Essays on Spatial Econometrics
Paulo Gustavo de Sampaio Grahl

Essays on Spatial Econometrics

Dissertação para obtenção do grau de Doutor apresentada à Escola de Pós-Graduação em Economia

Área de concentração: Econometria, Finanças e Economia Internacional

Orientador: Renato Galvão Flores Junior

Rio de Janeiro
2012
Grahl, Paulo Gustavo de Sampaio
356 f.

Tese (Doutorado) - Fundação Getulio Vargas, Escola de Pós-
Graduação em Economia.
Orientador: Renato Galvão Flóres Junior.
Inclui bibliografia.


CDD – 330.015195
PAULO GUSTAVO DE SAMPAIO GRAHL

ESSAYS ON SPATIAL ECONOMETRICS


Data da defesa: 22/12/2012
Aprovada em: 22/12/2012

ASSINATURA DOS MEMBROS DA BANCA EXAMINADORA

Renato Galvão Flôres Junior
Orientador (a)

André Luis Squarize Chagas

Alexandre Sartoris Neto

Alexandre Xavier de Carvalho Ywata

Filipe Lage de Sousa
Dedico esta tese à Tatiane, minha esposa e companheira, às minhas filhas Ana Luisa e Aline, pelo apoio e compreensão ao longo desta jornada. Aos meus pais, pela educação e valores que permeam minha vida.
Agradecimentos

Ao mestre e orientador professor Renato Flôres, pelas brilhantes aulas que despertaram meu interesse pela estatística e econometria, pelo apoio e orientação durante a elaboração da tese.

Aos professores Aloísio Araujo, Carlos Eugênio da Costa, Rubens Penha Cysne, cujas aulas, projetos e conversas foram importantes para complementar minha formação e enriqueceram a experiência na EPGE. Ao professor Luiz Renato Lima pelo apoio e estímulo para que eu viesse para o doutorado na EPGE, e que muito ajudou na minha mudança para o Rio de Janeiro e na minha adaptação à escola. Ao professor John Kolassa cujo livro e troca de emails foi importante para o desenvolvimento das séries de Edgeworth e saddlepoint.

Aos colegas e amigos, que muito contribuíram para enfrentar o ritmo intenso de aulas e provas, e o período mais calmo, mas não menos tenso, de elaboração da dissertação. Um centro de excelência é também formado pelos alunos, e isso ficou claro durante a convivência com os amigos na EPGE.

Ao pessoal do Núcleo de Computação, pela ajuda na utilização dos recursos computacionais. A Marcia, Mary e Tania, pelo apoio e auxílio, entre outros, em encontrar tempo disponível na agenda dos professores para as reuniões.
Resumo

Esta dissertação concentra-se nos processos estocásticos espaciais definidos em um reticulado, os chamados modelos do tipo Cliff & Ord. Minha contribuição nesta tese consiste em utilizar aproximações de Edgeworth e saddlepoint para investigar as propriedades em amostras finitas do teste para detectar a presença de dependência espacial em modelos SAR (autoregressivo espacial), e propor uma nova classe de modelos econômicos espaciais na qual os parâmetros que afetam a estrutura da média são distintos dos parâmetros presentes na estrutura da variância do processo. Isto permite uma interpretação mais clara dos parâmetros do modelo, além de generalizar uma proposta de taxonomia feita por Anselin (2003). Eu proponho um estimador para os parâmetros do modelo e derivo a distribuição assintótica do estimador. O modelo sugerido na dissertação fornece uma interpretação interessante ao modelo SARAR, bastante comum na literatura. A investigação das propriedades em amostras finitas dos testes expande com relação a literatura permitindo que a matriz de vizinhança do processo espacial seja uma função não-linear do parâmetro de dependência espacial. A utilização de aproximações ao invés de simulação (mais comum na literatura), permite uma maneira fácil de comparar as propriedades dos testes com diferentes matrizes de vizinhança e corrigir o tamanho ao comparar a potência dos testes. Eu obtenho teste invariante ótimo que é também localmente uniformemente mais potente (LUMPI). Construo o envelope de potência para o teste LUMPI e mostro que ele é *virtualmente* UMP, pois a potência do teste está muito próxima ao envelope (considerando as estruturas espaciais definidas na dissertação). Eu sugiro um procedimento prático para construir um teste que tem boa potência em uma gama de situações onde talvez o teste LUMPI não tenha boas propriedades. Eu concluo que a potência do teste aumenta com o tamanho da amostra e com o parâmetro de dependência espacial (o que está de acordo com a literatura). Entretanto, dispute a visão consensual que a potência do teste diminui à medida que a matriz de vizinhança fica mais densa. Isto reflete um erro de medida comum na literatura, pois a distância estatística entre a hipótese nula e a alternativa varia muito com a estrutura da matriz. Fazendo a correção, concluo que a potência do teste aumenta com a distância da alternativa à nula, como esperado.

Palavras-chave: Econometria espacial, Mínimos quadrados em dois estágios, Método generalizado de momentos, Amostras finitas, Heteroscedasticidade, Assintótica, Cliff & Ord, Edgeworth, Saddlepoint, Teste de hipóteses, Testes invariants, Testes UMP
Abstract

This dissertation focuses on spatial stochastic process on a lattice (Cliff & Ord-type of models). My contribution consists of using Edgeworth and saddlepoint series to investigate small sample size and power properties of tests for detecting spatial dependence in spatial autoregressive (SAR) stochastic processes, and proposing a new class of spatial econometric models where the spatial dependence parameters that enter the mean structure are different from the ones in the covariance structure. This allows a clearer interpretation of models’ parameters and generalizes the set of local and global models suggested by Anselin (2003) as an alternative to the traditional Cliff & Ord models. I propose an estimation procedure for the model’s parameters and derive the asymptotic distribution of the parameters’ estimators. The suggested model provides some insights on the structure of the commonly used mixed regressive, spatial autoregressive model with spatial autoregressive disturbances (SARAR). The study of the small sample properties of tests to detect spatial dependence expands on the existing literature by allowing the neighborhood structure to be a nonlinear function of the spatial dependence parameter. The use of series approximations instead of the often used Monte Carlo simulation allows a simple way to compare test properties across different neighborhood structures and to correct for size when comparing power. I obtain the power envelope for testing the presence of spatial dependence in the SAR process using the optimal invariant test statistic, which is also locally uniformly most powerful invariant (LUMPI). I have found that the LUMPI test is virtually UMP since its power is very close to the power envelope. I suggest a practical procedure to build a test that, while not UMP, retain good power properties in a wider range of the spatial parameter when compared to the LUMPI test. I find that power increases with sample size and with the spatial dependence parameter – which agrees with the literature. However, I call into question the consensus view that power decreases as the spatial weight matrix becomes more densely connected. This finding in the literature reflects an error of measure because the hypothesis being compared are at very different statistical distance from the null. After adjusting for this, the power is larger for alternative hypothesis further away from the null – as one would expect.

Keywords: Spatial econometrics, Spatial autoregressive and moving average models, Two-stage least squares, Generalized moments estimation, Small sample, Heteroskedasticity, Asymptotics, Cliff & Ord model, Edgeworth series, Saddlepoint series, Hypothesis testing, Invariant tests, Uniformly most powerful tests
# Contents

1 Spatial Econometrics
   1.1 Introduction and Motivation ........................................... 1
   1.2 Spatial Econometrics ................................................ 3
   1.3 Spatial Stochastic Processes ....................................... 4
      1.3.1 Definition ..................................................... 4
      1.3.2 Neighborhood Structure ..................................... 6
      1.3.3 Parallel Between Graph Theory and Specification of Lattice Models .......... 7
   1.4 Using Models with Spatial Dependence in Economics ............... 8
      1.4.1 Underlying Model Assumes Spatial Dependence ............... 8
      1.4.2 Spatial Dependence Results from Specification Errors, Omitted Variables 9
      1.5 Spatial Dependence in the First Two Moments .................... 9
         1.5.1 Cressie's Critique ........................................ 10
      1.6 Intent of this Dissertation ..................................... 10

2 Small Sample Inference in Spatial Econometric Models ................ 13
   2.1 Introduction ...................................................... 13
   2.2 Basic Model, Parameter Space ..................................... 14
      2.2.1 Uniformly Bounded Matrices ................................ 19
   2.3 Testing Problem .................................................. 20
   2.4 Asymptotic Expansions ............................................ 21
      2.4.1 Edgeworth Series .......................................... 21
      2.4.2 Saddlepoint Series ....................................... 25
   2.5 Testing for Spatial Dependence (Known Variance) ................. 29
      2.5.1 Edgeworth Series - Optimum Test ................................ 32
      2.5.2 Saddlepoint Series ....................................... 35
      2.5.3 Simulation-based Testing .................................. 37
   2.6 Testing for Spatial Dependence (Unknown Variance) .............. 38
      2.6.1 Testing Problem with Unknown Variance ...................... 38
      2.6.2 Edgeworth Approximation .................................. 40
      2.6.3 Saddlepoint Approximation ................................ 44
   2.7 Evaluating the Accuracy of the Proposed Approximations .......... 45
      2.7.1 Size Properties of the LUMP Test ................................ 50
      2.7.2 Size Properties of the LUMPI Test .......................... 51
      2.7.3 Power Envelope Using Edgeworth Series ...................... 67
      2.7.4 Power Envelope Using Saddlepoint .......................... 72
      2.7.5 Virtually Most Powerful Test ................................ 77
   2.8 Conclusion ......................................................... 80
3 Specification and Estimation of a New Class of Spatial ARMA Models 83
  3.1 Introduction .................................................. 83
  3.2 Current Spatial Models ....................................... 84
  3.3 Proposed Spatial Model ....................................... 86
  3.4 The General Spatial Model .................................... 87
    3.4.1 Assumptions .............................................. 89
    3.4.2 Proposed Estimation Procedure ......................... 90
  3.5 Instrumental Variables Estimator ............................ 93
  3.6 Method of Moments Estimator ................................ 97
    3.6.1 Identification .......................................... 100
    3.6.2 Uniform Convergence ................................... 106
    3.6.3 Consistency ............................................. 109
    3.6.4 Asymptotic Normality ................................... 109
  3.7 Conclusion ................................................... 117

4 Example: Spatial Econometrics Applied to International Trade 119
  4.1 Introduction .................................................. 119
  4.2 Traditional Gravity Equations ................................. 121
  4.3 Spatial Dispersion of the Residuals - Exploratory Analysis 123
  4.4 Moran I Test for Spatial Correlation ......................... 129
    4.4.1 Neighborhood Structure for Flows ....................... 129
    4.4.2 Neighborhood Structure for Countries .................. 130
    4.4.3 Moran Plot ............................................. 131
    4.4.4 Bootstrapping Moran I Test ............................. 131
  4.5 General Spatial Model ....................................... 133
    4.5.1 Estimation .............................................. 133
  4.6 Conclusion ................................................... 137

5 Conclusion: Final Remarks ..................................... 139

A Appendices ...................................................... 143
  A.1 Appendix to Chapter 2 ....................................... 143
    A.1.1 Circulant Spatial Weight Matrices ....................... 143
    A.1.2 Toeplitz Matrices ...................................... 145
  A.2 Appendix to Chapter 3 ....................................... 146
    A.2.1 Mathematical definitions and some useful lemmas .... 146
    A.2.2 Moments of quadratic forms in nonnormal variables ... 148
    A.2.3 Miscellaneous ........................................... 151
  A.3 Appendix to Chapter 4 ....................................... 152
    A.3.1 Neighborhood Structure Maps ............................ 152
    A.3.2 Moran Plot ............................................. 156
  A.4 R codes ..................................................... 178
    A.4.1 Codes used in Chapter 2 ................................ 178
    A.4.2 Codes used in Chapter 4 ................................ 329

Bibliography ..................................................... 351
List of Figures

2.1 Standardized critical values (design 1, \(m = 2, \rho_1 = 0.1\)) ........................................ 29
2.2 Simulated critical value ............................................................................................................ 38
2.3 Example of designs 1 and 2 (\(n = 20, \text{and } m = 2\)) ..................................................... 46
2.4 Example of designs 3 and 4 (\(n = 20, \text{and } m = 2\)) ..................................................... 47
2.5 Contiguity structure for Rio de Janeiro's municipalities (design 5a) ........................................ 48
2.6 Approximate critical value (using Edgeworth) for different sample sizes ................................ 52
2.7 Approximate critical value (using \(BNJA\), varying \(m\)) .................................................... 53
2.8 Approximate critical value (using \(BNJA\), varying \(J\)) .................................................... 53
2.9 Power envelope using Edgeworth (\(m = 2\)) ........................................................................ 67
2.10 Edgeworth approximation for \(F_{\rho_1, \nu_0(\rho_1)}\) (design 1, \(m = 2, n = 92\)) ......................... 68
2.11 Power envelope for design 1 (different values of \(m\)) ....................................................... 69
2.12 Power envelope against information distance ........................................................................ 71
2.13 Comparison of power envelope for different designs ......................................................... 72
2.14 Power envelope using Saddlepoint (\(m = 2\)) ..................................................................... 73
2.15 Power envelope: Edgeworth versus Saddlepoint approximations (\(n = 30\)) ....................... 74
2.16 Power envelope (saddlepoint) against information distance ............................................. 75
2.17 Comparison of power envelope (saddlepoint) for different designs ...................................... 76
2.18 Power loss of the LUMPI test .............................................................................................. 78
2.19 Power loss of LUMPI test vs. alternative test for design 3 (\(n = 92, J = 16\)) ...................... 79

3.1 Some examples of moment conditions for spatial AR(1) process ........................................ 103
3.2 Some examples of moment conditions for spatial MA(1) process ...................................... 105

4.1 Argentina - residuals from standard gravity equation ........................................................ 124
4.2 Australia - residuals from standard gravity equation ......................................................... 125
4.3 Brazil - residuals from standard gravity equation ............................................................. 126
4.4 Poland - residuals from standard gravity equation ........................................................... 127

A.1 Design 1 - Contiguity ........................................................................................................ 152
A.2 Design 2 - Triangulation ..................................................................................................... 152
A.3 Design 3 - Sphere of Influence (SOI) ............................................................................ 153
A.4 Design 4 - Gabriel ............................................................................................................ 153
A.5 Design 5 - Relative .......................................................................................................... 153
A.6 Design 6 - Nearest Neighbor 1 (NN1) ............................................................................ 154
A.7 Design 7 - Nearest Neighbor 3 (NN3) ............................................................................ 154
A.8 Design 8 - Nearest Neighbor 5 (NN5) ............................................................................ 154
A.9 Design 9 - Geodesic Distance (\(d_1 = 6.2\) thousand Km) ............................................... 155
A.10 Design 10 - Geodesic Distance (\(d_2 = 7.4\) thousand Km) .............................................. 155
A.11 Design 11 - Geodesic Distance (\(d_3 = 9.2\) thousand Km) ................................................ 155
A.12 Moran plot for the residuals of the exports gravity equation - Design 1 ......................... 156
A.13 Moran plot for the residuals of the exports gravity equation - Design 2 ......................... 157
A.14 Moran plot for the residuals of the exports gravity equation - Design 3 ......................... 158
A.15 Moran plot for the residuals of the exports gravity equation – Design 4 159
A.16 Moran plot for the residuals of the exports gravity equation – Design 5 160
A.17 Moran plot for the residuals of the exports gravity equation – Design 6 161
A.18 Moran plot for the residuals of the exports gravity equation – Design 7 162
A.19 Moran plot for the residuals of the exports gravity equation – Design 8 163
A.20 Moran plot for the residuals of the exports gravity equation – Design 9 164
A.21 Moran plot for the residuals of the exports gravity equation – Design 10 165
A.22 Moran plot for the residuals of the exports gravity equation – Design 11 166
A.23 Moran plot for the residuals of the imports gravity equation – Design 1 167
A.24 Moran plot for the residuals of the imports gravity equation – Design 2 168
A.25 Moran plot for the residuals of the imports gravity equation – Design 3 169
A.26 Moran plot for the residuals of the imports gravity equation – Design 4 170
A.27 Moran plot for the residuals of the imports gravity equation – Design 5 171
A.28 Moran plot for the residuals of the imports gravity equation – Design 6 172
A.29 Moran plot for the residuals of the imports gravity equation – Design 7 173
A.30 Moran plot for the residuals of the imports gravity equation – Design 8 174
A.31 Moran plot for the residuals of the imports gravity equation – Design 9 175
A.32 Moran plot for the residuals of the imports gravity equation – Design 10 176
A.33 Moran plot for the residuals of the imports gravity equation – Design 11 177
List of Tables

2.1 Coefficients of Gram-Charlier series ........................................... 23
2.2 Spatial weight matrices - proposed designs ..................................... 49
2.3 Skewness and kurtosis of the $T_n$ statistic ($\rho_1 = 0$) .......... 55
2.4 *Design 1* - critical levels and actual sizes ($n = 92, \alpha = 10\%$) .......................................................... 55
2.5 *Design 2* - critical levels and actual sizes ($n = 92, \alpha = 10\%$) .......................................................... 56
2.6 *Design 3* - critical levels and actual sizes ($n = 92, \alpha = 10\%$) .......................................................... 56
2.7 *Design 4* - critical levels and actual sizes ($n = 92, \alpha = 10\%$) .......................................................... 56
2.8 *Design 1* - critical levels and actual sizes ($n = 92, \alpha = 5\%$) .......................................................... 57
2.9 *Design 2* - critical levels and actual sizes ($n = 92, \alpha = 5\%$) .......................................................... 57
2.10 *Design 3* - critical levels and actual sizes ($n = 92, \alpha = 5\%$) .......................................................... 57
2.11 *Design 5* - critical levels and actual sizes ($n = 92, \alpha = 5\%$) .......................................................... 58
2.12 *Design 1* - critical levels and actual sizes ($n = 30, \alpha = 10\%$) .......................................................... 58
2.13 *Design 2* - critical levels and actual sizes ($n = 30, \alpha = 10\%$) .......................................................... 58
2.14 *Design 3* - critical levels and actual sizes ($n = 30, \alpha = 10\%$) .......................................................... 59
2.15 *Design 1* - critical levels and actual sizes ($n = 30, \alpha = 5\%$) .......................................................... 59
2.16 *Design 2* - critical levels and actual sizes ($n = 30, \alpha = 5\%$) .......................................................... 59
2.17 *Design 3* - critical levels and actual sizes ($n = 30, \alpha = 5\%$) .......................................................... 60
2.18 Empirical variance, skewness and kurtosis of the $V_n$ statistic ($\rho_1 = 0$) .......................................................... 61
2.19 *Design 1* - critical levels and actual sizes ($n = 92, \alpha = 10\%$) .......................................................... 61
2.20 *Design 2* - critical levels and actual sizes ($n = 92, \alpha = 10\%$) .......................................................... 62
2.21 *Design 3* - critical levels and actual sizes ($n = 92, \alpha = 10\%$) .......................................................... 62
2.22 *Design 4* - critical levels and actual sizes ($n = 92, \alpha = 10\%$) .......................................................... 62
2.23 *Design 1* - critical levels and actual sizes ($n = 92, \alpha = 5\%$) .......................................................... 63
2.24 *Design 2* - critical levels and actual sizes ($n = 92, \alpha = 5\%$) .......................................................... 63
2.25 *Design 3* - critical levels and actual sizes ($n = 92, \alpha = 5\%$) .......................................................... 63
2.26 *Design 4* - critical levels and actual sizes ($n = 92, \alpha = 5\%$) .......................................................... 64
2.27 *Design 1* - critical levels and actual sizes ($n = 30, \alpha = 10\%$) .......................................................... 64
2.28 *Design 2* - critical levels and actual sizes ($n = 30, \alpha = 10\%$) .......................................................... 64
2.29 *Design 3* - critical levels and actual sizes ($n = 30, \alpha = 10\%$) .......................................................... 65
2.30 *Design 1* - critical levels and actual sizes ($n = 30, \alpha = 5\%$) .......................................................... 65
2.31 *Design 2* - critical levels and actual sizes ($n = 30, \alpha = 5\%$) .......................................................... 65
2.32 *Design 3* - critical levels and actual sizes ($n = 30, \alpha = 5\%$) .......................................................... 66
2.33 Statistical curvature (square of) .................................................. 79

3.1 Anselin's taxonomy of spatial models .............................................. 86
3.2 General model $M$ ........................................................................ 87
3.3 Roots of moment equations ......................................................... 102

4.1 Traditional gravity equation – export flows .................................... 122
4.2 Traditional gravity equation – import flows .................................... 123
4.3 Total actual trade minus total calculated trade (2003) .................. 128
4.4 Neighborhood structures .............................................................. 135
4.5 General Spatial Model – Export Flows – Design 3 ........................................ 136
Listings

1 chart01.R .................................................. 178
2 chart04modified.R ........................................... 181
3 chart05.R .................................................. 185
4 chart06.R .................................................. 186
5 chart07modified.R .......................................... 189
6 chart08.R .................................................. 191
7 chart10.R .................................................. 192
8 chart11.R .................................................. 192
9 chart12modified.R ......................................... 194
10 comparesaddlemethodsforCV.R ............................. 197
11 correlacao.stat.suf.R ....................................... 198
12 edgeworth.functions.R ..................................... 198
13 edgeworth.R ............................................... 200
14 empiricalmatrices.R ....................................... 206
15 invarianttest.R ............................................ 208
16 Matriz.k.ahead.R .......................................... 211
17 mle.rho.R ................................................ 212
18 penz.change.m.R ........................................... 214
19 power.envelope.null.zero.R ............................... 218
20 saddlepoint.functions.R ................................... 219
21 simulajedgeworthalternativa.R ......................... 220
22 simulajedgeworth.R ....................................... 222
23 art1.chart03.R ............................................ 224
24 art1.compare.Curvature.V.R ............................... 224
25 art1.compare.optimal.vs.LUMPI.test.Edgeworth.R .... 226
26 art1.compare.optimal.vs.LUMPI.test.R .................. 227
27 art1.compare.powerevelope.saddle.vs.empirical.design5.R 232
28 art1.create.size.tables.R ................................ 233
29 art1.CValue.as.n.increases.R ........................... 233
30 art1.CValue.change.avg.degree.R ....................... 235
31 art1.Edgeworth.vs.true.R ................................ 236
32 art1.MonteCarlo.curvature.V.R .......................... 237
33 art1.MonteCarlo.CValue.T.R ............................. 240
34 art1.MonteCarlo.CValue.V.R .............................. 242
35 art1.power.at.some.infodistance.points.R ............ 245
36 art1.powerevelope.design1.changing.m.R ............... 245
37 art1.powerevelope.Edgeworth.R .......................... 246
38 art1.powerevelope.Edgeworth.vs.Saddlepoint.R ....... 250
40 art1.powerevelope.Saddlepoint.R ....................... 251
41 art1.powerevelope.vs.info.distance.R .................. 254
42 art1.powerevelope.vs.info.distance.Saddlepoint.R ... 258
art1 size tables 10percent.R .................................................. 261
art1 size tables 10percent V.R ............................................. 265
art1 size tables 5percent.R .................................................. 269
art1 size tables 5percent V.R ................................................. 272
art1 skew kurtosis T.R ....................................................... 277
art1 skew kurtosis V.R ....................................................... 280
art compare CV vs skew kurt T.R ...................................... 284
art compare CV vs skew kurt V.R ...................................... 285
chart designs.R ................................................................. 292
matrix rj.R ..................................................................... 293
Alinha.R ....................................................................... 295
ar.R ............................................................................. 295
BNJACV T.R ........................................................................... 295
bnjacv v.r .............................................................................. 296
bnjapt.r .................................................................................. 296
bnjapv.r ............................................................................... 296
BootstrapOLS.R ...................................................................... 296
BootstrapSpatial.R ............................................................ 297
b.r .................................................................................. 297
c0.r .................................................................................. 297
Ca.R .................................................................................. 298
circularM.R ........................................................................... 298
ConfidenceInterval.R ........................................................ 298
CornFiCV T.R .......................................................................... 298
createHermite.R ............................................................... 298
Cr.R .................................................................................. 298
cumulantes T.R ..................................................................... 299
cumulantes V.R ..................................................................... 299
Dv.R .................................................................................. 299
E2SLS.R ................................................................................ 299
e4.r .................................................................................. 299
e5.r .................................................................................. 300
EGMM2.R ................................................................................ 300
EGMM.R ................................................................................ 301
eigenRange.R ................................................................. 302
Ew4CV T.R ................................................................................ 302
Ew4CV V.R ................................................................................ 303
Ew4pv.r .................................................................................. 303
Ew5CV T.R ................................................................................ 303
Ew5CV V.R ................................................................................ 304
FisherInfoRho.R .................................................................... 304
gencirculantM.R ............................................................ 304
genDesign1.R ......................................................................... 304
genDesign2.R ......................................................................... 304
genDesign3.R ......................................................................... 304
genDesign4 a.R ......................................................................... 304
genDesign4 b.R ......................................................................... 305
genToeplitzM.R ................................................................. 305
GMMIV.R ................................................................................ 305
GS2SLS.R ................................................................................ 308
<table>
<thead>
<tr>
<th></th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>94</td>
<td>IGMM.R</td>
<td>308</td>
</tr>
<tr>
<td>95</td>
<td>invariants.R</td>
<td>309</td>
</tr>
<tr>
<td>96</td>
<td>kt.r</td>
<td>309</td>
</tr>
<tr>
<td>97</td>
<td>KV.r</td>
<td>309</td>
</tr>
<tr>
<td>98</td>
<td>Linv.R</td>
<td>310</td>
</tr>
<tr>
<td>99</td>
<td>l.r</td>
<td>310</td>
</tr>
<tr>
<td>100</td>
<td>LRACV T.R</td>
<td>310</td>
</tr>
<tr>
<td>101</td>
<td>lrcv.v.r</td>
<td>310</td>
</tr>
<tr>
<td>102</td>
<td>LRAPT.R</td>
<td>310</td>
</tr>
<tr>
<td>103</td>
<td>lrap.v.r</td>
<td>311</td>
</tr>
<tr>
<td>104</td>
<td>matrixpower.R</td>
<td>311</td>
</tr>
<tr>
<td>105</td>
<td>MLEgeneralSAR.R</td>
<td>311</td>
</tr>
<tr>
<td>106</td>
<td>MLESAR.R</td>
<td>313</td>
</tr>
<tr>
<td>107</td>
<td>MLESE.R</td>
<td>314</td>
</tr>
<tr>
<td>108</td>
<td>MonteCarloCurvature V.R</td>
<td>315</td>
</tr>
<tr>
<td>109</td>
<td>MonteCarloCV H1 V.R</td>
<td>315</td>
</tr>
<tr>
<td>110</td>
<td>MonteCarloCV T.R</td>
<td>316</td>
</tr>
<tr>
<td>111</td>
<td>MonteCarloCV V.R</td>
<td>316</td>
</tr>
<tr>
<td>112</td>
<td>MonteCarloPower V.R</td>
<td>316</td>
</tr>
<tr>
<td>113</td>
<td>neggradLLc.R</td>
<td>316</td>
</tr>
<tr>
<td>114</td>
<td>neggradLLc sar.R</td>
<td>317</td>
</tr>
<tr>
<td>115</td>
<td>neggradLLc se.R</td>
<td>317</td>
</tr>
<tr>
<td>116</td>
<td>negLLc.R</td>
<td>317</td>
</tr>
<tr>
<td>117</td>
<td>negLLc sar.R</td>
<td>318</td>
</tr>
<tr>
<td>118</td>
<td>negLLc se.R</td>
<td>318</td>
</tr>
<tr>
<td>119</td>
<td>negLLfull.R</td>
<td>318</td>
</tr>
<tr>
<td>120</td>
<td>negLLfull sar.R</td>
<td>318</td>
</tr>
<tr>
<td>121</td>
<td>NLSSAR.R</td>
<td>319</td>
</tr>
<tr>
<td>122</td>
<td>plotMatrix.R</td>
<td>319</td>
</tr>
<tr>
<td>123</td>
<td>powerBNJ V.R</td>
<td>320</td>
</tr>
<tr>
<td>124</td>
<td>powerE4.R</td>
<td>320</td>
</tr>
<tr>
<td>125</td>
<td>powerE4 V.R</td>
<td>321</td>
</tr>
<tr>
<td>126</td>
<td>powerLR V.R</td>
<td>321</td>
</tr>
<tr>
<td>127</td>
<td>powerM.R</td>
<td>321</td>
</tr>
<tr>
<td>128</td>
<td>pt.r</td>
<td>321</td>
</tr>
<tr>
<td>129</td>
<td>PV.r</td>
<td>322</td>
</tr>
<tr>
<td>130</td>
<td>RACV T.R</td>
<td>322</td>
</tr>
<tr>
<td>131</td>
<td>racv.v.r</td>
<td>322</td>
</tr>
<tr>
<td>132</td>
<td>RAPTR.R</td>
<td>323</td>
</tr>
<tr>
<td>133</td>
<td>rapv.r</td>
<td>323</td>
</tr>
<tr>
<td>134</td>
<td>rholimiteneg.R</td>
<td>323</td>
</tr>
<tr>
<td>135</td>
<td>rholimite.R</td>
<td>323</td>
</tr>
<tr>
<td>136</td>
<td>saddlepinvCV T.R</td>
<td>323</td>
</tr>
<tr>
<td>137</td>
<td>saddlepointCV T.R</td>
<td>324</td>
</tr>
<tr>
<td>138</td>
<td>saddlepointCV V.R</td>
<td>324</td>
</tr>
<tr>
<td>139</td>
<td>saddlepoint T.R</td>
<td>324</td>
</tr>
<tr>
<td>140</td>
<td>saddlepoint V.R</td>
<td>325</td>
</tr>
<tr>
<td>141</td>
<td>SignedFisherInfoRho.R</td>
<td>325</td>
</tr>
<tr>
<td>142</td>
<td>SymmetricCV V.R</td>
<td>325</td>
</tr>
<tr>
<td>143</td>
<td>Tn.R</td>
<td>325</td>
</tr>
<tr>
<td>144</td>
<td>trpowermatrix.R</td>
<td>325</td>
</tr>
<tr>
<td>File Name</td>
<td>Page</td>
<td></td>
</tr>
<tr>
<td>------------------------------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>tr.R</td>
<td>325</td>
<td></td>
</tr>
<tr>
<td>VarHgeneralSAR.R</td>
<td>325</td>
<td></td>
</tr>
<tr>
<td>VarIISAR.R</td>
<td>326</td>
<td></td>
</tr>
<tr>
<td>Var1MgeneralSAR.R</td>
<td>326</td>
<td></td>
</tr>
<tr>
<td>Var1MSAR.R</td>
<td>327</td>
<td></td>
</tr>
<tr>
<td>vc.v.R</td>
<td>328</td>
<td></td>
</tr>
<tr>
<td>Vn.R</td>
<td>328</td>
<td></td>
</tr>
<tr>
<td>w.r</td>
<td>328</td>
<td></td>
</tr>
<tr>
<td>gamma tables.R</td>
<td>329</td>
<td></td>
</tr>
<tr>
<td>gamma tables RN port.R</td>
<td>331</td>
<td></td>
</tr>
<tr>
<td>moran test.R</td>
<td>332</td>
<td></td>
</tr>
<tr>
<td>prepare data gamma.R</td>
<td>335</td>
<td></td>
</tr>
<tr>
<td>worldtrade gamma.R</td>
<td>336</td>
<td></td>
</tr>
<tr>
<td>generate world neighbor matrix.R</td>
<td>337</td>
<td></td>
</tr>
<tr>
<td>bootstrap moran.R</td>
<td>339</td>
<td></td>
</tr>
<tr>
<td>gamma functions.R</td>
<td>341</td>
<td></td>
</tr>
<tr>
<td>generate gamma estimates.R</td>
<td>345</td>
<td></td>
</tr>
<tr>
<td>generate maps.R</td>
<td>347</td>
<td></td>
</tr>
<tr>
<td>genGarmaestimate.R</td>
<td>348</td>
<td></td>
</tr>
<tr>
<td>traditional gravity.R</td>
<td>348</td>
<td></td>
</tr>
</tbody>
</table>
Chapter 1

Spatial Econometrics

1.1 Introduction and Motivation

Models that consider the spatial dependence among sample points have only recently become present in mainstream economics and econometric models. A spatial stochastic process is nothing more than a standard stochastic process where (i) a particular interpretation is given to the index set (set of spatial units), and (ii) the departure from the classical paradigm of iid is based on the dependence structure among the spatial units, with the degree of dependence proportional to the distance among observations. A broad definition of the terms space and distance allows spatial models to be used in several areas: economics (new economic geography, peer influence, neighbor effects, spillover effects, housing decisions, unemployment), sociology (externalities, diffusion and contagion in crime analysis), political science (individual voting behavior, international relations), among others.

My dissertation focuses on spatial stochastic process on a lattice, where the index set is a countable set of regularly or irregularly spaced spatial units. Models or data generating processes in this class are often called Cliff & Ord-type of models. This class of models has a parallel with time series (which can be seen as a particular case of a spatial stochastic process on a lattice) and thus borrows from the time series terminology of autoregressive and moving average processes.

In this chapter I define the spatial processes, highlight the parallel with graph theory and call attention to the fact that models with spatial dependence are suitable both in cases where there is a theory suggesting spatial interaction among observations, or in cases where there is no such theoretical foundation but nevertheless the observations can be regarded as having spatial dependence. This could arise from omitted variables that depend on the location of the sample point in space, from misspecification of the true underlying data generating process, or from measurement problems. Ignoring spatial effects in these cases could lead, for instance, to biased variance estimators, inefficient estimators, bias and inconsistency in OLS estimators. Some authors suggest as good practice to consider spatial dependence explicitly in any model to start with, and then to test the hypothesis of spatial dependence. This would likely make the proposed models robust to specification errors, omitted variables, and measurement errors.

The interpretation of the spatial dependence parameters in the traditional Cliff & Ord models varies among authors. When it is present in the covariance structure it is often called small scale variation, or spatial heterogeneity, or interaction among units. When it is present in the mean structure it is often referred to as large scale variation, spatial dependence, or reaction to covariates. But a key drawback in the Cliff & Ord-type of models is that both spatial effects (mean and covariance) are mixed and this makes more difficult to interpret the estimated parameters. Some authors (Cressie, 1993) suggest it might be good practice to disentangle the spatial dependence parameters from the mean and the covariance structure (Cressie's critique).

In Small Sample Inference in Spatial Econometric Models I investigate small sample size and power properties of tests for detecting spatial dependence in spatial autoregressive (SAR) stochas-
tic processes. I expand on the existing literature by allowing the spatial weight matrix (or neighborhood structure) to be a nonlinear function of the spatial dependence parameter. The choice of the spatial weight matrix is often ad hoc and thus it is important to be able to compare test properties among different specifications. Nearly all the research in this area relies on Monte Carlo simulation and the experiments usually do not correct for size when comparing power of the tests. Instead of simulation, I use Edgeworth and saddlepoint series to approximate the small sample distribution of the test statistics. Simulation shows that these approximations are accurate in small samples for a variety of designs for the spatial weight matrix. Moreover, under this framework it is straightforward to correct for size when comparing power of the tests.

In order to avoid the usual procedure of comparing several test statistics, I obtain the power envelope for testing the presence of spatial dependence in the SAR process using (i) the optimal test statistic (in the Neyman-Pearson sense) when the variance of the error term is known, and (ii) the optimal invariant test statistic when the variance is unknown. The optimal invariant test converges to the score test (Lagrange multiplier) when the alternative hypothesis converges to the null of no spatial dependence and is, therefore, locally uniformly most powerful invariant (LUMP). I have found that the LUMP test is virtually uniformly most powerful in the sense that, for practical purposes, its power is very close to the power envelope. But this near optimality might not hold for spatial weight matrices with designs different from the ones I have used. Therefore, I suggest a practical procedure to build a test that, while not UMP, retain good power properties in a wider range for the spatial parameter when compared to the LUMP test. The proposed test statistics allow me to analyze how the power envelope behaves for different sample sizes and different designs for the weight matrix. I find that power increases with sample size and with the spatial dependence parameter — which agrees with the literature. However, I call into question the consensus view that power decreases as the spatial weight matrix becomes more densely connected. This finding in the literature reflects an error of measure because changing the weight matrix changes the structure of the process. Therefore when looking at the same spatial dependence parameter the hypothesis being compared are at very different statistical distance from the null. After adjusting for this, the power is larger for alternative hypothesis further away from the null — as one would expect.

In Specification and Estimation of a New Class of Spatial ARMA Models I build on the second suggested good practice: to use models where the spatial dependence parameters that enter the mean structure are different from the ones in the covariance structure. Anselin (2003) suggests a set of local and global models as an alternative to the traditional Cliff & Ord–type of spatial models to address this problem. However, global and local specifications are non-nested and Anselin himself noted this drawback. The author speculated about the possibility of considering models where, for instance, local externalities would be present in the error term and global externalities in the covariates. He also highlighted that none of these possibilities had received attention in the literature and, to the best of my knowledge, this remains an unexplored area both in theoretical as well as in applied econometrics. In order to overcome this problem, I propose a more general parametric spatial stochastic process that nests Anselin's suggested local and global models into an encompassing structure and allows for a clearer interpretation of the model's parameters vis-a-vis the existing models. This model provides some insights on the structure of the commonly used mixed regressive, spatial autoregressive model with spatial autoregressive disturbances (SARAR) but they are non-nested. Therefore they become competing models whose validity needs to be tested empirically. I also propose an estimation procedure for the model's parameters that has the benefit of being easily implementable and computationally simple even when the dimension of the parameter space is not small. The asymptotic distribution of the parameter's estimators is also derived in the chapter.

In Example: Spatial Econometrics Applied to International Trade I apply the model developed in Chapter 3 to estimate spatial dependence parameters for a modified international trade gravity equation. The underlying idea of gravity models is to express a flow (exports, imports,
or total trade) from country $i$ to country $j$ as being directly proportional to their economic mass (e.g., income) and inversely proportional to the distance between them. The economic size of a country would reflect demand and supply factors while the distance between them would be a proxy for transportation costs and other frictions. Additional dummy variables to control for common trade agreements, common language, etc., are also included. Gravity equations started as an empirical model of trade but it has been shown that it can be generated from a set of primitives in the economy. In this chapter I argue that it is reasonable to consider that gravity equations in international trade could still present spatial dependence even accounting for variables such as distance and common border.

I estimate a traditional gravity equation and then use Moran I test to check the residuals for the presence of spatial dependence. The limiting distribution of Moran I test statistic is known to be Gaussian even in the presence of heteroskedasticity (which is likely to be the case) but its dispersion parameter depends on the specific structure of the model. Therefore I perform a residual wild bootstrap of Moran's I test, which confirms that the residuals of the gravity equation are spatially dependent. I use several neighborhood structures (or spatial weight matrices) to compute Moran I test, all elaborated using different graph structures based on the concepts mentioned in the Spatial Econometrics chapter. It is interesting to highlight that the observations in the gravity model of trade are the flows between a pair of countries rather than the countries themselves. Therefore the neighborhood structure needs to be adapted to reflect this fact. I apply the procedure developed in chapter 3 to estimate the new class of spatial autoregressive moving average processes that is suggested in my dissertation. I do not attempt to qualify the estimated parameters nor to test trade models, but only to check, with a simple example, whether the spatial econometrics toolbox would be appropriate to deal with international trade gravity equations.

1.2 Spatial Econometrics

Models that consider the spatial structure of the data under investigation have a long tradition in the regional science and economic geography literature but its use in economics is relatively new. Baltagi et al. (2007) and Anselin et al., eds (2004) highlighted that spatial econometric models have only recently become present in mainstream economics and econometric methods. The literature on spatially dependent data usually refers to spatial econometrics or spatial statistics as the set of theory and methods to deal with models that encompass some type of spatial structure, but the distinction between these terms is not at all clear. The data structure and models used in spatial statistics, spatial econometrics, and econometrics are very similar. Anselin (1988) argues that the main difference between econometrics and spatial econometrics lies in the fact that spatial econometrics deals with specific aspects of models and data – spatial dependence and heterogeneity – but the author acknowledges this is a narrow view and that several methods used in econometrics can also be used to deal with specific aspects of spatial data.

The distinction between the terms spatial statistics and spatial econometrics is also debated in the literature. The differentiation proposed by Anselin (1988) between spatial econometrics (guided by models) and spatial statistics (guided by data) is strongly refuted by Cressie (1993), where the author mentions that “this distinction is not particularly helpful from a statistician’s perspective... mostly because both deal with similar methodological issues”. One important difference (Anselin, 2003) is that conditional models are common in spatial statistics, while simultaneous models are more used in spatial econometrics. Even though some references (e.g., Arbía, 2006) elaborate on both conditional and simultaneous models, most of the recent spatial econometrics literature focus almost exclusively on simultaneous models, despite the fact that conditional models are widely used in standard econometrics.

The use of the term spatial in a broader context (not limited to its geographical meaning) allows spatial models to be considered more generally as models for cross section dependence.
The idea of broadening the scope of space and distance is not new. Isard and Smith (1969), for instance, expanded the notion of physical distance to include policy space, inter-personal distance, social networks, among others, allowing for the idea of spatial dependence to be applied to a wide range of situations in the social sciences.

Indeed, spatial models have been used in various fields: economics (new economic geography, peer influence, neighbor effects, spillover effects, housing decisions, unemployment), sociology (externalities, diffusion and contagion in crime analysis), political science (individual voting behavior, international relations), among others. The rapid spread of social media and the use of the internet allowing to track peoples’ behaviors and relationships will likely provide a rich dataset to be used under the realm of spatial econometrics.

1.3 Spatial Stochastic Processes

1.3.1 Definition

Before going into the motivation for using spatial econometric models, I will formally define a stochastic process and highlight that a spatial stochastic process is nothing but a stochastic process where a particular interpretation to the index set is given.

1.3.1 Definition (Stochastic process). Let \((\Omega, \mathcal{F}, P)\) be a probability space, \(\Gamma\) any set, and \(\mathbb{R}^\Gamma\) the product space. A stochastic process is a measurable function \(x: \Omega \to \mathbb{R}^\Gamma\) where

\[x(\omega) = \{X_\gamma(\omega), \gamma \in \Gamma\}\]

\(\Gamma\) is the index set, and the random variable \(X_\gamma(\omega)\) is the process coordinate. If \(\Gamma\) is a finite set, then \(x\) is just a random vector. If \(\Gamma\) is a countable, totally ordered set, then \(x\) is a stochastic sequence. For any \(\gamma \in \Gamma\), \(X_\gamma(\omega)\) is a random variable, i.e., a \(\mathcal{F}\)-measurable function \(X_\gamma: \Omega \to \mathbb{R}\). For any \(\omega \in \Omega\), the function \(X(\omega): \Gamma \to \mathbb{R}\) is called sample path.

1.3.2 Definition (Spatial stochastic process). Is nothing more than an stochastic process where we can give a particular interpretation to the index set \(\Gamma \in \mathbb{R}^p\) (also called a \(p\)-dimensional random field). In this context, we will refer to the index \(\gamma \in \Gamma\) as a spatial unit.

Remark (Space-time process). A spatial stochastic process that includes also the time dimension can be included in definition 1.3.2. However, due to the unidirectional nature of the time dimension, it is usual to write the index set explicitly as \(\Gamma \times T\), where \(\Gamma \subset \mathbb{R}^p\), \(T \subset \mathbb{R}\), and the space-time stochastic process is a measurable function \(x: \Omega \to \mathbb{R}^{\Gamma \times T}\) where \(x(\omega) = \{X_{\gamma t}(\omega), (\gamma, t) \in \Gamma \times T\}\) which is a \(p + 1\)-dimensional random field.

Under this approach, time series models can be considered as a sub-set of spatial models, where the space is the time dimension and location and distance are well-defined concepts. An interesting example of this link between time series and spatial econometric models is the test for autocorrelation. The Durbin-Watson test for autocorrelation in a time-series context, can be written as a particular case of Moran I spatial autocorrelation test (see section 4.4). Indeed, Durbin and Watson’s first article on the subject in Biometrika, 1951, (Testing for Serial Correlation in Least Squares Regression: I) cites Moran’s 1950 article (Notes on Continuous Stochastic Phenomena) in the same journal. In the time series context, however, the space is a totally ordered set, which allows the use of more specific methods when compared to the general case where the space set cannot be ordered.

I will need a further definition in order to specify the neighborhood structure (in section 1.3.2).

1.3.3 Definition (Derived Measure). The random variable \(X_\gamma(\omega)\) lives in the space \((\mathbb{R}, \mathcal{B}, \mu)\) where \(\mathcal{B}\) is a Borel \(\sigma\)-algebra of \(\mathbb{R}\), and \(\mu_\gamma\) is the derived measure given by:

\[\mu_\gamma(B) = P(X_\gamma^{-1}(B)) = P(\{\omega \mid X_\gamma(\omega) \in B\}), \forall B \in \mathcal{B}\]
For each stochastic process we can have several associated finite-dimensional distributions (measures). For \( \gamma, \gamma' \in \Gamma \), \( \mu_{\gamma,\gamma'}(B_1 \times B_2) = P(X_{\gamma} \in B_1, X_{\gamma'} \in B_2), \forall B_1, B_2 \in B \). For \( \gamma_1, \gamma_2, \ldots, \gamma_r \in \Gamma \), \( \mu_{\gamma_1,\gamma_2,\ldots,\gamma_r}(B_1 \times B_2 \times \cdots \times B_r) = P(X_{\gamma_1} \in B_1, \ldots, X_{\gamma_r} \in B_r) \). Some restrictions need to be imposed for all \( \gamma, \gamma' \in \Gamma \) and all \( B_1, B_2 \in B^1 \):

\begin{itemize}
  \item \( \mu_{\gamma,\gamma'}(B_1 \times B_2) = \mu_{\gamma',\gamma}(B_2 \times B_1) = P(X_{\gamma} \in B_1, X_{\gamma'} \in B_2) \)
  \item \( \mu_{\gamma,\gamma}(B_1 \times \mathbb{R}) = \mu_{\gamma}(B_1) \)
\end{itemize}

This definition of spatial stochastic process is broad enough to encompass most of the models using spatial data found in the literature. Cressie (1993), for example, classifies the typical spatial statistics problems into:

**Geo-statistical data** it is the study of a random variable in a continuous region; in this case, the index set \( \Gamma \subset \mathbb{R}^p \) is fixed and the spatial units \( \gamma \) change continuously in its domain.

**Lattice data** in this case, the index set \( \Gamma \) is a countable set (or a finite set) of points; if the spatial units (points) are regularly spaced in \( \mathbb{R}^p \) we say it is a regular lattice, otherwise an irregular lattice; these are the models most used in the spatial econometrics literature.

**Point patterns** concerns the occurrence of events randomly disposed in space, for example, the location of new firms in a region; in this model the index set \( \Gamma \) is a point process; this model can be used, for instance, to investigate whether the location of spatial units is random, if it exhibits some regularities, or if an agglomeration pattern is present.

**Objects**\(^2\) it is an extension of the point pattern model, where \( X_\gamma \) is a random set\(^3\) instead of a random variable.

The spatial econometrics literature focus mostly in the lattice model's family, and this will be the focus of this dissertation.

The analysis of spatial stochastic process defined in a lattice is similar to stochastic processes where the index set is \( T \) (time): we observe only one path of the stochastic process (among many possible), with finite dimension, and try to make inference on the underlying process. One key difference, however, is that for time series usually two simplifying hypothesis are used: (i) the stochastic process started at distant past \( (t = -\infty) \) and proceeds indefinitely into the future \( (t = +\infty) \), and (ii) a transformation of the process is stationary and ergodic. For spatial stochastic processes, generally the stationary hypothesis is not valid (not even weak-stationarity), therefore some additional structure needs to be imposed on the spatial stochastic processes.

In general, for any stochastic process, the objective of the analysis is to characterize the distribution of the random variables observed in the process' path. Therefore, we need to make hypothesis on the distribution of each \( X_\gamma \) and on the dependence structure among these random variables. The underlying idea of the spatial stochastic processes is that the index set \( \Gamma \), even though not totally ordered, can be used to define a dependence structure among the random variables \( X_\gamma \). When the time dimension is also being considered (space-time stochastic process) and we assume that the dependence structure of the stochastic process remains unchanged in one of the dimensions (time or cross-section) then panel data tools could be used. However, when the time dimension is not available, or one is not willing to assume constancy of the stochastic process in this dimension, the econometrician needs to impose some structure to the data in order to estimate model's parameters or to make inference.

---

1 Called natural consistency conditions.
2 Mentioned here for completeness, as I could not find applications of this model in economics.
3 A random set is a measurable function from a probability space into a measure space of subsets of \( \mathbb{R}^p \); for details see Cressie (1993); a random set can also be viewed as a special case of random functions, called excursion sets (Adler, 1981).
Methods developed for the analysis of spatially dependent data aim at departing, in a parsimonious way, from the classical paradigm of standard statistical models, where a sample from an stochastic process is independent and identically distributed (iid). For spatial data, the dependence structure is related to the location of spatial units $\gamma$ in the space (or location of sample points in space), where space and location are concepts that can be broadly defined, not restricted to its geographical meaning. For example, space can be defined as a set of characteristics (age, education, sex, etc.) and an individual would be a point in this multi-dimensional space of individual characteristics, where some notion of distance or proximity can be established\(^4\).

### 1.3.2 Neighborhood Structure

The classical way to look at the dependence structure in a lattice spatial stochastic process is to impose, a priori, based on specific knowledge of the problem being investigated, a structure to the index set $\Gamma$ based on the location of the spatial units in the space. In analogy to the geographical meaning of location and space, the structure imposed to the index set is usually called neighborhood structure, but neighborhood matrix, spatial weight matrix or just weight matrix are also commonly used.

The neighborhood structure can be defined in a myriad of ways, but often it is constructed in terms (i) of a neighborhood relation, (ii) of the distance among spatial units, or (iii) both.

For a spatial stochastic process, one possibility to form the set of neighbors of a spatial unit $\gamma$ is to consider all $\gamma'$ units that are not pairwise independent of unit $\gamma$. In this case, the set of neighbors of unit $\gamma$ could be defined by:

$$\mathcal{N}_\gamma = \{ \gamma' \in \Gamma \mid \mu_{\gamma'\gamma}(B_1 \times B_2) \neq \mu_{\gamma\gamma}(B_1)\mu_{\gamma'\gamma}(B_2) \}$$

For all $B_1, B_2 \in \mathcal{B}$ such that $\mu_{\gamma\gamma}(B_1) \neq 1$ and $\mu_{\gamma'\gamma}(B_2) \neq 1$.

However, to introduce more structure in the spatial data, often a metric is also used to define the neighbor set. Assume $(\Gamma, \rho)$ is a metric space, where the metric $\rho : \Gamma \times \Gamma \to \mathbb{R}_+$ satisfy the usual conditions\(^5\). In Anselin (1988), for instance, the author uses a cutoff distance to define the set of neighbors. In this case, the set of neighbors of unit $\gamma \in \Gamma$ would be $\mathcal{N}_\gamma \cap \{ \gamma' \in \Gamma \mid \rho(\gamma, \gamma') \leq \delta_\gamma \}$ where $\delta_\gamma$ is an arbitrary cut-off\(^6\). With this definition, all spatial units that are not pairwise independent of $\gamma$ but are further away from it ($\rho(\gamma, \gamma') > \delta_\gamma$) are still connected to spatial unit $\gamma$ via higher order lags (higher order neighbors).

#### 1.3.4 Definition (Spatial Neighbors)

The set of neighbors of a spatial unit $\gamma \in \Gamma$ is usually defined based on the underlying economic model. It will be denoted by $\mathcal{N}_\gamma$. Note that the concept of neighborhood can also be described by a binary relation in the set $\Gamma \times \Gamma$, or $\mathcal{N} \subset \Gamma \times \Gamma$, where $aN b$ means $(a, b) \in \mathcal{N}$. This binary relation is:

(a) Irreflexive: $\nexists a \in \Gamma \mid aNa$

(b) Symmetric: $aN b \Rightarrow bNa$ for all $a, b \in \Gamma$

**Remark.** The fact that the neighbor sets can be described by a binary relation that is irreflexive and symmetric provides a natural representation of the neighboring structure as a graph, where the spatial units represent the vertices's (or nodes) of the graph and the binary neighboring relation is represented by the edges (or lines) of the graph (see Definition 1.3.5).

---

\(^4\)Isard and Smith (1969) expand the notion of physical distance to include policy space, inter-personal distance, social networks, among others, allowing the idea of spatial dependence to be applied to a wide range of situations in the social sciences.

\(^5\)In some circumstances we may need to add more structure to the process by considering not only the distance but also the direction between two spatial units, or the number of neighbors of each spatial unit.

\(^6\)In most of the practical applications, the cut-off is arbitrarily chosen and some sensitivity analysis is made in order to check how robust the parameters estimates are to changes in the cut-off level. See Carvalho et al. (2006) for an example of an study of growth in Brazilian municipalities.
The concept of distance among spatial units can be defined in a broad context. Conley (1999), for instance, argues that the distance among spatial units (agents) reflect the proximity of these units regarding some non-observable variables, a notion of economic distance. This structure could also be represented by a weighted graph, where the edges weights' are (inversely) proportional to the distance of its vertices. Anselin (1988) reminds the importance that the neighborhood structure be chosen according to solid concepts:

"...the structure of spatial dependence incorporated in the spatial weight matrix should be chosen judiciously, and related to general concepts from spatial interaction theory, such as the notions of accessibility and potential. In line with a model driven approach to spatial econometrics, the weight matrix should bear a direct relation to a theoretical conceptualization of the structure of dependence rather than reflecting an ad hoc description of spatial pattern".

To represent a neighborhood structure with a graph is interesting as it can sometimes simplify the understanding of some properties of the structure. For instance, in the case of a neighborhood relation, the adjacency matrix of the graph (spatial weight matrix, or just weight matrix) would be a binary matrix $W$, where the elements $w_{ij}$ would be unity when $i, j$ are neighbors and zero otherwise. In this case, the element $i, j$ of the matrix $W^k$ would represent the number of paths of length $k$ between spatial units $i$ and $j$. Also, $trW^2$ represent $2 \times$ the number of edges in the graph, $trW^3$ represent $6 \times$ the number of triangles in the graph, and so on. As it will be shown later, the properties of the spatial stochastic process will depend on the traces of powers of the spatial weight matrix, therefore this geometrical interpretation can be useful. Also, there is a vast literature on how the graph eigenvalues are related to the structure of the graph, its degree of connectivity, etc., and this could be exploited to better understand the properties of the associated spatial stochastic processes, but this line of research will not be pursued in this work.

Finally, it is worth mentioning that several empirical work use a standardized version of the neighborhood structure, where the rows of the weight matrix ($W$) add to unity. This approach allows the interpretation of the weight matrix as a spatial lag operator. Although widely used, it is not always highlighted that this normalization could materially change the covariance structure of the underlying stochastic process.

1.3.3 Parallel Between Graph Theory and Specification of Lattice Models

The underlying structure of lattice models in spatial econometrics translates smoothly to the mathematical definition of a graph, and some results and structures from graph theory can be used to gain insights into the underlying economic problem being modeled by the spatial neighbors structure described in definition 1.3.4. Definitions below are from Diestel (2005).

1.3.5 Definition (Graph, Vertex, Edge). A graph is a pair of sets $G = (V, E)$ where $E \subset V \times V$. Elements of $V$ are the vertices (or nodes) of the graph $G$ (denoted $V(G)$), and elements of $E$ are the edges of graph $G$ (denote $E(G)$). The number of vertices and edges of a graph $G$ is represented by $|V(G)|$ and $|E(G)|$, respectively. $|V|$ is also called the order of the graph. Graphs may be finite or infinite, depending on its order.

1.3.6 Definition (Neighbor, Degree). Let $G = (V, E)$ be a non-empty graph. Vertices $x, y \in V$ are neighbors if $(x, y) \in E \subset V \times V$ (if they are connected by an edge). The set of neighbors of a vertex $x \in V$ is denoted by $N(x)$. The degree of a vertex $x \in V$, denoted $d(x)$ is the number of edges at $x$ ($|E(x)|$), which is equal to the number of neighbors of $x$. If all the vertices of $V$ have the same degree $k$, then $G$ is called a $k$-regular graph. The minimum degree of graph $G$ is $\delta(G) = \min \{d(x) \mid x \in V\}$ and the maximum degree is $\Delta(G) = \max \{d(x) \mid x \in V\}$.
Two quantities are usually illustrative of the graph structure:

\[ d(G) = \frac{1}{|V|} \sum_{x \in V} d(x) \]

is the average degree of graph \( G \), and \( \varepsilon(G) = \frac{|E|}{|V|} \) is the average number of edges per vertex. Both are related: \( d(G) = 2\varepsilon(G) \).

From the above mentioned definitions, it is clear that a lattice model can be represented by a graph structure. Set \( V = \Gamma \) and \( E = \mathcal{N} \). Then the graph \( G = (V, E) = (\Gamma, \mathcal{N}) \) represents the neighboring structure embedded in the model. This will allow us to borrow some results from graph theory, when needed, to facilitate the interpretation and analysis of the lattice models.

In particular, graph theory allows some interesting ways to create different spatial weight matrices. For instance, Anselin (1988) builds spatial weight matrices for a grid of regions by letting two regions that share a border to be considered neighbors (rook-type). If one allows for neighbors to share either a border or a vertex the resulting structure is called queen-type. Alternatively, one may define the nearest neighbor graph where spatial unit \( \gamma \in \Gamma \) is neighbor of \( \gamma' \) if the distance from \( \gamma \) to \( \gamma' \) is no larger than from \( \gamma \) to any other spatial unit \( \gamma'' \in \Gamma \). Related to that is the \( k \)-nearest neighbor graph where two spatial units \( \gamma, \gamma' \) are considered neighbors if the distance between them is among the \( k \)-th smallest distance from \( \gamma \) to other spatial units in \( \Gamma \).

There are also some algorithms that use the Delaunay triangulation (which is the dual graph of a Voronoi diagram) to construct a neighbor relation (Bivand et al., 2010). For instance, the relative neighbor graph defines that two spatial units \( \gamma \) and \( \gamma' \) are neighbors if there does not exist a \( \gamma'' \in \Gamma \) that is closer to both \( \gamma \) and \( \gamma' \) than they are to each other. The Gabriel graph expresses a notion of nearness among spatial units. Two spatial units \( \gamma \) and \( \gamma' \) are neighbors if the closed ball of which the line segment between \( \gamma \) and \( \gamma' \) is a diameter contains no other spatial unit in \( \Gamma \). In other words, \( \gamma \) and \( \gamma' \) are neighbors if \( \rho(\gamma, \gamma') \leq \min \left( \sqrt{\rho(\gamma, \gamma'')^2 + \rho(\gamma', \gamma'')^2} \mid \gamma'' \in \Gamma \right) \).

The sphere of influence graph or SOI is defined in the following way: let \( \rho_\gamma \) be the distance between the spatial unit \( \gamma \) and its nearest neighbor in \( \Gamma \) and let \( C_\gamma \) be the circle centered in \( \gamma \) with radius \( \rho_\gamma \). Then \( \gamma \) and \( \gamma' \) are considered neighbors if \( C_\gamma \) and \( C_{\gamma'} \) intersect in at least two points.

Most of this structures will be used to illustrate the results obtained in the subsequent chapters.

1.4 Using Models with Spatial Dependence in Economics

The use of models with spatial dependence in economics might be appropriate both in cases where there is some theoretical ground for modeling spatial interaction among observations, or when no such theoretical basis exist but nevertheless the observations can be regarded as having spatial dependence. These points are addressed below.

1.4.1 Underlying Model Assumes Spatial Dependence

The general idea of spatial dependence can be illustrated by Tobler’s (1979) first law of geography: "everything is related to everything else, but near things are more related than distant things". This idea applies directly to some economic models, e.g., trade models, technology spillovers, etc., where the spatial dependence structure results directly from the underlying economic model. Case (1991) is an example of the use of spatial models in the analysis of economic problems. His article discuss some processes that can generate a spatial dependence structure, focusing

\footnote{Note, however, that in competition models one can expect neighbors to be different; this would be the case, for instance, when neighbor areas compete for scarce natural resources.}
on demand models where consumer's utility is higher when its consumption basket is similar to its neighbors' consumption basket. The author compares spatial dependence models with panel data models with fixed effects, and highlights the trade-offs between the two approaches. Specifically, the estimation of fixed effects is not possible when there is no intra-regional variation in the variables of interest, as these are perfectly correlated with the fixed effects. Moreover, in some cases, the interest might be specifically in the spatial component. For instance, if one is interested in testing if consumers' choices are related to the behavior of some reference groups, then the magnitude and direction of interactions among the consumers is the relevant variable, and this would be difficult to extract from a fixed effect model. The author also emphasizes that if a spatial process is indeed the underlying source behind regional effects, then a spatial model that limits the random effects to be spatially correlated is more efficient than a model that does not impose such restriction, therefore yielding more precise estimates for the model's parameters.

Another example, Pinkse and Slade (1998) develop tests to detect the presence of spatial dependence in the covariance structure and estimation methods applied to discrete choice models, and apply the methodology to test for agglomeration or dispersion in the retail gasoline contracts. Pinkse et al. (2002) estimate a model of spatial competition applied to the retail gasoline market, aiming at discriminating between models of global competition and models of local competition, where products compete only with their neighbors.

1.4.2 Spatial Dependence Results from Specification Errors, Omitted Variables

Even when a spatial dependence structure is not directly explicit in the underlying economic model, it can be present in the sample data. Spatial dependence can result (i) from omitted variables that depend on the location of the sample point in space, (ii) from misspecification of the true underlying data generating process (e.g., the DGP in the North region is different from the South region - regional heterogeneity), or even (iii) from measurement problems. For example, if the boundaries defined for each sample point (e.g., municipalities) do not reflect the true DGP, it is likely that the measured variables for the different sample points will show patterns of spatial dependence.8

Ignoring spatial effects could lead to biased variance estimators, inefficient estimators, bias and inconsistency in OLS estimators, among others (Anselin, 1988). Due to the variety of reasons that can originate a spatial dependence structure in the data and the problems of ignoring it, some author suggest as good practice to consider spatial dependence explicitly in any model to start with, and then to test whether the hypothesis of spatial dependence can be rejected. This would likely make the proposed models more robust to specification errors, omitted variables, and measurement errors.

1.5 Spatial Dependence in the First Two Moments

As previously emphasized, a key issue in the analysis of spatially dependent data is to model its dependence structure. In some cases, the problem under scrutiny might help to define whether the dependence structure will be modeled trough its mean process or trough its covariance structure. However, for most cases, this differentiation is not clear. Cressie (1993) refers to large scale variation when the spatial dependence is captured by the process mean, and small scale variation when its captured by the covariance structure. Alternatively, Anselin (1988) classifies the spatial effects into spatial dependence and spatial heterogeneity, where spatial dependence, in general, refers to the mean structure of the stochastic process, while spatial heterogeneity is a mix between the spatial effects in the covariance structure and heterogeneous models whose structure

8For instance, unemployment rate in neighboring municipalities usually show patterns of spatial dependence, as the work force can move with lower cost between adjacent areas.
depends on its location in space\textsuperscript{9}. Cliff and Ord (1981a) mention the concept of interaction (among spatial units) and reaction (to some other covariate) to refer to spatial dependence in the covariance and in the mean structure of the process, respectively. To avoid digging into specifics, throughout my dissertation the term spatial dependence will be used in a broad context, and when needed a distinction will be made between modeling the spatial dependence via mean structure or via covariance structure.

1.5.1 Cressie’s Critique

Note, however, that in the majority of the spatial econometrics literature, the models are defined in a way where both spatial effects (in the mean and covariance structure of the process) are mixed. For instance, the following model is usually found in the literature:

\[
\begin{align*}
Y &= \rho W_1 Y + X\beta + U \\
U &= \lambda W_2 U + \varepsilon \\
\varepsilon &\sim iid(0, \sigma^2 I)
\end{align*}
\]

where \(Y, U\) are \(n \times 1\) vectors, \(\rho, \lambda, \sigma\) are scalars, \(\beta\) is a \(k \times 1\) vector associated with \(k\) exogenous covariates \(X\), and \(W_1, W_2\) are \(n \times n\) matrices representing the spatial structure (spatial weight matrix) associated to the spatial autoregressive process for the independent variable \(Y\) and for the error term \(U\), respectively. This model implies the following structure for the first two moments of \(Y\):

\[
\begin{align*}
E[Y] &= (I - \rho W_1)^{-1} X\beta \\
Var[Y] &= \sigma^2 (I - \rho W_1)^{-1}(I - \lambda W_2)^{-1}(I - \lambda W_2')^{-1}(I - \rho W_1')^{-1}
\end{align*}
\]

which shows that both the mean and the covariance of the process depend on the parameter \(\rho\) and the matrix \(W_1\). Cressie (1993) argues that the decision to include the spatial dependence in the mean or in the covariance structure depends on the problem under analysis, and it is usually a trade-off between model’s fit and parsimony. Nevertheless the author suggests as good practice to model both effects separately. Arnaiz et al. (2008) is a good example where the author struggles to interpret the estimated model’s parameters due to the fact that one parameter affects both the mean and the covariance structure of the model (section 3.2 explains this in more details).

1.6 Intent of this Dissertation

I have highlighted in the previous sections that it might be a good practice to consider spatial dependence in any model to start with, and then test for its presence. This could potentially make the suggested models more robust to specification errors, omitted variables, and measurement errors. It also might be a good practice to disentangle the spatial dependence parameters from the mean and the covariance structure – following Cressie’s critique.

However, by starting with some spatial dependence structure there is a risk of introducing a spurious spatial dependence in the model – therefore highlighting the importance of having a good understanding of the tests to detect spatial dependence and its properties.

Chapter 2 of my dissertation investigates size and power properties of tests for detecting spatial dependence in a specific but commonly used set of spatial stochastic process defined on a lattice – called SAR, or spatial autoregressive process. I expand on the existing literature by

\textsuperscript{9}In parametric models this heterogeneity could happen, for instance, in models where the parameters were functions of the sample’s location (or relative location) in the space. Note, however, that some problems caused by the spatial heterogeneity can be solved using standard techniques, e.g., varying parameters, random coefficients, structural breaks, etc., but in some cases knowledge of the data spatial structure could lead to more efficient procedures (Anselin, 1988).
allowing the spatial weight matrix (or neighborhood structure) to be a nonlinear function of the spatial dependence parameter. Moreover, as highlighted in section 1.3.2, the choice of the spatial weight matrix is often ad hoc and thus it is important to be able to compare test properties among different specifications. Instead of the commonly used Monte Carlo simulation I use Edgeworth and saddlepoint series to approximate the small sample distribution of the test statistics. Moreover, to my knowledge, none of the Monte Carlo experiments in the literature correct for size when comparing power of the tests – something that is easily done in the framework I have used.

I also obtain the power envelope for testing the presence of spatial dependence in the SAR process using (i) the optimal test statistic (in the Neyman-Pearson sense) when the variance of the error term is known, and (ii) the optimal invariant test statistic when the variance is unknown. The optimal invariant test converges to the score test (Lagrange multiplier) when the alternative hypothesis converges to the null of no spatial dependence and is, therefore, locally uniformly most powerful invariant (LUMPI). I have found that the LUMPI test is virtually uniformly most powerful in the sense that, for practical purposes, its power is very close to the power envelope. But this near optimality might not hold for spatial weight matrices with designs different than the ones I have used. Therefore, I suggest a practical procedure to build a test that, while not UMP, retain good power properties in a wider range for the spatial parameter when compared to the LUMPI test. The proposed test statistic allows me to analyze how the power envelope behaves for different sample sizes and different designs for the weight matrix. I find that power increases with sample size and with the spatial dependence parameter – which agrees with the literature. However, I call into question the consensus view that power decreases as the spatial weight matrix becomes more densely connected. This finding in the literature reflects an error of measure because changing the weight matrix changes the structure of the process. Therefore when looking at the same spatial dependence parameter the hypothesis being compared are at very different statistical distance from the null. After adjusting for this, the power is larger for alternative hypothesis further away from the null – as one would expect.

Chapter 3 of my dissertation builds on the second suggested good practice: to use models where the spatial dependence parameters that enter the mean structure are different from the ones in the covariance structure. Anselin (2003) suggests a set of local and global models as an alternative to the traditional Cliff & Ord-type of spatial models to address this problem. However, global and local specifications are non-nested and Anselin himself noted this drawback. The author speculated about the possibility of considering models where, for instance, local externalities would be present in the error term and global externalities in the covariates. He also highlighted that none of these possibilities had received attention in the literature and, to the best of my knowledge, this remains an unexplored area both in theoretical as well as in applied econometrics. In order to overcome this problem and to build on Anselin’s suggestion, I propose a more general parametric spatial stochastic process that nests Anselin’s suggested local and global models into an encompassing structure and allows for a clearer interpretation of the model’s parameters vis-a-vis the existing models. This model provides some insights on the structure of the commonly used model MSAR but they are non-nested. Therefore they become competing models whose validity needs to be tested empirically. I also propose an estimation procedure for the model’s parameters that has the benefit of being easily implementable and computationally simple even when the dimension of the parameter space is not small. The asymptotic distribution of the parameter’s estimators is also derived in the chapter.

Chapter 4 provides an example of an area where the application of spatial econometrics is straightforward: international trade. Tinbergen (1962) and Linnemann (1966) are probably the pioneers in bringing gravity models to use in international trade. The simple idea of gravity models is to express a flow (exports, imports, or total trade) from country i to country j as being directly proportional to their economic mass (e.g., income) and inversely proportional to the distance between them. Tinbergen and Linnemann rationalize the model by arguing that
the exports from \( i \) to \( j \) are greater, the larger is the economic size of \( j \) due to demand factors, and the larger is the economic size of the exporting \( i \) due to supply factors. The volume of trade should also decline with transportation costs and with difficulties of commercial contact, and geographical distance is used as a proxy for them. Both authors also include additional variables, such as dummies for common trade agreements, and dummies for countries that share a border. What has started as an empirical model of trade was followed by the theoretical literature. Several authors (see Anderson, 1979; Bergstrand, 1985; Eaton and Kortum, 2002; Anderson and van Wincoop, 2003, among others) have shown that the gravity equation can be generated from a set of primitives and thus broadened the scope of the gravity equations. For a variety of reasons (see detailed examples in chapter 4) it is reasonable to consider that gravity equations in international trade could still present spatial dependence even after accounting for the distance as an important explanatory variable. The goal of this chapter, therefore, is to provide an informal assessment of the usefulness of adding a spatial dimension to international trade gravity equations. I start by estimating a traditional gravity equation and then use Moran I test statistic to check the residuals for the presence of spatial dependence. The limiting distribution of Moran I test statistic is known to be Gaussian even in the presence of heteroskedasticity (which is likely to be the case) but its dispersion parameter depends on the specific structure of the model. Therefore I perform a residual wild bootstrap of Moran's I test, which confirms that the residuals of the gravity equation are spatially dependent.

Several neighborhood structures (or spatial weight matrices) are used in Moran's test, all elaborated using different graph structures based on the concepts mentioned in sections 1.3.2 and 1.3.3. It is interesting to highlight that in the gravity model of trade the observations are the flows between a pair of countries rather than the countries themselves. Therefore the neighborhood structure needs to be adapted to reflect this fact. I apply the procedure developed in chapter 3 to estimate the new class of spatial autoregressive moving average processes that is suggested in my dissertation. I do not attempt to qualify the estimated parameters nor to test trade models, but only to check, with a simple example, whether the spatial econometrics toolbox would be appropriate to deal with international trade gravity equations.

Finally, the appendices contain some additional theorems and proofs, maps with the neighborhood structures used, additional figures, and the R (R Development Core Team, 2011) programming codes that were used to generate the statistics, tables, figures, and simulations that were used throughout my dissertation.
Chapter 2

Small Sample Inference in Spatial Econometric Models

2.1 Introduction

I have argued in chapter 1 that assuming a spatial dependence structure from the outset has the benefit of potentially making the model under investigation robust to specification errors, omitted variables, and measurement errors. However, there is also a risk of introducing a spurious spatial dependence in the model, thus stressing the importance of understanding clearly the properties of the tests used to detect whether spatial dependence is present.

This chapter of my dissertation aims at shedding some light on the procedures used for testing the presence of spatial dependence on a widely used type of spatial stochastic process (spatial autoregressive process, or SAR) defined on an irregular lattice. I allow for the spatial weight matrix to be a nonlinear function of the spatial dependence parameter, therefore expanding on the existing literature which focus on linear functions.

The classical testing procedures (Wald, score, likelihood ratio), which are valid asymptotically, present some serious size distortions for small to moderate sample sizes. Therefore the need for a closer look at the size and power properties of testing procedures for small samples.

A deeper investigation of the the small sample properties of the classical tests, however, would require either an extensive Monte Carlo study, or more refined approximations to the asymptotic distribution of the statistics of interest\(^1\). As an example of the first approach, Anselin and Florax (1995) consider the properties of tests for several different designs for the spatial weight matrix, and Das et al. (2003) consider the finite sample performance of some estimators for the spatial dependence parameter, but their Monte Carlo experiment uses only one specific design for the spatial weight matrix. The second approach, second or higher order asymptotic expansions, would be simpler for the score statistic, as it is already written as a function only of a random vector which is assumed to follow a SAR stochastic process. For the likelihood ratio and Wald statistics, we would need to obtain a small sample approximation for the distribution of the maximum likelihood (ML) estimator of the spatial dependence parameter, which is asymptotically Gaussian. One such approximation was obtained recently by Bao and Ullah (2007), that developed the second-order bias and mean square error for the ML estimator of the SAR process using a Nagar-type expansion.

In this chapter I will follow the second approach, but instead of looking at the classical tests, I will investigate the size and power properties of testing for the presence of spatial dependence in the SAR model by devising optimum tests from first principles and obtaining higher order asymptotic expansions for their statistics. I use the optimal (in the Neyman-Pearson sense) test statistic to construct the power envelope for testing the presence of spatial dependence when

\(^1\)By adding terms of lower order in the sample size, e.g., terms \(O(n^{-1/2})\) or \(O(n^{-1})\).
the variance of the error term is known, and the optimal invariant test when the variance is unknown.

The characteristics of spatial stochastic processes defined on a lattice were introduced in section 1.3 where some key terms (e.g. spatial weight matrix, neighborhood structure) used in the subsequent sections are defined. I define the spatial autoregressive stochastic process (SAR) in section 2.2, present the assumptions on the spatial weight matrix, define the parameter space, and provide some general lemmas that will be used throughout this article. Section 2.3 defines the testing problem under consideration and draws attention to the size distortions of the classical testing procedures in small samples. Section 2.5 derives the optimum test for the case of known variance, and shows that even in this simple situation, an uniformly most powerful test is not available. In section 2.6, I focus on the class of invariant tests and derive an optimal invariant test for the case of unknown variance, and, again, an uniformly most powerful test is not available. In both cases (known and unknown variances) the optimal tests converge to the score test when the alternative hypothesis is close to the null hypothesis, therefore yielding locally uniformly most powerful tests. To study the small sample properties and to build the power envelope for the optimal tests, I obtain two asymptotic expansions for the distribution of the optimal test statistics: Edgeworth expansion and saddlepoint expansion. These expansions are introduced in sections 2.4.1 and 2.4.2.

Monte Carlo simulations are provided in section 2.7 to assess the accuracy of the proposed approximations, and I investigate also how the properties of the tests change for different designs of the spatial weight matrices. I argue that, for the specific designs of the spatial weight matrices considered in this paper, the locally uniformly most powerful invariant test can be regarded as virtually uniformly most powerful, in the sense that for practical purposes its power is very close to the power envelope for (invariant) testing the presence of spatial dependence in the SAR stochastic process. Nevertheless, as this nearly optimality of the LUMPI test might not hold for a different design for the spatial weight matrix, I suggest a practical procedure to build a test that, while not uniformly most powerful, retain good power properties in a wider range for the spatial dependence parameter when compared to the locally most powerful invariant test (score). Section 2.8 concludes.

2.2 Basic Model, Parameter Space

There are basically three approaches to estimate the dependence structure in a spatial stochastic process: (i) to specify the stochastic process through a data generating process, (ii) to represent the covariance structure parametrically, and (iii) to estimate the covariance structure non-parametrically. For this last approach, one usually needs a large sample, which may not be available. Moreover, for the non-parametric estimation of the covariance structure, the process also needs to be (weakly) stationary, an hypothesis that usually do not hold for stochastic processes defined on an irregular lattice. The parametric representation of the covariance structure, although intuitive, has not been widely used, perhaps due to difficulties in finding a parsimonious, representative parametrization yielding a positive definite covariance matrix which could be defined for a general design of the spatial units. Therefore, the first approach is the one most used in econometrics, and the one followed in this article. Important to highlight that this approach depends strongly on the design of the spatial weight matrix, the way it is normalized, etc., and that in some cases the covariance structure resulting from a given specification ends up being different from the notion of neighborhood that was explicitly or implicitly used to build up the weight matrix. Kelejian and Prucha (1997) draws attention to some potential pitfalls in using a row-normalized weight matrix, and Martelllosio (2008) provides some examples showing some unexpected covariances resulting from a given neighborhood structure and the effects of row-normalization in the covariance structure. These points, however, will not be exploited further in this article.
Drawing from time series terminology, the basic underlying structure of spatial stochastic processes with lattice data \(^2\) is called *spatial autoregressive* (SAR) process \(^3\) (see Anselin, 2003; Cressie, 1993; Cliff and Ord, 1981a). It is a parametric model \((M)\), whose typical data generating process (DGP) \(m \in M\) can be written as

\[
Y = w(\rho)Y + U
\]

\[
U \sim N(0, \sigma^2 I)\tag{2.2.1}
\]

where \(Y\) is an \(n \times 1\) vector of observables, \(w\) is an \(n \times n\) matrix (spatial weight matrix) of known real-valued functions of the spatial autocorrelation parameter \(\rho \in \mathbb{R}\), and \(U\) is an \(n \times 1\) vector of homoskedastic Gaussian disturbances.

Let \(\theta : M \to \Theta\) be the parameter-defining map of the model \(M \equiv (Y, \{F_{\theta}, \theta \in \Theta\})\), thus for each DGP \(m_0 \in M\) there is an unique associated vector of parameters \(\theta_0 \equiv \theta(m_0) = (\rho_0, \sigma_0^2)\) we are interested into (estimating, testing).

Most of the recent literature (both empirical and theoretical), focus only in the particular case where the spatial weight matrix is linear \((w(\rho) = \rho M)\), where \(M\) is an \(n \times n\) matrix of known values. In this paper I allow for nonlinearities in the functions which compose the elements of the spatial weight matrix.

This SAR stochastic process can be regarded as an equilibrium relationship as long as \(\det L(\rho) \equiv \det (I - w(\rho)) \neq 0\). In this case

\[
Y = L(\rho)^{-1}U = (I - w(\rho))^{-1}U
\]

and the parameter space for the spatial autocorrelation coefficient \(\rho\), therefore, could be considered as the entire real line \(\mathbb{R}\), excluding the set where \(\det L(\rho) = 0\) (a countable union of disjoint intervals). However, as the focus of the empirical research is usually to test whether spatial dependence is present or not, it seems natural to focus on the largest interval containing the origin.

In order to properly define the parameter space for \(\rho\), we proceed with the following assumptions:

**2.2.1 Assumption (Elements of spatial weight matrix).** Assume that:

(a) the elements of the spatial weight matrix, \(w_{ij}(\rho)\), are real-valued, twice differentiable, non-decreasing functions defined in a compact set \(\mathcal{F} \subset \mathbb{R}\) containing the origin, for all \(i, j = 1, \ldots, n\);

(b) \(w_{ii}(\rho) = 0\) for all \(\rho \in \mathbb{R}\) and \(i = 1, \ldots, n\);

(c) \(w_{ij}(0) = 0\) for all \(i, j = 1, \ldots, n\);

(d) for each spatial unit \(i\), there is at least another spatial unit \(j\) such that \(W_{ij}(\rho) \equiv \frac{\partial w_{ij}(\rho)}{\partial \rho} > 0\); and \(^4\)

(e) \(\text{rank}(w(\rho))/n\), for \(\rho \neq 0\), does not vanish as \(n \to \infty\).

---

\(^2\)Where the index set which define the stochastic process is a finite or a countable collection of points, called **spatial units**.

\(^3\)This acronym is also used in the spatial statistics literature to represent the *simultaneous autoregressive* random field; in the present context, however, both stochastic processes are the same.

\(^4\)This ensures that for each spatial unit \(i\), there is at least another spatial unit \(j\) whose association with \(i\) increases with \(\rho\). This assumption can be eliminated if existence of a valid parameter space can be validated by other methods; as an example, consider the *time series AR(1)* process, where the eigenvalues of the spatial weight matrix are all zeros and therefore the parameter space is the whole real line. In this case, assumption (d) is not satisfied, but the parameter space is well defined.
Assumptions 2.2.1 are satisfied for the typical setting where the spatial weight matrix is linear (see Kelejian and Prucha, 1999; Lee, 2004, for example). They aim at establishing the existence of a parameter space for \( \rho \) around the origin, where the inverse of \( L(\rho) \) exists. Assumption (e) will be used in sections 2.5 and 2.6 to establish the asymptotic normality of the proposed test statistics.

The following two propositions characterize the behavior of \( \det L(\rho) \equiv \det (I - w(\rho)) \) for \( \rho \) in the neighborhood of zero, and help to delineate the relevant parameter space for the SAR stochastic process.

**2.2.2 Proposition.** The function \( \det L(\rho) \) attains a local maximum at \( \rho = 0 \).

**Proof.** Taking the differential of \( \det L(\rho) \equiv |I - w(\rho)| \):

\[
d \det L(\rho) = \det L(\rho) \text{tr} \left( L(\rho)^{-1}(-1) \frac{\partial w(\rho)}{\partial \rho} d\rho \right)
= - \det L(\rho) \text{tr} \left( L(\rho)^{-1}W(\rho) \right) d\rho
\]

therefore

\[
\frac{\partial \det L(\rho)}{\partial \rho} = - \det L(\rho) \text{tr}(L(\rho)^{-1}W(\rho))
\]  

(2.2.2)

Taking the differential of \( \frac{\partial \det L(\rho)}{\partial \rho} \):

\[
d \left( \frac{\partial \det L(\rho)}{\partial \rho} \right) = -d(\det L(\rho)) \text{tr}(L(\rho)^{-1}W(\rho)) - \det L(\rho) \text{tr}(d(L(\rho)^{-1}W(\rho)))
= \det L(\rho) \left[ \text{tr}^2(L(\rho)^{-1}W(\rho)) - \text{tr} \left( (L(\rho)^{-1}W(\rho))^2 + L(\rho)^{-1}W^{(1)}(\rho) \right) \right] d\rho
\]

therefore

\[
\frac{\partial^2 \det L(\rho)}{\partial \rho^2} = \det L(\rho) \left[ \text{tr}^2(L(\rho)^{-1}W(\rho)) - \text{tr} \left( (L(\rho)^{-1}W(\rho))^2 + L(\rho)^{-1}W^{(1)}(\rho) \right) \right]
\]  

(2.2.3)

where \( W^{(1)}(\rho) \equiv \frac{\partial w(\rho)}{\partial \rho} \). Evaluating the first and second derivatives at \( \rho = 0 \) results

\[
\frac{\partial \det L(\rho)}{\partial \rho} \bigg|_{\rho=0} = - \text{tr} W(0) = 0
\]  

(2.2.4)

\[
\frac{\partial^2 \det L(\rho)}{\partial \rho^2} \bigg|_{\rho=0} = \text{tr}^2 W(0) - \text{tr} \left( W(0)^2 + W^{(1)}(0) \right) = - \text{tr} W(0)^2 < 0
\]  

(2.2.5)

thus \( \det L(\rho) \) has a local maximum at \( \rho = 0 \). \( \square \)

**2.2.3 Proposition.** There are positive real numbers \( \rho_n, \bar{\rho}_n \in \mathbb{R}_{+}, \) function of the eigenvalues of the spatial weight matrix \( w(\rho) \), such that the function \( \det L(\rho) \) is strictly positive in the open interval \( (-\bar{\rho}_n, \rho_n) \). The subscript \( n \) reminds that the bounds of the interval usually change with the sample size.

**Proof.** Assume \( \rho > 0 \). By assumption 2.2.1, \( w(\rho) \) is a nonnegative matrix. By the extension of the Perron-Frobenius theorem to reducible matrices (see Lancaster, 1969, theorem 9.3.1, p.288), the spectral radius of \( w(\rho) \), denoted \( \lambda_{w(\rho)} \), is a real eigenvalue of the matrix \( w(\rho) \) and all the
other eigenvalues are within the circle with radius \( \lambda_{w(\rho)} \) in the complex plane. Moreover, the spectral radius can be bounded by:

\[
\rho(\rho) = \min_i \sum_{j} w_{ij}(\rho) \leq \lambda_{w(\rho)} \leq \max_i \sum_{j} w_{ij}(\rho) = r(\rho)
\]

Note that \( \lambda_{w(0)} = 0 \), as \( r(\rho) = r(\rho^*) = 0 \). By the mean value theorem, \( w_{ij}(\rho) = w_{ij}(0) + W_{ij}(\rho_1^*) \rho \), where \( \rho_1^* \in (0, \rho) \). Therefore, \( r(\rho) = \rho \min_i \sum_j W_{ij}(\rho_1^*) \), which by assumption 2.2.1 is a continuous, strictly increasing function of \( \rho \).

Therefore, there is an \( \overline{P} \in \mathbb{R}_+ \) such that \( \lambda_{w(\overline{P})} = 1 \). As a result, for all \( \rho \in [0, \overline{P}] \) the eigenvalues of \( w(\rho) \) are within the unit circle and the real eigenvalues of \( L(\rho) \) are all in the interval \((0, 2)\), thus \( \det L(\rho) > 0 \) for all \( \rho \in [0, \overline{P}] \).

The case \( \rho < 0 \) is not analogous. In this case we could define \( m(\rho) \equiv -w(\rho) \) to be a nonnegative matrix. In order to set the bounds on the real eigenvalues of \( L(\rho) = I - w(\rho) = I + m(\rho) \) to be larger than zero, we need bounds on the real negative eigenvalues of \( m(\rho) \).

However, there is nothing that guarantees that there will be any real negative eigenvalue for \( m(\rho) \). And, if there is a negative eigenvalue (with multiplicity greater than 1), it is possible that a small change in \( \rho \) will split it into distinct eigenvalues, possibly complex. As a result, tight bounds for negative values of the parameter \( \rho \) need to be established in a case by case basis, which could become cumbersome in the case where the spatial weight matrix is not composed of linear functions of \( \rho \).

To overcome this problem, we can impose bounds for the negative values of \( \rho \) that do not pretend to be tight. With an argument analogous to the case \( \rho > 0 \) we can find \( \underline{\rho} > 0 \) such that all the eigenvalues of \( m(\rho) \) will lie in the unit circle for \( \rho \in (\underline{\rho}, 0] \). As a result, even if \( m(\rho) \) has any real negative eigenvalues, the real eigenvalues of \( L(\rho) = I + m(\rho) \) will be strictly greater than zero and therefore \( \det L(\rho) > 0 \) for all \( \rho \in (\underline{\rho}, 0] \).

Remark. Note that for specific matrices \( w(\rho) \), the bounds for the negative values for the parameter \( \rho \) can be sharpened, i.e., the interval where \( \det L(\rho) \neq 0 \) can be enlarged. For example, it is usually the case that for symmetric matrices, with linear functions of \( \rho \), the modulus of the largest negative eigenvalue is significantly smaller than the spectral radius \( \overline{P} \), therefore the parameter space could be extended to \( (\rho_{\text{min}}, \overline{P}) \supset (\underline{\rho}, \overline{P}) \), where \( -1 < \lambda_{\text{min}}(\rho_{\text{min}}) \) is the smallest eigenvalue of \( w \) (real, as the matrix is symmetric).

Remark. Despite the possibility of finding broader parameter space for the spatial autoregressive parameter \( \rho \) for specific settings, often it is useful to focus on the interval \( (\underline{\rho}, \overline{P}) \) as for all \( \rho \) in this interval the inverse of \( I - w(\rho) \) can be expanded into a convergent infinite series:

\[
(I - w(\rho))^{-1} = \sum_{i=0}^{+\infty} w(\rho)^i
\]

Remark. An even more conservative parameter space could be established by setting \( \overline{P} > 0 \) such that \( \max_i \sum_j w_{ij}(\overline{P}) = 1 \). This implies the absolute value of the eigenvalues of \( w(\rho) \) would be smaller than unity for all \( \rho \in (\underline{P}, \overline{P}) \).

Propositions 2.2.2 and 2.2.3 allow us to characterize the relevant parameter space \((P)\) for \( \rho \) as any compact subset of \( (\rho_{\text{min}}, \rho_{\text{max}}) \cap (\underline{P}, \overline{P}) \) containing the origin. However, as it is clear from the proof of proposition 2.2.3, the parameter space can shrink to \( \{0\} \) as \( n \to \infty \), unless some bounds on the sum of elements of the spatial weight matrix are imposed as the sample size grows. This is the subject of the following assumption:

---

\(^5\)The eigenvalues are continuous functions of the elements of the matrix, see Lancaster (1969)

\(^6\)In this case, the parameter space would become \((\rho_{\text{min}}, \rho_{\text{max}}) \cap (\underline{P}, \overline{P})\).
2.2.4 Assumption (Derivatives of spatial weight matrix uniformly bounded). The row and column sums of the matrix $W(\rho)$ are uniformly bounded in absolute value, i.e., there exist positive constant $c$ that do not depend on $n$, such that

\[
\max_{i \in \{1, \ldots, n\}} \sum_{j=1}^{n} |W_{ij}(\rho)| \leq c, \quad \text{and} \\
\max_{j \in \{1, \ldots, n\}} \sum_{i=1}^{n} |W_{ij}(\rho)| \leq c, \quad \forall n \in \mathbb{N}, \quad \forall \rho \in \mathcal{P}
\]  

(2.2.7) (2.2.8)

This assumption is satisfied, for instance, in (i) the typical case where the spatial weight matrix is row normalized (i.e., \( \sum_j w_{ij}(\rho) = \rho \)), (ii) when each spatial unit has a finite number of neighbors, among others.

The following proposition shows that, under assumption 2.2.4, the parameter space $\mathcal{P}$ is not degenerated asymptotically.

2.2.5 Proposition. If the row sums of the matrix of derivatives of the spatial weight matrix, $W(\rho)$, are uniformly bounded in absolute value, then there exists a non-degenerated set $\mathcal{P}$, including the origin, independent of $n$, such that $L(\rho)$ is non-singular for all $\rho \in \mathcal{P}$. The parameter space $\mathcal{P}$, then, can be chosen as any compact subset of $\mathcal{P} \cap \mathcal{P}$ containing the origin.

Proof. By the Geršgorin’s theorem (see Lancaster, 1969), all the eigenvalues of $w(\rho)$ satisfy

\[
|\lambda_i(w(\rho))| \leq \max_{i} \sum_{j=1}^{n} |w_{ij}(\rho)|, \quad \forall i = 1, \ldots, n
\]

(2.2.9)

Define $c_n(\rho) \equiv \sum_{j=1}^{n} |w_{ij}(\rho)|$, a continuous function of $\rho$. Then, for $\rho \neq 0$, by the mean-value theorem:

\[
c_n(\rho) = |\rho| \sum_{j=1}^{n} W_{ij}(\rho^*_{ij}(\rho))
\]

(2.2.10)

\[
\lim_{n \to \infty} c_n(\rho) = |\rho| c_i(\rho)
\]

(2.2.11)

where the limit $c_i(\rho)$ exists and is continuous by assumption 2.2.4. Taking limits of equation 2.2.9

\[
|\lambda_i(w(\rho))| \leq |\rho| \max_{i} c_i(\rho) \leq |\rho| c, \quad \forall i = 1, \ldots
\]

(2.2.12)

the right hand side is an increasing function of $|\rho|$, which is zero for $\rho = 0$ and reaches its maximum value at $m = \rho_{\text{max}} - c$, where $\rho_{\text{max}} = \max \{ \max \{ \mathcal{P} \}, -\min \{ \mathcal{P} \} \}$. If $m < 1$ then $L(\rho)$ is non-singular for all $\rho \in \mathcal{P} \equiv \mathcal{P}$. If $m \geq 1$, then there is an $\mathcal{P} \in (0, \rho_{\text{max}})$ such that $|\mathcal{P}| c = 1$ and det $L(\rho) \neq 0$ for all $\rho \in \mathcal{P} \equiv (\mathcal{P} \cap \mathcal{P})$. Then the parameter space $\mathcal{P}$ can be taken as any compact subset of $\mathcal{P}$ containing zero.

Corollary. The row sums of $w(\rho)$ are uniformly bounded in absolute value, for all $\rho \in \mathcal{P}$.

Proof. By the mean-value theorem, $w_{ij}(\rho) = \rho W_{ij}(\rho^*_{ij}(\rho))$. Thus, $\sum_{j=1}^{n} |w_{ij}(\rho)| = |\rho| \sum_{j=1}^{n} |W_{ij}(\rho^*_{ij}(\rho))|$. Taking limits, $\lim_{n \to \infty} \sum_{j=1}^{n} |w_{ij}(\rho)| \leq |\rho| c$, for all $i = 1, \ldots$ and $\rho \in \mathcal{P}$, therefore the row sums of $w(\rho)$ are uniformly bounded in absolute value.

Under assumptions 2.2.1 and 2.2.4 then, we can find an asymptotically non-degenerate compact set $\mathcal{P} \subseteq \mathcal{P}$ for the spatial parameter $\rho$, and therefore we can define the parameter space for the SAR stochastic process as $\Theta \equiv \mathcal{P} \times \mathbb{R}_{++}$. As det $L(\rho)$ is positive for all $\theta \in \Theta$, then $\{F_\theta, \theta \in \Theta\}$ is a family of probability distributions, whose density is given by:

\[
f(y, \theta) = (2\pi\sigma^2)^{-n/2} \det L(\rho) e^{-\frac{\rho^2(y, \theta)}{2\sigma^2}}
\]

(2.2.13)
The following section states some lemmas regarding traces and products of matrices whose row and column sums are uniformly bounded in absolute value. They will be useful in the remaining sections in order to establish the order of magnitude of the cumulants of some statistics of interest.

### 2.2.1 Uniformly Bounded Matrices

When the row and column sums of the elements of a given matrix are uniformly bounded in absolute value (see Assumption 2.2.4 for this definition) we will say that the matrix is uniformly bounded, for short.

#### 2.2.6 Lemma (Product of uniformly bounded matrices). Let $A$ and $B$ be two $n \times n$ matrices whose row and column sums are uniformly bounded in absolute value. Then row and column sums of $AB$ are uniformly bounded in absolute value.

\[
\sum_{j=1}^{n} |ab_{ij}| = \sum_{j=1}^{n} \left| \sum_{k=1}^{n} a_{ik} b_{kj} \right| \leq \sum_{j=1}^{n} \sum_{k=1}^{n} |a_{ik}||b_{kj}| \leq c^2, \quad \forall n \in \mathbb{N}
\]

\[\square\]

#### 2.2.7 Lemma (Trace of uniformly bounded matrices). Let $A$ be an $n \times n$ matrix whose row sums are bounded in absolute value. Then $\text{tr} A = O(n)$.

\[
|\text{tr} A| = \left| \sum_{i=1}^{n} \lambda_i(A) \right| \leq \sum_{i=1}^{n} |\lambda_i(A)| \leq nc
\]

where the second inequality derives from Geršgorin’s theorem an the fact that $A$’s row sums are bounded in absolute value. \[\square\]

#### 2.2.8 Lemma (Cumulants of quadratic forms in Gaussian variables\(^{8}\)). If $A$ is a symmetric $n \times n$ matrix and $v \sim N(0, I_n)$, then the cumulants of the quadratic form $v'Av$ are given by $k_s = 2^{s-1}(s-1)!\text{tr} A^s$.

\[
v'Av = v'^tS'ASv^* = \sum_{i=1}^{n} \lambda_i(A) v_i'^2
\]

where $v_i'^2 \sim \chi_i^2 \quad (i = 1, \ldots, n)$ are independent random variables. The cumulant generating function of $x_i \equiv \lambda_i v_i'^2$ is $K_{x_i}(\beta) = -\frac{1}{2} \ln (1 - 2\lambda_i \beta)$. The domain of the cumulant generating function is given by (i) $\beta \in \left(\frac{1}{2\lambda_{\min}}, \frac{1}{2\lambda_{\max}}\right)$ when the largest and smallest eigenvalues of $A$ are, respectively, positive and negative, (ii) $\beta \in (-\infty, \frac{1}{2\lambda_{\max}})$ when all eigenvalues are non-negative, and (iii) $\beta \in \left(\frac{1}{2\lambda_{\min}}, \infty\right)$ when all the eigenvalues are non-positive. In all the situations the domain of the cumulant generating function includes an open set around the origin, therefore all

---

\(^7\)See Lee (2004) for a reference.

\(^8\)Lancaster (1954) is an earlier reference for this proof, but the author does not mention that the matrix $A$ needs to be symmetric, or more generally, normal ($A^* A = AA^*$).
the moments (and cumulants) of $x$, are finite (see Kolassa, 1997, Lemma 2.3.6) and are given by the derivative of the moment (cumulant) generating function evaluated at $\beta = 0$. Define $x \equiv \sum_{i=1}^{n} x_i$. Then the cumulant generating function of $x$ is just the sum of the cumulant generating function of $x_i$,

$$K_x(\beta) = -\frac{1}{2} \sum_{i=1}^{n} \ln (1 - 2\lambda_i \beta)$$

Taking derivatives

$$\frac{dK(\beta)}{d\beta} = -\sum_{i=1}^{n} \frac{\lambda_i}{1 - 2\lambda_i \beta}$$

$$\frac{d^2 K(\beta)}{d\beta^2} = 2 \sum_{i=1}^{n} \frac{\lambda_i^2}{(1 - 2\lambda_i \beta)^2}$$

$$\frac{d^3 K(\beta)}{d\beta^3} = 8 \sum_{i=1}^{n} \frac{\lambda_i^3}{(1 - 2\lambda_i \beta)^3}$$

$$\vdots$$

$$\frac{d^s K(\beta)}{d\beta^s} = 2^{s-1}(s-1)! \sum_{i=1}^{n} \frac{\lambda_i^s}{(1 - 2\lambda_i \beta)^s}$$

therefore $k_s = \frac{d^s K(\beta)}{d\beta^s} \big|_{\beta=0} = 2^{s-1}(s-1)! \text{tr} A^s$. $\square$

### 2.3 Testing Problem

The testing procedure under consideration is defined by the model $(\mathcal{Y}, \{F_0, \theta \in \Theta\})$ and partitioning the family of probability distributions (i.e., the parameter space) in two disjoint sets: $\Theta_0 = \{0\} \times \mathbb{R}_{++}$ and $\Theta_1 = \mathcal{P} \setminus \{0\} \times \mathbb{R}_{++}$, where $\Theta_0$ represents the absence of spatial dependence in the model, i.e., $Y \sim N(0, \sigma^2 I)$ is a Gaussian stochastic process.

The goal of the testing procedure is, given a realization “$y$” of the spatial autoregressive stochastic process (SAR), to decide in which set $\Theta_0$ or $\Theta_1$ the underlying DGP lies.

One of the sets is called the null hypothesis ($H_0$) and the other, the alternative hypothesis ($H_1$). A non-randomized test is a function $\psi : \mathcal{Y} \rightarrow \{0, 1\}$, where 0 is defined as the decision to accept the null hypothesis and 1 the decision to reject it. The Neyman testing principle aims at controlling for the risk of Type I error, i.e., to establish an upper limit in the probability of incorrectly rejecting $H_0$ when it is true, and to minimize the Type II error (to accept $H_0$ when it is false). Therefore, which one of the sets is chosen to be the null hypothesis largely depends on the objective of the analysis. Throughout this paper we will call the absence of spatial dependence the null hypothesis.

The natural starting point for testing for the presence of spatial dependence is to use the three classical testing principles - likelihood ratio, Lagrange multiplier (score test), and Wald. As the maximum likelihood estimator of $\theta$ in the SAR model (equation 2.2.1) is asymptotically Gaussian (see Lee, 2004, for the case where $w(\rho)$ is linear), these three test statistics have known asymptotic distributions.

The $\alpha$-size test $\psi$ for testing for the presence of spatial dependence ($H_0 : \theta \in \Theta_0$ vs. $H_1 : \theta \in \Theta_1$) using the classical testing principles can be summarized as

$$\psi(y) = \begin{cases} 1 & ST(y) > c_\alpha \\ 0 & ST(y) < c_\alpha \end{cases}$$

(2.3.1)
where $c_\alpha$ is the $1 - \alpha$ quantile of a chi-square distribution with one degree of freedom ($\chi^2_1$), and

$$ST'(y) = 2 \ln \det L(\hat{\beta}) - n \ln \left( \frac{y' L(\hat{\beta}) L(\hat{\beta}) y}{y'y} \right)$$

(likelihood ratio) \hspace{1cm} (2.3.2)

$$ST(y) = \left( \frac{y' W(0)y}{y'y} \right)^2 \frac{n^2}{\text{tr} [W(0)^2 - W(0)yW(0)'y]}$$

(score test) \hspace{1cm} (2.3.3)

$$ST(y) = \frac{(\hat{\beta})^2}{\text{Var}(\hat{\beta})}$$

(Wald test) \hspace{1cm} (2.3.4)

where $\hat{\beta}$ is the ML estimate for the spatial dependence parameter with no restrictions imposed.

My goal is to investigate the small sample properties (size, power) of the testing procedures for detecting the presence of spatial dependence in the SAR stochastic process, and how they are affected by different designs of the spatial weight matrix $w(\rho)$.

A simple Monte Carlo simulation shows that even for very simple and well-behaved designs for the spatial weight matrix, the actual size of the classical tests is very different from the nominal size when the critical value is obtained from the asymptotic distribution, even for sample sizes as large as 500 spatial units. Consider, for instance, a circular world design with 500 spatial units, each with two neighbors. Using tests of nominal size $\alpha = 0.05$ and running 999 simulations, we obtain 3.5%, 4.8%, and 14.4%, for the Type I error probability for likelihood ratio, score and Wald tests, respectively. With a denser spatial weight matrix (each spatial unit with 25 neighbors) we obtain 4.3%, 5.4%, and 16.1%, respectively.

These size distortions, therefore, point towards the need for an in-depth look at the test properties for small sample sizes, which is the subject of the next sections.

### 2.4 Asymptotic Expansions

#### 2.4.1 Edgeworth Series

In Phillips (1980), the author exemplifies the use of a truncated Edgeworth series to approximate an exact distribution (or density) of a statistic of interest. The Edgeworth series has its origins related to the Gram-Charlier A-series, which, for an statistic $T_n$ with density function $f_{T_n}(t)$, can be written as:

$$f_{T_n}(t) = a_0 \phi(t) + \frac{a_1}{1!} \phi^{(1)}(t) + \frac{a_2}{2!} \phi^{(2)}(t) + \ldots$$

(2.4.1)

where $a_r, r = 0, 1, \ldots$ are constants, $\phi(t) = (2\pi)^{-\frac{1}{2}} e^{-\frac{t^2}{2}}$ is the standard Gaussian density function, and

$$\phi^{(r)}(t) = \frac{d^r \phi(t)}{dt^r} = (-1)^r H_r(t) \phi(t)$$

where $H_r(t)$ is a polynomial in $t$ of degree $r$ and the set $\{H_r(t) \mid r = 1, 2, \ldots\}$ is an orthogonal set of polynomials (Hermite polynomials) with regard to the Gaussian distribution\(^{11}\).

In particular:

$$E(H_r(T)H_s(T)) = \begin{cases} r! & \text{if } s = r \\ 0 & \text{if } s \neq r \end{cases}$$

Integrating the Hermite polynomials:

$$E(H_r(T)) = \int_{-\infty}^{+\infty} H_r(t) f_{T_n}(t) dt = a_r (-1)^r$$

\(^9\) See table 2.2 for a summary of the spatial weight designs used in this article.

\(^{10}\) Larger than the typical empirical setting for spatial econometrics.

\(^{11}\) The polynomials are orthogonal considering the inner product $<x, y>$ as the expected value $E(xy)$ with the Gaussian measure.
one obtains the coefficients of the Gram-Charlier series as a linear combination of the moments of the statistic $T_n$.

\[ a_r = (-1)^r E(H_r(T)) = (-1)^r (b_0 + b_1 E[T] + \ldots + b_r E[T^r]) \]

Table 2.1 shows the first nine coefficients of the Gram-Charlier series in terms of the cumulants $(k)$ and the invariants $(\kappa)$ of the statistic $T_n$.

In general, the conditions for the convergence of the Gram-Charlier series are restrictive; nevertheless, even though convergence can not be assured in some cases, the truncated Gram-Charlier series can sometimes be a good approximation for the density function of the statistic of interest (see Phillips, 1980).

The Edgeworth series is a re-ordering of the Gram-Charlier series coefficients $(a_r)$, but with the important property that after truncating the series, the remainder term has the same order of magnitude of the first eliminated term. In other words, it turns the approximation into an asymptotic series (see Kolassa, 1997).

If we assume\(^{12}\) that the cumulants (and invariants) of the statistic $T_n$ have the same order than the standardized sum of independent and identically distributed random variables, i.e. $\kappa_s = O(n^{1-s/2})$, then only the first and second cumulants do not vanish as $n \to \infty$, and therefore it would make sense to use the Gaussian density function as the baseline in equation 2.4.1.

Using the coefficients from table 2.1 and assuming the invariants of the statistic $T_n$ have order $\kappa_s = O(n^{1-s/2})$, the Edgeworth series for the density function of the standardized statistic $T'_n = \frac{T_n}{\sqrt{\kappa_2}}$ can be expressed as:

\[
f_{T'_n}(t) = \phi(t) \left[ (1) + \left( \frac{\kappa_3 H_3(t)}{3!} \right) + \left( \frac{\kappa_4 H_4(t)}{4!} + \frac{10\kappa_3^2 H_6(t)}{6!} \right) \right. \]
\[
+ \left( \frac{\kappa_5 H_5(t)}{5!} \right) + \frac{35\kappa_3 \kappa_4 H_7(t)}{7!} + \frac{280\kappa_3^2 \kappa_6 H_6(t)}{9!} \right] \]
\[
+ \left( \kappa_6 H_6(t) \right) + \frac{(56\kappa_3 \kappa_5 + 35\kappa_4^3) H_8(t)}{8!} + \frac{2100\kappa_3^2 \kappa_4 H_{10}(t)}{10!} \right] + \ldots \quad (2.4.2) \]

where the terms within parenthesis are of order $O(n^{1-s/2})$ for $s = 2, 3, \ldots$

This series can be integrated term by term to produce an analogous expression for the distribution of $T'_n$:

\[
F_{T'_n}(t) = \Phi(t) - \phi(t) \left[ \left( \frac{\kappa_3 H_3(t)}{3!} \right) + \left( \frac{\kappa_4 H_4(t)}{4!} + \frac{10\kappa_3^2 H_6(t)}{6!} \right) \right. \]
\[
+ \left( \frac{\kappa_5 H_5(t)}{5!} \right) + \frac{35\kappa_3 \kappa_4 H_7(t)}{7!} + \frac{280\kappa_3^2 \kappa_6 H_6(t)}{9!} \right] \]
\[
+ \left( \kappa_6 H_6(t) \right) + \frac{(56\kappa_3 \kappa_5 + 35\kappa_4^3) H_8(t)}{8!} + \frac{2100\kappa_3^2 \kappa_4 H_{10}(t)}{10!} \right] + \ldots \quad (2.4.3) \]

The Edgeworth approximation for the distribution function ($\mathcal{E}$) is obtained by truncating the above series after a finite number of terms, and the approximation error is of the order of the

\(^{12}\) Note that in the following sections the order of the cumulants and the invariants of each statistic of interest will be derived from the underlying assumption that both matrices $W(\rho)$ and $(I - w(\rho))^{-1}$ are uniformly bounded (see section 2.2.1).
<table>
<thead>
<tr>
<th>Coefficients</th>
<th>General cumulants</th>
<th>Invariants</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a_0$</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$a_1$</td>
<td>$-k_1$</td>
<td>0</td>
</tr>
<tr>
<td>$a_2$</td>
<td>$k_2 + k_1^2 - 1$</td>
<td>0</td>
</tr>
<tr>
<td>$a_3$</td>
<td>$3k_1 - k_3 - 3k_1k_2 - k_1^3$</td>
<td>$-\kappa_3$</td>
</tr>
<tr>
<td>$a_4$</td>
<td>$k_4 + 4k_1k_3 + 3k_2^2 + 6k_1^2k_2 - 6k_2 + k_1^4 - 6k_1^2 + 3$</td>
<td>$\kappa_4$</td>
</tr>
<tr>
<td>$a_5$</td>
<td>$-k_5 - 5k_1k_4 - 10k_2k_3 - 10k_3^2k_3 + 10k_3 - 15k_1k_2^2 - 10k_1^2k_2 + 30k_1k_2 - k_1^2 + 10k_2^2 - 15k_1$</td>
<td>$-\kappa_5$</td>
</tr>
<tr>
<td>$a_6$</td>
<td>$k_6 + 6k_1k_5 + 15k_2k_4 + 15k_3^2k_4 - 15k_4 - 10k_2^2 + 60k_1k_2k_3 + 20k_2^3k_3 - 60k_1k_3 + 15k_2^2 + 45k_1^2k_2^2 - 45k_2^2 + 15k_1k_2 - \kappa_6 + 10\kappa_3^2$</td>
<td>$90k_2^2k_2 + 45k_2 + k_1^2 - 15k_1^4 + 15k_1^2 - 15$</td>
</tr>
<tr>
<td>$a_7$</td>
<td>$-k_7 - 7k_1k_6 - 21k_2k_5 - 21k_2^2k_5 + 21k_5 - 35k_3k_4 - 105k_1k_2k_4 - 35k_1^2k_4 + 105k_1k_2k_3 - 105k_2^2k_3 - 210k_2^3k_3 + 210k_1k_3 + 35k_3^2k_3 + 35k_1k_3 - 105k_3 - \kappa_7 - 35\kappa_3\kappa_4$</td>
<td>$-315k_1k_2 - k_1^2 + 21k_1^5 - 105k_1^3 + 105k_1$</td>
</tr>
<tr>
<td>$a_8$</td>
<td>$k_8 + 8k_1k_7 + 28k_2k_6 + 28k_2^2k_6 - 28k_6 + 56k_3k_5 + 168k_1k_2k_5 + 56k_1^2k_5 + 168k_1k_5 + 35k_2^2 + 280k_1k_3k_4 + 210k_2^3 + 420k_2^2k_4 + 420k_2k_4 + 70k_3^2k_4 + 420k_2^2k_4 + 210k_4 + 280k_2k_2^2 + 280k_2^2k_3^2 + 280k_3^2 + 840k_1k_2^3k_3 + 560k_1k_2k_3 + 56k_1^2k_2k_3 + 560k_1k_3^3 + 560k_1^2k_3 + 840k_1k_3 + 105k_2^3 + 420k_2k_2^2 + 420k_2k_2^2 + 120k_1k_2^2k_2 + 120k_1k_2^2k_2 + 630k_2^3 + 28k_6^2k_2 - 120k_1k_2k_2 + 1260k_1k_2k_2 - 420k_2k_2 + 28k_2^2 + 210k_4^2 + 105k_1^2 + 105$</td>
<td>$k_8 + 56\kappa_3\kappa_5 + 35\kappa_3^2$</td>
</tr>
<tr>
<td>$a_9$</td>
<td>$-k_9 - 9k_1k_8 - 36k_2k_7 - 36k_2^2k_7 + 36k_7 - 84k_3k_6 - 252k_1k_2k_6 - 84k_2^2k_6 + 252k_1k_6 - 126k_4k_5 - 504k_1k_3 - 378k_2^3k_5 + 756k_2^2k_5 + 756k_2k_5 - 126k_4^2k_5 + 756k_2^3k_5 - 378k_5 - 315k_1k_2^2 - 1260k_2k_5k_4 - 1260k_1k_2k_4 + 1260k_3k_4 - 1890k_1k_2^2k_4 - 1260k_1k_2k_4 + 3780k_1k_2k_4 - 126k_4k_5 + 1260k_4k_5 - 1260k_4^2k_5 - 1890k_1k_2^2k_4 - 1980k_1k_2k_4 - 280k_3^3 + 280k_2^2k_3^2 - 2520k_1^2k_2k_3 - 840k_2^3k_2^2 + 2520k_1^2k_2k_3 - 1260k_1^2k_2k_3 + 3780k_1^2k_2k_3 - 1260k_1^2k_2k_3 + 3780k_1^2k_2k_3 - 3780k_1^2k_2k_3 - 7560k_2^2k_3 + 3780k_2k_3 + 3780k_2k_3 - 84k_1^6k_3 + 1260k_1^4k_3 + 3780k_1^3k_3 + 1260k_1k_3 - 945k_1^4k_3 - 1260k_1^3k_3 + 3780k_1^3k_2^2 - 3780k_1^3k_2^2 - 5670k_1k_2^2 - 36k_1^2k_2 + 756k_1^3k_2 - 3780k_1^2k_2 - 3780k_1k_2 - k_1^4 + 36k_1^2 - 378k_1 + 1260k_1^3 - 945k_1$</td>
<td>$-k_9 - 84\kappa_3\kappa_6 - 126\kappa_4\kappa_5 - 280\kappa_3^3$</td>
</tr>
</tbody>
</table>

Table 2.1: Coefficients of Gram-Charlier series
first neglected term. For example, using the first three terms (see equation 2.4.3) we have:

\[
\mathcal{E}_5(t, \kappa) = \Phi(t) - \phi(t) \left[ \frac{\kappa_3 H_2(t)}{3!} + \frac{\kappa_4 H_2(t)}{4!} + \frac{10\kappa_5 H_2(t)}{6!} \right. \\
+ \left. \frac{\kappa_5 H_4(t)}{5!} + \frac{35\kappa_5 \kappa_4 H_6(t)}{7!} + \frac{280\kappa_5^3 H_8(t)}{9!} \right] \tag{2.4.4}
\]

where the subscript \((s = 5)\) in \(\mathcal{E}_s\) represents the use of the \(s^{th}\) invariant in the Edgeworth approximation. As a result we can write:

\[
F_{T_2}(t) = \mathcal{E}_s(t, \kappa) + O \left( \frac{1}{n^{(1-s)/2}} \right) \tag{2.4.5}
\]

The approximation error goes to zero faster than the error obtained using a Gaussian approximation (i.e., using only the first term in the Edgeworth series) as \(n \to \infty\); however, we can make assertions only about the asymptotic order of magnitude of the approximation error, and usually very little can be said about the error absolute value. Theorems trying to establish numeric bounds for the error of Gaussian approximation are known in the literature as Berry-Esséen Theorems, but this line of research has diminished as usually the approximations perform better than the bounds in the theorems would predict. One must be aware that, even when absolute bounds can be obtained (either analytically or numerically), the relative error of an Edgeworth approximation can be very large, mainly near the tails of the distribution.

More formally, the conditions for the validity of an Edgeworth approximation, to the order expected, in the case of a standardized sum of independent non-identically distributed random variables is given in Kolassa (1997, Theorem 3.11.2), reproduced below:

2.4.1 Theorem (Edgeworth approximation for the independent non-id case). Suppose \(X_i\) are independent random variables, with characteristic functions \(\xi_{X_i}(\beta)\), zero means and variance \(\sigma_i^2\), and \(s \geq 2\) is an integer, such that for all \(i\), \(X_i\) has a cumulant of order \(s\). Define

\[
\zeta_n^2 = n^{-1} \sum_{i=1}^{n} \sigma_i^2
\]

\[
\omega_{n,s} = n^{-1} \sum_{i=1}^{n} \frac{E(|X_i|^s)}{\zeta_n^s} \tag{2.4.6}
\]

and assume that \(\omega_{n,s}\) remains bounded as \(n\) increases, and that \(\zeta_n^2\) is bounded away from zero. Also assume that

\[
\prod_{i=1}^{n} \xi_{X_i}(\beta) = o(n^{-a}) \text{ uniformly for } |\beta| > \delta, \quad \forall \delta, a > 0 \tag{2.4.8}
\]

Then the distribution function of \(Z = \sum_{i=1}^{n} X_i / \sqrt{n \zeta_n^2}\) using the first \(s\) cumulants and dropping terms of order \(O(n^{1-s/2})\) can be approximated by \(F_Z(z) = \mathcal{E}_{s-1}(z, \kappa) + O(n^{1-s/2})\), where \(\mathcal{E}_s(\cdot, \cdot)\) is the Edgeworth approximation for the distribution function of standardized sums of iid random variables (see equation 2.4.4), and \(\kappa\) are the invariants of \(Z\).
2.4.2 Saddlepoint Series

The use of a truncated Edgeworth series is well-known to not provide a good approximation at the tails of the distribution. This point has been emphasized at least since Daniels (1954), where the author mentions that errors in the tail regions of the density function can be sometimes comparable with the frequency themselves (i.e., high relative errors), and that the Edgeworth approximation can assume negative values in these regions. Moreover, in the current context, section 2.7.3 highlights that for the spatial dependence parameter $\rho$ near the upper bound of the set $\hat{P}$ the Edgeworth approximation to the relevant test statistic ($T_n$ or $V_n$, defined in sections 2.5 and 2.6) is not well-behaved even for values not that far in the tails. As a result, the approximation for the distribution of the test statistics under the alternative hypothesis might be poorly approximated and therefore the power of the optimum test calculated using the Edgeworth approximation is likely to be materially different from its true (unknown) value.

One possibility to tackle this problem is to look for alternative ways to approximate the distribution of the statistic of interest when the spatial dependence parameter $\rho$ is large. This section focuses on the use of the saddlepoint series to approximate the distribution of the test statistics. Goutis and Casella (1999) provide a good elementary motivation and explanation of this approximation technique.

Consider a statistic $T_n$, whose density (distribution) $f_{T_n}(t)$ ($F_{T_n}(t)$) is to be approximated, and we know its cumulant generating function $K_{T_n}(\gamma)$. Define

$$g_{T_n}(t, \gamma) = e^{\gamma - K_{T_n}(\gamma)} f_{T_n}(t)$$

(2.4.9)

The moment generating function of the random variable whose density is given by equation 2.4.9 is given by:

$$M_{T_n}(\beta) = E(e^{\beta T_n}) = \int e^{\beta t} e^{t \gamma - K_{T_n}(\gamma)} f_{T_n}(t) dt_n$$

(2.4.10)

$$= e^{-K_{T_n}(\gamma)} M_{T_n}(\beta + \gamma)$$

and therefore the cumulant generating function is:

$$K_{T_n}(\beta) = K_{T_n}(\beta + \gamma) - K_{T_n}(\gamma)$$

(2.4.11)

The cumulants for the random variable within the embedded exponential family are then given by:

$$k_{s, \gamma} = \left. \frac{d^s K_{T_n}(\beta)}{d\beta^s} \right|_{\beta=0} = K_{T_n}^{(s)}(\gamma)$$

(2.4.12)

Knowing the cumulants (function of $\gamma$) of the statistic embedded in the exponential family allow us to approximate its density function by a truncated Edgeworth series (see section 2.4.1) for each chosen value for $\gamma$. As we are interested in the value of $f_{T_n}(t)$, the best alternative is to approximate $g_{T_n}(t, \gamma)$ at a point where the expected value of $T_n, \gamma$ is $t$, i.e., one chooses a value for $\gamma$ (called the saddlepoint $\hat{\gamma}$) such that $K_{T_n}'(\hat{\gamma}) = t$. The rational behind this choice is explained formally in Kolassa (1997); Goutis and Casella (1999).

In analogy with equation 2.4.4 the Edgeworth approximation for the density function ($e$) of a standardized random variable is obtained by truncating the Edgeworth series (equation 2.4.2) after a finite number of terms. For example:

$$e_{5}(t, \kappa) = \phi(t) \left[ 1 + \frac{\kappa_3 H_3(t)}{3!} + \frac{\kappa_4 H_4(t)}{4!} + \frac{10\kappa_5^2 H_6(t)}{6!} \right. \left. + \frac{\kappa_5 H_5(t)}{5!} + \frac{35\kappa_3\kappa_4 H_7(t)}{7!} + \frac{280\kappa_5^3 H_9(t)}{9!} \right]$$

(2.4.13)
where the subscript \((s = 5)\) in \(\varepsilon_s\) represents the use of the \(s^{th}\) invariant in the approximation. Thus any density function of a standardized random variable can be written as:

\[
f(t) = \varepsilon_s(t, \kappa) + \mathcal{O}(n^{1-s}/2) \tag{2.4.14}
\]

Therefore, an Edgeworth approximation for \(g_{T_n}(t, \gamma)\) would be given by:

\[
g_{T_n}(t, \gamma) \approx \frac{1}{\sqrt{K''_{T_n}(\gamma)}} \varepsilon_s \left( \frac{t - K''_{T_n}(\gamma)}{\sqrt{K''_{T_n}(\gamma)}}, \kappa_n \right) \tag{2.4.15}
\]

where the invariants \(\kappa_n\) are given by \(\kappa_{s, \gamma} = k_{s, \gamma}k_{2, \gamma}^{-s/2}\) for \(s = 2, 3, \ldots\).

The approximated density function for the statistic \(T_n\), evaluated at the saddlepoint \((\hat{\gamma})\) is then:

\[
f_{T_n}(t) \approx \frac{e^{K_{T_n}(\gamma; \hat{\gamma})}}{\sqrt{2\pi K''_{T_n}(\hat{\gamma})}} \varepsilon_s(0, \kappa_n) \tag{2.4.16}
\]

Interesting to note (Daniels, 1980) that the renormalization of the saddlepoint approximation to the density function usually has a very low relative error in the whole range of density. The author mentions that for some specific densities (normal, gamma, and inverse normal), and only for those, the renormalization matches exactly with the true density.

**Gaussian Approximation**

The Gaussian approximation \((\varepsilon_2)\) for the density of \(T_n\) (see equation 2.4.16) is

\[
f_{T_n}(t) \approx \frac{e^{K_{T_n}(\gamma; \hat{\gamma})}}{\sqrt{2\pi K''_{T_n}(\hat{\gamma})}} \tag{2.4.17}
\]

Note, however, that since the Hermite polynomials of odd order are null when evaluated at zero, this approximation is actually of order \(O\left(n^{-1}\right)\) even using only the first two cumulants. The underlying reason is that for each \(t\) the value of \(\hat{\gamma}(t)\) adjusts, improving the order of the approximation. Using up to the fourth cumulant, the order of the error term would then be \(O\left(n^{-2}\right)\). The drawback is that if one needs to calculate the approximated value of the density function for several values of \(t\) (for instance in the case of calculating the approximate distribution function), the saddlepoint equation has to be solved several times numerically, as an analytical expression for \(\hat{\gamma}(t)\) is not available. A change of variables, however, usually reduces this burden (Kolassa, 1997).

The distribution function of the statistic of interest \((T_n)\) can thus be calculated using the approximate density function:

\[
1 - F_{T_n}(t) = P(T_n > t) = \int_t^\infty f_{T_n}(u)du = \int_t^\infty \frac{e^{K_{T_n}(\gamma(u); \hat{\gamma}(u))}}{\sqrt{2\pi K''_{T_n}(\hat{\gamma}(u))}}du + \mathcal{O}(n^{-1}) \tag{2.4.18}
\]

but \(K''_{T_n}(\hat{\gamma}) = u\), therefore \(du = K''_{T_n}(\hat{\gamma})d\hat{\gamma}\), and

\[
P(T_n > t) = \int_{\hat{\gamma}(t)}^{\hat{\gamma}(t) + \infty} \frac{K''_{T_n}(\hat{\gamma})}{2\pi} e^{K_{T_n}(\gamma; \hat{\gamma})}d\hat{\gamma} + \mathcal{O}(n^{-1}) \tag{2.4.19}
\]

Alternatively, to avoid numerical integration of the above formula, one can use an analytical approximation to this integral. Bahadur and Rao (1960), Robinson (1982), Lugannani and Rice
Barndorff-Nielsen (1990a), are some examples of such approximation. Daniels (1987) provided several numerical examples showing a high accuracy of the Lugannani-Rice formula. Below we illustrate some of these approximations.

2.4.2 Lemma (Robinson approximation). Let \( T_n = \sum_{i=1}^n X_i \) be the sum of independent and non-identically distributed random variables, satisfying the assumptions of theorem 2.4.1. Then the tail probability of \( T_n \) can be approximated by:

\[
P(T_n > t) = e^{\frac{\gamma^2 \sigma^2}{2}} \left( 1 - \Phi(b) \right) \left( 1 - \frac{\kappa_3 T_n b^3}{6} \right) + \frac{\phi(b)(b^2 - 1)\kappa_3 T_n}{6} + \phi(a)O(n^{-1}) \tag{2.4.20}
\]

where the saddlepoint \( \gamma \) solves \( K'_{T_n}(\gamma) = t \), \( a = \text{sign}(\gamma) \sqrt{2 \left( \gamma K''_{T_n}(\gamma) - K_{T_n}(\gamma) \right)} \), \( b = \gamma \sqrt{K''_{T_n}(\gamma)} \), and \( \kappa_3 T_n \equiv \frac{K''_{T_n}(0)}{K''_{T_n}(0)^{3/2}} \).

Proof. Robinson’s approximation formula shown in Kolassa (2007) holds for the mean of independent and non-identically distributed random variables. Therefore we define \( T_n^* = \frac{T_n}{n} \) and apply Robinson’s approximation to \( T_n^* \) evaluated at the \( \frac{t}{n} \), and noting that the cumulant function of \( T_n^* \) satisfy

\[
K_{T_n^*}(\gamma) = K_{T_n}(\frac{\gamma}{n})
\]

\[
K_{T_n^*}(\gamma) = K_{T_n}(\frac{\gamma}{n}) \frac{1}{n^s}
\]

Robinson’s approximation, then simplifies to the equation in the lemma’s statement.

2.4.3 Lemma (Lugannani-Rice approximation). Let \( T_n \) be the sum of non-identically distributed random variables. Then the following approximation to its distribution function is valid:

\[
F_{T_n}(t) = \begin{cases} 
\Phi(a) - \phi(a) \left( \frac{b}{a} - \frac{1}{a} \right) + \phi(a)O\left(n^{-3/2}\right) & \text{for } \gamma \neq 0 \\
\Phi(a) - \frac{b}{a} \phi(a) \kappa_3 T_n \sqrt{n} + \phi(a)O\left(n^{-3/2}\right) & \text{for } \gamma = 0
\end{cases}
\]

Proof. Analogous to lemma 2.4.2.

Another approximation, which is asymptotically equivalent, is given by Barndorff-Nielsen (1990a); Jensen (1992):

2.4.4 Lemma. The distribution of the statistic \( T_n \) is given by:

\[
F_{T_n}(t) = \Phi(a^*) + \phi(a)O(n^{-1})
\]

where \( a^* = a + \frac{1}{a} \ln \left( \frac{b}{a} \right) \).

Proof. analogous to 2.4.2.

Kolassa (2007) proves that under some regularity conditions, both Robinson and Lugannani and Rice approximations agree to \( O(n^{-1}) \), even for the case non-identically distributed random variables. Moreover, the Lugannani-Rice approximation is appropriate both for values \( t > E(T_n) \) and \( t < E(T_n) \), but Robinson’s approximation is not valid for ordinates below the mean.

One may conjecture whether, based on these saddlepoint approximations, it is possible to obtain an approximation for the quantile function similar to the Cornish-Fisher (see equation 2.5.12). Arevalillo (2003) provides such inversion of the saddlepoint approximation, which is used in the following lemma:
2.4.5 Lemma (Saddlepoint inversion). Let $T_n$ be the sum of non-identically distributed random variables satisfying assumptions in theorem 2.4.1. Let $\mathcal{K}_T(\beta)$, $k_1$, and $k_2$ be, respectively, the cumulant generating function, mean and variance of $T_n$. For a given $\alpha \in (0,1)$ define $z_\alpha = \Phi^{-1}(1 - \alpha)$, and $\bar{z}_\alpha = z_\alpha / \sqrt{k_2}$. Let

$$
\bar{t} = \frac{z_\alpha [k_2 + \mathcal{K}_T''(z_\alpha)] - \mathcal{K}_T'(z_\alpha) + k_1}{\mathcal{K}_T'(z_\alpha)}
$$

$$
\bar{s} = \mathcal{K}_T'(\bar{t}) + \frac{a(\bar{t}) [z_\alpha - r(\bar{t})]}{\bar{t}}
$$

where

$$
a(t) \equiv \text{sign}(t) \sqrt{2 \left[t \mathcal{K}_T''(t) - \mathcal{K}_T(t)\right]}
$$

$$
b(t) \equiv t \sqrt{\mathcal{K}_T''(t)}
$$

$$
r(t) \equiv a(t) + \frac{1}{a(t)} \ln \frac{b(t)}{a(t)}
$$

Then, the quantile function is given by

$$
F^{-1}_T(1 - \alpha) = \bar{s} + O(n^{-1}) \quad (2.4.21)
$$

Proof. Define $T^* = \frac{T_n - k_1}{\sqrt{k_2}}$, which is asymptotically standard Gaussian. Then apply Arevalillo (2003, Theorem 1) and the result of the lemma follows. Note that Arevalillo’s main theorem applies to random sample of identically distributed variables, but the theorem’s proof is also valid when the invariants $\kappa_s$ of the statistic of interest are of order $O(n^{1-s/2})$. \hfill \square

Remark. Note that the saddlepoint inversion may present some problems for $\alpha$ close to the bounds of the interval $(0,1)$. The reason is that $\bar{t} \to \pm \infty$ for $\alpha$ close to the upper or lower bounds but the cumulant generating function $\mathcal{K}_T(t)$ is defined only for a bounded domain (see lemma 2.2.8). This can be illustrated in figure 2.1, which compares, for a simple circular world design, the critical values obtained by the saddlepoint inversion formula, by numerically solving equation 2.4.18, and the critical values for the standard Gaussian distribution.

In this case, the saddlepoint inversion approximation is inaccurate for $\alpha < 0.04$ for small sample sizes ($n = 25$). The saddlepoint approximation, however, is remarkably faster to calculate than the numerical solution to equation 2.4.18.
2.5 Testing for Spatial Dependence (Known Variance)

Assuming the $n \times 1$ disturbance vector $U$ is homoskedastic with known variance ($\sigma^2$) reduces the parameter space to $P \subset \tilde{P} \cap \tilde{P}$ (see proposition 2.2.5). Therefore we can write the parametric model $M \equiv (\mathcal{Y}, \{F_\rho, \rho \in P\})$ where $\mathcal{Y} \subset \mathbb{R}^n$ is the sample space, and $F_\rho$ is a family of probability distributions, whose density is given by:

$$f(y, \rho) = (2\pi\sigma^2)^{-n/2} \det L(\rho) \exp \left( -\frac{y' L(\rho)' L(\rho) y}{2\sigma^2} \right)$$  \hspace{1cm} (2.5.1)

This is a curved exponential family. Depending on the structure of the spatial weight matrix $w(\rho)$, the dimension of the embedding exponential family can be reduced, but even in the simplest case of a linear spatial weight matrix (i.e., $w(\rho) = \rho M$, where $M$ is a known constant matrix), the sufficient statistic and natural parameter belong to $\mathbb{R}^2$. In this case $\eta = \sigma^{-2}(\rho/2, \rho^2/2) \in \mathbb{R}^2$, and $t(y) = (y'(M + M')y, -y'MMy) \in \mathbb{R}^2$, and the canonical form for this density would be:

$$f(y, \eta) = h(y)e^{\eta t(y) - k(\eta)}$$

where $\eta$ is the natural parameter, $t(y)$ is a sufficient statistic, $h(y) \equiv (2\pi\sigma^2)^{-n/2}e^{-y'y/2\sigma^2}$, and $k(\eta(\rho)) \equiv -\ln \det L(\rho)$. This is a curved (2,1) exponential family model, therefore, despite having only one unknown parameter we cannot look for a monotone likelihood property to justify an uniformly most powerful test.

We will then look for an optimum test (most powerful test) against each specific (simple) alternative hypothesis, and use this test to calculate the power envelope for testing the null of no spatial dependence in a SAR stochastic process. An optimum $\alpha$-size test can be found by solving the following problem:

$$\max_{\psi} \beta_\psi(\rho_1) = E_{\rho_1} \psi(Y)$$  \hspace{1cm} (2.5.2)

subject to $E_\rho \psi(Y) \leq \alpha$, $\forall \rho \in P_0$  \hspace{1cm} (2.5.3)

where $\beta_\psi(\rho) = E_\rho \psi(Y)$ evaluated for $\rho \in P_1$ is the power function. Assuming the null hypothesis is also simple, the optimum test of size $\alpha$ for testing $H_0 : \rho = \rho_0$ vs. $H_1 : \rho = \rho_1$ is given by the
Neyman-Pearson lemma (Lehmann, 1986):

$$\psi(y) = \begin{cases} 1 & f(y, \rho_1) > c_\alpha f(y, \rho_0) \\ 0 & f(y, \rho_1) < c_\alpha f(y, \rho_0) \end{cases}$$

where $c_\alpha$ is determined by $E_0 \psi(Y) = \alpha$.

The following theorem characterizes the test statistic that will be used in the subsequent analysis.

2.5.1 Theorem (Optimum test). Define the statistic $T_n(Y, \rho_1) = \frac{Y' A(\rho_1) Y}{\sigma^2 \rho_1}$. The optimum $\alpha$-size test for testing $H_0 : \rho = 0$ vs. $H_1 : \rho = \rho_1$ is given by

$$\psi(y) = \begin{cases} 1 & T_n(y, \rho_1) > c_\alpha \\ 0 & T_n(y, \rho_1) < c_\alpha \end{cases} \quad (2.5.4)$$

for all $\rho_1 > 0$ where $A(\rho) \equiv w(\rho) + w(\rho)' - w(\rho)'w(\rho)$ is a symmetric matrix, and $c_\alpha$ is determined by $E_0 \psi(T_n) = \alpha$. For the case $\rho_1 < 0$, all the inequalities in equation 2.5.4 are reversed. Moreover, when $\rho_1 \to 0$, $\rho_1 > 0$ this test is also locally uniformly most powerful (LUMP) for testing $H_0 : \rho \leq 0$ vs. $H_1 : \rho > 0$.

Proof. Follows from the Neyman-Pearson lemma. The critical region of the test is given by $\frac{f(y, \rho_1)}{f(y, 0)} > c_\alpha$. Expanding:

$$\frac{f(y, \rho_1)}{f(y, 0)} > c_\alpha$$

$$\frac{-y' L(\rho_1)' L(\rho_1) y + y'y}{2\sigma^2} > c_\alpha$$

$$\frac{y' A(\rho_1) y}{\sigma^2} > c_\alpha$$

$$\frac{y' A(\rho_1) y}{\sigma^2 \rho_1} > c_\alpha \text{ for } \rho_1 > 0$$

$$\frac{y' A(\rho_1) y}{\sigma^2 \rho_1} < c_\alpha \text{ for } \rho_1 < 0$$

Note that in the limiting case $\rho_1 \to 0$, $\frac{A(\rho_1)}{\rho_1} \to W(0)' + W(0)$, where $W(\rho) = \frac{\partial w(\rho)}{\partial \rho}$ is the first derivative of the spatial weight matrix. The test statistic becomes $T_n(Y) = \frac{2Y'W(0)Y}{\sigma^2}$, therefore, in this case, the critical region of the test is equivalent to the one based on the score statistic $s(Y) = \frac{\partial \ln f(Y, \rho)}{\partial \rho} \bigg|_{\rho = 0}$, so this test is also locally uniformly most powerful (LUMP) for testing $H_0 : \rho \leq 0$ vs. $H_1 : \rho > 0$ (see Gourieroux and Monfort, 1995). The preceding inequalities will depend on whether $\rho_1 \to 0$ is a right or left limit.

Remark. In the typical case where the spatial weight matrix is linear, $w(\rho) = \rho W$, $\rho_1 \to 0$, the test statistic becomes $T_n(Y) = \frac{2Y'WY}{\sigma^2}$, which is similar to the popular Moran’s $I$ test statistic $I = \frac{1}{\sum_{i,j}W_{ij}Y_i Y_j/n}$ (see Moran, 1950; Cliff and Ord, 1981b). Burridge (1980) had already noted that Moran’s $I$ test is equivalent to a Lagrange multiplier test when the alternative to $\rho_0 = 0$ is a SAR or a SMA stochastic process.

In the general case where the random variable $Y$ is generated by a SAR model $m \in M$ with associated parameter $\rho$, it follows that the statistic $T_n$ has the same distribution as a quadratic form in standard Gaussian variables:

$$T_n(Y, \rho_1; \rho) \sim \frac{U'}{\sigma} C(\rho_1, \rho) \frac{U}{\sigma} \quad (2.5.5)$$
where \( C(\rho_1, \rho) \equiv L^{-1}(\rho)' \frac{\partial^2 C}{\partial \rho^2} L^{-1}(\rho) \) is a symmetric matrix, and \( \frac{U}{\sigma} \sim N(0, I_n) \). The following theorems characterize the asymptotic distribution of the statistic \( T_n \).

### 2.5.2 Proposition

The cumulants for the statistic \( T_n \) are given by

\[
k_{s,n}(\rho_1, \rho) = 2^{s-1} (s-1)! \text{tr} C(\rho_1, \rho)^s
\]

\[\text{for all } s = 1, 2, \ldots \] (2.5.6)

*Proof.* Immediate from lemma 2.2.8. \(\square\)

### 2.5.3 Proposition

(Convergence rate for \( C(\rho_1, \rho) \)). Let \( w(\rho) \) be a spatial weight matrix satisfying assumption 2.2.4. Assume further that the row and column sums of the matrix \( L^{-1}(\rho) \) are uniformly bounded in absolute value for all \( \rho \in \mathbb{R}^3 \). Then it follows that \( \text{tr} C(\rho_1, \rho)^s = O(n) \)

*Proof.* From assumption 2.2.4 the row and column sums of the matrix \( W(\rho) \) are uniformly bounded in absolute value. Corollary following proposition 2.2.5 shows that under this assumption the row and column sums of \( w(\rho) \) are uniformly bounded in absolute value. From lemma 2.2.6 we conclude \( w(\rho)'w(\rho) \) is uniformly bounded, and thus \( \frac{\partial^2 C}{\partial \rho^2} \) is uniformly bounded. By hypothesis the row and column sums of \( L^{-1}(\rho) \) are uniformly bounded in absolute value, then it follows again from lemma 2.2.6 that the matrix product \( C(\rho_1, \rho) = L^{-1}(\rho)' \frac{\partial^2 C}{\partial \rho^2} L^{-1}(\rho) \) is uniformly bounded. The conclusion then follows from lemma 2.2.7. \(\square\)

### 2.5.4 Theorem

(Optimum test statistic is asymptotically Gaussian). The statistic \( T_n \) is asymptotically Gaussian, \( \forall \rho, \rho_1 \in \mathbb{R} \).

*Proof.* Billingsley (1986) shows in theorem 30.2 that if the distribution of a random variable \( X \) is determined by its moments (which is the case for the Gaussian distribution) and \( X_n \) has moments of all orders converging to the moments of \( X \), then \( X_n \) converges in distribution to \( X \).

The cumulants of the statistic \( T_n \) are given in proposition 2.5.2, and proposition 2.5.3 establishes \( \text{tr} C(\rho_1, \rho)^s = O(n), \forall s \). Consider \( T_n^* \equiv \frac{T_n - k_{2,n}}{\sqrt{k_{2,n}}} \). The proof of proposition 2.5.5 establishes that the variance of \( T_n \) grows exactly at a rate proportional to the sample size \( n \), therefore the cumulants of \( T_n^* \) are of order \( k_s^* = \frac{k_{s,n}^*}{k_{2,n}} = O(n^{1-s/2}) \), \( s \geq 2 \) and all the cumulants of order greater than two vanish as \( n \to \infty \). Therefore \( T_n^* \overset{d}{\to} N(0, 1) \), and \( T_n \overset{d}{\to} N(k_{1,n}, k_{2,n}) \), where \( k_{1,n}, k_{2,n} \) depend on \( (\rho_1, \rho) \) and the structure of the spatial weight matrix \( w(\rho) \). \(\square\)

In order to calculate the critical level \( c_\alpha \) one needs the distribution of the \( T_n \) statistic under the null hypothesis. Even though we can calculate all the cumulants of the statistic for a given sample size, its distribution is only known asymptotically, and in most empirical applications the asymptotic values are often used. However, as illustrated in section 2.3, using the critical values obtained from the limiting distribution is likely to yield an actual size for the test which is different from the nominal size, potentially leading to significant under or over rejection of the null hypothesis.

In the following sections we compare three approaches to obtain a more accurate critical value to be used in small samples: the use of Edgeworth series, saddlepoint series, and Monte Carlo simulation. Then, in section 2.7 these approaches are used to obtain the power envelope for testing the presence of spatial dependence in an SAR stochastic process in small samples.

---

Note that this assumption is also needed to show consistency and asymptotic normality of the ML estimator of the SAR model (Lee, 2004).
2.5.1 Edgeworth Series - Optimum Test

The following proposition shows that an Edgeworth expansion that approximates the distribution function of $T_n$ based on its cumulants $k_{s,n}$ is valid to the correct order.

2.5.5 Proposition. The distribution function of the statistic $T_n$, approximated by an Edgeworth truncated series by dropping terms of size $O\left(n^{1-s/2}\right)$, satisfy:

$$F_{T_n}(t) = E_{s-1} \left( \frac{t - k_{1,n}}{\sqrt{k_{2,n}}} \kappa \right) + O\left(n^{1-s/2}\right) \quad (2.5.7)$$

Proof. The statistic $T_n$ is the sum of independent but non-identically distributed random variables. Therefore, in order to show that an Edgeworth approximation to the distribution function of $T_n$ has the correct order for the error term, we will check the assumptions of theorem 2.4.1 (reproduced from Kolassa, 2007).

Let $X_i \equiv \lambda_i \left( \frac{U_i}{\sigma} \right)^2 - 1$, $i = 1, \ldots, n$, where $U \sim N(0, \sigma^2 I_n)$ and $\lambda_i$ are the eigenvalues of $C(\rho_1, \rho)$. Then $E X_i = 0$ and $\sigma_i^2 = \text{Var} X_i = 2\lambda_i^2$, $\forall i = 1, \ldots, n$. Define

$$\varsigma_n^2 = \frac{1}{n} \sum_{i=1}^{n} \sigma_i^2$$

$$= \frac{2}{n} \sum_{i=1}^{n} \lambda_i^2$$

$$= \frac{2}{n} \text{tr} C(\rho_1, \rho)^2 = O(1)$$

In order to show that $\varsigma_n^2$ is bounded away from zero, recall that from assumption 2.2.1 the number of zero eigenvalues of the spatial weight matrix is finite, therefore the number of non-zero eigenvalues $\lambda_i$ say, $K \geq 0$, is of order $K = o(n)$. Denote by $b > 0$ the lower bound of the absolute values of the non-null eigenvalues. Therefore we have, for $n > K$

$$\varsigma_n^2 = \frac{2}{n} \sum_{i=1}^{n} \lambda_i^2 \geq \frac{2}{n} \sum_{i=1}^{n-K} b^2 = \frac{2b^2 (n-K)}{n} \xrightarrow{n \to \infty} 2b^2 > 0 \quad (2.5.8)$$

Now consider the characteristic function of $X_j$:

$$\zeta_{X_j}(\beta) = E e^{i\beta X_j} = E e^{i\beta \lambda_j \left( \frac{U_j}{\sigma} \right)^2}$$

$$= e^{-i\beta \lambda_j} E e^{i\beta \lambda_j \left( \frac{U_j}{\sigma} \right)^2} = e^{-i\beta \lambda_j} \zeta_{X_j}^2 (\beta \lambda_j)$$

$$= e^{-i\beta \lambda_j} (1 - 2i \beta \lambda_j)^{-1/2}$$

where $\zeta_{X_j}^2(\beta)$ is the characteristic function of a chi-square random variable with one degree of freedom. One can show that

$$i^s E X_j^s = \frac{d^s \zeta_{X_j}(\beta)}{d\beta^s} \bigg|_{\beta=0} = i^s \mu_s \lambda_j^s$$
where \( \mu_s \) is the \( s^{th} \) central moment of a chi-square random variable with one degree of freedom, therefore

\[
E X_j^s = \mu_s \lambda_j^s < \infty
\]

as the eigenvalues of \( C(\rho_1, \rho) \) are real and bounded. For \( s \) a positive even integer, the existence of the \( s^{th} \) derivative of the characteristic function evaluated at zero implies that \( E |X_j|^s < \infty \) and this implies \( E |X_j|^s < \infty, \forall s \leq s \). As \( s \) is an arbitrary even integer, we conclude that \( X_j \) has absolute moments of all orders. Let \( B > 0 \) such that \( E |X_j|^s < B, \forall j, s \). Thus

\[
\bar{\omega}_{n,s} = \frac{1}{n} \sum_{j=1}^{n} \frac{E |X_j|^s}{\zeta_n^s} \leq \frac{1}{n} \sum_{j=1}^{n} \frac{B}{(2b^2)^{s/2}} = \frac{B}{(b\sqrt{2})^s} < \infty
\]  

(2.5.9)

showing that \( \bar{\omega}_{n,s} \) remains bounded as \( n \) increases.

Finally, we need to show that:

\[
\prod_{i=1}^{n} \zeta_i(\beta) = o(n^{-a}) \text{ uniformly for } |\beta| > \delta, \quad \forall \delta, a > 0
\]  

(2.5.10)

Consider

\[
\prod_{j=1}^{n} \zeta_j(\beta) = \prod_{j=1}^{n} e^{-i\beta \lambda_j} (1 - 2i\beta \lambda_j)^{-1/2}
\]

\[
= e^{-i \sum_{j=1}^{n} (\beta \lambda_j + \theta_j/2)} \prod_{j=1}^{n} (1 + 4\beta^2 \lambda_j^2)^{-1/4}
\]

where \( \theta_j \) is the angle of the complex number \( 1 - 2i\beta \lambda_j \) in polar coordinates. Therefore,

\[
\left| \prod_{j=1}^{n} \zeta_j(\beta) \right| = \prod_{j=1}^{n} (1 + 4\beta^2 \lambda_j^2)^{-1/4}
\]

Now, take any \( \delta > 0, a > 0, \varepsilon > 0 \). For \( |\beta| > \delta \) we have:

\[
n^a \left| \prod_{j=1}^{n} \zeta_j(\beta) \right| = n^a \prod_{j=1}^{n} (1 + 4\beta^2 \lambda_j^2)^{-1/4}
\]

\[
< n^a \prod_{j=1}^{n} (1 + 4\beta^2 b^2)^{-1/4} = n^a \left( 1 + 4\delta^2 b^2 \right)^{-\frac{a}{4} - g(n)}
\]

where \( g(n) \) is a strictly decreasing function of \( n \), for \( n > \frac{n}{\log(1 + 4\delta^2 b^2)} \). Therefore, given \( \varepsilon > 0 \), there is an \( n^* \) such that for \( n > n^* \), \( g(n) < \varepsilon \). As \( \delta, a, \varepsilon \) were arbitrary, then the hypothesis in equation 2.5.10 holds.

Considering also equations 2.5.8 and 2.5.9 the hypothesis of theorem 2.4.1 are satisfied and therefore the cumulative distribution function of \( \frac{\sum_{i=1}^{n} X_i^2}{\sqrt{n} \sigma^2} = \frac{T_n - tr C(\rho_1, \rho)}{\sqrt{2 tr C(\rho_1, \rho)^2}} \) can be approximated by a truncated Edgeworth series. Dropping the invariants of order \( O(n^{1-s/2}) \) the distribution function of \( T_n \) can be then approximated by:

\[
F_{T_n}(t) = \mathcal{E}_{s-1} \left( \frac{t - k_{1,n}}{\sqrt{k_{2,n}^2}}, \kappa \right) + O \left( n^{1-s/2} \right)
\]
where $\mathcal{E}_s$ is the Edgeworth approximation for the distribution function of standardized sums of iid random variables (see equation 2.4.4), and $\kappa$ are the invariants of $T_n$.  

Knowing the distribution function of the statistic $T_n$ allow us to calculate the critical value $c_\alpha$ of the optimum test.

\[
E_0 \psi(T_n(Y, \rho_1)) = \alpha \\
\Pr_0(T_n(Y, \rho_1) > c_\alpha) = \alpha \\
F_{0,T_n(\rho_1)}(c_\alpha) = 1 - \alpha
\] (2.5.11)

In order to obtain the $1 - \alpha$ quantile of the distribution $F_{0,T_n(\rho_1)}(t)$ one can use the Cornish-Fisher expansion (see Cornish and Fisher, 1938; Fisher and Cornish, 1960; Kolassa, 1997). It is basically a Taylor series expansion of equation 2.5.11 around $t_\alpha$, neglecting terms of order $o(n^{-1})$, and solving for $c_{\alpha,n}(\rho_1)$, to obtain:

\[
c_\alpha \simeq k_1 + \sqrt{k_2} \left( t_\alpha + \frac{\kappa_3 H_2(t_\alpha)}{3!} + \frac{\kappa_4 H_3(t_\alpha)}{4!} - \frac{\kappa_3^2 (2t_\alpha^2 - 5t_\alpha)}{36} \right) \\
\simeq k_1 + \frac{\sqrt{k_2}}{72} \left( (3\kappa_4 - 4\kappa_3^2) t_\alpha^3 + 12\kappa_3 t_\alpha^2 + (-9\kappa_4 + 10\kappa_3^2 + 72)t_\alpha - 12\kappa_3 \right) \] (2.5.12)

where $t_\alpha = \Phi^{-1}(1 - \alpha)$ is the $1 - \alpha$ quantile of the standard Gaussian distribution and the dependence on $(n, \rho_1)$ is implicit in the context.

Remark. Note that, as highlighted in Hall (1992, Theorem 2.4), the Cornish-Fisher expansion is not uniform in $0 < \alpha < 1$. Indeed, in some cases as $\alpha \to 0$ the right-hand side of the Cornish-Fisher expansion can diverge to $+\infty$, but the expected critical value would be $c_\alpha \to -\infty$. Nevertheless, under some moment conditions the approximation can be established uniformly in $n^{-c} \leq \alpha \leq 1 - n^{-c}$, for a given $c > 0$.

Having obtained the critical level $c_{\alpha,n}(\rho_1)$ for an optimum size-$\alpha$ test, one can calculate the power of the test by:

\[
\beta_n(\rho_1) = E_{\rho_1} \psi(T_n(Y, \rho_1)) \\
= 1 - F_{\rho_1,T_n(\rho_1)}(c_{\alpha,n}(\rho_1))
\] (2.5.13)

Summing up, in order to calculate the power envelope for testing the null of no spatial dependence in the SAR stochastic process ($H_0: \rho = 0$) against the alternative that the spatial dependence parameter is not zero ($H_1: \rho = \rho_1 \neq 0$) for all $\rho_1 \in P \setminus \{0\}$, one should follow the procedure:
Box 2.5.1 Calculating the power envelope

(a) pick $\rho_1 \in P, \rho_1 \neq 0$.

(b) calculate the cumulants of $T_n$ under the null, $k_{s,n}(\rho_1) = 2^{s-1}(s-1)! \text{tr} \left( \frac{A(\rho_1)}{\rho_1} \right)^s$, and its respective invariants $\kappa_{s,n}$.

(c) find the critical level $c_{\alpha,n}(\rho_1)$ for a size-$\alpha$ test by using the Cornish-Fisher expansion or by numerically solving $F_{0,T_n(\rho_1)}(c_{\alpha,n}) = 1 - \alpha$ (when $\rho_1 > 0$) or solving $F_{0,T_n(\rho_1)}(c_{\alpha,n}) = \alpha$ (when $\rho_1 < 0$), where the distribution function of $T_n$ under the null hypothesis is approximated by the truncated Edgeworth series, $F_{0,T_n(\rho_1)}(t) \approx \mathcal{E}_s \left( (t - k_{1,n}(\rho_1))/\sqrt{k_{2,n}(\rho_1)}, \kappa_n(\rho_1) \right)$.

(d) calculate the cumulants of $T_n$ under the alternative, i.e.,

$k_{s,n}^a(\rho_1) = 2^{s-1}(s-1)! \text{tr} \left( L^{-1}(\rho_1) \frac{A(\rho_1)}{\rho_1} L^{-1}(\rho_1) \right)^s$, and its respective invariants $\kappa_{s,n}^a$.

(e) approximate the distribution function of $T_n$ under the alternative hypothesis by

$F_{\rho_1,T_n(\rho_1)}(t) \approx \mathcal{E}_s \left( (t - k_{1,n}^a(\rho_1))/\sqrt{k_{2,n}^a(\rho_1)}, \kappa_n^a(\rho_1) \right)$

and calculate the power of the test against the alternative, i.e., $\beta = 1 - F_{\rho_1,T_n(\rho_1)}(c_{\alpha})$ for $\rho_1 > 0$ and $\beta = F_{\rho_1,T_n(\rho_1)}(c_{\alpha})$ for $\rho_1 < 0$.

(f) repeat the above steps for a different $\rho_1 \in P \setminus \{0\}$.

2.5.2 Saddlepoint Series

The saddlepoint series was introduced in section 2.4.2. In this section we will apply those developments to approximate $F_{\rho,T_n(\rho)}(t)$, the distribution of the statistic $T_N(\mathbf{Y}, \rho_1) = \frac{Y^T A(\rho_1) Y}{\sigma^2 \rho_1}$, when the true DGP is given by the parameter $\rho$, in order to calculate the critical level for the optimum test and the power envelope. The cumulant generating function of $T_n$ is:

$$K_{\rho,T_n(\rho_1)}(\gamma) = -\frac{1}{2} \ln \det \left( I - 2\gamma C(\rho_1, \rho) \right) \tag{2.5.14}$$

which is well defined for $\gamma$ in an open set containing the origin (see lemma 2.2.8).

The following proposition characterizes the cumulants of the random variable in the embedding exponential family shown in equation 2.4.12.

2.5.6 Proposition (Cumulants for saddlepoint approximation). In the SAR stochastic process considered, an explicit formula for the cumulants of the random variable embedded in the exponential family is given by:

$$k_{s,\rho,\gamma}(\rho_1) = K_{\rho,T_n(\rho_1)}^{(s)}(\gamma) = 2^{s-1}(s-1)! \text{tr} \left[ \left( I - 2\gamma C(\rho_1, \rho) \right)^{-1} C(\rho_1, \rho) \right]^s \tag{2.5.15}$$

Proof. The cumulant function for $T_n$ is $K_{\rho,T_n(\rho_1)}(\gamma) = -\frac{1}{2} \ln \det \left( I - 2\gamma C(\rho_1, \rho) \right)$. Taking the first differential:

$$dK_{\rho,T_n(\rho_1)}(\gamma) = \frac{1}{2} \text{tr} \left[ \left( I - 2\gamma C(\rho_1, \rho) \right)^{-1} (-2C(\rho_1, \rho)) \right]$$

therefore

$$K'_{\rho,T_n(\rho_1)}(\gamma) = \text{tr} \left[ \left( I - 2\gamma C(\rho_1, \rho) \right)^{-1} C(\rho_0, \rho_1, \rho) \right]$$
Taking the differential of $K'_{\rho,T_n(\rho_1)}(\gamma)$:

$$dK'_{\rho,T_n(\rho_1)}(\gamma) = \text{tr} \left[ - (I - 2\gamma C(\rho_1, \rho))^{-1} (-2C(\rho_1, \rho)d\gamma) (I - 2\gamma C(\rho_1, \rho))^{-1} C(\rho_1, \rho) \right]$$

therefore

$$K''_{\rho,T_n(\rho_1)}(\gamma) = 2\text{tr} \left[ (I - 2\gamma C(\rho_1, \rho))^{-1} C(\rho_1, \rho) \right]^2$$

Proceeding in this way we find the formula for the generic $s$ cumulant.

The saddlepoint Gaussian approximation for the density of $T_n$ is then (see equation 2.4.17):

$$f_{\rho,T_n(\rho_1)}(t) \approx \frac{\exp \left( -\hat{\gamma}(t) \text{tr} (I - 2\hat{\gamma}(t)C)^{-1} C \right)}{2\sqrt{\pi} \left( \det (I - 2\hat{\gamma}(t)C) \text{tr} \left( (I - 2\hat{\gamma}(t)C)^{-1} C \right)^2 \right)^{1/2}}, \ C \equiv C(\rho_1, \rho) \tag{2.5.16}$$

where $\hat{\gamma}(t)$ is the solution of $\text{tr} (I - 2\hat{\gamma} C(\rho_1, \rho))^{-1} C(\rho_1, \rho) = t$. The existence of a solution for the saddlepoint equation is shown in the following proposition.

**2.5.7 Proposition.** For any $t \in \mathbb{R}$ there is a $\hat{\gamma}$ in the domain of the cumulant generating function such that $\text{tr} (I - 2\hat{\gamma} C(\rho_1, \rho))^{-1} C(\rho_1, \rho) = t$ holds.

**Proof.** Note that the matrix $C(\rho_1, \rho) = L^{-1}(\rho)\Lambda(\rho_1)\Lambda(\rho_1)^{-1} L^{-1}(\rho)$ has both positive and negative eigenvalues (as the underlying spatial weight matrix $w(\rho_1)$ has at least one positive and one negative real eigenvalue). Let $\lambda_{\text{max}}$ and $\lambda_{\text{min}}$ denote, respectively, the largest (positive) and the smallest (negative) eigenvalue of $C(\rho_1, \rho)$. Therefore the domain for $\gamma$ is the interval $\left( \frac{1}{2\lambda_{\text{min}}}, \frac{1}{2\lambda_{\text{max}}} \right)$. Note that $\text{tr} (I - 2\gamma C(\rho_1, \rho))^{-1} C(\rho_1, \rho)$ is a weighted sum of the eigenvalues of $C(\rho_1, \rho)$, with weights given by the eigenvalues of $(I - 2\gamma C(\rho_1, \rho))^{-1}$. As $\gamma$ approaches the lower (upper) bound of its domain, it gives an infinitely large weight to the smallest negative (largest positive) eigenvalue of $C$, therefore the trace ranges from $-\infty$ to $+\infty$ as $\gamma$ ranges in its domain. As a result, for any $t \in \mathbb{R}$, there is an $\gamma \in \left( \frac{1}{2\lambda_{\text{min}}}, \frac{1}{2\lambda_{\text{max}}} \right)$ such that the saddlepoint equation holds.

The tail probability can be calculated from:

$$P_{\rho}(T_n(\rho_1) > t) \approx \int_{\gamma(t)} \left( \frac{\text{tr} ((I - 2\gamma C)^{-1} C)}{\pi \det (I - 2\gamma C)} \right)^{1/2} \exp \left( -\gamma \text{tr} (I - 2\gamma C)^{-1} C \right) d\gamma \tag{2.5.17}$$

where $\hat{\gamma}(t)$ is the solution to the saddlepoint equation, $C \equiv C(\rho_1, \rho)$, and $\lambda_{\text{max}}$ is the largest eigenvalue of $C(\rho_1, \rho)$.

In order to calculate the tail probabilities or the distribution function of $T_n$ without calculating the above integral, define

$$a = \text{sign}(\hat{\gamma}) \sqrt{2 \left( \hat{\gamma} \text{tr} (I - 2\hat{\gamma} C)^{-1} C + \frac{1}{2} \log \det (I - 2\hat{\gamma} C) \right)} \tag{2.5.18}$$

$$b = \hat{\gamma} \sqrt{2 \text{tr} \left( (I - 2\hat{\gamma} C)^{-1} C \right)^2} \tag{2.5.19}$$

$$\kappa_{3,T_n} = \frac{8 \text{tr} \left( (I - 2\hat{\gamma} C)^{-1} C \right)^3}{\left( 2 \text{tr} \left( (I - 2\hat{\gamma} C)^{-1} C \right)^2 \right)^{3/2}} \tag{2.5.20}$$
and use one of the approximations described in section 2.4.2: Robinson’s approximation (RA, lemma 2.4.2), Lugannani-Rice’s approximation (LRA, lemma 2.4.3), or Barndorff-Nielsen / Jensen’s approximation (BNJA, lemma 2.4.4), which are reproduced below for convenience:

\[
P_p(T_n(\rho_1) > t) \approx e^{\frac{b^2-a^2}{2}} \left(1 - \Phi(b) \left(1 - \frac{\kappa_3T_n b^3}{6}\right) + \frac{\phi(b)(b^2 - 1)\kappa_3T_n}{6}\right) \quad (\text{RA}) \quad (2.5.21)
\]

\[
P_p(T_n(\rho_1) > t) \approx \begin{cases} 
1 - \Phi(a) + \phi(a) \left(\frac{b}{a} - \frac{1}{a}\right) & \text{for } \gamma \neq 0 \\
1 - \Phi(a) + \frac{\kappa_3T_n}{6} & \text{for } \gamma = 0
\end{cases} \quad (\text{LRA}) \quad (2.5.22)
\]

\[
P_p(T_n(\rho_1) > t) \approx 1 - \Phi(a^*), \text{ where } a^* \equiv a + \frac{1}{a} \ln \left(\frac{b}{a}\right) \quad (\text{BNJA}) \quad (2.5.23)
\]

The procedure outlined in box 2.5.2 illustrates the steps needed to calculate the power envelope for testing the null of no spatial dependence in the SAR stochastic process \((H_0: \rho_0 = 0)\) against the alternative that the spatial dependence parameter is not zero \((H_1: \rho_1 \neq 0)\) for all \(\rho_1 \in P \setminus \{0\}\).

**Box 2.5.2 Calculating power envelope (saddlepoint)**

(a) pick \(\rho_1 \in P \setminus \{0\}\).

(b) numerically solve \(P_0(T_n(\rho_1) > t) = \alpha\) using either of the approximations (RA, LRA, BNJA) or the integral in equation 2.5.17 to find \(\hat{\gamma}_\alpha\), in case \(\rho_1 > 0\) or solve \(P_0(T_n(\rho_1) > t) = 1 - \alpha\) for \(\rho_1 < 0\).

(c) calculate the critical value \(c_\alpha = \operatorname{tr} (I - 2\hat{\gamma}_\alpha C(\rho_1, 0))^{-1} C(\rho_1, 0)\). Alternatively, skip step (b) and calculate the critical value directly from the saddlepoint inversion (lemma 2.4.5).

(d) obtain, numerically, the saddlepoint \((\hat{\gamma})\) associated to the critical level, \(\operatorname{tr} (I - 2\hat{\gamma} C(\rho_1, \rho_1))^{-1} C(\rho_1, \rho_1) = c_\alpha\).

(e) the power of the test against the alternative is then calculated from \(\beta = P_{\rho_1}(T_n(\rho_1) > c_\alpha)\) for \(\rho_1 > 0\) or \(\beta = 1 - P_{\rho_1}(T_n(\rho_1) > c_\alpha)\) for \(\rho_1 < 0\), using either RA, LRA, BNJA, or equation 2.5.17.

(f) repeat the above steps.

### 2.5.3 Simulation-based Testing

An alternative to both Edgeworth and saddlepoint approximation is to perform a Monte Carlo (MC) simulation to calculate the critical value of the optimum test \(\psi(T_n)\). This is also called a Monte Carlo test. The test statistic \(T_n(Y, \rho_1) = (\sigma^2 \rho_1)^{-1} Y' A(\rho_1) Y\) is pivotal both under the null \(H_0: \rho = 0\) and under the alternative \(H_1: \rho = \rho_1\) hypothesis. Therefore, provided the number of simulated data sets \((S)\) is chosen such that \(\alpha(S + 1)\) is an integer (where \(\alpha\) is the test’s significance level), then the Monte Carlo test is exact (see Dufour and Khalaf, 2001; MacKinnon, 2007).

In order to calculate the critical level \(c_\alpha\) for the optimum test one needs to generate \(S\) independent simulated data sets under the null hypothesis and, for each sample generated \((y_s \in \mathbb{R}^n, s = 1, \ldots, S)\), calculate the value of \(t_{s,n} = T_n(y_s, \rho_1)\). The simulated critical value is then the largest \(\alpha(S + 1)\) statistic \(t_{s,n}\) obtained in the simulation. In other words, \(c_\alpha = t_{\alpha(1-\alpha)(S+1)}\), where \(t_{\alpha}\) is the order statistic. Figure 2.2 compares the simulated critical values against those obtained by the procedures described in the previous sections: (i) numerically inverting an Edgeworth approximation, (ii) using the Cornish-Fisher expansion for the quantile function, (iii) numerically inverting the saddlepoint integral (equation 2.5.17), and (iv) calculating the inverse
saddlepoint (lemma 2.4.5). In this simple example, one can show that despite the fact that MC simulation yields an exact test, it is subject to random fluctuations and the number of simulations needed to achieve convergence may be large. Moreover, this example also illustrates that the analytical procedures yield critical values that are close to the true (unknown) critical value. For this example, MC (with 9999 simulations) yield a critical value of 1.691, while procedures (i)-(iv) result in, respectively, 1.689, 1.681, 1.717, and 1.589. As a reference, the critical value based on the asymptotic distribution of $T_n$ for this example is 1.645. The poor performance of the saddlepoint inversion procedure observed in this example is common when the sample size is too small and $\alpha$ is close to the bounds of the interval $(0, 1)$ - see lemma 2.4.5 and the subsequent remark.

Monte Carlo simulation can also be used to generate the power envelope for the optimum test. However, this could potentially be cumbersome as we would need a large number of simulations $S$ to calculate the empirical distribution function $\hat{F}_{\rho_1, T_n(\rho_1)}(t)$ for each $\rho_1 \in P$.

### 2.6 Testing for Spatial Dependence (Unknown Variance)

When the $n \times 1$ disturbance vector $U$ is homoskedastic with unknown variance, the parameter space is defined by $\Theta = (\rho, \sigma^2) \in P \times \mathbb{R}_{++}$ and the parametric model is $\mathcal{M} = (\mathcal{Y}, \{F_\theta, \theta \in \Theta\})$, where $\mathcal{Y} \subset \mathbb{R}^n$ is the sample space and $F_\theta$ is a family of probability distributions with density given by equation 2.5.1.

In this case, the test presented in section 2.5 is not anymore an optimum test. In the quest for an optimality result, we will restrict attention to invariant tests.

#### 2.6.1 Testing Problem with Unknown Variance

Consider the following group of transformations acting on the sample space $\mathcal{Y}$:

$$y = (y_1, \ldots, y_n) \rightarrow (cy_1, \ldots, cy_n), \ c \neq 0$$

(2.6.1)
The statistics $V \equiv (Y_i, \ldots, Y_{n-1})$ is a maximal invariant (see Lehmann, 1986) under this group of transformation. If $Y_i = 0$ one can choose another $Y_i$, and this division can always be done provided $Y \neq (0, \ldots, 0)$. But the probability of this event is zero, so the maximal invariant can be constructed with probability 1. Consider the bijective function $Z : Y \rightarrow Y$ defined by $Z_i = \frac{Y_i}{Y_n}, i = 1, \ldots, n - 1$ and $Z_n = Y_n$. Define $\tilde{Z} \equiv (V', 1)'$, therefore we can write $Y = Z_n \tilde{Z}$. The density of $Z$ can be obtained from

\[
f(z, \theta) = f(y(z), \theta) \left| \det \frac{\partial y(z)}{\partial z} \right| \tag{2.6.2}
\]

\[
= (2\pi\sigma^2)^{-\frac{n}{2}} \det L(\rho) \frac{z_n^2 \lambda_L(\rho) L(\rho) z}{2\sigma^2} \left| \det \frac{\partial y(z)}{\partial z} \right| \tag{2.6.3}
\]

\[
= (2\pi\sigma^2)^{-\frac{n}{2}} \det L(\rho) e^{-\frac{z_n^2 \lambda_L(\rho) L(\rho) z}{2\sigma^2}} \left| \frac{\partial y(z)}{\partial z} \right| z_n^{n-1} \tag{2.6.4}
\]

To obtain the density of the maximal invariant $V$ we integrate $f(z)$ with respect to $z_n$:

\[
f(v, \theta) = (2\pi\sigma^2)^{-\frac{n}{2}} \det L(\rho) \int_{-\infty}^{\infty} e^{-\frac{z_n^2 \lambda_L(\rho) L(\rho) z}{2\sigma^2}} |z_n|^{n-1} dz_n 
\]

\[
= (2\pi\sigma^2)^{-\frac{n}{2}} \det L(\rho) 2 \int_{0}^{\infty} e^{-\frac{z_n^2 \lambda_L(\rho) L(\rho) z}{2\sigma^2}} z_n^{n-1} dz_n 
\]

\[
= (2\pi\sigma^2)^{-\frac{n}{2}} \det L(\rho) 2 \int_{0}^{\infty} z_n^{n-1} e^{-K z_n^2} dz_n, \text{ where } K \equiv \frac{\lambda_L(\rho) L(\rho) z}{2\sigma^2} 
\]

\[
= K^{-\frac{n}{2}} (2\pi\sigma^2)^{-\frac{n}{2}} \det L(\rho) \left( \frac{1}{K} \right)^{-\frac{n}{2}} \int_{0}^{+\infty} t^{-\frac{n}{2}-1} e^{-t} dt 
\]

\[
= \frac{n}{2} \det L(\rho) \left( \frac{\lambda_L(\rho) L(\rho) z}{2\sigma^2} \right)^{-\frac{n}{2}} \Gamma \left( \frac{n}{2} \right) \tag{2.6.5}
\]

Note that, as expected, the distribution of the maximal invariant $V$ does not depend on $\sigma^2$, therefore we can write $f(v, \rho)$ and we are back to the one parameter case, where the Newman-Pearson lemma can be applied to find an optimum invariant test.

2.6.1 Proposition (Optimum invariant test). Define the statistic $V_n(Y, \rho_1) \equiv \sqrt{n} Y^A(\rho_1) Y$. The optimum invariant size $\alpha$ test for testing $H_0 : (\rho = 0, \sigma^2 > 0)$ vs. $H_1 : (\rho = \rho_1 > 0, \sigma^2 > 0)$ is given by

\[
\psi(y) = \begin{cases} 
1 & V_n(y, \rho_1) > c_\alpha \\
0 & V_n(y, \rho_1) < c_\alpha 
\end{cases}
\]

where $c_\alpha$ is determined from $E_\theta \psi(V_n) = \alpha$. For $\rho_1 < 0$ the inequalities are reversed. Moreover, when $\rho_1 \rightarrow 0$ this test is also locally uniformly most powerful invariant (LUMPIT) for testing $H_0 : (\rho \leq 0, \sigma^2 > 0)$ vs. $H_1 : (\rho > 0, \sigma^2 > 0)$. 
Proof. From the Neyman-Pearson lemma an optimum invariant test of size $\alpha$ for testing $H_0 : \rho = 0$ vs. $H_1 : \rho = \rho_1$ rejects the null when \( \frac{f(v, \rho_1)}{f(v, 0)} > c_\alpha \). Expanding:

\[
\det L(\rho_1) \left( \frac{\varepsilon' L(\rho_1)' L(\rho_1) \varepsilon}{\varepsilon' \varepsilon} \right)^{-\frac{d}{2}} > c_\alpha \quad \frac{\varepsilon' L(\rho_1)' L(\rho_1) \varepsilon}{\varepsilon' \varepsilon} < c_\alpha
\]

\[
1 - \frac{\varepsilon' A(\rho_1) \varepsilon}{\rho_1 \varepsilon' \varepsilon} < c_\alpha \quad \sqrt{n} \frac{\varepsilon' A(\rho_1) \varepsilon}{\rho_1 \varepsilon' \varepsilon} > c_\alpha \quad \text{for } \rho_1 > 0, \text{ and}
\]

\[
\sqrt{n} \frac{\varepsilon' A(\rho_1) \varepsilon}{\rho_1 \varepsilon' \varepsilon} < c_\alpha \quad \text{for } \rho_1 < 0
\]

But $\varepsilon = \frac{v}{\sqrt{n}}$, therefore the critical region for the optimum test can be written as

\[
\sqrt{n} \frac{y' A(\rho_1) y}{\rho_1 y' y} > c_\alpha, \text{ for } \rho_1 > 0
\]

with the inequality reversed for $\rho_1 < 0$. Note also that $\lim_{\rho_1 \to 0} \frac{A(\rho_1)}{\rho_1} = W(0)' + W(0)$ therefore the critical region is equivalent to $\frac{\partial \log f(v, \cdot)}{\partial \rho} \bigg|_{\rho = 0} > c$ and thus this test is also locally uniformly most powerful invariant (LUMPI) for $H_0 : \rho \leq 0$ vs. $H_1 : \rho > 0$ (see Lehmann, 1986; Gourieroux and Monfort, 1995; Kariya, 1980).

In order to find the approximate distribution for the statistic $V_n$ of the optimal invariant test, we develop in the following sections asymptotic approximations (Edgeworth, saddlepoint) for a modified statistic $V_{n,v}$.

### 2.6.2 Edgeworth Approximation

Define the following statistic

\[
V_{n,v}^* \equiv \left( \frac{U}{\sigma} \right)' L(\rho)^{-1} \left( \frac{A(\rho_1)}{\rho_1} - \frac{v I}{\sqrt{n}} \right) L(\rho)^{-1} \frac{U}{\sigma} \quad \text{(2.6.7)}
\]

and let $D(v, \rho_1, \rho) \equiv L(\rho)^{-1} (A(\rho_1)/\rho_1 - v I/\sqrt{n}) L(\rho)^{-1}$. The following lemma characterizes the cumulants of $V_{n,v}^*$.

#### 2.6.2 Lemma. The cumulants of $V_{n,v}^*$ are given by

\[
k_{s,v} V_{n,v}^* = 2^{s-1} (s - 1)! \text{ tr } D(v, \rho_1, \rho)^s, \text{ for } s = 1, 2, \ldots
\]

Proof. Direct application of lemma 2.2.8.

#### 2.6.3 Lemma (Convergence rate for $D(v, \rho_1, \rho)$). Let $w(\rho)$ be a spatial weight matrix satisfying assumptions 2.2.1 and 2.2.4. Assume further that the row and column sums of the matrix $L^{-1}(\rho)$ are uniformly bounded in absolute value for all $\rho \in P$. Then it follows that $\text{ tr } D(v, \rho_1, \rho)^s = O(n), \forall \rho_1, \rho \in P$ and $v \in \mathbb{R}$.
Proof. Corollary following proposition 2.2.5 shows that the row and column sums of \( w(\rho) \) and its transpose are uniformly bounded in absolute value. Lemma 2.2.6 assures that \( w(\rho_1)'w(\rho_1) \) has its row and column sums uniformly bounded in absolute value, and therefore, \( \frac{A(\rho_1)}{\rho_1} = \frac{w(\rho_1)'w(\rho_1)}{\rho_1} + o(1) \) and \( D(v, \rho_1, \rho) = L^{-1}(\rho)'(\frac{A(\rho_1)}{\rho_1} - \frac{v}{\sqrt{n}})L^{-1}(\rho) \) are bounded in absolute value. The conclusion follows from lemma 2.2.7. \( \square \)

2.6.4 Proposition. The distribution function of the statistic \( V^*_{n,v} \), approximated by an Edgeworth truncated series by dropping terms of size \( O(n^{-1/2}) \), satisfy:

\[
F_{\rho,V^*_{n,v}(\rho_1)}(x) = \mathcal{E}_{s-1}\left(\frac{x - k_1V^*_{n,v}}{\sqrt{k_2V^*_{n,v}}}, \kappa\right) + O(n^{-1/2}) \tag{2.6.8}
\]

Proof. analogous to proposition 2.5.5, replacing matrix \( C(\rho_1, \rho) \) by \( D(v, \rho_1, \rho) \). \( \square \)

Note that

\[
F_{\rho,V^*_{n,v}(\rho_1)}(0) = P(V^*_{n,v}(\rho_1, \rho) \leq 0) = P\left(\frac{U}{\sigma}D(v, \rho_1, \rho)\frac{U}{\sigma} \leq 0\right)
\]

\[
= P\left(\frac{Y'}{\sqrt{n}}\frac{A(\rho_1)}{\rho_1}Y \leq 0\right) = P\left(\frac{vY'}{\sqrt{n}} \leq \frac{vY'}{\sqrt{n}}\right) = F_{\rho,V_n(\rho_1)}(v)
\]

As a result, the approximated distribution function of \( V_n \) at a point \( v \) can be calculated by evaluating the (approximated) distribution of \( V^*_{n,v} \) at the origin. For instance, in order to find the critical value for an optimum size \( \alpha \) test, we need to solve \( F_{0, V_n(\rho_1)}(c_\alpha) = 1 - \alpha \). But this is equivalent to solving \( F_{0, V_n,c_\alpha}(\rho_1)(0) = 1 - \alpha \), where the matrix \( D \) is given by \( D = \frac{A(\rho_1)}{\rho_1} - \frac{v\mathcal{I}}{\sqrt{n}}. \)

The drawback is that this approximation has to be calculated for each value of \( v \), but this was already the case in the saddlepoint approximation of the statistic \( T_n \) (section 2.5). The equations to find the critical value of the test can be simplified in the typical case where the spatial weight matrix is symmetric, and we consider a LUMPI test, as shown in the theorem below.

2.6.5 Theorem (Critical value with symmetric spatial weight matrix). Assume the spatial weight matrix \( w(\rho) \) satisfies assumptions 2.2.1 and 2.2.4, and let \( W(\rho) = \frac{\partial w(\rho)}{\partial \rho} \). Consider the locally uniformly most powerful invariant (LUMPI) level \( \alpha \) test for testing the null of no spatial dependence \( H_0 : \rho \leq 0 \) against \( H_1 : \rho > 0 \). This test rejects the null hypothesis when \( V_n = \frac{2\sqrt{n}Y'W(0)Y}{VY} > c_\alpha. \) Then, the critical level for the test is given by

\[
c_\alpha = t_\alpha \left(\frac{8\text{tr} W(0)^2}{n}\right)^{1/2} + \frac{32(t_\alpha^2 - 1)\text{tr} W(0)^3}{3\sqrt{n}} + o(n^{-1/2})
\]

or

\[
c_\alpha = c_{\alpha,i^*} + o(n^{-1})
\]

where \( t_\alpha = \Phi^{-1}(1 - \alpha) \) is the \( 1 - \alpha \) quantile of the standard Gaussian distribution, and \( c_{\alpha,i^*} \) is the smallest positive real root of the cubic equation \( a_3c_{a,i}^3 + a_1c_{a,i} + a_0 = 0 \), where \( a_i, i = 0, 1, 3 \), are functions of \( t_\alpha \), of the sample size \( n \), and of powers of \( \text{tr} W(0)^j, j = 2, 3, 4 \). The expressions for \( a_i, i = 0, 1, 3 \) and for the roots of the cubic equation are given in the proof.
In most practical situations,

\[ c_\alpha = -\frac{a_0}{a_1} = \left( t_\alpha + \frac{\text{tr} W(0)}{(\text{tr} W(0))^2} \sqrt{\frac{2}{3}} (t_\alpha^2 - 1) + \frac{(\text{tr} W(0))^3}{(\text{tr} W(0))^2} \left( \frac{10t_\alpha^2 - 4t_\alpha}{9} \right) + \frac{\text{tr} W(0)^4}{(\text{tr} W(0))^2} \left( \frac{t_\alpha^2 - 3t_\alpha}{2} \right) \right) \left( \frac{n}{8 \text{tr} W(0)^2} \right)^{1/2} \left( 1 + \frac{2(t_\alpha^2 - 1)}{n} \right) \]

proves to be a good approximation, avoiding the calculation of the roots of the cubic equation.

**Proof.** In order to find the critical value for an \( \alpha \)-size test we need to solve \( F_{0,V_n,\rho}(c_\alpha) = 1 - \alpha \), but this is equivalent to solving \( F_{0,V_n,\rho}^{\rho}(0) = 1 - \alpha \), with \( D = 2W(0) - \frac{c_\alpha I}{\sqrt{n}} \). From the Cornish-Fisher approximation (equation (2.5.12) we have

\[ 0 = k_{1,V_n} + k_{2,V_n} \left( t_\alpha + \frac{k_{3,V_n} H_2(t_\alpha)}{(k_{2,V_n})^{3/2} 3!} + \frac{k_{4,V_n} H_3(t_\alpha)}{(k_{2,V_n})^{3/2} 4!} - \frac{(k_{3,V_n})^2 (2t_\alpha^2 - 5t_\alpha)}{(k_{2,V_n})^2 36} + o(n^{-1}) \right) \]

therefore

\[ -\frac{k_{1,V_n}}{k_{2,V_n}} = t_\alpha + \frac{k_{3,V_n} H_2(t_\alpha)}{(k_{2,V_n})^{3/2} 3!} + \frac{k_{4,V_n} H_3(t_\alpha)}{(k_{2,V_n})^{3/2} 4!} - \frac{(k_{3,V_n})^2 (2t_\alpha^2 - 5t_\alpha)}{(k_{2,V_n})^{3/2} 36} + o(n^{-1}) \]

(2.6.10)

where \( t_\alpha \) is the \( 1 - \alpha \) quantile of the Gaussian distribution, \( H_s(t_\alpha), s = 2, 3 \), are the Hermite polynomials, and the cumulants of \( V_n \) are given by:

\[ k_s, V_n = 2^{s-1} (s-1)! \text{tr} \left( 2W(0) - \frac{c_\alpha I}{\sqrt{n}} \right)^s \]

Expanding the calculations we obtain:

\[ k_{1,V_n} = -c_\alpha \sqrt{n} \]
\[ k_{2,V_n} = 8 \text{tr} W(0)^2 + 2c_\alpha^2 \]
\[ k_{3,V_n} = 64 \text{tr} W(0)^3 - \frac{96c_\alpha \text{tr} W(0)^2}{\sqrt{n}} - \frac{8c_\alpha^3}{\sqrt{n}} \]
\[ k_{4,V_n} = 768 \text{tr} W(0)^4 - \frac{1536c_\alpha \text{tr} W(0)^3}{\sqrt{n}} + \frac{1152c_\alpha^2 \text{tr} W(0)^2}{n} + \frac{c_\alpha^4}{n} \]

Calculating the ratios of the cumulants that appear in equation 2.6.10 we obtain:

\[ -\frac{k_{1,V_n}}{k_{2,V_n}} = c_\alpha \left( \frac{n}{8 \text{tr} W(0)^2} \right)^{1/2} \left( 1 - \frac{c_\alpha^2}{8 \text{tr} W(0)^2} \right) + o(n^{-1}) \]
\[ \frac{k_{3,V_n}}{(k_{2,V_n})^{3/2}} = \left( \frac{n}{8 \text{tr} W(0)^2} \right)^{3/2} \left( \frac{64 \text{tr} W(0)^3}{n^{3/2}} - \frac{96c_\alpha \text{tr} W(0)^2}{n^{3/2}} \right) + o(n^{-1}) \]
\[ \frac{k_{4,V_n}}{(k_{2,V_n})^{2}} = \left( \frac{n}{8 \text{tr} W(0)^2} \right)^2 \frac{768 \text{tr} W(0)^4}{n^2} + o(n^{-1}) \]
\[ \frac{(k_{3,V_n})^2}{(k_{2,V_n})^{3}} = \left( \frac{n}{8 \text{tr} W(0)^2} \right)^3 \left( \frac{64 \text{tr} W(0)^3}{n^{3/2}} \right)^2 + o(n^{-1}) \]

Substituting this ratios back in equation 2.6.10 and neglecting terms of order \( o(n^{-1}) \) we obtain a cubic equation in \( c_\alpha \) of the following form:

\[ a_3 c_\alpha^3 + a_1 c_\alpha + a_0 = 0 \]
where

\[
a_3 = -\frac{1}{n} \left( \frac{n}{8 \text{ tr } W^2} \right)^{3/2}
\]

\[
a_1 = \left( \frac{n}{8 \text{ tr } W^2} \right)^{1/2} \left( 1 + \frac{2(t_\alpha^2 - 1)}{n} \right)
\]

\[
a_0 = -\left( t_\alpha + \frac{\text{ tr } W^3}{(\text{ tr } W^2)^{3/2}} \sqrt{\frac{2}{3}} (t_\alpha^2 - 1) + \left( \frac{\text{ tr } W^3}{(\text{ tr } W^2)^{2}} \right) \left( \frac{10t_\alpha - 4t_\alpha^3}{9} \right) + \frac{\text{ tr } W^4}{(\text{ tr } W^2)^2} \left( \frac{t_\alpha^3 - 3t_\alpha}{2} \right) \right)
\]

Note that \( a_3 \) is usually a small number (it goes to zero as the sample size increases) and therefore the discriminant of the cubic equation is negative (provided also that \( a_1 \) is positive, which will happen if the sample size is larger than 2), resulting in three real roots for the above equation. The critical value \( c_\alpha \) is then the smallest positive root\(^{14} \) of the cubic equation. The three roots are given by:

\[
c_{\alpha,1} = Q^{1/3} - \frac{a_1}{3a_3} Q^{-1/3}
\]

\[
c_{\alpha,2} = zQ^{1/3} - \frac{a_1}{3a_3} zQ^{-1/3}
\]

\[
c_{\alpha,3} = \bar{z}Q^{1/3} - \frac{a_1}{3a_3} \bar{z}Q^{-1/3}
\]

where \( z = -\frac{1}{2} + \frac{\sqrt{3}}{2} i, \bar{z} \) is the complex conjugate of \( z, Q = \frac{\sqrt{2}}{\delta_Y \sqrt{a_3}} - \frac{a_0}{a_1}, \) and \( \Delta = 27a_3^2 + \frac{4a_3^3}{a_5} \).

The critical value is then the smallest positive root among the \( c_{\alpha,i}, i = 1, 2, 3 \) from above.

Note that as \( n \to \infty, a_3 \to 0, a_1 \to \left( \frac{n}{8 \text{ tr } W(0)^2} \right)^{1/2} \), and \( a_0 \to -t_\alpha \), therefore for large \( n \) the critical value is given by \( c_\alpha = -\frac{a_0}{a_1} = t_\alpha \left( \frac{8 \text{ tr } W(0)^2}{n} \right)^{1/2} \), which is the expected value if we recall that in the current case \( \text{EV}_n \to 0 \) and \( \text{Var } V_n \to \frac{8 \text{ tr } W(0)^2}{n} \).

Repeating the calculations, but now dropping terms of order \( o(n^{-1/2}) \) leads to the first approximation to the critical value in this theorem. \( \square \)

Summing up, the scheme in box 2.6.1 shows how to use an Edgeworth approximation to calculate the power envelope for an optimal invariant procedure for testing the null of no spatial dependence in the SAR stochastic process, \( H_0 : (\rho_0 = 0, \sigma^2 > 0) \), against the alternative that the spatial dependence parameter is not zero, \( H_1 : (\rho_1 \neq 0, \sigma^2 > 0) \), for all \( \rho_1 \in P \setminus \{0\} \).

\(^{14}\)There is always a positive root, since the sum of all roots add to zero.
Box 2.6.1 Calculating the power envelope for the invariant test (Edgeworth)

(a) pick \( \rho_1 \neq 0 \).

(b) define the cumulants of \( V_{n}^*(\nu) \) under the null as function of \( \nu \), i.e., \( k_{s,V_{n}}(\nu) = 2^{s-1}(s-1)! \text{tr} \left(A(\rho_1) - \frac{\nu}{\sqrt{n}} \right)^s \).

(c) approximate the distribution function of \( V_{n}^*(\nu) \) at zero using an Edgeworth expansion, e.g.,
\[
F_{0,V_{n},(0)}(0) \approx \mathcal{E}_4 \left( -\frac{k_{2,V_{n}}(\nu)}{\sqrt{k_{2,V_{n}}(\nu)}} \kappa_{3,V_{n}}(\nu), \kappa_{4,V_{n}}(\nu) \right)
\]

(d) find the critical level \( c_\alpha \) for a size-\( \alpha \) test by numerically solving \( F_{0,V_{n,c_{\alpha}}}(0) = 1 - \alpha \) (when \( \rho_1 > 0 \)) or solving \( F_{0,V_{n,c_{\alpha}}}(0) = \alpha \) (when \( \rho_1 < 0 \)).

(e) calculate the cumulants of \( V_{n}^*(\nu) \) under the alternative, i.e.,
\[
k_{s,V_{n}}(\nu) = 2^{s-1}(s-1)! \text{tr} \left[ \left(L^{-1}(\rho_1) \left( A(\rho_1) - \frac{\nu}{\sqrt{n}} \right) \right)^s \right]
\]

(f) approximate the distribution function of \( V_{n}^*(\nu) \) under the alternative hypothesis by
\[
F_{\rho_1,V_{n},(\nu)}(0) \approx \mathcal{E}_4 \left( -\frac{k_{2,V_{n}}(\nu)}{\sqrt{k_{2,V_{n}}(\nu)}} \kappa_{3,V_{n}}(\nu), \kappa_{4,V_{n}}(\nu) \right), \text{and calculate the power of the test against the alternative, i.e., } \beta = 1 - F_{\rho_1,V_{n,c_{\alpha}}}(0) \text{ for } \rho_1 > 0, \text{ and } \beta = F_{\rho_1,V_{n,c_{\alpha}}}(0) \text{ for } \rho_1 < 0.
\]

(g) repeat the above steps for a different \( \rho_1 \in \mathbb{P} \setminus \{0\} \).

2.6.3 Saddlepoint Approximation

Instead of approximating the distribution of \( V_{n,v}^* \) using an Edgeworth expansion (see equation 2.6.8) we could use a saddlepoint approximation. For that we follow the same steps than in section 2.4.2. The cumulant generating function of \( V_{n,v}^* \) is:

\[
K_{\rho_1,V_{n,v}^*}(\gamma) = -\frac{1}{2} \ln \det \left( I - 2\gamma D(\nu, \rho_1, \rho) \right)
\]

and the saddlepoint Gaussian approximation for its density is given by

\[
f_{\rho_1,V_{n,v}^*}(\nu^*) \approx \frac{\exp \left( \gamma(\nu^*) \text{tr} \left( I - 2\gamma(\nu^*) D \right)^{-1} D \right)}{\left( 4\pi \det \left( I - 2\gamma(\nu^*) D \right) \text{tr} \left( (I - 2\gamma(\nu^*) D)^{-1} D \right)^2 \right)^{1/2}}, \quad D \equiv D(\nu, \rho_1, \rho)
\]

where \( \gamma(\nu^*) \) solves the saddlepoint equation \( \text{tr} \left( I - 2\gamma D(\nu, \rho_1, \rho) \right)^{-1} D(\nu, \rho_1, \rho) = \nu^* \). In order to calculate the tail probability, we follow equation 2.4.19:

\[
P(V_n > \nu) = P(V_{n,v}^* > 0)
= \frac{1}{2^{n_{\max}}} \left( \frac{\text{tr} \left( (I - 2\gamma D)^{-1} D \right)^2}{\pi \det (I - 2\gamma D)} \right)^{1/2} \exp \left( -\gamma \text{tr} (I - 2\gamma D)^{-1} D \right) d\gamma
\]

where \( \gamma(\nu) \) solves \( \text{tr} (I - 2\gamma(\nu) D(\nu, \rho_1, \rho))^{-1} D(\nu, \rho_1, \rho) = 0 \), and \( \lambda_{\max} \) is the largest eigenvalue of \( D \equiv D(\nu, \rho_1, \rho) \).
Alternatively, the tail probabilities can be calculated using the approximations defined in section 2.4.2. Define

\[
a = \text{sign}(\gamma) \sqrt{\log \det (I - 2\gamma D(v, \rho_1, \rho))}
\]

(2.6.14)

\[
b = \gamma \sqrt{2 \text{tr} \left( (I - 2\gamma D(v, \rho_1, \rho))^{-1} D(v, \rho_1, \rho) \right)^2}
\]

(2.6.15)

\[
\kappa_{V_{\alpha, v}} = \frac{8 \text{tr} \left( (I - 2\gamma D(v, \rho_1, \rho))^{-1} D(v, \rho_1, \rho) \right)^3}{\left( 2 \text{tr} \left( (I - 2\gamma D(v, \rho_1, \rho))^{-1} D(v, \rho_1, \rho) \right)^2 \right)^{3/2}}
\]

(2.6.16)

and then use one of Robinson’s approximation (RA), Lugannani-Rice’s approximation (LRA), or Barndorff-Nielsen/Jensen’s approximation (BNJA) to calculate the tail probabilities of \( V_n \), as in equations 2.5.21 to 2.5.23.

The saddlepoint approximation for the invariant test statistics \( V_n \) is computationally more demanding, as both for the integral equation 2.6.13 and for the RA, LRA, and BNJA approximations, the quantile function depend on the saddlepoint \( \gamma \) and on the ordinate \( v \), and there is no analytical form relating both. Box 2.6.2 summarizes the steps needed to calculate the power envelope for the invariant procedure for testing the presence of spatial dependence in the SAR stochastic process based on the statistic \( V_n \), using a saddlepoint approximation.

---

**Box 2.6.2 Calculating power envelope (saddlepoint)**

(a) pick \( \rho_1 \neq 0 \).

(b) numerically solve \( P_0(V_n(\rho_1) > c_\alpha) = \alpha \) (when \( \rho_1 > 0 \)) using equation 2.6.13 or the approximations RA, LRA, BNJA. For \( \rho_1 < 0 \) solve \( P_0(V_n(\rho_1) > c_\alpha) = 1 - \alpha \).

(c) obtain, numerically, the saddlepoint \( \gamma_\alpha \) associated to the critical level \( c_\alpha \), i.e., \( \gamma_\alpha \) solves \( \text{tr} \left( I - 2\gamma_\alpha D(c_\alpha, \rho_1, \rho_1) \right)^{-1} D(c_\alpha, \rho_1, \rho_1) = 0 \).

(d) the power of the test against an alternative \( \rho_1 > 0 \) can then be calculated from \( \beta = P_{\rho_1}(V_n(\rho_1) > c_\alpha) = P_{\rho_1}(V_{n, c_\alpha}^*(\rho_1) > 0) \) using again equation 2.6.13 or the approximations RA, LRA, BNJA. For \( \rho_1 < 0 \), \( \beta = 1 - P_{\rho_1}(V_{n, c_\alpha}^*(\rho_1) > 0) \).

(e) repeat the above steps for a different \( \rho_1 \in P \setminus \{0\} \).

---

### 2.7 Evaluating the Accuracy of the Proposed Approximations

In this section we will chose some specific designs for the spatial weight matrix in order to exemplify the procedure to (i) obtain the small sample critical value for testing the null hypothesis of no spatial dependence in the SAR stochastic process, and to (ii) derive the power envelope of the test. The goal is to verify how sensitive the properties of the optimum test are to changes in the sample size and in the structure of the spatial weigh matrix. In order to do that we need flexibility to create the weight matrices with a well defined structure and with some sort of regularity that would allow us to increase the sample size while maintaining the underlying structure. The easiest way to achieve this is to work with circulant (or more generally, with Toeplitz) matrices (see appendices A.1.1 and A.1.2 for definition and properties of these matrices).

---

15The results in this section were obtained using The R Project for Statistical Computing, 2011.
The first design is called circular world\textsuperscript{16}, which is applied in several Monte Carlo simulations (see, for example, Kelejian and Prucha, 1999). In this design, the spatial weight matrix is defined based on a binary $n \times n$ matrix $M$, which is a circulant matrix generated by the vector:

$$(0, 1, \ldots, 1, 0, \ldots, 0, 1, \ldots, 1) \in \mathbb{R}^n$$

which is also the first row of $M$ (see definition A.1.1 in the appendix). Lines 2, \ldots, $n$ are defined in a way that each row is a right cyclic shift of the previous one, i.e., the elements of the matrix are defined by $M_{ij} = M_{1,1+(j-i) \mod n}$ for $i, j = 1, \ldots, n$. Figure 2.3a shows an example of a graph that has this neighborhood structure.

![Circular world](image1)

![Asymmetric neighbor relation](image2)

**Figure 2.3:** Example of designs 1 and 2 ($n = 20$, and $m = 2$).

The second design is similar to the first design, but instead of a circular world with $m$-ahead and $m$-behind neighbors, we introduce an asymmetry by considering each neighbor having $(2m-1)$-ahead and 1-behind neighbors. It is also a circulant matrix, but generated by the vector:

$$(0, 1, \ldots, 1, 0, \ldots, 1) \in \mathbb{R}^n$$

Note that both designs 1 and 2 represent regular structures\textsuperscript{17} in the sense that each spatial unit in a given design has the same number of neighbors. Also, if we define the degree ($d_i$) of each spatial unit $i$ as the sum of its weights attributed to spatial unit $j$ (i.e., $d_i = \sum_j w_{ij}(\rho)$), then both designs are regular with the same degree. The structure of design 2 can be represented by a directed graph, as shown in Figure 2.3b, where one of the spatial units is highlighted.

The third design introduces variability in the number of neighbors of each spatial unit, while keeping the average degree of the spatial units equal to the previous three designs. To achieve this, we create a vector with $2n - 1$ elements of the following form:

$$t = (0, 1, \ldots, 1, 0, \ldots, 0, 1, 0, 0, \ldots, 0, 1, \ldots, 1, 0) \in \mathbb{R}^{2n-1}$$

and define the vector $t^* \in \mathbb{R}^{2n-1}$ by $t^* = \frac{2mn}{2((n-1)+(J-1)(J+2))} \times t$. Then we use $t^*$ to generate an $n \times n$ Toeplitz matrix (see Definition A.1.4 in appendix) representing the neighborhood structure.

\textsuperscript{16}Also referred to as $m$-ahead and $m$-behind spatial weight matrices.

\textsuperscript{17}Definition comes from the underlying regular graph.
This spatial weight matrix has an average degree of spatial units equal to designs 1, and 2, but it does not represent a regular structure anymore. Instead, it allows for the number of neighbors of each spatial unit to range from 2 to $J + 1$. See figure 2.4a for an example.

For designs 1 to 3, the spatial weight matrix is defined as $w_j(\rho) = \rho M_j$, where $M_j$, $j = 1, 2, 3$ are the matrices just defined. Therefore all are linear functions of the parameter $\rho$. The fourth design introduces some non-linearity by considering the notion of distance implied in the graph whose adjacency structure is represented by a given spatial weight matrix to construct second order SAR stochastic process. Consider, for example, a circular world (design 1) with $k$-ahead and $k$-behind neighbors, represented by the matrix $M_k$. In this design, each spatial unit $i$ has $2k$ neighbors, and we assume the weights attributed to each neighbor is $\rho$. We can also say that the shortest distance among neighboring spatial units is 1. Now consider a binary matrix $M'_k$ in which the elements $\{i, j\}$ are non-zero if the shortest-distance between spatial units $i$ and $j$ are 2, and define $w(\rho) = \rho M_k + \rho^2 M'_k$, where a weight $\rho^2$ is attributed to spatial units whose distance is 2. For the circular world design, it can be shown that the matrix $w(\rho)$ is also a circulant matrix generated by the vector:

$$(0, \rho, \ldots, \rho, \rho^2, \ldots, \rho^2, 0, \ldots, 0, \rho^2, \ldots, \rho^2, \rho, \ldots, \rho) \in \mathbb{R}^n$$

This design is also a regular structure (as designs 1 and 2) in which each spatial unit has $4k$ neighbors, but it does not have the same average degree than designs 1 to 3. Figure 2.4b illustrates this design, where the bold lines represent $\rho$ weights and the thin lines represent $\rho^2$ weights.

Finally, the fifth design consists of a real world example of some possible neighborhood structures among Rio de Janeiro’s 92 municipalities. For this design it is not possible to check the asymptotic behavior as the sample size is fixed, but we can define spatial weight matrices that will have different degrees of connectivity, and assess the effect of the neighborhood structure’s choice on the properties of the tests to detect the presence of spatial dependence. The first structure (design 5a) builds the neighborhood relation based on a contiguity criteria, i.e., two municipalities (spatial units) are neighbors if they share one or more boundary points. For the case of Rio de Janeiro, this results in a spatial weight matrix with an average number of neighbors of 4.89, with one spatial unit having 10 neighbors and three having 1 neighbor (see figure 2.5).

---

18 This is also known as queen neighborhood structure (see Anselin, 1988).
The second and third structures are based on the geodesic distance among the cities. For the second structure (design 5b), two spatial units (municipalities) are neighbors if the geodesic distance between them is smaller than 44.5Km. This creates a spatial weight matrix with an average number of neighbors of 10.2, with one spatial unit having 21 neighbors and one with 1 neighbor. In the third structure (design 5c), two spatial units are connected if the geodesic distance between their cities is smaller than 63Km. This results in the number of neighbors ranging from 1 to 32, with an average of 18.1.

Figure 2.5: Contiguity structure for Rio de Janeiro’s municipalities (design 5a)

Proposition 2.2.5 assures that, if the spatial weight matrices satisfy the assumptions 2.2.1 and 2.2.4, then there is a non-degenerate open interval, containing zero, independent of the sample size \( n \), such that \( L(\rho) = I - w(\rho) \) is non-singular for all \( \rho \) in this interval. However, as designs 1 to 4 are represented by Toeplitz and circulant matrices their eigenvalues can be analytically calculated using proposition A.1.2, and it can be shown that the matrix \( L(\rho) \) is non-singular for all \( \rho \in \left(-\frac{1}{2m}, \frac{1}{2m}\right)^{19} \) for designs 1 and 2. For design 3 the parameter space could be chosen as \( \left(-\frac{1}{m}, \frac{1}{m}\right) \), where \( m^* = \frac{2m}{J+1} \), and for design 4, \( \rho \in \left(-\frac{\sqrt{1+4m^{-T}-1}}{2}, \frac{\sqrt{1+4m^{-T}-1}}{2}\right) \). For designs 5a, 5b, and 5c the parameter space is \((-0.31, 0.16), (-0.23, 0.06), (-0.18, 0.04)\), respectively\(^{20}\). Table 2.2 summarizes the proposed designs.

\(^{19}\)The lower bound is not tight, while the upper bound is the best possible.

\(^{20}\)Note that the upper bound of the parameter space for design 5 is similar to the one we would have obtained by choosing \( m \) in design 1 to match the average degree of designs 5a to 5c.
<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design 1</td>
<td>sample size $(n)$, number of neighbors $(2m)$ &lt;br&gt;$ Circular world$ structure, in which each spatial unit has $m$-ahead and $m$-behind neighbors. The spatial weight matrix is symmetric, regular, with an average degree of $2m\rho$.</td>
</tr>
<tr>
<td>Design 2</td>
<td>$(n, m)$ &lt;br&gt;$ Circular world$ structure, but each spatial unit has $(2m - 1)$-ahead and 1-behind neighbors. The spatial weight matrix is regular, not symmetric, and has an average degree of $2m\rho$.</td>
</tr>
<tr>
<td>Design 3</td>
<td>$(n, m)$ and $J$ &lt;br&gt;The parameter $J$ introduces an asymmetry in the number of neighbors of each spatial unit, with few spatial units with near $J$ neighbors and most with 2 neighbors*. The spatial weight matrix is symmetric and it is multiplied by a constant to yield an average degree of $2m\rho$.</td>
</tr>
<tr>
<td>Design 4</td>
<td>$(n, m/2)$, $m$ even &lt;br&gt;Similar to design 1, with each spatial unit having $2m$ neighbors, half with weight $\rho$ and half with $\rho^2$. The spatial weight matrix is also symmetric, regular, but it has an average degree of $(1 + \rho)m\rho$.</td>
</tr>
<tr>
<td>Design 5</td>
<td>sample size $n = 92$ &lt;br&gt;Neighborhood structure based on Rio de Janeiro's municipalities. Contiguity and geodesic distance are used to build three spatial weight matrices with varying average degree and asymmetric number of neighbors.</td>
</tr>
</tbody>
</table>

*This comment is not precise on purpose, as the exact distribution of the number of neighbors depend on both $J$ and the sample size $n$.

Table 2.2: Spatial weight matrices - proposed designs
2.7.1 Size Properties of the LUMP Test

In this section we compare the accuracy of the proposed approximations for the distribution of the statistic $T_n$ of the locally uniformly most powerful (LUMP) test for testing the presence of spatial dependence in the SAR process ($H_0 : \rho = 0$ vs. $H_1 : \rho > 0$). Table 2.3 shows some descriptive statistics for $T_n$ for two sample sizes ($n = 30$ and $n = 92$) for the designs 1 to 5 described in table 2.2. It is apparent that for all the designs and for the chosen sample sizes the statistic $T_n$ is not symmetric (positive skewness) and is leptokurtic (positive kurtosis)\textsuperscript{21}, i.e., both values near the mean and extreme values are more likely than in the Gaussian distribution. The table also shows that both skewness and kurtosis decrease as the sample size increases, which is the expected behavior as the statistic $T_n$ has a limiting Gaussian distribution (skewness and kurtosis are zero). Note that the values for the parameters $m$ and $J$ in designs 1 to 4 were chosen to yield a binary spatial weight matrix matching the average number of neighbors of the neighborhood structure based on Rio de Janeiro's municipalities (designs 5a to 5c). This choice also yielded third and fourth invariants for designs 1 to 3 that were close to design 5. Design 4 was not considered in this comparison as for the LUMP test its equivalent to design 1 with half the number of neighbors.

In tables 2.4 to 2.17 we report the standardized critical values and the actual size of the test for the different designs and sample sizes. The critical values were obtained from (i) Monte Carlo simulation with 49999 simulated data sets (the true critical value), (ii) numerically inverting the Edgeworth approximation ($\mathcal{E}_4$, see equation 2.4.5), (iii) using the Cornish-Fisher expansion for the quantile function (equation 2.5.12), (iv) numerically inverting the saddlepoint integral (equation 2.4.19), (v) calculating the inverse saddlepoint (lemma 2.4.5), (vi) numerically inverting Robinson's approximation (equation 2.5.21), (vii) numerically inverting Lugannani-Rice's approximation (equation 2.5.22), and (viii) numerically inverting Barndorff-Nielsen / Jensen's approximation (equation 2.5.23). The size reported in the tables is the true size for the Monte Carlo simulation, and the actual minus nominal size for the approximations. A negative entry means the Type I error is smaller than specified by the nominal size of the test (i.e., the test under rejects the null hypothesis). The actual size of the test is calculated by plugging the calculated critical values into the true distribution function of the $T_n$ statistic, which is taken (as a proxy) as the empirical cumulative distribution obtained from the Monte Carlo simulations.

No hard conclusion can be taken from this small experiment, but some regularities were evident and deserve some comments:

- The saddlepoint inverse approximation for the critical value (see lemma 2.4.5) performed poorly for the majority of the cases, yielding it not usable for practical purposes.

- The Barndorff-Nielsen / Jensen's approximation (BNJA) performed, on average, better than the other approximations:
  - For testing the presence of spatial dependence at level $\alpha = 5\%$, the median absolute error ranged from 0.12 percentage points (Barndorff-Nielsen / Jensen's approximation) to 0.58pp (Robinson's approximation). Comparing the third-quartile, the BNJA also performed better, with an absolute error of 0.19 percentage points, while the Edgeworth approximation fared worst with an absolute error of 0.92pp.
  - For $\alpha = 10\%$, the median absolute error ranged from 0.16pp (BNJA) to 1.10pp (RA). Comparing the third-quartile, BNJA again performed better, with an absolute error of 0.40pp, and the Edgeworth approximation was the worst with an absolute error of 2.32pp.

\textsuperscript{21}Skewness and kurtosis are the third and fourth invariants, represented in this text by $\kappa_3$ and $\kappa_4$ in the Edgeworth and saddlepoint expansions.
• The absolute error for Edgeworth and saddlepoint approximations are positively correlated to skewness and kurtosis, which is not unexpected since the approximations for the distribution of the statistic $T_n$ would likely be worse the farther away from the Gaussian distribution it is.

• The unexpected behavior happens with the Lugannani-Rice's and Barndorff-Nielsen / Jensen's approximation, for which the absolute error bear no linear (or quadratic) relationship to skewness and kurtosis. Therefore these methods are apparently more robust to calculate the critical values for densely populated and asymmetric spatial weight matrices. One example is given by design 2 with $n = 30$ and $m = 9$ (see tables 2.13 and 2.16), where it is clear that Edgeworth and saddlepoint approximations worked poorly when compared to LRA and BNJA.

• Using the asymptotic critical value for the LUMP test ($1.28$ for $\alpha = 10\%$, $1.64$ for $\alpha = 5\%$) seems appropriate when the significance level is $10\%$, the sample size is $n = 92$, and the spatial weight matrix is regular. For smaller sample size and for irregular design matrices the asymptotic value is far from the true value. Also, for the significance level of $5\%$, the asymptotic critical value was far away from the true even for well-behaved spatial weight matrices and sample size of 92. Since a sample size around 100 and a significance level of $5\%$ is standard in spatial econometrics, this highlights the importance of calculating a more accurate critical value for an appropriate inference.

2.7.2 Size Properties of the LUMPI Test

In this section we perform the same analysis than in section 2.7.1 but with the statistic $V_n$ of the locally uniformly most powerful invariant test (LUMPI) for testing $H_0 : \rho = 0$ vs. $H_1 : \rho > 0$. Table 2.18 shows descriptive statistics for $V_n$, and as in the case for $T_n$ in all designs and sample sizes the density function of $V_n$ is not symmetric (positive skewness) and has positive kurtosis, although, in general, both skewness and kurtosis were smaller for $V_n$ than for $T_n$. Note however, that the cumulants and invariants of $V_n$ were obtained empirically, as no formula for them is available. However, the smaller skewness and kurtosis for $V_n$ persist even if we compare them to their empirical counterparts for $T_n$.

Tables 2.19 to 2.32 report the standardized critical values and the actual size of the LUMPI test for the different designs and sample sizes. The critical values were obtained from (i) Monte Carlo simulation with 49999 simulated data sets (referred as the true critical value), (ii) numerically inverting the Edgeworth approximation ($E_4$, see equation 2.4.5), (iii) using the critical value proposed in theorem 2.6.5 (Symmetric CV), (iv) numerically inverting the saddlepoint integral (equation 2.4.16), (v) numerically inverting Robinson's approximation (equation 2.5.21), (vi) numerically inverting Lugannani-Rice's approximation (equation 2.5.22), and (vii) numerically inverting Barndorff-Nielsen / Jensen's approximation (equation 2.5.23). The size column in the tables reports the true size for the Monte Carlo simulation and the actual minus nominal size for the approximations. Negative values in this column means the Type I error is smaller than specified by the nominal size of the test. The actual size of the test is calculated using the empirical cumulative distribution of $V_n$ as a proxy for the true unknown distribution.

Some comments on the results:

• The Barndorff-Nielsen / Jensen approximation again performed better, on average:
  - At level $\alpha = 5\%$, the median absolute error ranged from 0.12 percentage points (for BNJA) to 0.64pp (Robinson's approximation). Comparing the third quartile of the absolute error we obtain 0.17pp for BNJA and 0.89pp for Robinson's approximation.

---

22 We could have used Laplace's approximation to the moments of a ratio of quadratic forms (see Lieberman, 1994) instead of the empirical moments, but this approximation did not perform well for $n = 30$. 

- For $\alpha = 10\%$, the median absolute error ranged from 0.25pp to 1.26pp and the best and worst performers were again BNJA and RA, respectively. Comparing the third quartile reveals the same picture: an absolute error of 0.27pp for BNJA and of 1.87pp for RA.

- Edgeworth approximation for the statistic $V_n$ had smaller absolute error than for $T_n$ for almost all designs and sample sizes when the nominal level of the test was $\alpha = 10\%$. For $\alpha = 5\%$, the approximation for $V_n$ performed better than for $T_n$ mainly for the designs that were too far from normality (e.g., design 2 with $n = 30$ and $m = 9$). This likely reflects the fact that skewness and kurtosis for the statistic $V_n$ is smaller than for $T_n$.

- Saddlepoint approximation also performed better for statistic $V_n$ than for $T_n$, but the reduction in the absolute error was smaller than the observed for the Edgeworth approximation.

- For Robinson, Lugannani-Rice, and Barndorff-Nielsen / Jensen approximations there was no evident improvement or deterioration in the computed absolute errors comparing statistics $V_n$ and $T_n$.

Another interesting aspect to investigate is the minimum sample size after which the critical value for the LUMPI test would be close to the asymptotic critical value obtained from the Gaussian distribution. Figure 2.6 shows that, depending on the neighborhood structure, the convergence to the asymptotic critical value can indeed be slow. It plots the critical value obtained from Edgeworth approximation (a proxy for the true unknown critical value) for designs 1 to 3 as the sample size increases. Designs 1 to 3 were calibrated to yield an average number of neighbors of 18, similar to design 5c (figure 2.6a), and to yield an average number of neighbors of 4, similar to design 5a (figure 2.6b). The figures show that even for large samples (500 spatial units) the true critical value can still be materially different from the asymptotic one. Therefore, using the asymptotic critical value can result in a type I error significantly different that the one established by the nominal level of the test.

![Graph](image-url)

(a) Larger average number of neighbors  
(b) Smaller average number of neighbors

Figure 2.6: Approximate critical value (using Edgeworth) for different sample sizes

The relationship among critical values and average degree or average number of neighbors is explored in figures 2.7 and 2.8. While for designs 1 and 2 the average number of neighbors and the average degree are linked to the same parameter ($m$), design 3 allows for an interplay between both. A fixed value for $m$ sets the average degree and a fixed value for $J$ sets the average number of neighbors. For designs 1 and 2 the critical value is an increasing function of

---

23 The simulated critical values, obtained from 49999 simulated sets, are also shown in the figure for $n = 510$. 

the average degree, therefore the larger \( m \), the larger the difference between the actual size of the test compared to the size of the asymptotic test. For design 3, this relationship does not hold (the critical value remains constant if we multiply the spatial weight matrix by a scalar). This suggest that the average number of neighbors is more relevant for the properties of the test than the average degree of the spatial weight matrix. Indeed, design 3 has a constant average number of neighbors and its increasing average degree comes from larger weights for the non-zero elements in the spatial matrix. On the other hand, for design 1 and 2 the increasing average degree comes from an increasing number of neighbors.

From figure 2.7 we could conjecture that a larger average number of neighbors yields a higher critical value, regardless of the average degree. However, figure 2.8 reveals that the distribution of the number of neighbors might also be an important factor to consider in the analysis. The higher the parameter \( J \) (keeping \( m \) constant) the higher both the average number of neighbors and the dispersion of the distribution of the number of neighbors\(^{24}\), and the graph illustrates that the critical value is not a monotone function of the average number of neighbors anymore.

\(^{24}\) The dispersion measured by the interquartile range, for example, is zero for \( J = 1, \ldots, 11 \), then increases in unitary steps to 23 (when \( J = 34 \)) and stabilizes at 23 thereafter.
This simple exercise highlights the importance of obtaining a more accurate critical value for the test in order to have proper control of its size in small to moderate sample sizes. It was shown that the specific design of the neighborhood structure can significantly change the actual size of the LUMPI test if one is using the asymptotic critical value, even for moderate sample sizes, and no clear direction for the size's bias can be obtained simply from the underlying neighborhood structure.
Table 2.3: Skewness and kurtosis of the $T_n$ statistic ($\rho_1 = 0$)

<table>
<thead>
<tr>
<th>Design</th>
<th>Variance/n</th>
<th>Skewness</th>
<th>Kurtosis</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$m = 2$</td>
<td>32.00</td>
<td>0.39</td>
</tr>
<tr>
<td></td>
<td>$m = 5$</td>
<td>80.00</td>
<td>0.98</td>
</tr>
<tr>
<td>$n = 30$</td>
<td>$m = 9$</td>
<td>144.00</td>
<td>1.46</td>
</tr>
<tr>
<td></td>
<td>$m = 2$</td>
<td>32.00</td>
<td>0.22</td>
</tr>
<tr>
<td>$n = 92$</td>
<td>$m = 5$</td>
<td>80.00</td>
<td>0.56</td>
</tr>
<tr>
<td></td>
<td>$m = 9$</td>
<td>144.00</td>
<td>0.83</td>
</tr>
<tr>
<td></td>
<td>$m = 2$</td>
<td>24.00</td>
<td>0.60</td>
</tr>
<tr>
<td>$n = 30$</td>
<td>$m = 5$</td>
<td>48.00</td>
<td>1.40</td>
</tr>
<tr>
<td></td>
<td>$m = 9$</td>
<td>100.00</td>
<td>2.26</td>
</tr>
<tr>
<td>$n = 92$</td>
<td>$m = 5$</td>
<td>48.00</td>
<td>0.80</td>
</tr>
<tr>
<td></td>
<td>$m = 9$</td>
<td>80.00</td>
<td>1.18</td>
</tr>
<tr>
<td>$J = 9$</td>
<td></td>
<td>38.93</td>
<td>0.67</td>
</tr>
<tr>
<td>$J = 15$</td>
<td></td>
<td>78.93</td>
<td>0.69</td>
</tr>
<tr>
<td>$J = 22$</td>
<td></td>
<td>149.87</td>
<td>1.28</td>
</tr>
<tr>
<td>$J = 16$</td>
<td></td>
<td>39.30</td>
<td>0.42</td>
</tr>
<tr>
<td>$J = 27$</td>
<td></td>
<td>81.39</td>
<td>0.41</td>
</tr>
<tr>
<td>$J = 38$</td>
<td></td>
<td>144.52</td>
<td>0.35</td>
</tr>
<tr>
<td>Design 5</td>
<td>$n = 92$</td>
<td>a</td>
<td>39.13</td>
</tr>
<tr>
<td></td>
<td>b</td>
<td>81.74</td>
<td>0.64</td>
</tr>
<tr>
<td></td>
<td>c</td>
<td>144.52</td>
<td>0.89</td>
</tr>
</tbody>
</table>

Table 2.4: Design 1 - critical levels and actual sizes ($n = 92, \alpha = 10\%$)

<table>
<thead>
<tr>
<th>Design 1 - statistic $T_n$</th>
<th>$m = 2$</th>
<th>$m = 5$</th>
<th>$m = 9$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$c_\alpha$</td>
<td>size</td>
<td>$c_\alpha$</td>
</tr>
<tr>
<td>Monte Carlo</td>
<td>1.28</td>
<td>10.00</td>
<td>1.31</td>
</tr>
<tr>
<td>Edgeworth</td>
<td>1.29</td>
<td>0.11</td>
<td>1.30</td>
</tr>
<tr>
<td>Cornish-Fisher</td>
<td>1.29</td>
<td>0.09</td>
<td>1.30</td>
</tr>
<tr>
<td>Saddlepoint</td>
<td>1.30</td>
<td>-0.02</td>
<td>1.32</td>
</tr>
<tr>
<td>Saddlepoint inversion</td>
<td>1.29</td>
<td>0.08</td>
<td>1.30</td>
</tr>
<tr>
<td>Robinson</td>
<td>1.28</td>
<td>0.17</td>
<td>1.29</td>
</tr>
<tr>
<td>Lugannani-Rice</td>
<td>1.29</td>
<td>0.08</td>
<td>1.30</td>
</tr>
<tr>
<td>Barndorff-Nielsen / Jensen</td>
<td>1.29</td>
<td>0.08</td>
<td>1.30</td>
</tr>
</tbody>
</table>
Table 2.5: Design 2 - critical levels and actual sizes \((n = 92, \alpha = 10\%)\)

<table>
<thead>
<tr>
<th>Design 2 - statistic (T_n)</th>
<th>(m = 2)</th>
<th>(m = 5)</th>
<th>(m = 9)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(c_\alpha)</td>
<td>size</td>
<td>(c_\alpha)</td>
</tr>
<tr>
<td>Monte Carlo</td>
<td>1.30</td>
<td>10.00</td>
<td>1.30</td>
</tr>
<tr>
<td>Edgeworth</td>
<td>1.29</td>
<td>0.20</td>
<td>1.29</td>
</tr>
<tr>
<td>Cornish-Fisher</td>
<td>1.30</td>
<td>0.17</td>
<td>1.31</td>
</tr>
<tr>
<td>Saddlepoint</td>
<td>1.31</td>
<td>-0.01</td>
<td>1.33</td>
</tr>
<tr>
<td>Saddlepoint inversion</td>
<td>1.30</td>
<td>0.17</td>
<td>1.25</td>
</tr>
<tr>
<td>Robinson</td>
<td>1.29</td>
<td>0.27</td>
<td>1.29</td>
</tr>
<tr>
<td>Lugannami-Rice</td>
<td>1.30</td>
<td>0.16</td>
<td>1.31</td>
</tr>
<tr>
<td>Barndoff-Nielsen / Jensen</td>
<td>1.30</td>
<td>0.16</td>
<td>1.31</td>
</tr>
</tbody>
</table>

Table 2.6: Design 3 - critical levels and actual sizes \((n = 92, \alpha = 10\%)\)

<table>
<thead>
<tr>
<th>Design 3 - statistic (T_n)</th>
<th>(J = 16)</th>
<th>(J = 27)</th>
<th>(J = 38)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(c_\alpha)</td>
<td>size</td>
<td>(c_\alpha)</td>
</tr>
<tr>
<td>Monte Carlo</td>
<td>1.22</td>
<td>10.00</td>
<td>1.18</td>
</tr>
<tr>
<td>Edgeworth</td>
<td>1.17</td>
<td>0.91</td>
<td>1.04</td>
</tr>
<tr>
<td>Cornish-Fisher</td>
<td>1.21</td>
<td>0.24</td>
<td>1.13</td>
</tr>
<tr>
<td>Saddlepoint</td>
<td>1.26</td>
<td>-0.54</td>
<td>1.30</td>
</tr>
<tr>
<td>Saddlepoint inversion</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Robinson</td>
<td>1.10</td>
<td>2.02</td>
<td>1.06</td>
</tr>
<tr>
<td>Lugannami-Rice</td>
<td>1.18</td>
<td>0.78</td>
<td>1.15</td>
</tr>
<tr>
<td>Barndoff-Nielsen / Jensen</td>
<td>1.18</td>
<td>0.73</td>
<td>1.16</td>
</tr>
</tbody>
</table>

Table 2.7: Design 5 - critical levels and actual sizes \((n = 92, \alpha = 10\%)\)

<table>
<thead>
<tr>
<th>Design 5 - statistic (T_n)</th>
<th>(5a)</th>
<th>(5b)</th>
<th>(5c)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(c_\alpha)</td>
<td>size</td>
<td>(c_\alpha)</td>
</tr>
<tr>
<td>Monte Carlo</td>
<td>1.29</td>
<td>10.00</td>
<td>1.29</td>
</tr>
<tr>
<td>Edgeworth</td>
<td>1.28</td>
<td>-0.03</td>
<td>1.27</td>
</tr>
<tr>
<td>Cornish-Fisher</td>
<td>1.29</td>
<td>-0.06</td>
<td>1.29</td>
</tr>
<tr>
<td>Saddlepoint</td>
<td>1.29</td>
<td>-0.14</td>
<td>1.27</td>
</tr>
<tr>
<td>Saddlepoint inversion</td>
<td>1.28</td>
<td>-0.03</td>
<td>-</td>
</tr>
<tr>
<td>Robinson</td>
<td>1.27</td>
<td>0.13</td>
<td>1.22</td>
</tr>
<tr>
<td>Lugannami-Rice</td>
<td>1.28</td>
<td>-0.04</td>
<td>1.26</td>
</tr>
<tr>
<td>Barndoff-Nielsen / Jensen</td>
<td>1.28</td>
<td>-0.04</td>
<td>1.26</td>
</tr>
</tbody>
</table>
Table 2.8: *Design 1* - critical levels and actual sizes ($n = 92, \alpha = 5\%$)

<table>
<thead>
<tr>
<th>Design 1 - statistic $T_n$</th>
<th>$m = 2$</th>
<th>$m = 5$</th>
<th>$m = 9$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$c_\alpha$</td>
<td>size</td>
<td>$c_\alpha$</td>
</tr>
<tr>
<td>Monte Carlo</td>
<td>1.70</td>
<td>5.00</td>
<td>1.77</td>
</tr>
<tr>
<td>Edgeworth</td>
<td>1.70</td>
<td>0.09</td>
<td>1.79</td>
</tr>
<tr>
<td>Cornish-Fisher</td>
<td>1.70</td>
<td>0.11</td>
<td>1.78</td>
</tr>
<tr>
<td>Saddlepoint</td>
<td>1.71</td>
<td>0.06</td>
<td>1.80</td>
</tr>
<tr>
<td>Saddlepoint inversion</td>
<td>1.70</td>
<td>0.12</td>
<td>1.73</td>
</tr>
<tr>
<td>Robinson</td>
<td>1.69</td>
<td>0.20</td>
<td>1.76</td>
</tr>
<tr>
<td>Lugannani-Rice</td>
<td>1.70</td>
<td>0.12</td>
<td>1.78</td>
</tr>
<tr>
<td>Barndorff-Nielsen / Jensen</td>
<td>1.70</td>
<td>0.12</td>
<td>1.78</td>
</tr>
</tbody>
</table>

Table 2.9: *Design 2* - critical levels and actual sizes ($n = 92, \alpha = 5\%$)

<table>
<thead>
<tr>
<th>Design 2 - statistic $T_n$</th>
<th>$m = 2$</th>
<th>$m = 5$</th>
<th>$m = 9$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$c_\alpha$</td>
<td>size</td>
<td>$c_\alpha$</td>
</tr>
<tr>
<td>Monte Carlo</td>
<td>1.74</td>
<td>5.00</td>
<td>1.81</td>
</tr>
<tr>
<td>Edgeworth</td>
<td>1.74</td>
<td>0.15</td>
<td>1.84</td>
</tr>
<tr>
<td>Cornish-Fisher</td>
<td>1.73</td>
<td>0.18</td>
<td>1.83</td>
</tr>
<tr>
<td>Saddlepoint</td>
<td>1.74</td>
<td>0.10</td>
<td>1.85</td>
</tr>
<tr>
<td>Saddlepoint inversion</td>
<td>1.73</td>
<td>0.20</td>
<td>-</td>
</tr>
<tr>
<td>Robinson</td>
<td>1.72</td>
<td>0.26</td>
<td>1.80</td>
</tr>
<tr>
<td>Lugannani-Rice</td>
<td>1.73</td>
<td>0.20</td>
<td>1.82</td>
</tr>
<tr>
<td>Barndorff-Nielsen / Jensen</td>
<td>1.73</td>
<td>0.19</td>
<td>1.82</td>
</tr>
</tbody>
</table>

Table 2.10: *Design 3* - critical levels and actual sizes ($n = 92, \alpha = 5\%$)

<table>
<thead>
<tr>
<th>Design 3 - statistic $T_n$</th>
<th>$J = 16$</th>
<th>$J = 27$</th>
<th>$J = 38$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$c_\alpha$</td>
<td>size</td>
<td>$c_\alpha$</td>
</tr>
<tr>
<td>Monte Carlo</td>
<td>1.70</td>
<td>5.00</td>
<td>1.67</td>
</tr>
<tr>
<td>Edgeworth</td>
<td>1.77</td>
<td>-0.41</td>
<td>1.79</td>
</tr>
<tr>
<td>Cornish-Fisher</td>
<td>1.73</td>
<td>-0.12</td>
<td>1.70</td>
</tr>
<tr>
<td>Saddlepoint</td>
<td>1.76</td>
<td>-0.36</td>
<td>1.83</td>
</tr>
<tr>
<td>Saddlepoint inversion</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Robinson</td>
<td>1.57</td>
<td>1.20</td>
<td>1.57</td>
</tr>
<tr>
<td>Lugannani-Rice</td>
<td>1.65</td>
<td>0.51</td>
<td>1.66</td>
</tr>
<tr>
<td>Barndorff-Nielsen / Jensen</td>
<td>1.66</td>
<td>0.45</td>
<td>1.67</td>
</tr>
</tbody>
</table>
Table 2.11: Design 5 - critical levels and actual sizes (n = 92, $\alpha = 5\%$)

<table>
<thead>
<tr>
<th>Design 5 - statistic $T_n$</th>
<th>5a $c_\alpha$ size</th>
<th>5b $c_\alpha$ size</th>
<th>5c $c_\alpha$ size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monte Carlo</td>
<td>1.70 5.00</td>
<td>1.77 5.00</td>
<td>1.82 5.00</td>
</tr>
<tr>
<td>Edgeworth</td>
<td>1.71 -0.14</td>
<td>1.81 -0.29</td>
<td>1.88 -0.36</td>
</tr>
<tr>
<td>Cornish-Fisher</td>
<td>1.71 -0.12</td>
<td>1.79 -0.16</td>
<td>1.85 -0.18</td>
</tr>
<tr>
<td>Saddlepoint</td>
<td>1.71 -0.13</td>
<td>1.76 0.12</td>
<td>1.83 -0.10</td>
</tr>
<tr>
<td>Saddlepoint inversion</td>
<td>1.70 -0.04</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Robinson</td>
<td>1.69 0.03</td>
<td>1.68 0.71</td>
<td>1.73 0.63</td>
</tr>
<tr>
<td>Lugannani-Rice</td>
<td>1.70 -0.06</td>
<td>1.73 0.29</td>
<td>1.78 0.25</td>
</tr>
<tr>
<td>Barndorff-Nielsen / Jensen</td>
<td>1.70 -0.06</td>
<td>1.74 0.26</td>
<td>1.79 0.22</td>
</tr>
</tbody>
</table>

Table 2.12: Design 1 - critical levels and actual sizes (n = 30, $\alpha = 10\%$)

<table>
<thead>
<tr>
<th>Design 1 - statistic $T_n$</th>
<th>$m = 2$ $c_\alpha$ size</th>
<th>$m = 5$ $c_\alpha$ size</th>
<th>$m = 9$ $c_\alpha$ size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monte Carlo</td>
<td>1.26 10.00</td>
<td>1.28 10.00</td>
<td>1.25 10.00</td>
</tr>
<tr>
<td>Edgeworth</td>
<td>1.26 0.27</td>
<td>1.21 0.97</td>
<td>1.06 2.57</td>
</tr>
<tr>
<td>Cornish-Fisher</td>
<td>1.27 0.11</td>
<td>1.27 0.23</td>
<td>1.23 0.26</td>
</tr>
<tr>
<td>Saddlepoint</td>
<td>1.30 -0.41</td>
<td>1.34 -0.61</td>
<td>1.36 -1.19</td>
</tr>
<tr>
<td>Saddlepoint inversion</td>
<td>1.26 0.23</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Robinson</td>
<td>1.25 0.35</td>
<td>1.22 0.82</td>
<td>1.10 1.87</td>
</tr>
<tr>
<td>Lugannani-Rice</td>
<td>1.27 0.02</td>
<td>1.27 0.26</td>
<td>1.21 0.56</td>
</tr>
<tr>
<td>Barndorff-Nielsen / Jensen</td>
<td>1.27 0.01</td>
<td>1.27 0.23</td>
<td>1.22 0.40</td>
</tr>
</tbody>
</table>

Table 2.13: Design 2 - critical levels and actual sizes (n = 30, $\alpha = 10\%$)

<table>
<thead>
<tr>
<th>Design 2 - statistic $T_n$</th>
<th>$m = 2$ $c_\alpha$ size</th>
<th>$m = 5$ $c_\alpha$ size</th>
<th>$m = 9$ $c_\alpha$ size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monte Carlo</td>
<td>1.28 10.00</td>
<td>1.27 10.00</td>
<td>1.20 10.00</td>
</tr>
<tr>
<td>Edgeworth</td>
<td>1.26 0.50</td>
<td>1.11 2.03</td>
<td>0.89 3.83</td>
</tr>
<tr>
<td>Cornish-Fisher</td>
<td>1.28 0.20</td>
<td>1.25 0.29</td>
<td>1.19 0.29</td>
</tr>
<tr>
<td>Saddlepoint</td>
<td>1.32 -0.40</td>
<td>1.34 -0.72</td>
<td>1.43 -1.84</td>
</tr>
<tr>
<td>Saddlepoint inversion</td>
<td>1.23 0.91</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Robinson</td>
<td>1.26 0.55</td>
<td>1.13 1.76</td>
<td>1.06 1.60</td>
</tr>
<tr>
<td>Lugannani-Rice</td>
<td>1.28 0.17</td>
<td>1.22 0.66</td>
<td>1.19 0.22</td>
</tr>
<tr>
<td>Barndorff-Nielsen / Jensen</td>
<td>1.28 0.16</td>
<td>1.23 0.55</td>
<td>1.22 -0.04</td>
</tr>
</tbody>
</table>
Table 2.14: Design 3 - critical levels and actual sizes ($n = 30, \alpha = 10\%$)

<table>
<thead>
<tr>
<th>Design 3 - statistic $T_n$</th>
<th>$J = 9$</th>
<th>$J = 15$</th>
<th>$J = 22$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$c_\alpha$</td>
<td>size</td>
<td>$c_\alpha$</td>
</tr>
<tr>
<td>Monte Carlo</td>
<td>1.22</td>
<td>10.00</td>
<td>1.18</td>
</tr>
<tr>
<td>Edgeworth</td>
<td>1.14</td>
<td>1.05</td>
<td>1.03</td>
</tr>
<tr>
<td>Cornish-Fisher</td>
<td>1.21</td>
<td>0.16</td>
<td>1.15</td>
</tr>
<tr>
<td>Saddlepoint</td>
<td>1.29</td>
<td>-0.90</td>
<td>1.33</td>
</tr>
<tr>
<td>Saddlepoint inversion</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Robinson</td>
<td>1.10</td>
<td>1.73</td>
<td>1.07</td>
</tr>
<tr>
<td>Lugannani-Rice</td>
<td>1.18</td>
<td>0.49</td>
<td>1.17</td>
</tr>
<tr>
<td>Barndorff-Nielsen / Jensen</td>
<td>1.19</td>
<td>0.43</td>
<td>1.17</td>
</tr>
</tbody>
</table>

Table 2.15: Design 1 - critical levels and actual sizes ($n = 30, \alpha = 5\%$)

<table>
<thead>
<tr>
<th>Design 1 - statistic $T_n$</th>
<th>$m = 2$</th>
<th>$m = 5$</th>
<th>$m = 9$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$c_\alpha$</td>
<td>size</td>
<td>$c_\alpha$</td>
</tr>
<tr>
<td>Monte Carlo</td>
<td>1.74</td>
<td>5.00</td>
<td>1.84</td>
</tr>
<tr>
<td>Edgeworth</td>
<td>1.75</td>
<td>0.01</td>
<td>1.92</td>
</tr>
<tr>
<td>Cornish-Fisher</td>
<td>1.73</td>
<td>0.15</td>
<td>1.86</td>
</tr>
<tr>
<td>Saddlepoint</td>
<td>1.76</td>
<td>-0.11</td>
<td>1.90</td>
</tr>
<tr>
<td>Saddlepoint inversion</td>
<td>1.59</td>
<td>1.39</td>
<td>-</td>
</tr>
<tr>
<td>Robinson</td>
<td>1.71</td>
<td>0.35</td>
<td>1.78</td>
</tr>
<tr>
<td>Lugannani-Rice</td>
<td>1.73</td>
<td>0.19</td>
<td>1.82</td>
</tr>
<tr>
<td>Barndorff-Nielsen / Jensen</td>
<td>1.73</td>
<td>0.19</td>
<td>1.83</td>
</tr>
</tbody>
</table>

Table 2.16: Design 2 - critical levels and actual sizes ($n = 30, \alpha = 5\%$)

<table>
<thead>
<tr>
<th>Design 2 - statistic $T_n$</th>
<th>$m = 2$</th>
<th>$m = 5$</th>
<th>$m = 9$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$c_\alpha$</td>
<td>size</td>
<td>$c_\alpha$</td>
</tr>
<tr>
<td>Monte Carlo</td>
<td>1.79</td>
<td>5.00</td>
<td>1.85</td>
</tr>
<tr>
<td>Edgeworth</td>
<td>1.81</td>
<td>0.04</td>
<td>2.16</td>
</tr>
<tr>
<td>Cornish-Fisher</td>
<td>1.78</td>
<td>0.21</td>
<td>1.92</td>
</tr>
<tr>
<td>Saddlepoint</td>
<td>1.82</td>
<td>-0.04</td>
<td>1.97</td>
</tr>
<tr>
<td>Saddlepoint inversion</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Robinson</td>
<td>1.75</td>
<td>0.47</td>
<td>1.74</td>
</tr>
<tr>
<td>Lugannani-Rice</td>
<td>1.77</td>
<td>0.31</td>
<td>1.83</td>
</tr>
<tr>
<td>Barndorff-Nielsen / Jensen</td>
<td>1.77</td>
<td>0.29</td>
<td>1.84</td>
</tr>
</tbody>
</table>
Table 2.17: Design 3 - critical levels and actual sizes \((n = 30, \alpha = 5\%)\)

<table>
<thead>
<tr>
<th>Design 3 - statistic ( T_n )</th>
<th>( J = 9 )</th>
<th>( J = 15 )</th>
<th>( J = 22 )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( c_\alpha )</td>
<td>size</td>
<td>( c_\alpha )</td>
</tr>
<tr>
<td>Monte Carlo</td>
<td>1.72</td>
<td>5.00</td>
<td>1.72</td>
</tr>
<tr>
<td>Edgeworth</td>
<td>1.88</td>
<td>-0.92</td>
<td>1.98</td>
</tr>
<tr>
<td>Cornish-Fisher</td>
<td>1.78</td>
<td>-0.38</td>
<td>1.77</td>
</tr>
<tr>
<td>Saddlepoint</td>
<td>1.83</td>
<td>-0.69</td>
<td>1.90</td>
</tr>
<tr>
<td>Saddlepoint inversion</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Robinson</td>
<td>1.61</td>
<td>0.82</td>
<td>1.62</td>
</tr>
<tr>
<td>Lugannani-Rice</td>
<td>1.70</td>
<td>0.18</td>
<td>1.71</td>
</tr>
<tr>
<td>Barndorff-Nielsen / Jensen</td>
<td>1.71</td>
<td>0.11</td>
<td>1.72</td>
</tr>
</tbody>
</table>
Table 2.18: Empirical variance, skewness and kurtosis of the $V_n$ statistic ($\rho_1 = 0$)

<table>
<thead>
<tr>
<th>Design 1</th>
<th>Mean</th>
<th>Variance</th>
<th>Skewness</th>
<th>Kurtosis</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n = 30$</td>
<td>$m = 2$</td>
<td>0.02</td>
<td>30.30</td>
<td>0.35</td>
</tr>
<tr>
<td></td>
<td>$m = 5$</td>
<td>0.07</td>
<td>75.35</td>
<td>0.89</td>
</tr>
<tr>
<td></td>
<td>$m = 9$</td>
<td>0.06</td>
<td>135.89</td>
<td>1.33</td>
</tr>
<tr>
<td></td>
<td>$n = 92$</td>
<td>$m = 2$</td>
<td>0.03</td>
<td>31.56</td>
</tr>
<tr>
<td></td>
<td>$m = 5$</td>
<td>0.06</td>
<td>78.20</td>
<td>0.56</td>
</tr>
<tr>
<td></td>
<td>$m = 9$</td>
<td>0.08</td>
<td>139.56</td>
<td>0.80</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Design 2</th>
<th>Mean</th>
<th>Variance</th>
<th>Skewness</th>
<th>Kurtosis</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n = 30$</td>
<td>$m = 2$</td>
<td>0.01</td>
<td>22.86</td>
<td>0.55</td>
</tr>
<tr>
<td></td>
<td>$m = 5$</td>
<td>0.03</td>
<td>45.39</td>
<td>1.28</td>
</tr>
<tr>
<td></td>
<td>$m = 9$</td>
<td>0.07</td>
<td>95.65</td>
<td>2.05</td>
</tr>
<tr>
<td></td>
<td>$n = 92$</td>
<td>$m = 2$</td>
<td>-0.00</td>
<td>23.90</td>
</tr>
<tr>
<td></td>
<td>$m = 5$</td>
<td>0.03</td>
<td>46.83</td>
<td>0.77</td>
</tr>
<tr>
<td></td>
<td>$m = 9$</td>
<td>0.01</td>
<td>78.27</td>
<td>1.13</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Design 3</th>
<th>Mean</th>
<th>Variance</th>
<th>Skewness</th>
<th>Kurtosis</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n = 30$</td>
<td>$J = 9$</td>
<td>-0.02</td>
<td>36.90</td>
<td>0.63</td>
</tr>
<tr>
<td></td>
<td>$J = 15$</td>
<td>0.01</td>
<td>75.96</td>
<td>0.63</td>
</tr>
<tr>
<td></td>
<td>$J = 22$</td>
<td>0.03</td>
<td>143.49</td>
<td>1.15</td>
</tr>
<tr>
<td></td>
<td>$n = 92$</td>
<td>$J = 16$</td>
<td>-0.01</td>
<td>38.97</td>
</tr>
<tr>
<td></td>
<td>$J = 27$</td>
<td>-0.04</td>
<td>80.56</td>
<td>0.46</td>
</tr>
<tr>
<td></td>
<td>$J = 38$</td>
<td>0.01</td>
<td>143.36</td>
<td>0.42</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Design 5</th>
<th>Mean</th>
<th>Variance</th>
<th>Skewness</th>
<th>Kurtosis</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n = 92$</td>
<td>$a$</td>
<td>-0.00</td>
<td>37.95</td>
<td>0.23</td>
</tr>
<tr>
<td></td>
<td>$b$</td>
<td>0.06</td>
<td>79.73</td>
<td>0.58</td>
</tr>
<tr>
<td></td>
<td>$c$</td>
<td>0.06</td>
<td>140.45</td>
<td>0.83</td>
</tr>
</tbody>
</table>

Table 2.19: Design 1 - critical levels and actual sizes ($n = 92, \alpha = 10\%$)

<table>
<thead>
<tr>
<th>Design 1 - statistic $V_n$</th>
<th>$m = 2$</th>
<th>$m = 5$</th>
<th>$m = 9$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$c_\alpha$</td>
<td>size</td>
<td>$c_\alpha$</td>
</tr>
<tr>
<td>Monte Carlo</td>
<td>1.31</td>
<td>10.00</td>
<td>1.32</td>
</tr>
<tr>
<td>Edgeworth</td>
<td>1.29</td>
<td>0.13</td>
<td>1.31</td>
</tr>
<tr>
<td>Symmetric CV</td>
<td>1.29</td>
<td>0.13</td>
<td>1.31</td>
</tr>
<tr>
<td>Saddlepoint</td>
<td>1.30</td>
<td>0.08</td>
<td>1.32</td>
</tr>
<tr>
<td>Robinson</td>
<td>1.28</td>
<td>0.24</td>
<td>1.30</td>
</tr>
<tr>
<td>Lugannani-Rice</td>
<td>1.29</td>
<td>0.13</td>
<td>1.31</td>
</tr>
<tr>
<td>Barndorff-Nielsen / Jensen</td>
<td>1.29</td>
<td>0.13</td>
<td>1.31</td>
</tr>
</tbody>
</table>
Table 2.20: Design 2 - critical levels and actual sizes ($n = 92, \alpha = 10\%$)

<table>
<thead>
<tr>
<th>Design 2 - statistic $V_n$</th>
<th>$m = 2$</th>
<th></th>
<th></th>
<th>$m = 5$</th>
<th></th>
<th></th>
<th>$m = 9$</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$c_\alpha$</td>
<td>size</td>
<td></td>
<td>$c_\alpha$</td>
<td>size</td>
<td></td>
<td>$c_\alpha$</td>
<td>size</td>
<td></td>
</tr>
<tr>
<td>Monte Carlo</td>
<td>1.32</td>
<td>10.00</td>
<td></td>
<td>1.32</td>
<td>10.00</td>
<td></td>
<td>1.32</td>
<td>10.00</td>
<td></td>
</tr>
<tr>
<td>Edgeworth</td>
<td>1.30</td>
<td>0.21</td>
<td></td>
<td>1.30</td>
<td>0.11</td>
<td></td>
<td>1.26</td>
<td>0.61</td>
<td></td>
</tr>
<tr>
<td>Symmetric CV</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Saddlepoint</td>
<td>1.31</td>
<td>0.11</td>
<td></td>
<td>1.34</td>
<td>-0.35</td>
<td></td>
<td>1.35</td>
<td>-0.47</td>
<td></td>
</tr>
<tr>
<td>Robinson</td>
<td>1.29</td>
<td>0.33</td>
<td></td>
<td>1.29</td>
<td>0.30</td>
<td></td>
<td>1.25</td>
<td>0.66</td>
<td></td>
</tr>
<tr>
<td>Lugannami-Rice</td>
<td>1.30</td>
<td>0.21</td>
<td></td>
<td>1.31</td>
<td>-0.03</td>
<td></td>
<td>1.30</td>
<td>0.11</td>
<td></td>
</tr>
<tr>
<td>Barndorff-Nielsen / Jensen</td>
<td>1.30</td>
<td>0.21</td>
<td></td>
<td>1.31</td>
<td>-0.03</td>
<td></td>
<td>1.30</td>
<td>0.07</td>
<td></td>
</tr>
</tbody>
</table>

Table 2.21: Design 3 - critical levels and actual sizes ($n = 92, \alpha = 10\%$)

<table>
<thead>
<tr>
<th>Design 3 - statistic $V_n$</th>
<th>$J = 16$</th>
<th></th>
<th></th>
<th>$J = 27$</th>
<th></th>
<th></th>
<th>$J = 38$</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$c_\alpha$</td>
<td>size</td>
<td></td>
<td>$c_\alpha$</td>
<td>size</td>
<td></td>
<td>$c_\alpha$</td>
<td>size</td>
<td></td>
</tr>
<tr>
<td>Monte Carlo</td>
<td>1.24</td>
<td>10.00</td>
<td></td>
<td>1.20</td>
<td>10.00</td>
<td></td>
<td>1.18</td>
<td>10.00</td>
<td></td>
</tr>
<tr>
<td>Edgeworth</td>
<td>1.19</td>
<td>0.71</td>
<td></td>
<td>1.06</td>
<td>2.08</td>
<td></td>
<td>0.99</td>
<td>2.81</td>
<td></td>
</tr>
<tr>
<td>Symmetric CV</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Saddlepoint</td>
<td>1.21</td>
<td>0.30</td>
<td></td>
<td>1.13</td>
<td>0.88</td>
<td></td>
<td>1.09</td>
<td>1.28</td>
<td></td>
</tr>
<tr>
<td>Robinson</td>
<td>1.25</td>
<td>-0.30</td>
<td></td>
<td>1.29</td>
<td>-1.31</td>
<td></td>
<td>1.29</td>
<td>-1.59</td>
<td></td>
</tr>
<tr>
<td>Lugannami-Rice</td>
<td>1.10</td>
<td>2.24</td>
<td></td>
<td>1.06</td>
<td>2.05</td>
<td></td>
<td>1.04</td>
<td>1.91</td>
<td></td>
</tr>
<tr>
<td>Barndorff-Nielsen / Jensen</td>
<td>1.18</td>
<td>0.83</td>
<td></td>
<td>1.16</td>
<td>0.53</td>
<td></td>
<td>1.14</td>
<td>0.45</td>
<td></td>
</tr>
</tbody>
</table>

Table 2.22: Design 5 - critical levels and actual sizes ($n = 92, \alpha = 10\%$)

<table>
<thead>
<tr>
<th>Design 5 - statistic $V_n$</th>
<th>$5a$</th>
<th></th>
<th></th>
<th>$5b$</th>
<th></th>
<th></th>
<th>$5c$</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$c_\alpha$</td>
<td>size</td>
<td></td>
<td>$c_\alpha$</td>
<td>size</td>
<td></td>
<td>$c_\alpha$</td>
<td>size</td>
</tr>
<tr>
<td>Monte Carlo</td>
<td>1.30</td>
<td>10.00</td>
<td></td>
<td>1.31</td>
<td>10.00</td>
<td></td>
<td>1.31</td>
<td>10.00</td>
</tr>
<tr>
<td>Edgeworth</td>
<td>1.29</td>
<td>-0.10</td>
<td></td>
<td>1.29</td>
<td>0.22</td>
<td></td>
<td>1.27</td>
<td>0.44</td>
</tr>
<tr>
<td>Symmetric CV</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Saddlepoint</td>
<td>1.29</td>
<td>-0.09</td>
<td></td>
<td>1.27</td>
<td>0.48</td>
<td></td>
<td>1.30</td>
<td>0.01</td>
</tr>
<tr>
<td>Robinson</td>
<td>1.28</td>
<td>0.13</td>
<td></td>
<td>1.22</td>
<td>1.30</td>
<td></td>
<td>1.22</td>
<td>1.26</td>
</tr>
<tr>
<td>Lugannami-Rice</td>
<td>1.29</td>
<td>-0.06</td>
<td></td>
<td>1.26</td>
<td>0.58</td>
<td></td>
<td>1.27</td>
<td>0.46</td>
</tr>
<tr>
<td>Barndorff-Nielsen / Jensen</td>
<td>1.29</td>
<td>-0.07</td>
<td></td>
<td>1.27</td>
<td>0.57</td>
<td></td>
<td>1.27</td>
<td>0.44</td>
</tr>
</tbody>
</table>
Table 2.23: Design 1 - critical levels and actual sizes $(n = 92, \alpha = 5\%)$

<table>
<thead>
<tr>
<th>Design 1 - statistic $V_n$</th>
<th>$m = 2$</th>
<th>$m = 5$</th>
<th>$m = 9$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$c_\alpha$</td>
<td>size</td>
<td>$c_\alpha$</td>
</tr>
<tr>
<td>Monte Carlo</td>
<td>1.70</td>
<td>5.00</td>
<td>1.78</td>
</tr>
<tr>
<td>Edgeworth</td>
<td>1.69</td>
<td>0.06</td>
<td>1.78</td>
</tr>
<tr>
<td>Symmetric CV</td>
<td>1.69</td>
<td>0.05</td>
<td>1.78</td>
</tr>
<tr>
<td>Saddlepoint</td>
<td>1.69</td>
<td>0.06</td>
<td>1.78</td>
</tr>
<tr>
<td>Robinson</td>
<td>1.68</td>
<td>0.13</td>
<td>1.75</td>
</tr>
<tr>
<td>Lugannani-Rice</td>
<td>1.69</td>
<td>0.07</td>
<td>1.77</td>
</tr>
<tr>
<td>Barndorff-Nielsen / Jensen</td>
<td>1.69</td>
<td>0.07</td>
<td>1.77</td>
</tr>
</tbody>
</table>

Table 2.24: Design 2 - critical levels and actual sizes $(n = 92, \alpha = 5\%)$

<table>
<thead>
<tr>
<th>Design 2 - statistic $V_n$</th>
<th>$m = 2$</th>
<th>$m = 5$</th>
<th>$m = 9$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$c_\alpha$</td>
<td>size</td>
<td>$c_\alpha$</td>
</tr>
<tr>
<td>Monte Carlo</td>
<td>1.74</td>
<td>5.00</td>
<td>1.82</td>
</tr>
<tr>
<td>Edgeworth</td>
<td>1.72</td>
<td>0.09</td>
<td>1.84</td>
</tr>
<tr>
<td>Symmetric CV</td>
<td>1.72</td>
<td>0.09</td>
<td>1.84</td>
</tr>
<tr>
<td>Saddlepoint</td>
<td>1.71</td>
<td>0.19</td>
<td>1.79</td>
</tr>
<tr>
<td>Robinson</td>
<td>1.72</td>
<td>0.12</td>
<td>1.81</td>
</tr>
<tr>
<td>Lugannani-Rice</td>
<td>1.72</td>
<td>0.12</td>
<td>1.81</td>
</tr>
<tr>
<td>Barndorff-Nielsen / Jensen</td>
<td>1.72</td>
<td>0.12</td>
<td>1.81</td>
</tr>
</tbody>
</table>

Table 2.25: Design 3 - critical levels and actual sizes $(n = 92, \alpha = 5\%)$

<table>
<thead>
<tr>
<th>Design 3 - statistic $V_n$</th>
<th>$J = 16$</th>
<th>$J = 27$</th>
<th>$J = 38$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$c_\alpha$</td>
<td>size</td>
<td>$c_\alpha$</td>
</tr>
<tr>
<td>Monte Carlo</td>
<td>1.70</td>
<td>5.00</td>
<td>1.68</td>
</tr>
<tr>
<td>Edgeworth</td>
<td>1.73</td>
<td>-0.26</td>
<td>1.69</td>
</tr>
<tr>
<td>Symmetric CV</td>
<td>1.72</td>
<td>-0.20</td>
<td>1.69</td>
</tr>
<tr>
<td>Saddlepoint</td>
<td>1.74</td>
<td>-0.33</td>
<td>1.81</td>
</tr>
<tr>
<td>Robinson</td>
<td>1.56</td>
<td>1.09</td>
<td>1.56</td>
</tr>
<tr>
<td>Lugannani-Rice</td>
<td>1.64</td>
<td>0.38</td>
<td>1.65</td>
</tr>
<tr>
<td>Barndorff-Nielsen / Jensen</td>
<td>1.65</td>
<td>0.34</td>
<td>1.66</td>
</tr>
</tbody>
</table>
Table 2.26: Design 5 - critical levels and actual sizes \((n = 92, \alpha = 5\%)\)

<table>
<thead>
<tr>
<th>Design 5 - statistic (v_n)</th>
<th>(5a) (c_\alpha) size</th>
<th>(5b) (c_\alpha) size</th>
<th>(5c) (c_\alpha) size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monte Carlo</td>
<td>1.71  5.00</td>
<td>1.78  5.00</td>
<td>1.83  5.00</td>
</tr>
<tr>
<td>Edgeworth</td>
<td>1.70 -0.13</td>
<td>1.80 -0.31</td>
<td>1.87 -0.43</td>
</tr>
<tr>
<td>Symmetric CV</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Saddlepoint</td>
<td>1.69 -0.09</td>
<td>1.74  0.17</td>
<td>1.82 -0.03</td>
</tr>
<tr>
<td>Robinson</td>
<td>1.68  0.05</td>
<td>1.67  0.77</td>
<td>1.72  0.70</td>
</tr>
<tr>
<td>Lugannani-Rice</td>
<td>1.69 -0.07</td>
<td>1.72  0.30</td>
<td>1.78  0.27</td>
</tr>
<tr>
<td>Barndorff-Nielsen / Jensen</td>
<td>1.69 -0.07</td>
<td>1.73  0.28</td>
<td>1.78  0.24</td>
</tr>
</tbody>
</table>

Table 2.27: Design 1 - critical levels and actual sizes \((n = 30, \alpha = 10\%)\)

<table>
<thead>
<tr>
<th>Design 1 - statistic (V_n)</th>
<th>(m = 2) (c_\alpha) size</th>
<th>(m = 5) (c_\alpha) size</th>
<th>(m = 9) (c_\alpha) size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monte Carlo</td>
<td>1.33  10.00</td>
<td>1.34  10.00</td>
<td>1.31  10.00</td>
</tr>
<tr>
<td>Edgeworth</td>
<td>1.29  0.23</td>
<td>1.28  0.39</td>
<td>1.15  1.59</td>
</tr>
<tr>
<td>Symmetric CV</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Saddlepoint</td>
<td>1.28  0.28</td>
<td>1.29  0.21</td>
<td>1.24  0.32</td>
</tr>
<tr>
<td>Robinson</td>
<td>1.29  0.19</td>
<td>1.34  -0.38</td>
<td>1.36  -0.98</td>
</tr>
<tr>
<td>Lugannani-Rice</td>
<td>1.29  0.25</td>
<td>1.29  0.20</td>
<td>1.23  0.43</td>
</tr>
<tr>
<td>Barndorff-Nielsen / Jensen</td>
<td>1.29  0.25</td>
<td>1.29  0.18</td>
<td>1.24  0.32</td>
</tr>
</tbody>
</table>

Table 2.28: Design 2 - critical levels and actual sizes \((n = 30, \alpha = 10\%)\)

<table>
<thead>
<tr>
<th>Design 2 - statistic (V_n)</th>
<th>(m = 2) (c_\alpha) size</th>
<th>(m = 5) (c_\alpha) size</th>
<th>(m = 9) (c_\alpha) size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monte Carlo</td>
<td>1.35  10.00</td>
<td>1.32  10.00</td>
<td>1.25  10.00</td>
</tr>
<tr>
<td>Edgeworth</td>
<td>1.30  0.27</td>
<td>1.20  1.13</td>
<td>0.92  3.72</td>
</tr>
<tr>
<td>Symmetric CV</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Saddlepoint</td>
<td>1.31  0.04</td>
<td>1.35  -0.75</td>
<td>1.45  -1.85</td>
</tr>
<tr>
<td>Robinson</td>
<td>1.27  0.68</td>
<td>1.14  1.83</td>
<td>1.10  1.46</td>
</tr>
<tr>
<td>Lugannani-Rice</td>
<td>1.30  0.26</td>
<td>1.24  0.50</td>
<td>1.23  0.01</td>
</tr>
<tr>
<td>Barndorff-Nielsen / Jensen</td>
<td>1.30  0.26</td>
<td>1.25  0.42</td>
<td>1.25  -0.17</td>
</tr>
</tbody>
</table>
Table 2.29: *Design 3* - critical levels and actual sizes ($n = 30, \alpha = 10\%$)

<table>
<thead>
<tr>
<th>Design 3 - statistic $V_n$</th>
<th>$J = 9$</th>
<th>$J = 15$</th>
<th>$J = 22$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$c_\alpha$</td>
<td>size</td>
<td>$c_\alpha$</td>
</tr>
<tr>
<td>Monte Carlo</td>
<td>1.27</td>
<td>10.00</td>
<td>1.24</td>
</tr>
<tr>
<td>Edgeworth</td>
<td>1.21</td>
<td>0.37</td>
<td>1.11</td>
</tr>
<tr>
<td>Symmetric CV</td>
<td>1.22</td>
<td>0.18</td>
<td>1.15</td>
</tr>
<tr>
<td>Saddlepoint</td>
<td>1.27</td>
<td>-0.60</td>
<td>1.31</td>
</tr>
<tr>
<td>Robinson</td>
<td>1.11</td>
<td>1.95</td>
<td>1.08</td>
</tr>
<tr>
<td>Lugannani-Rice</td>
<td>1.20</td>
<td>0.47</td>
<td>1.18</td>
</tr>
<tr>
<td>Barndorff-Nielsen / Jensen</td>
<td>1.20</td>
<td>0.42</td>
<td>1.19</td>
</tr>
</tbody>
</table>

Table 2.30: *Design 1* - critical levels and actual sizes ($n = 30, \alpha = 5\%$)

<table>
<thead>
<tr>
<th>Design 1 - statistic $V_n$</th>
<th>$m = 2$</th>
<th>$m = 5$</th>
<th>$m = 9$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$c_\alpha$</td>
<td>size</td>
<td>$c_\alpha$</td>
</tr>
<tr>
<td>Monte Carlo</td>
<td>1.76</td>
<td>5.00</td>
<td>1.87</td>
</tr>
<tr>
<td>Edgeworth</td>
<td>1.69</td>
<td>0.24</td>
<td>1.85</td>
</tr>
<tr>
<td>Symmetric CV</td>
<td>1.71</td>
<td>0.12</td>
<td>1.86</td>
</tr>
<tr>
<td>Saddlepoint</td>
<td>1.70</td>
<td>0.20</td>
<td>1.84</td>
</tr>
<tr>
<td>Robinson</td>
<td>1.67</td>
<td>0.45</td>
<td>1.75</td>
</tr>
<tr>
<td>Lugannani-Rice</td>
<td>1.70</td>
<td>0.21</td>
<td>1.80</td>
</tr>
<tr>
<td>Barndorff-Nielsen / Jensen</td>
<td>1.70</td>
<td>0.21</td>
<td>1.80</td>
</tr>
</tbody>
</table>

Table 2.31: *Design 2* - critical levels and actual sizes ($n = 30, \alpha = 5\%$)

<table>
<thead>
<tr>
<th>Design 2 - statistic $V_n$</th>
<th>$m = 2$</th>
<th>$m = 5$</th>
<th>$m = 9$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$c_\alpha$</td>
<td>size</td>
<td>$c_\alpha$</td>
</tr>
<tr>
<td>Monte Carlo</td>
<td>1.81</td>
<td>5.00</td>
<td>1.91</td>
</tr>
<tr>
<td>Edgeworth</td>
<td>1.75</td>
<td>0.12</td>
<td>1.97</td>
</tr>
<tr>
<td>Symmetric CV</td>
<td>1.75</td>
<td>0.09</td>
<td>1.93</td>
</tr>
<tr>
<td>Saddlepoint</td>
<td>1.71</td>
<td>0.44</td>
<td>1.72</td>
</tr>
<tr>
<td>Robinson</td>
<td>1.74</td>
<td>0.17</td>
<td>1.82</td>
</tr>
<tr>
<td>Lugannani-Rice</td>
<td>1.74</td>
<td>0.17</td>
<td>1.83</td>
</tr>
<tr>
<td>Barndorff-Nielsen / Jensen</td>
<td>1.74</td>
<td>0.17</td>
<td>1.83</td>
</tr>
</tbody>
</table>
Table 2.32: Design 3 - critical levels and actual sizes ($n = 30, \alpha = 5\%$)

<table>
<thead>
<tr>
<th>Design 3 - statistic $V_n$</th>
<th>$J = 9$</th>
<th>$J = 15$</th>
<th>$J = 22$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$c_\alpha$</td>
<td>size</td>
<td>$c_\alpha$</td>
</tr>
<tr>
<td>Monte Carlo</td>
<td>1.76</td>
<td>5.00</td>
<td>1.76</td>
</tr>
<tr>
<td>Edgeworth</td>
<td>1.74</td>
<td>-0.19</td>
<td>1.71</td>
</tr>
<tr>
<td>Symmetric CV</td>
<td>1.76</td>
<td>-0.37</td>
<td>1.75</td>
</tr>
<tr>
<td>Saddlepoint</td>
<td>1.77</td>
<td>-0.41</td>
<td>1.83</td>
</tr>
<tr>
<td>Robinson</td>
<td>1.59</td>
<td>0.92</td>
<td>1.60</td>
</tr>
<tr>
<td>Lugannani-Rice</td>
<td>1.69</td>
<td>0.17</td>
<td>1.70</td>
</tr>
<tr>
<td>Barndorff-Nielsen / Jensen</td>
<td>1.69</td>
<td>0.13</td>
<td>1.71</td>
</tr>
</tbody>
</table>
2.7.3 Power Envelope Using Edgeworth Series

This section exemplifies the procedure to obtain the power envelope for testing the null hypothesis of no spatial dependence in the SAR stochastic process (see box 2.6.1) using designs 1 to 5 defined in table 2.2. Figure 2.9 illustrates some interesting aspects. First, the Edgeworth approximation for the power envelope for testing the null hypothesis of no spatial dependence in a SAR stochastic process is badly behaved for the alternative hypothesis \( (\rho_1) \) near the bounds of the parameter space \( P = (\rho, \rho) \). Moreover, this problem does not vanish as \( n \to \infty \), even for the simple and well-behaved example of a circular world design (design 1). The reason is that for each sample size \( n \) we can pick \( \rho_1 \) arbitrarily close to \( \rho \) and therefore the eigenvalues of \( L(\rho_1)^{-1} \) can be arbitrarily large, more than offsetting the factor \( n^{-1/2} \) in the invariants formulas under the alternative hypothesis, which results in arbitrarily large invariants and an Edgeworth approximation for the distribution function of \( F_{\rho_1, \rho_0(\rho_1)}(v) \) under \( H_1 \) which is not well behaved even for small values of \( v \). For this reason, the parameter space is usually assumed a closed interval contained in \( (\rho, \rho) \), therefore putting an upper bound in the eigenvalues of \( L^{-1}(\rho_1) \).

Second, for a given degree of connectivity in the spatial weight matrix (represented by the parameters \( m, J \), in these examples) and for reasonable values of \( \rho_1 \in P \setminus \{0\} \), the power function approaches unity with \( n \to \infty \).

![Power envelope using Edgeworth (m=2)](image-url)
Figure 2.10: Edgeworth approximation for $F_{\rho_1, V_n(\rho_1)}$ in a circular world design with $n = 92$ and $m = 2$

The first point highlighted can be appreciated in figure 2.10. There we compare the Edgeworth approximation for $F_{\rho_1, V_n(\rho_1)}$ in a circular world design for two different values of $\rho_1$. The figure plots values for the coordinates in the range of two standard deviations, and it is clear that the truncated Edgeworth series is a poor approximation to the true distribution $F_{\rho_1, V_n(\rho_1)}$ for $\rho_1$ near $\rho$ even for values of $v$ smaller than two standard deviations. However, one needs to note that the values of $\rho_1$ for which the approximation is poor might not be of practical interest. In this simple circular world design example (design 1, with $n = 92, m = 2$), the value $\rho_1 = 0.24$ would imply that under the alternative hypothesis $E(y_i^2) \approx 29\sigma^2$, which looks unreasonable.  

For figure 2.10a the same calculation yields $E(y_i^2) \approx 8.6\sigma^2$.

The question on how the power envelope behaves as the degree of connectivity increases (in these examples, as $m, J$ increases) is a little more trickier. In design 1, for example, for a given value of $\rho_1$, the power function obviously increases as $m$ grows. The reason is that as $m$ increases the parameter space $P = (\rho, \rho)$ shrinks. Thus, for instance, $\rho_1 = 0.1$ represents a value almost halfway through the maximum $\rho = 0.25$ for $m = 2$, but the same value for $\rho_1$ is near the maximum $\rho = 0.125$ when $m = 4$. And the preceding paragraph pointed that the power function increases as $\rho_1 \rightarrow \rho$, except for values too close to the bounds of the parameter space where approximation used is not appropriated.

One alternative is to compare the power functions for a normalized parameter space. One normalization which is implicitly or explicitly widely used in the applied literature\footnote{In design 1, $Y$ is homoskedastic under the alternative hypothesis, and this value of $\rho_1$ would imply the variance of each spatial unit would be more than 29 times larger than the variance of the disturbance term.} is to rescale the parameter space by setting $\rho = 1$. In this case, for instance, a normalized value $\rho_1 = 0.5$ would map into a non normalized $\rho_1 = 0.125$ for the case $m = 2$ and $\rho_1 = 0.05$ for $m = 5$.

With this choice of normalization, however, one would still be inclined to conclude that the power of the test would decrease as $m$ increases (see figure 2.11), which is a counter-intuitive result since one would expect the power against an alternative represented by a spatial weight matrix that is more densely populated to be larger than for an alternative less densely populated and therefore closer to the null hypothesis, at least when comparing within models with the same underlying structure for the spatial weight matrix, as it is the case for design 1 with varying values for $m$.

The underlying reason for this apparently contradictory result is that the effects of $\rho_1 \neq 0$ into the covariance structure of the SAR stochastic process is strongly non-linear. This reflects\footnote{This would be the case, for instance, when the spatial weight matrix is row normalized.}
the fact that the values of $\rho$ being compared are not at the same “statistical” distance from the null hypothesis. In the example of figure 2.11, for instance, a normalized $\rho$ of 0.5 is 4.0, 2.7, and 2.0 away\(^{27}\) from the null ($\rho = 0$) for the models with $m = 2$, $m = 5$, and $m = 9$ respectively. Therefore it is no surprise that the power of the optimum test is larger for the model with $m = 2$ as it is expected that the power will increase with the distance from the null hypothesis.

As just argued, one possibility to assess the relative power performance of different designs for the spatial weight matrix is to compare not the parameter $\rho_1$ but its implications in terms of the covariance of the stochastic process. For example, given a value for $\rho_1$ for the model with $m = 2$, we could calculate which is the maximum variance $\sigma^2_{\text{max}}(\rho_1) \equiv \max \{ E(Y_i^2) \} \text{ under the alternative } H_1$, and then choose values for $\rho_1$ which would yield the same maximum variance for different values of $m$.

A more general way to extend the rational of the previous paragraph to compare different spatial weight designs would be to use a notion of distance to compare, for example, the covariance matrices under the null and under the alternative hypothesis. Several choices of matrix norm could then be used to construct this metric. One example could be the Frobenius norm $\left( \| \Delta(\rho) \|_F = \sqrt{\text{tr} \Delta(\rho)^T \Delta(\rho)} \right)$, where $\Delta(\rho) \equiv L^{-1}(\rho)L^{-1}(\rho)^T - I$ would be proportional to the difference between the covariance matrices of the stochastic SAR process under the alternative ($\rho \neq 0$) and the null ($\rho = 0$) hypothesis. To keep in the statistical context, however, I choose to use the average information distance defined by

$$ id(\rho_0, \rho_1) \equiv \frac{1}{n} \int_{\rho_0}^{\rho_1} \sqrt{I(\rho)} d\rho $$ \hspace{1cm} (2.7.1) 

where

$$ I(\rho) = -E_\theta \frac{\partial^2 \log f(Y, \theta)}{\partial \rho^2} $$

\(^{27}\)Based on the information distance defined in equation 2.7.1.
is the Fisher information. In the current context, the Fisher information would measure the amount of information that (ex-ante) the sample \(Y\) carries about the parameter \(\rho\), and the information distance \(\text{id}(\rho_0, \rho_1)\) would be a measure of how far apart any two members of the parametric family of distributions \(\{F_\theta, \theta \in \mathcal{P}\}\) are (see Kass, 1989).

The Fisher information depends on both \(\rho\) and on the spatial weight matrix. For the designs 1 to 5 (which are summarized in table 2.2) the following holds:

\[
I(\rho) = \text{tr}\left((L_i^{-1}(\rho)W_i(\rho))^2 + (W_i(\rho)L_i^{-1}(\rho))^t(W_i(\rho)L_i^{-1}(\rho))\right)
\]

\[L_i(\rho) = I - w_i(\rho), \quad \text{and}\]

\[W_i(\rho) = \frac{\partial w_i(\rho)}{\partial \rho}, \quad \text{for } i = 1, \ldots, 5\] (2.7.2)

This allows the comparison of the power curves for different designs based not on the value of the parameter \(\rho\) itself, but on the information distance between the alternative and the null hypothesis. For instance, when the sample size is \(n = 30\), the values of \(\rho\) that are at information distance 1 from the origin are 0.21, 0.23, 0.15, and 0.31 for designs 1, 2, 3, and 4, respectively. Figure 2.12 shows that for moderate values of the information distance there is not a significant difference in the maximum power of the test. One pattern that seems to appear is that larger values of the average number of neighbors lead to slightly higher powers when \(\rho\) is positive and slightly lower powers when it is negative. For design 2, with an asymmetric spatial weight matrix, this bias was more pronounced. Figure 2.13 highlights that for moderate values of the information distance the difference of optimum power is small across different designs. For \(\text{id} = 0.17\), for instance, the optimum power for design 1 is 0.50 and the absolute difference of optimum power among designs 1 to 5 reaches a maximum of 2.5 percentage points. For reference, the values of \(\rho\) that are at information distance \(\text{id} = 0.17\) from the origin are 0.06 for designs 1 and 2, 0.5 for designs 3 and 5, and 0.8 for design 4. Disregarding design 3 (which has too high kurtosis to have a good fit by the Edgeworth approximation) it is apparent that the difference of optimum power is small for all values of information distance (therefore for all \(\rho\)), the differences arising probably from the errors of the Edgeworth approximation.

The possibility of comparing the power envelope for the optimum invariant test to detect the presence of spatial dependence in a SAR stochastic process for different designs of the spatial weight matrix might be a valuable tool in the process of checking the robustness of some specification of the neighborhood structure. Even though some authors (see Anselin, 1988, for instance) emphasize that the weight matrix should arise from the underlying theory specifying the dependence relations, in practice it is often based on an ad hoc notion of distance and dependence. Therefore, most empirical applications propose alternative specifications of the spatial weight matrix to check for robustness, and the possibility of building the power envelope for each specification allows the econometrician to assess how the competing designs for \(w(\rho)\) fare regarding the probability of rejecting the hypothesis of no spatial dependence when it is false. It might happen that the alternative designs proposed to check for "robustness" are in fact very similar to the original design.
Figure 2.12: Power envelope against information distance
2.7.4 Power Envelope Using Saddlepoint

The same designs 1 to 5 (see table 2.2) are used to illustrate the procedure to obtain the power envelope for the optimum (invariant) test for the presence of spatial dependence in a SAR stochastic process using saddlepoint approximation. The saddlepoint approximation is, in general, better suited for approximating the power envelope near the bounds of the parameter space, where the Edgeworth approximation fails. Also, the simulations to obtain the critical values for the optimum tests described in sections 2.7.1 and 2.7.2 also suggest the saddlepoint technique might be more appropriate than the Edgeworth approximation when the design matrix is such that the test statistic has high skewness and kurtosis. The saddlepoint approximation (using Barndorff-Nielsen / Jensen) is shown in figure 2.14 (compare to figure 2.9). Figure 2.15 compares both Edgeworth and Saddlepoint approximations for designs 1 to 5 and a small sample size ($n = 30$).
Figure 2.14: Power envelope using Saddlepoint ($m=2$)
Figure 2.15: Power envelope: Edgeworth versus Saddlepoint approximations (n = 30)
Figure 2.16: Power envelope (saddlepoint) against information distance
Figure 2.17: Comparison of power envelope (saddlepoint) for different designs
2.7.5 Virtually Most Powerful Test

The construction of the power envelope for the optimum test allows, among others, to evaluate the loss of power of using a non-optimal test procedure. For instance, in section 2.6.1 we argue that in the limiting case $\rho_1 \to 0$ the test statistic $V_n(Y, 0)$ is locally uniformly most powerful invariant (LUMP) for testing $H_0 : \rho \leq 0$ vs. $H_1 : \rho > 0$, and in the particular case where the spatial weight matrix is linear, the test statistic would also be equivalent to Moran I test statistic. If we define the following $\alpha$-size test:

$$
\psi^*(y) = \begin{cases} 
1 & V_n(y, 0) > c \\
0 & V_n(y, 0) < c 
\end{cases}
$$

where the critical value $c$ is determined by $E_0 \psi^*(V_n(Y, 0)) = \alpha$, then we can calculate the loss of power associated to the LUMP test (or, equivalently, score or Moran I tests) by $\beta^*(\rho) - \beta(\rho) = F_{\rho, V_n(0)}(c_\alpha(\rho)) - F_{\rho, V_n(0)}(c)$, for all $\rho \in \mathcal{P}$. This measures how far the power of the LUMP test is from the optimum test, and therefore we could check whether there would be room for improvement in the testing procedure.

We calculate the power loss of the LUMP test for designs 1 to 5, for sample sizes $n = 30$ and $n = 92$, and for different degrees of connectivity of the spatial weight matrix (see table 2.18 for the combinations used). Figure 2.18 illustrates that the loss of power by using the LUMP test is small for all the designs considered. It reaches a maximum loss close to 1.2 percentage point for design 3 ($n = 92, J = 16$). Therefore, we could say that for the designs analyzed the LUMP test is virtually uniformly most powerful invariant, in the sense that for practical purposes we can consider it achieves the power envelope.

The nearly optimal behavior of the LUMP test for the proposed designs is probably related to the curvature of the of the problem, in the sense of Efron (1975). In that paper, Efron defines statistical curvature for curved exponential models (which is the case for the SAR stochastic model) and more broadly for arbitrary family of density functions indexed by a scalar parameter, and argues that under large curvature the use of a locally most powerful test (LMP) can be questioned\textsuperscript{28}. The author establishes an ad-hoc large value as: $(curvature)^2 \geq \frac{1}{\lambda}$. Using this threshold, none of the curvature parameters for the different combination of designs, sample size, and connectivity (see table 2.33) can be considered large, therefore it is no surprise that the LUMP test performs well in all the situations.

But, what if the statistical curvature of the model under investigation is large and the loss of power of the LUMP test is larger than what the researcher is willing to accept? One possibility is to use the Neyman-Pearson optimal test for a specific alternative hypothesis, as suggested in Efron (1975). In the context of the SAR stochastic process, we could use the square-root of Fisher information (see equation 2.7.2) evaluated at zero to approximate the standard deviation of $\rho$ and then consider an alternative hypothesis that is one standard deviation away from the null hypothesis, i.e., $\rho_1 = \sqrt{\chi^2(0)}$. The resulting power loss of this alternative testing procedure is in figure 2.19, where we plot the power loss curves for both the LUMP and this alternative testing procedure for design 3, with $n = 92$, and $J = 16$. The figure illustrates that this alternative testing procedure has a larger power loss than the LUMP test for values of $\rho$ near the null hypothesis, but a better performance for larger values of $\rho$. Overall, the power of this alternative testing procedure is closer to achieving the power envelope than the LUMP test for a wider range of values for $\rho$.

\textsuperscript{28}The higher the curvature, the higher the expected power loss of using a linear approximation (LUMP).
(a) Sample size $n = 30$

(b) Sample size $n = 92$

Figure 2.18: Power loss of the LUMPI test
Table 2.33: Statistical curvature (square of)

<table>
<thead>
<tr>
<th>Model</th>
<th>Sample size</th>
<th>(a)</th>
<th>(b)</th>
<th>(c)(^a)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design 1</td>
<td>(n = 30)</td>
<td>0.014</td>
<td>0.022</td>
<td>0.047</td>
</tr>
<tr>
<td></td>
<td>(n = 92)</td>
<td>0.004</td>
<td>0.006</td>
<td>0.011</td>
</tr>
<tr>
<td>Design 2</td>
<td>(n = 30)</td>
<td>0.011</td>
<td>0.064</td>
<td>0.064</td>
</tr>
<tr>
<td></td>
<td>(n = 92)</td>
<td>0.003</td>
<td>0.018</td>
<td>0.042</td>
</tr>
<tr>
<td>Design 3</td>
<td>(n = 30)</td>
<td>0.043</td>
<td>0.063</td>
<td>0.070</td>
</tr>
<tr>
<td></td>
<td>(n = 92)</td>
<td>0.050</td>
<td>0.088</td>
<td>0.107</td>
</tr>
<tr>
<td>Design 4</td>
<td>(n = 30)</td>
<td>0.013</td>
<td>0.016</td>
<td>0.020</td>
</tr>
<tr>
<td></td>
<td>(n = 92)</td>
<td>0.003</td>
<td>0.005</td>
<td>0.005</td>
</tr>
<tr>
<td>Design 5</td>
<td>(n = 92)</td>
<td>0.007</td>
<td>0.014</td>
<td>0.018</td>
</tr>
</tbody>
</table>

\(^a\)\(a, b, c\), means \(m = 2, 6, 10\) for designs 1, 2 and 4, \(J = 9, 15, 22\) for design 3 with \(n = 30\), \(J = 16, 27, 38\) for design 3 with \(n = 92\).

Figure 2.19: Power loss of LUMPI test vs. alternative test for design 3 (\(n = 92, J = 16\))
2.8 Conclusion

In this paper I investigate size and power properties of tests to detect spatial dependence in spatial autoregressive (SAR) stochastic processes defined on a lattice. The article focuses mainly on the small sample properties of the tests and on the impact of different designs for the spatial weight matrix on test's properties.

There are several reasons why spatial models are an important area for research. They have been used in various fields, such as economics (new economic geography, peer influence, neighbor effects), sociology (externalities, diffusion and contagion in crime analysis), political science (individual voting behavior, international relations), among others. Spatial dependence structure can also arise even when not explicitly in the model, due, for instance, to omitted variables, misspecification of the true data generating process, and measuring problems. Therefore, it might be a good practice to consider the spatial dependence explicitly in any model to start with and then test whether or not this hypothesis can be rejected. Moreover, in regression analysis, biased variance estimators, inefficient estimators or even bias and inconsistency in OLS estimators can result from ignoring spatial dependence.

The importance of studying the consequences of different designs for the spatial weight matrix comes up because, in practical applications, the researcher seldom can count on a structure of spatial dependence that is tied to a well defined theoretical model. Often, the spatial weight matrix is an arbitrary description of the spatial dependence that tries to capture the interaction suggested by the underlying model. Therefore, the importance of investigating the effects of changes in the spatial weight matrix on the properties of the tests being used.

This paper contributes to the literature on spatial models testing by (i) using Edgeworth and saddlepoint to approximate the small sample distribution of test statistics to detect spatial dependence in SAR processes (nearly all the research in this area rely on Monte Carlo simulation), (ii) providing a performance comparison among Edgeworth and several saddlepoint approximations (Robinson, Lugannani-Rice, Barndorff-Nielsen / Jensen), and suggesting which characteristics of the spatial weight matrix make either preferable, (iii) using the optimal (in the Neyman-Pearson sense) statistic to construct the power envelope for testing the presence of spatial dependence in the SAR model when (a) the variance of the error term is known, and (b) the optimal invariant test when the variance is unknown (closed formulas for the critical values of these optimal test statistics are provided), (iv) allowing a simple way to correct for size when comparing power of the tests (to my knowledge most of the Monte Carlo experiments do not make this correction), and (v) allowing the spatial weight matrix to be a nonlinear function of the spatial dependence parameter, therefore expanding on existing literature which focus mostly on linear functions.

In this article I find that Edgeworth and saddlepoint approximations are accurate for small samples ($n = 30$) for a variety of designs for the spatial weight matrix. Also, the actual size of the tests when using the critical values computed using the approximations are materially closer to the nominal size of the tests when compared to using asymptotic critical values. The optimal invariant test converges to the score (Lagrange multiplier) test when the alternative hypothesis converges to the null of no spatial dependence, and is therefore locally uniformly most powerful invariant (LUMPI). In this research, I found that, for the designs of the spatial weight matrix analyzed, the LUMPI test is virtually uniformly most powerful, in the sense that for practical purposes its power is very close to the power envelope. This probably reflects the small curvature (in the sense of Efron (1975)) of the model for the designs proposed. Since this nearly optimality might not hold for different designs, I suggest a practical procedure to build a test that, while not UMP, retain good power properties in a wider range for the spatial parameter when compared to the LUMPI test.

With the proposed test statistics, I analyze how the power envelope behaves for different sample sizes and different designs for the weight matrix. I find that power increases with sample size and with the spatial dependence parameter, which is in tandem with the literature. How-
ever, I call into question the consensus view that power *decreases* as the spatial weight matrix becomes more densely connected (see Anselin and Florax, 1995; Florax and de Graaff, 2004, and the references there in). The reduction of power observed in the literature reflects an error of measure where, despite looking at the same spatial dependence parameter, the hypothesis being compared are at different "statistical" distance from the null. I use the average information distance (based on Fisher information) to measure the distance between the null and the alternative hypothesis and, after correcting for this, the power is larger for alternatives further a way from the null, as expected. For alternatives at the same distance from the null, the power of the tests are very similar, regardless of the degree of connectivity of the weight matrix.
Chapter 3

Specification and Estimation of a New Class of Spatial Autoregressive Moving Average Models Under Unknown Heteroskedasticity

3.1 Introduction

The current set of spatial models typically used in both applied and theoretical research is called Cliff & Ord--type of spatial models. A taxonomy for these models has been proposed by Anselin (1988) and it is generally agreed that these set of models are fairly general. However, some authors complain the parameters do not have a clear interpretation in these type of spatial models, since the same parameter affects both the mean and the variance structure of the stochastic process. For instance, Arraiz et al. (2008) is a recent example where this concern is raised.

An alternative would be to consider spatial models where a different set of parameters are used to model spatial dependence in the mean structure and in the covariance structure. Cressie (1993) is among the authors that called attention to this problem and, since it was the first reference where I have read about it, I have named this as Cressie’s critique (see section 1.5.1).

Anselin (2003) proposes a new taxonomy of spatial models as an alternative to the traditional Cliff & Ord--type of spatial models, where the spatial dependence structure is either local or global. However, global and local specifications are non-nested and Anselin himself noted this drawback. The author speculated about the possibility of considering models where, for instance, local externalities would be present in the error term and global externalities in the covariates. He also highlighted that none of these possibilities had received attention in the literature and, to the best of my knowledge, this remains an unexplored area both in theoretical as well as in applied econometrics.

In order to overcome Cressie’s critique and to build on Anselin’s suggestion, I propose a more general parametric spatial stochastic process that nests Anselin’s suggested local and global models into an encompassing structure and allows for a clearer interpretation of the model’s parameters vis-a-vis the existing models. This model provides some insights on the structure of the commonly used model \( \mathbf{M}_{SARAR} \) but they are non-nested. Therefore they become competing models whose validity needs to be tested empirically. I also propose an estimation procedure for the model’s parameters that has the benefit of being easily implementable and computationally simple even when the dimension of the parameter space is not small. The asymptotic distribution of the parameter’s estimators is also obtained.

The remaining of the chapter is organized as follows: a review of the current spatial econometrics models and the literature on their parameter estimation is briefly stated in section 3.2.
Section 3.3 contains the proposed model, and its comparison with existing models. In section 3.4 I detail the proposed model and its assumptions, and suggest an estimation procedure. Sections 3.5 and 3.6 details the proposed parameter estimators and establish their large sample properties. Additionally, estimators for the large sample covariance matrices of the estimators are provided, and their consistency demonstrated. Concluding remarks are given in section 3.7, while some additional details are left to the Appendices.

3.2 Current Spatial Models

Probably one of the most referenced model of spatial dependence is due to Cliff and Ord (1973) (see also Ord, 1975) which itself makes reference to Whittle (1954). Since then, several Cliff & Ord-type of models have been investigated in the literature. These type of models refer, more specifically, to a subset of spatial stochastic process called (regular or irregular) lattice models (see Cressie, 1993). To my knowledge, Anselin (1988) is one of the earliest attempts to provide a taxonomy for Cliff & Ord-type of spatial econometric models. He proposed the following model, called mixed regressive, spatial autoregressive model with spatial autoregressive disturbances (MSAR, or simply SARAR)\(^1\):

\[
\begin{align*}
Y &= \lambda M_1 Y + X \beta + U \\
U &= \rho M_2 U + \varepsilon \\
\varepsilon &\sim N(0, \Sigma) \\
\Sigma &\text{ diagonal}
\end{align*}
\]

(3.2.1)

where \(Y\) is an \(n\)-vector of observations on \(n\) spatial units, \(M_1, M_2\) are two \(n \times n\) matrices representing the spatial dependence structure (or neighborhood structure) on the spatial units, \(X\) is an \(n \times k\) matrix of covariates, and \(\varepsilon\) is the error term.\(^2\)

When \(\rho = 0\) the resulting model would be called mixed regressive, spatial autoregressive model or spatial lag model (MRSAR, MSAR, or simply SAR). When \(\lambda = 0\) the model would be called a linear regression with spatial autoregressive disturbances, or simply a generalized least squares (GLS) model where the covariance matrix has a spatial autoregressive form. This taxonomy, however, is far from consensual. In LeSage and Pace (2009), for instance, the authors call model \(M_{\text{SARAR}}\) by SAC\(^3\), model \(M_{\text{SAR}}\) simply spatial autoregressive model or spatial Durbin model (SDM) when the regressors also include spatial lags of the covariates, and model \(M_{\text{GLS}}\) spatial error model (SEM).

The theoretical literature on parameter estimation in this class of spatial econometric models (\(M_{\text{SARAR}}\)) has recently been formalized in a series of papers mostly by Lee, and Kelejian and Prucha. Maximum likelihood estimation has been a tool for estimation of parametric spatial econometric models since the very beginning (see, Cliff and Ord, 1973; Anselin, 1988, for instance), but only somewhat recently, Lee (2004) provided a rigorous analysis of the ML (and QML) estimator for model \(M_{\text{SAR}}\). Mostly as alternatives to computer-intensive maximum likelihood estimation, Kelejian and Prucha (1999) proposed a generalized moments estimator for the autoregressive parameter in a linear regression with spatial autoregressive disturbances (\(M_{\text{GLS}}\)). Kelejian and Prucha (1998) extends the previous work to present a generalized spatial two-stage least squares (GS2SLS) procedure for estimation of \(M_{\text{SARAR}}\) model’s parameters. The debate quickly followed to the choice of the optimal set of instruments (for GS2SLS estimator), and Lee (2003) proposed a best generalized spatial two-stage least squares estimator (BGS2SLS). The finite sample superiority of the BGS2SLS estimator, however, was questioned in Das et al. (2003). In Lee (2007), the author proposed a GMM plus 2SLS framework to estimate model

\(^1\)There is also the analogous \(M_{\text{SARMA}}\), where the disturbances follow a moving average process (Anselin et al., eds, 2004).

\(^2\)Section 3.4 will specify in details the model’s components.

\(^3\)I looked hard but could not find the meaning of this acronym; perhaps spatial autocorrelation?
$M_{SAR}$, and showed that his estimator could be more efficient than Prucha’s GS2SLS estimator and as efficient as the ML estimator when the disturbances are normally distributed.

The drawback of all the above estimation procedures, however, was that they were all based on homoskedastic disturbances, even though the original model $M_{SAR}$ allowed for the presence of heteroskedasticity in the covariance matrix $\Sigma$ (Anselin, 1988). As highlighted in Lin and Lee (2009); Kelejian and Prucha (2009), spatial units are often heterogeneous in some characteristics (e.g., size) and often result from aggregation (e.g., unemployment); therefore heteroskedasticity is likely to be present. Almost simultaneously, these authors extended their earlier work to allow for unknown heteroskedastic innovations and provided proofs of the consistency and asymptotic normality of their proposed estimators. Some Monte Carlo experiments were also provided (Das et al., 2003; Arreliz et al., 2008) to investigate the small sample performance of their estimators and to support the proposed estimation methods.

The $M_{SAR}$ model is largely seen as comprehensive, by potentially encompassing several different types of stochastic processes. Anselin (1988), for instance, argues that “most situations of interest are taken into account” in this model, a view that is also repeated more recently in Kelejian and Prucha (2009): “...this model is fairly general in that it allows for spatial spillovers in the endogenous variables, exogenous variables and disturbances”. This model, however, has the disadvantage of mixing the effects of a single parameter into both the mean and the covariance structure of the spatial stochastic process. This might be one of the reasons behind the comment on the results of the Monte Carlo simulation in Arreliz et al. (2008, p.13):

Intuitive explanations of the table results thus far discussed as they relate to the values of $\rho$ and $\lambda$ are not straightforward. As one example, the reduced form for $Y$ is $Y = (I - \lambda W)^{-1} X \beta + (I - \lambda W)^{-1} (I - \rho W)^{-1} \varepsilon$

If $\lambda$ is large in absolute value, say close to 1.0, the variances of the elements of the error vector will, ceteris paribus, tend to be large, since $\lambda = 1$ is a singular point of the inverse matrix. These large variances will obviously have a negative effect on estimation precision. On the other hand, increased variation on the vector $Y$ will, ceteris paribus, increase the variation in $WY$, which is a right hand side variable, and this should increase estimation precision. The net effect on estimation precision of a large value of $\lambda$ will obviously be the result of these two effects and it is not clear which of these two effects would dominate in a particular case. Similar concerns relate to the value of $\rho$ since, on the negative side, it also enters the error term in the same fashion as $\lambda$; on the positive side, $\rho$ can be viewed as a regression parameter and the larger the value of $\rho$ the more $WU$ varies and so the more precision is increased! Of course, intuitive interpretations of our results are made still more complex by the interactive effects of $\rho$ and $\lambda$.

The problem of having a single parameter affecting both the mean and the covariance structure had already been pointed out by, among others, Cressie (1993), where the author suggests as good practice to model both effects separately. Along these lines, Anselin has updated his views (he calls his previous proposed taxonomy “too simplistic”) and suggests (Anselin, 2003) another taxonomy that classifies the spatial stochastic process according to whether the spatial effects (or externalities, as he calls it) are (i) local, or (ii) global, and whether the spatial effects occur in (a) the covariates (explanatory variables), in (b) the error term, or in (c) both. This classification survives Cressie’s critique.

Below we present the article’s table with the structural forms of the proposed models, where $Y$ is the dependent variable, $X$ are the covariates, $\varepsilon$ are the innovations, and $M_1, M_2$ are, respectively, the spatial weight matrices that represent the dependence structure in the covariates and in the innovations. Anselin also remarks that, in his proposed taxonomy (table 3.1), “while the unconstrained forms for both global and local models are nested within their own category (i.e., a simple model can be obtained by imposing a zero restriction on a parameter of a more
Table 3.1: Anselin’s taxonomy of spatial models

<table>
<thead>
<tr>
<th>Local</th>
<th>Global</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\varepsilon$</td>
<td>$M_{SL}$</td>
</tr>
<tr>
<td>$X$</td>
<td>$M_{LC}$</td>
</tr>
<tr>
<td>$\varepsilon, X$</td>
<td>$M_{SL}$</td>
</tr>
</tbody>
</table>

Note: $\alpha$ and $\gamma$ represent the local effects, and $\rho$ and $\lambda$ represent the global effects; the spatially lagged explanatory variables ($M_1 X, M_2 X$) does not include a constant, since there could be identification problems.

complex model), this is not the case across categories*, i.e., the global and local specifications are non-nested. The author also speculated on the possibility of considering models, for example, with local externalities in the error term and global externalities in the covariates (or vice-versa). He highlighted that none of these possibilities had received attention in the literature and, to my knowledge, this remains an unexplored area both in the theoretical as well as in the applied econometrics.

It is also interesting to note that when the global effects parameters in model $M_{SG}$ are the same ($\rho = \lambda$), and the spatial weight matrices are also the same ($M_1 = M_2$), then the model simplifies to the model $M_{SAR}$ considered, for instance, in Lee (2004). On the other hand, no restrictions imposed in the models $M_{SL}$ or $M_{SG}$ (table 3.1) would yield the commonly used model $M_{SAR}$. Anselin, for example, mentions “It is also striking that the often cited model with both a spatially autoregressive dependent variable and a spatially autoregressive error structure does not follow from either global or local spatial externalities...”, adding “It remains an empirical matter whether or not it (i.e., model $M_{SAR}$) may be more appropriate in given situations, but its interpretation involves a more complex spatial pattern of externalities than for the models included in table 3.1” (recall Aroraiz et al., 2008, remarks on the difficulty of interpreting the parameters in model $M_{SAR}$).

3.3 Proposed Spatial Model

Building up on Anselin’s proposed taxonomy and supported by Cressie’s critique, I suggest a more general spatial stochastic process that nests local model $M_{SL}$ and global model $M_{SG}$ (see table 3.1, section 3.2) into an encompassing structure. Let $M$ denote the following model (or family of DGP’s):

$$
Y = \rho M_2 Y + \lambda M_1 Y - (\rho \lambda) M_2 M_1 Y + X\beta + M_1 X_1 \gamma - M_2 X (\rho \beta) \\
- M_2 M_1 X_1 (\rho \gamma) + \varepsilon - \alpha M_2 \varepsilon - \lambda M_1 \varepsilon + \lambda \alpha M_1 M_2 \varepsilon
$$

$$
\varepsilon \sim N (0, \Sigma), \Sigma \text{ diagonal}
$$

(3.3.1)

where the variables have the same meaning than in equation 3.2.1, $X_1$ is a subset of the explanatory variables $X$ for which it makes sense to think of a “spatial lag”, and matrices $M_1, M_2$ commute.4 Table 3.2 highlights that most of the commonly used spatial stochastic process, Anselin’s proposed models, as well as some newly suggested models, are nested in the proposed model $M$. Note that, by looking independently at the local and global spatial effects in model $M$, it becomes clear that the commonly used model $M_{SAR}$ has a hidden interpretation (which is

4This includes, for instance, the possibility of both matrices being the same (any matrix commutes with itself), one of them being the null matrix, or being orthogonal matrices $M_1 M_2 = 0$.}
Table 3.2: General model $M$

<table>
<thead>
<tr>
<th>Parameter restriction</th>
<th>Resulting model</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\rho = \lambda = 0$</td>
<td>Local externalities in both covariates and error ($M_{SL}$)</td>
</tr>
<tr>
<td>$\rho = \lambda = \alpha = 0$</td>
<td>Local externalities in covariates ($M_{SLC}$)</td>
</tr>
<tr>
<td>$\rho = \lambda = 0, \gamma = 0$</td>
<td>Local externalities in error ($M_{SLp}$)</td>
</tr>
<tr>
<td>$\alpha = 0, \gamma = 0$</td>
<td>Global externalities in both covariates and error ($M_{SG}$)</td>
</tr>
<tr>
<td>$\alpha = \rho = 0, \gamma = 0$</td>
<td>Global externalities in covariates ($M_{SGC}$)</td>
</tr>
<tr>
<td>$\alpha = \lambda = 0, \gamma = 0$</td>
<td>Global externalities in error ($M_{SGP}$)</td>
</tr>
</tbody>
</table>

$\alpha = \lambda = 0$  | Local externalities in covariates and global in error $Y = \rho M_2 Y + X \beta + M_1 X_1 \gamma - M_2 X (\rho \beta) - M_2 M_1 X_1 (\rho \gamma) + \epsilon$

$\gamma = 0, \rho = 0$  | Global externalities in covariates and local in error $Y = \lambda M_1 Y + X \beta + \epsilon - \alpha M_1 \epsilon - \lambda M_1 \epsilon + \lambda \alpha M_1 M_2 \epsilon$

$\rho = \lambda, M_1 = M_2$  | Global externalities in covariates and error are the same $Y = \lambda M_1 Y + X \beta + M_1 X_1 \gamma + \epsilon - \alpha M_1 \epsilon$

$\rho = \lambda, M_1 = M_2, \gamma = 0, \alpha = 0$  | Global externalities in covariates and error are the same and no local externalities (this is model $M_{SAR}$): $Y = \lambda M_1 Y + X \beta + \epsilon$

$\gamma = 0, \alpha M_2 = -\lambda M_1 (I - \lambda M_1)^{-1}$  | Global externalities in covariates and error, and structure of local dependence in the errors given as a function of $M_1, \alpha, \lambda$; these restrictions yield model $M_{SAR}$: $Y = \lambda M_1 Y + X \beta + U$, $U = \rho M_2 U + \epsilon$

not clear from its standard model representation): it includes a local spatial effect in the innovations that is linked to the global spatial effect assumed for the covariates. Failing to consider this clearly complicates the analysis of the meaning of model $M_{SAR}$ parameters, as highlighted in Arraiz et al. (2008). Also note that if we impose $M_1 = M_2$ (as it is usual in the applied literature), then model $M_{SAR}$ fails to be nested in model $M$. The exception happens when $\lambda$ is small, since in this case $\alpha M_2 \approx -\lambda M_1$ and this would then imply that the parameter for the local spatial effect in the error would be equal (in absolute value) to the parameter for the global spatial effect in the covariates, i.e., $\alpha = -\lambda$.

Besides standing up to Cressie’s critique, the proposed family of DGP’s represented in $M$ also allows new insights on the structure of the commonly used model $M_{SAR}$. However, the fact that, in general, models $M$ and $M_{SAR}$ are non-nested, makes them competing models whose validity and relevance needs to be tested empirically. I hope to contribute to this challenge by presenting, in the following sections, a simple procedure for estimating the parameters of model $M$.

### 3.4 The General Spatial Model

In the previous section (3.3), I introduced a parametric class of general spatial stochastic processes, whose main appeal was to blend local spillovers (or moving average), global spillovers (or autoregressive), or both spatial effects, affecting the covariates and the error structure (independently or simultaneously), while providing a clear interpretation of model’s parameters. In other words, a model where the parameters that characterize the conditional mean of the dependent variable are (a priori) different from the parameters that characterize its higher conditional moments.
To clarify this point, consider the reduced form of model $M$ in equation 3.3.1:

$$Y = f(\delta; M_1, X) + U$$

$$f(\delta; M_1, X) \equiv (I - \lambda M_1)^{-1} (X \beta + M_1 X_1 \gamma)$$

$$\delta \equiv [\lambda \ \ \beta' \ \ \gamma'], \quad S(X) \subset S(X)$$

$$U = (I - \rho M_2)^{-1} (I - \alpha M_2) \varepsilon$$

(3.4.1)

where it becomes clear that $E(Y | X)$ is a function of $\delta$ and $\text{Var}(Y | X)$ is a function of $\theta \equiv [\alpha \ \ \rho]$. The general spatial model $M$ assumes that $Y$ is an vector of observations on $n$ spatial units, $X$ is an $n \times k$ matrix of regressors (including non-stochastic regressors, such as the constant and dummy variables), $S(X)$ is the space spanned by the columns of $X$, $X_1$ is an $n \times k_1$ matrix of stochastic regressors (for which it makes sense to think of a "spatial lag"), $f(\cdot)$ is the regression function that determines the mean value of $Y$ conditional on $\delta$ and on the stochastic variables in $X$. $M_1, M_2$ are $n \times n$ matrices of known constants representing, respectively, the spatial dependence structure in the covariates and in the innovations $\varepsilon$.

The spatial structure of model $M$ is embedded in the form of matrices $M_1$ and $M_2$. Consider, for instance, the model with no spatial dependence structure in the covariates ($\lambda = 0, \gamma = 0$). Assume also that the observations $i = 1, \ldots, n$ represent a set of $n$ spatial units and that some neighborhood (binary, irreflexive) relation $\mathcal{N}$ has been established, i.e., for each spatial unit $i$ there is a set $N_i \equiv \{\text{spatial units } j \mid j \in \mathcal{N}_i\}$. Now define $M_2$ structure as $m_{2,ij} \neq 0, \forall j \in N_i$ and $m_{2,ij} = 0$ otherwise. This results in, for all $i = 1, \ldots, n, y_i = x_i' \beta + u_i$, where $u_i = \varepsilon_i + \sum_{j \in N_i} m_{2,ij} (\rho u_j + \alpha \varepsilon_j)$, i.e., the spatial unit $i$ is affected by its own and by its neighbors disturbances (and by the disturbances of the neighbors of its neighbors, etc.). In this setting, the parameters $\alpha, \rho$ are scalars that try to capture this spatial dependence effect.

The parameters $\beta \in \mathbb{R}^k$, in the absence of spatial effects, are the standard parameters in typical linear regression models. $\lambda \in \mathbb{R}$ and $\gamma \in \mathbb{R}^{k_1}$ are the global and local spatial effects in the covariates, and $\rho, \alpha \in \mathbb{R}$ are the global and local spatial effects in the error structure. Define $\Phi \equiv A \times R \times \mathcal{N} \times B \times G$, where $\Phi \subset \mathbb{R}^{3+k+k_1}$, and let $\phi : \mathbb{M} \to \Phi$ be the parameter-defining map of the model $M$. Thus, for each DGP $m_0 \in \mathbb{M}$ there is an associated parameter $\phi_0 = \phi(m_0) = [\phi_0 \ \ \rho_0 \ \ \lambda_0 \ \ \gamma_0]' \in \Phi$ that we are interested in (estimating, testing)\(^6\). The pair $(\mathbb{M}, \phi)$ is called a parametrized model (Davidson and MacKinnon, 1993), and in this case the DGP is only partially characterized by the model parameters, since no distribution for the innovations $\varepsilon_i$ are assumed. Important to note that we are ruling out from the beginning DGPs where both $\beta = 0$ and $\gamma = 0$, since in this case the value of $\lambda$ would be irrelevant and $\phi(\cdot)$ would not be a mapping from the family of DGPs to the parameter space. For the same reason models where $\alpha = \rho$ are precluded.

The question of asymptotic identification of the parametrized model $(\mathbb{M}, \phi)$ will be dealt with in an affirmative manner, i.e., I will show the existence of consistent estimators for model’s parameters. To be precise, the definition of asymptotic identifiability follows (Davidson and MacKinnon, 1993):

**3.4.1 Definition** (Asymptotic identifiability). A parametrized model $(\mathbb{M}, \phi)$ is said to be asymptotically identified if for any $\phi_1, \phi_2 \in \Phi$ with $\phi_1 \neq \phi_2$ there exists some sequence of functions $\{E_n\}$ such that

$$\lim_{n \to \infty} E_n(Y, X, M_1, M_2) \neq \lim_{n \to \infty} E_n(Y, X, M_1, M_2)$$

where at least one of the plims exists and is a finite constant.

---

\(^5\)Contrast this result with model $\mathbb{M}_{SAVAR}$, where the conditional variance depends on the same parameter that appears in the conditional mean specification.

\(^6\)Observe that we have intentionally excluded the $n$ elements of $\Sigma$ from the parameter-defining map, since there is no hope of estimating them consistently.
In other words, if there is a consistent estimator \( \hat{\phi} \) of the model parameters, it would be the obvious candidate to replace function \( F \) in the above definition. However, as it will be shown throughout the paper, I will only be able to present a consistent estimator of the conditional mean parameters \( \delta \). The parameters of the conditional variance \( \theta \) will be only locally identified, i.e., the model will be asymptotically identified only if, for each \( m_0 \in M \), we restrict the parameter space of \( \theta \) to be a neighborhood of the (unknown) \( \theta_0 \)! This problem is not present in the current theoretical literature cited in the introduction since both Kelejian and Prucha (2009); Lin and Lee (2009) do not consider local spillover in the error structure. In the case where only the global spatial effect is present in the error structure its parameter is globally identified (see section 3.6.1) and therefore the full model is asymptotically identified. For practical purposes, however, this might not be a serious problem. Even without being able to provide a global identification condition for \( \theta \) in the general case, Monte Carlo simulation and some examples in the text highlight that this parameter is indeed uniquely identified in the usually adopted parameter space.

In the following sections I will lay down the set of assumptions of the model \((M, \phi)\) and the proposed estimation procedure for its parameters.

### 3.4.1 Assumptions

In this section I will consolidate all the assumptions on the general spatial model proposed \((M)\). For clarity and easy of future reference, they will be classified into assumptions on the spatial matrices, covariates and instruments, parameter space, moment equations, and innovations.

#### 3.4.2 Assumption (Spatial Matrices). \( M_1, M_2 \) are \( n \times n \) matrices of known constants, whose elements may depend on \( n \), and that satisfy: (i) diagonal elements of \( M_i, i = 1, 2 \) are zero, (ii) matrices \( M_i, i = 1, 2 \) are uniformly bounded\(^7\) by 1, i.e., \( \| M_i \|_1 \leq 1, \| M_i \|_\infty \leq 1 \), for \( i = 1, 2, \forall n \), and (iii) matrices \( M_1, M_2 \) commute.

#### 3.4.3 Assumption (Covariates and instruments). (i) the elements of the \( n \times k \) matrix of regressors \( X \) (which may depend on \( n \)) are bounded, (ii) \( \frac{1}{\sqrt{n}} X \) has full column rank for all \( n \), (iii) \( \frac{1}{n} X' X \) is positive definite for all \( n \), (iv) the \( n \times p \) (\( p = k + k_1 \)) matrix \( \frac{1}{\sqrt{n}} [ X \; M_1 X_1 ] \) has full column rank, (v) for given values of \( a, b \), define the matrix of potential instruments as \( H = [ X \; \ldots \; M_1^2 X \; M_2^2 \; \ldots \; M_2^2 M_1^2 ] \), let \( H \) be an \( n \times \bar{p} \) full rank matrix of instruments \((\bar{p} \geq p + 1)\) composed of a subset of the linearly independent columns of \( H \) (containing at least the linearly independent columns of \( X \) and \( M_1 X_1 \)), and let \( \Delta = [ M_1 Y \; \ldots \; M_1 X ] \); assume that \( \frac{1}{n} H' \Delta \) is finite and has full column rank for \( n \) large.

#### 3.4.4 Assumption (Parameter Space). (i) the parameter space \( \Phi \) is assumed to be a compact set, (ii) the parameter space for \( \lambda, \alpha, \rho \) \((L, A, R, \) respectively) are a closed intervals contained in \((-1, 1)\), containing zero, (iii) the DGP \( m_0 \) is such that \( \phi_0 \) is in the interior of the parameter space, (iv) the parameters \( \beta \) and \( \gamma \) cannot both be null, since in this case \( \phi(\cdot) \) would not be a mapping from the family of DGPs to the parameter space, and (v) the parameters \( \alpha \) and \( \rho \) cannot be equal, since this would reduce to the case of no spatial effect in the error term.

#### 3.4.5 Assumption (Moment equations). (i) let \( \hat{W}_n \) be a \( l \times l \) matrix, \( \lim_{n \to \infty} \hat{W}_n = W \), where \( W \) is a deterministic matrix, symmetric, positive definite; and (ii) let \( K_i, i = 1, \ldots, l \) be a set of \( n \times n \) matrices, uniformly bounded, with zeros in the diagonal.

#### 3.4.6 Assumption (Innovations). (i) the elements of \( \varepsilon \) are independent, but not necessarily identically distributed, with (non-null) variance \( E \varepsilon_i^2 = \sigma_i^2 \), (ii) for some \( \epsilon > 0, E[\varepsilon_i^{4+\epsilon}] < \infty, i = 1, \ldots, n, \forall n \), and (iii) \( \varepsilon \) is an innovation with regard to \( X \), i.e., \( E(\varepsilon_i | X) = 0, \forall i = 1, \ldots, n, \forall n \). Denote by \( \Sigma \) the diagonal matrix whose elements are \( \sigma_i^2 \).

\(^7\) We say an \( n \times n \) matrix \( A \) is uniformly bounded iff \( \lim_{n \to \infty} \| A \|_1 < \infty \) and \( \lim_{n \to \infty} \| A \|_\infty < \infty \); see appendix.
Assumption 3.4.2(i) is a normalization of the model commonly used in the literature (no spatial unit is viewed as its own neighbor).

Assumption 3.4.2(ii) together with assumption 3.4.4(ii) is a stability condition for the specifications of $M_1, M_2$ and assures that the error term $U$ is uniquely identified in terms of the innovation $\varepsilon$ (Kelejian and Prucha, 1999) and that the conditional mean function $f$ is well defined. These assumptions also imply that the covariance matrix is bounded, thus limiting the degree of correlation, which is required for the asymptotic results. It is usual in the literature to require only that the spatial matrices $M_1, M_2$ are uniformly bounded (by, say, a finite constant $c$) but with the added assumption that the inverses of $L_\lambda = I - \lambda M_1, A_\alpha = I - \alpha M_2, R_\rho = I - \rho M_1$ exist and are uniformly bounded as well. Instead, I took the route to bound $M_i$ by 1 (which requires the parameter space be adjusted accordingly) and the conclusion that the inverses of $L_\lambda, A_\alpha, R_\rho$ exist and are uniformly bounded follow as a lemma (see lemma 3.5.4). Assumption 3.4.2(ii) is usually satisfied, for instance, when a spatial unit has a limited number of neighbors, or when the spatial matrices are row-normalized. Assumption 3.4.4(ii), on the other hand, despite generally employed in the literature (Kelejian and Prucha, 1999, 1998; Lee, 2003) is a restrictive assumption. Both Kelejian and Prucha (2009) and chapter 2 bring a somewhat detailed description of the parameter space.

Assumption 3.4.2(iii) is key to allow Anselin (2003) proposed specifications for local and global spillovers (see section 3.2) to be nested within one encompassing model. For practical applications it is not as restrictive as it appears, since it includes, for instance, the possibility of both matrices being the same (which is adopted in most empirical applications), one of them being the null matrix, or the neighborhood structure be non-overlapping, i.e., satisfy $M_1 M_2 = 0$ (see Lee, 2003, for comments on this structure).

Assumptions 3.4.3(i)-(iv) are standard in the linear regression setting; in particular, they rule out multicollinearity among the regressors $X$ and $M_1 X_1$. Note that $M_1 X_1$ will include only spatial lags of the stochastic variables in $X$, since it is not straightforward to interpret the meaning of spatial lags of non-stochastic variables. Moreover, often the space spanned by the spatial lags of the non-stochastic variables are similar to the one spanned by the original (non-spatially lagged) variables. The extreme example of this situation is when a constant vector of ones (denoted by $\iota$) is included as the only non-stochastic covariate and the matrix $M_1$ is row-normalized. In this case, $M_1 \iota = \iota (\iota)$, so it is meaningless to speak of “spatial lag” of a constant vector. Throughout the text, all the analysis will be conditioned on the realized values of $X$, therefore it will subsequently be viewed as a matrix of constants.

Assumption 3.4.3(v) is an identification assumption similar to the ones in Lin and Lee (2009); Lee (2003); Kelejian and Prucha (1998, 2009). It allows the identification of the conditional mean parameters. Since the choice of the potential instruments ($\mathcal{H}$) may not be immediately clear, in section 3.5 I will detail its origin.

Assumptions 3.4.4(i) and (iii) are required to show consistency and asymptotic normality of the general method of moments (GMM) estimator of the covariance parameters $\theta$, and are usual when dealing with extremum estimators. Items (iv) and (v) assure that $\phi : \mathcal{M} \to \Phi$ is a mapping from the family of DGP to the parameter space; without it, identification would not be possible. Section 3.6 will state additional assumptions required for identification of the conditional covariance parameters.

Finally, assumptions 3.4.5 are needed for the moment equations proposed in the GMM framework and assumptions 3.4.6 are standard in the recent literature with heteroskedastic innovations (Lin and Lee, 2009; Kelejian and Prucha, 2009).

### 3.4.2 Proposed Estimation Procedure

In this section I will introduce the proposed estimation procedure for model ($\mathcal{M}, \phi$). It is basically a two step procedure (similar to the one adopted in Kelejian and Prucha, 2009, for the model $M_{SARAR}$) where in the first step the conditional mean parameters $\delta$ are estimated by instrumental
variables (IV) and the residual of this estimation is used in a GMM framework to estimate the conditional covariance parameters $\theta$. The main attractive of this approach vis-a-vis, for instance, the estimation procedure proposed in Lin and Lee (2009) for model $\mathcal{M}_{SAR}$, is the low dimensionality of the non-linear optimization step.

Going into the specifics, note that model $\mathcal{M}$ (equation 3.3.1) can also be put in the following form:

$$Y = \Delta\delta + V$$
$$\Delta = \begin{bmatrix} M_1 Y & X & M_1 X_1 \end{bmatrix}, \quad \delta' = \begin{bmatrix} \lambda & \beta' & \gamma' \end{bmatrix}$$
$$V = (I - \lambda M_1) (I - \rho M_2)^{-1} (I - \alpha M_2) \varepsilon$$

(3.4.2)

This form resembles the $\mathcal{M}_{SARAR}$ model studied in Kelejian and Prucha (1998); Lee (2003), except for the fact that (i) the regressors includes a spatial lagged $M_1 X_1$, and (ii) the disturbances follow an moving average, autoregressive spatial process instead of the autoregressive spatial process presented in those papers. This suggests that I could estimate the proposed model's parameters $\phi = [\alpha, \rho, \lambda, \beta', \gamma]'$ by using a procedure similar to the ones developed in those papers.

For the sake of completeness, note that the explanatory variables $\Delta$ are not exogenous, i.e., the randomness in the covariates (via $M_1 Y$) are not independent of the error term and therefore the OLS estimator of $\delta$ will in general be biased and inconsistent (see Lee, 2002, for specific designs where OLS estimator can be consistent), and thus instrumental variables estimator is usually needed for the parameters that enter into the conditional mean function.

Fortunately, finding instruments for the model in equation 3.4.2 is not controversial, at least in theory. Expanding the argument in Davidson and MacKinnon (2004, p. 318) for the generalized IV (GIV) case (i.e., when the covariance matrix of the error terms $V$ is known), we conclude that the optimal instrumental variables for model in equation 3.4.2, by the criterion of asymptotic variance, is $\Upsilon^\prime \hat{\Delta}$, where $\hat{\Delta} = \text{E}(\Delta | X, M_1, M_2)$ and $\Upsilon \Upsilon' = \text{Var}(V)^{-1}$. However, since $\text{Var}(V)$ is unknown, one would have to resort to feasible generalized IV (FGIV). However, even assuming there is a way to come up with a consistent estimator $\text{Var}(V)$ and thus $\hat{\Upsilon}$, the difficulties in showing the asymptotic equivalence of FGIV and GIV could be large, as pointed out in Mandy and Martins-Filho (1994, 1997). In the current framework, this could be even more cumbersome due to the fact that the same parameter ($\lambda$) appears both in the mean and in the covariance specification. Therefore, I opted to choose an easier route to find instruments, which is the one used in Kelejian and Prucha (2009); Lin and Lee (2009), for example, where the authors simply used the set of instruments developed in their earlier work for the homoscedastic case without questioning or qualifying them. As a side note, observe that the models used in those papers ($\mathcal{M}_{SARAR}$ and $\mathcal{M}_{SAR}$, respectively) have the property that, in the structural form (see equation 3.2.1), the parameters that appear in the covariance structure do not enter into the mean structure. Therefore it might be less complicated to improve on their estimation procedures by proposing a set of optimal instruments for their IV (or 2SLS) stages. This would be similar in spirit to the best generalized spatial two-stage least squares estimator (BGS2SLS) proposed by Lee (2003) for the homoscedastic case, but this approach is not followed in the current paper and is left for future research.

So, taking the simpler route and assuming away the covariance structure of the error term $V$, the instruments would be given by the $n \times (p + 1)$ matrix $\hat{\Delta} = \begin{bmatrix} M_1 E Y & X & M_1 X_1 \end{bmatrix}$, where, assuming the true DGP is $m_0 \in \mathcal{M}$, $E Y = (I - \lambda_0 M_1)^{-1} (X \beta_0 + M_1 X_1 \gamma_0)$. Assuming the eigenvalues of $\lambda_0 M_1$ are all inside the unitary circle, $(I - \lambda_0 M_1)^{-1} = \sum_{i=0}^{\infty} (\lambda_0 M_1)^i$, and the first

*Lemma 3.5.4 will derive this result from the model's assumptions.*
column of \( \tilde{\Delta} \) can be written as

\[
M_1 (I - \lambda_0 M_1)^{-1} (X_0 \beta_0 + M_1 X_1 \gamma_0) = M_1 (I + \lambda_0 M_1 + \lambda_0^2 M_1^2 + \cdots) (X_0 \beta_0 + M_1 X_1 \gamma_0)
\]

\[
= M_1 (I + \lambda_0 M_1 + \lambda_0^2 M_1^2 + \cdots) (X_0 \beta_{00} + X_1 \beta_{01} + M_1 X_1 \gamma_0)
\]

\[
= M_1 X_1 \beta_{01} + M_1^2 X_1 (\lambda_0 \beta_{01} + \gamma_0) + M_1^3 X_1 \lambda_0 (\lambda_0 \beta_{01} + \gamma_0) + \cdots
\]

\[
+ M_1 X_0 \beta_{00} + M_1^2 X_0 (\lambda_0 \beta_{00}) + M_1^3 X_0 (\lambda_0^2 \beta_{00}) + \cdots
\]

That is, the first column of \( \tilde{\Delta} \) is in the space spanned by the columns of \( M_1 X, M_1^2 X, \ldots \). This justifies the usual practice (see Kelejian and Prucha, 1998, for instance) of choosing

\[
\mathcal{H} = [X \quad M_1 X \quad M_1^2 X \quad \cdots \quad M_1^k X]
\]

(3.4.3)

as a potential set of instruments, at least for an initial consistent estimation. Observe that the value of \( b \) needs to be large enough to assure that \( \text{rank}(\mathcal{H}) \) will be at least as large as \( p + 1 \). The larger the value of \( b \) the higher the precision of the IV estimates, but at the cost of a larger sample bias (see Davidson and MacKinnon, 1993). Indeed, as reported in Lee and Liu (2006), the bias increases substantially for model \( M_{SARAR} \) when \( b \) is larger than 2.

Under the constraint that the instruments are linear combinations of the columns of \( \mathcal{H} \), it can be shown (see Davidson and MacKinnon, 2004) that the optimal instruments are \( P_H \Delta = [P_H M_1 Y \quad X \quad M_1 X_1] \), where \( P_H \) is the orthogonal projection in the space spanned by the columns of \( \mathcal{H} \).

Note that the structural form of model \( M \) can also be written as

\[
Y = \rho M_2 Y + \Delta \delta + M_2 \Delta (-\rho \delta) + E
\]

\[
= \Pi \pi + U
\]

\[
\Pi = [M_2 Y \quad M_1 Y \quad X \quad M_1 X_1 \quad M_2 M_1 Y \quad M_2 X \quad M_2 M_1 X_1]
\]

(3.4.4)

\[
\pi' = \begin{bmatrix}
\rho \\ \lambda \\ \beta' \\ \gamma' \\ -\rho \lambda \\ \rho \beta' \\ -\rho \gamma'
\end{bmatrix}
\]

\[
E = (I - \lambda M_1) (I - \alpha M_2) \varepsilon
\]

Following the previous steps, the set of instruments could be expanded to

\[
\mathcal{H} = [X \quad M_1 X \quad M_1^2 X \quad \cdots \quad M_1^k X \quad M_2 X \quad M_2 M_1 X \quad M_2 M_1^2 X \quad \cdots \quad M_2 M_1^k X]
\]

(3.4.5)

and the optimal instruments (again, restricted to be a linear combination of the instruments in \( \mathcal{H} \)) are

\[
P_H \Pi = [P_H M_2 Y \quad P_H M_1 Y \quad X \quad M_1 X_1 \quad P_H M_2 M_1 Y \quad M_2 X \quad M_2 M_1 X_1]
\]

where \( \text{rank}(\mathcal{H}) \) is at least as large as the rank of \( \Pi \). Moreover, since the space spanned by the columns of \( \mathcal{H} \) in 3.4.3 is a subset of the space spanned by the columns of \( \mathcal{H} \) in 3.4.5, \( P_H \Pi \) could also be used as a valid instrument for \( \delta \) in equation 3.4.2, and this is the approach followed in Kelejian and Prucha (1998).

It is possible to re-write the model in yet another structural form

\[
(I - \alpha M_2)^{-1} (I - \rho M_2) Y = (I - \alpha M_2)^{-1} (I - \rho M_2) (I - \lambda M_1)^{-1} (X \beta + M_1 X_1 \gamma) + \varepsilon
\]

\[
Y = (\rho - \alpha) M_2 Y + \alpha (\rho - \alpha) M_2^2 Y + \cdots
\]

\[
+ (I - \alpha M_2)^{-1} (I - \rho M_2) (I - \lambda M_1)^{-1} (X \beta + M_1 X_1 \gamma) + \varepsilon
\]

(3.4.6)

which suggests another set of instruments

\[
\mathcal{H} = [X \quad \cdots \quad M_1^k X \quad M_2 X \quad \cdots \quad M_2 M_1^k X \quad \cdots \quad M_2^2 M_1^k X]
\]

(3.4.7)

In this case, the typical block of instruments would be \( M_2 M_1^k X \). These will be the instruments used to estimate \( \delta \) in the model given by equation 3.4.2.
For the parameters of the conditional covariance structure ($\theta$), we will follow Lin and Lee (2009); Kelejian and Prucha (2009) and propose moment conditions that allow the consistent estimation of these parameters in the presence of heteroskedastic innovations.

Summing up, we propose to estimate $\phi = [\alpha \ \rho \ \delta]'$ in the general spatial model proposed ($M$) in two steps: (i) first estimating $\delta$ by instrumental variables, then (ii) using the residuals of equation 3.4.1 in a GMM framework to estimate $\theta$. In practice, this procedure could be iterated recursively by either using the consistent estimator $\hat{\theta}$ obtained in the second step to obtain a (a) feasible generalized nonlinear least squares (FGNLS) estimator $\hat{\delta}_{FGNLS}$ in equation 3.4.1, or (b) a feasible generalized IV (FGIV) estimator $\hat{\delta}_{FGIV}$ in equation 3.4.2, potentially even using the "best" instruments in this last case. Sections 3.5 and 3.6 will detail the proposed two step estimation method, showing consistency and asymptotic normality of the proposed estimators. While the procedure draws from the contribution of the above mentioned papers, it is applied to a different model (as argued in section 3.3) and the tools used to show consistency are, in my view, simpler and easier to follow.

3.5 Instrumental Variables Estimator

This section introduces the IV estimator $\hat{\delta}$:

$$
\hat{\delta} = (\Delta'P_H\Delta)^{-1} \Delta'P_HY
$$

(3.5.1)

and the main result are the following propositions:

3.5.1 Proposition (Consistence of IV estimator). Let $H$ be instruments satisfying assumption 3.4.3(v). Let $m_0 \in M$ be the true DGP. Then the IV estimator of $\delta$ defined in equation 3.5.1 is consistent.

3.5.2 Proposition (Asymptotic normality of IV estimator). Assume that 3.4.2, 3.4.3, and 3.4.6 hold. Then

$$
\sqrt{n}(\hat{\delta} - \delta_0) \overset{d}{\to} N\left(0, \lim_{n \to \infty} n (\Delta'P_H\Delta)^{-1} \Delta'P_HY_0P_H\Delta (\Delta'P_H\Delta)^{-1}\right)
$$

where $Y_0 \equiv L_0R_0^{-1}A_0\Sigma_0(\lambda L_0R_0^{-1}A_0)'$.

Before proving these propositions, some lemmas are needed.

3.5.3 Lemma. The elements of $[X \ M_1X_1]$ are bounded.

Proof. By assumptions 3.4.3 item (i) and 3.4.2 item (ii), the elements of $X$ are bounded (therefore the elements of $X_1$, which is a subset of the columns of $X$, are bounded), and matrix $M_1$ is uniformly bounded. It then follows that the elements of $M_1X_1$ are bounded (see lemma A.2.2).

3.5.4 Lemma. The $n \times n$ matrices $L^{-1} = L(\lambda)^{-1} \equiv (I - \lambda M_1)^{-1}$, $A^{-1} = A(\alpha)^{-1} \equiv (I - \alpha M_2)^{-1}$, and $R^{-1} = R(\rho)^{-1} \equiv (I - \rho M_2)^{-1}$ exist for all $\lambda \in \mathcal{L}, \alpha \in \mathcal{A}, \rho \in \mathcal{R}$, can be expanded in a power series, and are uniformly bounded.

Proof. The spectral radius of any matrix (the absolute value of its largest eigenvalue) is bounded by any induced norm, in particular by $\|\cdot\|_1, \|\cdot\|_\infty$, therefore the absolute value of the largest eigenvalue of $M_i, i = 1, 2$ is bounded by 1. Let $\psi$ represent any of $\lambda, \alpha, \rho$. Since the parameter space is restricted to $|\psi| < 1$, it follows that all the eigenvalues of $\psi M_i$ are strictly smaller then 1, which implies the eigenvalues of $I - \psi M_i$ are bounded away from zero, and thus that $(I - \psi M_i)^{-1}$
exists. Since all the eigenvalues of ψM_i are within the unit circle, it follows that the inverse can be expanded into a power series:

\[(I - ψM_i)^{-1} = \sum_{j=1}^{∞} ψ^j M_i^j\]

But, by corollary to lemma A.2.3 (since the matrices are bounded by 1 by assumption 3.4.2), matrices \((M_i)^j, j = 0, \ldots \) are uniformly bounded by 1. Let \(m_{kl}^2\) be an element of \((M_i)^j\). The elements of \(B = (I - ψM_i)^{-1}\) can be written as \(b_{kl} = \sum_{j=0}^{∞} ψ^j m_{kl}^2\). Then, for all columns \(l = 1, \ldots, n\)

\[
\sum_{k=1}^{n} |b_{kl}| = \sum_{k=1}^{n} \sum_{j=0}^{∞} |ψ^j m_{kl}^2| \leq \sum_{k=1}^{n} \sum_{j=0}^{∞} |ψ^j||m_{kl}^2| \\
= \sum_{j=0}^{∞} |ψ^j| \sum_{k=1}^{n} |m_{kl}^2| \quad \text{(since it is a convergent series)} \\
\leq \frac{1}{1 - |ψ|} \quad \text{(since \((M_i)^j\) is uniformly bounded by 1, \(∀j\)}

The same applies to the rows of \(B\), concluding the proof.

Additionally, the following results will be used:

3.5.5 Lemma.

(a) The instruments \(H\) (see assumption 3.4.3 item (v)) are asymptotically uncorrelated with errors in model 3.4.8, \(\text{plim}_{n→∞} \frac{1}{n} H'V = 0\); as a corollary, note that for any \(n × \bar{p}\) matrix \(C\), with bounded elements, \(\text{plim}_{n→∞} \frac{1}{n} C'ε = 0\)

(b) \(S_{H'H} = \lim_{n→∞} \frac{1}{n} H'H\) is finite, positive definite; and

(c) \(S_{H'Δ} = \text{plim}_{n→∞} \frac{1}{n} H'Δ\) exists and has full column rank (i.e., instruments \(H\) are asymptotically correlated with covariates \(Δ\)).

Proof. Item (a): The instrument matrix \(H\) is a linearly independent subset of \(H\) (see assumption 3.4.3), whose columns can be separated in blocks of the form \(M_2^i M_1^i X\). Since \(M_1, M_2\) are uniformly bounded (assumption 3.4.2) and \(X\) has bounded elements, it follows that the elements of such block are also bounded and as a result elements of \(H\) are bounded. Note that \(H'V\) is

\[H'V = H' \underbrace{L_0 R_0^{-1} A_0}_C ε\]

Since \(L_0, A_0, R_0^{-1}\) are uniformly bounded (lemma 3.5.4), lemma A.2.3 assures \(L_0 R_0^{-1} A_0\) is also uniformly bounded. Since \(H\) has bounded elements, lemma A.2.2 assures \(C\) also has bounded elements. Let \(ξ_i^{(k)} \equiv c_{ik}ε_i\). A generic \(k\) element of the vector \(\frac{1}{n} C'ε\) is \(\frac{1}{n} \sum_i ξ_i^{(k)}\). Since the elements of \(C\) are bounded, and by assumption 3.4.6 the innovations \(ε\) are independent, with bounded variance, the conclusion follows from Chebyshev’s form of the LLN (Gnedenko, 1978) for each element of vector \(\frac{1}{n} C'ε\).

Item (b): recall that the columns of \(H\) will be of the form \(M_2^i M_1^k x_k\), where \(x_k\) is in \(k\) column of matrix \(X\). Therefore, the typical \(\{i,j\}\) element of the \(\bar{p} × \bar{p}\) matrix \(H'H\) is \(\sum_{m=1}^{n} \sum_{l=1}^{∞} b_{m,l} x_m x_l\), where \(b_{kl}\) is the element of the \(n × n\) uniformly bounded matrix \(B = (M_2^i M_1^i)' M_2^i M_1^i\). By assumption 3.4.3 item (i) the elements of \(X\) are bounded, thus there exists
\( c_1 < \infty \) such that \(|x_{ij}| < c_1\) for all \(i = 1, \ldots, n, j = 1, \ldots, k, \forall n\). Also, since \(B\) is uniformly bounded, there is \(c_2 < \infty\) such that \(\sum_{j=1}^{n}|b_{ij}| < c_2\) for all \(i = 1, \ldots, n, \forall n\). Thus

\[
\sum_{m=1}^{n} \sum_{l=1}^{n} b_{ml}x_{mi}x_{lj} \leq \sum_{m=1}^{n} \sum_{l=1}^{n} |b_{ml}||x_{mi}||x_{lj}|
\]

\[
\leq c_1^2 \sum_{m=1}^{n} \sum_{l=1}^{n} |b_{ml}| \quad \text{(since } X \text{ is bounded)}
\]

\[
\leq c_1^2 \sum_{m=1}^{n} c_2 \quad \text{(since } B \text{ is uniformly bounded)}
\]

\[
= \tilde{c}n
\]

therefore each element of \(H'H\) is \(O(n)\) and the conclusion \(S_{H'H} = O(1)\) follows. The fact that this limit is positive definite follows from the full rank assumption of \(H\) (3.4.3 item (v)).

Item (c): Note again that each column of the instruments \(H\) takes the following form, \(M_2^2 M_1^2 z_{ik}\). Therefore, each \(k\) line of \(H'\Delta\) is given by the following row vector

\[
[x'_{ik} M_1^2 M_2^2 M_1 Y \quad x'_{ik} M_1^2 M_2^2 X \quad x'_{ik} M_1^2 M_2^2 M_1 X_1]
\]

where the \(n \times n\) uniformly bounded matrix \(B\) depends on \(a, b\). Then,

\[
\lim_{n \to \infty} \frac{1}{n} (H'\Delta)_{k} = \lim_{n \to \infty} \left[ \frac{1}{n} x'_{ik} M_1^2 M_2^2 M_1 Y \quad \frac{1}{n} x'_{ik} M_1^2 M_2^2 X \quad \frac{1}{n} x'_{ik} M_1^2 M_2^2 M_1 X_1 \right]
\]

where

\[
\lim_{n \to \infty} \frac{1}{n} x'_{ik} BM_{1} Y = \lim_{n \to \infty} \frac{1}{n} x'_{ik} BM_{1} L_{0}^{-1} X \beta_{0} + \lim_{n \to \infty} \frac{1}{n} x'_{ik} BM_{1} L_{0}^{-1} M_{1} X_{1} \gamma_{0}
\]

\[
+ \lim_{n \to \infty} \frac{1}{n} x'_{ik} BM_{1} R_{0}^{-1} A_{0} \varepsilon
\]

\[
= \left( S_{X'BM_{1}L_{0}^{-1}X} \right)_{k} \beta_{0} + \left( S_{X'BM_{1}L_{0}^{-1}M_{1}X_{1}} \right)_{k} \gamma_{0} + 0
\]

which is \(O(1)\) provided \(|\beta_{0}| < \infty\) and \(|\gamma_{0}| < \infty\). As a result, each line of \(S_{H'\Delta}\) can be written in the following form:

\[
(S_{H'\Delta})_{k} = \left[ \left( S_{X'BM_{1}L_{0}^{-1}X} \right)_{k} \beta_{0} + \left( S_{X'BM_{1}L_{0}^{-1}M_{1}X_{1}} \right)_{k} \gamma_{0} \right] (S_{X'BX})_{k} (S_{X'BM_{1}X_{1}})_{k}
\]

The full column rank is assumption 3.4.3 item (v). Note that item (iv) of assumption 3.4.4 rules out DGPs in which \(\beta_{0} = 0, \gamma_{0} = 0\) (i.e., the case with no covariates at all), since in this case the first column of \(S_{H'\Delta}\) would be null. But if this were the case, the conditional covariance parameters could be consistently estimated by GMM (see section 3.6). Also, when \(\gamma_{0} = 0, A_{0} = 0\) (i.e., \(L_{0}^{-1} = I\), the case where no spatial effect is present in the covariates), the first column would be a linear combination of the last column plus a linear combination of the deterministic covariates. So, depending on the true DGP (for instance, one in which the parameters of the non-stochastic regressors are null) it could be unfeasible to obtain full column rank for the matrix \(S_{H'\Delta}\). In this case, however, \(\beta_{0}\) could be consistently estimated by OLS (see Kelejian and Prucha, 1998, for details).

The following lemma is required to shown asymptotic normality of the IV estimator:

3.5.6 Lemma. The term \(\frac{1}{\sqrt{n}} H'V\) follows a central limit theorem.
Proof. Recall from lemma 3.5.5 item (a) that $\frac{1}{\sqrt{n}}\mathbf{H}' \mathbf{V} = \frac{1}{\sqrt{n}} \mathbf{C}' \mathbf{\varepsilon}$, where $\mathbf{C}$ is an $p \times p$ non-stochastic matrix with bounded elements. Let $\xi_i^{(p)} \equiv c_{ip} \mathbf{\varepsilon}_i$, then a generic $p$ element of the vector $\frac{1}{\sqrt{n}} \mathbf{C}' \mathbf{\varepsilon}$ is $\frac{1}{\sqrt{n}} \sum_{i=1}^{n} \xi_i^{(p)}$. It follows that, for each $p$, $\xi_i$ is a sequence of independent random variables with finite variance, since the innovations are independent and have finite variance, and the elements of $\mathbf{C}$ are bounded. $E \xi_i = 0$, and $\text{Var}(\xi_i) = E \xi_i^2 = c_{ip}^2 \sigma_i^2$. Given the assumption on innovations (see 3.4.6) assures the existence of finite moments of order four, in order to assure that Lyapunov's condition is satisfied, one needs $s_{n,1} = \left( \sum_{i=1}^{n} \text{Var}(\xi_i) \right)^{1/2}$ to diverge faster than $n$. But it is sufficient that only a finite number of $c_{ip}$ be different from zero. As a result, and recalling that $\mathbf{H}$ has full rank,

$$\frac{1}{\sqrt{n}} \mathbf{H}' \mathbf{V} \xrightarrow{d} \mathcal{N} \left( 0, \lim_{n \to \infty} \frac{1}{n} \mathbf{H}' \mathbf{L}_0 \mathbf{R}_0^{-1} \mathbf{A}_0 \mathbf{\Sigma}_0 \left( \mathbf{L}_0 \mathbf{R}_0^{-1} \mathbf{A}_0 \right)' \mathbf{H} \right)$$

The above lemmata allows to prove propositions 3.5.1 and 3.5.2 stated at the onset of this section.

Proof of proposition 3.5.1. Let $\hat{\delta}$ be the IV estimator for $\delta_0$, with instruments $\mathbf{H}$. Denote $\mathbf{P}_H$ the orthogonal projection into the subspace generated by $\mathbf{H}$. Note that

$$\lim_{n \to \infty} \frac{1}{n} \Delta' \mathbf{P}_H \Delta = \lim_{n \to \infty} \frac{1}{n} \Delta' \mathbf{H} \lim_{n \to \infty} \left( \frac{1}{n} \mathbf{H}' \mathbf{H} \right)^{-1} \lim_{n \to \infty} \frac{1}{n} \mathbf{H}' \Delta$$

thus from lemma 3.5.5 we conclude that $\lim_{n \to \infty} \frac{1}{n} \Delta' \mathbf{P}_H \Delta$ exists and is positive definite, since each term above exists and is positive definite. Thus

$$\lim_{n \to \infty} \hat{\delta} = \lim_{n \to \infty} \left( \frac{1}{n} \Delta' \mathbf{P}_H \Delta \right)^{-1} \Delta' \mathbf{P}_H (\Delta \delta_0 + \mathbf{V})$$

$$= \delta_0 + \lim_{n \to \infty} \left( \frac{1}{n} \Delta' \mathbf{P}_H \Delta \right)^{-1} \Delta' \mathbf{H} (\mathbf{H}' \mathbf{H})^{-1} \mathbf{H}' \mathbf{V}$$

$$\lim_{n \to \infty} \left( \frac{1}{n} \Delta' \mathbf{P}_H \Delta \right)^{-1} \lim_{n \to \infty} \left( \frac{\Delta' \mathbf{H}}{n} \right) \lim_{n \to \infty} \left( \frac{\mathbf{H}' \mathbf{H}}{n} \right)^{-1} \lim_{n \to \infty} \left( \frac{\mathbf{H}' \mathbf{V}}{n} \right)$$

$$= 0$$

where the last steps follow again from lemma 3.5.5.

Proof of proposition 3.5.2. From proposition 3.5.1

$$\sqrt{n} \left( \hat{\delta} - \delta_0 \right) = \left( \frac{\Delta' \mathbf{P}_H \Delta}{n} \right)^{-1} \frac{\Delta' \mathbf{P}_H \mathbf{V}}{\sqrt{n}}$$

where

$$\frac{\Delta' \mathbf{P}_H \mathbf{V}}{\sqrt{n}} = \frac{\Delta' \mathbf{H}}{n} \left( \frac{\mathbf{H}' \mathbf{H}}{n} \right)^{-1} \frac{\mathbf{H}' \mathbf{V}}{\sqrt{n}}$$

Since the first two terms in the right-side of the above converge in probability to well-defined matrices, and by lemma 3.5.6 the last term is asymptotically normal, it follows that $\frac{\Delta' \mathbf{P}_H \mathbf{V}}{\sqrt{n}}$ is asymptotically normal with covariance matrix

$$\lim_{n \to \infty} \frac{1}{n} \Delta' \mathbf{P}_H \mathbf{L}_0 \mathbf{R}_0^{-1} \mathbf{A}_0 \mathbf{\Sigma}_0 \left( \mathbf{L}_0 \mathbf{R}_0^{-1} \mathbf{A}_0 \right)' \mathbf{P}_H \Delta$$

Therefore,

$$\sqrt{n} \left( \hat{\delta} - \delta_0 \right) \xrightarrow{d} \mathcal{N} \left( 0, \lim_{n \to \infty} \frac{1}{n} \Delta' \mathbf{P}_H \mathbf{\Upsilon}_0 \mathbf{P}_H \Delta \left( \Delta' \mathbf{P}_H \Delta \right)^{-1} \right)$$
3.6 Method of Moments Estimator

Let \( m_0 \in \mathbb{M} \) be the true data generating process. The second step of the proposed estimating procedure, after the IV estimator \( \hat{\delta} \) has been estimated, is to use the residuals \( \hat{U} = Y - f(\hat{\delta}) \) to estimate the remaining parameters \( \theta_0 = [\alpha_0 \ \rho_0] \). Note that the error term \( U \) in model \( \mathbb{M} \) follows a spatial moving average, autoregressive process with heteroskedastic innovations:

\[
U = \rho M_{2U} U + \varepsilon - \alpha M_{2\varepsilon}
\]

\[
E \varepsilon = 0, \quad E \varepsilon \varepsilon' = \Sigma \text{ (diagonal)}
\]  

(3.6.1)

Let’s call this model \( M_{ARMA} \subseteq \mathbb{M} \). Before proposing a GMM estimator for the parameter \( \theta_0 \), recall that the value of \( U \) is not observed, instead we observe a consistently estimated residual \( \hat{U} \). The following remark characterizes \( U \) and \( \hat{U} \):

Remark. The true error \( U \) from model \( \mathbb{M} \) can be written as

\[
u_i = h_i(\delta_0; M_1, X, y_i) \equiv y_i - f_i(\delta_0; M_1, X), \ i = 1, \ldots, n
\]  

(3.6.2)

where \( f(\delta; M_1, X) = (I - \lambda M_1)^{-1}(X \beta + M_1 X_1 \gamma) \), \( X \) is an \( n \times k \) matrix of regressors, \( M_1 \) is an \( n \times n \) matrix of known constants representing the spatial dependence structure in the covariates, and \( \delta_0 = [\lambda_0 \ \beta_0 \ \gamma_0] \) is a parameter vector representing the true DGP \( m_0 \) (see assumptions in section 3.4.1). Define \( D \equiv \mathcal{L} \times \mathcal{B} \times \mathcal{G} \) and note that it is a compact set (assumption 3.4.4). The functions \( h_i(\cdot; M_1, X, y_i) : D \to \mathbb{R} \) are twice continuously differentiable on \( D \) for each \( M_1, X, y_i \).

Let

\[
\tilde{h}_i(M_1, Z, y_i) = \sup_{\delta \in D} \left\{ h_i(\delta; M_1, X, y_i), \left| \frac{\partial h_i(\delta; M_1, X, y_i)}{\partial \delta} \right|, \left| \frac{\partial^2 h_i(\delta; M_1, X, y_i)}{\partial \delta \partial \delta'} \right| \right\}
\]  

(3.6.3)

and note that, for all \( n, \sup_{1 \leq i \leq n} E(\tilde{h}_i^2) \leq \infty \) (Kelejian and Prucha, 2001). The consistent estimator of the residual is therefore

\[
\hat{U} = h(\hat{\delta}; \ M_1, X, Y) = Y - f(\hat{\delta}; \ M_1, X)
\]  

(3.6.4)

where \( \hat{\delta} \) is a \( \sqrt{n} \) consistent estimator of the true \( \delta_0 \), i.e., \( \sqrt{n}(\hat{\delta} - \delta_0) = O_p(1) \).

Now, the proposed GMM estimator is defined:

3.6.1 Definition. the GMM estimator of \( \theta_0 \) is given by

\[
\hat{\theta} = \arg\max_{\theta \in A \times R} \hat{Q}_n(\theta)
\]  

(3.6.5)

where the objective function is

\[
\hat{Q}_n(\theta) = -\frac{1}{2} \hat{g}_n(\theta)' \hat{W}_n \hat{g}_n(\theta)
\]  

(3.6.6)

and the moment conditions are defined by the function \( \hat{g}_n : A \times R \to \mathbb{R}^k \)

\[
\hat{g}_n(\theta) = \frac{1}{n} \left[ \begin{array}{c}
\hat{U}'(A^{-1}_\alpha R_\rho)'K_1 A^{-1}_\alpha R_\rho \hat{U} \\
\vdots \\
\hat{U}'(A^{-1}_\alpha R_\rho)'K_k A^{-1}_\alpha R_\rho \hat{U}
\end{array} \right]
\]  

(3.6.7)

where \( \hat{W}_n \), and \( K_i, i = 1, \ldots, l \) satisfy assumption 3.4.5.

The consistency and asymptotic normality of the proposed GMM estimator \( \hat{\theta} \) is shown in the following sections. The proof relies on showing that conditions of theorems 2.1 and 3.2 in Newey and McFadden (1994) are satisfied for \( \theta \) when the actual DGP \( m_0 \) belongs to the family of model \( \mathbb{M} \) (see sections 3.6.3, 3.6.4). To get there, however, some lemmas and definitions will be needed. The following lemmas are adapted from Kelejian and Prucha (2001, 2009); Lee (2007).
3.6.2 Lemma. Let $B$ be any $n \times n$ non-stochastic matrix. Then for $\varepsilon$ as defined in assumption 3.4.6,

(a) $E(\varepsilon' B \varepsilon) = \text{tr}(\Sigma B)$, and

(b) $\text{Var}(\varepsilon' B \varepsilon) = \text{tr}(\Sigma B \Sigma B^*) + \sum_{i=1}^{n} b_{ii}^2 \left[ E(\varepsilon_i^4) - 3 \left( E(\varepsilon_i^2) \right)^2 \right]$, where $B^* \equiv B + B'$.

Proof. For the expected value

$$E(\varepsilon' B \varepsilon) = E \left( \sum_{i=1}^{n} \sum_{j=1}^{n} b_{ij} \varepsilon_i \varepsilon_j \right) = \sum_{i=1}^{n} b_{ii} E(\varepsilon_i^2) = \text{tr}(\Sigma B)$$

since $\varepsilon_i, \varepsilon_j$ are independent for $i \neq j$

For the variance, write first the second non-centered moment:

$$E(\varepsilon' B \varepsilon' B \varepsilon) = E \left( \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} \sum_{l=1}^{n} b_{ij} b_{kl} \varepsilon_i \varepsilon_j \varepsilon_k \varepsilon_l \right)$$

$$= \sum_{i=1}^{n} b_{ii}^2 E(\varepsilon_i^4) + \sum_{i=1}^{n} \sum_{j=1}^{n} b_{ij} b_{ij} \sigma_i^2 \sigma_j^2 + \sum_{i=1}^{n} \sum_{j=1}^{n} b_{ij} b_{ij} \sigma_j^2 \sigma_j^2 + \sum_{i=1}^{n} \sum_{j=1}^{n} b_{ij} b_{ij} \sigma_i^2 \sigma_j^2$$

$$= \sum_{i=1}^{n} b_{ii}^2 \left( E(\varepsilon_i^4) - 3 \sigma_i^4 \right) + \sum_{i=1}^{n} \sum_{j=1}^{n} (b_{ij} b_{ij} + b_{ij} b_{ij} + b_{ij} b_{ij}) \sigma_i^2 \sigma_j^2$$

$$= \sum_{i=1}^{n} b_{ii}^2 \left( E(\varepsilon_i^4) - 3 \sigma_i^4 \right) + \text{tr}(\Sigma B \Sigma B^*) + \text{tr}(\Sigma B)$$

therefore,

$$\text{Var}(\varepsilon' B \varepsilon) = \sum_{i=1}^{n} b_{ii}^2 \left( E(\varepsilon_i^4) - 3 \sigma_i^4 \right) + \text{tr}(\Sigma B \Sigma B^*)$$

\[\Box\]

3.6.3 Lemma. If $B$ is an $n \times n$ non-stochastic uniformly bounded matrix, and $\varepsilon$ satisfy assumption 3.4.6, then $E(\varepsilon' B \varepsilon) = O(n)$, $\text{Var}(\varepsilon' B \varepsilon) = O(n)$, $\frac{1}{n} E' B \varepsilon = O_p(1)$, and $\lim_{n \to \infty} \frac{1}{n} E' B \varepsilon = \lim_{n \to \infty} \frac{1}{n} \text{tr}(\Sigma B)$.

Proof. Assumption 3.4.6 implies $\Sigma$ is an uniformly bounded matrix. Product of uniformly bounded matrices is uniformly bounded (lemma A.2.3) and trace of uniformly bounded matrices is $O(n)$ (lemma A.2.4), therefore $\text{tr}(\Sigma B)$ and $\text{tr}(\Sigma B \Sigma B^*)$ are $O(n)$. Since $E \varepsilon_i^4$ is bounded for all $i$, $\sum_{i=1}^{n} b_{ii}^2 \left( E(\varepsilon_i^4) - 3 \sigma_i^4 \right)$ is also $O(n)$. Therefore $\lim_{n \to \infty} \text{Var}(\frac{1}{n} E' B \varepsilon) = 0$, and this implies convergence in mean square to a non-stochastic limit, $\lim_{n \to \infty} \frac{1}{n} \text{tr}(\Sigma B) = 0$, and therefore implies convergence in probability to this same limit. Since $\frac{1}{n} E' B \varepsilon$ converges in probability to a non-stochastic, finite limit, it is bounded in probability, i.e., it is $O_p(1)$.

\[\Box\]

---

Note that there is an error in this formula in Lin and Lee (2009); it does not compromise the conclusion of their paper, but it might render an error if one tries to implement their formulas.
Remark: If $B$ has zero diagonal, then the conclusion of the previous lemma follows without the assumption of bounded fourth moments.

3.6.4 Lemma. Consider a DGP $m_0 \in \mathbb{M}$ (equation 3.4.1), assume assumption 3.4.6 holds, and let $\hat{U}$ be defined as in equation 3.6.4. Let $B$ be an $n \times n$ uniformly bounded matrix. Then

$$\lim_{n \to \infty} \frac{1}{\sqrt{n}} \hat{U}'B\hat{U} = \lim_{n \to \infty} \frac{1}{\sqrt{n}} U'BU$$

Proof. This proof is analogous to the one in Kleijer and Prucha (2001). Since the functions $h_i(\delta), i = 1, \ldots, n$ (defined in equation 3.6.2) are twice continuously differentiable, it follows that $h(\delta)'Bh(\delta)$ also is twice continuously differentiable. Therefore, using a second order Taylor expansion to $n^{-1/2} \hat{U}'B\hat{U} = n^{-1/2}h(\delta)'Bh(\delta)$:

$$n^{-1/2} \hat{U}'B\hat{U} = \frac{1}{\sqrt{n}}U'BU + \frac{1}{\sqrt{n}} U'B \frac{\partial h(\delta_0)}{\partial \delta}(\hat{\delta} - \delta_0) + \frac{1}{2n}(\hat{\delta} - \delta_0)' \frac{\partial^2 h(\delta')Bh(\delta')}{\partial \delta \partial \delta'}(\hat{\delta} - \delta_0)$$

where $\delta^*$ is a convex combination of $\delta_0$ and $\hat{\delta}$. We need to show that a.1 and a.2 converge in probability to zero.

Item (a.1):

$$a.1 = \frac{1}{\sqrt{n}}(\hat{\delta} - \delta_0)' \left( \frac{\partial h(\delta_0)}{\partial \delta} \right)' B^*U$$

$$= -\sqrt{n}(\hat{\delta} - \delta_0)' \frac{1}{n} \left( \frac{\partial f(\delta_0)}{\partial \delta} \right)' B^*R_0^{-1}A_0 \varepsilon$$

$$= -\sqrt{n}(\hat{\delta} - \delta_0)' \frac{1}{n} \left( B' \frac{\partial f(\delta_0)}{\partial \delta} \right)' \varepsilon$$

thus, if $\psi_1 \equiv o_p(1)$ the conclusion $a.1 = o_p(1)$ follows. But note that

$$\frac{\partial f(\delta_0)}{\partial \delta} = [(I - \lambda_0 M_1)^{-1} M_1 (X \beta_0 + M_1 X_1 \gamma_0) - (I - \lambda_0 M_1)^{-1} X (I - \lambda_0 M_1)^{-1} M_1 X_1]$$

is a non-stochastic matrix with bounded elements (lemma A.2.2), and $B$ is uniformly bounded. Therefore the result $\lim_{n \to \infty} \psi_1 = 0$ follows from item (a) in lemma 3.5.5 (Chebyshev’s LLN).

Item (a.2):

$$a.2 = \frac{1}{2n}(\hat{\delta} - \delta_0)' \frac{\partial^2 h(\delta')Bh(\delta')}{\partial \delta \partial \delta'}(\hat{\delta} - \delta_0)$$

$$= \frac{1}{2n} \sum_{l=1}^{p+1} \sum_{m=1}^{p+1} (\hat{\delta}_l - \delta_{l,0})(\hat{\delta}_m - \delta_{m,0}) \frac{\partial^2 h(\delta')Bh(\delta')}{\partial \delta_m \partial \delta_l}$$

$$= \frac{1}{2n} \sum_{l=1}^{p+1} \sum_{m=1}^{p+1} \sqrt{n}(\hat{\delta}_l - \delta_{l,0})(\hat{\delta}_m - \delta_{m,0}) \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{n} b_{ij} \left[ \frac{\partial h_i(\delta')}{\partial \delta_m} \frac{\partial h_j(\delta')}{\partial \delta_l} + h_i(\delta') \frac{\partial^2 h_i(\delta')}{\partial \delta_m \partial \delta_l} \right]$$

therefore, recalling equation 3.6.3, we have

$$|a.2| \leq \frac{1}{2n} \sum_{l=1}^{p+1} \sum_{m=1}^{p+1} \sqrt{n}(\hat{\delta}_l - \delta_{l,0})(\hat{\delta}_m - \delta_{m,0}) \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{n} |b_{ij}| \bar{h}_i \bar{h}_j \psi_{2n}$$
and thus, in order to show that \(a.2 = o_p(1)\) it is enough to show that \(\psi_{2n}\) is bounded in probability, i.e., \(\psi_{2n} = O_p(1)\). But, since \(B\) is uniformly bounded and \(E(\tilde{h}_i^2) < \infty\) for all \(i = 1, \ldots, n\) and all \(n\), there is a constant \(\tilde{c} < \infty\), independent of \(n\), such that \(\sum_{j=1}^{n} |b_{ij}| < \tilde{c}\) and \(E(\tilde{h}_i^2) < \tilde{c}\). Thus

\[
E \psi_{2n} = \frac{4}{n} \sum_{i=1}^{n} \sum_{j=1}^{n} |b_{ij}| E(\tilde{h}_i \tilde{h}_j)
\]

\[
\leq \frac{4}{n} \sum_{i=1}^{n} \sum_{j=1}^{n} |b_{ij}| \left( E(\tilde{h}_i^2) \right)^{\frac{1}{2}} \left( E(\tilde{h}_j^2) \right)^{\frac{1}{2}} \quad \text{(Hölder's inequality)}
\]

\[
\leq \frac{4\tilde{c}}{n} \sum_{i=1}^{n} \sum_{j=1}^{n} |b_{ij}| \leq \frac{4\tilde{c}}{n} \tilde{c} = 4\tilde{c}^2 < \infty
\]

and therefore \(\psi_{2n}\) is bounded in probability (see lemma 3.6.14 and recall that \(\psi_{2n}\) is a positive random variable).

The above lemmata will be used to show that the empirical moments converge in probability to well-defined non-stochastic continuous functions.

**3.6.5 Proposition.** The empirical moments converge in probability to a non-stochastic continuous function for all \(\theta \in A \times \mathcal{R}:

\[
\text{plim}_{n \to \infty} \hat{\theta}_n(\theta) = \begin{bmatrix}
\lim_{n \to \infty} \frac{1}{n} \text{tr} \left( \Sigma_0 \left( A_{\alpha}^{-1} R_{\rho} R_0^{-1} A_0 \right)' K_i A_{\alpha}^{-1} R_{\rho} R_0^{-1} A_0 \right) \\
\vdots \\
\lim_{n \to \infty} \frac{1}{n} \text{tr} \left( \Sigma_0 \left( A_{\alpha}^{-1} R_{\rho} R_0^{-1} A_0 \right)' K_i A_{\alpha}^{-1} R_{\rho} R_0^{-1} A_0 \right)
\end{bmatrix} \equiv g_0(\theta)
\]

and \(g_0(\theta_0) = 0\).

**Proof.** Consider any \(i\) element of the empirical moments vector

\[
\text{plim}_{n \to \infty} \hat{\theta}_{n,i}(\theta) = \text{plim}_{n \to \infty} \frac{1}{n} \text{tr} \left( \Sigma_0 B_i(\theta) \right)
\]

\[
= \text{plim}_{n \to \infty} \frac{1}{n} \text{tr} \left( \Sigma_0 B_i(\theta) \right) \equiv g_{0,i}(\theta_0)
\]

since this holds for all \(i = 1, \ldots, l\), the conclusion follows. The last statement also holds by noting that, since \(K_i\) has zeros in its diagonal:

\[
g_{0,i}(\theta_0) = \lim_{n \to \infty} \frac{1}{n} \text{tr} \left( \Sigma_0 B_i(\theta_0) \right) = \lim_{n \to \infty} \frac{1}{n} \text{tr} \left( \Sigma_0 K_i \right) = 0, \forall i = 1, \ldots, l
\]

**Corollary.** By continuity of multiplication, \(Q_0(\theta) \equiv \text{plim}_{n \to \infty} \hat{Q}_n(\theta) = -\frac{1}{2} g_0(\theta)' W g_0(\theta)\).

**3.6.1 Identification**

One of the important assumptions to show consistency of \(\hat{\theta}\) is that, for a given DGP \(m_0 \in M\), the the limiting function \(Q_0(\theta)\) has a unique maximum at \(\theta_0\). The sections below show the conditions needed for the model \(M_{ARMA}\) to be identified.
M_{AR} Case

Consider first the case of an autoregressive spatial error term, i.e., consider model M_{AR} \subset M_{ARMA}, defined by imposing \alpha = 0 on M_{ARMA}. The parameter vector then reduces to a scalar \rho. Define \bar{\rho} = \rho - \rho_0 and note that \((I - \rho M_2)(I - \rho_0 M_2)^{-1} = I - \bar{\rho} M_2(I - \rho_0 M_2)^{-1}\). Substituting into the moment conditions \(g_{0,i}(0, \rho)\) (equation 3.6.7), yields:

\[
g_{0,i}(0, \rho) = \lim_{n \to \infty} \frac{1}{n} \text{tr} \left[ \Sigma_0 (I - \bar{\rho} R_0^{-1} M_2') K_i (I - \bar{\rho} M_2 R_0^{-1}) \right]
\]

\[
= \lim_{n \to \infty} \frac{1}{n} \left[ \text{tr} \Sigma_0 K_i - \bar{\rho} \text{tr} \left( \Sigma_0 (K_i + K_i') M_2 R_0^{-1} \right) + \bar{\rho}^2 \text{tr} \left( \Sigma_0 R_0^{-1} M_2' K_i M_2 R_0^{-1} \right) \right]
\]

Therefore, the equation \(g_{0,i}(0, \rho) = 0\) has two real roots, \(\bar{\rho}_{i,1}\) and \(\bar{\rho}_{i,2}\). The first root is \(\bar{\rho}_{i,1} = 0\) which implies \(\rho = \rho_0\) is a solution for all \(i = 1, \ldots, l\) moment equations. The second root is \(\bar{\rho}_{i,2} = \lim_{n \to \infty} \frac{\text{tr}(\Sigma_0 K_i M_2 R_0^{-1})}{\text{tr}(\Sigma_0 R_0^{-1} M_2' M_2 R_0^{-1})}\). Thus, at least \(l = 2\) moment conditions are required to identify the true parameter \(\rho_0\), and one needs to be sure that the quadratic polynomials in \(\bar{\rho}\) which result from these moment conditions share only the null root. This leads to the following proposition:

3.6.6 Proposition (Asymptotic identification (global)). The spatial autoregressive model M_{AR}, whose moment conditions are obtained from equation 3.6.7 by setting \(\alpha = \alpha_0 = 0\), is asymptotically identified if there exists at least two \(i_1, i_2\) such that \(\bar{\rho}_{i_1,2} \neq \bar{\rho}_{i_2,2}\).

\[ \text{Proof.} \text{ If the condition is satisfied, } g_0(0, \rho_0) = 0 \text{ (since } \rho_0 \text{ is a root for all } i = 1, \ldots, l \text{ moment equations) and } g_0(0, \rho) \neq 0, \forall \rho \in \mathcal{R}, \rho \neq \rho_0. \text{ By Cholesky decomposition, } W = U^T U, \text{ where } U \text{ is upper triangular with positive diagonal entries (full rank). Therefore for } \rho \neq \rho_0, U g_0(0, \rho) \neq 0 \text{ and thus } Q_0(0, \rho) = -(U g_0(0, \rho))^T U g_0(0, \rho) < Q_0(0, \rho_0) = 0. \]

The condition for asymptotic identification of spatial autoregressive model M_{AR} is equivalent to the identification condition on Kelejian and Prucha (2009); Lin and Lee (2009), which are also equivalent between themselves.

To illustrate, consider an homoskedastic unit variance model \((\Sigma_0 = I)\) with spatial weight matrix \(M_2\) given by the typical circular world where each spatial unit has two neighbors, and for each neighbor the equivalent element of matrix \(M_2\) is 0.5 (\(||M_2||_1 = 1\)). The parameter space for \(\rho\) can be defined as any closed interval, containing zero, such that \(\mathcal{R} \subset (-1, 1)\). Using Kelejian and Prucha’s proposed two moment conditions based on \(K_1 = M_2\) and \(K_2 = M_2^2 - \text{diag}(M_2^2)\), the limiting function of the moment conditions is:

\[
g_0(0, \rho) = \begin{bmatrix} \lim_{n \to \infty} \frac{1}{n} \text{tr} \left[ (R_0 R_0^{-1})' M_2 R_0^{-1} \right] \\ \lim_{n \to \infty} \frac{1}{n} \text{tr} \left[ (R_0 R_0^{-1})' (M_2^2 - \text{diag}(M_2^2)) R_0 R_0^{-1} \right] \end{bmatrix} = \begin{bmatrix} g_{0,1}(\rho) \\ g_{0,2}(\rho) \end{bmatrix}
\]

where we note that the convergence happens quickly, i.e., the asymptotic values are obtained for \(n\) as small as 20.\(^{10}\) Table 3.3 shows, for some examples of DGP’s \(m_0 \in M_{AR}\) (represented by different values for \(\rho_0\) in the first and fourth column), the second root of each moment equation \(g_{0,1}(\rho)\) and \(g_{0,2}(\rho)\), i.e., the root that is \textit{not equal} to the true \(\rho_0\).

Some points are worth mentioning in this example: (a) the moment equations \(g_{0,1}\) and \(g_{0,2}\) are more similar (similar roots) the closer the true parameter \(\rho_0\) is (in absolute value) to the bounds of the parameter space, (b) the roots that are different from the true \(\rho_0\) might lie either inside or outside the parameter space, therefore using only one moment condition carries the risk

\(^{10}\) Except for the fact that Lin and Lee consider a generic set of \(l\) moment conditions and Kelejian and Prucha works with two specific moment conditions, the assumptions are equivalent.

\(^{11}\) This is due to the well-behaved form of the spatial weight matrix \(M_2\), which is a circulant matrix.
of identifying the wrong parameter, even when the search is restricted to the interval \( \mathcal{R} \). Further analysis is required, but this simple example shows that it is important to choose carefully the matrices \( K_i \) and also to restrict the search for the solutions of the moment equations to the proper parameter space. This contrasts to the observation made in Kelejian and Prucha (2009) where the authors suggest the search for \( \rho \) be “any compact interval that contains the true parameter space”. In the (admittedly extreme) case the true parameter is \( \rho_0 = 0.99 \), for instance, one could wrongly identify 1.02 as the true parameter since it satisfies both moment equations (if precision of root finding algorithm is low).

In Lee (2007) it is shown that, for the case of homoskedastic error terms, there exists a “best” moment condition, and it is given by the matrix \( K = M_2(I - \rho_0M_2)^{-1} \). The author claims that this one moment condition is best relative to any finite number of \( K_i \)’s. To illustrate, consider Lee’s additional moment condition

\[
g_{0,3}(\rho) = \lim_{n \to \infty} \frac{1}{n} \text{tr} \left( R_p R_0^{-1} \right)' \left( M_2 R_0^{-1} - \text{diag}(M_2 R_0^{-1}) \right) R_p R_0^{-1}
\]

which is plotted (along with \( g_{0,1}, g_{0,2} \)) in figure 3.1. It illustrates that, in this example, any of the three moment conditions could be used alone to identify the model, provided the parameter space is restricted to \( \mathcal{R} \). Otherwise, the researcher would not be able to chose between the two real roots of the individual moment equation. This also shows that, although best from a variance perspective, using only one moment condition (as proposed in Lee (2007)) is flawed, since it does not guarantee asymptotic identification. Also interesting to note that, even though global identification is assured in all the four cases shown in figure 3.1, in practice, if the true \( \rho_0 \) is near the bounds of \( \mathcal{R} \), the moment equations (depending on the precision of the root finding algorithm) could potentially point to the wrong model, as it would be the case if one were using only moment equations \( g_{0,2} \) and \( g_{0,3} \) when \( \rho_0 = 0.9 \) – see the fourth chart in figure 3.1. This goes against the recommendation in Kelejian and Prucha (2009) and highlights the importance of properly considering the parameter space in the optimization routines.

Table 3.3: Roots of moment equations

<table>
<thead>
<tr>
<th>( \rho_0 )</th>
<th>( \rho_{1.2} )</th>
<th>( \rho_{2.2} )</th>
<th>( \rho_0 )</th>
<th>( \rho_{1.2} )</th>
<th>( \rho_{2.2} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.98</td>
<td>-1.049</td>
<td>-1.039</td>
<td>0.98</td>
<td>1.049</td>
<td>1.039</td>
</tr>
<tr>
<td>-0.80</td>
<td>-1.455</td>
<td>-1.189</td>
<td>0.80</td>
<td>1.455</td>
<td>1.189</td>
</tr>
<tr>
<td>-0.60</td>
<td>-2.077</td>
<td>-1.182</td>
<td>0.60</td>
<td>2.077</td>
<td>1.182</td>
</tr>
<tr>
<td>-0.40</td>
<td>-3.241</td>
<td>-0.983</td>
<td>0.40</td>
<td>3.241</td>
<td>0.983</td>
</tr>
<tr>
<td>-0.20</td>
<td>-6.622</td>
<td>-0.569</td>
<td>0.20</td>
<td>6.622</td>
<td>0.569</td>
</tr>
<tr>
<td>0</td>
<td>+\infty</td>
<td>0</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
\( \mathbb{M}_M \) Case

Consider the case of a spatial moving average process for the error term, i.e., consider model \( \mathbb{M}_M \subset \mathbb{M}_{ARMA} \) defined by imposing \( \rho = 0 \) on \( \mathbb{M}_{ARMA} \). The parameter vector then reduces to a scalar \( \alpha \). However, differently from the spatial autoregressive stochastic process, a global identification condition is difficult to postulate for model \( \mathbb{M}_M \). To overcome this problem, we state the condition needed for local identification (see Newey and McFadden, 1994).

3.6.7 Assumption (Local asymptotic identification). Assume that for some \( i \in \{1, \ldots, l\} \) the following condition holds

\[
\lim_{n \to \infty} \frac{1}{n} \text{tr} \Sigma_0 K_i^* A_0^{-1} M_2 \neq 0
\]

With this assumption, the proposition below shows that model \( \mathbb{M}_M \) is asymptotically locally identified.

3.6.8 Proposition (Local asymptotic identification). The spatial moving average model \( \mathbb{M}_M \), whose moment conditions are obtained from equation 3.6.7 by setting \( \rho = \rho_0 = 0 \), is asymptotically locally identified if there exists \( i \in \{1, \ldots, l\} \) such that

\[
\lim_{n \to \infty} \frac{1}{n} \text{tr} \Sigma_0 K_i^* A_0^{-1} M_2 \neq 0
\]
Proof. In this model, the empirical moment function converges to (see lemma 3.6.5)

$$g_{0,i}(\alpha, 0) = \lim_{n \to \infty} \frac{1}{n} \text{tr} \left( \Sigma_0 \left( A_\alpha^{-1} A_0 \right)' K_i A_\alpha^{-1} A_0 \right), \forall i = 1, \ldots, I$$

where \( f_{n,i} = O(1) \) by lemmas A.2.3 and A.2.4.

A sufficient condition for local identification (see Newey and McFadden, 1994; Rothenberg, 1971) is

$$\frac{dg_{0,i}(\alpha_0, 0)}{d\alpha} \neq 0.$$ But the sequence

$$f'_{n,i}(\alpha) = \frac{d}{d\alpha} f_{n,i}(\alpha) = \frac{1}{n} \text{tr} \left( \Sigma_0 \left( A_\alpha^{-1} A_0 \right)' K_i^* A_\alpha^{-1} M_2 A_\alpha^{-1} A_0 \right)$$

is bounded (since it is continuous and \( \mathcal{A} \) is compact) and equicontinuous (which follows by the mean-value theorem by noting that its second derivative is bounded, see A.2.11); thus, Arzelá-Ascoli theorem (A.2.12) assures that \( f'_{n,i}(\alpha) \) converges uniformly. Theorem A.2.13 then allows the interchange between differentiation and limit,

$$\frac{d}{d\alpha} g_{0,i}(\alpha_0, 0) = \lim_{n \to \infty} \frac{d}{d\alpha} f_{n,i}(\alpha_0) = \lim_{n \to \infty} \frac{1}{n} \text{tr} \left( \Sigma_0 K_i^* A_\alpha^{-1} M_2 \right)$$

but, by assumption 3.6.7 this is non null for at least some \( i \), therefore \( \frac{d}{d\alpha} g_{0,i}(\alpha_0, 0) \neq 0 \), and the model is identified in a neighborhood of \( \alpha_0 \).

To illustrate, consider again an homoskedastic unit variance model (\( \Sigma_0 = I \)) with spatial weight matrix \( (M_2) \) given by the typical circular world where each spatial unit has two neighbors, and for each neighbor, the equivalent element of matrix \( M_2 \) is 0.5 (\( ||M_2||_1 = 1 \)). In this setting, the parameter space for \( \alpha \) can be defined as any closed interval, containing zero, such that \( \mathcal{A} \subset (-1, 1) \). Using Kelejian and Prucha's proposed two moment conditions based on \( K_1 = M_2 \) and \( K_2 = M_2^2 - \text{diag}(M_2^2) \), and considering Lee (2007) "best" moment condition for the homoskedastic case\(^{12} \), \( K_3 = M_2 \left( I - \alpha_0 M_2 \right)^{-1} \), the limiting function of the moment conditions is given by:

$$g_0(\alpha, 0) = \left[ \begin{array}{c} \lim_{n \to \infty} \frac{1}{n} \text{tr} \left( (A_\alpha^{-1} A_0)' M_2 A_\alpha^{-1} A_0 \right) \\ \lim_{n \to \infty} \frac{1}{n} \text{tr} \left( (A_\alpha^{-1} A_0)' \left( M_2^2 - \text{diag}(M_2^2) \right) A_\alpha^{-1} A_0 \right) \\ \lim_{n \to \infty} \frac{1}{n} \text{tr} \left( (A_\alpha^{-1} A_0)' \left( M_2 A_\alpha^{-1} - \text{diag}(M_2 A_\alpha^{-1}) \right) A_\alpha^{-1} A_0 \right) \end{array} \right] = \left[ \begin{array}{c} g_{0,1}(\alpha) \\ g_{0,2}(\alpha) \\ g_{0,3}(\alpha) \end{array} \right]$$

In this example (see figure 3.2) we can see that moment condition \( g_{0,1}(\alpha) \) alone would be enough to identify the true parameter \( \alpha_0 \), even globally\(^{13} \). Moment condition \( g_{0,2}(\alpha) \), on the other hand, is relatively flat in the neighborhood of the true parameter \( \alpha_0 \) and therefore does not help in identifying the model; this function (for some values of the parameter) has two roots within the interval \( \mathcal{A} \) and therefore is an example of a moment condition that satisfies the local identification condition but does not identify the model globally (even when restricted to the interval \( \mathcal{A} \)). The figure also shows that Lee's proposed "best" moment condition \( (g_{0,3}(\alpha)) \) could be used in the spatial moving average case as well and, in this example, it provides better local identification (compared to the other two moment conditions) and provides also "global" identification in the interval \( \mathcal{A} \).

\(^{12}\)Note that Lee argued that this moment condition is "best" (from a variance perspective) for the case of an spatial autoregressive process; it is shown here, in the moving average case, for illustrative purposes only.

\(^{13}\)In the figure, the \( x - \alpha \) is plotted only for values of \( \alpha \) in the interval \( \mathcal{A} \), but if one extend the axis it is possible to verify that \( g_{0,3}(\alpha) \) has only one root, for instance, in the interval \([-20, 20]\); note, however, that for \( \alpha \notin \mathcal{A} \) we cannot guarantee that \( (I - \alpha M_2)^{-1} \) is uniformly bounded and therefore the convergences of the functions can not be assured.
\( \mathbb{M}_{\text{ARMA}} \) Case

Consider now the spatial autoregressive moving average process represented by model \( \mathbb{M}_{\text{ARMA}} \). Due to the moving average component, a global identification condition is not straightforward, and therefore we state the condition for local identification, which is analogous to the case for model \( \mathbb{M}_{\text{MA}} \).

3.6.9 Assumption (Local asymptotic identification). Assume that the following matrix has rank 2

\[
\begin{bmatrix}
\lim_{n \to \infty} \frac{1}{n} \text{tr} (\Sigma_0 K_1^* A_0^{-1} M_2) & \lim_{n \to \infty} \frac{1}{n} \text{tr} (\Sigma_0 K_1^* R_0^{-1} M_2) \\
\vdots & \vdots \\
\lim_{n \to \infty} \frac{1}{n} \text{tr} (\Sigma_0 K_1^* A_0^{-1} M_2) & \lim_{n \to \infty} \frac{1}{n} \text{tr} (\Sigma_0 K_1^* R_0^{-1} M_2)
\end{bmatrix}
\]

With this assumption, the proposition below shows that model \( \mathbb{M}_{\text{ARMA}} \) is asymptotically locally identified.

3.6.10 Proposition. The spatial autoregressive moving average model \( \mathbb{M}_{\text{ARMA}} \), whose moment conditions given by equation 3.6.7, is locally asymptotically identified if it satisfies assumption 3.6.9.

\textit{Proof.} Analogous to proposition 3.6.8. \( \square \)
Remark. Note that if \( \rho_0 = \alpha_0 \), then \( A_0 = R_0 \) and therefore the columns of the above matrix are linearly dependent, so the model cannot be identified. But this is no surprise since in this case we would have:

\[
U = \rho_0 M_2 U + \varepsilon - \alpha_0 M_2 \varepsilon \\
(I - \rho_0 M_2) U = (I - \alpha_0 M_2) \varepsilon \\
U = \varepsilon
\]

and there would be no spatial effect in the model’s residual. Assumption 3.4.4 rules out this case.

3.6.2 Uniform Convergence

Corollary to proposition 3.6.5 has established point-wise convergence of the objective function to a limiting non-stochastic function, which is maximized (locally) only at the true parameter value. However, in order to show consistency of the GMM estimator \( \hat{\theta} \), we need to show that the convergence is uniform. Before that, some useful lemmas and definitions are stated.

3.6.11 Definition (Uniform convergence in probability). \( \hat{Q}_n(\theta) \) converges uniformly in probability to \( Q_0(\theta) \) iff

\[
\lim_{n \to \infty} \sup_{\theta \in A \times \mathbb{R}} |\hat{Q}_n(\theta) - Q_0(\theta)| = 0
\]

3.6.12 Definition (Bounded in probability). An stochastic process \( \{X_n\} \) is bounded in probability, denoted by \( X_n = O_p(1) \), if for any \( \varepsilon > 0 \), exists a finite \( M_\varepsilon \in \mathbb{R} \), and \( N \in \mathbb{N} \), such that \( \Pr(|X_n| > M_\varepsilon) \leq \varepsilon \) for all \( n > N \).

3.6.13 Lemma. If an stochastic process \( \{X_n\} \) converges in distribution then it is bounded in probability.

Proof. This is exercise III.3.68 of Dorogovtsev et al. (1997, p.117). Let \( F_n(x) = \Pr(X_n \leq x) \) denote the distribution function of \( X_n \), and let \( F(x) \) be its limit. Since \( F(x) \) is a non-decreasing function of \( x \), \( \Pr(|X| \leq M) = F(M) - F(-M) \) is also non-decreasing, ranging from \( \Pr(X = 0) \) (when \( M = 0 \)) to 1 (when \( M = +\infty \)). Then, \( \Pr(|X| > M) \) is non-increasing, ranging from \( 1 - \Pr(X = 0) \) to 0 as \( M \) increases. As a result, given any \( \varepsilon > 0 \) there is an \( M_\varepsilon < \infty \) such that \( \Pr(|X| > M_\varepsilon) < \varepsilon \). Moreover, since \( \{X_n\} \) converges in distribution,

\[
\lim_{n \to \infty} \Pr(|X_n| > M_\varepsilon) = \lim_{n \to \infty} \int_{|x|>M_\varepsilon} dF_n(x) = \int_{|x|>M_\varepsilon} dF(x) < \varepsilon
\]

therefore \( X_n = O_p(1) \).

Corollary. If \( \{X_n\} \) converges in probability to a non-stochastic limit, then \( X_n = O_p(1) \)

3.6.14 Lemma. If an stochastic process \( \{X_n\} \) has finite absolute first moment (\( E|X_n| < \infty \)) then \( X_n = O_p(1) \).

Proof. Let \( K > 0 \) be an upper bound for \( E|X_n| \). Then, given \( \varepsilon > 0 \), define \( M_\varepsilon = \frac{K}{\varepsilon} \), finite. Then for all \( n \)

\[
\Pr(|X_n| \geq M_\varepsilon) \leq \frac{E|X_n|}{M_\varepsilon} \text{ by Markov’s inequality} \\
\leq \frac{K}{M_\varepsilon} = \varepsilon
\]
3.6.15 Lemma. If an stochastic process \( \{X_n\} \) has finite second moments \( (EX_n^2 < +\infty, \forall n) \) then \( X_n = O_p(1) \).

Proof. Let \( K > 0 \) be an upper bound for \( EX_n^2 \). Then, given \( \varepsilon > 0 \), define \( M_\varepsilon = \left( \frac{K}{\varepsilon} \right)^{1/2} \), finite. Then for all \( n \)

\[
\Pr(|X_n| \geq M_\varepsilon) \leq \frac{EX_n^2}{M_\varepsilon^2} \leq \frac{K}{M_\varepsilon^2} = \varepsilon
\]

by Chebyshev’s inequality.

\[ \blacksquare \]

The following definition and lemmas are adapted from Newey and McFadden (1994).

3.6.16 Definition (Stochastic equicontinuity). \( \hat{Q}_n(\theta) \) is stochastically equicontinuous if for every \( \varepsilon, \eta > 0 \) there exists a sequence of random variables \( \Delta_n \) and a sample size \( N \) such that for \( n \geq N \),

\[
\Pr(|\Delta_n| > \varepsilon) < \eta \quad \text{and for each } \theta \text{ there is an open neighborhood of } \theta, \text{ denoted by } \mathcal{N}, \text{ with}
\]

\[
\sup_{\theta \in \mathcal{N}} |\hat{Q}_n(\tilde{\theta}) - \hat{Q}_n(\theta)| \leq \Delta_n, \quad \forall n \geq N
\]

3.6.17 Lemma ("In probability" Lipschitz). If there is an \( \alpha > 0 \) and a random variable \( B_n = O_p(1) \) such that for all \( \tilde{\theta}, \theta \in \Theta \), \( |\hat{Q}_n(\tilde{\theta}) - \hat{Q}_n(\theta)| \leq B_n |\tilde{\theta} - \theta|^\alpha \), then \( \hat{Q}_n(\theta) \) is stochastically equicontinuous.

Proof. See lemma 2.9 of Newey and McFadden (1994). Choose \( \delta > 0 \). Since \( B_n = O_p(1) \), there is a finite \( M_\delta \) and \( N_\delta \in \mathbb{N} \) such that \( \Pr(|B_n| > M_\delta) < \delta \), for all \( n > N_\delta \). Choose \( \varepsilon > 0 \), and define \( \Delta_n = \frac{B_n M_\delta}{\varepsilon} \) and \( \mathcal{N} = \left\{ \tilde{\theta} : |\tilde{\theta} - \theta|^\alpha < \frac{\varepsilon}{B_n M_\delta} \right\} \). Then \( \Pr(|\Delta_n| > \varepsilon) = \Pr(|B_n| > M_\delta < \delta \), and for all \( \theta, \tilde{\theta} \in \mathcal{N} \),

\[
|\hat{Q}_n(\tilde{\theta}) - \hat{Q}_n(\theta)| \leq B_n |\tilde{\theta} - \theta|^\alpha = \Delta_n M_\delta |\tilde{\theta} - \theta|^\alpha < \Delta_n, \quad \forall n > N_\delta
\]

\[ \blacksquare \]

3.6.18 Lemma. Assume \( \Theta \) compact and \( Q_0(\theta) \) continuous. Then \( \hat{Q}_n(\theta) \) converges uniformly in probability to \( Q_0(\theta) \), i.e., \( \sup_{\theta \in \Theta} |\hat{Q}_n(\theta) - Q_0(\theta)| \xrightarrow{p} 0 \), if and only if \( \hat{Q}_n(\theta) \xrightarrow{D} Q_0(\theta), \forall \theta \in \Theta \) and \( \hat{Q}_n(\theta) \) is stochastically equicontinuous.


\[ \blacksquare \]

The previous lemma allows to show uniform convergence in probability of the objective function \( \hat{Q}_n(\theta) \) by showing that it is stochastically equicontinuous. Before that, one additional proposition is needed:

3.6.19 Proposition. The first derivative of the empirical moments vector converges in probability to a non-stochastic function, i.e., for all \( i = 1, \ldots, l \)

\[
\text{plim}_{n \to \infty} \frac{d}{d\alpha} \hat{g}_{n,i}(\theta) = \frac{d}{d\alpha} g_{0,i}(\theta) = \left[ \frac{\partial}{\partial \alpha} g_{0,i}(\theta) \right] \left( \frac{\partial}{\partial \rho} g_{0,i}(\theta) \right), \quad \forall \theta \in A \times \mathcal{R}
\]

Proof. Consider the derivative of a generic element of the empirical moment vector \( \hat{g}_{n,i}(\theta) \):

\[
\frac{\partial}{\partial \alpha} \hat{g}_{n,i}(\theta) = \frac{1}{n} \left( \frac{U'}{A_{-1} R^2} \right) \begin{bmatrix} K_1^{-1} & A_{-1}^{-1} M A_{-1}^{-1} R_{\alpha} \end{bmatrix} C_{i,\alpha}(\theta) \tag{3.6.12}
\]

\[
\text{plim}_{n \to \infty} \frac{\partial}{\partial \alpha} \hat{g}_{n,i}(\theta) = \text{plim}_{n \to \infty} \frac{1}{n} \left( \frac{U'}{A_{-1} R^2} \right) C_{i,\alpha}(\theta) U \tag{3.6.13}
\]
(by lemma 3.6.4, since \( C_{i,\alpha}(\theta) \) is uniformly bounded)

\[
\lim_{n \to \infty} \frac{1}{n} e'(R_0^{-1}A_0)'C_{i,\alpha}R_0^{-1}A_0 e_{\alpha} = (DGP \ m_0) \quad (3.6.14)
\]

\[
\lim_{n \to \infty} \frac{1}{n} \text{tr} \left( \Sigma_0 (R_0^{-1}A_0)'C_{i,\alpha}R_0^{-1}A_0 \right) = \text{(lemma 3.6.3)} \quad (3.6.15)
\]

similarly

\[
\frac{\partial}{\partial \theta} \hat{g}_{n,i}(\theta) = \frac{-1}{n} \hat{U}' (A^{-1}_\alpha R_\theta)'K_i A^{-1}_\alpha M \hat{U} \quad (3.6.16)
\]

\[
\lim_{n \to \infty} \frac{\partial}{\partial \theta} \hat{g}_{n,i}(\theta) = \lim_{n \to \infty} \frac{-1}{n} U' C_{i,\theta}(\theta) U \quad (3.6.17)
\]

(again lemma 3.6.4, since \( C_{i,\theta}(\theta) \) is uniformly bounded)

\[
\lim_{n \to \infty} \frac{-1}{n} \hat{g}'(R_0^{-1}A_0)'C_{i,\theta}R_0^{-1}A_0 e_{\alpha} = (DGP \ m_0) \quad (3.6.18)
\]

\[
\lim_{n \to \infty} \frac{-1}{n} \text{tr} \left( \Sigma_0 (R_0^{-1}A_0)'C_{i,\theta}R_0^{-1}A_0 \right) = \text{(lemma 3.6.3)} \quad (3.6.19)
\]

Recall from equation 3.6.11 that

\[
g_{0,i}(\theta) = \lim_{n \to \infty} \frac{1}{n} \text{tr} \left( \Sigma_0 B_i(\theta) \right), \quad \text{where} \quad B_i(\theta) = (R_0^{-1}A_0)'(A^{-1}_\alpha R_\theta)'K_i A^{-1}_\alpha R_\theta R_0^{-1}A_0
\]

and note that \( \frac{\partial f_n}{\partial \alpha} = f'_n, \alpha \) and \( \frac{\partial f_n}{\partial \theta} = f'_n, \theta \). Since both \( f'_n, \alpha \), \( f'_n, \theta \) converges uniformly, using theorem A.2.13 (interchange of differentiation and limit) we conclude that

\[
\lim_{n \to \infty} \frac{d}{d\theta} \hat{g}_{n,i}(\theta) = \frac{d}{d\theta} g_{0,i}(\theta)
\]

The following proposition uses lemma 3.6.17 to show that the objective function is stochastically equicontinuous.

3.6.20 Proposition. The objective function \( \hat{Q}_n(\theta) \) (equation 3.6.6) is stochastically equicontinuous.

Proof. The first derivative of the objective function is

\[
\frac{d}{d\theta} \hat{Q}_n(\theta) = -\hat{g}_n(\theta)'W_n \frac{d}{d\theta} \hat{g}_n(\theta)
\]

which by continuity and propositions 3.6.5 and 3.6.19 converges in probability to a non-stochastic function

\[
\lim_{n \to \infty} \frac{d}{d\theta} \hat{Q}_n(\theta) = -g_0(\theta)'W \frac{d}{d\theta} g_0(\theta)
\]

therefore, by corollary following lemma 3.6.13, \( \frac{d}{d\theta} \hat{Q}_n(\theta) = O_p(1) \).

By the mean-value theorem

\[
\hat{Q}_n(\theta) = \hat{Q}_n(\bar{\theta}) + \frac{d}{d\theta} \hat{Q}_n(\theta) \bigg|_{\theta = \bar{\theta}} (\theta - \bar{\theta})
\]
where \( \theta^* \) is a convex combination of \( \theta \) and \( \tilde{\theta} \). Taking absolute values

\[
\left| \hat{Q}_n(\theta) - \hat{Q}_n(\tilde{\theta}) \right| = \left| \frac{d}{d\theta} \hat{Q}_n(\theta^*)(\theta - \tilde{\theta}) \right| \leq \left\| \frac{d}{d\theta} \hat{Q}_n(\theta^*) \right\|_F \left\| \theta - \tilde{\theta} \right\|_F
\]

and using lemma 3.6.17 we conclude that \( \hat{Q}_n(\theta) \) is stochastically equicontinuous. \( \square \)

The developments so far in this section, allow to conclude that:

3.6.21 Proposition. \( \hat{Q}_n(\theta) \) converges uniformly in probability to \( Q_0(\theta) \).

Proof. The parameter space \( \mathcal{A} \times \mathcal{R} \) is compact, \( \hat{Q}_n(\theta) \) converges in probability (point-wise) to a continuous function \( Q_0(\theta) \) (corollary to proposition 3.6.5), and \( \hat{Q}_n(\theta) \) is stochastically equicontinuous (proposition 3.6.20). Therefore, the conclusion follows from lemma 3.6.18. \( \square \)

3.6.3 Consistency

In order to show consistency of the GMM estimator \( \hat{\theta} = \arg\max_{\theta \in \mathcal{A} \times \mathcal{R}} \hat{Q}_n(\theta) \), I will rely on theorem 2.1 of Newey and McFadden (1994), reproduced below:

3.6.22 Theorem. If there is a function \( Q_0(\theta), \theta \in \Theta \) such that (i) \( Q_0(\theta) \) is uniquely maximized at \( \theta_0 \), (ii) the parameter space \( \Theta \) is compact, (iii) \( Q_0(\theta) \) is continuous, and (iv) \( \hat{Q}_n(\theta) \) converges uniformly in probability to \( Q_0(\theta) \), then \( \lim_{n \to \infty} \hat{\theta} = \theta_0 \).

We have established that (a) \( Q_0(\theta) \) is uniquely maximized at \( \theta_0 \) (proposition 3.6.10), (b) \( \mathcal{A} \times \mathcal{R} \) is compact (assumption 3.4.4), (c) \( Q_0(\theta) \) is continuous, and (d) \( \hat{Q}_n(\theta) \) converges uniformly in probability to \( Q_0(\theta) \) (proposition 3.6.21). Therefore, by the above theorem, we conclude \( \hat{\theta} \overset{D}{=} \theta_0 \).

A practical "solution" to the problem of global GMM identification, that has often been adopted, is to simply assume identification. This practice is reasonable, given the difficulty of formulating primitive conditions, but it is important to check that it is not a vacuous assumption whenever possible, by showing identification in some special cases. In simple models it may be possible to show identification under particular forms for conditional distributions (Newey and McFadden, 1994).

3.6.4 Asymptotic Normality

In order to show asymptotic normality, we rely on theorem 3.2 of Newey and McFadden (1994), reproduced below:

3.6.23 Theorem. Let \( \hat{\theta} \) be the GMM estimator defined in equation 3.6.5. \( \lim_{n \to \infty} \hat{\theta} = \theta_0 \), \( \lim_{n \to \infty} W_n = W \), \( W \) symmetric, positive definite, and assume (i) \( \theta_0 \) is in the interior of \( \mathcal{A} \times \mathcal{R} \), (ii) \( g_n(\theta) \) is continuously differentiable in a neighborhood \( N \) of \( \theta_0 \), (iii) \( \sqrt{n} g_n(\theta_0) \overset{d}{\to} N(0, \Omega_0) \), (iv) \( \frac{d}{d\theta} g_n(\theta) \) converges uniformly in probability, in the neighborhood \( N \), to a function \( G(\theta) \) which is continuous at \( \theta_0 \), and (v) \( G(\theta_0)WG(\theta_0) \) is nonsingular. Then

\[
\sqrt{n} \left( \hat{\theta} - \theta_0 \right) \overset{d}{\to} N \left( 0, (G_0'WG_0)^{-1} G_0' W \Omega_0 W G_0 (G_0'WG_0)^{-1} \right), \quad G_0 \equiv G(\theta_0)
\]

or if the weighting matrix is optimal, \( W = \Omega_0^{-1} \),

\[
\sqrt{n} \left( \hat{\theta} - \theta_0 \right) \overset{d}{\to} N \left( 0, (G_0'\Omega_0^{-1}G_0)^{-1} \right)
\]
Pointwise convergence in probability of the GMM estimator \( \hat{\theta} \overset{p}{\to} \theta_0 \) was shown in section 3.6.3 using theorem 2.1 of Newey and McFadden (1994). Convergence of the weighting matrix \( \hat{W}_n \overset{p}{\to} W \) to a symmetric, positive definite matrix, and assumption (i) of the above theorem are a restatement of assumptions 3.4.5 and 3.4.4. Assumption (ii) follows from the definition of the moment equations \( \hat{g}_n(\theta) \) (see equation 3.6.7). The remaining assumptions – (iii), (iv), and (v) – will be shown in the sequence.

In order to show that the empirical moment conditions converges in distribution (assumption (iii)), I will use a simplified version of the central limit theorem for linear quadratic forms proved in Kelejian and Prucha (2001, theorem 1), reproduced below:

**3.6.24 Theorem (Central limit theorem for linear quadratic forms).** Let \( m_n = \frac{1}{\sqrt{n}} \varepsilon' B \varepsilon + \frac{1}{\sqrt{n}} b' \varepsilon \), and assume (i) \( E \varepsilon_i = 0, \forall i = 1, \ldots, n, \forall n \), and \( \varepsilon_1, \ldots, \varepsilon_n \) are independent for all \( n \), (ii) the \( n \times n \) matrix \( B \) is uniformly bounded, (iii) the \( n \times 1 \) vector \( b \) satisfy, for some \( \delta_1 > 0 \), \( \frac{1}{n} \sum_{i=1}^{n} |b_i|^{2+\delta_2} < \infty \), \( \forall n \), (iv) for some \( \delta_2 > 0 \), \( E|\varepsilon_i|^{4+\delta_2} < \infty, \forall i = 1, \ldots, n, \forall n \), and (v) the variance of the quadratic form \( m_n \) is bounded away from zero. Then

\[
\frac{m_n - \mu_{m_n}}{\sigma_{m_n}} \xrightarrow{d} N(0, 1)
\]

where

\[
\begin{align*}
\mu_{m_n} &= \frac{1}{\sqrt{n}} \text{tr} (\Sigma B) \\
\sigma^2_{m_n} &= \frac{1}{n} \left\{ \text{tr} (\Sigma B \Sigma B') + b' \Sigma b + \sum_{i=1}^{n} \left[ b_i^2 \left( E \varepsilon_i^4 - 3 (E \varepsilon_i^2)^2 \right) + 2 b_i b_i E \varepsilon_i^3 \right] \right\} \\
\Sigma &= E \varepsilon \varepsilon'
\end{align*}
\]

We now proceed to show that assumptions (iii), (iv), and (v) of theorem 3.6.23 are satisfied:

**3.6.25 Proposition (Assumption (iii)).** \( \sqrt{n} \hat{g}_n(\theta_0) \overset{d}{\to} N(0, \Omega_0) \), where

\[
\Omega_0 = \begin{bmatrix}
\hat{\sigma}_{11} & \cdots & \hat{\sigma}_{1k} \\
\vdots & \ddots & \vdots \\
\hat{\sigma}_{k1} & \cdots & \hat{\sigma}_{kk}
\end{bmatrix}
\]

\[
\hat{\sigma}_{ij} = \lim_{n \to \infty} \frac{1}{n} \text{tr} (\Sigma_0 K_i \Sigma_0 K_i' \Sigma_0 K_j \Sigma_0 K_j'), \quad K_i' \equiv K_i + K_i', \quad \forall i, j = 1, \ldots, l
\]

\[
\Sigma_0 = E \varepsilon \varepsilon'
\]

**Proof.** Note that each element of the vector of empirical moments, evaluated at \( \theta_0 \), is given by

\[
\hat{g}_{n,i}(\theta_0) = \frac{1}{n} \hat{U}' (A_0^{-1} R_0)' K_i A_0^{-1} R_0 \hat{U}
\]

therefore,

\[
\text{plim}_{n \to \infty} \sqrt{n} \hat{g}_{n,i}(\theta_0) = \text{plim}_{n \to \infty} \frac{1}{\sqrt{n}} \hat{U}' (A_0^{-1} R_0)' K_i A_0^{-1} R_0 \hat{U}
\]

which, by lemma 3.6.4 (since \( (A_0^{-1} R_0)' K_i A_0^{-1} R_0 \) is uniformly bounded for all \( i \)), yields

\[
= \text{plim}_{n \to \infty} \frac{1}{\sqrt{n}} \hat{U}' (A_0^{-1} R_0)' K_i A_0^{-1} R_0 U
\]

\[
= \text{plim}_{n \to \infty} \frac{1}{\sqrt{n}} \varepsilon' K_i \varepsilon
\]
for all \( i = 1, \ldots, l \). Thus \( \sqrt{n} \hat{g}_{n,i}(\theta_0) = \left[ \frac{1}{\sqrt{n}} e' K_i e + o_p(1) \cdots \frac{1}{\sqrt{n}} e' K_l e + o_p(1) \right]' \). Since the innovation \( \varepsilon \) and the matrices \( K_i, i = 1, \ldots, l \) satisfy the assumptions of theorem 3.6.24, each element of the empirical moments vector is asymptotically normal:

\[
\sqrt{n} \hat{g}_{n,i}(\theta_0) \xrightarrow{d} N \left( 0, \sigma^2_{g_{n,i}}(\theta_0) \right)
\]

\[
\mu_{g_{n,i}}(\theta_0) = \lim_{n \to \infty} \frac{1}{\sqrt{n}} \text{tr} \Sigma_0 K_i = 0
\]

\[
\sigma^2_{g_{n,i}}(\theta_0) = \lim_{n \to \infty} \frac{1}{n} \text{tr} (\Sigma_0 K_i \Sigma_0 K_i^*), \quad K_i^* = K_i + K_i^t
\]

where the simplification arises since \( K_i \) has zero diagonal\(^{14}\). Also, note that \( \sigma^2_{g_{n,i}}(\theta_0) = O(1) \) by lemma A.2.4, and is bounded away from zero.

Let \( \hat{\sigma}_{ij} \equiv \lim_{n \to \infty} \text{Cov}(\sqrt{n} \hat{g}_{n,i}(\theta_0), \sqrt{n} \hat{g}_{n,j}(\theta_0)) = \lim_{n \to \infty} n E \hat{g}_{n,i}(\theta_0) \hat{g}_{n,j}(\theta_0) \), and note that for all \( i, j = 1, \ldots, l \)

\[
\hat{\sigma}_{ij} = \lim_{n \to \infty} \frac{1}{n} \text{tr} (\Sigma_0 K_j \Sigma_0 K_i^t) = O(1), \quad K_i^t \equiv K_i + K_i^t
\]

To prove that \( \sqrt{n} \hat{g}_{n}(\theta_0) \) is asymptotically normal we use the Cramér-Wold device (see Billingsley, 1986, theorem 29.4), which asserts that, given two random vectors \( X_n \in \mathbb{R}^l \) and \( Z \in \mathbb{R}^l \), then \( X_n \overset{d}{\to} Z \) iff \( t' X_n \overset{d}{\to} t' Z \), \( \forall t \in \mathbb{R}^l \). Let \( Z \in \mathbb{R}^l \) be a Gaussian random vector \( Z \sim N(0, \Omega_0) \), where \( \Omega_0 \) is the \( l \times l \) matrix with typical element \( (\Omega_0)_{ij} = \hat{\sigma}_{ij} \). Then it is straightforward to show that \( t'(\sqrt{n} \hat{g}_{n}(\theta_0)) \) converges in distribution to \( t' Z \) for all \( t \in \mathbb{R}^l \), thus \( \sqrt{n} \hat{g}_{n}(\theta_0) \) is asymptotically normal with zero mean and covariance matrix given by \( \Omega_0 \).

\[\text{□}\]

3.6.26 Proposition (Assumption (iv)). \( \hat{G}_n(\theta) = \frac{d}{d\theta} \hat{g}_{n}(\theta) \) converges uniformly in probability to a continuous function \( G_0(\theta) \).

Proof. Proposition 3.6.19 has already established the pointwise convergence to a continuous function

\[
\text{plim}_{n \to \infty} \hat{G}_n(\theta) = \frac{d}{d\theta} g_0(\theta) = G_0(\theta)
\]

Thus, we need only to show that \( \hat{G}_n(\theta) \) is stochastically equicontinuous (see lemma 3.6.18). By the mean-value theorem

\[
\text{vec} \hat{G}_n(\theta) = \text{vec} \hat{G}_n(\hat{\theta}) + \frac{d}{d\theta} \text{vec} \hat{G}_n(\theta^*) \left( \theta - \hat{\theta} \right)
\]

where \( \theta^* \) is a convex combination of \( \theta \) and \( \hat{\theta} \); therefore,

\[
\| \text{vec} \hat{G}_n(\theta) - \text{vec} \hat{G}_n(\hat{\theta}) \|_F \leq \left\| \frac{d}{d\theta} \text{vec} \hat{G}_n(\theta^*) \right\|_F \| \theta - \hat{\theta} \|_F
\]

\[
\leq \sqrt{n} \left( \frac{d}{d\theta} \text{vec} \hat{G}_n(\theta^*) \right) \| \theta - \hat{\theta} \|_F
\]

and in order to use the "in probability" Lipschitz condition it is enough to show that all the elements of the \( 2l \times 2 \) matrix (\( * \)) are \( O_p(1) \), i.e., that \( \frac{\partial^2}{\partial \alpha^2} \hat{g}_{n,i}(\theta) \), \( \frac{\partial^2}{\partial \alpha \beta} \hat{g}_{n,i}(\theta) \), and \( \frac{\partial^2}{\partial \alpha \theta} \hat{g}_{n,i}(\theta) \) are bounded in probability for all \( i = 1, \ldots, l \). But, following the same steps than in the proof of proposition 3.6.19,

\[
\frac{\partial^2}{\partial \alpha^2} \hat{g}_{n,i}(\theta) = \frac{1}{n} \hat{U}'(A^{-1} R_p)' \left( (K_i^t A^{-1} M_2 + 2K_i^t A^{-1} M_2) A^{-1} M_2 A^{-1} R_p \hat{U}_p \right)
\]

\(^{14}\)Note that in this particular case, the assumptions of theorem 3.6.24 could be weakened to \( E|\varepsilon_i|^{2+\delta} < \infty \) for some \( \delta > 0 \) (Kelejian and Prucha, 2001).
\[ \lim_{n \to \infty} \frac{\partial^2}{\partial \alpha \partial \rho} \hat{g}_{n,i}(\theta) = \lim_{n \to \infty} \frac{1}{n} \epsilon' (A'_{\alpha} R_{\alpha} R_{\alpha}^{-1} A_0) \left[ [K'_{\alpha} A_{\alpha}^{-1} M_2] + 2K'_{\alpha} A_{\alpha}^{-1} M_2 \right] A_{\alpha}^{-1} M_2 A_{\alpha}^{-1} R_{\alpha} R_{\alpha}^{-1} A_0 \epsilon \]

\[ = \lim_{n \to \infty} \frac{1}{n} \text{tr} \Sigma_0 D_{i,\alpha,\rho}(\theta) \]

\[ \frac{\partial^2}{\partial \rho^2} \hat{g}_{n,i}(\theta) = \frac{1}{n} \hat{U}'(A_{\alpha}^{-1} M_2) K_{\alpha} A_{\alpha}^{-1} M_2 \hat{U} \]

\[ \lim_{n \to \infty} \frac{1}{n} \epsilon' (A_{\alpha}^{-1} M_2 R_{\alpha} R_{\alpha}^{-1} A_0) \left[ [K'_{\alpha} A_{\alpha}^{-1} M_2] + 2K'_{\alpha} A_{\alpha}^{-1} M_2 \right] A_{\alpha}^{-1} R_{\alpha} R_{\alpha}^{-1} A_0 \epsilon \]

\[ = \lim_{n \to \infty} -\frac{1}{n} \text{tr} \Sigma_0 D_{i,\alpha,\rho}(\theta) \]

we conclude that the second derivatives of the elements of the moment vector converge in probability to non-stochastic functions, and therefore are \( O_p(1) \). Therefore 3.6.17 assures \( \hat{G}_n(\theta) \) is stochastically equicontinuous and it then follows that the convergence in probability \( \hat{G}_n(\theta) \xrightarrow{P} G_0(\theta) \) is uniform.

\[ \frac{\partial}{\partial \theta} g_0(\theta) \equiv G_0(\theta) \]

\[ 3.6.27 \text{ Proposition (Assumption (v)). } G_0(\theta_0)/W G_0(\theta_0) \text{ is non-singular.} \]

\[ \text{Proof. Let } W = W^\frac{1}{2} W^\frac{1}{2}, \text{ which exists since } W \text{ is symmetric and positive definite. Note that (see equations 3.6.15 and 3.6.19)} \]

\[ G_0(\theta_0) = \left[ \begin{array}{c} \frac{\partial}{\partial \alpha} g_0(\theta_0) \\ \frac{\partial}{\partial \rho} g_0(\theta_0) \end{array} \right] \]

\[ = \left[ \begin{array}{c} \lim_{n \to \infty} \frac{1}{n} \text{tr} \Sigma_0 K_{\alpha}^i A_{\alpha}^{-1} M_2 \\ \lim_{n \to \infty} \frac{1}{n} \text{tr} \Sigma_0 K_{\alpha}^i R_{\alpha}^{-1} M_2 \\ \vdots \\ \lim_{n \to \infty} \frac{1}{n} \text{tr} \Sigma_0 K_{\alpha}^i A_{\alpha}^{-1} M_2 \\ \lim_{n \to \infty} \frac{1}{n} \text{tr} \Sigma_0 K_{\alpha}^i R_{\alpha}^{-1} M_2 \end{array} \right] \]

but by assumption 3.6.9 (local identification for the model \( \mathbb{M}_{ARMA} \)) the columns of this matrix are linearly independent, and therefore \( W^\frac{1}{2} G_0(\alpha_0) \) has rank 2 and the matrix \( \left( W^\frac{1}{2} G_0(\alpha_0) \right)' W^\frac{1}{2} G_0(\alpha_0) \) is non-singular.

Summing up, the conditions for theorem 3.2 of Newey and McFadden (1994) (theorem 3.6.23) are satisfied in the present setting, and thus, using a weighting matrix \( W_n \) that converges in
probability to $\Omega_0^{-1}$, yields:

$$\sqrt{n}(\theta - \theta_0) \xrightarrow{d} N \left( 0, (G_0\Omega_0^{-1}G_0)^{-1} \right)$$

(3.6.20)

$$\Omega_0 = \lim_{n \to \infty} \frac{1}{n} \begin{bmatrix}
\text{tr} \Sigma_0 K_1 \Sigma_0 K_1^* & \cdots & \text{tr} \Sigma_0 K_k \Sigma_0 K_k^*
\vdots & \ddots & \vdots
\text{tr} \Sigma_0 K_1 \Sigma_0 K_k^* & \cdots & \text{tr} \Sigma_0 K_k \Sigma_0 K_k^*
\end{bmatrix}$$

(3.6.21)

$$G_0 = \lim_{n \to \infty} \frac{1}{n} \begin{bmatrix}
\text{tr} \Sigma_0 K_1^* A_0^{-1} M_2 & -\text{tr} \Sigma_0 K_1^* R_0^{-1} M_2
\vdots & \vdots
\text{tr} \Sigma_0 K_k^* A_0^{-1} M_2 & -\text{tr} \Sigma_0 K_k^* R_0^{-1} M_2
\end{bmatrix}$$

(3.6.22)

**Covariance Matrix Estimator**

Since $\Sigma_0, A_0 = I - \alpha_0 M_2$, and $R_0 = I - \rho_0 M_2$ are not known, one needs estimates $\hat{\Omega}_0$ and $\hat{C}_0$ of the true $\Omega_0$ and $G_0$ in order to estimate the asymptotic variance of $\hat{\theta}$. The following lemmas and propositions characterize those estimates and show their convergence to the true matrices.

**3.6.28 Lemma.** Let $\Sigma_0 = E \varepsilon \varepsilon'$ with entries $(\Sigma_0)_{ij} = \sigma_{ij}^2$, $(\Sigma_0)_{ij} = 0, i \neq j$, let $\hat{\Sigma}$ be a diagonal matrix with entries $(\hat{\Sigma})_{ii} = \bar{e}_i^2$ and $A, B$ be $n \times n$ uniformly bounded matrices with zeros elements in the diagonal. Then

$$\lim_{n \to \infty} \frac{1}{n} \text{tr} \left( \hat{\Sigma} A \hat{\Sigma} B \right) = \lim_{n \to \infty} \frac{1}{n} \text{tr} \left( \Sigma_0 A \Sigma_0 B \right) = O(1)$$

*Proof.* Note that

$$\text{tr} \hat{\Sigma} A \hat{\Sigma} B = \sum_{i=1}^{n} \sum_{j=1}^{n} \bar{e}_i^2 a_{ij} b_{ij} \bar{e}_j^2 = \sum_{i=1}^{n} \sum_{j=1}^{n} \bar{e}_i^2 c_{ij} \bar{e}_j^2 = \eta' C \eta$$

$$\text{tr} \Sigma_0 A \Sigma_0 B = \sum_{i=1}^{n} \sum_{j=1}^{n} \sigma_{ij}^2 a_{ij} b_{ij} \sigma_{ij}^2 = \sum_{i=1}^{n} \sum_{j=1}^{n} \sigma_{ij}^2 c_{ij} \sigma_{ij}^2 = \bar{\eta}' C \bar{\eta}$$

where $C \equiv A \ast B'$, $\eta \equiv [\bar{e}_1^2 \ldots \bar{e}_n^2]'$, and $\bar{\eta} = E \eta = [\bar{\sigma}_1^2 \ldots \bar{\sigma}_n^2]'$. Therefore,

$$E \eta' C \eta = E \eta' \eta^* + 0 + \bar{\eta}' C \bar{\eta}, \text{ where } \eta^* = \eta - \bar{\eta}$$

Since $\eta^*$ is a vector of zero mean, independent random variables, and $C$ is a uniformly bounded matrix with zeros in the diagonal, by using lemma A.2.8 we conclude $E \eta' \eta^* = 0$ and therefore $E \eta' C \eta = \bar{\eta}' C \bar{\eta}$, i.e., $E \left( \text{tr} \hat{\Sigma} A \hat{\Sigma} B \right) = \text{tr} \Sigma_0 A \Sigma_0 B$. The conclusion that $\text{tr} \Sigma_0 A \Sigma_0 B = O(n)$ comes from lemmas A.2.3 and A.2.4, since $A, B, \Sigma_0$ are uniformly bounded.

Note also that

$$\eta' C \eta = (\eta' + \bar{\eta}') C (\eta^* + \bar{\eta}) = \eta' \bar{C} \bar{\eta}^* + \eta' \bar{C} \bar{\eta} + \bar{\eta}' \bar{C} \eta^* + \bar{\eta}' \bar{C} \eta$$

$$= \eta' \bar{C} \eta^* + \bar{\eta}' C \eta^* + \bar{\eta}' C \eta$$

$$(\eta' C \eta - \bar{\eta}' \bar{C} \eta)^2 = (\eta' \bar{C} \eta)^2 + 2\eta' \bar{C} \eta \eta' C \eta^* + (\bar{\eta}' C \eta)^2$$

where $\Xi \equiv E \eta' \eta'^*$ is uniformly bounded (since the elements of $\eta'^*$ are independent, with zero mean, and the innovation $\varepsilon$ has bounded fourth moments), and $C, C^*$ are uniformly bounded (lemma A.2.6). Therefore, taking expected values

$$\text{Var}(\eta' C \eta) = \text{tr} \Xi C \Xi + 0 + \sum_{k} \left( \sum_{i} \bar{\eta}_k c_{ki}^2 \right) E \eta_k^* = O(n)$$

$O(1)$
where the first term results from lemma A.2.8 item (b) by recalling that $A, B$ have zero diagonal, an the second term results from item (e) of the same lemma. From these results, we conclude that $\frac{1}{n} \text{tr}(\hat{\Sigma} A \hat{\Sigma} B) = O(1)$ and $\text{Var}\left(\frac{1}{n} \text{tr}(\hat{\Sigma} A \hat{\Sigma} B)\right) = o(1)$, thus $\text{plim}_{n \to \infty} \frac{1}{n} \text{tr}(\hat{\Sigma} A \hat{\Sigma} B) = \lim_{n \to \infty} \frac{1}{n} \text{tr} \Sigma_0 A \Sigma_0 B$.

\[ \square \]

### 3.6.29 Lemma

Let $\hat{\Sigma}$ be a diagonal matrix with elements $(\hat{\Sigma})_{ii} = \varepsilon_i^2$, $\hat{\Sigma}$ be a diagonal matrix with elements $(\hat{\Sigma})_{ii} = \varepsilon_i^2$ and $A, B$ be $n \times n$ uniformly bounded matrices with zeros in the diagonal. Then

$$
\text{plim}_{n \to \infty} \frac{1}{n} \text{tr} \left( \hat{\Sigma} A \hat{\Sigma} B \right) = \text{plim}_{n \to \infty} \frac{1}{n} \text{tr} \left( \hat{\Sigma} A \hat{\Sigma} B \right)
$$

**Proof.** Recall that the true residual $\varepsilon$ arises from the model

$$
\varepsilon(\phi) = (I - \alpha M_2)^{-1} (I - \rho M_2) h(\delta)
$$

where

$$
\begin{align*}
    h(\delta) &= Y - f(\delta) = Y - (I - \lambda M_1)^{-1} (X \beta + M_1 X_1 \gamma) \\
    \delta &= \begin{bmatrix} \lambda & \beta' \end{bmatrix}' \in \mathcal{D} \\
    \phi &= \begin{bmatrix} \alpha & \rho & \delta' \end{bmatrix}' \in \Phi \equiv \mathcal{A} \times \mathcal{R} \times \mathcal{D}
\end{align*}
$$

Note that, for all $i = 1, \ldots, n$, the functions $\varepsilon_i(\cdot; M_1, M_2, X, Y) : \Phi \to \mathbb{R}$ are continuously differentiable on $\Phi$ for each $M_1, M_2, X, Y$. Let

$$
\varepsilon_i(M_1, M_2, X, Y) = \sup_{\phi \in \Phi} \left\{ \left| \varepsilon_i(\phi) \right|, \left\| \frac{\partial \varepsilon_i(\phi)}{\partial \phi} \right\| \right\}
$$

(3.6.23)

and note that, for all $n$, $\sup_{1 \leq i \leq n} E(\varepsilon_i^2) < \infty$. Using the notation of lemma 3.6.28, define $\eta(\phi) = [\varepsilon_1(\phi)^2 \ldots \varepsilon_n(\phi)^2]'$, and note that $\text{tr} \hat{\Sigma} A \hat{\Sigma} B = \eta(\phi_0)' C \eta(\phi_0)$ and $\text{tr} \hat{\Sigma} A \hat{\Sigma} B = \eta(\hat{\phi})' C \eta(\hat{\phi})$, where $C = A + B'$ is an uniformly bounded matrix. Therefore, by the mean-value theorem,

$$
\frac{1}{n} \eta(\hat{\phi})' C \eta(\hat{\phi}) = \frac{1}{n} \eta(\phi_0)' C \eta(\phi_0) + \frac{1}{n} \eta(\phi^*)' C + \frac{\partial \eta(\phi^*)}{\partial \phi} \left( \hat{\phi} - \phi_0 \right)
$$

where $C^* = C + C'$ is an uniformly bounded matrix, and $\phi^*$ is a convex combination of $\hat{\phi}$ and $\phi_0$. Expanding the calculations:

$$
\frac{1}{n} \text{tr} \hat{\Sigma} A \hat{\Sigma} B = \frac{1}{n} \text{tr} \hat{\Sigma} A \hat{\Sigma} B + \sum_{l=1}^{p+3} \epsilon_l - \phi_{l,0} + \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{n} 2 \epsilon_i \epsilon_j \left( \phi^* \right)^2 \frac{\partial \varepsilon_i(\phi^*)}{\partial \phi_l} \left( \hat{\phi} - \phi_0 \right)
$$
therefore, it is enough to show that $\psi_{n,l}$ is bounded in probability, i.e., $\psi_{n,l} = \mathcal{O}_p(1)$.

$$|\psi_{n,l}| \leq \frac{2}{n} \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ji}^s ||\epsilon_j(\phi^*)||^2 |\epsilon_i(\phi^*)| \left| \frac{\partial \epsilon_i(\phi^*)}{\partial \phi_l} \right|$$

recalling equation 3.6.23,

$$|\psi_{n,l}| \leq \frac{2}{n} \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ji}^s \epsilon_j^2 \epsilon_i^2$$

taking expected values,

$$\mathbb{E}|\psi_{n,l}| \leq \frac{2}{n} \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ji}^s \mathbb{E}(\epsilon_j^2 \epsilon_i^2)$$

$$\leq \frac{2}{n} \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ji}^s \left( \mathbb{E}(\epsilon_j^4) \right)^{\frac{1}{2}} \left( \mathbb{E}(\epsilon_i^4) \right)^{\frac{1}{2}}$$

But since $C^s$ is uniformly bounded and $\mathbb{E}(\epsilon_i^4) < \infty$ for all $i = 1, \ldots, n$ and all $n$, there is a constant $\bar{c} < \infty$, independent of $n$, such that $\sum_{j=1}^{n} |c_{ji}^s| < \bar{c}$ and $\mathbb{E}(\epsilon_i^4) < \bar{c}$, thus

$$\mathbb{E}|\psi_{n,l}| \leq \frac{2\bar{c}}{n} \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ji}^s \leq \frac{2\bar{c}}{n} \bar{c} = 2\bar{c}^2 < \infty$$

therefore $\psi_{n,l}$ is bounded in probability (see lemma 3.6.14), and the conclusion $\frac{1}{n} \text{tr} \hat{\Sigma}A\hat{\Sigma}B = \frac{1}{n} \text{tr} \hat{\Sigma}A\hat{\Sigma}B + o_p(1)$ follows. \hfill \Box

**3.6.30 Proposition.** The matrix $\hat{\Omega}$ constructed by replacing $\Sigma_0$ in $\Omega_0$ by $\hat{\Sigma}$ satisfy

$$\text{plim}_{n \to \infty} \hat{\Omega} = \Omega_0$$

**Proof.** The proof follows by noting that each element of the matrix $\hat{\Omega}$ is of the form $\frac{1}{n} \text{tr} \hat{\Sigma}A\hat{\Sigma}B$, where $A, B$ are known uniformly bounded matrices with zeros in the diagonal, and using lemmas 3.6.29 and 3.6.28. \hfill \Box

**3.6.31 Lemma.** Let $\Sigma_0 = \Sigma'\Sigma$ with elements $(\Sigma_0)_{ii} = \sigma_i^2$, let $\hat{\Sigma}$ be a diagonal matrix with elements $(\hat{\Sigma})_{ii} = \hat{\epsilon}_i^2$ and $B$ be a uniformly bounded matrix. Then

$$\text{plim}_{n \to \infty} \frac{1}{n} \hat{\Sigma}B = \lim_{n \to \infty} \frac{1}{n} \text{tr} \Sigma_0B$$

**Proof.** Note that

$$\text{tr} \hat{\Sigma}B = \sum_{i=1}^{n} b_i \hat{\epsilon}_i^2$$

$$\text{tr} \Sigma_0B = \sum_{i=1}^{n} b_i \sigma_i^2$$
thus, \( E\left( \frac{1}{n} \text{tr} \Sigma B \right) = \frac{1}{n} \text{tr} \Sigma B \). Also, the variance is given by

\[
\text{Var}\left( \frac{1}{n} \text{tr} \Sigma B \right) = E\left( \frac{1}{n} \text{tr} \Sigma B - \frac{1}{n} \text{tr} \Sigma B \right)^2 = \frac{1}{n^2} E\left( \sum_{i=1}^{n} b_{ii} (\varepsilon_i^2 - \sigma_i^2) \right)^2
\]

\[
= \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} b_{ii} b_{jj} E(\varepsilon_i^2 \varepsilon_j^2)
\]

where \( \varepsilon_i^2 \equiv \varepsilon_i^2 - \sigma_i^2 \), and \( \varepsilon_i^2 \equiv [\varepsilon_1^2 \cdots \varepsilon_n^2] \) is a vector of zero mean independent random variables; therefore,

\[
= \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} b_{ii} b_{jj} E \varepsilon_i^2 E \varepsilon_j^2 + \frac{1}{n^2} \sum_{i=1}^{n} b_{ii}^2 E \varepsilon_i^4
\]

\[
= 0 + O(n^{-1}) \rightarrow 0
\]

where the last step follows since the innovation \( \varepsilon \) has bounded fourth moment and \( b_{ii} \) is bounded for all \( i = 1, \ldots, n \). Therefore, \( \lim_{n \rightarrow \infty} \frac{1}{n} \text{tr} \Sigma B = \lim_{n \rightarrow \infty} \frac{1}{n} \text{tr} \Sigma_0 B \). \( \Box \)

3.6.32 Lemma. Let \( \hat{\Sigma} \) be a diagonal matrix with elements \( (\hat{\Sigma})_{ii} = \hat{\varepsilon}_i^2 \), \( \hat{\Sigma} \) be a diagonal matrix with elements \( (\hat{\Sigma})_{ii} = \hat{\varepsilon}_i^2 \), \( B, M \) two uniformly bounded matrices, and \( C(M; \phi) \) an uniformly bounded matrix for all \( \phi \in \Phi \), whose elements are continuously differentiable functions of \( \phi \). Let \( \hat{\phi} \) be a consistent estimator of \( \phi_0 \). Then

\[
\lim_{n \rightarrow \infty} \frac{1}{n} \text{tr} \hat{\Sigma} B \hat{C} M = \lim_{n \rightarrow \infty} \frac{1}{n} \text{tr} \hat{\Sigma} B \hat{C} M
\]

Proof. Similarly to lemma 3.6.29 recall that each element of the vector \( \varepsilon(\phi) = A^{-1}_n R_p h(\delta) \) is a continuously differentiable function an let

\[
\varepsilon_i(M_1, M_2, X, Y) = \sup_{\phi \in \Phi} \left\{ |\varepsilon_i(\phi)|, \left\| \frac{\partial \varepsilon_i(\phi)}{\partial \phi} \right\| \right\} \quad (3.6.24)
\]

and note that, for all \( n, \sup_{1 \leq i \leq n} E(\varepsilon_i^2) < \infty \). By assumption, \( C(M; \phi) \) is an uniformly bounded matrix for all \( \phi \in \Phi \), where each element is continuously differentiable. Let

\[
\varepsilon_{ij}(M) = \sup_{\phi \in \Phi} \left\{ |c_{ij}(M; \phi)|, \left\| \frac{\partial c_{ij}(M; \phi)}{\partial \phi} \right\| \right\} \quad (3.6.25)
\]

Note that \( \text{tr} \hat{\Sigma} B \hat{C} M = \text{tr} \Sigma(\phi_0) B \Sigma(\phi_0) M \) and \( \text{tr} \hat{\Sigma} B \hat{C} M = \text{tr} \Sigma(\hat{\phi}) B \Sigma(\hat{\phi}) M \). By the mean-value theorem:

\[
\frac{1}{n} \text{tr} \Sigma(\hat{\phi}) B \Sigma(\hat{\phi}) M = \frac{1}{n} \text{tr} \Sigma(\phi_0) B \Sigma(\phi_0) M
\]

\[
+ \sum_{l=1}^{p+3} \frac{1}{o_{p(1)}} \left( \phi_l - \phi_{l,0} \right) \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} b_{ij} m_{ki} \varepsilon_i(\phi^*) \frac{\partial \varepsilon_i(\phi^*)}{\partial \phi_l} c_{jk}(\phi^*) \psi_{1,n,l}
\]

\[
+ \sum_{l=1}^{p+3} \frac{1}{o_{p(1)}} \left( \phi_l - \phi_{l,0} \right) \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} b_{ij} m_{ki} \varepsilon_i(\phi^*) \frac{\partial c_{jk}(\phi^*)}{\partial \phi_l} \psi_{2,n,l}
\]

\[
\frac{1}{n} \text{tr} \hat{\Sigma} B \hat{C} M = \frac{1}{n} \text{tr} \hat{\Sigma} B \hat{C} M + \sum_{l=1}^{p+3} \frac{1}{o_{p(1)}} (\psi_{1,n,l} + \psi_{2,n,l})
\]
Therefore it suffices to show that $\psi_{1,n,l}$ and $\psi_{2,n,l}$ are bounded in probability.

$$|\psi_{1,n,l}| \leq \frac{2}{n} \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} |b_{ij}||m_{ki}| \alpha_i(\phi^*) \left| \frac{\partial \alpha_i(\phi^*)}{\partial \phi_l} \right| |c_{jk}(\phi^*)|$$

recalling equations 3.6.24 and 3.6.25,

$$|\psi_{1,n,l}| \leq \frac{2}{n} \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} |b_{ij}||m_{ki}| \epsilon_i \epsilon_j$$

taking expected values

$$E|\psi_{1,n,l}| \leq \frac{2}{n} \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} |b_{ij}||m_{ki}| \epsilon_j E \epsilon_i^2$$

But since $B, M$ are uniformly bounded, $E \epsilon_i^2 < \infty$, and $\epsilon_i < \infty$ for all $i, j = 1, \ldots, n$ and all $n$, there is a constant $\bar{c} < \infty$, independent of $n$, such that $\sum_{j=1}^{n} |b_{ij}| < \bar{c}$, $\sum_{k=1}^{n} |m_{ki}| < \bar{c}$, $E \epsilon_i^2 < \bar{c}$, and $\epsilon_j < \bar{c}$; therefore,

$$E|\psi_{1,n,l}| \leq \frac{2\bar{c}^2}{n} \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} |b_{ij}||m_{ki}| \leq \frac{2\bar{c}^2}{n} \bar{c} = 2\bar{c}^3 < \infty$$

(3.6.26)

and thus, by lemma 3.6.14, $\psi_{1,n,l} = O_p(1)$. The conclusion $\psi_{2,n,l} = O_p(1)$ follows by taking the same steps. As a result, $\operatorname{plim}_{n \to \infty} \frac{1}{n} \text{tr} \Sigma BC = \lim_{n \to \infty} \frac{1}{n} \text{tr} \Sigma BC$.

\[\square\]

3.6.33 Proposition. The matrix $\hat{G}$ constructed by replacing $\Sigma_0$ by $\hat{\Sigma}$, $A_0^{-1}$ by $A^{-1}$, and $R_0^{-1}$ by $\hat{R}^{-1}$ in $G_0$, satisfies

$$\operatorname{plim}_{n \to \infty} \hat{G} = G_0$$

Proof. A generic element of the first column of matrix $G_0$ is $\lim_{n \to \infty} \frac{1}{n} \text{tr} \Sigma_0 K_i^* A_0^{-1} M_2$, where $K_i^*, M_2$ are uniformly bounded matrices and $A_0^{-1} = A(M_2; \alpha_0)^{-1}$, where $A(M_2; \alpha)^{-1} = (I - \alpha M_2)^{-1}$ is an uniformly bounded matrix for all $\alpha \in A$ (see lemma 3.5.4), whose elements are continuously differentiable functions of $\alpha$. Therefore the conditions for lemma 3.6.32 are satisfied and it follows $\lim_{n \to \infty} \frac{1}{n} \text{tr} \Sigma K^*_i A(\hat{\alpha})^{-1} M_2 = \lim_{n \to \infty} \frac{1}{n} \Sigma K^*_i A^{-1} M_2$. Now, define $\hat{B} = K_i^* A_0^{-1} M_2$ and observe that $\hat{B}$ is a non-stochastic uniformly bounded matrix (lemma A.2.3). Therefore, using lemma 3.6.31,

$$\lim_{n \to \infty} \frac{1}{n} \Sigma K^*_i A_0^{-1} M_2 = \lim_{n \to \infty} \frac{1}{n} \Sigma \hat{B} = \lim_{n \to \infty} \frac{1}{n} \Sigma_0 \hat{B} = \lim_{n \to \infty} \frac{1}{n} \Sigma_0 K^*_i A^{-1} M_2$$

The same rational applies to the second column of matrix $G_0$, by using $R(M; \rho)^{-1}$ instead of $A(M; \alpha)^{-1}$, thus concluding the proof.

\[\square\]

3.7 Conclusion

The current set of spatial models typically used in both applied and theoretical research is called Cliff & Ord-type of spatial models. A taxonomy for these models has been proposed by Anselin, and it is generally agreed that these set of models are fairly general, encompassing several different types of stochastic processes. These models flexibility is usually praised since
they “allow for spillovers in the endogenous variables, exogenous variables and disturbances” (Kelejian and Prucha, 2009). The main drawback, however, is that these models mix the effects of a single parameter into both the mean and the covariance structure of the spatial stochastic process (Cressie’s critique). In order to circumvent this problem, Anselin (2003) proposes a taxonomy classifying the spatial stochastic process according to whether the spatial effects (or externalities) are local or global, and whether the spatial effects occur in the error term, or in both. The problem with this suggestion, however, is that global and local specifications are non-nested. Anselin himself noted this drawback and speculated about the possibility of considering models, for instance, with local externalities in the error term and global externalities in the covariates. The author also highlighted that none of these possibilities had received attention in the literature and, to my knowledge, this remains an unexplored area both in the theoretical as well as in the applied econometrics.

In order to address Cressie’s critique and with Anselin’s suggestion in mind, I proposed a more general spatial stochastic process that nests local and global models into an encompassing structure. Several spatial stochastic process frequently used in both theoretical and applied research are nested into my proposed model. It is interesting to note, for instance, that one commonly used model (M_{SARAR}) seems to have a hidden interpretation: when looked under the optics of the model I proposed, the M_{SARAR} model implies a local spatial effect in the innovations that is linked to the global spatial effect assumed to the covariates in a very specific way! This is definitely not an interpretation that is evident from the outset. But the model I propose and the M_{SARAR} model are, in general, non-nested which makes them competing models whose validity and relevance needs to be tested empirically.

My contribution to this literature is thus to propose a very general form for a spatial stochastic process and to present a simple procedure for estimating the parameters of the model as well as the asymptotic distribution of the proposed estimators. The theoretical literature on parameter estimation in the Cliff & Ord--type of models has only recently been formalized in a series of papers mostly by Lee, and Kelejian and Prucha. Maximum likelihood estimation has been a common practice but only recently Lee (2004) provided a rigorous analysis of the ML (and QML) estimator for a specific subset of the Cliff & Ord models. Kelejian and Prucha (1999, 1998) proposed generalized moments estimator and a two-stage least squares procedure for parameter estimation. A debate on the the optimal set of instruments and a search for the most efficient estimator and the estimator with better small sample properties quickly followed. These estimation procedures, however, were all based on homoskedastic disturbances, which is a critical problem since spatial units are often heterogeneous in some characteristics (e.g., size) or result from aggregation (e.g., unemployment), which would suggest heteroskedasticity is likely to be present. More recently, Lee, and Kelejian and Prucha extended their earlier models to include heteroskedastic disturbances. Building up on their work, I propose a two step procedure to estimate the parameters of the model, where in the first step the conditional mean parameters are estimated by instrumental variables and the residual of this step is used in a GMM framework to estimate the conditional covariance parameters. The main attractive of this approach vis-a-vis, for instance, the estimation procedure proposed in Lin and Lee (2009) is the low dimensionality of the non-linear optimization step (GMM). This makes the procedure computationally simple even when the dimension of the parameter space is not small. I then proceed to show consistency and asymptotic normality of the proposed estimators. The procedure draws from the contribution of the above mentioned papers, but it is applied to a different model and the tools used to show consistency are, in my view, simpler and easier to follow. Moreover, contrary to what was obtained in Kelejian and Prucha (2009), the asymptotic distribution of the estimator of the mean does not depend on the instrumental variable estimator. This only happens because I used nonlinear least squares procedure to obtain an estimator for the residuals, while Kelejian and Prucha use the instrumental variable. Therefore, the covariance matrix of the GMM obtained by the authors could potentially become simpler following the idea that I developed in this paper.
Chapter 4

Example: Spatial Econometrics Applied to International Trade

4.1 Introduction

The earliest empirical studies in international trade are often attributed to Beckerman (1956), Tinbergen (1962), Linnemann (1966) (see Leamer and Levinsohn, 1995, for a historical account on this subject). However, as Isard and Peck (1954) highlight, the tension between classical trade theory and observation predates these studies. For instance, Isard and Peck recalls Alfred Weber's (1911) criticism that classical theory ignores the fact that “the significant amount of industry which is transport-oriented, and whose geographic distribution, internationally speaking, is governed primarily by considerations of transport cost of raw materials, fuels, and finished products”. Both Isard and Peck, and Beckerman provide evidence that spatial frictions do exist by analyzing the pattern of trade flows vis-a-vis distance.

Tinbergen, the first Nobel Prize winner in economic sciences\(^1\), and Linnemann are probably the pioneers in bringing gravity models to use in international trade (according to Sen and Smith, 1995, gravity models in social sciences have been used since at least 1858). The simple idea of gravity models is to express a flow (exports, imports, or total trade) from country \(i\) to country \(j\) as being directly proportional to their economic mass (e.g., income) and inversely proportional to the distance between them. Tinbergen and Linnemann rationalize the model by arguing that the exports from \(i\) to \(j\) are greater, the larger is the economic size of \(j\) due to demand factors, and the larger is the economic size of the exporting \(i\) due to supply factors. The volume of trade should also decline with transportation costs and with difficulties of commercial contact, and geographical distance is used as a proxy for them. Both authors also include additional variables, such as dummies for common trade agreements, and dummies for countries that share a border.

Gravity models have been vastly investigated in trade literature, in spite of Tinbergen’s caveat in 1962: “...the present analysis is a fairly crude one which clearly needs to be supplemented by further research. The results reported here must be used with a good deal of care; only provisional and qualified conclusions are possible.”). The original aim of the trade gravity models was to compare actual trade volume with “theoretical” expected trade volume in order to draw conclusions on the discrepancies indicating that a country’s trade flows are either receiving preferential treatment or being discriminated against. The assumption is that discriminating trade impediments were an independent stochastic process with zero expected value.

The empirical success of gravity models led Leamer and Levinsohn (1995) to state that they “have produced some of the clearest and most robust empirical findings in economics”, just to add that “paradoxically, they have had virtually no affect on the subject of international economics”.

\(^1\)Shared with Ragnar Frisch in 1969 “for having developed and applied dynamic models for the analysis of economic processes”.

Fortunately, this later complaint has been addressed and the theoretical literature on trade shows that the gravity equation can be generated from a set of economic primitives (see Anderson, 1979; Bergstrand, 1985; Eaton and Kortum, 2002; Anderson and van Wincoop, 2003, among others). All these advances broadened the scope of the gravity equations that started to be used, among others, to assess the impact of trade agreements, monetary unions, exchange rate volatilities, and to test trade theories.

It is interesting to note that, despite the early and mid 1900's debate among classical trade theorists and location theorists (see Isard and Peck, 1954) and the work of Krugman (1991) – who brought to attention the work in economic geography – the spatial econometric techniques failed to be widely incorporated in the traditional gravity models of trade. As noted in chapter 1, the use of models with spatial dependence might be appropriate both in cases where there is some theoretical ground for modeling spatial interaction among observations, or when no such theoretical basis exist but nevertheless the observations can be regarded as having spatial dependence. Anderson and van Wincoop (2003) model, for instance, provide a theoretical argument in favor of using spatial econometric tools. They derive a gravity equation with two additional terms (labeled multilateral resistance), which are functions of all bilateral trade barriers. Therefore, omitting these variables might induce bias. Moreover, if we assume that the bilateral trade barriers are not independent (for instance, trade barrier from country i to j is related to trade barrier from country k to j) then the residuals of the traditional gravity equations would not be independent anymore, as it is often assumed. Even when no such theoretical basis is posted, it is reasonable to think that traditional gravity models are likely to omit some explanatory variables that might be spatially correlated. Think of productivity, for example. It is not explicitly included in the standard gravity equations (nor in its augmented versions) but one may consider that if there is a productivity shock in country i, that will affect all the trade flows between i and the remaining countries, therefore violating the independent residuals assumption. Explicitly considering spatial dependence in trade equations would likely make the proposed models more robust to specification errors, omitted variables, and measuring errors.

Porojan (2001) can be cited as one of the few articles that incorporate spatial econometric tools to gravity equations, and its author claim to be the first to explore the performance of the gravity model when spatial effects are explicitly accounted for. However, her work was harshly criticized by Johnston et al. (2003): "It is always encouraging to see the gravity model used in new contexts, but Porojan’s empirical tests are badly mis-specified, and as such her findings have no import". Carrington (2003)\(^2\) answers back refuting each of the critiques. That criticism was overstated, in my view, and the main point of highlighting that standard least squares (OLS) might be ill-posed to deal with spatial data was unfortunately missed.

The goal of this chapter, therefore, is to provide an informal assessment of the usefulness of adding a spatial dimension to international trade gravity equations. I start by estimating a traditional gravity equation and then use Moran I test statistic to check the residuals for the presence of spatial dependence. The limiting distribution of Moran I test statistic is known to be Gaussian even in the presence of heteroskedasticity (which is likely to be the case) but its dispersion parameter depends on the specific structure of the model. Therefore I perform a residual wild bootstrap of Moran’s I test, which confirms that the residuals of the gravity equation are spatially dependent.

Another interesting aspect of using spatial econometrics to estimate gravity equations relates to the construction of the neighborhood structure. Section 1.3.2 describes the importance of defining a neighborhood structure (or neighborhood matrix, or spatial weight matrix, or spatial connectivity matrix) in the standard spatial econometrics models (lattice spatial stochastic processes). When the observations are located in a metric space (e.g., countries in a map) there are several algorithms that can be used to build a neighborhood structure. Most of these algorithms are borrowed from Graph Theory (see section 1.3.3), but gravity equations adds an

\(^2\)Who had previously published under the name A. Porojan.
additional complexity for building the spatial weight matrix, since for a given set of \( n \) countries there are \( n(n - 1) \) flows from country \( i \) to country \( j \) and one needs to specify a dependence structure among the flows, not the countries. LeSage and Pace (2008) provides a clever way of building these dependence structures (or spatial weight matrices). The authors suggest a simple way of building an origin-based spatial dependence (for the flows sharing the same origin), a destination-based spatial dependence (sharing the same destination), and an origin-to-destination dependence structure, which would reflect an average of flows from neighbors to the origin to neighbors of the destination.

Beyond investigating the standard spatial econometric models (SAR, SARAR), I also apply the procedure developed in chapter 3 to estimate the new class of spatial autoregressive moving average processes that is suggested in my dissertation. I do not attempt to qualify the estimated parameters nor to test trade models, but only to check, with a simple example, whether the spatial econometrics toolbox would be appropriate to deal with international trade gravity equations.

### 4.2 Traditional Gravity Equations

The starting point of this exercise is the typical augmented gravity model used in the literature (see Rose, 1999, for example).

\[
\ln(f_{ij}) = \alpha_0 + \alpha_1 \ln(y_i) + \alpha_2 \ln(y_i C) + \alpha_3 \ln(y_j) + \alpha_4 \ln(y_j C) + \alpha_5 \ln(dist_{ij}) + \alpha_6 Cont_{ij} + \alpha_7 Lang_{ij} + \alpha_8 Rta_{ij} + \alpha_9 Cn_{ij} + \alpha_{10} Cc_{ij} + \alpha_{11} Col_{ij} + \epsilon_{ij}
\]  

(4.2.1)

where \( i \) and \( j \) denotes the countries, and the remaining variables and their sources are described below:

- \( f_{ij} \) is the flow (exports, imports) from country \( i \) to country \( j \) for the year 2003;
- \( y_i \) is real GDP at PPP, and \( y_i C \) is real GDP per capita;
- \( dist_{ij} \) is the geodesic distance between the main cities in countries \( i \) and \( j \);
- \( Cont_{ij} \) is a dummy variable indicating whether countries \( i,j \) are contiguous;
- \( Lang_{ij} \) is a dummy variable indicating whether the two countries share a common language, which is defined either as the official country language or a language spoken by an important share of the population (e.g., US and Mexico share a common language);
- \( Rta_{ij} \) is a dummy variable which is one if countries \( i \) and \( j \) belong to the same regional trade agreement (e.g., NAFTA, Mercosur, EU); and
- \( Cn_{ij}, Cc_{ij}, \) and \( Col_{ij} \) indicate whether the two countries, respectively, are part of the same nation, have had a common colonizer after 1945, and have had a colonial relationship after 1945.

The above data comes from the CEPII gravity dataset (Head et al., 2010), from the Penn World Tables (Heston et al., 2009), and from the UN Comtrade database. The countries used consist of the largest 33 countries (by PPP adjusted GDP in 2003) in the world: Argentina, Australia, Bangladesh, Belgium, Brazil, Canada, China, Colombia, Egypt, France, Germany, India, Indonesia, Iran, Italy, Japan, Malaysia, Mexico, Netherlands, Pakistan, Philippines, Poland, Rep. of Korea, Russian Federation, Saudi Arabia, South Africa, Spain, Sweden, Thailand, Turkey, Ukraine, United Kingdom, United States. These countries represented around 86% of the world GDP in 2003. There are 1056 (33*32) observed trade flows, and for each country \( i \), the remaining countries \( j \) in the list represent between 53% (Iran) and 94% (Mexico and Canada) of total trade flows (with a median value of 75%).
Table 4.1 contains the estimated coefficients of the traditional augmented gravity equation for this dataset, using export flows. Since heteroskedasticity is often a concern in this cross section setting, an heteroskedasticity-consistent estimation of the covariance matrix is reported (using HC3, see Davidson and MacKinnon, 1993) together with OLS estimates of the coefficients. In this simple example, the gravity equation explained almost 70% of the total variation in the regressand, which is in line with the well-documented result of the good performance of this model in explaining trade flows. Using import flows, the results are qualitatively the same, except for the dummy variables “contiguity” and “language” which had its significance changed, as shown in table 4.2.

Table 4.1: Traditional gravity equation – export flows

|        | Estimate | Std. Error | t value | Pr(>|t|) |
|--------|----------|------------|---------|----------|
| (Intercept) | -24.8474 | 1.4024 | -17.72 | 0.0000 *** |
| Yo      | 0.9159   | 0.0403   | 22.74   | 0.0000 *** |
| YoC     | 0.9357   | 0.0502   | 18.64   | 0.0000 *** |
| Yd      | 0.8316   | 0.0422   | 19.72   | 0.0000 *** |
| YdC     | 0.7748   | 0.0536   | 14.44   | 0.0000 *** |
| Dist    | -0.8020  | 0.0627   | -12.80  | 0.0000 *** |
| Cont    | 0.2532   | 0.1642   | 1.54    | 0.1231   |
| Lang    | 0.1897   | 0.1050   | 1.81    | 0.0708   |
| Rta     | 0.5899   | 0.1834   | 3.22    | 0.0013 ** |
| Cn      | -0.1225  | 0.5736   | -0.21   | 0.8309   |
| Cc      | 0.4118   | 0.6070   | 0.68    | 0.4976   |
| Col     | 0.3543   | 0.1649   | 2.15    | 0.0316 *  |

Signif. codes: ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ’.’ 0.1
Residual standard error: 1.298 on 1044 degrees of freedom
Multiple R-squared: 0.6924, Adjusted R-squared: 0.6892
F-statistic: 213.7 on 11 and 1044 DF, p-value: < 2.2e-16
Table 4.2: Traditional gravity equation – import flows

| Estimate | Std. Error | t value | Pr(>|t|) |
|----------|------------|---------|----------|
| (Intercept) | -25.3878 | 1.3545 | -18.74 | 0.0000 *** |
| Yo | 0.8281 | 0.0414 | 20.01 | 0.0000 *** |
| YoC | 0.8494 | 0.0515 | 16.48 | 0.0000 *** |
| Yd | 0.9185 | 0.0383 | 23.98 | 0.0000 *** |
| YdC | 0.8245 | 0.0490 | 16.82 | 0.0000 *** |
| Dist | -0.6824 | 0.0618 | -11.04 | 0.0000 *** |
| Cont | 0.3999 | 0.1627 | 2.46 | 0.0140 * |
| Lang | 0.0773 | 0.1036 | 0.75 | 0.4556 |
| Rta | 0.6241 | 0.1781 | 3.51 | 0.0005 *** |
| Cn | -0.3958 | 0.5062 | -0.78 | 0.4342 |
| Cc | 0.4568 | 0.5313 | 0.86 | 0.3899 |
| Col | 0.3430 | 0.1466 | 2.34 | 0.0193 * |

Signif. codes: '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1

Residual standard error: 1.241 on 1044 degrees of freedom
Multiple R-squared: 0.6993, Adjusted R-squared: 0.6961
F-statistic: 220.7 on 11 and 1044 DF, p-value: < 2.2e-16

4.3 Spatial Dispersion of the Residuals – Exploratory Analysis

In order to investigate the presence of spatial dependence in the residuals of the gravity equation (eq. 4.2.1) I proceed first with a visual inspection of the residuals. The idea is to plot the spatial pattern of the residuals of the gravity equation for each of the countries considered. This conveys several information in one picture. Consider, for instance, figure 4.1 which shows the residuals of the gravity equation for Argentina (the referenced country is not plotted in the map). Each of Argentina’s trade partners is highlighted in a gray scale, ranging from light gray for residuals in the bottom 20 percentiles to black in the top 20 percentiles. From the legend we can see, for instance, that the median residual for Argentina’s exports is positive, which shows that, on average, the gravity equation tends to underestimate the country’s exports. The opposite happens to imports, where the model overestimates, on average, the actual imports. Figure 4.1 also highlights that the Argentinean export flows are underestimated by the gravity equation for Bangladesh, China, Egypt, India, Netherlands, Philippines, and Thailand, which (except for Netherlands) are geographically close, thus raising the question whether or not the residuals could present signs of spatial correlation.

For imports, one can also observe that, except for Canada, the other countries for which the model overestimates the imports flow (Bangladesh, Egypt, Iran, Saudi Arabia, Turkey, Ukraine) are also clustered together. As an additional example, consider Poland. Figure 4.4 shows that its exports are higher than predicted by the gravity model for Belgium, Germany, Italy, Netherlands, Sweden, Turkey and Ukraine, which again suggest evidence of spatial agglomeration. Additional examples for Australia (figure 4.2) and Brazil (figure 4.3) suggest that this pattern might be the rule rather than exception. One interesting curiosity relates to China. It appears as an outlier for all the countries imports, i.e., imports from China were larger than what was predicted by using the gravity equations. In the spirit of Tinbergen’s remarks, this discrepancy would mean China’s imports were receiving preferential treatment from all the other countries.

\[^3\text{The charts for the remaining countries are available upon request.}\]

\[^4\text{Note that bilateral real exchange rate has not been included in the gravity model.}\]
A less informal assessment of the degree of spatial correlation is left to section 4.4 where the presence of spatial correlation in the residuals is tested for using Moran I test, using several proposed neighborhood structures for bilateral trade flows.

Figure 4.1: Argentina – residuals from standard gravity equation
Figure 4.2: Australia – residuals from standard gravity equation
Figure 4.3: Brazil – residuals from standard gravity equation
Figure 4.4: Poland – residuals from standard gravity equation
As a final remark, Table 4.3 resembles the ones published in Tinbergen (1962). It is clear that, despite the high \( R^2 \) of the OLS estimate, still a material portion of the variance is not explained and the gravity equations overestimates total export flows by almost 20\% (US\$ 1.1 trillion of US\$ 5.7 tn) and total import flows by about 8\%.

**Table 4.3: Total actual trade minus total calculated trade (2003)**

<table>
<thead>
<tr>
<th>Country</th>
<th>Export deviation</th>
<th>Import deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>US$ bn</td>
<td>% of actual value</td>
</tr>
<tr>
<td>Argentina</td>
<td>7.18</td>
<td>34.59</td>
</tr>
<tr>
<td>Australia</td>
<td>18.69</td>
<td>36.07</td>
</tr>
<tr>
<td>Bangladesh</td>
<td>2.98</td>
<td>50.68</td>
</tr>
<tr>
<td>Belgium</td>
<td>16.66</td>
<td>7.69</td>
</tr>
<tr>
<td>Brazil</td>
<td>32.63</td>
<td>56.47</td>
</tr>
<tr>
<td>Canada</td>
<td>-312.92</td>
<td>-118.61</td>
</tr>
<tr>
<td>China</td>
<td>199.07</td>
<td>66.92</td>
</tr>
<tr>
<td>Colombia</td>
<td>2.32</td>
<td>25.80</td>
</tr>
<tr>
<td>Egypt</td>
<td>-5.16</td>
<td>-140.70</td>
</tr>
<tr>
<td>France</td>
<td>-325.53</td>
<td>-119.00</td>
</tr>
<tr>
<td>Germany</td>
<td>-192.28</td>
<td>-35.82</td>
</tr>
<tr>
<td>India</td>
<td>7.06</td>
<td>17.17</td>
</tr>
<tr>
<td>Indonesia</td>
<td>37.18</td>
<td>77.58</td>
</tr>
<tr>
<td>Iran</td>
<td>0.74</td>
<td>5.45</td>
</tr>
<tr>
<td>Italy</td>
<td>-30.76</td>
<td>-14.54</td>
</tr>
<tr>
<td>Japan</td>
<td>173.46</td>
<td>48.54</td>
</tr>
<tr>
<td>Malaysia</td>
<td>58.41</td>
<td>82.22</td>
</tr>
<tr>
<td>Mexico</td>
<td>109.68</td>
<td>69.31</td>
</tr>
<tr>
<td>Netherlands</td>
<td>-73.69</td>
<td>-40.60</td>
</tr>
<tr>
<td>Pakistan</td>
<td>3.23</td>
<td>39.06</td>
</tr>
<tr>
<td>Philippines</td>
<td>22.21</td>
<td>81.64</td>
</tr>
<tr>
<td>Poland</td>
<td>13.06</td>
<td>33.35</td>
</tr>
<tr>
<td>Rep. of Korea</td>
<td>76.91</td>
<td>54.27</td>
</tr>
<tr>
<td>Russian Federation</td>
<td>-5.73</td>
<td>-7.25</td>
</tr>
<tr>
<td>Saudi Arabia</td>
<td>56.46</td>
<td>77.52</td>
</tr>
<tr>
<td>South Africa</td>
<td>12.50</td>
<td>56.61</td>
</tr>
<tr>
<td>Spain</td>
<td>-57.08</td>
<td>-50.17</td>
</tr>
<tr>
<td>Sweden</td>
<td>15.02</td>
<td>22.02</td>
</tr>
<tr>
<td>Thailand</td>
<td>46.58</td>
<td>80.46</td>
</tr>
<tr>
<td>Turkey</td>
<td>19.36</td>
<td>61.63</td>
</tr>
<tr>
<td>Ukraine</td>
<td>0.39</td>
<td>2.78</td>
</tr>
<tr>
<td>United Kingdom</td>
<td>-295.23</td>
<td>-126.94</td>
</tr>
<tr>
<td>United States</td>
<td>-733.58</td>
<td>-124.85</td>
</tr>
</tbody>
</table>

Total          | -1100.20 | -469.78
4.4 Moran I Test for Spatial Correlation

As a preliminary check for the presence of spatial correlation in the residuals of the gravity equation (equation 4.2.1) I will use the Moran I test statistic, defined as:

$$ I = \frac{N}{\sum_{i,j} m_{ij}} \bar{\hat{e}}' M \bar{\hat{e}} $$

(4.4.1)

where $\hat{e}$ is the residual of the gravity equation, $N$ is the sample size, $M$ is the spatial weight matrix (Moran, 1950; Cliff and Ord, 1981b).

Under the null hypothesis of no spatial correlation, Moran I test statistic is asymptotically normal and under the alternative the test’s power tends to unity (Kelejian and Prucha, 2001). Moran’s I test is regarded as a “diffuse test” indicating that rejecting the null hypothesis is indicative of spatial dependence, but the test do not point towards any specific alternative (Florax and de Graaff, 2004). Indeed, Burridge (1980) noted that Moran’s I test is equivalent to a Lagrange multiplier test when the alternative hypothesis is either a spatial autoregressive or a spatial moving-average stochastic process. Florax and de Graaff (2004) also indicates that Moran I test is not sensitive to departures from a Gaussian error distribution, however the same study also suggests that it lacks power in the presence of heteroskedasticity. However, Kelejian and Robinson (2004) results suggest that Moran I test is valid in the presence of heteroskedasticity provided heteroskedasticity itself is not spatially correlated. Even when the heteroskedastic structure is spatially correlated the authors show that, under some regularity assumptions, the test statistic is still asymptotically normal under the null of no spatial autocorrelation, albeit with a different dispersion. It is likely that the distribution of Moran I statistic, given the above mentioned concerns, will not be well approximated by the asymptotic distribution, even in the current case of a sample of $N = 1056$ bilateral trade flows. Thus, I resort to bootstrap technique, which is explained in details in section 4.4.4. Sections 4.4.1 and 4.4.2 detail how the neighborhood structure for bilateral trade flows were constructed and section 4.4.3 explores visually the spatial correlation in the residuals of the gravity equation.

4.4.1 Neighborhood Structure for Flows

The classical way to look at the spatial dependence structure in lattice spatial stochastic processes is to impose, a priori and based on specific knowledge of the problem being investigated, a structure to the observations based on the location of the spatial units in the space. In the current case, the spatial units or observations, are trade flows from country $i$ to country $j$, rather than the countries themselves. Several possibilities might be considered to come up with a neighborhood structure for these trade flows, but since the goal of the current study is just to provide an informal assessment of the usefulness of the spatial toolbox to estimate gravity equations, I will simply adapt the methodology used in LeSage and Pace (2008).

Assume there are $i = 1, \ldots, n$ spatial units (countries) and that we observe the $n^2$ trade flows $f_{ij}$ from country $i$ to country $j$ (note here that intra country flows are considered; later on this will be removed from the model since they are not observed). Let $F$ be an $n^2 \times 1$ vector defined as $F' \equiv [f_{11} \ f_{12} \ \cdots \ f_{1n} \ \cdots \ f_{n1} \ \cdots \ f_{nn}]$, so the first $n$ elements of this vector represent trade flows from country 1 to the remaining countries, the second set of $n$ elements represent trade flows from country 2, and so on. The goal is to propose a neighborhood structure relating trade flows $f_{ij}$ to $f_{kl}$. The methodology proposed in LeSage and Pace (2008) relates this neighborhood structure to the neighborhood structure of the underlying spatial units (countries) in a straightforward way. Let $M$ be an $n \times n$ matrix representing the neighborhood structure of the countries (e.g., a matrix where $m_{ij} = 1/n_i$ if countries $i$ and $j$ are neighbors, $m_{ij} = 0$ otherwise, and $n_i$ chosen to impose $\sum_j m_{ij} = 1$, i.e., a row-normalized matrix).

---

5 In the bootstrap test (section 4.4.4) the sample size $N$ is adjusted to take into account that some spatial units might have no neighbors.
At least two neighborhood structures could be proposed. The first one is designed to capture the destination-based dependence structure and is constructed as \( M_d = I \otimes M \). Then the spatially lagged \( M_d F \) is a vector containing the spatial average of flows around each destination, keeping the origin constant\(^6\). To see this, decompose \( F' = [F_1' \cdots F_n'] \), where \( F_i' = [f_{i1} \ f_{i2} \cdots \ f_{in}] \), and note that (using the same ordering of indices than in vector \( F \)):

\[
(M_d F)_{ij} = (MF)_{ij} = \sum_{k=1}^{n} m_{jk} f_{ik}
\]

which could be interpreted as: the spatial lag of trade flows from country \( i \) to country \( j \) is the weighted average of trade flows from \( i \) to \( k \), where country \( k \) is a neighbor of the destination country \( j \). In other words, \( M_d \) represents the association between the origin and the destination’s neighbors. In this case, the presence of spatial correlation between \( F \) and \( M_d F \) would represent the extent to which the flows from \( i \) to \( j \) are “enhanced or diminished in accordance with the propensity of attractiveness of its [meaning, \( j \)] neighboring destination locations” (see Griffith and Jones, 1980). Translating this notion of destination-based dependence to the residuals of the gravity equation, for instance, means that the residuals \( \varepsilon_{ij} \) and \( \varepsilon_{ik} \) would be correlated whenever \( k \) and \( j \) are neighbors.

The second neighborhood structure is designed to capture the origin-based dependence structure and is constructed as \( M_o = M \otimes I \). In this case

\[
(M_o F)_{ij} = \sum_{k=1}^{n} m_{ik} f_{kj}
\]

and the interpretation is: the spatial lag of trade flows from \( i \) to \( j \) is the weighted average of trade flows from \( k \) to \( j \), where country \( k \) is a neighbor of the origin country \( i \). In this case, the presence of spatial correlation between \( F \) and \( M_o F \) represents the degree that trade flows from \( i \) to \( j \) are “enhanced or diminished in accordance with the propensity of emissiveness of its neighboring origin locations” (Griffith and Jones, 1980). This would mean, in the case of the residuals of the gravity equation, that \( \varepsilon_{ij} \) would depend on \( \varepsilon_{kj} \) whenever countries \( k \) and \( i \) are neighbors.

Finally, the \( n^2 \times n^2 \) matrices \( M_d, M_o \) need to be adjusted to accommodate the fact that the intra-country flows \( f_{ii}, i = 1, \ldots, n \) are not observed. Therefore the dimension of the neighbor matrices need to be reduced to match the \( n \times (n-1) \) dimension of the \( F \) vector of bilateral trade flows. This is done simply by eliminating the \( n \) row and columns \((1, n+2, 2n+3, 3n+4, \ldots, n^2)\) of matrices \( M_d, M_o \).

### 4.4.2 Neighborhood Structure for Countries

In the previous section, a neighborhood structure for bilateral trade flows was suggested in order to capture either destination-based or origin-based dependence structures. Both were based on an underlying neighborhood structure for the countries (spatial units), which, by itself, can be generated in a myriad of ways. Since the focus of this chapter is not on any underlying theoretical model for the bilateral trade flows, I will concentrate on purely geographical definitions of dependence structure. This also avoids the risk of introducing endogeneity in the definition of the neighborhood structure if the neighboring relation were to be defined based on some other profile of country’s characteristics. For instance, the “space” where countries are located could, in principle, be based on some characteristics (e.g., income distribution, average education, natural resources), a “distance” between countries could be defined in this space, and a neighborhood relation could be based on this metric space. However, if any of these characteristics are correlated

---

\(^6\)This interpretation is allowed when the \( M \) matrix is row-normalized; for different specifications of \( M \) it is better to interpret it as a weighted sum, where the weights are given by the elements of \( M \).
with the determinants of bilateral trade flows that are omitted from the gravity equation, then spatial matrices $M_d, M_o$ could not be assumed exogenous anymore.

I will consider 11 designs for the spatial weight matrix $M$, whose characteristics are described in Table 4.4. The neighborhood structure and maps were constructed using the routines supplied in Bivand et al. (2010); Pebesma and Bivand (2005); Bivand et al. (2008). Figures A.1 – A.11 show the graphs generated by the neighborhood structure depicted in designs 1 to 11. Each of the proposed designs originates two slightly different spatial weight matrices: binary and row-normalized. Therefore, there is a total of 22 spatial weight matrices for the countries and, for each, there are two spatial weight matrices for the trade flows, one origin-based and the other destination-based. Each of these 44 matrices will be used in Moran I test (equation 4.4.1) to test for the presence of spatial correlation in the residuals of the gravity equation.

4.4.3 Moran Plot

In the previous sections I described how the spatial weight matrices that represent the neighborhood of bilateral trade flows were constructed. Before moving on to using Moran’s I test to more formally investigate the presence of spatial dependence in the residuals of the gravity equation, it is worth to explore visually the residuals spatial correlation using the suggested neighborhood structures. Figures A.12–A.33 contain the scatter-plot of the bilateral flows $F$ and the “spatially lagged” bilateral flows $M_i F$, $i = \{d, o\}$. Each figure is split in four panels showing:

(a) destination-based neighborhood structure based on binary spatial weight matrices for the countries, (b) origin-based neighborhood structure based on binary spatial weight matrices, and (c) and (d) replace the binary weight matrices by row-normalized ones. Figures A.12–A.22 use bilateral export flows, while figure A.23–A.33 use bilateral imports. Each figure also plots a regression line of $M_i F$ in $F$, and its robust version (function rlm() in Venables and Ripley, 2002).

What is clear from all the figures is that, regardless of the countries' underlying neighborhood structure, there is evidence of spatial correlation in the residuals of the gravity equation. Another regularity observed is that, for each neighborhood design (Table 4.4), destination-based matrices $M_d$ tend to capture better the spatial correlation in the residuals of the export equation, and origin-based matrices $M_o$ tend to capture better the spatial correlation in the residuals of the import equation.

4.4.4 Bootstrapping Moran I Test

In this section I will briefly describe the wild bootstrap procedure used in the current context to investigate whether spatial autocorrelation is present in the residuals of the gravity equation (equation 4.2.1). Consider the general spatial model $M$, which is described in details in Chapter 3, adapted to the the traditional gravity equation:

$$F = g(\delta; M_1, X) + U$$

$$g(\delta; M_1, X) \equiv (I - \lambda M_1)^{-1} \times \beta, \quad \delta \equiv \left[ \lambda \quad \beta \right]^{\top}$$

$$U = (I - \rho M_2)^{-1} \times (I - \alpha M_2) \varepsilon, \quad E \varepsilon \varepsilon' = \Sigma \text{ (diagonal)}$$

(4.4.2)

where $F$ is an vector of observations on $n \times (n - 1)$ bilateral trade flows, $X$ is an $n(n - 1) \times 12$ matrix of regressors for the gravity model, described in equation 4.2.1, $g(\cdot)$ is the regression function that determines the mean value of $F$ conditional on $\delta$ and on the stochastic variables in $X$. $M_1, M_2$ are $n(n - 1) \times n(n - 1)$ matrices of known constants (see section 4.4.1) representing, respectively, the spatial dependence structure in the covariates and in the innovations $\varepsilon$. The parameters $\beta \in \mathbb{R}^{12}$ are the standard parameters in typical linear regression models, $\lambda \in \mathbb{R}$ is the global spatial effect in the covariates, and $\rho, \alpha \in \mathbb{R}$ are the global and local spatial effects in the error structure. Let $\phi: \mathcal{M} \to \mathcal{A} \times \mathcal{R} \times \mathcal{L} \times \mathbb{R}^{12}_+ \times \mathbb{R}^{n(n-1)}_{++}$ be the parameter-defining
map of the model \( \mathcal{M} \). Thus, for each DGP \( m_0 \in \mathcal{M} \) there is an associated parameter vector \( \phi_0 = \phi(m_0) = [\alpha_0, \rho_0, \lambda_0, \beta_0^0 (\text{diag } \Sigma_0)]' \).

The testing problem under consideration is defined by partitioning the parameter space in two disjoint sets: \( \Phi_0 = \{0\} \times \{0\} \times \{0\} \times \mathbb{R}_+^{12} \times \mathbb{R}_+^{(n-1)} \) and \( \Phi_1 = \mathcal{A} \setminus \{0\} \times \mathcal{R} \setminus \{0\} \times \mathcal{L} \setminus \{0\} \times \mathbb{R}_+^{12} \times \mathbb{R}_+^{(n-1)} \), where \( \Phi_0 \) represents the absence of spatial dependence in model \( \mathcal{M} \). The goal of the testing procedure is, given a realization \((F, X)\) of the spatial process, to decide in which set \( \Phi_0 \) or \( \Phi_1 \) the underlying data generating process lies.

The \( \alpha \)-size Moran I test (MI for short) for testing the presence of spatial dependence \((H_0 : \phi \in \Phi_0 \text{ vs. } H_1 : \phi \in \Phi_1)\) can be summarized as

\[
\text{MI}(F, X) = \begin{cases} 
1 & |I| > c_\alpha \\
0 & |I| < c_\alpha 
\end{cases}
\]  

(4.4.3)

where \( c_\alpha \) is the \( 1 - \alpha \) quantile of the distribution of \( I \) (Moran’ I statistic) defined in equation 4.4.1, and \( \text{MI}(F, X) = 1 \) means that the null hypothesis \( H_0 \) is rejected. The \( p \)-value of an one-sided test of \( I \) is \( p(I) = 1 - F_I(I) \), where \( F_I(\cdot) \) is the cumulative distribution function of \( I \) under the null hypothesis. Although the limiting distribution of \( I \) is known to be normal even in the presence of unknown heteroskedasticity (Kelejian and Robinson, 2004; Kelejian and Prucha, 2001), its dispersion parameter depends on the specific structure of the model. Therefore a residual wild bootstrap test can be performed (MacKinnon, 2007). Assuming the residuals of the spatial gravity equation (equation 4.4.2) are independent but possibly heteroskedastic, the bootstrap data generating process is given by

\[
F^* = g(\hat{\delta}) + \varepsilon \bar{\varepsilon} \ast \nu^*
\]  

(4.4.4)

where \( \hat{\delta} \) is an estimator of \( \delta \) in equation 4.4.2 under \( H_0 \), \( \varepsilon \bar{\varepsilon} \) are the residuals, \( e(\varepsilon) \) is a function of the residuals, and \( \nu^* \) is a random variable with zero mean and unit variance.

Following Davidson and Flachaire (2001), I define \( e_i(\varepsilon_i) = \frac{\varepsilon_i}{\sqrt{h_i}}, i = 1, \ldots, n(n-1) \), where \( h_i \) is the \( i \)-th element of the diagonal of the hat matrix \( X(X'X)^{-1}X' \). The distribution of \( \nu^* \) is assumed to be the Rademacher distribution, where \( \nu^* = 1 \) with probability \( \frac{1}{2} \) and \( \nu^* = -1 \) otherwise (see Davidson and Flachaire, 2001, for an argument in favor of using this distributions vs. the popularly used distribution suggested in Mammen (1993)).

In practice, the wild bootstrap procedure consist of producing \( B \) bootstrap samples that satisfy the null hypothesis by generating \( B \) draws from the Rademacher distribution, and for each \( b = 1, \ldots, B \) compute \( F^*_b = (X\hat{\delta} + e(\varepsilon) \ast \nu^*) \), estimate the regression 4.2.1 and compute the bootstrap statistic \( I_b^* \) with the new residuals. If one wants to reject the null \( H_0 \), the (equal tail) bootstrap p-value is

\[
p^*(I) = 2 \min \left\{ \frac{1}{B} \sum_{b=1}^{B} I(I_b^* \leq I), \frac{1}{B} \sum_{b=1}^{B} I(I_b^* > I) \right\}
\]  

(4.4.5)

and the null would be rejected if \( p^*(I) < \alpha \).

This procedure is implemented for the gravity equation and the proposed neighborhood matrices \( M_d, M_r \), for designs 1 to 11 (see table 4.4), for both binary and row-normalized spatial matrices, using the tools described in Davison and Hinkley (1997); Canty and Ripley (2009). There are thus 88 combinations: 2 trade flows (exports, imports), 2 spatial neighborhood structures for trade flows (origin-based, destination-based), and 22 neighborhood structures for countries (binary and row-normalized matrices for each of the 11 spatial designs). For all of them, the above described bootstrap procedure, using \( B = 1999 \), agree with the asymptotic result for Moran I test in rejecting the null hypothesis of no spatial dependence in the residuals, with the I statistic in the [0.13, 0.52] range and \( p \)-value (\( p^*(I) \)) zero in all cases. Also, the visual impression that for export flows the spatial dependence were higher when using destination-based designs
for the neighborhood structure (figures A.12–A.22) is confirmed by Moran I statistic. It ranges from 0.28 to 0.52 (median: 0.44) when destination-based matrices are used, and from 0.13 to 0.26 (median: 0.23) when origin-based matrices are used. The opposite happens when bilateral import flows are used as the regressand.

Summing up, the bootstrap of Moran I test confirms that the residuals of the gravity equation are spatially dependent, and that this was not the result heteroskedastic innovations.

### 4.5 General Spatial Model

In this section I use the framework of the traditional gravity equations in international trade to illustrate the general spatial model (GSM) introduced in Chapter 3.

The structural form of the GSM model, adapted to the current setting of modeling bilateral trade flows, is given by:

\[
\begin{align*}
F &= (I - \lambda M)^{-1} (X_1 \beta_1 + X_2 \beta_2) + U \\
U &= (I - \rho M)^{-1} (I - \alpha M) \varepsilon \\
E \varepsilon \varepsilon' &= \Sigma, \quad \Sigma(\text{diagonal})
\end{align*}
\]

where \( F \) is an vector of observations on the logarithm of \( n \times (n - 1) \) bilateral trade flows \( (f_{ij}) \), \( X \equiv [X_1 \ X_2] \) is an \( n(n - 1) \times 12 \) matrix of regressors for the gravity model (eq. 4.2.1) where the covariates \( X \) are divided in the the stochastic regressors, \( X_1 \), containing the log of real GDP \( (y_i) \) and log of real per capita GDP \( (y_i/c) \), and \( X_2 \) are the non-stochastic regressors, consisting of a constant, logarithm of the distance between countries \( (Dist_{ij}) \), and the dummy variables described in section 4.2. \( M \) is an \( n(n - 1) \times n(n - 1) \) matrix of known constants (see section 4.4.1) representing the suggested spatial dependence structure in the model. The parameters \( \beta \equiv [\beta_1 \ \beta_2] \in \mathbb{R}^{12} \) are the standard parameters in typical linear regression models, \( \lambda \in \mathbb{R} \) is the global spatial effect in the covariates, and \( \rho, \alpha \in \mathbb{R} \) are the global and local spatial effects in the error structure. Let \( \phi : \mathbb{M} \to \mathbb{A} \times \mathbb{R} \times \mathbb{L} \times \mathbb{R}_+^{12} \times \mathbb{R}_+^{n(n - 1)} \) be the parameter-defining map of the model \( \mathbb{M} \). Thus, for each DGP \( m_0 \in \mathbb{M} \) there is an associated parameter vector \( \phi_0 = \phi(m_0) = [\alpha_0 \ \rho_0 \ \lambda_0 \ \beta'_0 (\text{diag } \Sigma_0)]' \).

#### 4.5.1 Estimation

In order to estimate model \( \mathbb{M} \) in equation 4.5.1, I will use the procedure explained in Chapter 3, sections 3.5 and 3.6. The instruments needed to estimate \( \lambda, \beta \) are \( H \equiv [X \ MX_1 \ M^2 X_1] \), and five moment conditions are used to estimate \( \alpha, \rho \). These moment conditions are based on matrices \( K_i = M^i - \text{diag}(M^i), i = 1, \ldots, 5 \) (see eq. 3.6.7), where \( M \) are origin-based or destination-based spatial weight matrices defined in section 4.4.1. The codes used to estimate the model was developed in R (R Development Core Team, 2011) and the main functions are listed in appendix A.4.

I have estimated the gravity equation with spatial effects (eq. 4.5.1) for export and import flows using all the 11 neighborhood structures for countries suggested in table 4.4. For each neighborhood structure two spatial matrices were created for the flows: a destination-based matrix and an origin-based matrix (see section 4.4.1). As suggested by the visual inspection of the Moran plots in figures A.12–A.33 export flows were better fit using destination-based matrices, while import flows were better described by origin-based neighborhood structure.

The overall fit of the model (including the estimated coefficients) was very similar across all the neighborhood structures used, but the ones with designs 2, 3, 4, and 6 were slightly better (see table 4.4). They were based on the graph structure "triangulation", "sphere of influence", "Gabriel", and "nearest neighbor 1" respectively. Figures A.2, A.3, A.4, and A.6 in the appendix show the graph structure of these neighborhood structures overlapping in a world map.
Tables 4.5 and 4.5 contain the estimated coefficients of the general spatial model applied to gravity equation for exports and imports using designs 2 and 3 for the neighborhood structure. Results using the other neighborhood structures produce similar estimates. As expected, the $R^2$ of the spatial model is larger than the traditional gravity equation. The F-statistic for testing $\lambda = \alpha = \rho = 0$ clearly rejects the null. The global spatial effect in the covariates ($\lambda$), despite being significant, is close to zero. The global ($\rho$) and local ($\alpha$) spatial effect in the error structure are more pronounced, as expected by the visual inspection of the Moran plot in the previous section. Despite the clear presence of a spatial effect, the estimated coefficients for the remaining parameters are very similar to the ones estimated without using the spatial structure. The coefficients for real GDP and real GDP per capita for country of origin and country of destination are practically the same. The coefficient for distance between two countries is also very similar in the two specifications of the gravity equations.
Table 4.4: Neighborhood structures

<table>
<thead>
<tr>
<th>Design*</th>
<th>Non-zero weights (%)</th>
<th>Average number of links</th>
<th>Range of links</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 - Contiguity</td>
<td>4.41</td>
<td>1.45</td>
<td>[0, 4]</td>
<td>The neighborhood structure is based on regions with contiguous boundaries, that is ( m_{ij} = 1 ) if country ( i ) and ( j ) share a boundary point and ( m_{ij} = 0 ) otherwise.</td>
</tr>
<tr>
<td>2 - Triangulation</td>
<td>16.35</td>
<td>5.39</td>
<td>[4, 8]</td>
<td>Based on the Delaunay triangulation for a set of points in the plane, which is a triangulation such that no point is inside the circumcircle of any triangle (Weisstein, n.d.).</td>
</tr>
<tr>
<td>3 - Sphere of Influence</td>
<td>9.73</td>
<td>3.21</td>
<td>[1, 5]</td>
<td>Let ( d_i ) be the distance from country ( i ) to its nearest neighbor, and define ( C_i ) as the circle centered on ( i ) with radius ( d_i ); then we say that country ( j ) is in the sphere of influence of ( i ) if the circles ( C_i ) and ( C_j ) intercept more than once. This is a subgraph of the Delaunay triangulation.</td>
</tr>
<tr>
<td>4 - Gabriel</td>
<td>8.63</td>
<td>2.85</td>
<td>[1, 5]</td>
<td>The Gabriel graph of a set of points expresses one notion of proximity of those points; countries ( i ) and ( j ) are Gabriel neighbors if ( d(i,j) &lt; \min_k \sqrt{d(i,k)^2 + d(j,k)^2} ). It is also a subgraph of the Delaunay triangulation.</td>
</tr>
<tr>
<td>5 - Relative</td>
<td>6.06</td>
<td>2.00</td>
<td>[1, 3]</td>
<td>The relative neighborhood is defined as ( i ) and ( j ) are neighbors if ( d(i,j) &lt; \min_k \max {d(i,k), d(j,k)} ).</td>
</tr>
<tr>
<td>6 - Nearest Neighbor 1</td>
<td>4.77</td>
<td>1.58</td>
<td>[1, 3]</td>
<td>A country ( j ) is defined to be the “nearest neighbor” of ( i ) if the geodesic distance ( d(i,j) &lt; \min_k d(i,k) ), for ( k \neq {i, j} ).</td>
</tr>
<tr>
<td>7 - Nearest Neighbor 3</td>
<td>12.49</td>
<td>4.12</td>
<td>[3, 7]</td>
<td>A country ( j ) is defined to be the “3-nearest neighbor” of ( i ) if the geodesic distance between them is among the 3 smallest distances from ( i ) the any other country.</td>
</tr>
<tr>
<td>8 - Nearest Neighbor 5</td>
<td>20.02</td>
<td>6.61</td>
<td>[5, 9]</td>
<td>A country ( j ) is defined to be the “5-nearest neighbor” of ( i ) if the geodesic distance between them is among the 5 smallest distances from ( i ) the any other country.</td>
</tr>
<tr>
<td>9 - Geodesic Distance ( d_1 )</td>
<td>36.55</td>
<td>12.06</td>
<td>[1, 21]</td>
<td>Countries ( i ) and ( j ) are neighbors if the geodesic distance between them is smaller than ( d_1 = 6.2 ) thousand kilometers.</td>
</tr>
<tr>
<td>10 - Geodesic Distance ( d_2 )</td>
<td>44.63</td>
<td>14.73</td>
<td>[2, 25]</td>
<td>Countries ( i ) and ( j ) are neighbors if the geodesic distance between them is smaller than ( d_2 = 7.4 ) thousand kilometers.</td>
</tr>
<tr>
<td>11 - Geodesic Distance ( d_3 )</td>
<td>62.99</td>
<td>20.79</td>
<td>[5, 27]</td>
<td>Countries ( i ) and ( j ) are neighbors if the geodesic distance between them is smaller than ( d_3 = 9.2 ) thousand kilometers.</td>
</tr>
</tbody>
</table>

*Note that designs 4 – 8 generate an asymmetric neighborhood structure, where the fact that country \( i \) is neighbor of country \( j \) does not imply country \( j \) is neighbor of \( i \) (i.e., generates a directed graph); to avoid this awkward structure, the resulting spatial weight matrix is forced to be symmetric. The statistics of each design (columns 2–4) already consider a symmetric structure.
Table 4.5: General Spatial Model – Export Flows – Design 3

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>Std. Error</th>
<th>t value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\lambda$</td>
<td>-0.04</td>
<td>0.02</td>
<td>-1.74</td>
</tr>
<tr>
<td>(Intercept)</td>
<td>-25.37</td>
<td>2.40</td>
<td>-10.56</td>
</tr>
<tr>
<td>$Y_0$</td>
<td>0.95</td>
<td>0.09</td>
<td>10.58</td>
</tr>
<tr>
<td>$Y_{0C}$</td>
<td>0.97</td>
<td>0.11</td>
<td>8.98</td>
</tr>
<tr>
<td>$Y_d$</td>
<td>0.83</td>
<td>0.04</td>
<td>19.60</td>
</tr>
<tr>
<td>$Y_{dC}$</td>
<td>0.80</td>
<td>0.07</td>
<td>10.71</td>
</tr>
<tr>
<td>Dist</td>
<td>-0.80</td>
<td>0.11</td>
<td>-7.23</td>
</tr>
<tr>
<td>Cont</td>
<td>0.14</td>
<td>0.18</td>
<td>0.76</td>
</tr>
<tr>
<td>Lang</td>
<td>0.17</td>
<td>0.14</td>
<td>1.24</td>
</tr>
<tr>
<td>$R_t$</td>
<td>0.57</td>
<td>0.26</td>
<td>2.20</td>
</tr>
<tr>
<td>$C_n$</td>
<td>-0.16</td>
<td>0.52</td>
<td>-0.31</td>
</tr>
<tr>
<td>$C_c$</td>
<td>0.42</td>
<td>0.48</td>
<td>0.88</td>
</tr>
<tr>
<td>Col</td>
<td>0.35</td>
<td>0.20</td>
<td>1.74</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>0.57</td>
<td>0.03</td>
<td>19.45</td>
</tr>
<tr>
<td>$\rho$</td>
<td>0.87</td>
<td>0.07</td>
<td>12.11</td>
</tr>
</tbody>
</table>

Residual standard error: 0.90 on 1041 degrees of freedom
R-squared: 0.694
F-statistic: 304.6 on 3 and 1041 DF, p-value: < 2.2e-16

Table 4.6: General Spatial Model – Import Flows – Design 2

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>Std. Error</th>
<th>t value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\lambda$</td>
<td>-0.07</td>
<td>0.03</td>
<td>-2.24</td>
</tr>
<tr>
<td>(Intercept)</td>
<td>-26.05</td>
<td>3.14</td>
<td>-8.31</td>
</tr>
<tr>
<td>$Y_0$</td>
<td>0.82</td>
<td>0.03</td>
<td>24.48</td>
</tr>
<tr>
<td>$Y_{0C}$</td>
<td>0.88</td>
<td>0.06</td>
<td>14.90</td>
</tr>
<tr>
<td>$Y_d$</td>
<td>0.98</td>
<td>0.12</td>
<td>8.11</td>
</tr>
<tr>
<td>$Y_{dC}$</td>
<td>0.88</td>
<td>0.16</td>
<td>5.62</td>
</tr>
<tr>
<td>Dist</td>
<td>-0.68</td>
<td>0.12</td>
<td>-5.66</td>
</tr>
<tr>
<td>Cont</td>
<td>0.28</td>
<td>0.19</td>
<td>1.42</td>
</tr>
<tr>
<td>Lang</td>
<td>0.05</td>
<td>0.16</td>
<td>0.32</td>
</tr>
<tr>
<td>$R_t$</td>
<td>0.61</td>
<td>0.28</td>
<td>2.19</td>
</tr>
<tr>
<td>$C_n$</td>
<td>-0.47</td>
<td>0.46</td>
<td>-1.02</td>
</tr>
<tr>
<td>$C_c$</td>
<td>0.49</td>
<td>0.47</td>
<td>1.04</td>
</tr>
<tr>
<td>Col</td>
<td>0.32</td>
<td>0.19</td>
<td>1.71</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>0.62</td>
<td>0.02</td>
<td>28.03</td>
</tr>
<tr>
<td>$\rho$</td>
<td>0.92</td>
<td>0.06</td>
<td>14.25</td>
</tr>
</tbody>
</table>

Residual standard error: 0.92 on 1041 degrees of freedom
R-squared: 0.702
F-statistic: 232 on 3 and 1041 DF, p-value: < 2.2e-16
4.6 Conclusion

This chapter used the traditional gravity equations applied to international trade to illustrate the process to identify and estimate a spatial econometric model. The simple idea of gravity models is to express a flow (exports, imports, or total trade) from country $i$ to country $j$ as being directly proportional to their economic mass (e.g., income) and inversely proportional to the distance between them. Timmergen and Linnemann rationalize the model by arguing that the exports from $i$ to $j$ are greater, the larger is the economic size of $j$ due to demand factors, and the larger is the economic size of the exporting $i$ due to supply factors. The volume of trade should also decline with transportation costs and with difficulties of commercial contact, and geographical distance is used as a proxy for them. Both authors also include additional variables, such as dummies for common trade agreements, and dummies for countries that share a border.

In this chapter I investigated whether the traditional gravity equations could still have spatial dependence even after accounting for the distance as an important explanatory variable. It is clear from the Moran plot, from Moran I test, and from the estimation of the general spatial model that spatial dependence is indeed present in the residuals of the traditional gravity equation. This finding is robust to different specifications of the neighborhood matrix. However, in spite of the clear presence of spatial dependence in the residuals of the gravity equation, I find that the magnitude and statistical significance of the estimated parameters did not change substantially compared to the traditional gravity equation. This contrasts with Porojan (2001) who found a substantial change in the magnitude and significance of the estimated parameters when using a more restrict spatial model.
Chapter 5

Conclusion: Final Remarks

Models that consider the spatial dependence among sample points have only recently become present in mainstream economics and econometric models. A spatial stochastic process is nothing more than a standard stochastic process where a particular interpretation is given to the index set (set of spatial units), and the departure from the classical paradigm of iid is based on the dependence structure among the spatial units, with the degree of dependence proportional to the distance among observations. A broad definition of the terms space and distance allows spatial models to be used in several areas: economics (new economic geography, peer influence, neighbor effects, spillover effects, housing decisions, unemployment), sociology (externalities, diffusion and contagion in crime analysis), political science (individual voting behavior, international relations), among others.

It is interesting to highlight the fact that models with spatial dependence are suitable both in cases where there is a theory suggesting spatial interaction among observations, or in cases where there is no such theoretical foundation but nevertheless the observations can be regarded as having spatial dependence. This spatial dependence could arise, for instance, from omitted variables that depend on the location of the sample point in space, from misspecification of the true underlying data generating process, or from measurement problems. Ignoring spatial effects in these cases could lead, for instance, to biased variance estimators, inefficient estimators, bias and inconsistency in OLS estimators. Some authors therefore suggest as good practice to consider spatial dependence explicitly in any model to start with, and then to test the hypothesis of spatial dependence. This would likely make the proposed models robust to specification errors, omitted variables, and measurement errors.

This dissertation focuses on spatial stochastic processes on a lattice, where the index set is a countable set of regularly or irregularly spaced spatial units (Cliff & Ord-type of models). This class of models has a parallel with time series and thus borrows from the time series terminology of autoregressive and moving average processes. My contribution to the literature in the area consists of using Edgeworth and saddlepoint series to investigate small sample size and power properties of tests for detecting spatial dependence in spatial autoregressive (SAR) stochastic processes, and proposing a new class of spatial econometric models where the spatial dependence parameters that enter the mean structure are different from the ones in the covariance structure.

In the chapter on Small Sample Inference in Spatial Econometric Models I investigate size and power properties of tests to detect spatial dependence in spatial autoregressive (SAR) stochastic processes defined on a lattice, focusing mostly on the small sample properties of the tests and on the impact of different designs for the spatial weight matrix on test's properties. The importance of studying the consequences of different designs for the spatial weight matrix comes up because, in practical applications, the researcher seldom can count on a structure of spatial dependence that is tied to a well-defined theoretical model. Often, the spatial weight matrix is an arbitrary description of the spatial dependence that tries to capture the interaction
suggested by the underlying model. Therefore, the importance of investigating the effects of changes in the spatial weight matrix on the properties of the tests being used.

My contribution to the literature on spatial models arises from (i) using Edgeworth and saddlepoint to approximate the small sample distribution of test statistics to detect spatial dependence in SAR processes (nearly all the research in this area rely on Monte Carlo simulation), (ii) providing a performance comparison among Edgeworth and several saddlepoint approximations (Robinson, Lugannani-Rice, Barndorff-Nielsen / Jensen), and suggesting which characteristics of the spatial weight matrix make either preferable, (iii) using the optimal (in the Neyman-Pearson sense) statistic to construct the power envelope for testing the presence of spatial dependence in the SAR model when (a) the variance of the error term is known, and (b) the optimal invariant test when the variance is unknown (closed formulas for the critical values of these optimal test statistics are provided), (iv) allowing a simple way to correct for size when comparing power of the tests (to my knowledge most of the Monte Carlo experiments do not make this correction), and (v) allowing the spatial weight matrix to be a nonlinear function of the spatial dependence parameter, therefore expanding on existing literature which focus mostly on linear functions.

I find that Edgeworth and saddlepoint approximations are accurate for small samples ($n = 30$) for a variety of designs for the spatial weight matrix. Also, the actual size of the tests when using the critical values computed using the approximations are materially closer to the nominal size of the tests when compared to using asymptotic critical values. The optimal invariant test converges to the score (Lagrange multiplier) test when the alternative hypothesis converges to the null of no spatial dependence, and is therefore locally uniformly most powerful invariant (LUMPI). In this research, I found that, for the designs of the spatial weight matrix analyzed, the LUMPI test is virtually uniformly most powerful, in the sense that for practical purposes its power is very close to the power envelope. This probably reflects the small curvature (in the sense of Efron (1975)) of the model for the designs proposed. Since this nearly optimality might not hold for different designs, I suggest a practical procedure to build a test that, while not UMP, retain good power properties in a wider range for the spatial parameter when compared to the LUMPI test.

With the proposed test statistics, I analyze how the power envelope behaves for different sample sizes and different designs for the weight matrix. I find that power increases with sample size and with the spatial dependence parameter, which is in tandem with the literature. However, I call into question the consensus view that power decreases as the spatial weight matrix becomes more densely connected (see Anselin and Florax, 1995, Florax and de Graaff, 2004, and the references therein). The reduction of power observed in the literature reflects an error of measure where, despite looking at the same spatial dependence parameter, the hypothesis being compared are at different ‘statistical’ distance from the null. I use the average information distance (based on Fisher information) to measure the distance between the null and the alternative hypothesis and, after correcting for this, the power is larger for alternatives further a way from the null, as expected. For alternatives at the same distance from the null, the power of the tests are very similar, regardless of the degree of connectivity of the weight matrix.

In chapter Specification and Estimation of a New Class of Spatial ARMA Models I propose a more general stochastic process that nests into an encompassing structure models with local and global spatial effects. The traditional Cliff&Ord-type of spatial models is generally agreed to be fairly general and flexible. The main drawback, however, is that these models mix the effects of a single parameter into both the mean and the covariance structure of the spatial stochastic process (Cressie’s critique). In order to circumvent this problem, Anselin (2003) proposes a taxonomy classifying the spatial stochastic process according to whether the spatial effects (or externalities) are local or global, and whether the spatial effects occur in the covariates, in the error term, or in both. the problem with this suggestion, however, is that global and local specifications are non-nested. Anselin himself noted this drawback and speculated about the possibility of considering models, for instance, with local externalities in the error term and global externalities in the
covariates. The author also highlighted that none of these possibilities had received attention in the literature and, to my knowledge, this remained an unexplored area both in the theoretical as well as in the applied econometrics.

The model I introduce in this dissertation aims at addressing Cressie's critique and builds on Anselin's suggestion. My model is more general and several spatial stochastic process frequently used in both theoretical and applied research are nested into my proposed model. It is interesting to note, however, that one commonly used model (SARAR) seems to have a hidden interpretation: when looked under the optics of the model I proposed, the SARAR model implies a local spatial effect in the innovations that is linked to the global spatial effect assumed to the covariates in a very specific way! This is definitely not an interpretation that is evident from the outset. But the model I propose and the SARAR model are, in general, non-nested which makes them competing models whose validity and relevance needs to be tested empirically.

My contribution to this literature is thus to introduce a very general form for a spatial stochastic process and to present a simple procedure for estimating the parameters of the model as well as the asymptotic distribution of the proposed estimators. I propose a two step procedure to estimate the parameters of the model, where in the first step the conditional mean parameters are estimated by instrumental variables and the residual of this step is used in a GMM framework to estimate the conditional covariance parameters. The main attractive of this approach vis-a-vis, for instance, the estimation procedure proposed in Lin and Lee (2009) is the low dimensionality of the non-linear optimization step (GMM). This makes the procedure computationally simple even when the dimension of the parameter space is not small. I then proceed to show consistency and asymptotic normality of the proposed estimators. Contrary to what was obtained in Kelejian and Prucha (2009), the asymptotic distribution of the estimator of the mean does not depend on the instrumental variable estimator. This only happens because I used nonlinear least squares procedure to obtain an estimator for the residuals, while Kelejian and Prucha use the instrumental variable. Therefore, the covariance matrix of the GMM obtained by the authors could potentially become simpler following the idea that I developed in this paper.

In chapter 4, Example: Spatial Econometrics Applied to International Trade, I use the traditional gravity equations applied to international trade to illustrate the process to identify and estimate a spatial econometric model. I investigated whether the traditional gravity equations could still have spatial dependence even after accounting for the distance as an important explanatory variable. It is clear from the Moran plot, from Moran I test, and from the estimation of the general spatial model proposed in chapter 3 that spatial dependence is indeed present in the residuals of the traditional gravity equation. This finding is robust to different specifications of the neighborhood matrix. However, in spite of the clear presence of spatial dependence in the residuals of the gravity equation, I find that the magnitude and statistical significance of the estimated parameters did not change substantially compared to the traditional gravity equation. This contrasts with Porojan (2001) who found a substantial change in the magnitude and significance of the estimated parameters when using a more restrict spatial model.

The paragraphs above highlight my main contributions to the literature on spatial econometrics. But the motivation for this dissertation goes beyond those contributions – the dissertation allowed me to dig deep into several areas within econometrics and to learn a lot about the tools used in small sample analysis, asymptotic analysis, bootstrapping, hypothesis testing, among others, which helped to build the foundation necessary to be an economist and econometrician.

There are several promising areas for future research and extensions of the ideas developed in this dissertation. The tools developed in chapter 2 could be extended to include a more general spatial process than the SAR (spatial autoregressive) and to develop the asymptotic expansions when the spatial process is not observed directly, but is estimated as the residual from another stochastic process. Additionally, one could use the J-test for the (non-nested) model selection between the traditional SARMA process and the spatial model I developed in the chapter 3. This would provide some tools to allow a proper comparison of these competing models.
Appendix A

Appendices

A.1 Appendix to Chapter 2

A.1.1 Circulant Spatial Weight Matrices

This section establishes the important result on the order of magnitude of $\text{tr} C^s$, for $s = 2, 3, \ldots$, when the spatial weight matrix has a circulant form (see theorem A.1.3). We begin with some definitions and propositions taken from Lancaster (1969) and Gray (2006):

**A.1.1 Definition** (Circulant Matrices). An $n \times n$ matrix $R$, with elements in $\mathbb{C}$, is called *circulant* if and only if it has the following form:

$$
\begin{pmatrix}
  a_0 & a_1 & \ldots & a_{n-2} & a_{n-1} \\
  a_{n-1} & a_0 & \ldots & a_{n-3} & a_{n-2} \\
  \vdots & \vdots & \ddots & \vdots & \vdots \\
  a_1 & a_2 & \ldots & a_{n-1} & a_0
\end{pmatrix}
$$

(A.1.1)

A vector $a = (a_0, \ldots, a_{n-1}) \in \mathbb{C}^n$ then defines a circulant matrix.

The following proposition (see Gray, 2006) characterizes the eigenvalues and eigenvector of circulant matrices.

**A.1.2 Proposition** (Eigenvalues and eigenvectors of circulant matrices). *Every circulant matrix $R$ has eigenvectors $v_m = \frac{1}{\sqrt{n}}(1, e^{-2\pi im/n}, \ldots, e^{-2\pi im(n-1)/n})'$, for $m = 0, 1, \ldots, n-1$, and eigenvalues

$$
\lambda_m = \sum_{k=0}^{n-1} a_k e^{-2\pi imk/n}
$$

and can be expressed in the form $R = U\Lambda U^*$, where $U$ is a unitary matrix whose columns are the eigenvectors of $R$ and $\Lambda = \text{diag}(\lambda_m)$. Note also that $U$ is the same for all circulant matrices $R$.*

The circulant $n \times n$ matrices, with elements in $\mathbb{C}$, form a commutative sub-algebra of the algebra of $n \times n$ matrices, i.e., the subset of circulant matrices is closed under addition, multiplication and scalar multiplication, and for any two circulant matrices $R_1$ and $R_2$, we have $R_1R_2 = R_2R_1$.

Several spatial weight matrices found in the theoretical literature can be written as circulant matrices (see Das et al., 2003; Anselin and Florax, 1995, for example).

As the following theorem shows, if the spatial weight matrix $w(\rho)$ is a circulant matrix, then the trace of powers of the matrix $C$ is of order $O(n)$, and therefore the residual of the truncated Edgeworth series has the correct order of magnitude.
A.1.3 Theorem. If \( w(\rho) \) is a circulant matrix, then \( C(\rho_1, \rho) \equiv L(\rho)^{-1} A(\rho_1) L(\rho)^{-1} \) is also circulant. Moreover, if the sequence \( \{ c_k \}_{k=0}^\infty \) which defines the circulant matrix \( C \) is absolutely summable, then \( \text{tr} \, C(\rho_0, \rho_1)^s = O(n) \).

Proof. \( C \) is a circulant matrix since the circulant matrices form a sub-algebra. Then, its eigenvalues are given by proposition A.1.2:

\[
\lambda_m = \sum_{k=0}^{n-1} c_k z^{mk}, \text{ for } m = 0, \ldots, n - 1
\]

where \( c_k \) are the elements of \( C \) and \( z \equiv e^{-2\pi i/n} \).

\[
\lambda_m^s = \left( \sum_{k=0}^{n-1} c_k z^{mk} \right)^s
= \sum_{j_0, \ldots, j_{n-1}} \left( \begin{array}{c} s \\ j_0, \ldots, j_{n-1} \end{array} \right) c_{j_0}^{j_0} \cdots c_{j_{n-1}}^{j_{n-1}} z^{m(j_1+2j_2+\cdots+(n-1)j_{n-1})}
\]

where the summation is taken over all sequences of non-negative integers \( j_0, \ldots, j_{n-1} \) such that \( \sum_{i=0}^{n-1} j_i = s \). We can then write \( \text{tr} \, C^s \) as:

\[
\text{tr} \, C^s = \sum_{m=0}^{n-1} \lambda_m^s
= \sum_{m=0}^{n-1} \sum_{j_0, \ldots, j_{n-1}} \left( \begin{array}{c} s \\ j_0, \ldots, j_{n-1} \end{array} \right) c_{j_0}^{j_0} \cdots c_{j_{n-1}}^{j_{n-1}} z^{m(j_1+2j_2+\cdots+(n-1)j_{n-1})}
\]

\[
= \sum_{j_0, \ldots, j_{n-1}} \left( \begin{array}{c} s \\ j_0, \ldots, j_{n-1} \end{array} \right) c_{j_0}^{j_0} \cdots c_{j_{n-1}}^{j_{n-1}} \delta_{j_1+2j_2+\cdots+(n-1)j_{n-1} \mod n}
\]

where \( \delta \) is the Kronecker delta. Therefore

\[
|\text{tr} \, C^s| = n \sum_{j_0, \ldots, j_{n-1}} \left( \begin{array}{c} s \\ j_0, \ldots, j_{n-1} \end{array} \right) c_{j_0}^{j_0} \cdots c_{j_{n-1}}^{j_{n-1}} \delta_{j_1+2j_2+\cdots+(n-1)j_{n-1} \mod n}
\]

\[
\leq n \sum_{j_0, \ldots, j_{n-1}} \left( \begin{array}{c} s \\ j_0, \ldots, j_{n-1} \end{array} \right) |c_0|^{j_0} \cdots |c_{n-1}|^{j_{n-1}} \delta_{j_1+2j_2+\cdots+(n-1)j_{n-1} \mod n}
\]

\[
\leq n \sum_{j_0, \ldots, j_{n-1}} \left( \begin{array}{c} s \\ j_0, \ldots, j_{n-1} \end{array} \right) |c_0|^{j_0} \cdots |c_{n-1}|^{j_{n-1}}
\]

\[
= n (|c_0| + \cdots + |c_{n-1}|)^s
\]

Taking limits \( n \to \infty \) and noting that the sequence \( \{ c_k \}_{k=0}^\infty \) is absolutely summable, we then conclude that \( \text{tr} \, C(\rho_0, \rho_1)^s = O(n) \). \qed
A.1.2 Toeplitz Matrices

A.1.4 Definition (Toeplitz Matrices). An $n \times n$ matrix $T_n$, with elements in $\mathbb{C}$, is called Toeplitz if and only if it has the following form:

$$
\begin{pmatrix}
 t_0 & t_{-1} & t_{-2} & \ldots & t_{-(n-2)} & t_{-(n-1)} \\
 t_1 & t_0 & t_{-1} & \ldots & t_{-(n-3)} & t_{-(n-2)} \\
 t_2 & t_1 & t_0 & \ldots & t_{-(n-4)} & t_{-(n-3)} \\
 \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
 t_{n-2} & t_{n-3} & t_{n-4} & \ldots & t_0 & t_{-1} \\
 t_{n-1} & t_{n-2} & t_{n-3} & \ldots & t_{1} & t_0
\end{pmatrix}
$$

(A.1.2)

A vector $t = (t_{-(n-1)}, \ldots, t_0, \ldots, t_{n-1}) \in \mathbb{C}^{2n-1}$ then defines a Toeplitz matrix.

The sequence $\{t_k\}_{k=-\infty}^{+\infty}$ then defines a sequence $\{T_n\}$ of Toeplitz matrices. Assuming $\{t_k\}_{k=-\infty}^{+\infty}$ is absolutely summable, then the Fourier series $f(\lambda)$ defined by:

$$
f(\lambda) = \sum_{k=-\infty}^{+\infty} t_k e^{ik\lambda} = \lim_{n \to +\infty} \sum_{k=-n}^{n} t_k e^{ik\lambda}
$$

exists and is uniformly convergent for all $\lambda$. Additionally, the elements of the sequence $\{t_k\}_{k=-\infty}^{+\infty}$ can be obtained by the inversion of the Fourier transform:

$$
t_k = \frac{1}{2\pi} \int_{0}^{2\pi} f(\lambda) e^{-ik\lambda} d\lambda
$$

Therefore, a sequence of Toeplitz matrices can also be defined by the function $f(\lambda)$, i.e., $T_n(f)$.

As shown in Gray (2006), every sequence of Toeplitz matrices $T_n(f)$ can be approximated by at least one sequence of circulant matrices $C_n(f)$, provided the sequence $\{t_k\}$ is absolutely summable. It also shows that the average value of powers of the eigenvalues of both matrices converge and are equal:

$$
\lim_{n \to \infty} \frac{1}{n} \sum_{k=0}^{n-1} \lambda_k(C_n)^s = \lim_{n \to \infty} \frac{1}{n} \sum_{k=0}^{n-1} \lambda_k(T_n)^s = \frac{1}{2\pi} \int_{0}^{2\pi} f(\lambda)^s d\lambda
$$

Since all circulant matrices are also Toeplitz matrices, this also shows that $\text{tr} C^s = O(n)$ for general circulant matrices (see proposition A.1.2).

By the definitions, it is clear that sum of Toeplitz matrices are also Toeplitz, however, the same result does not generalize to products and inverses of Toeplitz matrices. Nevertheless, Gray (2006) shows that (i) the inverse of a Toeplitz matrix is asymptotically equivalent to the inverse of a circulant matrix and is asymptotically Toeplitz, (ii) the product of Toeplitz matrices are not necessarily Toeplitz, however, for every sequence of product of Toeplitz matrices, there is a sequence of Toeplitz matrices that is asymptotically equivalent to it and they commute asymptotically.

Based on these remarks, we can conclude that if the spatial weight matrix $w(\rho)$ is Toeplitz, then the matrix $C(\rho_1, \rho) = L(\rho)^{-1} A(\rho_1) L(\rho)^{-1}$ is asymptotically equivalent to a Toeplitz matrix and therefore the result $\text{tr} C(\rho_1, \rho)^s = O(n)$ also holds in this case.
A.2 Appendix to Chapter 3

A.2.1 Mathematical definitions and some useful lemmas

**A.2.1 Definition** (Uniformly bounded matrices). An \( n \times n \) matrix \( A_n \) is uniformly bounded if the row and column sums of its elements are uniformly bounded in absolute value, i.e., there exist positive constant \( c \) that do not depend on \( n \), such that

\[
\|A_n\|_\infty = \max_{i \in \{1, \ldots, n\}} \sum_{j=1}^{n} |a_{ij,n}| \leq c, \quad \text{and} \quad (A.2.1)
\]

\[
\|A_n\|_1 = \max_{j \in \{1, \ldots, n\}} \sum_{i=1}^{n} |a_{ij,n}| \leq c, \quad \forall n \in \mathbb{N} \quad (A.2.2)
\]

**A.2.2 Lemma** (Product of uniformly bounded and element wise bounded matrices). Let \( A_n \) be a \( n \times n \) uniformly bounded matrix and \( B_n \) an \( n \times k \) matrix whose elements are bounded. Then the elements of \( A_n B_n \) are bounded.

*Proof.* Since \( A_n \) is uniformly bounded, there exists \( c_1 > 0 \), independent of \( n \), such that

\[
\max_{i \in \{1, \ldots, n\}} \sum_{j=1}^{n} |a_{ij,n}| \leq c_1, \quad \forall n \in \mathbb{N}
\]

and there exists \( c_2 > 0 \) such that \( |b_{ij,n}| \leq c_2, \forall i = 1, \ldots, n, \forall n, j = 1, \ldots, k. \)

A typical element of \( A_n B_n \) is \( a_{ij,n} b_{ij,n} = \sum_{l=1}^{n} a_{il,n} b_{lj,n} \), therefore

\[
|ab_{ij,n}| = \left| \sum_{l=1}^{n} a_{il,n} b_{lj,n} \right| \leq \sum_{l=1}^{n} |a_{il,n}||b_{lj,n}|
\]

\[
\leq c_1 \sum_{l=1}^{n} |a_{il,n}|
\]

\[
\leq c_1 c_2
\]

\[ \square \]

**A.2.3 Lemma** (Product of uniformly bounded matrices). Let \( A_n \) and \( B_n \) be two \( n \times n \) matrices whose row and column sums are uniformly bounded in absolute value. Then row and column sums of \( A_n B_n \) are uniformly bounded in absolute value.

*Proof.* The typical element of the matrix \( A_n B_n \) is \( ab_{ij,n} = \sum_{k=1}^{n} a_{ik,n} b_{kj,n} \). Then:

\[
\sum_{j=1}^{n} |ab_{ij,n}| = \sum_{j=1}^{n} \left( \sum_{k=1}^{n} a_{ik,n} b_{kj,n} \right) \leq \sum_{j=1}^{n} \sum_{k=1}^{n} |a_{ik,n}||b_{kj,n}| \leq c^2, \quad \forall n \in \mathbb{N}
\]

and analogous for the row sums.

*Corollary* (Infinite product of uniformly bounded matrices). Let \( \{A_{k,n}\} \) be an infinite sequence of uniformly bounded matrices. Let \( c_k \) be the smallest value which bounds the sum of the absolute values of row and columns of \( A_{k,n} \), and \( c = \max \{c_1, \ldots, c_k, \ldots\} \), which is finite. If \( c \leq 1 \) then \( \prod_{k=1}^{\infty} A_{k,n} \) is uniformly bounded.

**A.2.4 Lemma** (Trace of uniformly bounded matrices). Let \( A_n \) be an \( n \times n \) matrix whose row sums are bounded in absolute value. Then \( \text{tr} A_n = O(n) \).
Proof. Let $\lambda_i(A_n)$, for $i = 1, \ldots, n$ be the eigenvalues of $A_n$ (real or complex).

$$|\text{tr} \ A_n| = \left| \sum_{i=1}^{n} \lambda_i(A_n) \right| \leq \sum_{i=1}^{n} |\lambda_i(A_n)| \leq nc$$

where the second inequality derives from the fact that the spectral radius of a matrix (the absolute value of its largest eigenvalue) is bounded by any induced matrix norm, in particular by the norms $\|A_n\|_1 = \max_{1 \leq i \leq n} \sum_{j=1}^{n} |a_{ij,n}|$, and $\|A_n\|_{\infty} = \max_{1 \leq i \leq n} \sum_{j=1}^{n} |a_{ij,n}|$, which are bounded since $A_n$ is uniformly bounded.

A.2.5 Lemma (Trace of product of uniformly bounded and elementwise bounded matrices). Let $A$ be a $n \times n$ uniformly bounded matrix, and $B$ a $n \times n$ matrix whose elements are bounded. Then $\text{tr}(AB) = O(n)$.

Proof. Since $A$ is uniformly bounded, there exists $c > 0$, independent of $n$, such that

$$\max_{i \in \{1, \ldots, n\}} \sum_{j=1}^{n} |a_{ij,n}| \leq c, \quad \text{and}$$

$$\max_{j \in \{1, \ldots, n\}} \sum_{i=1}^{n} |a_{ij,n}| \leq c, \quad \forall n \in \mathbb{N} \quad (A.2.3)$$

and there exists $M > 0$ such that $|b_{ij,n}| \leq M$, $\forall i, j = 1, \ldots, n$, $\forall n$. Therefore,

$$|\text{tr}(AB)| = \left| \sum_{i=1}^{n} \sum_{k=1}^{n} a_{ik,n} b_{ki,n} \right| \leq \sum_{i=1}^{n} \sum_{k=1}^{n} |a_{ik,n}||b_{ki,n}| \leq M \sum_{i=1}^{n} \sum_{k=1}^{n} |a_{ik,n}| \leq Mcn$$

$$\Rightarrow$$

$$\frac{|\text{tr}(AB)|}{n} \leq M < \infty \quad (A.2.4)$$

A.2.6 Lemma (Hadamard product of uniformly bounded matrices). Let $A, B$ be $n \times n$ matrices whose row and column sums are bounded in absolute value. Then $C = A \ast B$, where $\ast$ is the Hadamard (elementwise) product, is uniformly bounded.

Proof. Since $A, B$ are uniformly bounded, there exists $c > 0$, independently of $n$, such that

$$\max_{i \in \{1, \ldots, n\}} \sum_{j=1}^{n} |a_{ij,n}| \leq c, \quad \max_{i \in \{1, \ldots, n\}} \sum_{j=1}^{n} |b_{ij,n}| \leq c, \quad \text{and}$$

$$\max_{j \in \{1, \ldots, n\}} \sum_{i=1}^{n} |a_{ij,n}| \leq c, \quad \max_{j \in \{1, \ldots, n\}} \sum_{i=1}^{n} |b_{ij,n}| \leq c \quad \forall n \in \mathbb{N} \quad (A.2.5)$$

and it follows that $|a_{ij}| \leq c$ and $|b_{ij}| \leq c$. The typical element of $C$ is given by $c_{ij} = a_{ij}b_{ij}$, therefore the sum of absolute values of a typical row $i$ of $C$ is

$$\sum_{j=1}^{n} |c_{ij}| = \sum_{j=1}^{n} |a_{ij}||b_{ij}| \leq c \sum_{j=1}^{n} |a_{ij}| = c^2$$

and therefore the row sums of the elements of $C$ are uniformly bounded in absolute value. Similar reasoning shows the column sums of $C$ are uniformly bounded in absolute value, thus $A \ast B$ is uniformly bounded.
A.2.7 Lemma (Kronecker of uniformly bounded matrices). Let \( A, B \) be \( n \times n \) matrices whose row and column sums are bounded in absolute value. Then \( A \otimes B \) is uniformly bounded.

Proof. Since \( A, B \) are uniformly bounded, there exists \( c > 0 \), independently of \( n \), such that

\[
\max_{i \in \{1, \ldots, n\}} \sum_{j=1}^{n} |a_{ij,n}| \leq c, \quad \max_{i \in \{1, \ldots, n\}} \sum_{j=1}^{n} |b_{ij,n}| \leq c, \quad \text{and} \quad (A.2.7)
\]

\[
\max_{j \in \{1, \ldots, n\}} \sum_{i=1}^{n} |a_{ij,n}| \leq c, \quad \max_{j \in \{1, \ldots, n\}} \sum_{i=1}^{n} |b_{ij,n}| \leq c \quad \forall n \in \mathbb{N} \quad (A.2.8)
\]

Let \( C = A \otimes B \). The typical element of \( C \) is given by \( C_{\alpha\beta,n} = a_{ij,n}b_{kl,n} \), for \( \alpha \equiv n(i - 1) + k \), \( \beta \equiv n(j - 1) + l \). Therefore the sum of absolute values of a typical row of \( C \) is

\[
\sum_{\beta=1}^{n^2} |c_{\alpha\beta,n}| = \sum_{j=1}^{n} \sum_{i=1}^{n} |a_{ij,n}||b_{kl,n}| = \sum_{j=1}^{n} |a_{ij,n}| \sum_{l=1}^{n} |b_{kl,n}| \leq c^2 \quad \forall i, k = 1, \ldots, n
\]

and the row sums of the elements of \( C \) are uniformly bounded in absolute value. Similar reasoning shows the columns sums of \( C \) are uniformly bounded in absolute value, thus \( A \otimes B \) is uniformly bounded.

A.2.2 Moments of quadratic forms in nonnormal variables

This section collects some results on quadratic forms from Lin and Lee (2009), Kelejian and Prucha (2009), and corrects minor mistakes in the derivations contained in Lin and Lee (2009).

A.2.8 Lemma. Let \( A, B \) be \( n \times n \) matrices, and assume \( \varepsilon \) is an independent \( n \times 1 \) random vector with zero mean and covariance \( \Sigma \). Then:

(a) \( \mathbb{E} \varepsilon' A \varepsilon = \text{tr} \Sigma A \)

(b) \( \mathbb{E} \varepsilon' A \varepsilon \varepsilon' B \varepsilon = \sum_{i=1}^{n} a_{ii}b_{ii} \left( \mathbb{E} \varepsilon_i^2 \right) - 3\sigma_i^4 \) + \( \text{tr} \Sigma A \text{tr} \Sigma B + \text{tr} \Sigma A \Sigma B^* \)

(c) \( \text{Cov}(\varepsilon' A \varepsilon, \varepsilon' B \varepsilon) = \sum_{i=1}^{n} a_{ii}b_{ii} \left( \mathbb{E} \varepsilon_i^4 \right) - 3\sigma_i^4 \) + \( \text{tr} \Sigma A \Sigma B^* \)

(d) \( \mathbb{E}(\varepsilon' A \varepsilon + a' \varepsilon) = \text{tr} \Sigma A \)

(e) \( \mathbb{E} a' \varepsilon \varepsilon' A \varepsilon = \sum_{i=1}^{n} a_{ii} \mathbb{E} \varepsilon_i^2 \)

\[
\mathbb{E} \left[ (\varepsilon' A \varepsilon + a' \varepsilon) (\varepsilon' B \varepsilon + b' \varepsilon) \right] = \text{tr} \Sigma A \text{tr} \Sigma B + \text{tr} \Sigma A \Sigma B^* + a' \Sigma b + \sum_{i=1}^{n} a_{ii}b_{ii} \left( \mathbb{E} \varepsilon_i^4 \right) - 3\sigma_i^4 \]

(f)

\[
+ \sum_{i=1}^{n} (b_{ii}a_{ii} + a_{ii}b_{ii}) \mathbb{E} \varepsilon_i^3
\]

\( \text{Cov}(\varepsilon' A \varepsilon + a' \varepsilon, \varepsilon' B \varepsilon + b' \varepsilon) = \text{tr} \Sigma A \Sigma B^* + a' \Sigma b + \sum_{i=1}^{n} a_{ii}b_{ii} \left( \mathbb{E} \varepsilon_i^4 \right) - 3\sigma_i^4 \)

(g)

\[
+ \sum_{i=1}^{n} (b_{ii}a_{ii} + a_{ii}b_{ii}) \mathbb{E} \varepsilon_i^3
\]
Proof. Item (g):

\[ \text{Cov} (\varepsilon' A \varepsilon + a' \varepsilon', \varepsilon' B \varepsilon + b' \varepsilon) = E \left[ (\varepsilon' A \varepsilon + a' \varepsilon') (\varepsilon' B \varepsilon + b' \varepsilon) \right] - E (\varepsilon' A \varepsilon + a' \varepsilon) E (\varepsilon' B \varepsilon + b' \varepsilon) \]

which, by items (f) and (d) yield:

\[ = \text{tr} \Sigma A \Sigma B^* + a' \Sigma b + \sum_{i=1}^{n} a_{ii} b_{ii} \left( E(\varepsilon_i^4) - 3 \sigma_i^4 \right) + \sum_{i=1}^{n} (b_{ii} a_{ii} + a_{ii} b_{ii}) E(\varepsilon_i^2) \]

Item (f):

\[ E \left[ (\varepsilon' A \varepsilon + a' \varepsilon) (\varepsilon' B \varepsilon + b' \varepsilon) \right] = E (\varepsilon' A \varepsilon' B \varepsilon) + E (\varepsilon' A \varepsilon' b \varepsilon) + E (a' \varepsilon b' \varepsilon) \]

by using items (b) and (e):

\[ = \sum_{i=1}^{n} a_{ii} b_{ii} \left( E(\varepsilon_i^4) - 3 \sigma_i^4 \right) + \text{tr} \Sigma A \text{tr} \Sigma B + \text{tr} \Sigma A \Sigma B^* \\
+ \sum_{i=1}^{n} b_{ii} a_{ii} E(\varepsilon_i^2) + \sum_{i=1}^{n} a_{ii} b_{ii} E(\varepsilon_i^2) + \sum_{i=1}^{n} a_{ii} b_{ii} E(\varepsilon_i^2) \]

\[ = \sum_{i=1}^{n} a_{ii} b_{ii} \left( E(\varepsilon_i^4) - 3 \sigma_i^4 \right) + \text{tr} \Sigma A \text{tr} \Sigma B + \text{tr} \Sigma A \Sigma B^* + \sum_{i=1}^{n} (b_{ii} a_{ii} + a_{ii} b_{ii}) E(\varepsilon_i^2) + a' \Sigma b \]

Item (e):

\[ E (a' \varepsilon \varepsilon' A \varepsilon) = \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} a_{i} a_{jk} E(\varepsilon_i \varepsilon_j \varepsilon_k) = \sum_{i=1}^{n} a_{ii} E(\varepsilon_i^2) \]

Item (a):

\[ E (\varepsilon' A \varepsilon) = E \left( \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} \varepsilon_i \varepsilon_j \right) \]

\[ = \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{j \neq i} a_{ij} E(\varepsilon_i \varepsilon_j) + \sum_{i=1}^{n} a_{ii} E(\varepsilon_i^2) \]

\[ = \sum_{i=1}^{n} a_{ii} \sigma_i^2 = \text{tr} \Sigma A \]
Item (b):

\[
E(\epsilon' A\epsilon' \epsilon B \epsilon) = \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} \sum_{l=1}^{n} a_{ij}b_{kl} E(\epsilon_i \epsilon_j \epsilon_k \epsilon_l)
\]

\[
= \sum_{i=1}^{n} a_{ii} E(\epsilon_i^4) + \sum_{i=1}^{n} \sum_{j=1}^{n} (a_{ii}b_{jj} + a_{ij}b_{ij} + a_{ij}b_{ji}) E(\epsilon_i^2) E(\epsilon_j^2)
\]

\[
= \sum_{i=1}^{n} a_{ii} E(\epsilon_i^4) - 3\sigma_i^4 + \sum_{i=1}^{n} \sum_{j=1}^{n} (a_{ii}b_{jj} + a_{ij}b_{ij} + a_{ij}b_{ji}) E(\epsilon_i^2) E(\epsilon_j^2)
\]

\[
= \sum_{i=1}^{n} a_{ii} E(\epsilon_i^4) - 3\sigma_i^4 + \sum_{i=1}^{n} \sum_{j=1}^{n} (\sigma_i^2 a_{ii} \sigma_j^2 b_{jj} + \sigma_i^2 a_{ij} \sigma_j^2 b_{ij} + \sigma_i^2 a_{ij} \sigma_j^2 b_{ji})
\]

\[
= \sum_{i=1}^{n} a_{ii} E(\epsilon_i^4) - 3\sigma_i^4 + \sum_{i=1}^{n} \sum_{j=1}^{n} ((\Sigma A)_{ii} (\Sigma B)_{jj} + (\Sigma A)_{ij} (\Sigma B)_{ij} + (\Sigma A)_{ij} (\Sigma B)_{ji})
\]

\[
= \sum_{i=1}^{n} a_{ii} E(\epsilon_i^4) - 3\sigma_i^4 + tr \Sigma A tr \Sigma B + tr \Sigma A \Sigma B + tr \Sigma A \Sigma B^t
\]

\[
= \sum_{i=1}^{n} a_{ii} E(\epsilon_i^4) - 3\sigma_i^4 + tr \Sigma A tr \Sigma B + \underbrace{tr \Sigma A \Sigma B^t}_{(*)}
\]

Note that the expression (*) in Lin and Lee (2009) is written as \(tr (\Sigma AB^t \Sigma + \Sigma A \Sigma B)\), which is not equivalent.

Item (c): Follows from (a) and (b). \(\Box\)

The following is an extension of the CLT for linear quadratic forms (Kelejian and Prucha, 2001) to the multivariate case. It was proved in Kelejian and Prucha (2009). For simplicity, I will state the theorem for the bivariate case, but the generalization is straightforward.

**A.2.9 Theorem** (Multivariate central limit theorem for linear quadratic forms). Let \(m_1 = \frac{1}{\sqrt{n}} \epsilon' A_1 \epsilon + \frac{1}{\sqrt{n}} a_1 \epsilon, m_2 = \frac{1}{\sqrt{n}} \epsilon' A_2 \epsilon + \frac{1}{\sqrt{n}} a_2 \epsilon\), and assume (i) \(E(\epsilon_i) = 0, \forall i = 1, \ldots, n, \forall n\), and \(\epsilon_1, \ldots, \epsilon_n\) are independent for all \(n\), (ii) the \(n \times n\) matrices \(A_i, i = 1, 2\) are uniformly bounded, (iii) the \(n \times 1\) vectors \(a_i, i = 1, 2\) satisfy, for some \(\delta_i > 0, \frac{1}{n} \sum_{j=1}^{n} |a_{ij}|^{2+\delta_i} < \infty, \forall n, i = 1, 2\), (iv) for some \(\delta_2 > 0, E|\epsilon|^4 < \infty, \forall n, \forall n\), and (v) the variance of the linear quadratic forms \(m_i, i = 1, 2\) are bounded away from zero. Then

\[
\Xi_m^{-\frac{1}{2}} (m - \mu_m) \xrightarrow{d} N(0, I_2)
\]

where

\[
\mu_m = \left[ \frac{1}{\sqrt{n}} tr (\Sigma A_1) \right] \left[ \frac{1}{\sqrt{n}} tr (\Sigma A_2) \right]
\]

\[
\sigma_{m_1 m_2} = \frac{1}{n} \left\{ tr \Sigma A_1 \Sigma A_2^t + a_2^t \Sigma a_2 + \sum_{i=1}^{n} (A_1)_{ii} (A_2)_{ii} (E(\epsilon_i^4) - 3\sigma_i^4) \right. \\
+ \left. \sum_{i=1}^{n} ((a_2)_{i} (A_1)_{ii} + (a_1)_{i} (A_2)_{ii}) E(\epsilon_i^4) \right\}
\]

\[
\Xi_m = \begin{bmatrix}
\sigma_{m_1 m_1} & \sigma_{m_1 m_2} \\
\sigma_{m_2 m_1} & \sigma_{m_2 m_2}
\end{bmatrix}
\]

\[\Sigma \equiv E(\epsilon \epsilon')\]
Proof. See Kelejian and Prucha (2009, theorem A.1, p.11).

A.2.3 Miscellanea

A.2.10 Definition (Uniform convergence (Bartle, 1967)). A sequence \( \{ f_n \} \) of functions on \( D \subset \mathbb{R}^p \) to \( \mathbb{R}^q \) converges uniformly on a subset \( D_0 \) of \( D \) to a function \( f \) in case for each \( \epsilon > 0 \) there is a natural number \( N_\epsilon \) (depending on \( \epsilon \) but not on \( x \in D_0 \)) such that if \( n > N_\epsilon \), and \( x \in D_0 \), then

\[ |f_n(x) - f(x)| < \epsilon \]

In other words,

\[ \lim_{n \to \infty} \sup_{x \in D} |f_n(x) - f(x)| = 0 \]

A.2.11 Definition (Equicontinuity (Bartle, 1967)). A sequence \( \{ f_n \} \) of functions on a compact \( D \subset \mathbb{R}^p \) to \( \mathbb{R}^q \) is equicontinuous on \( D \) if, for each \( \epsilon > 0 \) there is a \( \delta_\epsilon > 0 \) such that \( \| f_n(x) - f_n(y) \| < \epsilon \) for all \( \| x - y \| < \delta_\epsilon \), and all \( n \).

A.2.12 Theorem (Arzelà-Ascoli (Bartle, 1967)). Let \( D \subset \mathbb{R}^p \) be compact, and let \( \{ f_n \} \) be a sequence of continuous functions on \( D \) to \( \mathbb{R}^q \). The following properties are equivalent:

(a) The sequence \( \{ f_n \} \) is bounded and equicontinuous on \( D \).

(b) The sequence \( \{ f_n \} \) has a subsequence that is uniformly convergent on \( D \).

A.2.13 Theorem (Interchange of limit and derivative (Bartle, 1967)). Let \( \{ f_n \} \) be a sequence of functions, \( f_n : J \to \mathbb{R}, \ J \subset \mathbb{R} \) is an interval. Suppose that there is a point \( x_0 \in J \) at which the sequence \( \{ f_n(x_0) \} \) converges, that the derivatives \( f'_n \) exist on \( J \), and that the sequence \( \{ f'_n \} \) converges uniformly on \( J \) to a function \( g \). Then the sequence \( \{ f_n \} \) converges uniformly on \( J \) to a function \( f \) which has a derivative at every point of \( J \) and \( f' = g \).
A.3 Appendix to Chapter 4

A.3.1 Neighborhood Structure Maps

Figure A.1: Design 1 – Contiguity

Figure A.2: Design 2 – Triangulation
Figure A.3: Design 3 – Sphere of Influence (SOI)

Figure A.4: Design 4 – Gabriel

Figure A.5: Design 5 – Relative
Figure A.6: Design 6 – Nearest Neighbor 1 (NN1)

Figure A.7: Design 7 – Nearest Neighbor 3 (NN3)

Figure A.8: Design 8 – Nearest Neighbor 5 (NN5)
Figure A.9: Design 9 – Geodesic Distance ($d_1 = 6.2$ thousand Km)

Figure A.10: Design 10 – Geodesic Distance ($d_2 = 7.4$ thousand Km)

Figure A.11: Design 11 – Geodesic Distance ($d_3 = 9.2$ thousand Km)
A.3.2 Moran Plot

(a) Destination-based neighbor structure  (b) Origin-based neighbor structure

(c) Destination-based neighborhood (normalized)  (d) Origin-based neighborhood (normalized)

Figure A.12: Moran plot for the residuals of the exports gravity equation – Design 1
(a) Destination-based neighbor structure

(b) Origin-based neighbor structure

(c) Destination-based neighborhood (normalized)

(d) Origin-based neighborhood (normalized)

Figure A.13: Moran plot for the residuals of the exports gravity equation – Design 2
Figure A.14: Moran plot for the residuals of the exports gravity equation – Design 3
(a) Destination-based neighbor structure

(b) Origin-based neighbor structure

(c) Destination-based neighborhood (normalized)

(d) Origin-based neighborhood (normalized)

Figure A.15: Moran plot for the residuals of the exports gravity equation – Design 4
(a) Destination-based neighbor structure  
(b) Origin-based neighbor structure  
(c) Destination-based neighborhood (normalized)  
(d) Origin-based neighborhood (normalized)  

Figure A.16: Moran plot for the residuals of the exports gravity equation – Design 5
(a) Destination-based neighbor structure

(b) Origin-based neighbor structure

(c) Destination-based neighborhood (normalized)

(d) Origin-based neighborhood (normalized)

Figure A.17: Moran plot for the residuals of the exports gravity equation – Design 6
(a) Destination-based neighbor structure  
(b) Origin-based neighbor structure

(c) Destination-based neighborhood (normalized)  
(d) Origin-based neighborhood (normalized)

Figure A.18: Moran plot for the residuals of the exports gravity equation – Design 7
(a) Destination-based neighbor structure

(b) Origin-based neighbor structure

(c) Destination-based neighborhood (normalized)

(d) Origin-based neighborhood (normalized)

Figure A.19: Moran plot for the residuals of the exports gravity equation – Design 8
(a) Destination-based neighbor structure

(b) Origin-based neighbor structure

(c) Destination-based neighborhood (normalized)

(d) Origin-based neighborhood (normalized)

Figure A.20: Moran plot for the residuals of the exports gravity equation – Design 9
(a) Destination-based neighbor structure  
(b) Origin-based neighbor structure  
(c) Destination-based neighborhood (normalized)  
(d) Origin-based neighborhood (normalized)

Figure A.21: Moran plot for the residuals of the exports gravity equation – Design 10
(a) Destination-based neighbor structure
(b) Origin-based neighbor structure
(c) Destination-based neighborhood (normalized)
(d) Origin-based neighborhood (normalized)

Figure A.22: Moran plot for the residuals of the exports gravity equation – Design 11
(a) Destination-based neighbor structure  
(b) Origin-based neighbor structure  
(c) Destination-based neighborhood (normalized)  
(d) Origin-based neighborhood (normalized) 

Figure A.23: Moran plot for the residuals of the imports gravity equation – Design 1
(a) Destination-based neighbor structure
(b) Origin-based neighbor structure

(c) Destination-based neighborhood (normalized)
(d) Origin-based neighborhood (normalized)

Figure A.24: Moran plot for the residuals of the imports gravity equation – Design 2
(a) Destination-based neighbor structure  
(b) Origin-based neighbor structure  
(c) Destination-based neighborhood (normalized)  
(d) Origin-based neighborhood (normalized)  

Figure A.25: Moran plot for the residuals of the imports gravity equation – Design 3
(a) Destination-based neighbor structure
(b) Origin-based neighbor structure
(c) Destination-based neighborhood (normalized)
(d) Origin-based neighborhood (normalized)

Figure A.26: Moran plot for the residuals of the imports gravity equation – Design 4
(a) Destination-based neighbor structure  
(b) Origin-based neighbor structure  

(c) Destination-based neighborhood (normalized)  
(d) Origin-based neighborhood (normalized)  

Figure A.27: Moran plot for the residuals of the imports gravity equation – Design 5
(a) Destination-based neighbor structure  
(b) Origin-based neighbor structure

(c) Destination-based neighborhood (normalized)  
(d) Origin-based neighborhood (normalized)

Figure A.28: Moran plot for the residuals of the imports gravity equation – Design 6
(a) Destination-based neighbor structure  
(b) Origin-based neighbor structure

(c) Destination-based neighborhood (normalized)  
(d) Origin-based neighborhood (normalized)

Figure A.29: Moran plot for the residuals of the imports gravity equation – Design 7
Figure A.30: Moran plot for the residuals of the imports gravity equation – Design 8
(a) Destination-based neighbor structure

(b) Origin-based neighbor structure

(c) Destination-based neighborhood (normalized)

(d) Origin-based neighborhood (normalized)

Figure A.31: Moran plot for the residuals of the imports gravity equation – Design 9
Figure A.32: Moran plot for the residuals of the imports gravity equation – Design 10
(a) Destination-based neighbor structure

(b) Origin-based neighbor structure

(c) Destination-based neighborhood (normalized)

(d) Origin-based neighborhood (normalized)

Figure A.33: Moran plot for the residuals of the imports gravity equation – Design 11
A.4  R codes

A.4.1  Codes used in Chapter 2

This section includes R codes that were used to generate the statistics, tables, figures, and simulations used in chapter 2.

Listing 1: chart01.R

```r
# This program calculates the power envelope for testing
# H0: rho_0 = 0 vs. H1: rho_1 <> 0
# in the SAR model y = w(rho)y = u
rm(list=ls()) # Apaga todas variaveis da memoria
source("/medias/EGE/Docs/programas/all_functions.R")
X11.options(width=5, height=5, type="pdfcairo")
ps.options(horizontal=FALSE, onefile=FALSE, height=4.5,
       width=6, paper="special", pointsize=10)

# Begin program
epsilon <- .Machine$double.eps^0.5 # defines an arbitrarily small value
# nmax <- 4 # max cumulant to be calculated (4 for 0(1/n), 5 for 0(1/n^(-3/2))
# alpha <- 0.05 # size alpha test (limits the type I error)

# Calculate the power envelope for m=2 and different n
m <- 2 # with m-ahead and m-behind neighbors
J <- 9
n.range <- c(36,100,200)
lens.rho.range<150 # HAS to be EVEN
rho.range<array(NA, dim=c(4, length(n.range), len.rho.range))
M<-vector("list",4)

# matrix to store the power calculated on each rho in rho.range
fischer.info<array(NA, dim=c(4, length(n.range), len.rho.range))
#fsher.at.zero<array(NA, dim=c(4, length(n.range), 1))

for (i in 1:length(n.range)) {
  M[[1]]<-genDesign1(n.range[i],m)
  M[[2]]<-genDesign2(n.range[i],m)
  M[[3]]<-genDesign3(n.range[i],m)
  M[[4]]<-genDesign4(n.range[i],m,J)
  rr<-1/eigenRange(M[[1]]))
  rho.range[i,1,]<-c(seq(from=rh[1]+sqrt(epsilon), to=-sqrt(epsilon), len=len.rho.range/2)
   ,0,
   seq(from=rr[1]+sqrt(epsilon), to=rr[2]-sqrt(epsilon), len=len.rho.range/2-1))
  fischer.info[i,1,]<-supply(rho.range[i,1],FisherInfoRho,M[[1]])
  #fsher.at.zero[i,1]<FisherInfoRho(0,M[[1]])
  rr<-1/eigenRange(M[[2]])
  rho.range[i,2,]<-c(seq(from=rh[1]+sqrt(epsilon), to=-sqrt(epsilon), len=len.rho.range/2)
   ,0,
   seq(from=rr[1]+sqrt(epsilon), to=rr[2]-sqrt(epsilon), len=len.rho.range/2-1))
  fischer.info[i,2,]<-supply(rho.range[i,2],FisherInfoRho,M[[2]])
  #fsher.at.zero[i,2]<FisherInfoRho(0,M[[2]])
  rr<-1/eigenRange(M[[3]])
  rho.range[i,3,]<-c(seq(from=rh[1]+sqrt(epsilon), to=-sqrt(epsilon), len=len.rho.range/2)
   ,0,
   seq(from=rr[1]+sqrt(epsilon), to=rr[2]-sqrt(epsilon), len=len.rho.range/2-1))
  fischer.info[i,3,]<-supply(rho.range[i,3],FisherInfoRho,M[[3]])
  #fsher.at.zero[i,3]<FisherInfoRho(0,M[[3]])
  rr<-1/eigenRange(M[[4]])
  rho.range[i,4,]<-c(seq(from=rh[1]+sqrt(epsilon), to=-sqrt(epsilon), len=len.rho.range/2)
   ,0,
   seq(from=rr[1]+sqrt(epsilon), to=rr[2]-sqrt(epsilon), len=len.rho.range/2-1))
  fischer.info[i,4,]<-supply(rho.range[i,4],FisherInfoRho,M[[4]])
  #fsher.at.zero[i,4]<FisherInfoRho(0,M[[4]])
  power[i,1,]<powerE4(rho.range[i,1],alpha,M[[1]],1,TRUE)
  power[i,2,]<powerE4(rho.range[i,2],alpha,M[[2]],floor(sqrt(n.range[i])),TRUE)
  power[i,3,]<powerE4(rho.range[i,3],alpha,M[[3]],1,TRUE)
  power[i,4,]<powerE4(rho.range[i,4],alpha,M[[4]],1,TRUE)
}
```

61}

# Functions to calculate the square root of Fisher info (fi)
# and the information distance (id), i.e., the integral from the null
# to the alternative.
# obs: Note that design 2 has a different normalizing factor to obtain the average info

f11 <- approxfun(rho.range[1,1], sqrt(fisher.info[1,1]/n.range[1]))
id11 <- function(rho) integrate(f11, 0, rho)$value
id11f <- function(rho) sapply(rho, id11)

f12 <- approxfun(rho.range[1,2], sqrt(fisher.info[1,2]/n.range[2]))
id12 <- function(rho) integrate(f12, 0, rho)$value
id12f <- function(rho) sapply(rho, id12)

f13 <- approxfun(rho.range[1,3], sqrt(fisher.info[1,3]/n.range[3]))
id13 <- function(rho) integrate(f13, 0, rho)$value
id13f <- function(rho) sapply(rho, id13)

f21 <- approxfun(rho.range[2,1], sqrt(fisher.info[2,1]/n.range[1]/floor(sqrt(n.range[1]))))
id21 <- function(rho) integrate(f21, 0, rho)$value
id21f <- function(rho) sapply(rho, id21)

f22 <- approxfun(rho.range[2,2], sqrt(fisher.info[2,2]/n.range[2]/floor(sqrt(n.range[2]))))
id22 <- function(rho) integrate(f22, 0, rho)$value
id22f <- function(rho) sapply(rho, id22)

f23 <- approxfun(rho.range[2,3], sqrt(fisher.info[2,3]/n.range[3]/floor(sqrt(n.range[3]))))
id23 <- function(rho) integrate(f23, 0, rho)$value
id23f <- function(rho) sapply(rho, id23)

f31 <- approxfun(rho.range[3,1], sqrt(fisher.info[3,1]/n.range[1]))
id31 <- function(rho) integrate(f31, 0, rho)$value
id31f <- function(rho) sapply(rho, id31)

f32 <- approxfun(rho.range[3,2], sqrt(fisher.info[3,2]/n.range[2]))
id32 <- function(rho) integrate(f32, 0, rho)$value
id32f <- function(rho) sapply(rho, id32)

f33 <- approxfun(rho.range[3,3], sqrt(fisher.info[3,3]/n.range[3]))
id33 <- function(rho) integrate(f33, 0, rho)$value
id33f <- function(rho) sapply(rho, id33)

f41 <- approxfun(rho.range[4,1], sqrt(fisher.info[4,1]/n.range[1]))
id41 <- function(rho) integrate(f41, 0, rho)$value
id41f <- function(rho) sapply(rho, id41)

f42 <- approxfun(rho.range[4,2], sqrt(fisher.info[4,2]/n.range[2]))
id42 <- function(rho) integrate(f42, 0, rho)$value
id42f <- function(rho) sapply(rho, id42)

f43 <- approxfun(rho.range[4,3], sqrt(fisher.info[4,3]/n.range[3]))
id43 <- function(rho) integrate(f43, 0, rho)$value
id43f <- function(rho) sapply(rho, id43)

for (i in 1:4) for (j in 1:length(n.range))
assign(paste("pwe", i, j, sep=""), approxfun(rho.range[i,j], power[i,j]))

for (i in 1:4) for (j in 1:length(n.range)) {
  x.id <- eval(call(paste("id", i, j, sep=""), rho.range[i,j]))
  assign(paste("pwe_id", i, j, sep=""), approxfun(x.id, power[i,j]))
}

# plots all the power envelopes
# Design 1
postscript("/media/EPGE/Docs/figures/chart01a.eps")
xlim <- range(rho.range[1,])
curve(pwe11, xlim[1], xlim[2], ylim=c(0,1), type="l", lty=1, xlab=expression(rho[1]), ylab= expression(beta(rho[1])))
curve(pwe12, xlim[1], xlim[2], lty=2, add=TRUE)
## Graficos alternativos

```r
# n1
plot(rho.range[1,1], power[1,1], ylim=c(0,1), type="l", lty=1,
     xlab=expression(rho[1]), ylab= expression(beta(rho[1])))
for (i in 2:4) {
    lines(rho.range[1,i], power[1,i], lty=i)
}
legend.text <- c(paste("Design", 1:4))
legend("bottomleft", legend=legend.text, lty=1:4,
       bty="n")

curve(pwe_id11, -1,1, ylim=c(0,1), type="l", lty=1,
      xlab="Information distance (id)", ylab= expression(beta(id)))
curve(pwe_id21, -1,1, lty=2, add=TRUE)
curve(pwe_id31, -1,1, lty=3, add=TRUE)
curve(pwe_id41, -1,1, lty=4, add=TRUE)
legend.text <- c(paste("Design", 1:4))
legend("bottomleft", legend=legend.text, lty=1:4,
       bty="n")

# n2
plot(rho.range[1,2], power[1,2], ylim=c(0,1), type="l", lty=1,
     xlab=expression(rho[1]), ylab= expression(beta(rho[1])))
for (i in 2:4) {
    lines(rho.range[1,i], power[1,i], lty=i)
}
legend.text <- c(paste("Design", 1:4))
legend("bottomleft", legend=legend.text, lty=1:4,
       bty="n")

# n3
plot(rho.range[1,3], power[1,3], ylim=c(0,1), type="l", lty=1,
     xlab=expression(rho[1]), ylab= expression(beta(rho[1])))
for (i in 2:4) {
    lines(rho.range[1,i], power[1,i], lty=i)
}
legend.text <- c(paste("Design", 1:4))
legend("bottomleft", legend=legend.text, lty=1:4,
       bty="n")

# CURIOSIDADES

curve(pwe_id23, -1,1, ylim=c(0,1), type="l", lty=1,
      xlab="Information distance (id)", ylab= expression(beta(id)))
curve(pwe_id33, -1,1, lty=2, add=TRUE)
curve(pwe_id43, -1,1, lty=3, add=TRUE)
legend.text <- c(paste("Design", 1:4))
legend("bottomleft", legend=legend.text, lty=1:4,
       bty="n")
unroot(function(c) pwe_id41(c)-1,c(0,rho.range[4,1,len.rho.range]))
```

Listing 2: chart04modified.R
# Power envelope for circular world design
# with m=2 for different values of n
# using saddlepoint approx
# Modified to introduct all the 4 different designs

rm(list=ls())
source("/media/EPGE/Docs/programas/all_functions.R")
X11.options(width=6, height=6, type="ncairo")
ps.options(horizontal=FALSE, onefile=FALSE, height=4.5,
           width=6, paper="special", fontsize=10)
epsilon <- .Machine$double.eps^0.25
m <- 2 # with m-ahead and m-behind neighbors
<-.9
alpha <- .0.05
n.range <- c(36, 100, 200)
l rnd.range<-150 # HAS to be EVEN
rho.range <- array(NA, dim=c(4, length(n.range), len.rho.range))
M <- vector("list", 4)
power <- array(NA, dim=c(4, length(n.range), len.rho.range)) # matrix to
#store the power calculated on each rho in rho.range
fisher.info <- array(NA, dim=c(4, length(n.range), len.rho.range))

for (i in 1:length(n.range)) {
  M[[1]] <- genDesign1(n.range[i], m)
  M[[2]] <- genDesign2(n.range[i], m)
  M[[3]] <- genDesign3(n.range[i], m)
  M[[4]] <- genDesign4(n.range[i], m)

  rr <- -1/eigenRange(M[[1]])
  rho.range[1, i] <- c(seq(from = rr, to = -sqrt(epsilon), len = len.rho.range/2), 0,
                      seq(from = sqrt(epsilon), to = -rr - sqrt(epsilon), len = len.rho.range/2 - 1))
  fisher.info[1, i] <- apply(rho.range[1, i], FisherInfoRho, M[[1]])

  rr <- -1/eigenRange(M[[2]])
  rho.range[2, i] <- c(seq(from = rr, to = -sqrt(epsilon), len = len.rho.range/2), 0,
                      seq(from = sqrt(epsilon), to = -rr - sqrt(epsilon), len = len.rho.range/2 - 1))
  fisher.info[2, i] <- apply(rho.range[2, i], FisherInfoRho, M[[2]])

  rr <- -1/eigenRange(M[[3]])
  rho.range[3, i] <- c(seq(from = rr, to = -sqrt(epsilon), len = len.rho.range/2), 0,
                      seq(from = sqrt(epsilon), to = -rr - sqrt(epsilon), len = len.rho.range/2 - 1))
  fisher.info[3, i] <- apply(rho.range[3, i], FisherInfoRho, M[[3]])

  rr <- -1/eigenRange(M[[4]])
  rho.range[4, i] <- c(seq(from = rr, to = -sqrt(epsilon), len = len.rho.range/2), 0,
                      seq(from = sqrt(epsilon), to = -rr - sqrt(epsilon), len = len.rho.range/2 - 1))
  fisher.info[4, i] <- apply(rho.range[4, i], FisherInfoRho, M[[4]])

  power[1, i] <- powerLR(rho.range[1, i], alpha, M[[1]], TRUE, FALSE) # NORMALIZE - FALSE !!
  power[2, i] <- powerLR(rho.range[2, i], alpha, M[[2]], floor(sqrt(n.range[i])), TRUE, FALSE)
  power[3, i] <- powerLR(rho.range[3, i], alpha, M[[3]], TRUE, FALSE)
  power[4, i] <- powerLR(rho.range[4, i], alpha, M[[4]], TRUE, FALSE)
}

# save(power, rho.range, fisher.info, file="/media/EPGE/Docs/dados/saddle1.Rdata")
# save(power, rho.range, fisher.info, file="/media/EPGE/Docs/dados/saddle2.Rdata")
# save(list=ls(all=TRUE), file="/media/EPGE/Docs/dados/saddleSession.Rdata")

# Functions to calculate the square root of Fisher info (fi)
# and the information distance (id), i.e., the integral from the null
# to the alternative.
# obs: Note that design 2 has a different normalizing factor to obtain the average info

fi11 <- approxfun(rho.range[1, i], sqrt(fisher.info[1, i]/n.range[1]))
id11 <- function(rho) integrate(fi11, 0, rho)

id11 <- function(rho) apply(rho, id11)
bty="n"

# Design 2
163 xlim<-range(rho.range[2,])
164 curve(pwe21,xlim[1],xlim[2],ylim=c(0,1), type="l", lty=1,
165 xlab=expression(rho[1]), ylab= expression(beta(rho[1])))
166 curve(pwe22,xlim[1],xlim[2], lty=2,add=TRUE)
167 curve(pwe23,xlim[1],xlim[2], lty=3,add=TRUE)
168 legend.text <- c(paste("Matrix", n.range, "x",n.range))
169 legend("bottomleft", legend=legend.text, lty=1:length(n.range),
170 bty="n")

172 # Design 3
173 xlim<-range(rho.range[3,])
174 curve(pwe31,xlim[1],xlim[2],ylim=c(0,1), type="l", lty=1,
175 xlab=expression(rho[1]), ylab= expression(beta(id)))
176 curve(pwe32,xlim[1],xlim[2], lty=2,add=TRUE)
177 curve(pwe33,xlim[1],xlim[2], lty=3,add=TRUE)
178 legend.text <- c(paste("Matrix", n.range, "x",n.range))
179 legend("bottomleft", legend=legend.text, lty=1:length(n.range),
180 bty="n")

178 # Design 4
181 xlim<range(rho.range[4,])
182 curve(pwe41,xlim[1],xlim[2],ylim=c(0,1), type="l", lty=1,
183 xlab=expression(rho[1]), ylab= expression(beta(id)))
184 curve(pwe42,xlim[1],xlim[2], lty=2,add=TRUE)
185 curve(pwe43,xlim[1],xlim[2], lty=3,add=TRUE)
186 legend.text <- c(paste("Matrix", n.range, "x",n.range))
187 legend("bottomleft", legend=legend.text, lty=1:length(n.range),
188 bty="n")

188 # Comparing Saddlepoint and Edgeworth
197 mlist=ls()  
198 source("/media/EPGE/Docs/programs/all_functions.R")
199 X11.options(width=5,height=5, type="cairo")
200 ps.options(horizontal=FALSE, onefile=FALSE, height=4.5,
201 width=6, paper="special", pointsize=10)
203 epsilon <- .Machine$double.eps^0.25
204 m <- 2 # with m-ahead and m-behind neighbors
205 j<-9
206 alpha<-0.05
207 n <- 200
208 len.rho.range<60 # HAS to be EVEN
209 power <- matrix(NA, nrow=3, ncol=len.rho.range) # matrix to
210 fisher.info<-matrix(NA,nrow=2,ncol=len.rho.range)
212 M <- genDesign4(n,n,1)
Listing 3: chart05.R

```r
rm(list=ls())
source("/media/EPGE/Docs/programas/all_functions.R")
X11.options(width=6, height=6, type="ncairo")
ps.options(horizontal=FALSE, onefile=FALSE, height=4.5, width=6, paper="special", pointsize=10)

m<25
a.range <- seq(from=0.001, to=0.2, len=50)
rho1<0.1
M<circum(M,n,m)
C<CO(0, rho1, M)
k1<-KelinhT(1,0,C)
k2<-KelinhT(2,0,C)
cv1<-cv2<-cv3<-matrix(NA,nrow=length(a.range),ncol=1)
for (j in 1:length(a.range)) {
a.range[j]
cat("alpha=")
cat(a,"\n")
cv1[j]<-(saddlepointCV(a,C)-k1)/sqrt(k2)
cv2[j]<-(saddlepointCV(a,C)-k1)/sqrt(k2)
cv3[j]<-qnorm(1-a)
}
postscript("/media/EPGE/Docs/figures/chart05a.eps")
plot(1-a.range,cv1,type="l",lty=1,xlab=expression(1-alpha),
     ylab="Standardized critical value")
lines(1-a.range,cv2,lty=2)
lines(1-a.range,cv3,lty=3)
legend.text<-c("numerical solution", "saddlepoint inversion", "standard normal")
legend("topleft", legend=legend.text, lty=1:3, bty="n")
dev.off()
## muda n
re(list=ls())
source("/media/EPGE/Docs/programas/all_functions.R")
X11.options(width=5, height=5, type="ncairo")
ps.options(horizontal=FALSE, onefile=FALSE, height=4.5, width=6, paper="special", pointsize=10)
m<50
```
```r
a.range <- seq(from=0.001, to=0.2, len=50)
mc<-2
rho1<-0.1
M<-circularM(n,m)
C<-CO(0,rho1,M)
k1<-xslinhaT(1,0,C)
k2<-xslinhaT(2,0,C)
cv1<-cv2<-cv3<-matrix(NA,nrow=length(a.range),ncol=1)

for (j in 1:length(a.range)) {
a<-a.range[j]
cat("alpha= ",a,"
")
cv1[j]<-/(saddlepointCV(a,C)-k1)/sqrt(k2)
cv2[j]<-/(saddlepointCV(a,C)-k2)/sqrt(k2)
cv3[j]<-qnorm(1-a)
}

dev.off()

# Critical value vs. rho1
# using saddlepoint
rm(list="ls())
source("/media/EPGE/Docs/programas/all_functions.R")
X11.options(width=5, height=5, type="ncairo")
ps.options(horizontal=FALSE, onefile=FALSE, height=4.5,
width=6, paper="special", pointsize=10)

c<-10
mc<-1
M<-circularM(n,m)
rho_.0<-0
epsilon <- .Machines$double.eps*0.5
rho.range <- seq(from=0, to=1/(2*m)-sqrt(epsilon), len=100)
alpha<0.05
c<-NA
powerfix<-NA

C<-CO(0,0,M)
attributes(C)<-c(attributes(M),list("rhopositive"=TRUE))

c<-saddlepointCV(alpha,C)
c<-saddlepointCV(alpha,C)
Bfix <- B(0,0,M)

for (i in 1:length(rho.range)) {
 rho1<-rho.range[i]
 C<-C-Ca(0,rho1,M)
 C<-Linv(C,M)
 attributes(C) <- c(attributes(M), list("rhopositive"=TRUE))
 powerfix[i]<-LagmanniRicePT(Ca,CA)
cat("+-")
cat("n")
}

poweropt <- powerLR(rho.range, alpha,M,TRUE)
#poweropt2 <- powerES(rho.range, alpha,M,TRUE)
summary(powerloss <- (powerfix - poweropt))

postscript("/media/EPGE/Docs/figures/chart06b.pdf")
par(mar=c(5,4,2,2)+0.1)
layout(matrix(c(1,2,2,1))
```

Listing 4: chart06.R
```r
plot(rho.range, poweropt, ylim=c(0,1), type="l", lty=1,
     xlab=expression(rho[1]), ylab= expression(beta(rho[1])))
lines(rho.range, powerfix, lty=1)
legend("bottomright", legend=legend.text, lty=1:2, byy="n")
plot(rho.range, powerloss, type="l", xlab=expression(rho[1]), ylab="Difference")
device.off()

## o MESMO NO RANGE NEGATIVO
rm(list=ls())
source("/media/EPGE/Docs/programas/all_functions.R")
X11.options(width=5, height=5, type="nbcairo")
ps.options(horizontal=FALSE, onefile=FALSE, height=4.5,    
            width=6, paper="special", pointsize=10)
mx<50
m<4
M< circularM(n,m)
 rho_0 <- 0
epsilon <- .Machine$double.eps^0.5
rho.range <- seq(from=-1/(2*m)+sqrt(epsilon), to=0, len=50)
alpha<0.1
cA<-NA
powerfix<-NA
 C<-C0(0,0,M) # aqui eu posso colocar rho1=0 mesmo!!
ca<-saddlepinCV(alpha,C)

for (i in 1:length(rho.range)) {
si<-rho.range[i]
CtA<-Ca(0,rho1,M)
powerfix[i]<-1-LugannaniRicePT(ca,CtA) # mudar aqui tbm
}
poweropt<-powerLR(rho.range, alpha,M,TRUE)
powerloss <- (powerfix - poweropt)

postscript("/media/EPGE/Docs/figures/chart06a.ep")

par(mar=c(5,4,2,2)+0.1)
layout(matrix(c(1,2,2,1)))
plot(rho.range, poweropt, ylim=c(0,1), type="l", lty=1,
     xlab=expression(rho[1]), ylab= expression(beta(rho[1])))
lines(rho.range, powerfix, lty=1)
legend("bottomleft", legend=legend.text, lty=1:2, byy="n")
plot(rho.range, powerloss, type="l", xlab=expression(rho[1]), ylab="Difference")
device.off()

## Agora para outros testes que não o score (ou seja para diferentes
## valores de rho1
rm(list=ls())
source("/media/EPGE/Docs/programas/all_functions.R")
X11.options(width=5, height=5, type="nbcairo")
poweropt<-powerLR(rho.range, alpha,M,TRUE)
powerloss <- (powerfix - poweropt)

postscript("/media/EPGE/Docs/figures/chart06a.ep")
```

This code snippet appears to be a R script for generating plots and performing statistical tests. It includes functions for calculating power and confidence intervals, and it is used to visualize the power of statistical tests across different rho values. The script also includes options for customizing the appearance of the plots, such as changing the size and type of markers and the thickness of lines.
\begin{verbatim}
117  rho1<-rho.range[i]
118  CA<-Ca(0, rho1, M)
119  powerfix[i]<-LugannaniRicePT(ca, CA)
120  cat("+-")
121  cat("\n")
122  poweropt <- powerLR(rho.range, alpha,M,TRUE)
123  powerloss <- (powerfix - poweropt)
124
125  postscript("/media/EPGE/Docs/figures/chart06b.eps")
126  par(mar=c(5,4,2,2)+0.1)
127  layout(matrix(c(1,2),2,1))
128  plot(rho.range, poweropt, ylim=c(0,1), type="l", lty=1,
129   xlab=expression(rho[1]), ylab= expression(beta(rho[1])))
130  lines(rho.range, powerfix, lty=1)
131  legend.text <- c("optimum test power", "score test power")
132  legend("bottomright", legend=legend.text, lty=1:2, bty="n")
133  plot(rho.range, powerloss, type="l",xlab=expression(rho[1]),ylab="Difference")
134  dev.off()
135
136  ## REAL LIFE EXAMPLE
137  rm(list=ls())
138  source("/media/EPGE/Docs/programs/all.functions.R")
139  X1t.options(width=5, height=5, type="nbczero")
140  ps.opt=option(horizontal=FALSE, onefile=FALSE, height=4.5,
141   width=6, paper="special", fontsize=10)
142  load("/media/EPGE/Docs/dados/matrix_exemplo/RJWexample.Rdata")
143  MB<-M
144  rm(M)
145
146  M<MB[["Contiguity1_N"]]
147  print(rho.lims <- 1/eigenRange(M))
148  epsilon <- .Machine$double.eps^0.5
149  rho.range <- seq(from=0, to=rho.lims[2]-0.05, len=100)
150  alpha<-0.01
151  ca<-NA
152  powerfix<-NA
153  C<-CO(0,0,M)
154  attributes(C)<-c(attributes (M),list("rhopositive"=TRUE))
155  ca<-saddlepointCV(alpha,C)
156  #ca<-saddlepointCV(alpha,C)
157  Bfix <- B(0,0,M)
158
159  for (i in 1:length(rho.range)) {
160    rho1<-rho.range[i]
161    #CA<-Ca(0,rho1,M)
162    CA<-t(Linr(rho1,M))%*%Bfix %*% Linr(rho1,M)
163    # attributes(CA) <- c(attributes (M), list("rhopositive"=TRUE))
164    powerfix[i]<-LugannaniRicePT(ca,CA)
165    cat("+-")
166    cat("\n")
167  poweropt <- powerLR(rho.range, alpha,M,TRUE)
168  #poweropt2<-powerEd(rho.range, alpha,M,TRUE)
169  summary(powerloss <- (powerfix - poweropt))
170
171  postscript("/media/EPGE/Docs/figures/chart06b.eps")
172  par(mar=c(5,4,2,2)+0.1)
173  layout(matrix(c(1,2),2,1))
174  plot(rho.range, poweropt, ylim=c(0,1), type="l", lty=1,
175   xlab=expression(rho[1]), ylab= expression(beta(rho[1])))
176  lines(rho.range, powerfix, lty=1)
177  legend.text <- c("optimum test power", "score test power")
178  legend("bottomright", legend=legend.text, lty=1:2, bty="n")
179  plot(rho.range, powerloss, type="l",xlab=expression(rho[1]),ylab="Difference")
\end{verbatim}
Listing 5: chart07modified.R

```r
# Power vs size - using Edgeworth
# Modified to calculate using information distance (id) as parameter

eval(list ls()) # Apaga todas variáveis da memória
source("/media/EPGE/Docs/programas/all_functions.R")
par(mai=c(5,5,5,5), mar=c(5,5,5,5), mgp=c(5,2,0.5))
ps.options(horizontal=FALSE, onefile=FALSE, height=4.5,
width=6, paper="special", pointsize=10)

epsilon <- .Machine$double.eps^0.5 # defines an arbitrarily small value
m<-36
m <- 2 # with m-ahead and m-behind neighbors
J<-15 # so it has a mean degree 8.55 (comparable to the other 3 designs range 4 a 12
len.rho.range<-150 # HAS to be EVEN
rhow.range<-array(NA, dim=c(4,1,len.rho.range))
a.range <- seq(from=0-.Machine$double.eps^0.25, to=1-.Machine$double.eps^0.25, len=50)
#powerf <- array(NA, dim=c(4,1,len.rho.range, length(a.range)))
power <- array(NA, dim=c(4,4,len.rho.range),5))
fisher.info<-array(NA, dim=c(4,4,len.rho.range))
rho.set<-matrix(NA,4,5) # designs, number of elements rho

dx<-vector("list",4)
dx[[1]]<-genDesign1(n,m)
dx[[2]]<-genDesign2(n,m)
dx[[3]]<-genDesign3(n,m)
dx[[4]]<-genDesign4(n,m)

r<1/eigenRange(M[[1]])
rho.range[1,1]<-c(seq(from=r[1]+sqrt(epsilon), to=sqrt(epsilon),len=len.rho.range/2)
,0,
seq(from=sqrt(epsilon), to=r[2]-sqrt(epsilon),len=len.rho.range/2-1))
fisher.info[1,1]<-sapply(rho.range[1,1],FisherInfoRho,M[[1]])

t<1/eigenRange(M[[2]])
rho.range[2,1]<-c(seq(from=r[1]+sqrt(epsilon), to=sqrt(epsilon),len=len.rho.range/2)
,0,
seq(from=sqrt(epsilon), to=r[2]-sqrt(epsilon),len=len.rho.range/2-1))
fisher.info[2,1]<-sapply(rho.range[2,1],FisherInfoRho,M[[2]])

r<1/eigenRange(M[[3]])
rho.range[3,1]<-c(seq(from=r[1]+sqrt(epsilon), to=sqrt(epsilon),len=len.rho.range/2)
,0,
seq(from=sqrt(epsilon), to=r[2]-sqrt(epsilon),len=len.rho.range/2-1))
fisher.info[3,1]<-sapply(rho.range[3,1],FisherInfoRho,M[[3]])

r<1/eigenRange(M[[4]])
rho.range[4,1]<-c(seq(from=r[1]+sqrt(epsilon), to=sqrt(epsilon),len=len.rho.range/2)
,0,
seq(from=sqrt(epsilon), to=r[2]-sqrt(epsilon),len=len.rho.range/2-1))
fisher.info[4,1]<-sapply(rho.range[4,1],FisherInfoRho,M[[4]])

f11<-approxfun(rho.range[1,1],sqrt(fisher.info[1,1,])/n)
id11<-function(rho) integrate(f11,0,rho)$value
f12<-approxfun(rho.range[2,1],sqrt(fisher.info[2,1,]/(n/floor(sqrt(n)))))
id12<-function(rho) integrate(f12,0,rho)$value
f13<-approxfun(rho.range[3,1],sqrt(fisher.info[3,1,])/n)
id13<-function(rho) integrate(f13,0,rho)$value
f14<-approxfun(rho.range[4,1],sqrt(fisher.info[4,1,])/n)
id14<-function(rho) integrate(f14,0,rho)$value

# calcula qual o valor de rho dado alguns valores id
id.range<-seq(0,.8,len=5)
for (j in 1:5) {
 rho.set[1,j] <- uniroot(function(c) id11(c)-id.range[j],c(0,rho.range[1,1,len.rho.range]))$root
}
```

rho.set[2,j] <- univar(lfunction(c) id21(c)-id.range[j],c(0,rho.range[2,1],len.rho.range[1]))
}$root
rho.set[3,j] <- univar(lfunction(c) id31(c)-id.range[j],c(0,rho.range[3,1],len.rho.range[1]))
}$root
rho.set[4,j] <- univar(lfunction(c) id41(c)-id.range[j],c(0,rho.range[4,1],len.rho.range[1]))
}$root

for (i in 1:length(a.range)) {
  alpha<-a.range[i]
  power[1,i]<-powerE4(rho.set[1,i],alpha,M[[1]],1,TRUE)
  power[2,i]<-powerE4(rho.set[2,i],alpha,M[[2]],floor(sqrt(n)),TRUE)
  power[3,i]<-powerE4(rho.set[3,i],alpha,M[[3]],1,TRUE)
  power[4,i]<-powerE4(rho.set[4,i],alpha,M[[4]],1,TRUE)
}

## Design 1
#postscript("/media/EPGE/Docs/figures/chart07.eps")
plot(a.range, power[1,i], ylim=c(0,1), type="l", lty=1,
     xlab=expression(alpha), ylab= expression(beta(rho[1])))
for (i in 2:length(rho.range)) {
  lines(a.range,power[1,i],lty=i)
}

text(.45,.45,substitute(list(rho)x, list(x=rho.range[1,i])),pos=4)
text(.33,.6,substitute(list(rho)x, list(x=rho.range[1,2])),pos=4)
text(.18,.75,substitute(list(rho)x, list(x=rho.range[1,3])),pos=4)
text(.05,.83,substitute(list(rho)x, list(x=rho.range[1,4])),pos=4)
text(-.02,1,substitute(list(rho)x, list(x=rho.range[1,5])),pos=4)
dev.off()

## Design 2
#postscript("/media/EPGE/Docs/figures/chart07.eps")
plot(a.range, power[2,i], ylim=c(0,1), type="l", lty=1,
     xlab=expression(alpha), ylab= expression(beta(rho[1])))
for (i in 2:length(rho.range)) {
  lines(a.range,power[2,i],lty=i)
}

text(.45,.45,substitute(list(rho)x, list(x=rho.range[2,1])),pos=4)
text(.33,.6,substitute(list(rho)x, list(x=rho.range[2,2])),pos=4)
text(.18,.75,substitute(list(rho)x, list(x=rho.range[2,3])),pos=4)
text(.05,.83,substitute(list(rho)x, list(x=rho.range[2,4])),pos=4)
text(-.02,1,substitute(list(rho)x, list(x=rho.range[2,5])),pos=4)
dev.off()

## Design 3
#postscript("/media/EPGE/Docs/figures/chart07.eps")
plot(a.range, power[3,i], ylim=c(0,1), type="l", lty=1,
     xlab=expression(alpha), ylab= expression(beta(rho[1])))
i<-.5
for (i in 2:length(rho.range)) {
  lines(a.range,power[3,i],lty=i)
}

text(.45,.45,substitute(list(rho)x, list(x=rho.range[3,1])),pos=4)
text(.33,.6,substitute(list(rho)x, list(x=rho.range[3,2])),pos=4)
text(.18,.75,substitute(list(rho)x, list(x=rho.range[3,3])),pos=4)
text(.05,.83,substitute(list(rho)x, list(x=rho.range[3,4])),pos=4)
text(-.02,1,substitute(list(rho)x, list(x=rho.range[3,5])),pos=4)
dev.off()

## Design 4
#postscript("/media/EPGE/Docs/figures/chart07.eps")
plot(a.range, power[4,i], ylim=c(0,1), type="l", lty=1,
     xlab=expression(alpha), ylab= expression(beta(rho[1])))}
for (i in 2:length(rho.range)) {
  lines(a.range, power[4, i], lty=1)
}

text(.45, .45, substitute(list(rho)==x, list(x=rho.range[4:1])), pos=4)
text(.33, .6, substitute(list(rho)==x, list(x=rho.range[4:2])), pos=4)
text(.18, .75, substitute(list(rho)==x, list(x=rho.range[4:3])), pos=4)
text(.05, .85, substitute(list(rho)==x, list(x=rho.range[4:4])), pos=4)
text(-.02, .1, substitute(list(rho)==x, list(x=rho.range[4:5])), pos=4)

dev.off()

# Compare the critical values obtained using different methods
rm(list=ls()) # Apaga todas variáveis da memória
source("/media/EPGE/Docs/programas/all_functions.R")
X11.options(width=5, height=5, type="ncbcairo")
ps.options(horizontal=FALSE, onefile=FALSE, height=4.5, 
           width=6, paper="special", pointsize=10)

n<-.26
m<-2
M<.circularM(n,m)
S<-1999
alpha<-.05
sigma2<-1
rho0<-0
rho1<-0.3/(2*m)
Li<-Linv(rho0,M)
t<-.NA

c<.CO(rho0,rho1,M)
##PT(c<,C)
k1<.KslinhaT(1,0,C)
k2<.KslinhaT(2,0,C)

print(ca.ew <- (Ew4CV(alpha,C)-k1)/sqrt(k2))
print(ca.cf <- (CnF4CV(alpha,C)-k1)/sqrt(k2))
print(ca.sp <- (saddlepointCV(alpha,C)-k1)/sqrt(k2))
print(ca.zi <- (saddlepointCV(alpha,C)-k1)/sqrt(k2))

postscript("/media/EPGE/Docs/figures/chart08.eps")
set.seed(2000)
chart<-FALSE
c<.NA
for (s in 1:5) {
  U<.rnorm(n,0,sqrt(sigma2))
  Y<.LIX*U
  ts[s]<.Tn(Y,rho0,rho1,M,smga2)
  ord<.alpha*(s+1)
  if (ceiling(ord)==floor(ord))#(chart) {
    ca[s]<.sort(ts,decreasing=TRUE)[ord]-mean(ts)/sd(ts)
    plot(s, ca[s], xlim=c(1,5), ylim=c(0,3), pch=21, cex=.5, bg="white",
         xlab="Simulation",
         ylab=expression(paste("Critical values (standardized, level ",
         alpha," )"))
    abline(h=.(ca.ew,ca.cf,ca.sp,ca.si), lty=2:5)
    leg.text<.-c("Simulated", "Edgeworth", "Cornish-Fisher", "Saddlepoint",
               "Inverse saddlepoint")
    legend("bottomright", lty=.(NA,2:5), pch=.(21,NA,NA,NA,NA), bty="n",
           legend=leg.text)
    chart<.TRUE
  }
  else if (ceiling(ord)==floor(ord)) {
    ca[s]<.sort(ts,decreasing=TRUE)[ord]-mean(ts)/sd(ts)
    points(s, ca[s], pch=21, cex=.5, bg="white")
  }
}

dev.off()
Listing 7: chart10.R

```r
rm(list="ls")
source("/media/EPGE/Docs/programs/all_functions.R")
X11.options(width=5, height=5, type="ncairo")
ps.options(horizontal=FALSE, onefile=FALSE, height=4.5,
           width=6, paper="special", pointsize=10)
library(igraph)

n<-20
m<-2
j<-7
N<-500
calpha<-matrix(NA,N,m)
talpha <- qnorm(1-0.05)

for (n in n0:N)
  M<-vector("list",4)
  M[[1]]<-genDesign1(n,m)
  M[[2]]<-genDesign2(n,m)
  M[[3]]<-genDesign3(n,m)
  M[[4]]<-genDesign4(n,m,j)
  k<-vector("list",4)
  ki<-vector("list",4)
  for (i in 1:4)
    ki[i]<-cumulantsT(4,0,0,0,M[[1]], ifelse(i==2,floor(sqrt(n)),1))
    ki[i]<-invariants(k[i])
    calpha[n,m+1,i]<-(3*ki[i][4] -4*ki[i][3]*talpha^2 -12*ki[i][3]*talpha + 12*ki[i][3]*talpha^2 + (72+10)*ki[i][3]*talpha + 12*ki[i][3]*talpha^2 )/72
  cat("...",n)

# Plot
postscript("/media/EPGE/Docs/figures/chart10.ps")
plot(n0:n, calpha[,1], ylim=c(1.5,2), type="l", lty=2, xlab="Sample size (n)", ylab=expression(paste("Critical values (standardized, level \(\alpha\), alpha,"=5\%\)")))
lines(n0:n, calpha[,2], lty=3)
lines(n0:n, calpha[,3], lty=4)
lines(n0:n, calpha[,4], lty=5)
abline(h=talpha, lty=1)
legend("topright", lty=c(2,6,1), bty="n", legend=leg.text)
dev.off()
```

Listing 8: chart11.R

```r
rm(list="ls")
source("/media/EPGE/Docs/programs/all_functions.R")
X11.options(width=5, height=5, type="ncairo")
ps.options(horizontal=FALSE, onefile=FALSE, height=4.5,
           width=6, paper="special", pointsize=10)
library(igraph)

n<-100
m<-2
j<-25
mM<-1
mM<-20
calpa<-matrix(NA,mM,m0+1,4)
navgiz<-matrix(NA,mM,m0+1,4)
talpha <- qnorm(1-0.05)

for (m in m0:mM)
  M<-vector("list",4)
  M[[1]]<-genDesign1(n,m)
  M[[2]]<-genDesign2(n,m)
  M[[3]]<-genDesign3(n,m)
  M[[4]]<-genDesign4(n,m,j)
```
Listing 9: chart12modified.R

```r
rm(list=ls()) # Apaga todas variáveis da memória
source("all_functions.R")
X1I.options(width=5, height=5, type="nbcairo")
p.s.options(horizontal=FALSE, onefile=FALSE, height=4.5,
width=6, paper="special", pointsize=8)

# Plot power Envelopes for circular world design with n=100, m=2, m=5, m=10.
epsilon <- Machine$double.eps=0.5 # defines an arbitrarily small value
alpha <- 0.05 # size alpha test (limits the type I error)
## n changed from original file chart12.R
n <- 36 # n x n matrix
Jc<9
len.rho.range<-150 # HAS to be EVEN
m.range <- c(2,5,10)
rho.range<-array(NA, dim=c(4,length(m.range),len.rho.range))
fisher.info<-array(NA, dim=c(4,length(m.range),len.rho.range))
powerf <- array(NA, dim=c(4,length(m.range),len.rho.range))
M<-vector("list",4)

for (i in 1:length(m.range)) {
  M[[i]]<-genDesign1(n, m.range[i])
  M[[2]]<-genDesign2(n, m.range[i])
  M[[3]]<-genDesign3(n, m.range[i])
  M[[4]]<-genDesign4(n, m.range[i], J)
  rr<-1/eigenRange(M[[i]])
  rho.range[i,1,] <- c(seq(from=rr[1]+sqrt(epsilon), to=-sqrt(epsilon), len=len.rho.range/2)
,0,
    seq(from=sqrt(epsilon), to=rr[2]-sqrt(epsilon), len=len.rho.range/2-1)
  )
  fisher.info[i,1,] <- sapply(rho.range[i,1], FisherInfoRho,M[[i]])
  #fisher.at.zero[1,1,1]<-FisherInfoRho(0,M[[1]])
  rr<-1/eigenRange(M[[2]])
  rho.range[i,2,] <- c(seq(from=rr[1]+sqrt(epsilon), to=-sqrt(epsilon), len=len.rho.range/2)
,0,
    seq(from=sqrt(epsilon), to=rr[2]-sqrt(epsilon), len=len.rho.range/2-1)
  )
  fisher.info[i,2,1,] <- sapply(rho.range[i,2], FisherInfoRho,M[[2]])
  #fisher.at.zero[2,1,1]<-FisherInfoRho(0,M[[2]])
  rr<-1/eigenRange(M[[3]])
  rho.range[i,3,] <- c(seq(from=rr[1]+sqrt(epsilon), to=-sqrt(epsilon), len=len.rho.range/2)
,0,
    seq(from=sqrt(epsilon), to=rr[2]-sqrt(epsilon), len=len.rho.range/2-1)
  )
  fisher.info[i,3,1,] <- sapply(rho.range[i,3], FisherInfoRho,M[[3]])
  #fisher.at.zero[3,1,1]<-FisherInfoRho(0,M[[3]])
  rr<-1/eigenRange(M[[4]])
  rho.range[i,4,] <- c(seq(from=rr[1]+sqrt(epsilon), to=-sqrt(epsilon), len=len.rho.range/2)
,0,
    seq(from=sqrt(epsilon), to=rr[2]-sqrt(epsilon), len=len.rho.range/2-1)
  )
  fisher.info[i,4,1,] <- sapply(rho.range[i,4], FisherInfoRho,M[[4]])
  #fisher.at.zero[4,1,1]<-FisherInfoRho(0,M[[4]])
```

\[ \text{rho} \cdot \text{range} [3, i, j] <- \text{c(seq(from=rho[1] + \sqrt{\text{epsilon}}, to=-\sqrt{\text{epsilon}}), len=len \cdot \text{rho} \cdot \text{range}/2)} \] 
\[ 0, \] 
\[ \text{seq(from=sqrt(\text{epsilon}), to=rho[2] - \sqrt{\text{epsilon}}), len=len \cdot \text{rho} \cdot \text{range}/2-1)} \] 
\[ \text{fisher.info}[3, i, j] <- \text{apply(rho.range[3, i, j], FisherInfRho, M[[3]])} \] 
\[ \text{#fisher.at.zero[3, i, j] <- FisherInfRho(0, M[[3]])} \] 
\[ \text{rr} <- 1/eigenRange(M[[4]]) \] 
\[ \text{rho.range[4, i, j] <- c(seq(from=rho[1] + \sqrt{\text{epsilon}}, to=-\sqrt{\text{epsilon}}), len=len \cdot \text{rho} \cdot \text{range}/2)} \] 
\[ 0, \] 
\[ \text{seq(from=sqrt(\text{epsilon}), to=rho[2] - \sqrt{\text{epsilon}}), len=len \cdot \text{rho} \cdot \text{range}/2-1)} \] 
\[ \text{fisher.info}[4, i, j] <- \text{apply(rho.range[4, i, j], FisherInfRho, M[[4]])} \] 
\[ \text{#fisher.at.zero[4, i, j] <- FisherInfRho(0, M[[4]])} \] 
\[ \text{power}[1, i, j] <- \text{powerE4(rho.range[1, i, j], alpha, M[[1]], 1, TRUE)} \] 
\[ \text{power}[2, i, j] <- \text{powerE4(rho.range[2, i, j], alpha, M[[2]], floor(\sqrt{n})), TRUE} \] 
\[ \text{power}[3, i, j] <- \text{powerE4(rho.range[3, i, j], alpha, M[[3]], 1, TRUE)} \] 
\[ \text{power}[4, i, j] <- \text{powerE4(rho.range[4, i, j], alpha, M[[4]], 1, TRUE)} \] 
\] 
\[ \text{fill1} <- \text{approxfun(rho.range[1, i, j], sqrt(fisher.info[1, i, j]/n))} \] 
\[ \text{id11} <- \text{function(rho) integrate(fill1, 0, rho)$value} \] 
\[ \text{id11f} <- \text{function(rho) sapply(rho, id11)} \] 
\[ \text{id12} <- \text{approxfun(rho.range[1, 2, i, j], sqrt(fisher.info[1, 2, i, j]/n))} \] 
\[ \text{id12f} <- \text{function(rho) sapply(rho, id12)} \] 
\[ \text{id13} <- \text{approxfun(rho.range[1, 3, i, j], sqrt(fisher.info[1, 3, i, j]/n))} \] 
\[ \text{id13f} <- \text{function(rho) sapply(rho, id13)} \] 
\[ \text{id21} <- \text{approxfun(rho.range[2, 1, i, j], sqrt(fisher.info[2, 1, i, j]/(n/floor(\sqrt{n})}))} \] 
\[ \text{id21f} <- \text{function(rho) sapply(rho, id21)} \] 
\[ \text{id22} <- \text{approxfun(rho.range[2, 2, i, j], sqrt(fisher.info[2, 2, i, j]/(n/floor(\sqrt{n}))})} \] 
\[ \text{id22f} <- \text{function(rho) sapply(rho, id22)} \] 
\[ \text{id23} <- \text{approxfun(rho.range[2, 3, i, j], sqrt(fisher.info[2, 3, i, j]/(n/floor(\sqrt{n}))})} \] 
\[ \text{id23f} <- \text{function(rho) sapply(rho, id23)} \] 
\[ \text{id31} <- \text{approxfun(rho.range[3, 1, i, j], sqrt(fisher.info[3, 1, i, j]/n))} \] 
\[ \text{id31f} <- \text{function(rho) sapply(rho, id31)} \] 
\[ \text{id32} <- \text{approxfun(rho.range[3, 2, i, j], sqrt(fisher.info[3, 2, i, j]/n))} \] 
\[ \text{id32f} <- \text{function(rho) sapply(rho, id32)} \] 
\[ \text{id33} <- \text{approxfun(rho.range[3, 3, i, j], sqrt(fisher.info[3, 3, i, j]/n))} \] 
\[ \text{id33f} <- \text{function(rho) sapply(rho, id33)} \] 
\[ \text{id41} <- \text{approxfun(rho.range[4, 1, i, j], sqrt(fisher.info[4, 1, i, j]/n))} \] 
\[ \text{id41f} <- \text{function(rho) sapply(rho, id41)} \] 
\[ \text{id42} <- \text{approxfun(rho.range[4, 2, i, j], sqrt(fisher.info[4, 2, i, j]/n))} \] 
\[ \text{id42f} <- \text{function(rho) sapply(rho, id42)} \] 
\[ \text{id43} <- \text{approxfun(rho.range[4, 3, i, j], sqrt(fisher.info[4, 3, i, j]/n))} \] 
\[ \text{id43f} <- \text{function(rho) sapply(rho, id43)} \]
111)

# Grafico Design 1
112 postscript("/media/EPG/Docs/figures/chart12a.eps")
113 curve(pwe_id11,-1.1,ylim=c(0.1), type="l", lty=1,
114 xlab="information distance (id)", ylab=expression(beta(id)))
115 curve(pwe_id12,-1.1,lty=2,add=TRUE)
116 curve(pwe_id13,-1.1,lty=3,add=TRUE)
117 legend.text <- c(paste("Matrix", n, " x", n, " with m
118 \"ahead\" behind neighbors"), paste("Matrix with m
119 \"ahead\" behind neighbors"), paste("Matrix with m"., m.range[-1]))
120 legend("bottomleft", legend=legend.text, lty=1:length(m.range),
121 bty="n")
122 dev.off()

# Grafico Design 2
123 postscript("/media/EPG/Docs/figures/chart12b.eps")
124 curve(pwe_id21,-1.1,ylim=c(0.1), type="l", lty=1,
125 xlab="information distance (id)", ylab=expression(beta(id)))
126 curve(pwe_id22,-1.1,lty=2,add=TRUE)
127 curve(pwe_id23,-1.1,lty=3,add=TRUE)
128 legend.text <- c(paste("Matrix", n, " x", n, " with m
129 \"ahead\" behind neighbors"), paste("Matrix with m"., m.range[-1]))
130 legend("bottomleft", legend=legend.text, lty=1:length(m.range),
131 bty="n")
132 dev.off()

# Grafico Design 3
133 postscript("/media/EPG/Docs/figures/chart12c.eps")
134 curve(pwe_id31,-1.1,ylim=c(0.1), type="l", lty=1,
135 xlab="information distance (id)", ylab=expression(beta(id)))
136 curve(pwe_id32,-1.1,lty=2,add=TRUE)
137 curve(pwe_id33,-1.1,lty=3,add=TRUE)
138 legend.text <- c(paste("Matrix", n, " x", n, " with m
139 \"ahead\" behind neighbors"), paste("Matrix with m"., m.range[-1]))
140 legend("bottomleft", legend=legend.text, lty=1:length(m.range),
141 bty="n")
142 dev.off()

# Grafico Design 4
143 postscript("/media/EPG/Docs/figures/chart12d.eps")
144 curve(pwe_id41,-1.1,ylim=c(0.1), type="l", lty=1,
145 xlab="information distance (id)", ylab=expression(beta(id)))
146 curve(pwe_id42,-1.1,lty=2,add=TRUE)
147 curve(pwe_id43,-1.1,lty=3,add=TRUE)
148 legend.text <- c(paste("Matrix", n, " x", n, " with m
149 \"ahead\" behind neighbors"), paste("Matrix with m"., m.range[-1]))
150 legend("bottomleft", legend=legend.text, lty=1:length(m.range),
151 bty="n")
152 dev.off()

# comparison of different designs for m=2
153 postscript("/media/EPG/Docs/figures/chart12e.eps")
154 curve(pwe_id11,-1.1,ylim=c(0.1), type="l", lty=1,
155 xlab=paste("information distance (id), sample size n="n"),
156 ylab=expression(beta(id) ))
157 curve(pwe_id21,-1.1,lty=2,add=TRUE)
158 curve(pwe_id31,-1.1,lty=3,add=TRUE)
159 curve(pwe_id41,-1.1,lty=4,add=TRUE)
160 legend.text <- c(paste("Design", i=":1:4"))
161 legend.text[4] <- paste(legend.text[4], " (J="J","), sep="")
162 legend("bottomleft", legend=legend.text, lty=1:4, bty="n")
163 dev.off()

# Power difference at id=0.15 is negligible
164 pwe_id11(0.15)
165 (pwe_id31(0.15)-pwe_id11(0.15))
166 (pwe_id41(0.15)-pwe_id11(0.15))
167 # The values of rho that correspond to id=0.15 are
168 r2<unroot(function(c) id21(c)-0.15, c0=rho.range[2,1,len.rho.range])
169 r3<unroot(function(c) id31(c)-0.15, c0=rho.range[3,1,len.rho.range])
170 r4<unroot(function(c) id41(c)-0.15, c0=rho.range[4,1,len.rho.range])
# And the respective maximum variance (sigma2=1)
A1 <- genDesign1(n, m.range[1])
A2 <- genDesign2(n, m.range[1])
A3 <- genDesign3(n, m.range[1])
A4 <- genDesign4(n, m.range[1])

range(diag(VCV(r1, A1, 1))))
range(diag(VCV(r2, A2, 1))))
range(diag(VCV(r3, A3, 1))))
range(diag(VCV(r4, A4, 1))))

image(cov2cor(VCV(r1, A1, 1))))
image(cov2cor(VCV(r2, A2, 1))))
image(cov2cor(VCV(r3, A3, 1))))
image(cov2cor(VCV(r4, A4, 1))))

Listing 10: comparesaddlemethodsforCV.R

rm(list=ls())
source("/media/EPGE/Docs/programs/all_functions.R")
X11.options(width=5, height=5, type="pdfcairo")
ps.options(horizontal=FALSE, onefile=FALSE, height=4.5, width=6, paper="special", pointsize=8)
n<25
a.range <- seq(from=0.001, to=0.2, len=50)

rho1<-0.1
M<-circularM(n,m)
C<-CO(0,rho1,M)
K1<-Kshelf(1,0,C)
K2<-Kshelf(2,0,C)

cv1<-cv2<-cv3<-matrix(NA, nrow=length(a.range), ncol=1)

for (j in 1:length(a.range)) {
  a<-a.range[j]
cat("alpha = ",a,\n")
cv1[j]<-saddlepointCV(a,C,K1)/sqrt(k2)
cv2[j]<-saddlepointinVC(a,C,K1)/sqrt(k2)
cv3[j]<-qnorm(1-a)
}

postscript("/media/EPGE/Docs/figures/chart05.eps")

plot(1-a.range, cv1, type="l", lty=1, xlab=expression(1-alpha),
ylab="Standardized critical value")
lines(1-a.range, cv2, lty=2)
lines(1-a.range, cv3, lty=3)
legend.text<-c("numerical solution", "saddlepoint inversion", "standard normal")
legend("topleft", legend=legend.text, lty=1:3, bty="n")
dev.off()

### FIM

cv1<-cv2<-cv3<-matrix(NA, nrow=length(n.range), ncol=length(alpha.range))
cv1.time<-cv2.time<-cv3.time<-
  matrix(NA, nrow=length(n.range), ncol=length(alpha.range))

for (j in 1:length(alpha.range)) {
  a<-alpha.range[j]
cv1.time[1,j] <- system.time(cv1[i,j]<-saddlepointCV(a,C,K1)/sqrt(k2))[3]
cv2.time[1,j] <- system.time(cv2[i,j]<-saddlepointinVC(a,C,K1)/sqrt(k2))[3]
cv3.time[1,j] <- system.time(cv3[i,j]<-qnorm(1-a))[3]
}

n<25
M<-circularM(n,m)
C<-CO(0,rho1,M)
alpha<-0.01

saddlepointinVC(a,C)

def in funcao de alpha
tbarf<-function(alpha) {
  K1<-Kshelf(1,0,C)
}
Listing 11: correlacao stat suf.R

```r
rm(list="ls")
source("/media/EPGE/Docs/programs/all_functions.R")
par(options(width=6, height=5, type="nbcairo"))
ps.options(horizontal=FALSE, onefile=FALSE, height=4.5,
      width=6, paper="special", pointsize=10)

n<25
m<4
M<: circularM(n,m)
rho_range <- 1/eigenRange(M)+.Machine$double.eps^.25*c(1,-1)
corr1t2 <- function(rho) {
  Lm4 <- Linv(rho,M)
  Lm4 <- powerM(Lm4,4)
  return(t(tr(powerM(M,3)%*%Lm4)/sqrt(t(powerM(M,2)%*%Lm4)*tr(powerM(M,4)%*%Lm4))))
}
corr1t2.vec <- function(rho) sapply(rho,corr1t2)
curve(corr1t2.vec, rho_range[1], rho_range[2])
corr1t2(0)
```

Listing 12: edgeworth functions R

```r
# Edgeworth functions
# cria os polinimos de hermite ate orden D
createHermite <- function(D=10) {
  require(Polynom)
x<-polynom()
H<-polynomial(1,x)
for(i in 2:D)
  H[[n+1]] <- x*H[[n]] - (n-1)*H[[n-1]]
  return(H[2:(D+1)])
}

# Approx de Edgeworth da distribucio aeth orden D(n-1)
E4 <- function(x,k) {
  h<:createHermite(5)
  return(pnorm(x) - dnorm(x) * (k[3]*h[[2]])(x)/factorial(3) +
    k[4]*h[[3]](x)/factorial(4) +
    10*k[3]^2*h[[5]](x)/factorial(6))
}

# Approx de Edgeworth da distribucio aeth orden D(n-3/2)
E6 <- function(x,k) {
  h<:createHermite(8)
  return(pnorm(x) - dnorm(x) * (k[3]*h[[2]])(x)/factorial(3) +
    k[4]*h[[3]](x)/factorial(4) +
    10*k[3]^2*h[[5]](x)/factorial(6) +
    k[5]*h[[4]](x)/factorial(5) +
    35*k[3]*k[4]*h[[6]](x)/factorial(7) +
    280*k[3]^3*h[[8]](x)/factorial(9))
}
# given the vector of cumulants k, returns the invariants
```
invariants <- function(k) {
  sizek<-length(k)
  ki <- k * k[2]^-((1:sizek)/2) #invariants
  return(c(0,1,ki[3:sizek]))
}

# Plots truncated Edgeworth and Gaussian as a ref
plotE4 <- function(x, nstdev=2.5, xrange=NULL, pfile=NULL) {
  if (is.null(xrange)) xrange<-c(k[1]-nstdev*sqrt(k[2]), k[1]+nstdev*sqrt(k[2]))
  ki <- invariants(k)
  if (!is.null(pfile)) postscript(paste("/media/EPGE/Docs/figures/",
                               pfile, sep=".", horizontal=FALSE, onefile=FALSE, height=4.5, width=6,
                               pointsize=10, paper="special")
  plot(function(a) E4((a-k[1])/sqrt(k[2]),ki), xrange[1], xrange[2],
       main="Edgeworth Approximation", xlab="t", ylab="F(t)",
       ylim=c(0,1), lty=1)
  curve(pnorm(x,mean=k[1],sd=sqrt(k[2])), add=TRUE, col="gray", lwd=1, lty=2)
  mtext("Edgeworth approx.", adj=0, lty=1)
  mtext("Gaussian", col="gray", adj=1, lty=2)
  if (!is.null(pfile)) dev.off()
}

plotE4v2 <- function(x, nstdev=2.5, xrange=NULL, pfile=NULL) {
  if (is.null(xrange)) xrange<-c(k[1]-nstdev*sqrt(k[2]), k[1]+nstdev*sqrt(k[2]))
  ki <- invariants(k)
  if (!is.null(pfile)) postscript(paste("/media/EPGE/Docs/figures/",
                                    pfile, sep=".", horizontal=FALSE, onefile=FALSE, height=4.5, width=6,
                                    pointsize=10, paper="special")
  layout(matrix(c(1,2),2,1))
  plot(function(a) E4((a-k[1])/sqrt(k[2]),ki), xrange[1], xrange[2],
       main="Edgeworth Approx", xlab="x", ylab="F(x)", ylim=c(0,1))
  curve(pnorm(x,mean=k[1],sd=sqrt(k[2])), add=TRUE, col="red", lwd=2)
  mtext("Edgeworth app.", adj=0)
  mtext("Gaussian", col="red", adj=1)
  plot(function(a) E4((a-k[1])/sqrt(k[2]),ki)-pnorm(a,mean=k[1],sd=sqrt(k[2])),
        xrange[1], xrange[2], main="Edgeworth - Gaussian", xlab="x",
        ylab="Distance")
  if (!is.null(pfile)) dev.off()
}

powerE4 <- function(rho.range, alpha=0.05, md=mddefault, verbose=FALSE) {
  # repeat the procedure outlined in the paper for all rho in rho.range
  h<-createHermite5() # for Cornish - Fisher
  power<-NA # re-sets the output variable
  nx<-dim(N)[1]
  smax <- 4 # as it is an E4 approximation
  for (ind in 1:length(rho.range)) {
    rho <- rho.range[ind]
    # Calculates the cumulants of T_n under the NULL and ALTERNATIVE
    k <- NA
    ka <- NA
    for (s in 1:smax) {
      ks <- 2*sqrt(factorial(s-1)*tr(powerM(A.rho.md,s))/n^(s/2))
      k(1)[s]<-2*(s-1)*factorial(s-1)*
      tr(powerM(k((linv(rho.md))%t% A.rho.md)%t%linv(rho.md,s))/ n^(s/2))
    }
    # calculate the invariants
    ki <- invariants(k)
    kai <- invariants(ka)
      # obtain the critical level c_a by Cornish - Fisher (only valid when smax=4)
    if (rho > 0) t_a <- qnorm(1-alpha) # the 1-alpha quantile of the Gaussian
    else t_a <- qnorm(alpha)
      # The critical value changes depending on whether rho is positive
    c_a <- k[1] + sqrt(k[2])*t_a + k[3]*h[2](t_a)/factorial(3) +
       k[4]*h[3](t_a)/factorial(4) -
       k[3]*2*(2*t_a^3 -5*t_a)/36
      # obtain the critical level by solving numerically
    if (rho > 0) {
      guess1<-qnorm(1-alpha,mean=k[1],sd=sqrt(k[2])) # base o valor critico N
      guess2<-qnorm(1-alpha/4,mean=k[1],sd=sqrt(k[2]))
      c_a.n <- uniroot(function(c) E4((c-k[1])/sqrt(k[2]),ki)+ alpha-1,}
lower=guess1, upper=guess2, tol=.Machine$double.eps^0.5)$root

else (guess1<qnorm(alpha/10, mean=k[1], sd=sqrt(k[2]));

    guess2<qnorm(alpha*10, mean=k[1], sd=sqrt(k[2]));

    c_a_n<uniroot(function(c) E4((c-k[1])/sqrt(k[2]), k1-alpha,

    lower=guess1, upper=guess2, tol=.Machine$double.eps^0.5)$root)

if (verbose) {

    # cat("For rho: ",rho," the critical levels are ",c_a, " and ", c_a_n,

    # " and ", t_a=sqrt(k[2])/k[1],"\n")

    # cat("For rho: ",rho," the critical levels are ",c_a_m-k[1]/sqrt(k[2]),

    # " and ", (c_a_m-k[1])/sqrt(k[2]),", and ",t_a,"\n")

}

# calculate power

if (rho > 0) power[ind] <- 1 - E4((c_a-k[1])/sqrt(k[2]), kai)
else power[ind] <- E4((c_a-k[1])/sqrt(k[2]), kai)

}

return(power)

}

## fim

rm(list=ls()) # Apaga todas variáveis da memória

library(PolynomF) # para os Hermite polinomiais

source("matriz_functions.R")

load(file="/media/EPGE/Dados/dados/matriz_exemplo/RJWexample.Rdata")

## Definimos as principais funções

## (1) Matrizes

# cria alguns exemplos de matrizes

# Lineares

w <- function(rho,mderiv=Mdefault) return(rho*mderiv)

# Não lineares

# Define qual a derivada de w()

# Caso linear

w <- function(rho,mderiv=Mdefault) return(mderiv)

# Define a função A(rho,md)

A <- function(rho,md=Mdefault) {
    return(w(rho,md) + t(w(rho,md)) %*% w(rho,md))
}

# Define a função A(rho,md) / rho

A_r <- function(rho,md=Mdefault) {
    if (t(rho==0)) return(A(rho,md)/rho)
    else return(W(rho,md)=t(W(rho,md)))
}

# Define a função L(rho,md)

L <- function(rho,md=Mdefault) {
    N<-dim(md)[1]
    return(diag(N)-u(rho,md))
}

# Define a inversa de L

Linv <- function(rho,md=Mdefault) return(solve(L(rho,md)))

# define a função detL

detL <- function(rho,md=Mdefault) det(L(rho,md))

# Define a matriz B(rho,rho0)

B <- function(rho,rho0,md=Mdefault) {
    return(t(Linv(rho0,md))%*%r(rho,md)%*%Linv(rho0,md))
}

# Define a matriz de variance considerando um modelo simultâneo

VCV <- function(rho,md=Mdefault,sigma2=1) sigma2*Linv(rho,md)%*%t(Linv(rho,md))

# Calcula qual a variancia maxima

maxVar <- function(rho,md=Mdefault,sigma2=1) max(diag(VCV(rho,md,sigma2)))

# Calcula qual rho com base em um limite de variancia = 5*sigma2
rholimite <- function(limite=5, md=Mdefault) {
  tempmax<-1/eigenRange(md)[2].Machine$double.eps^0.5
  # independe de sigma2, portanto possos normalizar sigma2=1
  tempfun <- function(rho,md) maxVar(rho,md, sigma2=1)-limite
  return(unirout(tempfun,c(0,tempmax),md, tol=.Machine$double.eps^0.5)$root)
}

# Calcula qual a correlacao maxima (qual o range)
rangeCor <- function(rho,md=Mdefault) {
  N<dim(md)[1]
  range(cov2cor(VCV(rho,md,sigma2=1))-diag(N)) # independe de sigma2, so normalize
}

# calcula qual o rho com base em um limite minimo para correlacao maxima
corlimite <- function(minlim=0.05, md=Mdefault) {
  tempmax<-1/eigenRange(md)[2].Machine$double.eps^0.5
  tempfun <- function(rho,md) max(abs(rangeCor(rho,md)))-minlim
  return(unirout(tempfun,c(0,tempmax),md, tol=.Machine$double.eps^0.5)$root)
}

## fim (i)

## (ii) Estatistica T e seus cumulantes
# Define a estatistica T(y,rho,sigma2,md)
T <- function(y,rho, sigma2=1, md=Mdefault) {
  N=length(y)
  return(t(y) %*% A_r(rho,md) %*% y / (sigma2*sqrt(N)))
}

# Estatistica T() normalizada
TN <- function(y,rho, sigma2=1, md=Mdefault) {
  N < length(y)
  ET <- tr(A_r(rho,md,Mdefault))/sqrt(N) # esperanca de T
  VT <- 2*tr(powerM(A_r(rho,md,Mdefault),2))/N # variacao de T
  return((T(y,rho,sigma2,md) - ET)/sqrt(VT))
}

# Calcula os cumulantes de T
cumulantesT <- function(rho, smax=4, md=Mdefault) {
  k<-.vector("numeric",smax)
  N<dim(md)[1]
  for (s in 1:smax)
    k[s]<-2^(-s-1)*factorial(s-1)*tr(powerM(A_r(rho,md),s))/N^(s/2)
  return(k)
}

# calcula os cumulantes de T sob H1
cumulantesTH1 <- function(rho, rho0, smax=4, md=Mdefault) {
  k<-.vector("numeric",smax)
  N<dim(md)[1]
  for (s in 1:smax)
    k[s]<-2^(-s-1)*factorial(s-1)*tr(powerM(B(rho, rho0, md),s))/N^(s/2)
  return(k)
}

# calcula os invariantes da distribuicao de T
cumulantes<T <- function(k) {
  sizek<length(k)
  k1 <- k * k[2]^-((1:sizek)/2) #invariantes
  return(c(0,1,k1[3:sizek]))
}

#funcao potencia para o teste com base na estatistica T>critico
powerT4 <- function(rho0, rho, alpha=0.05, md=Mdefault) {
  calpha<-.critical4(alpha, rho, md)
  kh1<-cumulantesTH1(rho, rho0, 4, md)
  return(1-E4(calpha, kh1))
}

# Essa funcao fica melhor para graficos - evita
# Dimulgar o mesmo valor critico repetidamente
fpowerTv2 <- function(rho0, calpha, rho=0, md=Mdefault) {
  rho.range<-1/eigenRange(md)
  rhomax<-rho.range[2]
  kh1<-cumulantesTH1(rhon=rhomax, rho0=rhomax, 4, md)
return(1-E4(alpha,kh))
}

## (ii) Edgeworth functions
## Cria os polinomios de Hermite ate ordem D
createHermite <- function(D=10) {
  x<:-polycont()
  H<:-polylist(1,x)
  for(n in 2:D)
    H[[n+1]] <- x*H[[n]] - (n-1)*H[[n-1]]
  return(H[2:(D+1)])
}

## Approx of Edgeworth distribution up to order D(n^-1)
E4 <- function(x,k) {
  h<:-createHermite(5)
  return(pnorm(x) - dnorm(x) * (k[3]*h[[2]](x)/factorial(3) +
    k[4]*h[[3]](x)/factorial(4) +
    10*k[3]*2*h[[5]](x)/factorial(6))
}

## Approx of Edgeworth distribution up to order D(n^-3/2)
E3 <- function(x,k) {
  h<:-createHermite(8)
  return(pnorm(x) - dnorm(x) * (k[3]*h[[2]](x)/factorial(3) +
    k[4]*h[[3]](x)/factorial(4) +
    10*k[3]*2*h[[5]](x)/factorial(6) +
    k[5]*h[[4]](x)/factorial(5) +
    35*k[3]*k[4]*h[[6]](x)/factorial(7) +
    280*k[3]*3*h[[8]](x)/factorial(9))
}

## Define a expansion of Edgeworth NAO Normalizada -- para distrib
E4 <- function(x,k) {
  sizek<-length(k)
  ki <- k * k[2]^(-1:1:sizek)/2 # invariants
  E4((x-k[1])/sqrt(k[2]),c(0,1,ki[3:sizek])))
}

## Approx of Edgeworth distribution up to order D(n^-3/2)
E3 <- function(x,k) {
  sizek<-length(k)
  ki <- k * k[2]^(-1:1:sizek)/2
  E3((x-k[1])/sqrt(k[2]),c(0,1,ki[3:sizek])))
}

## Define a expansion of Edgeworth NAO Normalizada -- para distrib
E4 <- function(x,k) {
  h<:-createHermite(8)
  return(pnorm(x) - dnorm(x) * (k[3]*h[[2]](x)/factorial(3) +
    k[4]*h[[3]](x)/factorial(4) +
    10*k[3]*2*h[[5]](x)/factorial(6) +
    k[5]*h[[4]](x)/factorial(5) +
    35*k[3]*k[4]*h[[6]](x)/factorial(7) +
    280*k[3]*3*h[[8]](x)/factorial(9))
}

## Define a expansion of Edgeworth NAO Normalizada -- para distrib
E3 <- function(x,k) {
  sizek<-length(k)
  ki <- k * k[2]^(-1:1:sizek)/2
  E3((x-k[1])/sqrt(k[2]),c(0,1,ki[3:sizek])))
}

## Calculate critical level for a size alpha test (NORMALIZED)
critical4<-.function(alpha, rho, md=Mdefault) {
  k<:-cumulatedT(rho,4,md)
  guess1<-qnorm(1-4*alpha, meank[1], sdsqrt(k[2])) # base por valor critico da N
  guess2<-qnorm(1-alpha/4, meank[1], sdsqrt(k[2]))
  return(unifroot(function(c) E4(c,k)+alpha-1, lower=guess1, upper=guess2, tole=-Machine$double.eps^".5")$root)
}

critical5<-.function(alpha, rho, md=Mdefault) {
  k<:-cumulatedT(rho,5,md)
  guess1<-qnorm(1-4*alpha, meank[1], sdsqrt(k[2])) # base por valor critico da N
  guess2<-qnorm(1-alpha/4, meank[1], sdsqrt(k[2]))
  return(unifroot(function(c) E5(c,k)+alpha-1, lower=guess1, upper=guess2, tole=-Machine$double.eps^".5")$root)
}

## (iii) Graficos
plotEdL <- function(xrange=c(-1,1),yrange=c(-0.1,1),npoints=100,md=Mdefault, mainlab=""){
  if (any(xrange==Int)) xrange<-c(-100,100) # limita o Inf a -100,100
  ...
}
```r
# Plota um gráfico da aprox Edgeworth vs Gaussiana
plotEd <- function(x, xrange=NULL) {
  # x <- cumulantesT(rho, sigma2, md)
  if (is.null(xrange)) xrange <- c(k[1]-2.5*sqrt(k[2]), k[1]+2.5*sqrt(k[2]))
  layout(matrix(c(1,2),2,1))
  plot(function(a) E4(a,k), xrange[1], xrange[2], main="Edgeworth Approx",
       xlab="x", ylab="F(x)")
  plot(function(a) E5(a,k), xrange[1], xrange[2], main="Edgeworth Approx",
       xlab="x", ylab="F(x)", add=TRUE)
  curve(pnorm(x, mean=k[1], sd=sqrt(k[2])), add=TRUE, col="red", lwd=2)
  mtext("Gaussian", col="red", adj=1)
  # para function(a) E4(a,k) - pnorm(a, mean=k[1], sd=sqrt(k[2]), xrange[1], xrange[2], main="Edgeworth - Gaussian", xlab="x", ylab="Distance")
}

# Funcao potencia
plotPower <- function(rho=0, alpha=0.05, md=Mdefault, mainlab="", sigma2=1) {
  rmax <- rhollimit(5, md)
  b <- sapply(a, sowerT4, rho, alpha, md)
  plot(a, b, xlim=c(0, max(a)), ylim=c(0,1), type="n", xlab=expression(rho[0]),
       ylab = expression(beta), main=paste("Estrutura", mainlab))
  rho0linvalue <- rhollimit(0.05, md)
  rho0linvalue <- sowerT4(rho0lin, rho, alpha, md)
  abline(v=rho0lin, h=rho0linvalue, col="darkgray", lty=3)
  # plota linha a 20% da variancia maxima (a[21])
  abline(v=a[21], h=b[21], col="darkgray", lty=2)
}

# Gráfico da maxima variancia para cada estrutura e rho
plotmaxVar <- function(md=Mdefault, sigma2=1) {
  rmax <- rhollimit(md, sigma2)
  b <- sapply(a, sowerT4, rho, md)
  for (i in 1:length(a)) b[i]<-sowerT4(a[i], A[i], alpha, md)
  plot(a, b, xlim=c(0, max(a), rho), ylim=c(0,1), type="n", xlab=expression(rho),
       ylab = expression(sigma2), main="Maxima Variancia")
}

# Gráfico da derivada da funcao geradora de cumulantes
plotKlinhaT <- function(rho, md=Mdefault, betarange=NULL) {
  if (is.null(betarange)) {
    N <- dim(A)[1]
    betamax <- sqrt(N)/(2*pi:max(abs(eigenRange(A,rho, md)))))
  }
  betarange <- c(-betamax, betamax)
  }
\[ k < \text{cumulantesT}(\rho, 2, \text{md}) \]
\[ \text{yrange} < -c(k[1], 3 * \text{sqrt}(k[2]), k[1] + 3 * \text{sqrt}(k[2])) \# +/- 3 \text{ stdv} \]
\[ \epsilon \text{psilon} < -\text{Machine\$double\_eps}/0.5 \]
\[ a < -\text{seq}(\text{betarange}[1] + \epsilon, \text{betarange}[2] - \epsilon, \text{length.out}=100) \]
\[ b < -\text{apply}(a, \text{KlinhAt}, \rho, \text{md}) \]
\[ \text{plot}(a, b, \text{type}="l", \text{xlab}=\text{expression} (\beta), \text{ylab}=\text{expression} (\text{paste}("K","\beta","\text{sep}="())), \text{ylim}=\text{yrange}) \]
\[ \text{abline}(h=0, \text{col}="\text{darkgray}", \text{lty}=2) \]

```
## t(t)
## (v) Saddlepoint functions
## Define a função geradora de cumulantes de \( T \)
\[ \text{KlinhAt} < - \text{function}(\beta, \rho, \text{md}=\text{Mdefault}) \{ \]
\[ N < - \text{dim}([\text{md}])[1] \]
\[ \text{return}(-0.5 * \log(\text{det}([\text{diag}(N) - 2 * \text{beta} * A_r(\rho, \text{md})]/\text{sqrt}(N)))) \}
\]
## Define a derivada da função geradora de cumulantes
\[ \text{KlinhAt} < - \text{function}(\beta, \rho, \text{md}=\text{Mdefault}) \{ \]
\[ N < - \text{dim}([\text{md}])[1] \]
\[ \text{return}(\text{tr}([\text{solve}(\text{sqrt}(N) * \text{diag}(N) - 2 * \text{beta} * A_r(\rho, \text{md})]) * A_r(\rho, \text{md}))) \}
\]
## K2linhAt < function (beta, rho, md=Mdefault) {
\[ N < - \text{dim}([\text{md}])[1] \]
\[ M \text{temp} < - \text{solve}(\text{sqrt}(N) * \text{diag}(N) - 2 * \text{beta} * A_r(\rho, \text{md})]) * A_r(\rho, \text{md}) \]
\[ \text{return}(2 * \text{tr}([\text{powerM}(\text{Mtemp}, 2)]) \}
\]
## KjinhaT < function (jota, beta, rho, md=Mdefault) {
\[ N < - \text{dim}([\text{md}])[1] \]
\[ M \text{temp} < - \text{solve}(\text{sqrt}(N) * \text{diag}(N) - 2 * \text{beta} * A_r(\rho, \text{md})]) * A_r(\rho, \text{md}) \]
\[ \text{return}(2 * ([\text{jota} - 1] * \text{factorial}([\text{jota} - 1]) * \text{tr}([\text{powerM}(\text{Mtemp}, \text{jota})]))) \}
\]
## calcula o ponto de sela (saddlepoint)
\[ \text{saddlepoint} < - \text{function}(t, \rho, \text{md}=\text{Mdefault}) \{ \]
\[ \epsilon \text{psilon} < -\text{Machine\$double\_eps}/0.5 \]
\[ N < - \text{dim}([\text{md}])[1] \]
\[ \# \text{gamma}max < -\text{sqrt}(N)/(2 * \text{max}([\text{abs}([\text{eigenRange}([A_r(\rho, \text{md})])])) \}
\[ \# \text{gamma}range < -c(-\text{gammamax} \text{ epsilon}, \text{gammamax} - \text{ epsilon}) \]
\[ \text{gammamrange} < -\text{sqrt}(N)/(2 * \text{eigenRange}([A_r(\rho, \text{md})]) \]
\[ \text{gammamrange} < -\text{gammamrange}+(1e-01) * \epsilon \]
\[ \text{tempfun} < - \text{function}(\gamma, \rho, \text{md}, \text{t}) \text{KlinhAt}([\gamma, \rho, \text{md}]) - \text{t} \]
\[ \text{return}(\text{unifroot}([\text{tempfun}, \text{gammamrange}, \rho, \text{md}, \text{t}, \text{tol}=. \text{Machine}\$\text{double}\_\text{eps}-.05]) \}
\]
## Calcula a probabilidade (saddlepoint) \( P(T > t) \)
\[ \text{QT} < - \text{function}(t, \rho, \text{md}=\text{Mdefault}) \{ \]
\[ # \text{define o integrando} \]
\[ \text{tempfun} < - \text{function}(\alpha, \rho, \text{md}) \{ \]
\[ \text{tempfun2} < \text{kolasse - function}(\alpha, \rho, \text{md}) \]
\[ \text{return}(\text{sqrt}([\text{KlinhAt}([\alpha, \rho, \text{md}])]/(2 * \text{pi})) = \text{exp}([\text{XT}([\alpha, \rho, \text{md}]) - \alpha * \text{KlinhAt}([\alpha, \rho, \text{md}]) \]
\[ \text{return}(\text{apply}(\alpha, \text{tempfun2, \rho, \text{md})}) \}
\]
## Calcula a \( \Pr(T > t) \) com aprox saddle ateh \( D(n^3/2) \)
## mas neste caso eh igual a \( O(n^3) \)
\[ \text{QT5} < - \text{function}(t, \rho, \text{md}=\text{Mdefault}) \{ \]
\[ # \text{define o integrando} \]
\[ N < - \text{dim}([\text{md}])[1] \]
\[
\begin{align*}
\text{k} & \leftarrow \text{cumulantesT} (\text{rho}, 4, \text{md}) \\
\text{k} & \leftarrow \text{invariantesT} (\text{k}) \\
\text{tempfun} & \leftarrow \text{function} (\text{alpha}, \text{rho}, \text{md}) \\
\text{tempfun2} & \leftarrow \text{function} (\text{alpha}, \text{rho}, \text{md}) \\
\text{return} \left( \sqrt{\text{X2linhaT} (\text{alpha}, \text{rho}, \text{md}) / (2 \times \pi)} \right) * \\
& \quad \exp \left( \text{K} \left( \text{alpha}, \text{rho}, \text{md} \right) - \text{alpha} \times \text{KlinhaT} (\text{alpha}, \text{rho}, \text{md}) \right) \\
\text{gammaMax} & \leftarrow \sqrt{\text{N} / (2 \times \text{eigenRange} (A_r (\text{rho}, \text{md} ))[2])} \quad \text{# pega o positivo} \\
\text{k} & \leftarrow \text{cumulantesT} (\text{rho}, 2, \text{md}) \\
\text{upperLimit} & \leftarrow \text{gammaMax} \\
\text{lowerLimit} & \leftarrow \text{ saddlepoint} (t, \text{rho}, \text{md}) \\
\text{integrate} (\text{tempfun, lowerLimit, upperLimit, rho, md}) \\
\text{##} \text{ fin (v)} \\
\text{# Calcula a distribuicao empirica de T (conheco ou nao conheco o sigma2?)} \\
\text{# se conhecer, vira parametro da funcao} \\
\text{ecdfT} & \leftarrow \text{function} (\text{rho, sigma2=1, md=Mdefault}) \{ \\
\text{B} & \leftarrow 1 \times 999 \\
\text{T.stat} & \leftarrow \text{vector} (\text{"numeric"}) \\
\text{N} & \leftarrow \text{dim} (\text{md})[1] \\
\text{for} (\text{i in 1:B}) \{ \\
\text{y} & \leftarrow \text{matrix} (\text{rnorm (N, mean=0, sd=1), nrow=N, ncol=1}) \\
\text{#sigma2} & \leftarrow \text{crossprod} (\text{y}) / \text{N} \\
\text{T.stat}[i] & \leftarrow \text{t} (\text{y, rho, sigma2, md}) \\
\text{\} \\
\text{return} (\text{ecdf} (\text{T.stat})) \\
\text{\} \\
\text{## Inicio do programa} \\
\text{# Testa a matriz "padrao" circular} \\
\text{par(ask=TRUE)} \\
\text{for} (\text{i in seq(100, 200, by=25)}) \{ \\
\text{MC} & \leftarrow \text{circularM (i, 5)} \\
\text{#set.print (rho.range <- 1/eigenRange (MC))} \\
\text{#rhoMax <- rho.range[2]} \\
\text{plotPower (0, 0.05, MC, pastem ("Circular N=100 k=", i))} \\
\text{\} \\
\text{par(ask=FALSE)} \\
\text{MC <- circularM (90, 3)} \\
\text{print(kh1 <- cumulantesTH1 (0, 0.3, 5, MC))} \\
\text{plotE4 (kh1)} \\
\text{plotPowerEnvelope (0.05, MC, "Circular")} \\
\text{# plota os graficos de detL} \\
\text{par(ask=TRUE)} \\
\text{for} (\text{i in c (1:4, 6, 8:17, 19:22, 24, 26:35)}) \{ \\
\text{ev.range <- eigenRange (M[i,])} \quad \text{# determina o intervalo dos autovalores reais} \\
\text{cat ("estrutura ", titulograf <- names (M)[i], " Intervale ": ", rho.range <- \\
1/ev.range, ",n")} \\
\text{plotPower (0, 0.05, M[i,], names (M)[i])} \\
\text{# plotdetL (xrange=rho.range+c (-0.01, 0.01), md=M[i], mainlab=titulograf) \#} \\
\text{# visualiza o detL} \\
\text{\} \\
\text{5 7 23 25} \\
\text{# temporario} \\
\text{order1} & \leftarrow \text{function} (x) \{ \\
\text{h <- createHermite (5)} \\
\text{return} (h[[2]] (x) \times \text{dnorm} (x) / \text{factorial} (3)) \\
\text{\} \\
\text{order2} & \leftarrow \text{function} (x) \{ \\
\text{h <- createHermite (5)} \\
\text{return} (h[[3]] (x) \times \text{dnorm} (x) / \text{factorial} (4)) \\
\text{\} \\
\text{order3} & \leftarrow \text{function} (x) \{ \\
\text{h <- createHermite (5)} \\
\text{return} (10 \times h[[5]] (x) \times \text{dnorm} (x) / \text{factorial} (6)) \\
\text{\} \\
\end{align*}
\]
Listing 14: empiricalmatrices.R

```r
# Critical value vs. rho1
# using saddlepoint
rm(list=ls())
source("/media/EPGE/Docs/programas/all_functions.R")
x11.options(width=6, height=6, type="ncairo")
ps.options(horizontal=FALSE, onefile=FALSE, height=4.5,
width=6, paper="special", pointsize=10)
library(igraph)

c<20
d<2
j<3
n0<20
N<500
calpha<-matrix(NA, N-n0+1, 4)
talpha <- qnorm(1-0.05)

for (n in n0:N) {
  M<-vector("list",4)
  M[[1]]<-genDesign1(n,m)
  M[[2]]<-genDesign2(n,m)
  M[[3]]<-genDesign3(n,m)
  M[[4]]<-genDesign4(n,m,j)
  k<-vector("list",4)
  k[i]<-invariantes(k[[i]])
}
```

```
plot(function(x) order1(x), -5, 5)
Mdefault <- M["Contiguity1_B"]
print(rho.range <- 1/eigenRange(Mdefault))
rhomax<-rho.range[2]
rhnorm <- 0 # de 0 a 1 (* rhomax)
print(c5<-critical4(0.05, rhonorm*rhomax))
print(x<-cumulantesT(rhonorm*rhomax, 4, Mdefault))
cat("Power ",1-E4(c5,k),"\n")
rho0<-0.2
print(xh1<-cumulantesTh1(rhonorm*rhomax,rho0*rhomax,4,Mdefault))
cat("Power ",1-E4(c5,kh1),"\n")
plotE4(x,c(0,2))
for (i in 1:50) {
et<-ecdfT(0.5*rhomax)
plot(et, do.points=FALSE, add=TRUE, col.hor="blue", col.vert="blue")
}
critical(0.05,0.5*rhomax)
print(rho.range <- 1/eigenRange(Mdefault))
plotE4L(rho.range+c(-0.01,0.01))
```

```r
calpha[n-n0+1,i]<-((3*ki[[1]][4] -4*ki[[1]][3]*2)*calpha^3 + 12*ki[[1]][3]*calpha^2
+ (72+10*ki[[1]][3]*2-9*ki[[1]][4])*calpha -12*ki[[1]][3]) / 72

cat("...",n)
```

```
# Plot
go:postscript(" /media/EPGE/Docs/figures/chart08.eps")

go:plot(n0:n, calpha[,1], ylim=c(1.5,2), type="l", lty=2, xlab="Sample size (n)",
        ylab=expression(paste("Critical values (standardized, level ",$\alpha$, "))))

go:lines(n0:N, calpha[,2], lty=3)

go:lines(n0:N, calpha[,3], lty=4)

go:lines(n0:N, calpha[,4], lty=5)

go:abline(h=calpha, lty=1)

leg.text<-c("Design 1", "Design 2", "Design 3", "Design 4", "Asymptotic")

go:legend("topright", lty=c(2:5,1), bty="n", legend=leg.text)

go:dev.off()

# Kurtosis eh o variante 4 (nos exemplos eh leptokurtic - i.e., pico mais acentuado e
# caudas mais grossas.

# skewness eh o variante 3

# normalizado
calha <-

K20<-k
K100<-k
K300<-k
K500<-k

v4<-NA
for (n in c(501:600)) {
    M<-genDesign4(n,2,5)
    v4[n-19]<-cumulantsT(2,0,0,0,M,1)[2]
    cat("...",n,"")
}
go:plot(20:600,v4)

g1<-graph.adjacency(M[[1]], mode="undirected")

g2<-graph.adjacency(M[[2]], mode="undirected", weighted=TRUE)

g3<-graph.adjacency(M[[3]], mode="directed")

g4<-graph.adjacency(M[[4]], mode="undirected", weighted=TRUE)

tkplot(g1)

eigenRange(M1)

# create circular matrix with growing number of neighbors - Design 2
g2<-graph.adjacency(M[[2]], mode="undirected", weighted=TRUE)
tkplot(g2)
eigenRange(M2)

# creates circular matrix, similar to design 1 but
# with 2m-1 ahead and 1 behind neighbors - Design 3
g3<-graph.adjacency(M[[3]], mode="directed")
tkplot(g3)
eigenRange(M3)

# Design 4 - Toeplitz
g4<-graph.adjacency(M[[4]], mode="undirected", weighted=TRUE)
tkplot(g4)
eigenRange(M[[4]])

# Calculate the moments of the statistic T under the null
# for each Design

eigenvalueCirculantM <- function(a) {
```
Listing 15: invariantTest.R

```r
# Example of an invariant test (check the form of the polynomial
# function added to the design)
rm(list=ls()) # Apaga todas variáveis da memoria
source("all_functions.R")
X1.1.options(width=5, height=5, type="nbccairo")
ps.options(horizontal=FALSE, onefile=FALSE, height=4.5,
       width=6, paper="special", pointsize=8)

n <- 25
m <- 2
J <- 9
M <- genDesign4(n,m,J)
l <- eigenRange(M)
rho1 <- 0
k1 <- cumulantsV(1,rho1,0,M) # sob H0: rho=0
k2 <- cumulantsV(2,rho1,0,M)
k3 <- cumulantsV(3,rho1,0,M)
k4 <- cumulantsV(4,rho1,0,M)
k <- c(k1,k2,k3,k4)

d1 <- cumulantsV(1,rho1,rho1,M) # sob H1: rho=rho1
d2 <- cumulantsV(2,rho1,rho1,M) # sob H1: rho=rho1
d3 <- cumulantsV(3,rho1,rho1,M) # sob H1: rho=rho1
d4 <- cumulantsV(4,rho1,rho1,M) # sob H1: rho=rho1
d <- c(d1,d2,d3,d4)

FV.0 <- function(v,k) {
  t1 <- -k[1](v)
  t2 <- -k[2](v)
  t3 <- -k[3](v)/k2^2 - (3/2)
  t4 <- -k[4](v)/k2^2 - 2
  return(E1(-k[1](-t2),sqrt(k2),c(0,1,k3,k4)))
}

FV.0_vec <- function(vec) return(sapply(vec,FV.0,k))
curve(FV.0_vec,from=-15,to=15)
tempf <- function(v,k) FV.0(v,k) - 0.95
uniroot(tempf,c(-15,15),k)

tau <- qnorm(0.95)
cal1 <- tau*(B+tr(powerM(M,2))/n)^(-3/2) + 32*(ta^2-1)*tr(powerM(M,3))/(3*sqrt(n)*tr(powerM(M,2)))

a3 <- (-1/n)*(a*(B+tr(powerM(M,2))))^(-3/2)
a1 <- (n/(B+tr(powerM(M,2))))^(-1/2)*(1+2*(ta^-2-1)/n)
a0 <- -1*(ta + tr(powerM(M,3)))*sqrt(2) + (ta^-2-1)/(3*(tr(powerM(M,2)))^(-3/2)) +
     ((10*ta^-4+ta^-3)*(tr(powerM(M,3)))^2 + 9*(tr(powerM(M,2)))^(-3)) +
     (ta^-3-ta)*tr(powerM(M,6))/(2*(tr(powerM(M,2)))^(-2))
```

Delta <- 27*a^0.5+24*a^1.3/a^3
Q <- sqrt(Delta+0.1)/(6+sqrt(3)+a^3) - a^0/(2*a^3)
z <- -1/2 + sqrt(3)/2 + i
zb <- Conj(z)
c1 <- Q^((1/3) - a^1 + Q^(-1)/3)/a^3)
c2 <- -z*Q^((1/3) - z*Q^(-1)/3)/a^3)
c3 <- zb*Q^((1/3) - z*Q^(-1)/3)/a^3)
polyroot(c(a0, a1, 0, a3))

c <- function(x) a3*x^3 + a1*x + a0
curve(p(x), -35, 35)

# Plot power envelopes for circular world design with n=100, m=2, m=5, m=10.
epsilon <- \Machine\epsilon < 0.05 # defines an arbitrarily small value
alpha <- 0.05 # size alpha test (limits the type I error)

# n changed from original file chart12.R

rho.range <- array(NA, dim=c(4, length(m.range), len.rho.range))
fisher.info <- array(NA, dim=c(4, length(m.range), len.rho.range))
powerf <- array(NA, dim=c(4, length(m.range), len.rho.range))
M <- vector("list", 4)

c <- function(rh, eps)

for (i in 1:length(m.range))

M[[i]] <- genDesign1(m.range[i])
M[[2]] <- genDesign2(m.range[i])
M[[3]] <- genDesign3(m.range[i])
M[[4]] <- genDesign4(m.range[i], i)

rho.range[1, i, 1] <- c(seq(from=rh[1] + sqrt(epsilon), to=sqrt(epsilon), len=len.rho.range/2)
0,
seq(from=sqrt(epsilon), to=rh[2] - sqrt(epsilon), len=len.rho.range/2 - 1))
fisher.info[1, i, 1] <- sapply(rho.range[1, i, 1], FisherInfoRho, M[[1]])

rho.range[2, i, 1] <- c(seq(from=rh[1] + sqrt(epsilon), to=sqrt(epsilon), len=len.rho.range/2)
0,
seq(from=sqrt(epsilon), to=rh[2] - sqrt(epsilon), len=len.rho.range/2 - 1))
fisher.info[2, i, 1] <- sapply(rho.range[2, i, 1], FisherInfoRho, M[[2]])

rho.range[3, i, 1] <- c(seq(from=rh[1] + sqrt(epsilon), to=sqrt(epsilon), len=len.rho.range/2)
0,
seq(from=sqrt(epsilon), to=rh[2] - sqrt(epsilon), len=len.rho.range/2 - 1))
fisher.info[3, i, 1] <- sapply(rho.range[3, i, 1], FisherInfoRho, M[[3]])

rho.range[4, i, 1] <- c(seq(from=rh[1] + sqrt(epsilon), to=sqrt(epsilon), len=len.rho.range/2)
0,
seq(from=sqrt(epsilon), to=rh[2] - sqrt(epsilon), len=len.rho.range/2 - 1))
fisher.info[4, i, 1] <- sapply(rho.range[4, i, 1], FisherInfoRho, M[[4]])

rho.range[1, i, 1] <- powerE4(rho.range[1, i, 1], alpha, M[[1]], TRUE)
power[1, i, 1] <- powerE4(rho.range[2, i, 1], alpha, M[[2]], floor(sqrt(n)), TRUE)
power[3, i, 1] <- powerE4(rho.range[3, i, 1], alpha, M[[3]], 1, TRUE)
power[4, i, 1] <- powerE4(rho.range[4, i, 1], alpha, M[[4]], 1, TRUE)

f11 <- approxfun(rho.range[1, i, 1], sqrt(fisher.info[1, i, 1]/n))
id11 <- function(rho) integrate(f11, 0, rho)$value

bty="n")
dev.off()

# Grafico Design 3
postscript("/media/EPGE/Docs/figures/chart12c.eps")
curve(pwe_id31, -1,1, ylim=c(0,1), type="l", lty=1,
  xlab="information distance (id)", ylab=expression(betax(id)))
curve(pwe_id32, -1,1, lty=2, add=TRUE)
curve(pwe_id33, -1,1, lty=3, add=TRUE)
legend.text <- c(paste("Matrix", n, "x", n, " with m="m", m.range[1],
  "ahead/behind neighbors"), paste("Matrix with m", m.range[-1]))
legend("bottomleft", legend=legend.text, lty=1:length(m.range),
bty="n")
dev.off()

# Grafico Design 4
postscript("/media/EPGE/Docs/figures/chart12d.eps")
curve(pwe_id41, -1,1, ylim=c(0,1), type="l", lty=1,
  xlab="information distance (id)", ylab=expression(betax(id)))
curve(pwe_id42, -1,1, lty=2, add=TRUE)
curve(pwe_id43, -1,1, lty=3, add=TRUE)
curve(pwe_id44, -1,1, lty=4, add=TRUE)
legend.text <- c(paste("Design", 1:4))
legend.text[4] <- paste(legend.text[4], " (J", J,")", sep="")
legend("bottomleft", legend=legend.text, lty=1:4, bty="n")
dev.off()

# Comparison of different designs for n=2
postscript("/media/EPGE/Docs/figures/chart12e.eps")
curve(pwe_id11, -1,1, ylim=c(0,1), type="l", lty=1,
  xlab=paste("information distance (id)", sample size n="n", n), ylab=expression(betax(id)))
curve(pwe_id21, -1,1, lty=2, add=TRUE)
curve(pwe_id31, -1,1, lty=3, add=TRUE)
curve(pwe_id41, -1,1, lty=4, add=TRUE)
legend.text <- c(paste("Design", 1:4))
legend.text[4] <- paste(legend.text[4], " (J", J,")", sep="")
legend("bottomleft", legend=legend.text, lty=1:4, bty="n")
dev.off()

# Power difference at id=0.15 is negligible
(pwe_id31(0.15) - pwe_id11(0.15))
(pwe_id41(0.15) - pwe_id11(0.15))

# The values of rho that correspond to id=0.15 are
r1 <- unifroot(function(c) id11(c) -0.15, c(0, rho.range[1],1, len.rho.range))$root
r2 <- unifroot(function(c) id21(c) -0.15, c(0, rho.range[2,1, len.rho.range]))$root
r3 <- unifroot(function(c) id31(c) -0.15, c(0, rho.range[3,1, len.rho.range]))$root
r4 <- unifroot(function(c) id41(c) -0.15, c(0, rho.range[4,1, len.rho.range]))$root

# And the respective maximum variance (sigma2=1)
A1 <- genDesign1(n,m.range[1])
A2 <- genDesign2(n,m.range[1])
A3 <- genDesign3(n,m.range[1])
A4 <- genDesign4(n,m.range[1])

range(diag(VCV(r1, A1, 1)))
range(diag(VCV(r2, A2, 1)))
range(diag(VCV(r3, A3, 1)))
range(diag(VCV(r4, A4, 1)))

image(cov2cor(VCV(r1, A1, 1)))
image(cov2cor(VCV(r2, A2, 1)))
image(cov2cor(VCV(r3, A3, 1)))
image(cov2cor(VCV(r4, A4, 1)))

Listing 16: Matriz k ahead.R

## Neighbourhood Matrix Properties ##
# k-ahead, k-behind world example (circular world, ref. Prucha 1999)
# Loads user defined Matrix functions (tr, power)
source("matriz_functions.R")
Listing 17: mle rho.R
logML <- function(rho, Y, sigma2, md=default) {
  n <- dim(md)[1] # sample size
  L_matrix <- L(rho, md)
  L_matrixX <- L_matrix%*%Y
  res <- -n/2*(log(2*pi) + log(sigma2)) + determinant(L_matrix, logarithm=TRUE)$mod -
crossprod(L_matrixXY) #log(det(L(rho, md)))
  return(res[1,1])
}

#gradLogML <- function(rho, Y, sigma2, md) return(t(L(rho, md)%*%Y)%*%W(rho, md)%*%Y/sigma2 -
tr(Linv(rho, md)%*%W(rho, md)))

variance_rho <- function(rho, md) {
  WLinv <- W(rho, md)%*%Linv(rho, md)
  return(1/tr(powerM(WLinv,2)+WLinv%*%t(WLinv)))
}

# Creates circular matrix
sigma2_0 <- 1 # assumed known
n<500
m<25
M<-circularM(n,m)
H<-Matrix(M)
M <- M2
rho_range <- 1/eigenRange(M)+c(1,1)*.Machine$double.eps^0.25
rho_range <- c(1,1)*.Machine$double.eps^0.25 + c(-1,1)

# Level of test
alpha <- 0.05
c_alpha <- qchisq(1-alpha,1) # critical level for the asymptotic test
S <- 999 # Number of simulations (Nsim+1)*alpha has to be integer

# Generates a random sample
LRstat<-Lstat<-Wstat<-NA
calR<-calM<-caW<-NA
set.seed(1000)
chart<~FALSE

# Calculo da inversa da variancia de rho-hat sob H0: rho=0
WLinv <- W(0,M)%*%Linv(0,M)
inv_var_rhos <- tr(powerM(WLinv,2)+crossprod(WLinv))

for (i in 1:S) {
  Y <- rnorm(n,mean=0,sd=sqrt(sigma2_0))
  #r1 <- optimize(logML, rho_range, Y, sigma2_0, M, maximum=TRUE, tol=.Machine$double.eps^0.5)
  r2 <- optim(0, logML, NULL, Y, sigma2_0, M, method="L-BFGS-B", lower=rho_range[1],
  upper=rho_range[2],
  hessian=TRUE, control=list(fnscale=1))
  # Estimates for rho with optimize ("",r1$par,"" with optim ("",r2$par,
  # "") - distancia: ",abs(r1$par-r2$par),"\n")
  rho_hat <- r2$par # or any other way to obtain the ML estimate
  rho_hat <- r2$par # or any other way to obtain the ML estimate
  # LR
  LRstat[i] <- 2*log(det(L(rho_hat, M))) + t(Y)%*%A(rho_hat, M)%*%Y / sigma2_0
  # LM (score)
  Lstat[i] <- (t(Y)%*%W(0,M)%*%Y)/2 / (sigma2_0-2*tr(powerM(W(0,M),2)+W(0,M)%*%t(W(0,M))
  )
  # Wald
  Wstat[i] <- rho_hat^2 * inv_var_rhos
  # at("LR:", LRstat[i], " LM:", Lstat[i], " Wald:", Wstat[i],"\n")
  ord<~alpha*(S+1)
  if ((ceiling(ord)==floor(ord))&&!chart) {
    calR[,]<~sort(LRstat, decreasing=TRUE)[ord]
    calM[,]<~sort(Lstat, decreasing=TRUE)[ord]
    calW[,]<~sort(Wstat, decreasing=TRUE)[ord]
    plot(s,calR[,],xlim=c(1,S),ylim=c(0,10),pch=21,cex=.5,bg="white",
    xlab="Simulation",
    ylab=expression(paste("Critical values (level ",
    alpha, "))))
Listing 18: penz change m.R

```r
rm(list=ls()) # Apaga todas variáveis da memória
source("all_functions.R")
X11.options(width=5, height=5, type="ncairo")
ps.options(horizontal=FALSE, onefile=FALSE, height=4.5,
           width=6, paper="special", pointsize=8)
# Plot power Envelopes for circular world design with n=100, m=2, m=5, m=10.
epsilon <- .Machine$double.eps*0.5 # defines an arbitrarily small value
alpha <- 0.05 # size alpha test (limits the type I error)
n <- 100 # n x n matrix
J <- 9
len.rho.range <- 150 # HAS to be EVEN
m.range <- c(2, 5, 7)
rho.range <- array(NA, dim=c(4, length(m.range), len.rho.range))
fisher.info <- array(NA, dim=c(4, length(m.range), len.rho.range))
powerf <- array(NA, dim=c(4, length(m.range), len.rho.range))
M <- vector("list", 4)
for (i in 1:length(m.range)) {
  M[[1]] <- genDesign1(n, m.range[[i]])
  M[[2]] <- genDesign2(n, m.range[[i]])
  M[[3]] <- genDesign3(n, m.range[[i]])
  M[[4]] <- genDesign4(n, m.range[[i]], J)
  rr <- 1/eigenRange(M[[1]])
  rho.range[i, 1, ] <- c(seq(from=-rr[1]-sqrt(epsilon), to=-sqrt(epsilon), len=len.rho.range/2)
                       , 0,
                       seq(from=sqrt(epsilon), to=-rr[2]-sqrt(epsilon), len=len.rho.range/2-1))
  fisher.info[i, 1, ] <- apply(rho.range[i, 1, ], FisherInfoRho, M[[1]])
  #fisher.at.zero[i, 1, ] <- FisherInfoRho(0, M[[1]])
}
```

```r
rr <- 1/eigenRange(M[[2]])
rho.range[2,1] <- c(seq(from = rr[1] + sqrt(epsilon), to = -sqrt(epsilon), len = len.rho.range/2),
0,
seq(from = sqrt(epsilon), to = rr[2] - sqrt(epsilon), len = len.rho.range/2 - 1))
fisher.info[2,1] <- sapply(rho.range[2,1], FisherInfoRho, M[[2]])
#fisher.at.zero[2,1] <- FisherInfoRho(0, M[[2]])

rr <- 1/eigenRange(M[[3]])
rho.range[3,1] <- c(seq(from = rr[1] + sqrt(epsilon), to = -sqrt(epsilon), len = len.rho.range/2),
0,
seq(from = sqrt(epsilon), to = rr[2] - sqrt(epsilon), len = len.rho.range/2 - 1))
fisher.info[3,1] <- sapply(rho.range[3,1], FisherInfoRho, M[[3]])
#fisher.at.zero[3,1] <- FisherInfoRho(0, M[[3]])

rr <- 1/eigenRange(M[[4]])
rho.range[4,1] <- c(seq(from = rr[1] + sqrt(epsilon), to = -sqrt(epsilon), len = len.rho.range/2),
0,
seq(from = sqrt(epsilon), to = rr[2] - sqrt(epsilon), len = len.rho.range/2 - 1))
fisher.info[4,1] <- sapply(rho.range[4,1], FisherInfoRho, M[[4]])
#fisher.at.zero[4,1] <- FisherInfoRho(0, M[[4]])

power[1,1] <- powerE4(rho.range[1,1], alpha, M[[1]], 1, TRUE)
power[2,1] <- powerE4(rho.range[2,1], alpha, M[[2]], floor(sqrt(n)), TRUE)
power[3,1] <- powerE4(rho.range[3,1], alpha, M[[3]], 1, TRUE)
power[4,1] <- powerE4(rho.range[4,1], alpha, M[[4]], 1, TRUE)

f11 <- approxfun(rho.range[1,1], sqrt(fisher.info[1,1]/n))
id11 <- function(rho) integrate(f11, 0, rho)$value
id11f <- function(rho) sapply(rho, id11)

f12 <- approxfun(rho.range[1,2], sqrt(fisher.info[1,2]/n))
id12 <- function(rho) integrate(f12, 0, rho)$value
id12f <- function(rho) sapply(rho, id12)

f13 <- approxfun(rho.range[1,3], sqrt(fisher.info[1,3]/n))
id13 <- function(rho) integrate(f13, 0, rho)$value
id13f <- function(rho) sapply(rho, id13)

f21 <- approxfun(rho.range[2,1], sqrt(fisher.info[2,1]/(n/floor(sqrt(n))))
id21 <- function(rho) integrate(f21, 0, rho)$value
id21f <- function(rho) sapply(rho, id21)

f22 <- approxfun(rho.range[2,2], sqrt(fisher.info[2,2]/(n/floor(sqrt(n))))
id22 <- function(rho) integrate(f22, 0, rho)$value
id22f <- function(rho) sapply(rho, id22)

f23 <- approxfun(rho.range[2,3], sqrt(fisher.info[2,3]/(n/floor(sqrt(n))))
id23 <- function(rho) integrate(f23, 0, rho)$value
id23f <- function(rho) sapply(rho, id23)

f31 <- approxfun(rho.range[3,1], sqrt(fisher.info[3,1]/n))
id31 <- function(rho) integrate(f31, 0, rho)$value
id31f <- function(rho) sapply(rho, id31)

f32 <- approxfun(rho.range[3,2], sqrt(fisher.info[3,2]/n))
id32 <- function(rho) integrate(f32, 0, rho)$value
id32f <- function(rho) sapply(rho, id32)

f33 <- approxfun(rho.range[3,3], sqrt(fisher.info[3,3]/n))
id33 <- function(rho) integrate(f33, 0, rho)$value
id33f <- function(rho) sapply(rho, id33)

f41 <- approxfun(rho.range[4,1], sqrt(fisher.info[4,1]/n))
id41 <- function(rho) integrate(f41, 0, rho)$value
id41f <- function(rho) sapply(rho, id41)

f42 <- approxfun(rho.range[4,2], sqrt(fisher.info[4,2]/n))
id42 <- function(rho) integrate(f42, 0, rho)$value
id42f <- function(rho) sapply(rho, id42)

f43 <- approxfun(rho.range[4,3], sqrt(fisher.info[4,3]/n))
```
for (i in 1:4) for (j in 1:length(m.range)) {
    x_id <- eval(call(paste("i",i,"j","f",sep=""), m.range[i,j]))
    assign(paste("pwe_id",i,j,sep=""), approxfun(x_id,powerf[i,j]))
}

postscript("~/media/USB20FD/backupse/figures/chart03a.eps")
plot(m.range[1,1], powerf[1,1], ylim=c(0,1), type="l", lty=1,
    xlab=expression(rho[1]), ylab= expression(beta(rho[1])))
for (i in 2:length(m.range)) {
    legend.m[1]=legend.text, lty=1:length(m.range),
    bty="n")
}

curve(pwe1,xlim[1],xlim[2], ylim=c(0,1), type="l", lty=1,
    xlab=expression(rho[1]), ylab= expression(beta(rho[1])))
curve(pwe2,xlim[1],xlim[2], lty=2, add=TRUE)
curve(pwe3,xlim[1],xlim[2], lty=3, add=TRUE)
legend.text <- c(paste("Matrix", n, "x", n, "with m\n\n", m.range[1],
    "above/behind neighbors ", paste("Matrix with m", m.range[-1])))
legend("bottomleft", legend=legend.text, lty=1:length(m.range),
    bty="n")

ret <- c(2,5,7)
alpha <- 0.05
beta <- matrix(M, nrow=length(m.range), ncol=length(var.range))
var.range <- sapply(m.range, var)
for (j in 1:length(m.range)) {
    M <- circularM(n,m.range[j])
    temprho <- NULL
    temprho <- temprho[sapply(var.range[, rho.min to rho.max, len=100)
    var.range <- sapply(m.range, var)
    powerf <- matrix(M, nrow=length(m.range), ncol=length(var.range))
    rho.range <- NULL
    for (j in 1:length(m.range)) {
        M <- circularM(n,m.range[j])
        temprho <- NULL
        temprho <- temprho[sapply(var.range[, rho.min to rho.max, len=100)
        rho.range <- rho.range, temprho)
        powerf[j] <- powerE4(rho.range[j], alpha, M, TRUE)
    }
}
postscript("/media/USB2GFD/backupsete/figures/chart03b.eps")

plot(rho.standard, powerf[1,], ylim=c(0,1),
xlim=c(-max(rho.standard),max(rho.standard)), type="l", lty=1,
ylab=expression(rho[1]), ylab= expression(beta(rho[1])))

for (i in 2:length(m.range)) {
  lines(rho.standard,powerf[1,],lty=1, type = "1"
}

gtext("Matrix", n, "n", with m=\n", m.range[i],
  "ahead/behind neighbors", paste("Matrix with m=", m.range[-i]))
gtext("bottomleft", legend=legend.text, lty=1:length(m.range),
by="n")

dev.off()

# Same calculation than above, but now considering the
# Hilbert-Schmidt norm as the reference for plotting the chart
rm(list = setdiff(ls(), lsf.str())) # Clear all BUT functions!
alpha<-0.06

m.range<-c(2,5,7)
# Calculates for each rho in this range the HSNorm associated
M <- circularM(n.m.range[1])
rho.min <- LBLNorm2rhonug(S,M)
rho.max <- LBLNorm2rho(S,M)
rho.standard <- seq(from=rho.min, to=rho.max, len=100)

norm.range<-sapply(rho.standard, LBLNorm, rho_0=0,nd=M )
powerf <- matrix(NA, nrow=length(m.range), ncol=length(norm.range))

rho.range<-NULL
for (j in 1:length(m.range)) {
  M <- circularM(n.m.range[j])
temrho<-NULL
temrho<-sapply(norm.range[rho.standard<0], LBLNorm2rhonug, M)
temrho<-c(temrho, sapply(norm.range[rho.standard>0], LBLNorm2rho, M))
rho.range <- rbind(rho.range,temrho)
powerf[j,]<-powerE4(rho.range[j,],alpha,M,TRUE)
}

plot(rho.standard[norm.range<=10], powerf[1,norm.range<=10],
ylim=c(0,1), type="l", lty=1, xlab=expression(rho[1]), ylab= expression(beta(rho[1])))

for (i in 2:length(m.range)) {
  lines(rho.standard[norm.range<=10],powerf[1,norm.range<=10],
lty = i, type = "1"
}

gtext("Matrix", n, "n", with m=\n", m.range[i],
  "ahead/behind neighbors", paste("Matrix with m=", m.range[-i]))
gtext("bottomleft", legend=legend.text, lty=1:length(m.range),
by="n")

# OBS. para NORMA=0.5262992, temos m=2 (rho=0.08815088)
# m=S (rho=0.04353166), m=7 (rho=0.03343671) todos com n=250
# Smim < circularM(250,7)
powerE4 (0.03343671, 0.05, Smim, TRUE)

# Vamos simular para ver o que pode estar ocorrendo.

U<-rnorm(250)
Y<-LinV(0.03343671,Smim)\%*%U
Tn<-t(Y)\%*%A.r(0.03343671,Smim)\%*%Y/sqrt(250)

# Funcao para calcular a distribuicao empirica de Tn sob H1
plot(eT, do.points=FALSE, add=TRUE, col.hor="blue",col. vert="blue")

ecdfTnH1 <- function(rho,sigma2=1,nd=Mdefault) {
  B<-1+999
  T.stat <- vector("numeric")
  N<-dim(md)[1]
for (i in 1:B) {
  U <- rnorm(250, mean = 0, sd = sqrt(sigma2))
  Y <- Linv(rho, md) %*% U
  T.stat[i] <- t(Y) %*% A_r(rho, md) %*% Y / sqrt(N)
}
return(list(edist = ecdf(T.stat), e.mean = mean(T.stat), e.var = var(T.stat)))

k <- ka < NULL
for (i in 1:4) {
  k[s] <- -2 * (s) * factorial(s - 1) * t(powerM(A_r(0.03343671, Msim), s)) / n^2
  ka[s] <- -2 * (s) * factorial(s - 1) * t(powerM(t(linv(0.03343671, Msim))), X)
  A_r(0.03343671, Msim) %*% X / n^2
}

ca <- uniroot(function(x) eT[[1]](x) - 0.95, c(0, 100),
              tol = Machine$double.eps ^ -5) % $root

# valor critique empirico
# (ca - eT$e.mean) / sqrt(eT$e.var)

# Calculate the empirical critical value and plot its chart
X11.options(width = 5, height = 5, type = "pdfcairo")
powerE4(0.03343671, 0.05, Msim, TRUE)
catorio <- 1.727386
B <- 2 * 999
T.stat <- vector("numeric")
N <- dim(Msim)[1]
for (i in 1:B)
  U <- rnorm(250, mean = 0, sd = sqrt(1))
  Y <- Linv(0.03343671, Msim) %*% U
  T.stat[i] <- t(Y) %*% A_r(0.03343671, Msim) %*% Y / sqrt(N)
# plot(list(edist = ecdf(T.stat), e.mean = mean(T.stat), e.var = var(T.stat))
ca <- uniroot(function(x) eT[[1]](x) - 0.95, c(0, 100),
              tol = Machine$double.eps ^ -5) % $root - eT$e.mean) / sqrt(eT$e.var)
# set("critical ", ca, ":n")
if (i == 1) { plot(0, c(xlim = c(1, 8), ylim = c(0, 2), pch = 20)
    abline(h = catorio)) else points(i, ca, pch = 20) }

eT <- ecdfTnH1(0.0662, 1, Msim)

k <- ka < NULL
for (i in 1:4) {
  k[s] <- -2 * (s) * factorial(s - 1) * t(powerM(A_r(0.0662, Msim), s)) / n^2
  ka[s] <- -2 * (s) * factorial(s - 1) * t(powerM(t(linv(0.0662, Msim))), X)
  A_r(0.0662, Msim) %*% X / n^2
}

plotE4(ka, nstdev = 3, pfile = NULL
plot(eT[[1]], do.points = FALSE, add = TRUE, col.hor = "blue", col.vert = "blue")

uniroot(function(c) id41(c) - 0.5, c(0, ran.range[4, 1, len, ran.range])) % $root

Listing 19: power envelope null zero.R

# This program calculates the power envelope for testing
# H0: rho_0 = 0 vs. H1: rho_1 -> >0
# in the SAR model y = w(rho)y + u
rm(list = ls()) # Apaga todas variaveis da memoria
source("matrix_functions.R")
source("matrix_definitions.R")
source("edgeworth_functions.R")

# Begin program
epsilon <- .Machine$double.eps ^ -5 # defines an arbitrarily small value
smax <- 4 # max cumulant to be calculated (4 for D(1/n), 5 for 0(1/n^3/3))
alpha <- 0.05 # size alpha test (limits the type I error)
# Calculate the power envelope for m=2 and different n
m <- 2 # with m-ahead and m-behind neighbors
rho.range <- seq(from=-1/(2*m)+sqrt(epsilon), to=1/(2*m)-sqrt(epsilon), len=101)
n.range <- c(25,100,1000)
power <- matrix(NA, nrow=length(n.range), ncol=length(rho.range)) # matrix to store the power calculated on each rho in rho.range
for (i in 1:length(n.range)) {
  M <- circularM(n.range[i], m) # creates a "circular world" matrix
  power[i,] <- powerE4(rho.range, alpha, M, TRUE)
}

# plots all the power envelopes
postscript("/media/EPGE/Docs/figures/chart01a.eps", horizontal=FALSE, onefile=FALSE, height=4.5, width=6, pointsize=10, paper="special")
plot(rho.range, power[,], ylim=c(0,1), type="l", lty=1, xlab=expression(paste("\(\rho\)")), ylab= expression(beta(rho[1])))
for (i in 2:length(n.range)) {
  lines(rho.range, power[,], lty=i)
}
legend.text <- c(paste("Matrix", n.range[1], "x", n.range[1], "with\n", m, "ahead/behind neighbors"), paste("Matrix", n.range[-1], "x", n.range[-1]))
legend("bottomleft", legend=legend.text, lty=1:length(n.range), bty="n")
dev.off()

# Charts the Edgeworth Approximation for two values of rho
n<100
M< circularM(n,2)
smax<6
rho< 0.248
k< k< NULL
for (s in 1:smax) {
  k[s]<2^(-s-1)*factorial(s-1)*tr(powerM(A_r(rho,M),s))/n^(s/2)
  ka[s]<2^(-s-1)*factorial(s-1)*
    tr(powerM(t(tinv(rho,M))%*% A_r(rho,M) %*% t(tinv(rho,M),s))/ n^(s/2)
}
powerE4(ka, nstddev=2, pfile=NULL)
powerE4(ka, nstddev=2, pfile="chart02a.eps") # rho=0.20
powerE4(ka, nstddev=2, pfile="chart02b.eps") # rho=0.248
# calculates which is the max Variance when rho=0.248
print(maxVar(0.248, circularM(100,2), sigma2=1))
# calculates which is the max Variance when rho=0.20
print(maxVar(0.20, circularM(100,2), sigma2=1))

# Plot power Envelopes for circular world design with n=100, m=2, m=5, n=10.
epsilon <- .Machine$double.eps^0.5 # defines an arbitrarily small value
alpha <- 0.05 # size alpha test (limits the type I error)
m <- 30 # n x n matrix
m.range <- c(2,5,7)
rho.range <- seq(from=-0.99, to=0.99, len=100)
powerf <- matrix(NA, nrow=length(m.range), ncol=length(rho.range)) # matrix to store the power calculated on each m in m.range
for (i in 1:length(m.range)) {
  M <- circularM(n, m.range[i]) # creates a "circular world" matrix
  powerf[i,] <- powerE4(rho.range/2*m.range[i], alpha, M, TRUE)
}
plot(rho.range, powerf[,], ylim=c(0,1), type="l", lty=1, xlab=expression(paste("\(\rho/2\)")), ylab= expression(beta(rho[1])))
for (i in 2:length(m.range)) {
  lines(rho.range, powerf[,], lty=i)
}

# Defines the saddle point functions
KT <- function(gamma, C) {
  n< dim(C)[1]
  return (-0.5*1og(det(diag(n)-2*gamma*C/sqrt(n)))))
}

Listing 20: saddlepoint functions.R
# Defines the derivatives of Tn's cumulant generating function
KlinhaT <- function(s, gamma, C) {
  n <- dim(C)[1]
  Mtemp <- solve(diag(n) - 2*gamma*sqrt(n))*%*%C
  return((2^(-s-1))*factorial(s-1)*tr(powerM(Mtemp, s)) / n^-((s+2))
}

# Finds the saddlepoint KlinhaT(gamma)-t
saddlepoint <- function(t, C) {
  n <- dim(C)[1]
  gammarange <- sqrt(n)/(2*eigenRange(C))
  gammarange <- gammrange + c(1, -1) * .Machine$double.eps^0.5
  tempfun <- function(gamma, t, C)
    return(unroot(gamma, gammarange, t, C,
      tol = .Machine$double.eps^0.5)%*%root)
}

# Calculates tail probabilities Pr(T>t) based on the integration
# of Gaussian approximations - O(1/n)
PT <- function(t, C) {
  # define the integrand as function "vectorizable"
  tempfun <- function(gammavec, C) {
    tempfun2 <- function(gamma, C)
      sqrt(KlinhaT(2, gammavec, C)/(2*pi))*
      exp(KT(gammavec, C) - gamma*KlinhaT(1, gammavec, C))
    return(zapply(gammavec, tempfun2, C))
  }
  n <- dim(C)[1]
  gammarange <- sqrt(n)/(2*eigenRange(C))
  gammarange <- gammarange + c(1, -1) * .Machine$double.eps^0.5
  renorfactor <- integrate(tempfun, gammarange[1], gammarange[2], C,
    rel.tol = .Machine$double.eps^0.5)$value
  lowerlimit <- saddlepoint(t, C)
  return(integrate(tempfun, lowerlimit, gammarange[2], C,
    rel.tol = .Machine$double.eps^0.5)$value/renorfactor)
}

LugannaniRicePT <- function(t, C) {
  gammabar <- saddlepoint(t, C)
  <- sign(gammabar)*sqrt(2*(gammabar*KlinhaT(1, gammabar, C) - KT(gammabar, C)))
  k2gammabar <- KlinhaT(2, gammabar, C)
  k3gammabar <- KlinhaT(3, gammabar, C)
  if (abs(gammabar) < .Machine$double.eps^0.25)
    return((1-pnorm(a) - dnorm(a)*k2gammabar/6) * k3gammabar)
  else return((1-pnorm(a) - dnorm(a) * (k2gammabar + k3gammabar)) - 1/a))
}

# Define functions gerals
# Define a função traceo "tr"
tr <- function(m) if (is.matrix(m)) sum(diag(m)) else return(0)

# Define a função potencia de matriz
powerM <- function(M, n){
  if (n<0)
    return(-1)
  else if (n=0)
    return(diag(nrow(M)))
}
else
  return(M%*%rho*M(N,n-1))
}

## Funções que dependem da matriz vizinhança ##

# Define a matriz de vizinhança
w <- function(rho) {  # define aqui a matriz de vizinhança NxN
  rho%*%W
}
# Define a função "lagrangiano"
L <- function(rho) diag(N) - w(rho)
# define o determinante do lagrangiano
DetL <- function(rho) det(L(rho))
# Define a derivada da matriz de vizinhança
W <- function(rho) {  # define aqui a derivada
  M
}

# define a matriz A
A <- function(rho) return(w(rho)%*%t(w(rho))%*%t(w(rho)))
# define a expansão de Edgeworth
Ed.padrao <- function(x,k) pnorm(x) - dnorm(x)*(k[3]*(x^2-1)/6 + k[4]*(x^3-3*x)/24 + 10*k
  [3]*2*(x^5-10*x^3+15*x)/720)

Ed <- function(x,k) {
  A <- k * k[2]^-((1:4)/2) # invariantes
  Ed.padrao((x-k[1])/sqrt(k[2]),c(0,1,k[3:4]))
}

## Início
N <- 10  # define o tamanho da amostra
sigma2 <- 1  # variancia
V <- vector("list",4)

library(igraph)

# Exemplo de matriz vizinhança
G <- graph.full(N)
V[[1]] <- get.adjacency(G)

# Exemplo 2 matriz vizinhança
G <- graph.ring(N)
tkplot(g)
V[[2]] <- get.adjacency(G)

# Exemplo 3
G <- graph.star(N,mode="undirected")
tkplot(g)
V[[3]] <- get.adjacency(G)

# Exemplo 4
G <- graph.ring(N,directed=TRUE,circular=FALSE)
tkplot(g)
V[[4]] <- get.adjacency(G)

detach("package:igraph")

# Escolhe um dos exemplos
M <- V[[2]]  # 1- full, 2-mundo circular, 3- Star, 4- TS

eigen(M)$val

unroot(DetL,interval=c(-.99,0),tol=Machine$double.eps)$root
unroot(DetL,interval=c(0,.99),tol=Machine$double.eps)$root

# neste caso o intervalo para rho eh (-1,1/9)
cat("Intervalo para rho: ", rhointerval <- c((-1.001),1.001),"\n")

# Início simulacao
nsim <- 999
rho <- 0.01  # DEFINE rho
rho.range <- seq(-0.4,0.4,by=0.1)
T.stat <- array(NA, dim=c(length(rho.range), nsim))
T.null <- vector("numeric")

for (i in 1:nsim) {
ex <- matrix(rnorm(N, mean=0, sd=1), nrow=N, ncol=1)
for (r in 1:length(rho.range)) {
y <- solve(L(rho.range[r]))%*%e
T.stat[r, i] <- sigma2*(t(y)%*%A(rho)%*%y)/rho
cat("+ - ")
}
T.null[i] <- sigma2*(t(e)%*%A(rho)%*%e)/rho
cat("FIM! \n")
}
eF1 <- ecdf(T.stat[1,])
eF2 <- ecdf(T.stat[2,])
eF3 <- ecdf(T.stat[3,])
eF4 <- ecdf(T.stat[4,])
eF5 <- ecdf(T.stat[5,])
eF6 <- ecdf(T.stat[6,])
eF7 <- ecdf(T.stat[7,])
eF8 <- ecdf(T.stat[8,])
eF9 <- ecdf(T.stat[9,])
e.F.null <- ecdf(T.null)
p1 <- 1-eF1(19.25)
p2 <- 1-eF2(19.25)
p3 <- 1-eF3(19.25)
p4 <- 1-eF4(19.25)
p5 <- 1-eF5(19.25)
p6 <- 1-eF6(19.25)
p7 <- 1-eF7(19.25)
p8 <- 1-eF8(19.25)
p9 <- 1-eF9(19.25)

plot(rho.range, c(p1, p2, p3, p4, p5, p6, p7, p8, p9), type="l")

plot(eF6)
lines(e.F.null)

# Aplicacao de EW

k <- vector("numeric") # armazena os cumulantes de T

for (s in 1:4) { k[s] <- 2^(-s-1)*factorial(s-1)*tr(powerM(t(solve(L(rho))), %*% A(rho))%*% solve(L(rho))/rho, s))
}

a <- seq(-100, 100, by=0.1)
b <- vector("numeric")

for (i in 1:length(a)) { b[i] <- E4(a[i], k)

plot(eF, xlim=c(-100,100), do.points=FALSE)
lines(a, b, col="red")

Listing 22: simulaedgeworth.R
# Define a matriz de vizinhanca
w <- function(rho) {  # define aqui a matriz de vizinhanca N\times N
  rho\times N}

# Define a funcao "lagrangiano"
L <- function(rho) diag(N) - w(rho)

# define o determinante do lagrangiano
DetL <- function(rho) det(L(rho))

# Define a derivada da matriz de vizinhanca
W <- function(rho) {  # define aqui a derivada
  \nu }

# define a matriz A
A <- function(rho) return(w(rho)+t(w(rho))-t(w(rho))%*%w(rho))

# define a expansao de Edgeworth
E4.padrao <- function(x,k) pnorm(x)-dnorm(x)*(k[3]1/(2)+(x-3/24+10+k
  3)]-2*(x-5/10*y-3+15*x)/720)

E4 <- function(x,k) {
  k1 <- k * k[2]"^"(-(1/4)/2)  # invariantes
  E4.padrao((x-k[1])/sqrt(k[2]),c(0,1,k[3:4]))
}

## Inicio
N <- 10  # define o tamanho da amostra
sigma2 <- 1  # variancia
V <- vector("list",4)

library(igraph)
# Exemplo de matriz vizinhanca
g<-graph.full(N)
V[[1]]<-get.adjacency(g)

# Exemplo 2 matriz vizinhanca
x<-graph.full(N)
#tplot(g)
V[[2]]<-get.adjacency(g)

# Exemplo 3
g<-graph.star(N,mode="undirected")
#tplot(g)
V[[3]]<-get.adjacency(g)

# Exemplo 4
x<-graph.full(N,directed=TRUE,circular=FALSE)
#tplot(g)
V[[4]]<-get.adjacency(g)

detach("package:igraph")

# Escolhe um dos exemplos
M <- V[[1]] # 1- full, 2- mundo circular, 3- Star, 4- TS

eigen(M)$val

unisolve(DetL,interval=c(-0.99,0),tol=Machine$double.eps)$root
unisolve(DetL,interval=c(0.0,0.99),tol=Machine$double.eps)$root

# neste caso o intervalo para rho eh (-1,1/9)
cat("Intervalo para rho: ", rhointerval <- c(-1.0001,1-0.001)"\n")

# Inicio simulacao
nsim<-999
T.stat <- vector("numeric")
 rho <- 0.00001 # DEFINE rho !

for (i in 1:nsim) {
  y<-matrix(rnorm(N,mean=0,sd=1),nrow=N,ncol=1)
  T.stat[i] <- sigma2*(t(y)%*%A(rho)%*%y)/rho
cat("+--")
}

Listing 23: art1 chart03.R

```r
# Compare the critical values obtained using different methods
rm(list=ls()) # Apaga todas variáveis da memória
source("/media/EPGE/Docs/programas/all_functions.R")
x11.options(width=6,height=5,type="ncairo")
ps.options(horizontal=FALSE, onfile=FALSE, height=4.5,
          width=6, paper="special", pointsize=10)
nc<-30
m<-2
M<-circularM(n,m)
alpha<-0.05
rho1<-0 # LUMP test
Ev4CV_V(alpha, rho1,M)
Ev5CV_V(alpha, rho1,M)
SymmetricCV_V(alpha, rho1,M)
RACV_V(alpha, rho1,M)
LRACV_V(alpha, rho1,M)
print(ca_ev <- Ev4CV_T(alpha, rho1,M))
print(ca_evb <- Ev5CV_T(alpha, rho1,M))
print(ca_cf <- CornFlCV_T(alpha, rho1,M))
print(ca_sp <- saddlepointCV_T(alpha, rho1,M))
print(ca_s1i <- saddlepointCV_T(alpha, rho1,M))
print(ca_ra <- RACV_T(alpha, rho1,M))
print(ca_lra <- LRACV_T(alpha, rho1,M))
print(ca_bnja <- BNJACV_T(alpha, rho1,M))
load("/media/EPGE/Docs/dados/MC.CV.Design1_30.2.Rdata") # returns ca_mc
ca_sim_norm<-round(ca_mc[[21]][!is.na(ca_mc[[21]])],2)
ca_sim_std<-round(ca_mc[[3]][!is.na(ca_mc[[3]])],2)
tam<-length(ca_sim_norm)
print(ca_true <- list(norm=ca_sim_norm[tam],std=ca_sim_std[tam]))
load("/media/EPGE/Docs/dados/MC.CV.V.Design1_30.2.Rdata") # returns ca_mc
ca_sim_norm<-round(ca_mc[[21]][!is.na(ca_mc[[21]])],2)
ca_sim_std<-round(ca_mc[[3]][!is.na(ca_mc[[3]])],2)
tam<-length(ca_sim_norm)
print(ca_true <- list(norm=ca_sim_norm[tam],std=ca_sim_std[tam]))
x<-ecdf(ca_mc[[1]])
x(9.73)
```

Listing 24: art1 compare Curvature V.R

```r
# This program compares calculates the curvature of the different models
rm(list=ls()) # Apaga todas variáveis da memória
curve<-NA
```
nlist <- c(30, 92)
mlist <- c(2, 6, 10)
Jlist1 <- c(9, 15, 22)
Jlist2 <- c(16, 27, 38)
idx <- 0
name <- NULL

for (n in nlist) for (i in 1:3) {
  # loads the ldot (l1) and ldotdot (l2d) variables and calculates the curvature
  load(paste("/media/EPGE/Docs/dados/MC_Curvature_V_Design1", n, ",", mlist[i], ".Rdata", sep = " "))
  info <- mean(ca.mc$l1d)
  v1i <- mean(ca.mc$l1d * ca.mc$l1dd)
  vo2i <- mean((ca.mc$l1dd - 2) - info - 2)
  curv[idx] <- sqrt((vo2i/info - 2)/v1i^2/info3)
  dname <- c(dname, paste("Design1", n, ",", mlist[i], sep = " " ))

dx <- idx + 1
  load(paste("/media/EPGE/Docs/dados/MC_Curvature_V_Design2", n, ",", mlist[i], ".Rdata", sep = " "))
  info <- mean(ca.mc$l1d)
  v1i <- mean(ca.mc$l1d * ca.mc$l1dd)
  vo2i <- mean((ca.mc$l1dd - 2) - info - 2)
  curv[idx] <- sqrt((vo2i/info - 2)/v1i^2/info3)
  dname <- c(dname, paste("Design2", n, ",", mlist[i], sep = " " ))

idx <- idx + 1
  load(paste("/media/EPGE/Docs/dados/MC_Curvature_V_Design3", n, ",", mlist[i], ".Rdata", sep = 
" "))
  info <- mean(ca.mc$l1d)
  v1i <- mean(ca.mc$l1d * ca.mc$l1dd)
  vo2i <- mean((ca.mc$l1dd - 2) - info - 2)
  curv[idx] <- sqrt((vo2i/info - 2)/v1i^2/info3)
  dname <- c(dname, paste("Design3", n, ",", mlist[i], sep = " " ))

idx <- idx + 1
  load(paste("/media/EPGE/Docs/dados/MC_Curvature_V_Design4", n, ",", mlist[i], ".Rdata", sep = 
"
" ))
  info <- mean(ca.mc$l1d)
  v1i <- mean(ca.mc$l1d * ca.mc$l1dd)
  vo2i <- mean((ca.mc$l1dd - 2) - info - 2)
  curv[idx] <- sqrt((vo2i/info - 2)/v1i^2/info3)
  dname <- c(dname, paste("Design4", n, ",", mlist[i], sep = " " ))

dx <- idx + 1
  load(paste("/media/EPGE/Docs/dados/MC_Curvature_V_Design5.a.Rdata")
  info <- mean(ca.mc$l1d)
  v1i <- mean(ca.mc$l1d * ca.mc$l1dd)
  vo2i <- mean((ca.mc$l1dd - 2) - info - 2)
  curv[idx] <- sqrt((vo2i/info - 2)/v1i^2/info3)
  dname <- c(dname, "Design5.a")

idx <- idx + 1
  load("/media/EPGE/Docs/dados/MC_Curvature_V_Design5.b.Rdata")
  info <- mean(ca.mc$l1d)
  v1i <- mean(ca.mc$l1d * ca.mc$l1dd)
  vo2i <- mean((ca.mc$l1dd - 2) - info - 2)
  curv[idx] <- sqrt((vo2i/info - 2)/v1i^2/info3)
  dname <- c(dname, "Design5.b")

dx <- idx + 1
  load("/media/EPGE/Docs/dados/MC_Curvature_V_Design5.c.Rdata")
  info <- mean(ca.mc$l1d)
  v1i <- mean(ca.mc$l1d * ca.mc$l1dd)
  vo2i <- mean((ca.mc$l1dd - 2) - info - 2)
  curv[idx] <- sqrt((vo2i/info - 2)/v1i^2/info3)
  dname <- c(dname, "Design5.c")
# Compare all Designs
rm(list=ls())
source("/media/EPGE/Docs/programas/all_functions.R")
X11.options(width=5, height=5, type="ncairo")
ps.options(horizontal=FALSE, onefile=FALSE, height=4.5, width=6, paper="special", pointsize=10)

alpha <- 0.05
epsilon <- .Machine$double.eps^0.5
M <- vector("list", 5)
c.value <- vector("list", 4)
len.rho.range <- 50
rhol <- array(NA, dim=c(4, length(m.list), len.rho.range))
poveryx <- array(NA, dim=c(4, length(m.list), len.rho.range))
poveryen <- array(NA, dim=c(4, length(m.list), len.rho.range))
powerenv <- array(NA, dim=c(4, length(m.list), len.rho.range))
powerloss <- array(NA, dim=c(4, length(m.list), len.rho.range))

for (i in 1:3) {
cat("Starting designs with i=" , i, "\n")
M[[1]] <- genDesign1(m.list[i])
M[[2]] <- genDesign2(m.list[i])
M[[3]] <- genDesign3(m.list[i])
M[[4]] <- genDesign4.a(m.list[i])
M[[5]] <- genDesign4.b(m.list[i])

table <- data.frame(dname[,1], curv[,1]^2, dname[,2], curv[,2]^2, dname[,3], curv[,3]^2)
library(xtable)
xtable(table)

table <- data.frame(curv^2, dname[1])
curv^2 >= 1/8

cat("Calculates power of LUMPI\n")
for (j in 1:1:1) {
  cat("Calculates power of LUMPI\n")
  (powerfix1[1, i, j] <- Ev4PV(c.value[[1]], norm=0, rho.range[1, i, j], M[[1]], NULL))
  if (j>1) if (powerfix1[1, i, j]<powerfix1[1, i, j-1])
  cat("Warning! Power has DECREASED for Design 1 at rho=" , rho.range[1, i, j], ",\n")
  (powerfix2[1, i, j] <- Ev4PV(c.value[[2]], norm=0, rho.range[2, i, j], M[[2]], NULL))
  if (j>1) if (powerfix2[1, i, j]<powerfix2[1, i, j-1])
  cat("Warning! Power has DECREASED for Design 2 at rho=" , rho.range[2, i, j], ",\n")
  (powerfix3[1, i, j] <- Ev4PV(c.value[[3]], norm=0, rho.range[3, i, j], M[[3]], NULL))
  if (j>1) if (powerfix3[1, i, j]<powerfix3[1, i, j-1])
  cat("Warning! Power has DECREASED for Design 3 at rho=" , rho.range[3, i, j], ",\n")
  (powerfix4[1, i, j] <- Ev4PV(c.value[[4]], norm=0, rho.range[4, i, j], M[[4]], M[[5]]))
  if (j>1) if (powerfix4[1, i, j]<powerfix4[1, i, j-1])
  cat("Warning! Power has DECREASED for Design 4 at rho=" , rho.range[4, i, j], ",\n")
}
cat("Calculates the power envelope\n")

Listing 25: art1 compare optimal vs LUMPI test Edgeworth.R
Listing 26: art1 compare optimal vs LUMPI test.R

```r
# Critical value vs. rho1
# using saddlepoint
# Compare all Designs
ze(list=ls())
source("/media/EPGE/Docs/programs/all_functions.R")
X11.options(width=8, height=5, type="pdfcairo")
ps.options(horizontal=FALSE, onefile=FALSE, height=4.5, width=6, paper="special", pointsize=10)
```
m.list <- c(2, 6, 10)
J.list <- c(9, 8, 22)
alpha <- 0.05
epsilon <- .Machine$double.eps * 0.5
M <- vector("list", 5)
c.value <- vector("list", 4)
len.rho.range <- 50
rho.range <- array(NA, dim = c(4, length(m.list), len.rho.range))
powerfix <- array(NA, dim = c(4, length(m.list), len.rho.range))
powerloss <- array(NA, dim = c(4, length(m.list), len.rho.range))

for (i in 1:3) {
cat("Starting designs with i =", i, "\n")
M[1] <- genDesign1(m, m.list[1])
M[2] <- genDesign2(m, m.list[2])
M[3] <- genDesign3(n, (2*n-1)*(J.list[i]-1) + (J.list[i]+2)/(2*n), J.list[i])
M[4] <- genDesign4_a(m, m.list[1])
M[5] <- genDesign4_b(m, m.list[1])

rho.range[i, i, j] <- seq(from = 0, to = 1/eigenRange(M[[i]])[2] - sqrt(epsilon), length.out = len.rho.range)
power.range[i, i, j] <- seq(from = 0, to = 1/eigenRange(M[[i]])[2] - sqrt(epsilon), length.out = len.rho.range)

c.value[[1]] <- BNJACV_V(alpha, 0, M[[1]], NULL)
c.value[[2]] <- BNJACV_V(alpha, 0, M[[2]], NULL)
c.value[[3]] <- BNJACV_V(alpha, 0, M[[3]], NULL)
c.value[[4]] <- BNJACV_V(alpha, 0, M[[4]], M[[5]])

cat("Calculates power of LUMPI\n")
for (j in 1:len.rho.range) {
  (powerfix[1, i, j] <- BNJACV(c.value[[1]]$norm, 0, rho.range[i, i, j], M[[1]], NULL))
  if (j > i) if (powerfix[1, i, j] < powerfix[1, i, j-1])
    cat("Warning! Povice has DECREASED for Design 1 at rho =", rho.range[i, i, j], "\n")
  (powerfix[2, i, j] <- BNJACV(c.value[[2]]$norm, 0, rho.range[2, i, j], M[[2]], NULL))
  if (j > i) if (powerfix[2, i, j] < powerfix[2, i, j-1])
    cat("Warning! Povice has DECREASED for Design 2 at rho =", rho.range[2, i, j], "\n")
  (powerfix[3, i, j] <- BNJACV(c.value[[3]]$norm, 0, rho.range[3, i, j], M[[3]], NULL))
  if (j > i) if (powerfix[3, i, j] < powerfix[3, i, j-1])
    cat("Warning! Povice has DECREASED for Design 3 at rho =", rho.range[3, i, j], "\n")
  (powerfix[4, i, j] <- BNJACV(c.value[[4]]$norm, 0, rho.range[4, i, j], M[[4]], M[[5]]))
  if (j > i) if (powerfix[4, i, j] < powerfix[4, i, j-1])
    cat("Warning! Povice has DECREASED for Design 4 at rho =", rho.range[4, i, j], "\n")
}

cat("Calculates the power envelope\n")
cat("For Design 1\n")
powerenv[1, i, j] <- powerBNJ_V(rho.range[1, i, j], alpha, M[[1]], NULL, FALSE)
cat("For Design 2\n")
powerenv[2, i, j] <- powerBNJ_V(rho.range[2, i, j], alpha, M[[2]], NULL, FALSE)
cat("For Design 3\n")
powerenv[3, i, j] <- powerBNJ_V(rho.range[3, i, j], alpha, M[[3]], NULL, FALSE)
cat("For Design 4\n")
powerenv[4, i, j] <- powerBNJ_V(rho.range[4, i, j], alpha, M[[4]], M[[5]], FALSE)

(powerloss[1, i, j] <- (powerfix[1, i, j] - powerenv[1, i, j]))
(powerloss[2, i, j] <- (powerfix[2, i, j] - powerenv[2, i, j]))
(powerloss[3, i, j] <- (powerfix[3, i, j] - powerenv[3, i, j]))
(powerloss[4, i, j] <- (powerfix[4, i, j] - powerenv[4, i, j]))

save(m.list, J.list, alpha, epsilon, len.rho.range, rho.range, powerfix, powerenv, powerloss, file="/media/EPGE/Docs/dados/optimal_vs_LUMPI_test_n30.Rdata"
i<-1
 summary(powerloss[1,1,] <- (powerfix[1,1,] - powerenv[1,1,]))
 summary(powerloss[2,1,] <- (powerfix[2,1,] - powerenv[2,1,]))
 summary(powerloss[3,1,] <- (powerfix[3,1,] - powerenv[3,1,]))
 summary(powerloss[4,1,] <- (powerfix[4,1,] - powerenv[4,1,]))

# Prepare box plot
tempvalue<-NULL
tempchar<-NULL
for (i in 1:4) for (j in 1:3) {
tempvalue<c(tempvalue,powerloss[i,j])
tempchar<c(tempchar,paste("Design",i,
"","i",j,"m-",m="",i="3",J.list[j],m.list[j],"",sep=""),len.
rho.range))
}

oldpar<-par()
postscript("/media/EPGE/Docs/figures/art1_chart11g.eps")
par(mar=c(6,6,4,2)+0.1,adj=0.5)
boxplot(100*tempvalue,tempchar,data=data.frame(tempvalue,tempchar),horizontal=TRUE,
las=1,xlab="Power difference (in percentuals)",cex.axis=0.8,cex.lab=0.8)
dev . off()
par(oldpar)

## Compare all Designs (n=92)
rm(list=ls())
source("/media/EPGE/Docs/programs/all_functions.R")
X11(options(width=5,height=5,type="ncairo")
ps.options(horizontal=FALSE,onefile=FALSE,height=4.5,
width=6,paper="special",pointsize=10)

n<-92
J.list<-c(2,6,10)
J.list1<-c(16,27,38)
alpha<-0.05
epsilon <- .Machine$double.$eps/0.5
M<-vector("list",6)
c.value<-vector("list",4)
len_rho.range<-50
rho.range <- array(NA,dim=c(5,length(M.list),len_rho.range))
powerfix<-array(NA,dim=c(5,length(M.list),len_rho.range))
powerenv<-array(NA,dim=c(5,length(M.list),len_rho.range))
powerloss<-array(NA,dim=c(5,length(M.list),len_rho.range))

load(file="/media/EPGE/Docs/dados/design5.Rdata")
MSa<"design5a"
MSb<"design5b"
MSC<"design5c"

for (i in 1:3) {
cat("Starting designs with i="i,"\n"
M[1][i]<-genDesign1(a,m.list[1])
M[2][i]<-genDesign2(a,m.list[1])
M[3][i]<-genDesign3(2n,J.list[1],(J.list[1]+2))/2n,J.list[1])
M[4][i]<-genDesign4_a(a,m.list[1])
M[5][i]<-genDesign5_b(m.m.list[1])
M[6][i]<if (i==1) Msa else if (i==2) MSb else MSC

rho.range[1,1,] <- seq(from=0,to=1/eigenRange(M[1][1])[2]-sqrt(epsilon),length.out=len.rho.
rho.range)
rho.range[2,1,] <- seq(from=0,to=1/eigenRange(M[2][1])[2]-sqrt(epsilon),length.out=len.rho.
rho.range)
rho.range[3,1,] <- seq(from=0,to=1/eigenRange(M[3][1])[2]-sqrt(epsilon),length.out=len.rho.
rho.range)
rho.range[4,1,] <- seq(from=0,to=sqrt(1-4/m.list[1])-1/2-sqrt(epsilon),length.out=len.rho.
rho.range)
rho.range[5,1,] <- seq(from=0,to=1/eigenRange(M[6][1])[2]-sqrt(epsilon),length.out=len.rho.
rho.range)
cat("Obtain the critical value for the LUMP test\n")
c.value[1][1]<-BNJACV_V(alpha,0,M[1][1],NULL)
c.value[2][1]<-BNJACV_V(alpha,0,M[2][1],NULL)
c.value[3][1]<-BNJACV_V(alpha,0,M[3][1],NULL)
(c.value[[4]] <- BNJAVC_V(alpha, M[[4]], M[[5]]))

(c.value[[5]] <- BNJAVC_V(alpha, M[[6]], NULL))

cat("Calculates power of LUMP \\
")
for (j in 1: len.rho.range)
{
  (powerfix[1, i, j] <- BNJAVF(c.value[[1]], norm, 0, rho.range[1, i, j], M[[1]], NULL))
  if (j > 1) if (powerfix[1, i, j] < powerfix[1, i, j-1])
    cat("Warning! Power has DECREASED for Design 1 at rho", rho.range[1, i, j], \\
"\n")
  (powerfix[2, i, j] <- BNJAVF(c.value[[2]], norm, 0, rho.range[2, i, j], M[[2]], NULL))
  if (j > 1) if (powerfix[2, i, j] < powerfix[2, i, j-1])
    cat("Warning! Power has DECREASED for Design 2 at rho", rho.range[2, i, j], \\
"\n")
  (powerfix[3, i, j] <- BNJAVF(c.value[[3]], norm, 0, rho.range[3, i, j], M[[3]], NULL))
  if (j > 1) if (powerfix[3, i, j] < powerfix[3, i, j-1])
    cat("Warning! Power has DECREASED for Design 3 at rho", rho.range[3, i, j], \\
"\n")
  (powerfix[4, i, j] <- BNJAVF(c.value[[4]], norm, 0, rho.range[4, i, j], M[[4]], M[[3]]))
  if (j > 1) if (powerfix[4, i, j] < powerfix[4, i, j-1])
    cat("Warning! Power has DECREASED for Design 4 at rho", rho.range[4, i, j], \\
"\n")
  (powerfix[5, i, j] <- BNJAVF(c.value[[5]], norm, 0, rho.range[5, i, j], M[[6]], NULL))
  if (j > 1) if (powerfix[5, i, j] < powerfix[5, i, j-1])
    cat("Warning! Power has DECREASED for Design 5 at rho", rho.range[5, i, j], \\
"\n")
}


cat("Calculates the power envelope")
cat("For Design 1")
powerenv[1, i, j] <- powerBNJ_V(rho.range[1, i, j], alpha, M[[1]], NULL, FALSE)
cat("For Design 2")
powerenv[2, i, j] <- powerBNJ_V(rho.range[2, i, j], alpha, M[[2]], NULL, FALSE)
cat("For Design 3")
powerenv[3, i, j] <- powerBNJ_V(rho.range[3, i, j], alpha, M[[3]], NULL, FALSE)
cat("For Design 4")
powerenv[4, i, j] <- powerBNJ_V(rho.range[4, i, j], alpha, M[[4]], M[[3]], FALSE)
cat("For Design 5")
powerenv[5, i, j] <- powerBNJ_V(rho.range[5, i, j], alpha, M[[5]], FALSE)

(powers[1, i, j] <- (powerfix[1, i, j] - powerenv[1, i, j])
(powers[2, i, j] <- (powerfix[2, i, j] - powerenv[2, i, j])
(powers[3, i, j] <- (powerfix[3, i, j] - powerenv[3, i, j])
(powers[4, i, j] <- (powerfix[4, i, j] - powerenv[4, i, j])
(powers[5, i, j] <- (powerfix[5, i, j] - powerenv[5, i, j])

# gravados depois das correcoes e colocando BNJ
save(n,m.list,J.list, alpha, epsilon, len.rho.range, rho.range,
powerfix,powerenv,powers, file="/media/EPGE/Docs/dados/optimal_vs_LUMP/test_n92.Rdata")

# Corrige manualmente alguns bugs no calculo de power
# normalmente proximos ao ponto onde a segunda derivada eh zero (sela)
powers[5,2,22] <- powers[5,2,22] + powerenv[5,2,22]
powers[1,3,19] <- powers[1,3,18] + powers[1,3,20]/2
powers[1,3,19] <- powers[1,3,19] + powerenv[1,3,19]
\begin{verbatim}
222
223  # Prepare box plot
tempvalue <- NULL
tempchar <- NULL
chardesigns <- c("a", "b", "c")
for (i in 1:5) for (j in 1:3) {
tempvalue <- c(tempvalue, powerloss[i, j,])
  tempchar <- c(tempchar, rep(paste("Design", i, "", "i=", j, "", m=""),
                   ifelse(i==3, "J=", ifelse(i==5, chardesigns[j], "m=")),
                   ifelse(i==3, J.list[j], ifelse(i==5, "m.list[j],
                   " "))))
}
oldpar <- par()
postscript("/media/EPGE/Docs/figures/art1_chart11f.eps")
par(mar=c(0,6,4,2)+0.1, adj=0.5)
boxplot(100 + tempvalue ~ tempchar, data=data.frame(tempvalue, tempchar), horizontal=TRUE,
        las=1, xlab="Power Difference (in percentuals)", cex.axis=0.8, cex.lab=0.8)
dev.off()
par(oldpar)

244  ## Design 5a
245  postscript("/media/EPGE/Docs/figures/art1_chart11a.eps")
246  par(mar=c(6,4,2,2)+0.1)
247  layout(matrix(c(1,2,2,1), 2))
248  plot(rho.range[5,1,1], powerenv[5,1,], ylim=c(0,1), type="l", lty=1,
        xlab=expression(rho[1]), ylab= expression(beta(rho[1])))
249  lines(rho.range[5,1,1], powerfix[5,1,], lty=2)
250  legend("bottomright", legend=legend.text, lty=1:2, bty="n")
251  plot(rho.range[5,1,1], powerloss[5,1,], type="l", xlab=expression(rho[1]), ylab="Difference")
252  dev.off()

256  ## Design 5b
257  postscript("/media/EPGE/Docs/figures/art1_chart11b.eps")
258  par(mar=c(6,4,2,2)+0.1)
259  layout(matrix(c(1,2,2,1), 2))
260  plot(rho.range[5,2,1], powerenv[5,2,], ylim=c(0,1), type="l", lty=1,
        xlab=expression(rho[1]), ylab= expression(beta(rho[1])))
261  lines(rho.range[5,2,1], powerfix[5,2,], lty=2)
262  legend("bottomright", legend=legend.text, lty=1:2, bty="n")
263  plot(rho.range[5,2,1], powerloss[5,2,], type="l", xlab=expression(rho[1]), ylab="Difference")
264  dev.off()

268  ## Design 5c
269  postscript("/media/EPGE/Docs/figures/art1_chart11c.eps")
270  par(mar=c(6,4,2,2)+0.1)
271  layout(matrix(c(1,2,2,1), 2))
272  plot(rho.range[5,3,1], powerenv[5,3,], ylim=c(0,1), type="l", lty=1,
        xlab=expression(rho[1]), ylab= expression(beta(rho[1])))
273  lines(rho.range[5,3,1], powerfix[5,3,], lty=2)
274  legend("bottomright", legend=legend.text, lty=1:2, bty="n")
275  plot(rho.range[5,3,1], powerloss[5,3,], type="l", xlab=expression(rho[1]), ylab="Difference")
276  dev.off()

280  ## Design 3.1-6 - worst performance with largest powerloss
281  postscript("/media/EPGE/Docs/figures/art1_chart11d.eps")
282  par(mar=c(6,4,2,2)+0.1)
283  layout(matrix(c(1,2,2,1), 2))
284  plot(rho.range[3,1,1], powerenv[3,1,], ylim=c(0,1), type="l", lty=1,
        xlab=expression(rho[1]), ylab= expression(beta(rho[1])))
285  lines(rho.range[3,1,1], powerfix[3,1,], lty=2)
286  legend("topleft", legend=legend.text, lty=1:2, bty="n")
287  plot(rho.range[3,1,1], powerloss[3,1,], type="l", xlab=expression(rho[1]), ylab="Difference")
288  dev.off()
\end{verbatim}
Listing 27: art1 compare powerenvelope saddle vs empirical design5.R
```r
# empirical power
for (i in 1:3) for (j in 1:n) e.power[i, j] <- 1 - ecdf(e.values[i, j])(cvalue[i])
```
source("/media/EPGE/Docs/programas/all_functions.R")
X11.options(width=5,height=5,type="ncairo")
ps.options(horizontal=FALSE, onefile=FALSE, height=4.5,
width=6, paper="special", pointsize=10)

nrange<-seq(30,510, by=15)
calpha<-matrix(NA, length(nrange),3)

alpha<-0.05 # size of the test
m<-9 # comparable to design 5c

for (j in 1:length(nrange)) {
  M<-vector("list",3)
  n<-nrange[j]
  M[[1]]<-genDesign1(n,m)
  M[[2]]<-genDesign2(n,m)
  if (j<22) # same average than design 5c (when n=30)
    m3<-(2*(n-1)+(J-1)*(J+2))/(2*n)
    M[[3]]<-genDesign3(n,m3,J)
  for (i in 1:J) {
    print(calpha[j,i]<-Ew4CV_V(alpha,0,M[[i]])$std)
  }
  cat("...",n)
}
save(nrange,calpha,file="/media/EPGE/Docs/dados/CV_designs1_2_3_m9_increasing_n.Rdata")
calpha_BNJ <- calpha
calpha2_BNJ <- calpha2

# Plot
postscript("/media/EPGE/Docs/figures/arti_chart03a.eps")
plot(nrange, calpha[,][,1], ylim=c(1.6,2), type="l", lty=2,xlab="Sample size (n)",
ylab=expression(paste("Critical values (standardized, level ",alpha,"=5%"))))
lines(nrange, calpha[,][,2], lty=3)
lines(nrange, calpha[,][,3], lty=4)
abline(h=qnorm(1-alpha), lty=1)
pointr(510,mc1[[3]][49999],pch=21)
pointr(510,mc2[[3]][49999],pch=21)
pointr(510,mc3[[3]][49999],pch=21)
legend("topright", lty=c(2:4,1,NA),pch=c(NA,NA,NA,NA,21),bty="n", legend=legend)
dev.off()

# outra simulacao agora com m=2
calpha2<-matrix(NA, length(nrange),3)
m<-2 # comparable to design 5a

for (j in 1:length(nrange)) {
  M<-vector("list",3)
  n<-nrange[j]
  M[[1]]<-genDesign1(n,m)
  M[[2]]<-genDesign2(n,m)
  if (j<9) # keep avg number of neighbors = 4 when n=30
    m3<-(2*(n-1)+(J-1)*(J+2))/(2*n)
    M[[3]]<-genDesign3(n,m3,J)
  for (i in 1:J) {
    print(calpha2[j,i]<-Ew4CV_V(alpha,0,M[[i]])$std)
  }
  cat("...",n)
}
save(nrange,calpha2,file="/media/EPGE/Docs/dados/CV_designs1_2_3_m2_increasing_n.Rdata")

# Plot
postscript("/media/EPGE/Docs/figures/arti_chart03b.eps")
plot(nrange, calpha2[,][,1], ylim=c(1.6,1.8), type="l", lty=2,xlab="Sample size (n)",
ylab=expression(paste("Critical values (standardized, level ",alpha,"=5%"))))
Listing 30: art1 CValue change avg degree.R

```r
# compare the changes in the critical values
# resulting from changing either the average degree
# or the average number of neighbors
rm(list="ls")
source("/media/EPGE/Docs/programs/all functions.R")
X11.options(width=5, height=5, type="pdfcairo")
ps.options(horizontal=FALSE, onefile=FALSE, height=4.5, width=6, paper="special", pointsize=10)
library(igraph)

n<92
m<2
J<16
m0<1
mM<20
calpha<-matrix(NA,mM-m0+1,3)
navgViz<-matrix(NA,mM-m0+1,3)
alpha<-0.06
for (m in m0:mM) {
  M<vector("list",3)
  M[[1]]<-genDesign1(n,m)
  M[[2]]<-genDesign2(n,m)
  M[[3]]<-genDesign3(n,m)
  navgViz[m,1]<-mean(degree(graph.adjacency(M[[1]],mode="undirected")))
  navgViz[m,2]<-mean(degree(graph.adjacency(M[[2]],mode="directed"),mode="all")) #mode used for directed graphs
  navgViz[m,3]<-mean(degree(graph.adjacency(M[[3]],mode="undirected",weighted=TRUE)))
  for (i in 1:3) print(calpha[m-m0+1,i]<-BMJACV(alpha,0,M[[i]],NULL,TRUE)$std)
  cat("...",m)
}
```
Listing 31: art1 Edgeworth vs true.R

```r
# This program calculates the power envelope for testing
```
# H0: rho_0 = 0 vs. H1: rho_1 <> 0
# in the SAR model y=v(rho)y+u
rm(list=ls()) # Apaga todas variáveis da memória
source("/media/EPGE/Docs/programas/all_functions.R")
X11.options(width=5, height=5, type="ncairo")
ps.options(horizontal=FALSE, onefile=FALSE, height=4.5,
width=6, paper="special", pointsize=10)

# Charts the Edgeworth Approximation for two values of rho
alpha<-0.05
n<92
m<2
M<-genDesign1(n,m)
sim1<MonteCarloCV_H1_V(49999, alpha, 0.20, M, NULL, TRUE)
sim2<MonteCarloCV_H1_V(49999, alpha, 0.24, M, NULL, TRUE)

FV1<ecdf(sim1[[1]])
FV2<ecdf(sim2[[1]])
k1<-mean(sim1[[1]])
k2<-var(sim1[[1]])
xrange1<-c(k1-2*sqrt(k2), k1+2*sqrt(k2))
k1<-mean(sim2[[1]])
k2<-var(sim2[[1]])
xrange2<-c(k1-2*sqrt(k2), k1+2*sqrt(k2))

FV1_Ew <- function(a) {
k <- cumulantsV(4,a,0.20,0.20, M, NULL)
k1<-invariantes(k)
return(E4(-k[1]/sqrt(k[2]),k1))
}
FV1_Ew_vec <- function(vec) sapply(vec,FV1_Ew)

FV2_Ew <- function(a) {
k <- cumulantsV(4,a,0.248,0.248, M, NULL)
k1<-invariantes(k)
return(E4(-k[1]/sqrt(k[2]),k1))
}
FV2_Ew_vec <- function(vec) sapply(vec,FV2_Ew)

# rho=0.20
postscript("/media/EPGE/Docs/figures/art1_chart06a.eps")
plot(FV1_Ew_vec, xrange1[1], xrange1[2],
main="Edgeworth Approximation", xlab="v", ylab="F(v)",
ylim=c(0,1), lty=1)
points(FV1, add=TRUE, col="gray",lwd=1, lty=2,do.points=FALSE)
text("Edgeworth approx.", adj=0, lty=1)
text("True distribution", col="gray", adj=1, lty=2)
dev.off()

# rho=0.24
postscript("/media/EPGE/Docs/figures/art1_chart06b.eps")
plot(FV2_Ew_vec, xrange2[1], xrange2[2],
main="Edgeworth Approximation", xlab="v", ylab="F(v)",
ylim=c(0,1), lty=1)
points(FV2, add=TRUE, col="gray",lwd=1, lty=2,do.points=FALSE)
text("Edgeworth approx.", adj=0, lty=1)
text("True distribution", col="gray", adj=1, lty=2)
dev.off()

# calculates which is the max Variance when rho=0.248
max(diag(Linv(0.24,M, NULL))%*%t(LinV(0.24,M, NULL))))

# calculates which is the max Variance when rho=0.20
max(diag(LinV(0.20,M, NULL))%*%t(LinV(0.24,M, NULL))))
rm(list=ls()) # Apaga todas variáveis da memória
source("/media/EPGE/Docs/programas/all_functions.R")
ps.options(horizontal=FALSE, onefile=FALSE, height=4.5,
           width=6, paper="special", pointsize=10)
S<-49999

## Design 1 - (Vn)
# n<-30
m<-2
M<-genDesign1(n,m)
camc<-MonteCarloCurvature_V(S,M,NULL,TRUE)
summary(cbind(camc[[1]], camc[[2]]))
save(camc, file="/media/EPGE/Docs/dados/MC_Curvature_V_Design1_30.2.Rdata")

# n<-6
m<-2
M<-genDesign1(n,m)
camc<-MonteCarloCurvature_V(S,M,NULL,TRUE)
summary(cbind(camc[[1]], camc[[2]]))
save(camc, file="/media/EPGE/Docs/dados/MC_Curvature_V_Design1_30.6.Rdata")

# n<-10
m<-2
M<-genDesign1(n,m)
camc<-MonteCarloCurvature_V(S,M,NULL,TRUE)
summary(cbind(camc[[1]], camc[[2]]))
save(camc, file="/media/EPGE/Docs/dados/MC_Curvature_V_Design1_30.10.Rdata")

# n<-92
m<-2
M<-genDesign1(n,m)
camc<-MonteCarloCurvature_V(S,M,NULL,TRUE)
summary(cbind(camc[[1]], camc[[2]]))
save(camc, file="/media/EPGE/Docs/dados/MC_Curvature_V_Design1_92.2.Rdata")

# n<-6
m<-2
M<-genDesign1(n,m)
camc<-MonteCarloCurvature_V(S,M,NULL,TRUE)
summary(cbind(camc[[1]], camc[[2]]))
save(camc, file="/media/EPGE/Docs/dados/MC_Curvature_V_Design1_92.6.Rdata")

# n<-10
m<-2
M<-genDesign1(n,m)
camc<-MonteCarloCurvature_V(S,M,NULL,TRUE)
summary(cbind(camc[[1]], camc[[2]]))
save(camc, file="/media/EPGE/Docs/dados/MC_Curvature_V_Design1_92.10.Rdata")

## Design 2 - (Tn)
# n<-30
m<-2
M<-genDesign2(n,m)
camc<-MonteCarloCurvature_V(S,M,NULL,TRUE)
save(camc, file="/media/EPGE/Docs/dados/MC_Curvature_V_Design2_30.2.Rdata")

# n<-6
m<-2
M<-genDesign2(n,m)
camc<-MonteCarloCurvature_V(S,M,NULL,TRUE)
save(camc, file="/media/EPGE/Docs/dados/MC_Curvature_V_Design2_30.6.Rdata")

# n<-10
m<-2
M<-genDesign2(n,m)
camc<-MonteCarloCurvature_V(S,M,NULL,TRUE)
save(camc, file="/media/EPGE/Docs/dados/MC_Curvature_V_Design2_30.10.Rdata")

# n<-92
m<-2
M<-genDesign2(n,m)
camc<-MonteCarloCurvature_V(S,M,NULL,TRUE)
save(camc, file="/media/EPGE/Docs/dados/MC_Curvature_V_Design2_92.2.Rdata")

m<-6
M<-genDesign2(n,m)
ca_mc<-MonteCarloCurvature_V(S,M,NULL,TRUE)
save(ca_mc,file="/media/EPGE/Docs/dados/MC_Curvature_V_Design2_92_6.Rdata")

m<-10
M<-genDesign2(n,m)
cac<--MonteCarloCurvature_V(S,M,NULL,TRUE)
save(cac,"/media/EPGE/Docs/dados/MC_Curvature_V_Design2_92_10.Rdata")

## Design3
# Design 5a has 450 non zeros, 5b has 960, 5c has 1662
# the values of J will be chosen to match the number of non-zeros
# on these designs. For smaller n=30, it will be chosen to match the average

n<-30
J<-round((sqrt((1+4*(2*n+450/92-2*(n-1))-1))/2,0)
m<-((2*(n-1)+(J-1)*(J+2))/(2*n)
M<-genDesign3(n,m)
ca_mc<-MonteCarloCurvature_V(S,M,NULL,TRUE)
save(ca_mc,"/media/EPGE/Docs/dados/MC_Curvature_V_Design3_30_9.Rdata")

J<-round((sqrt((1+4*(2*n+940/92-2*(n-1))-1))/2,0)
m<-((2*(n-1)+(J-1)*(J+2))/(2*n)
M<-genDesign3(n,m)
ca_mc<-MonteCarloCurvature_V(S,M,NULL,TRUE)
save(ca_mc,"/media/EPGE/Docs/dados/MC_Curvature_V_Design3_30_15.Rdata")

J<-round((sqrt((1+4*(2*n+1662/92-2*(n-1))-1))/2,0)
m<-((2*(n-1)+(J-1)*(J+2))/(2*n)
M<-genDesign3(n,m)
ca_mc<-MonteCarloCurvature_V(S,M,NULL,TRUE)
save(ca_mc,"/media/EPGE/Docs/dados/MC_Curvature_V_Design3_30_22.Rdata")

J<-92
J<-round((sqrt((1+4*(2*n+450/92-2*(n-1))-1))/2,0)
m<-((2*(n-1)+(J-1)*(J+2))/(2*n)
M<-genDesign3(n,m)
ca_mc<-MonteCarloCurvature_V(S,M,NULL,TRUE)
save(ca_mc,"/media/EPGE/Docs/dados/MC_Curvature_V_Design3_92_16.Rdata")

J<-round((sqrt((1+4*(2*n+940/92-2*(n-1))-1))/2,0)
m<-((2*(n-1)+(J-1)*(J+2))/(2*n)
M<-genDesign3(n,m)
ca_mc<-MonteCarloCurvature_V(S,M,NULL,TRUE)
save(ca_mc,"/media/EPGE/Docs/dados/MC_Curvature_V_Design3_92_27.Rdata")

J<-round((sqrt((1+4*(2*n+1662/92-2*(n-1))-1))/2,0)
m<-((2*(n-1)+(J-1)*(J+2))/(2*n)
M<-genDesign3(n,m)
ca_mc<-MonteCarloCurvature_V(S,M,NULL,TRUE)
save(ca_mc,"/media/EPGE/Docs/dados/MC_Curvature_V_Design3_92_38.Rdata")

## Design 4 - (Va)

n<-30
m<-2
M<-genDesign4_a(n,m)
ca_mc<-MonteCarloCurvature_V(S,M,M1,TRUE)
save(ca_mc,"/media/EPGE/Docs/dados/MC_Curvature_V_Design4_30_2.Rdata")

m<-6
M<-genDesign4_a(n,m)
ca_mc<-MonteCarloCurvature_V(S,M,M1,TRUE)
save(ca_mc,"/media/EPGE/Docs/dados/MC_Curvature_V_Design4_30_6.Rdata")

m<-10
M<-genDesign4_a(n,m)
ca_mc<-MonteCarloCurvature_V(S,M,M1,TRUE)
save(ca_mc,"/media/EPGE/Docs/dados/MC_Curvature_V_Design4_30_10.Rdata")

n<-92
m<-2
Listing 33: art1 MonteCarlo CVaule T.R

# Simulate the distribution of the various designs
# to calculate the critical Values for the LUMP and LUMPI tests
# save the results for future reference
rm(list=ls()) # Apaga todas variáveis da memoria
source("/media/EPGE/Docs/programs/all_functions.R")
X11.options(width=5, height=5, type="cairo")
ps.options(horizontal=FALSE, onefile=FALSE, height=4.5,
width=6, paper="special", pointsize=10)
S<-49999
alpha<-0.05
sigma2<-1
rho1<-0

## Design 1 - (Tn)
nc<-30
mc<-2
M<-genDesign1(n, m)
cvmc<-MonteCarloCV_T(S, alpha, rho1, sigma2, M, NULL, FALSE)
save(cvmc, file="/media/EPGE/Docs/dados/MC_CV_Design1_30.Rdata")
mc<-5
M<-genDesign1(n, m)
cvmc<-MonteCarloCV_T(S, alpha, rho1, sigma2, M, NULL, FALSE)
save(cvmc, file="/media/EPGE/Docs/dados/MC_CV_Design1_30_2.Rdata")
mc<-9
M<-genDesign1(n, m)
cvmc<-MonteCarloCV_T(S, alpha, rho1, sigma2, M, NULL, FALSE)
save(cvmc, file="/media/EPGE/Docs/dados/MC_CV_Design1_30_5.Rdata")
mc<-92
mc<-2
M<-genDesign1(n, m)
cvmc<-MonteCarloCV_T(S, alpha, rho1, sigma2, M, NULL, FALSE)
save(cvmc, file="/media/EPGE/Docs/dados/MC_CV_Design1_92.Rdata")
m<5
40 M<genDesign1(n,m)
41 ca_mc<MonteCarloCV_T(S, alpha, rho1, sigma2, M, NULL, FALSE)
42 save(ca_mc,file="/media/EPGE/Docs/dados/MC.CV_Design1_92.9.Rdata")
44 m<9
45 M<genDesign1(n,m)
46 ca_mc<MonteCarloCV_T(S, alpha, rho1, sigma2, M, NULL, FALSE)
47 save(ca_mc,file="/media/EPGE/Docs/dados/MC.CV_Design1_92_9.Rdata")
49 ## Design 2  - (Tn)
50 m<30
51 m<2
52 M<genDesign2(n,m)
53 ca_mc<MonteCarloCV_T(S, alpha, rho1, sigma2, M, NULL, FALSE)
54 save(ca_mc,file="/media/EPGE/Docs/dados/MC.CV_Design2_30.2.Rdata")
56 m<5
57 M<genDesign2(n,m)
58 ca_mc<MonteCarloCV_T(S, alpha, rho1, sigma2, M, NULL, FALSE)
59 save(ca_mc,file="/media/EPGE/Docs/dados/MC.CV_Design2_30.9.Rdata")
61 m<9
62 M<genDesign2(n,m)
63 ca_mc<MonteCarloCV_T(S, alpha, rho1, sigma2, M, NULL, FALSE)
64 save(ca_mc,file="/media/EPGE/Docs/dados/MC.CV_Design2_92.2.Rdata")
66 m<92
67 m<2
68 M<genDesign2(n,m)
69 ca_mc<MonteCarloCV_T(S, alpha, rho1, sigma2, M, NULL, FALSE)
70 save(ca_mc,file="/media/EPGE/Docs/dados/MC.CV_Design2_92.9.Rdata")
72 m<5
73 M<genDesign2(n,m)
74 ca_mc<MonteCarloCV_T(S, alpha, rho1, sigma2, M, NULL, FALSE)
75 save(ca_mc,file="/media/EPGE/Docs/dados/MC.CV_Design2_92_9.Rdata")
77 m<9
78 M<genDesign2(n,m)
79 ca_mc<MonteCarloCV_T(S, alpha, rho1, sigma2, M, NULL, FALSE)
80 save(ca_mc,file="/media/EPGE/Docs/dados/MC.CV_Design2_92_9.Rdata")
82 ## Design 3
83 # Design 3a has 450 non zeros, 5b has 940, 5c has 1662
84 # the values of J will be chosen to match the number of non-zeros
85 # on these designs. For smaller n<30, it will be chosen to match the average
87 n<30
88 J<round((sqrt(14*(2*n+450/92-2*(n-1)))-1)/2,0)
89 m<(2*(n-1)+(J-1)*(J+2))/(2*n)
90 M<genDesign3(n,m)
91 ca_mc<MonteCarloCV_T(S, alpha, rho1, sigma2, M, NULL, FALSE)
92 save(ca_mc,file="/media/EPGE/Docs/dados/MC.CV_Design3_30.9.Rdata")
94 J<round((sqrt(14*(2*n+940/92-2*(n-1)))-1)/2,0)
95 m<(2*(n-1)+(J-1)*(J+2))/(2*n)
96 M<genDesign3(n,m)
97 ca_mc<MonteCarloCV_T(S, alpha, rho1, sigma2, M, NULL, FALSE)
98 save(ca_mc,file="/media/EPGE/Docs/dados/MC.CV_Design3_30.15.Rdata")
100 J<round((sqrt(14*(2*n+1662/92-2*(n-1)))-1)/2,0)
101 m<(2*(n-1)+(J-1)*(J+2))/(2*n)
102 M<genDesign3(n,m)
103 ca_mc<MonteCarloCV_T(S, alpha, rho1, sigma2, M, NULL, FALSE)
104 save(ca_mc,file="/media/EPGE/Docs/dados/MC.CV_Design3_30.22.Rdata")
106 n<92
107 J<round((sqrt(14*(2*n+450/92-2*(n-1)))-1)/2,0)
108 m<(2*(n-1)+(J-1)*(J+2))/(2*n)
109 M<genDesign3(n,m)
110 ca_mc<MonteCarloCV_T(S, alpha, rho1, sigma2, M, NULL, FALSE)
111 save(ca_mc,file="/media/EPGE/Docs/dados/MC.CV_Design3_92.16.Rdata")
J <- round((sqrt(1 + 4 * (2 + n^940 / 92 - 2 * (n - 1))) - 1) / 2, 0)
c <- (2 * (n - 1) - (j - 1) * (j + 2)) / (2 * n)
c_a_mc <- MonteCarloCV_T(S, alpha, rho1, sigma2, M, NULL, FALSE)
save(c_a_mc, file = "/media/EPGE/Docs/dados/MC_CV_Design3_92_27.Rdata")

J <- round((sqrt(1 + 4 * (2 + n^1662 / 92 - 2 * (n - 1))) - 1) / 2, 0)
c <- (2 * (n - 1) - (j - 1) * (j + 2)) / (2 * n)
c_a_mc <- MonteCarloCV_T(S, alpha, rho1, sigma2, M, NULL, FALSE)
save(c_a_mc, file = "/media/EPGE/Docs/dados/MC_CV_Design3_92_38.Rdata")

## Design 5
load(file = "/media/EPGE/Docs/dados/design5.Rdata")
c_a_mc <- MonteCarloCV_T(S, alpha, rho1, sigma2, M, NULL, FALSE)
save(c_a_mc, file = "/media/EPGE/Docs/dados/MC_CV_Design5_a.Rdata")
c_a_mc <- MonteCarloCV_T(S, alpha, rho1, sigma2, M, NULL, FALSE)
save(c_a_mc, file = "/media/EPGE/Docs/dados/MC_CV_Design5_b.Rdata")
c_a_mc <- MonteCarloCV_T(S, alpha, rho1, sigma2, M, NULL, FALSE)
save(c_a_mc, file = "/media/EPGE/Docs/dados/MC_CV_Design5_c.Rdata")

## FIM!!

### Listing 34: art1 MonteCarlo CV Value R

1. # Simulate the distribution of the various designs
2. # to calculate the critical Values for the LUMP and LUMPY tests
3. # saves the results for future reference
4. rm(list = ls()) # Apaga todas variáveis da memória
5. source("/media/EPGE/Docs/programas/all_functions.R")
6. ps.options(horizontal = FALSE, onefile = FALSE, height = 4.5,
7.     width = 6, paper = "special", pointsize = 10)
8. S <- 49999
9. alpha <- 0.05
10. #sigma2 <- 1
11. rho1 <- 0  # LUMPY Test
12. ## Design 1 - (Mn)
13. n <- 30
14. c <- 2
15. c_a_mc <- MonteCarloCV_V(S, alpha, rho1, M, NULL, FALSE)
16. save(c_a_mc, file = "/media/EPGE/Docs/dados/MC_CV_V_Design1_30_2.Rdata")
17. c <- 5
18. c_a_mc <- MonteCarloCV_V(S, alpha, rho1, M, NULL, FALSE)
19. save(c_a_mc, file = "/media/EPGE/Docs/dados/MC_CV_V_Design1_30_5.Rdata")
20. c <- 9
21. c_a_mc <- MonteCarloCV_V(S, alpha, rho1, M, NULL, FALSE)
22. save(c_a_mc, file = "/media/EPGE/Docs/dados/MC_CV_V_Design1_30_9.Rdata")
23. n <- 92
24. c <- 2
25. c_a_mc <- MonteCarloCV_V(S, alpha, rho1, M, NULL, FALSE)
26. save(c_a_mc, file = "/media/EPGE/Docs/dados/MC_CV_V_Design1_92_2.Rdata")
27. c <- 5
28. c_a_mc <- MonteCarloCV_V(S, alpha, rho1, M, NULL, FALSE)
29. save(c_a_mc, file = "/media/EPGE/Docs/dados/MC_CV_V_Design1_92_5.Rdata")
m<9
M<genDesign2(n,m)
ca_mc<-MonteCarloCV_V(S, alpha, rho1, M, NULL, FALSE)
save(ca_mc, file="/media/EPGE/Docs/dados/MC_V_Design1_92_9.Rdata")
## Design 2 - (Tn)
	n<30
m<2
M<genDesign2(n,m)
ca_mc<-MonteCarloCV_V(S, alpha, rho1, M, NULL, FALSE)
save(ca_mc, file="/media/EPGE/Docs/dados/MC_V_Design2_30_2.Rdata")

m<5
M<genDesign2(n,m)
ca_mc<-MonteCarloCV_V(S, alpha, rho1, M, NULL, FALSE)
save(ca_mc, file="/media/EPGE/Docs/dados/MC_V_Design2_30_5.Rdata")

m<9
M<genDesign2(n,m)
ca_mc<-MonteCarloCV_V(S, alpha, rho1, M, NULL, FALSE)
save(ca_mc, file="/media/EPGE/Docs/dados/MC_V_Design2_30_9.Rdata")

m<92
m<2
M<genDesign2(n,m)
ca_mc<-MonteCarloCV_V(S, alpha, rho1, M, NULL, FALSE)
save(ca_mc, file="/media/EPGE/Docs/dados/MC_V_Design2_92_2.Rdata")

m<5
M<genDesign2(n,m)
ca_mc<-MonteCarloCV_V(S, alpha, rho1, M, NULL, FALSE)
save(ca_mc, file="/media/EPGE/Docs/dados/MC_V_Design2_92_5.Rdata")

m<9
M<genDesign2(n,m)
ca_mc<-MonteCarloCV_V(S, alpha, rho1, M, NULL, FALSE)
save(ca_mc, file="/media/EPGE/Docs/dados/MC_V_Design2_92_9.Rdata")

## Design3
# Design 5a has 450 non zeros, 5b has 940, 5c has 1662
# the values of J will be chosen to match the number of non-zeros
# on these designs. For smaller n=30, it will be chosen to match the average

m<30
J<round((sqrt(1+4*(2*n+450/92-2*(n-1)))-1)/2,0)
m<(2*(n-1)+(J-1)*(J+2))/2m)
M<genDesign3(n,m,J)
ca_mc<-MonteCarloCV_V(S, alpha, rho1, M, NULL, FALSE)
save(ca_mc, file="/media/EPGE/Docs/dados/MC_V_Design3_30.Rdata")

J<round((sqrt(1+4*(2*n+940/92-2*(n-1)))-1)/2,0)
m<(2*(n-1)+(J-1)*(J+2))/2m)
M<genDesign3(n,m,J)
ca_mc<-MonteCarloCV_V(S, alpha, rho1, M, NULL, FALSE)
save(ca_mc, file="/media/EPGE/Docs/dados/MC_V_Design3_30_15.Rdata")

J<round((sqrt(1+4*(2*n+1662/92-2*(n-1)))-1)/2,0)
m<(2*(n-1)+(J-1)*(J+2))/2m)
M<genDesign3(n,m,J)
ca_mc<-MonteCarloCV_V(S, alpha, rho1, M, NULL, FALSE)
save(ca_mc, file="/media/EPGE/Docs/dados/MC_V_Design3_30_22.Rdata")

m<92
m<92
J<round((sqrt(1+4*(2*n+450/92-2*(n-1)))-1)/2,0)
m<(2*(n-1)+(J-1)*(J+2))/2m)
M<genDesign3(n,m,J)
ca_mc<-MonteCarloCV_V(S, alpha, rho1, M, NULL, FALSE)
save(ca_mc, file="/media/EPGE/Docs/dados/MC_V_Design3_92.Rdata")

J<round((sqrt(1+4*(2*n+940/92-2*(n-1)))-1)/2,0)
m<(2*(n-1)+(J-1)*(J+2))/2m)
M<genDesign3(n,m,J)
ca_mc<-MonteCarloCV_V(S, alpha, rho1, M, NULL, FALSE)
save(ca_mc, file="/media/EPGE/Docs/dados/MC_V_Design3_92_16.Rdata")

J<round((sqrt(1+4*(2*n+1662/92-2*(n-1)))-1)/2,0)
m<(2*(n-1)+(J-1)*(J+2))/2m)
M<genDesign3(n,m,J)
ca_mc<-MonteCarloCV_V(S, alpha, rho1, M, NULL, FALSE)
save(ca_mc, file="/media/EPGE/Docs/dados/MC_V_Design3_92_30.Rdata")
ca_mc<-MonteCarloCV(S, alpha, rho1, M, NULL, FALSE)
save(ca_mc, file="/media/EPGE/Docs/dados/MC.CV.V_Design3_92_27.Rdata")

J<-round((sqrt((1+4*(2+n)*1662/92-2*(n-1)))-1)/2,0)
m<-((n-1)+(J-1)*(J+2))/(2*n)
M<-genDesign3(n,m)
ca_mc<-MonteCarloCV(S, alpha, rho1, M, NULL, FALSE)
save(ca_mc, file="/media/EPGE/Docs/dados/MC.CV.V_Design3_92_37.Rdata")

## Design 5
load(file="/media/EPGE/Docs/dados/design5.Rdata")
M<-design5a
ca_mc<-MonteCarloCV(S, alpha, rho1, M, NULL, FALSE)
save(ca_mc, file="/media/EPGE/Docs/dados/MC.CV.V_Design5_a.Rdata")

M<-design5b
calculate
ca_mc<-MonteCarloCV(S, alpha, rho1, M, NULL, FALSE)
save(ca_mc, file="/media/EPGE/Docs/dados/MC.CV.V_Design5_b.Rdata")

M<-design5c
ca_mc<-MonteCarloCV(S, alpha, rho1, M, NULL, FALSE)
save(ca_mc, file="/media/EPGE/Docs/dados/MC.CV.V_Design5_c.Rdata")

## FIM!!

## Generates an extra data set for n=500
S<-49999
alpha<-0.05
sig2<-1
rho1<-0

## Design 1 - (Vn)
n<-500
m<-2
M<-genDesign1(n,m)
ca_mc<-MonteCarloCV(S, alpha, rho1, M, NULL, FALSE)
save(ca_mc, file="/media/EPGE/Docs/dados/MC.CV.V_Design1_500_2.Rdata")

m<-5
M<-genDesign1(n,m)
ca_mc<-MonteCarloCV(S, alpha, rho1, M, NULL, FALSE)
save(ca_mc, file="/media/EPGE/Docs/dados/MC.CV.V_Design1_500_5.Rdata")

m<-9
M<-genDesign1(n,m)
ca_mc<-MonteCarloCV(S, alpha, rho1, M, NULL, FALSE)
save(ca_mc, file="/media/EPGE/Docs/dados/MC.CV.V_Design1_500_9.Rdata")

## Design 2 - (Vn)
n<-500
m<-2
M<-genDesign2(n,m)
ca_mc<-MonteCarloCV(S, alpha, rho1, M, NULL, FALSE)
save(ca_mc, file="/media/EPGE/Docs/dados/MC.CV.V_Design2_500_2.Rdata")

m<-5
M<-genDesign2(n,m)
ca_mc<-MonteCarloCV(S, alpha, rho1, M, NULL, FALSE)
save(ca_mc, file="/media/EPGE/Docs/dados/MC.CV.V_Design2_500_5.Rdata")

m<-9
M<-genDesign2(n,m)
ca_mc<-MonteCarloCV(S, alpha, rho1, M, NULL, FALSE)
save(ca_mc, file="/media/EPGE/Docs/dados/MC.CV.V_Design2_500_9.Rdata")

## Design 3
# Design 5a has 450 non zeros, 5b has 940, 5c has 1662
# the values of J will be chosen to match the number of non-zeros
# on these designs. For smaller n=50, it will be chosen to match the average
n<-500
J<-round((sqrt((1+4*(2+n)*450/92-2*(n-1)))-1)/2,0)
m<-((n-1)+(J-1)*(J+2))/(2*n)
M <- genDesign3(n,m,J)
cm <- MonteCarloCV(S, alpha, rho1, M, NULL, FALSE)
save(cm, file="/media/EPGE/Docs/dados/MC_CV_Design3_500_38.Rdata")

J <- round((1+4*2*n^940/92-2*(n-1))/2,0)
m <- (2*(n-1)+(J-1)*(J+2))/(2*m)
M <- genDesign3(n,m,J)
cm <- MonteCarloCV(S, alpha, rho1, M, NULL, FALSE)
save(cm, file="/media/EPGE/Docs/dados/MC_CV_Design3_500_64.Rdata")

J <- round((1+4*2*n^1662/92-2*(n-1))/2,0)
m <- (2*(n-1)+(J-1)*(J+2))/(2*m)
M <- genDesign3(n,m,J)
cm <- MonteCarloCV(S, alpha, rho1, M, NULL, FALSE)
save(cm, file="/media/EPGE/Docs/dados/MC_CV_Design3_500_89.Rdata")

Listing 35: art1 power at some infodistance points.R

rm(list=ls()) # Apaga todas variáveis da memória
source("/media/EPGE/Docs/programas/all_functions.R")
x11.options(width=5, height=5, type="ncairo")
ps.options(horizontal=FALSE, onefile=FALSE, height=4.5,
width=6, paper="special", pointsize=10)

## Design 1 - (Tn)
n <- 92
M1 <- genDesign1(n,2)
M2 <- genDesign1(n,5)
M3 <- genDesign1(n,9)
rho1 <- ((1/eigenRange(M1))[2]) + 0.5
rho2 <- ((1/eigenRange(M2))[2]) + 0.5
rho3 <- ((1/eigenRange(M3))[2]) + 0.5
FisherInfoRho(rho1, M1, NULL)
FisherInfoRho(rho2, M2, NULL)
FisherInfoRho(rho3, M3, NULL)

f1 <- function(c,M) sqrt(sapply(c,FisherInfoRho,M))
print(round(integrate(f1,lower=0,upper=rho1,M1$Yval,1)))
print(round(integrate(f1,lower=0,upper=rho2,M2$Yval,1)))
print(round(integrate(f1,lower=0,upper=rho3,M3$Yval,1)))

Listing 36: art1 powerenvelope design1 changing m.R

rm(list=ls()) # Apaga todas variáveis da memória
source("/media/EPGE/Docs/programas/all_functions.R")
ps.options(horizontal=FALSE, onefile=FALSE, height=4.5,
width=6, paper="special", pointsize=10)

# Plot power Envelopes for circular world design with n=92, m=2, m=5, m=9.
epsilon <- 0.001 # defines an arbitrarily small value
alpha <- 0.05 # size alpha test (limits the type I error)
n <- 92 # n x n matrix
len.rho.range<-101
m.range <- c(2,5,9)
rho.range<-array(NA, dim=c(1, length(m.range), len.rho.range))
powerf <- array(NA, dim=c(1, length(m.range), len.rho.range))
rr<-c(seq(-0.99,-sqrt(epsilon), length.out=50),0,seq(sqrt(epsilon),0.99,length.out=50))
for (i in 1:length(m.range)) {
  M <- genDesign1(n,m.range[i])
  maxrho<-1/(2*m.range[i])
  rho.range[i,1] <- rr*maxrho
  powerf[i,1,] <- powerE4_V(rho.range [i,], alpha, M, NULL, TRUE)
}
for (i in 1:l) for (j in 1:length(m.range))
assign("pwe", i, j, sep = "", approxfun(rr, powerf[i, j], ))

## Design 1
postscript("/media/EPGE/Docs/figures/art1_chart07.eps")
curve(pue1, -0.99, 0.99, ylim = c(0, 1), type = "l", lty = 1,
lab = expression(rho[1]), ylab = expression(beta(rho[1])))
curve(pue2, -0.99, 0.99, lty = 2, add = TRUE)
curve(pue3, -0.99, 0.99, lty = 3, add = TRUE)

legend.text <- c(paste("Matrix", n, "x", n, "with m=", m.range[1]),
paste("Matrix with m="
, m.range[-1]))

legend("bottomleft", legend = legend.text, lty = 1:length(m.range), bty = "n")
dev.off()

Listing 37: art1 power envelope Edgeworth.R
\[\text{rho.range}[i, 1:(\text{len.rho.range}/2)]\] <- c(0, seq(from=sqrt(\text{epsilon}), to=\text{rr}[2]-sqrt(\text{epsilon}),
len=\text{len.rho.range}/2-1))

\text{fisher.info}[4, i, 1:(\text{len.rho.range}/2)]\) <- \text{sapply}(\text{rho.range}[4, i, 1:(\text{len.rho.range}/2)],
\text{FisherInfoRho}, M[[4]], M[[6]]))

# fisher.at.zero[4, i, 1] <- FisherInfoRho(0, M[[4]])

\text{power}[1, i, j] <- \text{power4.V}(\text{rho.range}[1, i, j], \text{alpha}, M[[1]], \text{NULL}, \text{TRUE})

\text{power}[2, i, j] <- \text{power4.V}(\text{rho.range}[2, i, j], \text{alpha}, M[[2]], \text{NULL}, \text{TRUE})

\text{power}[3, i, j] <- \text{power4.V}(\text{rho.range}[3, i, j], \text{alpha}, M[[3]], \text{NULL}, \text{TRUE})

\text{power}[4, i, j] <- \text{power4.V}(\text{rho.range}[4, i, j], \text{alpha}, M[[4]], M[[6]], \text{TRUE})

\text{save}(\text{rho.range}, \text{power}, \text{fisher.info}, \text{len.rho.range}, \text{n.range}, z, j, \text{epsilon}, \text{alpha},
\text{file}="/\text{media/EPGE/Docs/dados/powerenvelope\_edgeworth\_V\_Rdata}\)"

# Functions to calculate the square root of Fisher info (fi)
# and the information distance (id), i.e., the integral from the null
# to the alternative.

# obs: Note that design 2 has a different normalizing factor to obtain the average info

\text{fi11} <- \text{approxfun}(\text{rho.range}[1, 1, .], \sqrt{\text{fisher.info}[1, 1, /n.range[1]])

\text{id11} <- \text{function}(\text{rho}) \text{ integrate}(\text{fi11, 0, rho}) \text{ value}

\text{id11f} <- \text{function}(\text{rho}) \text{ sapply}(\text{rho}, \text{id11})

\text{fi12} <- \text{approxfun}(\text{rho.range}[1, 2, .], \sqrt{\text{fisher.info}[1, 2, /n.range[2]])

\text{id12} <- \text{function}(\text{rho}) \text{ integrate}(\text{fi12, 0, rho}) \text{ value}

\text{id12f} <- \text{function}(\text{rho}) \text{ sapply}(\text{rho}, \text{id12})

\text{fi13} <- \text{approxfun}(\text{rho.range}[1, 3, .], \sqrt{\text{fisher.info}[1, 3, /n.range[3]])

\text{id13} <- \text{function}(\text{rho}) \text{ integrate}(\text{fi13, 0, rho}) \text{ value}

\text{id13f} <- \text{function}(\text{rho}) \text{ sapply}(\text{rho}, \text{id13})

\text{fi21} <- \text{approxfun}(\text{rho.range}[2, 1, .], \sqrt{\text{fisher.info}[2, 1, /n.range[1]))

\text{id21} <- \text{function}(\text{rho}) \text{ integrate}(\text{fi21, 0, rho}) \text{ value}

\text{id21f} <- \text{function}(\text{rho}) \text{ sapply}(\text{rho}, \text{id21})

\text{fi22} <- \text{approxfun}(\text{rho.range}[2, 2, .], \sqrt{\text{fisher.info}[2, 2, /n.range[2]])

\text{id22} <- \text{function}(\text{rho}) \text{ integrate}(\text{fi22, 0, rho}) \text{ value}

\text{id22f} <- \text{function}(\text{rho}) \text{ sapply}(\text{rho}, \text{id22})

\text{fi23} <- \text{approxfun}(\text{rho.range}[2, 3, .], \sqrt{\text{fisher.info}[2, 3, /n.range[3]])

\text{id23} <- \text{function}(\text{rho}) \text{ integrate}(\text{fi23, 0, rho}) \text{ value}

\text{id23f} <- \text{function}(\text{rho}) \text{ sapply}(\text{rho}, \text{id23})

\text{fi31} <- \text{approxfun}(\text{rho.range}[3, 1, .], \sqrt{\text{fisher.info}[3, 1, /n.range[1])}

\text{id31} <- \text{function}(\text{rho}) \text{ integrate}(\text{fi31, 0, rho}) \text{ value}

\text{id31f} <- \text{function}(\text{rho}) \text{ sapply}(\text{rho}, \text{id31})

\text{fi32} <- \text{approxfun}(\text{rho.range}[3, 2, .], \sqrt{\text{fisher.info}[3, 2, /n.range[2]])

\text{id32} <- \text{function}(\text{rho}) \text{ integrate}(\text{fi32, 0, rho}) \text{ value}

\text{id32f} <- \text{function}(\text{rho}) \text{ sapply}(\text{rho}, \text{id32})

\text{fi33} <- \text{approxfun}(\text{rho.range}[3, 3, .], \sqrt{\text{fisher.info}[3, 3, /n.range[3]])

\text{id33} <- \text{function}(\text{rho}) \text{ integrate}(\text{fi33, 0, rho}) \text{ value}

\text{id33f} <- \text{function}(\text{rho}) \text{ sapply}(\text{rho}, \text{id33})

\text{fi41} <- \text{approxfun}(\text{rho.range}[4, 1, .], \sqrt{\text{fisher.info}[4, 1, /n.range[1]])

\text{id41} <- \text{function}(\text{rho}) \text{ integrate}(\text{fi41, 0, rho}) \text{ value}

\text{id41f} <- \text{function}(\text{rho}) \text{ sapply}(\text{rho}, \text{id41})

\text{fi42} <- \text{approxfun}(\text{rho.range}[4, 2, .], \sqrt{\text{fisher.info}[4, 2, /n.range[2]])

\text{id42} <- \text{function}(\text{rho}) \text{ integrate}(\text{fi42, 0, rho}) \text{ value}

\text{id42f} <- \text{function}(\text{rho}) \text{ sapply}(\text{rho}, \text{id42})

\text{fi43} <- \text{approxfun}(\text{rho.range}[4, 3, .], \sqrt{\text{fisher.info}[4, 3, /n.range[3]])

\text{id43} <- \text{function}(\text{rho}) \text{ integrate}(\text{fi43, 0, rho}) \text{ value}

\text{id43f} <- \text{function}(\text{rho}) \text{ sapply}(\text{rho}, \text{id43})

\text{for (i in 1:4) for (j in 1:length(n.range))
\text{assign(paste("pve", i, j, sep=""), approxfun(rho.range[i, j], power[i, j]))
                      for (i in 1:3) for (j in 1:length(n.range))
\text{evaluate(paste("id", i, j, "f", sep=""), rho.range[i, j]))}
Listing 38: art1 powerenvelope Edgeworth vs Saddlepoint.R

```r
## Comparing Saddlepoint and Edgeworth
rm(list=ls()) # Apaga todas variáveis da memória
source("/media/EPGE/Docs/programas/all_functions.R")
X11.options(width=5,height=5, type="pdfcairo")
ps.options(horizontal=FALSE, onefile=FALSE, height=4.5, width=6, paper="special", pointsize=10)

# Loads the saved runs
#saddlepoint
load("/media/EPGE/Docs/dados/powerenvelope_saddlepoint_V.Rdata")
#Returns: rho_range,power,fisher.info,len.rho.range,n.range,m,J,M,epsilon,alpha,
#Creates power envelope functions (only for the smallest sample sizes)
for (i in 1:4) for (j in 1:1)
assign(paste("pwe",i,j,"_sp","sep=""), approxfun(rho.range[i,j],power[i,j],))

# Edgeworth
load("/media/EPGE/Docs/dados/powerenvelope_edgeworth_V.Rdata")
#Returns: rho_range,power,fisher.info,len.rho.range,n.range,m,J,M,epsilon,alpha,
#Creates power envelope functions (only for the smallest sample sizes)
for (i in 1:4) for (j in 1:1)
assign(paste("pwe",i,j,"_ew","sep=""), approxfun(rho.range[i,j],power[i,j],))

## Design 1
postscript("/media/EPGE/Docs/figures/art1_chart10a.eps")
layout(matrix(c(1,2,2,1),2))
xlim<range(rho.range[1,])
curve(pwe1_ev,xlim[1],xlim[2],ylim=c(0,1),type="l",lty=1,xlab=expression(rho[1]), ylab= expression(beta(rho[1])))
curve(pwe1_sp,xlim[1],xlim[2],lty=2,add=TRUE)
mtext("Edgeworth app.", adj=0)
mtext("Saddlepoint app.", col="red", adj=1)
plot(function(x) return(pwe1_ev(x)-pwe1_sp(x)),xlim[1],xlim[2],main="Edgeworth - Saddlepoint",
xlab=expression(rho[1]),ylab="Power difference",lty=1)
abline(h=0,lty=2)
legend("bottomleft", legend="Edgeworth approximation","Saddlepoint approximation")
legend("bottomleft", legend=legend.text, lty=1:2, bty="n")
dev.off()

layout(matrix(c(1,2,2,1),2))
plot(function(a) Ed((a-k[1])/sqrt(k[2]),ki), xrange[1], xrange[2], main="Edgeworth Approx", xlab="x", ylab="F(x)", ylim=c(0,1))
curve(pnorm(x,mean=k[1],sd=sqrt(k[2])), add=TRUE, col="red", lwd=2)
plot(function(a) Ed((a-k[1])/sqrt(k[2]),ki)-pnorm(a,mean=k[1],sd=sqrt(k[2])),
   xrange[1], xrange[2], main="Edgeworth - Gaussian", xlab="x",
   ylab="Distance")

## Design 2
postscript("/media/EPGE/Docs/figures/art1_chart10b.eps")
xlim<range(rho.range[2,])
```

Listing 39: art1 powerenvelope Saddlepoint MonteCarlo.R

```r
# Calculate the TRUE power envelope for some specific points in the alternative
rm(list=ls()) # Apaga todas variáveis da memória
source("/media/EPGE/Docs/programas/all_functions.R")

## Table for Design 5 with n=92, alpha = 5% for Vn
alpha=0.05
load("/media/EPGE/Docs/dados/design5.Rdata") # returns design5
designlist=c("a","b","c")
S<49999 # number of simulations
len.rho.range<21
e.values<-array(NA,dim=c(3,len.rho.range,2)) # matrix to store the empirical values.
rho.range<-array(NA,dim=c(3,len.rho.range))
cvalue<-array(NA,dim=c(3,1))
for (i in 1:3) {
  ca<-vector("list",7)
dc<-designlist[i]
MC<design5[[1]]
rr<-1/eigenRange(M)
rho.range[i,]<-(seq(from=0,to=rr[2]-sqrt(epsilon),len=len.rho.range)
load(paste("/media/EPGE/Docs/dados/MC_CV_V_Deposit5",d,".Rdata",sep="")) # returns ca_mc
truecdf<-ecdf(ca_mc[[1]])
Nsim<-length(ca_mc[[1]])
cvalue[i,1]<-sort(ca_mc[[1]],decreasing=TRUE)[alpha*Nsim]
for (j in 1:len.rho.range) e.values[i,j]<-MonteCarloPower_V(S,rho.range[i,j],M,NULL)
}
save(rho.range, e.values, file="/media/EPGE/Docs/dados/MC_ECDF_V_Deposit5.Rdata")
```

Listing 40: art1 powerenvelope Saddlepoint.R

```r
# Power envelope for circular world design
# with n=2 for different values of n
# using saddlepoint approx
# uses BNJ Approximation to obtain (i) the critical values and
# (ii) to calculate the tail probabilities.
# The reason is that it performed better than the other alternatives
# in some simulation -- see paper
```
rm(list=ls()) # Apaga todas variáveis da memória
source("/media/EPGE/Docs programas/all_functions.R")
X11.options(width=5, height=5, type="ncairo")
ps.options(horizontal=FALSE, onefile=FALSE, height=4.5, width=6, paper="special", pointsize=10)

# Begin program
epsilon <- .Machine$double.eps ^ 0.5 # defines an arbitrarily small value
#max <- 4 # max cumulative to be calculated (4 for 0(1/n), 5 for 0(1/n^2)
alpha <- 0.05 # size alpha test (limits the type I error)

m <- Calculate the power envelope for m=2 and different n
m < 2 # with m-ahead and m-behind neighbors

n.range <- c(30, 92, 200)
len. rho.range <- 100 # HAS to be EVEN
rho.range <- array(N, dim=c(4, length(n.range), len. rho.range))
M <- vector("list"), 5)

power <- array(N, dim=c(4, length(n.range), len. rho.range)) # matrix to
#store the power calculated on each rho in rho.range
fisher.info <- array(N, dim=c(4, length(n.range), len. rho.range))

for (i in 1:length(n.range)) {
M[1][i] <- genDesign1(n.range[i], m)
M[2][i] <- genDesign2(n.range[i], m)
M[3][i] <- genDesign3(n.range[i], 2 * (n.range[i] - 1) + (J - 1) * (J - 2)) / 2 * n.range[i] / 1,)
M[4][i] <- genDesign4_a(n.range[i], m)
M[5][i] <- genDesign4_b(n.range[i], m)
vec <- eigenRange(M[[1]])
rho.range[1, i,] <- c(seq(from=rr[1] + sqrt(epsilon), to=-sqrt(epsilon), len=len. rho.range / 2)
0,
seq(from=sqrt(epsilon), to=rr[2] - sqrt(epsilon), len=len. rho.range / 2 - 1))
fisher.info[1, i,] <- sapply(rho.range[1, i,], FisherInfoRho, M[[1]], NULL)

vec <- eigenRange(M[[2]])
rho.range[2, i,] <- c(seq(from=rr[1] + sqrt(epsilon), to=-sqrt(epsilon), len=len. rho.range / 2)
0,
seq(from=sqrt(epsilon), to=rr[2] - sqrt(epsilon), len=len. rho.range / 2 - 1))
fisher.info[2, i,] <- sapply(rho.range[2, i,], FisherInfoRho, M[[2]], NULL)

vec <- eigenRange(M[[3]])
rho.range[3, i,] <- c(seq(from=rr[1] + sqrt(epsilon), to=-sqrt(epsilon), len=len. rho.range / 2)
0,
seq(from=sqrt(epsilon), to=rr[2] - sqrt(epsilon), len=len. rho.range / 2 - 1))
fisher.info[3, i,] <- sapply(rho.range[3, i,], FisherInfoRho, M[[3]], NULL)

vec <- c(0, (sqrt(144/m) - 1)/2)
rho.range[4, i,] <- c(seq(from=sqrt(epsilon), to=rr[2] - sqrt(epsilon), len=len. rho.range / 2 - 1))
fisher.info[4, i,] <- sapply(rho.range[4, i,], FisherInfoRho, M[[4]], M[[5]],)

power[1, i,] <- powerBNJ.V(rho.range[1, i,], alpha, M[[1]], NULL, TRUE)
power[2, i,] <- powerBNJ.V(rho.range[2, i,], alpha, M[[2]], NULL, TRUE)
power[3, i,] <- powerBNJ.V(rho.range[3, i,], alpha, M[[3]], NULL, TRUE)
power[4, i,] <- powerBNJ.V(rho.range[4, i,], alpha, M[[4]], M[[5]], TRUE)
}

save(rho.range, power, fisher.info, len. rho.range, n.range, m, J, epsilon, alpha,
file="/media/EPGE/Docs dados/powerenvelope_saddlepoint_V.Rdata")

# Functions to calculate the square root of Fisher info (f1)
# and the information distance (id), i.e., the integral from the null
# to the alternative
# obs: Note that design 2 has a different normalizing factor to obtain the average info
f11_sp <- approxfun(rho.range[1, i,], sqrt(fisher.info[1, i,] / n.range[i]))
id11_sp <- function(rho) integrate(f11_sp, 0, rho)$value
id11_sp <- function(rho) sapply(rho, id11_sp)
f12_sp <- approxfun(rho.range[1, 2, ], sqrt(fisher.info[1, 2,] / n.range[2]))
id12_sp <- function(rho) integrate(f12_sp, 0, rho)$value
id12f_sp <- function(rho) sapply(rho, id12_sp)
f13_sp<-approxfun(rho.range[1,3], sqrt(fisher.info[1,3]/n.range[3]))
id13_sp<-function(rho) integrate(f13_sp, 0, rho)

d13f_sp<-function(rho) sapply(rho, id13_sp)

f21_sp<-approxfun(rho.range[2,1], sqrt(fisher.info[2,1]/n.range[1]))
id21_sp<-function(rho) integrate(f21_sp, 0, rho)

d21f_sp<-function(rho) sapply(rho, id21_sp)

f22_sp<-approxfun(rho.range[2,2], sqrt(fisher.info[2,2]/n.range[2]))
id22_sp<-function(rho) integrate(f22_sp, 0, rho)

d22f_sp<-function(rho) sapply(rho, id22_sp)

f31_sp<-approxfun(rho.range[3,1], sqrt(fisher.info[3,1]/n.range[1]))
id31_sp<-function(rho) integrate(f31_sp, 0, rho)

d31f_sp<-function(rho) sapply(rho, id31_sp)

f32_sp<-approxfun(rho.range[3,2], sqrt(fisher.info[3,2]/n.range[2]))
id32_sp<-function(rho) integrate(f32_sp, 0, rho)

d32f_sp<-function(rho) sapply(rho, id32_sp)

f41_sp<-approxfun(rho.range[4,1], sqrt(fisher.info[4,1]/n.range[1]))
id41_sp<-function(rho) integrate(f41_sp, 0, rho)

d41f_sp<-function(rho) sapply(rho, id41_sp)

f42_sp<-approxfun(rho.range[4,2], sqrt(fisher.info[4,2]/n.range[2]))
id42_sp<-function(rho) integrate(f42_sp, 0, rho)

d42f_sp<-function(rho) sapply(rho, id42_sp)

f43_sp<-approxfun(rho.range[4,3], sqrt(fisher.info[4,3]/n.range[3]))
id43_sp<-function(rho) integrate(f43_sp, 0, rho)

d43f_sp<-function(rho) sapply(rho, id43_sp)

for (i in 1:4) for (j in 1:length(n.range))
assign(paste("pwe","i",","_sp","sep=""), approxfun(rho.range[i,j], power[i,j,]))

for (i in 1:3) for (j in 1:length(n.range)) {
  x_id<-eval(call(paste("id","i",","_sp",sep=""),rho.range[i,j,]))
  assign(paste("pwe.id","i",","_sp",sep=""), approxfun(x_id,power[i,j,]))
}

1<-4
for (j in 1:length(n.range)) {
  x_id<-eval(call(paste("id","i",","_sp",sep=""),rho.range[i,j,1:(len.rho.range/2)]))
  assign(paste("pwe.id","i",","_sp",sep=""), approxfun(x_id,power[i,j,1:(len.rho.range/2)]))
}

# plots all the power envelopes

postscript("/media/EPGE/Docs/figures/art1_chart09a.eps")
xlim<-range(rho.range[1,])
curve(pwe11_sp,xlim[1],xlim[2], ylim=c(0,1), type="l", lty=1,
  xlab=expression(rho[1]), ylab= expression(beta(rho[1])))
curve(pwe12_sp,xlim[1],xlim[2], lty=2, add=TRUE)
curve(pwe21_sp,xlim[1],xlim[2], lty=3, add=TRUE)
legend.text <- c(paste("Matrix", n.range, "x",n.range))
legend("bottomleft", legend=legend.text, lty=1:length(n.range),
bty="n")
dev.off()

# Design 2

tiserviced("/media/EPGE/Docs/figures/art1_chart09b.eps")
xlim<-range(rho.range[2,])
curve(pwe21_sp,xlim[1],xlim[2], ylim=c(0,1), type="l", lty=1,
  xlab=expression(rho[1]), ylab= expression(beta(rho[1])))
Listing 41: art1 power envelope vs info distance.R

```
150 curve(pwe22_sp,xlim[1],xlim[2],lty=2,add=TRUE)
151 curve(pwe23_sp,xlim[1],xlim[2],lty=3,add=TRUE)
152 legend.text <- c(paste("Matrix", n.range, ",n.range"))
153 legend("bottomleft", legend=legend.text, lty=1:length(n.range),
154 bty="n")
155 dev.off()

157 # Design 3
158 postscript("/media/EPGE/Docs/figures/art1_chart09c.eps")
159 xlim.range=rho.range[3,4]
160 curve(pwe31_sp,xlim[1],ylim=c(0,1), type="l", lty=1,
161 xlab="expression(rho[1])", ylab= expression(beta(rho[1])))
162 curve(pwe32_sp,xlim[1],ylim=c(0,1), type="l", lty=2,add=TRUE)
163 curve(pwe33_sp,xlim[1],ylim=c(0,1), type="l", lty=3,add=TRUE)
164 legend.text <- c(paste("Matrix", n.range, ",n.range"))
165 legend("bottomleft", legend=legend.text, lty=1:length(n.range),
166 bty="n")
167 dev.off()

169 # Design 4
170 postscript("/media/EPGE/Docs/figures/art1_chart09d.eps")
171 xlim.range=rho.range[4,5]
172 curve(pwe41_sp,xlim[1],ylim=c(0,1), type="l", lty=1,
173 xlab="expression(rho[1])", ylab= expression(beta(rho[1])))
174 curve(pwe42_sp,xlim[1],ylim=c(0,1), type="l", lty=2,add=TRUE)
175 curve(pwe43_sp,xlim[1],ylim=c(0,1), type="l", lty=3,add=TRUE)
176 legend.text <- c(paste("Matrix", n.range, ",n.range"))
177 legend("bottomleft", legend=legend.text, lty=1:length(n.range),
178 bty="n")
179 dev.off()
```

```
Listing 41: art1 power envelope vs info distance.R

1 rm(list="ls") \# Apaga todas variáveis da memória
2 source("/media/EPGE/Docs/programs/all_functions.R")
3 X11.options(width=5,height=5,type="pbcairo")
4 ps.options(horizontal=FALSE, onefile=FALSE, height=4.5,
5 width=6, paper="special", pointsize=10)
6 # Plot power Envelopes for circular world design with n=100, m=2, m=5, m=10.
7 epsilon <- .Machine$double.eps^0.5 \# defines an arbitrarily small value
8 alpha <- 0.05 \# size alpha test (limits the type I error)
9 m <- 92 \# n x n matrix
10 m.range<-c(2,6,10)
11 J.range<-c(16,27,38)
12 len.rho.range<-150 \# HAS to be EVEN
13 rho.range<array(MA, dim=c(5,length(m.range),len.rho.range))
14 fisher.info<array(MA, dim=c(5,length(m.range),len.rho.range))
15 power< array(MA, dim=c(5,length(m.range),len.rho.range))
16 M<"vector("list",6)
17 load(file="/media/EPGE/Docs/dados/design5.Rdata")
18 M5a<-design5$A
19 M5b<-design5$B
20 M5c<-design5$C
21 for (i in 1:length(m.range)) {
22  M[i]<-genDesign1(m,m.range[i])
23  M[2]<-genDesign2(m,m.range[i])
24  M[3]<-genDesign3(n,2*(n-1)+(J.range[i]-1)/(2*n),J.range[i])
25  M[4]<-genDesign4.a(m,m.range[i])
26  M[5]<-genDesign4.b(m,m.range[i])
27  M[6]<-if (i==1) M5a else if (i==2) M5b else M5c
28  rr<"/eigenRange/M[1][1]"
29  rho.range[1,i]<-c(seq(from=rr[1]+sqrt(epsilon), to=-sqrt(epsilon), len=len.rho.range/2)
30 ,0,
31   seq(from=-sqrt(epsilon), to=rr[2]-sqrt(epsilon), len=len.rho.range/2-1))
32  fisher.info[1,i]<-sapply(rho.range[1,i],FisherInfoRho,M[[1]],NULL)
```
\[
\begin{align*}
&\text{rrc}<-1/
\chi^2(
\text{eigenRange}(\text{M}[[2]])) \\
&\text{rho.range}[2,1,]<-c(seq(from=\text{rrc}+\sqrt{\epsilon}, to=\sqrt{\epsilon}, len=\text{len.rho.range}/2), 0) \\
&\text{seq(from=\sqrt{\epsilon}, to=\text{rrc}-\sqrt{\epsilon}, len=\text{len.rho.range}/2-1)) \\
&\text{fisher.info}[2,1,]<-\text{apply(rho.range}[2,1,], \text{FisherInfoRho}, \text{M}[[2]], \text{NULL}) \\
&\text{rrc}<-1/
\chi^2(
\text{eigenRange}(\text{M}[[3]])) \\
&\text{rho.range}[3,1,]<-c(seq(from=\text{rrc}+\sqrt{\epsilon}, to=\sqrt{\epsilon}, len=\text{len.rho.range}/2), 0) \\
&\text{seq(from=\sqrt{\epsilon}, to=\text{rrc}-\sqrt{\epsilon}, len=\text{len.rho.range}/2-1)) \\
&\text{fisher.info}[3,1,]<-\text{apply(rho.range}[3,1,], \text{FisherInfoRho}, \text{M}[[3]], \text{NULL}) \\
&\text{rrc}<-c(0, (\sqrt{1+4/\text{m.range}[1]})-1)/2 \\
&\text{rho.range}[4,1,1:(\text{len.rho.range}/2)]<-c(0, seq(from=\sqrt{\epsilon}, to=\text{rrc}-\sqrt{\epsilon}, len=\text{len.rho.range}/2-1)) \\
&\text{fisher.info}[4,1,1:(\text{len.rho.range}/2)]<-\text{apply(rho.range}[4,1,1:(\text{len.rho.range}/2)], \text{FisherInfoRho}, \text{M}[[4]], \text{M}[[6]]) \\
&\text{rrc}<-1/
\chi^2(
\text{eigenRange}(\text{M}[[6]])) \\
&\text{rho.range}[5,1,]<-c(seq(from=\text{rrc}+\sqrt{\epsilon}, to=\sqrt{\epsilon}, len=\text{len.rho.range}/2), 0) \\
&\text{seq(from=\sqrt{\epsilon}, to=\text{rrc}-\sqrt{\epsilon}, len=\text{len.rho.range}/2-1)) \\
&\text{fisher.info}[5,1,]<-\text{apply(rho.range}[5,1,], \text{FisherInfoRho}, \text{M}[[6]], \text{NULL}) \\
&\text{powerf}[1,1]<-\text{power4.V}(\text{rho.range}[1,1,], \alpha, \text{M}[[1]], \text{NULL}, \text{TRUE}) \\
&\text{powerf}[2,1]<-\text{power4.V}(\text{rho.range}[2,1,], \alpha, \text{M}[[2]], \text{NULL}, \text{TRUE}) \\
&\text{powerf}[3,1]<-\text{power4.V}(\text{rho.range}[3,1,], \alpha, \text{M}[[3]], \text{NULL}, \text{TRUE}) \\
&\text{powerf}[4,1]<-\text{power4.V}(\text{rho.range}[4,1,1:(\text{len.rho.range}/2)], \alpha, \text{M}[[4]], \text{M}[[6]], \text{TRUE}) \\
&\text{powerf}[5,1]<-\text{power4.V}(\text{rho.range}[5,1,], \alpha, \text{M}[[6]], \text{NULL}, \text{TRUE}) \\
&\text{)} \\
&\text{f111}<-\text{approxfun}(\text{rho.range}[1,1,], \sqrt{\text{fisher.info}[1,1,]/n}) \\
&\text{id111}<\text{function(rho)} \text{ integrate(f111,0, rho)}$value \\
&\text{id111}<\text{function(rho)} \text{ apply(rho, id111) \\
&\text{f112}<\text{approxfun}(\text{rho.range}[1,2,], \sqrt{\text{fisher.info}[1,2,]/n}) \\
&\text{id112}<\text{function(rho)} \text{ integrate(f112,0, rho)}$value \\
&\text{id112}<\text{function(rho)} \text{ apply(rho, id112) \\
&\text{f113}<\text{approxfun}(\text{rho.range}[1,3,], \sqrt{\text{fisher.info}[1,3,]/n}) \\
&\text{id113}<\text{function(rho)} \text{ integrate(f113,0, rho)}$value \\
&\text{id113}<\text{function(rho)} \text{ apply(rho, id113) \\
&\text{f121}<\text{approxfun}(\text{rho.range}[2,1,], \sqrt{\text{fisher.info}[2,1,]/n}) \\
&\text{id211}<\text{function(rho)} \text{ integrate(f121,0, rho)}$value \\
&\text{id211}<\text{function(rho)} \text{ apply(rho, id211) \\
&\text{f122}<\text{approxfun}(\text{rho.range}[2,2,], \sqrt{\text{fisher.info}[2,2,]/n}) \\
&\text{id222}<\text{function(rho)} \text{ integrate(f122,0, rho)}$value \\
&\text{id222}<\text{function(rho)} \text{ apply(rho, id222) \\
&\text{f123}<\text{approxfun}(\text{rho.range}[2,3,], \sqrt{\text{fisher.info}[2,3,]/n}) \\
&\text{id233}<\text{function(rho)} \text{ integrate(f123,0, rho)}$value \\
&\text{id233}<\text{function(rho)} \text{ apply(rho, id233) \\
&\text{f131}<\text{approxfun}(\text{rho.range}[3,1,], \sqrt{\text{fisher.info}[3,1,]/n}) \\
&\text{id311}<\text{function(rho)} \text{ integrate(f131,0, rho)}$value \\
&\text{id311}<\text{function(rho)} \text{ apply(rho, id311) \\
&\text{f132}<\text{approxfun}(\text{rho.range}[3,2,], \sqrt{\text{fisher.info}[3,2,]/n}) \\
&\text{id322}<\text{function(rho)} \text{ integrate(f132,0, rho)}$value \\
&\text{id322}<\text{function(rho)} \text{ apply(rho, id322) \\
&\text{f133}<\text{approxfun}(\text{rho.range}[3,3,], \sqrt{\text{fisher.info}[3,3,]/n}) \\
&\text{id333}<\text{function(rho)} \text{ integrate(f133,0, rho)}$value \\
&\text{id333}<\text{function(rho)} \text{ apply(rho, id333) \\
&\text{f141}<\text{approxfun}(\text{rho.range}[4,1,], \sqrt{\text{fisher.info}[4,1,]/n}) \\
&\text{id411}<\text{function(rho)} \text{ integrate(f141,0, rho)}$value \\
&\text{id411}<\text{function(rho)} \text{ apply(rho, id411) \\
&\text{f142}<\text{approxfun}(\text{rho.range}[4,2,], \sqrt{\text{fisher.info}[4,2,]/n}) \\
&\text{id422}<\text{function(rho)} \text{ integrate(f142,0, rho)}$value \\
\end{align*}

id42f <- function(rho) sapply(rho, id42)

if43 <- approxfun(rho.range[4,3,], sqrt(fisher.info[4,3,]/n))

id43 <- function(rho) integrate(id43, 0, rho)$value

id43f <- function(rho) sapply(rho, id43)

if51 <- approxfun(rho.range[5,1,], sqrt(fisher.info[5,1,]/n))

id51 <- function(rho) integrate(id51, 0, rho)$value

id51f <- function(rho) sapply(rho, id51)

if52 <- approxfun(rho.range[5,2,], sqrt(fisher.info[5,2,]/n))

id52 <- function(rho) integrate(id52, 0, rho)$value

id52f <- function(rho) sapply(rho, id52)

if53 <- approxfun(rho.range[5,3,], sqrt(fisher.info[5,3,]/n))

id53 <- function(rho) integrate(id53, 0, rho)$value

id53f <- function(rho) sapply(rho, id53)

for (i in 1:5) for (j in 1:length(m.range))
assign(paste("pwe",i,j,sep="", approxfun(rho.range[i,j,], power[i,j,]))

for (i in c(1:3,5)) for (j in 1:length(m.range)) {
  x_id <- eval(call(paste("id",i,j,"f",sep=""), rho.range[i,j,]))
  assign(paste("pwe_id",i,j,sep=""), approxfun(x_id, power[i,j,]))
}

i <- 4
for (j in 1:length(m.range)) {
  x_id <- eval(call(paste("id",i,j,"f",sep=""), rho.range[i,j,1:(len.rho.range/2)]))
  assign(paste("pwe_id",i,j,sep=""), approxfun(x_id, power[i,j,1:(len.rho.range/2)]))
}

# Grafico Design 1
postscript("/media/EPGE/Docs/figures/arti_chart08a.eps")
curve(pwe_id1,1,ylim=c(0,1), type="l", lty=1,
  xlab="information distance (id)", ylab=expression(beta(id)))
curve(pwe_id2,1,ylim=c(0,1), type="l", lty=2, add=TRUE)
curve(pwe_id3,1,ylim=c(0,1), type="l", lty=3, add=TRUE)
legend.text <- c(paste("Design 1 (n","m","m.range1,""), paste("m"," m.range[-1]"))
legend("bottomleft", legend=legend.text, lty=1:length(m.range),
  bty="n")
dev.off()

# Grafico Design 2
postscript("/media/EPGE/Docs/figures/arti_chart08b.eps")
curve(pwe_id1,1,ylim=c(0,1), type="l", lty=1,
  xlab="information distance (id)", ylab=expression(beta(id)))
curve(pwe_id2,1,ylim=c(0,1), type="l", lty=2, add=TRUE)
curve(pwe_id3,1,ylim=c(0,1), type="l", lty=3, add=TRUE)
legend.text <- c(paste("Design 2 (n","m","m.range1,""), paste("m"," m.range[-1]"))
legend("bottomleft", legend=legend.text, lty=1:length(m.range),
  bty="n")
dev.off()

# Grafico Design 3
postscript("/media/EPGE/Docs/figures/arti_chart08c.eps")
curve(pwe_id1,1,ylim=c(0,1), type="l", lty=1,
  xlab="information distance (id)", ylab=expression(beta(id)))
curve(pwe_id2,1,ylim=c(0,1), type="l", lty=2, add=TRUE)
curve(pwe_id3,1,ylim=c(0,1), type="l", lty=3, add=TRUE)
legend.text <- c(paste("Design 3 (n","m","J.range1,""), paste("J"," J.range[-1]"))
legend("bottomleft", legend=legend.text, lty=1:length(m.range),
  bty="n")
dev.off()

# Grafico Design 4
postscript("/media/EPGE/Docs/figures/arti_chart08d.eps")
curve(pwe_id1,0,1,ylim=c(0,1), type="l", lty=1,
  xlab="information distance (id)", ylab=expression(beta(id)))
curve(pwe_id2,0,1,ylim=c(0,1), type="l", lty=2, add=TRUE)
curve(pwe_id3,0,1,ylim=c(0,1), type="l", lty=3, add=TRUE)
legend.text <- c(paste("Design 4 (n","m","m.range1,""), paste("m"," m.range[-1]"))
legend("bottomright", legend=legend.text, lty=1:length(m.range),
  bty="n")
dev.off()
```r
# Load necessary libraries
library(ggplot2)
library(grid)
library(gridExtra)

# Set seed for reproducibility
set.seed(123)

# Generate data
n <- 100
rho <- abs(rnorm(n))
sigma <- rnorm(n)

# Calculate correlation matrix
C <- matrix(c(1, rho, rho, 1), nrow = 2)
svdC <- svd(C)

# Plot power envelopes
for (i in 1:length(rho)) {
  M[1] <- genDesign1(n, m.range[i])
  M[2] <- genDesign2(n, m.range[i])
  M[3] <- genDesign3(n, (2 * (n - 1) + 1) * (J.range[i] - 1) * (J.range[i] + 2) / (2 * n))
  M[4] <- genDesign4_a(n, m.range[i])
  M[5] <- genDesign4_b(n, m.range[i])

  # If condition
  if (i == 1) M5a else if (i == 2) M5b else M5c

  # Calculate power
  rho.range <- c(seq(from = rr[1] + sqrt(epsilon), to = sqrt(epsilon), len = len.rho.range / 2),
                 seq(from = -sqrt(epsilon), to = -sqrt(epsilon), len = len.rho.range / 2))
  fisher.info <- array(NA, dim = c(5, length(m.range), len.rho.range))
  power <- array(NA, dim = c(5, length(m.range), len.rho.range))
  vector("list", 6)

  # Load data
  load(file = "~/media/EPGE/Docs/dados/design5.Rdata")

  # Calculate Fisher information
  fisher.info <- apply(rho.range[1, i], 1, function(x) fisherInfoRho, M[1], NULL)
  fisher.info[2, i] <- apply(rho.range[2, i], 1, function(x) fisherInfoRho, M[2], NULL)
  fisher.info[3, i] <- apply(rho.range[3, i], 1, function(x) fisherInfoRho, M[3], NULL)
  fisher.info[4, i, 1:(len.rho.range / 2)] <- apply(rho.range[4, i, 1:(len.rho.range / 2)],
                                                    1, function(x) fisherInfoRho, M[4], M[5])
seq(from = sqrt(epsilon), to = rr[2] - sqrt(epsilon), len = len.rho.range/2 - 1)

fisher.info[5, i, ] <- sapply(rho.range[5, i, ], FisherInfoRho, M[[6]], NULL)

power[1, i, ] <- powerNJ_V(rho.range[1, i, ], alpha, M[[1]], NULL, TRUE)
power[2, i, ] <- powerNJ_V(rho.range[2, i, ], alpha, M[[2]], NULL, TRUE)
power[3, i, ] <- powerNJ_V(rho.range[3, i, ], alpha, M[[3]], NULL, TRUE)
power[4, i, ] <- powerNJ_V(rho.range[4, i, ], (len.rho.range/2), alpha, M[[4]], M[[5]], TRUE)
power[5, i, ] <- powerNJ_V(rho.range[5, i, ], alpha, M[[6]], NULL, TRUE)

save(epsilon, alpha, n, m.range, J.range, len.rho.range, rho.range, fisher.info, power, M, file="/media/EPGE/Docs/dados/powerenvelope_saddlepoint_vs_info_distance.Rdata")

f11 <- approxfun(rho.range[1, ,], sqrt(fisher.info[1, 1, ]/n))
id11 <- function(rho) integrate(f11, 0, rho)$value

f12 <- approxfun(rho.range[1, 2,], sqrt(fisher.info[1, 2, ]/n))
id12 <- function(rho) integrate(f12, 0, rho)$value

f12f <- function(rho) sapply(rho, id12)

f13 <- approxfun(rho.range[1, 3,], sqrt(fisher.info[1, 3, ]/n))
id13 <- function(rho) integrate(f13, 0, rho)$value

f13f <- function(rho) sapply(rho, id13)

f121 <- approxfun(rho.range[2, 1,], sqrt(fisher.info[2, 1, ]/n))
id21 <- function(rho) integrate(f121, 0, rho)$value

f121f <- function(rho) sapply(rho, id21)

f122 <- approxfun(rho.range[2, 2,], sqrt(fisher.info[2, 2, ]/n))
id22 <- function(rho) integrate(f122, 0, rho)$value

f122f <- function(rho) sapply(rho, id22)

f123 <- approxfun(rho.range[2, 3,], sqrt(fisher.info[2, 3, ]/n))
id23 <- function(rho) integrate(f123, 0, rho)$value

f123f <- function(rho) sapply(rho, id23)

f131 <- approxfun(rho.range[3, 1,], sqrt(fisher.info[3, 1, ]/n))
id31 <- function(rho) integrate(f131, 0, rho)$value

f131f <- function(rho) sapply(rho, id31)

f132 <- approxfun(rho.range[3, 2,], sqrt(fisher.info[3, 2, ]/n))
id32 <- function(rho) integrate(f132, 0, rho)$value

f132f <- function(rho) sapply(rho, id32)

f133 <- approxfun(rho.range[3, 3,], sqrt(fisher.info[3, 3, ]/n))
id33 <- function(rho) integrate(f133, 0, rho)$value

f133f <- function(rho) sapply(rho, id33)

f141 <- approxfun(rho.range[4, 1,], sqrt(fisher.info[4, 1, ]/n))
id41 <- function(rho) integrate(f141, 0, rho)$value

f141f <- function(rho) sapply(rho, id41)

f142 <- approxfun(rho.range[4, 2,], sqrt(fisher.info[4, 2, ]/n))
id42 <- function(rho) integrate(f142, 0, rho)$value

f142f <- function(rho) sapply(rho, id42)

f143 <- approxfun(rho.range[4, 3,], sqrt(fisher.info[4, 3, ]/n))
id43 <- function(rho) integrate(f143, 0, rho)$value

f143f <- function(rho) sapply(rho, id43)

f151 <- approxfun(rho.range[5, 1,], sqrt(fisher.info[5, 1, ]/n))
id51 <- function(rho) integrate(f151, 0, rho)$value

f151f <- function(rho) sapply(rho, id51)

f152 <- approxfun(rho.range[5, 2,], sqrt(fisher.info[5, 2, ]/n))
id52 <- function(rho) integrate(f152, 0, rho)$value

f152f <- function(rho) sapply(rho, id52)

f153 <- approxfun(rho.range[5, 3,], sqrt(fisher.info[5, 3, ]/n))
id53 <- function(rho) integrate(f153, 0, rho)$value

f153f <- function(rho) sapply(rho, id53)
for (i in 1:5) for (j in 1:length(m.range))
  assign(paste("pwe_",i,j,sep=""), approxfun(rho.range[i,j],power[i,j]))

for (i in c(1,3,5)) for (j in 1:length(m.range)) {
  x_id_eval(call(paste("x",i,j,"f",sep=""),rho.range[i,j]))
  assign(paste("pwe_id_",i,j,sep=""), approxfun(x_id,power[i,j]))
}

# Gráfico Design 1
postscript("/media/EPGE/Docs/figures/art1_chart1a.eps")
curve(pwe.id11,-1,1,ylim=c(0,1), type="l", lty=1,
     xlab="information distance (id)", ylab=expression(beta(id)))
curve(pwe.id12,-1,1,ylty=2,add=TRUE)
curve(pwe.id13,-1,1,ylty=3,add=TRUE)
legend.text <- c(paste("Design 1 (n","n","m","m.range[1],")"), paste("m","m.range[-1]"))
legend("bottomleft", legend.legend.text, lty=1:length(m.range),
       bty="n")
dev.off()

# Gráfico Design 2
postscript("/media/EPGE/Docs/figures/art1_chart1b.eps")
curve(pwe.id21,-1,1,ylim=c(0,1), type="l", lty=1,
     xlab="information distance (id)", ylab=expression(beta(id)))
curve(pwe.id22,-1,1,ylty=2,add=TRUE)
curve(pwe.id23,-1,1,ylty=3,add=TRUE)
legend.text <- c(paste("Design 2 (n","n","m","m.range[1],")"), paste("m","m.range[-1]"))
legend("bottomleft", legend.legend.text, lty=1:length(m.range),
       bty="n")
dev.off()

# Gráfico Design 3
postscript("/media/EPGE/Docs/figures/art1_chart1c.eps")
curve(pwe.id31,-1,1,ylim=c(0,1), type="l", lty=1,
     xlab="information distance (id)", ylab=expression(beta(id)))
curve(pwe.id32,-1,1,ylty=2,add=TRUE)
curve(pwe.id33,-1,1,ylty=3,add=TRUE)
legend.text <- c(paste("Design 3 (n","n","J","J.range[1],")"), paste("J","J.range[-1]"))
legend("bottomleft", legend.legend.text, lty=1:length(m.range),
       bty="n")
dev.off()

# Gráfico Design 4
postscript("/media/EPGE/Docs/figures/art1_chart1d.eps")
curve(pwe.id41,0,1,ylim=c(0,1), type="l", lty=1,
     xlab="information distance (id)", ylab=expression(beta(id)))
curve(pwe.id42,0,1,ylty=2,add=TRUE)
curve(pwe.id43,0,1,ylty=3,add=TRUE)
legend.text <- c(paste("Design 4 (n","n","m","m.range[1],")"), paste("m","m.range[-1]"))
legend("bottomright", legend.legend.text, lty=1:length(m.range),
       bty="n")
dev.off()

# Gráfico Design 5
postscript("/media/EPGE/Docs/figures/art1_chart1e.eps")
curve(pwe.id51,-1,1,ylim=c(0,1), type="l", lty=1,
     xlab="information distance (id)", ylab=expression(beta(id)))
curve(pwe.id52,-1,1,ylty=2,add=TRUE)
curve(pwe.id53,-1,1,ylty=3,add=TRUE)
legend.text <- c("Design 5a", "b", "c")
legend("bottomleft", legend.legend.text, lty=1:length(m.range),bty="n")
dev.off()

# comparison of different designs for m=2
postscript("/media/EPGE/Docs/figures/art1_chart1f.eps")
curve(pwe.id11,-1,1,ylim=c(0,1), type="l", lty=1,
Listing 43: art1 size tables 10percent.R

# Compare the critical values obtained using different methods
# LUMP test for statistic T_m
rm(list=ls()) # Apaga todas variáveis da memoria
source("/media/EPGE/Dos programs/all_functions.r")

# Table for Design 1 with n=92, alpha = 10% for Tn
Design1_table92 <- matrix(NA, 8, 6)
n <- 92
alpha <- 0.10
mlist <- c(2, 5, 9)
for (i in 1:3) {
c <- vector("list", 7)
mc <- mlist[i]
M <- genDesign1(n, m)
load(paste("/media/EPGE/Dos dados/MC.CV_Design1_" , n, "," , m, ",.Rdata", sep="")) # returns ca_mc
truecdf <- ecdf(ca_mc[[1]])
Nsim <- length(ca_mc[[1]])
ca_true <- sort(ca_mc[[1]], decreasing = TRUE)[alpha*Nsim+1]
c_true_std <- (ca_true-mean(ca_mc[[1]])/sd(ca_mc[[1]]))
Design1_table92[i, 1:1] <- ca_true_std
Design1_table92[i, 2:1] <- 100*(1-truecdf(ca_true)) - ifelse(any(ca_mc[[1]]==ca_true), 1/Nsim, 0)) # actual size
print(ca[[1]]) <- EwqcvT(alpha, 0, M))
print(ca[[2]]) <- CornFqCV_T(alpha, 0, M))
print(ca[[3]]) <- saddlepointCV_T(alpha, 0, M))
print(ca[[4]]) <- saddlepintCV_T(alpha, 0, M))
print(ca[[5]]) <- RAqCV_T(alpha, 0, M))
print(ca[[6]]) <- LRqCV_T(alpha, 0, M))
print(ca[[7]]) <- BJqCV_T(alpha, 0, M))
for (j in 1:7) {
if ((is.na(ca[[j]])$norm)) {
Design1_table92[j, 1:1] <- ca[[j]]$std
Design1_table92[j, 1:1] <- 100*(1-truecdf(ca[[j]])$norm) - ifelse(any(ca_mc[[1]]==ca[[j]]$norm), 1/Nsim, 0)) - Design1_table92[j, 1:1]
}
# rm(ca_mc,M,Nsim, truecdf, ca_true, ca_true_std, ca,m)
save(Design1_table92, file="/media/EPGE/Dos dados/Design1_T_table92_10.Rdata")

## Table for Design 1 with n=30, alpha = 10% for Tn
Design1_table30 <- matrix(NA, 8, 6)
n <- 30
alpha <- 0.10
mlist <- c(2, 5, 9)
for (i in 1:3) {
c <- vector("list", 7)
mc <- mlist[i]
M <- genDesign1(n, m)
load(paste("/media/EPGE/Dos dados/MC.CV_Design1_" , n, "," , m, ",.Rdata", sep="")) # returns ca_mc
truecdf <- ecdf(ca_mc[[1]])
Nsim <- length(ca_mc[[1]])
ca_true <- sort(ca_mc[[1]], decreasing = TRUE)[alpha*Nsim+1]
c_true_std <- (ca_true-mean(ca_mc[[1]])/sd(ca_mc[[1]]))
Design1_table30[i, 1:1] <- ca_true_std
Design1_table30[i, 2:1] <- 100*(1-truecdf(ca_true)) - ifelse(any(ca_mc[[1]]==ca_true), 1/Nsim, 0)) # actual size
print(ca[[1]]) <- EwqcvT(alpha, 0, M))
print(ca[[2]]) <- CornFqCV_T(alpha, 0, M))
print(ca[[3]]) <- saddlepointCV_T(alpha, 0, M))
print(ca[[4]]) <- saddlepintCV_T(alpha, 0, M))
print(ca[[5]]) <- RAqCV_T(alpha, 0, M))
print(ca[[6]]) <- LRqCV_T(alpha, 0, M))
print(ca[[7]]) <- BJqCV_T(alpha, 0, M))
for (j in 1:7) {
if ((is.na(ca[[j]])$norm)) {
Design1_table30[j, 1:1] <- ca[[j]]$std
Design1_table30[j, 1:1] <- 100*(1-truecdf(ca[[j]])$norm) - ifelse(any(ca_mc[[1]]==ca[[j]]$norm), 1/Nsim, 0)) - Design1_table30[j, 1:1]
}
}
# rm(ca_mc,M,Nsim, truecdf, ca_true, ca_true_std, ca,m)
}
save(Design1_table30, file="/media/EPGE/Docs/dados/Design1_T_table30_10.Rdata")

## Table for Design 2 with n=92, alpha = 10% for Tn
Design2_table92 <- matrix(NA, 8, 6)
nc <- 92
alpha <- 0.10
mlist <- c(2, 5, 9)
for (i in 1:3) {
  ca <- vector("list", 7)
  m <- mlist[i]
  M <- genDesign2(n, m)
  load(paste("/media/EPGE/Docs/dados/MC CV/Design2",".n",".m",".Rdata",sep="")) # returns ca_mc
  truedcf <- ecdf(ca_mc[[1]]))
  Nsim <- length(ca_mc[[1]])
  ca_true <- sort(ca_mc[[1]], decreasing = TRUE)[alpha*(Nsim+1)]
  ca_true_std <- (ca_true - mean(ca_mc[[1]]))/sd(ca_mc[[1]])
  Design2_table92[1, 2*i-1] <- ca_true_std
  Design2_table92[1, 2*i] <- 100*(1 - truedcf (ca_true))
  for (j in 1:7) {
    if (!is.na(ca[j])) {
      Design2_table92[j, 1:2*i-1] <- ca[j]
    }
    Design2_table92[j, 1:2*i] <- 100*(1 - truedcf (ca[[j]])
      norm, 1/Nsim, 0)) - Design2_table92[1, 2*i-1])
  }
  rm(ca_mc, M, Nsim, truedcf, ca_true, ca_true_std, ca)
}
save(Design2_table92, file="/media/EPGE/Docs/dados/Design2_T_table92_10.Rdata")

## Table for Design 2 with n=30, alpha = 10% for Tn
Design2_table30 <- matrix(NA, 8, 6)
nc <- 30
alpha <- 0.10
mlist <- c(2, 5, 9)
for (i in 1:3) {
  ca <- vector("list", 7)
  m <- mlist[i]
  M <- genDesign2(n, m)
  load(paste("/media/EPGE/Docs/dados/MC CV/Design2",".n",".m",".Rdata",sep="")) # returns ca_mc
  truedcf <- ecdf(ca_mc[[1]]))
  Nsim <- length(ca_mc[[1]])
  ca_true <- sort(ca_mc[[1]], decreasing = TRUE)[alpha*(Nsim+1)]
  ca_true_std <- (ca_true - mean(ca_mc[[1]]))/sd(ca_mc[[1]])
  Design2_table30[1, 2*i-1] <- ca_true_std
  Design2_table30[1, 2*i] <- 100*(1 - truedcf (ca_true))
  for (j in 1:7) {
    if (!is.na(ca[j])) {
      Design2_table30[j, 1:2*i-1] <- ca[j]
    }
    Design2_table30[j, 1:2*i] <- 100*(1 - truedcf (ca[[j]])
      norm, 1/Nsim, 0)) - Design2_table30[1, 2*i-1])
  }
  rm(ca_mc, M, Nsim, truedcf, ca_true, ca_true_std, ca)
}
save(Design2_table30, file="/media/EPGE/Docs/dados/Design2_T_table30_10.Rdata")
## Table for Design 3 with n=92, alpha = 10% for Tn

Design3_table92 <- matrix(NA, 8, 6)
nc <- 92
alpha <- 0.10
Jlist <- c(16, 17, 18, 19, 20, 32)
for (i in 1:3) {
c <- vector("list", 7)
J <- Jlist[i]
mc <- (2*(n-1)+(J-1)\*(J+2))\ /(2*n)
M <- genDesign3(n, m)
load(paste("/media/EPGE/Docs/dados/MC_CV_Design3_","n","_","J",".Rdata",sep="")) # returns ca_mc
truecdf <- ecdf(ca_mc[[1]])
Nsim <- length(ca_mc[[1]])
ca_true <- sort(ca_mc[[1]], decreasing=TRUE)[alpha*(Nsim+1)]
ca_true_std <- (ca_true-mean(ca_mc[[1]]))/sd(ca_mc[[1]])
Design3_table92[1, 2:1] <- ca_true_std
Design3_table92[1, 2:1] <- 100*(1-truecdf(ca_mc[[1]]==ca_true, 1/Nsim )) # actual size
print(ca[[1]] <- Ev4CV_T(alpha, 0, M))
print(ca[[2]] <- CornFiCV_T(alpha, 0, M))
print(ca[[3]] <- saddlepointCV_T(alpha, 0, M))
print(ca[[4]] <- saddlepointCV_T(alpha, 0, M))
print(ca[[5]] <- RACV_T(alpha, 0, M))
print(ca[[6]] <- LRACV_T(alpha, 0, M))
print(ca[[7]] <- BJACV_T(alpha, 0, M))
for (j in 1:7) {
  if (is.na(ca[[j]])$norm) {
    Design3_table92[j, 1:2:1] <- ca[[j]]$std
    Design3_table92[j, 1:2:1] <- 100*(1-truecdf(ca[[j]]$norm)-1 unless (any(ca_mc[[1]]==ca[[j]]$norm), 1/Nsim )) # actual size
    } else {
    
    }
    # rm(ca_mc,M,Nsim,truecdf,ca_true,ca_true_std,ca,m)
    }
save(Design3_table92, file="/media/EPGE/Docs/dados/Design3_T_table92.10.Rdata"

## Table for Design 3 with n=30, alpha = 10% for Tn

Design3_table30 <- matrix(NA, 8, 6)
nc <- 30
alpha <- 0.10
Jlist <- c(9, 15, 22)

for (i in 1:3) {
c <- vector("list", 7)
J <- Jlist[i]
mc <- (2*(n-1)+(J-1)\*(J+2))\ /(2*n)
M <- genDesign3(n, m)
load(paste("/media/EPGE/Docs/dados/MC_CV_Design3_","n","_","J",".Rdata",sep="")) # returns ca_mc
truecdf <- ecdf(ca_mc[[1]])
Nsim <- length(ca_mc[[1]])
ca_true <- sort(ca_mc[[1]], decreasing=TRUE)[alpha*(Nsim+1)]
ca_true_std <- (ca_true-mean(ca_mc[[1]]))/sd(ca_mc[[1]])
Design3_table30[1, 2:1] <- ca_true_std
Design3_table30[1, 2:1] <- 100*(1-truecdf(ca_true)-1 unless (any(ca_mc[[1]]==ca_true, 1/Nsim )) # actual size
    } else {
    
    }
    # rm(ca_mc,M,Nsim,truecdf,ca_true,ca_true_std,ca,m)
    }
save(Design3_table30, file="/media/EPGE/Docs/dados/Design3_T_table30.10.Rdata"
Listing 44: art1 size tables 10percent V.R
for (j in 1:7) {
  if (!is.na(ca[[j]])$norm)) {
    Design1_table92[j+1,2+i-1]<-ca[[j]]$std
    Design1_table92[j+1,2+i]<-100*(1-(truecdf(ca[[j]]$norm)-ifelse(any(ca_mc[[1]]==ca[[j]]$norm),1/Nsim,0))) - Design1_table92[1,2+i])
  }

  # rm(ca_mc,M,Nsim,truecdf,ca_true,ca_true_std,ca)
  save(Design1_table92, file="/media/EPGE/Docs/dados/Design1_V_table92_10.Rdata")
}

## Table for Design 1 with n=30, alpha = 10% for Vn
Design1_table30<-matrix(NA,8,6)
nc<30
alpha<.10
mlist<-c(2,5,9)

for (i in 1:3) {
  ca<-vector("list",?)
  nc<-list[1]
  M<-genDesign(n,m)
  load(paste("/media/EPGE/Docs/dados/MC.CV.Design_"",n,"_"",m,".Rdata",sep="")) # returns ca_mc
  truecdf<-ecdf(ca_mc[[1]])
  Nsim<-length(ca_mc[[1]])
  ca_true<-sort(ca_mc[[1]],decreasing=TRUE)[alpha*(Nsim+1)]
  ca_true_std<-ca_true-mean(ca_mc[[1]])/sd(ca_mc[[1]])
  Design1_table30[1,2+i-1]<-ca_true_std
  Design1_table30[1,2+i]<-100*(1-(truecdf(ca_true)-ifelse(any(ca_mc[[1]]==ca[[j]]$norm),1/Nsim,0))) # actual size
  print(ca[[1]]) <- Ev4CV_V(alpha,0,0)
  print(ca[[2]]) <- SymmetricCV_V(alpha,0,0)
  print(ca[[3]]) <- saddlepointCV_V(alpha,0,0)
  print(ca[[4]]) <- list(norm=NA, std=NA)
  print(ca[[5]]) <- RNCV_V(alpha,0,0)
  print(ca[[6]]) <- LRACV_V(alpha,0,0)
  print(ca[[7]]) <- BJNCV_V(alpha,0,0)

  for (j in 1:7) {
    if (!is.na(ca[[j]])$norm)) {
      Design1_table92[j+1,2+i-1]<-ca[[j]]$std
      Design1_table92[j+1,2+i]<-100*(1-(truecdf(ca[[j]]$norm)-ifelse(any(ca_mc[[1]]==ca[[j]]$norm),1/Nsim,0)))-Design1_table92[1,2+i])
    }
  }

  # rm(ca_mc,M,Nsim,truecdf,ca_true,ca_true_std,ca)

  save(Design1_table30, file="/media/EPGE/Docs/dados/Design1_V_table30_10.Rdata")
}

## Table for Design 2 with n=92, alpha = 10% for Vn
Design2_table92<-matrix(NA,8,6)
n<92
alpha<.10
mlist<-c(2,5,9)

for (i in 1:3) {
  ca<-vector("list",?)
  nc<-list[1]
  M<-genDesign2(n,m)
  load(paste("/media/EPGE/Docs/dados/MC.CV.Design2_"",n,"_"",m,".Rdata",sep="")) # returns ca_mc
  truecdf<-ecdf(ca_mc[[1]])
  Nsim<-length(ca_mc[[1]])
  ca_true<-sort(ca_mc[[1]],decreasing=TRUE)[alpha*(Nsim+1)]
  ca_true_std<-ca_true-mean(ca_mc[[1]])/sd(ca_mc[[1]])
  Design2_table92[1,2+i-1]<-ca_true_std
  Design2_table92[1,2+i]<-100*(1-(truecdf(ca_true)-ifelse(any(ca_mc[[1]]==ca[[j]]$norm),1/Nsim,0))) # actual size
  print(ca[[1]]) <- Ev4CV_V(alpha,0,0)
  print(ca[[2]]) <- list(norm=NA, std=NA)
  print(ca[[3]]) <- saddlepointCV_V(alpha,0,0)
  print(ca[[4]]) <- list(norm=NA, std=NA)
  print(ca[[5]]) <- RNCV_V(alpha,0,0)
  print(ca[[6]]) <- LRACV_V(alpha,0,0)
  print(ca[[7]]) <- BJNCV_V(alpha,0,0)
for (j in 1:7)
  if (!is.na(ca[j]@norm)) {
    Design2_table92[j,1:2+1-1] <- design2_index[1,2+1-1] * ca[j]@std
    Design2_table92[j,1:2+1] <= 100 * (1 - (truecf((ca[[j]]@norm) - ifelse(any(ca_mc[[j]] == ca[[j]]@norm, 1/NSim, 0)))) - Design2_table92[1,2+1])
  }
  # rm(ca_mc, M.Nsim, truecf, ca_true, ca_true_std, ca, m)
  }
  }
  save(Design2_table, file = "~/media/EPGE/Docs/dados/Design2_V_table92_10.Rdata")
  }
  }
  ## Table for Design 2 with n=30, alpha = 10% for Vn
  Design2_table30 < matrix(NA, 8, 6)
  n< 30
  alpha< 0.10
  list<- c(2,5,9)
  for (i in 1:3) {
    ca<- vector("list",7)
    m<- list[i]
    M<- genDesign2(n, m)
    load(paste("~/media/EPGE/Docs/dados/MC_CV_V_Design2","n","_",m,"."Rdata",sep="")) # returns ca_mc
    truecf <- ecdf(ca_mc[[1]])
    Nsim<- length(ca_mc[[1]])
    ca_true<- sort(ca_mc[[1]], decreasing=TRUE)[alpha*(Nsim+1)]
    ca_true_std<- (ca_true-mean(ca_mc[[1]]))/sd(ca_mc[[1]])
    Design2_table30[1:2+1-1] <- ca_true_std
    Design2_table30[1:2+1] <= 100 * (1 - (truecf((ca_true@norm) - ifelse(any(ca_mc[[j]] == ca[[j]]@norm, 1/NSim, 0)))) - Design2_table30[1,2+1])
  }
  }
  save(Design2_table30, file = "~/media/EPGE/Docs/dados/Design2_V_table30_10.Rdata")
  }
  }
  ## Table for Design 3 with n=92, alpha = 10% for Vn
  Design3_table92 < matrix(NA, 8, 6)
  n< 92
  alpha< 0.10
  list<- c(16,27,38)
  for (i in 1:3) {
    ca<- vector("list",7)
    J<- list[i]
    M<- genDesign3(n, m,J)
    load(paste("~/media/EPGE/Docs/dados/MC_CV_V_Design3","n","_",J,"."Rdata",sep="")) # returns ca_mc
    truecf <- ecdf(ca_mc[[1]])
    Nsim<- length(ca_mc[[1]])
    ca_true<- sort(ca_mc[[1]], decreasing=TRUE)[alpha*(Nsim+1)]
    ca_true_std<- (ca_true-mean(ca_mc[[1]]))/sd(ca_mc[[1]])
    Design3_table92[1:2+1-1] <- ca_true_std
    Design3_table92[1:2+1] <= 100 * (1 - (truecf((ca_true@norm) - ifelse(any(ca_mc[[j]] == ca[[j]]@norm, 1/NSim, 0)))) - Design3_table92[1,2+1])
  }
  }
  save(Design3_table92, file = "~/media/EPGE/Docs/dados/Design3_V_table92_10.Rdata")
  }
  }
  # Load data
  print(ca[[1]]) <- cv4CV_V(alpha,0,M)
  print(ca[[2]]) <- list(norm=NA, std=NA)
  print(ca[[3]]) <- saddlepointCV_V(alpha,0,M)
  print(ca[[4]]) <- list(norm=NA, std=NA)
  print(ca[[5]]) <- RACV_V(alpha,0,M)
  print(ca[[6]]) <- LRAV_V(alpha,0,M)
  print(ca[[7]]) <- BNJACV_V(alpha,0,M)
  for (j in 1:7) {
    if (!is.na(ca[j]@norm)) {
      Design2_table30[j,1:2+1-1] <- design2_index[1,2+1-1] * ca[j]@std
      Design2_table30[j,1:2+1] <= 100 * (1 - (truecf((ca[j]@norm) - ifelse(any(ca_mc[[j]] == ca[[j]]@norm, 1/NSim, 0)))) - Design2_table30[1,2+1])
    }
  }
  save(Design2_table30, file = "~/media/EPGE/Docs/dados/Design2_V_table30_10.Rdata")
  }
  }
  ## Table for Design 3 with n=92, alpha = 10% for Vn
  Design3_table92 < matrix(NA, 8, 6)
  n< 92
  alpha< 0.10
  list<- c(16,27,38)
  for (i in 1:3) {
    ca<- vector("list",7)
    J<- list[i]
    M<- genDesign3(n, m,J)
    load(paste("~/media/EPGE/Docs/dados/MC_CV_V_Design3","n","_",J,"."Rdata",sep="")) # returns ca_mc
    truecf <- ecdf(ca_mc[[1]])
    Nsim<- length(ca_mc[[1]])
    ca_true<- sort(ca_mc[[1]], decreasing=TRUE)[alpha*(Nsim+1)]
    ca_true_std<- (ca_true-mean(ca_mc[[1]]))/sd(ca_mc[[1]])
    Design3_table92[1:2+1-1] <- ca_true_std
    Design3_table92[1:2+1] <= 100 * (1 - (truecf((ca_true@norm) - ifelse(any(ca_mc[[j]] == ca[[j]]@norm, 1/NSim, 0)))) - Design3_table92[1,2+1])
  }
  }
  save(Design3_table92, file = "~/media/EPGE/Docs/dados/Design3_V_table92_10.Rdata")
  }
  }
  # Load data
  print(ca[[1]]) <- cv4CV_V(alpha,0,M)
  print(ca[[2]]) <- list(norm=NA, std=NA)
  print(ca[[3]]) <- saddlepointCV_V(alpha,0,M)
  print(ca[[4]]) <- list(norm=NA, std=NA)
  print(ca[[5]]) <- RACV_V(alpha,0,M)
  print(ca[[6]]) <- LRAV_V(alpha,0,M)
  print(ca[[7]]) <- BNJACV_V(alpha,0,M)
for (j in 1:7) {
  if (!is.na(ca[[j]]$norm)) {
    Design3_table92[j,1,2,1] <- ca[[j]]$std
    Design3_table92[j,1,2,1] <- 100*(1-(truecdf (ca[[j]]$norm)-ifelse(any(ca_mc[[1]]==ca[[j]]$norm),1/NSim,0))) - Design3_table92[j,1,2,1]
  }
  # rm(ca_mc,M,NSim,truecdf,ca_true,ca_true_std,ca,n)
  save(Design3_table92 ,file="/media/EPGE/Docs/dados/Design3_V_table92_10.Rdata")
}

## Table for Design 3 with n=30, alpha = 10% for Vn
Design3_table30<-matrix(NA,8,6)
n<-30
alpha<-0.10
Jlist<-c(9,15,22)

for (i in 1:3) {
  ca<-vector("list",7)
  J<-Jlist[i]
  n<-2*(n-1)-(J-1)*(J+2)/(2*n)
  M<-genDesign3(n,m,J)

  load(paste("/media/EPGE/Docs/dados/MC.CV.V.Design3_","","","J",".Rdata",sep="")) # returns ca_mc
  truecdf<-ecdf(ca_mc[[1]])
  NSim<-length(ca_mc[[1]])
  ca_true<-sort(ca_mc[[1]],decreasing=TRUE)[alpha*(NSim+1)]
  ca_true_std<-ca_true-mean(ca_mc[[1]])/sd(ca_mc[[1]])
  Design3_table30[1,2,1]<-ca_true_std
  Design3_table30[1,2,1]<-100*(1-(truecdf (ca_true)-ifelse(any(ca_mc[[1]]==ca_true),1/NSim,0))) # actual size
  print(ca[[1]] <- Ev4CV.V(alpha, 0, M))
  print(ca[[2]] <- SymmetricCV.V(alpha, M))
  print(ca[[3]] <- saddlepointCV.V(alpha, 0, M))
  print(ca[[4]] <- list(norm=NA,std=NA))
  print(ca[[5]] <- RACV.V(alpha, 0, M))
  print(ca[[6]] <- LRACV.V(alpha, 0, M))
  print(ca[[7]] <- BNJACV.V(alpha, 0, M))
  for () in 1:7 {
    if (!is.na(ca[[j]]$norm)) {
      Design3_table30[j,1,2,1] <- ca[[j]]$std
      Design3_table30[j,1,2,1] <- 100*(1-(truecdf (ca[[j]]$norm)-ifelse(any(ca_mc[[1]]==ca[[j]]$norm),1/NSim,0))) - Design3_table30[j,1,2,1]
    }
    # rm(ca_mc,M,NSim,truecdf,ca_true,ca_true_std,ca,n)
    save(Design3_table30 ,file="/media/EPGE/Docs/dados/Design3_V_table30_10.Rdata")
  }
}

## Table for Design 5 with n=92, alpha = 10% for Vn
Design5_table92<-matrix(NA,8,6)
n<-92
alpha<-0.10
Jlist<-c(16,27,38)

load("/media/EPGE/Docs/dados/design5.Rdata") # returns design5

designlist<-c("a","b","c")

for (i in 1:3) {
  ca<-vector("list",7)
  M<design5[i]
  load(paste("/media/EPGE/Docs/dados/MC.CV.V.Design5_","d",".Rdata",sep="")) # returns ca_mc
  truecdf<-ecdf(ca_mc[[1]])
  NSim<-length(ca_mc[[1]])
  ca_true<-sort(ca_mc[[1]],decreasing=TRUE)[alpha*(NSim+1)]
  ca_true_std<-ca_true-mean(ca_mc[[1]])/sd(ca_mc[[1]])
  Design5_table92[1,2,1]<-ca_true_std
  Design5_table92[1,2,1]<-100*(1-(truecdf (ca_true)-ifelse(any(ca_mc[[1]]==ca_true),1/NSim,0))) # actual size
  print(ca[[1]] <- Ev4CV.V(alpha, 0, M))
  print(ca[[2]] <- list(norm=NA,std=NA))
  print(ca[[3]] <- saddlepointCV.V(alpha, 0, M))
  print(ca[[4]] <- list(norm=NA,std=NA))
  print(ca[[5]] <- RACV.V(alpha, 0, M))
  print(ca[[6]] <- LRACV.V(alpha, 0, M))
  print(ca[[7]] <- BNJACV.V(alpha, 0, M))

for (i in 1:3) {
  ca<-vector("list",7)
  M<design5[i]
  load(paste("/media/EPGE/Docs/dados/MC.CV.V.Design5_","d",".Rdata",sep="")) # returns ca_mc
  truecdf<-ecdf(ca_mc[[1]])
  NSim<-length(ca_mc[[1]])
  ca_true<-sort(ca_mc[[1]],decreasing=TRUE)[alpha*(NSim+1)]
  ca_true_std<-ca_true-mean(ca_mc[[1]])/sd(ca_mc[[1]])
  Design5_table92[1,2,1]<-ca_true_std
  Design5_table92[1,2,1]<-100*(1-(truecdf (ca_true)-ifelse(any(ca_mc[[1]]==ca_true),1/NSim,0))) # actual size
  print(ca[[1]] <- Ev4CV.V(alpha, 0, M))
  print(ca[[2]] <- list(norm=NA,std=NA))
  print(ca[[3]] <- saddlepointCV.V(alpha, 0, M))
  print(ca[[4]] <- list(norm=NA,std=NA))
  print(ca[[5]] <- RACV.V(alpha, 0, M))
  print(ca[[6]] <- LRACV.V(alpha, 0, M))

```r
# Compare the critical values obtained using different methods
# LUMP test for statistic T_n

rm(list=ls()) # Apaga todas variáveis da memória

source("/media/EPGE/Docs/programas/all_functions.R")

## Table for Design 1 with n=92, alpha = 5% for Tn
Design1_table92<-matrix(NA,8,6)
alpha<-0.05
mlist<-c(2,5,9)
for (i in 1:3) {
  ca<-vector("list",7)
  m<-mlist[i]
  M<-genDesign1(n,m)
  load(paste("/media/EPGE/Docs/dados/MC_CV_Design1","n","_","m",".Rdata",sep="")) # returns ca_mc
  truedcf<-ecdf(ca_mc[[1]])
  Nsim<-length(ca_mc[[1]])
  ca_true<-sort(ca_mc[[1]],decreasing=TRUE)[alpha*(Nsimm+1)]
  ca_true_std<-<-(ca_true-mean(ca_mc[[1]]))/sd(ca_mc[[1]])
  Design1_table92[1,2*i-1]<-ca_true_std
  Design1_table92[1,2*i]<-100*(1-(truedcf(ca_true)-ifelse(any(ca_mc[[1]]==ca[[1]],norm,1/Nsim,0)))) # actual size
  print(ca[1]) <- BNACV_V(alpha,0,M))
  for (j in 1:7) {
    if (!is.na(ca[j])) {
      Design1_table92[j-1,2*i-1]<-ca[j]$
      Design1_table92[j-1,2*i]<-100*(1-(truedcf(ca[j])$norm)-ifelse(any(ca_mc[[1]]==ca[[1]],norm,1/Nsim,0))) - Design1_table92[1,2*i])
    }
  }
  # rm() ca_mc,M,Nsim,truedcf,ca_true,ca_true_std,ca,m)
  save(Design1_table92 ,file="/media/EPGE/Docs/dados/Design1_T_table92.10.Rdata")
```

Listing 45: art1 size tables 5percent.R
Design_table30[1,2+1]<-100*(1-(truecdf(ca_true)-ifelse(any(ca_mc[1] == ca_true),1/Nsim,0))) # actual size
print(ca[1]) <- Ev4CV_T(alpha,0,M)
print(ca[2]) <- CornPlCV_T(alpha,0,M)
print(ca[3]) <- saddlepointCV_T(alpha,0,M)
print(ca[4]) <- saddlepointCV_T(alpha,0,M)
print(ca[5]) <- RACV_T(alpha,0,M)
print(ca[6]) <- LRACV_T(alpha,0,M)
print(ca[7]) <- BJACV_T(alpha,0,M)
for (j in 1:7) {
  if (is.na(ca[[j]])){
    Design_table30[j+1,2+1]<-ca[[j]]
    Design_table30[1,2+1]<-100*(1-(truecdf(ca[[j]])-ifelse(any(ca_mc[1] == ca[[j]]),norm,1/Nsim,0)))-Design_table30[1,2+1]
  }
}
# rm(ca_mc,M,Nsim,truecdf,ca_true,ca_true_std,ca)
save(Design_table30,file="/media/EPGE/Docs/dados/Design1_T_table30_5.Rdata")

## Table for Design 2 with n=92, alpha = 5% for Tn
Design_table92<-matrix(NA,8,6)
nc<92
alpha<0.05
mlist<-c(2.5,9)
for (i in 1:3) {
c<vector("list",7)
m<mlist[i]
M<-genDesign2(n,m)
load(paste("/media/EPGE/Docs/dados/MC_CV_Design2_{1,2,3}.n","m",".Rdata",sep="")) # returns ca mc
truecdf<-ecdf(ca_mc[1])
Nsim<-length(ca_mc[1])
ca_true<sort(ca_mc[1],decreasing=TRUE)[alpha*Nsim+1]
ca_true_std<ca_true-mean(ca_mc[1])/sd(ca_mc[1])
Design_table92[1,2+1]<-ca_true_std
Design_table92[1,2+1]<-100*(1-(truecdf(ca_true)-ifelse(any(ca_mc[1] == ca_true),1/Nsim,0)))-Design_table92[1,2+1]
}
# rm(ca_mc,M,Nsim,truecdf,ca_true,ca_true_std,ca)
save(Design2_table92,file="/media/EPGE/Docs/dados/Design2_T_table92_5.Rdata")

## Table for Design 2 with n=30, alpha = 5% for Tn
Design_table30<-matrix(NA,8,6)
nc<30
alpha<0.05
mlist<-c(2.5,9)
for (i in 1:3) {
c<vector("list",7)
m<mlist[i]
M<-genDesign2(n,m)
load(paste("/media/EPGE/Docs/dados/MC_CV_Design2_{1,2,3}.n","m",".Rdata",sep="")) # returns ca mc
truecdf<-ecdf(ca_mc[1])
Nsim<-length(ca_mc[1])
ca_true<sort(ca_mc[1],decreasing=TRUE)[alpha*Nsim+1]
ca_true_std<ca_true-mean(ca_mc[1])/sd(ca_mc[1])
Design_table30[1,2+1]<-ca_true_std
Design_table30[1,2+1]<-100*(1-(truecdf(ca_true)-ifelse(any(ca_mc[1] == ca_true),1/Nsim,0)))-Design_table30[1,2+1]
}
Design2_table30[1,2+i]<-100*(1-(truecdf(ca_true)-ifelse(any(ca_mc[1][1]==ca_true),1/Nsim,0))) # actual size
print(ca[[1]])<- EvCV_T(alpha,0,M))
print(ca[[2]])<- CornFinCV_T(alpha,0,M))
print(ca[[3]])<- saddlepointCV_T(alpha,0,M))
print(ca[[4]])<- saddlepointNVCV_T(alpha,0,M))
print(ca[[5]])<- RACV_T(alpha,0,M))
print(ca[[6]])<- LRACV_T(alpha,0,M))
print(ca[[7]])<- BNJACV_T(alpha,0,M))
for (j in 1:7) {
if (!(is.na(ca[[j]])$norm))) {
Design2_table30[j+1,2+i-1]<-ca[[j]]$std
Design2_table30[j+1,2+i]<-100*(1-(truecdf(ca[[j]]$norm)-ifelse(any(ca_mc[[1]][1]==ca[[j]]$norm),1/Nsim,0))) - Design2_table30[1,2+i]
}
rm(ca_mc,M,Nsim,truecdf,ca_true,ca_true_std,ca,m)
save(Design2_table30,file="/media/EPGE/Docs/dados/Design2_T_table30_5.Rdata")
}
# Table for Design 3 with n=92, alpha = 5% for Tn
Design3_table92<-matrix(NA,8,6)
x<-92
alpha<-0.05
Jlist<-c(16,27,38)
for (i in 1:3) {
cx<-vector("list",7)
J<-Jlist[i]
m<(2*(n-1)+(J-1)*(J+2))/(2*m)
M<-genDesign3(n,m,J)
load(paste("/media/EPGE/Docs/dados/MC.CV.Design3_\"n\",\"_\",J,".Rdata",sep="\") # returns ca_mc
truecdf<-ecdf(ca_mc[[1]]))
Nsim<-length(ca_mc[[1]])
ca_true<-sort(ca_mc[[1]],decreasing=TRUE)[alpha*(Nsim+1)]
ca_true_std<-ca_true-mean(ca_mc[[1]])/sd(ca_mc[[1]])
Design3_table92[1,2+i-1]<-ca_true_std
Design3_table92[1,2+i]<-100*(1-(truecdf(ca_true)-ifelse(any(ca_mc[1][1]==ca_true),1/Nsim,0))) # actual size
print(ca[[1]])<- EvCV_T(alpha,0,M))
print(ca[[2]])<- CornFinCV_T(alpha,0,M))
print(ca[[3]])<- saddlepointCV_T(alpha,0,M))
print(ca[[4]])<- saddlepointNVCV_T(alpha,0,M))
print(ca[[5]])<- RACV_T(alpha,0,M))
print(ca[[6]])<- LRACV_T(alpha,0,M))
print(ca[[7]])<- BNJACV_T(alpha,0,M))
for (j in 1:7) {
if (!(is.na(ca[[j]])$norm))) {
Design3_table92[j+1,2+i-1]<-ca[[j]]$std
Design3_table92[j+1,2+i]<-100*(1-(truecdf(ca[[j]]$norm)-ifelse(any(ca_mc[[1]][1]==ca[[j]]$norm),1/Nsim,0))) - Design3_table92[1,2+i]
}
rm(ca_mc,M,Nsim,truecdf,ca_true,ca_true_std,ca,m)
save(Design3_table92,file="/media/EPGE/Docs/dados/Design3_T_table92_5.Rdata")
}
# Table for Design 3 with n=30, alpha = 5% for Tn
Design3_table30<-matrix(NA,8,6)
x<-30
alpha<-0.05
Jlist<-c(9,15,22)
for (i in 1:3) {
cx<-vector("list",7)
J<-Jlist[i]
m<(2*(n-1)+(J-1)*(J+2))/(2*m)
M<-genDesign3(n,m,J)
load(paste("/media/EPGE/Docs/dados/MC.CV.Design3_\"n\",\"_\",J,".Rdata",sep="\") # returns ca_mc
truecdf<-ecdf(ca_mc[[1]]))
Nsim<-length(ca_mc[[1]])
ca_true<-sort(ca_mc[[1]],decreasing=TRUE)[alpha*(Nsim+1)]
ca_true_std<-ca_true-mean(ca_mc[[1]])/sd(ca_mc[[1]])
Listing 46: art1 size tables 5percent V.R
15 m<-mlist[1]
16 MC<-genDesign1(n,m)
17 load(paste("/media/EPGE/Docs/dados/MC_CV_V/Design1","n","_","m",".Rdata",sep="")) # returns ca_mc
18 truecdf<-ecdf(ca_mc[[1]])
19 Nsim<-length(ca_mc[[1]])
20 ca_true<-sort(ca_mc[[1]],decreasing=TRUE)[alpha*(Nsim+1)]
21 ca_true_std<-(-ca_true-mean(ca_mc[[1]]))/sd(ca_mc[[1]]))
22 Design_table92[1,2*i-1]<-ca_true_std
23 Design_table92[1,2*i]<-100*(1-(truecdf(ca_true)-ifelse(any(ca_mc[[1]]==ca_true),1/Nsim,0))) # actual size
24 print(ca[[1]])<-EW4CV_V(alpha,0,M)
25 print(ca[[2]])<-SymmetricCV_V(alpha,M)
26 print(ca[[3]])<-saddlepointCV_V(alpha,0,M)
27 print(ca[[4]])<-list(norm=NA,std=NA)
28 print(ca[[5]])<-RACV_V(alpha,0,M)
29 print(ca[[6]])<-LRACV_V(alpha,0,M)
30 print(ca[[7]])<-BNJACV_V(alpha,0,M)
31 for (i in 1:7) {
32 if (((is.na(ca[[1]]))$norm)) {
33 Design_table92[1,2*i-1]<-ca[[1]]$std
34 Design_table92[1,2*i]<-100*(1-(truecdf(ca[[1]]))$norm)-ifelse(any(ca_mc[[1]]==ca[[1]]))$norm,1/Nsim,0))
35 }
36 # rm(ca_mc,M,Nsim,truecdf,ca_true,ca_true_std,ca,m)
37 }
38 save(Design_table92,file="/media/EPGE/Docs/dados/Design1_V_table92_5.Rdata")
39 # Table for Design 1 with n=30, alpha = 5% for Vn
40 Design_table30<-matrix(NA,8,6)
41 n<-30
42 alpha<-0.05
43 mlis<-c(2,5,9)
44 for (i in 1:3) {
45 ca<-vector("list",7)
46 m<-mlis[i]
47 MC<-genDesign1(n,m)
48 load(paste("/media/EPGE/Docs/dados/MC_CV_V/Design1","n","_","m",".Rdata",sep="")) # returns ca_mc
49 truecdf<-ecdf(ca_mc[[1]])
50 Nsim<-length(ca_mc[[1]])
51 ca_true<-sort(ca_mc[[1]],decreasing=TRUE)[alpha*(Nsim+1)]
52 ca_true_std<-(-ca_true-mean(ca_mc[[1]]))/sd(ca_mc[[1]]))
53 Design_table30[1,2*i-1]<-ca_true_std
54 Design_table30[1,2*i]<-100*(1-(truecdf(ca_true)-ifelse(any(ca_mc[[1]]==ca_true),1/Nsim,0))) # actual size
55 print(ca[[1]])<-EW4CV_V(alpha,0,M)
56 print(ca[[2]])<-SymmetricCV_V(alpha,M)
57 print(ca[[3]])<-saddlepointCV_V(alpha,0,M)
58 print(ca[[4]])<-list(norm=NA,std=NA)
59 print(ca[[5]])<-RACV_V(alpha,0,M)
60 print(ca[[6]])<-LRACV_V(alpha,0,M)
61 print(ca[[7]])<-BNJACV_V(alpha,0,M)
62 for (j in 1:7) {
63 if (((is.na(ca[[1]]))$norm)) {
64 Design_table30[1,2*i-1]<-ca[[1]]$std
65 Design_table30[1,2*i]<-100*(1-(truecdf(ca[[1]])$norm)-ifelse(any(ca_mc[[1]]==ca[[1]])$norm,1/Nsim,0)))
66 }
67 }
68 save(Design_table30,file="/media/EPGE/Docs/dados/Design1_V_table30_5.Rdata")
69 # Table for Design 2 with n=92, alpha = 5% for Vn
70 Design_table92<-matrix(NA,8,6)
71 n<-92
72 alpha<-0.05
73 mlis<-c(2,5,9)
74 for (i in 1:3) {
75
ca <- vector("list", 7)
c <- mlist[1]
M <- genDesign2(n, m)
load.paste("/media/EPGE/Docs/dados/MC_CV_V_Design2_.", n, ",", m, ".Rdata", sep="") # returns ca.mc
truecdf <- ecdf(ca.mc[[1]])
Nsim <- length(ca.mc[[1]])
ca_true <- sort(ca.mc[[1]], decreasing=TRUE)[alpha*(Nsim+1)]
ca_true_std <- (ca_true-mean(ca.mc[[1]]))/sd(ca.mc[[1]])
Design2_table92[1,2*i-1] <- ca_true_std
Design2_table92[1,2*i-1+100*(1-truecdf(ca_true)-ifelse(any(ca.mc[[1]]==ca_true),1/Nsim,0))] # actual size
print(ca[[1]] <- Ev4CV_V(alpha, 0, M))
print(ca[[2]] <- list(norm=NA, std=NA))
print(ca[[3]] <- saddlepointCV_V(alpha, 0, M))
print(ca[[4]] <- list(norm=NA, std=NA))
print(ca[[5]] <- RACV_V(alpha, 0, M))
print(ca[[6]] <- LRACV_V(alpha, 0, M))
print(ca[[7]] <- BNJACV_V(alpha, 0, M))
for (j in 1:7) {
  if (fis.na(ca[[j]]$norm)) {
    Design2_table92[j+1,2*i-1] <- ca[[j]]$std
    Design2_table92[j+1,2*i-1+100*(1-truecdf(ca[[j]]$norm)-ifelse(any(ca.mc[[1]]==ca[[j]]$norm),1/Nsim,0)) - Design2_table92[1,2*i+1])
  }
  # rm(ca.mc,M,Nsim,truecdf,ca_true,ca_true_std,ca)
  save(Design2_table92, file="/media/EPGE/Docs/dados/Design2_V_table92.5.Rdata")
}
# Table for Design 2 with n=30, alpha = 5% for Vn
Design2_table30 <- matrix(NA, 8, 6)
c <- 30
alpha <- 0.05
mlist <- c(2, 5, 9)
for (i in 1:3) {
  ca <- vector("list", 7)
c <- mlist[1]
M <- genDesign2(n, m)
load.paste("/media/EPGE/Docs/dados/MC_CV_V_Design2_.", n, ",", m, ".Rdata", sep="") # returns ca.mc
truecdf <- ecdf(ca.mc[[1]])
Nsim <- length(ca.mc[[1]])
ca_true <- sort(ca.mc[[1]], decreasing=TRUE)[alpha*(Nsim+1)]
ca_true_std <- (ca_true-mean(ca.mc[[1]]))/sd(ca.mc[[1]])
Design2_table30[1,2*i-1] <- ca_true_std
Design2_table30[1,2*i-1+100*(1-truecdf(ca_true)-ifelse(any(ca.mc[[1]]==ca_true),1/Nsim,0))] # actual size
print(ca[[1]] <- Ev4CV_V(alpha, 0, M))
print(ca[[2]] <- list(norm=NA, std=NA))
print(ca[[3]] <- saddlepointCV_V(alpha, 0, M))
print(ca[[4]] <- list(norm=NA, std=NA))
print(ca[[5]] <- RACV_V(alpha, 0, M))
print(ca[[6]] <- LRACV_V(alpha, 0, M))
print(ca[[7]] <- BNJACV_V(alpha, 0, M))
for (j in 1:7) {
  if (fis.na(ca[[j]]$norm)) {
    Design2_table30[j+1,2*i-1] <- ca[[j]]$std
    Design2_table30[j+1,2*i-1+100*(1-truecdf(ca[[j]]$norm)-ifelse(any(ca.mc[[1]]==ca[[j]]$norm),1/Nsim,0)) - Design2_table30[1,2*i+1])
  }
  # rm(ca.mc,M,Nsim,truecdf,ca_true,ca_true_std,ca)
  save(Design2_table30, file="/media/EPGE/Docs/dados/Design2_V_table30.5.Rdata")
}
## Table for Design 3 with n=92, alpha = 5% for Vn
Design3_table92 <- matrix(NA, 8, 6)
n <- 92
alpha <- 0.05
jlist <- c(16, 27, 38)
for (i in 1:3) {
  ca <- vector("list", 7)
truecdf <- ecdf(ca_mc[[1]])

if (!is.na(ca[[j]]$snorm)) {
  Design3_table92[[j+1,2*i+1]] <- ca[[j]]$snorm
  Design3_table92[[j+1,2*i+1]] <- 100*(1-truecdf(ca[[j]]$snorm)-ifelse(any(ca_mc[[1]]==ca[[j]]$snorm),1/NSim,0))
}

for (j in 1:7) {
  if (((is.na(ca[[j]]$snorm))) {
    Design3_table92[[j+1,2*i+1]] <- ca[[j]]$snorm
    Design3_table92[[j+1,2*i+1]] <- 100*(1-truecdf(ca[[j]]$snorm)-ifelse(any(ca_mc[[1]]==ca[[j]]$snorm),1/NSim,0))
  }
  # rm(ca_mc,M,NSim,truecdf,ca_true,ca_true_std,ca,m)
  save(Design3_table92,file="/media/EPGE/Docs/dados/Design3_V_table92_5.Rdata")

  # Table for Design 3 with n=30, alpha = 5% for Vn
  Design3_table30 <- matrix(NA,8,6)
  n<30
  alpha<0.05
  Jlist<-c(9,15,22)

  for (i in 1:3) {
    ca<-vector("list",7)
    J<-Jlist[1]
    M<-(2*(n-1)+(J+1)*(J+2))/(2*n)
    MX<-genDesign3(n,M)
    load(paste("/media/EPGE/Docs/dados/Design3_V_Design3_",n,"_","J",".Rdata",sep="")) # returns ca_mc
    truecdf <- ecdf(ca_mc[[1]])
    NSim=length(ca_mc[[1]])
    ca_true <- sort(ca_mc[[1]],decreasing=TRUE)[alpha*(NSim+1)]
    ca_true_std <- (ca_true-mean(ca_mc[[1]]))/sd(ca_mc[[1]])
    Design3_table92[[1,2*i+1]] <- ca_true_std
    Design3_table30[[1,2*i+1]] <- ca_true_std
    Design3_table30[[1,2*i+1]] <- 100*(1-truecdf(ca[[j]]$snorm)-ifelse(any(ca_mc[[1]]==ca[[j]]$snorm),1/NSim,0))
  }

  for (j in 1:7) {
    if (((is.na(ca[[j]]$snorm))) {
      Design3_table30[[j+1,2*i+1]] <- ca[[j]]$snorm
      Design3_table30[[j+1,2*i+1]] <- 100*(1-truecdf(ca[[j]]$snorm)-ifelse(any(ca_mc[[1]]==ca[[j]]$snorm),1/NSim,0))
    }
    # rm(ca_mc,M,NSim,truecdf,ca_true,ca_true_std,ca,m)
    save(Design3_table30,file="/media/EPGE/Docs/dados/Design3_V_table30_5.Rdata")

    # Table for Design 5 with n=92, alpha = 5% for Vn
    Design5_table92 <- matrix(NA,8,6)
    n<92
    alpha<0.05
    Jlist<-c(16,27,38)
    load("/media/EPGE/Docs/dados/design5.Rdata") # returns design5
    designlist<-c("a","b","c")
for (i in 1:3) {
ca <- vector("list", 7)
d <- design5[1]
MC_design5[11]
load(paste("/media/EPGE/Docs/dados/MC.CV_V_Design5\".d",".Rdata",sep="")) # returns ca_mc
truecdf <- ecdf(ca_mc[[1]])
Nsim <- length(ca_mc[[1]])
ca_true <- sort(ca_mc[[1]], decreasing = TRUE)[alpha*(Nsim+1)]
ca_true Std <- (ca_true - mean(ca_mc[[1]]))/sd(ca_mc[[1]])
Design5_table92[1,2*i-1] <- ca_true Std
Design5_table92[1,2*i-1] <- 100*(1-(truecdf(ca[[j]]$norm)-ifelse(any(ca_mc[[1]]==ca[[j]]$norm),1/Nsim,0))) # actual size
print(ca[[1]] <- Ev4CV.V(alpha,0,M))
print(ca[[2]] <- list(norm=NA, std=NA))
print(ca[[3]] <- saddlepointCV.V(alpha,0,M))
print(ca[[4]] <- list(norm=NA, std=NA))
print(ca[[5]] <- RACV.V(alpha,0,M))
print(ca[[6]] <- LRACV.V(alpha,0,M))
print(ca[[7]] <- BNJACV.V(alpha,0,M))
for (j in 1:7) {
if (!is.na(ca[[j]]$norm)) {
Design5_table92[j,1,2*i-1] <- ca[[j]]$std
Design5_table92[j,1,2*i-1] <- 100*(1-(truecdf(ca[[j]]$norm)-ifelse(any(ca_mc[[1]]==ca[[j]]$norm),1/Nsim,0))) # Design5_table92[1,2*i1]
}
rm(ca_mc,M,Nsim,truecdf,ca_true,ca_true Std,ca,n)
}
save(Design5_table92, file="/media/EPGE/Docs/dados/Design5_V_table92.5.Rdata")
## EXTRA tables for n=500
## Table for Design 1 with n=500, alpha = 5% for Vn
rm(list=ls()) # Apaga todas variáveis da memória
source("/media/EPGE/Docs/programas/all.functions.R")
Design1_table500 <- matrix(NA,8,6)
n<-500
alpha<-0.05
mlist<-c(2,5,9)
for (i in 1:3) {
ca <- vector("list",7)
m<-mlist[1]
MC_genDesign1(n,m)
load(paste("/media/EPGE/Docs/dados/MC.CV_V_Design1\".n\".m\".Rdata",sep="")) # returns ca_mc
truecdf <- ecdf(ca_mc[[1]])
Nsim <- length(ca_mc[[1]])
ca_true <- sort(ca_mc[[1]], decreasing = TRUE)[alpha*(Nsim+1)]
ca_true Std <- (ca_true - mean(ca_mc[[1]]))/sd(ca_mc[[1]])
Design1_table500[1,2*i-1] <- ca_true Std
Design1_table500[1,2*i-1] <- 100*(1-(truecdf(ca[[j]]$norm)-ifelse(any(ca_mc[[1]]==ca[[j]]$norm),1/Nsim,0))) # actual size
print(ca[[1]] <- Ev4CV.V(alpha,0,M))
print(ca[[2]] <- SymmetricCV.V(alpha,M))
print(ca[[3]] <- saddlepointCV.V(alpha,0,M))
print(ca[[4]] <- list(norm=NA, std=NA))
print(ca[[5]] <- RACV.V(alpha,0,M))
print(ca[[6]] <- LRACV.V(alpha,0,M))
print(ca[[7]] <- BNJACV.V(alpha,0,M))
for (j in 1:7) {
if (!is.na(ca[[j]]$norm)) {
Design1_table500[j,1,2*i-1] <- ca[[j]]$std
Design1_table500[j,1,2*i-1] <- 100*(1-(truecdf(ca[[j]]$norm)-ifelse(any(ca_mc[[1]]==ca[[j]]$norm),1/Nsim,0))) # Design1_table500[1,2*i1]
}
rm(ca_mc,M,Nsim,truecdf,ca_true,ca_true Std,ca,n)
}
save(Design1_table500, file="/media/EPGE/Docs/dados/Design1_V_table500.5.Rdata")
## Table for Design 2 with n=500, alpha = 5% for Vn
Design2_table500 <- matrix(NA,8,6)
n<-500
alpha<-0.05
mlist<-c(2,5,9)
for (i in 1:3) {
c<-vector("list",7)
m<-mlist[i]
M<-genDesign2(n,m)
load(paste("/media/EPGE/Docs/dados/MC_CV_V_Design2","n","m",".Rdata",sep="")) # returns ca_mc
truecdf<-ecdf(ca_mc[[1]])
Nsim<-length(ca_mc[[1]])
ca_true<-sort(ca_mc[[1]],decreasing=TRUE)[alpha*(Nsim+1)]
ca_true_std<-ca_true-mean(ca_mc[[1]])/sd(ca_mc[[1]])
Design2_table500[1,2*i-1]<-ca_true_std
Design2_table500[1,2*i+1]<-100*(1-(truecdf(ca_true)-ifelse(any(ca_mc[[1]]==ca_true),1/Nsim,0))) # actual size
print(ca[[1]]) <- Ev4CV_V(alpha,0,M)
print(ca[[2]]) <- list(norm=NA, std=NA)
print(ca[[3]]) <- saddlepointCV_V(alpha,0,M)
print(ca[[4]]) <- list(norm=NA, std=NA)
print(ca[[5]]) <- RACV_V(alpha,0,M)
print(ca[[6]]) <- LRACV_V(alpha,0,M)
print(ca[[7]]) <- BNJACV_V(alpha,0,M)
for (j in 1:7) {
  if (!is.na(ca[[j]]$norm)) {
    Design2_table500[1,2*i-1]<ca[[j]]$std
    Design2_table500[1,2*i+1]<100*(1-(truecdf(ca[[j]]$norm)-ifelse(any(ca_mc[[1]]==ca[[j]]$norm),1/Nsim,0))) - Design2_table500[1,2*i+1]
  }
# rm(ca_mc,M,Nsim,truecdf,ca_true,ca_true_std,ca,m)
}
save(Design2_table500, file="/media/EPGE/Docs/dados/Design2_V_table500_S.Rdata")

## Table for Design 3 with n=500, alpha = 5% for Vn
Design3_table500<-matrix(NA,8,6)
n<500
alpha<0.05
Jlist<-c(38,64,89)
for (i in 1:3) {
c<-vector("list",7)
J<Jlist[i]
m<-(2*(n-1)+(J-1)*(J+2))/(2*m)
M<-genDesign3(n,m,J)
load(paste("/media/EPGE/Docs/dados/MC_CV_V_Design3","n","J",".Rdata",sep="")) # returns ca_mc
truecdf<-ecdf(ca_mc[[1]])
Nsim<-length(ca_mc[[1]])
ca_true<-sort(ca_mc[[1]],decreasing=TRUE)[alpha*(Nsim+1)]
ca_true_std<-ca_true-mean(ca_mc[[1]])/sd(ca_mc[[1]])
Design3_table500[1,2*i-1]<ca_true_std
Design3_table500[1,2*i+1]<100*(1-(truecdf(ca_true)-ifelse(any(ca_mc[[1]]==ca_true),1/Nsim,0))) # actual size
print(ca[[1]]) <- Ev4CV_V(alpha,0,M)
print(ca[[2]]) <- SymmetricCV_V(alpha,M)
print(ca[[3]]) <- saddlepointCV_V(alpha,0,M)
print(ca[[4]]) <- list(norm=NA, std=NA)
print(ca[[5]]) <- RACV_V(alpha,0,M)
print(ca[[6]]) <- LRACV_V(alpha,0,M)
print(ca[[7]]) <- BNJACV_V(alpha,0,M)
for (j in 1:7) {
  if (!is.na(ca[[j]]$norm)) {
    Design3_table500[1,2*i-1]<ca[[j]]$std
    Design3_table500[1,2*i+1]<100*(1-(truecdf(ca[[j]]$norm)-ifelse(any(ca_mc[[1]]==ca[[j]]$norm),1/Nsim,0))) - Design3_table500[1,2*i+1]
  }
# rm(ca_mc,M,Nsim,truecdf,ca_true,ca_true_std,ca,m)
}
save(Design3_table500, file="/media/EPGE/Docs/dados/Design3_V_table500_S.Rdata")

# this program calculates the skewness and kurtosis
# of the statistic Tn under all the different
# designs, considering rho1=0
rm(list=ls()) # Apaga todas variáveis da memória
source("/media/EPGE/Docs/programas/all_functions.R")
X11.options(width=5, height=5, type="ncairo")
ps.options(horizontal=FALSE, onefile=FALSE, height=4.5,
width=6, paper="special", pointsize=10)

tabela<-NA

## Design 1 - (Tn)
n<-30
m<-2
M<-genDesign1(n,m)
>k<-cumulantsT(4,0,0,M,NULL)
k<-invariantes(k)
tabela<-c(k[1],k[2]/n,k[3],k[4])

m<-5
M<-genDesign1(n,m)
>k<-cumulantsT(4,0,0,M,NULL)
k<-invariantes(k)
tabela<-rbind(tabela,c(k[1],k[2]/n,k[3],k[4]))

m<-9
M<-genDesign1(n,m)
>k<-cumulantsT(4,0,0,M,NULL)
k<-invariantes(k)
tabela<-rbind(tabela,c(k[1],k[2]/n,k[3],k[4]))

n<-92
m<-2
M<-genDesign1(n,m)
>k<-cumulantsT(4,0,0,M,NULL)
k<-invariantes(k)
tabela<-rbind(tabela,c(k[1],k[2]/n,k[3],k[4]))

m<-5
M<-genDesign1(n,m)
>k<-cumulantsT(4,0,0,M,NULL)
k<-invariantes(k)
tabela<-rbind(tabela,c(k[1],k[2]/n,k[3],k[4]))

m<-9
M<-genDesign1(n,m)
>k<-cumulantsT(4,0,0,M,NULL)
k<-invariantes(k)
tabela<-rbind(tabela,c(k[1],k[2]/n,k[3],k[4]))

## Design 2 - (Tn)
n<-30
m<-2
M<-genDesign2(n,m)
>k<-cumulantsT(4,0,0,M,NULL)
k<-invariantes(k)
tabela<-rbind(tabela,c(k[1],k[2]/n,k[3],k[4]))

m<-5
M<-genDesign2(n,m)
>k<-cumulantsT(4,0,0,M,NULL)
k<-invariantes(k)
tabela<-rbind(tabela,c(k[1],k[2]/n,k[3],k[4]))

m<-9
M<-genDesign2(n,m)
>k<-cumulantsT(4,0,0,M,NULL)
k<-invariantes(k)
tabela<-rbind(tabela,c(k[1],k[2]/n,k[3],k[4]))

n<-92
m<-2
M<-genDesign2(n,m)
>k<-cumulantsT(4,0,0,M,NULL)
k<-invariantes(k)
tabela<-rbind(tabela,c(k[1],k[2]/n,k[3],k[4]))

M<-genDesign2(n,m)
k <- cumulantsT(4,0,0,M, NULL)
k1 <- invariants(k)
tabla <- rbind(tabla, c(k[1], k[2]/n, k[3], k[4]))

m <- 5
M <- genDesign2(n, m)
k <- cumulantsT(4,0,0,M, NULL)
k1 <- invariants(k)
tabla <- rbind(tabla, c(k[1], k[2]/n, k[3], k[4]))

m <- 9
M <- genDesign2(n, m)
k <- cumulantsT(4,0,0,M, NULL)
k1 <- invariants(k)
tabla <- rbind(tabla, c(k[1], k[2]/n, k[3], k[4]))

## Design 3
# Design 5a has 450 non zeros, 5b has 940, 5c has 1662
# the values of J will be chosen to match the number of non-zeros
# on these designs. For smaller n=30, it will be chosen to match the average

m <- 30
J <- round((sqrt(1+4*(2*n+450/92-2*(n-1)))-1)/2, 0)
m <- (2*n-1)*(J+2)/(2*m)
M <- genDesign3(n, m, J)
k <- cumulantsT(4,0,0, M, NULL)
k1 <- invariants(k)
tabla <- rbind(tabla, c(k[1], k[2]/n, k[3], k[4]))

J <- round((sqrt(1+4*(2*n+940/92-2*(n-1)))-1)/2, 0)
m <- (2*n-1)*(J+2)/(2*m)
M <- genDesign3(n, m, J)
k <- cumulantsT(4,0,0, M, NULL)
k1 <- invariants(k)
tabla <- rbind(tabla, c(k[1], k[2]/n, k[3], k[4]))

J <- round((sqrt(1+4*(2*n+1662/92-2*(n-1)))-1)/2, 0)
m <- (2*n-1)*(J+2)/(2*m)
M <- genDesign3(n, m, J)
k <- cumulantsT(4,0,0, M, NULL)
k1 <- invariants(k)
tabla <- rbind(tabla, c(k[1], k[2]/n, k[3], k[4]))

J <- round((sqrt(1+4*(2*n+940/92-2*(n-1)))-1)/2, 0)
m <- (2*n-1)*(J+2)/(2*m)
M <- genDesign3(n, m, J)
k <- cumulantsT(4,0,0, M, NULL)
k1 <- invariants(k)
tabla <- rbind(tabla, c(k[1], k[2]/n, k[3], k[4]))

load(file = "~/media/EPGE/Docs/dados/design5.Rdata")
M <- design5$a
k <- cumulantsT(4,0,0,M, NULL)
k1 <- invariants(k)
tabla <- rbind(tabla, c(k[1], k[2]/n, k[3], k[4]))
M <- design5$b
Listing 48: arti skew kurtosis V.R

```r
# this program calculates the skewness and kurtosis
# of the statistic Vn under all the different
# designs, considering rho1=0

rm(list="ls") # Apaga todas variaveis da memoria

# since there is no closed formula for the cumulants of Vn
# we will use the empirical estimates from the Monte Carlo
# simulation. We could use liebermann's approximation,
# but not sure whether it would hold for small samples

tabela<-NA

table<-read.table(tabela)

save(tabela.file="/media/EPGE/Docs/dados/skew_kurtosis_T.Rdata")

## FIM!!
```

variância <- var(ca.mc[[1]])
skew <- (sum(demean -3)/nsim)/((sum(demean^2)/nsim)^((3/2))*sqrt(nsim*(nsim-1))/(nsim-2) # unbiased estimator
kurt <- (nsim+1)*nsim*(nsim-1)*sum(demean^4)/((nsim-2)*(nsim-3)*sum(demean^2)^2) -3*(nsim-1)^2/((nsim-2)*(nsim-3))
tabela <- rbind(tabela, c(media, variância, skew, kurt))

m <- 9
load("/media/EPGE/Docs/dados/MC_V_Design1_92_9.Rdata")
demean <- ca.mc[[1]]
nsim <- length(demean)
variância <- var(ca.mc[[1]])
skew <- (sum(demean -3)/nsim)/((sum(demean^2)/nsim)^((3/2))*sqrt(nsim*(nsim-1))/(nsim-2) # unbiased estimator
kurt <- (nsim+1)*nsim*(nsim-1)*sum(demean^4)/((nsim-2)*(nsim-3)*sum(demean^2)^2) -3*(nsim-1)^2/((nsim-2)*(nsim-3))
tabela <- rbind(tabela, c(media, variância, skew, kurt))

# Design 2 - (Tn)
m <- 30
load("/media/EPGE/Docs/dados/MC_V_Design2_30_2.Rdata")
demean <- ca.mc[[1]]
nsim <- length(demean)
variância <- var(ca.mc[[1]])
skew <- (sum(demean -3)/nsim)/((sum(demean^2)/nsim)^((3/2))*sqrt(nsim*(nsim-1))/(nsim-2) # unbiased estimator
kurt <- (nsim+1)*nsim*(nsim-1)*sum(demean^4)/((nsim-2)*(nsim-3)*sum(demean^2)^2) -3*(nsim-1)^2/((nsim-2)*(nsim-3))
tabela <- rbind(tabela, c(media, variância, skew, kurt))

m <- 5
load("/media/EPGE/Docs/dados/MC_V_Design2_30_5.Rdata")
demean <- ca.mc[[1]]
nsim <- length(demean)
variância <- var(ca.mc[[1]])
skew <- (sum(demean -3)/nsim)/((sum(demean^2)/nsim)^((3/2))*sqrt(nsim*(nsim-1))/(nsim-2) # unbiased estimator
kurt <- (nsim+1)*nsim*(nsim-1)*sum(demean^4)/((nsim-2)*(nsim-3)*sum(demean^2)^2) -3*(nsim-1)^2/((nsim-2)*(nsim-3))
tabela <- rbind(tabela, c(media, variância, skew, kurt))
load("/media/EPGE/Docs/dados/MC_CV_Design2_30_9.Rdata")
media<-mean(ca_mc[[1]])
demean<-ca_mc[[1]]-media
nsim<-length(demean)
variancia<-var(ca_mc[[1]])
skew<-sum(demean^3)/nsim/((sum(demean^2)/nsim)^2/3)*sqrt(nsim/(nsim-1))/(nsim-2) # unbiased estimator
kurt<-((nsim+1)*nsim*(nsim-1)*nsim-3)*sum(demean^4)/((nsim-2)*(nsim-3)*sum(demean^2)^2)-3*(nsim-1)/(nsim-2)*(nsim-3)
tabela<-rbind(tabela,c(media,variancia,skew,kurt))

n<-92
m<-2
load("/media/EPGE/Docs/dados/MC_CV_Design2_92_2.Rdata")
media<-mean(ca_mc[[1]])
demean<-ca_mc[[1]]-media
nsim<-length(demean)
variancia<-var(ca_mc[[1]])
skew<-sum(demean^3)/nsim/((sum(demean^2)/nsim)^2/3)*sqrt(nsim/(nsim-1))/(nsim-2) # unbiased estimator
kurt<-((nsim+1)*nsim*(nsim-1)*nsim-3)*sum(demean^4)/((nsim-2)*(nsim-3)*sum(demean^2)^2)-3*(nsim-1)/(nsim-2)*(nsim-3)
tabela<-rbind(tabela,c(media,variancia,skew,kurt))

m<-5
load("/media/EPGE/Docs/dados/MC_CV_Design2_92_5.Rdata")
media<-mean(ca_mc[[1]])
demean<-ca_mc[[1]]-media
nsim<-length(demean)
variancia<-var(ca_mc[[1]])
skew<-sum(demean^3)/nsim/((sum(demean^2)/nsim)^2/3)*sqrt(nsim/(nsim-1))/(nsim-2) # unbiased estimator
kurt<-((nsim+1)*nsim*(nsim-1)*nsim-3)*sum(demean^4)/((nsim-2)*(nsim-3)*sum(demean^2)^2)-3*(nsim-1)/(nsim-2)*(nsim-3)
tabela<-rbind(tabela,c(media,variancia,skew,kurt))

m<-9
load("/media/EPGE/Docs/dados/MC_CV_Design2_92_9.Rdata")
media<-mean(ca_mc[[1]])
demean<-ca_mc[[1]]-media
nsim<-length(demean)
variancia<-var(ca_mc[[1]])
skew<-sum(demean^3)/nsim/((sum(demean^2)/nsim)^2/3)*sqrt(nsim/(nsim-1))/(nsim-2) # unbiased estimator
kurt<-((nsim+1)*nsim*(nsim-1)*nsim-3)*sum(demean^4)/((nsim-2)*(nsim-3)*sum(demean^2)^2)-3*(nsim-1)/(nsim-2)*(nsim-3)
tabela<-rbind(tabela,c(media,variancia,skew,kurt))

# Design3
# Design 3a has 450 non zeros, 5b has 940, 5c has 1652
# the values of J will be chosen to match the number of non-zeros
# on these designs. For smaller n=30, it will be chosen to match the average

n<-30
J<-round(sqrt(1+4*(2+n*450/92-2*(n-1)))-1)/2,0)
m<-2*(n-1)-(J+2)/(2*n)
load("/media/EPGE/Docs/dados/MC_CV_Design3_30_9.Rdata")
media<-mean(ca_mc[[1]])
demean<-ca_mc[[1]]-media
nsim<-length(demean)
variancia<-var(ca_mc[[1]])
skew<-sum(demean^3)/nsim/((sum(demean^2)/nsim)^2/3)*sqrt(nsim/(nsim-1))/(nsim-2) # unbiased estimator
kurt<-((nsim+1)*nsim*(nsim-1)*nsim-3)*sum(demean^4)/((nsim-2)*(nsim-3)*sum(demean^2)^2)-3*(nsim-1)/(nsim-2)*(nsim-3)
tabela<-rbind(tabela,c(media,variancia,skew,kurt))

J<-round(sqrt(1+4*(2+n*940/92-2*(n-1)))-1)/2,0)
m<-2*(n-1)-(J+2)/(2*n)
load("/media/EPGE/Docs/dados/MC_CV_Design3_30_15.Rdata")
media<-mean(ca_mc[[1]])
demean<-ca_mc[[1]]-media
nsim<-length(demean)
variancia<-var(ca_mc[[1]])
skew<-sum(demean^3)/nsim/((sum(demean^2)/nsim)^2/3)*sqrt(nsim/(nsim-1))/(nsim-2) # unbiased estimator
kurt<-((nsim+1)*nsim*(nsim-1)*nsim-3)*sum(demean^4)/((nsim-2)*(nsim-3)*sum(demean^2)^2)-3*(nsim-1)/(nsim-2)*(nsim-3)
tabela<-rbind(tabela,c(media,variancia,skew,kurt))

J<-round(sqrt(1+4*(2+n*1652/92-2*(n-1)))-1)/2,0)
m<-2*(n-1)-(J+2)/(2*n)
load("/media/EPGE/Docs/dados/MC_CV_Design3_30_15.Rdata")
media<-mean(ca_mc[[1]])
demean<-ca_mc[[1]]-media
nsim<-length(demean)
variancia<-var(ca_mc[[1]])
skew<-sum(demean^3)/nsim/((sum(demean^2)/nsim)^2/3)*sqrt(nsim/(nsim-1))/(nsim-2) # unbiased estimator
kurt<-((nsim+1)*nsim*(nsim-1)*nsim-3)*sum(demean^4)/((nsim-2)*(nsim-3)*sum(demean^2)^2)-3*(nsim-1)/(nsim-2)*(nsim-3)
tabela<-rbind(tabela,c(media,variancia,skew,kurt))
\begin{verbatim}
167  variancia <- var(ca_mc[[1]])
168  skew<-(sum(demean^3)/nsim)/((sum(demean^2)/nsim)-(3/2))*sqrt(nsim*(nsim-1))/(nsim-2) # unbiased estimator
169  kurt<-(nsim+1)*nsim*(nsim-1)*sum(demean^4)/((nsim-2)*(nsim-3)*sum(demean^2)^2) -3*(nsim-1)
170     -2/((nsim-2)*nsim-3))
171  tabela <- rbind(tabela,c(media, variancia, skew, kurt))
172  
173  J<round((sqrt(1+4*(2+n*1662/92-2*(n-1)))-1)/2,0)
174  m<2*(n-1)+(J-1)*J/2)
175  load("/media/EPGE/Docs/dados/DCV_V_Design3_30_32.Rdata")
176  media <- mean(ca_mc[[1]])
177  demean <- ca_mc[[1]] - media
178  nsim <- length(demean)
179  variancia <- var(ca_mc[[1]])
180  skew<-(sum(demean^3)/nsim)/((sum(demean^2)/nsim)-(3/2))*sqrt(nsim*(nsim-1))/(nsim-2) # unbiased estimator
181  kurt<-(nsim+1)*nsim*(nsim-1)*sum(demean^4)/((nsim-2)*(nsim-3)*sum(demean^2)^2) -3*(nsim-1)
182     -2/((nsim-2)*nsim-3))
183  tabela <- rbind(tabela,c(media, variancia, skew, kurt))
184  
185  
186  
187  
188  
189  
190  
191  
192  
193  
194  
195  
196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
\end{verbatim}
```r
# This program compares the actual size of LUMP test (Tn)
# vs skew and kurtosis


colnames(tabcon) <- ten
library(xtable)

# For the 5% level
load("/media/EPGE/Docs/dados/Design1_T_table30_5.Rdata")
tabcon[1:10,1:4] <- t(tabcon)
rownames(tabcon) <- rownames(tabela)
load("/media/EPGE/Docs/dados/Design2_T_table30_5.Rdata")
tabcon[1:10,1:4] <- t(tabcon)
rownames(tabcon) <- rownames(tabela)
load("/media/EPGE/Docs/dados/Design3_T_table30_5.Rdata")
tabcon[1:10,1:4] <- t(tabcon)
rownames(tabcon) <- rownames(tabela)
load("/media/EPGE/Docs/dados/Design5_T_table92_5.Rdata")
tabcon[19:21,1:4] <- t(tabcon)
rownames(tabcon) <- rownames(tabela)


# For the 10% level

tabcon10 <- matrix(NA, 21,11)

Listing 49: art compare CV vs skew kurt T.R
```
load("/media/EPGE/Docs/dados/skew_kurtosis_T.Rdata") # return tabela
tabcon10[,1:4]<-tableauro
rownames(tabcon10)<-rownames(tabela)
# Loads the size of the tests calculates for each approximation (alpha=5%)
load("/media/EPGE/Docs/dados/Design1_T_table30_10.Rdata")
tabcon10[1:3,5:11]<-t(Design1_table30[-1,c(2,4,6)])
load("/media/EPGE/Docs/dados/Design1_T_table92_10.Rdata")
tabcon10[4:6,5:11]<-t(Design1_table92[-1,c(2,4,6)])
load("/media/EPGE/Docs/dados/Design2_T_table30_10.Rdata")
tabcon10[7:9,5:11]<-t(Design2_table30[-1,c(2,4,6)])
load("/media/EPGE/Docs/dados/Design2_T_table92_10.Rdata")
tabcon10[10:12,5:11]<-t(Design2_table92[-1,c(2,4,6)])
load("/media/EPGE/Docs/dados/Design3_T_table30_10.Rdata")
tabcon10[13:15,5:11]<-t(Design3_table30[-1,c(2,4,6)])
load("/media/EPGE/Docs/dados/Design3_T_table92_10.Rdata")
tabcon10[16:18,5:11]<-t(Design3_table92[-1,c(2,4,6)])
load("/media/EPGE/Docs/dados/Design5_T_table92_10.Rdata")
tabcon10[19:21,5:11]<-t(Design5_table92[-1,c(2,4,6)])


# summaries
summary(abs(tabcon))
summary(abs(tabcon[,-c(1:4)])) # reports the median absolute error
summary(abs(tabcon10[,-c(1:4)])) # reports the median absolute error

# comparing

# some regressions
df5<-data.frame(abs(tabcon[,-9]))
df10<-data.frame(abs(tabcon10[,-9]))

summary(reg5<-lm(Edgeworth~Skewness+Kurtosis+Skew2+Kurt2,df5))
summary(reg5<-lm(Cornish.Fisher~Skewness+Kurtosis+Skew2+Kurt2,df5))
summary(reg5<-lm(Saddlepoint~Skewness+Kurtosis+Skew2+Kurt2,df5))
summary(reg5<-lm(Robinson~Skewness+Kurtosis,df5))
summary(reg5<-lm(Lugannani.Rice~Skewness+Kurtosis+Skew2+Kurt2,df5))
summary(reg5<-lm(Barndorff.Nielsen...Jensen~Skewness+Kurtosis+Skew2+Kurt2,df5))

oldnames<-colnames(df5)
df5<-cbind(df5,(df5$Skewness)^2,(df5$Kurtosis)^2)
colnames(df5)<-c(oldnames,"Skew2","Kurt2")

summary(reg10<-lm(Edgeworth~Skewness+Kurtosis,df10))
summary(reg10<-lm(Cornish.Fisher~Skewness+Kurtosis,df10))
summary(reg10<-lm(Saddlepoint~Skewness+Kurtosis,df10))
summary(reg10<-lm(Robinson~Skewness+Kurtosis,df10))
summary(reg10<-lm(Lugannani.Rice~Skewness+Kurtosis,df10))
summary(reg10<-lm(Barndorff.Nielsen...Jensen~Skewness+Kurtosis,df10))

# comparing design 1 n=92 with design 5

tabcon[c(4:6,19:21),c(5:7,9:11)]

library(scatterplot3d)
scatterplot3d(tabcon[,1],tabcon[,2],tabcon[,5],xlab="Skew",ylab="Kurtosis",zlab="Edgeworth",box=T,angle=45)
plot(tabcon[,1],tabcon[,5])
reg<-lm(tabcon[-9,9]~tabcon[-9,1]+tabcon[-9,2])
summary(reg)
plot(reg)
cloud(tabcon[-9,2]-tabcon[-9,1]+tabcon[-9,2],zlab="Sadd",xlab="Skew",ylab="Kurtosis", scales=list(arrows=FALSE),screen = list(x = 0, y = 0, z = 90))

Listing 50: art compare CV vs skew kurt V.R
# This program compares the actual size of LUMPI test (Vn)
# vs skew and kurtosis
rm(list=ls())  # Apaga todas variáveis da memoria

# For the 5% level
tabcon5<- matrix(NA, 21, 10)
load("/media/EPGE/Docs/dados/skew_kurtosis_V.Rdata")  # return tabela
rownames(tabcon5)<- rownames(tabela)

# Loads the size of the tests calculates for each approximation (alpha=5%) 
load("/media/EPGE/Docs/dados/Design1_V_table30_5.Rdata")
tabcon5[1:3,5:10]<- t(Design1_table30[-c(1,5),c(2,4,6)])
load("/media/EPGE/Docs/dados/Design1_V_table92_5.Rdata")
tabcon5[4:6,5:10]<- t(Design1_table92[-c(1,5),c(2,4,6)])
load("/media/EPGE/Docs/dados/Design2_V_table30_5.Rdata")
tabcon5[7:9,5:10]<- t(Design2_table30[-c(1,5),c(2,4,6)])
load("/media/EPGE/Docs/dados/Design2_V_table92_5.Rdata")
tabcon5[10:12,5:10]<- t(Design2_table92[-c(1,5),c(2,4,6)])
load("/media/EPGE/Docs/dados/Design3_V_table30_5.Rdata")
tabcon5[13:15,5:10]<- t(Design3_table30[-c(1,5),c(2,4,6)])
load("/media/EPGE/Docs/dados/Design3_V_table92_5.Rdata")
tabcon5[16:18,5:10]<- t(Design3_table92[-c(1,5),c(2,4,6)])
load("/media/EPGE/Docs/dados/Design5_V_table92_5.Rdata")
tabcon5[19:21,5:10]<- t(Design5_table92[-c(1,5),c(2,4,6)])


# For the 10% level 
load("/media/EPGE/Docs/dados/skew_kurtosis_V.Rdata")  # return tabela
rownames(tabcon10)<- rownames(tabela)

# Loads the size of the tests calculates for each approximation (alpha=5%) 
load("/media/EPGE/Docs/dados/Design1_V_table30_10.Rdata")
tabcon10[1:3,5:10]<- t(Design1_table30[-c(1,5),c(2,4,6)])
load("/media/EPGE/Docs/dados/Design1_V_table92_10.Rdata")
tabcon10[4:6,5:10]<- t(Design1_table92[-c(1,5),c(2,4,6)])
load("/media/EPGE/Docs/dados/Design2_V_table30_10.Rdata")
tabcon10[7:9,5:10]<- t(Design2_table30[-c(1,5),c(2,4,6)])
load("/media/EPGE/Docs/dados/Design2_V_table92_10.Rdata")
tabcon10[10:12,5:10]<- t(Design2_table92[-c(1,5),c(2,4,6)])
load("/media/EPGE/Docs/dados/Design3_V_table30_10.Rdata")
tabcon10[13:15,5:10]<- t(Design3_table30[-c(1,5),c(2,4,6)])
load("/media/EPGE/Docs/dados/Design3_V_table92_10.Rdata")
tabcon10[16:18,5:10]<- t(Design3_table92[-c(1,5),c(2,4,6)])
load("/media/EPGE/Docs/dados/Design5_V_table92_10.Rdata")
tabcon10[19:21,5:10]<- t(Design5_table92[-c(1,5),c(2,4,6)])


# summaries 
summary(tabcon5[, -(1:4)])
summary(tabcon10[, -(1:4)])

# Comparing only the symmetric designs 
summary(abs(tabcon5[, c(7:12, 19:21)], -(1:4)))
summary(abs(tabcon10[, c(7:12, 19:21)], -(1:4)))

# comparing Vn and Tn 
# Tn loaded from the other file and named tabcon5T and tabcon10T 
# Vn renamed to tabcon5V and tabcon10V

# Vn loaded from tabcon5V and tabcon10V
73  tabcon5T <- tabcon
74  tabcon10T <- tabcon10
75
76  abserrTS <- (abs(tabcon5T[, -c(1:4, 8)]))  # elimina coluna saddle
77  abserrT10 <- (abs(tabcon10T[, -c(1:4, 8)]))  # elimina coluna saddle
78
79  abserrV5 <- (abs(tabcon5V[, -c(1:4)]))
80  abserrV10 <- (abs(tabcon10V[, -c(1:4)]))
81
82  summary(abserrTS)
83  summary(abserrV5)
84
85  # Edgeworth 10%
86  x <- cbind(abserrT10[, "Edgeworth"], abserrV10[, "Edgeworth"], abserrT5[, "Edgeworth"] -
87  abserrV5[, "Edgeworth"])
88
89  summary(x)
90
91  V1      V2      V3
92  Min.   0.030   0.046   -0.0660
93  1st Qu. 0.266   0.212    0.1080
94  Median 0.810   0.394    0.1960
95  Mean   1.277   0.932    0.3456
96  3rd Qu. 2.318   1.476    0.5720
97  Max.   3.840   3.722    1.1000
98  [,1]        [,2]        [,3]
99  Design10_30 0.2660053  0.2260045  0.04000080
100  Design10_300 0.9660193  0.39000788  0.57201144
101  Design10_3_2 2.5680514  1.59203184  0.97601982
102  Design10_9_2 0.1120022  0.13200264  -0.02000040
103  Design10_9_2 0.2300046  0.04600092  0.18600368
104  Design10_2_9 0.2840057  0.09600192  0.18800376
105  Design10_2_9 0.4980100  0.26600532  0.23200464
106  Design10_2_5 2.0280406  1.12802256  0.90001800
107  Design10_2_9 3.3800766  3.72207448  0.10800216
108  Design10_2_9 0.2040064  0.21200424  -0.00800016
109  Design10_2_9 0.2420048  0.11000220  0.13200264
110  Design10_2_9 0.8100162  0.60601212  0.20400408
111  Design10_3_9 1.0500210  0.36600732  0.68401368
112  Design10_3_15 2.5760515  1.47602952  1.10002200
113  Design10_3_22 3.8400768  2.84805696  0.99201984
114  Design10_9_16 0.9060181  0.71201424  0.19400388
115  Design10_2_27 2.3180464  2.07804156  0.24000480
116  Design10_2_38 3.1960639  2.80805616  0.38800776
117  Design_a  0.0300006  0.09600192  -0.06600132
118  Design_b  0.3420068  0.22200444  0.12000240
119  Design_c  0.5300106  0.43600872  0.09600188

120  # Edgeworth 5%
121  x <- cbind(abserrT5[, "Edgeworth"], abserrV5[, "Edgeworth"], abserrT5[, "Edgeworth"] -
122  abserrV5[, "Edgeworth"])
123
124  summary(x)
125
126  V1      V2      V3
127  Min.   0.0140   0.0620   -0.2240
128  1st Qu. 0.1520   0.1220   -0.4000
129  Median 0.3560   0.2640    0.0580
130  Mean  0.7985   0.3437    0.4518
131  3rd Qu. 0.9580   0.4200    0.7260
132  Max.  4.5321   1.3980    3.1341
133  [,1]        [,2]        [,3]
134  Design10_30 0.01400028  0.23800476  -0.22400448
135  Design10_30 0.38200764  0.12200244  0.26000520
136  Design10_3_9 1.88603772  0.74801496  1.13802276
137  Design10_9_2 0.09400188  0.06200124  0.03200064
138  Design10_2_9 0.12200245  0.08800176  0.03400068
139  Design10_2_9 0.30000600  0.36200724  -0.06200124
140  Design10_2_9 0.04200084  0.12200244  -0.08000160
141  Design10_2_9 1.42802856  0.59801196  0.83001660
142  Design10_2_9 4.53209064  1.39802796  3.13406268
143  Design10_2_9 0.15200304  0.09600188  0.05800188
144  Design10_2_9 0.20600412  0.30600612  -0.10000200
145  Design10_2_9 0.32600652  0.36600732  -0.04000080
146  Design3_30_9 0.91801836  0.19200384  0.72601652
147  Design3_30_1b 1.28602572  0.08600172  1.20002400
Design3_30_22 2.43604872 0.69401388 1.74203484
Design3_92_16 0.40800816 0.26400052 0.14400288
Design3_92_27 0.74801496 0.19000038 0.58801126
Design3_92_36 0.63401268 0.42000840 0.21400428
Design5_a 0.14400288 0.13000260 0.01400028
Design5_b 0.29200584 0.31200624 -0.02000040
Design5_c 0.35600712 0.42600852 -0.07000140

# Saddlepoint 10%
xc< cbind(abserrT10 [,"Saddlepoint"],abserrV10 [,"Saddlepoint"],abserrT10 [,"Saddlepoint"]-
abserrV10 [,"Saddlepoint"])

summary(x)

V1                V2                V3
Min.  0.0120  Min.  0.0080  Min.  -0.1280
1st Qu. 0.3520  1st Qu. 0.1860  1st Qu.  -0.0120
Median 0.5340  Median 0.4680  Median  0.0960
Mean  0.7554  Mean  0.6413  Mean  0.1141
3rd Qu. 1.1880  3rd Qu. 0.9840  3rd Qu.  0.2240
Max.  2.2560  Max.  1.9480  Max.  0.3660

[,1] [,2] [,3]
[1,] 0.40600812 0.18600372 0.22000440
[2,] 0.60800716 0.38400768 0.22400448
[3,] 1.18802376 0.98401968 0.20400408
[4,] 0.01800036 0.07800156 -0.06600120
[5,] 0.13800276 0.22000440 -0.08200164
[6,] 0.53401068 0.53001060 0.00400008
[7,] 0.40200804 0.03600072 0.36600732
[8,] 0.71801436 0.74601492 -0.02800568
[9,] 1.83603672 1.84803696 -0.01200024
[10,] 0.01200024 0.11200224 -0.10000200
[11,] 0.38000760 0.35400708 0.02600052
[12,] 0.50001000 0.46800936 0.03200064
[13,] 0.89601792 0.60201204 0.29400588
[14,] 1.53803076 1.18602372 0.39200704
[15,] 2.25604512 1.94803896 0.30800616
[16,] 0.54401088 0.30400608 0.24000480
[17,] 1.51003200 1.31402628 0.19600392
[18,] 1.78003560 1.59003180 0.19000380
[19,] 0.14400288 0.09000180 0.05400108
[20,] 0.35200704 0.48000960 -0.12800256
[21,] 0.10400208 0.08000160 0.09600192

# Saddlepoint 5%
xc< cbind(abserrT5 [,"Saddlepoint"],abserrV5 [,"Saddlepoint"],abserrT5 [,"Saddlepoint"]-
abserrV5 [,"Saddlepoint"])

summary(x)

V1                V2                V3
Min.  0.0400  Min.  0.0340  Min.  -0.09000
1st Qu. 0.122  1st Qu. 0.0880  1st Qu.  0.00400
Median 0.260  Median 0.2760  Median  0.05600
Mean  0.452  Mean  0.3835  Mean  0.06848
3rd Qu. 0.754  3rd Qu. 0.6420  3rd Qu.  0.14400
Max.  1.242  Max.  1.0740  Max.  0.28001

[,1] [,2] [,3]
[1,] 0.11400228 0.20400408 -0.09000180
[2,] 0.26000520 0.08200164 0.17800356
[3,] 0.75401508 0.64201284 0.11200224
[4,] 0.06200124 0.05800116 0.00400008
[5,] 0.15200304 0.09000180 0.06200124
[6,] 0.33600672 0.34600692 -0.01000200
[7,] 0.04000080 0.08800176 -0.04800096
[8,] 0.60201204 0.40400808 0.19800396
[9,] 0.96801936 0.93201864 0.03600072
[10,] 0.10200204 0.08800176 0.01400028
[11,] 0.21800436 0.27600552 -0.05800116
[12,] 0.22400448 0.17000340 0.05400108
[13,] 0.69201384 0.41200824 0.28000560
[14,] 0.91201824 0.69201384 0.22000440
[15,] 1.24204284 1.07402148 0.16800336
[16,] 0.36400728 0.33200664 0.03200064
[17,] 0.97401948 0.89601792 0.07800156
[18,] 1.12402248 0.98001960 0.14400288
Design5_a  0.13000260  0.08600172  0.04400088
Design5_b  0.12200244  0.16800336  -0.04600092
Design5_c  0.10000020  0.03600068  0.06600132

# Robinson 10%
xx <- cbind(abserrT10[, "Robinson"], abserrV10[, "Robinson"], abserrT10[, "Robinson"]-abserrV10[, "Robinson"])
summary(x)

Min. 1st Qu.  Median    Mean 3rd Qu.    MAX.  
V1   :0.132  :0.288  :0.1098  :1.063  :1.760  :2.020  
V2   :0.126  :0.334  :1.262   :0.6000 :1.868  :2.236  
V3   :0.2540  :0.268  :0.1360  :0.0905  :0.0100  :0.1360

[,1] [,2] [,3]
Design1_30_0 0.3520070 0.6600121 -0.25400508
Design1_30_5 0.8240165 0.9200184 -0.09600192
Design1_30_9 1.8720374 1.8680374  0.04000088
Design1_92_2 0.1740035 0.2380048 -0.06400128
Design1_92_5 0.2880058 0.2300068  0.05800116
Design1_92_9 0.2500050 0.2320064  0.01800036
Design2_30_0 0.5460109 0.6820136 -0.13600272
Design2_30_5 1.7600352 1.8320366 -0.07200144
Design2_30_9 1.5980320 1.4620292  0.13600272
Design2_92_0 0.2740055 0.3340067 -0.06000120
Design2_92_5 0.2700054 0.2960059 -0.02600052
Design2_92_9 0.6520130 0.6620132  0.01000020
Design3_30_9 1.7340347 1.9500390 -0.21600432
Design3_30_5 1.8760375 1.9140383 -0.03800076
Design3_30_2 1.6420328 1.6720334  0.03000060
Design3_92_16 2.0200404 2.2360447 -0.21600432
Design3_92_27 1.9560381 2.0480410 -0.14200284
Design3_92_38 1.8840377 1.9100382  0.02600052
Design4_4_0 0.1320026 0.1260025  0.00600012
Design4_5  1.1740235 1.2960259 -0.12200244
Design4_5_c 1.0980220 1.2620252 -0.16400328

# Robinson 5%
xx <- cbind(abserrT5[, "Robinson"], abserrV5[, "Robinson"], abserrT5[, "Robinson"]-abserrV5[, "Robinson"])
summary(x)

Min. 1st Qu.  Median    Mean 3rd Qu.    MAX.  
V1   :0.0300  :0.0460  :0.1940  :0.1020  :0.7700  :1.2020  
V2   :0.0460  :0.6400  :0.0660  :0.0464  :0.8780  :0.0020  
V3   :0.2000  :0.1020  :0.0104  :0.1080

[,1] [,2] [,3]
Design1_30_0 0.34600692 0.45000900 -0.10400208
Design1_30_5 0.50801016 0.62601252 -0.11800236
Design1_30_9 0.71801436 0.85201704 -0.13400268
Design1_92_2 0.20200404 0.13200266  0.07000110
Design1_92_5 0.09800196 0.12600252 -0.02800056
Design1_92_9 0.04400088 0.05800116 -0.01400028
Design2_30_0 0.46800936 0.44000880  0.02800056
Design2_30_5 0.77001540 0.97001940 -0.20000400
Design2_92_0 0.54601092 0.72401448 -0.17800356
Design2_92_2 0.26000520 0.19400388  0.06000132
Design2_92_5 0.07000140 0.07200144 -0.00200044
Design2_92_9 0.42000840 0.42600852 -0.00600012
Design3_30_9 0.82001640 0.92201844 -0.10200240
Design3_30_5 0.90601812 0.89201784  0.01400028
Design3_30_2 0.57001156 0.66001280 -0.06200114
Design3_92_16 1.20202404 1.09602188 -0.02800044
Design3_92_27 0.79601592 0.88201766 -0.08600173
Design3_92_38 0.80001600 0.87801756 -0.07800156
Design5_a  0.03000060 0.04600092  0.01600032
Design5_b  0.71001420 0.77001540 -0.06000120
Design5_c  0.63001260 0.70401408 -0.07400148
# Lugannani_Rice 10%

x <- cbind(aberrT10[, "Lugannani_Rice"], aberrV10[, "Lugannani_Rice"], aberrT10[, "Lugannani_Rice"], aberrV10[, "Lugannani_Rice"])

summary(x)

V1 V2 V3
Min. :0.0220 Min. :0.0080 Min. : -0.232005
1st Qu.:0.0840 1st Qu.:0.1140 1st Qu.: -0.054001
Median :0.2220 Median :0.2540 Median : -0.026001
Mean : :0.2916 Mean : :0.2971 Mean : : -0.005629
3rd Qu.:0.4700 3rd Qu.:0.4580 3rd Qu.: :0.030001
Max. : :0.7760 Max. : :0.8300 Max. : :0.214004

Design1_30_2 0.02200064 0.25400058 -0.32300464
Design1_30_5 0.26400528 0.20400408 0.06000120
Design1_30_9 0.56001120 0.43000860 0.13000260
Design1_92_2 0.08400168 0.12800256 -0.04400088
Design1_92_5 0.11600322 0.01200204 0.10400208
Design1_92_9 0.06600132 0.09200184 -0.02600052
Design2_30_2 0.17000340 0.26400528 -0.09400188
Design2_30_5 0.66201324 0.49600992 0.16000332
Design2_30_9 0.22200444 0.00800016 0.21400428
Design2_92_2 0.16000320 0.21000420 -0.05000100
Design2_92_5 0.03600072 0.02800056 0.00800016
Design2_92_9 0.07400148 0.11400228 -0.04000080
Design3_30_2 0.48800976 0.46800936 0.02000040
Design3_30_5 0.47000940 0.44000880 0.03000060
Design3_30_9 0.19200384 0.17200344 0.02000040
Design3_92_16 0.77601552 0.83001660 -0.05400108
Design3_92_27 0.43200864 0.53401068 -0.10200204
Design3_92_38 0.44400888 0.44800896 -0.00400008
Design5_a 0.03600072 0.06400128 -0.02800056
Design5_b 0.49600992 0.58400118 -0.08800176
Design5_c 0.35400708 0.45800916 -0.10400208

# Lugannani_Rice 5%

x <- cbind(aberrT5[, "Lugannani_Rice"], aberrV5[, "Lugannani_Rice"], aberrT5[, "Lugannani_Rice"], aberrV5[, "Lugannani_Rice"])

summary(x)
```r
summary(x)
```

### Summary Output

```
  Min. :0.0160  Min. :0.0080  Min. :0.23600  Min. :0.086002
  1st Qu.:0.0500  1st Qu.:0.1000  1st Qu.:0.08400  1st Qu.:0.024000
  Median:0.1580  Median:0.2500  Median:0.02800  Median:0.03193
  Mean :0.2375  Mean :0.2691  Mean :0.03152  Mean :0.05201
  3rd Qu.:0.3900  3rd Qu.:0.4160  3rd Qu.:0.01600
  Max. :0.7320  Max. :0.7820  Max. :0.12800
```

### Design Variable Summary

```
[,1]          [,2]          [,3]
Design1_30_2 0.00140 028 0.025020 00 0.032600472
Design1_30_5 0.230000460 0.18200364 0.04800096
Design1_30_9 0.40200080 0.32200644 0.08000160
Design1_1_2 0.08200164 0.12600252 -0.04600088
Design1_1_5 0.10800216 0.00800016 0.10000200
Design2_1_9 0.03400016 0.25600512 -0.09800196
Design2_1_5 0.54801096 0.42000840 0.12800256
Design2_1_9 0.03600072 0.17200346 -0.13600272
Design2_1_2 0.15600312 0.20800616 -0.05200104
Design2_1_5 0.05000100 0.03400068 0.01600032
Design3_1_9 0.43200864 0.41600832 0.01600032
Design3_1_5 0.39000278 0.37600752 0.14000288
Design3_1_2 0.02800056 0.01600228 0.01600228
Design3_1_18 0.73601566 0.05001010 0.50001010
Design3_1_2 0.33800676 0.48000960 -0.14200284
Design3_1_38 0.34000680 0.36800736 -0.2800056
Design5_a 0.03800076 0.06600372 -0.02800056
Design5_b 0.47800956 0.56601132 -0.08800176
Design5_c 0.32800656 0.43600872 -0.10800216

# Barndorff-Nielsen / Jensen 5%
summary(x)
```

### Summary Output

```
  Min. :0.0020  Min. :0.0060  Min. :0.086002
  1st Qu.:0.0800  1st Qu.:0.0740  1st Qu.:0.024000
  Median:0.1180  Median:0.1160  Median:0.004000
  Mean :0.1425  Mean :0.1333  Mean :0.009143
  3rd Qu.:0.1900  3rd Qu.:0.1700  3rd Qu.:0.052001
  Max. :0.4460  Max. :0.3420  Max. :0.124002
```

### Design Variable Summary

```
[,1]          [,2]          [,3]
Design1_30_2 0.19000380 0.21400428 -0.02400048
Design1_30_5 0.16800336 0.16800336 0.00000000
Design1_30_9 0.08000160 0.11200224 -0.03200064
Design1_1_2 0.11800236 0.06600132 0.05200104
Design1_1_5 0.00200006 0.06000112 -0.04600088
Design1_1_9 0.10400208 0.09600192 0.08600016
Design2_1_2 0.29400588 0.17000340 0.12400248
Design2_1_5 0.12800256 0.21400428 -0.08600172
Design2_1_9 0.08000160 0.02200044 0.05800116
Design2_2_5 0.19200384 0.12000240 0.07200144
Design2_2_2 0.07200144 0.11600232 -0.04600088
Design2_2_9 0.08500172 0.12200244 -0.03600072
Design3_1_2 0.11400228 0.12600252 -0.01200024
Design3_1_5 0.14200284 0.10600208 0.03800076
Design3_1_9 0.18000360 0.10800216 0.07200144
Design3_1_16 0.44600892 0.34200684 0.10400208
Design3_1_27 0.04200084 0.02800056 0.01400028
Design3_1_28 0.01200025 0.07000140 -0.05800116
Design5_a 0.06200124 0.07400148 -0.01200024
Design5_b 0.26200526 0.28400568 -0.02200044
Design5_c 0.21800436 0.23800476 -0.02000040
```

### Summary (abs(tabcon))

```
summary(abs(tabcon))
```

### Summary Output

```
```

### Summary (abs(tabcon5[, -c(1:4)]))

```
summary(abs(tabcon5[, -c(1:4)])) # reports the median absolute error
```

### Summary Output

```
```

### Summary (abs(tabcon10[, -c(1:4)]))

```
summary(abs(tabcon10[, -c(1:4)])) # reports the median absolute error
```

### Summary Output

```
```
# some regressions
df5 <- data.frame(abs(tabcon[-9,]))
df10 <- data.frame(abs(tabcon10[-9,]))
summary(reg5 <- lm(Edgeworth ~ Skewness + Kurtosis + Skew2 + Kurt2, df5))
summary(reg5 <- lm(Cornish.Fisher ~ Skewness + Kurtosis + Skew2 + Kurt2, df5))
summary(reg5 <- lm(Saddlepoint ~ Skewness + Kurtosis + Skew2 + Kurt2, df5))
summary(reg5 <- lm(Robinson ~ Skewness + Kurtosis, df5))
summary(reg5 <- lm(Lugannani.Rice ~ Skewness + Kurtosis + Skew2 + Kurt2, df5))
summary(reg5 <- lm(Berndorf.Nielsen ... Jensen ~ Skewness + Kurtosis + Skew2 + Kurt2, df5))

oldnames <- colnames(df5)
df5 <- cbind(df5, (df5$Skewness)^2, (df5$Kurtosis)^2)
colnames(df5) <- c(oldnames, "Skew2", "Kurt2")

summary(reg10 <- lm(Eedgeworth ~ Skewness + Kurtosis, df10))
summary(reg10 <- lm(Cornish.Fisher ~ Skewness + Kurtosis, df10))
summary(reg10 <- lm(Saddlepoint ~ Skewness + Kurtosis, df10))
summary(reg10 <- lm(Robinson ~ Skewness + Kurtosis, df10))
summary(reg10 <- lm(Lugannani.Rice ~ Skewness + Kurtosis, df10))
summary(reg10 <- lm(Berndorf.Nielsen ... Jensen ~ Skewness + Kurtosis, df10))

# comparing design 1 n = 92 with design 5
tabcon[c(4:6,19:21), c(5:7,9:11)]

library(scatterplot3d)
scatterplot3d(tabcon[,1], tabcon[,2], tabcon[,5], xlab="Skew", ylab="Kurtosis", zlab="Edgeworth", box=T, angle=45)
plot(tabcon[,1], tabcon[,5])
reg <- lm(tabcon[-9,9] ~ tabcon[-9,1] + tabcon[-9,2])
s lines(reg)
plot(reg)

c $tabcon[-9,2] ~ tabcon[-9,1] + tabcon[-9,2], zlab="Sadd", ylab="Kurtosis",
scales=list(arrows=FALSE, screen = list(z = 0, x = 0, y = 90))

rm(list="ls")
source("/media/EPGE/Docs/programas/all.function.R")
par(options(width=5, height=5, type="ncbcairo")
ps.options(horizontal=FALSE, onefile=FALSE, height=4.5, width=6, paper="special", pointsize=10)
library(ligraph)

t<20
t<2
J<5
M1<-genDesign1(n,m)
M2<-genDesign2(n,m)
M3<-genDesign3(n,m,J)
M4a<-genDesign4_a(n,m)
M4b<-genDesign4_b(n,m)
load("/media/EPGE/Docs/dados/design5.Rdata")
MS<-design5 # a, b, and c

# Plot
g1<-graph.adjacency(M1, mode="undirected", weighted=TRUE)
g2<-graph.adjacency(M2, mode="directed", weighted=TRUE)
g3<-graph.adjacency(M3, mode="undirected", weighted=TRUE)
g4<-graph.adjacency(M4a+0.5*M4b, mode="undirected", weighted=TRUE)
tkplot(g1)
tkplot(g2)
tkplot(g3)
tkplot(g4)

# Creates table with skewness and kurtosis for each design

Listing 51: chart designs.R
Listing 52: matrix rj.R

```r
rm(list=ls()) # Apaga todas variáveis da memória
tmp <- system.time(source("/media/EPGE/Docs/programas/all_functions.R")
library(sp)
library(spdep)
library(rgeos)
library(tripack)
library(lattice)
library(igraph)

#source("matriz_functions.R")

# Define nbcairo para imprimir mais rápido na tela
X11.options(width=5, height=5, type="nbcairo")
ps.options(horizontal=FALSE, onefile=FALSE, height=4.5, width=6, paper="special", pointsize=10)

# Rotina para ler o mapa (do RJ)
# retorna uma lista com (map,IDs,ID.Names)
readMap <- function(mapfile) {
  map <- readGDER("/media/EPGE/Docs/dados/mapas/rj", mapfile,
  proj="+proj=longlat") # "33nu00gc" reads #RJ map (source: IBGE)
  # mapProjString(projargs="+longlat") # define as coordenadas como
  # lat long
  # isto eh usado em varias funcoes, como distancias,
  # graficos, etc. verifique proj4string(rj) e
  # is.projection(rj)

  # Seleciona apenas os com SEDE=TRUE
  map <- map[map$SEDE,"SEDE"]== "True",]
  IDs <- row.names(as(map, "data.frame")) # identifica os municipios
  ID.Names <- as.character(map$data[,2]) # falta corrigir os nomes
  #coords <- coordinates(rj) # obtem coordenadas dos municipios (latitude,
  # longitude)
  # setup(currentdir)
  return(list(map,IDs,ID.Names))
}

# Rotina para criar diferentes estruturas de vizinhan a
# retorna uma lista com as estruturas "nb" criadas
createNB <- function(map,IDs) {
  coords <- coordinates(map)
  N <- dim(map$data)[1]
  map_nb01 <- poly2nb(map, row.names=IDs, queen=TRUE)
  # nearest neighbour (K=1)
  map_aux <- knn2nb(knnearneigh(coords, k=1, longlat=T), row.names=IDs, sym=T)
  # Distance based neighbours
  data$nbdist <- unlist(map$dist(map_aux, coords,longlat=TRUE)) # base eh estrutura NN k1
  map_nb11 <- max(data$nbdist)
  map_nb02 <- dnearneigh(coords, d=0, d2=1.1*max$nb, row.names=IDs, longlat=T)
  map_nb03 < dnearneigh(coords, d=0, d2=1.56*max$nb, row.names=IDs, longlat=T)

  # Cria lista com todas as estruturas acima definidas
  nb_list <- list(Contiguity=map_nb01, Distd1=map_nb02, Distd2=map_nb03)
  return(nb_list)
}

# Rotina para gerar
# Alguns infos sobre as estruturas de vizinhan a criadas
summarynb <- function(nb_list) {
  #apply(nb_list, summary)
  cat("Verifica quais estruturas s o simetricas \n")
  print(sapply(nb_list, function(x) is.symmetric.nb(x, verbose=FALSE, force=TRUE))
  cat("Verifica numero de subgrafos disjuntos \n")
  print(sapply(nb_list, function(x) n.com. nb(x)$inc))
  cat("Verifica quais estruturas s o strongly connected\n")
  print(sapply(nb_list, function(x) is.connected( graph.adjacency( nbmat( x, style="B", zero.policy=T), mode="directed"), mode="strong"))) # verifica se o digraph eh
  #strongly connected
```

# return(0)

# funcao para imprimir mapa na tela
plotMap <- function(map, nb) {
  plot(map, border="grey60", axes=TRUE)
  plot(nb, coordinates(map), add=TRUE, pch=19, cex=0.6, arrows=TRUE, length=0.1/2,
       points=FALSE)
}

# funcao para plotar a comparacao entre vizinhos
comparnb <- function(map, nb1, nb2) {
  # oopar <- par(mar=c(3,3,1,1)+0.1)
  plot(map, border="grey60", axes=TRUE)
  # text(coordinates(map),cex=0.5) # label=row.names(as(map, "data.frame"))
  plot(nb1, coordinates(map), add=TRUE, pch=19, cex=0.6, arrows=TRUE,
       length=0.1/2, points=FALSE)
  plot(diffnb(nb1, nb2, verbose=FALSE), coordinates(map),
       add=TRUE, pch=19, cex=0.6, col="red", arrows=TRUE, length=0.1/2, points=FALSE)
  # par(oopar)
}

# funcao para imprimir uma matriz (heat)
# se for um array com dimenso maior do que 2
# a terceira dimenso sao os fatores
plotMatrix <- function(array, order=FALSE) {
  if (order) ord<-order.dendrogram(as.dendrogram(hclust(dist(array))))
  dim(array)[1] # nao verifica as dimensoes...
  levelplot(array[1:N,1:N], col.regions=gray(100:0/100),
            scales=list(draw=FALSE), xlab="spatial unit", ylab="spatial unit")
}

# Cria as matrizes de vizinhanca com base nas estruturas de
# vizinhanca a da_nb_list
#nb2mat(nb, glist=NULL, style="W", zero.policy=TRUE) # style B-binary, W-rowstand
# retorna uma w_list: primeira metade com normalizacao binaria B e segunda com W
createList <- function(nb_list) {
  names<-c(paste(names(nb_list),"B",sep="_"),paste(names(nb_list),"W",sep="_"))
  w_list <- vector("list",length(names))
  names(w_list)<-names
  for (i in names(nb_list)) {
    w_list[[paste(i,"B",sep="_")]]<-nb2mat(nb_list[[i]], glist=NULL, style="B",
                                              zero.policy=T)
    w_list[[paste(i,"W",sep="_")]]<-nb2mat(nb_list[[i]], glist=NULL, style="W",
                                              zero.policy=T)
  }
  return(w_list)
}

# Cria matriz de distancias (normalizadas ou nao)
createDistMatrix <- function(map, nb, longlat=TRUE, normalize=FALSE) {
  distances<-nbdist(map, nb, coordinates(map), longlat=longlat)
  N <- length(distances)
  DM <- matrix(NA, nrow=N, ncol=N)
  diag(DM)<-0
  for (i in 1:N) DM[i, nb[[i]]] <- distances[i]
  if (normalize) return(DM/max(DM)) else return(DM)
}

## Inicio programa principal
mapinfo <- readMap("33mu500gc") # RJ
rj <- mapinfo[[1]]
IDs <- mapinfo[[2]]
ID.names <- mapinfo[[3]]
rj_nb <- createblist(rj, IDs)
summary(nb2listw(rj_nb[[1]]))
summarynb(rj_nb)

postscript("/media/EPGE/Docs/figures/art1_chart01.eps")
plotMap(rj, rj_nb[[1]])
dev.off()
Listing 53: Alinha.R

```r
# A linha
function (rho, md=Mdefault)
return (W(rho, md)+t(W(rho, md))-t(w(rho, md))/%*%W(rho, md)-t(W(rho, md)))
```

Listing 54: a.r

```r
# A
function (rho, md=Mdefault, m2=NULL)
return (w(rho, md, m2) + t(w(rho, md, m2)) %*% w(rho, md, m2))
```

Listing 55: A.r.R

```r
'Ar' <-
function (rho, m1, m2=NULL) {
if (is.null(rho)) return (A(rho, m1, m2)/rho)
else return (W(0, m1, m2)+t(W(0, m1, m2)))
}
```

Listing 56: BNJACV T.R

```r
# Calculates the critical value based on the solution to
# the tail probability integral.
'BNJACV_T' <-
function (alpha, rho1, m1, m2=NULL, normalize=TRUE)

k <- cumulantsT(2, rho1, 0, m1, m2) # Under the NULL

if (rho1<0)
  (guess1=dnorm(1, mean=k[1], sd=sqrt(k[2]))
  guess2=dnorm(1, mean=k[1], sd=sqrt(k[2]))
  ca <- uniroot(function(t) BNJAP(t, rho1, 0, m1, m2)- alpha,
  lower=guess1, upper=guess2, tol=Machine$double.eps^-0.5)$root
else
  (guess1=dnorm(1, mean=k[1], sd=sqrt(k[2]))
  guess2=dnorm(1, mean=k[1], sd=sqrt(k[2]))
  ca <- uniroot(function(t) BNJAP(t, rho1, 0, m1, m2)-1+alpha,
  lower=guess1, upper=guess2, tol=Machine$double.eps^-0.5)$root
return (list(norw=ca, std=(ca-x[i])/sqrt(k[2])))
```

MRJ1< nb2mat(rj_nb[[1]], glist=NULL, style="B", zero.policy=T)
MRJ2< nb2mat(rj_nb[[2]], glist=NULL, style="B", zero.policy=T)
MRJ3< nb2mat(rj_nb[[3]], glist=NULL, style="B", zero.policy=T)

# Interval for the parameter rho
1/eigenRange(MRJ1)
1/eigenRange(MRJ2)
1/eigenRange(MRJ3)

compareb(rj, rj_nb[[2]], rj_nb[[1]])
desing5 <- list(a=MRJ1, b=MRJ2, c=MRJ3)
save(listing5, file="/media/EPGE/Docs/dados/design5.Rdata")
M<-rj_w <- createW(rj_nb)

#save(M, file="/media/EPGE/Docs/dados/matrix_exemplo/RJWexample.Rdata")

## FIN

# Grafico com a frequencia de numero de vizinhos em cada estrutura
res <- sapply(rj_nb, function(a) table(card(a))

mx <- max(sapply(rj_nb, card))
res1 <- matrix(0, ncol=(mx+1), nrow=length(rj_nb))
rownames(res1) <- names(res)
columns(res1) <- as.character(0:mx)
for (i in 1:length(rj_nb))
res1[i, names(res[[i]][1:i])]<- res[[i]]

barplot(res1[c(1:3)], horizontal=F, legend.text=TRUE,
xlab="numbers of neighbours")
Listing 57: bnjcsv.v <
# Calculates the critical value based on the solution to
# the tail probability integral.
'BNJCSV' <-
function(alpha, rhol, m1, m2=NULL, normalize=TRUE){
  k1 <- tr(A_rhol, m1, m2))/sqrt(ncol(m1)) # mean of Vn (approx) Laplace
  k2 <- 2*trPowerM(A_rhol, m1, m2, 2)/ncol(m1) # var of Vn (approx) Laplace
  if (rhol>0)
    {guess1<-qnorm(0.55, mean=k1, sd=sqrt(k2))
     guess2<qnorm(0.999, mean=k1, sd=sqrt(k2))
     ca <- uniroot(function(t) BNJAPV(t, rhol, 0, m1, m2) - alpha,
       lower=guess1, upper=guess2, tol=.Machine$double.eps^-0.5)$root
     ca <- uniroot(function(t) BNJAPV(t, rhol, 0, m1, m2) - 1+alpha,
       lower=guess1, upper=guess2, tol=.Machine$double.eps^-0.5)$root
     return(list(normal=ca, std=(ca-k1)/sqrt(k2)))
    }
  ca <- uniroot(function(t) (1-pnorm(t, mean=k1, sd=sqrt(k2))))
  return(ca)
}

Listing 58: bnjapt.r

'BNJAPT' <-
function(t, rhol, rho1, rho2, m1, m2)
mat <- Cr(rhol, rho1, rho2, m2)
gammahat <- saddlepoint_T(t, rhol, rho1, rho2, m1, m2)
mat2 <- diag(ncol(m1)) - 2*gammahat * mat
mat2inv <- solve(mat2)
a <- sign(gammahat) * sqrt(2*(gammahat * tr(mat2inv%*%mat) + 0.5*log(det(mat2))))
b <- gammahat * sqrt(2*trPowerM(mat2inv%*%mat, 2))
a_star <- a + log(b/a)
return(1 - pnorm(a_star))

Listing 59: bnjapv.r

'BNJAPV' <-
function(v, rhol, rho1, rho2, m1, m2)
mat <- Dv(v, rhol, rho1, rho2, m1, m2)
gammahat <- saddlepoint_V(v, rhol, rho1, rho2, m1, m2)
if (gammahat == -Inf) return(1)
if (gammahat == Inf) return(0)
if (abs(gammahat) < .Machine$double.eps^-0.5) {
  cat("gammahat MENOR que tolerancia!!\n\n"
  }
mat2 <- diag(ncol(m1)) - 2*gammahat * mat
mat2inv <- solve(mat2)
a <- sign(gammahat) * sqrt(log(det(mat2)))
b <- gammahat * sqrt(2*trPowerM(mat2inv%*%mat, 2))
if (abs(a-b) < .Machine$double.eps^-0.5)
  { a_star <- a # VERIFICAR QUAL EH O LIMITE!
   cat("WARNING!!! a and b are too close!!\n\n"
  )
  } else a_star <- a + log(b/a)/a
return(1 - pnorm(a_star))

Listing 60: BootstrapOLS.R

'BootstrapOLS' <- function(olsmodel, nbo, seed=2000) {
  require(sandwich)
  ehat <- olsmodel$residuals
  n <- length(ehat)
  x <- olsmodel$x
  yhat <- olsmodel$fitted.values
  sdat <- sqrt(diag(vcovHC(olsmodel, "HC3")))
  h <- diag(x%*%solve(crossprod(x))%*%x(x)) # leverage points
  fehat <- ehat/(1-h) #HC3 see Davidson Flachaire 2008 for details
  # Using the Rademacher distribution (see Davidson Flachaire (2008) and
  # Davidsonstasea, Monticini and Peel (2007)
```r
bootstrap.t <- matrix(NA, nrow=nb, ncol=ncol(x))
set.seed(seed)
for (b in 1:nb) {
  v<-2*rbahin(n,1,0.5)-1 # generates random sequence of -1, 1
  eb<-v*fehat
  yb<-yb+eb
  bootmodel<-lm(yb~x-1)
  bootstrap.t[b,] <- (bootmodel$coeff-olsmodel$coeff) /
sqrt(diag(vcovHC(bootmodel,"HC3")))
}
return(list(bst=apply(bootstrap.t,2,sort), sd=sdat,
  coef=olsmodel$coef))

'Bootstratedb Spatial' <- function(spmodel, nb, seed=2000) {
  require(MASS)
  p <-length(spmodel$coefficients)
  rhobat <-spmodel$coeff[1:(p-3)]
  rhobat <-spmodel$coeff[p-2]
  lambdahat <-spmodel$coeff[p-1]
  sigma2hat <-spmodel$coeff[p]
  ehat <-spmodel$residuals
  n <- length(ehat)
  x <- spmodel$x
  yhat <- spmodel$fitted.values
  sdat<- sqrt(diag(spmodel$vcov))
  k <- 1(lambdahat,spmodel$coeff)
  l <- 1(rhobat,spmodel$coeff)
  xx <- x%*%x
  h <- diag(xx%*%ginv(crossprod(xx))%*%t(xx)) # leverage points
  fehat <- ehat/(1-h) # HC3 see Davidson Flachaire 2008 for details
  # Using the Rademacher distribution (see Davidson Flachaire (2008) and
  # Davidsen, Montecini and Peel (2007)
  bootstrap.t <- matrix(NA, nrow=nb, ncol=p)
  set.seed(seed)
  for (b in 1:nb) {
    v<-2*rbahin(n,1,0.5)-1 # generates random sequence of -1, 1
    eb<-v*fehat
    yb<-yb+eb
    bootmodel<-ML.EgeneralSAR(x, yb, spmodel$coeff, spmodel$coeff)
    bootstrap.t[b,] <- (bootmodel$coeff-olsmodel$coeff) /
sqrt(diag(vcovHC(bootmodel,"HC3")))
    cat("ln Bootstrap number ", nb, ",n")
  }
  return(list(bst=apply(bootstrap.t,2,sort), sd=sdat,
    coef=spmodel$coeff))
}

'B' <-
function(rh_0, rh_1, md=Mdefault) {
  if (rh_0==rh_1) return((A(rh_1,md)-A(rh_0,md))/(rh_1-rh_0))
  else Alinha(rh_0,md)
}

'C0' <-
function(rh_0, rh_1, md=Mdefault) {
  C<-.t(Lin(rh_0,md)) %*% B(rh_0, rh_1, md) %*% Linv(rh_0,md)
  attributes(C)<-attributes(md),
  list("rhopositive"=ifelse(rh_0-rh_0,TRUE,FALSE),
    "rho0"=rh_0, "rho1"=rh_1, "hypothesis"="NULL")
  return(C)
}
```

Listing 61: BootstrapSpatial.R

Listing 62: b.r

Listing 63: c0.r
Listing 64: Ca.R

```r
'Ca' <-
function(rho_0,rho_1,md=Mdefault) {
  C<-t(Lin(rho_0,rho_1,md) %*% B[rho_0,rho_1,md]) %*% Lin(rho_1,md)
  attributes(C)<-c(attributes(md),
    list("rhopositive"=ifelse(rho_1>rho_0,TRUE,FALSE),
      "rho0"=rho_0, "rho1"=rho_1,
      "hypothesis"="ALTERNATIVE")
  return(C)
}
```

Listing 65: circularM.R

```r
'circularM' <-
function(n,m) {
  # creates the vector that defines the circulant matrix
  a <- c(0,rep(1,m),rep(0,n-(2*m-1)),rep(1,m))
  return(gencirculantM(a))
}
```

Listing 66: ConfidenceInterval.R

```r
'ConfidenceInterval' <- function(bootresult,alpha) {
  b.coef<-bootresult$coef
c.co.int<-matrix(NA,length(b.coef),3)
  nb<-nrow(bootresult$bst)
t.low <- alpha/2*nb
t.high <- (1-alpha/2)*nb
  st.low <- bootresult$bst[t.low,]*bootresult$sd
  st.high<- bootresult$bst[t.high,]*bootresult$sd
c.co.int <- b.coef-cbind(0,st.high,st.low)
colnames(c.co.int)<-c("Estimate", "Lower", "Upper")
  return(c.co.int)
}
```

Listing 67: CornFiCV T.R

```r
'CornFiCV_T' <-
function(alpha,rho1,m1,m2=NULL) {
  h<-createHermite(b) # for Cornish-Fisher
  k<-cumulantsT(c,rho1,0,m1,m2) # under the NULL hypothesis
  k_i<-invariants(k)
  if (rho1>0) t_a <- qnorm(1-alpha) # the 1-alpha quantile of the Gaussian
      else t_a <- qnorm(alpha)
  # The critical value changes depending on whether rho is positive
  c_a <- k[1] + sqrt(k[2])*t_a + ki[3]*h[[2]](t_a)/factorial(3) +
       ki[4]*h[[31]](t_a)/factorial(4) -
       ki[3]-2*(2*t_a^-3 -5*t_a)/36
  return(list(norm=c_a, std=(c_a-k[1])/sqrt(k[2])))
}
```

Listing 68: createHermite.R

```r
'createHermite' <-
function(D=10) {
  require(Polynomial)
  x<-polynomial()
  H<-polylist(1,x)
  for(n in 2:D)
    H[[n+1]] <- x*H[[n]] - (n-1)*H[[n-1]]
  return(H[[2:(D+1)]])
}
```

Listing 69: Cr.R

```r
'Cr' <-
```
function (rho_1, rho_m, m1, m2 = NULL) {
  C <- t(Linv(rho_m, m1, m2)) %*% A_r(rho_1, m1, m2) %*% Linv(rho_m, m1, m2)
  return(C)
}

'cumulantesT' <-
function (s, rho_1, rho_m, m1, m2 = NULL) {
  mat <- Cx(rho_1, rho_m, m2)
  kT <- function (s) return (2^(s - 1) * factorial(s - 1) * trPowerM(mat, s))
  klista <- sapply (1:s, kT)
  return (klista)
}

'cumulantesV' <-
function (s, v, rho_1, rho_m, m1, m2 = NULL) {
  mat <- Dv(v, rho_1, rho_m, m1, m2)
  kV <- function (s) return (2^(s - 1) * factorial(s - 1) * trPowerM(mat, s))
  klista <- sapply (1:s, kV)
  return (klista)
}

'Dv' <-
function (v, rho_1, rho_m, m1, m2 = NULL) {
  n <- ncol(m1)
  return (t(Linv(rho_m, m1, m2)) %*% (A_r(rho_1, m1, m2) - v * diag(n) / sqrt(n)) %*% Linv(rho_m, m1, m2))
}

# Step ia to estimate 'delta' in the model
# y = X * beta + lambda * y + u
# y = Z * delta + u
# u = rho * M * u + e

'E2SLS' <- function (Xmat, Y, Wmat, Mmat) {
  WY <- Wmat %*% Y
  WX <- Wmat %*% Xmat
  W2X <- Wmat %*% WX
  if (identical(Wmat, Mmat))
    Hmat <- cbind(Xmat, WX, W2X)
  else
    Hmat <- cbind(Xmat, WY, WX, W2X, Mmat %*% Xmat)
  require(Matrix) # for RankMatrix
  p <- rankMatrix(Hmat)
  k <- ncol(X)
  if (p < k + 1) stop ('Rank of instruments smaller than number of covariates')
  require(MASS) # for ginv
  PH <- Hmat %*% ginv(crossprod(Hmat)) %*% t(Hmat)
  Zmat <- cbind(X, WY)
  Ztil <- cbind(Xmat, PH %*% WY)
  Dtil <- solve(crossprod(Ztil, Zmat)) %*% crossprod(Ztil, Y)
  return (Dtil)
}

'E4' <-
function (x, k) {
  h <- createHermite (k)
  return (pnorm(x) - dnorm(x) * (k[3] * h[[2]](x) / factorial(3)) +
          k[4] * h[[3]](x) / factorial(4) +
          10 * k[3] * 2 * h[[5]](x) / factorial(6))
}

Listing 70: cumulantesT.R

Listing 71: cumulantesV.R

Listing 72: Dv.R

Listing 73: E2SLS.R

Listing 74: e4.r
Listing 75: e5.r

```r
'ES' <-
function(x, k) {
  h <- createHermite(8)
  return(pnorm(x) - dnorm(x) * (k[3]*h[[2]])(x)/factorial(3) +
  k[4]*h[[3]](x)/factorial(4) +
  10*k[3]**2*h[[6]](x)/factorial(6) +
  k[5]*h[[4]](x)/factorial(5) +
  35*k[3]*k[4]*h[[6]](x)/factorial(7) +
  280*k[3]**3*h[[8]](x)/factorial(9))
}
```

Listing 76: EGMM2.R

```r
# Step 1c to estimate 'rho' in the model
# y = X beta + lambda W y + u
# u = rho M u + e

'EGMM2' <- function(Xmat,Y,Wmat,Mmat,verbose=TRUE) {
  Dhat <- GS2SLS(Xmat,Y,Wmat,Mmat,verbose)
  Zmat <- cbind(Xmat,Wmat*%*%Y)
  Uhat <- Y - Zmat%*%Dhat
  UBhat <- Mmat%*%Uhat
  N <- length(Y)
  g <- G <- NA
  g[1,1] <- (crossprod(Uhat)) + tr(Mmat%*%diag(Uhat))%*%t(Mmat))/N
  g[2,1] <- crossprod(Uhat,Uhat)/N
  G[1,1] <- 2*(crossprod(Uhat,Uhat)-tr(Mmat%*%diag(Uhat))%*%t(Mmat))/N
  G[1,2] <- -1*tr(crossprod(Uhat,Uhat) + tr(Mmat%*%diag(Uhat))%*%t(Mmat))/N
  G[2,1] <- -1*crossprod(Uhat,Uhat)/N
  N <- length(Y)
  # Finds the parameter space for rho
  M <- g - G%*%rhomat
  return(crossprod(MC))
}

# Find the parameter space for rho
EVAL <- eigen(Mmat,only.values=TRUE)$val
ReEVAL <- Re(EVAL)[abs(Im(EVAL)) <= .Machine$double.eps^0.5]] # selects only REAL eigenvalues
RHOSPACE <- 1/length(RHV)
# check for zero eigenvalues on the left
if (RHOSPACE[1]==Inf) RHOSPACE[1] <- -Inf
# check whether the parameter space is contained in (0,1)
if (RHOSPACE[1] < 1) RHOSPACE[1] <- 1
RHOSPACE <- RHOSPACE + c(1,-1)*.Machine$double.eps^0.25
if (verbose) cat("The parameter space for rho is: ", RHOSPACE,"\n")

OPTRES <- optimize(Meq, interval=RHOSPACE, tol=.Machine$double.eps^0.5)
if (verbose) {
  cat("The initial GMM estimator of rho based on 2SLS residuals is: ",OPTRES$minimum ,\n')
  cat("The absolute value of the moment vector is: ",sqrt(OPTRES$objective),\n')
}

RhoBar <- OPTRES$minimum
Hmat <- NA
if (identical(Mmat,Mmat)) # note that Hmat is also calculated in the E2SLS
  Hmat <- cbind(Xmat,WX,W2X)
else
  Hmat <- cbind(Xmat,WX,W2X,Mmat%*%X)
require(MASS) # for ginv
PH <- Hmat%*%ginv(crossprod(Hmat))%*%t(Hmat)
}

Le <- diag(N)-RhoBar*Mmat
Et1 <- Le%*%Uhat
SIGNEDat1 <- diag(Et1^2)
AI <- crossprod(Mmat)-diag(crossprod(Mmat))
```
```r
A2 <- Mmat
Alphat111 <- Alphat112 <- NA
Alphat111 <- (-1/N)*t(Zmat)%*%(t(Lm)%*%(A1+t(A1))%*%Lm%*%Util)
Alphat112 <- (-1/N)*t(Zmat)%*%(t(Lm)%*%(A2+t(A2))%*%Lm%*%Util)
HPt11 <- NA
HPt11 <- N*PH%*%Zmat%*%solve(crossprod(Zmat,PH%*%Zmat))
At11 <- At12 <- NA
At11 <- solve(t(Lm))%*%HPt11%*%Alphat11
At12 <- solve(t(Lm))%*%HPt11%*%Alphat12
phi <- NA
phi[1,1] <- t((A1+t(A1))%*%SIGMAt11%*%(A1+t(A1))%*%SIGMAt11)/(2*N) + crossprod(At11, SIGMAt11)%*%At11/N
phi[1,2] <- t((A1+t(A1))%*%SIGMAt11%*%(A2+t(A2))%*%SIGMAt12)/(2*N) + crossprod(At11, SIGMAt11)%*%At12/N
phi[2,1] <- t((A2+t(A2))%*%SIGMAt11%*%(A1+t(A1))%*%SIGMAt11)/(2*N) + crossprod(At12, SIGMAt11)%*%At11/N
phi[2,2] <- t((A2+t(A2))%*%SIGMAt11%*%(A2+t(A2))%*%SIGMAt12)/(2*N) + crossprod(At12, SIGMAt11)%*%At12/N
MEq2 <- function(rhovec)
{ MC <- g - G%*%rhovec
   return(crossprod(MC,solve(phi)%*%MC))
}
OPTRES2 <- optimize(MEq2,interval=RHOSPACE,tol=.Machine$double.eps^0.5)
if (verbose)
{
   cat('The EFFICIENT GMM estimator of rho based on 2SLS residuals is: ',OPTRES2$minimum,\n
   cat('The absolute value of the moment vector is: ',sqrt(OPTRES2$objective),\n
   return(OPTRES2$minimum)
}

# Step ic to estimate 'rho' in the model
# y = X beta + lambda W y + u
# y = Z delta + u
# u = rho M u + e
#'EGMM' <- function(Xmat,Y,Wmat,Mmat,verbose=TRUE) {

Dtil <- E2SLS(Xmat,Y,Wmat,Mmat)
Zmat <- cbind(Xmat,Wmat%*%Y)
Util <- Y - ZZ%*%Dtil
Util <- Mmat%*%Util
U2til <- Mmat%*%Util
N <- length(Y)
G <- G <- NA
G[1,1] <- (crossprod(U2til) - tr(Mmat%*%diag(Util^2))%*%t(Mmat))/N
G[2,1] <- crossprod(U2til,Util)/N
G[1,1] <- 2*(crossprod(U2til,Uutil) - tr(Mmat%*%diag(Uutil*Util))%*%t(Mmat))/N
G[2,1] <- -1*(crossprod(U2til) + tr(Mmat%*%diag(Uutil^2))%*%t(Mmat))/N
G[2,2] <- -1*crossprod(U2til,Uutil)/N
MEq <- function(rhovec)
{
   MC <- g - G%*%rhovec
   return(crossprod(MC))
}

# Finds the parameter space for rho
EVAL <- eigen(Mmat,only$values=TRUE)$val
ReEVAL <- Re(EVAL[abs(Im(EVAL)) <= .Machine$double.eps^-0.5]) # selects only REAL eigenvalues
RHOSPACE <- 1/range(ReEVAL)
#check for zero eigenvalues on the left
if (RHOSPACE[1]==Inf) RHOSPACE[1] <- -Inf
# check whether the parameter space is contained in [0,1)
if (RHOSPACE[1] < -1) RHOSPACE[1] <- -1
RHOSPACE <- RHOSPACE + c(1-1)*.Machine$double.eps^-0.25
```
if (verbose) cat("The parameter space for rho is: ", RHOSPACE,"\n")

OPTRES <- optimize(Meq, interval=RHOSPACE, tol=Machine$double.eps^0.5)

if (verbose) {
  cat("The initial GMM estimator of rho based on 2SLS residuals is: ", OPTRES$minimum ,"\n")
  cat("The absolute value of the moment vector is: ", sqrt(OPTRES$objective),"\n")
}

RhoBar <- OPTRES$minimum
Hmat <- NA
if (identical(Wmat, Hmat)) # note that Hmat is also calculated in the E2SLS
  Hmat <- cbind(Wmat ,WX ,W2X)
else
  Hmat <- cbind(Xmat ,WX ,WX*Wmat%*%X)
regext(MASS) # for ginv

PH <- Hmat%*%ginv(crossprod(Hmat))%*%t(Hmat)

Lm <- diag(N)-RhoBar*Mmat
Et11 <- Lm%*%Util
SIGMat11 <- diag(Et11^2)
A1 <- crossprod(Mmat)-diag(crossprod(Mmat))
A2 <- Mmat
Alphat11 <- Alphat12 <- NA
Alphat11 <- (-1/N)*t((Zmat)%*%t(Lm)%*%(A1+t(A1))%*%Lm%*%Util)
Alphat12 <- (-1/N)*t((Zmat)%*%t(Lm)%*%(A2+t(A2))%*%Lm%*%Util)

HPt11 <- NA
HPt11 <- N*PH%*%Zmat%*%solve(crossprod(Zmat,PH%*%Zmat))

At111 <- At112 <- NA
At111 <- solve(t(Lm))%*%HPt11%*%Alphat11
At112 <- solve(t(Lm))%*%HPt12%*%Alphat12

phi <- NA
phi[1,1] <- -t(A1+t(A1))%*%SIGMat11%*%(A1+t(A1))%*%SIGMat11/(2*K) + crossprod(At11, SIGMat11%*%At111/N)
phi[1,2] <- -t(A1+t(A1))%*%SIGMat11%*%(A2+t(A2))%*%SIGMat11/(2*K) + crossprod(At11, SIGMat11%*%At112/N)
phi[2,1] <- -t((A2+t(A2))%*%SIGMat11%*%(A1+t(A1))%*%SIGMat11)/(2*K) + crossprod(At12, SIGMat11%*%At111/N)
phi[2,2] <- -t((A2+t(A2))%*%SIGMat11%*%(A2+t(A2))%*%SIGMat11)/(2*K) + crossprod(At12, SIGMat11%*%At112/N)

MEq2 <- function(rhovec) {
  MC <- g - GX*rhovec
  return(crossprod(MC,solve(phi)%*%MC))
}

OPTRES2 <- optimize(Meq2, interval=RHOSPACE, tol=Machine$double.eps^0.5)
if (verbose) {
  cat("The EFFICIENT GMM estimator of rho based on 2SLS residuals is: ", OPTRES2$ minimum ,"\n")
  cat("The absolute value of the moment vector is: ", sqrt(OPTRES2$objective),"\n")
}
return(OPTRES2$minimum)

Listing 78: eigenRange.R

'eigenRange' <-
function(M) {
  ev <- eigen(M, only.values=TRUE)$val
  return(range(Re(ev[Im(ev)==0)]))
}

Listing 79: Ew4CV.T.R

'Eff4CV_T' <-
function(alpha, rho1, m1, m2=NULL) {
  k <- cumulantsT(4, rho1, 0, m1, m2) # under the NULL hypothesis
  xk <- invariants(k)
```
if (rho1 > 0) {
  
guess1 <- qnorm(0.55, mean = k[1], sd = sqrt(k[2]))
  
guess2 <- qnorm(0.999, mean = k[1], sd = sqrt(k[2]))
  
c_a_n <- uniroot(function(c) E4((c - k[1])/sqrt(k[2])), c = alpha - 1, 
                  lower = guess1, upper = guess2, tol = Machine$double$eps^0.5)$root)
} else {
  guess1 <- qnorm(0.001, mean = k[1], sd = sqrt(k[2]))
  guess2 <- qnorm(0.45, mean = k[1], sd = sqrt(k[2]))
  c_a_n <- uniroot(function(c) E3((c - k[1])/sqrt(k[2])), c = alpha, 
                   lower = guess1, upper = guess2, tol = Machine$double$eps^0.5)$root)
return(list(norm = c_a_n, std = (c_a_n - k[1])/sqrt(k[2])))
}
}

Listing 80: Ew4CV V.R

'Ev4CV_V' <-
function(alpha, rho1, m1, m2 = NULL) {
  f1 <- function(v) {
    mat <- Dv(v, rho1, 0, m1, m2) # under the NULL
    eigenmat <- eigen(mat, only.values = TRUE)$val
    k1 <- sum(eigenmat)
    k2 <- 2 * sum(eigenmat - 2)
    k31 <- 8 * sum(eigenmat - 3)/k2^(3/2)
    k41 <- 48 * sum(eigenmat - 4)/k2^2
    E4(k1/sqrt(k2), c(0, 1, k31, k41))
  }

  k1^tr(A_r(rho1, m1, m2))/sqrt(ncol(m1)) # mean of Vn (approx) Laplace
  k2^2*trPowerM(A_r(rho1, m1, m2), 2)/ncol(m1) # var of Vn (approx) Laplace

  if (rho1 > 0) {
    guess1 <- qnorm(0.55, mean = k1, sd = sqrt(k2))
    guess2 <- qnorm(0.999, mean = k1, sd = sqrt(k2))
    c_a_n <- uniroot(function(c) f1(c) + alpha - 1, 
                      lower = guess1, upper = guess2, tol = Machine$double$eps^0.5)$root)
  } else {
    guess1 <- qnorm(0.001, mean = k1, sd = sqrt(k2))
    guess2 <- qnorm(0.45, mean = k1, sd = sqrt(k2))
    c_a_n <- uniroot(function(c) f1(c) - alpha, 
                      lower = guess1, upper = guess2, tol = Machine$double$eps^0.5)$root)
  return(list(norm = c_a_n, std = (c_a_n - k[1])/sqrt(k[2])))
}
}

Listing 81: Ew4pY.r

'Ev4pY' <-
function(v, rho1, rho, m1, m2 = NULL) {
  k <- cumulantsV(4, v, rho1, rho, m1, m2)
  k1 <- invariants(k)
  if (rho1 > 0) return(1 - E4(-k[1]/sqrt(k[2]), k))
  else return(E4(-k[1]/sqrt(k[2]), k))
}

Listing 82: Ew5CV T.R

'Ev5CV_T' <-
function(alpha, rho1, m1, m2 = NULL) {
  k <- cumulantsT(6, rho1, 0, m1, m2) # under the NULL hypothesis
  k1 <- invariants(k)
  guess1 <- qnorm(Machine$double$eps^0.5, mean = k[1], sd = sqrt(k[2]))
  guess2 <- qnorm(1 - Machine$double$eps^0.5, mean = k[1], sd = sqrt(k[2]))

  if (rho1 > 0) {
    c_a_n <- uniroot(function(c) E5((c - k[1])/sqrt(k[2])), c = alpha - 1, 
                        lower = guess1, upper = guess2, tol = Machine$double$eps^0.5)$root)
  } else {
    c_a_n <- uniroot(function(c) E5((c - k[1])/sqrt(k[2])), c = alpha, 
                      lower = guess1, upper = guess2, tol = Machine$double$eps^0.5)$root)
  return(list(norm = c_a_n, std = (c_a_n - k[1])/sqrt(k[2])))
}
```
Listing 83: Ew5CV V.R

```r
'EWSCV.V' <-
function(alpha, rho1, m1, m2=NULL) {
  f1 <- function(v) {
    mat = Dv(v, rho1, 0, m1, m2) # under the NULL
    eigenmat = eigen(mat, only=TRUE)$val
    k1 = sum(eigenmat)
  }
  k2 = 2*sum(eigenmat^2)
  k3 = 8*sum(eigenmat^3)/k2^3/2
  k4 = 48*sum(eigenmat^4)/k2^2
  k5 = -384*sum(eigenmat^5)/k2^5/2
  ES = (k1/sqrt(k2), c(0.1, k3, k4, k5))
}
kv1 = tr(A_r(rho1, m1, m2)) / sqrt(ncol(m1)) # mean of Vn (approx) Laplace
kv2 = 2*trPowerM(A_r(rho1, m1, m2), 2)/ncol(m1) # var of Vn (approx) Laplace
guess1 = qnorm(1 - Machine$double.eps^0.5, mean = kv1, sd = 2*sqrt(kv2))
guess2 = qnorm(1 - Machine$double.eps^0.5, mean = kv1, sd = 2*sqrt(kv2))
if (rho1>0)
  c_a_n <- uniroot(function(c) f1(c)+alpha-1,
  lower=guess1, upper=guess2, tol=Machine$double.eps^0.5)$root
else
  c_a_n <- uniroot(function(c) f1(c)-alpha,
  lower=guess1, upper=guess2, tol=Machine$double.eps^0.5)$root
return(list(norm=c_a_n, std=(c_a_n-kv1)/sqrt(kv2)))
}
```

Listing 84: FisherInfoRho.R

```r
'FisherInfoRho' <-
function(rho1, m1, m2=NULL) {
  temp1 = Linv(rho1, m1, m2)^%>%W(rho1, m1, m2)
  temp2 = W(rho1, m1, m2)^%>%Linv(rho1, m1, m2)
  return(tr(powerM(temp1, 2)+crossprod(temp2)))
}
```

Listing 85: gencirculantM.R

```r
'gencirculantM' <-
function(a) {
  n = length(a)
  M = a # initializes the circulant matrix
  for (i in 1:(n-1))
    M <- rbind(M, c(a[(n-i+1):n], a[1:(n-i)]))
  return(M)
}
```

Listing 86: genDesign1.R

```r
'genDesign1' <-
function(n, m) return(gencirculantM(c(0, rep(1, m), rep(0, n-(2*m+1)), rep(1, m))))
```

Listing 87: genDesign2.R

```r
'genDesign2' <-
function(n, m) return(gencirculantM(c(0, rep(1, 2*m+1), rep(0, n-(2*m+1)), 1)))
```

Listing 88: genDesign3.R

```r
'genDesign3' <-
function(n, m, j)
return(genToepplitzM((2*m+n/(2*n-1)+(j-1)*(J+2)))%*%c(0, rep(1, J-1), rep(0, n-J-2), 1, 0, 1, rep(0, n-J-2), rep(1, J-1), 0))
```

Listing 89: genDesign4 a.R

```r
'genDesign4_a' <-
function(n, m) {
  stopifnot(m%<=0)
```
return(gencirculantM(c(0, rep(1, n/2), rep(0, n-(n+1)), rep(1, n/2)))
}

Listing 90: gencirculantM R

`genDesign4_b` <-
function(n,m) {
stopifnot(n%/%2==0, n>2*m+1)
return(gencirculantM(c(0,rep(0,m/2),rep(1,n/2),rep(0,n-(2*m+1)),rep(1,m/2),rep(0,m/2))))
}

Listing 91: genToepIztlzMR

`genToeplitzM` <-
function(a) {
N <- length(a)
n <- (N+1)/2
T <- a[1:n] # inicializa matrix Toeplitz
for (i in 1:(n-1)) T[i]<-T[i]+(1+i)
return(T)
}

Listing 92: GMMIV.R

# This program performs the GMM/IV estimation for the model
# y = X beta + lambda W y + u
# y = Z delta + u
# u = rho H u + e
# as explained in Arraiz, Drukker, Kelejian, Prucha (2007)
# Define some auxiliar functions
# moment conditions matrices
getG <- function(delta, Y, Zm, Mm) {  
  uh <- as.vector(Y - Zm%*%delta)  
  vh <- as.vector(Mm%*%uh) # H^u  
  vb <- as.vector(Mm%*%vh) # H^v  
  N <- length(Y)  
  g <- matrix(NA,2,1)  
  G <- matrix(NA,2,2)
  g[1,1] <- (crossprod(vh)%*%diag(uh^2)*%t(Mm))/N  
  g[2,1] <- (crossprod(uh, uh)/N  
  G[1,1] <- 2*(crossprod(vh, vh)%*%diag(vh^2)*%t(Mm))/N  
  G[1,2] <- -1*(crossprod(vh, vh)%*%diag(vh^2)*%t(Mm))/N  
  G[2,1] <- (crossprod(vh, vh)/N  
  G[2,2] <- -1*crossprod(vh, vh)/N  
  return(list(g=g, G=G))
}
# moment equation
MEq <- function(rho, Glist, weight) {  
  rhovec <- c(rho, rho^2)  
  MC <- Glist%*%Glist%*%rhovec  
  return(crossprod(MC, solve(weight)%*%MC))
}
# Finds the parameter space for rho
rhospace <- function(Mmat) {  
  EVAL <- eigen(Mmat, only.values=TRUE)$val  
  ReEVAL <- Re(EVAL)  
  if (abs(Im(EVAL)) <= .Machine$double.eps^0.5) # selects only REAL eigenvalues  
    RHO_SPACE <- 1/range(ReEVAL)  
  if (RHO_SPACE[1]==Inf) RHO_SPACE[1]<- -Inf  
  # check whether the parameter space is contained in (0,1)  
  if (RHO_SPACE[1] < -1) RHO_SPACE[1]<- -1  
  RHO_SPACE <- c(1,-1)*.Machine$double.eps^0.25  
  return(RHO_SPACE)
}
# getWeight <- function(delta, rho, Y, Zm, MM, PH, opc2) {
#  uh <- Y - Zm%*%delta
47 # eh <- uh - rho*Mh*Xuh
48 # Sigma <- diag(eh^2)
49 # A1 <- crossprod(Mh) - diag(crossprod(Mm))
50 # A2 <- Mm
51 # N <- length(Y)
52 # Krho <- diag(N) - rho*Mm
53 # Krho.u <- Krho%*%u
54 # Krho.Z <- Krho%*%Zm
55 # alpha1 <- (-1/N)*(Krho.Z)%*%((A1+t(A1))%*%Krho.u)
56 # alpha2 <- (-1/N)*(Krho.Z)%*%((A2+t(A2))%*%Krho.u)
57 # if (opcao==1) {
58 # HP <- N*Phi%*%Zm%*%solve(crossprod(Zm,Phi%*%Zm))
59 # a1 <- solve(t(Krho))%*%HP%*%alpha1
60 # a2 <- solve(t(Krho))%*%HP%*%alpha2
61 # }
62 else{
63 # Zs <- Zm-rho*Mm%*%Zm
64 # HP <- N*Phi%*%Zs%*%solve(crossprod(Zs,Phi%*%Zs))
65 # a1 <- HP%*%alpha1
66 # a2 <- HP%*%alpha2
67 # }
68 # phi <- matrix(NA,2,2)
69 # A1S <- (A1+t(A1))%*%Sigma
70 # A2S <- (A2+t(A2))%*%Sigma
71 # Sa1 <- Sigma%*%a1
72 # Sa2 <- Sigma%*%a2
73 # phi[1,1] <- t(A1S%*%A1S)/(2*N) + crossprod(a1,Sa1)/N
74 # phi[1,2] <- t(A1S%*%A2S)/(2*N) + crossprod(a1,Sa2)/N
75 # phi[2,1] <- t(A2S%*%A1S)/(2*N) + crossprod(a2,Sa1)/N
76 # phi[2,2] <- t(A2S%*%A2S)/(2*N) + crossprod(a2,Sa2)/N
77 # return(phi)
78 # }

80 #getPhiOmega <- function(delta,rho,Y,Zm,Mm,Hm,Glist,opcao) {
81 require(MASS) # for ginv
82 Ph <- Hm%*%ginv(crossprod(Hm))%*%t(Hm)
83 uh <- as.vector(Y - Zm%*%delta)
84 eh <- as.vector(uh - rho*Mm%*%uh)
85 Sigma <- diag(eh^2)
86 # A1 <- crossprod(Mm) - diag(crossprod(Mm))
87 # A2 <- Mm
88 # N <- length(Y)
89 # Krho <- diag(N) - rho*Mm
90 # Krho.u <- Krho%*%u
91 # Krho.Z <- Krho%*%Zm
92 # alpha1 <- (-1/N)*t(Krho.Z)%*%((A1+t(A1))%*%Krho.u)
93 # alpha2 <- (-1/N)*t(Krho.Z)%*%((A2+t(A2))%*%Krho.u)
94 # if (opcao==1) {
95 # P <- N*ginv(crossprod(Hm))%*%t(Hm)%*%Zm%*%solve(crossprod(Zm,Ph%*%Zm))
96 # a1 <- solve(t(Krho))%*%HP%*%alpha1
97 # a2 <- solve(t(Krho))%*%HP%*%alpha2
98 # }
99 else{
100 # Zs <- Zm-rho*Mm%*%Zm
101 # P <- N*ginv(crossprod(Hm))%*%t(Hm)%*%Zs%*%solve(crossprod(Zs,Ph%*%Zs))
102 # a1 <- Hm%*%alpha1
103 # a2 <- Hm%*%alpha2
104 # }
105 # Phi.rr <- matrix(NA,2,2)
106 # A1S <- (A1+t(A1))%*%Sigma
107 # A2S <- (A2+t(A2))%*%Sigma
108 # Sa1 <- Sigma%*%a1
109 # Sa2 <- Sigma%*%a2
110 # Phi.rr[1,1] <- t(A1S%*%A1S)/(2*N) + crossprod(a1,Sa1)/N
111 # Phi.rr[1,2] <- t(A1S%*%A2S)/(2*N) + crossprod(a1,Sa2)/N
112 # Phi.rr[2,1] <- t(A2S%*%A1S)/(2*N) + crossprod(a2,Sa1)/N
113 # Phi.rr[2,2] <- t(A2S%*%A2S)/(2*N) + crossprod(a2,Sa2)/N
114 if (opcao==1) {
115 a <- (t(Hm)%*%Ph%*%Xt(Krho)%*%Hm)/N
116 Phi.rr <- (t(Hm)%*%Ph%*%Sigma%*%Xcbind(a,a))/N
117 }
118 else {
119
Phi.dd <- (t(Hm) %*% Sigma %*% Hm) / N
Phi.dr <- (t(Hm) %*% Sigma %*% cbind(a1, a2)) / N
)
Phi.o <- rbind(cbind(Phi.dd, Phi.dr), cbind(t(Phi.dr), Phi.rr))
J <- Glist %*% rbind(1, 2 * rho)
Phi.rr.inv <- solve(Phi.rr)
Zerol <- matrix(0, nrow=nrow(P), ncol=1)
Zer02 <- matrix(0, nrow=2, ncol=ncol(P))
Temp <- rbind(cbind(P, Zerol), cbind(Zer02, Phi.rr.inv %*% J %*% solve(t(t(J) %*% Phi.rr.inv %*% J))))
Omega <- t(Temp) %*% Phi.o %*% TtempM
return(list(Phi=Phi.rr, Omega=Omega))

# Main function
'GMMIV' <- function(xm, Ym, Mm, verbose=TRUE) {
  # step 1a: 2SLS estimator
call<-match.call()
if (identical(xm, Mm))
  Hm <- cbind(xm, Mm %*% Hm, Mm %*% Wm %*% XM)
else
  Hm <- cbind(xm, XM %*% XM, XM %*% XM, XM %*% XM, XM %*% XM, XM %*% XM)
require(Matrix) # for rankMatrix
p <- rankMatrix(Hm)
K <- ncol(xm)
if (p < K + 1) stop("Rank of instruments smaller than number of covariates")
require(MASS) # for ginv
Ph <- Hm %*% ginv(crossprod(Hm)) %*% X
Zm <- cbind(Xm, Wm %*% Y)
Zh <- Ph %*% Z
delta1 <- solve(crossprod(Zh, Zm)) %*% crossprod(Zh, Y)

# Step 1b: initial GMM Estimator of 'rho' based on 2SLS residuals
# Obtain the G matrices
G1 <- getG(deltal, Y, Zm, Mm)
Weight1 <- diag(2) # no weighting scheme in the first case

# Finds the parameter space for rho
rhorange <- rhospace(Mm)
if (verbose) cat("The parameter space for rho is: ", rhorange, "\n")

# obtain the estimated rho
opt1 <- optimize(MEq.interval=rhowrange, G1, Weight1, tol=.Machine$double.eps^0.5)
if (verbose) {
  cat("The initial GMM estimator of rho based on 2SLS residuals is: ", opt1$minimum, "\n")
  cat("The absolute value of the moment vector is: ", sqrt(opt1$objective), "\n")
}
rho1 <- opt1$minimum

# Step 1c: Efficient GMM estimator of 'rho' based on 2SLS residuals
#Weight2 <- getWeight(deltal, rh01, Y, Zm, Mm, Ph, 1) # note opcao
PhiOmega2 <- getPhiOmega(deltal, rh01, Y, Zm, Mm, Mm, G1, 1)
Weight2 <- PhiOmega2$Phi

# obtain the estimated rho
opt2 <- optimize(MEq.interval=rhowrange, G1, Weight2, tol=.Machine$double.eps^0.5)
if (verbose) {
  cat("The EFFICIENT GMM estimator of rho based on 2SLS residuals is: ", opt2$minimum, "\n")
  cat("The absolute value of the moment vector is: ", sqrt(opt2$objective), "\n")
}
rho2 <- opt2$minimum

# Step 2a: GS2SLS estimator
Ys <- Y - rho2 * Mm %*% Y
Xs <- Xm - rho2 %*% XM
Ws <- Wm - rho2 * Mm %*% Wm
Zs <- cbind(Xs, Ws %*% Y)
delta2 <- solve(crossprod(Zs, Ph %*% Zs)) %*% t(Zs) %*% Ph %*% Ys
# Step 2b: Efficient GMM estimator of rho based on GS2SLS residuals
G2 <- g2(delta.t, Y, Zm, Mm)
Weight3 <- getWeight(delta2, rho2, Y, Zm, Mm, P, 2) # note opcao
PhiOmega3 <- getPhiOmega(delta2, rho2, Y, Zm, Mm, Hm, G2, 2)
Weight3 <- PhiOmega3

opt3 <- optimize(Meq, interval = rhorange, G2, Weight3, tol = .Machine$double.eps^0.5)
if (verbose) {
  cat("The EFFICIENT GMM estimator of rho based on GS2SLS residuals is: ", opt3$minimum, 
        \n")
  cat("The absolute value of the moment vector is: ", sqrt(opt3$objective),\n")
}
rho3 <- opt3$minimum

# Step 2a (again)
Ys <- Y - rho3*Mm*X
Xs <- X - rho3*Mm*X
Ws <- Wm - rho3*Mm*Wm
Zs <- cbind(Xs, Ws*X)
delta3 <- solve(crossprod(Zs, Ph*x*Xs))%*%t(Zs)%*%Ph*X

G3 <- g3(delta3, Y, Zm, Mm)
PhiOmega4 <- getPhiOmega(delta3, rho3, Y, Zm, Mm, Hm, G3, 2)

theta.til <- c(delta1, rho2)
theta.hat <- c(delta3, rho3)
if (verbose) {
  cat("The estimate 1 (2SLS + GMM based on 2SLS) is: ", theta.til, \n")
  cat("The estimate 2 (GS2SLS + GMM based on GS2SLS) is: ", theta.hat, \n")
}
N <- length(Y)
residuals <- (diag(N) - rho3*Mm)%*%(Y - Zm*delta3)
ncoeF <- length(delta3)\nbeta.hat <- delta3[1:(ncoefs - 1)]
lambda.hat <- delta3[ncoeF]
fitted.values <- solve(diag(N) - lambda.hat*Wm)%*%X*X%*%beta.hat
rss <- sum(residuals)^2
mse <- sum((fitted.values - mean(fitted.values))^2) # note that there is a CONSTANT
r.squared <- mse/(ms)

return(list(call = call, coeff = coefficients(theta.hat), vcov = PhiOmega4$Omega/N, resid = residuals, 
            fitted = fitted.values, r.squared = r.squared, y = Y, x = X, m1 = Wm, m2 = Mm, 
            theta.hat = theta.hat, omegahat = PhiOmega4$Omega))

Listing 93: GS2SLS.R

# Step 2a to estimate 'delta' in the model
# y = X beta + lambda W y + u
# y = Z delta + u
# u = rho M u + e

'GS2SLS' <- function(Xmat, Ymat, Wmat, Mmat, verbose=TRUE) {
  Rho.til <- EGMM(Xmat, Ymat, Wmat, verbose)
  Ystar <- Y - Rho.til*Mmat%*%X
  Xstar <- X - Rho.til*Mmat%*%X
  Wstar <- Wmat - Rho.til*Mmat%*%Wmat
  Zstar <- cbind(Xstar, Wstar%*%Y)
  if (identical(Wmat, Mmat))
    Hmat <- cbind(Xmat, WX, W2X)
  else
    Hmat <- cbind(Xmat, WX, W2X)
  PH <- Hmat%*%ginv(crossprod(Hmat))%*%t(Hmat)
  Zhat <- PH%*%Zstar
  Dhat <- solve(crossprod(Zhat, Zstar)%*%crossprod(Zhat, Ystar)
  return(Dhat)
}

Listing 94: IGMM.R
# Step 1b to estimate 'rho' in the model
# y = X beta + lambda W y + u
y = Z delta + u
u = rho u + e

'gmm' <- function(Xmat, Ymat, Wmat,/MM, verbose=TRUE) {
  Dtilt <- E2SLS(Xmat, Ymat, Wmat, Mmat)
  Emat <- cbind(Xmat, Wmat%*%Y)
  Utilt <- Y - Z%*%Dtilt
  Ubtilt <- Mmat%*%Ubtilt
  Ub2tilt <- Mmat%*%Ub2tilt
  N <- length(Y)
  g <- G <- NA
  g[1,1] <- (crossprod(Ub2tilt) - tr(Mmat%*%diag(Ub2tilt)%*%t(Mmat)))/N
  g[2,1] <- (crossprod(Ubtilt)/Ubtilt)/N
  G[1,1] <- 2*(crossprod(U2btilt, Ubtilt) - tr(Mmat%*%diag(Ubtilt*Utilt)%*%t(Mmat)))/N
  G[1,2] <- -1*(crossprod(U2btilt) + tr(Mmat%*%diag(Ub2tilt)%*%t(Mmat)))/N
  G[2,1] <- (crossprod(Ubtilt, U2btilt) + crossprod(Ubtilt))/N
  G[2,2] <- -1*(crossprod(Ub2tilt, U2btilt))/N

  MEQ <- function(embvec) {
    MC <- g - G%*%embvec
    return(crossprod(MC))
  }

  # Finds the parameter space for rho
  EVAL <- eigen(Mmat, only.values=TRUE)$val
  ReEVAL <- Re(EVAL (abs(Im(EVAL)) <= .Machine$double.eps^0.5)) # selects only REAL eigenvalues
  RHOSPACE <- 1/range(ReEVAL)
  #check for zero eigenvalues on the left
  if (RHOSPACE[1]==Inf) RHOSPACE[1] <- -Inf
  # check whether the parameter space is contained in (0,1)
  if (RHOSPACE[1] < -1) RHOSPACE[1] <- -1
  RHOSPACE <- RHOSPACE + c(1, -1)*.Machine$double.eps^0.25
  if (verbose) cat("The parameter space for rho is: ", RHOSPACE,"\n")

  OPTRES <- optimize(MEQ, interval=RHOSPACE, tol=.Machine$double.eps^0.5)
  if (verbose) {
    cat("The initial GMM estimator of rho based on 2SLS residuals is: ", OPTRES$minimum ,"\n")
    cat("The absolute value of the moment vector is: ", sqrt(OPTRES$objective),"\n")
  }
  return(OPTRES$minimum)
}

'invariantes' <-
function(k) {
  sizek <- length(k)
  k1 <- k + k[2]^(-1:sizek)/2 # invariantes
  return(c(0,1,k1[3:sizek]))
}

'k.t' <-
function(omega, rho0, rho1, rho, md=Mdefault, hn=1) {
  n <- dim(md)[1]
  return(-0.5*log(det(diag(n)-2*omega*Cr(rho0, rho1, rho, md)/sqrt(n/hn)))))
}

'k.V' <-
function(omega, v, rho0, rho, md=Mdefault) {
  n <- dim(md)[1]
  return(-0.5*log(det(diag(n)-2*omega*V(v, rho0, rho, md)/sqrt(n)))))
}
Listing 98: Linv.R

```r
'Linv' <-
function (rho, m1, m2=NULL) return(solve(L(rho, m1, m2)))
```

Listing 99: lr

```r
'lr' <-
function (rho, m1, m2=NULL) {
  return(diag(ncol(m1))-w(rho, m1, m2))
}
```

Listing 100: LRACV.T.R

```r
# Calculates the critical value based on the solution to # the tail probability integral.
'LRACV.T' <-
function (alpha, rho1, m1, m2=NULL, normalize=TRUE) {
  k <- cumulantsT(2, rho1, 0, m1, m2) # Under the NULL
  if (rho1>0) {
    guess1 <- qnorm(0.55, mean=k[1], sd=sqrt(k[2]))
    guess2 <- qnorm(0.999, mean=k[1], sd=sqrt(k[2]))
    ca <- uniroot(function(t) LAPT(t, rho1, 0, m1, m2) - alpha,
                lower=guess1, upper=guess2, tol=Machine$double.eps^-0.5)$root
  } else {
    guess1 <- qnorm(0.001, mean=k[1], sd=sqrt(k[2]))
    guess2 <- qnorm(0.45, mean=k[1], sd=sqrt(k[2]))
    ca <- uniroot(function(t) LAPT(t, rho1, 0, m1, m2) -1+alpha,
                lower=guess1, upper=guess2, tol=Machine$double.eps^-0.5)$root
  }
  return(list(norm=ca, std=(ca-k[1])/sqrt(k[2])))
}
```

Listing 101: lrvc hurdle

```r
# Calculates the critical value based on the solution to # the tail probability integral.
'LRACV.V' <-
function (alpha, rho1, m1, m2=NULL, normalize=TRUE) {
  k2 <- 2*trPowerM(A_r(rho1, m1, m2))/ncol(m1) # mean of Vn (approx) Laplace
  k2 <- 2*trPowerM(A_r(rho1, m1, m2))/ncol(m1) # var of Vn (approx) Laplace
  if (rho1>0) {
    guess1 <- qnorm(0.55, mean=k1, sd=sqrt(k2))
    guess2 <- qnorm(0.999, mean=k1, sd=sqrt(k2))
    ca <- uniroot(function(t) LAPV(t, rho1, 0, m1, m2) - alpha,
                lower=guess1, upper=guess2, tol=Machine$double.eps^-0.5)$root
  } else {
    guess1 <- qnorm(0.001, mean=k1, sd=sqrt(k2))
    guess2 <- qnorm(0.45, mean=k1, sd=sqrt(k2))
    ca <- uniroot(function(t) LAPV(t, rho1, 0, m1, m2) -1+alpha,
                lower=guess1, upper=guess2, tol=Machine$double.eps^-0.5)$root
  }
  return(list(norm=ca, std=(ca-k1)/sqrt(k2)))
}
```

Listing 102: LRAPT.R

```r
'LAPT' <-
function(t, rho1, rho, m1, m2=NULL) {
  mat2 <- diag(ncol(m1)) -2*gammahat*mat
  mat2[1]<-solve(mat2)
  a<--sign(gammahat)*sqrt(2*(gammahat*tr(mat2inv%*%Xmat)+0.5*log(det(mat2))))
  b<--gammahat*sqrt(2*trPowerM(mat2inv%*%Xmat, 2))
  k3<--4*trPowerM(mat2inv%*%Xmat, 3)/8*trPowerM(mat2inv%*%Xmat, 2)^3/2
  if (abs(gammahat)<Machine$double.eps^-0.25) return(1-pnorm(a)+dnorm(a)*k3*sqrt(ncol(m1))/6)
  else return(1-pnorm(a)+dnorm(a)*(1-b^-1/a))
}
```
Listing 103: lapr.r

1 'LRAPV' <-
2 function(v, rho1, rho, m1, m2=NULL) {
3 mat <- Dv(v, rho1, rho, m1, m2)
4 gammahat <- saddlepoint.V(v, rho1, rho, m1, m2)
5 mat2 <- diag(ncol(mat)) - 2 * gammahat * mat
6 mat2inv <- solve(mat2)
7 if (abs(det(mat2) - 1) < .Machine$double.eps^(-0.5)) a <- 0
8 else a <- -sign(gammahat) * sqrt(log(det(mat2)))
9 b <- gammahat + sqrt(2 * trPowerM(mat2inv %*% mat, 2))
10 c <- trPowerM(mat2inv %*% mat, 3) / (2 * trPowerM(mat2inv %*% mat, 2))^(3/2)
11 if (abs(gammahat) < .Machine$double.eps^(-0.5))
12 { cat("gammahat MENOR que tolerancia!!!\n\n\n")
13 return(1 - pnorm(a) + dnorm(a) * k3 / sqrt(ncol(mat))/6) # VERIFICAR ESTE RESULTADO
14 }
15 else return(1 - pnorm(a) + dnorm(a) * (1/b - 1/a))
16 }

Listing 104: matrixpower.R

1 'matrix.power' <- function(mat, n)
2 {
3 # test if mat is a square matrix
4 # treat n < 0 and n = 0 -- this is left as an exercise
5 # trap non-integer n and return an error
6 if (n == 1) return(mat)
7 result <- diag(1, ncol(mat))
8 while (n > 0) {
9 if (n %% 2 != 0) {
10 result <- result %*% mat
11 n <- n - 1
12 }
13 mat <- mat %*% mat
14 n <- n / 2
15 }
16 return(result)
17 }

Listing 105: MLEgeneralSAR.R

1 # Estimates the following model by ML
2 # y = rho M1 y + X beta + u
3 # u = lambda M2 u + e
4 # e ~ N(0, sigma I)
5 'MLEgeneralSAR' <- function(x, y, m1, m2=m1, verbose=TRUE, vartype=1) {
6 # Finds the parameter space for rho
7 rhospace <- function(Mmat) {
8 EVAL <- eigen(Mmat, only.values=TRUE)$val
9 ReEVAL <- Re(EVAL[abs(Im(EVAL)) < .Machine$double.eps^0.5]) # selects only REAL
10 eigenvalues
11 RHOSPACEx <- 1/Range(ReEVAL)
12 # check for zero eigenvalues on the left
13 if (RHOSPACEx[1]==Inf) RHOSPACEx[1] <- -Inf
14 # check whether the parameter space is contained in (0,1)
15 if (RHOSPACEx[1] < -1) RHOSPACEx[1] <- -1
17 RHOSPACEx <- RHOSPACEx + c(1.0,1.0) * .Machine$double.eps^0.25
18 return(RHOSPACEx)
19 }
20 require(MASS)
21 call <- match.call()
22 rhospace <- rhospace(m1)
23 lambda.space <- rhospace(m2)
24 if (verbose) { cat("The parameter space for rho is: ", rhospace, "\n")
25 cat("The parameter space for lambda is: ", lambda.space, "\n")
26 }
27 opt <- vector("list",10)
absgrad<-1e6 # any LARGE value
modselected<-11 # larger than the number of random starting values
bigloops<-0
repeat {for (i in 1:10) { # tries some random starting values
  rho0 <- runif(1, min=rho_space[1], max=rho_space[2])
  lambda0 <- runif(1, min=lambda_space[1], max=lambda_space[2])
  opt[[1]]<-optim(c(rho0, lambda0), negLLc, gr=neggradLLc, x, y, m1, m2,
      method="L-BFGS-B", lower=c(rho_space[1], lambda_space[1]),
      upper=c(rho_space[2], lambda_space[2]), hessian=FALSE)
  currentabsgrad <- sqrt(crossprod(neggradLLc(opt[[1]]$par,x,y,m1,m2)))
  if (currentabsgrad < absgrad) {
    absgrad<-currentabsgrad
    modselected<-i
  }
  #cat("convergence: ", opt[[1]]$convergence)
  if (verbose) {
    cat("Solucao \("x\), ": ", opt[[1]]$par, "LokLike=",
      -1*negLLc(opt[[1]]$par,x,y,m1,m2),
      " abs(grad")", currentabsgrad, "msg", opt[[1]]$convergence, ",n")
    if (opt[[1]]$convergence==0 & currentabsgrad <=1e-4) break
  }
  theta0<-opt[[modselected]]$par
  finalopt<-optim(theta0, negLLc, gr=neggradLLc, x, y, m1, m2, method="L-BFGS-B",
      lower=c(rho_space[1], lambda_space[1]),
      upper=c(rho_space[2], lambda_space[2]), hessian=FALSE,
      control=list(factr=1e7, pgtol=1e-6))
  #control=list(factr=100)
  bigloops<-bigloops+1
  print(finalopt$message)
  if (finalopt$convergence==0 & bigloops==10) break
  }
  if (bigloops==10 & finalopt$convergence!=0) { # WARNING! Optimization did not achieve convergence
    cat("Solucao (final) ": " finalopt$par, "LokLike=",
      -1*negLLc(finalopt$par,x,y,m1,m2),
      " abs(grad") = sqrt(crossprod(neggradLLc(finalopt$par,x,y,m1,m2))), ",n")
  }
  rho_hat<-finalopt$par[[1]]
  lambda_hat<-finalopt$par[[2]]
  # Calculates the other parameters
  kh <- L(lambda_hat,m2)
  kh_x <- kh%*%X
  kh_lh_y <- kh%*%lh%*%y
  beta_hat <- solve(crossprod(kh_x))%*%(kh_x)%*%kh_lh_y
  kh_xhb <- kh_x%*%beta_hat
  sigma2_hat <- (crossprod(kh_lh_y) - 2*crossprod(kh_lh_y,kh_xhb) +
      crossprod(kh_xhb) )/length(y)
  theta_hat<-c(beta_hat,rho_hat,lambda_hat,sigma2_hat)
  names(theta_hat)<-c(colnames(x),"rho","lambda","sigma2")
  residuals <- kh_lh_y-kh_x hb
  fitted.values <- y-solve(kh%*%lh)%*%y
  residuals ~2
  rss <- sum(residuals^2)
  mss <- sum((fitted.values-mean(fitted.values))^2) # note that there is a CONSTANT
  r.squared <- mss/(mss+rss)
  if (verbose) { cat("ML estimate of theta is: 
")
    print(theta_hat)
  }
  if (vartype==1) vcov<-VarIMgeneralSAR(theta_hat,x,m1,m2)
  else vcov<-VarIMgeneralSAR(theta_hat,x,y,m1,m2)
  # Calculates likelihood
  # for the full model evaluated at theta_hat
  llfull <- -1*negLfull(theta_hat,x,y,m1,m2)
  # Calculate the estimated theta imposin NO SPATIAL (rho=0 and lambda=0)
  # is the same as OLS for beta
  beta_constrained <- solve(crossprod(x))%*%(x)%*%y
  # and the usual MLE for sigma2
  sigma2_constrained <- crossprod(y-x%*%beta_constrained)/length(y)
  theta_constrained <- c(beta_constrained,0,0,sigma2_constrained)
  # Calculates the likelihood for the constrained model
  llconstrained <- -1*negLfull(theta_constrained,x,y,m1,m2)
# Calculates the LR statistic for testing the null
# of NO SPATIAL effect (Chi-square(2))
LRnospatial <- 2*(llfull - llconstrained)
if (verbose) { cat("LR test for null of no spatial effect: ", LRnospatial, ", "
  cat("Critical levels (10%, 5%, 1%): ", qchisq(1-c(0.1, 0.05, 0.01), df=2), ", "
}
list(call=call, coefficients=theta_hat,
  vcov=vcov, min=finalopt$value, details=finalopt,
  museslog=muselog, negLLfull=llfull, method="L-BFGS-B", residuals=residuals,
  fitted.values=fitted.values, r.squared=r.squared, y=y, x=x, m1=m1, m2=m2,
  LRstat=LRnospatial)

# Estimates the following model by ML
# y = rho M y + X beta + e
# e ~ N(0, sigma2 I)

'MLESAR' <- function(x, y, m, verbose=TRUE, vartype=1) {
  if (any(abs(In(eigen(m)$val))>.Machine$double.eps^-0.5)) {
    cat("WARNING: this only works with SYMMETRIC matrices\n")
    stop("complex eigenvalues were found")
  }
  em <- Re(eigen(m, only.values=TRUE)$val)
  require(MASS)
call<-match.call()
rho.space <- 1/length(em)
# check for zero eigenvalues
if (rho.space[1]==Inf) rho.space[1]<- -Inf
# check whether the parameter space is contained in (0,1)
if (rho.space[1] < -1) rho.space[1]< -1
rho.space <- rho.space + c(1, -1)*.Machine$double.eps^-0.25
if (verbose) cat("The parameter space for rho is: ", rho.space, ",\n")
opt<-optimize(f=negLLc_sar, interval=rho.space, x, y, m,
  maximum=FALSE, tol=.Machine$double.eps^-0.25)
abograd <- abs(abgradLLc_sar(opt$minimum,x,y,m))

if (verbose) {
  cat("Solution (final): ", opt$minimum, "LokLike=",
    -1*negLLc_sar(opt$minimum,x,y,m),
    "abs(grad)="", abograd, ",\n")
}
 rho_hat<-opt$minimum
 lh<- 1/(rho_hat, m)
 lh_y <- log1p(y)
 beta_hat < solve(crossprod(x))*%t(x)*%x lh_y
 x_bh <- x%t(beta_hat)
 residuals <- lh_y - x_bh
 sigma2_hat <- crossprod(residuals)/length(y)
 theta_hat <- c(beta_hat, rho_hat, sigma2_hat)
 names(theta_hat)<-c(colnames(x), "rho", "sigma2")
 fitted.values <- y-solve(lh)x%t(residuals
 rss <- sum(residuals^2)
 mss <- sum((fitted.values-mean(fitted.values))^2) # note that there is a CONSTANT
 regressor in X
 r.squared <- mss/(mss+rss)

if (verbose) { cat("ML estimate of theta is: \n")
  print(theta_hat)
}
 if (vartype=1) vcov<-VarMSAR(theta_hat,x,m)
 else vcov<-VarHSAR(theta_hat,x,y,m)
 # Calculates likelihood
 # for the full model evaluated at theta_hat
 llfull <- -1*negLLfull_sar(theta_hat,x,y,m)
 # Calculate the estimated theta imposing NO SPATIAL (rho=0)
 # is the same as OLS for beta
 beta_constrained <- solve(crossprod(x))*%t(x)*%xy
 # and the usual MLE for sigma2
Listing 106: MLESAR.R
Listing 107: MLESE.R

```r
# Estimates the following model by ML
# y = X beta + u
# u = lambda M u + e
t # e ~ N(0, sigma^2)

MLESE <- function(x, y, m, verbose=TRUE, var.type=1) {
  if (any(abs(IM(eigen(User1.val)>Machine$double.eps*0.5))) {
    return("WARNING: this only works with SYMMETRIC matrices")
  }  
  diag <- Re(eigen(User1.val, only.values=TRUE)$val)
  require(MASS)
  call <- match.call()
  lambda.space <- 0:1/range(en)
  # check for zero eigenvalues
  if (any(lambda.space[1]==Inf) lambda.space[1]<- -Inf
  # check whether the parameter space is contained in (0,1)
  if (any(lambda.space[1]< -1 lambda.space[1]<1)
  lambda.space <- lambda.space + c(1, -1)*Machine$double.eps*0.25
  if (verbose) cat("The parameter space for lambda is: ", lambda.space,"\n")
  opt <- optimize(f=negLLC.se, interval=lambda.space, x, y, m,
                   maximum=FALSE, tol=Machine$double.eps*0.25)
  absgrad <- abs(neggradLLC.se(opt$minimum,x,y,m))
  if (verbose)
    cat("Solution (final): ", opt$minimum, "LokLike=",
         -1*negLLC.se(opt$minimum,x,y,m),
         "abs(Grad)= ", absgrad,"\n")
  lambda_hat <- opt$minimum
  kh <- L(lambda_hat, m)
  khx <- kh%*%x
  khy <- kh%*%y
  bh <- solve(crossprod(khx))%*%(khx)%*%khy
  x_bh <- x%*%beta_hat
  residuals <- kh%*%(y-x_bh)
  sigma2_hat <- c(residuals)/length(y)
  theta_hat <- c(beta_hat, lambda_hat, sigma2_hat)
  names(theta_hat) <- c(colnames(x), "lambda", "sigma2")
  fitted.values <- x_bh
  r^2 <- sum(residuals^2)
  mss <- sum((fitted.values-mean(fitted.values))^2)  
  if (verbose) { cat("ML estimate of theta is: \n")
    print(theta_hat)
  }
  if (var.type==1) vcv <- varMSE(theta_hat, x, m)
  else vcv <- varHSE(theta_hat, x, y, m)
  # Calculates likelihood
```

# for the full model evaluated at theta_hat
llfull <- -1*negLLfull_se(theta_hat,x,y,m)

# Calculate the estimated theta imposing NO SPATIAL (rho=0)
# is the same as OLS for beta
beta_constrained <- solve(crossprod(x))%*%t(x)%*%y
# and the usual MLE for sigma2
sigma2_constrained <- crossprod(y-x%*%beta_constrained)/length(y)
theta_constrained <- c(beta_constrained,0,sigma2_constrained)
# Computes the likelihood for the constrained model
llconstrained <- -1*negLLfull_se(theta_constrained,x,y,m)
# Computes the LR statistic for testing the null
# of NO SPATIAL effect (Chi-square(2))
llnspatial <- 2*(llfull-llconstrained)

if (verbose) { cat("LR test for null of no spatial effect:" , llnspatial, 
"\n")
cat("Critical levels (10%,5%,1%): ", qchisq(1-c(0.1,0.05,0.01), df=1), 
"\n")
}
list(call=call, coefficients=theta_hat, 
vcov=vcov, min=opt$p$objective, details=opt,
minorlogl=negLLfull_sar, method="optimize", residuals=residuals, 
fitting.values=fitted.values, r.squared=r.squared, y=y, x=x, m=m, 
llstat=llnspatial)

Listing 108: MonteCarloCurvature V.R

'MonteCarloCurvature_V' <- function(S,m1,m2=NULL,verbose=TRUE,seed=2000) {
  ldot<-NA
  lddot<-NA
  n<-ncol(m1)
  w1<-W(0,m1,m2)
  w2<-ifelse(is.null(m2),0,2*m2)
  set.seed(seed)
  for (i in 1:S) {
    U<-rnorm(n,0,sqrt(1)) # does not matter which standard deviation to use
    Y<-U # Under the NULL hypothesis (rho=0)
    ldot[i]<-n*t(Y)%*%w1%*%Y/crossprod(Y)
    lddot[i]<- -tr(matrix.power(w1,2)+w2) + n*( t(Y)%*%(w2-t(w1))%*%w1)%*%Y/crossprod(Y) + 2* (t(Y)%*%w1%*%Y/crossprod(Y))%*%2
    if (verbose) cat("Simulation ",s," ldot ",l[i]," lddot ",l[i]," \n")
  }
  res<-list(id=ldot,lrd=ldot)
  return(res)
}

Listing 109: MonteCarloCV H1 V.R

'MonteCarloCV_H1_V' <- function(S,alpha, rho1,m1,m2=NULL,verbose=TRUE,seed=2000) {
  vs<-NA
  ca<-NA
  ca_s<-NA # standardized critical value
  set.seed(seed)
  Lmenosum<-Linr(rho1,m1,m2)
  for (s in 1:S) {
    U<-rnorm(ncol(m1),0,sqrt(1)) # does not matter which standard deviation to use
    Y<-Lmenosum%*%U # Under the ALTERNATIVE hypothesis (rho=rho1)
    vs[s]<-Vn(Y,rho1,m1,m2)
    ord<-(alpha*(s+1))
    if (ceiling(ord)==floor(ord)) {
      ca[s]<-sort(vs,decreasing=TRUE)[ord]-mean(vs)/sd(vs)
    } else {
      ca[s]<-sort(vs,decreasing=TRUE)[ord]
    }
    if (verbose) cat("The critical value for simulation s=",s," is ",c(ca[s],ca_s[s])," \n")
  }
  res<-list(vs,ca,ca_s)
  return(res)
}
Listing 10: MonteCarloCV T.R

```r
'MonteCarloCV_T' <- function(S, alpha, rho1, sigma2, m1, m2=NULL, verbose=TRUE, seed=2000) {
  ts<-NA
  ca<-NA
  ca_s<-NA # standardized critical value
  set.seed(seed)
  for (s in 1:S) {
    U<-rnorm(ncol(m1),0,sqrt(sigma2))
    Y<-U # Under the NULL hypothesis (rho=0)
    ts[s]<-t(n(Y,rho1, sigma2, m1, m2))
    ord<-alpha*(s+1)
    if (ceiling(ord)==floor(ord)) {
      ca_s[s]<-(sort(ts, decreasing=TRUE)[ord]-mean(ts))/sd(ts)
      ca[s]<-sort(ts, decreasing=TRUE)[ord]
    }
    if (verbose) cat("The critical value for simulation s","s", " is ",c(ca[s],ca_s[s]),"\n")
  }
  res<-list(ts,ca,ca_s)
  return(res)
}
```

Listing 11: MonteCarloCV V.R

```r
'MonteCarloCV_V' <- function(S, alpha, rho1, m1, m2=NULL, verbose=TRUE, seed=2000) {
  vs<-NA
  ca<-NA
  ca_s<-NA # standardized critical value
  set.seed(seed)
  for (s in 1:S) {
    U<-rnorm(ncol,m1),0,sqrt(1)) # does not matter which standard deviation to use
    Y<-U # Under the NULL hypothesis (rho=0)
    vs[s]<-Vn(Y,rho1, m1, m2)
    ord<-alpha*(s+1)
    if (ceiling(ord)==floor(ord)) {
      ca_s[s]<-(sort(vs, decreasing=TRUE)[ord]-mean(vs))/sd(vs)
      ca[s]<-sort(vs, decreasing=TRUE)[ord]
    }
    if (verbose) cat("The critical value for simulation s","s", " is ",c(ca[s],ca_s[s]),"\n")
  }
  res<-list(vs,ca,ca_s)
  return(res)
}
```

Listing 12: MonteCarloPower V.R

```r
'MonteCarloPower_V' <- function(S, rho, m1, m2=NULL, verbose=TRUE, seed=2000) {
  vs<-NA
  set.seed(seed)
  Lmenosum<-Linv(rho, m1, m2)
  for (s in 1:S) {
    U<-rnorm(ncol,m1),0,sqrt(1)) # does not matter which standard deviation to use
    Y<-Lmenosum*X # Under the ALTERNATIVE hypothesis (rho=rho1)
    vs[s]<-Vn(Y,0,m1, m2) # note here that we consider only LUMPY test (rho1=0)
    if (verbose) cat("Simulation number s","s","\n")
  }
  return(vs)
}
```

Listing 13: neggradLLc.R

```r
'neggradLLc' <- function(theta, x, y, m1, m2=m1) {
  # Calculates the gradient of the CONCENTRATED log likelihood function
  n=length(y)
  rho<-theta[1]
  lambda<-theta[2]
  k<-L(lambda, m2)
  l<-L(rho, m1)
  kx<-k%*%x
  Pxx<-kx%*%ginv(crossprod(kx))%*%t(kx)
```
```r
Mx <- diag(n) - Px
ky <- kx %*% y
kwx <- kx %*% wx
zhat <- crossprod(kwx, Mwx %*% kwy)
eq_rho <- n * crossprod(kly, Mwx %*% kly) / zhat - tr(solve(1) %*% m1)
eq_lambda <- n * t(kly) %*% solve(Mx %*% x)
Mwx <- Mwx %*% m2 - Mwx %*% m2 -
xinv(crossprod(xx)) %*% t(kx) %*% Mwx %*% m2 -
tr(solve(k) %*% m2)
f <- c(eq_rho, eq_lambda)
return(-1 * f)
}

Listing 114: neggradLLe_sar.R

'neggradLLe_sar' <- function(rho, x, y, m) {
  # Calculates the gradient of the CONCENTRATED log likelihood function
  # for DGP
  # y = rho M y + Xb + e
  # e ~ N(0, sigma2 I)
  n <- length(y)
  l <- L(rho, m)
  Px <- x %*% solve(crossprod(x)) %*% t(x)
  Mx <- diag(n) - Px
  ly <- -1 %*% y
  wx <- -Mx %*% ly
  zhat <- crossprod(wx, Mwx %*% ly)
eq_rho <- n * crossprod(ly, Mwx %*% wx) / zhat - tr(solve(1) %*% m)
  return(-1 * eq_rho)
}

neggradMLE <- function(theta, m, w)
-1 * gradMLE(theta, m, w)

gradMLE <- function(theta, m, w) {
}

Listing 115: neggradLLe_sc.R

'neggradLLe_sc' <- function(lambda, x, y, m) {
  # Calculates the gradient of the CONCENTRATED log likelihood function
  # for DGP
  # y = Xb + u
  # u ~ lambda M u + e
  # e ~ N(0, sigma2 I)
  n <- length(y)
  l <- L(lambda, m)
  kx <- kx %*% kx
  Pkw <- kwx %*% x
  Mx <- diag(n) - Pkx
  ky <- kx %*% ky
  zhat <- crossprod(ky, Mwx %*% ky)
  Mwxm <- Mwx %*% kx
  ninv(crossprod(k) %*% t(kx)) %*% t(kx) %*% Mwxm %*% diag(n) -
xinv(crossprod(kx)) %*% t(kx) %*% ky %*% zhat -
  tr(solve(k) %*% m)
  return(-1 * eq_lambda)
}

neggradMLE <- function(theta, m, w)
-1 * gradMLE(theta, m, w)

gradMLE <- function(theta, m, w) {
}

Listing 116: negLLe_R

'negLLe' <- function(theta, x, y, m1, m2 = m1) {
  # calculates the negative log likelihood of the CONCENTRATED model
  rho <- theta[1]
```
\[
\begin{align*}
\lambda & \leftarrow \theta[2] \\
n & \leftarrow \text{length}(y) \\
k & \leftarrow \lambda[\lambda, 2] \\
l & \leftarrow \lambda[\lambda, 1] \\
kx & \leftarrow k^x \\
kly & \leftarrow k^x \cdot y \\
\text{# Mx} & \leftarrow \text{diag}(n) - k^x \cdot x \\
\text{# Mxx} & \leftarrow \text{diag}(n) - k^x \cdot y \cdot \text{solve}(\text{crossprod}(kx)) \cdot k^x \\
\text{# zhat} & \leftarrow \text{crossprod}(kly, MxX) \\
\text{# return} & \leftarrow (-1 * (-n/2 * (\log(x^2) + 1) + \text{determinant}(\text{I} \cdot \text{modulus}[1] + \\
& \text{determinant}(k \cdot \text{modulus}[1] - n/2 * \text{log}(\text{zhat})))) \\
\end{align*}
\]

```
'negLLc_sar' <- function(rho, x, y, m) {
  # calculates the negative log likelihood of the CONCENTRATED SAR model:
  # y = rho \cdot W \cdot y + X \cdot b + e
  # e \sim \mathcal{N}(0, \sigma_2^2)
  n <- length(y)
  l <- L(rho, m)
  ly <- l \cdot y
  Mx <- diag(n) - x \cdot x \cdot \text{solve}(\text{crossprod}(x)) \cdot x
  zhat <- crossprod(kly, Mxx)
  return(-1 * (-n/2 * (\log(x^2) + 1) + \text{determinant}(\text{I} \cdot \text{modulus}[1] - n/2 * \text{log}(\text{zhat}))))
}
```

```
'negLLc_se' <- function(lambda, x, y, m) {
  # calculates the negative log likelihood of the CONCENTRATED SAR model:
  # y = X \cdot b + u
  # u = lambda \cdot u + e
  # e \sim \mathcal{N}(0, \sigma_2^2)
  n <- length(y)
  k <- L(lambda, m)
  ky <- k \cdot y
  kx <- k \cdot x
  Mx <- diag(n) - k \cdot x \cdot x \cdot \text{solve}(\text{crossprod}(kx)) \cdot x
  zhat <- crossprod(kly, Mxx)
  return(-1 * (-n/2 * (\log(x^2) + 1) + \text{determinant}(\text{I} \cdot \text{modulus}[1] - n/2 * \text{log}(\text{zhat}))))
}
```

```
'negLLfull' <- function(theta, x, y, m, m2 = m) {
  # calculates the negative log likelihood of the FULL model
  p <- length(theta)
  n <- nrow(x)
  beta <- theta[1:(p-3)]
  rho <- theta[p-2]
  lambda <- theta[p-1]
  sigma2 <- theta[p]
  k <- L(lambda, m2)
  l <- L(rho, m1)
  kly <- k \cdot l
  kxx <- k \cdot k \cdot \beta
  z <- crossprod(kly) - 2 * crossprod(kly, kxx) + crossprod(kxx)
  return(-1 * (-n/2 * (\log(2 \cdot pi) + n/2 * \log(sigma2) - 1/(2 * sigma2) * z +
& \text{determinant}(x \cdot \text{modulus}[1] + \text{determinant}(1 \cdot \text{modulus}[1])))
}
```

```
'negLLfull_sar' <- function(theta, x, y, m) {
  # calculates the negative log likelihood of the FULL model
  p <- length(theta)
  n <- nrow(x)
  beta <- theta[1:(p-3)]
  rho <- theta[p-2]
}
```

```
'negLLfull_se' <- function(theta, x, y, m) {
  # calculates the negative log likelihood of the FULL model
  p <- length(theta)
  n <- nrow(x)
  beta <- theta[1:(p-3)]
  rho <- theta[p-2]
  ```
Listing 121: NLSSAR.R

```r
# This program performs the NLS estimation for the model
# y = X beta + u
# u = Rho M u + e

# re-write the model
# y = rho M y + (I - rho M) X beta + e
# y = X(gamma) + e
# gamma = [ beta, rho ]

'NLSSAR' <- function(Xm,Y,Mm,verbose=TRUE,tol=1e-6,maxit=100) {
  z <- function(gamma,Xm,Y,Mm) {
    k <- length(gamma)
    beta <- gamma[1:(k-1)]
    rho <- gamma[k]
    return(rho*Mm%*%Y + (diag(length(Y))-rho*Mm)%*%Xm%*%beta)
  }
  Q <- function(gamma,Xm,Y,Mm) crossprod(Y-z(gamma,Xm,Y,Mm))/length(Y)
  Z <- function(gamma,Xm,Y,Mm) {
    k <- length(gamma)
    beta <- gamma[1:(k-1)]
    rho <- gamma[k]
    n <- length(Y)
    return(cbind((diag(n)-rho*Mm)%*%Xm,Mm%*%(Y-Xm%*%beta)))
  }
  g <- function(gamma,Xm,Y,Mm) (-2*t(Z(gamma,Xm,Y,Mm))%*%(Y-z(gamma,Xm,Y,Mm)))/length(Y)

  # set the initial parameter vector (rho=0, beta=betaOLS)
  beta0 <- lm.fit(Xm,Y)$coefficients
  gamma0 <- c(beta0,0)

  repeat{
    # runs the GNR (GAUSS-NEWTON regression)
    if (verbose) cat("Coeff. estimate at iteration ","n")
    b0 <- lm.fit(Z(gamma0,Xm,Y,Mm),Y-z(gamma0,Xm,Y,Mm))$coefficients
    # checks convergence
    if (verbose) cat('Convergence criteria:','n')
    if (-1*t(gamma0,Xm,Y,Mm))%*%b0 < tol) break
    # finds alpha0
    alpha0 <- crossprod(b0,gamma0)/crossprod(b0)
    if (verbose) cat("alpha=-",alpha0,"n")
    # iterates
    gamma0 <- gamma0 + alpha0*b0
    nit <- nit+1
    if (nit>maxit) cat("Maximum number of iterations reached\n")
    break
  }
  return(gamma0)
}
```

Listing 122: plotMatrix.R

```r
# funcao para imprimir uma matriz (heat)
# se for um array com dimensao maior do que 2
# a terceira dimensao sao os fatores
'plotMatrix' <- function(array, order=FALSE) {
```

---

*Note: The above code listings are represented as text due to the limitations of the text-based model.*
if (order) ord<-order.dendrogram(as.dendrogram(hclust(dist(array))))
N<-dim(array)[1] #nao verifica as dimensoes...
levelplot(array[1:N,1:1], col.regions=gray(100:0/100),
scales=list(draw=FALSE), xlab="spatial unit", ylab="spatial unit")
}

'powerBNJ_V' <-
function(rho.range, alpha=0.05, m1,m2=NULL, verbose=FALSE) {
power<-NA
for (j in 1:length(rho.range)) {
  rho1 <- rho.range[j]
c_a <- BNJAV_V(alpha, rho1,m1,m2)
  if (rho1 > 0) {
    power[j] <- BNJAV(c_a$norm, rho1,rho1,m1,m2)
    if (j>1) if (power[j]<power[j-1]) {
      cat("WARNING! Power envelope has DECREASED!
    cat("Check rho=" ,rho.range[j-1]," and its neighbors.
    }
  }
  else power[j] <- 1-BNJAV(c_a$norm, rho1,rho1,m1,m2)
  if (verbose) cat("For rho: ",rho1," the critical level is ",c_a$std," and power = ",
power[j],"\n")
}
return(power)
}

'powerBNJ_E' <-
function(rho.range, alpha=0.05, md=Mdefault, hn=1, verbose=FALSE) {
# repeat the procedure outlined in the paper for all rho in rho.range
h<-createHermite(h) # for Cornish-Fisher
power<-NA # re-sets the output variable
for (ind in 1:length(rho.range)) {
  rho <- rho.range[ind]
  # Calculates the cumulants of T_n under the NULL and ALTERNATIVE
  k <- cumulantsT(4,0,rho,0,md,hn)
  ka <- cumulantsT(4,0,rho,md,hn)
  # calculate the invariants
  ki <- invariants(k)
  kai <- invariants(ka)
  # obtain the critical level c_a by Cornish-Fisher (only valid when smax=4)
  if (rho > 0) {t_a <- qnorm(1-alpha) # the 1-alpha quantile of the Gaussian
  else t_a <- qnorm(alpha)
  # The critical value changes depending on whether rho is positive
  c_a <- k[1] + sqrt(k[2])*(t_a + ki[3]*h[[2]](t_a)/factorial(3) +
          ki[4]*h[[3]](t_a)/factorial(4) -
          ki[3]*2*(2*t_a-3.5*t_a^3)/6 )
  # obtain the critical level by solving numerically
  if (rho > 0) {
    guess1<-qnorm(1-Machine$double.eps^0.5, mean=k[1], sd=sqrt(k[2]))
    guess2<-qnorm(1-Machine$double.eps^0.5, mean=k[1], sd=sqrt(k[2]))
    c_a_n <- uniroot(function(c) E4((c-k[1])/sqrt(k[2]),k1)+alpha-1,
                    upper=guess2, tol=Machine$double.eps^0.5)$root
    else (guess1<-qnorm(1-Machine$double.eps^0.5, mean=k[1], sd=sqrt(k[2]))
      guess2<-qnorm(1-Machine$double.eps^0.5, mean=k[1], sd=sqrt(k[2]))
      c_a_n <- uniroot(function(c) E4((c-k[1])/sqrt(k[2]),k1)+alpha,
                        lower=guess1, upper=guess2, tol=Machine$double.eps^0.5)$root)
  }

  if (verbose) {
    cat("For rho: ",rho," the critical levels are ",c_a , " and ", c_a_n,
    # " and " , t_a*sqrt(k[2]+k[1]),"\n"
    cat("For rho: ",rho," the critical levels are ",(c_a-k[1])/sqrt(k[2]),
    # " and ", (c_a_n-k[1])/sqrt(k[2]), " and ",t_a,"\n")
  }
}
```r
# calculate power (USING NUMERIC RATHER THEN CORNISH_FISHER
if (rho > 0)  power[ind] <- 1 - E4((c_a - n - ka[1])/sqrt(ka[2]),kai) 
else power[ind] <- E4((c_a - n - ka[1])/sqrt(ka[2]),kai)
)
}
return(power)

Listing 125: powerE4 V.R

'powerE4_V' <-
function(rho.range, alpha=0.05, m1, m2=NULL, verbose=FALSE) {
  # repeat the procedure outlined in the paper for all rho in rho.range
  power<-NA # re-sets the output variable
  for (ind in 1:length(rho.range)) {
    rho <- rho.range[ind]
    c_a <- EvCV_V(alpha, rho, m1, m2)
    if (verbose) cat("For rho: ",rho," the critical levels are ",c_a$norm," and ", c_a$std, "\n"")
    # Calculate the cumulants under the alternative
    k<-cumulantsV(4,c_a$norm,rho,rho,m1,m2)
    ki<-invariantes(k)
    # calculate power (USING NUMERIC RATHER THEN CORNISH_FISHER
    if (rho > 0)  power[ind] <- 1 - E4(-k[1]/sqrt(k[2]),ki)
    else power[ind] <- E4(-k[1]/sqrt(k[2]),ki)
  }
  return(power)
}

Listing 126: powerLR V.R

'powerLR_V' <-
function(rho.range, alpha=0.05, m1, m2=NULL, verbose=FALSE) {
  power<-NA
  for (j in 1:length(rho.range)) {
    rho1 <- rho.range[j]
    c_a <- LRACV_V(alpha, rho1, m1, m2)
    if (rho1 > 0)  power[j] <- LRAPV(c_a$norm, rho1, rho1, m1, m2)
    else power[j] <- 1-LRAPV(c_a$norm, rho1, rho1, m1, m2)
    if (verbose) cat("For rho: ",rho1," the critical level is ",c_a$std," and power =",
    power[j], "\n")
  }
  return(power)
}

Listing 127: powerM.R

'powerM' <-
function(M,n){
  if (n<0)
    return(-1)
  else if (n==0)
    return(diag(nrow(M)))
  else
    return(M%*%powerM(M,n-1))
}

Listing 128: pt.r

# Calculates TAIL probability P(T>t) based on the integral formula
'PT' <-
function(t, rho1, rho, m1, m2=NULL, normalize=TRUE) {
  mat <- Cr(rho1, rho, m1, m2) # define C
  gamma.range <- 1/(2*gammaRange(mat))
  gamma.range <- gamma.range + c(1, -1) * Machine$double.eps/0.5
  # define o integrando como funcao "vetorizavel"
  tempfun <- function(gammavec) {
    tempfun2 <- function(gamma) {
      mat2 <- diag(ncol(mat)) - 2*gamma*mat
```
Listing 129: PV.r

```r
# Calculates TAIL probability P(Vn>v) based on the integral formula
PV <- function(v, rho1, rho, m1, m2 = NULL, normalize = TRUE) {
  mat <- Dv(v, rho1, rho, m1, m2) # define D
  gammarange <- 1/(2 + eigenRange(mat))
  if (gammarange[1] > 0) return(1)
  if (gammarange[2] < 0) return(0)
  return(integrate(tempfun, gammarange[1], gammarange[2], rel.tol = Machine$double$eps$0.5)$value)
}
```

Listing 130: RACV.T.R

```r
# Calculates the critical value based on the solution to
# the tail probability integral.
'RACV.T' <- function(alpha, rho1, m1, m2 = NULL) {
  crit <- cumulantsT(2, rho1, 0, m1, m2) # Under the NULL
  guess1 <- qnorm(0.55, mean = k[1], sd = sqrt(k[2]))
  guess2 <- qnorm(0.999, mean = k[1], sd = sqrt(k[2]))
  if (rho1 > 0)
    ca <- unifroot(function(t) RAPT(t, rho1, 0, m1, m2) - alpha, lower = guess1, upper = guess2, tol = Machine$double$eps$0.5)$root
  else stop("Robinson's approximation is not valid for negative rho1")
  return(list(norm = ca, std = (ca - k[1])/sqrt(k[2])))
}
```

Listing 131: racv.v.r

```r
# Calculates the critical value based on the solution to
# the tail probability integral.
'RACV.V' <- function(alpha, rho1, m1, m2 = NULL, normalize = TRUE) {
  k1 <- tr(A_r(rho1, m1, m2))/sqrt(ncol(m1)) # mean of Vn (approx) Laplace
  k2 <- trPower2(A_r(rho1, m1, m2, 2))/ncol(m1) # var of Vn (approx) Laplace
  guess1 <- qnorm(0.55, mean = k1, sd = sqrt(k2))
  guess2 <- qnorm(0.999, mean = k1, sd = sqrt(k2))
  if (rho1 > 0)
    ca <- unifroot(function(t) RAPl(t, rho1, 0, m1, m2) - alpha, lower = guess1, upper = guess2, tol = Machine$double$eps$0.5)$root
  else stop("Robinson's approximation is not valid for negative rho1")
  return(list(norm = ca, std = (ca - k[1])/sqrt(k[2])))
}
```
Listing 132: RAPT.R

```r
'RAPT' <-
  function(t,rho1,rho,m1,m2=NULL) {
    mat<-Cr(rho1, rho, m1, m2)
    gammahat<-saddlepoint_T(t, rho1, rho, m1, m2)
    mat2<-diag(ncol(mat))-2*gammahat*mat
    mat2inv<-solve(mat2)
    a<-sign(gammahat)*sqrt(2*(gammahat*tr(mat2inv%*%mat)+0.5*log(det(mat2))))
    b<-gammahat*sqrt(2*trPowerM(mat2inv%*%mat,2))
    k3<-8*trPowerM(mat2inv%*%mat,3)/(2*trPowerM(mat2inv%*%mat,2))^(3/2)
    res<- exp((b^2-a^2)/2)*((-1-pnorm(b))*(1-k3*b^3/6)+dnorm(b)*(b^2-1)*k3/6)
    return(res)
  }

Listing 133: rapv.r

```r

```r
'RAPV' <-
  function(v,rho1,rho,m1,m2=NULL) {
    mat<-Dv(v, rho1, rho, m1, m2)
    gammahat<-saddlepoint_V(v, rho1, rho, m1, m2)
    mat2<-diag(ncol(mat))-2*gammahat*mat
    mat2inv<-solve(mat2)
    a<-sign(gammahat)*sqrt(2*log(det(mat2)))
    b<-gammahat*sqrt(2*trPowerM(mat2inv%*%mat,2))
    k3<-8*trPowerM(mat2inv%*%mat,3)/(2*trPowerM(mat2inv%*%mat,2))^(3/2)
    res<- exp((b^2-a^2)/2)*((-1-pnorm(b))*(1-k3*b^3/6)+dnorm(b)*(b^2-1)*k3/6)
    return(res)
  }

Listing 134: rholimiteneg.R

```r

```r
'rholimiteneg' <-
  function(limite=5, md=Mdefault) {
    tempmin <- 1/eigenRange(md)[1].Machine$double.eps^0.5
    tempfun <- function(rho, md) maxVar(rho, md, sigma2=1)-limite=1
    return(unroot(tempfun, c(tempmin,0), md, tol=.Machine$double.eps^0.5)$root)
  }

Listing 135: rholimit.R

```r

```r
'rholimit' <-
  function(limite=5, md=Mdefault) {
    tempmax<-1/eigenRange(md)[2].Machine$double.eps^0.5
    # independe de sigma2, portante posso normalizar sigma2=1
    tempfun <- function(rho, md) maxVar(rho, md, sigma2=1)-limite=1
    return(unroot(tempfun, c(0, tempmax), md, tol=.Machine$double.eps^0.5)$root)
  }

Listing 136: saddlepointCV_T.R

```r

```r
# Calculates the saddlepoint critical value based on Arevalillo
'saddlepointCV_T' <-
  function(alpha, rho0, m1,m2=NULL) {
    k<-cumulantsT(2, rho0, 0, m1, m2) # under the NULL
    if (rho0>0) z_a <- qnorm(1-alpha) else z_a <- qnorm(alpha)
    zabar <- z_a/sqrt(k[2])
    mat<-Cr(rho0, 0, m1, m2) # under the NULL
    mat2<-diag(ncol(mat))-2*zabar*mat
    mat2inv<-solve(mat2)
    k1_zabar <- tr(mat2inv%*%mat) # first derivative of cumulant gen function eval at zabar
    k2_zabar <- 2*trPowerM(mat2inv%*%mat,2)
    tbar <- (zabar * (k[2] + k2_zabar) - k1_zabar + k[1])/k2_zabar
```
mat2 <- diag(ncol(mat)) - 2*tbar*mat  # note the new definition of mat2
c2inv <- solve(mat2)
cumRange <- 1/(2* eigenRange(mat))
if ( (tbar <= cumRange[1]) || (tbar > cumRange[2]) )
    k0_tbar <- Inf
cat("WARNING!! - cumulant domain bounds reached! 
")
return(list(normNA, stdNA))
else
    k0_tbar <- -0.5*log(det(mat2))
k1_tbar <- tr(mat2inv%*%mat)
k2_tbar <- 2*trPowerM(mat2inv%*%mat, 2)
a_tbar <- sign(tbar) * sqrt(2*((tbar*k1_tbar - k0_tbar))
b_tbar <- tbar * sqrt(k2_tbar)
r_tbar <- a_tbar + log(b_tbar/a_tbar)/a_tbar
s_bar <- k1_tbar + a_tbar * (s_a_r_tbar)/tbar
return(list(norm=s_bar, std=(s_bar-k1)/sqrt(k2))))

# calculates the critical value based on the solution to
# the tail probability integral.
'saddlepointCV_T' <-
function(alpha, rho1, m1, m2=NULL, normalize=TRUE){
  k <- cumulantsT(2, rho1, 0, m1, m2)  # Under the NULL
  if (rho1>0) {
    guess1 <- qnorm(0.55, mean=k[1], sd=sqrt(k[2]))
    guess2 <- qnorm(0.999, mean=k[1], sd=sqrt(k[2]))
    ca <- uniroot(function(t) PT(t, rho1, 0, m1, m2, normalize) - alpha,
               lower=guess1, upper=guess2, tol=.Machine$double.eps^0.5)$root
    else {
      guess1 <- qnorm(0.001, mean=k[1], sd=sqrt(k[2]))
      guess2 <- qnorm(0.45, mean=k[1], sd=sqrt(k[2]))
      ca <- uniroot(function(t) PT(t, rho1, 0, m1, m2, normalize) - alpha,
               lower=guess1, upper=guess2, tol=.Machine$double.eps^0.5)$root
  return(list(norm=ca, std=(ca-k[1])/sqrt(k[2])))

'saddlepointCV_V' <-
function(alpha, rho1, m1, m2=NULL, normalize=TRUE){
  kv1 <- tr(A_r(rho1, m1, m2))/sqrt(ncol(m1))  # mean of Vn (approx) Laplace
  kv2 <- 2*trPowerM(A_r(rho1, m1, m2), 2)/ncol(m1)  # var of Vn (approx) Laplace
  if (rho1>0) {
    guess1 <- qnorm(0.55, mean=kv1, sd=sqrt(kv2))
    guess2 <- qnorm(0.999, mean=kv1, sd=sqrt(kv2))
    ca <- uniroot(function(v) PV(v, rho1, 0, m1, m2, normalize) - alpha,
               lower=guess1, upper=guess2, tol=.Machine$double.eps^0.5)$root
    else {
      guess1 <- qnorm(0.001, mean=kv1, sd=sqrt(kv2))
      guess2 <- qnorm(0.45, mean=kv1, sd=sqrt(kv2))
      ca <- uniroot(function(v) PV(v, rho1, 0, m1, m2, normalize) - alpha,
               lower=guess1, upper=guess2, tol=.Machine$double.eps^0.5)$root
  return(list(norm=ca, std=(ca-kv1)/sqrt(kv2)))

# calculates the solution to the saddlepoint equation
'saddlepoint_T' <-
function(t, rho1, rho, m1, m2)
mat <- Cr(rho1, rho, m1, m2)
gammarange <- 1/(2* eigenRange(mat))
gammarange <- gammarange * c(1,-1) * .Machine$double.eps^0.5
tempfun <- function(gamma, t) tr(solve(diag(ncol(mat)) - 2*gamma*mat)%*%mat) - t
return(uniroot(tempfun, gammarange, t, tol=.Machine$double.eps^0.5)$root)
Listing 140: saddlepoint V.R

```r
'saddlepoint.V' <-
function(v,rholi,rho,m1,m2=NULL) {
  mat <- Dv(v,rholi,rho,m1,m2)
  gammarange <- sort(1/(2*eigen(lambda(mat))))
  if (gammarange[1] > 1) return(-Inf)
  if (gammarange[2] <= 0) return(+Inf)
  gammarange <- gammarange*c(1,-1)*Machine$double.eps^-0.5
  tempfun <- function(gamma) tr(solve(diag(ncol(mat)) - 2*gamma*mat,rel=Machine$double.eps^-0.5)*sqrt)
  return(uniroot(tempfun,gammarange,tol=Machine$double.eps^-0.5)$root)
}
```

Listing 141: SignedFisherInfoRho.R

```r
'SignedFisherInfoRhos' <-
function(rho, md=Mdefault) {
  temp1 <- Linv(rho, md)*%W(rho, md) # note que eh o caso linear, onde mderiv=md
  temp2 <- W(rho, md)*%W(rho, md)
  return(ifelse(rho > 0, 1,-1)*tr(powerM(temp1,2)+crossprod(temp2)))
}
```

Listing 142: SymmetricCV V.R

```r
# Valid only for LUMPI tests, i.e., rholi=0
'SymmetricCV.V' <-
function(alpha,m1,m2=NULL) {
  n <- ncol(m1)
  mat <- W(0,m1,m2)
  tv2 <- trPowerM(mat,2)
  tv3 <- trPowerM(mat,3)
  tv4 <- trPowerM(mat,4)
  ta <- qnorm(1-alpha)
  a3 <- (-1/n)*((n/(8+tv2))-3/2)
  a2 <- -0
  a1 <- sqrt(n/(8+tv2)-1/2+ta-2-1)/n
  a0 <- -(ta + tv3+sqrt(2)*(ta-2-1)/(3+tv2*(3/2))) + tv2-2*(ta-3+ta)/(2+tv2)
  ca <- polyroot(c(a0,a1,a2,a3))
  if (!all(abs(In(ca))<.Machine$double.eps^0.5)) stop("Imaginary roots for the cubic polynomial!"))
  ca <- Re(ca)
  ca <- ca[ca>0] # critical value is the smallest positive root
  kv2 <- 8*tv2/n # var of Vn (approx) Laplace
  return(list(norm=ca, std=ca/sqrt(kv2)))
}
```

Listing 143: Tn.R

```r
'Tn' <-
function(y,rho,sigma2=1,m1,m2=NULL) return(t(y)%*%A_r(rho,m1,m2)%*%Y/sigma2)
```

Listing 144: trpowermatrix.R

```r
'trPowerM' <- function(mat, n) tr(matrix.power(mat,n))
```

Listing 145: tr.R

```r
'tr' <-
function(m) if (is.matrix(m)) sum(diag(m)) else stop("Non matrix argument to trace")
```

Listing 146: VarHgeneralSAR.R

```r
'VarHgeneralSAR' <- function(theta_hat,x,y,m1,m2) {
  BuildHessian <- function(theta,x,y,m1,m2) {
    n <- length(y)
```
p <- length(theta)
beta <- theta[1:(p-1)]
rho <- theta[p-2]
lambda <- theta[p-1]
sigma2 <- theta[p]
k <- L(lambda, m2)
l <- L(rho, m1)
kinv <- solve(k)
linv <- solve(l)
xx <- k*y**2
kwy <- k*x*y
xb_ly <- x*y/beta - 1/y
kxbb <- k%*%xb*y
kly <- x*y/beta
z <- crossprod(kly) - 2*crossprod(kly, kxb) + crossprod(kxb)
mtk_ktm <- t(m2)*k%*%t(k)*m2
bb <- crossprod(kx)/sigma2
br <- crossprod(kx, kwy)/sigma2
rr <- crossprod(kwy)/sigma2 - tr(powerM(linv%*%m1, 2))
rl <- (t(m1%*%ly)*mtk_ktm%*%x(b Ly))/sigma2
rs <- crossprod(kwy, k%*%(xb_ly))/sigma2 - 2
ll <- crossprod(m2%*%xb_ly)/sigma2 - tr(powerM(kinv%*%m2, 2))
l1 <- crossprod(m2%*%xb_ly, k%*%xb_ly)/sigma2 - 2
ls <- n/(2*sigma2 - 2) - z/sigma2 - 3
bl <- (t(x)%*%mtk_ktm%*%xb_ly)/sigma2
bs <- crossprod(kx, k%*%xb_ly)/sigma2 - 2
Hm <- cbind(bb, br, bl, bs)
Hm <- rbind(Hm, c(t(br), rr, rl, rs))
Hm <- rbind(Hm, c(t(bl), rl, 1, 1, ls))
Hm <- rbind(Hm, c(t(bs), rs, ls, ss))
colnames(Hm) <- as.list(colnames(x))
return(Hm)
}
return(-solve(BuildHessian(theta_hat, x, y, m1, m2)))

'VarHSAR' <- function(theta_hat, x, y, m) {
BuildHessian <- function(theta, x, y, m) {
  n <- length(y)
p <- length(theta)
beta <- theta[1:(p-1)]
rho <- theta[p-2]
sigma2 <- theta[p]
linv <- solve(l)
wy <- n*y
xb <- x*y/beta
ly <- 1/y
e <- ly - xb
bb <- crossprod(x)/sigma2
br <- crossprod(x, wy)/sigma2
rr <- crossprod(wy)/sigma2 - tr(powerM(linv%*%m, 2))
rs <- crossprod(wy, e)/sigma2 - 2
ss <- n/(2*sigma2 - 2) - 2*crossprod(e)/sigma2 - 3
bs <- crossprod(x, e)/sigma2 - 2
Hm <- cbind(bb, br, bs)
Hm <- rbind(Hm, c(t(br), rr, rs))
Hm <- rbind(Hm, c(t(bs), rs, ss))
colnames(Hm) <- as.list(colnames(x))
return(Hm)
}
return(-solve(BuildHessian(theta_hat, x, y, m)))

Listing 147: VarHSAR.R

Listing 148: VarIMgeneralSAR.R
'VarIMgeneralSAR' <- function(theta_hat, x, m1, m2=m1) {
  BuildInfoMatrix <- function(theta, x, m1, m2) {
    p <- length(theta)
    n <- nrow(x)
    beta <- theta[1:(p-3)]
    rho <- theta[p-2]
    lambda <- theta[p-1]
    sigma2 <- theta[p]
    k <- L(lambda, m2)
    l <- L(rho, m1)
    linv <- solve(l)
    kinv <- solve(k)
    kx <- k %*% %*
    kulinv <- k %*% m1 %*% linv
    kulinvxv <- kulinv %*% x %*% %*% beta
    kulinvkinv <- kulinv %*% kinv
    mkinv <- m2 %*% kinv
    bb <- -crossprod(kx)/sigma2
    br <- -crossprod(kx, kulinvxv)/sigma2
    rr <- -crossprod(kulinvxv)/sigma2 - tr(crossprod(kulinvkinv)) -
      tr(powerM(linv, 1, 2))
    ri <- -tr((m2 %*% linv %*% kinv)) - tr(crossprod(k %*% linv * %*% kinv, mkinv))
    res <- -tr(linv %*% kinv)/sigma2
    li <- -tr(crossprod(kinv)) - tr(powerM(mkinv, 2))
    lb <- -tr(mkinv)/sigma2
    ss <- -n/sigma2^2
    Hm <- cbind(bb, br, 0, 0)
    Hm <- rbind(Hm, c(t(br), rr, rl, rs))
    Hm <- rbind(Hm, c(rep(0, ncol(x)), rl, ll, ls))
    Hm <- rbind(Hm, c(rep(0, ncol(x)), rs, ls, ss))
    colnames(Hm) <- cnames(Hm) <- c(colnames(x), "rho", "lambda", "sigma2")
    return(Hm)
  }
  return(-solve(BuildInfoMatrix(theta_hat, x, m1, m2)))
}

Listing 149: VarIMSAR.R

'VarIMSAR' <- function(theta_hat, x, m) {
  BuildInfoMatrix <- function(theta, x, m) {
    p <- length(theta)
    n <- nrow(x)
    beta <- theta[1:(p-2)]
    rho <- theta[p-1]
    sigma2 <- theta[p]
    l <- L(rho, m)
    linv <- m %*% linv
    wlinv <- m %*% linv
    wlinvxv <- wlinv %*% x %*% %*% beta
    kwlinvkinv <- kwlinv %*% kinv
    #mkinv <- m2 %*% kinv
    bb <- -crossprod(x)/sigma2
    br <- -crossprod(x, wlinvxv)/sigma2
    rr <- -crossprod(wlinvxv)/sigma2 - tr(crossprod(wlinv)) -
      tr(powerM(wlinv, 2))
    rs <- -tr(wlinv)/sigma2
    ss <- -n/(2*sigma2^2)
    Hm <- cbind(bb, br, 0)
    Hm <- rbind(Hm, c(t(br), rr, rs))
    Hm <- rbind(Hm, c(rep(0, ncol(x)), rs, ss))
    colnames(Hm) <- cnames(Hm) <- c(colnames(x), "rho", "sigma2")
    return(Hm)
  }
  return(-solve(BuildInfoMatrix(theta_hat, x, m)))
Listing 150: vcv.r

```r
vcv <- function(rho, md = Mdefault, sigma2 = 1) sigma2 * Linv(rho, md) %*% t(Linv(rho, md))
```

Listing 151: Vn.R

```r
vn <- function(Y, rho1, m1 = NULL, m2 = NULL) return(sqrt(length(Y)) * t(Y) %*% A_r(rho1, m1, m2) %*% crossprod(Y))
```

Listing 152: w.r

```r
wr <- function(rho, M1, M2 = NULL) if (is.null(M2)) return(M1) else return(M1 + 2 * rho * M2)
```

Listing 153: w.R

```r
wr <- function(rho, M1, M2 = NULL) if (is.null(M2)) return(rho * M1) else return(rho * M1 + rho^2 * M2)
```
A.4.2 Codes used in Chapter 4

This section includes R codes that were used to generate the statistics, tables, figures, and simulations used in chapter 4.

Listing 154: garma tables.R

```r
# loads data
load('/media/EPGE/Docs/dados/artigo3/modelY.Rdata')

# prepare table for YsMd
modelY <- modelYMd
N <- length(modelY[[1]]$eback)

# prepare table for YsMd
modelY <- modelYMd
N <- length(modelY[[1]]$eback)

# prepare table for YsMo
modelY <- modelYMo
N <- length(modelY[[1]]$eback)

# prepare table for YsMw
modelY <- modelYMw
N <- length(modelY[[1]]$eback)

# prepare table for YmMd
modelY <- modelYMd
N <- length(modelY[[1]]$eback)

# prepare table for YmMo
modelY <- modelYM0

# prepare table for YmMw
modelY <- modelYMw
N <- length(modelY[[1]]$eback)
```
W <- length(modelY[[1]]$what)
tabYMo <- cbind(c(modelY[[1]]$coefficients, modelY[[1]]$ss$ssr/N, modelY[[1]]$convInfo$code ),
               c(modelY[[1]]$tstat, NA, NA))
rownames(tabYMo) <- c('lambda', 'Intercept', 'Yo', 'Yd', 'Dist', 'Rta', 'Col', 'alpha', 'rho', 'SSR/n', 'Conv?')
# note that Isky model '2' since it is the same as '1'
for (i in 1:3) tabYMo <- cbind(tabYMo, cbind(c(modelY[[1]]$coefficients, modelY[[1]]$ss$ssr/N, modelY[[1]]$convInfo$code ),
                                       c(modelY[[1]]$tstat, NA, NA)))
colnames(tabYMo) <- paste(rep(paste('M', i, 1:11, sep = ''), each = 2), c('(Est.)', '(tval)'), sep = ' ')

# prepare table for Ym_Mu
mod1Y <- modelYmuW
n <- length(modelYmuW[[1]]$what)
tabYMuW <- cbind(c(modelYmuW[[1]]$coefficients, modelYmuW[[1]]$ss$ssr/N, modelYmuW[[1]]$convInfo$code ),
                c(modelYmuW[[1]]$tstat, NA, NA))
rownames(tabYMuW) <- c('lambda', 'Intercept', 'Yo', 'Yd', 'Dist', 'Rta', 'Col', 'alpha', 'rho', 'SSR/n', 'Conv?')
# note that Isky model '2' since it is the same as '1'
for (i in 1:3) tabYMuW <- cbind(tabYMuW, cbind(c(modelYmuW[[1]]$coefficients, modelYmuW[[1]]$ss$ssr/N, modelYmuW[[1]]$convInfo$code ),
                                          c(modelYmuW[[1]]$tstat, NA, NA)))
colnames(tabYMuW) <- paste(rep(paste('M', i, 1:11, sep = ''), each = 2), c('(Est.)', '(tval)'), sep = ' ')

# save tabYMo, tabYMuW, tabYMu, tabYMo, tabYMW, file = '/media/EPGE/Docs/dados/artigo3_tables_modelY.Rdata'

# # print tables
library(xtable)
# prepares LaTeX tables
mtab <- xtable(tabYMo[, 1:12], digits = 2, caption = 'Model for exports using destination neighborhood', label = 'modYMo')
print(mtab, floating.environment = 'sidewaystable', include.colnames = FALSE)
mtab <- xtable(tabYMo[, 13:22], digits = 2, caption = 'Model for exports using destination neighborhood (cont.)', label = 'modYMocont')
print(mtab, floating.environment = 'sidewaystable', include.colnames = FALSE)
mtab <- xtable(tabYMu[, 1:12], digits = 2, caption = 'Model for exports using origin neighborhood', label = 'modYMu')
print(mtab, floating.environment = 'sidewaystable', include.colnames = FALSE)
mtab <- xtable(tabYMu[, 13:22], digits = 2, caption = 'Model for exports using origin neighborhood (cont.)', label = 'modYMucnt')
print(mtab, floating.environment = 'sidewaystable', include.colnames = FALSE)
mtab <- xtable(tabYMuW[, 1:12], digits = 2, caption = 'Model for exports using origin-destination neighborhood', label = 'modYMuW')
print(mtab, floating.environment = 'sidewaystable', include.colnames = FALSE)
mtab <- xtable(tabYMuW[, 13:22], digits = 2, caption = 'Model for exports using origin-destination neighborhood (cont.)', label = 'modYMucntw')
print(mtab, floating.environment = 'sidewaystable', include.colnames = FALSE)
Listing 155: garm tables RN.port.R

```r
mtab <- xtable(tabYmW[, , 13:22], digits = 2, caption = 'Model for imports using origin-destination neighborhood (cont.)', label = 'modYmWcont') print(mtab, floating_environment = 'sidewaystable', include = .colnames = FALSE)

# check which coefficients reached upper bound 
ubound <- 1 -.Machine$double.eps^0.3

for (i in 1:12) cat(quote('Model (', i, ') has:'), format(modelYXMo[[i]]$coefficients[9] - ubound, digits = 1), quote(',
')

# modelYxM is OK
# modelYxM [[6]] and modelYxM [[12]] reached upper bound
# modelYxM [[3, 4, 5, 6, 7, 8, 9]] reached upper bound
# modelYxM [[12]] reached upper bound
# modelYmMo is OK (but model 7 returned an error) -- OK checked. The error did not change
# coef and t-stat
```

```r
xs(list = ls())
# loads data
load('./media/EPGE/Docs/dados/artigo3/modelY_RN_nlmind.Rdata')
## define Functions
prepare_table <- function(modelY) {
  ubound <- 1 -.Machine$double.eps^0.3
  X <- length(modelY[[1]])$xhat
  tabY <- cbind(c(modelY[[1]]$coefficients, modelY[[1]]$ss$ssr/N, modelY[[1]]$qhat*100, modelY[[1]]$convInfo$code),
               c(modelY[[1]]$tstat, NA, NA, NA),
  colnames(tabY) <- paste(paste('M', i, 11, sep = ' '), each = 3), c('Est.', 'tval'), (bound? ')
  return(tabY)
}
}
## Main program

# modelYxM <- prepare_table(modelYxM)
tabYxM <- prepare_table(modelYxM)
tabYmM <- prepare_table(modelYmM)
tabYmMo <- prepare_table(modelYmMo)

# save(tabYxM, tabYxM, tabYmM, tabYmMo, file = './media/EPGE/Docs/dados/artigo3/tabs_modelY_RN_port.Rdata')
```

```r
```
# this program runs Moran I test - monte carlo
1 rm(list=ls())
2 # Loads DATA
3 # loads independent variables
4 load('dados/gravity_XM.garma.Rdata') # returns X, M
5 #renames imports and exports
6 Yx <- X
7 Ym <- M
8 rm(X)
9 rm(M)
10 # loads covariates (see prepare_data.garma.R)
11 load('dados/gravity_Z.covariates_garma.Rdata') # returns Z
12 #loads neighbor matrices
13 load('dados/world_neighbors_matrix.Rdata') # returns M
14 # renames the neighbor matrices list
15 Mlist <- M
16 rm(M)
17 # end DATA
18
20 # Starts MAIN program
21 selected <- (Yx!=0)
22 F <- Yx[selected]
23 X <- [selected,,c('Yx','Yc','Yd','YdC','Dist','Cont','Lang','Rta','Cn','Cc','Col')]
24 reg1 <- lm(F~X)
25 F <- Ym[selected]
26 reg2 <- lm(F~X)
27 nblist <- c(1,3,4,6,8,10,12,14,15,16,17) # These are the 11 designs in the paper (BINARY)
28 nblist_rn <- nblist + 18 # These are de 11 designs in the paper (ROW NORMALIZED)
30 # using BINARY matrices
31 x<-reg1$res
32 for (i in 1:length(nblist)) {
33 Md <- kronecker(diag(nrow(Mlist[[nblist[i]]])),Mlist[[nblist[i]]])
34 Mo <- kronecker(Mlist[[nblist[i]]],diag(nrow(Mlist[[nblist[i]]])))
35 wzd <- Md[selected,selected]*x
36 wxo <- Mo[selected,selected]*x
37 xrange <- range(x)
38 yrange <- range(c(wzd,wxo))
39 zerowzd <- wzd==0
40 zerowxo <- wxo==0
41 require(MASS)
42 wzd.lm <- lm(wzd[!zerowzd] ~ x[!zerowzd],method='MM')
43 wxd.lm <- lm(wxo[!zerowxo] ~ x[!zerowxo])
44 wzd.lm <- lm(wzd[!zerowzd] ~ x[!zerowzd])
45 wzd.lm <- lm(wzd[!zerowzd] ~ x[!zerowzd])
47 ps.options(horizontal=FALSE, onefile=FALSE, width=5.8, height=5.8,paper="special",
pointsize=10)
49 filename<paste("/media/EPGE/Docs/tese3_tes/figures/art3_chart03_YxMd_M'/i,'.eps',sep="'"),
50 postscript(filename)
```r
plot(x[!zerowxd], wxd[!zerowxd], xlab = 'Residuals of gravity equation',
ylab = 'Spatially lagged residuals', xlim=xrange, ylim=yrange)
abline(wxd.lm)
abline(wxd.lm, lty=2)
abline(h = mean(wxd[!zerowxd]), lty = 3)
abline(v = mean(x[!zerowxd]), lty = 3)
legend('topleft', legend=c('regression line', 'robust regression line'), lty=1:2, bty = 'n')
dev.off()

filename<-paste('/media/EPGE/Docs/tese3_tex/figures/art3_chart03_YxMo_M',i, '.eps', sep = '' )
postscript(filename)
plot(x[!zerowxo], wxo[!zerowxo], xlab = 'Residuals of gravity equation',
 ylab = 'Spatially lagged residuals', xlim=xrange, ylim=yrange)
abline(wxo.lm)
abline(wxo.lm, lty=2)
abline(h = mean(wxo[!zerowxo]), lty = 3)
abline(v = mean(x[!zerowxo]), lty = 3)
legend('topleft', legend=c('regression line', 'robust regression line'), lty=1:2, bty = 'n')
dev.off()

load('nbinlist_rn')
for (i in 1:length(nbinlist_rn)) {
Md <- kronecker(diag(nrow(Mlist[[nbinlist_rn[i]]])), Mlist[[nbinlist_rn[i]]])
Mo <- kronecker(Mlist[[nbinlist_rn[i]]], diag(nrow(Mlist[[nbinlist_rn[i]]])))
wxd <- Md[selected,selected]*%x%w xo <- Mo[selected,selected]*%x%
xrange <- range(x)
yrange <- range(c(xwd, wxo))
zerowxd <- wwd=0 zerowxo <- wxo=0
require(MASS)
wxd.lm <- lm(xwd[!zerowxd] ~ x[zerowxd], method='MM')
wxd.lm1 <- lm(wxd[!zerowxd] ~ x[zerowxd])
wxo.lm <- lm(xwo[!zerowxo] ~ x[zerowxo], method='MM')
wxo.lm1 <- lm(xwo[!zerowxo] ~ x[zerowxo])
ps.options(horizontal=FALSE, onefile=FALSE, width=5.8, height=5.8, paper = 'special', 
pointsize=10)
filename<-paste('/media/EPGE/Docs/tese3_tex/figures/art3_chart03_YxMo_RN_M',i, '.eps', sep = '' )
postscript(filename)
plot(x[!zerowxd], wxd[!zerowxd], xlab = 'Residuals of gravity equation',
 ylab = 'Spatially lagged residuals', xlim=xrange, ylim=yrange)
abline(wxd.lm)
abline(wxd.lm, lty=2)
abline(h = mean(wxd[!zerowxd]), lty = 3)
abline(v = mean(x[!zerowxd]), lty = 3)
legend('topleft', legend=c('regression line', 'robust regression line'), lty=1:2, bty = 'n')
dev.off()

filename<-paste('/media/EPGE/Docs/tese3_tex/figures/art3_chart03_YxMo_RN_M',i, '.eps', sep = '' )
postscript(filename)
plot(x[!zerowxo], wxo[!zerowxo], xlab = 'Residuals of gravity equation',
 ylab = 'Spatially lagged residuals', xlim=xrange, ylim=yrange)
abline(wxo.lm)
abline(wxo.lm, lty=2)
abline(h = mean(wxo[!zerowxo]), lty = 3)
abline(v = mean(x[!zerowxo]), lty = 3)
legend('topleft', legend=c('regression line', 'robust regression line'), lty=1:2, bty = 'n')
dev.off()
}
## repeat the chart producing routine for imports
# using BINARY matrices
xx<-reg2res
for (i in 1:length(nbinlist)) {
Md <- kronecker(diag(nrow(Mlist[[nbinlist[i]]])), Mlist[[nbinlist[i]]])
Mo <- kronecker(Mlist[[nbinlist[i]]], diag(nrow(Mlist[[nbinlist[i]]])))
wxd <- Md[selected,selected]*%x%w
```
wxo <- M0[select, select]X*Xx
xrange <- range(x)
yscale <- range(c(xd, wdo))
zeronxd <- xrd=0
zerowxo <- wdo=0
require(MASS)
wx.lm <- lm(xd[!zerowxd] ~ x[!zerowxd], method = 'MM')
x.lm <- lm(xd[!zerowxd] ~ x[!zerowxd])
w.xl <- lm(xdo[!zerowxo] ~ x[!zerowxo], method = 'MM')
xo.lm <- lm(xdo[!zerowxo] ~ x[!zerowxo])

ps.options(horizontal = FALSE, onefile = FALSE, width = 5.8, height = 5.8, paper = "special",
            pointsize = 10)

filename <- paste(’/media/EPGE/Docs/tese3_tex/figures/art3_plot03_YnMd_M’, i, ’.eps’, sep = ”)

postscript(filename)

plot(x[!zerowxd], wdo[!zerowxd], xlab = ’Residuals of gravity equation’,
ylab = ’Spatially lagged residuals’, xlab = xrange, ylab = yrange)

plot(x[!zerowxd], wdo[!zerowxd], xlab = ’Residuals of gravity equation’,
ylab = ’Spatially lagged residuals’, xlab = xrange, ylab = yrange)

plot(x[!zerowxd], wdo[!zerowxd], xlab = ’Residuals of gravity equation’,
ylab = ’Spatially lagged residuals’, xlab = xrange, ylab = yrange)

plot(x[!zerowxd], wdo[!zerowxd], xlab = ’Residuals of gravity equation’,
ylab = ’Spatially lagged residuals’, xlab = xrange, ylab = yrange)

plot(x[!zerowxd], wdo[!zerowxd], xlab = ’Residuals of gravity equation’,
ylab = ’Spatially lagged residuals’, xlab = xrange, ylab = yrange)

plot(x[!zerowxd], wdo[!zerowxd], xlab = ’Residuals of gravity equation’,
ylab = ’Spatially lagged residuals’, xlab = xrange, ylab = yrange)

for (i in 1:length(nblist.rn)) {
    M <- kronecker(diag(nrow(Mlist[[nblist.rn[i]]])), Mlist[[nblist.rn[i]]])
    M0 <- kronecker(Mlist[[nblist.rn[i]]], diag(nrow(Mlist[[nblist.rn[i]]])))
    wdo <- M0[select, select]X*Xx
    wxo <- M0[select, select]X*Xx
    xrange <- range(x)
    yrange <- range(c(wdo, wxo))
    zerowxd <- xrd=0
    zerowxo <- wdo=0
    require(MASS)
    wx.lm <- lm(xd[!zerowxd] ~ x[!zerowxd], method = 'MM')
    x.lm <- lm(xd[!zerowxd] ~ x[!zerowxd])
    wx.lm <- lm(xdo[!zerowxo] ~ x[!zerowxo], method = 'MM')
    xo.lm <- lm(xdo[!zerowxo] ~ x[!zerowxo])

    ps.options(horizontal = FALSE, onefile = FALSE, width = 5.8, height = 5.8, paper = "special",
               pointsize = 10)

    filename <- paste(’/media/EPGE/Docs/tese3_tex/figures/art3_plot03_YnMd_Rn_M’, i, ’.eps’, sep = ”)

    postscript(filename)

    plot(x[!zerowxd], wdo[!zerowxd], xlab = ’Residuals of gravity equation’,
ylab = ’Spatially lagged residuals’, xlab = xrange, ylab = yrange)

    plot(x[!zerowxd], wdo[!zerowxd], xlab = ’Residuals of gravity equation’,
ylab = ’Spatially lagged residuals’, xlab = xrange, ylab = yrange)

    plot(x[!zerowxd], wdo[!zerowxd], xlab = ’Residuals of gravity equation’,
ylab = ’Spatially lagged residuals’, xlab = xrange, ylab = yrange)

    plot(x[!zerowxd], wdo[!zerowxd], xlab = ’Residuals of gravity equation’,
ylab = ’Spatially lagged residuals’, xlab = xrange, ylab = yrange)

    plot(x[!zerowxd], wdo[!zerowxd], xlab = ’Residuals of gravity equation’,
ylab = ’Spatially lagged residuals’, xlab = xrange, ylab = yrange)

    plot(x[!zerowxd], wdo[!zerowxd], xlab = ’Residuals of gravity equation’,
ylab = ’Spatially lagged residuals’, xlab = xrange, ylab = yrange)

    ps.options(horizontal = FALSE, onefile = FALSE, width = 5.8, height = 5.8, paper = "special",
               pointsize = 10)

    filename <- paste(’/media/EPGE/Docs/tese3_tex/figures/art3_plot03_YnMd_Rn_M’, i, ’.eps’, sep = ”)

    postscript(filename)
plot(x[,zerowxo], xse[,zerowxo], xlab = 'Residuals of gravity equation',
ylab = 'Spatially lagged residuals', xlim=xrange, ylim=yrange)
abline(x[0], y=0)
abline(x=x[0], y=y, lty=2)
abline(h = mean(x[,zerowxo]), lty = 3)
abline(v = mean(x[,zerowxo]), lty = 3)
legend('topleft', legend=c('regression line', 'robust regression line'), lty=1:2, bty='n')
dev.off()
}
data(oldcol)
colw <- nb2listw(COL.nb, style="W")
nsim <- 99
set.seed(1234)
simi <- moran.mc(COL.OLD$CRIME, listw=colw, nsim=nsim)
simi
mean(simi$res[1:nsim])
var(simi$res[1:nsim])
summary(simi$res[1:nsim])
MoranI.boot <- function(var, i, ...) {
  var <- var[i]
  return(moran(x=var, ...)$i)
}
set.seed(1234)
library(boot)
boot1 <- boot(COL.OLD$CRIME, statistic=MoranI.boot, R=nsim,
  sim="permutation", listw=colw, n=nrow(COL.OLD), S=Szero(colw))
boot1
plot(boot1)
mean(boot1$t)
var(boot1$t)
summary(boot1$t)
colold.lags <- nb2lag(COL.nb, 3)
set.seed(1234)
simi <- moran.mc(COL.OLD$CRIME, nb2listw(colold.lags[[2]]),
  style="W"), nsim=nsim)
summary(simi$res[1:nsim])
simi <- moran.mc(COL.OLD$CRIME, nb2listw(colold.lags[[3]]),
  style="W"), nsim=nsim)
summary(simi$res[1:nsim])

# This program reads all the explanatory
# variables used in the gravity trade model
# data is for year 2003
# obs: data will be origin-centric

zscore <- function(y, ...)(y-mean(y, ...))/sd(y, ...)

DATADIR <- "~/media/EPGE/Docs/dados/gravity"
setwd(DATADIR)
DATAf <- read.csv('gdp_pop2003.csv',row.names=1) # source Penn-World tables
summary(DATAf)
Y <- as.matrix(DATAf[,c('GDP','CGDP')])
rownames(V) <- DATAf[,1202]
summary(V)

N <- nrow(Y)
V1 <- NA
V1[1:N] <- 1 # unit vector
rownames(V1) <- rownames(Y)

YD <- kronecker(V1, Y, make.dimnames=TRUE) # destination variables
YD <- kronecker(Y, V1, make.dimnames=TRUE) # origin variables

DATAf2 <- as.matrix(read.csv('dist_main.csv',row.names=1)) # distance 1
#diag(DATAf2) <- NA # ignores INTRA country data
D1 <- as.vector(DATAf2)
DATAf2 <- as.matrix(read.csv('dist_capital.csv',row.names=1)) # distance 2
#diag(DATAf2) <- NA # ignores INTRA country data
D2 <- as.vector(DATAf2)
DATAf2 <- as.matrix(read.csv('distw.csv',row.names=1)) # distance 3
# this program creates a world map containing the selected countries
# and adds to this database the selected variables and its standardized zscores

# Starts
rm(list=ls()) # Apaga todas variaveis da memoria
library(rgdal)

# Build matrix with explanatory variables
Z <- cbind(log(YD[1], log(YD[2]), log(D1), CO, LA, RT, CN, CL, log(YD[2], 1), YD[1], 1))
Z[1, 1] <- apply(Z[1, 1], 2, 'zscore', na.rm = TRUE)

# common country
DATA5 <- as.matrix(read.csv('common_country.csv', row.names = 1))
# Table 8.4.2
DATA6 <- as.matrix(read.csv('same_country.csv', row.names = 1))

# regional trade agreement
DATA7 <- as.matrix(read.csv('same_colonizer.csv', row.names = 1))
DATA8 <- as.matrix(read.csv('colony.csv', row.names = 1))

# Calculates the World Map of Gravity (Z) and Covariates
save(Z, file = '/media/EPGE/Docs/dados/gravity/gravity_Z_covariates_garma.Rdata')

# Now reads EXPORTS
rm(list = ls())
DATA1 <- as.matrix(read.csv('exports2003.csv', row.names = 1))
DATA1 <- t(DATA1) # transforms to 'origin-centric' format
X <- as.matrix(DATA1)
X <- log(X)
names(X) <- t(outer(rownames(DATA1), rownames(DATA1), paste, sep = ""))

# saves the NA rows
naobs <- which(is.na(X))
# replaces NAs with zeroes
X[naobs] <- 0

DATA2 <- as.matrix(read.csv('imports2003.csv', row.names = 1))
DATA2 <- t(DATA2)
M <- as.matrix(DATA2)
M <- log(M)
names(M) <- t(outer(rownames(DATA2), rownames(DATA2), paste, sep = ""))

# replaces NAs with zeroes
M[naobs] <- 0

save(X, M, naobs, file = '/media/EPGE/Docs/dados/gravity/gravity_ZM_garma.Rdata')
library(spdep)
library(igraph)
library(tmap)
library(lattice)

# Define nbcairo para imprimir mais rapido na tela
X11.options(width=5,height=5, type="cairo")

map_nb01 <- poly2nb(map, row.names=IDs, queen=TRUE)
map_nb02 <- poly2nb(map, row.names=IDs, queen=FALSE)
map_nb03 <- tri2nb(coords, row.names=IDs)
map_nb04 <- graph2nb(xoi.graph(map_nb03, coords), row.names=IDs, sym=F)
# obs: SUI nao tem diferenca entre sym=T e sym=F
map_nb05 <- graph2nb(gabrielneigh(coords), row.names=IDs, sym=F)
map_nb06 <- graph2nb(relativeneigh(coords), row.names=IDs, sym=F)
map_nb07 <- knn2nb(kneighbour(coords, k=1, longlat=T), row.names=IDs, sym=F)
map_nb08 <- knn2nb(kneighbour(coords, k=3, longlat=T), row.names=IDs, sym=F)
map_nb09 <- knn2nb(kneighbour(coords, k=5, longlat=T), row.names=IDs, sym=F)
map_nb09s <- knn2nb(knearneig(coords, k=5, longlat=T), row.names=IDs, sym=T)
# Distance based neighbours
dists <- unlist(nb2dist(map_nb07, coords,longlat=TRUE)) # base eh estrutura NN k1
max_lnn <- max(dists)
map_nb10 <- dnearneig(coords, d1=0, d2=1*max_lnn, row.names=IDs, longlat=T)
map_nb11 <- dnearneig(coords, d1=0, d2=1.2*max_lnn, row.names=IDs, longlat=T)
map_nb12 <- dnearneig(coords, d1=0, d2=1.5*max_lnn, row.names=IDs, longlat=T)
map_nb13 <- knn2nb(knearneig(coords, k=N-1, longlat=T), row.names=IDs, sym=T)

# Cria lista com todas as estruturas acima definidas
nb_list <- list(Contiguity1=map_nb01, Contiguity2=map_nb02,
    Triangulation=map_nb03, S01=map_nb04, Gabriel=map_nb05,
    Gabriels=map_nb06, Relative=map_nb06, Relatives=map_nb06,
    NNk1=map_nb07, NNk3=map_nb08, NNk3s=map_nb08, NNk5=map_nb09, NNkbs=map_nb09s,
    Dist2=map_nb10, Dist2=map_nb11, Dist3=map_nb12, NNkfull=map_nb13)
return(nb_list)
}

# Rotina para gerar
# Algumas infos sobre as estruturas de vizinhanca criadas
summarynb <- function(nb_list) {
  cat("Verifica quais estruturas sao simetricas \n")
  print(sapply(nb_list, function(x) is.symmetric.nb(x, verbose=FALSE, force=TRUE)))
  cat("Verifica numero de subgrafos disjuntos \n")
  print(sapply(nb_list, function(x) n.comp.nb(x)$n))
  cat("Verifica quais estruturas sao strongly connected\n")
  print(sapply(nb_list, function(x) is.connected(graph.adjacency(nb2mat(x, style="E", zero.policy=T), mode="directed"),mode="strong"))) # verifica se o digraph eh #strongly connected
}

# funcao para imprimir mapa na tela
plotMap <- function(map, nb) {
  opar <- par(mar=c(2,2,1,1)+0.1, cex.axis=0.7)
  fig1<-plot(map, setParusrBB=TRUE, axes=TRUE, xlim=c(-160,160), ylim=c(-60,85), border="#grey60",
    cex=0.6)
  #plot(map, border="#grey60", axes=TRUE)
  fig1<-plot(nb, coordinates(map), add=TRUE, pch=19, cex=0.6, arrows=FALSE, length=0.1/2,
    points=FALSE)
  par(opar)
  return(fig1)
}

# funcao para plotar a comparacao entre vizinhos
comparenb <- function(map, nb1, nb2) {
  opar <- par(mar=c(3,3,1,1)+0.1)
  plot(map, border="#grey60", axes=TRUE)
  # text(coordinates(map), cex=0.5) # label=row.names(as(map, "data.frame"))
  plot(nb1, coordinates(map), add=TRUE, pch=19, cex=0.6, arrows=TRUE,
    length=0.1/2, points=FALSE)
  plot(differences(nb1, nb2, verbose=FALSE), coordinates(map),
    add=TRUE, pch=19, cex=0.6, col="#red", arrows=FALSE, length=0.1/2, points=FALSE)
  # par(opar)
}

# Cria as matrizes de vizinhanca com base nas estruturas de
# vizinhanca a da nb_list
# nb2mat(nb, glist=NULL, style="W", zero.policy=TRUE) # style B=binary, W=arowstand
# retorna uma w_list: primeira metade com normalizacao binaria B e segunda com W
createList <- function(nb_list, ID.names) {
  nomes<-(paste(names(nb_list), "B", sep=".")
  w_list <- vector("list", length(nomes))
  for (i in names(nb_list)) {
    w_list[[paste(i,"B",sep=".")]] <- nb2mat(nb_list[[i]], glist=NULL, style="B",
      zero.policy=T)
  rownames(w_list[[paste(i,"B",sep=".")]]) <- ID.names
  colnames(w_list[[paste(i,"B",sep=".")]]) <- ID.names
  w_list[[paste(i,"W",sep=".")]] <- nb2mat(nb_list[[i]], glist=NULL, style="W",
      zero.policy=T)
  rownames(w_list[[paste(i,"W",sep=".")]]) <- ID.names
  colnames(w_list[[paste(i,"W",sep=".")]]) <- ID.names
  }
  return(w_list)
Listing 160: bootstrap moran.R

```r
# this program runs Moran I test - monte carlo
rm(list=ls())
# loads DATA
load('dados/gravity_XM_garma.Rdata') # returns X, M
#renames imports and exports
Yx <- X
Ym <- M
rm(X)
rm(M)
# loads covariates (see prepare_data_garma.R)
load('dados/gravity_Z_covariates_garma.Rdata') # returns Z
#loads neighbor matrices
load('dados/world_neighbors_matrix.Rdata') # returns M
#renames the neighbor matrices list
Mlist <- M
```

```
# funcao para imprimir uma matriz (heat)
# se for um array com dimensao maior do que 2
# a terceira dimensao sao as fatores
plotMatrix <- function(array, order=FALSE)
{
  if (order) ord <- order.dendrogram(as.dendrogram(hclust(dist(array))))
  N <- dim(array)[1] #nao verificas dimensoes...
  levelplot(array[1:N,N:1], col.regions=gray(100:0/100),
  scales=list(draw=FALSE), xlab="spatial unit", ylab="spatial unit")
}

# inicio programa principal
load("dados/gravity_map.Rdata") #loads "SMAP"
# create neighborhood lists
WNB <- createNlist(SMAP,SMAP$ID.Names)
summaryW(N(WNB)
X11.options(width=5.8, height=2.81, type="cairo")
plotMap(SMAP,WNB[[2]])
summary(WNB[[1]])
compareW(SMAP,WNB[[15]],WNB[[16]])
M <- createList(WNB, as.character(SMAP$ID.Names))
#save(M,file="/media/EPGE/Docs/dados/gravity/world_neighbors_matrix.Rdata")

#object, when using and par( yaxs ) in addition to .
# plots all the maps
ps.options(horizontal=FALSE, onefile=FALSE, width=5.8, height=2.81, paper="special", pointsize=10)
nblist <- c(1,3,4,6,8,10,12,14,15,16,17)
for (i in 1:length(nblist)) {
  filename<-paste("/media/EPGE/Docs/tese3_tes/figures/art3_chart02_M',i,'.eps",sep="")
  postscript(filename)
  print(map)
  dev.off()
  filename2<-paste("/media/EPGE/Docs/tese3_tes/figures/art3_chart02_M',i,'.pdf",sep="")
  pdf(filename2,onefile=FALSE, width=5.8, height=2.81, paper="special", pointsize=10)
  map<-plotMap(SMAP,WNB[[nblist[i]]])
  print(map)
  dev.off()
}

oospar <- par(mar=c(3.3,1,1)+0.1)

#visualiza algumas estruturas
plotMatrix(M[[1]])
#mostra qual e maior autovvalor para cada estrutura
1/sapply(M,function(x) range(Re(eigen(x,only.values=T)$val)))
1/sapply(M,function(x) max(Mod(eigen(x,only.values=T)$val)))
## FIM
```
```r
17 rm(M)
18 # end DATA
19 # define some FUNCTIONS
20 MoranI.boot <- function(res, yhat, xmat, listw, SO) {
21 require(sandwich)
22 n <- length(res)
23 yb <- yhat + res
24 boot.lm <- lm(yb ~ xmat)
25 #boot.t <- boot.lm$coeff / sqrt(diag(vcovHC(boot.lm,"HC3")))
26 N <- length(which(card(listw$neighbors) > 0))
27 lres <- lag.listw(listw, res, zero.policy = TRUE)
28 m.i <- (N/S0) * (crossprod(res,lres)/crossprod(res))
29 #m.i <- moran(boot.lm$res, listw, n, S0, TRUE, FALSE)$I
30 return(m.i[1:1])
31 }
32 wildboot <- function(res,mle) {
33 fehat <- res/(1-mle$y) #HC3 see Davidson Flachaire 2008 for details
34 v <- 2*rbeta(length(res),1,0,5)-1 # generates random sequence of -1, 1
35 eb <- v*fehat
36 return(eb)
37 }
38 # end FUNCTIONS
39 # Starts MAIN program
40 require(spldep)
41 nblist <- c(1,3,4,6,8,10,12,14,15,16,17) # These are the 11 designs in the paper (BINARY)
42 nblist_rn <- nblist + 18 # These are de 11 designs in the paper (ROW NORMALIZED)
43 nsim <- 1999
44 set.seed(2000)
45 pvalue <- array(NA,dim=c(2,2,2,length(nblist)),dimnames=c("Yv", "Mv", "design"))
46 bootI <- array(NA,dim=c(2,2,2,length(nblist)),dimnames=c("Yv", "Mv", "design"))
47 j<-1
48 for ((Yv in list(Yx,Ym)) {
49 selected <- (Yv>0)
50 F <- Yv[selected]
51 X <- Z[selected, c("Y", "Yc", "Yd", "Yd2","Dist","Cont","Lang","Rta","Cn","Cc","Coi")]
52 regl <- lm (F~X)
53 h <- hatvalues(regl) # leverage points
54 for (i in 1:length(nblist)) {
55 Md <- kroncker(diag(nrow(Mlist[[nblist[i]]]),Mlist[[nblist[i]]]))
56 Mv <- kroncker(diag(nrow(Mlist[[nblist[i]]]),Mlist[[nblist[i]]]))
57 k <- 1
58 for (Mv in list(Md,Mv)) {
59 Mv <- Mv/max(abs(range(Re(eigen(Mv)$val)))))
60 lw <- mat2listw(Mv[selected,selected],row.names=names(Yv)[selected]) # creates a 'listw' object
61 cat("[j,k,i] = ,c(j,k,i),\"\"")
62 boot<boot (regI$res, statistic=MoranI.boot, R=nsim, sim="parametric", ran.gen = wildboot, mle=list(h=h),
63 yhat=regI$fitted, xmat=X, listw=lw, S0=Szero(1w))
64 cat("'bootstrap result: \',bootI[j,k,i]<-bootI$t0,\"\"")
65 pvalue[j,k,i]<-2*min(sum(bootI$t[,1] > bootI$t0)/nsim,sum(bootI$t[,1] <= bootI$t0)/nsim)
66 cat("p-value: \',pvalue[j,k,i],\"\"")
67 k <- k + 1
68 }
69 }
70 for (i in 1:11) {
71 Md <- kroncker(diag(nrow(Mlist[[nblist_rn[i]]]),Mlist[[nblist_rn[i]]]))
72 Mv <- kroncker(diag(nrow(Mlist[[nblist_rn[i]]]),diag(nrow(Mlist[[nblist_rn[i]]])))
73 k <- 1
74 for (Mv in list(Md,Mv)) {
75 Mv <- Mv/max(abs(range(Re(eigen(Mv)$val)))))
76 lw <- mat2listw(Mv[selected,selected],row.names=names(Yv)[selected]) # creates a 'listw' object
77 cat("[j,k,i] = ,c(j,k,i+11),\"\"")
78 boot<boot (regI$res, statistic=MoranI.boot, R=nsim, sim="parametric", ran.gen = wildboot, mle=list(h=h),
79 yhat=regI$fitted, xmat=X, listw=lw, S0=Szero(1w))
80 cat("'bootstrap result: \',bootI[j,k,i+11]<-bootI$t0,\"\"")
81 pvalue[j,k,i+11]<-2*min(sum(bootI$t[,1] > bootI$t0)/nsim,sum(bootI$t[,1] <= bootI$t0)/nsim)
82 cat("p-value: \',pvalue[j,k,i+11],\"\"")
83 k <- k + 1
84 ```
Listing 161: gamma functions.R

```
# Define Functions
gravity <- function(Y,X) {
  # runs classical linear regression (gravity)
  require(lmtest)
  require(sandwich)
  # Eliminates INTRA country flows Y1
  selected <- which(Y1==0)
  Y <- Y[selected]
  X <- X[selected,]
  print(summary(lm(Y~lm(X[Y1]))))
  print(coeftest(lm(Y~lm(X[Y1]),y=lm(X1)),df=Inf)) # check t-values when Heteroskedasticity is present
  return(reg)
}
# obtain world map
getWorldMap <- function() {
  require(rgdal)
  MAPDIR<-"/media/EPGE/Docs/dados/mapas/world/Country00"
  MAPFILE<-"country00"
  MAPFULL <- readOGR(MAPDIR,MAPFILE,pd="+proj=longlat")
  return(MAPFULL)
}
# plots the residuals in the map
plotDataMap <- function(uhat,country,MAPFULL) {
  require(sp)
  selctr<-grep(country,names(uhat)) # finds data
  selnames<-subset(country,','names(uhat[selctr])
  MAPISID2<-MAPFULL$data[,`ISO_2DIGIT`] # the NAs are ok, not the countries of interest
  SELECTED <- match(selnames,MAPISID2)
  MAP <- MAPFULL[SELECTED,]
  PID <- sapply(slot(MAP, "polygons"), function(i) slot(i, "ID"))
  atributos <- data.frame(ID.Name=as.factor(selnames),uhat[selctr],row.names=PID)
  SMAP <- SpatialPolygonsDataFrame(MAP,atributos)
  map2print<-.splot(SMAP,zcol=2,col.regions=grey.colors(5,0.95,0),aspect=mapsp(SMAP),
  scale=0.15,draw=TRUE)
  at<-(min(uhat[selctr])-0.001,quantile(uhat[selctr],prob=seq(0.2,0.8,0.2),type=7),
  max(uhat[selctr])+0.001))
  return(map2print)
}
# note that this function estimates GARMA WITHOUT the term M1 X1
# Note that X MUST include the constant
estimateGARMA <- function(Y,X,M,M1,M2,M3,algs='optim') {
  require(Matrix)
  Delta <- cbind(M1*M1,Y,X)
  colnames(Delta)[1] <- 'MY'
  if (rankMatrix(H)[1] > ncol(Delta)) { # cat('OK!! There are enough instruments to go ahead.
    return(NA)
  }
  deltathat <- IV.gravity(Y,Delta,H)
  That <- solve(diag(length(Y)))+deltathat[1]*M1%*%X%*%deltathat[-1]
  what <- Y - That
  # moments
  klist <- genMoments(M2) # note that this function needs to be changed to use different Ks
  gmnoptim <- GM_arma(uhat,M2,klist,runif(2,-0.5,0.5),algs)
  gmnoptimpar <- gmnoptim$par
  if (algs=='optim') Objf <- gmnoptim$value
  if (algs=='port') Objf <- gmnoptim$objective
  vcttheta <- vcovTheta(thetahat,uhat,M2,klist)
  ehat <- genEhat(thetahat,uhat,M2)
  Uppsilonhat <- genUpsilon(deltathat[1],thetahat,ehat,M1,M2)
  vcDelta <- vcovDelta(Delta,H,Uppsilonhat)
```
# calculate standard deviations
thetahat.sd <- sqrt(diag(vctbeta))
deltahat.sd <- sqrt(diag(vcdelta))

# calculate t-values
thetahat.t <- thetahat/thetahat.sd
deltahat.t <- deltalhat/deltalhat.sd

# residuals
SSR <- sum((Yhat-mean(Y))~2) # note that there is a CONSTANT
ESS <- sum((Yhat-mean(Y)^2)
TSS <- sum((Y-mean(Y)^2)

pseudoR2 <- ESS/TSS

df.residuals <- length(ehat.length(thetahat.length(deltahat)

# now estimate the RESTRICTED model (imposing lambda=0, alpha=0, rho=0)
# it is a linear regression model with heteroskedastic innovations
# it can be estimated by OLS
olsmodel<lm(Y~1)

R pseudoR <- sum((olsmodel$res^2)

#pseudoF <- (SSR(restricted) - SSR(unrestricted)) / r
# SSR(unrestricted)/(n-k)
# r.PseudoF <- Chi^2(r)
pseudoF <- ((SSR - SSR) / SSR * df.residuals/3 # since there are r=3 restrictions
pFvalue <- pchisq(pseudoF, 3, lower.tail=FALSE)

return(list(data = list(Y=Y,X=X,H=H,M1=M1,M2=M2),
        convInfo = list(code=gmmoptim$convergence, msg=gmmoptim$message),
        Qhat = Db[,,
        coefficients = c(deltahat,thetahat),
        coef.sd = c(deltahat.sd,thetahat.sd),
        tstat = c(deltahat.t,thetahat.t),
        what = what,
        ehat = ehat,
        fitted.values = what,
        pseudoR2 = pseudoR2,
        pseudoF = list(stat=pseudoF, pvalue=pFvalue),
        ss = list(ss=SSR, ess=ESS, tss=TSS))
})

# plots spatial matrix
plotMatrix <- function(array, order=FALSE) {
if (order) ord<order.dendrogram(as.dendrogram(hclust(dist(array)))))
X<as.matrix(array)[1] # nna verify as dimensoes
levelplot(array[1:N,M1], col.regions=gray(100:0/100),

scales=list(draw=FALSE), xlab="spatial unit", ylab="spatial unit")
}

# estimates the covariance parameters of the GARMA model
GMM arma <- function(what,M2,klist,startval,algo='optim') {
require(gmm)
require(MASS)
Qhat <- function(theta,what,M2,klist,W) {
g<ghat(theta,what,M2,klist)
return(1*crossprod(g,W*x%*%g)/2) # this function will be MINIMIZED!!
}

ghat <- function(theta,what,M2,klist) {
alpha<-theta[1]
rho <- theta[2]
ARU < solve(Aa(alpha,M2))%*%R(rho,M2)%*%x%*%what
gx < matrix(NA,nrow=length(klist),ncol=1)
for (i in 1:length(klist)) gh[i,1] <- crossprod(ARU,klist[[i]]%*%ARU)
return(gh/length(uhat))
}
gQhat <- function(theta,uhat,M2,klist,W)
alpha<-theta[1]
rho <-theta[2]
Ainv <- solve(Aa(alpha,M2))
ARU <- Ainv%*%Rr(rho,M2)%*%uhat
AM <- Ainv%*%M2
ks <- vector(“list”,length(klist))
for (i in 1:length(klist)) ks[[i]] <- klist[[i]]+t(klist[[i]])
dg <- matrix(NA,nrow=length(klist),ncol=2)
for (i in 1:length(klist)) {
dg[i,1] <- crossprod(ARU,ks[[i]]%*%AM%*%ARU)
dg[i,2] <- -1*crossprod(ARU,ks[[i]]%*%AM%*%uhat)
}
dg <- dg/length(uhat)
g <- matrix(NA,nrow=length(klist),ncol=1)
for (i in 1:length(klist)) g[i,1] <- crossprod(ARU,klist[[i]]%*%ARU)
g <- g/length(uhat)
return(1+crossprod(g,W%*%dg)) # W is SYMMETRIC

# first GMM step with W=identity
cat(‘First GMM step...’)
W <- diag(length(klist))
if (algo=='optim') {
  opt1<-optim(par=theta0, gr=gQhat, uhat,M2,klist,W,method='L-BFGS-B',
    lower=(-1+.Machine$double.eps)^0.3*c(1,1)),
    upper=(1-.Machine$double.eps)^0.3*c(1,1)),
    control=list(trunc=3, factr=10, pgtol=0, fnscale=1))
}
if (algo=='nlm') {
  opt1<-nlm(b=theta0, gradient=gQhat, hessian=NULL, uhat,M2,klist,W,
    lower=(-1-.Machine$double.eps)^0.3*c(1,1)),
    upper=(1-.Machine$double.eps)^0.3*c(1,1)),
    control=list(trunc=2))
}
cat(‘DONE!\n’)
theta0<-opt1$par
cat(‘theta0: ‘,theta0,’\n’)
cat(‘(convergence: ‘,opt1$convergence,’(msg): ‘,opt1$message,’\n’)
cat(‘gQhat: ‘,gQhat(theta0,uhat,M2,klist),’\n’)

# second GMM step with W=Omega^{-1}
# what <- genEhat(theta0,uhat,M2) # generates the consistently estimated what
W <- genOmegaMatinv(ehat,klist)
#cat(‘The empirical vcov of g (inverse): ‘,round(ginv(tcrossprod(what(theta0,uhat,M2,
  klist))%*%length(uhat)),3),’\n’)
#cat(‘The theoretical vcov Omega (inv): ‘,round(W,3),’\n’)
cat(‘Eigenvalues of Omega^{-1} ‘,eigen(W)$val,’\n’)
cat(‘Second GMM step...’)
if (algo=='optim') {
  finalopt<-optim(theta0, Qhat, gr=gQhat, uhat,M2,klist,W, method=’L-BFGS-B’,
    lower=(-1+.Machine$double.eps)^0.3*c(1,1)),
    upper=(1-.Machine$double.eps)^0.3*c(1,1)),
    hessian=FALSE, control=list(trunc=3, factr=10, pgtol=0, fnscale=1))
}
if (algo=='nlm') {
  finalopt<-nlm(b=theta0, gradient=gQhat, hessian=NULL, uhat,M2,klist,W,
    lower=(-1-.Machine$double.eps)^0.3*c(1,1)),
    upper=(1-.Machine$double.eps)^0.3*c(1,1)),
    control=list(trunc=2))
}
cat(‘DONE!\n’)
cat(‘theta1: ‘,finalopt$par,’\n’)
cat(‘(convergence: ‘,(finalopt$convergence,’(msg): ‘,finalopt$message,’\n’)
cat(‘gQhat: ‘,gQhat(finalopt$par,uhat,M2,klist),’\n’)
return(finalopt)
}

# Generates a consistently estimated residual
genEhat <- function(theta,uhat,M2)
return(as.vector(solve(Aa(theta[1],M2))%*%Rr(theta[2],M2)%*%uhat))
# generates GO (consistent) estimator
208 genGO <- function(theta, ehat, M2, klist) {
209 ehat <- as.vector(ehat) # make sure ehat is appropriate for diag()
210 Shat <- diag(ehat^2)
211 ks <- vector('list', length(klist))
212 Sks <- vector('list', length(klist))
213 for (i in 1:length(klist)) {
214 ks[[i]] <- klist[[i]] + t(klist[[i]])
215 Sks[[i]] <- Shat %*% Sks[[i]]
216 }
217 }
218 AM <- solve(Aa(theta[1], M2)) %*% XM2
219 RM <- solve(Rr(theta[2], M2)) %*% XM2
220 GO <- matrix(NA, nrow=length(klist), ncol=length(klist))
221 for (i in 1:length(klist)) {
222 GO[i,1] <- sum(diag(Sks[[i]])) %*% XMAM
223 GO[i,2] <- -1*sum(diag(Sks[[i]])) %*% XRM
224 }
225 return(GO/length(ehat))
226 }
227 # create the moment matrices
228 genMoments <- function(M2) {
229 M2_2 <- M2 %*% XM2
230 M2_3 <- M2_2 %*% XM2
231 M2_4 <- M2_2 %*% M2_2
232 M2_5 <- M2_2 %*% M2_3
233 #M2_6 <- M2_5 %*% XM2
234 #M2_7 <- M2_6 %*% XM2
235 K1 <- M2_2
236 K2 <- M2_2 # note that I am playing around with this...
237 K3 <- M2_3
238 K4 <- M2_4
239 K5 <- M2_5
240 #K6 <- M2_6
241 #K7 <- M2_7
242 diag(K2)<0
243 diag(K3)<0
244 diag(K4)<0
245 diag(K5)<0
246 #diag(K6)<0
247 #diag(K7)<0
248 klist <- list(K1, K2, K3, K4, K5) #, K6, K7)
249 return(klist)
250 }
251 #
252 vecTheta <- function(theta, ehat, M2, klist) {
253 ehat <- genEhat(theta, ehat, M2)
254 GO <- genGO(theta, ehat, M2, klist)
255 Omegainv <- genOmegahatinv(ehat, klist)
256 return(crossprod(GO, Omegainv %*% XGO)/length(ehat))
257 }
258 #
259 genUpsilon <- function(lambda, theta, ehat, M1, M2) {
260 ehat <- as.vector(ehat)
261 Shat <- diag(ehat^2)
262 LRA <- (diag(length(ehat)) - lambda %*% %solve(Rr(theta[2], M2)) %*% Aa(theta[1], M2)
263 return(tcrossprod(LRA %*% %Shat, LRA))
264 }
265 #
266 vecDelta <- function(Delta, H, Upsilon) {
267 require(Matrix)
268 PH <- H %*% ginv(crossprod(H)) %*% t(H)
269 PhD <- PH %*% Delta
270 DPhDi <- solve(crossprod(Delta, PhD))
271 return(DPhDi %*% tcrossprod(PhD, Upsilon %*% PhD) %*% DPhDi)
272 }
273 # Classical IV regression
274 # note that the CONSTANT NEEDS to be included in matrix DELTA
275 IV.gravity <- function(Y, Delta, H, eta=diag(length(Y)), adjH=FALSE) {
276 require(Matrix)
277 require(MASS)
278 Delta <- cbind(eta %*% Delta[,1], eta %*% Delta[,2])
279 if (adjH) H <- eta %*% H
280 Y <- eta %*% Y

Listing 162: generate garma estimates.R

```r
Ph <- H%*%ginv(crossprod(H))%*%t(H)
PhDelta <- Ph%*%Delta
deltahat <- solve(crossprod(Delta,PhDelta))%*%crossprod(PhDelta,Y)
rownames(deltahat) <- colnames(Delta)
colnames(deltahat) <- 'Estimate'
return(deltahat)
}
# Generates Omega (inverse)
genOmegaHat <- function(epsilon,klist) {
  ehat <- as.vector(epsilon) # makes sure it is a vector so that diag() works properly
  Shat <- diag(epsilon^2)
  ks <- vector(paste1('list',length(klist)))
  Sk <- vector(paste1('list',length(klist)))
  Sks <- vector(paste1('list',length(klist)))
  for (i in 1:length(klist)) {
    ks[[i]] <- klist[[i]]
    Sks[[i]] <- Shat%*%klist[[i]]
  }
  OmegaHat <- matrix(NA,nrow=length(klist),ncol=length(klist))
  for (i in 1:length(klist))
    for (j in 1:length(klist))
      OmegaHat[i,j] <- sum(diag(Sk[[j]]%*%Sks[[i]]))
  return(OmegaHat)
}
# Generates estimates based on the previous work of KP
genGMMIVestimate <- function(Yx,Yd,Z,oblist,Mlist,Yvar,Mvar) {
  list <- vector(paste1('list',12))
  #returnlist <- vector('list',12)
  #returnlistcompare <- vector('list',12)
  if (Yvar == 'Yx') Yv <- Yx
  if (Yvar == 'Yd') Yv <- Yd
  # which independent variable ? (Yx, Ym, Yt)
  # select only non NULL elements
  selected <- which(Yv!=0)
  Y <- Yv[selected]
  X <- cbind(1,2[selected,1:6])
  for (i in 1:length(oblist))
    if (running < oblist[[i]])
      Y <- c(Y,mv)
    if (Mvar == 'Mv')
      Mv <- kron(ks[[i]],Mvar)
    if (Mvar == 'Mv')
      Mv <- kron(ks[[i]],Mvar)
    # INSTEAD of norm=1, I will make the LARGEST Eigenvalue correction
    Mv <- Mv/max(abs(range(real(eigen(Mv)$value))))
    cat('Using',Yvar,'...n')
    cat('Using',Mvar,'...n')
    # Define matrices M1,M2
    M1 <- M2 <- M[selected,selected]
    # define INSTRUMENTS
    H <- cbind(1,2[selected,1:6],
      M1%*%X[1:6],
      M1%*%X[1:6][selected,1:6])
    # NOTE: instruments are automatically selected in GMMIV
    returnlist[[1]] <- GMMIV(X,Y,H,M1,M2,verbose=TRUE)
    #returnlistcompare[[1]] <- estimateGARMA(Y,h,M1,M2)
    cat('CONGRATULATIONS!! Step(','i','') was finished!\n')
    cat('------\n')
  }
  return(returnlist)
}
## end Functions

rm(list=ls())
source('/home/pgralh/Drpbox/epge/docs/programs/tese3/functions/garma_functions.R')
## define Functions
source('/home/pgralh/Drpbox/epge/docs/programs/tese3/functions/genGarmaestimate.R')
prepare_table <- function(modelY) {
  ubound <- 1-Machine$double.eps*0.3
  N <- length(modelY[[1]]$epsilon)
  tab <- cbind(modelY[[1]]$coefficients,modelY[[1]]$ssssr/N,modelY[[1]]$Qhat*100,
    modelY[[1]]$convInfo$code),
```
c(modelY[[1]]$coefficients [16] == ubound, NA, NA, NA))
rownames(tabY) <- c('lambda', 'Intercept', 'Yo', 'YoC', 'Yd', 'YdC', 'Dist',
  'Cont', 'Lang', 'Rta', 'Cn', 'Col', 'alpha', 'rho', 'SSR/n', 'Qhat(x100)', 'Conv?')
# note that I skip model '/2 since it is the same as '/1
for (i in 3:12) tabY <- cbind(tabY, cbind(c(modelY[[i]]$coefficients, modelY[[i]]$s$ssr/N , modelY[[i]]$Qhat*x100, modelY[[i]]$convInfo$code),
# save(list = ls(), file = 'dados/modelY_BIN_port.Rdata')
modelYMobin <- genGARMAestimate(Yx, Ym, Z, nblist, Mlist, Ym, 'Mo', algo)
# save(list = ls(pattern = 'modelY'), file = 'dados/modelY_BIN_port.Rdata')

## end MAIN

Listing 163: generate maps.R

```r
rm(list=ls())
source('garma_functions.R')
## Loads DATA
## loads independent variables
load('dados/gravity_XM_garma.Rdata') # returns X, M
# removes import and exports
Yx <- X
Ym <- M
Yt <- log((exp(Yx) + exp(Ym))/2)
rm(X)
rm(M)
## loads covariates (see prepare_data_garma.R)
load('dados/gravity_Z_covariates_garma.Rdata') # returns Z
## end DATA
## Starts MAIN program

#X11.options(width=5, height=5, type='nbcairo')
ps.options(horizontal=FALSE, onefile=FALSE, width=5.8, height=2.81, paper="special",
            pointsize=10)

fullmap <- getWorldMap()

for (Yvar in c('X', 'M')) {
  if (Yvar == 'X') Yv <- Yx
  if (Yvar == 'M') Yv <- Ym
  selected <- which(Yv == 0)
  if (Yv[selected] == 0)
    X <- Z[selected, c('Ya', 'YaC', 'Yd', 'YdC', 'Dist', 'Cont', 'Lang', 'Rta', 'Cn', 'Cc', 'Col')]
  reg1 <- lm(YX)
  for (Cty in countrylist) {
    filename <- paste('~/media/EPGE/Docs/tese3_tex/figures/art3_chart01_1.Yvar', '
    substr(Cty, 1, 2)', '/', 'eps', '/sep')
    postscript(filename)
    mapi <- plotDataMap(reg1$res, Cty, fullmap)
    print(mapi)
    dev.off()
    filename2 <- paste('~/media/EPGE/Docs/tese3_tex/figures/art3_chart01_1.Yvar', '
    substr(Cty, 1, 2)', '/', 'pdf', '/sep')
    pdf(filename2, onefile=FALSE, width=5.8, height=2.81, paper="special", pointsize=10)
    print(mapi)
    dev.off()
    cat(Yvar, 'for country', Cty, 'saved!
    ')
  }
}
# check the results from some countries to write the text
# Argentina exports
selected <- which(YX == 0)
Y <- Yx[selected]
X <- Z[selected, c('Ya', 'YaC', 'Yd', 'YdC', 'Dist', 'Cont', 'Lang', 'Rta', 'Cn', 'Cc', 'Col')]
reg1 <- lm(YX)
selected <- grep('AR:', names(reg1$res)) # finds data
selnames <- sub('AR:', '', names(reg1$res[selcstr]))
print(tmp <- quantile(reg1$res[selcstr], probs = seq(0, 1, 0.22))
(reg1$res[selcstr])[reg1$res[selcstr] >= tmp[6]]
(reg1$res[selcstr])[reg1$res[selcstr] <= tmp[2]]
# Argentina imports
selected <- which(Ym == 0)
Y <- Ym[selected]
X <- Z[selected, c('Ya', 'YaC', 'Yd', 'YdC', 'Dist', 'Cont', 'Lang', 'Rta', 'Cn', 'Cc', 'Col')]
reg1 <- lm(YX)
selected <- grep('AR:', names(reg1$res)) # finds data
selnames <- sub('AR:', '', names(reg1$res[selcstr]))
```


Listing 164: genGarmaestimate.R

```r
#秦皇岛 estimate <- function (Xy, ym, z, nblist, Mlist, Yvar, Mvar, algo) {
  returnlist <- vector('list', length(nblist))
  for (i in 1:length(nblist)) {
    cat('

    # Define matrices M1, M2
    M1 <- M2 <- Mv[1]
    cat('

    # define instruments
    H <- cbind(1, Mlist[1], c('Yo', 'YoC', 'Yd', 'YdC', 'Dist', 'Cont', 'Lang', 'Rta', 'Cn', 'Cc', 'Col'))
    cat('

    # instead of norm=1, I will make the largest eigenvalue correction
    MV <- MV/abs(max(range(Re(eigen(MV)$value))))
    # Which independent variable? (Yx, Ym, Yt)
    if (Yvar == 'Yx') Yv <- X
    if (Yvar == 'Ym') Yv <- Y
    Y <- Yv[1]
    X <- cbind(1, z[1], c('Yo', 'YoC', 'Yd', 'YdC', 'Dist', 'Cont', 'Lang', 'Rta', 'Cn', 'Cc', 'Col'))
    colnames(X)[1] <- 'Intercept'
    cat('

    # define matrices M1, M2
    M1 <- M2 <- Mv[1]
    cat('

    # define instruments
    H <- cbind(1, Mlist[1], c('Yo', 'YoC', 'Yd', 'YdC', 'Dist', 'Cont', 'Lang', 'Rta', 'Cn', 'Cc', 'Col'))
    M1%*%X[1] <- estimateGarma (Yx, X, H, M1, M2, algo)
    returnlist[[i]] <- estimateGarma (Yx, X, H, M1, M2, algo)
  }
  return(returnlist)
}
```

Listing 165: traditional gravity.R

```r
# This program obtains the traditional gravity equations
rm(list=ls())
#Load DATA
# loads independent variables
load('/media/EPGE/Docs/dados/gravity/gravity_XM_garma.Rdata') # returns X, M
# names imports and exports
Yx <- X
Ym <- M
```
Yt <- log((exp(Yx) + exp(Ym))/2)
rm(X)
rm(M)
# loads covariates (see prepare_data_garma.R)
load('/media/EPGE/Docs/dados/gravity/gravity_Z.covariates_garma.Rdata') # returns Z
# loads neighbor matrices
load('/media/EPGE/Docs/dados/gravity/world_neighbors_matrix.Rdata') # returns M
# renames the neighbor matrices list
Mlist <- M
# end DATA
## Starts MAIN program
selected <- (Yx!=0)
F <- Yx[selected]
X <- Z[selected,c('Yo', 'YoC', 'Yd', 'YdC', 'Dist', 'Cont', 'Lang', 'Rta', 'Cn', 'C', 'Col')]
#popD <- Z[selected,'Yd'] - Z[selected,'YoC']
#popD <- Z[selected,'Yd'] - Z[selected,'YdC']
x1 <- X[, 'YoC']^2
x2 <- X[, 'YdC']^2
x3 <- X[, 'Yo']^2
x4 <- X[, 'Yd']^2
x5 <- X[, 'Dist']^2
sreg1 <- summary(reg1 <- lm(F~X))
require(lmtest)
require(sandwich)
tab <- print(coefTest(reg1, vcov=vcovHC(reg1,type='HC3'), df=Inf)) # check t-values when Heteroskedasticity is present
require(xtable)
latex.tab <- xtable(sreg1, caption='Traditional gravity equation (heteroskedastic robust standard errors)',
label='tab:gravity_results')
print(latex.tab)

# Repeat with imports
F <- Ym[selected]
sreg2 <- summary(reg2 <- lm(F~X))
tab2 <- print(coefTest(reg2, vcov=vcovHC(reg2,type='HC3'), df=Inf)) # check t-values when Heteroskedasticity is present

# Creates a table similar to the one in Tinbergen's book (page 272)
# note that I need ot use a taylor approx to calculate the expected value of exp(X)
# E exp(x) = exp(Ex) + exp(Ex)/2 * var(x)
library(e1071)
m20x < NA
for (i in 1:20) m20x[i] <- moment(reg1$res, order=i, center=TRUE)/factorial(i)
Yxhat <- (exp(reg1$fitted)+sum(m20x))
Yxactual <- (exp(Yx[selected]))
m20y < NA
for (i in 1:20) m20y[i] <- moment(reg2$res, order=i, center=TRUE)/factorial(i)
Ymhat <- (exp(reg2$fitted)+sum(m20y))
Ymactual <- (exp(Ym[selected]))
plot(Yxactual, Yxhat)

countrylist <- c("AR:", "AU:", "BD:", "BE:", "BR:", "CA:", "CN:", "CU:", "EG:",
"FR:", "DE:", "IN:", "ID:", "IT:", "JP:", "MY:", "MX:",
"NL:", "PK:", "PH:", "PL:", "KR:", "RU:", "SA:", "ZA:", "ES:",
"SE:", "TH:", "TR:", "UA:", "GB:", "US:")
countrynames <- c('Argentina', 'Australia', 'Bangladesh', 'Belgium', 'Brazil', 'Canada', 'China', 'Colombia', 'Egypt', 'France', 'Germany', 'India', 'Indonesia', 'Iran', 'Italy', 'Japan', 'Malaysia', 'Mexico', 'Netherlands', 'Pakistan', 'Philippines', 'Poland', 'Rep. of Korea', 'Russian Federation', 'Saudi Arabia', 'South Africa', 'Spain', 'Sweden', 'Thailand', 'Turkey', 'Ukraine', 'United Kingdom', 'United States')
tabinbergen <- matrix(NA, nrow=length(countrylist), ncol=4)
for (i in 1:length(countrylist)) {

country <- countrylist[i]
select <- grep(country, names(Yxactual)) # finds data

tot.actual <- sum(Yxactual[select]) / 10000000000 # in USD billion
tot.proj <- sum(Yxhat[select]) / 10000000000 # in USD billion
	xdev <- tot.actual - tot.proj
xper <- xdev / tot.actual * 100

tot.actual <- sum(Yxactual[select]) / 10000000000 # in USD billion
tot.proj <- sum(Yxhat[select]) / 10000000000 # in USD billion
mdev <- tot.actual - tot.proj
mper <- mdev / tot.actual * 100

# Tinbergen[i,] <- c(xdev, xper, mdev, mper)

rownames(tabTinbergen) <- countrynames

latex.tabTinbergen <- xtable(cbind(tabTinbergen[,1:2], NA, tabTinbergen[,3:4]),
caption = 'Total actual trade minus total calculated trade',
label = 'tab:tinbergen', digits = 2)

print(latex.tabTinbergen)

colSums(tabTinbergen)
Bibliography


Weisstein, E. W., “Delaunay Triangulation.”