

Pedro Rafael Diniz Marinho

**Estimadores Intervalares sob
Heteroscedasticidade de Forma Desconhecida
via Bootstrap Duplo**

Recife

27 de fevereiro de 2014

Pedro Rafael Diniz Marinho

Estimadores Intervalares sob Heteroscedasticidade de Forma Desconhecida via Bootstrap Duplo

Trabalho apresentado ao Programa de Pós-graduação em Estatística do Departamento de Estatística da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Mestre em Estatística.

Universidade Federal de Pernambuco
Centro de Ciências Exatas e da Natureza
Departamento de Estatística

Orientador: Francisco Cribari Neto

Área de Concentração: Estatística Aplicada

Recife

27 de fevereiro de 2014

Catálogo na fonte
Bibliotecária Jane Souto Maior, CRB4-571

Marinho, Pedro Rafael Diniz

**Estimadores intervalares sob heteroscedasticidade de
forma desconhecida via bootstrap duplo / Pedro Rafael Diniz
Marinho. - Recife: O Autor, 2014.**

192 f.: il., fig., tab.

Orientador: Francisco Cribari Neto.

**Dissertação (mestrado) - Universidade Federal de Pernambuco.
CCEN, Estatística, 2014.**

Inclui referências e apêndices.

**1. Estatística. 2. Regressão. 3. Bootstrap. I. Cribari Neto,
Francisco (orientador). II. Título.**

310

CDD (23. ed.)

MEI2014 – 026

Universidade Federal de Pernambuco
Pós-Graduação em Estatística

27 de fevereiro de 2014

Nós recomendamos que a Dissertação de Mestrado de autoria de

Pedro Rafael Diniz Marinho

Intitulada:

“Estimadores Intervalares sob Heteroscedasticidade de Forma Desconhecida via Bootstrap Duplo”

Seja aceita como cumprimento parcial dos requerimentos para o grau de Mestre em Estatística.

Coordenador da Pós-Graduação em Estatística

Banca Examinadora:

Francisco Cribari Neto

Orientador/UFPE

Raydonal Ospina Martínez

UFPE

Aluísio de Souza Pinheiro

UNICAMP

Este documento será anexado à versão final da tese.

A Deus e aos meus pais Walter e Wilta.

Agradecimentos

Aos meus pais por todo o incentivo e por acreditar em mim durante todos esses anos. Tudo que sou hoje devo a vocês.

Ao meu orientador, Francisco Cribari Neto, por toda ajuda, conselhos, sugestões e críticas que contribuíram para construção dessa dissertação.

Aos demais professores do Departamento de Estatística da UFPE, em especial aos professores Gauss Moutinho Cordeiro, Leandro Chaves Rego e Audrey Helen Mariz de Aquino Cysneiros, por uma parte de seus conhecimentos passados em suas disciplinas.

A minha grande amiga e namorada Emanuelle Waleska Almeida de Farias por todas as palavras de conforto e por acreditar sempre em mim.

Aos meus familiares por todo incentivo.

A meu primo e grande amigo Gilberto Wilson Diniz de Luna por todos os conselhos, incentivos e boas conversas que sempre tivemos.

Ao meu amigo Marco Túlio Porto, que sempre acreditou no meu potencial e muitas vezes me incentivou para fazer o mestrado em estatística.

A todos os amigos do curso pelas inúmeras horas de estudos.

A Valéria Bittencourt por me ajudar sempre que precisei e por todos os conselhos que me foi dado. Você tem todo meu respeito e carinho.

Ao Centro Nacional de Super Computação (CESUP) por disponibilizar seus computadores para a realização das simulações desse trabalho.

À Fundação de Amparo à Ciência e Tecnologia do Estado de Pernambuco (FACEPE) pelo auxílio financeiro.

Resumo

Uma parte considerável das análises empíricas nas mais variadas áreas do conhecimento emprega modelagem de regressão. Um dos modelos de regressão mais utilizados é o modelo linear e tipicamente está associada uma suposição sobre a constância das variâncias dos erros ($\text{var}(\varepsilon_i) = \sigma^2$, $i = 1, 2, \dots, n$). Este modelo é denominado de modelo linear homoscedástico de regressão. Inferências em modelos lineares homoscedásticos já estão bem desenvolvidas, sendo possível fazer estimativas pontuais e intervalares e também testar hipóteses com facilidade.

Em muitos problemas em que a modelagem linear é adequada a suposição de homoscedasticidade não é válida, ou seja, os erros são heteroscedásticos ($\text{var}(\varepsilon_i) = \sigma_i^2$, $i = 1, 2, \dots, n$), a heteroscedasticidade tipicamente sendo de forma desconhecida. A inferência que é válida para modelos lineares homoscedásticos não é mais aplicável a modelos lineares heteroscedásticos, sobretudo, quando se trata de estimativas intervalares e testes de hipóteses.

Esse trabalho propõe avaliar estratégias de estimação intervalar para os parâmetros que indexam o modelo linear heteroscedástico de regressão (β_j , $j = 1, \dots, p$). As metodologias para construção de estimativas intervalares foram avaliadas via simulações de Monte Carlo sob cenários diferentes. Buscou-se avaliar os desempenhos dos estimadores em pequenas amostras considerando dados balanceados e não balanceados, i.e., ausência e presença de pontos de alta alavancagem nos dados, sob diferentes níveis de heteroscedasticidade. Foram avaliados os desempenhos de estimadores intervalares para um parâmetro β_j construídos a partir de uma estimativa consistente do desvio padrão do estimador de β ($\hat{\beta}_j$). Os estimadores da estrutura de covariância dos estimadores dos parâmetros que indexam o modelo linear de regressão utilizam os estimadores HC0, HC2, HC3, HC4 e HC5. Também foram avaliadas inferências intervalares de estimadores que utilizam esquemas de reamostragem via bootstrap selvagem. Foram considerados os métodos bootstrap percentil e bootstrap- t em esquemas simples e duplo, i.e., em esquemas com apenas um nível bootstrap e naqueles que consideram um segundo nível de bootstrap.

As avaliações das diferentes estratégias de estimação intervalar tiveram custo computacional muito elevado. O programa utilizado para realizar as simulações foi escrito na linguagem C++, sendo necessário usar computação paralela (OpenMP - *Open Multi-Processing*) e realizar as simulações em um supercomputador, sendo elas executadas em simultâneo. Assim, as simulações puderam ser realizadas em tempo viável. Todas as metodologias bootstrap consideradas nesse trabalho foram reunidas no pacote `hcci` versão 1.0.0, que está disponível gratuitamente no site da linguagem R.

Palavras-chaves: bootstrap, bootstrap duplo, bootstrap percentil, bootstrap- t , estimação intervalar, `hcci`, heteroscedasticidade.

Abstract

Many empirical analyses in many fields employ regression modeling. One of the models most widely used models is the linear regression model. It is commonly assumed that the error variances are constant ($\text{var}(\varepsilon_i) = \sigma^2$, $i = 1, 2, \dots, n$). This model is called homoskedastic linear model regression. Inferences on linear homoskedastic models are already well developed, making it possible to obtain point and interval estimates and to test hypotheses.

In many problems in which the linear model is appropriate the assumption of homoskedasticity is not valid, i.e., the errors are heteroskedastic ($\text{var}(\varepsilon_i) = \sigma_i^2$, $i = 1, 2, \dots, n$), the heteroskedasticity typically being of unknown form. Inferences that are valid for linear homoskedastic models are no longer applicable to linear heteroskedastic models, especially, when it comes to interval estimation and hypotheses testing.

In this thesis we assess strategies for interval estimation of the parameters that index the linear heteroskedastic model regression. Different interval estimates were evaluated via Monte Carlo simulation under different scenarios. We evaluated their performances in small samples under balanced and unbalanced, data i.e., without and with of points of high leverage in the data, under different levels of heteroskedasticity. Were evaluated the performances of interval estimators for β_j constructed from consistent standard errors. The estimators of the covariance structure of the parameters estimators which index the linear model of regression we used are HC0, HC2, HC3, HC4 and HC5. Were also evaluated interval estimators that use wild bootstrap schemes. Were considered percentile and bootstrap- t schemes simple and double versions, i.e., in schemes with only one level bootstrap and in schemes that consider a second level of bootstrap.

The evaluations of the different estimation strategies interspaced entailed very high computational cost. The program used to carry out the simulations was written using the C++ language, being necessary to use parallel computing (OpenMP - *Open Multi-Processing*) and carry out the simulations in a supercomputer. All bootstrap estimators considered in this study were implemented into the `hcci` package version 1.0.0, which is available for download at the R language main web page.

Key-words: bootstrap, bootstrap percentile, bootstrap- t , double bootstrap, `hcci`, heteroscedasticity, interval estimation.

Resumen

Una parte considerable de los análisis empíricos en diversas áreas del conocimiento emplea modelos de regresión. Uno de los modelos de regresión más usado comúnmente es el modelo lineal que hace suposiciones sobre la constancia de la varianza del error ($\text{var}(\varepsilon_i) = \sigma^2$, $i = 1, 2, \dots, n$). Este modelo se denomina un modelo de regresión lineal homoscedástico. Inferencias en modelos lineales homoscedásticos ya están bien desarrolladas, siendo posible hacer estimativas puntuales, intervalares y probar hipótesis con facilidad.

En muchos problemas en que el modelo lineal es adecuado el supuesto de homoscedasticidad no es válido. La inferencia que es válida para los modelos lineales homoscedásticos ya no es aplicable a los modelos heteroscedásticos, especialmente, en el caso de las estimaciones de intervalos y pruebas de hipótesis.

Este trabajo tiene como objetivo evaluar las estrategias para la estimación de intervalos para los parámetros del modelo lineal heteroscedastico de regresión (β_j , $j = 1, \dots, p$). Las metodologías para la construcción de las estimaciones de intervalos fueron evaluadas a través de simulación de Monte Carlo, donde se consideran diversos escenarios. Hemos tratado de evaluar el desempeño de los estimadores en muestras pequeñas teniendo en cuenta los datos balanceados y no balanceados, es decir, ausencia y presencia de puntos de apalancamiento en los datos en diferentes niveles de heteroscedasticidad. Se evaluó el rendimiento de los estimadores de intervalo para un parámetro β_j construido a partir de una estimación consistente de la desviación estándar del estimador de β ($\hat{\beta}_j$). Los estimadores de la estructura de covarianzas de los estimadores de los parámetros del modelo lineal de regresión utilizan los estimadores HC0, HC2, HC3, HC4 y HC5. También fueron evaluadas inferencias intervalares de estimadores que utilizan esquemas de remuestreo vía bootstrap salvaje. Fueron considerados los métodos bootstrap percentil y bootstrap- t en esquemas simples y dobles, es decir, en esquemas con sólo un nivel bootstrap y en esquemas que consideran un segundo nivel de bootstrap.

Las evaluaciones de las diferentes estrategias de estimación del intervalo tuvieron costo computacional muy elevado. El programa para la realización de la simulación fue redactado utilizando el lenguaje C++, siendo necesario usar computación paralela (OpenMP - *Open Multi-Processing*) y realizar las simulaciones en un supercomputador, siendo estas ejecutadas en simultáneo. Así, las simulaciones pudieron ser realizadas en un tiempo viable. Todas las metodologías bootstrap consideradas en este trabajo fueron reunidas en el paquete `hcci` versión 1.0.0, que está disponible gratuitamente en la web del lenguaje R.

Palabras-clave: bootstrap, bootstrap doble, bootstrap percentil, bootstrap- t , estimación intervalar, `hcci`, heteroscedasticidad.

Lista de ilustrações

Figura 1 – Densidades consideradas na geração dos erros do modelo (4.1).	59
Figura 2 – Coberturas dos estimadores intervalares (sem uso de bootstrap) utilizando os quantis da distribuição normal padrão e da distribuição t de Student com $n - 2$ graus de liberdade - $n = 20$, $\lambda \approx 49$, erros normais e desenho não balanceado.	69
Figura 3 – Coberturas dos estimadores intervalares (sem uso de bootstrap) utilizando os quantis da distribuição normal padrão e da distribuição t de Student com $n - 2$ graus de liberdade - $n = 20$, $\lambda \approx 49$, erros t de Student e desenho não balanceado.	69
Figura 4 – Coberturas dos estimadores intervalares (sem uso de bootstrap) utilizando quantis da distribuição normal padrão e da distribuição t de Student com $n - 2$ graus de liberdade - $n = 20$, $\lambda \approx 49$, erros qui-quadrado e desenho não balanceado.	70
Figura 5 – Coberturas dos estimadores intervalares (sem uso de bootstrap) utilizando quantis da distribuição normal padrão e da distribuição t de Student com $n - 2$ graus de liberdade - $n = 20$, $\lambda \approx 49$, erros Weibull e desenho não balanceado.	70
Figura 6 – Amplitudes das estimativas intervalares via bootstrap- t duplo com $n = 20$, erros normais, $\lambda \approx 49$ e desenho não balanceado.	89
Figura 7 – Amplitudes das estimativas intervalares bootstrap- t duplo com $n = 20$, erros $t_{(3)}$, $\lambda \approx 49$ e desenho não balanceado.	90
Figura 8 – Amplitudes das estimativas intervalares bootstrap- t duplo com $n = 20$, erros $\chi^2_{(2)}$, $\lambda \approx 49$ e desenho não balanceado.	90
Figura 9 – Amplitudes das estimativas intervalares bootstrap- t duplo com $n = 20$, erros Weibull, $\lambda \approx 49$ e desenho não balanceado.	91
Figura 10 – Amplitudes das estimativas intervalares bootstrap- t duplo com $n = 60$, erros normais, $\lambda \approx 49$ e desenho não balanceado.	91

Figura 11 – Amplitudes das estimativas intervalares bootstrap- t duplo com $n = 60$, erros $t_{(3)}$, $\lambda \approx 49$ e desenho não balanceado.	92
Figura 12 – Amplitudes das estimativas intervalares bootstrap- t duplo com $n = 60$, erros $\chi^2_{(2)}$, $\lambda \approx 49$ e desenho não balanceado.	92
Figura 13 – Amplitudes das estimativas intervalares bootstrap- t duplo com $n = 60$, erros Weibull, $\lambda \approx 49$ e desenho não balanceado.	93
Figura 14 – Renda per capita e despesas per capita em escolas públicas.	100
Figura 15 – Renda per capita e despesas per capita em escolas públicas.	103

Lista de tabelas

Tabela 1	– Medida de máxima alavancagem e limiares para detecção de pontos de alta alavancagem.	60
Tabela 2	– Número de amostras e erros de acurácia de estimativas intervalares para os métodos bootstrap percentil duplo e bootstrap- t duplo para diferentes níveis de confiança.	60
Tabela 3	– Percentuais de cobertura dos intervalos HC sem uso de esquemas bootstrap e intervalos bootstrap percentil simples e duplo para erros normais - nível nominal de 90%.	62
Tabela 4	– Percentuais de cobertura dos intervalos HC sem uso de esquemas bootstrap e intervalos bootstrap percentil simples e duplo para erros normais - nível nominal de 95%.	63
Tabela 5	– Percentuais de cobertura dos intervalos HC sem uso de esquemas bootstrap e intervalos bootstrap percentil simples e duplo para erros normais - nível nominal de 99%.	63
Tabela 6	– Percentuais de cobertura dos intervalos HC sem uso de esquemas bootstrap e intervalos bootstrap percentil simples e duplo para erros $t_{(3)}$ - nível nominal de 90%.	64
Tabela 7	– Percentuais de cobertura dos intervalos HC sem uso de esquemas bootstrap e intervalos bootstrap percentil simples e duplo para erros $t_{(3)}$ - nível nominal de 95%.	64
Tabela 8	– Percentuais de cobertura dos intervalos HC sem uso de esquemas bootstrap e intervalos bootstrap percentil simples e duplo para erros $t_{(3)}$ - nível nominal de 99%.	65
Tabela 9	– Percentuais de cobertura dos intervalos HC sem uso de esquemas bootstrap e intervalos bootstrap percentil simples e duplo para erros $\chi^2_{(2)}$ - nível nominal de 90%.	65

Tabela 10 – Percentuais de cobertura dos intervalos HC sem uso de esquemas bootstrap e intervalos bootstrap percentil simples e duplo para erros $\chi^2_{(2)}$ - nível nominal de 95%.	66
Tabela 11 – Percentuais de cobertura dos intervalos HC sem uso de esquemas bootstrap e intervalos bootstrap percentil simples e duplo para erros $\chi^2_{(2)}$ - nível nominal de 99%.	66
Tabela 12 – Percentuais de cobertura dos intervalos HC sem uso de esquemas bootstrap e intervalos bootstrap percentil simples e duplo para erros Weibull(2,3) - nível nominal de 90%.	67
Tabela 13 – Percentuais de cobertura dos intervalos HC sem uso de esquemas bootstrap e intervalos bootstrap percentil simples e duplo para erros Weibull(2,3) - nível nominal de 95%.	67
Tabela 14 – Percentuais de cobertura dos intervalos HC sem uso de esquemas bootstrap e intervalos bootstrap percentil simples e duplo para erros Weibull(2,3) - nível nominal de 99%.	68
Tabela 15 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos balanceados verificando o ajuste da distribuição normal padrão à amostra $b_m, m = 1, 2, \dots, 10000$	75
Tabela 16 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos balanceados verificando o ajuste da distribuição t de Student com $n - 2$ graus de liberdade à amostra $b_m, m = 1, 2, \dots, 10000$	75
Tabela 17 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos não balanceados verificando o ajuste da distribuição normal padrão à amostra $b_m, m = 1, 2, \dots, 10000$	76
Tabela 18 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos não balanceados verificando o ajuste da distribuição t de Student com $n - 2$ graus de liberdade à amostra $b_m, m = 1, 2, \dots, 10000$	76
Tabela 19 – Percentuais de cobertura das estimativas intervalares bootstrap- t simples e bootstrap- t duplo com erros normais - nível nominal de 90%.	77
Tabela 20 – Percentuais de cobertura das estimativas intervalares bootstrap- t simples e bootstrap- t duplo com erros normais - nível nominal de 95%.	78
Tabela 21 – Percentuais de cobertura das estimativas intervalares bootstrap- t simples e bootstrap- t duplo com erros normais - nível nominal de 99%.	79

Tabela 22 – Percentuais de cobertura das estimativas intervalares bootstrap- t simples e bootstrap- t duplo com erros t de Student - nível nominal de 90%.	80
Tabela 23 – Percentuais de cobertura das estimativas intervalares bootstrap- t simples e bootstrap- t duplo com erros t de Student - nível nominal de 95%.	81
Tabela 24 – Percentuais de cobertura das estimativas intervalares bootstrap- t simples e bootstrap- t duplo com erros t de Student - nível nominal de 99%.	82
Tabela 25 – Percentuais de cobertura das estimativas intervalares bootstrap- t simples e bootstrap- t duplo com erros qui-quadrado - nível nominal de 90%.	83
Tabela 26 – Percentuais de cobertura das estimativas intervalares bootstrap- t simples e bootstrap- t duplo com erros qui-quadrado - nível nominal de 95%.	84
Tabela 27 – Percentuais de cobertura das estimativas intervalares bootstrap- t simples e bootstrap- t duplo com erros qui-quadrado - nível nominal de 99%.	85
Tabela 28 – Percentuais de coberturas das estimativas intervalares bootstrap- t simples e bootstrap- t duplo com erros Weibull - nível nominal de 90%.	86
Tabela 29 – Percentuais de cobertura das estimativas intervalares bootstrap- t simples e bootstrap- t duplo com erros Weibull - nível nominal de 95%.	87
Tabela 30 – Percentuais de cobertura das estimativas intervalares bootstrap- t simples e bootstrap- t duplo com erros Weibull - nível nominal de 99%.	88
Tabela 31 – Dados de gastos per capita em escolas públicas e renda per capita por estado em 1979 nos Estados Unidos.	99
Tabela 32 – Estimativas intervalares obtidas por bootstrap percentil e bootstrap- t em esquemas bootstrap simples e duplo para $J = 1000$ e $K = 500$ fixados.	105
Tabela 33 – Estimativas intervalares obtidas por bootstrap percentil e bootstrap- t em esquemas bootstrap simples e duplo com $J = 2199$ e $K = 88$ obtidos por minimização da função M_2 proposta por Booth e Hall (1994).	106

Tabela 34 – Estimativas intervalares obtidas por bootstrap percentil e bootstrap- t em esquemas bootstrap simples e duplo para $J = 1000$ e $K = 500$ fixados.	107
Tabela 35 – Estimativas intervalares obtidas por bootstrap percentil e bootstrap- t em esquemas bootstrap simples e duplo com $J = 2199$ e $K = 88$ obtidos por minimização da função M_2 proposta por Booth e Hall (1994).	108
Tabela 36 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos balanceados verificando o ajuste da distribuição normal padrão à amostra aleatória $b_m, m = 1, 2, \dots, 10000$	114
Tabela 37 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos balanceados verificando o ajuste da distribuição normal padrão à amostra aleatória $b_m, m = 1, 2, \dots, 10000$	115
Tabela 38 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos não balanceados verificando o ajuste da distribuição normal padrão à amostra aleatória $b_m, m = 1, 2, \dots, 10000$	115
Tabela 39 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos não balanceados verificando o ajuste da distribuição $t_{(n-2)}$ à amostra aleatória $b_m, m = 1, 2, \dots, 10000$	116
Tabela 40 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos balanceados verificando o ajuste da distribuição normal padrão à amostra aleatória $b_m, m = 1, 2, \dots, 10000$	116
Tabela 41 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos balanceados verificando o ajuste da distribuição $t_{(n-2)}$ à amostra aleatória $b_m, m = 1, 2, \dots, 10000$	117
Tabela 42 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos não balanceados verificando o ajuste da distribuição normal padrão à amostra aleatória $b_m, m = 1, 2, \dots, 10000$	117
Tabela 43 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos não balanceados verificando o ajuste da distribuição $t_{(n-2)}$ à amostra aleatória $b_m, m = 1, 2, \dots, 10000$	118
Tabela 44 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos balanceados verificando o ajuste da distribuição normal padrão à amostra aleatória $b_m, m = 1, 2, \dots, 10000$	118
Tabela 45 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos balanceados verificando o ajuste da distribuição $t_{(n-2)}$ à amostra aleatória $b_m, m = 1, 2, \dots, 10000$	119

Tabela 46 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos não balanceados verificando o ajuste da distribuição normal padrão à amostra aleatória $b_m, m = 1, 2, \dots, 10000$	119
Tabela 47 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos não balanceados verificando o ajuste da distribuição $t_{(n-2)}$ à amostra aleatória $b_m, m = 1, 2, \dots, 10000$	120
Tabela 48 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos balanceados verificando o ajuste da distribuição normal padrão à amostra aleatória $b_m, m = 1, 2, \dots, 10000$	120
Tabela 49 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos balanceados verificando o ajuste da distribuição $t_{(n-2)}$ à amostra aleatória $b_m, m = 1, 2, \dots, 10000$	121
Tabela 50 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos não balanceados verificando o ajuste da distribuição normal padrão à amostra aleatória $b_m, m = 1, 2, \dots, 10000$	121
Tabela 51 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos não balanceados verificando o ajuste da distribuição $t_{(n-2)}$ à amostra aleatória $b_m, m = 1, 2, \dots, 10000$	122

Sumário

1	Introdução e Suporte Computacional	17
1.1	Introdução	17
1.2	Organização da Dissertação	18
1.3	Plataforma Computacional	19
1.3.1	Linguagem C++	20
1.3.2	GSL - <i>GNU Scientific Library</i>	21
1.3.3	Biblioteca Armadillo	21
1.3.4	OpenMP	22
1.3.5	Vim - Vi Improved	26
1.3.6	Hardwares utilizados	26
1.3.7	Linguagem R	28
1.3.8	L ^A T _E X	29
2	Modelo e Estimadores	31
2.1	Introdução	31
2.2	Modelagem de Regressão	31
2.3	Modelos Lineares Heteroscedásticos de Regressão	33
3	Intervalos de Confiança Bootstrap e Bootstrap Duplo	39
3.1	Introdução	39
3.2	Intervalos de Confiança Paramétricos	40
3.3	Intervalos de Confiança Aproximados	41
3.3.1	Intervalos Normais Aproximados	41
3.3.2	Intervalo Bootstrap Studentizado	42
3.3.3	Intervalo Bootstrap Percentil	43
3.3.4	Intervalo Bootstrap Duplo Percentil	43
3.3.5	Intervalo Bootstrap Duplo Studentizado	46
3.4	Algoritmos para estimativas intervalares em modelos lineares com heteroscedasticidade de forma desconhecida	51
3.4.1	Bootstrap Percentil	52
3.4.2	Bootstrap Studentizado	52

3.4.3	Bootstrap Duplo Percentil	53
3.4.4	Bootstrap Duplo Studentizado	54
3.5	Estimação do número de amostras bootstrap	55
4	Resultados Numéricos	57
5	Aplicação	94
5.1	Introdução	94
5.2	Estatísticas quasi- t e quasi- F	94
5.3	Pacote <code>hcci</code>	96
5.3.1	Função HC	96
5.3.2	Funções QT e QF	97
5.3.3	Funções Pboot e Tboot	97
5.4	Aplicações empíricas	98
6	Considerações Finais	109
	Referências	111
APÊNDICE A	Estatísticas Cramér-von Mises (W^*) e Anderson-Darling (A^*) para a amostra aleatória $b_m = b_1, b_2, \dots, b_m$	114
A.1	Estimativa intervalar HC0	114
A.1.1	Desenho balanceado	114
A.1.2	Desenho não balanceado	115
A.2	Intervalo HC2	116
A.2.1	Desenho balanceado	116
A.2.2	Desenho não balanceado	117
A.3	Intervalo HC3	118
A.3.1	Desenho balanceado	118
A.3.2	Desenho não balanceado	119
A.4	Intervalo HC5	120
A.4.1	Desenho balanceado	120
A.4.2	Desenho não balanceado	121
APÊNDICE B	Programa - Avaliação das estimativas intervalares . .	123
B.1	Código C++	124
APÊNDICE C	Programa - Funções em R para o cálculo das estima- tivas intervalares bootstrap simples e duplo	188
C.1	Função HC	188
C.2	Função Pboot	189
C.3	Função Tboot	191

Introdução e Suporte Computacional

1.1 Introdução

Muitos problemas práticos podem ser resolvidos empregando-se análise de regressão. Várias modelagens empregam modelos lineares homoscedásticos, ou seja, a variância dos erros é assumida ser constante para todas as observações ($\text{var}(\varepsilon_i) = \sigma^2$, $i = 1, 2, \dots, n$). Frequentemente ocorre que a suposição de homoscedasticidade não é apropriada. Nesses problemas não é razoável supor constância na variância dos erros, ou seja, os erros são heteroscedásticos e tipicamente essa heteroscedasticidade é de forma desconhecida. A estratégia que é mais comumente utilizada para estimar os parâmetros que indexam o modelo linear com heteroscedasticidade de forma desconhecida é utilizar o método de mínimos quadrados ordinários. Contudo, dado que os erros são heteroscedásticos e não temos conhecimento sobre sua distribuição há dificuldades na obtenção de estimativas intervalares e na realização de testes de hipóteses para os parâmetros que indexam o modelo linear de regressão.

Quando os erros são heteroscedásticos, temos que a estrutura de covariância do estimador $\hat{\beta}$ passa a ser dada por $\Psi_{\hat{\beta}} = (X'X)^{-1}X'\Omega X(X'X)^{-1}$, em que $\Omega = \text{diag}\{\sigma_1^2, \dots, \sigma_n^2\}$. [White \(1980\)](#) propôs um estimador de $\Psi_{\hat{\beta}}$ que é consistente tanto sob homoscedasticidade quanto sob heteroscedasticidade de forma desconhecida. Esse estimador é conhecido na literatura como estimador HC0. Ele pode ser muito viesado em amostras de tamanhos pequeno a moderado. A partir de estudos de [Horn, Horn e Duncan \(1975\)](#), [MacKinnon e White \(1985\)](#) propuseram o estimador HC2, que é não viesado sob homoscedasticidade. O estimador HC3 foi proposto por [Davidson e MacKinnon \(1993\)](#). Uma sequência de estimadores HC0 corrigido por viés foi proposta por [Cribari-Neto, Ferrari e Cordeiro \(2000\)](#). Esses resultados foram posteriormente estendidos por [Cribari-Neto e Galvão \(2003\)](#). [Cribari-Neto \(2004\)](#) propôs um estimador alternativo denominado de HC4. Uma variante desse estimador foi proposta por [Cribari-Neto, Souza e Vasconcellos \(2007\)](#),

[Errata: v. 37, n. 20, p. 3329–3330, 2008](#)) e foi denominada de estimador HC5.

Utilizando um dos estimadores mencionados acima (HC0, HC2, HC3, HC4 e HC5) é possível construir intervalos de confiança e testar hipóteses sobre o vetor de parâmetros β que indexam o modelo linear de regressão sob heteroscedasticidade de forma desconhecida. [Cribari-Neto e Lima \(2009\)](#) avaliaram estimadores intervalares em modelos lineares com heteroscedasticidade de forma desconhecida. Os autores avaliaram, via simulação de Monte Carlo, estimadores intervalares consistentes utilizando esquemas de bootstrap simples (esquemas com um nível de bootstrap), sendo eles os métodos bootstrap- t e bootstrap percentil, sob diferentes níveis de heteroscedasticidade e tamanhos de amostras utilizando a distribuição normal para os erros.

O presente estudo avalia diferentes estimativas intervalares em modelos lineares com heteroscedasticidade de forma desconhecida em esquemas sem bootstrap, com bootstrap simples e com bootstrap duplo. Foram considerados esquemas de bootstrap selvagem proposto por [Wu \(1986\)](#). Através de simulações de Monte Carlo foi possível avaliar os desempenhos de intervalos de confiança construídos utilizando os estimadores HC0, HC2, HC3, HC4 e HC5 sem esquemas de bootstrap utilizando quantis obtidos da distribuição normal padrão e quantis obtidos de uma distribuição t de Student ($t_{(n-p)}$). Também foram avaliados os métodos bootstrap- t e bootstrap percentil em esquemas simples e duplo. Utilizamos diferentes níveis de heteroscedasticidade e consideramos desenhos balanceados e não balanceados, ou seja, foram considerados esquemas sem nenhum ponto de alavanca e esquemas com presença de pontos de alta alavancagem. Também foram consideradas diferentes distribuições para os erros do modelo linear com heteroscedasticidade de forma desconhecida.

O programa usado para a realização das simulações foi escrito na linguagem de programação C++. Mesmo com toda performance computacional de uma linguagem compilada, houve a necessidade de utilizar computação paralela através de paralelização multicore em um supercomputador. Foi necessário submeter varias simulações para serem executadas simultaneamente e de forma paralela utilizando todos os núcleos do processador dos nós em que cada simulação foi executada. As metodologias bootstrap para estimação intervalar em modelos lineares com heteroscedasticidade de forma desconhecida foram reunidas e escritas na linguagem R, dando origem ao pacote `hcci` versão 1.0.0.

1.2 Organização da Dissertação

Esta dissertação é composta por seis capítulos. O primeiro capítulo tem início com esta introdução, em que é apresentado ao leitor uma síntese dos problemas que serão abordados nesse trabalho. Este capítulo traz também uma análise sobre os softwares,

linguagens e bibliotecas utilizadas durante a elaboração desse estudo.

O segundo capítulo introduz o modelo de regressão linear com heteroscedasticidade de forma desconhecida e apresenta um estimador ($\hat{\beta}$) para os parâmetros que indexam o modelo. São ainda apresentados estimadores consistentes para estrutura de covariância desse estimador ($\Psi_{\hat{\beta}}$).

O terceiro capítulo discute brevemente intervalos de confiança e apresenta as metodologias de estimação intervalar que utilizam esquemas de bootstrap simples e duplo. Também são apresentados algoritmos para estimação intervalar em modelos lineares com heteroscedasticidade de forma desconhecida utilizando bootstrap selvagem proposto por [Wu \(1986\)](#).

O quarto capítulo apresenta avaliações numéricas via simulações de Monte Carlo sobre os desempenhos em pequenas amostras de estimadores intervalares para um parâmetro que indexa o modelo linear de regressão com heteroscedasticidade de forma desconhecida considerado nesse trabalho. Foram avaliados os desempenhos de estimadores intervalares sem uso de bootstrap, utilizando apenas um nível de bootstrap e de estimadores que fazem uso de um segundo nível de bootstrap. Os estimadores intervalares que se baseiam em uma transformação pivotal utilizaram os erros-padrão consistentes apresentados no Capítulo 2.

O quinto capítulo apresenta uma aplicação com dados reais utilizando os estimadores intervalares apresentados no Capítulo 3. Na aplicação utilizamos esquemas de bootstrap simples e duplo. Também como resultado da aplicação é construída uma biblioteca para linguagem R com as metodologias bootstrap consideradas nesse trabalho. Detalhes sobre a biblioteca também são apresentados nesse capítulo. Por fim, o sexto capítulo reúne as principais conclusões deste trabalho.

1.3 Plataforma Computacional

O presente estudo utilizou linguagens e bibliotecas matemáticas capazes de atender às exigências numéricas das metodologias utilizadas. O trabalho fez uso da linguagem de programação C++. Essa linguagem foi de vital importância devido a sua grande eficiência computacional para trabalhar com computação numérica e também por sua integração com bibliotecas, como **Armadillo** (C++ *linear algebra library*) e a **GSL** (*GNU Scientific Library*). A linguagem C++ é interessante por dar suporte ao padrão OpenMP, que também é suportado por linguagens como C e FORTRAN. Utilizamos também o sistema de preparação de documentos e composição tipográfica **L^AT_EX**.

1.3.1 Linguagem C++

C++ é uma linguagem multi-paradigma de propósito geral que foi concebida em 1983 no Bell Labs. O criador da linguagem foi o cientista da computação dinamarquês Bjarne Stroustrup, que atualmente é professor catedrático da Universidade do Texas A&M. C++ é uma linguagem de programação de médio nível, pois reúne características de linguagens de alto e baixo nível. Esse é um dos principais motivos para C++ ser uma linguagem bastante flexível.

Em sua fase inicial de desenvolvimento, a linguagem C++ era conhecida como “novo C”, “C84” ou ainda “C com classes”. O nome “C++” foi utilizado pela primeira vez em dezembro de 1983 por Rick Mascitti. Ele contém uma referência ao operador de incremento ++ e significa um acréscimo ou evolução na linguagem C.

Por muito tempo, C++ foi encarado como um superconjunto de C. Entretanto, em 1999 o padrão ISO para a linguagem C tornou as duas linguagens ainda mais diferentes. Dessa forma, muitas empresas que desenvolviam compiladores para C++ deixaram de dar suporte à linguagem C com o novo padrão ISO.

A biblioteca padrão de C++ incorpora a biblioteca padrão de C com algumas poucas modificações, que foram feitas para acomodar novas funcionalidades incorporadas na linguagem.

Programas escritos em C, em geral, podem ser compilados usando compiladores C++, mas o contrário não é verdadeiro. O compilador utilizado nesse estudo foi o g++, que é distribuído pela Free Software Foundation (FSF) sob os termos da licença GNU GPL. Esse compilador está disponível para os sistemas operativos Unix e Linux, bem como para sistemas operativos derivados como o Mac OS X.

As principais características da linguagem C++ são:

- C++ é uma linguagem de propósito geral;
- C++ pode ser utilizado mesmo sem um ambiente de desenvolvimento sofisticado;
- C++ é multi-paradigma, tornando possível, por exemplo, que se programe de forma procedural e também com paradigma de orientação a objetos, podendo também o usuário misturar os dois paradigmas;
- C++ foi desenvolvido para ser o máximo possível compatível com C;
- C++ é uma linguagem estaticamente tipada, sendo tão eficiente e portátil quanto a linguagem C.

1.3.2 GSL - *GNU Scientific Library*

A **GSL** (*GNU Scientific Library*) é uma biblioteca numérica para as linguagens de programação C e C++. Esta biblioteca é distribuída sob a licença GPL (*General Public License*) e está disponível em <http://www.gnu.org/software/gsl/>.

A biblioteca oferece uma grande variedade de rotinas matemáticas, tais como geradores de números pseudo-aleatórios, integração de Monte Carlo, suporte **BLAS** (*Basic Linear Algebra Subprograms*), entre outras. Ao todo, são mais de 1000 funções disponíveis para uso na **GSL**. Algumas das áreas abrangidas pela **GSL** são:

- Estatística;
- Sequências pseudo-aleatórias de números;
- Álgebra linear;
- Equações diferenciais;
- Interpolação.

Existem projetos que reimplementam a biblioteca **GSL** para outras linguagens de programação, entre elas **HASKELL**, **PYTHON**, **LUA** e **JAVA**. Um desses projetos é a biblioteca **GSL Shell**, que fornece uma interface interativa, em linhas de comandos, dando ao usuário fácil acesso a uma coleção de algoritmos numéricos e funções com base na **GSL**. **GSL Shell** é capaz de trabalhar com matrizes e vetores para executar operações de álgebra linear. Esta biblioteca faz uso de **LUA JIT** e utiliza o compilador **LuaJIT2**. Esse trabalho fez uso da biblioteca **GSL** versão 1.15.

1.3.3 Biblioteca Armadillo

Armadillo é uma biblioteca de C++ desenvolvida para se trabalhar com álgebra linear e visa atingir um bom equilíbrio entre velocidade e facilidade de uso. Sua sintaxe é deliberadamente semelhante à do software **MATLAB**. Apesar de ser uma biblioteca matricial, números inteiros, números de ponto flutuante e números complexos também são suportados. A biblioteca também possui um subconjunto de funções trigonométricas e estatísticas.

Armadillo foi desenvolvido e é mantido pelo NICTA (National Information Communications Technology Australia), em especial por Conrad Sanderson. Atualmente **Armadillo** é utilizado pela NASA, Boeing, Siemens, MIT e CMU.

A versão da biblioteca utilizada nesse trabalho é **Armadillo-3.900.6**, obtida em <http://arma.sourceforge.net/>. **Armadillo** está disponível para download para um

vasto número de plataformas, incluindo Unix, Linux, Mac OS X, entre outras. A biblioteca pode ser distribuída e modificada sob os termos da licença MPL (Mozilla Public License 2.0), sendo assim open-source.

Armadillo se integra com a biblioteca de alto nível **LAPACK** (<http://www.netlib.org/lapack/>), contudo a integração com essa biblioteca é opcional. O usuário pode escolher uma implementação de alta performance de **LAPACK**, como o multi-treaded Intel MKL, AMD ACML. Também pode ser feita a integração da biblioteca **Armadillo** com implementações otimizadas da biblioteca **BLAS**, como, por exemplo, a biblioteca **OpenBLAS**. Esse trabalho fez uso das bibliotecas **OpenBLAS** (<http://xianyi.github.io/OpenBLAS/>) e **MKL** (<http://software.intel.com/en-us/intel-mkl>).

1.3.4 OpenMP

Ao longo dos anos, o desenvolvimento científico vem exigindo das simulações numéricas resultados mais confiáveis. Para acompanhar esse avanço, a computação científica vem superando as barreiras que surgem em virtude da limitação física dos computadores. Entre tais limitações podemos citar a demanda por processamento numérico, armazenamento de dados, visualização, entre outras.

Nesse contexto, a computação de alto desempenho (*High Performance Computing* - HPC) tem se apresentado como uma importante frente de pesquisa nos últimos anos. O termo pode ser definido como qualquer conjunto de técnicas que visam otimizar ou viabilizar o processamento de experimentos numéricos. Podemos citar como uma dessas técnicas o uso de processamento paralelo em clusters e supercomputadores.

Entre as técnicas de processamento paralelo mais difundidas atualmente, podemos destacar as técnicas de memória distribuída e as de memória compartilhada. Técnicas de memória distribuída são aquelas que se aplicam aos computadores que possuem arquitetura de memória distribuída, ou seja, máquinas com vários processadores que possuem seu próprio recurso de memória e são interconectados por uma rede local. O padrão MPI (*Message Passing Interface*) é o mais utilizado atualmente nesse contexto e seu funcionamento se dá basicamente através da troca de dados entre os processadores.

Por sua vez, as técnicas de memória compartilhada são utilizadas em ambientes que possuem vários núcleos que compartilham o mesmo recurso de memória. Nesse contexto, a utilização do padrão OpenMP (*Open Multi-Processing*) tem crescido bastante nos últimos anos. Suas funcionalidades facilitam o desenvolvimento de aplicações em memória compartilhada.

Atualmente existem bibliotecas que dão suporte ao encadeamento de execução (*threads*). Entenda-se por (*thread*) a forma de um processo dividir a si mesmo em duas

ou mais tarefas que podem ser executadas de forma concorrente. Assim, por exemplo, uma *thread* permite que o usuário de um programa utilize uma funcionalidade do sistema operacional enquanto outras linhas de execuções realizam cálculos e operações.

Em hardwares que possuem apenas uma CPU (*Central Processing Unit*), cada *thread* é processada de forma aparentemente simultânea, pois a mudança de uma *thread* para outra é feita de forma tão rápida que para o usuário isso aparenta ocorrer paralelamente. Em hardwares *multi-cores*, as *threads* são realizadas realmente de forma simultânea. Dito isso, os sistemas que suportam apenas uma única *thread* são chamados de *monothread* enquanto sistemas que suportam múltiplas *threads* são chamados de *multithread*.

OpenMP é uma implementação de *multithreading*, um método de paralelização no qual *master threads* (séries de instruções executadas consecutivamente) bifurcam-se em um número específico de *threads* escravos e uma tarefa é dividida entre eles.

O padrão OpenMP é desenvolvido e mantido pelo grupo OpenMP ARB (*Architecture Review Board*), que é formado pelos maiores fabricantes de softwares do mundo, tais como SUN Microsystems, SGI, IBM, Intel, entre outros. O OpenMP não é uma linguagem de programação. Ele representa um padrão que define como os compiladores devem gerar códigos paralelos através da incorporação, nos programas sequenciais, de diretivas que indicam como o trabalho será dividido entre os processadores. Dessa forma, muitas aplicações podem tirar proveito desse padrão com pequenas modificações no código.

As funcionalidades do OpenMP podem atualmente ser utilizadas nas linguagens FORTRAN 77, FORTRAN 90, C e C++. A primeira versão do padrão OpenMP foi disponibilizada para uso no final de 1997. Atualmente o OpenMP encontra-se na versão 2.5. Algumas vantagens do OpenMP são:

- Requer poucas alterações no código sequencial;
- Fácil compreensão e uso das diretivas;
- Possibilita o ajuste dinâmico do número de *threads*;
- Possui uma estrutura robusta para suporte a programação paralela;
- Suportado por vários compiladores, entre eles o `gcc`, que incluiu suporte ao OpenMP desde sua versão 4.2. A *flag* `fopenmp` é responsável por instruir o compilador gcc a utilizar o padrão OpenMP;
- OpenMP é de fácil manutenção e de fácil depuração.

Quando um código que contém diretivas do padrão OpenMP é compilado por um compilador que não fornece suporte ao OpenMP ou quando o compilador utilizado fornece

o suporte mas a opção de compilação que habilita o seu uso não é utilizada, o compilador ignora as diretivas e compila o programa de forma sequencial. O arquivo cabeçalho de C/C++ que fornece as funções úteis para trabalhar com OpenMP é o arquivo `omp.h`. Para ilustrar o uso do OpenMP considere os dois exemplos abaixo.

Exemplo 1:

```
1 #include<stdio.h> /* Cabecalho de rotinas padrao de C */
2 #include<stdlib.h> /* Cabecalho com funcoes de entrada e saida */
3 #include<math.h> /* Cabecalho para uso de funcoes matematicas */
4 #include<omp.h> /* Cabecalho para uso de OpenMP */
5
6 int main(void){
7     double start = omp_get_wtime();
8     const int N = 90000;
9     int i, k, vetor;
10    float *v;
11    v = (float *) malloc(N*sizeof(float));
12
13    for (i = 0; i < N; i++){
14        v[i] = sin(i*cos(i));
15        for(k = 0; k<700; k++){
16            v[k] = sin(k*cos(k));
17        }
18    }
19
20    double end = omp_get_wtime();
21    printf("Tempo = %f\n", end-start);
22    free(v);
23    return 0;
24 }
```

O código abaixo é excessivamente parecido com o código apresentado logo acima. Note que o exemplo que segue acrescenta a linha de código 14 que informa ao compilador que o *loop* será executado nos múltiplos cores do processador em que o executável desse programa será executado. Assim, o trecho de código contido no bloco `for` será paralelizado.

Exemplo 2:

```

1  #include<stdio.h> /* Cabecalho de rotinas padrao de C */
2  #include<stdlib.h> /* Cabecalho com funcoes de entrada e saida */
3  #include<math.h> /* Cabecalho para uso de funcoes matematicas */
4  #include<omp.h> /* Cabecalho para uso de OpenMP */
5
6  int main(void){
7      double start = omp_get_wtime();
8      const int N = 90000;
9      int i, k, vetor;
10     float *v;
11     v = (float *) malloc(N*sizeof(float));
12
13     #pragma omp parallel for
14     for (i = 0; i < N; i++){
15         v[i] = sin(i*cos(i));
16         for(k = 0; k<700; k++){
17             v[k] = sin(k*cos(k));
18         }
19     }
20
21     double end = omp_get_wtime();
22     printf("Tempo = %f\n", end-start);
23     free(v);
24     return 0;
25 }
```

Uma forma de medir o ganho pela computação em paralelo é usando o fator *speed up*, que representa o ganho de velocidade de processamento de uma aplicação quando executada com n processadores. Quanto maior o *speed up*, mais rápido é executado o código paralelo. O *speed up* é dado por

$$S_n = \frac{T_s}{T_n},$$

em que T_s é o tempo de computação serial e T_n é o tempo da computação em paralelo do programa. Observemos que os exemplos acima são excessivamente parecidos. O que difere um do outro é a presença da diretiva *parallel for* no segundo exemplo. Em um computador com processador Intel(R) Core(TM) i3 CPU, M 330 e 2.13GHz com 4 núcleos o tempo de execução do código serial usando o compilador gcc versão 4.8.1 foi de 34.159207 segundos,

já utilizando a diretiva OpenMP foi de 9.977066 segundos, correspondendo a um *speed up* de 3.423773. Ou seja, o código paralelizado é aproximadamente 3.4 vezes mais rápido que o código serial. Dessa forma, com pouco esforço, uma redução bastante considerável no tempo de execução foi obtida. É importante entender que nem sempre paralelizar aumenta a performance computacional. Esse fato é garantido pela Lei de Amdahl ([Amdahl \(1967\)](#)) que diz que sempre existe um limite ao qual a capacidade de ganho pela paralelização estará sujeita. Esse fato, em geral, se devem a fatores como entrada e saída, dependência entre os dados bem como outros fatores intrínsecos à aplicação e à técnica de programação paralela utilizada.

1.3.5 Vim - Vi Improved

Para a implementação do código em C++ foi utilizado o editor Vim. Vim é um poderoso editor de texto para programação. Este editor é bastante flexível, pois permite ao programador utilizar todos os comandos do `shell script` do Linux sem que seja necessário sair do editor de texto. O Vim é uma derivação melhorada do editor Vi, sendo este um editor de texto para sistemas operacionais Unix e Linux. Outros sistemas operacionais podem fazer uso de Vim que deverá ser obtido em <http://www.vim.org/>.

Vi foi criado por Bill Joy em 1976 para o sistema operacional BSD. Em 1991, foi lançado o Vim (**Vi IM**proved ou Vi Melhorado). Ele está presente em quase todas as distribuições Linux e em clusters em diversas localidades do mundo.

A maioria dos supercomputadores no mundo dispõe de um sistema operacional Unix/Linux. Em geral, o acesso a esses clusters não se dá por meio de interfaces gráficas. O usuário se depara com um terminal do sistema operacional e tudo que precisa ser feito deverá ser realizado via linhas de comandos. Como se faz necessário utilizar no mínimo um editor de texto para programar em alguma linguagem de programação como C/C++ e o Vim pode ser executado no terminal e está presente em quase todos sistemas Unix/Linux, o uso do Vim se torna muito conveniente.

O editor Vim requer uma curva de aprendizado, pois o seu uso não é igual ao de outros editores em que o usuário clica sobre um ícone, escreve o que precisa, faz uso de operações copiar, colar, recortar e salva o que foi editado. Vim apresenta um conjunto de comandos que facilita a vida do usuário o que de início pode parecer complicado.

1.3.6 Hardwares utilizados

A lei que instituiu o II PLANIN (Plano de Informática e Automação), aprovada pelo Congresso Nacional Brasileiro em outubro de 1991, propõe a instalação de um Centro Nacional de Supercomputação (CESUP) para oferecer serviços computacionais avançados aos pesquisadores brasileiros. Esse centro foi instalado na Universidade Federal do Rio

Grande do Sul (UFRGS).

O Brasil dispõe de alguns Centros Nacionais de Processamento de Alto Desempenho (CENAPAD), entre eles o CENAPAD UFRGS, CENAPAD UFRJ, CENAPAD UNICAMP, CENAPAD UFPE, entre outros. Qualquer pesquisador com um projeto de pesquisa poderá desfrutar das capacidades computacionais desses CENAPADs. Em geral, o interessado deverá preencher uma documentação informando detalhes sobre o projeto de pesquisa e encaminhar essa documentação à secretaria responsável.

Essa dissertação fez uso dos hardwares disponibilizados pelo Centro Nacional de Supercomputação - CESUP, CENAPD UFRGS. O CESUP possui dois clusters: o cluster Sun Fire, apelidado de Newton, e o cluster SIG Altix, também conhecido como Gauss. As configurações do clusters estão descritas abaixo.

Cluster Sun Fire (Newton):

- 45 nós de processamento;
- 3 nós de gerência;
- 8 GPUs nVIDIA Tesla;
- 1 GPU AMD FireStream;
- 1 switch Voltaire InfiniBand;
- Total de 1296GB de memória RAM;
- Total de 188TB de capacidade de armazenamento, em que 158TB são compartilhado com o cluster SGI Altix Gauss;
- Performance teórica de pico de 12.94 Tflops.

Cluster SGI Altix (Gauss):

- 64 blades de processamento;
- 2 nós de serviço;
- Interconexão InfiniBand;
- Total de 4TB de memória RAM;
- Total de 174TB de capacidade de armazenamento, sendo 158TB compartilhados com o cluster Sun Fire (Newton);

- Performance teórica de pico de 15.97 Tflops.

Essa dissertação fez uso do cluster SGI Altix (Gauss). Cada uma das 64 unidades de processamento do cluster SGI Altix possui 2 processadores dodeca-core (24 núcleos) AMD Opteron trabalhando com uma frequência de 2.3GHz, 128KB de cache L1 por núcleo (dados + instruções), 512KB de cache L2 por núcleo e 12MB de cache L3 por soquete. O Gauss possui controlador de memória DDR3 integrado com suporte a frequências de até 1333MHz e largura de banda de até 42.7GB/s por CPU, totalizando 64GB de RAM por unidade.

Esses hardwares deram maior agilidade as simulações, pois, foi possível enviar várias simulações simultâneas sem que houvesse perda no desempenho do processamento das tarefas submetidas ao cluster. O usuário pode enviar dez simulações (*jobs*) por vez. É possível desconectar-se da máquina se necessário e retornar em um outro momento para observar os resultados das simulações, pois o Gauss e o Newton operam com o PBS (*Product Breakdown Structure*), que gerencia os trabalhos submetidos ao cluster. Dessa forma, o usuário pode submeter suas simulações para processamento através de um sistema de filas, o qual organiza e monitora os *jobs* em execução, otimizando o uso dos recursos disponíveis. Maiores informações sobre os hardwares disponíveis pelo CESUP podem ser encontradas em <http://www.cesup.ufrgs.br/>.

1.3.7 Linguagem R

R é uma linguagem de programação para computação estatística e gráficos. R é uma parte oficial do projeto GNU da Free Software Foundation's. A Fundação R é similar a outras fundações de softwares open-source como o Apache Foundation e a GNOME Foundation. Entre as metas da Fundação R, podem-se destacar o suporte de desenvolvimento contínuo do R, a exploração de novas metodologias, ensino e treinamento de computação estatística, promover reuniões e conferências sobre computação estatística, entre outros.

A linguagem R foi criada originalmente por Ross Ihaka e Robert Gentleman no Departamento de Estatística da Universidade de Auckland, Nova Zelândia em agosto de 1993. Apesar de não ser uma linguagem de propósito geral, R é bastante flexível e pode se comunicar com as linguagens FORTRAN, C e C++. Usuários experientes podem escrever códigos nessas linguagens para manipular diretamente objetos em R.

Um dos grandes motivos da grande popularidade da linguagem R se deve a grande quantidade de pacotes disponíveis para os usuários da linguagem. Atualmente há aproximadamente 5 mil pacotes disponíveis para R com foco em várias áreas. Por ser uma linguagem livre e devemos entender liberdade não apenas o caráter gratuito da linguagem

mas também o fato de seus códigos fontes serem disponíveis e poderem ser alterados por qualquer usuário é o que leva a linguagem R ser muito utilizada. Esse fato também proporciona que metodologias mais novas estejam disponíveis em R mais rapidamente em comparação com linguagens e programas de código fechado.

Nesse trabalho a linguagem R foi utilizada para construção dos resultados apresentados na aplicação e para confecção dos gráficos presentes no texto. Maiores detalhes sobre a linguagem R poderão ser obtidos em <http://cran.r-project.org/>.

1.3.8 L^AT_EX

L^AT_EX é uma linguagem de comandos macros de T_EX e está atualmente na versão L^AT_EX 2_ε. T_EX é um sistema de tipografia científica desenvolvido por Donald E. Knuth que é orientado à produção de textos técnicos e fórmulas matemáticas. A pedido da AMS (*American Mathematical Society*), Donald Knuth desenvolveu uma linguagem de computador para editoração de textos com muitas equações. O trabalho de criação se estendeu de 1977 a 1986, quando o T_EX foi disponibilizado gratuitamente. O T_EX possui aproximadamente 600 comandos que controlam a construção de uma página. Pode-se considerar o T_EX como sendo um compilador para textos científicos que produz documentos de alta qualidade tipográfica.

O T_EX atingiu um estado de desenvolvimento em que Beebe (1990) afirmou:

“Meu trabalho no desenvolvimento do T_EX, METAFONT e as fontes Computer Modern chegou ao final. Eu não irei realizar mudanças futuras, exceto corrigir sérios erros de programação.”

Quase que em paralelo foi desenvolvido por Leslie Lamport o L^AT_EX. Essas macros definem tipos de documentos, tais como livros, artigos, cartas, entre outros. Desde dezembro de 1994, o pacote L^AT_EX está sendo atualizado pela equipe L^AT_EX 3, dirigida por Frank Mittelbach, para incluir algumas melhorias que já vinham sendo solicitadas há muito tempo. A equipe também se preocupa em reunificar todas as versões modificadas que surgiram desde o aparecimento do L^AT_EX 2.09.

O L^AT_EX é um sistema estável, mas com crescimento constante, podendo ser instalado em quase todos os sistemas operacionais existentes. O usuário conta com uma imensa quantidade de pacotes que realizam inúmeras tarefas distintas na edição de textos científicos.

Para manter o texto desse trabalho compatível com as normas da ABNT (Associação Brasileira de Normas Técnicas) foi utilizado o pacote abnT_EX2 1.8.1. O pacote abnT_EX2 trata-se de um conjunto de customizações da classe memoir para elaboração de documentos técnicos e científicos condizentes com as normas da ABNT, especialmente

a ABNT NBR 6022:2003, ABNT NBR 10719:2011, ABNT NBR 14724:2011 e a ABNT NBR 6024:2012. O pacote poderá ser obtido em [<https://code.google.com/p/abntex2/>](https://code.google.com/p/abntex2/).

Modelo e Estimadores

2.1 Introdução

Modelos lineares heteroscedásticos apresentam uma grande aplicabilidade em diversos problemas práticos. A não constância das variâncias dos erros impede que os modelos lineares heteroscedásticos herdem certas propriedades matemáticas dos modelos lineares homoscedásticos. A estrutura usual de covariâncias dos estimadores dos parâmetros que indexam o modelo de regressão linear homoscedástico não é mais válida em casos de heteroscedasticidade. [White \(1980\)](#) propôs um estimador da matriz de covariâncias do estimador de mínimos quadrados ordinários do vetor de parâmetros de regressão que é consistente tanto sob homoscedasticidade quanto sob heteroscedasticidade de forma desconhecida. Notou-se na literatura que o estimador de White é viesado em amostras pequenas e moderadas, sobretudo quando os dados contêm pontos de alavanca. Variantes do estimador de White foram propostos por [MacKinnon e White \(1985\)](#), [Davidson e MacKinnon \(1993\)](#), [Cribari-Neto \(2004\)](#) e [Cribari-Neto, Souza e Vasconcellos \(2007, Errata: v. 37, n. 20, p. 3329–3330, 2008\)](#). Este capítulo apresentará tais estimadores.

2.2 Modelagem de Regressão

Seja $L(\cdot)$ uma função qualquer, tal que, $\forall 0 < u < v$, temos:

- i) $0 = L(0) \leq L(u) \leq L(v)$;
- ii) $0 = L(0) \leq L(-u) \leq L(-v)$.

Qualquer função $L(\cdot)$ que satisfaz as propriedades acima é chamada de função perda. Alguns exemplos de função perda são:

- Função perda quadrática, em que $L(u) = u^2$;
- Função perda absoluta, em que $L(u) = |u|$;
- Perda degrau, em que para algum $\delta > 0$, $L(u) = 0$ se $|u| < \delta$ e 1 caso contrário.

No caso mais simples, queremos prever o comportamento de uma variável de interesse y condicional a uma variável explicativa x . O melhor preditor de y condicional em x é aquele que minimiza a função perda esperada, ou seja, é aquele que resolve $\min \mathbb{E}(L(y - f)|x)$, em que f é um preditor e $\mathbb{E}(L(y - f)|x)$ é a função perda esperada. A função f é função de x e de quantidades fixas e desconhecidas. Em alguns casos, tais quantidades são estimáveis a partir dos dados. Elas são chamadas de parâmetros. Aqui serão agrupadas no vetor β .

Para o caso da função perda quadrática, o melhor preditor de y condicional a x é a média condicional de y dado x . No caso em que L é a função perda absoluta temos que a mediana condicional de y dado x é o melhor preditor para y . Os modelos de regressão, em geral, fazem uso da função perda quadrática. Regressão é qualquer aspecto da distribuição condicional de y em x tomado como função de x .

Modelos lineares de regressão foram uma das primeiras formas de análise regressiva estudadas com rigor matemático. Esses modelos são amplamente utilizados porque são mais fáceis de serem ajustados que os modelos não-lineares. Um outro motivo de sua vasta utilização diz respeito à facilidade de se obter as propriedades estatísticas dos estimadores resultantes.

Em termos mais gerais, modelagem de regressão refere-se à modelagem estocástica que relaciona matematicamente uma variável de interesse y_i a um conjunto de variáveis explicativas, $x_{1i}, x_{2i}, \dots, x_{pi}$, em que, $i = 1, 2, \dots, n$, sendo n a quantidade de dados. A variável y_i também pode ser referenciada como variável resposta, regressando ou variável dependente, enquanto as variáveis $x_{1i}, x_{2i}, \dots, x_{pi}$ podem ser chamadas de variáveis independentes, regressores ou covariáveis. Em forma matricial, y é um vetor $n \times 1$ e X é uma matriz de variáveis explicativas de dimensão $n \times p$, em que p é o número de parâmetros lineares de regressão. Dessa forma, temos que o comportamento de y é afetado por variações na matriz de regressores X e por quantidades desconhecidas presentes em qualquer modelo estocástico denominadas de erros aleatórios, que serão agrupados no vetor ε de dimensão $n \times 1$.

É importante frisar que quando mencionamos modelos lineares de regressão não estamos falando em uma relação linear entre as variáveis envolvidas, mas sim em linearidade nos parâmetros do modelo. Ou seja, a esperança condicional de y , $\mathbb{E}(y|X)$, é uma função linear dos parâmetros do vetor de parâmetros β de dimensão $p \times 1$.

2.3 Modelos Lineares Heteroscedásticos de Regressão

Os parâmetros dos modelos lineares de regressão são tipicamente estimados pelo Método dos Mínimos Quadrados ou Mínimos Quadrados Ordinários (MQO). MQO é o método de estimação mais amplamente utilizado em econometria. Essa metodologia consiste em minimizar a soma das perdas quadráticas e não requer suposição distribucional sobre eles. Observe que em nenhum momento será necessário conhecer a distribuição de y . Consideremos o modelo linear de regressão, dado por

$$y = X\beta + \varepsilon,$$

em que y é um vetor $n \times 1$ de observações de interesse (regressando), X é uma matriz fixa e conhecida de variáveis independentes de dimensão $n \times p$ com posto coluna completo, ou seja, $\text{posto}(X) = p < n$, $\beta = (\beta_1, \dots, \beta_p)'$ é um vetor de dimensão $p \times 1$ de parâmetros lineares e $\varepsilon = (\varepsilon_1, \dots, \varepsilon_n)'$ é um vetor $n \times 1$ de erros aleatórios. A solução de mínimos quadrados é alcançada minimizando $\varepsilon^2 = \varepsilon'\varepsilon = \sum_{i=1}^n \varepsilon_i^2$. Ou seja, $\hat{\beta} = \arg \min_{\beta \in \mathbb{R}^p} \sum_{i=1}^n \varepsilon_i^2$. Substituindo ε por $y - X\beta$ temos que

$$S(\beta) = (y - X\beta)'(y - X\beta) = y'y - y'X\beta - \beta'X'y + \beta'X'X\beta.$$

Derivando com respeito a β e igualando esta derivada a zero, segue que

$$\frac{\partial S}{\partial \beta} = -2X'y + 2X'X\beta = 0 \Rightarrow X'X\hat{\beta} = X'y.$$

Ou seja,

$$\hat{\beta} = (X'X)^{-1}X'y. \quad (2.1)$$

Em se tratando de modelos lineares de regressão, as suposições que tipicamente são feitas são:

S1: O modelo $y = X\beta + \varepsilon$ é, de fato, o modelo verdadeiro;

S2: $\mathbb{E}(\varepsilon_i) = 0$, $i = 1, \dots, n$;

S3: $\mathbb{E}(\varepsilon_i^2) = \text{var}(\varepsilon_i) = \sigma_i^2$, $i = 1, \dots, n$;

S3': $\text{var}(\varepsilon_i) = \sigma^2$, $i = 1, \dots, n$ ($0 < \sigma^2 < \infty$);

S4: $\mathbb{E}(\varepsilon_i \varepsilon_s) = 0$, $\forall i \neq s$;

S5: $\lim_{n \rightarrow \infty} n^{-1}(X'X) = Q$, em que Q é uma matriz positiva-definida.

Sob [S1] e [S2], temos que o estimador $\hat{\beta} = (X'X)^{-1}X'y$ é não viesado para β , ou seja, em média iguala-se ao parâmetro verdadeiro. Sob essas suposições, temos que $\mathbb{E}(y) = X\beta$. Logo, $\mathbb{E}(\hat{\beta}) = (X'X)^{-1}X'\mathbb{E}(y) = (X'X)^{-1}X'X\beta = I_p\beta = \beta$, em que I_p é uma matriz identidade de dimensão $p \times p$. Tal estimador coincide com o estimador de máxima verossimilhança sob normalidade dos erros.

Quando as suposições [S1], [S2], [S3] e [S4] são válidas, a matriz de covariâncias de ε é dada por

$$\Omega = \text{diag}\{\sigma_i^2, \dots, \sigma_n^2\}.$$

Essa matriz se reduz a $\Omega = \sigma^2 I_n$ quando $\sigma_i^2 = \sigma^2 > 0$ com $i = 1, \dots, n$ (vale [S3']), sendo I_n matriz identidade de dimensão $n \times n$. Esse resultado pode ser visto abaixo. Note que

$$\mathbb{E}(\varepsilon\varepsilon') = \begin{bmatrix} \mathbb{E}(\varepsilon_1^2) & \mathbb{E}(\varepsilon_1\varepsilon_2) & \cdots & \mathbb{E}(\varepsilon_1\varepsilon_n) \\ \mathbb{E}(\varepsilon_2\varepsilon_1) & \mathbb{E}(\varepsilon_2^2) & \cdots & \mathbb{E}(\varepsilon_2\varepsilon_n) \\ \vdots & \vdots & \ddots & \vdots \\ \mathbb{E}(\varepsilon_n\varepsilon_1) & \mathbb{E}(\varepsilon_n\varepsilon_2) & \cdots & \mathbb{E}(\varepsilon_n^2) \end{bmatrix}.$$

Usando [S1], [S2], [S3'] e [S4], obtemos

$$\Omega = \mathbb{E}(\varepsilon\varepsilon') = \mathbb{E}(\varepsilon^2) = \sigma^2 I_n.$$

O vetor de parâmetros β pode ser estimado usando o método de mínimos quadrados ordinários; equação (2.1). Sem que se faça suposição sobre a distribuição das respostas, minimiza-se a soma dos quadrados dos erros. É importante destacar também que mesmo quando a suposição [S3] é satisfeita o estimador de mínimos quadrados ordinários continua não-viesado. O estimador $\hat{\beta}$ possui as seguintes propriedades:

- (1) Quando as suposições [S1] e [S2] são válidas, $\hat{\beta}$ é um estimador não-viesado para β . Ou seja, em média o estimador iguala-se ao parâmetro em estimação;
- (2) A estrutura de covariância de $\hat{\beta}$ é $\text{cov}(\hat{\beta}) = \Psi_{\hat{\beta}} = (X'X)^{-1}X'\Omega X(X'X)^{-1}$;
- (3) Quando as suposições [S1], [S2] e [S5] valem, o estimador $\hat{\beta}$ é consistente para β , ou seja, converge em probabilidade para o vetor de parâmetros;
- (4) Sob as suposições [S1], [S2], [S3'] e [S4], vale o Teorema de Gauss-Markov, ou seja, $\hat{\beta}$ possui variância mínima na classe dos estimadores lineares e não-viesados de β ;
- (5) Sob [S1], [S2] e [S3], o estimador $\hat{\beta}$ é assintoticamente gaussiano.

É importante destacar que quando a suposição de homoscedasticidade é violada (i.e., os erros seguem padrão heteroscedástico) $\hat{\beta}$ deixa de ser eficiente, ou seja, não vale

mais o Teorema de Gauss Markov. Contudo, o estimador continua não-viesado, consistente e assintoticamente gaussiano. Dessa forma, $\hat{\beta}$ mantém muitas das propriedades desejáveis de um bom estimador.

Sob homoscedasticidade, ou seja, quando as variâncias dos erros são idênticas (suposição [S3']), temos que a estrutura de covariância de $\hat{\beta}$ é dada por $\text{cov}(\hat{\beta}) = \sigma^2(X'X)^{-1}$. Essa matriz pode ser facilmente estimada substituindo σ^2 pelo estimador $\hat{\sigma}^2 = \hat{\varepsilon}'\hat{\varepsilon}/(n-p)$, em que $\hat{\varepsilon}$ é o vetor de dimensão $n \times 1$ contendo os resíduos de mínimos quadrados. Esses resíduos são dados por $\hat{\varepsilon} = \{I_n - X(X'X)^{-1}X'\}y = (I_n - H)y$. Os erros-padrão fornecidos pela maioria dos softwares estatísticos são as raízes quadradas dos elementos diagonais de $\hat{\sigma}^2(X'X)^{-1}$.

A matriz $H = X(X'X)^{-1}X'$ é conhecida como “matriz chapéu”, pois $\hat{y} = Hy$. Os elementos da diagonal principal da matriz H assumem valores no intervalo $(0, 1)$ e somam p , i.e., $\text{tr}(H) = p$. Dessa forma, a média dos elementos diagonais de H é igual a p/n .

Os elementos diagonais da matriz H são denotados por h_1, \dots, h_n e são utilizados como medidas dos graus de alavancagem, i.e., o i -ésimo elemento diagonal de H (h_i) mede o grau de influência da i -ésima observação sobre o correspondente valor predito (\hat{y}_i). Uma regra muito usada é concluir que a i -ésima observação é ponto de alavanca se $h_i > 2p/n$ ou $h_i > 3p/n$. Maiores detalhes poderão se encontrados em [Judge, Hill e Griffiths \(1988, p. 893\)](#).

Em modelos de regressão heteroscedásticos quando se conhece a matriz Ω de covariâncias dos erros (o que quase nunca ocorre na prática), é possível transformar o modelo $y = \beta X + \varepsilon$ de forma que as suposições necessárias para a validade do Teorema de Gauss-Markov sejam satisfeitas. No contexto em que se conhece a matriz de covariâncias dos erros, o estimador de mínimos quadrados generalizados (EMQG) é dado por $\hat{\beta}_G = (X'\Omega^{-1}X)^{-1}X'\Omega^{-1}y$. Pode-se mostrar que $\mathbb{E}(\hat{\beta}_G) = \beta$ e

$$\text{cov}(\hat{\beta}_G) = \Psi_{\hat{\beta}_G} = (X'\Omega^{-1}X)^{-1}.$$

Quando a suposição [S3'] é válida, ou seja, sob homoscedasticidade, tem-se que $\hat{\beta}_G = \hat{\beta}$ e $\text{cov}(\hat{\beta}_G) = \text{cov}(\hat{\beta}) = \sigma^2(X'X)^{-1}$.

Estimação do vetor β pelo método de mínimos quadrados generalizados é inviável, uma vez que os elementos de Ω (i.e., as variâncias dos n erros) são desconhecidos. Uma alternativa é postular um modelo para as variâncias dos erros e usar esse modelo para obter uma estimativa de Ω . Essa estimativa seria então utilizada no lugar da matriz verdadeira no estimador de mínimos quadrados generalizados. O estimador resultante é conhecido como estimador de mínimos quadrados generalizados viável (EMQGV) e é dado por

$$\hat{\hat{\beta}} = (X'\hat{\Omega}^{-1}X)^{-1}X'\hat{\Omega}^{-1}y.$$

O Teorema de Gauss Markov, contudo, não vale para o EMQGV. De fato, esse estimador não é linear e não é possível mostrar, em geral, que ele é não-viesado. Adicionalmente, para se usar esse estimador é necessário postular um modelo para as n variâncias. Tipicamente, contudo, é mais difícil modelar variâncias do que efeitos médios.

Deve ser notado que mesmo não valendo o Teorema de Gauss Markov, ou seja, quando a suposição [S3'] não é válida (i.e., os erros seguem um padrão heteroscedástico), $\hat{\beta}$ permanece não-viesado, consistente e assintoticamente gaussiano. Dessa forma, podemos basear as estimativas pontuais nesse estimador. Para a obtenção de intervalos de confiança e realização de testes de hipóteses é necessário um estimador consistente da matriz de covariância de $\hat{\beta}$. Para tanto, deve-se usar um estimador $\hat{\Omega}$ de Ω tal que $X'\hat{\Omega}X$ seja consistente para $X'\Omega X$, ou seja, $\text{plim}((X'\Omega X)^{-1}(X'\hat{\Omega}X)) = I_p$, onde plim denota limite em probabilidade.

White (1980) propôs um estimador da matriz de covariâncias $\Psi_{\hat{\beta}}$ que é consistente tanto sob homoscedasticidade quanto sob heteroscedasticidade de forma desconhecida. Ele percebeu que não seria preciso estimar a matriz de covariâncias dos erros de forma consistente; note que essa matriz possui n quantidades desconhecidas. Ele notou que basta estimar consistentemente $(X'\Omega X)$, que possui $p(p+1)/2$ elementos desconhecidos independentemente do tamanho da amostra. O estimador de Halbert White é obtido substituindo-se o i -ésimo elemento da diagonal da matriz de covariâncias dos erros (σ_i^2) pelo i -ésimo resíduo MQO ao quadrado, ou seja,

$$\text{HC0} = (X'X)^{-1}X'\hat{\Omega}_0X(X'X)^{-1},$$

em que $\hat{\Omega}_0 = \text{diag}\{\hat{\varepsilon}_1^2, \dots, \hat{\varepsilon}_n^2\}$.

O estimador de White pode ser muito viesado em amostras de tamanho pequeno a moderado. O viés desse estimador tende a ser negativo, ou seja, ele é um estimador “otimista”, pois tipicamente subestima as variâncias verdadeiras; ver Cribari-Neto e Zarkos (1999), Cribari-Neto e Zarkos (2001) e MacKinnon e White (1985). O viés do estimador é mais acentuado em situações em que os dados incluem pontos de alavanca; ver Chesher e Jewitt (1987). Testes sobre os parâmetros que fazem uso desse estimador são tipicamente liberais, i.e., anticonservativos. Dessa forma, a hipótese nula tende a ser rejeitada acima do esperado.

Em busca de corrigir o viés do estimador HC0, uma sequência de estimadores HC0 corrigidos por viés foi obtida por Cribari-Neto, Ferrari e Cordeiro (2000). Esses resultados foram posteriormente estendidos por Cribari-Neto e Galvão (2003).

A partir de resultados obtidos por Horn, Horn e Duncan (1975), MacKinnon e White (1985) construíram um novo estimador denominado HC2, que usa

$$\hat{\Omega}_2 = \text{diag}\{\hat{\varepsilon}_1^2/(1-h_1), \dots, \hat{\varepsilon}_n^2/(1-h_n)\},$$

em que h_i denota o i -ésimo elemento diagonal da “matriz chapéu” $H = X(X'X)^{-1}X'$, $i = 1, \dots, n$. O estimador HC2 é não-viesado sob homoscedasticidade.

O estimador HC3 foi proposto por [Davidson e MacKinnon \(1993\)](#). Ele é uma aproximação ao estimador jackknife, que é obtido a partir da remoção sequencial de cada observação da amostra. O estimador jackknife é apresentado em [Davidson e MacKinnon \(1993, p. 308 a 309\)](#). O estimador da matriz de covariâncias dos erros utilizado no estimador HC3 é

$$\hat{\Omega}_3 = \text{diag}\{\hat{\varepsilon}_1^2/(1 - h_1)^2, \dots, \hat{\varepsilon}_n^2/(1 - h_n)^2\}.$$

Um estimador alternativo que também faz uso das medidas de alavancagem e inclui termos de ajustes para amostras finitas foi proposto por [Cribari-Neto \(2004\)](#). Tal estimador, denominado HC4, utiliza

$$\hat{\Omega}_4 = \text{diag}\{\hat{\varepsilon}_1^2/(1 - h_1)^{\delta_1}, \dots, \hat{\varepsilon}_n^2/(1 - h_n)^{\delta_n}\},$$

em que $\delta_i = \min\{4, h_i/\bar{h}\} = \min\{4, nh_i/p\}$. Note que $\bar{h} = n^{-1} \sum_{i=1}^n h_i = \text{tr}(H)/n = p/n$. O expoente $\delta_i > 0$ controla o nível de desconto para a i -ésima observação. Como $\delta_i > 0$ e $0 < (1 - h_i) < 1$, tem-se então que $(1 - h_i)^{\delta_i}$ também pertence a $(0,1)$. Dessa forma, o quadrado do i -ésimo resíduo será tanto mais inflacionado quanto maior for h_i relativamente ao seu valor médio. O desconto linear δ_i é truncado em 4, que corresponde ao dobro do desconto utilizado no estimador HC3, de modo que $\delta_i = 4$ quando $h_i > 4\bar{h} = 4p/n$.

Um outro estimador foi proposto por [Cribari-Neto, Souza e Vasconcellos \(2007, Errata: v. 37, n. 20, p. 3329–3330, 2008\)](#). Trata-se do estimador HC5, que usa

$$\hat{\Omega}_5 = \text{diag}\{\hat{\varepsilon}_1^2/\sqrt{(1 - h_1)^{\alpha_1}}, \dots, \hat{\varepsilon}_n^2/\sqrt{(1 - h_n)^{\alpha_n}}\},$$

em que

$$\alpha_i = \min \left\{ \frac{h_i}{\bar{h}}, \max \left\{ \frac{h_i}{\bar{h}}, \frac{kh_{\max}}{\bar{h}} \right\} \right\}.$$

Aqui, k é uma constante cujo valor foi escolhido numericamente a partir de simulações-piloto. Os autores sugerem utilizar $k = 0.7$. Esse estimador leva em conta a alavancagem maximal em todos os n termos de descontos utilizados.

Utilizando um dos estimadores consistentes para a matriz $X'\Omega X$, de dimensão $p \times p$, é possível realizar testes sobre os elementos do vetor β sem especificar a forma da heteroscedasticidade existente, ou seja, não é preciso modelar o comportamento do segundo momento da variável de interesse. O comportamento de alguns desses testes em amostras finitas foram analisados por [Cribari-Neto, Ferrari e Oliveira \(2005\)](#) através do uso de integração numérica. Há ainda resultados de avaliações utilizando Monte Carlo

sobre o comportamento em amostras finitas de testes cujas estatísticas utilizam os estimadores descritos acima. Resultados sobre estimação intervalar robusta à presença de heteroscedasticidade podem ser encontrados em [Cribari-Neto e Lima \(2009\)](#).

Intervalos de Confiança Bootstrap e Bootstrap Duplo

3.1 Introdução

O método bootstrap foi introduzido em 1979 por Bradley Efron. Tal método foi inspirado em uma metodologia anterior baseada em reamostragem denominada jackknife. [Efron \(1979\)](#) sintetizou as metodologias baseadas em reamostragem que até então existiam e estabeleceu uma nova área de pesquisa.

Inicialmente houve ceticismo sobre a metodologia bootstrap, tendo sido tal ceticismo superado à medida em que estudos acumularam evidências de que o bootstrap pode ser consideravelmente mais eficaz que metodologias tradicionais.

A ideia de substituir aproximações complicadas e muitas vezes imprecisas por métodos de simulação baseados em reamostragem tem atraído diversos pesquisadores a desenvolver metodologias baseadas em bootstrap para os mais variados fins. Com a popularização do método bootstrap, alguns pesquisadores começaram a estabelecer condições matemáticas sob as quais o bootstrap é justificável.

Na literatura existem muitos trabalhos que fazem uso de metodologias bootstrap. Em geral, o método bootstrap é utilizado para correção de viés de estimadores, construção de intervalos de confiança, testes de hipóteses, estimação do erro-padrão de um estimador, entre outros.

As metodologias bootstrap apresentam dois paradigmas, sendo eles o bootstrap paramétrico e o bootstrap não-paramétrico. Bootstrap paramétrico refere-se ao caso em que a reamostragem é feita com base em uma distribuição $F(\hat{\theta})$ conhecida ou estabelecida, em que $\hat{\theta}$ é um estimador para θ . Em contrapartida, no bootstrap não-paramétrico há o desconhecimento da distribuição F verdadeira. A reamostragem é feita com base na

função de distribuição empírica \hat{F}_n . Reamostrar de \hat{F}_n equivale a reamostrar dos dados com reposição.

O presente capítulo descreve metodologias de estimação intervalar utilizando bootstrap. Posteriormente serão descritos esquemas de estimação intervalar para os parâmetros de modelos lineares heteroscedásticos de regressão.

3.2 Intervalos de Confiança Paramétricos

Um intervalo de confiança (IC) é uma estimativa intervalar para um parâmetro de interesse de uma população. Em vez de estimar o parâmetro por um único valor (estimativa pontual), o intervalo de confiança fornece um conjunto de estimativas possíveis para esse parâmetro de interesse através de um intervalo aleatório. Estimativas intervalares são realizadas sob um nível de confiança $1 - \alpha$, com $\alpha \in (0, 1)$, em que α é o nível de significância adotado e fixado pelo pesquisador. Um intervalo I_γ (intervalo de nível γ) para o parâmetro θ é tal que

$$\Pr\{I_\gamma \text{ conter } \theta\} = \gamma. \quad (3.1)$$

Um intervalo de confiança bilateral é delimitado pelos limites inferior e superior $\ell_{\frac{\alpha_1}{2}}$ e $\ell_{1-\frac{\alpha_2}{2}}$ respectivamente, em que, α_1 e α_2 pertencem ao conjunto de valores possíveis de α , tal que

$$\Pr\left\{\theta < \ell_{\frac{\alpha_1}{2}}\right\} = \frac{\alpha_1}{2} \quad \text{e} \quad \Pr\left\{\theta < \ell_{1-\frac{\alpha_2}{2}}\right\} = 1 - \frac{\alpha_2}{2}. \quad (3.2)$$

A cobertura do intervalo $[\ell_{\alpha_1/2}, \ell_{1-\alpha_2/2}]$ é $\gamma = 1 - (\frac{\alpha_1}{2} + \frac{\alpha_2}{2})$, com $\alpha_1/2 + \alpha_2/2 = \alpha$, sendo $\alpha_1/2$ e $\alpha_2/2$ os erros de coberturas à esquerda e à direita do intervalo I_γ , respectivamente. As escolhas de α_1 e α_2 devem ser feitas de forma que a amplitude de I_γ seja a menor possível. Na prática é usual escolher α_1 e α_2 de modo que

$$\Pr\left\{\theta < \ell_{\frac{\alpha_1}{2}}\right\} = \Pr\left\{\theta > \ell_{1-\frac{\alpha_2}{2}}\right\} = \frac{\alpha}{2}.$$

Uma abordagem frequentemente utilizada na construção de intervalos de confiança paramétricos é considerar um estimador $\hat{\theta}$ para um parâmetro θ da população, em que $\hat{\theta}$ é usualmente um estimador de máxima verossimilhança de θ . Queremos encontrar um intervalo que contenha θ com $100\gamma\%$ de confiança. Seja T_n um estimador do escalar θ baseado em n observações e t sua estimativa. Por simplicidade, suponhamos que T_n seja uma variável aleatória contínua. Denotando-se o p -ésimo quantil da distribuição da variável aleatória $T_n - \theta$ por a_p , temos que

$$\Pr\left\{T_n - \theta \leq a_{\frac{\alpha_1}{2}}\right\} = \frac{\alpha}{2} = \Pr\left\{T_n - \theta \geq a_{1-\frac{\alpha_2}{2}}\right\}. \quad (3.3)$$

Como a quantidade $Q = T_n - \theta$ é inversível em θ e T_n depende apenas da amostra, podemos construir o intervalo de confiança para θ reescrevendo os eventos em (3.3), ou seja, podemos reescrever os eventos $T_n - \theta \leq a_{\frac{\alpha_1}{2}}$ e $T_n - \theta \geq a_{1-\frac{\alpha_2}{2}}$ como $\theta > T_n - a_{\frac{\alpha_1}{2}}$ e $\theta < T_n - a_{1-\frac{\alpha_2}{2}}$, respectivamente. Assim, o intervalo de confiança de nível γ é dado pelos limites

$$\ell_{\alpha/2} = t - a_{1-\frac{\alpha_2}{2}}, \quad \ell_{1-\alpha/2} = t - a_{\frac{\alpha_1}{2}}. \quad (3.4)$$

Em situações em que o intervalo bilateral é de interesse, a soma de $\alpha_1/2$ e $\alpha_2/2$ é igual a α . Quando estamos interessados em intervalos simétricos, temos que $\alpha_1 = \alpha_2 = \alpha$. Assim,

$$\ell_{\alpha/2} = t - a_{1-\alpha/2}, \quad \ell_{1-\alpha/2} = t - a_{\alpha/2}. \quad (3.5)$$

Para os casos em que apenas um dos limites é de interesse, ou seja, o pesquisador está interessado na construção de intervalos de confiança unilaterais, temos que os limites para construção dos intervalos unilateral inferior e unilateral superior são dados por $\ell_{1-\alpha}$ e ℓ_{α} respectivamente. Os limites serão obtidos de tal forma que $\Pr\{\theta < \ell_{\alpha}\} = \Pr\{\theta > \ell_{1-\alpha}\} = \alpha$.

3.3 Intervalos de Confiança Aproximados

Em um grande número de aplicações práticas a distribuição de $T_n - \theta$ é desconhecida, o que impossibilita calcularmos a_p . Este fato leva os pesquisadores a considerar vários métodos alternativos aproximados para construção de intervalos de confiança.

A maioria das metodologias propõe formas de estimar os quantis verdadeiros da distribuição desconhecida da variável aleatória $T_n - \theta$. A inferência estatística se preocupa em estabelecer metodologias para realização de testes hipóteses e construção de métodos para realizar estimativas pontuais e intervalares. Essas metodologias normalmente fazem uso de rigor matemático e se apoiam em suposições que precisam ser perfeitamente atendidas. Em se tratando de estimação intervalar, métodos baseados em bootstrap são amplamente utilizados. Entre tais metodologias, podemos citar os intervalos de confiança normais aproximados, o método bootstrap percentil, o bootstrap- t , o bootstrap duplo percentil e o bootstrap- t duplo. Essas metodologias serão descritas a seguir.

3.3.1 Intervalos Normais Aproximados

Um enfoque usual é utilizar a distribuição $\mathcal{N}(0, v)$ como uma aproximação assintótica da distribuição da variável aleatória $T_n - \theta$. Aqui suponha que T_n é um estimador assintoticamente gaussiano de θ . Um caso comum é quando T_n é o estimador de máxima

verossimilhança de θ . Essa aproximação fornece os limites de confiança aproximados

$$\ell_{\alpha/2}, \ell_{1-\alpha/2} = t \mp v^{1/2} z_{1-\alpha/2}, \quad (3.6)$$

em que $z_{1-\alpha/2} = \Phi^{-1}(1 - \alpha/2)$, Φ^{-1} sendo a função quantílica da distribuição normal padrão. Se $\hat{\theta}$ é um estimador assintoticamente normal de θ , a variância aproximada v pode ser obtida diretamente da função log-verossimilhança $\ell(\theta)$. Caso não haja parâmetros de incômodo, podemos utilizar o inverso da informação de Fisher observada, $v = -1/\ddot{\ell}(\hat{\theta})$ ou o inverso da informação de Fisher esperada dado por $v = 1/i(\hat{\theta})$, em que $i(\theta) = \mathbb{E}(-\ddot{\ell}(\theta)) = \text{var}(\dot{\ell}(\theta))$ e

$$\dot{\ell}(\theta) = \frac{\partial \ell(\theta)}{\partial \theta} \quad \text{e} \quad \ddot{\ell} = \frac{\partial^2 \ell(\theta)}{\partial \theta \partial \theta'}.$$

A equação (3.6) é a forma padrão para os limites de confiança construídos pela aproximação assintótica da distribuição da variável aleatória $T_n - \theta$ pela distribuição normal.

3.3.2 Intervalo Bootstrap Studentizado

Seja $x_n = \{X_1, X_2, \dots, X_n\}$ uma amostra aleatória de tamanho n de variáveis aleatórias supostamente independentes e identicamente distribuídas com distribuição F , em que F é desconhecida. Seja também x_n^* uma nova amostra obtida a partir da distribuição empírica de F , i.e, de (\hat{F}_n) . Considere T_n^* como sendo a estatística T_n com base na amostra x_n^* e t^* sua estimativa. Usando a forma geral para intervalos de confiança dada pela equação (3.5), podemos estimar os quantis $a_{\alpha/2}$ e $a_{1-\alpha/2}$, que correspondem aos quantis de $T_n^* - t$. No bootstrap studentizado assumimos a forma de uma aproximação normal para limites de confiança, mas substituímos a aproximação $\mathcal{N}(0, 1)$ por $z = (T_n - \theta)/V^{1/2}$ por uma aproximação bootstrap. Aqui não conhecemos v pois não conhecemos F , em que $V = \text{var}(T_n|F)$. Cada amostra simulada é utilizada para calcular t^* , uma estimativa consistente de V (v^*) e a versão bootstrap de z dada por $z^* = (t^* - t)/v^{*1/2}$. São calculados J (número de amostras bootstrap) valores z^* , que são posteriormente ordenados. O quantil p da distribuição de z é estimado por $z_{((J+1)p)}^*$, i.e, o valor na posição $(J+1)p$ dos valores ordenados de $z_1^*, z_2^*, \dots, z_J^*$. Em seguida, os limites de confiança dados em (3.6) são

$$\ell_{\alpha/2} = t - v^{1/2} z_{((J+1)(1-\alpha/2))}^*, \quad \ell_{1-\alpha/2} = t - v^{1/2} z_{((J+1)\alpha/2)}^*, \quad (3.7)$$

sendo estes os limites de confiança do bootstrap studentizado. Segundo Efron e Tibshirani (1993, p. 160), se $(J+1)\alpha/2$ não é inteiro devemos considerar $q = [(J+1)\alpha/2]$, em que $[\cdot]$ é a função maior inteiro. Assim, os quantis de interesse são dados pelo $(J+1-q)$ -ésimo e q -ésimo maior valor inteiro de $z_1^*, z_2^*, \dots, z_J^*$.

3.3.3 Intervalo Bootstrap Percentil

[Davison e Hinkley \(1997, p. 202\)](#) afirma que existe alguma transformação de T_n , $U = h(T_n)$, tal que U possui uma distribuição simétrica. Suponhamos que sabemos calcular o intervalo de confiança de nível $1 - \alpha$ para $\phi = h(\theta)$. Segundo [Davison e Hinkley \(1997\)](#) podemos utilizar bootstrap para obter uma aproximação da distribuição de $T_n - \theta$ utilizando a distribuição de $T_n^* - t$. Dessa forma, estimamos o p -ésimo quantil de $T_n - \theta$ pelo $(J + 1)p$ -ésimo valor ordenado de $t^* - t$, ou seja, o p -ésimo quantil de $T_n - \theta$ é estimado por $t_{((J+1)p)}^* - t$. Analogamente, o p -ésimo quantil de $h(T_n) - h(\theta) = U - \phi$ poderá ser estimado pelo $(J + 1)p$ -ésimo valor ordenado de $h(T_n^*) - h(t) = u^* - u$. Seja b_p o p -ésimo quantil de $U - \phi$. Como U tem distribuição simétrica, então $U - \phi$ também tem distribuição simétrica, logo é verdade que $b_{\frac{\alpha}{2}} = -b_{1-\frac{\alpha}{2}}$. Utilizando a forma geral para intervalos de confiança dada por (3.5) e a simetria de $U - \phi$, temos que $h(\ell_{\alpha/2}) = u + b_{\alpha/2}$ e $h(\ell_{1-\alpha/2}) = u + b_{1-\alpha/2}$. Como $b_{\alpha/2}$ e $b_{1-\alpha/2}$ são quantis da distribuição de $U - \phi$ e sabemos calcular os quantis dessa distribuição, temos que os limites inferior e superior de confiança são dados por $u + (u_{((J+1)\alpha/2)}^* - u)$ e $u + (u_{((J+1)(1-\alpha/2))}^* - u)$, respectivamente, implicando os limites

$$u_{((J+1)\alpha/2)}^*, \quad u_{((J+1)(1-\alpha/2))}^*,$$

cujas transformações para θ é

$$t_{(J+1)\alpha/2}^*, \quad t_{(J+1)(1-\alpha/2)}^*. \quad (3.8)$$

Observe que não precisamos conhecer a transformação h . O intervalo de nível $1 - \alpha$ para o parâmetro θ não envolve h e pode ser calculado sem o conhecimento desta transformação. O intervalo (3.8) é conhecido como intervalo bootstrap percentil. Segundo [Davison e Hinkley \(1997, p. 203\)](#) o método percentil poderá ser aplicado a qualquer estatística.

3.3.4 Intervalo Bootstrap Duplo Percentil

Quando usamos o método percentil, obtemos cobertura que pode ser diferente do nível desejado $(1 - \alpha)$. O interessante é que podemos continuar fazendo uso de bootstrap para corrigir tal discrepância. Esse fato mostra a flexibilidade do método bootstrap.

A ideia para se conseguir intervalos de confiança mais acurados é fazer uso de esquemas de bootstrap duplo, ou seja, para cada réplica do bootstrap original será realizado um outro bootstrap. Consideremos a situação em que apenas um limite de confiança é de interesse e seja ele o limite superior com nível de confiança nominal de $1 - \alpha$, em que

$$\Pr \{T_n - \theta \leq a_\alpha(F) \mid F\} = \Pr \{t(\hat{F}_n) - t(F) \leq a_\alpha(F) \mid F\} = \alpha.$$

Ignorando os erros de simulação, o que é realmente calculado é o limite de confiança $t(\hat{F}_n) - a_\alpha(\hat{F}_n)$. O viés do bootstrap percentil resulta do fato que $a_\alpha(\hat{F}_n) \neq a_\alpha(F)$, o que, em geral, implica

$$\Pr \left\{ t(F) \leq t(\hat{F}_n) - a_\alpha(\hat{F}_n) \mid F \right\} \neq 1 - \alpha. \quad (3.9)$$

Segundo [Davison e Hinkley \(1997\)](#), o que poderia ser feito para corrigir o viés é acrescentar uma correção para $a_\alpha(\hat{F}_n)$, contudo, uma abordagem mais bem sucedida é ajustar o índice α . Assim, podemos substituir $a_\alpha(\hat{F}_n)$ por $a_{q(\alpha)}(\hat{F}_n)$ e estimar qual o valor ajustado $\hat{q}(\alpha)$ deve ser utilizado. Portanto, queremos encontrar $q(\alpha)$ que satisfaça

$$\Pr \left\{ t(F) \leq t(\hat{F}_n) - a_{q(\alpha)}(\hat{F}_n) \mid F \right\} = 1 - \alpha. \quad (3.10)$$

Note que a solução $q(\alpha)$ depende de F , i.e., $q(\alpha) = q(\alpha, F)$. Como a distribuição F é desconhecida, estimaremos $q(\alpha)$ por $\hat{q}(\alpha) = q(\alpha, \hat{F}_n)$. Seja $x_n^* = \{X_1^*, X_2^*, \dots, X_n^*\}$ uma amostra obtida aleatoriamente com reposição de x_n e $x_n^{**} = \{X_1^{**}, X_2^{**}, \dots, X_n^{**}\}$ uma nova amostra obtida com reposição de x_n^* com funções de distribuições empíricas dadas por \hat{F}_n^* e \hat{F}_n^{**} , respectivamente. Sejam também T_n^* e T_n^{**} a estatística T_n calculadas em x_n^* e x_n^{**} , em que t^* e t^{**} são suas estimativas, respectivamente. Denotamos $\Pr^* \{ \cdot \}$ como uma probabilidade condicionada a \hat{F}_n^* e $\Pr^{**} \{ \cdot \}$ como uma probabilidade condicionada em \hat{F}_n^{**} . Obteremos $\hat{q}(\alpha)$ utilizando a versão bootstrap da equação (3.10), definida como

$$\Pr^* \left\{ t(\hat{F}_n) \leq t(\hat{F}_n^*) - a_{\hat{q}(\alpha)}(\hat{F}_n^*) \mid \hat{F}_n \right\} = 1 - \alpha. \quad (3.11)$$

A partir da definição (3.11) um esquema envolvendo um segundo nível de bootstrap é definido como

$$\Pr^* \left[\Pr^{**} \left\{ T_n^{**} \leq 2T_n^* - t \mid \hat{F}_n^* \right\} \geq \hat{q}(\alpha) \mid \hat{F}_n \right] = 1 - \alpha. \quad (3.12)$$

Considerando uma amostra de tamanho n , o estimador $\hat{\theta}_n$ do parâmetro de interesse θ baseado em n observações é dito ser de k -ésima ordem de precisão se sua razão de convergência é $O_p(n^{-k/2})$. Um conjunto de confiança C_n de θ é dito ser de k -ésima ordem de precisão se

$$|P(\theta \in C_n) - (1 - \alpha)| = O(n^{-k/2}), \quad (3.13)$$

sendo α o nível de significância adotado.

Segundo [Davison e Hinkley \(1997\)](#), a cobertura $1 - \alpha + O(n^{-a})$ é corrigida para $1 - \alpha + O(n^{-a-1/2})$, em que, para limites de confiança unilaterais, temos que $a = \frac{1}{2}$ ou $a = 1$. Para os casos em que o intervalo bilateral é de interesse temos que a cobertura $1 - \alpha + O(n^{-1})$ é corrigida para $1 - \alpha + O(n^{-2})$.

Em geral, especialmente em problemas não-paramétricos, o cálculo de (3.12) não pode ser feito de forma exata. Assim, métodos aproximados devem ser utilizados. Um algoritmo básico é dado como segue. Suponhamos que temos J amostras obtidas com base em \hat{F}_n e denotemos suas respectivas funções de distribuições empíricas por $\hat{F}_{n,1}^*, \hat{F}_{n,2}^*, \dots, \hat{F}_{n,J}^*$, em que $\hat{F}_{n,j}^*$ é a j -ésima função de distribuição empírica. Defina

$$u_j^* = \Pr(T_n^{**} \leq 2t_j^* - t \mid \hat{F}_{n,j}^*). \quad (3.14)$$

Os valores $u_1^*, u_2^*, \dots, u_J^*$ podem ser calculados por uma aproximação. Geramos K amostras de $\hat{F}_{n,j}^*$ e para cada uma delas obtemos os valores estimados $t_{j,1}^{**}, t_{j,2}^{**}, \dots, t_{j,k}^{**}$, com $k = 1, 2, \dots, K$. Assim,

$$u_{K,j}^* = K^{-1} \sum_{k=1}^K I\{t_{j,k}^{**} \leq 2t_j^* - t\}, \quad (3.15)$$

em que $I\{A\}$ é a função indicadora para um evento boreliano A . A versão de Monte Carlo de (3.12) é dada por

$$J^{-1} \sum_{j=1}^J I\{u_{K,j}^* \geq \hat{q}(\alpha)\} = 1 - \alpha, \quad (3.16)$$

em que $\hat{q}(\alpha)$ é o quantil α de $u_{K,j}^*$. A maneira mais simples de obter $\hat{q}(\alpha)$ é ordenando os valores $u_{K,j}^*$ em $u_{K,1}^* \leq u_{K,2}^* \leq \dots \leq u_{K,J}^* \in (0, 1)$, e então fazendo $\hat{q}(\alpha) = u_{K,(\alpha(J+1))}^*$. O $(J+1)\alpha$ -ésimo quantil de $u_{K,j}^*$ é utilizado para se obter o quantil corrigido de $t_j^* - t$. O algoritmo bootstrap duplo percentil para intervalos bilaterais é apresentado abaixo. Escolhas adequadas para J e K são apresentadas na Seção 3.5.

1. Para uma dada amostra x_n (amostra original) calcule a quantidade t ;
2. Gere J amostras (x_n^*) de x_n de forma não paramétrica e calcule t_j^* , com $j = 1, 2, \dots, J$;
3. Gere K novas amostras bootstrap (x_n^{**}) para cada uma das J amostras no passo anterior e, para cada uma, calcule $t_{j,k}^{**}$, com $k = 1, 2, \dots, K$;

4. Calcule

$$u_j^* = K^{-1} \sum_{k=1}^K I\{t_{j,k}^{**} \leq 2t_j^* - t\},$$

em que I é a função indicadora;

5. Ordene o vetor u^* com J posições e obtenha os quantis inferior e superior de u^* , que são dados, respectivamente, por $q_{\inf} = u_{(J+1)*\alpha/2}^*$ e $q_{\sup} = u_{(J+1)(1-\alpha/2)}^*$;

6. Ordene os valores $t_1^*, t_2^*, \dots, t_J^*$ (i.e, $t_{(1)}^* \leq t_{(2)}^* \leq \dots \leq t_{(J)}^*$) e construa o intervalo de confiança para a amostra original utilizando as estimativas dos quantis calculadas no passo anterior. Considerando os valores ordenados das estatísticas t_j^* , com $j = 1, 2, \dots, J$, os limites do intervalo de nível $1 - \alpha$ são dados por

$$t_{((J+1)\alpha/2)}^*, \quad t_{((J+1)(1-\alpha/2))}^*.$$

3.3.5 Intervalo Bootstrap Duplo Studentizado

Em alguns casos, a principal vantagem de utilizar o bootstrap duplo com respeito ao bootstrap simples é que os intervalos de confiança usando bootstrap duplo normalmente têm ordem superior de precisão. Consideremos o caso do bootstrap percentil. Nesse bootstrap calculamos $\hat{\theta}_1^*, \hat{\theta}_2^*, \hat{\theta}_3^*, \dots, \hat{\theta}_J^*$, para J suficientemente grande, em geral, o maior possível, e obtemos os quantis de interesse dessas quantidades. Esses quantis fornecem nossa estimativa intervalar para θ . Segundo [Beran \(1987\)](#), a acurácia do bootstrap percentil pode ser melhorada quando é utilizada uma transformação pivotal.

Seja $x_n = \{X_1, X_2, \dots, X_n\}$ uma amostra aleatória de tamanho n de variáveis aleatórias supostamente independentes e identicamente distribuídas com distribuição $F : \mathbb{R} \mapsto (0, 1)$ com parâmetro θ , sendo F desconhecida. Segundo [Beran \(1987, p. 458\)](#), as possíveis distribuições que podem aproximar F estão restritas a uma família de funções de distribuições acumuladas \mathcal{F} , em que \mathcal{F} poderá ser uma família paramétrica ou não-paramétrica. Para o caso de regiões de confiança assintóticas que não levam em consideração métodos bootstrap, temos que F pertencerá a uma família de distribuições paramétricas. Para os casos em que estamos a considerar regiões de confiança bootstrap, temos que F pertencerá a uma família de distribuições não-paramétricas. O parâmetro θ para o qual queremos construir uma região de confiança de nível $1 - \alpha$ é igual a $T(F)$ em que T é um funcional F .

Em um estudo numérico sobre intervalos de confiança bootstrap, [Efron \(1982\)](#) e [Hinkley e Wei \(1984\)](#) mostraram que transformações pivotais aumentam a precisão das estimativas intervalares. [Beran \(1987\)](#) argumentou que mesmo em casos que $R_n(\theta)$ não é uma quantidade pivotal, considerar uma transformação studentizada diminui a dependência de $R_n(\theta)$ com respeito a F . Segundo [Beran \(1987\)](#) $R_n(\theta)$ pode ser uma quantidade aproximadamente pivotal e não necessariamente uma quantidade exatamente pivotal. Em [Beran \(1987\)](#) essa quantidade é chamada de *root* (raiz) e aqui será denominada de quantidade studentizada e quando fizer necessário será chamada também de quantidade pivotal. Um caso comum é quando $\hat{\theta}_n$ segue uma distribuição normal. Suponhamos que $\hat{\theta}_n$ é um estimador normalmente distribuído e não viesado para θ . Dessa forma, temos que a distribuição de $\hat{\theta}_n - \theta$ é normal com média zero e variância desconhecida σ^2 , i.e., a distribuição da variável aleatória $\hat{\theta}_n - \theta$ depende de F através do parâmetro desconhecido

σ . Como $\hat{\theta}_n - \theta \sim \mathcal{N}(0, \sigma^2)$, temos que $(\hat{\theta}_n - \theta)/\sigma$ segue distribuição normal padrão, i.e., a quantidade $(\hat{\theta}_n - \theta)/\sigma$ não depende mais de F através de um parâmetro desconhecido. Seja $R_n(\theta) = R_n(x_n, \theta)$ uma quantidade pivotal dada por uma transformação studentizada. Seja também $c_n(\alpha)$ o quantil $1 - \alpha$ da distribuição de R_n e Θ o espaço paramétrico de θ . Então,

$$\{t \in \Theta : R_n(t) \leq c_n(\alpha)\} \quad (3.17)$$

é uma região de confiança de nível $1 - \alpha$ para θ . Um exemplo dessa construção é o intervalo de confiança para a média de uma distribuição normal que se baseia em uma quantidade pivotal.

Para o caso de regiões de confiança assintóticas, seja $H_n = H_n(\cdot, F)$ a função de distribuição acumulada da variável aleatória $R_n(\theta)$. Suponhamos que H_n converge fracamente para a função de distribuição $H = H(\cdot, F)$ quando n cresce. Além disso supõe-se que $H = H(x, F)$ é contínua em x e F é desconhecida. Seja \hat{F}_n uma estimativa consistente de F . Considere d como sendo uma métrica na família de distribuições acumuladas \mathcal{F} . Segundo Beran (1987, p. 459), supor que \hat{F}_n é uma estimativa consistente para F quer dizer que $d(\hat{F}_n, F)$ converge para zero em probabilidade. Uma estimativa natural de H é dada por $\hat{H} = H(\cdot, \hat{F}_n)$. Assim, pelas suposições acima e pela transformação integral de probabilidade temos que a variável aleatória $\hat{H}\{R_n(\theta)\}$ converge fracamente para a distribuição uniforme no intervalo $(0, 1)$. Uma região de confiança assintótica para θ baseada em R_n é dada por

$$A_n = \{t \in \Theta : \hat{H}\{R_n(t)\} \leq 1 - \alpha\}. \quad (3.18)$$

De forma equivalente e mais familiar podemos reescrever a região de confiança A_n dada por (3.18) como sendo

$$A_n = \{t \in \Theta : R_n(t) \leq \hat{H}^{-1}(1 - \alpha)\}, \quad (3.19)$$

em que $\hat{H}^{-1}(1 - \alpha)$ é o quantil $1 - \alpha$ calculado a partir da função de distribuição acumulada \hat{H} . A região apresentada em (3.18) é preferível pois pela transformação integral de probabilidade sabemos que a distribuição da variável aleatória $\hat{H}\{R_n(\theta)\}$ converge fracamente para a distribuição uniforme no intervalo $(0, 1)$.

Em se tratando de regiões de confiança bootstrap temos que \hat{F}_n pertence a uma família \mathcal{F} de distribuições acumuladas não-paramétricas, de tal forma que \hat{F}_n converge para a distribuição F . No caso não-paramétrico, uma estimativa consistente de F é obtida tomando a função de distribuição empírica (\hat{F}_n). Regiões de confiança bootstrap são mais utilizadas quando não conhecemos o limite H da distribuição da variável aleatória $R_n(\theta)$ quando n é suficientemente grande. A região de confiança bootstrap é análoga à região de confiança assintótica, em que

$$B_n = \{t \in \Theta : \hat{H}_n\{R_n(t)\} \leq 1 - \alpha\}. \quad (3.20)$$

A região de confiança apresentada em (3.20) pode ser escrita como

$$B_n = \{t \in \Theta : R_n(t) \leq \widehat{H}_n^{-1}(1 - \alpha)\}, \quad (3.21)$$

em que $\widehat{H}_n^{-1}(1 - \alpha)$ é o $1 - \alpha$ quantil da distribuição empírica, \widehat{H}_n . Temos que $\widehat{H}_n = \widehat{H}_n(\cdot, \widehat{F}_n)$ é uma estimativa bootstrap para a função de distribuição acumulada $H_n(\cdot, F)$ de $R_n(\theta)$.

Segundo Beran (1987, p. 459), se $\{F_n \in \mathcal{F}\}$ é uma sequência tal que F_n converge para F na métrica d , então $H_n(\cdot, F_n)$ converge fracamente para $H = H(\cdot, F)$ ($\sup|H_n(\cdot, F_n) - H(\cdot, F)| \xrightarrow{p} 0$), em que \xrightarrow{p} denota convergência em probabilidade, sendo H uma função de distribuição contínua que depende de F e não depende da sequência $\{F_n\}$. Beran (1987) apresenta também uma condição sob a qual a consistência de \widehat{F}_n para F implica que a estimativa bootstrap \widehat{H}_n converge fracamente para H . Assim, a distribuição da variável aleatória $R_{n,1}(\theta) = \widehat{H}_n\{R_n(\theta)\}$ converge para a distribuição uniforme no intervalo $(0,1)$. Seja x_n^* uma amostra bootstrap de tamanho n obtida da distribuição \widehat{F}_n e seja \widehat{F}_n^* uma estimativa de F obtida a partir de x_n^* . Sejam também $\widehat{\theta}_n = T(\widehat{F}_n)$ e $\widehat{\theta}_n^* = T(\widehat{F}_n^*)$, em que T é função que define o parâmetro θ . A ideia de metodologias bootstrap para estimação intervalar que fazem uso de uma transformação studentizada é utilizar a distribuição da variável aleatória $R_n(x_n^*, \widehat{\theta}_n) = (\widehat{\theta}_n^* - \widehat{\theta}_n)/\widehat{\sigma}_n^*$ para aproximar a distribuição da variável aleatória $R_n(x_n, \theta) = (\widehat{\theta}_n - \theta)/\widehat{\sigma}_n$, em que $\widehat{\sigma}_n$ e $\widehat{\sigma}_n^*$ são estimativas consistentes do desvio-padrão dos estimadores $\widehat{\theta}_n$ e $\widehat{\theta}_n^*$, respectivamente.

Temos que se $H_n(x, F) = \Pr\{R_n(x_n, \theta) \leq x|F\}$, então $H_n^{-1}(p, F) = \inf\{x \in \mathbb{R} : \Pr\{R_n(x_n, \theta) \leq x|F\} \geq p\}$, com $0 < p < 1$. A estimativa bootstrap da função de distribuição acumulada $H_n(x, F)$ é dada por

$$\widehat{H}_n(x) = H_n(x, \widehat{F}_n) = \Pr\{R_n(x_n^*, \widehat{\theta}_n) < x|\widehat{F}_n\}. \quad (3.22)$$

Como conhecemos a estimativa $\widehat{H}_n(x)$ de $H_n(x, F)$, temos que uma estimativa imediata para a função quantílica $H_n^{-1}(p, F)$ é dada por

$$\widehat{H}_n^{-1}(p) = \widehat{H}_n^{-1}(p, \widehat{F}_n) = \inf\{x \in \mathbb{R} : \Pr\{R_n(x_n^*, \widehat{\theta}_n) \leq x|\widehat{F}_n\} \geq p\}. \quad (3.23)$$

Os quantis estimados ($\widehat{t}_\alpha = \widehat{H}_n^{-1}(\alpha)$) da distribuição da variável aleatória $R_n(\theta)$ são utilizados para a obtenção dos intervalos unilateral superior e inferior, e o intervalo bilateral com $100(1 - \alpha)\%$ de confiança. Tais intervalos são, respectivamente, dados por

$$I_1 \equiv (-\infty, \widehat{\theta} - \widehat{t}_\alpha \widehat{\sigma}], \quad (3.24)$$

$$I_2 \equiv [\widehat{\theta} - \widehat{t}_{1-\alpha} \widehat{\sigma}, \infty), \quad (3.25)$$

$$I_3 \equiv [\widehat{\theta} - \widehat{t}_{1-\alpha/2} \widehat{\sigma}, \widehat{\theta} + \widehat{t}_{\alpha/2} \widehat{\sigma}]. \quad (3.26)$$

Na prática, seja J um número suficientemente grande de réplicas do primeiro nível de bootstrap. Para a j -ésima réplica de bootstrap, $j = 1, \dots, J$, é definida a variável aleatória

$$R_j(x_n^*, \hat{\theta}_n) = (\hat{\theta}_n^* - \hat{\theta}_n) / \hat{\sigma}_n^*, \quad (3.27)$$

em que $\hat{\theta}_n$ é calculado em x_n (amostra original) e $\hat{\theta}_n^*$ e $\hat{\sigma}_n^*$ são calculados utilizando a amostra x_n^* referente à j -ésima réplica de bootstrap. Após ordenar as quantidades $R_1(x_n^*, \hat{\theta}_n), R_2(x_n^*, \hat{\theta}_n), \dots, R_J(x_n^*, \hat{\theta}_n)$, as estimativas dos quantis para a construção das estimativas intervalares são dadas por

$$\hat{t}_\alpha = \widehat{H}_n^{-1}(\alpha) = R_n(x_n^*, \hat{\theta}_n)_{[(J+1)\alpha]}, \quad (3.28)$$

$$\hat{t}_{1-\alpha} = \widehat{H}_n^{-1}(1 - \alpha) = R_n(x_n^*, \hat{\theta}_n)_{[(J+1)(1-\alpha)]}, \quad (3.29)$$

$$\hat{t}_{\alpha/2} = \widehat{H}_n^{-1}(\alpha/2) = R_n(x_n^*, \hat{\theta}_n)_{[(J+1)\alpha/2]}, \quad (3.30)$$

$$\hat{t}_{1-\alpha/2} = \widehat{H}_n^{-1}(1 - \alpha/2) = R_n(x_n^*, \hat{\theta}_n)_{[(J+1)(1-\alpha/2)]}, \quad (3.31)$$

em que $[\cdot]$ é a função maior inteiro. Segundo [Beran \(1987, p. 459\)](#) a variável aleatória $R_{n,1}(\theta) = \widehat{H}_n\{R_n(\theta)\}$ terá distribuição exata uniforme no intervalo aberto $(0, 1)$ se $R_n(\theta)$ for de fato uma quantidade pivotal, contudo, a distribuição da variável aleatória $R_{n,1}(\theta)$ é menos dependente de F que a distribuição de $R_n(\theta)$ e está mais próxima de ser uma quantidade pivotal. Para reduzir o erro da região B_n , trataremos $R_{n,1}$ como sendo um novo pivô para a região de confiança de θ . Seja $H_{n,1} = H_{n,1}(\cdot, F)$ a função de distribuição acumulada da variável aleatória $R_{n,1}(\theta)$ e $\widehat{H}_{n,1} = H_{n,1}(\cdot, \widehat{F}_n)$ sua estimativa bootstrap. A nova região de confiança bootstrap corrigida é dada por

$$B_{n,1} = \{t \in \Theta : \widehat{H}_{n,1}\{R_{n,1}(t)\} \leq 1 - \alpha\} = \{t \in \Theta : \widehat{H}_{n,1}[\widehat{H}_n\{R_n(t)\}] \leq 1 - \alpha\}. \quad (3.32)$$

A região de confiança $B_{n,1}$ pode ser escrita como

$$B_{n,1} = \{t \in \Theta : R_n(t) \leq \widehat{H}_n^{-1}\{\widehat{H}_{n,1}^{-1}(1 - \alpha)\}\}. \quad (3.33)$$

Seja x_n^{**} uma nova amostra de tamanho n estimada a partir da distribuição \widehat{F}_n^* , em que cada elemento de x_n^{**} são condicionalmente independentes das amostras x_n e x_n^* . Temos também que a função de distribuição acumulada $H_{n,1}(x, F)$ da variável aleatória $R_{n,1}(\theta)$ pode ser escrita como sendo $H_{n,1}(x, F) = \Pr[\Pr\{R_n(x_n^*, \hat{\theta}_n) < R_n(x_n, \theta) | \widehat{F}_n\} < x | F]$. Assim, uma estimativa bootstrap para $H_{n,1}(x, F)$ é dada por

$$\widehat{H}_{n,1}(x) = H_{n,1}(x, \widehat{F}_n) = \Pr[\Pr\{R_n(x_n^{**}, \hat{\theta}_n^*) < R_n(x_n^*, \hat{\theta}_n) | \widehat{F}_n^*\} < x | \widehat{F}_n]. \quad (3.34)$$

Segundo [Beran \(1987, p. 461\)](#), uma aproximação para $\widehat{H}_{n,1}(x)$ pode ser obtida através da definição de uma nova variável aleatória Z_j referente à j -ésima réplica do bootstrap exterior (primeiro nível de bootstrap). Seja $\hat{\theta}_{n,j}^* = T(\widehat{F}_{n,j}^*)$. A variável Z_j refere-se

à fração dos valores $\{R_n(x_{j,k}^{**}, \hat{\theta}_{n,j}^*) : 1 \leq k \leq K\}$ que são menores do que a quantidade $R_n(x_j^*, \hat{\theta}_n)$, sendo K o número de réplicas do bootstrap interior (segundo nível de bootstrap). Para valores de J e K suficientemente grandes, temos que a distribuição empírica da variável aleatória $\{Z_j : 1 \leq j \leq J\}$ aproxima $\widehat{H}_{n,1}$. Assim, pela equação (3.33) calculamos o quantil $1 - \alpha$ dos valores ordenados de Z_1, Z_2, \dots, Z_J , que será um número real no intervalo $(0,1)$, sendo denotado por ξ . Dessa forma, $\widehat{H}_n^{-1}(\xi)$ será o ξ quantil de $H_n(x, \widehat{F}_n)$ que fornecerá uma região de confiança $B_{n,1}$ de nível próximo a $1 - \alpha$. Para obtermos um intervalo de confiança unilateral de nível $1 - \alpha$ para o parâmetro θ devemos inverter a quantidade studentizada $R_n(\theta)$ em (3.33). As estimativas dos quantis para construção de intervalos de confiança unilateral inferior e superior e intervalo bilateral são dadas por

$$\hat{t}_\alpha = R_n(x_n^*, \hat{\theta}_n)_{((J+1)Q_\alpha)}, \quad (3.35)$$

$$\hat{t}_{1-\alpha} = R_n(x_n^*, \hat{\theta}_n)_{((J+1)Q_{1-\alpha})}, \quad (3.36)$$

$$\hat{t}_{\alpha/2} = R_n(x_n^*, \hat{\theta}_n)_{((J+1)Q_{\alpha/2})}, \quad (3.37)$$

$$\hat{t}_{1-\alpha/2} = R_n(x_n^*, \hat{\theta}_n)_{((J+1)Q_{1-\alpha/2})}, \quad (3.38)$$

em que Q_α é o α -percentil do vetor Q referente aos valores ordenados de Z_1, Z_2, \dots, Z_J .

A ideia do bootstrap duplo é obter uma região de confiança com ordem de precisão superior à região de confiança dada pelo bootstrap simples, ou seja, em geral, a região $B_{n,1}$ tem ordem de precisão superior à da região B_n . Para melhor entendimento do bootstrap- t duplo, considere o algoritmo para construção de uma estimativa intervalar para o parâmetro λ de uma distribuição exponencial.

1. Sejam y_1, y_2, \dots, y_n variáveis aleatórias tais que $y_i \sim \text{Exp}(\lambda)$, com $i = 1, 2, \dots, n$. Calculemos $\hat{\lambda}$ e $\hat{\sigma}(\hat{\lambda})$ de forma consistente;
2. Para J suficientemente grande, com $j = 1, 2, \dots, J$, gere $y_{j,1}^*, y_{j,2}^*, \dots, y_{j,n}^*$, sendo que $y_{j,i}^*$, $i = 1, 2, \dots, n$, é obtida aleatoriamente com reposição de y_i , $i = 1, 2, \dots, n$;
3. Calcule $\hat{\lambda}_j^*$ e $\hat{\sigma}(\hat{\lambda}_j^*)$ de forma consistente a partir de $y_{j,1}^*, y_{j,2}^*, \dots, y_{j,n}^*$ e obtenha o pivô $\hat{R}_j^* = (\hat{\lambda}_j^* - \hat{\lambda})/\hat{\sigma}(\hat{\lambda}_j^*)$;
4. Para cada estágio do bootstrap exterior j , com $j = 1, 2, \dots, J$, um número suficientemente grande de K amostras do bootstrap interior (segundo nível de bootstrap) é gerado, ou seja, geramos $y_{k,1}^{**}, y_{k,2}^{**}, \dots, y_{k,n}^{**}$, $k = 1, 2, \dots, K$. Aqui, obtemos $y_{k,i}^{**}$ a partir do vetor $y_{j,i}^*$, $i = 1, 2, \dots, n$. Calculamos as quantidades $\hat{\lambda}_{j,k}^{**}$, $\hat{\sigma}(\hat{\lambda}_{j,k}^{**})$ e $\hat{R}_{j,k}^{**} = (\hat{\lambda}_{j,k}^{**} - \hat{\lambda}_j^*)/\hat{\sigma}(\hat{\lambda}_{j,k}^{**})$ de forma consistente;

5. No j -ésimo bootstrap exterior, em que K bootstrap interiores são realizados, defina Z_j como sendo a proporção de vezes em que $\hat{R}_{j,k}^{**} \leq \hat{R}_j^*$, i.e.,

$$Z_j = K^{-1} \sum_{k=1}^K I\{\hat{R}_{j,k}^{**} \leq \hat{R}_j^*\},$$

I sendo a função indicadora;

6. A variável Z_j definida no passo anterior será utilizada para refinar os intervalos calculados com base no bootstrap exterior. Ao final de todas as réplicas bootstrap, teremos as estimativas $\hat{\lambda}_j^*$, \hat{R}_j^* e Z_j , $j = 1, 2, \dots, J$. Note que a construção de Z_j faz com que essa variável pertença ao intervalo $(0,1)$, pois Z_j é uma proporção. Para determinar o intervalo bilateral com $100(1 - \alpha)\%$ de confiança calcule os quantis $1 - \alpha/2$ e $\alpha/2$ dos valores de Z_j . Note que Z_j pertence ao intervalo $(0,1)$ e os quantis $1 - \alpha/2$ e $\alpha/2$ calculados a partir dessas quantidades também pertencem ao intervalo $(0,1)$. Use esses quantis como novos quantis que serão obtidos a partir das quantidades ordenadas de $\hat{R}_1^*, \hat{R}_2^*, \dots, \hat{R}_J^*$. Esses quantis serão utilizados para calcular os limites inferior e superior do intervalo de confiança bilateral de nível de confiança $100(1 - \alpha)\%$.

3.4 Algoritmos para estimativas intervalares em modelos lineares com heteroscedasticidade de forma desconhecida

O uso de bootstrap para o cálculo de estimativas intervalares em modelos de regressão com heteroscedasticidade de forma desconhecida a partir de uma ponderação dos resíduos foi proposto por Wu (1986). Esse esquema bootstrap é conhecido como bootstrap ponderado ou selvagem. A seguir serão descritos algoritmos para construção de estimativas intervalares para um parâmetro β_j que indexa o modelo de regressão linear com heteroscedasticidade de forma desconhecida utilizando bootstrap selvagem. Nesse esquema bootstrap, os resíduos utilizados na construção de uma nova variável resposta (variável explicativa), y^* no caso do bootstrap exterior e y^{**} no bootstrap interior, são multiplicados por t^* e t^{**} , respectivamente, que são obtidos aleatoriamente de uma distribuição com média zero e variância um. As distribuições mais utilizadas para t^* são a distribuição de Rademacher ou a distribuição normal padrão. Uma variável aleatória que segue a distribuição de Rademacher possui densidade concentrada nos valores -1 e 1 , podendo ser -1 ou 1 com igual probabilidade. Outras variáveis aleatórias com distribuição com média zero e variância um podem ser consideradas. Nas estimativas intervalares pelo método de bootstrap studentizado e bootstrap duplo studentizado as quantidades pivotais ou aproximadamente pivotais fazem uso de estimadores consistentes para o desvio-padrão da

estimativa do parâmetro de interesse. Essas estimativas foram apresentadas no Capítulo 2.

3.4.1 Bootstrap Percentil

1. Para cada i , $i = 1, \dots, n$, obtenha t_i^* aleatoriamente de uma distribuição que possui média zero e variância um;
2. Construa uma amostra bootstrap (y^*, X)

$$y_i^* = x_i \hat{\beta} + t_i^* \hat{\varepsilon}_i / \sqrt{1 - h_i}, \quad i = 1, \dots, n,$$

em que x_i é a i -ésima linha da matriz de regressores X ;

3. Calcule a estimativa de mínimos quadrados ordinários de β : $\hat{\beta}^* = (X'X)^{-1}X'y^*$;
4. Repita os passos 1 a 3 um grande número de vezes (J vezes);
5. Os limites inferior e superior com $100(1 - \alpha)\%$ de confiança que formam um intervalo para β_j são os quantis $\alpha/2$ e $1 - \alpha/2$ das quantidades $\hat{\beta}_j^*$, $j = 1, \dots, J$, respectivamente.

3.4.2 Bootstrap Studentizado

1. Para cada i , $i = 1, \dots, n$, obtenha t_i^* aleatoriamente de uma distribuição que possui média zero e variância um;
2. Construa uma amostra bootstrap (y^*, X)

$$y_i^* = x_i \hat{\beta} + t_i^* \hat{\varepsilon}_i / \sqrt{1 - h_i}, \quad i = 1, \dots, n,$$

em que x_i é a i -ésima linha da matriz de regressores X ;

3. Calcule $\hat{\beta}^*$ e $z_j^* = (\hat{\beta}_j^* - \hat{\beta}_j) / \sqrt{\widehat{\text{var}}(\hat{\beta}_j^*)}$, em que $\sqrt{\widehat{\text{var}}(\hat{\beta}_j^*)}$ é uma estimativa consistente do desvio-padrão de $\hat{\beta}_j^*$ para a amostra bootstrap e $\hat{\beta}_j$ é a estimativa de mínimos quadrados ordinários de β_j calculada na amostra original;
4. Repita os passos de 1 a 3 um grande número de vezes (J vezes);
5. Os limites inferior e superior com $(1 - \alpha)100\%$ de confiança que formam um intervalo para β_j são, respectivamente, $\hat{\beta}_j - \hat{t}_{1-\alpha/2} \sqrt{\widehat{\text{var}}(\hat{\beta}_j)}$ e $\hat{\beta}_j - \hat{t}_{\alpha/2} \sqrt{\widehat{\text{var}}(\hat{\beta}_j)}$, em que \hat{t}_γ é o quantil γ ($0 < \gamma < 1$) de J valores de $z^* (z_1^*, \dots, z_J^*)$ e $\sqrt{\widehat{\text{var}}(\hat{\beta}_j)}$ é uma estimativa consistente do desvio-padrão utilizado no passo 3 (agora calculada com a amostra original e não mais para as amostras bootstrap).

3.4.3 Bootstrap Duplo Percentil

1. Para cada i , $i = 1, \dots, n$, obtenha t_i^* aleatoriamente de uma distribuição que possui média zero e variância um;
2. Construa uma amostra bootstrap (y^*, X)

$$y_i^* = x_i \hat{\beta} + t_i^* \hat{\varepsilon}_i / \sqrt{1 - h_i}, \quad i = 1, \dots, n,$$

em que x_i é a i -ésima linha da matriz de regressores X ;

3. Calcule $\hat{\beta}^* = (X'X)^{-1}X'y^*$;
4. Para a amostra bootstrap (y^*, X) definida no passo 2, construa K novas amostras (y^{**}, X) de tal forma que

$$y_i^{**} = x_i \hat{\beta}^* + t_i^{**} \hat{\varepsilon}_i^* / \sqrt{1 - h_i}, \quad i = 1, \dots, n,$$

em que x_i é a i -ésima linha da matriz de regressores X , $\hat{\beta}^*$ é obtido do modelo linear que utiliza a amostra do bootstrap exterior (y^*, X) e $\hat{\varepsilon}_i^*$ é o i -ésimo resíduo do modelo linear que utiliza a amostra (y^*, X) . Calcule $\hat{\beta}^{**} = (X'X)^{-1}X'y^{**}$;

5. Repita os passos de 1 a 4 um grande número de vezes (J vezes), sendo J o número de réplicas do bootstrap exterior;
6. Para cada réplica do bootstrap exterior ($j = 1, \dots, J$), calcule

$$u_j^* = K^{-1} \sum_{k=1}^K I\{\hat{\beta}_{j,k}^{**} \leq 2\hat{\beta}_j^* - \hat{\beta}\},$$

em que I é a função indicadora e K é o número de réplicas do segundo nível de bootstrap. Note que u_j^* pertence ao intervalo $(0,1)$;

7. Calcule os quantis $q_{\inf} = u_{(J+1)*\alpha/2}^*$ e $q_{\sup} = u_{(J+1)(1-\alpha/2)}^*$ a partir do vetor $u_1^*, u_2^*, \dots, u_J^*$;
8. Fazendo uso dos quantis obtidos no passo anterior, o intervalo de confiança de nível $(1 - \alpha)100\%$ é formado pelos quantis q_{\inf} e q_{\sup} calculados a partir dos valores $\hat{\beta}_1^*, \hat{\beta}_2^*, \dots, \hat{\beta}_J^*$, sendo eles os quantis inferior e superior, respectivamente. Assim, os quantis calculados no passo 7 serão utilizados para obter os quantis corrigidos a partir das estimativas dos parâmetros obtidas no bootstrap exterior, ou seja, q_{\inf} -ésimo quantil e q_{\sup} -ésimo quantil de $\hat{\beta}_{(1)}^*, \dots, \hat{\beta}_{(J)}^*$, sendo eles os limites de confiança para o intervalo bootstrap percentil duplo.

3.4.4 Bootstrap Duplo Studentizado

1. Para cada i , $i = 1, \dots, n$, obtenha t_i^* aleatoriamente de alguma distribuição que possua média zero e variância um;
2. Construa uma amostra bootstrap (y^*, X)

$$y_i^* = x_i \hat{\beta} + t_i^* \hat{\varepsilon}_i / \sqrt{1 - h_i}, \quad i = 1, \dots, n,$$

em que x_i é a i -ésima linha da matriz de regressores X ;

3. Calcule $\hat{\beta}^*$ e $z_j^* = (\hat{\beta}_j^* - \hat{\beta}_j) / \sqrt{\widehat{\text{var}}(\hat{\beta}_j^*)}$, em que $\sqrt{\widehat{\text{var}}(\hat{\beta}_j^*)}$ é uma estimativa consistente do desvio-padrão de $\hat{\beta}_j^*$ para a amostra bootstrap e $\hat{\beta}_j$ é estimativa de mínimos quadrados ordinários de β_j calculada na amostra original;
4. A partir da amostra obtida via bootstrap exterior (y^*, X) definida no passo 2, construa K novas amostras (y^{**}, X) referentes ao segundo nível de bootstrap. Cada amostra (y^{**}, X) será construída de tal forma que

$$y_i^{**} = x_i \hat{\beta}^* + t_i^{**} \hat{\varepsilon}_i^* / \sqrt{1 - h_i} \quad i = 1, \dots, n,$$

em que x_i é a i -ésima linha da matriz de regressores X , $\hat{\beta}^*$ é a estimativa de mínimos quadrados ordinários obtida da amostra do bootstrap exterior (y^*, X) e $\hat{\varepsilon}_i^*$ é o i -ésimo resíduo obtido usando (y^*, X) ;

5. Calcule $\hat{\beta}^{**}$ para cada amostra (y^{**}, X) construída no passo 4. Também para cada amostra (y^{**}, X) calcula-se $z_k^{**} = (\hat{\beta}_{j,k}^{**} - \hat{\beta}_j^*) / \sqrt{\widehat{\text{var}}(\hat{\beta}_j^{**})}$, em que $\sqrt{\widehat{\text{var}}(\hat{\beta}_j^{**})}$ é uma estimativa consistente do desvio padrão de $\hat{\beta}_j^{**}$;
6. Repita os passos de 1 a 5 um grande número de vezes (J vezes), sendo J o número de réplicas do bootstrap exterior, e calcule para cada réplica do bootstrap exterior a quantidade

$$Z_j = K^{-1} \sum_{k=1}^K I \{ z_{j,k}^{**} \leq z_j^* \},$$

$j = 1, 2, \dots, J$. Note que $Z_j \in (0, 1)$;

7. Calcule os quantis $q_{\text{inf}} = 1 - \alpha/2$ e $q_{\text{sup}} = \alpha/2$ dos valores Z_1, Z_2, \dots, Z_J , em que $Z_1 \leq Z_2 \leq \dots \leq Z_J$;
8. Use os quantis calculados no passo 7 como novos quantis corrigidos. Obtenha o q_{inf} -ésimo e q_{sup} -ésimo quantis de $z_{(1)}^* < z_{(2)}^* < \dots < z_{(J)}^*$ e sejam eles $\hat{t}_c^{(1-\alpha/2)}$ e $\hat{t}_c^{(\alpha/2)}$, respectivamente. Com essas estimativas dos quantis corrigidos pelo segundo nível de bootstrap, estime os limites inferior e superior de confiança por $\hat{\beta}_j - \hat{t}_c^{(1-\alpha/2)} \sqrt{\widehat{\text{var}}(\hat{\beta}_j)}$ e $\hat{\beta}_j - \hat{t}_c^{(\alpha/2)} \sqrt{\widehat{\text{var}}(\hat{\beta}_j)}$, respectivamente.

3.5 Estimação do número de amostras bootstrap

Na prática, uma dúvida comum se refere ao número de amostras que devem ser consideradas para os bootstrap, exterior e interior, ou seja, quais os valores devem ser escolhidos para J e K . Em muitos estudos que fazem uso de bootstrap a escolha da quantidade de amostras que irão ser consideradas é tipicamente feita de forma subjetiva e se baseia nos valores escolhidos em estudos anteriores ou em sugestões de outros trabalhos. A depender da finalidade do método bootstrap a escolha pode não ser ideal.

Em se tratando de estimativas intervalares aproximadas via bootstrap, [Booth e Hall \(1994\)](#) fornecem uma orientação para uma escolha adequada dos valores de J e K . Segundos eles, para um dado J , a precisão das estimativas intervalares diminui para um valor de K demasiadamente grande ou para um K muito pequeno. A ideia do artigo é propor uma estratégia para calibrar o bootstrap percentil duplo ou bootstrap- t duplo através de escolhas adequadas para J e K . Essas técnicas de calibração são denominadas de bootstrap iterados. Os autores constroem uma estimativa intervalar pelo método bootstrap percentil duplo e assumem que J e K são infinitos o que, segundo eles irá garantir obtermos um intervalo de confiança de nível exatamente igual à $(1 - \alpha)$. Na prática, é impossível considerarmos infinitos valores para J e K . Assim, [Booth e Hall \(1994, p. 335–336\)](#) definem um estimador que leva em consideração o limite superior do intervalo de confiança bootstrap percentil duplo com infinitas réplicas de bootstrap e o limite superior do intervalo de confiança bootstrap percentil duplo para um número finito de réplicas. O estimador se dá pela diferença desses limites. Foram construídos dessa forma um estimador para o intervalo de confiança unilateral e outro para o intervalo de confiança bilateral cujos os erros quadráticos médio foram calculados por eles e denominados de $M_1(J, K)$ e $M_2(J, K)$, respectivamente. Eles mostraram também que para intervalos de confiança bootstrap- t duplo também é possível fazer uso de $M_1(J, K)$ e $M_2(J, K)$ para obtermos os valores do número de réplicas do bootstrap exterior e interior. Segundo [Booth e Hall \(1994\)](#), para um intervalo unilateral de nível de confiança de $(1 - \alpha)$, o erro quadrático médio assintótico é proporcional a

$$M_1(J, K) = 2\alpha(1 - 2\alpha)J^{-1} + \left(\frac{1}{2} - \alpha\right)^2 K^{-2}. \quad (3.39)$$

A minimização da equação (3.39) em $\{J, K\}$ sujeita à restrição $JK = L$ fornece $J = \gamma_1 L^{2/3}$ e $K = \gamma_1^{-1} L^{1/3}$, em que $\gamma_1 = \{\alpha(1 - 2\alpha)(1/2 - \alpha)^{-2}\}^{1/3}$. Para intervalos bilaterais o erro quadrático médio assintótico é dado por

$$M_2(J, K) = \alpha \left(\frac{5}{4} - \alpha\right) J^{-1} + (1 - \alpha)^2 K^{-2}. \quad (3.40)$$

Minimizando a quantidade em (3.40) sujeito à $JK = L$, temos que os valores de J e K são $\gamma_2 L^{2/3}$ e $\gamma_2^{-1} L^{1/3}$, respectivamente, em que $\gamma_2 = \{(1/2)(1 - \alpha)^{-2}\alpha(5/4 - \alpha)\}^{1/3}$.

Para uma maior acurácia na aproximação, o produto $JK(= L)$ deverá ser pelo menos n^3 ($L \geq n^3$), em que n é o tamanho da amostra. Devido à descontinuidade da função de distribuição empírica, os valores $(J + 1)\alpha$, $(J + 1)/K$ e $K/2$ deverão ser números inteiros. Se não tomarmos esse cuidado, essa singularidade poderá afetar a qualidade da aproximação.

Resultados Numéricos

Os resultados numéricos apresentados nesse capítulo são baseados no modelo

$$y_i = \beta_1 + \beta_2 x_i + \sigma_i \varepsilon_i, \quad i = 1, \dots, n, \quad (4.1)$$

com $\beta_1 = 1$ e $\beta_2 = 1$, $x_i \sim t_{(3)}$, em que ε_i segue uma distribuição com média zero e variância um e $\mathbb{E}(\varepsilon_i \varepsilon_j) = 0 \quad \forall i \neq j$. Nas simulações, usamos os seguintes tamanhos amostrais: $n = 20, 60, 100$. A i -ésima variância é

$$\sigma_i^2 = \sigma^2 \exp\{a x_i\}, \quad (4.2)$$

em que $\sigma^2 = 1$ e a é uma constante real. Essa forma de calcular as variâncias dos n erros do modelo (4.1) se deve ao fato de querermos heteroscedasticidade presente nos dados. Assim, encontraremos $a \in \mathbb{R}$ que forneça o grau de heteroscedasticidade prefixado. O grau de heteroscedasticidade apresentado no modelo (4.1) é medido por

$$\lambda = \frac{\max\{\sigma_i^2, i = 1, \dots, n\}}{\min\{\sigma_i^2, i = 1, \dots, n\}}. \quad (4.3)$$

Sob homoscedasticidade, $\lambda = 1$. Quanto maior o valor de λ , mais intensa a heteroscedasticidade. Em todas as simulações consideradas neste capítulo o nível heteroscedasticidade é fixado previamente. Para esse estudo foi considerado o caso homoscedástico ($\lambda = 1$) e os casos heteroscedásticos com $\lambda \approx 9$ (heteroscedasticidade fraca) e $\lambda \approx 49$ (heteroscedasticidade forte). Para que o nível de heteroscedasticidade permaneça constante em diferentes tamanhos de amostra ($n = 20, n = 60$ e $n = 100$), devemos encontrar o valor de a em cada um dos tamanhos de amostras que forneça $\max\{\sigma_i^2\}$ e $\min\{\sigma_i^2\}$ de tal forma que (4.3) seja satisfatoriamente próximo do nível de heteroscedasticidade fixado.

Os valores de $x_i, i = 1, \dots, n$, são ser gerados pelo código apresentado no Apêndice B como replicações de um conjunto de 20 observações, ou seja, para $n = 60$ e $n = 100$ esses dados com 20 observações foram replicados três e cinco vezes, respectivamente.

Assim, geramos a matriz de regressores X com diferentes dimensões linha (20, 60 e 100) replicando uma matriz de regressores inicial de dimensão 20×2 . Também é possível gerar diretamente a matriz de regressores com dimensão desejada sem replicações respeitando o nível de heteroscedasticidade fixado.

O valor da constante a é gerado automaticamente pelo código após fixarmos o nível de heteroscedasticidade (λ) desejado. Os valores da constante a que fornecerá níveis de heteroscedasticidade aproximadamente iguais a 9 ou 49 são obtidos incrementando $a = 0$ por 0.00001 e com esse valor de a incrementado (a_c) são calculadas as variâncias e o novo valor de λ (λ_c) utilizando as equações (4.2) e (4.3). A busca pelo valor da constante a que fornecerá nível de heteroscedasticidade próximo ao desejado irá parar antes de obtermos um valor de λ_c tal que $\lambda_c > \lambda - 0.00001$.

As simulações levaram em consideração diferentes distribuições de probabilidade para geração dos erros do modelo (4.1). Foram consideradas quatro distribuições de probabilidade, sendo duas simétricas e duas assimétricas. As distribuições consideradas foram a distribuição normal padrão, a distribuição $t_{(3)}$, a distribuição qui-quadrado com 2 graus de liberdade $\chi^2_{(2)}$ e a distribuição Weibull(2,3). A Figura 1 apresenta as densidades dessas distribuições.

Além dos diferentes tamanhos amostrais, das diferentes distribuições para os erros do modelo (4.1) e dos diferentes níveis de heteroscedasticidade considerados, também foi levado em consideração o desenho balanceado e não balanceado do modelo (4.1), ou seja, dados com presença de pontos de alta alavancagem e sem pontos de alavanca, respectivamente. Seja h_i o i -ésimo elemento diagonal da matriz $H = X(X'X)^{-1}X'$. Se $h_i > 3p/n = 3\bar{h} = 6/n$, com $i = 1, \dots, n$, então a i -ésima observação é considerada como uma observação de alta alavancagem. A Tabela 1 apresenta os limiares de alta alavancagem para os diferentes tamanhos de amostras e desenhos balanceados e não balanceados.

As simulações apresentadas no decorrer desse capítulo avaliam diferentes métodos de estimação intervalar (intervalos bilaterais) para o parâmetro β_2 do modelo linear apresentado em (4.1). Foram realizadas 10000 réplicas de Monte Carlo, 1000 amostras para o primeiro nível de bootstrap ($J = 1000$) e 500 amostras para o segundo nível bootstrap ($K = 500$).

Os intervalos de confiança para β_2 levaram em consideração três níveis nominais de cobertura, sendo eles 90%, 95% e 99%. Observou-se que intervalos de confiança mais precisos poderiam ser conseguidos utilizando a sugestão de Booth e Hall (1994), contudo, as estimativas para o número de amostras dos dois níveis bootstrap certamente apresentariam um custo computacional bem mais elevado.

Com base no erro quadrático médio assintótico dado na equação M_2 (Seção 3.5

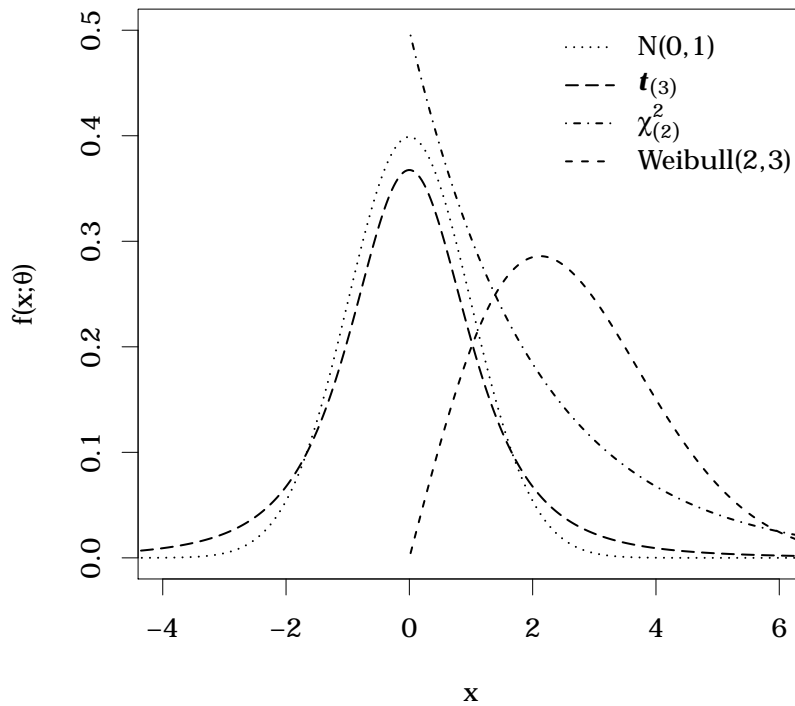


Figura 1 – Densidades consideradas na geração dos erros do modelo (4.1).

do Capítulo 3) de um estimador da acurácia de um intervalo bootstrap duplo bilateral apresentado por Booth e Hall (1994, p. 336), os erros ($\sqrt{M_2}$) na acurácia da estimação intervalar para $J = 1000$ e $K = 500$ considerados em todas as simulações foram aproximadamente iguais a 0.0109, 0.0080 e 0.0040 para os níveis de significância de 10%, 5% e 1%, respectivamente. Dessa forma, os erros alcançados em todas as simulações foram pequenos e assim podemos concluir que os valores de $J = 1000$ e $K = 500$ são adequados para avaliar os diferentes estimadores intervalares considerados nesse estudo. A Tabela 2 apresenta um comparativo das acurácias das estimativas intervalares utilizando $\sqrt{M_2}$ para os três níveis nominais de confiança (90%, 95% e 99%) levando em consideração as quantidades de amostras dos dois níveis de bootstrap utilizadas nas simulações apresentadas nesse capítulo ($J = 1000$ e $K = 500$) e comparando com o número de amostras sugeridas por Booth e Hall (1994).

Tabela 1 – Medida de máxima alavancagem e limiares para detecção de pontos de alta alavancagem.

Desenho	n	h_{\max}	Limiar		
			$2p/n$	$3p/n$	$4p/n$
Balanceado	20	0.1987	0.2000	0.3000	0.4000
	60	0.0662	0.0667	0.1000	0.1333
	100	0.0397	0.0400	0.0600	0.0800
Não Balanceado	20	0.6626	0.2000	0.3000	0.4000
	60	0.2209	0.0667	0.1000	0.1333
	100	0.1325	0.0400	0.0600	0.0800

Tabela 2 – Número de amostras e erros de acurácia de estimativas intervalares para os métodos bootstrap percentil duplo e bootstrap- t duplo para diferentes níveis de confiança.

Confiança (%)	J	K	Erro	$L = JK$
90	1000	500	0.0109	500000
95	1000	500	0.0080	500000
99	1000	500	0.0040	500000
90	4159	260	0.0056	1081340
95	3059	340	0.0052	1040060
99	1899	950	0.0028	1804050

Todos os valores de J e K apresentados na Tabela 2 diferentes de 1000 e 500, respectivamente, foram obtidos utilizando a sugestão de Booth e Hall (1994). Como podemos observar, a quantidade de amostras que devemos considerar para aumentar um pouco mais a acurácia das estimativas intervalares via bootstrap percentil duplo ou bootstrap- t duplo segundo a sugestão de Booth e Hall (1994) é consideravelmente maior. É importante frisar que em alguns casos, a minimização de M_2 fornece valores de J e K que não implicam custo computacional muito elevado para construção de estimativas intervalares via bootstrap duplo como poderá ser observado nas aplicações apresentadas no Capítulo 5. Contudo, para a avaliação dos métodos de estimação intervalar considerados nesse trabalho, considerar o número de amostras pela sugestão do Booth e Hall (1994) não justificaria o custo computacional envolvido. Na Tabela 2 observa-se que para o nível nominal de 99% de confiança o produto JK , com $J = 1899$ e $K = 950$, é 3.6081 vezes maior do que com $J = 1000$ e $K = 500$. Podemos nos convencer melhor sobre a inviabilidade de utilizar valores maiores para J e K como os sugeridos pela metodologia de Booth e Hall (1994) se levarmos em consideração o tempo gasto para realizarmos todas as simulações apresentadas nesse capítulo. Ao todo foram 72 simulações que totalizaram aproximadamente 6756 horas o que equivale a aproximadamente 281.5 dias. Necessitaríamos um pouco mais de 281 dias se as simulações fossem submetidas uma de cada vez em um computador com

processador AMD Opteron dodeca-core (24 núcleos) com frequência 2.3GHz utilizando OpenMP. Contudo, utilizando o Cluster SIG Altix (cluster Gauss) disponibilizado pelo CESUP foi possível submeter várias simulações simultaneamente e em aproximadamente quatro meses todas as 72 simulações foram concluídas.

Cada uma das 72 simulações avaliou os estimadores intervalares OLS, HC0, HC2, HC3, HC4 e HC5 sem uso de bootstrap como também os métodos bootstrap percentil e bootstrap- t em esquemas simples e duplo. O método bootstrap- t duplo levou em consideração os estimadores HC0, HC2, HC3, HC4 e HC5. A estimativa intervalar OLS utiliza a variância obtida por $\hat{\sigma}^2 c_{jj}$, onde c_{jj} é o elemento (jj) da matriz $(X'X)^{-1}$, em que nessas simulações $j = 2$. A quantidade studentizada $(\hat{\beta}_j - \beta_j)/\sqrt{\hat{\sigma}^2 c_{jj}}$ segue distribuição $t_{(n-2)}$ para erros normais. Os intervalos de confiança OLS utilizaram os quantis obtidos dessa distribuição mesmo em situações de erros não normais. Os outros intervalos que não fizeram uso de bootstrap (intervalos HC0, HC2, HC3, HC4 e HC5) utilizaram os quantis obtidos da distribuição normal padrão. Para esses intervalos, $\hat{\Psi}_{jj}^{(k)}$, com $k = 0, 2, 3, 4, 5$, é uma estimativa consistente da variância de $\hat{\beta}_j$. Como $\hat{\beta}_j$ é um estimador assintoticamente normal de β_j , temos que $(\hat{\beta}_j - \beta_j)/\sqrt{\hat{\Psi}_{jj}^{(k)}}$, $\forall k$ com $k = 0, 2, 3, 4, 5$, converge em distribuição para a distribuição normal padrão quando $n \rightarrow \infty$.

Os intervalos de confiança bilaterais OLS para β_2 de nível $(1 - \alpha)$ são obtidos por

$$\hat{\beta}_2 \pm t_{1-\alpha/2, n-2} \sqrt{\hat{\sigma}^2 c_{22}}, \quad (4.4)$$

em que $t_{1-\alpha/2, n-2}$ é o quantil $1 - \alpha/2$ da distribuição $t_{(n-2)}$. As estimativas intervalares bilaterais para β_2 que fazem uso de uma das estimativas consistentes do erro-padrão de $\hat{\beta}_2$ apresentado no Capítulo 2 (estimadores HC0, HC2, HC3, HC4 e HC5) são dadas por

$$\hat{\beta}_2 \pm z_{1-\alpha/2} \sqrt{\hat{\Psi}_{22}^{(k)}}, \quad (4.5)$$

em que $k = 0, 2, 3, 4, 5$ refere-se aos estimadores HC0, HC2, HC3, HC4 e HC5, respectivamente. A quantidade $z_{1-\alpha/2}$ é o quantil $1 - \alpha/2$ da distribuição normal padrão.

As Tabelas 3 a 14 apresentam os percentuais de cobertura dos intervalos de confiança bilaterais para o parâmetro β_2 do modelo (4.1) obtidos via simulações de Monte Carlo. Esses resultados referem-se aos intervalos de confiança sem uso de esquemas bootstrap calculados pelas equações (4.4) e (4.5). As tabelas também apresentam as coberturas bootstrap percentil, bootstrap duplo percentil e os tempos de simulações. Todas as simulações consideram diferentes tamanhos amostrais e desenhos balanceado e não balanceado. Para o desenho balanceado, a quantidade de pontos de alavanca (quantidade dos h_i tal que $h_i \geq 3p/n$, $i = 1, \dots, n$) foi igual a zero, ou seja, esquemas balanceados não consideraram nenhum ponto de alavanca. Já para o desenho não balanceado, considerou-se 2, 6 e 10 pontos de alavanca para os tamanhos de amostra 20, 60 e 100, respectivamente.

As Tabelas 3 a 5 apresentam os percentuais de cobertura das estimativas intervalares para erros normais para os níveis de 90%, 95% e 99% de confiança, respectivamente. Um fato interessante que pode ser observado refere-se aos intervalos bootstrap percentil duplo quando os dados não apresentaram pontos de alavanca (desenho balanceado). Note que os percentuais de cobertura do método bootstrap duplo percentil no caso balanceado ficaram muito próximos aos níveis nominais em casos de homoscedasticidade ($\lambda = 1$) e em casos de heteroscedasticidade ($\lambda \approx 9$ e $\lambda \approx 49$) para os tamanhos de amostra 20, 60 e 100. O método bootstrap duplo percentil mostrou-se consideravelmente superior ao método bootstrap percentil simples (um nível bootstrap). Note também que os percentuais de cobertura do método bootstrap percentil simples caem consideravelmente em situações em que os dados são não balanceados. Situações semelhantes foram observadas para erros não normais (erros $t_{(3)}$, $\chi^2_{(2)}$ e Weibull(2,3)), como pode ser observado nas Tabelas 6 a 14.

Tabela 3 – Percentuais de cobertura dos intervalos HC sem uso de esquemas bootstrap e intervalos bootstrap percentil simples e duplo para erros normais - nível nominal de 90%.

n	Desenho	λ	OLS	HC0	HC2	HC3	HC4	HC5	Percentil	Percentil Duplo	Tempo
20	Balanceado	$\lambda = 1$	88.20	84.26	86.96	89.48	88.28	86.24	87.04	89.44	7.4408
		$\lambda \approx 9$	79.54	80.40	81.86	83.82	85.46	85.08	85.90	89.50	7.3422
		$\lambda \approx 49$	78.00	81.32	82.70	84.02	85.42	83.76	84.64	89.20	7.3923
	Não Balanceado	$\lambda = 1$	93.90	77.36	85.18	86.08	87.90	86.06	78.48	79.20	7.3606
		$\lambda \approx 9$	79.84	75.06	80.86	81.70	83.38	84.72	82.08	85.58	7.3428
		$\lambda \approx 49$	77.84	76.90	81.04	83.70	84.66	84.64	83.64	85.58	7.3906
	Balanceado	$\lambda = 1$	90.14	88.92	89.72	90.32	89.96	89.48	86.74	90.34	55.2536
		$\lambda \approx 9$	87.12	88.76	89.44	87.08	89.66	89.16	83.46	89.12	54.8126
		$\lambda \approx 49$	85.08	84.32	85.96	89.64	89.10	88.78	80.80	88.64	56.1188
60	Balanceado	$\lambda = 1$	89.72	83.42	86.22	88.60	89.00	88.78	86.04	88.92	56.8720
		$\lambda \approx 9$	79.44	75.58	81.46	83.46	84.20	84.28	85.52	83.82	56.1946
		$\lambda \approx 49$	77.44	75.08	82.24	83.20	84.94	83.02	83.30	82.90	56.0587
	Não Balanceado	$\lambda = 1$	89.72	83.42	86.22	88.60	89.00	88.78	86.04	88.92	56.8720
		$\lambda \approx 9$	79.44	75.58	81.46	83.46	84.20	84.28	85.52	83.82	56.1946
		$\lambda \approx 49$	77.44	75.08	82.24	83.20	84.94	83.02	83.30	82.90	56.0587
	Balanceado	$\lambda \approx 1$	90.32	89.60	84.98	85.42	86.18	89.84	84.98	90.42	220.5600
		$\lambda \approx 9$	79.68	75.60	80.22	81.60	83.32	85.90	83.94	88.26	222.1320
		$\lambda \approx 49$	75.62	75.12	79.42	83.80	85.46	85.24	89.52	90.18	222.9650
100	Balanceado	$\lambda \approx 1$	89.26	85.34	86.86	88.52	90.90	88.64	86.88	88.70	222.3550
		$\lambda \approx 9$	75.18	76.06	81.24	82.20	83.74	85.96	78.88	82.70	221.8420
		$\lambda \approx 49$	77.26	76.66	80.46	83.58	83.16	84.12	81.46	81.90	221.3730
	Não Balanceado	$\lambda \approx 1$	89.26	85.34	86.86	88.52	90.90	88.64	86.88	88.70	222.3550
		$\lambda \approx 9$	75.18	76.06	81.24	82.20	83.74	85.96	78.88	82.70	221.8420
		$\lambda \approx 49$	77.26	76.66	80.46	83.58	83.16	84.12	81.46	81.90	221.3730
	Balanceado	$\lambda \approx 1$	90.32	89.60	84.98	85.42	86.18	89.84	84.98	90.42	220.5600
		$\lambda \approx 9$	79.68	75.60	80.22	81.60	83.32	85.90	83.94	88.26	222.1320
		$\lambda \approx 49$	75.62	75.12	79.42	83.80	85.46	85.24	89.52	90.18	222.9650

1 - Os tempos marcados estão apresentados em horas;

2 - As simulações foram realizadas utilizando o código em C++ apresentado no Apêndice B;

3 - Foram utilizados os quantis da distribuição normal padrão para construção das estimativas intervalares usando os métodos HC0, HC2, HC3, HC5 e HC5 e o quantil da distribuição t de Student com $n - 2$ graus de liberdade para o intervalo OLS.

Tabela 4 – Percentuais de cobertura dos intervalos HC sem uso de esquemas bootstrap e intervalos bootstrap percentil simples e duplo para erros normais - nível nominal de 95%.

n	Desenho	λ	OLS	HC0	HC2	HC3	HC4	HC5	Percentil	Percentil Duplo	Tempo
20	Balanceado	$\lambda = 1$	93.36	90.42	91.88	93.62	92.84	91.52	91.36	94.18	7.4408
		$\lambda \approx 9$	83.48	83.52	84.50	88.86	89.96	90.68	90.42	94.50	7.3422
		$\lambda \approx 49$	79.04	81.22	84.24	86.98	87.64	89.66	89.10	93.80	7.3923
	Não Balanceado	$\lambda = 1$	93.90	77.36	85.18	91.08	91.90	92.06	82.76	82.58	7.3606
		$\lambda \approx 9$	72.78	77.58	84.58	86.40	89.06	91.38	81.32	83.44	7.3428
		$\lambda \approx 49$	71.74	73.76	80.44	83.42	85.78	87.46	80.70	83.16	7.3906
	Balanceado	$\lambda = 1$	95.06	94.18	86.62	88.12	90.88	92.50	94.42	95.08	55.2536
		$\lambda \approx 9$	81.64	83.21	84.98	85.40	90.10	91.88	93.80	94.68	54.8126
		$\lambda \approx 49$	80.02	81.96	83.58	84.01	88.68	89.36	91.11	94.74	56.1188
60	Balanceado	$\lambda = 1$	95.42	82.60	84.54	88.90	91.26	92.06	90.82	93.06	56.8720
		$\lambda \approx 9$	82.64	81.22	84.58	85.92	87.46	90.80	85.90	91.34	56.1946
		$\lambda \approx 49$	79.90	81.90	86.14	87.42	88.90	87.46	89.58	90.94	56.0587
	Não Balanceado	$\lambda = 1$	95.42	82.60	84.54	88.90	91.26	92.06	90.82	93.06	56.8720
		$\lambda \approx 9$	82.64	81.22	84.58	85.92	87.46	90.80	85.90	91.34	56.1946
		$\lambda \approx 49$	79.90	81.90	86.14	87.42	88.90	87.46	89.58	90.94	56.0587
	Balanceado	$\lambda \approx 1$	94.78	84.32	86.70	88.02	92.90	94.52	94.38	94.86	220.5600
		$\lambda \approx 9$	84.82	83.31	85.42	87.70	88.52	90.36	90.58	91.84	222.1320
		$\lambda \approx 49$	82.74	83.16	84.58	85.76	87.64	90.42	87.22	95.14	222.9650
100	Balanceado	$\lambda \approx 1$	94.88	81.15	86.38	89.44	92.52	93.54	89.78	91.80	222.3550
		$\lambda \approx 9$	77.46	73.50	84.20	84.98	86.08	90.84	88.78	90.72	221.8420
		$\lambda \approx 49$	78.46	75.33	85.32	85.96	86.74	88.84	83.60	89.74	221.3730
	Não Balanceado	$\lambda \approx 1$	94.88	81.15	86.38	89.44	92.52	93.54	89.78	91.80	222.3550
		$\lambda \approx 9$	77.46	73.50	84.20	84.98	86.08	90.84	88.78	90.72	221.8420
		$\lambda \approx 49$	78.46	75.33	85.32	85.96	86.74	88.84	83.60	89.74	221.3730
	Balanceado	$\lambda \approx 1$	94.78	84.32	86.70	88.02	92.90	94.52	94.38	94.86	220.5600
		$\lambda \approx 9$	84.82	83.31	85.42	87.70	88.52	90.36	90.58	91.84	222.1320
		$\lambda \approx 49$	82.74	83.16	84.58	85.76	87.64	90.42	87.22	95.14	222.9650

1 - Os tempos marcados estão apresentados em horas;

2 - As simulações foram realizadas utilizando o código em C++ apresentado no Apêndice B;

3 - Foram utilizados os quantis da distribuição normal padrão para construção das estimativas intervalares usando os métodos HC0, HC2, HC3, HC5 e HC5 e o quantil da distribuição t de Student com $n - 2$ graus de liberdade para o intervalo OLS.

Tabela 5 – Percentuais de cobertura dos intervalos HC sem uso de esquemas bootstrap e intervalos bootstrap percentil simples e duplo para erros normais - nível nominal de 99%.

n	Desenho	λ	OLS	HC0	HC2	HC3	HC4	HC5	Percentil	Percentil Duplo	Tempo
20	Balanceado	$\lambda = 1$	97.94	92.24	93.16	95.00	95.60	96.80	90.00	98.58	7.4408
		$\lambda \approx 9$	85.94	84.94	88.84	89.62	91.20	93.60	89.51	99.22	7.3422
		$\lambda \approx 49$	82.78	83.22	85.14	87.42	90.36	92.76	87.30	96.38	7.3923
	Não Balanceado	$\lambda = 1$	98.28	87.02	90.31	94.32	98.13	96.04	87.86	95.80	7.3606
		$\lambda \approx 9$	82.70	83.88	83.08	90.03	90.68	91.92	91.34	94.78	7.3428
		$\lambda \approx 49$	80.84	78.88	82.62	81.80	87.94	90.80	90.16	95.52	7.3906
	Balanceado	$\lambda = 1$	98.66	92.30	91.56	94.72	95.64	95.44	98.30	98.80	55.2536
		$\lambda \approx 9$	83.56	82.26	84.52	90.70	90.58	91.38	93.98	98.68	54.8126
		$\lambda \approx 49$	81.12	81.98	83.22	85.44	89.28	92.08	90.60	97.60	56.1188
60	Balanceado	$\lambda = 1$	98.62	92.52	93.42	93.20	95.28	96.30	95.18	96.64	56.8720
		$\lambda \approx 9$	82.68	81.40	84.82	86.26	88.64	88.24	89.96	93.90	56.1946
		$\lambda \approx 49$	79.52	81.40	80.74	85.90	88.96	88.90	90.16	95.76	56.0587
	Não Balanceado	$\lambda = 1$	98.62	92.52	93.42	93.20	95.28	96.30	95.18	96.64	56.8720
		$\lambda \approx 9$	82.68	81.40	84.82	86.26	88.64	88.24	89.96	93.90	56.1946
		$\lambda \approx 49$	79.52	81.40	80.74	85.90	88.96	88.90	90.16	95.76	56.0587
	Balanceado	$\lambda \approx 1$	98.78	91.54	92.68	94.82	95.72	95.62	90.46	98.98	220.5600
		$\lambda \approx 9$	83.78	84.46	82.60	87.70	88.64	90.50	93.44	98.70	222.1320
		$\lambda \approx 49$	80.32	79.42	83.54	86.62	88.54	91.48	91.22	98.68	222.9650
100	Balanceado	$\lambda \approx 1$	96.66	94.58	96.18	97.74	98.34	97.78	86.20	97.86	222.3550
		$\lambda \approx 9$	84.72	85.30	88.70	90.96	90.28	92.94	87.98	93.88	221.8420
		$\lambda \approx 49$	80.46	80.33	84.58	86.50	89.74	90.50	88.64	90.42	221.3730
	Não Balanceado	$\lambda \approx 1$	96.66	94.58	96.18	97.74	98.34	97.78	86.20	97.86	222.3550
		$\lambda \approx 9$	84.72	85.30	88.70	90.96	90.28	92.94	87.98	93.88	221.8420
		$\lambda \approx 49$	80.46	80.33	84.58	86.50	89.74	90.50	88.64	90.42	221.3730
	Balanceado	$\lambda \approx 1$	98.78	91.54	92.68	94.82	95.72	95.62	90.46	98.98	220.5600
		$\lambda \approx 9$	83.78	84.46	82.60	87.70	88.64	90.50	93.44	98.70	222.1320
		$\lambda \approx 49$	80.32	79.42	83.54	86.62	88.54	91.48	91.22	98.68	222.9650

1 - Os tempos marcados estão apresentados em horas;

2 - As simulações foram realizadas utilizando o código em C++ apresentado no Apêndice B;

3 - Foram utilizados os quantis da distribuição normal padrão para construção das estimativas intervalares usando os métodos HC0, HC2, HC3, HC5 e HC5 e o quantil da distribuição t de Student com $n - 2$ graus de liberdade para o intervalo OLS.

Tabela 6 – Percentuais de cobertura dos intervalos HC sem uso de esquemas bootstrap e intervalos bootstrap percentil simples e duplo para erros $t_{(3)}$ - nível nominal de 90%.

n	Desenho	λ	OLS	HC0	HC2	HC3	HC4	HC5	Percentil	Percentil Duplo	Tempo
20	Balanceado	$\lambda = 1$	88.54	82.72	83.52	85.02	86.72	87.76	83.20	90.30	7.3946
		$\lambda \approx 9$	81.60	78.72	82.12	83.32	85.94	85.20	79.80	88.22	7.3811
		$\lambda \approx 49$	77.76	77.54	81.46	84.70	85.24	85.70	82.30	87.56	7.4061
	Não Balanceado	$\lambda = 1$	88.34	73.02	79.68	82.94	85.44	88.66	80.74	84.78	7.3308
		$\lambda \approx 9$	82.02	80.32	82.30	83.30	84.36	85.58	82.68	84.86	7.4995
		$\lambda \approx 49$	75.16	77.12	80.14	82.94	82.30	85.96	80.66	83.46	7.5976
	Balanceado	$\lambda = 1$	89.78	83.35	84.82	84.52	85.06	85.56	85.38	90.20	55.0695
		$\lambda \approx 9$	81.82	83.70	82.50	83.20	83.68	84.26	84.20	90.02	56.7594
		$\lambda \approx 49$	80.08	82.66	82.44	84.08	85.54	86.14	85.90	88.96	55.9323
60	Balanceado	$\lambda = 1$	89.84	84.40	83.22	84.10	83.88	85.32	83.02	85.20	55.9530
		$\lambda \approx 9$	80.62	83.76	84.70	84.78	85.56	84.62	81.44	83.98	55.8338
		$\lambda \approx 49$	79.26	78.68	82.98	83.02	83.60	84.64	81.56	83.14	55.8939
	Não Balanceado	$\lambda = 1$	89.84	84.40	83.22	84.10	83.88	85.32	83.02	85.20	55.9530
		$\lambda \approx 9$	80.62	83.76	84.70	84.78	85.56	84.62	81.44	83.98	55.8338
		$\lambda \approx 49$	79.26	78.68	82.98	83.02	83.60	84.64	81.56	83.14	55.8939
	Balanceado	$\lambda \approx 1$	90.50	82.50	83.94	83.36	84.98	85.86	88.72	91.24	221.8200
		$\lambda \approx 9$	80.54	80.52	83.08	83.50	84.20	85.82	85.70	89.50	219.9650
		$\lambda \approx 49$	80.82	81.06	83.40	84.08	84.62	83.31	83.38	90.34	222.5220
100	Balanceado	$\lambda \approx 1$	90.00	83.24	84.10	85.58	85.70	85.13	84.66	86.91	221.9190
		$\lambda \approx 9$	84.92	82.48	84.62	84.38	85.94	84.38	81.92	85.86	212.6090
		$\lambda \approx 49$	78.46	79.98	83.26	84.32	84.44	85.06	84.74	85.86	212.8480
	Não Balanceado	$\lambda \approx 1$	90.00	83.24	84.10	85.58	85.70	85.13	84.66	86.91	221.9190
		$\lambda \approx 9$	84.92	82.48	84.62	84.38	85.94	84.38	81.92	85.86	212.6090
		$\lambda \approx 49$	78.46	79.98	83.26	84.32	84.44	85.06	84.74	85.86	212.8480
	Balanceado	$\lambda \approx 1$	90.50	82.50	83.94	83.36	84.98	85.86	88.72	91.24	221.8200
		$\lambda \approx 9$	80.54	80.52	83.08	83.50	84.20	85.82	85.70	89.50	219.9650
		$\lambda \approx 49$	80.82	81.06	83.40	84.08	84.62	83.31	83.38	90.34	222.5220

1 - Os tempos marcados estão apresentados em horas;

2 - As simulações foram realizadas utilizando o código em C++ apresentado no Apêndice B;

3 - Foram utilizados os quantis da distribuição normal padrão para construção das estimativas intervalares usando os métodos HC0, HC2, HC3, HC5 e HC5 e o quantil da distribuição t de Student com $n - 2$ graus de liberdade para o intervalo OLS.Tabela 7 – Percentuais de cobertura dos intervalos HC sem uso de esquemas bootstrap e intervalos bootstrap percentil simples e duplo para erros $t_{(3)}$ - nível nominal de 95%.

n	Desenho	λ	OLS	HC0	HC2	HC3	HC4	HC5	Percentil	Percentil Duplo	Tempo
20	Balanceado	$\lambda = 1$	93.90	85.12	85.82	86.28	88.56	90.42	87.80	95.24	7.3946
		$\lambda \approx 9$	84.46	82.60	83.38	82.94	89.90	91.82	84.26	93.92	7.3811
		$\lambda \approx 49$	79.26	81.94	82.78	83.28	85.36	88.14	84.30	93.56	7.4061
	Não Balanceado	$\lambda = 1$	93.12	80.30	87.66	89.11	90.70	90.60	80.56	84.50	7.3308
		$\lambda \approx 9$	80.46	81.20	83.62	83.16	89.96	90.14	83.58	84.49	7.4995
		$\lambda \approx 49$	78.02	77.12	80.58	82.86	88.54	91.86	87.84	90.92	7.5976
	Balanceado	$\lambda = 1$	94.86	90.20	92.78	93.20	94.06	94.60	88.02	94.94	55.0695
		$\lambda \approx 9$	85.20	84.18	84.62	88.00	90.72	90.46	89.10	94.92	56.7594
		$\lambda \approx 49$	82.08	83.88	84.42	86.88	88.52	89.22	86.76	93.70	55.9323
60	Balanceado	$\lambda = 1$	94.36	88.32	83.56	84.88	90.78	91.00	89.08	90.32	55.9530
		$\lambda \approx 9$	82.66	83.54	85.90	86.20	88.48	90.54	85.20	90.52	55.8338
		$\lambda \approx 49$	79.14	81.86	82.12	84.12	88.86	90.54	79.36	88.72	55.8939
	Não Balanceado	$\lambda = 1$	94.36	88.32	83.56	84.88	90.78	91.00	89.08	90.32	55.9530
		$\lambda \approx 9$	82.66	83.54	85.90	86.20	88.48	90.54	85.20	90.52	55.8338
		$\lambda \approx 49$	79.14	81.86	82.12	84.12	88.86	90.54	79.36	88.72	55.8939
	Balanceado	$\lambda \approx 1$	95.42	90.30	91.54	93.76	94.62	95.46	89.30	95.16	221.8200
		$\lambda \approx 9$	85.60	85.16	86.67	86.86	89.64	90.42	92.96	95.18	219.9650
		$\lambda \approx 49$	80.22	83.72	85.92	86.63	90.02	92.86	89.28	94.90	222.5220
100	Balanceado	$\lambda \approx 1$	94.86	88.18	87.36	88.30	91.44	90.38	86.62	93.38	221.9190
		$\lambda \approx 9$	84.60	81.45	84.86	87.48	91.42	90.86	86.92	91.56	212.6090
		$\lambda \approx 49$	80.14	79.61	82.34	85.90	90.66	90.82	86.36	90.14	212.8480
	Não Balanceado	$\lambda \approx 1$	94.86	88.18	87.36	88.30	91.44	90.38	86.62	93.38	221.9190
		$\lambda \approx 9$	84.60	81.45	84.86	87.48	91.42	90.86	86.92	91.56	212.6090
		$\lambda \approx 49$	80.14	79.61	82.34	85.90	90.66	90.82	86.36	90.14	212.8480
	Balanceado	$\lambda \approx 1$	95.42	90.30	91.54	93.76	94.62	95.46	89.30	95.16	221.8200
		$\lambda \approx 9$	85.60	85.16	86.67	86.86	89.64	90.42	92.96	95.18	219.9650
		$\lambda \approx 49$	80.22	83.72	85.92	86.63	90.02	92.86	89.28	94.90	222.5220

1 - Os tempos marcados estão apresentados em horas;

2 - As simulações foram realizadas utilizando o código em C++ apresentado no Apêndice B;

3 - Foram utilizados os quantis da distribuição normal padrão para construção das estimativas intervalares usando os métodos HC0, HC2, HC3, HC5 e HC5 e o quantil da distribuição t de Student com $n - 2$ graus de liberdade para o intervalo OLS.

Tabela 8 – Percentuais de cobertura dos intervalos HC sem uso de esquemas bootstrap e intervalos bootstrap percentil simples e duplo para erros $t_{(3)}$ - nível nominal de 99%.

n	Desenho	λ	OLS	HC0	HC2	HC3	HC4	HC5	Percentil	Percentil Duplo	Tempo
20	Balanceado	$\lambda = 1$	98.50	85.42	86.32	87.96	90.70	92.10	93.12	98.48	7.3946
		$\lambda \approx 9$	88.70	83.42	85.22	84.84	91.52	92.04	93.98	96.76	7.3811
		$\lambda \approx 49$	83.36	81.82	83.68	84.36	89.88	90.28	90.06	98.66	7.4061
	Não Balanceado	$\lambda = 1$	96.88	85.32	84.94	90.92	95.84	97.14	83.64	87.54	7.3308
		$\lambda \approx 9$	83.14	82.78	85.82	86.94	92.56	92.94	90.84	91.15	7.4995
		$\lambda \approx 49$	82.66	83.28	84.22	89.64	91.88	93.58	91.26	94.93	7.5976
	Balanceado	$\lambda = 1$	99.04	92.84	91.02	93.14	96.08	96.94	94.56	98.98	55.0695
		$\lambda \approx 9$	83.04	84.58	85.68	87.84	91.68	93.64	91.18	98.97	56.7594
		$\lambda \approx 49$	83.64	81.38	83.58	86.80	90.64	92.48	92.88	96.86	55.9323
60	Balanceado	$\lambda = 1$	94.32	93.10	95.86	94.40	93.12	92.44	92.08	94.32	55.9530
		$\lambda \approx 9$	84.60	83.90	86.16	87.30	91.62	92.30	90.74	94.02	55.8338
		$\lambda \approx 49$	82.62	82.43	84.50	86.68	88.83	90.68	87.20	90.44	55.8939
	Não Balanceado	$\lambda = 1$	98.88	93.90	94.14	91.26	92.20	93.10	91.68	98.84	221.8200
		$\lambda \approx 9$	85.04	82.96	87.99	90.14	91.04	93.00	92.56	98.86	219.9650
		$\lambda \approx 49$	82.68	85.84	86.90	85.98	92.94	93.88	92.24	97.98	222.5220
	Balanceado	$\lambda = 1$	98.54	92.34	93.78	96.06	95.36	93.06	89.86	93.90	221.9190
		$\lambda \approx 9$	83.70	85.22	85.46	87.64	89.80	91.64	90.74	95.09	212.6090
		$\lambda \approx 49$	81.58	83.44	85.62	88.78	90.86	91.76	87.30	92.28	212.8480

1 - Os tempos marcados estão apresentados em horas;

2 - As simulações foram realizadas utilizando o código em C++ apresentado no Apêndice B;

3 - Foram utilizados os quantis da distribuição normal padrão para construção das estimativas intervalares usando os métodos HC0, HC2, HC3, HC5 e HC5 e o quantil da distribuição t de Student com $n - 2$ graus de liberdade para o intervalo OLS.Tabela 9 – Percentuais de cobertura dos intervalos HC sem uso de esquemas bootstrap e intervalos bootstrap percentil simples e duplo para erros $\chi^2_{(2)}$ - nível nominal de 90%.

n	Desenho	λ	OLS	HC0	HC2	HC3	HC4	HC5	Percentil	Percentil Duplo	Tempo
20	Balanceado	$\lambda = 1$	88.58	81.58	82.36	83.70	85.58	86.64	84.04	90.10	7.3532
		$\lambda \approx 9$	81.20	82.02	83.06	84.32	85.94	85.50	82.00	89.54	7.5438
		$\lambda \approx 49$	77.04	79.92	81.22	83.44	82.74	81.24	81.94	86.92	7.3637
	Não Balanceado	$\lambda = 1$	88.66	71.40	80.72	88.64	89.36	89.48	79.32	80.12	7.4816
		$\lambda \approx 9$	79.26	78.44	81.86	83.80	84.04	85.12	82.30	84.66	7.5684
		$\lambda \approx 49$	77.28	80.80	82.14	83.50	85.08	84.38	80.50	85.32	7.3189
	Balanceado	$\lambda = 1$	89.49	83.58	84.42	84.14	85.76	85.16	84.24	89.08	55.2214
		$\lambda \approx 9$	80.74	80.48	83.28	83.96	84.40	86.92	82.26	89.88	55.9804
		$\lambda \approx 49$	78.16	80.18	82.02	83.74	85.34	84.74	85.74	88.06	55.9323
60	Balanceado	$\lambda = 1$	90.00	83.08	84.04	85.16	85.24	84.36	84.62	87.56	55.8701
		$\lambda \approx 9$	79.06	80.14	81.60	82.54	83.64	85.42	80.34	85.54	55.9991
		$\lambda \approx 49$	78.64	80.02	81.24	82.44	84.14	86.46	82.42	84.28	56.2784
	Não Balanceado	$\lambda = 1$	89.32	82.62	83.14	84.78	85.56	86.96	84.12	89.84	221.0050
		$\lambda \approx 9$	82.72	83.30	84.74	84.38	85.94	85.68	80.64	89.72	221.4560
		$\lambda \approx 49$	80.22	79.18	81.64	83.00	84.66	86.48	84.58	89.34	221.6140
	Balanceado	$\lambda = 1$	90.32	82.40	83.84	84.96	86.10	87.02	85.50	88.88	221.3460
		$\lambda \approx 9$	78.86	81.92	83.24	82.32	84.12	85.24	82.08	84.06	222.5300
		$\lambda \approx 49$	76.16	79.16	81.16	83.64	85.02	85.30	81.26	84.86	211.4460

1 - Os tempos marcados estão apresentados em horas;

2 - As simulações foram realizadas utilizando o código em C++ apresentado no Apêndice B;

3 - Foram utilizados os quantis da distribuição normal padrão para construção das estimativas intervalares usando os métodos HC0, HC2, HC3, HC5 e HC5 e o quantil da distribuição t de Student com $n - 2$ graus de liberdade para o intervalo OLS.

Tabela 10 – Percentuais de cobertura dos intervalos HC sem uso de esquemas bootstrap e intervalos bootstrap percentil simples e duplo para erros $\chi^2_{(2)}$ - nível nominal de 95%.

n	Desenho	λ	OLS	HC0	HC2	HC3	HC4	HC5	Percentil	Percentil Duplo	Tempo
20	Balanceado	$\lambda = 1$	94.11	88.72	90.82	91.06	91.40	91.30	90.56	95.64	7.3532
		$\lambda \approx 9$	84.92	83.70	84.38	85.72	86.96	88.41	90.72	95.12	7.5438
		$\lambda \approx 49$	82.68	80.60	83.10	86.52	87.50	89.56	89.34	94.36	7.3637
	Não Balanceado	$\lambda = 1$	93.74	84.54	86.64	88.82	90.94	91.42	80.64	83.24	7.4816
		$\lambda \approx 9$	83.76	81.50	83.64	85.38	90.70	90.64	85.70	90.89	7.5684
		$\lambda \approx 49$	79.82	80.48	81.72	84.30	86.44	88.41	85.62	91.04	7.3189
	Balanceado	$\lambda = 1$	93.74	89.34	88.15	90.46	92.24	93.80	88.34	94.04	55.2214
		$\lambda \approx 9$	82.46	81.92	82.52	85.12	86.72	90.34	85.12	95.08	55.9804
		$\lambda \approx 49$	78.74	80.64	80.67	84.44	88.10	90.84	90.62	94.74	56.8088
60	Balanceado	$\lambda = 1$	94.64	83.86	87.76	94.26	94.02	95.96	90.16	92.12	55.8701
		$\lambda \approx 9$	87.30	83.90	82.72	85.82	90.46	89.02	85.16	90.66	55.9991
		$\lambda \approx 49$	82.56	81.42	83.38	86.56	88.62	90.64	84.56	92.52	56.2784
	Não Balanceado	$\lambda = 1$	94.64	83.86	87.76	94.26	94.02	95.96	90.16	92.12	55.8701
		$\lambda \approx 9$	87.30	83.90	82.72	85.82	90.46	89.02	85.16	90.66	55.9991
		$\lambda \approx 49$	82.56	81.42	83.38	86.56	88.62	90.64	84.56	92.52	56.2784
	Balanceado	$\lambda \approx 1$	94.88	87.34	87.66	91.08	93.88	94.54	89.34	95.00	221.0050
		$\lambda \approx 9$	84.46	83.34	83.52	84.88	85.64	86.48	88.26	94.77	221.4560
		$\lambda \approx 49$	82.36	81.94	82.24	83.54	90.30	92.10	86.78	92.88	221.6140
100	Balanceado	$\lambda \approx 1$	94.86	85.96	87.84	89.72	92.18	90.70	92.06	93.24	221.3460
		$\lambda \approx 9$	83.70	81.44	83.36	86.12	90.10	90.14	85.68	91.02	222.5300
		$\lambda \approx 49$	80.00	82.14	84.02	86.80	89.88	90.08	85.64	92.78	211.4460
	Não Balanceado	$\lambda \approx 1$	94.86	85.96	87.84	89.72	92.18	90.70	92.06	93.24	221.3460
		$\lambda \approx 9$	83.70	81.44	83.36	86.12	90.10	90.14	85.68	91.02	222.5300
		$\lambda \approx 49$	80.00	82.14	84.02	86.80	89.88	90.08	85.64	92.78	211.4460
	Balanceado	$\lambda \approx 1$	94.88	87.34	87.66	91.08	93.88	94.54	89.34	95.00	221.0050
		$\lambda \approx 9$	84.46	83.34	83.52	84.88	85.64	86.48	88.26	94.77	221.4560
		$\lambda \approx 49$	82.36	81.94	82.24	83.54	90.30	92.10	86.78	92.88	221.6140

1 - Os tempos marcados estão apresentados em horas;

2 - As simulações foram realizadas utilizando o código em C++ apresentado no Apêndice B;

3 - Foram utilizados os quantis da distribuição normal padrão para construção das estimativas intervalares usando os métodos HC0, HC2, HC3, HC5 e HC5 e o quantil da distribuição t de Student com $n - 2$ graus de liberdade para o intervalo OLS.Tabela 11 – Percentuais de cobertura dos intervalos HC sem uso de esquemas bootstrap e intervalos bootstrap percentil simples e duplo para erros $\chi^2_{(2)}$ - nível nominal de 99%.

n	Desenho	λ	OLS	HC0	HC2	HC3	HC4	HC5	Percentil	Percentil Duplo	Tempo
20	Balanceado	$\lambda = 1$	98.36	90.42	92.24	94.78	95.46	96.02	91.72	98.18	7.3532
		$\lambda \approx 9$	85.82	84.32	85.64	86.50	92.04	92.16	90.18	99.06	7.5438
		$\lambda \approx 49$	83.50	82.14	84.92	85.14	90.32	92.40	90.94	98.94	7.3637
	Não Balanceado	$\lambda = 1$	93.18	88.44	90.42	93.06	95.22	97.04	84.08	93.84	7.4816
		$\lambda \approx 9$	84.20	83.62	85.72	86.14	91.26	91.50	83.08	94.69	7.5684
		$\lambda \approx 49$	83.46	81.92	85.56	86.38	91.82	90.32	89.38	94.96	7.3189
	Balanceado	$\lambda = 1$	95.36	86.16	87.44	90.74	94.62	95.36	90.74	97.76	55.2214
		$\lambda \approx 9$	85.98	82.92	86.16	88.46	90.28	91.10	93.74	99.13	55.9804
		$\lambda \approx 49$	82.98	80.98	84.22	89.44	90.32	92.18	92.72	99.00	56.8088
60	Balanceado	$\lambda = 1$	96.34	92.70	93.32	95.00	95.76	95.92	90.98	96.64	55.8701
		$\lambda \approx 9$	86.64	84.66	83.16	87.54	92.84	93.54	88.50	92.34	55.9991
		$\lambda \approx 49$	84.79	82.78	83.36	89.64	92.94	93.64	87.74	94.48	56.2784
	Não Balanceado	$\lambda = 1$	96.34	92.70	93.32	95.00	95.76	95.92	90.98	96.64	55.8701
		$\lambda \approx 9$	86.64	84.66	83.16	87.54	92.84	93.54	88.50	92.34	55.9991
		$\lambda \approx 49$	84.79	82.78	83.36	89.64	92.94	93.64	87.74	94.48	56.2784
	Balanceado	$\lambda \approx 1$	98.74	84.68	87.76	90.92	92.82	92.72	90.44	98.96	221.0050
		$\lambda \approx 9$	85.50	83.94	86.02	88.14	93.06	93.98	88.60	98.86	221.4560
		$\lambda \approx 49$	83.76	82.38	85.52	90.62	92.56	93.48	87.14	98.62	221.6140
100	Balanceado	$\lambda \approx 1$	93.64	91.36	93.82	94.20	94.52	94.92	90.66	97.14	221.3460
		$\lambda \approx 9$	92.64	91.66	93.90	94.22	93.58	92.30	90.48	94.20	212.6090
		$\lambda \approx 49$	89.54	91.00	92.40	93.80	93.44	91.80	88.90	94.36	211.4460
	Não Balanceado	$\lambda \approx 1$	93.64	91.36	93.82	94.20	94.52	94.92	90.66	97.14	221.3460
		$\lambda \approx 9$	92.64	91.66	93.90	94.22	93.58	92.30	90.48	94.20	212.6090
		$\lambda \approx 49$	89.54	91.00	92.40	93.80	93.44	91.80	88.90	94.36	211.4460
	Balanceado	$\lambda \approx 1$	98.74	84.68	87.76	90.92	92.82	92.72	90.44	98.96	221.0050
		$\lambda \approx 9$	85.50	83.94	86.02	88.14	93.06	93.98	88.60	98.86	221.4560
		$\lambda \approx 49$	83.76	82.38	85.52	90.62	92.56	93.48	87.14	98.62	221.6140

1 - Os tempos marcados estão apresentados em horas;

2 - As simulações foram realizadas utilizando o código em C++ apresentado no Apêndice B;

3 - Foram utilizados os quantis da distribuição normal padrão para construção das estimativas intervalares usando os métodos HC0, HC2, HC3, HC5 e HC5 e o quantil da distribuição t de Student com $n - 2$ graus de liberdade para o intervalo OLS.

Tabela 12 – Percentuais de cobertura dos intervalos HC sem uso de esquemas bootstrap e intervalos bootstrap percentil simples e duplo para erros Weibull(2,3) - nível nominal de 90%.

n	Desenho	λ	OLS	HC0	HC2	HC3	HC4	HC5	Percentil	Percentil Duplo	Tempo
20	Balanceado	$\lambda = 1$	85.72	81.58	87.02	89.64	88.18	86.44	80.06	89.42	7.3920
		$\lambda \approx 9$	78.74	75.08	81.26	82.34	84.04	85.32	83.14	90.12	7.5388
		$\lambda \approx 49$	74.40	73.24	80.40	82.58	82.90	83.68	83.55	89.62	7.4891
	Não Balanceado	$\lambda = 1$	79.46	69.24	78.82	82.45	85.28	86.20	83.62	88.30	7.6432
		$\lambda \approx 9$	79.36	77.67	80.64	83.44	84.46	83.62	80.98	82.64	7.4351
		$\lambda \approx 49$	76.76	74.34	81.96	82.68	84.64	84.89	79.38	85.50	7.6669
	Balanceado	$\lambda = 1$	86.44	83.38	84.20	84.90	85.46	85.04	80.02	90.76	55.8420
		$\lambda \approx 9$	79.20	77.84	82.54	83.34	84.81	84.36	79.64	90.08	55.1949
		$\lambda \approx 49$	76.26	73.92	80.38	82.10	82.12	83.20	80.13	90.11	55.4217
60	Balanceado	$\lambda = 1$	84.30	81.50	82.20	83.80	84.38	85.06	80.04	83.30	55.7006
		$\lambda \approx 9$	80.76	79.12	83.21	83.66	84.18	84.46	81.76	82.05	55.1443
		$\lambda \approx 49$	79.92	77.56	81.53	82.12	84.08	84.78	81.87	82.21	55.4890
	Não Balanceado	$\lambda = 1$	83.94	82.94	83.62	84.16	84.38	85.28	83.80	84.68	212.5570
		$\lambda \approx 9$	79.68	77.00	81.34	81.60	82.86	83.34	76.21	87.36	215.2940
		$\lambda \approx 49$	76.96	74.28	79.54	80.56	82.86	81.26	75.32	82.85	212.4170
	Balanceado	$\lambda \approx 1$	90.08	84.22	84.84	85.36	86.98	84.62	89.67	90.30	211.7410
		$\lambda \approx 9$	81.46	79.34	80.82	82.20	84.90	84.62	80.84	90.09	211.4520
		$\lambda \approx 49$	79.26	78.02	80.40	83.72	84.52	83.32	81.38	90.00	212.7690
100	Balanceado	$\lambda \approx 1$	83.94	82.94	83.62	84.16	84.38	85.28	83.80	84.68	212.5570
		$\lambda \approx 9$	79.68	77.00	81.34	81.60	82.86	83.34	76.21	87.36	215.2940
		$\lambda \approx 49$	76.96	74.28	79.54	80.56	82.86	81.26	75.32	82.85	212.4170
	Não Balanceado	$\lambda \approx 1$	83.94	82.94	83.62	84.16	84.38	85.28	83.80	84.68	212.5570
		$\lambda \approx 9$	79.68	77.00	81.34	81.60	82.86	83.34	76.21	87.36	215.2940
		$\lambda \approx 49$	76.96	74.28	79.54	80.56	82.86	81.26	75.32	82.85	212.4170
	Balanceado	$\lambda \approx 1$	90.08	84.22	84.84	85.36	86.98	84.62	89.67	90.30	211.7410
		$\lambda \approx 9$	81.46	79.34	80.82	82.20	84.90	84.62	80.84	90.09	211.4520
		$\lambda \approx 49$	79.26	78.02	80.40	83.72	84.52	83.32	81.38	90.00	212.7690

1 - Os tempos marcados estão apresentados em horas;

2 - As simulações foram realizadas utilizando o código em C++ apresentado no Apêndice B;

3 - Foram utilizados os quantis da distribuição normal padrão para construção das estimativas intervalares usando os métodos HC0, HC2, HC3, HC5 e HC5 e o quantil da distribuição t de Student com $n - 2$ graus de liberdade para o intervalo OLS.

Tabela 13 – Percentuais de cobertura dos intervalos HC sem uso de esquemas bootstrap e intervalos bootstrap percentil simples e duplo para erros Weibull(2,3) - nível nominal de 95%.

n	Desenho	λ	OLS	HC0	HC2	HC3	HC4	HC5	Percentil	Percentil Duplo	Tempo
20	Balanceado	$\lambda = 1$	89.66	90.68	92.24	93.98	93.10	91.86	86.70	94.52	7.3920
		$\lambda \approx 9$	83.32	80.32	85.44	84.04	86.94	90.84	85.76	95.08	7.5388
		$\lambda \approx 49$	80.34	79.82	84.56	85.40	88.14	90.98	87.48	94.52	7.4891
	Não Balanceado	$\lambda = 1$	92.98	87.46	89.54	90.02	91.68	91.68	80.18	88.68	7.6432
		$\lambda \approx 9$	82.88	83.22	83.38	85.30	88.98	89.38	83.27	90.14	7.4351
		$\lambda \approx 49$	80.51	79.24	84.20	87.82	90.11	90.34	85.26	89.14	7.6669
	Balanceado	$\lambda = 1$	84.96	84.14	84.70	87.06	89.86	91.56	87.56	95.03	55.8420
		$\lambda \approx 9$	81.06	82.16	84.78	88.22	90.28	91.14	86.38	95.47	55.1949
		$\lambda \approx 49$	79.40	80.26	82.06	87.48	88.12	89.96	86.14	95.00	55.4217
60	Balanceado	$\lambda = 1$	89.48	83.82	84.98	84.56	90.20	92.74	88.55	92.68	55.7006
		$\lambda \approx 9$	83.98	83.43	84.52	85.60	89.34	90.46	86.02	95.04	55.1443
		$\lambda \approx 49$	79.02	77.43	86.72	88.70	90.98	89.62	86.00	90.10	55.4890
	Não Balanceado	$\lambda = 1$	89.48	83.82	84.98	84.56	90.20	92.74	88.55	92.68	55.7006
		$\lambda \approx 9$	83.98	83.43	84.52	85.60	89.34	90.46	86.02	95.04	55.1443
		$\lambda \approx 49$	79.02	77.43	86.72	88.70	90.98	89.62	86.00	90.10	55.4890
	Balanceado	$\lambda \approx 1$	92.76	89.14	94.42	94.74	94.62	94.34	88.40	95.02	211.7410
		$\lambda \approx 9$	82.12	81.20	83.44	87.74	89.50	91.40	86.30	94.94	211.4520
		$\lambda \approx 49$	80.56	78.88	84.20	88.48	90.24	90.04	86.02	94.97	212.7690
100	Balanceado	$\lambda \approx 1$	89.16	86.54	88.72	89.00	90.52	91.06	90.18	94.12	212.5570
		$\lambda \approx 9$	84.94	82.35	83.89	86.50	89.38	90.50	88.54	92.06	215.2940
		$\lambda \approx 49$	82.74	82.18	85.92	87.56	89.52	89.44	86.73	92.45	212.4170
	Não Balanceado	$\lambda \approx 1$	89.16	86.54	88.72	89.00	90.52	91.06	90.18	94.12	212.5570
		$\lambda \approx 9$	84.94	82.35	83.89	86.50	89.38	90.50	88.54	92.06	215.2940
		$\lambda \approx 49$	82.74	82.18	85.92	87.56	89.52	89.44	86.73	92.45	212.4170
	Balanceado	$\lambda \approx 1$	92.76	89.14	94.42	94.74	94.62	94.34	88.40	95.02	211.7410
		$\lambda \approx 9$	82.12	81.20	83.44	87.74	89.50	91.40	86.30	94.94	211.4520
		$\lambda \approx 49$	80.56	78.88	84.20	88.48	90.24	90.04	86.02	94.97	212.7690

1 - Os tempos marcados estão apresentados em horas;

2 - As simulações foram realizadas utilizando o código em C++ apresentado no Apêndice B;

3 - Foram utilizados os quantis da distribuição normal padrão para construção das estimativas intervalares usando os métodos HC0, HC2, HC3, HC5 e HC5 e o quantil da distribuição t de Student com $n - 2$ graus de liberdade para o intervalo OLS.

Tabela 14 – Percentuais de cobertura dos intervalos HC sem uso de esquemas bootstrap e intervalos bootstrap percentil simples e duplo para erros Weibull(2,3) - nível nominal de 99%.

n	Desenho	λ	OLS	HC0	HC2	HC3	HC4	HC5	Percentil	Percentil Duplo	Tempo
20	Balanceado	$\lambda = 1$	92.38	90.45	93.66	94.34	94.06	95.36	89.70	98.72	7.3920
		$\lambda \approx 9$	86.10	84.66	88.32	90.02	93.54	94.04	90.21	98.94	7.5388
		$\lambda \approx 49$	84.40	81.14	86.96	89.99	92.18	93.72	90.32	99.38	7.4891
	Não Balanceado	$\lambda = 1$	92.60	86.28	91.82	92.50	93.12	93.58	87.40	91.78	7.6432
		$\lambda \approx 9$	83.70	82.52	87.88	88.18	90.66	92.08	90.90	94.78	7.4351
		$\lambda \approx 49$	82.86	84.56	85.56	88.90	90.92	92.86	89.90	94.36	7.6669
	Balanceado	$\lambda = 1$	92.80	91.34	93.62	94.78	95.68	95.46	90.24	98.94	55.8420
		$\lambda \approx 9$	84.92	82.44	86.64	90.74	91.68	90.58	87.24	99.06	55.1949
		$\lambda \approx 49$	82.34	83.34	85.44	87.58	91.48	90.38	91.08	98.92	55.4217
60	Balanceado	$\lambda = 1$	90.84	88.30	93.28	94.12	94.30	93.14	90.04	94.44	55.7006
		$\lambda \approx 9$	85.84	83.30	87.82	88.12	90.56	90.14	89.92	92.67	55.1443
		$\lambda \approx 49$	83.76	81.38	86.72	88.82	90.94	91.82	89.12	91.31	55.4890
	Não Balanceado	$\lambda = 1$	90.84	88.30	93.28	94.12	94.30	93.14	90.04	94.44	55.7006
		$\lambda \approx 9$	85.84	83.30	87.82	88.12	90.56	90.14	89.92	92.67	55.1443
		$\lambda \approx 49$	83.76	81.38	86.72	88.82	90.94	91.82	89.12	91.31	55.4890
	Balanceado	$\lambda \approx 1$	95.86	90.50	94.66	95.82	95.74	91.64	90.46	98.98	211.7410
		$\lambda \approx 9$	84.94	83.64	87.74	88.84	91.76	92.68	90.42	98.95	211.4520
		$\lambda \approx 49$	82.42	82.04	83.56	88.70	90.62	90.52	88.12	98.89	212.7690
100	Balanceado	$\lambda \approx 1$	91.04	88.10	90.60	90.94	92.48	84.98	90.76	94.67	212.5570
		$\lambda \approx 9$	87.90	84.77	86.13	88.28	89.60	90.22	86.42	93.96	215.2940
		$\lambda \approx 49$	83.18	82.32	85.46	86.69	89.84	91.64	90.96	94.02	212.4170
	Não Balanceado	$\lambda \approx 1$	91.04	88.10	90.60	90.94	92.48	84.98	90.76	94.67	212.5570
		$\lambda \approx 9$	87.90	84.77	86.13	88.28	89.60	90.22	86.42	93.96	215.2940
		$\lambda \approx 49$	83.18	82.32	85.46	86.69	89.84	91.64	90.96	94.02	212.4170
	Balanceado	$\lambda \approx 1$	95.86	90.50	94.66	95.82	95.74	91.64	90.46	98.98	211.7410
		$\lambda \approx 9$	84.94	83.64	87.74	88.84	91.76	92.68	90.42	98.95	211.4520
		$\lambda \approx 49$	82.42	82.04	83.56	88.70	90.62	90.52	88.12	98.89	212.7690

1 - Os tempos marcados estão apresentados em horas;

2 - As simulações foram realizadas utilizando o código em C++ apresentado no Apêndice B;

3 - Foram utilizados os quantis da distribuição normal padrão para construção das estimativas intervalares usando os métodos HC0, HC2, HC3, HC4 e HC5 e o quantil da distribuição t de Student com $n - 2$ graus de liberdade para o intervalo OLS.

Também foi observado que obter as estimativas intervalares pelos métodos HC0, HC2, HC3, HC4 e HC5 para o parâmetro β_2 do modelo (4.1) utilizando a equação (4.5) e quantis $t_{(n-2)}$ ao invés de normais forneceram, em alguns casos, melhores coberturas, ou seja, coberturas mais próximas do níveis nominais considerados. As Figuras 2 a 5 apresentam um comparativo dos percentuais de cobertura obtidos via simulações de Monte Carlo utilizando os estimadores intervalares OLS, HC2, HC3, HC4 e HC5 usando quantis normais e usando $t_{(n-2)}$ para desenhos não balanceados, nível nominal de cobertura de 95%, $\lambda \approx 49$ e $n = 20$. Pode-se observar que as coberturas em pequenas amostras em geral são mais próximas ao nível nominal de cobertura (95%) quando utilizamos quantis da distribuição t de Student com $n - 2$ graus de liberdade. Note, por exemplo, que na Figura 2 referente às coberturas dos intervalos de confiança para β_2 sob erros normais, os intervalos de confiança que utilizaram quantis da distribuição $t_{(n-2)}$ foram claramente melhores do que os intervalos que fizeram uso de quantis da distribuição normal padrão. Houve aumentos de 6.36%, 6.06% e 2.66% nas coberturas dos intervalos HC3, HC4 e HC5, respectivamente, o que é um aumento razoavelmente significativo. Em alguns casos, o aumento percentual de cobertura não foi muito significativo podendo até mesmo haver coberturas um pouco inferiores às obtidas quando considerados os quantis da distribuição normal padrão. Contudo, em geral, os aumentos no nível de cobertura das estimativas intervalares que utilizaram os quantis da distribuição $t_{(n-2)}$ foram significativos e se sobressaíram em relação aos casos em que não houve um aumento significativo.

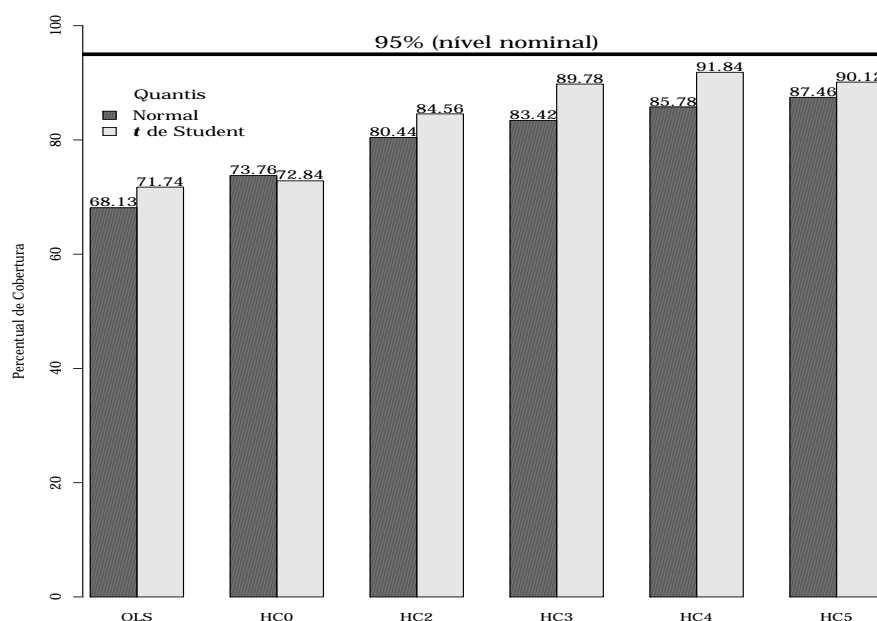


Figura 2 – Coberturas dos estimadores intervalares (sem uso de bootstrap) utilizando os quantis da distribuição normal padrão e da distribuição t de Student com $n - 2$ graus de liberdade - $n = 20$, $\lambda \approx 49$, erros normais e desenho não balanceado.

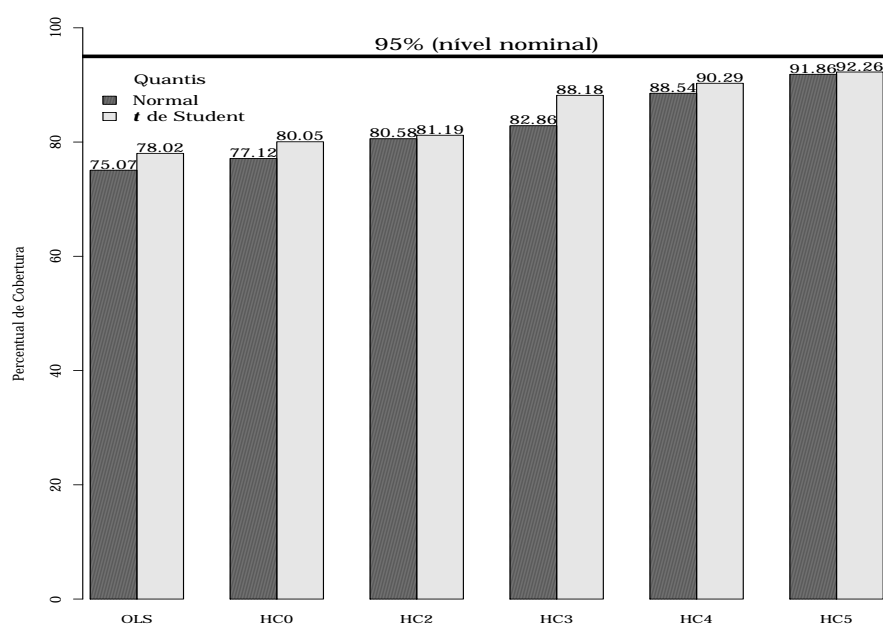


Figura 3 – Coberturas dos estimadores intervalares (sem uso de bootstrap) utilizando os quantis da distribuição normal padrão e da distribuição t de Student com $n - 2$ graus de liberdade - $n = 20$, $\lambda \approx 49$, erros t de Student e desenho não balanceado.

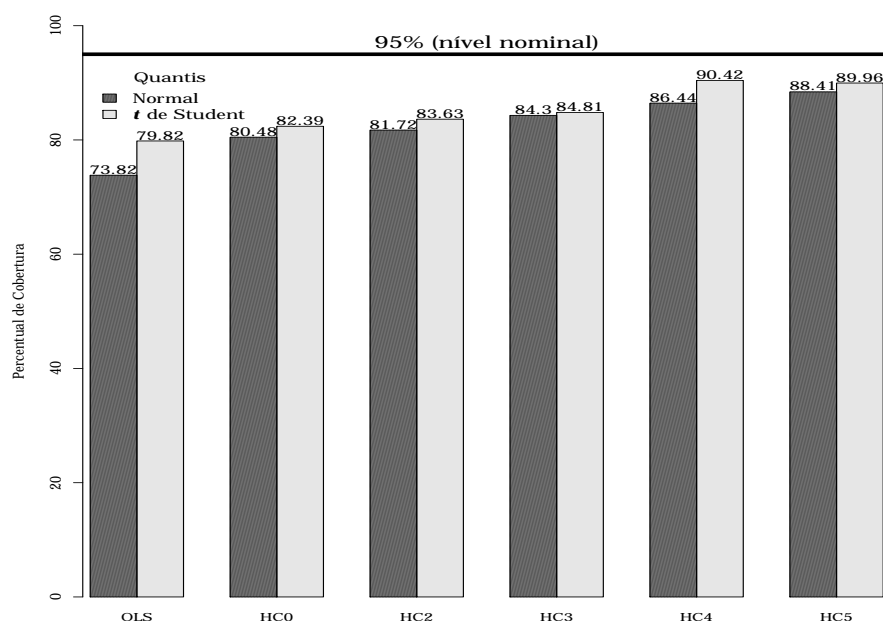


Figura 4 – Coberturas dos estimadores intervalares (sem uso de bootstrap) utilizando quantis da distribuição normal padrão e da distribuição t de Student com $n - 2$ graus de liberdade - $n = 20$, $\lambda \approx 49$, erros qui-quadrado e desenho não balanceado.

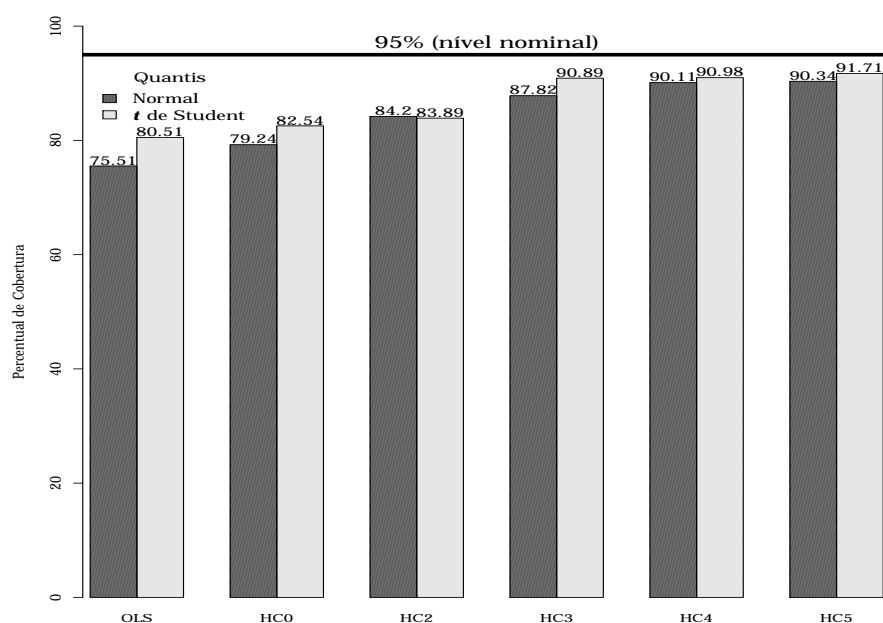


Figura 5 – Coberturas dos estimadores intervalares (sem uso de bootstrap) utilizando quantis da distribuição normal padrão e da distribuição t de Student com $n - 2$ graus de liberdade - $n = 20$, $\lambda \approx 49$, erros Weibull e desenho não balanceado.

Uma forma que deixa mais evidente que o uso dos quantis da distribuição $t_{(n-2)}$, em alguns casos, poderá ser mais interessante que o de quantis normais é utilizando as estatísticas de Cramér-von Mises e Anderson-Darling propostas por [Chen e Balakrishnan \(1995\)](#), estatísticas estas construídas com base nas sugestões de [Stephens \(1986\)](#). Utilizamos essas estatísticas quando temos uma amostra aleatória $x_n = \{x_1, \dots, x_n\}$ com função de distribuição empírica $F_n(x)$ e queremos testar se a amostra vem de uma determinada distribuição. As estatísticas de Cramér-von Mises (A^*) e Anderson-Darling (W^*) são dadas, respectivamente, por

$$W^* = \left\{ n \int_{-\infty}^{+\infty} \{F_n(x) - F(x; \hat{\theta}_n)\}^2 dF(x; \hat{\theta}_n) \right\} \left(1 + \frac{0.5}{n} \right) = W^2 \left(1 + \frac{0.5}{n} \right), \quad (4.6)$$

$$\begin{aligned} A^* &= \left\{ n \int_{-\infty}^{+\infty} \frac{\{F_n(x) - F(x; \hat{\theta}_n)\}^2}{\{F(x; \hat{\theta}_n)(1 - F(x; \hat{\theta}_n))\}} dF(x; \hat{\theta}_n) \right\} \left(1 + \frac{0.75}{n} + \frac{2.25}{n^2} \right) \\ &= A^2 \left(1 + \frac{0.75}{n} + \frac{2.25}{n^2} \right), \end{aligned} \quad (4.7)$$

em que $F_n(x)$ é a função de distribuição empírica, $F(x; \hat{\theta}_n)$ é a função de distribuição que postulamos avaliada no estimador de máxima verossimilhança $\hat{\theta}_n$ de θ , onde W^2 e A^2 são as estatísticas de Cramér-von Mises e Anderson-Darling, respectivamente. Para maiores detalhes sobre as estatísticas W^2 e A^2 veja [Cramér \(1928\)](#), [von Mises \(1931\)](#) e [Anderson e Darling \(1952\)](#). Note que as estatísticas W^* e A^* são dadas pela diferença entre $F_n(x)$ e $F(x; \hat{\theta}_n)$. Assim, quanto menor forem as estatísticas W^* e A^* mais evidências teremos que $F(x; \hat{\theta}_n)$ gerou a amostra. A hipótese nula testada usando as estatísticas (4.6) e (4.7) é de que a amostra aleatória x_1, \dots, x_n segue distribuição $F(x; \theta)$. Segundo [Chen e Balakrishnan \(1995, p. 155\)](#), as estatísticas A^2 e W^2 podem ser calculadas facilmente como

$$W^2 = \sum_{i=1}^n [u_i - \{(2i-1)/(2n)\}]^2 + 1/(12n) \quad (4.8)$$

e

$$A^2 = -n - n^{-1} \sum_{i=1}^n \{(2i-1)\ln(u_i) + (2n+1-2i)\ln(1-u_i)\}, \quad (4.9)$$

em que $u_i = \Phi((y_i - \bar{y})/s_y)$, onde $v_i = F(x_i; \hat{\theta}_n)$, $y_i = \Phi^{-1}(v_i)$ (Φ é a função de distribuição acumulada normal padrão) e s_y o desvio padrão amostral de y_i , com $i = 1, 2, \dots, n$. O algoritmo abaixo detalha a obtenção das estatísticas W^* e A^* .

1. Estime θ por $\hat{\theta}_n$ (de forma consistente), ordene de forma crescente os valores da amostra e calcule $v_i = F(x_i; \hat{\theta}_n)$. Aqui também podemos prefixar uma distribuição, caso haja suspeita que cada elemento da amostra foi gerado por essa distribuição;
2. Calcule $y_i = \Phi^{-1}(v_i)$, em que Φ é a função de distribuição acumulada normal padrão e Φ^{-1} a função quantílica normal padrão;

3. Calcule $u_i = \Phi\{(y_i - \bar{y})/s_y\}$, em que $\bar{y} = n^{-1} \sum_{i=1}^n y_i$ e $s_y^2 = (n-1)^{-1} \sum_{i=1}^n (y_i - \bar{y})^2$;
4. Calcule W^2 e A^2 utilizando as equações (4.8) e (4.9), respectivamente;
5. Obtenha as quantidades $W^* = W^2(1 + 0.5/n)$ e $A^* = A^2(1 + 0.75/n + 2.25/n^2)$, em que n é o tamanho da amostra.
6. Rejeitamos \mathcal{H}_0 ao nível de significância α se as estatísticas de teste excedem os valores críticos apresentados por [Chen e Balakrishnan \(1995, p. 155\)](#).

O teste é realizado com base em valores críticos estabelecidos por [Chen e Balakrishnan \(1995, p. 155\)](#) para diferentes níveis de significância para as estatísticas W^* e A^* , respectivamente. Isso se deve ao fato de não conhecermos as distribuições exatas das estatísticas de teste. Para $\alpha = 0.05$, temos que os valores críticos para W^* e A^* são, respectivamente, 0.1260 e 0.7520.

O que é comumente feito na prática é utilizar W^* e A^* para comparar duas ou mais distribuições de probabilidade. As distribuições de probabilidade que fornecem os valores menores de W^* ou A^* são as que melhor se adequam à amostra aleatória em mãos. Inúmeros artigos na área de distribuições de probabilidade utilizam as estatísticas (4.6) e (4.7) para comparar diferentes distribuições e escolhem a distribuição que fornece os menores valores de A^* e W^* como a distribuição que melhor se ajusta (distribuição mais adequada) ao conjunto de dados do pesquisador. Entre esses artigos podemos citar [Ramos, Cordeiro e Marinho \(2013\)](#), [Ramos, Marinho e Silva \(2013\)](#) e [Lemonte \(2013\)](#).

Para este trabalho foi observado que, em geral, os intervalos de confiança HC0, HC2, HC3, HC4 e HC5 apresentaram melhores coberturas quando foram utilizados quantis $t_{(n-2)}$ do que quando foram usados quantis normais sob desenho não balanceado. Talvez isso tenha ocorrido nos casos considerados nesse estudo porque as quantidades assintoticamente pivotais utilizadas para construção dos intervalos de confiança em para pequenas amostras, são melhor aproximadas pela distribuição $t_{(n-2)}$ na presença de pontos de alta alavancagem (desenho não balanceado).

Consideremos o caso do intervalo de confiança HC4. Para a m -ésima réplica de Monte Carlo ($m = 1, 2, \dots, 10000$), as quantidades $b_m = (\hat{\beta}_{2(m)} - 1)/\sqrt{\hat{\Psi}_{22(m)}^{(k)}}$, com $k = 4$, deram origem a uma amostra de 10000 observações. Assim, buscou-se verificar o nível de ajustamento da distribuição normal padrão e da distribuição $t_{(n-2)}$ a tal amostra. As Tabelas 15 e 16 apresentam as estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para o caso de homoscedasticidade ($\lambda = 1$) e heteroscedasticidade forte ($\lambda \approx 49$) considerando o nível nominal de cobertura de 95% para o intervalo de confiança HC4 e erros obtidos das quatro distribuições apresentadas na Figura 1 para o desenho balanceado. Na Tabela 15 foi testada a hipótese de que $b_m, m = 1, 2, \dots, 10000$, segue

distribuição normal padrão; já na Tabela 16 a hipótese \mathcal{H}_0 é de que $b_m, m = 1, 2, \dots, 10000$, segue distribuição $t_{(n-2)}$. As Tabelas 17 e 18 apresentam os mesmos cenários das Tabelas 15 e 16 considerando dados com pontos de alta alavancagem (desenho não balanceado). Para o cálculo das estatísticas A^* e W^* foi utilizado o pacote `AdequacyModel` versão 1.0.5 para linguagem R.

Como pode-se observar nas Tabelas 15 e 16, em que as hipóteses nulas testadas são de que $b_m, m = 1, 2, \dots, 10000$, segue distribuição normal padrão e distribuição $t_{(n-2)}$, respectivamente, houve várias situações em que ambas as hipóteses nulas foram rejeitadas ao nível de significância de 5% ($W^* > 0.1260$ e $A^* > 0.7520$). As situações em que não rejeitamos \mathcal{H}_0 estão destacadas nas Tabelas 15 e 16. O fato de termos que b_m não segue distribuição normal padrão nem distribuição $t_{(n-2)}$ poderá comprometer a acurácia das estimativas intervalares em pequenas amostras, situações estas em que não podemos garantir escolhas adequadas do quantil que fornecerá um intervalo com nível de confiança desejado.

Em diversas situações consideradas para o caso não balanceado, o uso de quantis normais pode não ser conveniente para o cálculo das estimativas intervalares HC4. Esse fato pode ser observado na Tabela 17. Note que as estatísticas W^* e A^* são menores quando testamos que b_m segue distribuição $t_{(n-2)}$ em comparação às estatísticas obtidas quando testamos que b_m segue distribuição normal padrão (caso não balanceado). É importante notar que o fato da distribuição se ajustar melhor aos dados não necessariamente garante que os dados seguem essa distribuição.

Assim, em situações em que o desenho é não balanceado (dados com pontos de alta alavancagem), é preferível construir intervalo de confiança HC4 usando quantis da distribuição $t_{(n-2)}$, como pode ser observado na Tabela 18. Esse fato talvez justifique as melhores coberturas obtidas quando são considerados os quantis da distribuição $t_{(n-2)}$ nos casos não balanceados apresentados nas Figuras 2 a 5. As estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para a quantidades assintoticamente pivotal $b_m = (\hat{\beta}_j - 1)/\sqrt{\hat{\Psi}_{jj}^{(k)}}$, com $k = 0, 2, 3, 5$ que são utilizadas para construção dos intervalos HC0, HC2, HC3 e HC5, estão presentes no Apêndice A. Nesses casos, também observou-se uma melhoria nas estimativas intervalares ao utilizar os quantis da distribuição $t_{(n-2)}$ quando os dados possuíam pontos de alta alavancagem. Um outro motivo de observarmos coberturas um pouco melhores quando utilizamos os quantis $t_{(n-2)}$ é o fato dessa distribuição possuir caudas mais pesadas do que as da distribuição normal, proporcionando assim, intervalos de confiança mais amplos.

Contudo, em pequenas amostras ($n = 20, 60, 100$), as estimativas intervalares HC0, HC2, HC3, HC4 e HC5 podem apresentar coberturas menores do que os níveis de confiança desejados. Observamos anteriormente, através de simulações de Monte Carlo, que o

bootstrap percentil duplo para dados balanceados fornece percentuais de cobertura muito próximos aos níveis nominais considerados. Observamos também que para o caso não balanceado, o intervalo de confiança HC4 com quantis $t_{(n-2)}$ possui boa cobertura. Nos casos em que não há pontos de alavanca nos dados a melhor solução é construir estimativas intervalares via bootstrap duplo percentil e em casos de dados com pontos de alta alavancagem o ideal é usar $\hat{\beta}_2 \pm t_{1-\alpha/2} \sqrt{\Psi_{22}^{(4)}}$, em que $t_{1-\alpha/2}$ é o quantil $1 - \alpha/2$ da distribuição $t_{(n-2)}$.

Ótimos resultados foram obtidos utilizando bootstrap- t duplo. Observou-se melhorias significativas dos níveis de coberturas com um segundo nível de bootstrap. Pode-se observar nas Tabelas 19 a 30 ganhos significativos no percentual de cobertura principalmente em relação aos estimadores HC3, HC4 e HC5 quando comparados os percentuais de coberturas do método bootstrap- t simples com os percentuais do bootstrap- t duplo, com exceção do caso de homoscedasticidade ($\lambda = 1$). Para esses casos, os percentuais de cobertura do bootstrap- t simples foram muito próximos aos percentuais obtidos com o bootstrap- t duplo.

Bons níveis de cobertura via bootstrap- t duplo ocorreram nos diversos cenários considerados nesse estudo, ou seja, foram observados percentuais de cobertura próximos dos níveis nominais considerados nesse estudo em caso de dados balanceados e não balanceados e para $\lambda \approx 9$ e $\lambda \approx 49$. Também foram observados ótimos níveis de coberturas do método bootstrap- t duplo independente da distribuição dos erros. Em geral, os intervalos de confiança bootstrap- t duplo HC4 e os intervalos bootstrap- t duplo HC5 foram os que apresentaram os melhores percentuais de cobertura.

Cada linha das Tabelas 19 a 30 apresenta o respectivo tempo (em horas) de simulação. Como mencionado anteriormente, foram consideradas 72 simulações, em que cada uma delas avaliou os estimadores OLS, HC0, HC2, HC3, HC4 e HC5, bem como os estimadores bootstrap percentil e bootstrap- t HC0, HC2, HC3, HC4 e HC5 em esquemas simples e duplos para três níveis de cobertura nominal (90%, 95% e 99%). Assim, por exemplo, a simulação apresentada na primeira linha da Tabela 19 durou 7.4408 horas. Contudo, esse tempo não se refere apenas às avaliações dos estimadores intervalares via bootstrap- t simples e duplo para o nível de confiança de 90%, mas também ao tempo das avaliações de Monte Carlo dos estimadores OLS, HC0, HC2, HC3, HC4 e HC5, para os três níveis de confiança considerados. Porém, a parte do código responsável pelas avaliações dos estimadores intervalares sem uso de bootstrap não acrescentou muito ao tempo de processamento, pois a mesma réplica de Monte Carlo que foi utilizada para avaliar as estimativas intervalares OLS, HC0, HC2, HC3, HC4, HC5 foi utilizada para avaliar as estimativas intervalares via bootstrap percentil e bootstrap- t em esquemas simples e duplo. Dessa forma, o tempo apresentado em cada linha das Tabelas 19 a 30 é o tempo

que reflete o custo computacional da simulação correspondente.

Tabela 15 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos balanceados verificando o ajuste da distribuição normal padrão à amostra $b_m, m = 1, 2, \dots, 10000$.

λ	Distribuições	n					
		20		60		100	
		W^*	A^*	W^*	A^*	W^*	A^*
$\lambda = 1$	$\mathcal{N}(0, 1)$	0.8192	5.4930	0.1390	0.9427	0.0908	0.6409
	$t_{(3)}$	0.0856	0.6159	0.0631	0.4410	0.1745	0.9963
	$\chi^2_{(2)}$	0.0655	0.5663	0.1551	0.8796	0.0139	0.1139
	Weibull(2, 3)	0.3934	2.5595	0.1557	1.1073	0.0502	0.3826
$\lambda \approx 49$	$\mathcal{N}(0, 1)$	0.8419	5.2308	0.2527	1.7073	0.1010	0.7397
	$t_{(3)}$	0.1853	1.0272	0.0948	0.5743	0.1167	0.6823
	$\chi^2_{(2)}$	10.3933	62.1225	4.5853	28.4372	3.233	19.7577
	Weibull(2, 3)	0.3972	2.7276	0.2255	1.5016	0.0923	0.6306

1 - $\mathcal{H}_0 : b_1, b_2, \dots, b_{10000}$ segue distribuição normal padrão;

2 - Os resultados aqui apresentados referem-se ao caso balanceado.

Tabela 16 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos balanceados verificando o ajuste da distribuição t de Student com $n - 2$ graus de liberdade à amostra $b_m, m = 1, 2, \dots, 10000$.

λ	Distribuições	n					
		20		60		100	
		W^*	A^*	W^*	A^*	W^*	A^*
$\lambda = 1$	$\mathcal{N}(0, 1)$	0.8153	5.4676	0.1380	0.9359	0.0902	0.6363
	$t_{(3)}$	0.0861	0.6171	0.0637	0.4441	0.0753	0.7138
	$\chi^2_{(2)}$	0.0655	0.5647	0.1551	0.8787	0.0139	0.1136
	Weibull(2, 3)	0.3910	2.5444	0.1550	1.1020	0.0497	0.3796
$\lambda \approx 49$	$\mathcal{N}(0, 1)$	0.8378	5.2057	0.2511	1.6972	0.1003	0.7352
	$t_{(3)}$	0.1448	1.0217	0.0955	0.5781	0.1173	0.6860
	$\chi^2_{(2)}$	10.3723	62.0008	4.5771	28.3867	3.2260	19.7172
	Weibull(2, 3)	0.3948	2.7113	0.2241	1.4922	0.0919	0.6272

1 - $\mathcal{H}_0 : b_1, b_2, \dots, b_{10000}$ segue distribuição $t_{(n-2)}$;

2 - Os resultados aqui apresentados referem-se ao caso balanceado.

Tabela 17 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos não balanceados verificando o ajuste da distribuição normal padrão à amostra $b_m, m = 1, 2, \dots, 10000$.

λ	Distribuições	n					
		20		60		100	
		W^*	A^*	W^*	A^*	W^*	A^*
$\lambda = 1$	$\mathcal{N}(0, 1)$	5.5544	32.2310	2.3182	14.6789	0.5458	3.8828
	$t_{(3)}$	7.5716	45.9578	0.1855	1.5457	0.1327	0.8630
	$\chi^2_{(2)}$	33.1663	176.5681	3.7669	21.7054	3.5280	19.7087
	Weibull(2, 3)	35.1834	185.7718	2.5147	14.8889	0.7020	4.5678
$\lambda \approx 49$	$\mathcal{N}(0, 1)$	8.3452	94.1321	1.6170	9.4934	0.3941	2.3759
	$t_{(3)}$	14.4137	100.0711	2.6641	16.0826	1.6712	9.6067
	$\chi^2_{(2)}$	125.3059	159.6019	3.5805	21.2443	2.8540	17.2831
	Weibull(2, 3)	8.7274	63.6751	1.750119	10.8774	0.7515	4.5005

1 - $\mathcal{H}_0 : b_1, b_2, \dots, b_{10000}$ segue distribuição normal padrão;

2 - Os resultados aqui apresentados referem-se ao caso não balanceado.

Tabela 18 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos não balanceados verificando o ajuste da distribuição t de Student com $n - 2$ graus de liberdade à amostra $b_m, m = 1, 2, \dots, 10000$.

λ	Distribuições	n					
		20		60		100	
		W^*	A^*	W^*	A^*	W^*	A^*
$\lambda = 1$	$\mathcal{N}(0, 1)$	1.5089	5.8509	0.3120	4.6398	0.5434	3.8665
	$t_{(3)}$	1.0412	2.6182	0.1212	0.5382	0.1032	0.5648
	$\chi^2_{(2)}$	13.1326	50.3925	1.7632	11.6840	0.5260	1.6964
	Weibull(2, 3)	12.1900	43.5463	2.5076	14.9461	0.2993	1.5500
$\lambda \approx 49$	$\mathcal{N}(0, 1)$	8.3410	30.4536	1.6189	9.5052	0.3952	2.3834
	$t_{(3)}$	14.3995	94.5243	2.6664	16.0976	1.6735	9.6210
	$\chi^2_{(2)}$	10.2488	51.2828	3.5828	10.2589	1.8548	7.2889
	Weibull(2, 3)	8.7169	63.6314	0.7520	1.8752	0.7531	4.5106

1 - $\mathcal{H}_0 : b_1, b_2, \dots, b_{10000}$ segue distribuição $t_{(n-2)}$;

2 - Os resultados aqui apresentados referem-se ao caso não balanceado.

Tabela 19 – Percentuais de cobertura das estimativas intervalares bootstrap- t simples e bootstrap- t duplo com erros normais - nível nominal de 90%.

n	Desenho	λ	HC0		HC2		HC3		HC4		HC5		Tempo
			Simple	Duplo	Simple	Duplo	Simple	Duplo	Simple	Duplo	Simple	Duplo	
20	Balanceado	$\lambda = 1$	81.52	85.84	82.48	85.86	83.60	87.90	86.72	90.00	86.52	89.80	7.4408
		$\lambda \approx 9$	79.56	83.84	79.56	87.82	81.76	88.88	83.54	87.96	83.58	88.84	7.3422
		$\lambda \approx 49$	79.48	85.86	82.44	88.78	83.32	88.72	84.32	89.79	85.42	88.78	7.3923
	Não Balanceado	$\lambda = 1$	81.42	80.14	80.06	83.77	79.98	84.44	81.18	88.58	83.68	87.36	7.3606
		$\lambda \approx 9$	81.60	85.08	81.40	87.50	83.44	86.26	84.94	89.95	86.10	89.74	7.3428
		$\lambda \approx 49$	80.64	84.10	83.76	86.71	86.84	89.02	86.50	90.07	83.12	90.66	7.3906
60	Balanceado	$\lambda = 1$	87.42	90.62	84.42	90.60	90.46	90.56	90.44	90.54	90.44	90.58	55.2536
		$\lambda \approx 9$	83.26	86.34	80.26	87.34	85.39	88.27	88.26	90.00	87.58	90.30	54.8126
		$\lambda \approx 49$	80.74	86.22	84.04	85.22	87.02	88.20	88.72	90.07	88.02	90.24	56.1188
	Não Balanceado	$\lambda = 1$	80.26	83.78	80.00	85.91	84.26	88.26	85.74	90.09	86.85	90.40	56.8720
		$\lambda \approx 9$	81.94	84.72	85.88	86.68	85.74	88.78	89.66	89.77	88.12	90.76	56.1946
		$\lambda \approx 49$	84.88	84.06	84.92	86.14	87.68	88.42	87.72	91.62	86.94	93.50	56.0587
100	Balanceado	$\lambda = 1$	86.58	86.52	84.58	85.52	86.60	86.56	90.60	90.54	90.60	90.52	220.5600
		$\lambda \approx 9$	80.60	81.85	80.58	84.58	85.58	85.90	89.58	90.54	90.58	90.58	222.1320
		$\lambda \approx 49$	80.22	82.30	81.11	83.68	85.22	86.28	90.22	90.28	90.22	90.28	222.9650
	Não Balanceado	$\lambda = 1$	84.94	89.42	88.96	89.40	89.00	89.48	89.20	89.99	89.08	89.46	222.3550
		$\lambda \approx 9$	82.68	84.52	80.66	86.50	84.60	89.42	85.54	89.78	87.56	89.42	221.8420
		$\lambda \approx 49$	82.12	85.80	81.98	86.72	85.88	88.64	86.58	87.89	85.19	86.56	221.3730

1 - Os tempos marcados estão apresentados em horas;

2 - As simulações foram realizadas utilizando o código em C++ apresentado no Apêndice B;

3 - Foram utilizadas 10000 réplicas de Monte Carlo, $J = 1000$ e $K = 500$;

4 - As simulações foram realizadas no cluster SIG Altix (cluster Gauss do CESUP).

Tabela 20 – Percentuais de cobertura das estimativas intervalares bootstrap- t simples e bootstrap- t duplo com erros normais - nível nominal de 95%.

n	Desenho	λ	HC0		HC2		HC3		HC4		HC5		Tempo
			Simple	Duplo	Simple	Duplo	Simple	Duplo	Simple	Duplo	Simple	Duplo	
20	Balanceado	$\lambda = 1$	92.12	93.34	92.08	94.44	94.06	94.40	94.06	94.48	94.08	94.44	7.4408
		$\lambda \approx 9$	88.88	91.12	89.86	94.14	91.88	94.12	93.86	95.07	92.88	94.10	7.3422
		$\lambda \approx 49$	89.44	92.78	90.38	92.80	90.44	92.82	91.44	93.01	92.40	93.46	7.3923
	Não Balanceado	$\lambda = 1$	84.66	86.46	83.74	85.78	85.22	91.78	92.72	94.02	93.58	95.16	7.3606
		$\lambda \approx 9$	83.96	86.24	82.76	89.44	87.50	90.61	89.78	95.04	93.86	95.00	7.3428
		$\lambda \approx 49$	84.86	86.64	87.16	89.16	89.42	93.48	90.74	94.94	93.92	95.12	7.3906
60	Balanceado	$\lambda = 1$	87.44	92.46	86.40	91.48	88.38	94.48	93.38	94.98	94.38	95.21	55.2536
		$\lambda \approx 9$	84.98	87.06	84.92	90.06	90.93	93.04	94.94	94.97	94.91	95.13	54.8126
		$\lambda \approx 49$	86.86	88.94	86.84	92.94	92.84	93.94	92.84	94.99	91.84	94.93	56.1188
	Não Balanceado	$\lambda = 1$	83.22	85.66	82.24	84.76	90.40	92.60	91.58	94.96	93.52	93.86	56.8720
		$\lambda \approx 9$	81.58	85.36	81.60	84.43	85.60	91.22	90.14	95.08	92.50	95.20	56.1946
		$\lambda \approx 49$	80.31	83.92	80.68	87.98	87.76	91.06	92.84	95.14	93.76	96.18	56.0587
100	Balanceado	$\lambda = 1$	89.06	90.14	87.06	89.18	93.06	95.14	94.06	95.14	95.06	95.14	220.5600
		$\lambda \approx 9$	87.48	84.84	85.94	84.84	88.94	90.67	92.94	94.88	94.94	95.47	222.1320
		$\lambda \approx 49$	86.00	88.82	90.00	93.82	93.00	93.82	92.00	95.00	93.00	94.82	222.9650
	Não Balanceado	$\lambda = 1$	84.74	87.10	83.74	87.10	90.74	94.18	95.11	93.24	94.80	94.12	222.3550
		$\lambda \approx 9$	84.68	85.84	81.68	88.78	89.56	92.74	93.47	94.86	94.48	95.58	221.8420
		$\lambda \approx 49$	80.68	83.36	83.56	89.40	90.44	92.44	92.30	95.00	92.42	95.14	221.3730

1 - Os tempos marcados estão apresentados em horas;

2 - As simulações foram realizadas utilizando o código em C++ apresentado no Apêndice B;

3 - Foram utilizadas 10000 réplicas de Monte Carlo, $J = 1000$ e $K = 500$;

4 - As simulações foram realizadas no cluster SIG Altix (cluster Gauss do CESUP).

Tabela 21 – Percentuais de cobertura das estimativas intervalares bootstrap- t simples e bootstrap- t duplo com erros normais - nível nominal de 99%.

n	Desenho	λ	HC0		HC2		HC3		HC4		HC5		Tempo
			Simple	Duplo	Simple	Duplo	Simple	Duplo	Simple	Duplo	Simple	Duplo	
20	Balanceado	$\lambda = 1$	86.26	87.26	83.24	87.18	88.28	92.12	94.28	98.87	93.24	98.16	7.4408
		$\lambda \approx 9$	83.26	87.24	86.24	92.20	93.28	92.16	94.24	99.14	94.24	98.18	7.3422
		$\lambda \approx 49$	85.08	88.20	86.04	88.14	92.04	98.12	93.02	97.60	95.04	98.14	7.3923
	Não Balanceado	$\lambda = 1$	88.44	87.42	87.70	88.08	87.98	89.02	94.00	98.44	94.60	95.02	7.3606
		$\lambda \approx 9$	86.10	94.64	89.30	92.68	91.00	94.74	93.22	99.06	94.86	99.92	7.3428
		$\lambda \approx 49$	85.78	88.06	88.54	94.04	91.38	96.00	92.78	99.02	94.22	97.92	7.3906
60	Balanceado	$\lambda = 1$	90.76	95.86	90.78	95.76	91.49	98.08	95.80	99.01	94.78	98.86	55.2536
		$\lambda \approx 9$	85.84	90.81	83.84	87.12	86.42	94.82	93.82	98.82	93.94	98.82	54.8126
		$\lambda \approx 49$	86.74	88.82	85.74	88.82	88.72	98.80	93.70	98.89	98.72	98.86	56.1188
	Não Balanceado	$\lambda = 1$	87.22	90.90	87.22	92.92	93.22	96.84	94.26	99.08	95.18	96.90	56.8720
		$\lambda \approx 9$	84.94	87.74	81.88	84.68	90.86	95.64	93.78	98.90	96.92	98.52	56.1946
		$\lambda \approx 49$	85.82	89.68	84.80	87.77	93.72	94.64	95.74	99.58	97.72	99.62	56.0587
100	Balanceado	$\lambda = 1$	88.80	91.74	85.80	87.74	92.80	95.74	94.80	98.74	94.80	98.49	220.5600
		$\lambda \approx 9$	84.90	88.84	83.90	84.84	94.90	94.84	96.99	99.00	97.91	98.84	222.1320
		$\lambda \approx 49$	98.74	98.88	98.74	98.88	98.74	98.88	98.74	98.88	98.74	98.88	222.9650
	Não Balanceado	$\lambda = 1$	92.90	93.12	90.94	96.14	94.96	98.14	95.00	98.16	96.70	97.20	222.3550
		$\lambda \approx 9$	85.84	88.86	87.82	83.84	88.80	91.81	94.84	99.36	94.74	98.80	221.8420
		$\lambda \approx 49$	80.56	83.56	82.54	87.67	90.52	96.54	95.52	99.07	96.52	99.58	221.3730

1 - Os tempos marcados estão apresentados em horas;

2 - As simulações foram realizadas utilizando o código em C++ apresentado no Apêndice B;

3 - Foram utilizadas 10000 réplicas de Monte Carlo, $J = 1000$ e $K = 500$;

4 - As simulações foram realizadas no cluster SIG Altix (cluster Gauss do CESUP).

Tabela 22 – Percentuais de cobertura das estimativas intervalares bootstrap- t simples e bootstrap- t duplo com erros t de Student - nível nominal de 90%.

n	Desenho	λ	HC0		HC2		HC3		HC4		HC5		Tempo
			Simple	Duplo	Simple	Duplo	Simple	Duplo	Simple	Duplo	Simple	Duplo	
20	Balanceado	$\lambda = 1$	86.00	88.17	84.02	87.86	88.14	87.80	90.19	88.90	89.10	90.80	7.3946
		$\lambda \approx 9$	84.52	87.52	83.50	87.49	85.56	90.44	88.60	90.08	86.54	90.38	7.3811
		$\lambda \approx 49$	84.00	86.96	82.98	85.98	86.90	90.98	88.94	89.86	89.94	89.94	7.4061
	Não Balanceado	$\lambda = 1$	84.46	86.28	83.58	87.74	82.62	85.86	86.00	90.11	86.18	88.92	7.3308
		$\lambda \approx 9$	78.42	84.34	84.36	87.38	83.84	85.56	86.30	89.88	86.12	87.32	7.4995
		$\lambda \approx 49$	83.46	85.04	83.88	85.30	86.60	88.44	87.58	90.08	86.14	90.26	7.5976
60	Balanceado	$\lambda = 1$	80.28	84.32	79.24	82.34	83.28	86.28	85.30	89.78	87.26	90.32	55.0695
		$\lambda \approx 9$	83.12	84.06	84.10	86.04	86.10	90.06	85.10	90.04	88.10	90.04	56.7594
		$\lambda \approx 49$	80.14	82.24	78.14	82.24	84.14	90.24	87.14	90.00	87.14	90.24	55.9323
	Não Balanceado	$\lambda = 1$	81.02	83.70	83.10	85.79	81.30	89.98	90.74	90.01	86.32	88.08	55.9530
		$\lambda \approx 9$	82.66	84.76	80.08	83.08	82.00	88.14	83.79	90.16	87.90	90.11	55.8338
		$\lambda \approx 49$	80.30	83.40	82.38	83.44	85.24	87.42	87.10	90.06	88.04	90.61	55.8939
100	Balanceado	$\lambda = 1$	85.92	86.64	84.90	86.69	88.90	90.02	88.90	91.03	89.90	90.62	221.8200
		$\lambda \approx 9$	80.10	84.16	80.10	86.16	87.10	88.14	86.06	90.04	87.10	90.17	219.9650
		$\lambda \approx 49$	79.10	83.78	81.10	85.78	85.10	89.78	88.10	89.89	86.10	89.05	222.5220
	Não Balanceado	$\lambda = 1$	83.76	86.16	81.78	86.18	86.90	88.22	87.10	90.22	89.96	91.14	221.9190
		$\lambda \approx 9$	80.01	83.54	81.98	85.66	85.96	87.64	87.17	89.99	87.82	90.68	212.6090
		$\lambda \approx 49$	79.02	83.46	79.92	84.36	85.90	87.24	88.52	89.97	87.78	90.20	212.8480

1 - Os tempos marcados estão apresentados em horas;

2 - As simulações foram realizadas utilizando o código em C++ apresentado no Apêndice B;

3 - Foram utilizadas 10000 réplicas de Monte Carlo, $J = 1000$ e $K = 500$;

4 - As simulações foram realizadas no cluster SIG Altix (cluster Gauss do CESUP).

Tabela 23 – Percentuais de cobertura das estimativas intervalares bootstrap- t simples e bootstrap- t duplo com erros t de Student - nível nominal de 95%.

n	Desenho	λ	HC0		HC2		HC3		HC4		HC5		Tempo
			Simple	Duplo	Simple	Duplo	Simple	Duplo	Simple	Duplo	Simple	Duplo	
20	Balanceado	$\lambda = 1$	83.46	85.48	82.40	85.48	91.40	92.46	90.93	95.04	92.08	95.12	7.3946
		$\lambda \approx 9$	82.06	85.22	83.10	84.24	90.18	90.18	92.18	94.88	90.14	95.26	7.3811
		$\lambda \approx 49$	79.96	83.86	85.86	85.86	84.84	89.90	92.88	95.00	92.98	94.92	7.4061
	Não Balanceado	$\lambda = 1$	81.64	85.56	88.32	91.60	94.10	95.06	93.80	94.98	93.14	95.06	7.3308
		$\lambda \approx 9$	82.66	87.10	81.82	86.58	87.72	92.16	90.72	95.20	91.24	93.98	7.4995
		$\lambda \approx 49$	80.76	83.49	81.16	85.76	85.54	91.10	92.48	94.72	93.08	94.64	7.5976
60	Balanceado	$\lambda = 1$	86.84	89.92	86.84	88.88	91.84	94.90	92.88	94.99	93.84	94.90	55.0695
		$\lambda \approx 9$	82.12	86.44	83.12	85.97	89.14	93.94	93.14	95.06	92.14	94.96	56.7594
		$\lambda \approx 49$	84.92	87.90	83.92	86.90	88.92	94.90	92.90	94.96	93.92	94.92	55.9323
	Não Balanceado	$\lambda = 1$	84.60	87.86	84.68	86.74	89.72	93.82	91.04	95.00	92.88	94.88	55.9530
		$\lambda \approx 9$	83.62	87.60	83.56	86.58	87.50	90.50	92.30	95.05	92.44	95.06	55.8338
		$\lambda \approx 49$	83.72	87.84	85.66	88.76	87.72	95.72	91.66	94.58	92.70	95.72	55.8939
100	Balanceado	$\lambda = 1$	85.52	86.38	83.52	85.40	91.52	94.40	93.54	95.00	93.52	95.41	221.8200
		$\lambda \approx 9$	82.48	85.36	85.48	87.36	89.48	95.34	93.48	95.34	93.88	95.51	219.9650
		$\lambda \approx 49$	82.10	84.78	82.10	85.78	90.10	93.78	91.11	94.78	90.10	95.79	222.5220
	Não Balanceado	$\lambda = 1$	85.36	88.66	90.42	92.72	85.42	92.72	90.62	95.14	92.50	95.68	221.9190
		$\lambda \approx 9$	82.88	85.80	85.88	90.53	91.84	95.68	93.72	95.08	94.84	95.05	212.6090
		$\lambda \approx 49$	83.28	86.22	84.26	86.20	90.14	93.08	93.04	95.18	93.18	95.22	212.8480

1 - Os tempos marcados estão apresentados em horas;

2 - As simulações foram realizadas utilizando o código em C++ apresentado no Apêndice B;

3 - Foram utilizadas 10000 réplicas de Monte Carlo, $J = 1000$ e $K = 500$;

4 - As simulações foram realizadas no cluster SIG Altix (cluster Gauss do CESUP).

Tabela 24 – Percentuais de cobertura das estimativas intervalares bootstrap- t simples e bootstrap- t duplo com erros t de Student - nível nominal de 99%.

n	Desenho	λ	HC0		HC2		HC3		HC4		HC5		Tempo
			Simple	Duplo	Simple	Duplo	Simple	Duplo	Simple	Duplo	Simple	Duplo	
20	Balanceado	$\lambda = 1$	83.00	85.76	84.98	86.76	93.96	97.74	96.88	99.16	95.98	98.76	7.3946
		$\lambda \approx 9$	85.96	88.88	83.96	87.90	90.00	95.90	94.98	98.94	95.98	98.90	7.3811
		$\lambda \approx 49$	84.54	86.87	85.58	90.87	92.60	97.80	96.62	97.96	95.58	98.82	7.4061
	Não Balanceado	$\lambda = 1$	85.66	89.50	83.12	89.84	92.32	95.80	92.04	99.00	90.56	99.00	7.3308
		$\lambda \approx 9$	81.64	84.54	83.94	86.38	91.76	94.63	95.32	99.20	94.90	99.06	7.4995
		$\lambda \approx 49$	79.06	82.32	83.78	87.40	91.52	97.38	97.58	99.03	97.66	99.48	7.5976
60	Balanceado	$\lambda = 1$	86.96	88.96	84.00	87.96	92.96	98.76	94.96	99.00	97.77	98.96	55.0695
		$\lambda \approx 9$	83.88	88.72	82.88	87.74	90.86	94.74	95.88	98.72	95.90	98.24	56.7594
		$\lambda \approx 49$	80.14	89.24	90.14	93.24	90.14	94.24	94.14	99.00	96.14	98.84	55.9323
	Não Balanceado	$\lambda = 1$	88.06	93.86	88.10	92.88	94.08	97.76	96.08	98.90	95.06	97.74	55.9530
		$\lambda \approx 9$	82.02	88.90	86.00	94.84	93.96	98.70	95.90	98.99	95.98	98.66	55.8338
		$\lambda \approx 49$	84.56	86.44	84.56	87.44	90.52	93.42	97.40	99.38	97.46	99.71	55.8939
100	Balanceado	$\lambda = 1$	88.98	90.96	86.00	88.96	98.00	98.98	98.70	98.98	98.00	99.06	221.8200
		$\lambda \approx 9$	85.98	88.78	84.98	82.98	86.98	92.98	94.98	99.00	97.00	98.99	219.9650
		$\lambda \approx 49$	82.88	85.84	85.88	87.84	93.88	98.84	94.88	98.84	96.88	98.14	222.5220
	Não Balanceado	$\lambda = 1$	80.98	90.08	90.02	95.10	98.00	99.03	97.14	99.02	96.98	99.04	221.9190
		$\lambda \approx 9$	84.50	87.46	82.50	86.46	95.54	95.46	97.94	99.10	96.13	99.16	212.6090
		$\lambda \approx 49$	80.78	84.72	80.74	87.72	93.74	97.12	97.72	99.00	96.35	99.10	212.8480

1 - Os tempos marcados estão apresentados em horas;

2 - As simulações foram realizadas utilizando o código em C++ apresentado no Apêndice B;

3 - Foram utilizadas 10000 réplicas de Monte Carlo, $J = 1000$ e $K = 500$;

4 - As simulações foram realizadas no cluster SIG Altix (cluster Gauss do CESUP).

Tabela 25 – Percentuais de cobertura das estimativas intervalares bootstrap- t simples e bootstrap- t duplo com erros qui-quadrado - nível nominal de 90%.

n	Desenho	λ	HC0		HC2		HC3		HC4		HC5		Tempo
			Simples	Duplo	Simples	Duplo	Simples	Duplo	Simples	Duplo	Simples	Duplo	
20	Balanceado	$\lambda = 1$	82.68	84.30	83.64	85.32	85.66	87.24	88.72	90.07	88.64	90.12	7.3532
		$\lambda \approx 9$	81.90	83.14	82.88	84.32	84.92	87.34	86.86	90.16	86.87	90.30	7.5438
		$\lambda \approx 49$	79.10	82.56	82.08	84.50	85.08	88.48	85.60	88.05	86.12	87.48	7.3637
	Não Balanceado	$\lambda = 1$	81.98	85.94	83.40	85.34	84.46	86.52	86.68	90.04	88.82	89.60	7.4816
		$\lambda \approx 9$	79.24	82.96	80.44	83.80	85.20	86.68	85.90	91.12	88.30	89.74	7.5684
		$\lambda \approx 49$	85.30	86.26	84.74	86.52	85.94	87.70	85.76	90.16	87.30	90.52	7.3189
60	Balanceado	$\lambda = 1$	82.30	85.60	85.30	88.62	86.30	88.23	87.28	90.12	88.30	90.14	55.2214
		$\lambda \approx 9$	83.96	84.92	84.96	86.90	87.96	90.92	88.94	90.00	89.96	90.51	55.9804
		$\lambda \approx 49$	79.90	82.88	80.90	81.88	85.96	88.83	87.88	90.05	88.91	90.00	56.8088
	Não Balanceado	$\lambda = 1$	84.82	86.12	84.74	86.08	85.74	87.19	86.80	90.02	87.84	90.12	55.8701
		$\lambda \approx 9$	82.74	84.16	85.66	88.16	85.60	87.08	87.42	90.00	88.52	90.18	55.9991
		$\lambda \approx 49$	81.32	84.27	83.15	87.21	86.88	89.43	87.00	88.03	87.84	88.12	56.2784
100	Balanceado	$\lambda = 1$	81.02	83.14	84.04	85.18	86.02	88.22	87.02	90.12	87.60	90.18	221.0050
		$\lambda \approx 9$	82.12	83.04	81.10	88.06	85.10	88.06	86.10	90.06	87.10	90.11	221.4560
		$\lambda \approx 49$	83.94	84.80	84.92	87.80	85.91	88.82	87.90	90.11	88.90	90.00	221.6140
	Não Balanceado	$\lambda = 1$	83.32	84.34	82.34	86.36	85.30	87.40	85.82	90.05	86.58	90.36	221.3460
		$\lambda \approx 9$	84.70	85.60	82.72	85.71	87.60	86.70	88.34	90.00	87.60	90.33	222.5300
		$\lambda \approx 49$	82.90	84.50	84.86	86.44	86.80	88.02	87.24	91.10	88.64	89.74	211.4460

1 - Os tempos marcados estão apresentados em horas;

2 - As simulações foram realizadas utilizando o código em C++ apresentado no Apêndice B;

3 - Foram utilizadas 10000 réplicas de Monte Carlo, $J = 1000$ e $K = 500$;

4 - As simulações foram realizadas no cluster SIG Altix (cluster Gauss do CESUP).

Tabela 26 – Percentuais de cobertura das estimativas intervalares bootstrap- t simples e bootstrap- t duplo com erros qui-quadrado - nível nominal de 95%.

n	Desenho	λ	HC0		HC2		HC3		HC4		HC5		Tempo
			Simples	Duplo	Simples	Duplo	Simples	Duplo	Simples	Duplo	Simples	Duplo	
20	Balanceado	$\lambda = 1$	83.82	84.16	85.86	85.10	88.92	93.14	92.84	95.12	93.82	95.14	7.3532
		$\lambda \approx 9$	80.84	83.74	85.92	86.76	88.96	93.80	92.10	95.05	92.94	93.74	7.5438
		$\lambda \approx 49$	84.64	85.72	85.68	88.66	89.70	90.66	91.70	93.68	93.64	94.15	7.3637
	Não Balanceado	$\lambda = 1$	83.38	84.84	84.06	88.48	91.40	84.62	92.72	95.12	94.00	95.08	7.4816
		$\lambda \approx 9$	85.06	88.34	89.24	91.78	91.70	92.43	90.56	95.06	91.30	95.03	7.5684
		$\lambda \approx 49$	85.68	87.58	88.98	90.46	92.40	94.94	93.22	95.00	93.04	95.00	7.3189
60	Balanceado	$\lambda = 1$	84.52	86.42	86.50	89.42	92.48	94.44	93.50	94.96	93.48	95.10	55.2214
		$\lambda \approx 9$	83.14	84.90	84.12	87.90	89.12	92.90	93.12	94.98	93.12	95.16	55.9804
		$\lambda \approx 49$	81.80	84.72	83.80	85.70	90.80	91.11	92.80	94.83	91.70	94.80	56.8088
	Não Balanceado	$\lambda = 1$	83.48	84.20	83.50	92.00	93.32	93.98	93.05	95.08	92.32	95.90	55.8701
		$\lambda \approx 9$	82.88	83.12	84.94	85.22	91.02	92.10	93.98	95.28	90.20	94.96	55.9991
		$\lambda \approx 49$	83.44	85.98	84.44	90.12	92.48	93.10	93.88	95.08	93.62	95.16	56.2784
100	Balanceado	$\lambda = 1$	82.02	84.14	83.02	85.14	91.02	93.14	92.02	95.03	93.02	94.71	221.0050
		$\lambda \approx 9$	83.92	84.92	85.92	87.92	90.92	94.92	94.92	95.14	93.92	95.92	221.4560
		$\lambda \approx 49$	82.66	84.58	85.66	86.58	90.64	91.58	92.64	92.58	90.66	95.08	221.6140
	Não Balanceado	$\lambda = 1$	83.58	85.62	87.66	91.62	88.64	92.62	92.58	94.58	93.62	95.29	221.3460
		$\lambda \approx 9$	83.90	86.20	84.00	86.24	87.06	90.24	91.10	95.06	93.08	95.28	222.5300
		$\lambda \approx 49$	87.50	88.64	90.42	92.58	91.44	94.60	93.40	94.96	93.40	94.69	211.4460

1 - Os tempos marcados estão apresentados em horas;

2 - As simulações foram realizadas utilizando o código em C++ apresentado no Apêndice B;

3 - Foram utilizadas 10000 réplicas de Monte Carlo, $J = 1000$ e $K = 500$;

4 - As simulações foram realizadas no cluster SIG Altix (cluster Gauss do CESUP).

Tabela 27 – Percentuais de cobertura das estimativas intervalares bootstrap- t simples e bootstrap- t duplo com erros qui-quadrado - nível nominal de 99%.

n	Desenho	λ	HC0		HC2		HC3		HC4		HC5		Tempo
			Simples	Duplo	Simples	Duplo	Simples	Duplo	Simples	Duplo	Simples	Duplo	
20	Balanceado	$\lambda = 1$	87.90	92.80	88.90	92.80	93.94	95.78	95.90	99.16	96.90	98.83	7.3532
		$\lambda \approx 9$	87.32	88.74	90.30	94.76	93.32	97.78	97.30	98.87	97.32	98.72	7.5438
		$\lambda \approx 49$	85.32	87.56	86.40	92.56	90.46	96.60	95.54	98.62	95.40	97.58	7.3637
	Não Balanceado	$\lambda = 1$	90.02	93.26	89.56	92.52	89.64	95.24	93.02	99.31	96.54	99.12	7.4816
		$\lambda \approx 9$	90.14	94.28	92.74	94.40	95.90	98.86	96.28	99.02	95.92	99.15	7.5684
		$\lambda \approx 49$	88.64	90.16	89.70	92.22	93.28	97.16	95.06	99.00	97.22	99.16	7.3189
60	Balanceado	$\lambda = 1$	88.82	90.06	92.82	94.06	95.82	97.04	97.82	99.04	97.12	99.06	55.2214
		$\lambda \approx 9$	87.58	90.80	93.56	95.76	94.56	97.76	96.56	98.89	96.56	98.76	55.9804
		$\lambda \approx 49$	86.56	92.80	86.38	93.82	93.54	96.82	97.54	99.08	96.54	99.32	56.8088
	Não Balanceado	$\lambda = 1$	88.36	91.62	87.14	93.56	95.10	97.48	96.76	98.94	97.01	98.42	55.8701
		$\lambda \approx 9$	85.44	90.36	92.48	94.32	93.50	94.24	94.32	98.84	95.44	99.24	55.9991
		$\lambda \approx 49$	85.64	89.62	90.64	92.62	93.60	94.60	97.64	99.21	98.63	99.52	56.2784
100	Balanceado	$\lambda = 1$	92.94	94.92	94.94	96.92	98.94	99.05	94.94	99.00	96.94	98.92	221.0050
		$\lambda \approx 9$	98.02	98.08	98.02	98.08	98.02	98.08	98.02	98.08	98.02	98.08	221.4560
		$\lambda \approx 49$	87.44	91.38	92.44	95.38	95.44	98.38	95.44	98.97	96.44	99.11	221.6140
	Não Balanceado	$\lambda = 1$	83.92	86.80	92.92	96.74	95.90	97.68	96.82	98.93	97.88	99.00	221.3460
		$\lambda \approx 9$	84.32	86.50	88.40	90.44	95.40	97.52	97.52	98.76	96.40	99.51	222.5300
		$\lambda \approx 49$	90.24	93.46	93.26	95.46	95.28	97.10	93.28	99.04	95.28	99.48	211.4460

1 - Os tempos marcados estão apresentados em horas;

2 - As simulações foram realizadas utilizando o código em C++ apresentado no Apêndice B;

3 - Foram utilizadas 10000 réplicas de Monte Carlo, $J = 1000$ e $K = 500$;

4 - As simulações foram realizadas no cluster SIG Altix (cluster Gauss do CESUP).

Tabela 28 – Percentuais de coberturas das estimativas intervalares bootstrap- t simples e bootstrap- t duplo com erros Weibull - nível nominal de 90%.

n	Desenho	λ	HC0		HC2		HC3		HC4		HC5		Tempo
			Simple	Duplo	Simple	Duplo	Simple	Duplo	Simple	Duplo	Simple	Duplo	
20	Balanceado	$\lambda = 1$	84.64	87.88	85.72	89.90	86.78	89.96	87.76	90.00	87.39	89.97	7.3920
		$\lambda \approx 9$	83.76	85.10	84.70	86.04	85.72	89.98	85.76	90.00	89.70	90.02	7.5388
		$\lambda \approx 49$	80.84	84.12	84.82	86.18	85.78	90.14	87.66	90.10	87.82	90.18	7.4891
	Não Balanceado	$\lambda = 1$	80.62	84.82	82.56	86.06	85.64	88.36	85.70	88.41	86.38	87.11	7.6432
		$\lambda \approx 9$	81.40	83.08	83.04	85.60	85.36	87.40	86.48	89.97	87.54	89.36	7.4351
		$\lambda \approx 49$	79.58	83.74	80.62	84.46	85.34	88.54	87.66	90.00	87.50	90.00	7.6669
60	Balanceado	$\lambda = 1$	83.54	85.62	84.54	87.62	86.54	88.17	87.54	90.12	89.56	90.58	55.8420
		$\lambda \approx 9$	82.46	83.51	83.44	86.65	88.44	89.19	89.00	90.04	88.56	90.46	55.1949
		$\lambda \approx 49$	81.56	83.32	82.18	84.89	85.46	87.53	88.54	89.79	88.68	90.37	55.4217
	Não Balanceado	$\lambda = 1$	84.28	85.06	85.48	86.26	85.64	89.42	88.98	89.92	88.60	89.54	55.7006
		$\lambda \approx 9$	80.92	82.70	84.70	85.62	85.64	86.54	86.42	90.07	87.44	90.04	55.1443
		$\lambda \approx 49$	78.76	81.86	82.74	84.96	84.60	86.96	87.36	90.26	88.52	91.69	55.4890
100	Balanceado	$\lambda = 1$	83.22	85.98	84.22	85.98	86.22	90.00	86.48	91.03	87.22	90.00	211.7410
		$\lambda \approx 9$	82.14	84.18	85.14	86.18	88.14	90.20	86.14	89.96	88.23	90.12	211.4520
		$\lambda \approx 49$	82.08	84.82	84.08	85.82	84.15	86.82	85.08	89.82	86.08	89.71	212.7690
	Não Balanceado	$\lambda = 1$	84.60	86.94	85.70	88.04	85.74	89.96	88.84	90.10	87.84	89.98	212.5570
		$\lambda \approx 9$	83.70	84.60	84.60	86.58	86.53	87.52	88.33	90.12	89.76	90.56	215.2940
		$\lambda \approx 49$	81.30	83.90	84.20	85.76	86.40	88.30	87.62	89.96	86.00	90.85	212.4170

1 - Os tempos marcados estão apresentados em horas;

2 - As simulações foram realizadas utilizando o código em C++ apresentado no Apêndice B;

3 - Foram utilizadas 10000 réplicas de Monte Carlo, $J = 1000$ e $K = 500$;

4 - As simulações foram realizadas no cluster SIG Altix (cluster Gauss do CESUP).

Tabela 29 – Percentuais de cobertura das estimativas intervalares bootstrap- t simples e bootstrap- t duplo com erros Weibull - nível nominal de 95%.

n	Desenho	λ	HC0		HC2		HC3		HC4		HC5		Tempo
			Simple	Duplo	Simple	Duplo	Simple	Duplo	Simple	Duplo	Simple	Duplo	
20	Balanceado	$\lambda = 1$	83.46	88.72	84.52	89.78	87.58	92.74	93.58	94.78	94.50	94.52	7.3920
		$\lambda \approx 9$	84.96	87.04	87.92	92.04	89.90	95.08	92.92	95.00	93.90	95.00	7.5388
		$\lambda \approx 49$	83.78	85.38	84.82	88.36	91.80	93.32	93.74	95.14	94.43	95.36	7.4891
	Não Balanceado	$\lambda = 1$	84.18	83.42	83.16	83.50	83.08	84.62	84.98	86.72	83.04	84.88	7.6432
		$\lambda \approx 9$	93.82	81.96	82.79	84.38	90.57	91.66	90.20	94.88	90.93	95.42	7.4351
		$\lambda \approx 49$	81.94	85.61	83.28	86.12	86.22	92.22	92.46	95.06	93.78	94.72	7.6669
60	Balanceado	$\lambda = 1$	85.30	86.34	85.28	88.36	90.22	93.38	92.28	95.13	93.28	95.43	55.8420
		$\lambda \approx 9$	83.30	85.28	85.30	86.14	86.30	92.22	92.30	95.11	92.30	95.19	55.1949
		$\lambda \approx 49$	83.31	85.18	84.06	95.16	89.15	92.14	92.06	94.94	93.06	95.14	55.4217
	Não Balanceado	$\lambda = 1$	85.03	86.44	85.84	89.14	90.18	93.56	92.17	94.82	93.08	94.62	55.7006
		$\lambda \approx 9$	85.24	87.86	84.08	85.90	86.06	91.44	93.08	94.82	93.04	94.74	55.1443
		$\lambda \approx 49$	83.70	85.62	83.78	85.47	90.15	93.16	93.64	95.54	90.12	93.63	55.4890
100	Balanceado	$\lambda = 1$	88.88	90.66	89.88	92.66	93.88	94.61	93.84	94.64	93.88	94.89	211.7410
		$\lambda \approx 9$	84.86	87.86	85.86	88.88	90.86	92.65	92.27	95.00	93.86	94.87	211.4520
		$\lambda \approx 49$	83.37	86.48	85.57	90.43	90.36	92.90	94.62	95.00	93.00	94.66	212.7690
	Não Balanceado	$\lambda = 1$	84.24	87.46	85.67	89.63	90.67	93.94	94.93	95.13	93.64	95.24	212.5570
		$\lambda \approx 9$	84.78	85.30	84.00	85.79	90.69	93.00	93.63	95.40	94.58	95.45	215.2940
		$\lambda \approx 49$	83.45	84.53	86.75	87.18	90.34	93.12	94.12	95.00	94.26	95.00	212.4170

1 - Os tempos marcados estão apresentados em horas;

2 - As simulações foram realizadas utilizando o código em C++ apresentado no Apêndice B;

3 - Foram utilizadas 10000 réplicas de Monte Carlo, $J = 1000$ e $K = 500$;

4 - As simulações foram realizadas no cluster SIG Altix (cluster Gauss do CESUP).

Tabela 30 – Percentuais de cobertura das estimativas intervalares bootstrap- t simples e bootstrap- t duplo com erros Weibull - nível nominal de 99%.

n	Desenho	λ	HC0		HC2		HC3		HC4		HC5		Tempo
			Simple	Duplo	Simple	Duplo	Simple	Duplo	Simple	Duplo	Simple	Duplo	
20	Balanceado	$\lambda = 1$	87.46	91.66	90.44	94.62	92.42	96.68	95.50	98.92	96.44	98.79	7.3920
		$\lambda \approx 9$	85.72	90.98	86.72	93.92	89.76	95.94	94.72	98.97	96.74	98.92	7.5388
		$\lambda \approx 49$	82.74	86.92	89.76	93.96	91.76	95.90	94.78	95.84	95.76	96.96	7.4891
	Não Balanceado	$\lambda = 1$	88.50	91.72	87.62	92.24	95.32	98.38	96.62	99.00	97.54	99.11	7.6432
		$\lambda \approx 9$	85.92	90.25	88.56	92.44	90.20	95.34	95.40	99.10	95.08	97.47	7.4351
		$\lambda \approx 49$	84.86	86.22	86.60	88.20	94.30	98.14	97.94	98.06	97.22	98.71	7.6669
60	Balanceado	$\lambda = 1$	88.88	94.88	91.90	96.86	95.92	98.86	96.94	97.86	96.80	98.86	55.8420
		$\lambda \approx 9$	85.88	88.82	87.37	89.84	93.88	96.84	95.88	98.84	96.88	98.33	55.1949
		$\lambda \approx 49$	83.92	86.94	86.92	87.94	92.92	96.94	96.92	98.99	96.92	98.94	55.4217
	Não Balanceado	$\lambda = 1$	90.78	93.54	91.78	94.48	95.82	97.48	96.76	99.02	95.82	97.34	55.7006
		$\lambda \approx 9$	85.74	88.20	87.64	94.20	95.56	98.12	96.30	98.84	96.44	99.21	55.1443
		$\lambda \approx 49$	82.82	87.70	88.82	94.70	96.80	99.70	97.74	99.07	97.80	99.13	55.4890
100	Balanceado	$\lambda = 1$	85.34	88.85	90.14	93.28	95.76	97.67	96.19	98.98	97.31	98.66	211.7410
		$\lambda \approx 9$	80.88	84.86	86.50	88.17	88.73	91.36	94.88	98.86	96.88	99.16	211.4520
		$\lambda \approx 49$	83.84	86.90	85.82	88.90	94.82	95.90	95.82	98.97	96.82	98.90	212.7690
	Não Balanceado	$\lambda = 1$	88.37	92.44	91.34	93.40	95.36	98.62	95.38	98.78	96.11	98.46	212.5570
		$\lambda \approx 9$	85.16	88.98	87.12	90.96	91.14	93.94	90.10	99.26	99.06	98.96	215.2940
		$\lambda \approx 49$	79.60	84.64	87.58	90.66	91.62	93.74	94.64	98.72	96.62	99.32	212.4170

1 - Os tempos marcados estão apresentados em horas;

2 - As simulações foram realizadas utilizando o código em C++ apresentado no Apêndice B;

3 - Foram utilizadas 10000 réplicas de Monte Carlo, $J = 1000$ e $K = 500$;

4 - As simulações foram realizadas no cluster SIG Altix (cluster Gauss do CESUP).

Dado que as estimativas intervalares ao utilizar bootstrap- t duplo (bootstrap- t duplo HC4 e bootstrap- t duplo HC5) apresentaram boas coberturas, discutiremos um pouco sobre suas amplitudes. Todos os gráficos de amplitude a seguir referem-se ao nível nominal de 95% de confiança. Foi observado empiricamente com o uso de gráficos que, em média, as amplitudes das estimativas intervalares via bootstrap- t duplo mostraram-se razoáveis, i.e, em média as amplitudes dos 10000 intervalos de confiança foram pequenas para todos os tamanhos amostrais considerados nesse estudo, em todos os níveis de heteroscedasticidade ($\lambda \approx 9$, $\lambda \approx 49$) e no caso homoscedástico ($\lambda = 1$) para desenhos balanceados e não balanceados. Contudo, houve amplitudes bastante elevadas para $n = 20$ em várias réplica de Monte Carlo, principalmente para o intervalo bootstrap- t duplo HC4, como pode-se observar nas Figuras 6 a 9 no caso não balanceado e $\lambda \approx 49$. Essas figuras contém diagramas de caixa formados por 10000 amplitudes dos intervalos de confiança obtidos pelas metodologias bootstrap- t duplo consideradas nesse estudo. Tanto em situações balanceadas como não balanceadas, o valor de λ não afetou de forma significativa as amplitudes das estimativas intervalares bootstrap- t duplo. Alterações nas amplitudes foram observadas ao aumentar o tamanho da amostra. Observou-se amplitudes menores com $n = 60$ e $n = 100$, ou seja, quando aumentou-se o tamanho da amostra, houve redução das amplitudes. As Figuras 10 a 13 são diagramas de caixa compostos pelas 10000 amplitudes dos intervalos de confiança obtidos pelos métodos bootstrap- t duplo HC0, HC2, HC3, HC4 e HC5 para $\lambda \approx 49$ e desenho não balanceados com $n = 60$.

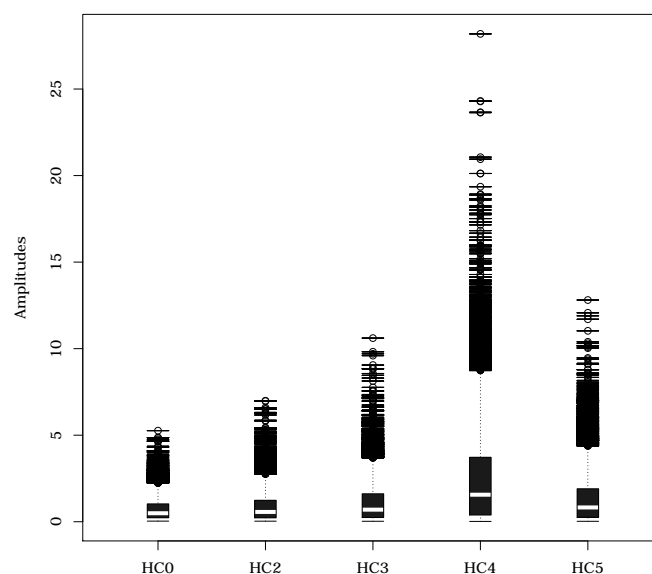


Figura 6 – Amplitudes das estimativas intervalares via bootstrap- t duplo com $n = 20$, erros normais, $\lambda \approx 49$ e desenho não balanceado.

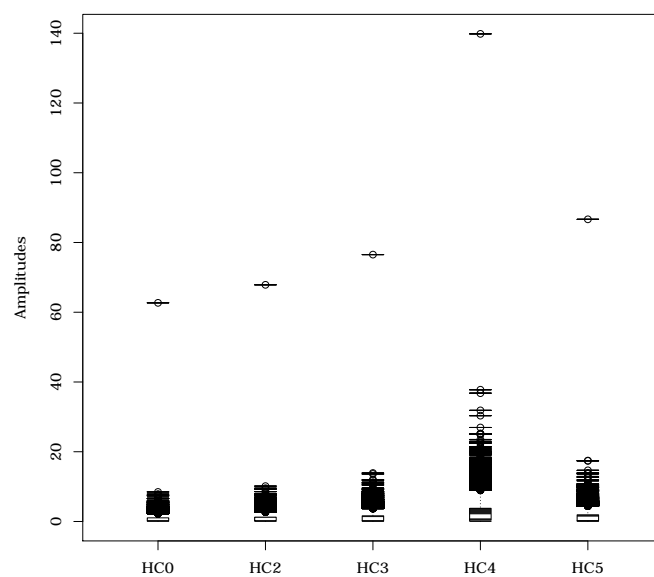


Figura 7 – Amplitudes das estimativas intervalares bootstrap- t duplo com $n = 20$, erros $t_{(3)}$, $\lambda \approx 49$ e desenho não balanceado.

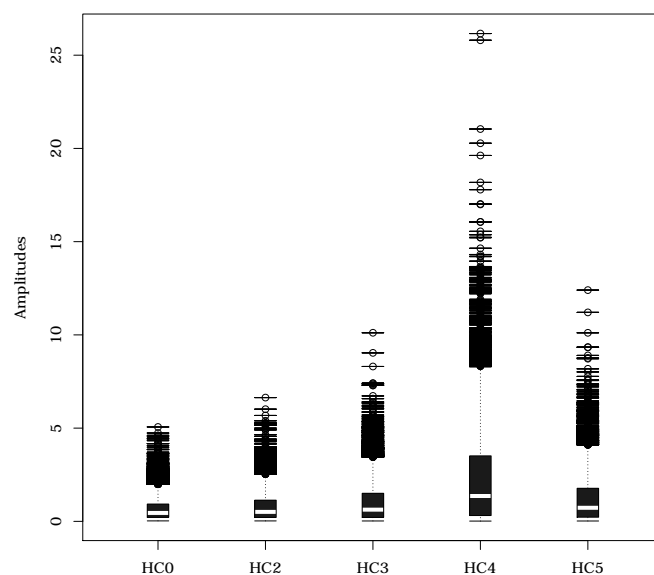


Figura 8 – Amplitudes das estimativas intervalares bootstrap- t duplo com $n = 20$, erros $\chi^2_{(2)}$, $\lambda \approx 49$ e desenho não balanceado.

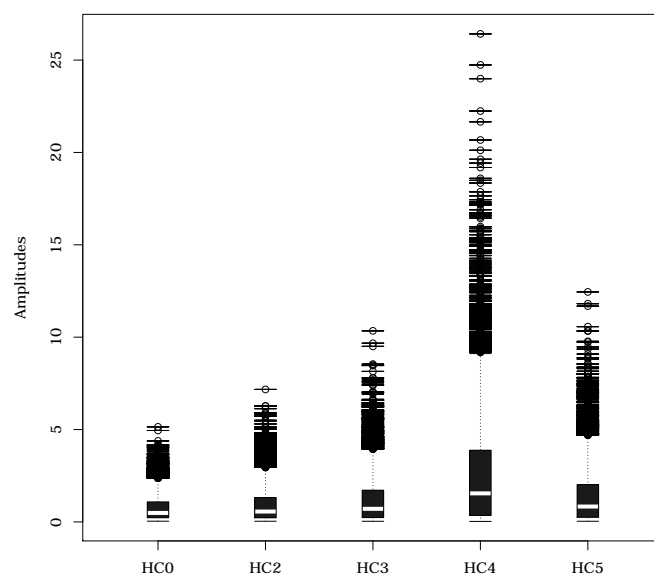


Figura 9 – Amplitudes das estimativas intervalares bootstrap- t duplo com $n = 20$, erros Weibull, $\lambda \approx 49$ e desenho não balanceado.

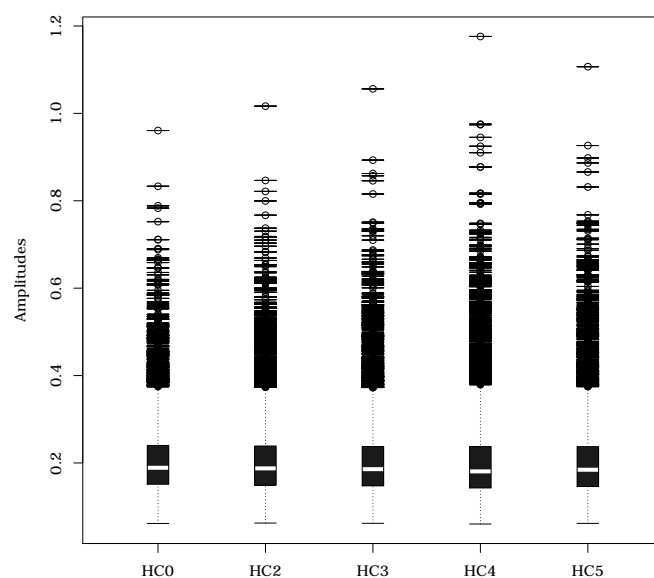


Figura 10 – Amplitudes das estimativas intervalares bootstrap- t duplo com $n = 60$, erros normais, $\lambda \approx 49$ e desenho não balanceado.

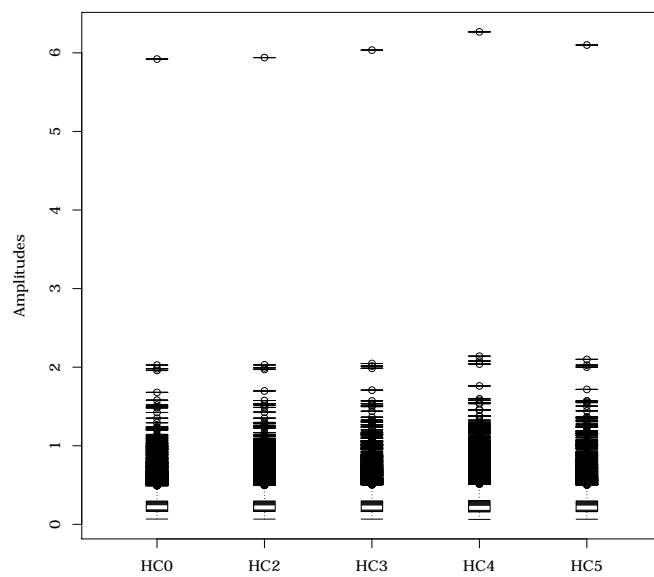


Figura 11 – Amplitudes das estimativas intervalares bootstrap- t duplo com $n = 60$, erros $t_{(3)}$, $\lambda \approx 49$ e desenho não balanceado.

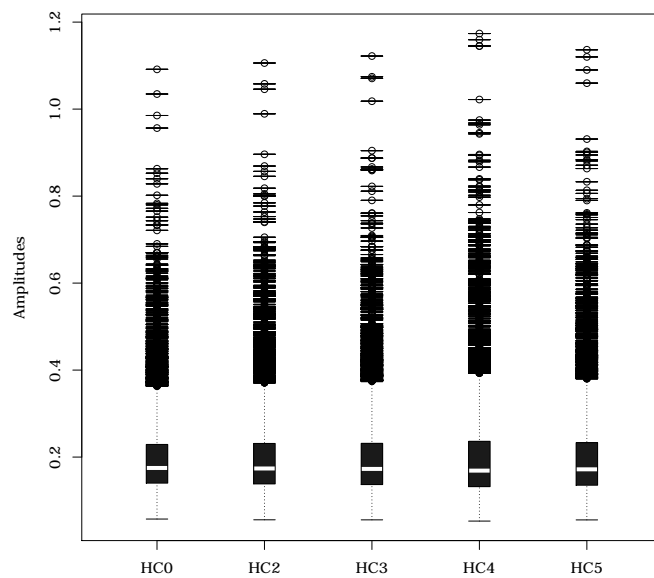


Figura 12 – Amplitudes das estimativas intervalares bootstrap- t duplo com $n = 60$, erros $\chi^2_{(2)}$, $\lambda \approx 49$ e desenho não balanceado.

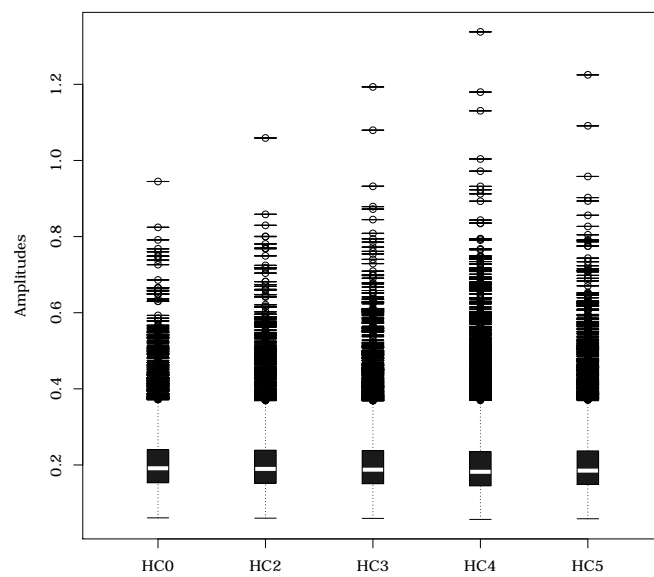


Figura 13 – Amplitudes das estimativas intervalares bootstrap- t duplo com $n = 60$, erros Weibull, $\lambda \approx 49$ e desenho não balanceado.

Aplicação

5.1 Introdução

Esse capítulo contém uma aplicação para um conjunto de dados utilizando as metodologias discutidas nos capítulos anteriores. Serão considerados dados reais e o interesse recairá na construção de intervalos de confiança utilizando as metodologias descritas no Capítulo 3. Para a construção das estimativas intervalares foram implementadas funções escritas utilizando a linguagem R para fácil uso das metodologias. Tais funções realizam estimações intervalares em modelos lineares com heteroscedasticidade de forma desconhecida. Elas foram agrupadas no pacote `hcci` disponível no *Comprehensive R Archive Network* - CRAN em <http://cran.r-project.org/>. O pacote `hcci` foi implementado pelo autor dessa dissertação. Contribuições teóricas necessárias para construção do pacote foram obtidas com o Dr. Francisco Cribari Neto que orientou esse trabalho. Detalhes sobre o funcionamento do pacote também serão apresentados.

5.2 Estatísticas quasi- t e quasi- F

Quando estamos trabalhando com modelos de regressão é comum testarmos hipóteses sobre os parâmetros que os indexam. Comumente, nosso interesse reside em testar se há relação de regressão entre a variável resposta y_i e os regressores $x_{i2}, x_{i3}, \dots, x_{ip}$, $i = 1, 2, \dots, n$, em que p é a quantidade de parâmetros de regressão no modelo. Um outro interesse é testar se determinado regressor contribui ou não para explicar variações em y . Quando a suposição [S3'] (homoscedasticidade) é verdadeira, frequentemente utilizamos a suposição de normalidade para a distribuição dos erros e sob essas suposições são realizados testes de hipóteses e são construídas estimativas intervalares em modelos lineares homoscedásticos. Em se tratando de testes de hipóteses, utilizamos os testes F e t para verificar a adequação global do modelo linear e a importância de um regressor para predi-

ção de y , respectivamente. Com respeito ao teste F (adequação global), quando os erros são homoscedásticos (vale [S3']), queremos testar

$$\mathcal{H}_0 : \beta_2 = \beta_3 = \dots = \beta_p = 0 \text{ contra } \mathcal{H}_1 : \text{Não } \mathcal{H}_0,$$

em que \mathcal{H}_0 é a hipótese nula e \mathcal{H}_1 é a hipótese alternativa. Seja, $\hat{\beta}_s = (\hat{\beta}_2, \dots, \hat{\beta}_p)$ o estimador de mínimos quadrados ordinários de $\beta_s = (\beta_2, \dots, \beta_p)$ e $\text{cov}(\hat{\beta}_s)$ sua estrutura de covariância (bloco $(p-1) \times (p-1)$ inferior) da matriz $\sigma^2(X'X)^{-1}$ com $s = 2, 3, \dots, p$. A estatística de teste utilizada é dada por

$$F = \frac{\hat{\beta}_s' [\widehat{\text{cov}}(\hat{\beta}_s)]^{-1} \hat{\beta}_s}{p-1}, \quad (5.1)$$

em que $\widehat{\text{cov}}(\hat{\beta}_s)$ é obtido substituindo σ^2 por $\hat{\sigma}^2$ em $\text{cov}(\hat{\beta}_s)$.

A distribuição da estatística de teste (5.1) sob \mathcal{H}_0 e sob a suposição de normalidade dos erros é $F(p-1, n-p)$. Rejeitamos a hipótese nula se $F > F(p-1, n-p)$ a um determinado nível de significância. Uma outra hipótese que usualmente estamos interessados em testar é

$$\mathcal{H}_0 : \beta_j = \beta_j^{(0)} \text{ contra } \mathcal{H}_1 : \beta_j \neq \beta_j^{(0)},$$

em que $j = 2, \dots, p$ e $\beta_j^{(0)}$ é uma dada constante, por exemplo, $\beta_j^{(0)} = 0$. A estatística de teste é dada por

$$t = \frac{\hat{\beta}_j - \beta_j^{(0)}}{\sqrt{\hat{\sigma}^2 c_{jj}}}, \quad (5.2)$$

em que c_{jj} é o elemento jj da matriz $(X'X)^{-1}$. A distribuição da estatística (5.2) sob a hipótese nula, normalidade dos erros e homoscedasticidade, é t_{n-p} . Rejeitamos a hipótese nula se $|t| > |t_{n-p}|$ a um determinado nível de significância.

Quando a suposição [S3'] não é verdadeira (não há homoscedasticidade) temos que as estatísticas (5.1) e (5.2) não poderão mais serem utilizadas, pois elas não possuem mais distribuições $F(p-1, n-p)$ e t_{n-p} , respectivamente, quando a hipótese nula é verdadeira. Em se tratando da estatística (5.2), ela não possui distribuição exata t de Student sob a hipótese nula e nem mesmo sua distribuição assintótica sob a hipótese nula é normal padrão. Contudo, podemos definir a estatística de teste (5.2) com base em uma das estimativas consistentes da estrutura de covariância dos estimadores dos parâmetros de indexam um modelo linear de regressão com heteroscedasticidade de forma desconhecida apresentadas no Capítulo 2. Quando definimos a estatística (5.2) considerando em seu denominador uma estimativa consistente do erro-padrão de $\hat{\beta}_j$, obtemos o que é conhecido como estatística *quasi-t*. Sua distribuição nula assintótica é $\mathcal{N}(0, 1)$.

Em relação ao teste de adequação global (teste F), no caso mais geral é de interesse testar mais de uma restrição, i.e., $\mathcal{H}_0 : R\beta - r = 0$ contra $\mathcal{H}_1 : R\beta - r \neq 0$, em que R é uma matriz $k \times p$, em que k é o número de restrições em teste e r é um vetor $k \times 1$. A estatística de teste quando vale [S3] é dada por

$$W = \widehat{m}' R(X'X)^{-1} X' \widehat{\Omega}_l X (X'X)^{-1} R' \widehat{m}, \quad l = 0, 2, 3, 4, 5, \quad (5.3)$$

em que $\widehat{m} = R\widehat{\beta} - r$ e $\widehat{\Omega}_l$ é de tal forma que $X'\widehat{\Omega}_l X$ é uma estimativa consistente para $X'\Omega X$. Sob \mathcal{H}_0 , temos que $W \xrightarrow{d} \chi^2_{(k)}$, em que \xrightarrow{d} denota convergência em distribuição.

5.3 Pacote `hcci`

O pacote `hcci` até o presente momento contém cinco funções. Uma delas faz a estimação da estrutura de covariância dos estimadores dos parâmetros que indexam o modelo linear de regressão, duas funções se destinam à construção de estimativas intervalares usando *bootstrap-t* e *bootstrap percentil* em esquemas simples e duplo e outras duas funções realizam os testes de hipóteses *quasi-F* e *quasi-t*. Os dados utilizados na aplicação desse capítulo, apresentada na Seção (5.4), estão também disponibilizados no pacote. Os dados poderão ser carregados utilizando o comando `data(schools)`. O pacote disponibiliza as funções `HC`, `Tboot`, `Pboot`, `QF` e a função `QT`.

A linguagem R apresenta uma interface com as linguagens de programação de propósito geral C/C++ e FORTRAN. Essa interface fornece uma grande flexibilidade e ajuda a aumentar o desempenho computacional de programas escritos em R. Atualmente o pacote foi escrito utilizando apenas a linguagem R. Futuramente o que se pretende é transcrever as funções de R para C/C++ visando à melhoria na performance computacional para o cálculo das estimativas intervalares. É de interesse também fornecer suporte à GPU (*Graphics Processing Unit*), o que diminuirá bastante o tempo de processamento para usuários que tenham computadores com uma placa de vídeo dedicada com suporte a GPU.

5.3.1 Função `HC`

A função `HC` calcula várias estimativas consistentes da estrutura de covariância dos parâmetros que indexam um modelo linear de regressão. Três argumentos são passados para a função: `model`, `method` e `k`. Como R é uma linguagem orientada a objetos, R suporta classes. O argumento `model` deve ser um objeto pertencente à classe `lm`. No caso da linguagem R, a classe se confunde com a função de mesmo nome chamada `lm`. A função `HC` retornará um erro caso `class(model) == "lm"` retorne `FALSE`. O segundo argumento da função `HC` (argumento `method`) refere-se ao método `HC` que será utilizado

para a construção da matriz de covariâncias dos estimadores dos parâmetros que indexam o modelo linear heteroscedástico passado como argumento para `model`. Os valores possíveis para serem passados para o argumento `method` são 0, 2, 3, 4 e 5, referindo-se aos métodos HC0, HC2, HC3, HC4 e HC5, respectivamente. Caso nenhuma opção seja passada ao argumento `method`, o padrão será utilizar `method=4`, ou seja, será utilizado o estimador HC4. O terceiro e último argumento da função HC (argumento `k`) é a constante k , que por padrão é 0.7. Esse argumento apenas influenciará no caso em que `method=5`. A constante k faz parte do cálculo da matriz diagonal $\hat{\Omega}_5$. A forma geral da função é `HC(model, method, k)`.

5.3.2 Funções QT e QF

As funções QT e QF realizam os testes quasi- t e quasi- F descritos na Seção 5.2. A função QT recebe quatro argumentos: `model`, `significance`, `hc` e `h0`. Assim como na função HC, o argumento `model` da função QT deve pertencer à classe `lm`. Se um objeto de classe diferente for passado como argumento para `model` ocorrerá um erro e uma mensagem de advertência será exibida no *prompt* de comandos do R. O segundo e terceiro argumentos da função QT referem-se ao nível de significância do teste e ao método HC que será utilizado para o cálculo da estatística de teste. Esses argumentos são denotados por `significance` e `hc`, respectivamente, em que por padrão `significance=0.05` e `hc=4`. O último argumento da função QT refere-se ao vetor de restrições para as hipóteses que estão sendo testadas. Por padrão, temos que $\beta_1^{(0)} = 0, \beta_2^{(0)} = 0, \dots, \beta_p^{(0)} = 0$ (`h0=0`). Todos os argumentos da função QF são iguais aos da função QT, exceto o argumento `h0`, que deve ser uma matriz de restrições ($k \times p$), em que k é o número de restrições e p a quantidade de parâmetros no modelo.

5.3.3 Funções Pboot e Tboot

A função `Pboot` calcula as estimativas intervalares para modelos lineares de regressão usando o método bootstrap percentil com inicialização selvagem proposta por Wu (1986). Assim como na função HC, a função `Pboot` requer como opção para o primeiro argumento da função um modelo linear pertencente à classe `lm`, sendo essa opção passada para o argumento `model`. Da mesma forma que nas funções HC, QT e QF, caso um objeto de classe diferente seja passado como argumento uma mensagem de aviso será exibida. O segundo argumento (argumento `significance`) refere-se ao nível de significância adotado para a construção das estimativas intervalares. Caso nenhum nível seja passado o nível de significância adotado será $\alpha = 0.05$. O terceiro argumento (argumento `double`) da função `Pboot` refere-se ao tipo de esquema bootstrap percentil que será realizado, ou seja, se `double=FALSE` (que é o padrão da função), será utilizado o método bootstrap percentil,

caso contrário (`double=TRUE`), será utilizado o esquema bootstrap duplo percentil para o cálculo das estimativas intervalares. Essas estimativas são feitas para todos os parâmetros do modelo.

O quarto e quinto argumentos (`J` e `K`) da função `Pboot` referem-se ao número de amostras do bootstrap exterior e interior, respectivamente. Em casos que `double=FALSE` (padrão da função) se for passado algum valor para o argumento `K`, este será desconsiderado. Sendo assim, apenas as estimativas do bootstrap percentil serão calculadas. Se ao menos um valor não for passado para os argumentos `J` e `K`, ou seja, `J=NULL` ou `K=NULL`, a função calculará automaticamente a quantidade de réplicas do bootstrap exterior e interior, respectivamente. Para esse cálculo será considerada a sugestão apresentada por Booth e Hall (1994). Essa sugestão foi comentada na Seção 3.5 do Capítulo 3. O sexto e último argumento da função `Pboot` (argumento `distribution`) refere-se à distribuição das variáveis aleatórias t_i^* e t_i^{**} , $i = 1, \dots, n$, que multiplicam as quantidades $\hat{\varepsilon}_i/\sqrt{1-h_i}$ e $\hat{\varepsilon}_i^*/\sqrt{1-h_i}$, $i = 1, \dots, n$, para construção das amostras (y^*, X) e (y^{**}, X) utilizadas no bootstrap percentil e bootstrap percentil duplo, respectivamente.

A função `Tboot` calcula estimativas intervalares para modelos lineares de regressão utilizando o método bootstrap- t ou bootstrap- t duplo. Todos os argumentos para a função `Pboot` e seus detalhes são válidos para a função `Tboot`, entretanto há um argumento a mais na função `Tboot` que se refere ao método HC utilizado para o cálculo dos erros-padrão utilizados pelos métodos bootstrap- t e bootstrap- t duplo. Esse argumento é denominado `hc` e pode assumir as opções 0, 2, 3, 4 e 5 referentes aos métodos HC0, HC2, HC3, HC4 e HC5, respectivamente.

5.4 Aplicações empíricas

A variável de interesse (y) refere-se às despesas per capita em escolas públicas e a variável independente (x) são as rendas per capita dos estados norte americanos em 1979. Os dados estão apresentados na Tabela 31. Os valores da variável renda serão reescalados por 10^{-4} , i.e., cada valor de x será dividido por 10^4 antes de se realizar a análise. O estado de Wisconsin será retirado da amostra por apresentar informação incompleta. A fonte original dos dados é o Departamento de Comércio dos Estados Unidos da América. Esses dados também estão disponíveis em Greene (1997, Tabela 12.1, p. 541). Consideremos o modelo de regressão linear dado por

$$y_i = \beta_1 + \beta_2 x_i + \varepsilon_i, \quad i = 1, 2, \dots, 50. \quad (5.4)$$

Com base no teste de Breusch-Pagan a hipótese nula de homoscedasticidade foi rejeitada ao nível de significância de 5% (p -valor ≈ 0.0006), o que indica a presença de heteroscedasticidade nos dados. Assim, não temos $\text{cov}(\hat{\beta}) = \sigma^2(X'X)^{-1}$.

Tabela 31 – Dados de gastos per capita em escolas públicas e renda per capita por estado em 1979 nos Estados Unidos.

Estado	Gasto	Renda	Estado	Gasto	Renda	Estado	Gasto	Renda
Alab.	275	6247	La.	316	6640	Ohio	322	7812
Alas.	821	10851	Maine	327	6333	Okla.	320	6951
Ariz.	339	7374	Md.	427	8306	Oreg.	397	7839
Arka.	275	6183	Mass.	427	8063	Pa.	412	7733
Cal.	387	8850	Mich.	466	8442	R.I.	342	7526
Colo.	452	8001	Minn.	477	7847	S.C.	315	6242
Ct.	531	8914	Miss.	259	5736	S.Dak.	321	6841
Del.	424	8604	Mo.	274	7342	Tenn.	268	6489
Flor.	316	7505	Mont.	433	7051	Texas	315	7697
Ga.	265	6700	Nebr.	294	7391	Utah	417	6622
Hawaii	403	8380	Nev.	359	9032	Vt.	353	6541
Idaho	304	6813	N.H.	279	7277	Va.	356	7624
Ill.	437	8745	N.J.	423	8818	Wash.	415	8450
Ind.	345	7696	N.Mex	388	6505	D.C	428	10022
Iowa	431	7873	N.Y.	447	8267	W.Va.	320	6456
Kans.	355	8001	N.C.	335	6607	Wyo.	500	9096
Ky.	260	6615	N.Dak.	311	7478	Wisc.	*	7597

O símbolo * representa dados faltantes.

Utilizando o método de mínimos quadrados ordinários, obteve-se $\hat{\beta}_1 = -151.2651$ e $\hat{\beta}_2 = 689.3881$. O modelo estimado é dado por

$$\hat{y} = -151.2651 + 689.3881x, \quad (5.5)$$

em que \hat{y} é o valor predito da variável gasto per capita em escolas públicas e x é a renda per capita dos estados em um corte transversal no ano de 1979 nos Estados Unidos. Uma relação de regressão linear entre as variáveis y e x é clara, como pode ser observado na Figura 14. Com base no teste quasi- t concluiu-se que o parâmetro β_1 não é importante para o modelo, sendo o p -valor da estatística de teste igual a 0.1874. Já com um p -valor (≈ 0.0016) foi observado que a variável renda per capita por estado é importante para prever a variável despesas per capita em escolas públicas. É provável que esse fato seja refletido nos intervalos de confiança para os parâmetros que indexam o modelo (5.4) que serão estimados mais a frente. Ou seja, as estimativas intervalares para o parâmetro β_1 deverão conter o valor zero enquanto as estimativas dos intervalos para o parâmetro β_2 não deverão conter o valor zero.

O coeficiente de determinação do modelo (5.5) é $R^2 = 0.5868$, ou seja, ele explica aproximadamente 59% da variabilidade total de y . O coeficiente de determinação ajustado do modelo (5.5) é 0.5782.

Com base na matriz $H = X(X'X)^{-1}X'$ é possível calcular o grau de alavancagem de cada observação. Essas medidas são dadas pelos elementos diagonais da matriz H (h_i , $i = 1, 2, \dots, 50$). Foi verificado que as observações referentes aos estados do Alasca e Washington D.C. são pontos de alavanca, com h_2 (≈ 0.2144) e h_{48} (≈ 0.1277) $> 3p/n$ ($= 0.1200$), em que $3p/n = 3\bar{h}$, sendo \bar{h} a alavancagem média. Observou-se também que a segunda observação, que se refere ao estado do Alasca, é um ponto bastante influente, com medida de alavancagem ultrapassando $4p/n$ ($= 0.1600$).

Dessa forma, estamos sob um esquema não balanceado em que possivelmente os métodos bootstrap- t ou bootstrap- t duplo utilizando o erro-padrão obtido pelo método HC4 será uma boa opção para a construção das estimativas intervalares para os parâmetros β_1 , β_2 do modelo (5.4). A Figura 14 apresenta os dados considerados nessa aplicação juntamente com a reta estimada pelo modelo (5.5).

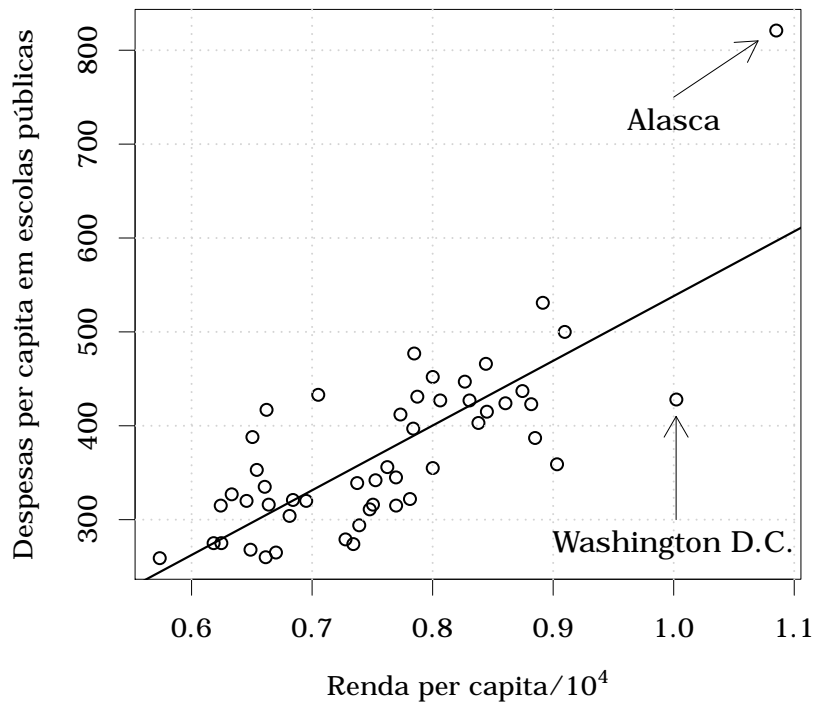


Figura 14 – Renda per capita e despesas per capita em escolas públicas.

Utilizando o pacote `hcci` foi possível calcular as estimativas intervalares obtidas pelos métodos bootstrap percentil e bootstrap- t em esquemas bootstrap simples e duplo. As estimativas referentes ao bootstrap- t (esquema simples e duplo) consideraram os estimadores HC0, HC2, HC3, HC4 e HC5 da matriz de covariâncias dos estimadores

dos parâmetros que indexam o modelo (5.4). Essas estimativas encontram-se na Tabela 32. O cálculo levou em consideração o nível de significância de 5% e foram utilizadas 1000 réplicas para o bootstrap exterior e 500 réplicas para o bootstrap interior, ou seja, $J = 1000$ e $K = 500$. O custo computacional para o bootstrap- t e bootstrap percentil em R, utilizando a função `Tboot` foi muito reduzido, sendo o maior tempo observado de 3.5400 segundos para esquemas que utilizaram apenas um nível de bootstrap. Contudo, o cálculo das estimativas intervalares por bootstrap- t utilizando um esquema duplo apresentou custo computacional considerável, chegando a ultrapassar os 27 minutos para essa aplicação utilizando o cluster SGI Altix - Gauss. Um custo computacional elevado também foi observado quando se utilizou a função `Pboot` quando consideramos um segundo nível bootstrap; nesse exemplo, o tempo de execução ficou próximo aos 21 minutos. Em um computador com hardware mais modesto, que utiliza processador Intel(R) Core(TM) i3 CPU M 330 2.13GHz e 3Gb de memória RAM, o maior tempo de execução do método bootstrap- t (apenas um nível bootstrap) foi de 6.8000 segundos. Já o maior tempo de execução observado das funções (`Pboot` e `Tboot`) que realizam estimativas intervalares e que fizeram uso de um segundo nível de bootstrap (`double=TRUE`) foi de 1.6652 horas nesse mesmo hardware. Os tempos de execução das funções `Pboot` e `Tboot` quando fazemos `double=TRUE`, ou seja, quando utilizamos o segundo nível de bootstrap, poderão ser bastante inconvenientes caso essas funções precisem ser chamadas repetidas vezes, como no caso de uma simulação de Monte Carlo. Para esses casos, o uso de linguagens compiladas, como C/C++, poderá ser a estratégia mais conveniente. Mesmo fazendo uso de uma máquina virtual para compilação das funções `Pboot` e `Tboot` usando compilação *Just-in-Time* (JIT), em R, observou-se que não há redução significativa nos tempos de execução dessas funções no cálculo das estimativas intervalares por bootstrap duplo. Em alguns casos é possível reduzir o custo computacional utilizando a sugestão apresentada por Booth e Hall (1994), que fornece escolhas adequadas e às vezes menos custosas de J e K . Essa sugestão foi apresentada na Seção 3.5 do Capítulo 3. Para essa aplicação e continuando a considerar o nível de significância de $\alpha = 0.05$, os valores de J e K sugeridos foram 2199 e 88, respectivamente. As estimativas com os tempos de execuções das funções usando $J = 2199$ e $K = 88$ e fazendo uso do cluster SIG Altix - Gauss estão apresentadas na Tabela 33. Escolhendo J e K pela proposta de Booth e Hall (1994), ou seja, utilizando o pacote `hcci` passando `NULL` como argumentos de J e K (`J=NULL` e `K=NULL`) nas funções `Pboot` ou `Tboot`, observou-se uma redução de 1.6652 horas para 0.5308 horas no maior tempo de execução nas estimativas realizadas em um computador com processador Intel(R) Core(TM) i3 CPU M 330 2.13GHz e 3Gb de memória RAM que levaram em consideração um segundo nível de bootstrap (`double=TRUE`).

Foi observado nessa aplicação que entre as estimativas intervalares que levaram em consideração esquemas bootstrap- t e bootstrap- t duplo os intervalos foram mais estreitos

considerando o estimador HC4. Intervalos mais estreitos foram também observados nas estimativas intervalares calculadas pelo bootstrap percentil. Observou-se também que todas as estimativas intervalares para o parâmetro β_2 apresentadas na Tabela 32 e Tabela 33 não contiveram o valor zero, em especial o intervalo obtido pelo método bootstrap- t HC4 que se mostrou nas simulações anteriores um dos mais adequados entre os estimadores considerados nesse trabalho. Pela relação estreita de intervalos de confiança e testes de hipóteses, esse fato mostra a importância da variável renda per capita por estado (regressor) para prever a variável despesas per capita em escolas públicas (regressando). Já em se tratando do parâmetro β_1 foi observado que o valor zero pertence a todos os intervalos, como pode ser observado nas Tabelas 32 e 33. Contudo, na prática quase sempre optamos em deixar o intercepto no modelo para podermos fazer uso de estatísticas que necessitam do uso do intercepto. Essa inclusão e não inclusão do valor zero nos intervalos para os parâmetros β_1 e β_2 , respectivamente, condizem com a conclusão obtida pelo teste quasi- t . Consideremos agora o modelo abaixo,

$$y_i = \beta_1 + \beta_2 x_i + \beta_3 x_i^2 + \varepsilon_i, \quad i = 1, 2, \dots, 50. \quad (5.6)$$

O modelo linear (5.6) acrescenta o efeito quadrático (x_i^2) ao modelo inicial apresentado em (5.4), em que, x_i^2 refere-se a variável renda per capita ao quadrado. Nesse modelo os valores da variável x_i também foram reescalados por 10^4 , ou seja, x_i , $i = 1, \dots, 50$ é dividido por 10^4 . O modelo estimado é dado por

$$\hat{y} = 832.9144 - 1834.2029x + 1587.0423x^2 \quad (5.7)$$

Assim como no modelo (5.4), pelo teste de Breusch-Pagan rejeitamos a hipótese nula de homoscedasticidade dos erros do modelo (5.6) a um nível de significância de 5%; p -valor ≈ 0.0004 . O modelo estimado descrito em (5.7) apresentou o coeficiente de determinação $R^2 = 0.6553$, ou seja, ele explica aproximadamente 66% da variabilidade total de y . O coeficiente de determinação ajustado do modelo (5.7) foi igual a 0.6407. A Figura 15 apresenta o modelo ajustado definido em (5.7).

As observações 2, 24 e 48 referentes aos estados do Alasca, Mississippi e Washington D.C. são pontos de alavancas, em que $h_2 (\approx 0.6508)$, $h_{24} (\approx 0.2000)$ e $h_{48} (\approx 0.2079) > 3p/n = 0.1800$. A observação referente ao estado do Alasca é um ponto de alta alavancagem, em que $h_2 > 4p/n = 0.2400$. Dessa forma, estamos utilizando um conjunto de dados não balanceados, ou seja, há a presença de ponto de alta alavancagem. Com base nas simulações apresentadas no Capítulo 3, utilizar os métodos bootstrap- t ou bootstrap- t duplo utilizando o estimador HC4 será uma boa estratégia para o cálculo dos intervalos de confiança para os parâmetros que indexam o modelo (5.6). Com base no teste quasi- t utilizando o estimador HC4 de $\hat{\beta}_j$, com $j = 1, 2, 3$, observou-se que o intercepto bem como

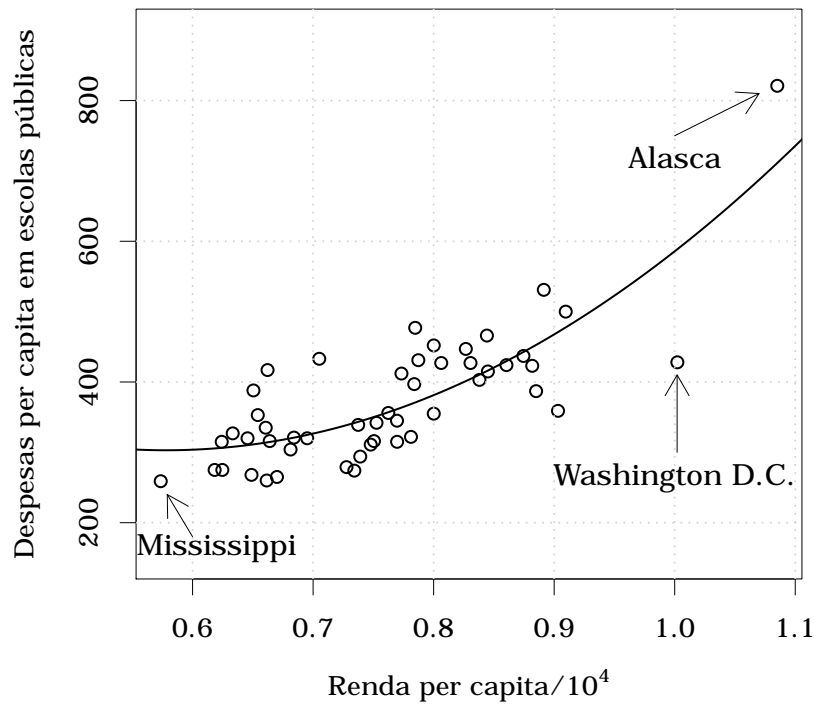


Figura 15 – Renda per capita e despesas per capita em escolas públicas.

as variáveis x e x^2 não são significativas para prever o comportamento da variável resposta a um nível de significância de 5% com p -valores aproximadamente iguais a 0.3909, 0.4113 e 0.3862, respectivamente.

As Tabelas 34 e 35 apresentam as estimativas intervalares utilizando as metodologias bootstraps apresentadas no Capítulo 3 para o modelo (5.6). É importante perceber que conclusões melhores sobre a importância das variáveis explicativas para predição da variável resposta (y) podem ser tomadas utilizando as estimativas intervalares via bootstrap- t simples ou bootstrap- t duplo utilizando o estimador HC4 que fornecem, em geral, regiões de aceitação mais precisas do que as obtidas pelo teste quasi- t .

As estatísticas do teste quasi- t bem como os p -valores calculados a partir dessas estatísticas para todos os parâmetros que indexaram os modelos (5.4) e (5.6), discutidas anteriormente, foram calculadas pela função QT do pacote `hcci`. O exemplo a seguir exemplifica o uso da função QT para os dois modelos considerados nessa aplicação.

Exemplo 1:

```

> library(hcci)
> data(schools)
> as.data.frame(schools[-50,],ncol=3)
> y = schools$Expenditure # Variável gastos em escolas públicas
> x = schools$Income/10000 # Variável renda/10^4
> modelo_1 = lm(y ~ x) # Modelo (5.5)
> modelo_2 = lm(y ~ x+I(x^2)) # Modelo (5.7)

> QT(modelo_1, significance = 0.05, hc=4, h0=0)$p_value

Intercept          x
0.18738684 0.00158113

> QT(modelo_2, significance = 0.05, hc=4, h0=0)$p_value

Intercept          x      I(x^2)
0.3909289 0.4113231 0.3862390

```

As hipóteses nulas para os dois testes realizados pelo código acima referem-se a $\mathcal{H}_0 : \beta_1 = \beta_2 = 0$ para o modelo (5.4) e $\mathcal{H}_0 : \beta_1 = \beta_2 = \beta_3 = 0$ para o modelo (5.6). O uso das funções `Pboot` e `Tboot` também é bastante simplificado. O trecho de código abaixo exemplifica o uso dessas funções para o cálculo de estimativas intervalares via bootstrap- t duplo HC4 e bootstrap percentil duplo para o modelo (5.6).

Exemplo 2:

```

# IC bootstrap-t duplo HC4 para o modelo (4.6)
> Tboot(model=modelo_2, significance = 0.05, hc = 4, double = TRUE,
        J=NULL, K=NULL, distribution = "rademacher")

# IC bootstrap percentil duplo para o modelo (4.6)
> Pboot(model=modelo_2, significance = 0.05, double = FALSE,
        J=NULL, K=NULL, distribution = "rademacher")

```

Tabela 32 – Estimativas intervalares obtidas por bootstrap percentil e bootstrap- t em esquemas bootstrap simples e duplo para $J = 1000$ e $K = 500$ fixados.

Método	Parâmetros				Tempo (minutos)	
	β_1		β_2		Simples	Duplo
	Simples	Duplo	Simples	Duplo		
Bootstrap-t HC0	[-541.415; 241.638]	[-533.257; 230.440]	[156.837; 1226.191]	[178.481; 1217.290]	0.047	26.465
Amplitude	(783.053)	(763.697)	(1069.354)	(1038.809)		
Bootstrap-t HC2	[-507.018; 206.695]	[-489.511; 193.494]	[204.311; 1178.271]	[225.690; 1164.706]	0.058	27.400
Amplitude	(713.713)	(683.005)	(973.960)	(939.016)		
Bootstrap-t HC3	[-474.470; 173.718]	[-456.454; 159.167]	[249.054; 1133.122]	[270.295; 1116.879]	0.051	25.042
Amplitude	(648.188)	(615.621)	(884.068)	(846.584)		
Bootstrap-t HC4	[-418.428; 118.943]	[-396.325; 100.409]	[325.462; 1056.003]	[359.113; 1032.313]	0.059	26.926
Amplitude	(537.371)	(496.734)	(730.541)	(673.200)		
Bootstrap-t HC5	[-476.962; 176.565]	[-458.802; 158.728]	[245.469; 1136.710]	[274.415; 1115.396]	0.053	26.944
Amplitude	(653.527)	(617.530)	(891.241)	(840.981)		
Bootstrap Percentil	[-355.154; 53.872]	[-397.607; 100.252]	[409.503; 968.659]	[342.100; 1057.420]	0.038	20.577
Amplitude	(409.026)	(497.859)	(559.156)	(715.320)		

1 - Estimativas intervalares para o modelo $y_i = \beta_1 + \beta_2 x_i + \varepsilon_i$, $i = 1, 2, \dots, 50$;

2 - Modelo estimado: $y = -151.2651 + 689.3881x$;

3 - Estimativas calculadas pelo pacote `hcci` utilizando os hardwares do cluster SIG Altix - Gauss.

Tabela 33 – Estimativas intervalares obtidas por bootstrap percentil e bootstrap- t em esquemas bootstrap simples e duplo com $J = 2199$ e $K = 88$ obtidos por minimização da função M_2 proposta por Booth e Hall (1994).

Método	Parâmetros				Tempo (minutos)	
	β_1		β_2		Simples	Duplo
	Simples	Duplo	Simples	Duplo		
Bootstrap-t HC0	[-554.938; 281.964]	[-563.067; 322.434]	[85.518; 1262.513]	[9.779; 1315.641]	0.071	8.723
Amplitude	(836.902)	(885.501)	(1176.995)	(1305.862)		
Bootstrap-t HC2	[-519.405; 244.485]	[-527.091; 280.726]	[138.176; 1210.663]	[71.0155; 1214.106]	0.115	9.828
Amplitude	(763.89)	(807.817)	(1072.487)	(1143.091)		
Bootstrap-t HC3	[-485.792; 208.838]	[-492.773; 212.734]	[189.620; 1161.584]	[128.750; 1164.710]	0.106	9.265
Amplitude	(694.630)	(705.507)	(971.964)	(1035.960)		
Bootstrap-t HC4	[-428.102; 147.164]	[-414.504; 150.398]	[277.469; 1078.234]	[271.239; 1080.208]	0.111	10.285
Amplitude	(575.266)	(564.902)	(800.765)	(808.969)		
Bootstrap-t HC5	[-488.472; 211.720]	[-495.486; 215.910]	[185.748; 1165.054]	[124.812; 1168.130]	0.104	10.776
Amplitude	(700.192)	(711.396)	(979.306)	(1043.318)		
Bootstrap Percentil	[-357.267; 58.536]	[-421.613; 115.903]	[406.565; 967.193]	[330.165; 1056.420]	0.081	7.341
Amplitude	(415.803)	(537.516)	(560.628)	(726.255)		

1 - Estimativas intervalares para o modelo $y_i = \beta_1 + \beta_2 x_i + \varepsilon_i$, $i = 1, 2, \dots, 50$;

2 - Modelo estimado: $y = -151.2651 + 689.3881x$;

3 - Estimativas calculadas pelo pacote `hcci` utilizando os hardwares do cluster SIG Altix - Gauss.

Tabela 34 – Estimativas intervalares obtidas por bootstrap percentil e bootstrap- t em esquemas bootstrap simples e duplo para $J = 1000$ e $K = 500$ fixados.

Método	Parâmetros				Tempo (minutos)	
	Simples	β_2 Duplo	Simples	β_3 Duplo	Simples	Duplo
Bootstrap-t HC0 Amplitude	[-47188.180; 44049.990] (91238.170)	[-60295.060; 52675.190] (112970.200)	[-30700.000; 33536.840] (64236.840)	[-35961.030; 41431.280] (77392.310)	0.066	26.369
Bootstrap-t HC2 Amplitude	[-43425.580; 40485.460] (83911.040)	[-54419.960; 49072.060] (103492.000)	[-27569.160; 30269.700] (57838.860)	[-32835.040; 37002.890] (69837.930)	0.069	34.139
Bootstrap-t HC3 Amplitude	[-38986.690; 36224.360] (75211.050)	[-46726.780; 45264.990] (91991.770)	[-24790.580; 26765.81] (51556.390)	[-29971.940; 31427.130] (61399.070)	0.052	26.717
Bootstrap-t HC4 Amplitude	[-29830.480; 30428.660] (60259.140)	[-39807.340; 38266.570] (78073.910)	[-20476.000; 21023.560] (41499.560)	[-24780.590; 26548.230] (51328.820)	0.057	28.023
Bootstrap-t HC5 Amplitude	[-32762.420; 34258.010] (67020.430)	[-44508.680; 43346.060] (87854.740)	[-23327.580; 23245.990] (46573.570)	[-28543.410; 30056.620] (58600.030)	0.053	32.130
Bootstrap Percentil Amplitude	[-4849.221; 1207.234] (6056.455)	[-5522.004; 1917.256] (7439.260)	[-416.055; 3579.675] (3995.730)	[-791.238; 3957.102] (4748.340)	0.031	18.505

1 - Estimativas intervalares para o modelo $y_i = \beta_1 + \beta_2 x_i + \beta_3 x_i^2 + \varepsilon_i$, $i = 1, 2, \dots, 50$;

2 - Modelo estimado: $\hat{y} = 832.9144 - 1834.2029x + 1587.0423x^2$;

3 - Estimativas calculadas pelo pacote `hcci` utilizando os hardwares do cluster SIG Altix - Gauss.

Tabela 35 – Estimativas intervalares obtidas por bootstrap percentil e bootstrap- t em esquemas bootstrap simples e duplo com $J = 2199$ e $K = 88$ obtidos por minimização da função M_2 proposta por Booth e Hall (1994).

Método	Parâmetros				Tempo (minutos)	
	β_2		β_3		Simple	Duplo
	Simple	Duplo	Simple	Duplo		
Bootstrap-t HC0	[-47387.160; 43678.740]	[-58203.060; 57540.360]	[-30336.970; 33605.750]	[-37894.350; 39774.510]	0.165	12.552
Amplitude	(91065.900)	(115743.400)	(63942.720)	(77668.860)		
Bootstrap-t HC2	[-42561.250; 39178.910]	[-53713.830; 53035.730]	[-27060.560; 29989.030]	[-34645.070; 36735.350]	0.140	13.115
Amplitude	(81740.160)	(106749.600)	(57049.590)	(71380.420)		
Bootstrap-t HC3	[-38176.450; 34107.650]	[-49450.120; 48318.350]	[-23498.620; 26618.430]	[-31298.520; 33523.170]	0.178	14.214
Amplitude	(72284.100)	(97768.470)	(50117.050)	(64821.690)		
Bootstrap-t HC4	[-29617.673; 27032.870]	[-42265.930; 39180.800]	[-18084.187; 20700.690]	[-25212.010; 28499.750]	0.164	14.116
Amplitude	(56650.540)	(81446.730)	(38784.880)	(53711.760)		
Bootstrap-t HC5	[-32586.290; 30155.640]	[-47080.860; 44925.420]	[-20607.500; 22985.360]	[-29190.310; 32235.960]	0.136	11.212
Amplitude	(62741.930)	(92006.280)	(43592.860)	(61426.270)		
Bootstrap Percentil	[-4848.882; 1121.528]	[-5738.786; 1901.945]	[-374.841; 3582.547]	[-809.436; 4080.056]	0.100	6.807
Amplitude	(5970.410)	(7640.731)	(3957.387)	(4889.492)		

1 - Estimativas intervalares para o modelo $y_i = \beta_1 + \beta_2 x_i + \beta_3 x_i^2 + \varepsilon_i$, $i = 1, 2, \dots, 50$;

2 - Modelo estimado: $\hat{y} = 832.9144 - 1834.2029x + 1587.0423x^2$;

3 - Estimativas calculadas pelo pacote `hcci` utilizando os hardwares do cluster SIG Altix - Gauss.

Considerações Finais

O presente estudo teve como objetivo avaliar, em pequenas amostras (20, 60, 100), algumas metodologias utilizadas para o cálculo das estimativas intervalares em modelos lineares de regressão com heteroscedasticidade de forma desconhecida. Nas avaliações numéricas foi considerado um modelo de regressão linear com dois parâmetros, sendo $\beta_1 = 1$, $\beta_2 = 1$ e o regressor obtido da distribuição $t_{(3)}$. As simulações levaram em considerações erros obtidos de diferentes distribuições de probabilidade sob diferentes níveis de heteroscedasticidade ($\lambda \approx 9$ e $\lambda \approx 49$) e o caso homoscedástico ($\lambda = 1$). Também foram avaliados os desempenhos dos estimadores intervalares em situações em que os dados não apresentavam pontos de alavanca (desenho balanceado) e em situações em que os dados apresentavam pontos de alta alavancagem (desenho não balanceado).

Inicialmente foram avaliados via simulações de Monte Carlo, levando em consideração os percentuais de cobertura, os intervalos de confiança OLS, HC0, HC2, HC3, HC4 e HC5. Observou-se que as estimativas intervalares HC0, HC2, HC3, HC4 e HC5, não são precisas em pequenas amostras. Foi observado que em casos de dados com presença de pontos de alta alavancagem, as coberturas dos intervalos de confiança HC0, HC2, HC3, HC4 e HC5, em geral, foram melhores quando utilizados quantis $t_{(n-2)}$. Contudo, essas melhorias não são suficientes para garantir ótimos percentuais de cobertura em todas as situações analisadas.

Estimadores intervalares bootstrap também foram considerados. Foram avaliados os métodos bootstrap percentil e bootstrap- t utilizando os estimadores pontuais HC0, HC2, HC3, HC4 e HC5. Ambas as metodologias foram avaliadas em esquemas simples e duplo. O método bootstrap percentil não apresentou boas estimativas em todos os cenários considerados nessa dissertação. Em contrapartida, observou-se bons percentuais de cobertura do estimador bootstrap percentil duplo em situações em que os dados não continham pontos de alta alavancagem (desenho balanceado). Coberturas razoáveis foram obtidas utilizando os métodos bootstrap- t HC4 e bootstrap- t HC5. Contudo, coberturas

melhores foram obtidas utilizando os métodos bootstrap- t duplo HC4 e bootstrap- t duplo HC5 nos diversos cenários avaliados nesse trabalho. Para facilitar o uso dos métodos bootstrap- t e bootstrap percentil em esquemas simples e duplo foi criado o pacote na `hcci` versão 1.0.0 para a linguagem R.

Referências

- AMDAHL, G. M. Validity of the single processor approach to achieving large scale computing capabilities. In: ACM. *Proceedings of the April 18-20, 1967, spring joint computer conference*. California, 1967. p. 483–485. Citado na página [26](#).
- ANDERSON, T. W.; DARLING, D. A. Asymptotic theory of certain "goodness of fit" criteria based on stochastic processes. *The annals of mathematical statistics*, JSTOR, p. 193–212, 1952. Citado na página [71](#).
- BEEBE, N. H. Comments on the future of TeX and METAFONT. *TUGboat*, v. 11, n. 4, p. 490–494, 1990. Citado na página [29](#).
- BERAN, R. Prepivoting to reduce level error of confidence sets. *Biometrika*, Biometrika Trust, v. 74, n. 3, p. 457–468, 1987. Citado 4 vezes nas páginas [46](#), [47](#), [48](#) e [49](#).
- BOOTH, J. G.; HALL, P. Monte Carlo approximation and the iterated bootstrap. *Biometrika*, v. 81, p. 331–340, 1994. Citado 10 vezes nas páginas [12](#), [13](#), [55](#), [58](#), [59](#), [60](#), [98](#), [101](#), [106](#) e [108](#).
- CHEN, G.; BALAKRISHNAN, N. A general purpose approximate goodness-of-fit test. *Journal of Quality Technology*, American Society for Quality, v. 27, n. 2, p. 154–161, 1995. Citado 2 vezes nas páginas [71](#) e [72](#).
- CHESHER, A.; JEWITT, I. The bias of a heteroskedasticity consistent covariance matrix estimator. *Econometrica: Journal of the Econometric Society*, JSTOR, v. 55, p. 1217–1222, 1987. Citado na página [36](#).
- CRAMÉR, H. On the composition of elementary errors. *Skand. Akt.*, v. 11, p. 141–180, 1928. Citado na página [71](#).
- CRIBARI-NETO, F. Asymptotic inference under heteroskedasticity of unknown form. *Computational Statistics & Data Analysis*, Elsevier, v. 45, n. 2, p. 215–233, 2004. Citado 3 vezes nas páginas [17](#), [31](#) e [37](#).
- CRIBARI-NETO, F.; FERRARI, S. L.; CORDEIRO, G. M. Improved heteroscedasticity-consistent covariance matrix estimators. *Biometrika*, v. 87, n. 4, p. 907–918, 2000. Citado 2 vezes nas páginas [17](#) e [36](#).

- CRIBARI-NETO, F.; FERRARI, S. L.; OLIVEIRA, W. A. Numerical evaluation of tests based on different heteroskedasticity-consistent covariance matrix estimators. *Journal of Statistical Computation and Simulation*, Taylor & Francis, v. 75, n. 8, p. 611–628, 2005. Citado na página [37](#).
- CRIBARI-NETO, F.; GALVÃO, N. M. A class of improved heteroskedasticity-consistent covariance matrix estimators. *Communications in Statistics-Theory and Methods*, Taylor & Francis, v. 32, n. 10, p. 1951–1980, 2003. Citado 2 vezes nas páginas [17](#) e [36](#).
- CRIBARI-NETO, F.; LIMA, M. da G. A. Heteroskedasticity-consistent interval estimators. *Journal of Statistical Computation and Simulation*, v. 79, p. 787–803, 2009. Citado 2 vezes nas páginas [18](#) e [38](#).
- CRIBARI-NETO, F.; SOUZA, T. C.; VASCONCELLOS, K. L. Inference under heteroskedasticity and leveraged data. *Communications in Statistics-Theory and Methods*, Taylor & Francis, v. 36, n. 10, p. 1877–1888, 2007, Errata: v. 37, n. 20, p. 3329–3330, 2008. Citado 3 vezes nas páginas [18](#), [31](#) e [37](#).
- CRIBARI-NETO, F.; ZARKOS, S. G. Bootstrap methods for heteroskedastic regression models: evidence on estimation and testing. *Econometric Reviews*, Taylor & Francis, v. 18, n. 2, p. 211–228, 1999. Citado na página [36](#).
- CRIBARI-NETO, F.; ZARKOS, S. G. Heteroskedasticity-consistent covariance matrix estimation: white's estimator and the bootstrap. *Journal of Statistical Computation and Simulation*, Taylor & Francis, v. 68, n. 4, p. 391–411, 2001. Citado na página [36](#).
- DAVIDSON, R.; MACKINNON, J. G. *Estimation and Inference in Econometrics*. New York: Oxford University Press, 1993. Citado 3 vezes nas páginas [17](#), [31](#) e [37](#).
- DAVISON, A.; HINKLEY, D. *Bootstrap methods and their application*. Cambridge: Cambridge University Press, 1997. Citado 2 vezes nas páginas [43](#) e [44](#).
- EFRON, B. Bootstrap methods: another look at the jackknife. *The Annals of Statistics*, v. 7, p. 1–26, 1979. Citado na página [39](#).
- EFRON, B. *The jackknife, the bootstrap and other resampling plans*. Philadelphia: SIAM, 1982. Citado na página [46](#).
- EFRON, B.; TIBSHIRANI, R. *An introduction to the bootstrap*. New York: CRC press, 1993. Citado na página [42](#).
- GREENE, W. H. *Econometric analysis, 3rd edition*. New York: Prentice Hall, 1997. Citado na página [98](#).
- HINKLEY, D.; WEI, B.-C. Improvements of jackknife confidence limit methods. *Biometrika*, v. 71, n. 2, p. 331–339, 1984. Citado na página [46](#).
- HORN, S. D.; HORN, R. A.; DUNCAN, D. B. Estimating heteroscedastic variances in linear models. *Journal of the American Statistical Association*, v. 70, p. 380–385, 1975. Citado 2 vezes nas páginas [17](#) e [36](#).
- JUDGE, G.; HILL, R.; GRIFFITHS, W. E.; LUTKEPOHL, H.; LEE, T. Introduction to the theory and practice of econometrics. 1988. Citado na página [35](#).

- LEMONTE, A. J. A new exponential-type distribution with constant, decreasing, increasing, upside-down bathtub and bathtub-shaped failure rate function. *Computational Statistics & Data Analysis*, Elsevier, 2013. Citado na página 72.
- MACKINNON, J. G.; WHITE, H. Some heteroskedasticity-consistent covariance matrix estimators with improved finite sample properties. *Journal of Econometrics*, v. 29, p. 305–325, 1985. Citado 3 vezes nas páginas 17, 31 e 36.
- RAMOS, W. A.; CORDEIRO, G. M.; MARINHO, P. R. D.; DIAS, C.R.B.; HAMEDANI, G.G. The zografos-balakrishnan log-logistic distribution: Properties and applications. *Journal of Statistical Theory and Applications*, Atlantis Press, v. 12, n. 3, p. 225–244, 2013. Citado na página 72.
- RAMOS, W. A.; MARINHO, P. R. D.; SILVA, R. V.; CORDEIRO, G.M. The exponentiated lomax poisson distribution with an application to lifetime data. *Advances and Applications in Statistics*, Pushpa Publishing House, v. 34, n. 2, p. 107–135, 2013. Citado na página 72.
- STEPHENS, M. A. Tests based on EDF statistics. *Goodness-of-Fit Techniques*, RB D'Agostino and MS Stephens, Eds. Marcel Dekker, New York, 1986. Citado na página 71.
- von Mises, R. *Wahrscheinlichkeitsrechnung und ihre Anwendung in der Statistik und theoretischen Physik*. Leipzig: Deuticke, 1931. Citado na página 71.
- WHITE, H. A heteroskedasticity-consistent covariance matrix estimator and a direct test for heteroskedasticity. *Econometrica*, v. 48, p. 817–838, 1980. Citado 3 vezes nas páginas 17, 31 e 36.
- WU, C. F. J. Jackknife, bootstrap and other resampling methods in regression analysis. *Annals of Statistics*, v. 14, p. 1261–1295, 1986. Citado 4 vezes nas páginas 18, 19, 51 e 97.

Estatísticas Cramér-von Mises (W^*) e Anderson-Darling (A^*) para a amostra aleatória $b_m = b_1, b_2, \dots, b_m$

A.1 Estimativa intervalar HC0

A.1.1 Desenho balanceado

Tabela 36 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos balanceados verificando o ajuste da distribuição normal padrão à amostra aleatória $b_m, m = 1, 2, \dots, 10000$.

λ	Distribuições	n					
		20		60		100	
		W^*	A^*	W^*	A^*	W^*	A^*
$\lambda = 1$	$\mathcal{N}(0, 1)$	0.7197	4.8149	0.1351	0.9163	0.0898	0.6345
	$t_{(3)}$	0.0861	0.5977	0.0649	0.4510	0.1749	1.0011
	$\chi^2_{(2)}$	0.0649	0.5350	0.1199	0.6840	0.0119	0.1026
	Weibull(2, 3)	0.3224	2.1111	0.1518	1.0848	0.0502	0.3846
$\lambda \approx 49$	$\mathcal{N}(0, 1)$	0.9310	5.7795	0.2552	1.7234	0.1012	0.7406
	$t_{(3)}$	0.1234	1.1059	0.0924	0.5621	0.1162	0.6794
	$\chi^2_{(2)}$	11.1227	66.3151	4.6095	28.5920	3.2399	19.8019
	Weibull(2, 3)	0.4617	3.1603	0.2277	1.5157	0.0923	0.6307

1 - $\mathcal{H}_0 : b_1, b_2, \dots, b_{10000}$ segue distribuição normal padrão;

2 - As simulações aqui apresentadas referem-se ao caso balanceado.

Tabela 37 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos balanceados verificando o ajuste da distribuição normal padrão à amostra aleatória $b_m, m = 1, 2, \dots, 10000$.

λ	Distribuições	n					
		20		60		100	
		W^*	A^*	W^*	A^*	W^*	A^*
$\lambda = 1$	$\mathcal{N}(0, 1)$	0.7153	4.7862	0.1340	0.9090	0.0892	0.6298
	$t_{(3)}$	0.0867	0.5997	0.0655	0.4544	0.1758	1.0057
	$\chi^2_{(2)}$	0.0651	0.5346	0.1199	0.6833	0.0118	0.1023
	Weibull(2, 3)	0.3198	2.0944	0.1510	1.0794	0.0497	0.3816
$\lambda \approx 49$	$\mathcal{N}(0, 1)$	0.9259	5.7477	0.2536	1.7127	0.1005	0.7359
	$t_{(3)}$	0.1226	1.0980	0.0932	0.5660	0.1168	0.6832
	$\chi^2_{(2)}$	11.0951	66.1556	4.6007	28.5383	3.2327	19.7599
	Weibull(2, 3)	0.4586	3.1389	0.2261	1.5057	0.0919	0.6272

1 - $\mathcal{H}_0 : b_1, b_2, \dots, b_{10000}$ segue distribuição $t_{(n-2)}$;

2 - As simulações aqui apresentadas referem-se ao caso balanceado.

A.1.2 Desenho não balanceado

Tabela 38 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos não balanceados verificando o ajuste da distribuição normal padrão à amostra aleatória $b_m, m = 1, 2, \dots, 10000$.

λ	Distribuições	n					
		20		60		100	
		W^*	A^*	W^*	A^*	W^*	A^*
$\lambda = 1$	$\mathcal{N}(0, 1)$	2.8716	17.2018	0.9937	6.6065	0.3305	2.4694
	$t_{(3)}$	1.5251	10.1473	0.0763	0.6549	0.1748	1.0624
	$\chi^2_{(2)}$	6.9310	40.6847	0.6316	3.6690	1.7990	9.9434
	Weibull(2, 3)	2.5688	15.3573	1.0482	6.4367	0.3940	2.6071
$\lambda \approx 49$	$\mathcal{N}(0, 1)$	3.4760	19.3682	0.6620	3.8600	0.1976	1.1684
	$t_{(3)}$	2.0676	12.6600	1.5930	9.5358	1.2523	7.0894
	$\chi^2_{(2)}$	1.5692	14.8589	2.8274	16.8843	3.0248	18.3131
	Weibull(2, 3)	3.3061	18.2303	0.7930	4.9770	0.7950	4.9900

1 - $\mathcal{H}_0 : b_1, b_2, \dots, b_{10000}$ segue distribuição normal padrão;

2 - As simulações aqui apresentadas referem-se ao caso não balanceado.

Tabela 39 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos não balanceados verificando o ajuste da distribuição $t_{(n-2)}$ à amostra aleatória $b_m, m = 1, 2, \dots, 10000$.

λ	Distribuições	n					
		20		60		100	
		W^*	A^*	W^*	A^*	W^*	A^*
$\lambda = 1$	$\mathcal{N}(0, 1)$	0.1145	0.7415	0.0976	0.5669	0.1221	0.4526
	$t_{(3)}$	0.1099	0.7498	0.0761	0.6518	0.1157	0.6671
	$\chi^2_{(2)}$	0.9033	1.5334	0.6303	2.6611	0.7983	3.9396
	Weibull(2, 3)	1.5461	5.2192	1.0419	6.3972	0.3914	1.5898
$\lambda \approx 49$	$\mathcal{N}(0, 1)$	0.0481	0.5967	0.6638	3.8715	0.1985	1.1743
	$t_{(3)}$	0.8724	3.6784	0.5958	2.5530	1.2547	4.1044
	$\chi^2_{(2)}$	0.5679	4.8216	0.8293	1.8960	0.0248	0.3137
	Weibull(2, 3)	0.3120	1.2593	0.0544	0.4363	0.0436	0.3146

1 - $\mathcal{H}_0 : b_1, b_2, \dots, b_{10000}$ segue distribuição $t_{(n-2)}$;

2 - As simulações aqui apresentadas referem-se ao caso não balanceado.

A.2 Intervalo HC2

A.2.1 Desenho balanceado

Tabela 40 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos balanceados verificando o ajuste da distribuição normal padrão à amostra aleatória $b_m, m = 1, 2, \dots, 10000$.

λ	Distribuições	n					
		20		60		100	
		W^*	A^*	W^*	A^*	W^*	A^*
$\lambda = 1$	$\mathcal{N}(0, 1)$	0.7537	5.0456	0.1365	0.9257	0.0896	0.6324
	$t_{(3)}$	0.0869	0.6097	0.0643	0.4473	0.1748	0.9997
	$\chi^2_{(2)}$	0.0551	0.4888	0.1308	0.7441	0.0124	0.1052
	Weibull(2, 3)	0.3456	2.2598	0.1532	1.0929	0.0502	0.3841
$\lambda \approx 49$	$\mathcal{N}(0, 1)$	0.9092	5.6461	0.2546	1.7200	0.1012	0.7409
	$t_{(3)}$	0.1214	1.0887	0.0932	0.5659	0.1163	0.6801
	$\chi^2_{(2)}$	10.9165	65.1336	4.6077	28.5779	3.2398	19.8004
	Weibull(2, 3)	0.4441	3.0410	0.2273	1.5133	0.0924	0.6312

1 - $\mathcal{H}_0 : b_1, b_2, \dots, b_{10000}$ segue distribuição normal padrão;

2 - As simulações aqui apresentadas referem-se ao caso balanceado.

Tabela 41 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos balanceados verificando o ajuste da distribuição $t_{(n-2)}$ à amostra aleatória $b_m, m = 1, 2, \dots, 10000$.

λ	Distribuições	n					
		20		60		100	
		W^*	A^*	W^*	A^*	W^*	A^*
$\lambda = 1$	$\mathcal{N}(0, 1)$	0.7498	5.0199	0.1354	0.9187	0.0902	0.6370
	$t_{(3)}$	0.0874	0.6113	0.0648	0.4506	0.1756	1.0041
	$\chi^2_{(2)}$	0.0552	0.4880	0.1308	0.7434	0.0123	0.1049
	Weibull(2, 3)	0.3433	2.2448	0.1524	1.0876	0.0498	0.3811
$\lambda \approx 49$	$\mathcal{N}(0, 1)$	0.9048	5.6187	0.2531	1.7097	0.1005	0.7363
	$t_{(3)}$	0.1208	1.0821	0.0939	0.5697	0.1169	0.6838
	$\chi^2_{(2)}$	10.8929	64.9977	4.5993	28.5265	3.2328	19.7595
	Weibull(2, 3)	0.4414	3.0227	0.2258	1.5037	0.0920	0.6278

1 - $\mathcal{H}_0 : b_1, b_2, \dots, b_{10000}$ segue distribuição $t_{(n-2)}$;

2 - As simulações aqui apresentadas referem-se ao caso balanceado.

A.2.2 Desenho não balanceado

Tabela 42 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos não balanceados verificando o ajuste da distribuição normal padrão à amostra aleatória $b_m, m = 1, 2, \dots, 10000$.

λ	Distribuições	n					
		20		60		100	
		W^*	A^*	W^*	A^*	W^*	A^*
$\lambda = 1$	$\mathcal{N}(0, 1)$	5.5543	32.2310	1.2095	7.9442	0.3694	2.7296
	$t_{(3)}$	7.5716	45.9578	0.0829	0.7245	0.1677	1.0237
	$\chi^2_{(2)}$	6.8282	40.0591	1.0716	6.2123	2.1148	11.7119
	Weibull(2, 3)	5.5687	32.5571	1.2865	7.8327	0.4504	2.9679
$\lambda \approx 49$	$\mathcal{N}(0, 1)$	4.1137	22.7088	0.8048	4.7001	0.2273	1.3494
	$t_{(3)}$	2.5182	16.0969	1.7716	10.6196	1.3261	7.5287
	$\chi^2_{(2)}$	2.6730	23.4897	2.9249	17.4215	2.9942	18.1215
	Weibull(2, 3)	4.2957	23.4228	0.9425	5.9097	0.5110	2.9926

1 - $\mathcal{H}_0 : b_1, b_2, \dots, b_{10000}$ segue distribuição normal padrão;

2 - As simulações aqui apresentadas referem-se ao caso não balanceado.

Tabela 43 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos não balanceados verificando o ajuste da distribuição $t_{(n-2)}$ à amostra aleatória $b_m, m = 1, 2, \dots, 10000$.

λ	Distribuições	n					
		20		60		100	
		W^*	A^*	W^*	A^*	W^*	A^*
$\lambda = 1$	$\mathcal{N}(0, 1)$	0.5271	2.0743	1.2095	2.9442	0.0369	0.7131
	$t_{(3)}$	0.0544	0.7103	0.0826	0.7205	0.1185	0.7113
	$\chi_{(2)}^2$	0.8053	9.2135	0.0698	0.2016	0.0114	0.7033
	Weibull(2, 3)	0.5368	2.3701	1.2802	7.7932	0.4478	1.9506
$\lambda \approx 49$	$\mathcal{N}(0, 1)$	0.1175	0.7265	0.1066	0.6115	0.0998	0.3555
	$t_{(3)}$	0.5210	2.1028	0.7742	1.6359	0.1215	0.5432
	$\chi_{(2)}^2$	1.6686	3.4361	1.9269	7.4340	0.9944	8.1232
	Weibull(2, 3)	0.3002	3.4422	0.9444	5.9223	0.5124	3.0020

1 - $\mathcal{H}_0 : b_1, b_2, \dots, b_{10000}$ segue distribuição $t_{(n-2)}$;

2 - As simulações aqui apresentadas referem-se ao caso não balanceado.

A.3 Intervalo HC3

A.3.1 Desenho balanceado

Tabela 44 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos balanceados verificando o ajuste da distribuição normal padrão à amostra aleatória $b_m, m = 1, 2, \dots, 10000$.

λ	Distribuições	n					
		20		60		100	
		W^*	A^*	W^*	A^*	W^*	A^*
$\lambda = 1$	$\mathcal{N}(0, 1)$	0.7918	5.3032	0.1379	0.9354	0.0906	0.6394
	$t_{(3)}$	0.0866	0.6160	0.0635	0.4433	0.1746	0.9981
	$\chi_{(2)}^2$	0.0553	0.5021	0.1425	0.8089	0.0130	0.1089
	Weibull(2, 3)	0.3721	2.4288	0.1546	1.1010	0.0502	0.3836
$\lambda \approx 49$	$\mathcal{N}(0, 1)$	0.8848	5.4964	0.2540	1.7164	0.1012	0.7411
	$t_{(3)}$	0.1193	1.0694	0.0939	0.5696	0.1164	0.6808
	$\chi_{(2)}^2$	10.6935	63.8527	4.6053	28.5600	3.2396	19.7981
	Weibull(2, 3)	0.4251	2.9139	0.2269	1.5108	0.0925	0.6316

1 - $\mathcal{H}_0 : b_1, b_2, \dots, b_{10000}$ segue distribuição normal padrão;

2 - As simulações aqui apresentadas referem-se ao caso não balanceado.

Tabela 45 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos balanceados verificando o ajuste da distribuição $t_{(n-2)}$ à amostra aleatória $b_m, m = 1, 2, \dots, 10000$.

λ	Distribuições	n					
		20		60		100	
		W^*	A^*	W^*	A^*	W^*	A^*
$\lambda = 1$	$\mathcal{N}(0, 1)$	0.7882	5.2802	0.1369	0.9288	0.0900	0.6349
	$t_{(3)}$	0.0870	0.6172	0.0641	0.4464	0.1755	1.0023
	$\chi^2_{(2)}$	0.0554	0.5010	0.1424	0.8082	0.0130	0.1086
	Weibull(2, 3)	0.3700	2.4152	0.1538	1.0959	0.0498	0.3807
$\lambda \approx 49$	$\mathcal{N}(0, 1)$	0.8810	5.4727	0.2526	1.7066	0.1006	0.7366
	$t_{(3)}$	0.1188	1.0639	0.0946	0.5733	0.1170	0.6844
	$\chi^2_{(2)}$	10.6736	63.7372	4.5972	28.5107	3.2328	19.7582
	Weibull(2, 3)	0.4228	2.8984	0.2255	1.5016	0.0921	0.6283

1 - $\mathcal{H}_0 : b_1, b_2, \dots, b_{10000}$ segue distribuição $t_{(n-2)}$;

2 - As simulações aqui apresentadas referem-se ao caso não balanceado.

A.3.2 Desenho não balanceado

Tabela 46 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos não balanceados verificando o ajuste da distribuição normal padrão à amostra aleatória $b_m, m = 1, 2, \dots, 10000$.

λ	Distribuições	n					
		20		60		100	
		W^*	A^*	W^*	A^*	W^*	A^*
$\lambda = 1$	$\mathcal{N}(0, 1)$	0.2985	3.2752	0.7669	3.5261	1.4129	6.0177
	$t_{(3)}$	2.0571	9.5789	0.5967	3.6497	0.1088	0.7090
	$\chi^2_{(2)}$	0.1459	2.2682	0.6435	1.0513	0.4613	3.6604
	Weibull(2, 3)	1.9904	8.9205	0.5730	2.5072	0.5130	3.3668
$\lambda \approx 49$	$\mathcal{N}(0, 1)$	0.0742	0.6948	0.0976	0.7122	0.0262	0.5626
	$t_{(3)}$	2.7090	2.1210	1.9725	1.8410	1.4061	1.0049
	$\chi^2_{(2)}$	4.8768	8.9558	3.0541	8.1504	2.9644	7.9372
	Weibull(2, 3)	4.2375	3.8836	1.1175	6.9944	0.5644	3.3264

1 - $\mathcal{H}_0 : b_1, b_2, \dots, b_{10000}$ segue distribuição normal padrão;

2 - As simulações aqui apresentadas referem-se ao caso não balanceado.

Tabela 47 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos não balanceados verificando o ajuste da distribuição $t_{(n-2)}$ à amostra aleatória $b_m, m = 1, 2, \dots, 10000$.

λ	Distribuições	n					
		20		60		100	
		W^*	A^*	W^*	A^*	W^*	A^*
$\lambda = 1$	$\mathcal{N}(0, 1)$	0.1003	1.1189	1.4610	9.4882	0.4106	3.0015
	$t_{(3)}$	0.5442	5.8003	0.0962	0.8449	0.1595	0.9824
	$\chi_{(2)}^2$	0.7339	1.1480	1.6413	9.4993	2.4601	13.6530
	Weibull(2, 3)	0.3546	6.5025	0.3666	4.4678	0.5104	3.3497
$\lambda \approx 49$	$\mathcal{N}(0, 1)$	0.7446	2.6993	0.9778	5.7163	0.2631	1.5689
	$t_{(3)}$	0.1089	0.7498	0.0950	1.8563	0.1084	0.6019
	$\chi_{(2)}^2$	0.8654	3.8704	0.1562	1.1632	0.2648	1.0400
	Weibull(2, 3)	0.2398	3.8893	0.1194	2.0066	0.5659	1.3357

1 - $\mathcal{H}_0 : b_1, b_2, \dots, b_{10000}$ segue distribuição $t_{(n-2)}$;

2 - As simulações aqui apresentadas referem-se ao caso não balanceado.

A.4 Intervalo HC5

A.4.1 Desenho balanceado

Tabela 48 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos balanceados verificando o ajuste da distribuição normal padrão à amostra aleatória $b_m, m = 1, 2, \dots, 10000$.

λ	Distribuições	n					
		20		60		100	
		W^*	A^*	W^*	A^*	W^*	A^*
$\lambda = 1$	$\mathcal{N}(0, 1)$	1.7650	5.1246	0.1370	0.9291	0.0903	0.6377
	$t_{(3)}$	1.0869	6.6121	1.0641	3.4463	0.1747	0.9990
	$\chi_{(2)}^2$	0.8538	8.4838	0.1367	0.7765	0.4127	3.1070
	Weibull(2, 3)	0.3543	2.3141	0.1537	1.0959	0.8502	3.3835
$\lambda \approx 49$	$\mathcal{N}(0, 1)$	0.3321	1.3443	0.2453	2.0353	0.4345	0.5536
	$t_{(3)}$	0.9876	1.4345	0.5424	0.6443	0.7643	0.6564
	$\chi_{(2)}^2$	2.5547	12.4353	1.4343	6.5534	1.3454	7.3434
	Weibull(2, 3)	1.5434	4.8981	0.7634	0.7854	0.4542	0.6435

1 - $\mathcal{H}_0 : b_1, b_2, \dots, b_{10000}$ segue distribuição normal padrão;

2 - As simulações aqui apresentadas referem-se ao caso não balanceado.

Tabela 49 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos balanceados verificando o ajuste da distribuição $t_{(n-2)}$ à amostra aleatória $b_m, m = 1, 2, \dots, 10000$.

λ	Distribuições	n					
		20		60		100	
		W^*	A^*	W^*	A^*	W^*	A^*
$\lambda = 1$	$\mathcal{N}(0, 1)$	0.7609	5.0977	0.1359	0.9220	0.0897	0.6331
	$t_{(3)}$	0.0875	0.6138	0.0646	0.4496	0.1756	1.0035
	$\chi^2_{(2)}$	0.0539	0.4828	0.1366	0.7758	0.0126	0.1067
	Weibull(2, 3)	0.3519	2.2982	0.1530	1.0905	0.0497	0.3805
$\lambda \approx 49$	$\mathcal{N}(0, 1)$	0.3450	3.3523	0.3443	7.3463	0.7643	4.3464
	$t_{(3)}$	0.2342	0.5340	0.5460	0.4467	0.2454	0.5436
	$\chi^2_{(2)}$	0.7533	1.4530	0.8435	1.6545	0.9753	1.3456
	Weibull(2, 3)	1.4575	6.4527	1.5463	5.5764	1.5642	4.3463

1 - $\mathcal{H}_0 : b_1, b_2, \dots, b_{10000}$ segue distribuição $t_{(n-2)}$;

2 - As simulações aqui apresentadas referem-se ao caso não balanceado.

A.4.2 Desenho não balanceado

Tabela 50 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos não balanceados verificando o ajuste da distribuição normal padrão à amostra aleatória $b_m, m = 1, 2, \dots, 10000$.

λ	Distribuições	n					
		20		60		100	
		W^*	A^*	W^*	A^*	W^*	A^*
$\lambda = 1$	$\mathcal{N}(0, 1)$	0.4348	5.4335	0.1450	3.5463	0.5455	1.6733
	$t_{(3)}$	0.6543	5.4535	0.5463	3.5435	0.4422	1.3465
	$\chi^2_{(2)}$	0.4364	1.3453	0.4324	2.3456	0.4653	1.4362
	Weibull(2, 3)	0.8445	7.4532	0.4563	1.4364	0.0452	0.2134
$\lambda \approx 49$	$\mathcal{N}(0, 1)$	0.2344	0.9834	0.4567	1.4565	0.1234	0.5435
	$t_{(3)}$	0.5335	20.5436	1.5463	10.4632	0.8754	9.5435
	$\chi^2_{(2)}$	0.6456	5.6532	0.6345	7.4353	0.5623	2.3456
	Weibull(2, 3)	0.3462	4.5634	0.6230	4.5643	0.2433	1.3456

1 - $\mathcal{H}_0 : b_1, b_2, \dots, b_{10000}$ segue distribuição normal padrão;

2 - As simulações aqui apresentadas referem-se ao caso não balanceado.

Tabela 51 – Estatísticas de Cramér-von Mises (W^*) e Anderson-Darling (A^*) para desenhos não balanceados verificando o ajuste da distribuição $t_{(n-2)}$ à amostra aleatória $b_m, m = 1, 2, \dots, 10000$.

λ	Distribuições	n					
		20		60		100	
		W^*	A^*	W^*	A^*	W^*	A^*
$\lambda = 1$	$\mathcal{N}(0, 1)$	0.0546	0.4235	0.0453	0.3553	0.06033	0.3453
	$t_{(3)}$	0.1223	0.5563	0.1143	0.6543	0.1133	0.6454
	$\chi^2_{(2)}$	0.6543	1.4235	0.5473	0.4564	0.5536	0.43743
	Weibull(2, 3)	0.1454	0.3473	0.6534	0.8643	1.4655	3.5653
$\lambda \approx 49$	$\mathcal{N}(0, 1)$	0.0456	0.5324	0.0335	0.3455	0.0533	0.5432
	$t_{(3)}$	0.1424	1.5435	0.5643	3.5343	0.4533	1.6534
	$\chi^2_{(2)}$	0.2434	0.8353	0.3456	4.2453	0.7643	3.4564
	Weibull(2, 3)	1.4536	4.6535	1.7639	10.5356	0.45623	8.3452

1 - $\mathcal{H}_0 : b_1, b_2, \dots, b_{10000}$ segue distribuição $t_{(n-2)}$;

2 - As simulações aqui apresentadas referem-se ao caso não balanceado.

Programa - Avaliação das estimativas intervalares

Esse apêndice apresenta o código fonte do programa escrito na linguagem de programação C++ utilizado para realizar as simulações contidas nos resultados desse trabalho. O programa realiza simulações de Monte Carlo para avaliar as coberturas das estimativas intervalares para os parâmetros que indexam o modelo linear de regressão com heteroscedasticidade de forma desconhecida. São consideradas as estimativas intervalares sem uso de métodos de bootstrap utilizando os estimadores da matriz de covariância de $\hat{\beta}$ (HC0, HC2, HC3, HC4 e HC5) com os quantis obtidos das distribuições normal padrão e t de Student com $n - p$ graus de liberdade. Também são considerados no mesmo código os estimadores intervalares bootstrap percentil e bootstrap- t em esquemas simples e duplo. É possível também definir o nível de heteroscedasticidade que serão consideradas nas simulações, escolher o tamanho da amostra e a distribuição de probabilidade para os erros. Também é permitido escolher entre as distribuições normal padrão e Rademacher para gerar os valores t^* e t^{**} , $i = 1, \dots, n$ considerados no bootstrap selvagem e também escolher entre esquemas balanceados e não balanceados, ou seja, dados sem pontos de alavancas e dados com pontos de alta alavancagem ($h_i > 3p/n$).

Esse programa foi compilado utilizando o compilador g++ versão 4.8.1. Também foram utilizadas além das bibliotecas padrão da linguagem de programação C++, a biblioteca GSL versão 1.15 e a biblioteca Armadillo versão 3.900.6. Para utilização desse código, aconselha-se utilizar a biblioteca OpenBLAS que tem uma performance computacional melhor que a biblioteca BLAS. Outras otimizações de BLAS podem ser utilizadas a depender dos hardwares disponíveis. Esse código foi compilado e executado no cluster SIG Altix - Gauss do CESUP utilizando a biblioteca MKL.

B.1 Código C++

```

1  /* *****
2      Modelos de Regressao Linear com
3      Heteroscedasticidade de Forma Desconhecida
4
5      Intervalos de Confianca Bootstrap simples e duplo
6
7  Esse programa realiza simulacoes de Monte Carlo das estimativas
8  intervalares em modelos lineares com heteroscedasticidade de
9  forma desconhecida. Tambem sao feitas simulacoes de Monte Carlo
10 das estimativas intervalares utilizando os estimadores HCO, HC2,
11 HC3, HC4 e HC5 sem uso de bootstrap. As estimativas intervalares
12 por bootstrap-t simples e bootstrap-t duplo tambem fizeram uso
13 dos estimadores HCO, HC2, HC3, HC4 e HC5.
14
15 Foram calculadas as coberturas das estimativas intervalares utili-
16 zando os niveis nominais de confiancas de 90%, 95% e 99%. Nesse
17 codigo apenas sera preciso informar o nivel de heteroscedasticidade
18 que o valor da constante "a" sera obtido automaticamente. Podera ser
19 escolhido 6 distribuicoes para os erros. Essas distribuicoes sao:
20 normal, t(3), chi-squared(2), weibull(2,3), gumbel type II (2.5,2) e
21 gamma(3,1.5). Tambem e possivel utilizar os geradores de numeros
22 pseudo-aleatorios fornecidos pela biblioteca GSL. Tais geradores
23 sao: tt800, mt19937, random256_bsd. Nesse trabalho foi utilizado
24 o gerador mt19937 de de Makoto Matsumoto e de Takuji Nishimura.
25
26 As estimativas intervalares avaliadas consideram os estimadores:
27
28 (1) HCO com quantis da distribuicao t e da distribuicao normal;
29 (2) HC2 com quantis da distribuicao t e da distribuicao normal;
30 (3) HC3 com quantis da distribuicao t e da distribuicao normal;
31 (4) HC4 com quantis da distribuicao t e da distribuicao normal;
32 (5) HC5 com quantis da distribuicao t e da distribuicao normal;
33 (6) Bootstrap percentil simpels (um nivel de bootstrap);
34 (7) Bootstrap percentil duplo (dois niveis de bootstrap);
35 (8) Bootstrap-t simples (um nivel de bootstrap) utilizando os estimadores
36 HCO, HC2, HC3, HC4 e HC5;
37 (9) Bootstrap-t duplo (dois niveis de bootstrap) utilizando os estimadores
38 HCO, HC2, HC3, HC4 e HC5.
39
40 Todas as avaliacoes das estimativas intervalares descritas acima levam
41 em consideracao tres tamanhos de amostras (n=20, n=60 e n=100) para os
42 niveis de confiancas 90%, 95% e 99%.
43
44 Tambem e possivel escolher o nivel de heteroscedasticidade e se o esquema
45 sera balanceado ou nao balanceado, ou seja, com presenca e ausencia de
46 pontos de alta alavancagem.
47
48 =====
49 Orientando: Pedro Rafael Diniz Marinho;
50 E-mail - pedro.rafael.marinho@gmail.com;
51 Mestrado em Estatistica - UFPE.
52
53 Orientador: Francisco Cribari Neto;
54 E-mail: cribari@de.ufpe.br;
55
56 ***** */

```

```
57
58 // NOTAS SOBRE O PROGRAMA:
59
60 // Esse programa faz a avaliacao dos intervalos de confiancas
61 // sem utilizar esquemas bootstrap e tambem utilizando esquemas
62 // bootstarp: bootstrap percetil, bootstrap percentil duplo,
63 // bootstrap t e bootstrap t duplo para todos os HC's: HC0, HC2,
64 // HC3, HC4 e HC5. e evidente que o bootstrap percentil e bootstrap
65 // duplo percentil nao faz uso dos estimadores HC.
66
67 // Esse programa faz uso de duas correcoes. Uma correcao na quanti-
68 // dade do denominador da variavel  $z^*$  e a outra correcao corrige o
69 // desvio que entra no calculo dos limites do intervalo de confinaca.
70 // Esse esquema nao e correto mas foi mantido nesse codigo fonte. 0
71 // esquema correto do bootstrap t duplo foi acrescentado. Esse esquema
72 // foi inspirado no algoritmo do artigo B.D.,McCULLOUGH, H.D.,VINOD.
73 // Implementing the Double Bootstrap, Computational Economics, 12,
74 // 79-95, 1998.
75
76
77 /* Versoes das biliotecas utilizadas
78     Armadillo - versao 3.900.6
79     g++ - versao 4.8.1
80     GSL - 1.15
81 */
82
83 /*Compilando o codigo usando a biblioteca armadillo, openblas e gsl.
84 O usuario deve se certificar que a biblioteca armadillo esta confi-
85 gurada para reconhecer o openblas. Caso nao esteja reconhecendo da-
86 ra um erro o uso da flag -lopenblas. O usuario tem duas opcao:
87 compilar sem usar a flag ou instalar a biblioteca openblas que e
88 mais otimizada. Comando para compilacao: g++ -O3 -o 04 C04.cpp
89 -lopenblas -lgsl -larmadillo
90 */
91
92 /*Instalacao da biblioteca armadillo no GNU/Linux
93 Ubuntu: apt-get install libarmadillo2 && libarmadillo-dev
94 Fedora: yum install armadillo
95 Manjaro: yaourt -S armadillo
96 Site: http://arma.sourceforge.net/
97
98 Para instalar a biblioteca pelo codigo fonte siga os seguintes passos:
99 (1) cmake
100 (2) Na pasta da biblioteca execute:
101 (2.1) ./configure
102 (2.2) make
103 (2.3) make install
104 */
105
106 /*A compilacao da biblioteca blas pode ser 32-bits. Logo, nao sera
107 eficiente a sua utilizacao. */
108 #define ARMA_DONT_USE_BLAS
109 //#define ARMA_USE_LAPACK
110
111 #include <iostream>
112 #include <omp.h>
113 #include <sstream>
114 #include <string.h>
```

```

115 #include <fstream> /* Biblioteca para leitura e escrita em arquivos */
116 #include <math.h>
117 #include <gsl/gsl_rng.h>
118 #include <gsl/gsl_randist.h>
119 #include <gsl/gsl_statistics.h>
120 #include <gsl/gsl_cdf.h>
121 #include <time.h>
122 #include "armadillo" /* Biblioteca de Algebra Linear para C++ */
123
124 using namespace arma;
125 using namespace std;
126
127 namespace myfunctions{
128     double quantil(vec dados, double p, int n){
129         vec xx = sort(dados);
130         double x[n];
131         for(int i = 0; i < n; i++){
132             x[i] = xx(i);
133         }return gsl_stats_quantile_from_sorted_data(x, 1, n, p);
134     }
135     // ESSE QUANTIL e O QUE e DESCRITO NA MAIORIA DOS ALGORITMOS BOOTSTRAP.
136     // ESSE QUANTIL PEGA A POSICAO (n+1)*alpha, EM QUE alfa e O PERCENTIL DE
137     // INTERESSE.
138     double quantil1(vec dados, double p, int n){
139         vec xx = sort(dados);
140         int indice = floor((n + 1) * p);
141         if(indice < 0)
142             indice = 0;
143         if(indice >= n)
144             indice = n - 1;
145         double resul = xx(indice);
146         return resul;
147     }
148 }
149
150 // (1) Esse namespace tata-se de funcoes gerais que nao estao implementadas
151 // na biblioteca armadillo.
152 // (2) Apesar de algumas das funcoes implementadas nesse namespace estarem
153 // implementadas em bibliotecas como por exemplo a GSL, as funcoes aqui im-
154 // plementadas podem trabalhar diretamente com os tipos de dados
155 // suportados pela biblioteca armadillo.
156 // (3) As funcoes buscam ser de facil uso.
157 // (4) Informacoes para o uso das funcoes implementadas nesse namespace po-
158 // dem ser encontradas nos comentarios destas funcoes.
159
160 // Para rodar o programa apenas mude os valores das variaveis definidas no
161 // painel de controle definido logo abaixo:
162
163 int nrep = 5000; // NUMERO DE REPLICAS DE MONTE CARLO.
164 int nrep_boot = 1000; // NUMERO DE REPLICAS DO BOOTSTRAP T-PERCENTIL.
165 int nrep_boot_duplo = 500; // NUMERO DE REPLICAS DO BOOTSTRAP DUPLO T-PERCENTIL.
166 int samplesize = 5; // NUMERO DE REPLICACOES DA MATRIZ X. A MATRIZ X SERA REPLI-
167 // CADAS samplisize VEZES.
168 int nobs = 20; // NUMERO DE OBSERVACOES. SE esquema = 1, A MATRIZ X TERA nobs
169 // LINHAS. NO CASO EM QUE esquema = 2 A MATRIZ X TERA nobs*samplesize LINHAS.
170 int esquema = 2; // SE esquema = 1 A OPCA O samplesize SERA DESCONSIDERADA. DESSA
171 // FORMA, A SEGUNDA COLUNA DA MATRIX X SERA GERADA DIRETAMENTE DE UMA DISTRIBUICAO
172 // T COM 3 GRAUS DE LIBERDADE. CASO A ESCOLHA SEJA esquema = 2 GERAMOS INICIALMENTE

```



```

173 // UMA MATRIZ COM nobs LINHAS E POSTERIORMENTE REPLICAMOS ESSA MATRIZ samplesize
174 // VEZES.
175 double lambda = 49; // BASTA FIXAR O VALOR DE LAMBDA QUE O VALOR DA CONSTANTE "a"
176 // e ESCOLHIDO AUTOMATICAMENTE. ASSIM O VALOR DE LAMBDA TRABALHADO SERA MUITO
177 // PROXIMO AO VALOR DE LAMBDA ESCOLHIDO. POR EXEMPLO, PARA "lambda = 9" 0
178 // LAMBDA ESCOLHIDO E IGUAL A 9.00017.
179 int balanceado = 2; // BALANCEADO (SEM PONTOS DE ALAVANCA) SE balanceado = 1.
180 // NAO BALANCEADO, OU SEJA, COM PONTOS DE ALAVANCA SE balanceado = 2;
181 int dist_erro = 4; // ECOLHA DA DISTRIBUICAO DOS ERROS: 1: normal, 2: t(3);
182 // 3: chi-squared(2), 4: weibull(2,3), 5: gumbel type II (2.5,2), 6: gamma(3,1.5).
183 int dist_t = 1; // 1: rademacher, 2: normal padrao.
184 int ncorrecoes = 2; // NUMERO DE CORRECOES UTILIZADAS. SE ncorrecoes = 1
185 // APENAS O ERRO PADRAO (QUANTIDADE NO DENOMINADOR DA VARIaVEL z^{*}) SERA
186 // CORRIGIDO. PARA ISSO, E UTILIZADO O BOOTSTRAP INTERIOR PARA ESTIMATIVA
187 // DO VIES. SE ncorrecoes = 2, TAMBEM SERA CORRIGIDO O DESVIO QUE ENTRA NO
188 // CALCULO DO INTERVALO DE CONFIANCA. PARA ISSO, E UTILIZADO O BOOTSTAP EXTERIOR.
189
190 int main()
191 {
192     time_t rawtime;
193     struct tm *timeinfo;
194     time(&rawtime);
195     timeinfo = localtime(&rawtime);
196
197     const clock_t tempo_inicial = clock();
198
199     ofstream saida("n100_l49_ew_nb.txt");
200
201     // BANCO DE DADOS AUXILIARES. ESSES BANCOS DE DADOS SERAO UTILIZADOS
202     // PARA CONSTRUCAO DE GRAFICOS NA LINGUAGEM R. TODOS OS DADOS REFEREM-SE
203     // AO NIVEL NOMINAL DE 95%.
204     ofstream lils_ols("lim_inf_lim_sup_ols.txt");
205     ofstream lils_hc0("lim_inf_lim_sup_hc0.txt");
206     ofstream lils_hc2("lim_inf_lim_sup_hc2.txt");
207     ofstream lils_hc3("lim_inf_lim_sup_hc3.txt");
208     ofstream lils_hc4("lim_inf_lim_sup_hc4.txt");
209     ofstream lils_hc5("lim_inf_lim_sup_hc5.txt");
210
211     ofstream lils_percentil("lim_inf_lim_sup_percentil.txt");
212     ofstream lils_percentil_duplo("lim_inf_lim_sup_percentil_duplo.txt");
213     ofstream lils_hc0_bootstrapt("lim_inf_lim_sup_hc0_bootstrapt.txt");
214     ofstream lils_hc0_bootstrapt_duplo("lim_inf_lim_sup_hc0_bootstrapt_duplo.txt");
215     ofstream lils_hc2_bootstrapt("lim_inf_lim_sup_hc2_bootstrapt.txt");
216     ofstream lils_hc2_bootstrapt_duplo("lim_inf_lim_sup_hc2_bootstrapt_duplo.txt");
217     ofstream lils_hc3_bootstrapt("lim_inf_lim_sup_hc3_bootstrapt.txt");
218     ofstream lils_hc3_bootstrapt_duplo("lim_inf_lim_sup_hc3_bootstrapt_duplo.txt");
219     ofstream lils_hc4_bootstrapt("lim_inf_lim_sup_hc4_bootstrapt.txt");
220     ofstream lils_hc4_bootstrapt_duplo("lim_inf_lim_sup_hc4_bootstrapt_duplo.txt");
221     ofstream lils_hc5_bootstrapt("lim_inf_lim_sup_hc5_bootstrapt.txt");
222     ofstream lils_hc5_bootstrapt_duplo("lim_inf_lim_sup_hc5_bootstrapt_duplo.txt");
223
224     ofstream pivo_ols("pivo_ols.txt");
225     ofstream pivo_hc0("pivo_hc0.txt");
226     ofstream pivo_hc2("pivo_hc2.txt");
227     ofstream pivo_hc3("pivo_hc3.txt");
228     ofstream pivo_hc4("pivo_hc4.txt");
229     ofstream pivo_hc5("pivo_hc5.txt");
230

```

```

231 // VETOR DE UNS. VETOR COM OS PARAMETROS VERDADEIROS.
232
233 vec beta = ones<vec>(2);
234
235 // Definicao do gerador
236 gsl_rng *r;
237
238 //GERADOR UTILIZADO.
239
240 if(gerador==1){
241     r = gsl_rng_alloc(gsl_rng_tt800);
242 }
243 if(gerador==2){
244     r = gsl_rng_alloc (gsl_rng_mt19937);
245 }
246 if(gerador==3){
247     r = gsl_rng_alloc(gsl_rng_random256_bsd);
248 }
249
250 // DEFININDO SEMENTE DO GERADOR.
251
252 gsl_rng_set(r,semente);
253
254 mat X(nobs,2);
255
256 // PRIMEIRO ESQUEMA PARA GERACAO DA MATRIZ X.
257
258 if(esquema==1){
259     X = ones<mat>(nobs,2);
260     if(balanceado == 2){
261         for(int linhas = 0; linhas < nobs; linhas++) {
262             X(linhas,1) = gsl_ran_tdist(r,3);
263         }
264     }else{
265         int contando_influencia = 1;
266         while(contando_influencia!=0){
267             contando_influencia = 0;
268             double alavanca = 4.0/nobs;
269             for(int i = 0; i < nobs; i++) {
270                 X(i,1) = gsl_ran_tdist(r,3);
271             }
272             mat matriz_chapeu = X*inv(sympd(trans(X)*X))*trans(X);
273             vec diag_chapeu = diagvec(matriz_chapeu);
274             for(int i = 0; i < nobs;i++){
275                 if(diag_chapeu(i) > alavanca){
276                     contando_influencia = 1;
277                 }
278             }
279         }
280     }
281 }
282
283 if(esquema == 2){
284     X = ones <mat>(nobs,2);
285     if(balanceado == 2){
286         for(int linhas = 0; linhas < nobs; linhas++){
287             X(linhas,1) = gsl_ran_tdist(r,3);
288         }

```

```

289         }
290     else{
291         int contando_influencia = 1;
292         while(contando_influencia!=0){
293             contando_influencia = 0;
294             double alavanca = 4.0/nobs;
295             for(int i = 0; i < nobs; i++) {
296                 X(i,1) = gsl_ran_tdist(r,3);
297             }
298             mat  matriz_chapeu = X*inv(sympd(trans(X)*X))*trans(X);
299             vec diag_chapeu = diagvec(matriz_chapeu);
300             for(int i = 0; i < nobs;i++){
301                 if(diag_chapeu(i) > alavanca){
302                     contando_influencia = 1;
303                 }
304             }
305         }
306     }
307
308     mat X1;
309     X1 = X;
310     int l = 1;
311
312     while(l<samplesize){
313         X = join_cols(X,X1);
314         l++;
315     }
316     nobs = nobs * samplesize;
317 }
318
319 // SEGUNDO ESQUEMA PARA GERACAO DA MATRIZ X.
320
321 // PREDITOR LINEAR.
322 //cout << X << endl;
323 mat eta = X*beta;
324
325 //  $P = (X'X)^{-1}X'$ 
326
327 mat P = inv(sympd(trans(X)*X))*trans(X);
328
329 // TRANSPONDA DA MATRIZ P.
330
331 mat Pt = trans(P);
332
333 // MATRIZ CHAPEU,  $H = X(X'X)^{-1}X'$ .
334
335 mat H = X*P;
336
337 // VETOR DE MEDIDAS DE ALAVANCAGEM.
338
339 vec h = diagvec(H);
340
341 // USADO EM HC0, HC2, HC3, HC4 E HC5.
342 vec weight0 = ones<vec>(nobs), weight2 = 1.0/(1.0-h),
343     weight3 = 1.0/pow((1.0-h), 2.0), weight4(nobs), weight5(nobs);
344
345 double media_h = mean(h);
346 for(int l=0; l<nobs; l++){

```

```

347         if((h(l)/media_h)<4.0){
348             weight4(l) = 1.0/pow((1-h(l)),h(l)/media_h);
349         }else{
350             weight4(l) = 1.0/pow((1-h(l)),4.0);
351         }
352     }
353
354     double max;
355     if(4.0>=(0.7*arma::max(h))/media_h)
356         max = 4.0;
357     else
358         max = 0.7*arma::max(h)/media_h;
359
360     for(int l=0; l<nobs; l++){
361         if((h(l) / media_h)<=max)
362             weight5(l) = 1.0/sqrt(pow((1-h(l)),h(l)/media_h));
363         else
364             weight5(l) = 1.0/sqrt(pow((1-h(l)),max));
365     }
366
367     // ARMAZENA OS PONTOS DE ALTA ALAVANCAGEM.
368     vec contador = zeros<vec>(nobs);
369
370     // CONTANDO O NUMERO DE PONTOS DE ALAVANCA.
371
372     for(int d=0;d<nobs;d++){
373         if(h(d)>4.0/nobs) // CONSIDERADO PONTO DE ALAVANCA OBSERVACOES
374             // MAIORES QUE 2p/n.
375             contador(d) = 1;
376         else
377             contador(d) = 0;
378     }
379
380     // "A" e UM VETOR COM POSSIVEIS CANDIDATOS A SER O VALOR DE "a"
381     // QUE NOS DARA UM LAMBDA PROXIMO DO VALOR DE lambda ESCOLHIDO.
382
383     vec A(4000000);
384     A(0) = 0;
385     for(int s=1;s<4000000;s++){
386         A(s) = A(s-1)+0.00001;
387     }
388
389     double lambda_utilizado, a_utilizado;
390
391     if(lambda==1){
392         a_utilizado=0;
393     }
394
395     if (lambda!=1){
396         int s = 0;
397         mat resultado;
398         while(lambda_utilizado<=lambda-0.00001){
399             resultado = exp(A(s)*X.col(1));
400             lambda_utilizado = resultado.max()/resultado.min();
401             s++;
402         }
403         a_utilizado = as_scalar(A(s));
404     }

```

```

405
406     vec sigma2(nobs), sigma(nobs);
407
408     // VETOR DE VARIANCIAS.
409
410     sigma2 = exp(a_utilizado*X.col(1));
411
412     // VETOR DE DESVIOS PADROES.
413     sigma = sqrt(sigma2);
414
415     // RAZAO ENTRE O MAXIMO E O MINIMO DAS VARIANCIAS.
416     lambda = sigma2.max()/sigma2.min();
417
418     // DADOS PRELIMINARES. INFORMACOES SOBRE O NUMERO DE REPLICAS DE
419     // MONTE CARLO, BOOTSTRAP, BOOTSTRAP DUPLO. TAMBEM E APRESENTADO
420     // INFORMACOES SOBRE O VALOR DE LAMBDA UTILIZADO E O VALOR DE "a"
421     // ESCOLHIDO, ASSIM COMO O NUMERO DE PONTOS DE ALTA ALAVANCAGEM.
422
423     saida << "\n \t \t DADOS DA SIMULACAO" << endl << endl;
424     saida << ">> [*] nobs = " << nobs << endl;
425     saida << ">> [*] lambda = " << lambda << endl;
426     saida << ">> [*] a = " << a_utilizado << endl;
427     saida << ">> [*] nrep = " << nrep << endl;
428     saida << ">> [*] nrep_boot = " << nrep_boot << endl;
429     saida << ">> [*] nrep_boot_duplo = " << nrep_boot_duplo << endl;
430     saida << ">> [*] ncorrecoes = " << ncorrecoes << endl;
431     saida << ">> [*] h_max = " << h.max() << ", 2p/n = " << 4.0/nobs
432         << " e 4p/n = " << 8.0/nobs << endl;
433     saida << ">> [*] Quant. de pontos de alavanca = " <<
434         arma::sum(contador) << endl;
435     if (dist_erro==1)
436         saida << ">> [*] Distribuicao do erro = normal" << endl;
437     if (dist_erro==2)
438         saida << ">> [*] Distribuicao do erro = t(3)" << endl;
439     if (dist_erro==3)
440         saida << ">> [*] Distribuicao do erro = qui-quadrado(2)" << endl;
441     if (dist_erro==4)
442         saida << ">> [*] Distribuicao do erro = weibull(2,3)" << endl;
443     if (dist_erro==5)
444         saida << ">> [*] Distribuicao do erro = gumbel(2.5,2)" << endl;
445     if (dist_erro==6)
446         saida << ">> [*] Distribuicao do erro = gama(3,1.5)" << endl;
447
448     if (dist_t==1)
449         saida << ">> [*] Distribuicao de t^* = rademacher" << endl;
450     if (dist_t==2)
451         saida << ">> [*] Distribuicao de t^* = normal padrao" << endl;
452
453     if(gerador==1)
454         saida << ">> [*] Gerador utilizado = tt800" << endl;
455     if(gerador==2)
456         saida << ">> [*] Gerador utilizado = mt19937" << endl;
457     if(gerador==3)
458         saida << ">> [*] Gerador utilizado = random256_bsd" << endl;
459
460     saida << ">> [*] Semente do gerador = " << semente << endl;
461     saida << ">> [*] Horário de inicio da simulacao: " << asctime(timeinfo);
462

```

```

463 //VARIAVEIS DO BOOTSTRAP.
464
465 vec epsilon_chapeu(nobs), y_estrela(nobs), beta_chapeu_boot;
466 vec beta2_chapeu_boot;
467 vec beta2_chapeu_boot_temp(nrep_boot);
468
469 vec z_estrela0(nrep_boot), z_estrela2(nrep_boot), z_estrela3(nrep_boot),
470 z_estrela4(nrep_boot), z_estrela5(nrep_boot), z0_estrela_duplo(nrep_boot),
471 z2_estrela_duplo(nrep_boot), z3_estrela_duplo(nrep_boot),
472 z4_estrela_duplo(nrep_boot), z5_estrela_duplo(nrep_boot),
473 z_estrela_estrela0(nrep_boot_duplo), z_estrela_estrela2(nrep_boot_duplo),
474 z_estrela_estrela3(nrep_boot_duplo), z_estrela_estrela4(nrep_boot_duplo),
475 z_estrela_estrela5(nrep_boot_duplo);
476
477 // VARIÁVEL UTILIZADA NO BOOTSTRAP PERCENTIL.
478
479 vec betaj_estrela_menos_betaj(nrep_boot);
480
481 // VARIÁVEL UTILIZADA NO BOOTSTRAP PERCENTIL.
482 vec beta2(nrep_boot);
483
484 vec cob95_percentil(nrep), cob99_percentil(nrep),
485 cob90_percentil(nrep), ncobesq95_percentil(nrep),
486 ncobdi95_percentil(nrep), ncobesq99_percentil(nrep),
487 ncobdi99_percentil(nrep), ncobesq90_percentil(nrep),
488 ncobdi90_percentil(nrep), ampl95_percentil(nrep),
489 ampl90_percentil(nrep), ampl99_percentil(nrep);
490
491 vec cob95_percentil_duplo(nrep), cob99_percentil_duplo(nrep),
492 cob90_percentil_duplo(nrep), ncobesq95_percentil_duplo(nrep),
493 ncobdi95_percentil_duplo(nrep), ncobesq99_percentil_duplo(nrep),
494 ncobdi99_percentil_duplo(nrep), ncobesq90_percentil_duplo(nrep),
495 ncobdi90_percentil_duplo(nrep), ampl95_percentil_duplo(nrep),
496 ampl90_percentil_duplo(nrep), ampl99_percentil_duplo(nrep);
497
498 vec cob_0_95_t_percentil(nrep), cob_0_99_t_percentil(nrep),
499 cob_0_90_t_percentil(nrep), ncobesq_0_95_t_percentil(nrep),
500 ncobdi_0_95_t_percentil(nrep), ncobesq_0_99_t_percentil(nrep),
501 ncobdi_0_99_t_percentil(nrep), ncobesq_0_90_t_percentil(nrep),
502 ncobdi_0_90_t_percentil(nrep), ampl_0_95_t_percentil(nrep),
503 ampl_0_90_t_percentil(nrep), ampl_0_99_t_percentil(nrep);
504
505 vec cob_2_95_t_percentil(nrep), cob_2_99_t_percentil(nrep),
506 cob_2_90_t_percentil(nrep), ncobesq_2_95_t_percentil(nrep),
507 ncobdi_2_95_t_percentil(nrep), ncobesq_2_99_t_percentil(nrep),
508 ncobdi_2_99_t_percentil(nrep), ncobesq_2_90_t_percentil(nrep),
509 ncobdi_2_90_t_percentil(nrep), ampl_2_95_t_percentil(nrep),
510 ampl_2_90_t_percentil(nrep), ampl_2_99_t_percentil(nrep);
511
512 vec cob_3_95_t_percentil(nrep), cob_3_99_t_percentil(nrep),
513 cob_3_90_t_percentil(nrep), ncobesq_3_95_t_percentil(nrep),
514 ncobdi_3_95_t_percentil(nrep), ncobesq_3_99_t_percentil(nrep),
515 ncobdi_3_99_t_percentil(nrep), ncobesq_3_90_t_percentil(nrep),
516 ncobdi_3_90_t_percentil(nrep), ampl_3_95_t_percentil(nrep),
517 ampl_3_90_t_percentil(nrep), ampl_3_99_t_percentil(nrep);
518
519 vec cob_4_95_t_percentil(nrep), cob_4_99_t_percentil(nrep),
520 cob_4_90_t_percentil(nrep), ncobesq_4_95_t_percentil(nrep),

```

```

521      ncobdi_4_95_t_percentil(nrep), ncobesq_4_99_t_percentil(nrep),
522      ncobdi_4_99_t_percentil(nrep), ncobesq_4_90_t_percentil(nrep),
523      ncobdi_4_90_t_percentil(nrep), ampl_4_95_t_percentil(nrep),
524      ampl_4_90_t_percentil(nrep), ampl_4_99_t_percentil(nrep);
525
526      vec cob_5_95_t_percentil(nrep), cob_5_99_t_percentil(nrep),
527      cob_5_90_t_percentil(nrep), ncobesq_5_95_t_percentil(nrep),
528      ncobdi_5_95_t_percentil(nrep), ncobesq_5_99_t_percentil(nrep),
529      ncobdi_5_99_t_percentil(nrep), ncobesq_5_90_t_percentil(nrep),
530      ncobdi_5_90_t_percentil(nrep), ampl_5_95_t_percentil(nrep),
531      ampl_5_90_t_percentil(nrep), ampl_5_99_t_percentil(nrep);
532
533      //double li95, li90, ls90, ls95, li99, ls99;
534
535      double li_0_90, li_2_90, li_3_90, li_4_90, li_5_90,
536      ls_0_90, ls_2_90, ls_3_90, ls_4_90, ls_5_90,
537      li_0_95, li_2_95, li_3_95, li_4_95, li_5_95,
538      ls_0_95, ls_2_95, ls_3_95, ls_4_95, ls_5_95,
539      li_0_99, li_2_99, li_3_99, li_4_99, li_5_99,
540      ls_0_99, ls_2_99, ls_3_99, ls_4_99, ls_5_99;
541
542      // VARIÁVEIS DO BOOTSTRAP DUPLO.
543
544      vec epsilon_chapeu_boot_duplo, beta_chapeu_boot_duplo,
545      beta2_chapeu_boot_duplo, y_estrela_estrela, t_estrela_estrela;
546
547      vec cob_0_95_t_percentil_duplo(nrep), cob_0_99_t_percentil_duplo(nrep),
548      cob_0_90_t_percentil_duplo(nrep), ncobesq_0_95_t_percentil_duplo(nrep),
549      ncobdi_0_95_t_percentil_duplo(nrep), ncobesq_0_99_t_percentil_duplo(nrep),
550      ncobdi_0_99_t_percentil_duplo(nrep), ncobesq_0_90_t_percentil_duplo(nrep),
551      ncobdi_0_90_t_percentil_duplo(nrep), ampl_0_95_t_percentil_duplo(nrep),
552      ampl_0_90_t_percentil_duplo(nrep), ampl_0_99_t_percentil_duplo(nrep);
553
554      vec cob_2_95_t_percentil_duplo(nrep), cob_2_99_t_percentil_duplo(nrep),
555      cob_2_90_t_percentil_duplo(nrep), ncobesq_2_95_t_percentil_duplo(nrep),
556      ncobdi_2_95_t_percentil_duplo(nrep), ncobesq_2_99_t_percentil_duplo(nrep),
557      ncobdi_2_99_t_percentil_duplo(nrep), ncobesq_2_90_t_percentil_duplo(nrep),
558      ncobdi_2_90_t_percentil_duplo(nrep), ampl_2_95_t_percentil_duplo(nrep),
559      ampl_2_90_t_percentil_duplo(nrep), ampl_2_99_t_percentil_duplo(nrep);
560
561      vec cob_3_95_t_percentil_duplo(nrep), cob_3_99_t_percentil_duplo(nrep),
562      cob_3_90_t_percentil_duplo(nrep), ncobesq_3_95_t_percentil_duplo(nrep),
563      ncobdi_3_95_t_percentil_duplo(nrep), ncobesq_3_99_t_percentil_duplo(nrep),
564      ncobdi_3_99_t_percentil_duplo(nrep), ncobesq_3_90_t_percentil_duplo(nrep),
565      ncobdi_3_90_t_percentil_duplo(nrep), ampl_3_95_t_percentil_duplo(nrep),
566      ampl_3_90_t_percentil_duplo(nrep), ampl_3_99_t_percentil_duplo(nrep);
567
568      vec cob_4_95_t_percentil_duplo(nrep), cob_4_99_t_percentil_duplo(nrep),
569      cob_4_90_t_percentil_duplo(nrep), ncobesq_4_95_t_percentil_duplo(nrep),
570      ncobdi_4_95_t_percentil_duplo(nrep), ncobesq_4_99_t_percentil_duplo(nrep),
571      ncobdi_4_99_t_percentil_duplo(nrep), ncobesq_4_90_t_percentil_duplo(nrep),
572      ncobdi_4_90_t_percentil_duplo(nrep), ampl_4_95_t_percentil_duplo(nrep),
573      ampl_4_90_t_percentil_duplo(nrep), ampl_4_99_t_percentil_duplo(nrep);
574
575      vec cob_5_95_t_percentil_duplo(nrep), cob_5_99_t_percentil_duplo(nrep),
576      cob_5_90_t_percentil_duplo(nrep), ncobesq_5_95_t_percentil_duplo(nrep),
577      ncobdi_5_95_t_percentil_duplo(nrep), ncobesq_5_99_t_percentil_duplo(nrep),
578      ncobdi_5_99_t_percentil_duplo(nrep), ncobesq_5_90_t_percentil_duplo(nrep),

```

```

579     ncobdi_5_90_t_percentil_duplo(nrep), ampl_5_95_t_percentil_duplo(nrep),
580     ampl_5_90_t_percentil_duplo(nrep), ampl_5_99_t_percentil_duplo(nrep);
581
582     vec cob_0_95_t_percentil_duplo1(nrep), cob_0_99_t_percentil_duplo1(nrep),
583     cob_0_90_t_percentil_duplo1(nrep), ncobesq_0_95_t_percentil_duplo1(nrep),
584     ncobdi_0_95_t_percentil_duplo1(nrep), ncobesq_0_99_t_percentil_duplo1(nrep),
585     ncobdi_0_99_t_percentil_duplo1(nrep), ncobesq_0_90_t_percentil_duplo1(nrep),
586     ncobdi_0_90_t_percentil_duplo1(nrep), ampl_0_95_t_percentil_duplo1(nrep),
587     ampl_0_90_t_percentil_duplo1(nrep), ampl_0_99_t_percentil_duplo1(nrep);
588
589     vec cob_2_95_t_percentil_duplo1(nrep), cob_2_99_t_percentil_duplo1(nrep),
590     cob_2_90_t_percentil_duplo1(nrep), ncobesq_2_95_t_percentil_duplo1(nrep),
591     ncobdi_2_95_t_percentil_duplo1(nrep), ncobesq_2_99_t_percentil_duplo1(nrep),
592     ncobdi_2_99_t_percentil_duplo1(nrep), ncobesq_2_90_t_percentil_duplo1(nrep),
593     ncobdi_2_90_t_percentil_duplo1(nrep), ampl_2_95_t_percentil_duplo1(nrep),
594     ampl_2_90_t_percentil_duplo1(nrep), ampl_2_99_t_percentil_duplo1(nrep);
595
596     vec cob_3_95_t_percentil_duplo1(nrep), cob_3_99_t_percentil_duplo1(nrep),
597     cob_3_90_t_percentil_duplo1(nrep), ncobesq_3_95_t_percentil_duplo1(nrep),
598     ncobdi_3_95_t_percentil_duplo1(nrep), ncobesq_3_99_t_percentil_duplo1(nrep),
599     ncobdi_3_99_t_percentil_duplo1(nrep), ncobesq_3_90_t_percentil_duplo1(nrep),
600     ncobdi_3_90_t_percentil_duplo1(nrep), ampl_3_95_t_percentil_duplo1(nrep),
601     ampl_3_90_t_percentil_duplo1(nrep), ampl_3_99_t_percentil_duplo1(nrep);
602
603     vec cob_4_95_t_percentil_duplo1(nrep), cob_4_99_t_percentil_duplo1(nrep),
604     cob_4_90_t_percentil_duplo1(nrep), ncobesq_4_95_t_percentil_duplo1(nrep),
605     ncobdi_4_95_t_percentil_duplo1(nrep), ncobesq_4_99_t_percentil_duplo1(nrep),
606     ncobdi_4_99_t_percentil_duplo1(nrep), ncobesq_4_90_t_percentil_duplo1(nrep),
607     ncobdi_4_90_t_percentil_duplo1(nrep), ampl_4_95_t_percentil_duplo1(nrep),
608     ampl_4_90_t_percentil_duplo1(nrep), ampl_4_99_t_percentil_duplo1(nrep);
609
610     vec cob_5_95_t_percentil_duplo1(nrep), cob_5_99_t_percentil_duplo1(nrep),
611     cob_5_90_t_percentil_duplo1(nrep), ncobesq_5_95_t_percentil_duplo1(nrep),
612     ncobdi_5_95_t_percentil_duplo1(nrep), ncobesq_5_99_t_percentil_duplo1(nrep),
613     ncobdi_5_99_t_percentil_duplo1(nrep), ncobesq_5_90_t_percentil_duplo1(nrep),
614     ncobdi_5_90_t_percentil_duplo1(nrep), ampl_5_95_t_percentil_duplo1(nrep),
615     ampl_5_90_t_percentil_duplo1(nrep), ampl_5_99_t_percentil_duplo1(nrep);
616
617     vec Y = ones<vec>(nobs);
618
619     // A MATRIZ C ABAIXO FORNECE OS ELEMENTOS c_jj QUE SERAO UTILIZADOS
620     // PARA AVALIAR O INTERVALO DE CONFIANCA OLS NO CASO DE HOMOSCEDASTICIDADE.
621
622     mat C = inv(sympd(trans(X)*X));
623
624     // ESSE PRODUTO SER FEITO FORA DE MONTE CARLO e IMPORTANTE
625     // PARA QUE EM CADA RePLICA DE MONTE CARLO NaO SEJA FEITO
626     // INVERSAS E PRODUTOS MATRICIAIS SEM NECESSIDADE.
627
628     mat produtos = C*trans(X);
629
630     // VALOR CRITICO
631     // QUANTIS 1-\alpha/2 PARA DAS DISTRIBUICOES NORMAL PADRAO E T-STUDENT(n-p)
632     // GRAUS DE LIBERDADE.
633
634     double vc_z1 = gsl_cdf_ugaussian_Pinv(1-0.01/2); // CONFIANCA DE 99%
635     double vc_z5 = gsl_cdf_ugaussian_Pinv(1-0.05/2); // CONFIANCA DE 95%
636     double vc_z10 = gsl_cdf_ugaussian_Pinv(1-0.10/2); // CONFIANCA DE 90%

```



```

637     double vc_t1 = gsl_cdf_tdist_Pinv(1-0.01/2,nobs-2); // CONFIANCA DE 99%
638     double vc_t5 = gsl_cdf_tdist_Pinv(1-0.05/2,nobs-2); // CONFIANCA DE 95%
639     double vc_t10 = gsl_cdf_tdist_Pinv(1-0.10/2,nobs-2); // CONFIANCA DE 90%
640
641     vec cob90_t_ols(nrep), cob95_t_ols(nrep), cob99_t_ols(nrep),
642         cob90_t_hc0(nrep), cob95_t_hc0(nrep), cob99_t_hc0(nrep),
643         cob90_t_hc2(nrep), cob95_t_hc2(nrep), cob99_t_hc2(nrep),
644         cob90_t_hc3(nrep), cob95_t_hc3(nrep), cob99_t_hc3(nrep),
645         cob90_t_hc4(nrep), cob95_t_hc4(nrep), cob99_t_hc4(nrep),
646         cob90_t_hc5(nrep), cob95_t_hc5(nrep), cob99_t_hc5(nrep);
647
648     vec ncobesq90_t_ols(nrep), ncobesq95_t_ols(nrep), ncobesq99_t_ols(nrep),
649         ncobesq90_t_hc0(nrep), ncobesq95_t_hc0(nrep), ncobesq99_t_hc0(nrep),
650         ncobesq90_t_hc2(nrep), ncobesq95_t_hc2(nrep), ncobesq99_t_hc2(nrep),
651         ncobesq90_t_hc3(nrep), ncobesq95_t_hc3(nrep), ncobesq99_t_hc3(nrep),
652         ncobesq90_t_hc4(nrep), ncobesq95_t_hc4(nrep), ncobesq99_t_hc4(nrep),
653         ncobesq90_t_hc5(nrep), ncobesq95_t_hc5(nrep), ncobesq99_t_hc5(nrep);
654
655     vec ncobdi90_t_ols(nrep), ncobdi95_t_ols(nrep), ncobdi99_t_ols(nrep),
656         ncobdi90_t_hc0(nrep), ncobdi95_t_hc0(nrep), ncobdi99_t_hc0(nrep),
657         ncobdi90_t_hc2(nrep), ncobdi95_t_hc2(nrep), ncobdi99_t_hc2(nrep),
658         ncobdi90_t_hc3(nrep), ncobdi95_t_hc3(nrep), ncobdi99_t_hc3(nrep),
659         ncobdi90_t_hc4(nrep), ncobdi95_t_hc4(nrep), ncobdi99_t_hc4(nrep),
660         ncobdi90_t_hc5(nrep), ncobdi95_t_hc5(nrep), ncobdi99_t_hc5(nrep);
661
662     vec cob90_z_ols(nrep), cob95_z_ols(nrep), cob99_z_ols(nrep),
663         cob90_z_hc0(nrep), cob95_z_hc0(nrep), cob99_z_hc0(nrep),
664         cob90_z_hc2(nrep), cob95_z_hc2(nrep), cob99_z_hc2(nrep),
665         cob90_z_hc3(nrep), cob95_z_hc3(nrep), cob99_z_hc3(nrep),
666         cob90_z_hc4(nrep), cob95_z_hc4(nrep), cob99_z_hc4(nrep),
667         cob90_z_hc5(nrep), cob95_z_hc5(nrep), cob99_z_hc5(nrep);
668
669     vec ncobesq90_z_ols(nrep), ncobesq95_z_ols(nrep), ncobesq99_z_ols(nrep),
670         ncobesq90_z_hc0(nrep), ncobesq95_z_hc0(nrep), ncobesq99_z_hc0(nrep),
671         ncobesq90_z_hc2(nrep), ncobesq95_z_hc2(nrep), ncobesq99_z_hc2(nrep),
672         ncobesq90_z_hc3(nrep), ncobesq95_z_hc3(nrep), ncobesq99_z_hc3(nrep),
673         ncobesq90_z_hc4(nrep), ncobesq95_z_hc4(nrep), ncobesq99_z_hc4(nrep),
674         ncobesq90_z_hc5(nrep), ncobesq95_z_hc5(nrep), ncobesq99_z_hc5(nrep);
675
676     vec ncobdi90_z_ols(nrep), ncobdi95_z_ols(nrep), ncobdi99_z_ols(nrep),
677         ncobdi90_z_hc0(nrep), ncobdi95_z_hc0(nrep), ncobdi99_z_hc0(nrep),
678         ncobdi90_z_hc2(nrep), ncobdi95_z_hc2(nrep), ncobdi99_z_hc2(nrep),
679         ncobdi90_z_hc3(nrep), ncobdi95_z_hc3(nrep), ncobdi99_z_hc3(nrep),
680         ncobdi90_z_hc4(nrep), ncobdi95_z_hc4(nrep), ncobdi99_z_hc4(nrep),
681         ncobdi90_z_hc5(nrep), ncobdi95_z_hc5(nrep), ncobdi99_z_hc5(nrep);
682
683     vec ampl90_t_ols(nrep), ampl95_t_ols(nrep), ampl99_t_ols(nrep),
684         ampl90_t_hc0(nrep), ampl95_t_hc0(nrep), ampl99_t_hc0(nrep),
685         ampl90_t_hc2(nrep), ampl95_t_hc2(nrep), ampl99_t_hc2(nrep),
686         ampl90_t_hc3(nrep), ampl95_t_hc3(nrep), ampl99_t_hc3(nrep),
687         ampl90_t_hc4(nrep), ampl95_t_hc4(nrep), ampl99_t_hc4(nrep),
688         ampl90_t_hc5(nrep), ampl95_t_hc5(nrep), ampl99_t_hc5(nrep);
689
690     vec ampl90_z_ols(nrep), ampl95_z_ols(nrep), ampl99_z_ols(nrep),
691         ampl90_z_hc0(nrep), ampl95_z_hc0(nrep), ampl99_z_hc0(nrep),
692         ampl90_z_hc2(nrep), ampl95_z_hc2(nrep), ampl99_z_hc2(nrep),
693         ampl90_z_hc3(nrep), ampl95_z_hc3(nrep), ampl99_z_hc3(nrep),
694         ampl90_z_hc4(nrep), ampl95_z_hc4(nrep), ampl99_z_hc4(nrep),

```

```

695         ampl90_z_hc5(nrep), ampl95_z_hc5(nrep), ampl99_z_hc5(nrep);
696
697 // UTILIZADO NA GERACAO DO VALOR DE t^*.
698 double numero;
699
700 // AQUI COMECA O LACO DE MONTE CARLO.
701 // #pragma omp parallel for
702 for(int i=0;i<nrep;i++){
703     if(dist_erro==1){
704         for(int v=0;v<nobs;v++){
705             Y(v) = eta(v)+sigma(v)*gsl_ran_gaussian(r,1.0);
706         }
707     }
708     if(dist_erro==2){
709         for(int v=0;v<nobs;v++){
710             Y(v) = eta(v)+sigma(v)*(gsl_ran_tdist(r,3)/sqrt(1.5));
711         }
712     }
713
714     if(dist_erro==3){
715         for(int v=0;v<nobs;v++){
716             Y(v) = eta(v)+sigma(v)*(gsl_ran_chisq(r,2)-2.0)/2.0;
717         }
718     }
719
720     if(dist_erro==4){
721         for(int v=0;v<nobs;v++){
722             Y(v) = eta(v)+sigma(v)*(gsl_ran_weibull(r,2,3)
723                                     -1.785959)/0.6491006;
724         }
725     }
726
727     if(dist_erro==5){
728         for(int v=0;v<nobs;v++){
729             Y(v) = eta(v)+sigma(v)*(gsl_ran_gumbel2(r,2.5,2)
730                                     -1.965001)/2.032706;
731         }
732     }
733
734 // DISTRIBUICAO GAMMA PARA OS ERROS USANDO O ALGORITMO DE KNUTH.
735 if(dist_erro==6){
736     for(int v=0;v<nobs;v++){
737         Y(v) = eta(v)+sigma(v)*(gsl_ran_gamma_knuth(r,2.5,2)
738                                 -1.550078)/1.052454;
739     }
740 }
741
742 mat temp = produtos*Y;
743 mat resid2 = pow((Y-X*temp),2.0); // EPSILON AO QUADRADO.
744
745 // MATRIZ OMEGA ESTIMADO. E UMA MATRIZ DIAGONAL N POR N.
746 mat omega0 = diagmat(resid2%weight0);
747 // MATRIZ OMEGA ESTIMADO. E UMA MATRIZ DIAGONAL N POR N.
748 mat omega2 = diagmat(resid2%weight2);
749 // MATRIZ OMEGA ESTIMADO. E UMA MATRIZ DIAGONAL N POR N.
750 mat omega3 = diagmat(resid2%weight3);
751 // MATRIZ OMEGA ESTIMADO. E UMA MATRIZ DIAGONAL N POR N.
752 mat omega4 = diagmat(resid2%weight4);

```

```

753      // MATRIZ OMEGA ESTIMADO. E UMA MATRIZ DIAGONAL N POR N.
754      mat omega5 = diagmat(resid2%weight5);
755
756      mat HC0 = P*omega0*Pt;
757      mat HC2 = P*omega2*Pt;
758      mat HC3 = P*omega3*Pt;
759      mat HC4 = P*omega4*Pt;
760      mat HC5 = P*omega5*Pt;
761
762      vec diagonal_hc4 = diagvec(HC4);
763      double variancia_maxima = diagonal_hc4.max();
764      double variancia_minima = diagonal_hc4.min();
765
766      double OLS = as_scalar(sum(resid2))/(nobs-2) * C(1,1);
767
768      //epsilon_chapeu = Y-X*temp; // ESTIMATIVAS DOS ERROS.
769      epsilon_chapeu = Y-X*temp; // ESTIMATIVAS DOS ERROS.
770
771      vec hc0_b(nrep_boot), hc2_b(nrep_boot), hc3_b(nrep_boot), hc4_b(nrep_boot),
772      hc5_b(nrep_boot), u_estrela(nrep_boot), Z0_j(nrep_boot), Z2_j(nrep_boot),
773      Z3_j(nrep_boot), Z4_j(nrep_boot), Z5_j(nrep_boot);
774
775      vec hc0_duplo(nrep_boot_duplo), hc2_duplo(nrep_boot_duplo),
776      hc3_duplo(nrep_boot_duplo), hc4_duplo(nrep_boot_duplo),
777      hc5_duplo(nrep_boot_duplo);
778
779      // CALCULO DAS QUANTIDADES PIVOTAIS CALCULADAS ( $\hat{\beta}_j - 1$ )/ $\sqrt{HC_k(1,1)}$ .
780      // ESSES PIVOS SERAO UTILIZADOS PARA VARIFICAR O AJUSTAMENTO DESSES VALORES A UMA
781      // DISTRIBUICAO t(n-p) OU A UMA DISTRIBUICAO NORMAL PADRAO. COM ESSES VALORES PODERA
782      // SER CALCULADO AS ESTATISTICAS DE CRAMER-VON MISSES E ANDERSON DARLING BEM COMO
783      // CONSTRUIR QQ-PLOTS OU PP-PLOTS.
784
785      double pivools, pivohc0, pivohc2, pivohc3, pivohc4, pivohc5;
786
787      pivools = (temp(1) - 1)/sqrt(OLS);
788      pivohc0 = (temp(1) - 1)/sqrt(HC0(1,1));
789      pivohc2 = (temp(1) - 1)/sqrt(HC2(1,1));
790      pivohc3 = (temp(1) - 1)/sqrt(HC3(1,1));
791      pivohc4 = (temp(1) - 1)/sqrt(HC4(1,1));
792      pivohc5 = (temp(1) - 1)/sqrt(HC5(1,1));
793
794      pivo_ols << pivools << endl;
795      pivo_hc0 << pivohc0 << endl;
796      pivo_hc2 << pivohc2 << endl;
797      pivo_hc3 << pivohc3 << endl;
798      pivo_hc4 << pivohc4 << endl;
799      pivo_hc5 << pivohc5 << endl;
800
801      // INTERVALOS PARA QUANTIL DE UMA DISTRIBUICAO T-STUDENT COM n-p
802      // GRAUS DE LIBERDADE. AVALIACAO DOS INTERVALOS SEM UTILIZAR BOOTSTRAP.
803      // CONFIANCA DE 90%
804
805      double li = temp(1) - vc_t10*sqrt(OLS);
806      double ls = temp(1) + vc_t10*sqrt(OLS);
807
808      if(li<=beta(1) && beta(1)<=ls)
809          cob90_t_ols(i) = 1;
810      else

```

```

811             cob90_t_ols(i) = 0;
812
813         if(beta(1)<li)
814             ncobesq90_t_ols(i) = 1;
815         else
816             ncobesq90_t_ols(i) = 0;
817
818         if(beta(1)>ls)
819             ncobdi90_t_ols(i) = 1;
820         else
821             ncobdi90_t_ols(i) = 0;
822         ampl90_t_ols(i) = ls - li;
823
824         li = temp(1) - vc_t10*sqrt(HC0(1,1));
825         ls = temp(1) + vc_t10*sqrt(HC0(1,1));
826
827         if(li<=beta(1) && beta(1)<=ls)
828             cob90_t_hc0(i) = 1;
829         else
830             cob90_t_hc0(i) = 0;
831
832         if(beta(1)<li)
833             ncobesq90_t_hc0(i) = 1;
834         else
835             ncobesq90_t_hc0(i) = 0;
836
837         if(beta(1)>ls)
838             ncobdi90_t_hc0(i) = 1;
839         else
840             ncobdi90_t_hc0(i) = 0;
841         ampl90_t_hc0(i) = ls - li;
842
843         li = temp(1) - vc_t10*sqrt(HC2(1,1));
844         ls = temp(1) + vc_t10*sqrt(HC2(1,1));
845
846         if(li<=beta(1) && beta(1)<=ls)
847             cob90_t_hc2(i) = 1;
848         else
849             cob90_t_hc2(i) = 0;
850
851         if(beta(1)<li)
852             ncobesq90_t_hc2(i) = 1;
853         else
854             ncobesq90_t_hc2(i) = 0;
855
856         if(beta(1)>ls)
857             ncobdi90_t_hc2(i) = 1;
858         else
859             ncobdi90_t_hc2(i) = 0;
860         ampl90_t_hc2(i) = ls - li;
861
862         li = temp(1) - vc_t10*sqrt(HC3(1,1));
863         ls = temp(1) + vc_t10*sqrt(HC3(1,1));
864
865         if(li<=beta(1) && beta(1)<=ls)
866             cob90_t_hc3(i) = 1;
867         else
868             cob90_t_hc3(i) = 0;

```

```

869
870         if(beta(1)<li)
871             ncobesq90_t_hc3(i) = 1;
872         else
873             ncobesq90_t_hc3(i) = 0;
874
875         if(beta(1)>ls)
876             ncobdi90_t_hc3(i) = 1;
877         else
878             ncobdi90_t_hc3(i) = 0;
879         ampl90_t_hc3(i) = ls - li;
880
881         li = temp(1) - vc_t10*sqrt(HC4(1,1));
882         ls = temp(1) + vc_t10*sqrt(HC4(1,1));
883
884         if(li<=beta(1) && beta(1)<=ls)
885             cob90_t_hc4(i) = 1;
886         else
887             cob90_t_hc4(i) = 0;
888
889         if(beta(1)<li)
890             ncobesq90_t_hc4(i) = 1;
891         else
892             ncobesq90_t_hc4(i) = 0;
893
894         if(beta(1)>ls)
895             ncobdi90_t_hc4(i) = 1;
896         else
897             ncobdi90_t_hc4(i) = 0;
898         ampl90_t_hc4(i) = ls - li;
899
900         li = temp(1) - vc_t10*sqrt(HC5(1,1));
901         ls = temp(1) + vc_t10*sqrt(HC5(1,1));
902
903         if(li<=beta(1) && beta(1)<=ls)
904             cob90_t_hc5(i) = 1;
905         else
906             cob90_t_hc5(i) = 0;
907
908         if(beta(1)<li)
909             ncobesq90_t_hc5(i) = 1;
910         else
911             ncobesq90_t_hc5(i) = 0;
912
913         if(beta(1)>ls)
914             ncobdi90_t_hc5(i) = 1;
915         else
916             ncobdi90_t_hc5(i) = 0;
917         ampl90_t_hc5(i) = ls - li;
918
919         // CONFIANCA DE 95%
920
921         li = temp(1) - vc_t5*sqrt(OLS);
922         ls = temp(1) + vc_t5*sqrt(OLS);
923
924         lils_ols << li << endl;
925         lils_ols << ls << endl;
926

```

```

927         if(li<=beta(1) && beta(1)<=ls)
928             cob95_t_ols(i) = 1;
929         else
930             cob95_t_ols(i) = 0;
931
932         if(beta(1)<li)
933             ncobesq95_t_ols(i) = 1;
934         else
935             ncobesq95_t_ols(i) = 0;
936
937         if(beta(1)>ls)
938             ncobdi95_t_ols(i) = 1;
939         else
940             ncobdi95_t_ols(i) = 0;
941         ampl95_t_ols(i) = ls - li;
942
943         li = temp(1) - vc_t5*sqrt(HC0(1,1));
944         ls = temp(1) + vc_t5*sqrt(HC0(1,1));
945
946         lils_hc0 << li << endl;
947         lils_hc0 << ls << endl;
948
949         if(li<=beta(1) && beta(1)<=ls)
950             cob95_t_hc0(i) = 1;
951         else
952             cob95_t_hc0(i) = 0;
953
954         if(beta(1)<li)
955             ncobesq95_t_hc0(i) = 1;
956         else
957             ncobesq95_t_hc0(i) = 0;
958
959         if(beta(1)>ls)
960             ncobdi95_t_hc0(i) = 1;
961         else
962             ncobdi95_t_hc0(i) = 0;
963         ampl95_t_hc0(i) = ls - li;
964
965         li = temp(1) - vc_t5*sqrt(HC2(1,1));
966         ls = temp(1) + vc_t5*sqrt(HC2(1,1));
967
968         lils_hc2 << li << endl;
969         lils_hc2 << ls << endl;
970
971         if(li<=beta(1) && beta(1)<=ls)
972             cob95_t_hc2(i) = 1;
973         else
974             cob95_t_hc2(i) = 0;
975
976         if(beta(1)<li)
977             ncobesq95_t_hc2(i) = 1;
978         else
979             ncobesq95_t_hc2(i) = 0;
980
981         if(beta(1)>ls)
982             ncobdi95_t_hc2(i) = 1;
983         else
984             ncobdi95_t_hc2(i) = 0;

```

```

985         ampl95_t_hc2(i) = ls - li;
986
987         li = temp(1) - vc_t5*sqrt(HC3(1,1));
988         ls = temp(1) + vc_t5*sqrt(HC3(1,1));
989
990         lils_hc3 << li << endl;
991         lils_hc3 << ls << endl;
992
993         if(li<=beta(1) && beta(1)<=ls)
994             cob95_t_hc3(i) = 1;
995         else
996             cob95_t_hc3(i) = 0;
997
998         if(beta(1)<li)
999             ncobesq95_t_hc3(i) = 1;
1000        else
1001            ncobesq95_t_hc3(i) = 0;
1002
1003        if(beta(1)>ls)
1004            ncobdi95_t_hc3(i) = 1;
1005        else
1006            ncobdi95_t_hc3(i) = 0;
1007        ampl95_t_hc3(i) = ls - li;
1008
1009        li = temp(1) - vc_t5*sqrt(HC4(1,1));
1010        ls = temp(1) + vc_t5*sqrt(HC4(1,1));
1011
1012        lils_hc4 << li << endl;
1013        lils_hc4 << ls << endl;
1014
1015        if(li<=beta(1) && beta(1)<=ls)
1016            cob95_t_hc4(i) = 1;
1017        else
1018            cob95_t_hc4(i) = 0;
1019
1020        if(beta(1)<li)
1021            ncobesq95_t_hc4(i) = 1;
1022        else
1023            ncobesq95_t_hc4(i) = 0;
1024
1025        if(beta(1)>ls)
1026            ncobdi95_t_hc4(i) = 1;
1027        else
1028            ncobdi95_t_hc4(i) = 0;
1029        ampl95_t_hc4(i) = ls - li;
1030
1031        li = temp(1) - vc_t5*sqrt(HC5(1,1));
1032        ls = temp(1) + vc_t5*sqrt(HC5(1,1));
1033
1034        lils_hc5 << li << endl;
1035        lils_hc5 << ls << endl;
1036
1037        if(li<=beta(1) && beta(1)<=ls)
1038            cob95_t_hc5(i) = 1;
1039        else
1040            cob95_t_hc5(i) = 0;
1041
1042        if(beta(1)<li)

```

```
1043         ncobesq95_t_hc5(i) = 1;
1044     else
1045         ncobesq95_t_hc5(i) = 0;
1046
1047     if(beta(1)>ls)
1048         ncobdi95_t_hc5(i) = 1;
1049     else
1050         ncobdi95_t_hc5(i) = 0;
1051     ampl95_t_hc5(i) = ls -li;
1052
1053     // CONFIANCA DE 99%
1054
1055     li = temp(1) - vc_t1*sqrt(OLS);
1056     ls = temp(1) + vc_t1*sqrt(OLS);
1057
1058     if(li<=beta(1) && beta(1)<=ls)
1059         cob99_t_ols(i) = 1;
1060     else
1061         cob99_t_ols(i) = 0;
1062
1063     if(beta(1)<li)
1064         ncobesq99_t_ols(i) = 1;
1065     else
1066         ncobesq99_t_ols(i) = 0;
1067
1068     if(beta(1)>ls)
1069         ncobdi99_t_ols(i) = 1;
1070     else
1071         ncobdi99_t_ols(i) = 0;
1072     ampl99_t_ols(i) = ls - li;
1073
1074     li = temp(1) - vc_t1*sqrt(HC0(1,1));
1075     ls = temp(1) + vc_t1*sqrt(HC0(1,1));
1076
1077     if(li<=beta(1) && beta(1)<=ls)
1078         cob99_t_hc0(i) = 1;
1079     else
1080         cob99_t_hc0(i) = 0;
1081
1082     if(beta(1)<li)
1083         ncobesq99_t_hc0(i) = 1;
1084     else
1085         ncobesq99_t_hc0(i) = 0;
1086
1087     if(beta(1)>ls)
1088         ncobdi99_t_hc0(i) = 1;
1089     else
1090         ncobdi99_t_hc0(i) = 0;
1091     ampl99_t_hc0(i) = ls - li;
1092
1093     li = temp(1) - vc_t1*sqrt(HC2(1,1));
1094     ls = temp(1) + vc_t1*sqrt(HC2(1,1));
1095
1096     if(li<=beta(1) && beta(1)<=ls)
1097         cob99_t_hc2(i) = 1;
1098     else
1099         cob99_t_hc2(i) = 0;
1100
```



```

1101         if(beta(1)<li)
1102             ncobesq99_t_hc2(i) = 1;
1103         else
1104             ncobesq99_t_hc2(i) = 0;
1105
1106         if(beta(1)>ls)
1107             ncobdi99_t_hc2(i) = 1;
1108         else
1109             ncobdi99_t_hc2(i) = 0;
1110         ampl99_t_hc2(i) = ls - li;
1111
1112         li = temp(1) - vc_t1*sqrt(HC3(1,1));
1113         ls = temp(1) + vc_t1*sqrt(HC3(1,1));
1114
1115         if(li<=beta(1) && beta(1)<=ls)
1116             cob99_t_hc3(i) = 1;
1117         else
1118             cob99_t_hc3(i) = 0;
1119
1120         if(beta(1)<li)
1121             ncobesq99_t_hc3(i) = 1;
1122         else
1123             ncobesq99_t_hc3(i) = 0;
1124
1125         if(beta(1)>ls)
1126             ncobdi99_t_hc3(i) = 1;
1127         else
1128             ncobdi99_t_hc3(i) = 0;
1129         ampl99_t_hc3(i) = ls - li;
1130
1131         li = temp(1) - vc_t1*sqrt(HC4(1,1));
1132         ls = temp(1) + vc_t1*sqrt(HC4(1,1));
1133
1134         if(li<=beta(1) && beta(1)<=ls)
1135             cob99_t_hc4(i) = 1;
1136         else
1137             cob99_t_hc4(i) = 0;
1138
1139         if(beta(1)<li)
1140             ncobesq99_t_hc4(i) = 1;
1141         else
1142             ncobesq99_t_hc4(i) = 0;
1143
1144         if(beta(1)>ls)
1145             ncobdi99_t_hc4(i) = 1;
1146         else
1147             ncobdi99_t_hc4(i) = 0;
1148         ampl99_t_hc4(i) = ls - li;
1149
1150         li = temp(1) - vc_t1*sqrt(HC5(1,1));
1151         ls = temp(1) + vc_t1*sqrt(HC5(1,1));
1152
1153         if(li<=beta(1) && beta(1)<=ls)
1154             cob99_t_hc5(i) = 1;
1155         else
1156             cob99_t_hc5(i) = 0;
1157
1158         if(beta(1)<li)

```

```

1159             ncobesq99_t_hc5(i) = 1;
1160         else
1161             ncobesq99_t_hc5(i) = 0;
1162
1163         if(beta(1)>ls)
1164             ncobdi99_t_hc5(i) = 1;
1165         else
1166             ncobdi99_t_hc5(i) = 0;
1167         ampl99_t_hc5(i) = ls - li;
1168
1169         // INTERVALOS PARA QUANTIL DE UMA DISTRIBUICAO NORMAL PADRAO.
1170         // AVALIACAO DOS INTERVALOS SEM UTILIZAR BOOTSTRAP.
1171         // CONFIANCA DE 90%
1172
1173         li = temp(1) - vc_z10*sqrt(OLS);
1174         ls = temp(1) + vc_z10*sqrt(OLS);
1175
1176         if(li<=beta(1) && beta(1)<=ls)
1177             cob90_z_ols(i) = 1;
1178         else
1179             cob90_z_ols(i) = 0;
1180
1181         if(beta(1)<li)
1182             ncobesq90_z_ols(i) = 1;
1183         else
1184             ncobesq90_z_ols(i) = 0;
1185
1186         if(beta(1)>ls)
1187             ncobdi90_z_ols(i) = 1;
1188         else
1189             ncobdi90_z_ols(i) = 0;
1190         ampl90_z_ols(i) = ls - li;
1191
1192         li = temp(1) - vc_z10*sqrt(HC0(1,1));
1193         ls = temp(1) + vc_z10*sqrt(HC0(1,1));
1194
1195         if(li<=beta(1) && beta(1)<=ls)
1196             cob90_z_hc0(i) = 1;
1197         else
1198             cob90_z_hc0(i) = 0;
1199
1200         if(beta(1)<li)
1201             ncobesq90_z_hc0(i) = 1;
1202         else
1203             ncobesq90_z_hc0(i) = 0;
1204
1205         if(beta(1)>ls)
1206             ncobdi90_z_hc0(i) = 1;
1207         else
1208             ncobdi90_z_hc0(i) = 0;
1209         ampl90_z_hc0(i) = ls - li;
1210
1211         li = temp(1) - vc_z10*sqrt(HC2(1,1));
1212         ls = temp(1) + vc_z10*sqrt(HC2(1,1));
1213
1214         if(li<=beta(1) && beta(1)<=ls)
1215             cob90_z_hc2(i) = 1;
1216         else

```

```

1217         cob90_z_hc2(i) = 0;
1218
1219         if(beta(1)<li)
1220             ncobesq90_z_hc2(i) = 1;
1221         else
1222             ncobesq90_z_hc2(i) = 0;
1223
1224         if(beta(1)>ls)
1225             ncobdi90_z_hc2(i) = 1;
1226         else
1227             ncobdi90_z_hc2(i) = 0;
1228         ampl90_z_hc2(i) = ls - li;
1229
1230         li = temp(1) - vc_z10*sqrt(HC3(1,1));
1231         ls = temp(1) + vc_z10*sqrt(HC3(1,1));
1232
1233         if(li<=beta(1) && beta(1)<=ls)
1234             cob90_z_hc3(i) = 1;
1235         else
1236             cob90_z_hc3(i) = 0;
1237
1238         if(beta(1)<li)
1239             ncobesq90_z_hc3(i) = 1;
1240         else
1241             ncobesq90_z_hc3(i) = 0;
1242
1243         if(beta(1)>ls)
1244             ncobdi90_z_hc3(i) = 1;
1245         else
1246             ncobdi90_z_hc3(i) = 0;
1247         ampl90_z_hc3(i) = ls - li;
1248
1249         li = temp(1) - vc_z10*sqrt(HC4(1,1));
1250         ls = temp(1) + vc_z10*sqrt(HC4(1,1));
1251
1252         if(li<=beta(1) && beta(1)<=ls)
1253             cob90_z_hc4(i) = 1;
1254         else
1255             cob90_z_hc4(i) = 0;
1256
1257         if(beta(1)<li)
1258             ncobesq90_z_hc4(i) = 1;
1259         else
1260             ncobesq90_z_hc4(i) = 0;
1261
1262         if(beta(1)>ls)
1263             ncobdi90_z_hc4(i) = 1;
1264         else
1265             ncobdi90_z_hc4(i) = 0;
1266         ampl90_z_hc4(i) = ls - li;
1267
1268         li = temp(1) - vc_z10*sqrt(HC5(1,1));
1269         ls = temp(1) + vc_z10*sqrt(HC5(1,1));
1270
1271         if(li<=beta(1) && beta(1)<=ls)
1272             cob90_z_hc5(i) = 1;
1273         else
1274             cob90_z_hc5(i) = 0;

```

```

1275
1276         if(beta(1)<li)
1277             ncobesq90_z_hc5(i) = 1;
1278         else
1279             ncobesq90_z_hc5(i) = 0;
1280
1281         if(beta(1)>ls)
1282             ncobdi90_z_hc5(i) = 1;
1283         else
1284             ncobdi90_z_hc5(i) = 0;
1285         ampl90_z_hc5(i) = ls - li;
1286
1287         // CONFIANCA DE 95%
1288
1289         li = temp(1) - vc_z5*sqrt(OLS);
1290         ls = temp(1) + vc_z5*sqrt(OLS);
1291
1292         if(li<=beta(1) && beta(1)<=ls)
1293             cob95_z_ols(i) = 1;
1294         else
1295             cob95_z_ols(i) = 0;
1296
1297         if(beta(1)<li)
1298             ncobesq95_z_ols(i) = 1;
1299         else
1300             ncobesq95_z_ols(i) = 0;
1301
1302         if(beta(1)>ls)
1303             ncobdi95_z_ols(i) = 1;
1304         else
1305             ncobdi95_z_ols(i) = 0;
1306         ampl95_z_ols(i) = ls - li;
1307
1308         li = temp(1) - vc_z5*sqrt(HC0(1,1));
1309         ls = temp(1) + vc_z5*sqrt(HC0(1,1));
1310
1311         if(li<=beta(1) && beta(1)<=ls)
1312             cob95_z_hc0(i) = 1;
1313         else
1314             cob95_z_hc0(i) = 0;
1315
1316         if(beta(1)<li)
1317             ncobesq95_z_hc0(i) = 1;
1318         else
1319             ncobesq95_z_hc0(i) = 0;
1320
1321         if(beta(1)>ls)
1322             ncobdi95_z_hc0(i) = 1;
1323         else
1324             ncobdi95_z_hc0(i) = 0;
1325         ampl95_z_hc0(i) = ls - li;
1326
1327         li = temp(1) - vc_z5*sqrt(HC2(1,1));
1328         ls = temp(1) + vc_z5*sqrt(HC2(1,1));
1329
1330         if(li<=beta(1) && beta(1)<=ls)
1331             cob95_z_hc2(i) = 1;
1332         else

```

```
1333             cob95_z_hc2(i) = 0;
1334
1335         if(beta(1)<li)
1336             ncobesq95_z_hc2(i) = 1;
1337         else
1338             ncobesq95_z_hc2(i) = 0;
1339
1340         if(beta(1)>ls)
1341             ncobdi95_z_hc2(i) = 1;
1342         else
1343             ncobdi95_z_hc2(i) = 0;
1344         ampl95_z_hc2(i) = ls - li;
1345
1346         li = temp(1) - vc_z5*sqrt(HC3(1,1));
1347         ls = temp(1) + vc_z5*sqrt(HC3(1,1));
1348
1349         if(li<=beta(1) && beta(1)<=ls)
1350             cob95_z_hc3(i) = 1;
1351         else
1352             cob95_z_hc3(i) = 0;
1353
1354         if(beta(1)<li)
1355             ncobesq95_z_hc3(i) = 1;
1356         else
1357             ncobesq95_z_hc3(i) = 0;
1358
1359         if(beta(1)>ls)
1360             ncobdi95_z_hc3(i) = 1;
1361         else
1362             ncobdi95_z_hc3(i) = 0;
1363         ampl95_z_hc3(i) = ls - li;
1364
1365         li = temp(1) - vc_z5*sqrt(HC4(1,1));
1366         ls = temp(1) + vc_z5*sqrt(HC4(1,1));
1367
1368         if(li<=beta(1) && beta(1)<=ls)
1369             cob95_z_hc4(i) = 1;
1370         else
1371             cob95_z_hc4(i) = 0;
1372
1373         if(beta(1)<li)
1374             ncobesq95_z_hc4(i) = 1;
1375         else
1376             ncobesq95_z_hc4(i) = 0;
1377
1378         if(beta(1)>ls)
1379             ncobdi95_z_hc4(i) = 1;
1380         else
1381             ncobdi95_z_hc4(i) = 0;
1382         ampl95_z_hc4(i) = ls - li;
1383
1384         li = temp(1) - vc_z5*sqrt(HC5(1,1));
1385         ls = temp(1) + vc_z5*sqrt(HC5(1,1));
1386
1387         if(li<=beta(1) && beta(1)<=ls)
1388             cob95_z_hc5(i) = 1;
1389         else
1390             cob95_z_hc5(i) = 0;
```

```

1391
1392         if(beta(1)<li)
1393             ncobesq95_z_hc5(i) = 1;
1394         else
1395             ncobesq95_z_hc5(i) = 0;
1396
1397         if(beta(1)>ls)
1398             ncobdi95_z_hc5(i) = 1;
1399         else
1400             ncobdi95_z_hc5(i) = 0;
1401         ampl95_z_hc5(i) = ls - li;
1402
1403         // CONFIANCA DE 99%
1404
1405         li = temp(1) - vc_z1*sqrt(OLS);
1406         ls = temp(1) + vc_z1*sqrt(OLS);
1407
1408         if(li<=beta(1) && beta(1)<=ls)
1409             cob99_z_ols(i) = 1;
1410         else
1411             cob99_z_ols(i) = 0;
1412
1413         if(beta(1)<li)
1414             ncobesq99_z_ols(i) = 1;
1415         else
1416             ncobesq99_z_ols(i) = 0;
1417
1418         if(beta(1)>ls)
1419             ncobdi99_z_ols(i) = 1;
1420         else
1421             ncobdi99_z_ols(i) = 0;
1422         ampl99_z_ols(i) = ls - li;
1423
1424         li = temp(1) - vc_z1*sqrt(HC0(1,1));
1425         ls = temp(1) + vc_z1*sqrt(HC0(1,1));
1426
1427         if(li<=beta(1) && beta(1)<=ls)
1428             cob99_z_hc0(i) = 1;
1429         else
1430             cob99_z_hc0(i) = 0;
1431
1432         if(beta(1)<li)
1433             ncobesq99_z_hc0(i) = 1;
1434         else
1435             ncobesq99_z_hc0(i) = 0;
1436
1437         if(beta(1)>ls)
1438             ncobdi99_z_hc0(i) = 1;
1439         else
1440             ncobdi99_z_hc0(i) = 0;
1441         ampl99_z_hc0(i) = ls - li;
1442
1443         li = temp(1) - vc_z1*sqrt(HC2(1,1));
1444         ls = temp(1) + vc_z1*sqrt(HC2(1,1));
1445
1446         if(li<=beta(1) && beta(1)<=ls)
1447             cob99_z_hc2(i) = 1;
1448         else

```

```

1449             cob99_z_hc2(i) = 0;
1450
1451         if(beta(1)<li)
1452             ncobesq99_z_hc2(i) = 1;
1453         else
1454             ncobesq99_z_hc2(i) = 0;
1455
1456         if(beta(1)>ls)
1457             ncobdi99_z_hc2(i) = 1;
1458         else
1459             ncobdi99_z_hc2(i) = 0;
1460         ampl99_z_hc2(i) = ls - li;
1461
1462         li = temp(1) - vc_z1*sqrt(HC3(1,1));
1463         ls = temp(1) + vc_z1*sqrt(HC3(1,1));
1464
1465         if(li<=beta(1) && beta(1)<=ls)
1466             cob99_z_hc3(i) = 1;
1467         else
1468             cob99_z_hc3(i) = 0;
1469
1470         if(beta(1)<li)
1471             ncobesq99_z_hc3(i) = 1;
1472         else
1473             ncobesq99_z_hc3(i) = 0;
1474
1475         if(beta(1)>ls)
1476             ncobdi99_z_hc3(i) = 1;
1477         else
1478             ncobdi99_z_hc3(i) = 0;
1479         ampl99_z_hc3(i) = ls - li;
1480
1481         li = temp(1) - vc_z1*sqrt(HC4(1,1));
1482         ls = temp(1) + vc_z1*sqrt(HC4(1,1));
1483
1484         if(li<=beta(1) && beta(1)<=ls)
1485             cob99_z_hc4(i) = 1;
1486         else
1487             cob99_z_hc4(i) = 0;
1488
1489         if(beta(1)<li)
1490             ncobesq99_z_hc4(i) = 1;
1491         else
1492             ncobesq99_z_hc4(i) = 0;
1493
1494         if(beta(1)>ls)
1495             ncobdi99_z_hc4(i) = 1;
1496         else
1497             ncobdi99_z_hc4(i) = 0;
1498         ampl99_z_hc4(i) = ls - li;
1499
1500         li = temp(1) - vc_z1*sqrt(HC5(1,1));
1501         ls = temp(1) + vc_z1*sqrt(HC5(1,1));
1502
1503         if(li<=beta(1) && beta(1)<=ls)
1504             cob99_z_hc5(i) = 1;
1505         else
1506             cob99_z_hc5(i) = 0;

```

```

1507
1508         if(beta(1)<li)
1509             ncobesq99_z_hc5(i) = 1;
1510         else
1511             ncobesq99_z_hc5(i) = 0;
1512
1513         if(beta(1)>ls)
1514             ncobdi99_z_hc5(i) = 1;
1515         else
1516             ncobdi99_z_hc5(i) = 0;
1517         ampl99_z_hc5(i) = ls - li;
1518
1519         u_estrela.zeros();
1520         double u_estrela_numerador = 0, contador0_Z_j = 0, contador2_Z_j = 0,
1521             contador3_Z_j = 0, contador4_Z_j = 0, contador5_Z_j = 0;
1522
1523         mat Xtemp = X*temp;
1524
1525         // AQUI COMECA O LACO BOOTSTRAP.
1526         // A PARTIR DESSE PONTO SERAO CONSTRUIDOS ESQUEMAS BOOTSTRAP E BOOTSTRAP
1527         // DUPLO PARA GERACAO DE INTERVALOS DE CONFIANCAS MAIS PRECISOS.
1528         for(int k=0;k<nrep_boot;k++){
1529             u_estrela_numerador = 0;
1530             // VARIABEL RESPOSTA UTILIZADA NO BOOTSTRAP.
1531             vec y_estrela(nobs), t_estrela(nobs);
1532             // NUMERO ALEATORIO COM MEDIA ZERO E VARINCIA UM.
1533             if(dist_t==2){
1534                 for(int t=0;t<nobs;t++){
1535                     numero = gsl_ran_gaussian(r,1.0);
1536                     t_estrela(t) = numero;
1537                 }
1538             }
1539
1540             if(dist_t==1){
1541                 for(int t=0;t<nobs;t++){
1542                     numero = gsl_rng_uniform(r);
1543                     if(numero <= 0.5)
1544                         t_estrela(t) = -1;
1545                     if(numero > 0.5)
1546                         t_estrela(t) = 1;
1547                 }
1548                 //media_t_estrela << mean(t_estrela) << endl;
1549             }
1550         }
1551
1552         y_estrela = Xtemp+t_estrela%epsilon_chapeu/sqrt(1-h); // CONFERIDO.
1553
1554         // cout << gsl_rng_uniform_int(r,nobs) << endl;
1555         // AQUI TEMOS AS ESTIMATIVAS DE \hat{\beta}_{j}.
1556         // LEMBRANDO QUE NOSSO INTERESSE EH \hat{\beta}_{2}
1557
1558         // ESTIMATIVA DOS BETAS ESTRELA (BOOTSTRAP). \hat{\beta}_{j}
1559         beta_chapeu_boot = produtos*y_estrela;
1560         beta2_chapeu_boot_temp(k) = as_scalar(beta_chapeu_boot(1));
1561
1562         mat Xtemp_b = X*beta_chapeu_boot;
1563         mat resid2_b = pow((y_estrela-X*beta_chapeu_boot),2.0);
1564

```



```

1565      // MATRIZ OMEGA ESTIMADA. E UMA MATRIZ DIAGONAL N POR N.
1566      mat omega0 = diagmat(resid2_b%weight0);
1567      // MATRIZ OMEGA ESTIMADA. E UMA MATRIZ DIAGONAL N POR N.
1568      mat omega2 = diagmat(resid2_b%weight2);
1569      // MATRIZ OMEGA ESTIMADA. E UMA MATRIZ DIAGONAL N POR N.
1570      mat omega3 = diagmat(resid2_b%weight3);
1571      // MATRIZ OMEGA ESTIMADA. E UMA MATRIZ DIAGONAL N POR N.
1572      mat omega4 = diagmat(resid2_b%weight4);
1573      // MATRIZ OMEGA ESTIMADA. E UMA MATRIZ DIAGONAL N POR N.
1574      mat omega5 = diagmat(resid2_b%weight5);
1575
1576      mat HC0_b = P*omega0*Pt;
1577      mat HC2_b = P*omega2*Pt;
1578      mat HC3_b = P*omega3*Pt;
1579      mat HC4_b = P*omega4*Pt;
1580      mat HC5_b = P*omega5*Pt;
1581
1582      hc0_b(k) = sqrt(as_scalar(HC0_b(1,1)));
1583      hc2_b(k) = sqrt(as_scalar(HC2_b(1,1)));
1584      hc3_b(k) = sqrt(as_scalar(HC3_b(1,1)));
1585      hc4_b(k) = sqrt(as_scalar(HC4_b(1,1)));
1586      hc5_b(k) = sqrt(as_scalar(HC5_b(1,1)));
1587
1588      z_estrela0(k) = (as_scalar(beta2_chapeu_boot_temp(k)
1589                          -temp(1)))/sqrt(HC0_b(1,1));
1590      z_estrela2(k) = (as_scalar(beta2_chapeu_boot_temp(k)
1591                          -temp(1)))/sqrt(HC2_b(1,1));
1592      z_estrela3(k) = (as_scalar(beta2_chapeu_boot_temp(k)
1593                          -temp(1)))/sqrt(HC3_b(1,1));
1594      z_estrela4(k) = (as_scalar(beta2_chapeu_boot_temp(k)
1595                          -temp(1)))/sqrt(HC4_b(1,1));
1596      z_estrela5(k) = (as_scalar(beta2_chapeu_boot_temp(k)
1597                          -temp(1)))/sqrt(HC5_b(1,1));
1598
1599      // BETA2 DA REPLICA DE BOOTSTRAP.
1600      beta2(k) = beta_chapeu_boot(1);
1601      // SERA UTILIZADO NO BOOTSTRAP DUPLO.
1602      epsilon_chapeu_boot_duplo = y_estrela-X*beta_chapeu_boot;
1603
1604      // VETOR QUE IRA ARMAZENAR AS ESTIMATIVAS HC DO BOOTSTRAP DUPLO QUE
1605      // SERA UTILIZADO PARA CORRIGIR O ERRO PADRAO DO BOOTSTRAP EXTERIOR.
1606      vec hc_duplo0(nrep_boot_duplo);
1607      vec hc_duplo2(nrep_boot_duplo);
1608      vec hc_duplo3(nrep_boot_duplo);
1609      vec hc_duplo4(nrep_boot_duplo);
1610      vec hc_duplo5(nrep_boot_duplo);
1611
1612      double desvio0_b = sqrt(as_scalar(HC0_b(1,1)));
1613      double desvio2_b = sqrt(as_scalar(HC2_b(1,1)));
1614      double desvio3_b = sqrt(as_scalar(HC3_b(1,1)));
1615      double desvio4_b = sqrt(as_scalar(HC4_b(1,1)));
1616      double desvio5_b = sqrt(as_scalar(HC5_b(1,1)));
1617
1618      contador0_Z_j = 0;
1619      contador2_Z_j = 0;
1620      contador3_Z_j = 0;
1621      contador4_Z_j = 0;
1622      contador5_Z_j = 0;

```

```

1623
1624 // AQUI COMECA O BOOTSTRAP DUPLO.
1625 // #pragma omp parallel for
1626 for (int m=0; m<nrep_boot_duplo; m++){
1627
1628     // VARIÁVEL RESPOSTA DENTRO DO BOOTSTRAP DUPLO.
1629     vec y_estrela_estrela(nobs);
1630     // NÚMERO ALEATORIO COM MÉDIA 0 E VARIÂNCIA 1.
1631     vec t_estrela_estrela(nobs);
1632
1633     if (dist_t==2){
1634         for (int t=0; t<nobs; t++){
1635             numero = gsl_ran_gaussian(r, 1.0);
1636             t_estrela_estrela(t) = numero;
1637         }
1638     }
1639
1640     if (dist_t==1){
1641         for (int t=0; t<nobs; t++){
1642             numero = gsl_rng_uniform(r);
1643             if(numero<=0.5)
1644                 t_estrela_estrela(t) = -1;
1645             if (numero>0.5)
1646                 t_estrela_estrela(t) = 1;
1647         }
1648     }
1649
1650     y_estrela_estrela = Xtemp_b+
1651         t_estrela_estrela*epsilon_chapeu_boot_duplo/sqrt(1-h);
1652
1653     // A VARIÁVEL PRODUTOS REFERE-SE A  $(X'X)^{-1}X'$ 
1654     beta_chapeu_boot_duplo = produtos*y_estrela_estrela;
1655     // RESÍDUO AO QUADRADO.
1656     mat resid2_b_duplo = pow((y_estrela_estrela
1657         -X*beta_chapeu_boot_duplo), 2.0);
1658
1659     // MATRIZ ÔMEGA ESTIMADA. e UMA MATRIZ DIAGONAL N POR N.
1660     mat omega0 = diagmat(resid2_b_duplo%weight0);
1661     // MATRIZ ÔMEGA ESTIMADA. e UMA MATRIZ DIAGONAL N POR N.
1662     mat omega2 = diagmat(resid2_b_duplo%weight2);
1663     // MATRIZ ÔMEGA ESTIMADA. e UMA MATRIZ DIAGONAL N POR N.
1664     mat omega3 = diagmat(resid2_b_duplo%weight3);
1665     // MATRIZ ÔMEGA ESTIMADA. e UMA MATRIZ DIAGONAL N POR N.
1666     mat omega4 = diagmat(resid2_b_duplo%weight4);
1667     // MATRIZ ÔMEGA ESTIMADA. e UMA MATRIZ DIAGONAL N POR N.
1668     mat omega5 = diagmat(resid2_b_duplo%weight5);
1669
1670     mat HC0_b_duplo = P*omega0*Pt;
1671     mat HC2_b_duplo = P*omega2*Pt;
1672     mat HC3_b_duplo = P*omega3*Pt;
1673     mat HC4_b_duplo = P*omega4*Pt;
1674     mat HC5_b_duplo = P*omega5*Pt;
1675
1676     hc0_duplo(m) = sqrt(as_scalar(HC0_b_duplo(1,1)));
1677     hc2_duplo(m) = sqrt(as_scalar(HC2_b_duplo(1,1)));
1678     hc3_duplo(m) = sqrt(as_scalar(HC3_b_duplo(1,1)));
1679     hc4_duplo(m) = sqrt(as_scalar(HC4_b_duplo(1,1)));
1680     hc5_duplo(m) = sqrt(as_scalar(HC5_b_duplo(1,1)));

```

```

1681
1682         z_estrela_estrela0(m) = (as_scalar(beta_chapeu_boot_duplo(1)
1683             -beta_chapeu_boot(1)))/sqrt(HC0_b_duplo(1,1));
1684         z_estrela_estrela2(m) = (as_scalar(beta_chapeu_boot_duplo(1)
1685             -beta_chapeu_boot(1)))/sqrt(HC2_b_duplo(1,1));
1686         z_estrela_estrela3(m) = (as_scalar(beta_chapeu_boot_duplo(1)
1687             -beta_chapeu_boot(1)))/sqrt(HC3_b_duplo(1,1));
1688         z_estrela_estrela4(m) = (as_scalar(beta_chapeu_boot_duplo(1)
1689             -beta_chapeu_boot(1)))/sqrt(HC4_b_duplo(1,1));
1690         z_estrela_estrela5(m) = (as_scalar(beta_chapeu_boot_duplo(1)
1691             -beta_chapeu_boot(1)))/sqrt(HC5_b_duplo(1,1));
1692
1693         if(z_estrela_estrela0(m)<=z_estrela0(k))
1694             contador0_Z_j = contador0_Z_j+1;
1695         if(z_estrela_estrela2(m)<=z_estrela2(k))
1696             contador2_Z_j = contador2_Z_j+1;
1697         if(z_estrela_estrela3(m)<=z_estrela3(k))
1698             contador3_Z_j = contador3_Z_j+1;
1699         if(z_estrela_estrela4(m)<=z_estrela4(k))
1700             contador4_Z_j = contador4_Z_j+1;
1701         if(z_estrela_estrela5(m)<=z_estrela5(k))
1702             contador5_Z_j = contador5_Z_j+1;
1703
1704         if(beta_chapeu_boot_duplo(1)<=2*beta_chapeu_boot(1)-temp(1))
1705             u_estrela_numerador = 1+u_estrela_numerador;
1706
1707     } // AQUI TERMINA O LACO DO BOOTSTRAP DUPLO.
1708
1709     Z0_j(k) = contador0_Z_j/nrep_boot_duplo;
1710     Z2_j(k) = contador2_Z_j/nrep_boot_duplo;
1711     Z3_j(k) = contador3_Z_j/nrep_boot_duplo;
1712     Z4_j(k) = contador4_Z_j/nrep_boot_duplo;
1713     Z5_j(k) = contador5_Z_j/nrep_boot_duplo;
1714     u_estrela(k) = u_estrela_numerador/nrep_boot_duplo;
1715     betaj_estrela_menos_betaj(k) = beta_chapeu_boot(1)-temp(1);
1716
1717     // BOOTSTRAP T CORRIGINDO A QUANTIDADE NO DENOMIZADOR DA VARIÁVEL
1718     // z^(errado). O ESQUEMA CORRETO TAMBEM ESTA SENDO CALCULADO NES-
1719     // SE CODIGO FONTE. ELE FAZ USO DA VARIÁVEL Z_j PARA CORRIGIR O
1720     // QUANTIL CALCULADO SOBRE z^*.
1721
1722     z0_estrela_duplo(k) = (as_scalar(beta2_chapeu_boot_temp(k)-temp(1)))/
1723         (2*sum(hc0_duplo)/nrep_boot_duplo-desvio0_b);
1724     z2_estrela_duplo(k) = (as_scalar(beta2_chapeu_boot_temp(k)-temp(1)))/
1725         (2*sum(hc2_duplo)/nrep_boot_duplo-desvio2_b);
1726     z3_estrela_duplo(k) = (as_scalar(beta2_chapeu_boot_temp(k)-temp(1)))/
1727         (2*sum(hc3_duplo)/nrep_boot_duplo-desvio3_b);
1728     z4_estrela_duplo(k) = (as_scalar(beta2_chapeu_boot_temp(k)-temp(1)))/
1729         (2*sum(hc4_duplo)/nrep_boot_duplo-desvio4_b);
1730     z5_estrela_duplo(k) = (as_scalar(beta2_chapeu_boot_temp(k)-temp(1)))/
1731         (2*sum(hc5_duplo)/nrep_boot_duplo-desvio5_b);
1732 } // AQUI TERMINA O LACO BOOTSTRAP.
1733
1734 // @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
1735 //                                     INTERVALOS PARA 90%
1736 // @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
1737
1738 // CONTANDO CONVERGENCIAS PARA O BOOSTRAP T DUPLO. (ESSE ESQUEMA

```

```

1739 // DE BOOTSTRAP DUPLO NAO e CORRETO, CONTUDO, ESTOU EM CASOS DE
1740 // HOMOSCEDASTICIDADE. PRECISA SER VERIFICADO.)
1741
1742 double quantil0_inferior90 = myfunctions::quantil1(z0_estrela_duplo,
1743                                           0.95,nrep_boot);
1744 double quantil0_superior90 = myfunctions::quantil1(z0_estrela_duplo,
1745                                           0.05,nrep_boot);
1746 double quantil2_inferior90 = myfunctions::quantil1(z2_estrela_duplo,
1747                                           0.95,nrep_boot);
1748 double quantil2_superior90 = myfunctions::quantil1(z2_estrela_duplo,
1749                                           0.05,nrep_boot);
1750
1751 double quantil3_inferior90 = myfunctions::quantil1(z3_estrela_duplo,
1752                                           0.95,nrep_boot);
1753 double quantil3_superior90 = myfunctions::quantil1(z3_estrela_duplo,
1754                                           0.05,nrep_boot);
1755
1756 double quantil4_inferior90 = myfunctions::quantil1(z4_estrela_duplo,
1757                                           0.95,nrep_boot);
1758 double quantil4_superior90 = myfunctions::quantil1(z4_estrela_duplo,
1759                                           0.05,nrep_boot);
1760
1761 double quantil5_inferior90 = myfunctions::quantil1(z5_estrela_duplo,
1762                                           0.95,nrep_boot);
1763 double quantil5_superior90 = myfunctions::quantil1(z5_estrela_duplo,
1764                                           0.05,nrep_boot);
1765
1766 if(ncorrecoes==2){
1767     // AQUI ESTAMOS CORRIGINDO O CALCULO DOS LIMITES INFERIORES E
1768     // SUPERIORES DO INTERVALO DE CONFIANCA. ESSA CORRECAO FAZ USO
1769     // DO BOOTSTRAP EXTERIOR.
1770     li_0_90 = temp(1,0)-quantil0_inferior90*(2*sum(hc0_b)/nrep_boot
1771             -sqrt(HC0(1,1)));
1772     ls_0_90 = temp(1,0)-quantil0_superior90*(2*sum(hc0_b)/nrep_boot
1773             -sqrt(HC0(1,1)));
1774     li_2_90 = temp(1,0)-quantil2_inferior90*(2*sum(hc2_b)/nrep_boot
1775             -sqrt(HC2(1,1)));
1776     ls_2_90 = temp(1,0)-quantil2_superior90*(2*sum(hc2_b)/nrep_boot
1777             -sqrt(HC2(1,1)));
1778     li_3_90 = temp(1,0)-quantil3_inferior90*(2*sum(hc3_b)/nrep_boot
1779             -sqrt(HC3(1,1)));
1780     ls_3_90 = temp(1,0)-quantil3_superior90*(2*sum(hc3_b)/nrep_boot
1781             -sqrt(HC3(1,1)));
1782     li_4_90 = temp(1,0)-quantil4_inferior90*(2*sum(hc4_b)/nrep_boot
1783             -sqrt(HC4(1,1)));
1784     ls_4_90 = temp(1,0)-quantil4_superior90*(2*sum(hc4_b)/nrep_boot
1785             -sqrt(HC4(1,1)));
1786     li_5_90 = temp(1,0)-quantil5_inferior90*(2*sum(hc5_b)/nrep_boot
1787             -sqrt(HC5(1,1)));
1788     ls_5_90 = temp(1,0)-quantil5_superior90*(2*sum(hc5_b)/nrep_boot
1789             -sqrt(HC5(1,1)));
1790 }
1791
1792 if(ncorrecoes==1){
1793
1794     // AQUI ESTAMOS CONSTRUINDO OS LIMITES DOS INTERVALOS DE
1795     // CONFIANCAS SEM USAR A CORRECAO DO DESVIO QUE ENTRA NO
1796     // CALCULO DOS LIMITES.

```

```

1797
1798             li_0_90 = temp(1,0)-quantil0_inferior90*sqrt(HC0(1,1));
1799             ls_0_90 = temp(1,0)-quantil0_superior90*sqrt(HC0(1,1));
1800             li_2_90 = temp(1,0)-quantil2_inferior90*sqrt(HC2(1,1));
1801             ls_2_90 = temp(1,0)-quantil2_superior90*sqrt(HC2(1,1));
1802             li_3_90 = temp(1,0)-quantil3_inferior90*sqrt(HC3(1,1));
1803             ls_3_90 = temp(1,0)-quantil3_superior90*sqrt(HC3(1,1));
1804             li_4_90 = temp(1,0)-quantil4_inferior90*sqrt(HC4(1,1));
1805             ls_4_90 = temp(1,0)-quantil4_superior90*sqrt(HC4(1,1));
1806             li_5_90 = temp(1,0)-quantil5_inferior90*sqrt(HC5(1,1));
1807             ls_5_90 = temp(1,0)-quantil5_superior90*sqrt(HC5(1,1));
1808         }
1809
1810         if(beta(1)>=li_0_90&&beta(1)<=ls_0_90){
1811             cob_0_90_t_percentil_duplo(i) = 1;
1812         }else
1813             cob_0_90_t_percentil_duplo(i) = 0;
1814
1815         if(beta(1)>=li_2_90&&beta(1)<=ls_2_90){
1816             cob_2_90_t_percentil_duplo(i) = 1;
1817         }else
1818             cob_2_90_t_percentil_duplo(i) = 0;
1819
1820         if(beta(1)>=li_3_90&&beta(1)<=ls_3_90){
1821             cob_3_90_t_percentil_duplo(i) = 1;
1822         }else
1823             cob_3_90_t_percentil_duplo(i) = 0;
1824
1825         if(beta(1)>=li_4_90&&beta(1)<=ls_4_90){
1826             cob_4_90_t_percentil_duplo(i) = 1;
1827         }else
1828             cob_4_90_t_percentil_duplo(i) = 0;
1829
1830         if(beta(1)>=li_5_90&&beta(1)<=ls_5_90){
1831             cob_5_90_t_percentil_duplo(i) = 1;
1832         }else
1833             cob_5_90_t_percentil_duplo(i) = 0;
1834
1835         if(beta(1)<li_0_90){
1836             ncobesq_0_90_t_percentil_duplo(i) = 1;
1837         }else
1838             ncobesq_0_90_t_percentil_duplo(i) = 0;
1839
1840         if(beta(1)<li_2_90){
1841             ncobesq_2_90_t_percentil_duplo(i) = 1;
1842         }else
1843             ncobesq_2_90_t_percentil_duplo(i) = 0;
1844
1845         if(beta(1)<li_3_90){
1846             ncobesq_3_90_t_percentil_duplo(i) = 1;
1847         }else
1848             ncobesq_3_90_t_percentil_duplo(i) = 0;
1849
1850         if(beta(1)<li_4_90){
1851             ncobesq_4_90_t_percentil_duplo(i) = 1;
1852         }else
1853             ncobesq_4_90_t_percentil_duplo(i) = 0;
1854

```

```

1855         if(beta(1)<li_5_90){
1856             ncobesq_5_90_t_percentil_duplo(i) = 1;
1857         }else
1858             ncobesq_5_90_t_percentil_duplo(i) = 0;
1859
1860         if(beta(1)>ls_0_90){
1861             ncobdi_0_90_t_percentil_duplo(i) = 1;
1862         }else
1863             ncobdi_0_90_t_percentil_duplo(i) = 0;
1864
1865         ampl_0_90_t_percentil_duplo(i) = ls_0_90-li_0_90;
1866
1867         if(beta(1)>ls_2_90){
1868             ncobdi_2_90_t_percentil_duplo(i) = 1;
1869         }else
1870             ncobdi_2_90_t_percentil_duplo(i) = 0;
1871
1872         ampl_2_90_t_percentil_duplo(i) = ls_2_90-li_2_90;
1873
1874         if(beta(1)>ls_3_90){
1875             ncobdi_3_90_t_percentil_duplo(i) = 1;
1876         }else
1877             ncobdi_3_90_t_percentil_duplo(i) = 0;
1878
1879         ampl_3_90_t_percentil_duplo(i) = ls_3_90-li_0_90;
1880
1881         if(beta(1)>ls_4_90){
1882             ncobdi_4_90_t_percentil_duplo(i) = 1;
1883         }else
1884             ncobdi_4_90_t_percentil_duplo(i) = 0;
1885
1886         ampl_4_90_t_percentil_duplo(i) = ls_4_90-li_4_90;
1887
1888         if(beta(1)>ls_5_90){
1889             ncobdi_5_90_t_percentil_duplo(i) = 1;
1890         }else
1891             ncobdi_5_90_t_percentil_duplo(i) = 0;
1892
1893         ampl_5_90_t_percentil_duplo(i) = ls_5_90-li_5_90;
1894
1895         // CONTANDO CONVERGENCIAS PARA O BOOTSTRAP T
1896         // (AQUI NAO E O BOOSTRAP DUPL0.)
1897
1898         quantil0_inferior90 = myfunctions::quantil1(z_estrela0,0.95,nrep_boot);
1899         quantil0_superior90 = myfunctions::quantil1(z_estrela0,0.05,nrep_boot);
1900         li_0_90 = temp(1,0)-quantil0_inferior90*sqrt(HC0(1,1));
1901         ls_0_90 = temp(1,0)-quantil0_superior90*sqrt(HC0(1,1));
1902
1903         quantil2_inferior90 = myfunctions::quantil1(z_estrela2,0.95,nrep_boot);
1904         quantil2_superior90 = myfunctions::quantil1(z_estrela2,0.05,nrep_boot);
1905         li_2_90 = temp(1,0)-quantil2_inferior90*sqrt(HC2(1,1));
1906         ls_2_90 = temp(1,0)-quantil2_superior90*sqrt(HC2(1,1));
1907
1908         quantil3_inferior90 = myfunctions::quantil1(z_estrela3,0.95,nrep_boot);
1909         quantil3_superior90 = myfunctions::quantil1(z_estrela3,0.05,nrep_boot);
1910         li_3_90 = temp(1,0)-quantil3_inferior90*sqrt(HC3(1,1));
1911         ls_3_90 = temp(1,0)-quantil3_superior90*sqrt(HC3(1,1));
1912

```

```

1913     quantil4_inferior90 = myfunctions::quantil1(z_estrela4,0.95,nrep_boot);
1914     quantil4_superior90 = myfunctions::quantil1(z_estrela4,0.05,nrep_boot);
1915     li_4_90 = temp(1,0)-quantil4_inferior90*sqrt(HC4(1,1));
1916     ls_4_90 = temp(1,0)-quantil4_superior90*sqrt(HC4(1,1));
1917
1918     quantil5_inferior90 = myfunctions::quantil1(z_estrela5,0.95,nrep_boot);
1919     quantil5_superior90 = myfunctions::quantil1(z_estrela5,0.05,nrep_boot);
1920     li_5_90 = temp(1,0)-quantil5_inferior90*sqrt(HC5(1,1));
1921     ls_5_90 = temp(1,0)-quantil5_superior90*sqrt(HC5(1,1));
1922
1923     if(beta(1)>=li_0_90 && beta(1)<=ls_0_90){
1924         cob_0_90_t_percentil(i) = 1;
1925     }else
1926         cob_0_90_t_percentil(i) = 0;
1927
1928     if(beta(1)>=li_2_90 && beta(1)<=ls_2_90){
1929         cob_2_90_t_percentil(i) = 1;
1930     }else
1931         cob_2_90_t_percentil(i) = 0;
1932
1933     if(beta(1)>=li_3_90 && beta(1)<=ls_3_90){
1934         cob_3_90_t_percentil(i) = 1;
1935     }else
1936         cob_3_90_t_percentil(i) = 0;
1937
1938     if(beta(1)>=li_4_90 && beta(1)<=ls_4_90){
1939         cob_4_90_t_percentil(i) = 1;
1940     }else
1941         cob_4_90_t_percentil(i) = 0;
1942
1943     if(beta(1)>=li_5_90 && beta(1)<=ls_5_90){
1944         cob_5_90_t_percentil(i) = 1;
1945     }else
1946         cob_5_90_t_percentil(i) = 0;
1947
1948     if(beta(1)<li_0_90){
1949         ncobesq_0_90_t_percentil(i) = 1;
1950     }else
1951         ncobesq_0_90_t_percentil(i) = 0;
1952
1953     if(beta(1)<li_2_90){
1954         ncobesq_2_90_t_percentil(i) = 1;
1955     }else
1956         ncobesq_2_90_t_percentil(i) = 0;
1957
1958     if(beta(1)<li_3_90){
1959         ncobesq_3_90_t_percentil(i) = 1;
1960     }else
1961         ncobesq_3_90_t_percentil(i) = 0;
1962
1963     if(beta(1)<li_4_90){
1964         ncobesq_4_90_t_percentil(i) = 1;
1965     }else
1966         ncobesq_4_90_t_percentil(i) = 0;
1967
1968     if(beta(1)<li_5_90){
1969         ncobesq_5_90_t_percentil(i) = 1;
1970     }else

```

```

1971         ncobesq_5_90_t_percentil(i) = 0;
1972
1973         if(beta(1)>ls_0_90){
1974             ncobdi_0_90_t_percentil(i) = 1;
1975         }else
1976             ncobdi_0_90_t_percentil(i) = 0;
1977
1978         ampl_0_90_t_percentil(i) = ls_0_90-li_0_90;
1979
1980         if(beta(1)>ls_2_90){
1981             ncobdi_2_90_t_percentil(i) = 1;
1982         }else
1983             ncobdi_2_90_t_percentil(i) = 0;
1984
1985         ampl_2_90_t_percentil(i) = ls_2_90-li_2_90;
1986
1987         if(beta(1)>ls_3_90){
1988             ncobdi_3_90_t_percentil(i) = 1;
1989         }else
1990             ncobdi_3_90_t_percentil(i) = 0;
1991
1992         ampl_3_90_t_percentil(i) = ls_3_90-li_3_90;
1993
1994         if(beta(1)>ls_4_90){
1995             ncobdi_4_90_t_percentil(i) = 1;
1996         }else
1997             ncobdi_4_90_t_percentil(i) = 0;
1998
1999         ampl_4_90_t_percentil(i) = ls_4_90-li_4_90;
2000
2001         if(beta(1)>ls_5_90){
2002             ncobdi_5_90_t_percentil(i) = 1;
2003         }else
2004             ncobdi_5_90_t_percentil(i) = 0;
2005
2006         ampl_5_90_t_percentil(i) = ls_5_90-li_5_90;
2007
2008         // INTERVALO BOOTSTRAP PERCENTIL.
2009         double li90 = myfunctions::quantil1(beta2,0.05,nrep_boot);
2010         double ls90 = myfunctions::quantil1(beta2,0.95,nrep_boot);
2011         if(beta(1)>=li90 && beta(1)<=ls90)
2012             cob90_percentil(i) = 1;
2013         else
2014             cob90_percentil(i) = 0;
2015
2016         ampl90_percentil(i) = ls90-li90;
2017
2018         if(beta(1)<li90)
2019             ncobesq90_percentil(i) = 1;
2020         else
2021             ncobesq90_percentil(i) = 0;
2022
2023         if(beta(1)>ls90)
2024             ncobdi90_percentil(i) = 1;
2025         else
2026             ncobdi90_percentil(i) = 0;
2027
2028         // INTERVALO PERCENTIL BOOTSTRAP DUPLO - 90% (BOOTSTRAP EXTERIOR).

```



```

2029         double hat_ql90 = myfunctions::quantil1(u_estrela,0.05,nrep_boot);
2030         double hat_qu90 = myfunctions::quantil1(u_estrela,0.95,nrep_boot);
2031         ls90 = myfunctions::quantil1(beta2,hat_qu90,nrep_boot);
2032         li90 = myfunctions::quantil1(beta2,hat_ql90,nrep_boot);
2033
2034         //ls90 = temp(1)-myfunctions::quantil1(betaj_estrela_menos_betaj ,
2035         //hat_ql90,nrep_boot);
2036         //li90 = temp(1)-myfunctions::quantil1(betaj_estrela_menos_betaj ,
2037         //hat_qu90,nrep_boot);
2038
2039         ampl90_percentil_duplo(i) = ls90-li90;
2040
2041         if(li90<=beta(1) && beta(1)<=ls90)
2042             cob90_percentil_duplo(i) = 1;
2043         else
2044             cob90_percentil_duplo(i) = 0;
2045
2046         if(beta(1)<li90)
2047             ncobesq90_percentil_duplo(i) = 1;
2048         else
2049             ncobesq90_percentil_duplo(i) = 0;
2050
2051         if(beta(1)>ls90)
2052             ncobdi90_percentil_duplo(i) = 1;
2053         else
2054             ncobdi90_percentil_duplo(i) = 0;
2055
2056         // INTERVALO BOORSTRAP T DUPLO (CORRETO). BASEADO NO ALGORITMO
2057         // DAS PAGINAS 84-85 DO ARTIGO: IMPLEMENTING THE DOUBLE BOOTSTRAP ,
2058         // MCCULLOUGH AND VINOD, COMPUTATIONAL ECONOMICS, 1998.
2059
2060         quantil0_inferior90 = myfunctions::quantil1(z_estrela0 ,
2061         myfunctions::quantil1(Z0_j ,
2062         0.95,nrep_boot),nrep_boot);
2063         quantil0_superior90 = myfunctions::quantil1(z_estrela0 ,
2064         myfunctions::quantil1(Z0_j ,
2065         0.05,nrep_boot),nrep_boot);
2066
2067         li_0_90 = temp(1,0)-quantil0_inferior90*sqrt(HC0(1,1));
2068         ls_0_90 = temp(1,0)-quantil0_superior90*sqrt(HC0(1,1));
2069
2070         quantil2_inferior90 = myfunctions::quantil1(z_estrela2 ,
2071         myfunctions::quantil1(Z2_j ,
2072         0.95,nrep_boot),nrep_boot);
2073         quantil2_superior90 = myfunctions::quantil1(z_estrela2 ,
2074         myfunctions::quantil1(Z2_j ,
2075         0.05,nrep_boot),nrep_boot);
2076         li_2_90 = temp(1,0)-quantil2_inferior90*sqrt(HC2(1,1));
2077         ls_2_90 = temp(1,0)-quantil2_superior90*sqrt(HC2(1,1));
2078
2079         quantil3_inferior90 = myfunctions::quantil1(z_estrela3 ,
2080         myfunctions::quantil1(Z3_j ,
2081         0.95,nrep_boot),nrep_boot);
2082         quantil3_superior90 = myfunctions::quantil1(z_estrela3 ,
2083         myfunctions::quantil1(Z3_j ,
2084         0.05,nrep_boot),nrep_boot);
2085
2086         li_3_90 = temp(1,0)-quantil3_inferior90*sqrt(HC3(1,1));

```

```

2087     ls_3_90 = temp(1,0)-quantil3_superior90*sqrt(HC3(1,1));
2088
2089     quantil4_inferior90 = myfunctions::quantil1(z_estrela4,
2090                                           myfunctions::quantil1(Z4_j,
2091                                           0.95,nrep_boot),nrep_boot);
2092     quantil4_superior90 = myfunctions::quantil1(z_estrela4,
2093                                           myfunctions::quantil1(Z4_j,
2094                                           0.05,nrep_boot),nrep_boot);
2095
2096     li_4_90 = temp(1,0)-quantil4_inferior90*sqrt(HC4(1,1));
2097     ls_4_90 = temp(1,0)-quantil4_superior90*sqrt(HC4(1,1));
2098
2099     quantil5_inferior90 = myfunctions::quantil1(z_estrela5,
2100                                           myfunctions::quantil1(Z5_j,
2101                                           0.95,nrep_boot),nrep_boot);
2102     quantil5_superior90 = myfunctions::quantil1(z_estrela5,
2103                                           myfunctions::quantil1(Z5_j,
2104                                           0.05,nrep_boot),nrep_boot);
2105
2106     li_5_90 = temp(1,0)-quantil5_inferior90*sqrt(HC5(1,1));
2107     ls_5_90 = temp(1,0)-quantil5_superior90*sqrt(HC5(1,1));
2108
2109     if(beta(1)>=li_0_90 && beta(1)<=ls_0_90){
2110         cob_0_90_t_percentil_duplo1(i) = 1;
2111     }else
2112         cob_0_90_t_percentil_duplo1(i) = 0;
2113
2114     if(beta(1)>=li_2_90 && beta(1)<=ls_2_90){
2115         cob_2_90_t_percentil_duplo1(i) = 1;
2116     }else
2117         cob_2_90_t_percentil_duplo1(i) = 0;
2118
2119     if(beta(1)>=li_3_90 && beta(1)<=ls_3_90){
2120         cob_3_90_t_percentil_duplo1(i) = 1;
2121     }else
2122         cob_3_90_t_percentil_duplo1(i) = 0;
2123
2124     if(beta(1)>=li_4_90 && beta(1)<=ls_4_90){
2125         cob_4_90_t_percentil_duplo1(i) = 1;
2126     }else
2127         cob_4_90_t_percentil_duplo1(i) = 0;
2128
2129     if(beta(1)>=li_5_90 && beta(1)<=ls_5_90){
2130         cob_5_90_t_percentil_duplo1(i) = 1;
2131     }else
2132         cob_5_90_t_percentil_duplo1(i) = 0;
2133
2134     if(beta(1)<li_0_90){
2135         ncobesq_0_90_t_percentil_duplo1(i) = 1;
2136     }else
2137         ncobesq_0_90_t_percentil_duplo1(i) = 0;
2138
2139     if(beta(1)<li_2_90){
2140         ncobesq_2_90_t_percentil_duplo1(i) = 1;
2141     }else
2142         ncobesq_2_90_t_percentil_duplo1(i) = 0;
2143
2144     if(beta(1)<li_3_90){

```

```

2145         ncobesq_3_90_t_percentil_duplo1(i) = 1;
2146     }else
2147         ncobesq_3_90_t_percentil_duplo1(i) = 0;
2148
2149     if(beta(1)<li_4_90){
2150         ncobesq_4_90_t_percentil_duplo1(i) = 1;
2151     }else
2152         ncobesq_4_90_t_percentil_duplo1(i) = 0;
2153
2154     if(beta(1)<li_5_90){
2155         ncobesq_5_90_t_percentil_duplo1(i) = 1;
2156     }else
2157         ncobesq_5_90_t_percentil_duplo1(i) = 0;
2158
2159     if(beta(1)>ls_0_90){
2160         ncobdi_0_90_t_percentil_duplo1(i) = 1;
2161     }else
2162         ncobdi_0_90_t_percentil_duplo1(i) = 0;
2163
2164     ampl_0_90_t_percentil_duplo1(i) = ls_0_90-li_0_90;
2165
2166     if(beta(1)>ls_2_90){
2167         ncobdi_2_90_t_percentil_duplo1(i) = 1;
2168     }else
2169         ncobdi_2_90_t_percentil_duplo1(i) = 0;
2170
2171     ampl_2_90_t_percentil_duplo1(i) = ls_2_90-li_2_90;
2172
2173     if(beta(1)>ls_3_90){
2174         ncobdi_3_90_t_percentil_duplo1(i) = 1;
2175     }else
2176         ncobdi_3_90_t_percentil_duplo1(i) = 0;
2177
2178     ampl_3_90_t_percentil_duplo1(i) = ls_3_90-li_3_90;
2179
2180     if(beta(1)>ls_4_90){
2181         ncobdi_4_90_t_percentil_duplo1(i) = 1;
2182     }else
2183         ncobdi_4_90_t_percentil_duplo1(i) = 0;
2184
2185     ampl_4_90_t_percentil_duplo1(i) = ls_4_90-li_4_90;
2186
2187     if(beta(1)>ls_5_90){
2188         ncobdi_5_90_t_percentil_duplo1(i) = 1;
2189     }else
2190         ncobdi_5_90_t_percentil_duplo1(i) = 0;
2191
2192     ampl_5_90_t_percentil_duplo1(i) = ls_5_90-li_5_90;
2193
2194     // @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
2195     //                                     INTERVALOS PARA 95%
2196     // @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
2197
2198     // CONTANDO CONVERGENCIAS PARA O BOOSTRAP T DUPLO. (ESSE ESQUEMA DE
2199     // BOOSTRAP DUPLO NAO e CORRETO, CONTUDO, ESTOU EM CASOS DE HOMOS-
2200     // CEDASTICIDADE. PRECISA SER VERIFICADO.)
2201
2202     double quantil0_inferior95 = myfunctions::quantil1(z0_estrela_duplo,

```

```

2203             0.975,nrep_boot);
2204 double quantil0_superior95 = myfunctions::quantil1(z0_estrela_duplo,
2205             0.025,nrep_boot);
2206
2207 double quantil2_inferior95 = myfunctions::quantil1(z2_estrela_duplo,
2208             0.975,nrep_boot);
2209 double quantil2_superior95 = myfunctions::quantil1(z2_estrela_duplo,
2210             0.025,nrep_boot);
2211
2212 double quantil3_inferior95 = myfunctions::quantil1(z3_estrela_duplo,
2213             0.975,nrep_boot);
2214 double quantil3_superior95 = myfunctions::quantil1(z3_estrela_duplo,
2215             0.025,nrep_boot);
2216
2217 double quantil4_inferior95 = myfunctions::quantil1(z4_estrela_duplo,
2218             0.975,nrep_boot);
2219 double quantil4_superior95 = myfunctions::quantil1(z4_estrela_duplo,
2220             0.025,nrep_boot);
2221
2222 double quantil5_inferior95 = myfunctions::quantil1(z5_estrela_duplo,
2223             0.975,nrep_boot);
2224 double quantil5_superior95 = myfunctions::quantil1(z5_estrela_duplo,
2225             0.025,nrep_boot);
2226
2227 if(ncorrecoes==2){
2228     // AQUI ESTAMOS CORRIGINDO O CALCULO DOS LIMITES INFERIORES E
2229     // SUPERIORES DO INTERVALO DE CONFIANCA. ESSA CORRECAO FAZ USO
2230     // DO BOOTSTRAP EXTERIOR.
2231     li_0_95 = temp(1,0)-quantil0_inferior95*(2*sum(hc0_b)/nrep_boot
2232     -sqrt(HC0(1,1)));
2233     ls_0_95 = temp(1,0)-quantil0_superior95*(2*sum(hc0_b)/nrep_boot
2234     -sqrt(HC0(1,1)));
2235     li_2_95 = temp(1,0)-quantil2_inferior95*(2*sum(hc2_b)/nrep_boot
2236     -sqrt(HC2(1,1)));
2237     ls_2_95 = temp(1,0)-quantil2_superior95*(2*sum(hc2_b)/nrep_boot
2238     -sqrt(HC2(1,1)));
2239     li_3_95 = temp(1,0)-quantil3_inferior95*(2*sum(hc3_b)/nrep_boot
2240     -sqrt(HC3(1,1)));
2241     ls_3_95 = temp(1,0)-quantil3_superior95*(2*sum(hc3_b)/nrep_boot
2242     -sqrt(HC3(1,1)));
2243     li_4_95 = temp(1,0)-quantil4_inferior95*(2*sum(hc4_b)/nrep_boot
2244     -sqrt(HC4(1,1)));
2245     ls_4_95 = temp(1,0)-quantil4_superior95*(2*sum(hc4_b)/nrep_boot
2246     -sqrt(HC4(1,1)));
2247     li_5_95 = temp(1,0)-quantil5_inferior95*(2*sum(hc5_b)/nrep_boot
2248     -sqrt(HC5(1,1)));
2249     ls_5_95 = temp(1,0)-quantil5_superior95*(2*sum(hc5_b)/nrep_boot
2250     -sqrt(HC5(1,1)));
2251 }
2252
2253 if(ncorrecoes==1){
2254
2255     // AQUI ESTAMOS CONSTRUINDO OS LIMITES DOS INTERVALOS DE CON-
2256     // FIANCAS SEM USAR A CORRECAO DO DESVIO QUE ENTRA NO CALCULO
2257     // DOS LIMITES.
2258
2259     li_0_95 = temp(1,0)-quantil0_inferior95*sqrt(HC0(1,1));
2260     ls_0_95 = temp(1,0)-quantil0_superior95*sqrt(HC0(1,1));

```

```

2261         li_2_95 = temp(1,0)-quantil2_inferior95*sqrt(HC2(1,1));
2262         ls_2_95 = temp(1,0)-quantil2_superior95*sqrt(HC2(1,1));
2263         li_3_95 = temp(1,0)-quantil3_inferior95*sqrt(HC3(1,1));
2264         ls_3_95 = temp(1,0)-quantil3_superior95*sqrt(HC3(1,1));
2265         li_4_95 = temp(1,0)-quantil4_inferior95*sqrt(HC4(1,1));
2266         ls_4_95 = temp(1,0)-quantil4_superior95*sqrt(HC4(1,1));
2267         li_5_95 = temp(1,0)-quantil5_inferior95*sqrt(HC5(1,1));
2268         ls_5_95 = temp(1,0)-quantil5_superior95*sqrt(HC5(1,1));
2269     }
2270
2271     if(beta(1)>=li_0_95&&beta(1)<=ls_0_95){
2272         cob_0_95_t_percentil_duplo(i) = 1;
2273     }else
2274         cob_0_95_t_percentil_duplo(i) = 0;
2275
2276     if(beta(1)>=li_2_95&&beta(1)<=ls_2_95){
2277         cob_2_95_t_percentil_duplo(i) = 1;
2278     }else
2279         cob_2_95_t_percentil_duplo(i) = 0;
2280
2281     if(beta(1)>=li_3_95&&beta(1)<=ls_3_95){
2282         cob_3_95_t_percentil_duplo(i) = 1;
2283     }else
2284         cob_3_95_t_percentil_duplo(i) = 0;
2285
2286     if(beta(1)>=li_4_95&&beta(1)<=ls_4_95){
2287         cob_4_95_t_percentil_duplo(i) = 1;
2288     }else
2289         cob_4_95_t_percentil_duplo(i) = 0;
2290
2291     if(beta(1)>=li_5_95&&beta(1)<=ls_5_95){
2292         cob_5_95_t_percentil_duplo(i) = 1;
2293     }else
2294         cob_5_95_t_percentil_duplo(i) = 0;
2295
2296     if(beta(1)<li_0_95){
2297         ncobesq_0_95_t_percentil_duplo(i) = 1;
2298     }else
2299         ncobesq_0_95_t_percentil_duplo(i) = 0;
2300
2301     if(beta(1)<li_2_95){
2302         ncobesq_2_95_t_percentil_duplo(i) = 1;
2303     }else
2304         ncobesq_2_95_t_percentil_duplo(i) = 0;
2305
2306     if(beta(1)<li_3_95){
2307         ncobesq_3_95_t_percentil_duplo(i) = 1;
2308     }else
2309         ncobesq_3_95_t_percentil_duplo(i) = 0;
2310
2311     if(beta(1)<li_4_95){
2312         ncobesq_4_95_t_percentil_duplo(i) = 1;
2313     }else
2314         ncobesq_4_95_t_percentil_duplo(i) = 0;
2315
2316     if(beta(1)<li_5_95){
2317         ncobesq_5_95_t_percentil_duplo(i) = 1;
2318     }else

```

```

2319             ncobesq_5_95_t_percentil_duplo(i) = 0;
2320
2321         if(beta(1)>ls_0_95){
2322             ncobdi_0_95_t_percentil_duplo(i) = 1;
2323         }else
2324             ncobdi_0_95_t_percentil_duplo(i) = 0;
2325
2326         ampl_0_95_t_percentil_duplo(i) = ls_0_95-li_0_95;
2327
2328         if(beta(1)>ls_2_95){
2329             ncobdi_2_95_t_percentil_duplo(i) = 1;
2330         }else
2331             ncobdi_2_95_t_percentil_duplo(i) = 0;
2332
2333         ampl_2_95_t_percentil_duplo(i) = ls_2_95-li_2_95;
2334
2335         if(beta(1)>ls_3_95){
2336             ncobdi_3_95_t_percentil_duplo(i) = 1;
2337         }else
2338             ncobdi_3_95_t_percentil_duplo(i) = 0;
2339
2340         ampl_3_95_t_percentil_duplo(i) = ls_3_95-li_0_95;
2341
2342         if(beta(1)>ls_4_95){
2343             ncobdi_4_95_t_percentil_duplo(i) = 1;
2344         }else
2345             ncobdi_4_95_t_percentil_duplo(i) = 0;
2346
2347         ampl_4_95_t_percentil_duplo(i) = ls_4_95-li_4_95;
2348
2349         if(beta(1)>ls_5_95){
2350             ncobdi_5_95_t_percentil_duplo(i) = 1;
2351         }else
2352             ncobdi_5_95_t_percentil_duplo(i) = 0;
2353
2354         ampl_5_95_t_percentil_duplo(i) = ls_5_95-li_5_95;
2355
2356         // CONTANDO CONVERGENCIAS PARA O BOOTSTRAP T
2357         // (AQUI Na0 e O BOOTSTRAP DUPL0.)
2358
2359         quantil0_inferior95 = myfunctions::quantil1(z_estrela0,0.975,nrep_boot);
2360         quantil0_superior95 = myfunctions::quantil1(z_estrela0,0.025,nrep_boot);
2361         li_0_95 = temp(1,0)-quantil0_inferior95*sqrt(HC0(1,1));
2362         ls_0_95 = temp(1,0)-quantil0_superior95*sqrt(HC0(1,1));
2363
2364         lils_hc0_bootstrap << li_0_95 << endl;
2365         lils_hc0_bootstrap << ls_0_95 << endl;
2366
2367         quantil2_inferior95 = myfunctions::quantil1(z_estrela2,0.975,nrep_boot);
2368         quantil2_superior95 = myfunctions::quantil1(z_estrela2,0.025,nrep_boot);
2369         li_2_95 = temp(1,0)-quantil2_inferior95*sqrt(HC2(1,1));
2370         ls_2_95 = temp(1,0)-quantil2_superior95*sqrt(HC2(1,1));
2371
2372         lils_hc2_bootstrap << li_2_95 << endl;
2373         lils_hc2_bootstrap << ls_2_95 << endl;
2374
2375         quantil3_inferior95 = myfunctions::quantil1(z_estrela3,0.975,nrep_boot);
2376         quantil3_superior95 = myfunctions::quantil1(z_estrela3,0.025,nrep_boot);

```

```

2377     li_3_95 = temp(1,0)-quantil3_inferior95*sqrt(HC3(1,1));
2378     ls_3_95 = temp(1,0)-quantil3_superior95*sqrt(HC3(1,1));
2379
2380     lils_hc3_bootstrapt << li_3_95 << endl;
2381     lils_hc3_bootstrapt << ls_3_95 << endl;
2382
2383     quantil4_inferior95 = myfunctions::quantil1(z_estrela4,0.975,nrep_boot);
2384     quantil4_superior95 = myfunctions::quantil1(z_estrela4,0.025,nrep_boot);
2385     li_4_95 = temp(1,0)-quantil4_inferior95*sqrt(HC4(1,1));
2386     ls_4_95 = temp(1,0)-quantil4_superior95*sqrt(HC4(1,1));
2387
2388     lils_hc4_bootstrapt << li_4_95 << endl;
2389     lils_hc4_bootstrapt << ls_4_95 << endl;
2390
2391     quantil5_inferior95 = myfunctions::quantil1(z_estrela5,0.975,nrep_boot);
2392     quantil5_superior95 = myfunctions::quantil1(z_estrela5,0.025,nrep_boot);
2393     li_5_95 = temp(1,0)-quantil5_inferior95*sqrt(HC5(1,1));
2394     ls_5_95 = temp(1,0)-quantil5_superior95*sqrt(HC5(1,1));
2395
2396     lils_hc5_bootstrapt << li_5_95 << endl;
2397     lils_hc5_bootstrapt << ls_5_95 << endl;
2398
2399     if(beta(1)>=li_0_95 && beta(1)<=ls_0_95){
2400         cob_0_95_t_percentil(i) = 1;
2401     }else
2402         cob_0_95_t_percentil(i) = 0;
2403
2404     if(beta(1)>=li_2_95 && beta(1)<=ls_2_95){
2405         cob_2_95_t_percentil(i) = 1;
2406     }else
2407         cob_2_95_t_percentil(i) = 0;
2408
2409     if(beta(1)>=li_3_95 && beta(1)<=ls_3_95){
2410         cob_3_95_t_percentil(i) = 1;
2411     }else
2412         cob_3_95_t_percentil(i) = 0;
2413
2414     if(beta(1)>=li_4_95 && beta(1)<=ls_4_95){
2415         cob_4_95_t_percentil(i) = 1;
2416     }else
2417         cob_4_95_t_percentil(i) = 0;
2418
2419     if(beta(1)>=li_5_95 && beta(1)<=ls_5_95){
2420         cob_5_95_t_percentil(i) = 1;
2421     }else
2422         cob_5_95_t_percentil(i) = 0;
2423
2424     if(beta(1)<li_0_95){
2425         ncobesq_0_95_t_percentil(i) = 1;
2426     }else
2427         ncobesq_0_95_t_percentil(i) = 0;
2428
2429     if(beta(1)<li_2_95){
2430         ncobesq_2_95_t_percentil(i) = 1;
2431     }else
2432         ncobesq_2_95_t_percentil(i) = 0;
2433
2434     if(beta(1)<li_3_95){

```

```

2435         ncobesq_3_95_t_percentil(i) = 1;
2436     }else
2437         ncobesq_3_95_t_percentil(i) = 0;
2438
2439     if(beta(1)<li_4_95){
2440         ncobesq_4_95_t_percentil(i) = 1;
2441     }else
2442         ncobesq_4_95_t_percentil(i) = 0;
2443
2444     if(beta(1)<li_5_95){
2445         ncobesq_5_95_t_percentil(i) = 1;
2446     }else
2447         ncobesq_5_95_t_percentil(i) = 0;
2448
2449     if(beta(1)>ls_0_95){
2450         ncobdi_0_95_t_percentil(i) = 1;
2451     }else
2452         ncobdi_0_95_t_percentil(i) = 0;
2453
2454     ampl_0_95_t_percentil(i) = ls_0_95-li_0_95;
2455
2456     if(beta(1)>ls_2_95){
2457         ncobdi_2_95_t_percentil(i) = 1;
2458     }else
2459         ncobdi_2_95_t_percentil(i) = 0;
2460
2461     ampl_2_95_t_percentil(i) = ls_2_95-li_2_95;
2462
2463     if(beta(1)>ls_3_95){
2464         ncobdi_3_95_t_percentil(i) = 1;
2465     }else
2466         ncobdi_3_95_t_percentil(i) = 0;
2467
2468     ampl_3_95_t_percentil(i) = ls_3_95-li_3_95;
2469
2470     if(beta(1)>ls_4_95){
2471         ncobdi_4_95_t_percentil(i) = 1;
2472     }else
2473         ncobdi_4_95_t_percentil(i) = 0;
2474
2475     ampl_4_95_t_percentil(i) = ls_4_95-li_4_95;
2476
2477     if(beta(1)>ls_5_95){
2478         ncobdi_5_95_t_percentil(i) = 1;
2479     }else
2480         ncobdi_5_95_t_percentil(i) = 0;
2481
2482     ampl_5_95_t_percentil(i) = ls_5_95-li_5_95;
2483
2484
2485     // INTERVALO BOOTSTRAP PERCENTIL.
2486     double li95 = myfunctions::quantil1(beta2,0.025,nrep_boot);
2487     double ls95 = myfunctions::quantil1(beta2,0.975,nrep_boot);
2488
2489     lils_percentil << li95 << endl;
2490     lils_percentil << ls95 << endl;
2491
2492     if(beta(1)>=li95 && beta(1)<=ls95)

```



```

2493         cob95_percentil(i) = 1;
2494     else
2495         cob95_percentil(i) = 0;
2496
2497     ampl95_percentil(i) = ls95-li95;
2498
2499     if(beta(1)<li95)
2500         ncobesq95_percentil(i) = 1;
2501     else
2502         ncobesq95_percentil(i) = 0;
2503
2504     if(beta(1)>ls95)
2505         ncobdi95_percentil(i) = 1;
2506     else
2507         ncobdi95_percentil(i) = 0;
2508
2509     // INTERVALO PERCENTIL BOOTSTRAP DUPLO - 95% (BOOTSTRAP EXTERIOR).
2510
2511     double hat_ql95 = myfunctions::quantil1(u_estrela,0.025,nrep_boot);
2512     double hat_qu95 = myfunctions::quantil1(u_estrela,0.975,nrep_boot);
2513     ls95 = myfunctions::quantil1(beta2,hat_qu95,nrep_boot);
2514     li95 = myfunctions::quantil1(beta2,hat_ql95,nrep_boot);
2515
2516     lils_percentil_duplo << li95 << endl;
2517     lils_percentil_duplo << ls95 << endl;
2518
2519     ampl95_percentil_duplo(i) = ls95-li95;
2520
2521     if(li95<=beta(1) && beta(1)<=ls95)
2522         cob95_percentil_duplo(i) = 1;
2523     else
2524         cob95_percentil_duplo(i) = 0;
2525
2526     if(beta(1)<li95)
2527         ncobesq95_percentil_duplo(i) = 1;
2528     else
2529         ncobesq95_percentil_duplo(i) = 0;
2530
2531     if(beta(1)>ls95)
2532         ncobdi95_percentil_duplo(i) = 1;
2533     else
2534         ncobdi95_percentil_duplo(i) = 0;
2535
2536     // INTERVALO BOORSTRAP T DUPLO (CORRETO). BASEADO NO ALGORITMO
2537     // DAS PAGINAS 84-85 DO ARTIGO: IMPLEMENTING THE DOUBLE BOOTSTRAP,
2538     // MCCULLOUGH AND VINOD, COMPUTATIONAL ECONOMICS, 1998.
2539
2540     quantil0_inferior95 = myfunctions::quantil1(z_estrela0,
2541         myfunctions::quantil1(Z0_j,
2542             0.975,nrep_boot),nrep_boot);
2543     quantil0_superior95 = myfunctions::quantil1(z_estrela0,
2544         myfunctions::quantil1(Z0_j,
2545             0.025,nrep_boot),nrep_boot);
2546
2547     li_0_95 = temp(1,0)-quantil0_inferior95*sqrt(HC0(1,1));
2548     ls_0_95 = temp(1,0)-quantil0_superior95*sqrt(HC0(1,1));
2549
2550     lils_hc0_bootstrappt_duplo << li_0_95 << endl;

```

```

2551         lils_hc0_bootstrap_duplo << ls_0_95 << endl;
2552
2553         quantil2_inferior95 = myfunctions::quantil1(z_estrela2,
2554             myfunctions::quantil1(Z2_j,
2555                 0.975,nrep_boot),nrep_boot);
2556         quantil2_superior95 = myfunctions::quantil1(z_estrela2,
2557             myfunctions::quantil1(Z2_j,
2558                 0.025,nrep_boot),nrep_boot);
2559
2560         li_2_95 = temp(1,0)-quantil2_inferior95*sqrt(HC2(1,1));
2561         ls_2_95 = temp(1,0)-quantil2_superior95*sqrt(HC2(1,1));
2562
2563         lils_hc2_bootstrap_duplo << li_2_95 << endl;
2564         lils_hc2_bootstrap_duplo << ls_2_95 << endl;
2565
2566         quantil3_inferior95 = myfunctions::quantil1(z_estrela3,
2567             myfunctions::quantil1(Z3_j,
2568                 0.975,nrep_boot),nrep_boot);
2569
2570         quantil3_superior95 = myfunctions::quantil1(z_estrela3,
2571             myfunctions::quantil1(Z3_j,
2572                 0.025,nrep_boot),nrep_boot);
2573
2574         li_3_95 = temp(1,0)-quantil3_inferior95*sqrt(HC3(1,1));
2575         ls_3_95 = temp(1,0)-quantil3_superior95*sqrt(HC3(1,1));
2576
2577         lils_hc3_bootstrap_duplo << li_3_95 << endl;
2578         lils_hc3_bootstrap_duplo << ls_3_95 << endl;
2579
2580         quantil4_inferior95 = myfunctions::quantil1(z_estrela4,
2581             myfunctions::quantil1(Z4_j,
2582                 0.975,nrep_boot),nrep_boot);
2583         quantil4_superior95 = myfunctions::quantil1(z_estrela4,
2584             myfunctions::quantil1(Z4_j,
2585                 0.025,nrep_boot),nrep_boot);
2586
2587         li_4_95 = temp(1,0)-quantil4_inferior95*sqrt(HC4(1,1));
2588         ls_4_95 = temp(1,0)-quantil4_superior95*sqrt(HC4(1,1));
2589
2590         lils_hc4_bootstrap_duplo << li_4_95 << endl;
2591         lils_hc4_bootstrap_duplo << ls_4_95 << endl;
2592
2593         quantil5_inferior95 = myfunctions::quantil1(z_estrela5,
2594             myfunctions::quantil1(Z5_j,
2595                 0.975,nrep_boot),nrep_boot);
2596         quantil5_superior95 = myfunctions::quantil1(z_estrela5,
2597             myfunctions::quantil1(Z5_j,
2598                 0.025,nrep_boot),nrep_boot);
2599
2600         li_5_95 = temp(1,0)-quantil5_inferior95*sqrt(HC5(1,1));
2601         ls_5_95 = temp(1,0)-quantil5_superior95*sqrt(HC5(1,1));
2602
2603         lils_hc5_bootstrap_duplo << li_5_95 << endl;
2604         lils_hc5_bootstrap_duplo << ls_5_95 << endl;
2605
2606         if(beta(1)>=li_0_95 && beta(1)<=ls_0_95){
2607             cob_0_95_t_percentil_duplo1(i) = 1;
2608         }else

```

```

2609             cob_0_95_t_percentil_duplo1(i) = 0;
2610
2611         if(beta(1)>=li_2_95 && beta(1)<=ls_2_95){
2612             cob_2_95_t_percentil_duplo1(i) = 1;
2613         }else
2614             cob_2_95_t_percentil_duplo1(i) = 0;
2615
2616         if(beta(1)>=li_3_95 && beta(1)<=ls_3_95){
2617             cob_3_95_t_percentil_duplo1(i) = 1;
2618         }else
2619             cob_3_95_t_percentil_duplo1(i) = 0;
2620
2621         if(beta(1)>=li_4_95 && beta(1)<=ls_4_95){
2622             cob_4_95_t_percentil_duplo1(i) = 1;
2623         }else
2624             cob_4_95_t_percentil_duplo1(i) = 0;
2625
2626         if(beta(1)>=li_5_95 && beta(1)<=ls_5_95){
2627             cob_5_95_t_percentil_duplo1(i) = 1;
2628         }else
2629             cob_5_95_t_percentil_duplo1(i) = 0;
2630
2631         if(beta(1)<li_0_95){
2632             ncobesq_0_95_t_percentil_duplo1(i) = 1;
2633         }else
2634             ncobesq_0_95_t_percentil_duplo1(i) = 0;
2635
2636         if(beta(1)<li_2_95){
2637             ncobesq_2_95_t_percentil_duplo1(i) = 1;
2638         }else
2639             ncobesq_2_95_t_percentil_duplo1(i) = 0;
2640
2641         if(beta(1)<li_3_95){
2642             ncobesq_3_95_t_percentil_duplo1(i) = 1;
2643         }else
2644             ncobesq_3_95_t_percentil_duplo1(i) = 0;
2645
2646         if(beta(1)<li_4_95){
2647             ncobesq_4_95_t_percentil_duplo1(i) = 1;
2648         }else
2649             ncobesq_4_95_t_percentil_duplo1(i) = 0;
2650
2651         if(beta(1)<li_5_95){
2652             ncobesq_5_95_t_percentil_duplo1(i) = 1;
2653         }else
2654             ncobesq_5_95_t_percentil_duplo1(i) = 0;
2655
2656         if(beta(1)>ls_0_95){
2657             ncobdi_0_95_t_percentil_duplo1(i) = 1;
2658         }else
2659             ncobdi_0_95_t_percentil_duplo1(i) = 0;
2660
2661         ampl_0_95_t_percentil_duplo1(i) = ls_0_95-li_0_95;
2662
2663         if(beta(1)>ls_2_95){
2664             ncobdi_2_95_t_percentil_duplo1(i) = 1;
2665         }else
2666             ncobdi_2_95_t_percentil_duplo1(i) = 0;

```

```

2667      ampl_2_95_t_percentil_duplo1(i) = ls_2_95-li_2_95;
2668
2669
2670      if(beta(1)>ls_3_95){
2671          ncobdi_3_95_t_percentil_duplo1(i) = 1;
2672      }else
2673          ncobdi_3_95_t_percentil_duplo1(i) = 0;
2674
2675      ampl_3_95_t_percentil_duplo1(i) = ls_3_95-li_3_95;
2676
2677      if(beta(1)>ls_4_95){
2678          ncobdi_4_95_t_percentil_duplo1(i) = 1;
2679      }else
2680          ncobdi_4_95_t_percentil_duplo1(i) = 0;
2681
2682      ampl_4_95_t_percentil_duplo1(i) = ls_4_95-li_4_95;
2683
2684      if(beta(1)>ls_5_95){
2685          ncobdi_5_95_t_percentil_duplo1(i) = 1;
2686      }else
2687          ncobdi_5_95_t_percentil_duplo1(i) = 0;
2688
2689      ampl_5_95_t_percentil_duplo1(i) = ls_5_95-li_5_95;
2690
2691      // #####
2692      //                                INTERVALOS PARA 99%
2693      // #####
2694
2695      // CONTANDO CONVERGENCIAS PARA O BOOSTRAP T DUPLO. (ESSE ESQUEMA
2696      // DE BOOTSTRAP DUPLO NAO e CORRETO, CONTUDO, ESTOU EM CASOS DE
2697      // HOMOSCEDASTICIDADE. PRECISA SER VERIFICADO.)
2698
2699      double quantil0_inferior99 = myfunctions::quantil1(z0_estrela_duplo,
2700          0.995,nrep_boot);
2701      double quantil0_superior99 = myfunctions::quantil1(z0_estrela_duplo,
2702          0.005,nrep_boot);
2703
2704      double quantil2_inferior99 = myfunctions::quantil1(z2_estrela_duplo,
2705          0.995,nrep_boot);
2706      double quantil2_superior99 = myfunctions::quantil1(z2_estrela_duplo,
2707          0.005,nrep_boot);
2708
2709      double quantil3_inferior99 = myfunctions::quantil1(z3_estrela_duplo,
2710          0.995,nrep_boot);
2711      double quantil3_superior99 = myfunctions::quantil1(z3_estrela_duplo,
2712          0.005,nrep_boot);
2713
2714      double quantil4_inferior99 = myfunctions::quantil1(z4_estrela_duplo,
2715          0.995,nrep_boot);
2716      double quantil4_superior99 = myfunctions::quantil1(z4_estrela_duplo,
2717          0.005,nrep_boot);
2718
2719      double quantil5_inferior99 = myfunctions::quantil1(z5_estrela_duplo,
2720          0.995,nrep_boot);
2721      double quantil5_superior99 = myfunctions::quantil1(z5_estrela_duplo,
2722          0.005,nrep_boot);
2723
2724      if(ncorrecoes==2){

```

```

2725
2726 // AQUI ESTAMOS CORRIGINDO O CALCULO DOS LIMITES INFERIORES
2727 // E SUPERIORES DO INTERVALO DE CONFIANCA. ESSA CORRECAO FAZ
2728 // USO DO BOOTSTRAP EXTERIOR.
2729
2730 li_0_99 = temp(1,0)-quantil0_inferior99*(2*sum(hc0_b)/nrep_boot
2731          -sqrt(HC0(1,1)));
2732 ls_0_99 = temp(1,0)-quantil0_superior99*(2*sum(hc0_b)/nrep_boot
2733          -sqrt(HC0(1,1)));
2734 li_2_99 = temp(1,0)-quantil2_inferior99*(2*sum(hc2_b)/nrep_boot
2735          -sqrt(HC2(1,1)));
2736 ls_2_99 = temp(1,0)-quantil2_superior99*(2*sum(hc2_b)/nrep_boot
2737          -sqrt(HC2(1,1)));
2738 li_3_99 = temp(1,0)-quantil3_inferior99*(2*sum(hc3_b)/nrep_boot
2739          -sqrt(HC3(1,1)));
2740 ls_3_99 = temp(1,0)-quantil3_superior99*(2*sum(hc3_b)/nrep_boot
2741          -sqrt(HC3(1,1)));
2742 li_4_99 = temp(1,0)-quantil4_inferior99*(2*sum(hc4_b)/nrep_boot
2743          -sqrt(HC4(1,1)));
2744 ls_4_99 = temp(1,0)-quantil4_superior99*(2*sum(hc4_b)/nrep_boot
2745          -sqrt(HC4(1,1)));
2746 li_5_99 = temp(1,0)-quantil5_inferior99*(2*sum(hc5_b)/nrep_boot
2747          -sqrt(HC5(1,1)));
2748 ls_5_99 = temp(1,0)-quantil5_superior99*(2*sum(hc5_b)/nrep_boot
2749          -sqrt(HC5(1,1)));
2750 }
2751
2752 if(ncorrecoes==1){
2753
2754 // AQUI ESTAMOS CONSTRUINDO OS LIMITES DOS INTERVALOS DE CONFI-
2755 // ANCAS SEM USAR A CORRECAO DO DESVIO QUE ENTRA NO CALCULO DOS
2756 // LIMITES.
2757
2758 li_0_99 = temp(1,0)-quantil0_inferior99*sqrt(HC0(1,1));
2759 ls_0_99 = temp(1,0)-quantil0_superior99*sqrt(HC0(1,1));
2760 li_2_99 = temp(1,0)-quantil2_inferior99*sqrt(HC2(1,1));
2761 ls_2_99 = temp(1,0)-quantil2_superior99*sqrt(HC2(1,1));
2762 li_3_99 = temp(1,0)-quantil3_inferior99*sqrt(HC3(1,1));
2763 ls_3_99 = temp(1,0)-quantil3_superior99*sqrt(HC3(1,1));
2764 li_4_99 = temp(1,0)-quantil4_inferior99*sqrt(HC4(1,1));
2765 ls_4_99 = temp(1,0)-quantil4_superior99*sqrt(HC4(1,1));
2766 li_5_99 = temp(1,0)-quantil5_inferior99*sqrt(HC5(1,1));
2767 ls_5_99 = temp(1,0)-quantil5_superior99*sqrt(HC5(1,1));
2768 }
2769
2770 if(beta(1)>=li_0_99&&beta(1)<=ls_0_99){
2771     cob_0_99_t_percentil_duplo(i) = 1;
2772 }else
2773     cob_0_99_t_percentil_duplo(i) = 0;
2774
2775 if(beta(1)>=li_2_99&&beta(1)<=ls_2_99){
2776     cob_2_99_t_percentil_duplo(i) = 1;
2777 }else
2778     cob_2_99_t_percentil_duplo(i) = 0;
2779
2780 if(beta(1)>=li_3_99&&beta(1)<=ls_3_99){
2781     cob_3_99_t_percentil_duplo(i) = 1;
2782 }else

```

```

2783         cob_3_99_t_percentil_duplo(i) = 0;
2784
2785     if(beta(1)>=li_4_99&&beta(1)<=ls_4_99){
2786         cob_4_99_t_percentil_duplo(i) = 1;
2787     }else
2788         cob_4_99_t_percentil_duplo(i) = 0;
2789
2790     if(beta(1)>=li_5_99&&beta(1)<=ls_5_99){
2791         cob_5_99_t_percentil_duplo(i) = 1;
2792     }else
2793         cob_5_99_t_percentil_duplo(i) = 0;
2794
2795     if(beta(1)<li_0_99){
2796         ncobesq_0_99_t_percentil_duplo(i) = 1;
2797     }else
2798         ncobesq_0_99_t_percentil_duplo(i) = 0;
2799
2800     if(beta(1)<li_2_99){
2801         ncobesq_2_99_t_percentil_duplo(i) = 1;
2802     }else
2803         ncobesq_2_99_t_percentil_duplo(i) = 0;
2804
2805     if(beta(1)<li_3_99){
2806         ncobesq_3_99_t_percentil_duplo(i) = 1;
2807     }else
2808         ncobesq_3_99_t_percentil_duplo(i) = 0;
2809
2810     if(beta(1)<li_4_99){
2811         ncobesq_4_99_t_percentil_duplo(i) = 1;
2812     }else
2813         ncobesq_4_99_t_percentil_duplo(i) = 0;
2814
2815     if(beta(1)<li_5_99){
2816         ncobesq_5_99_t_percentil_duplo(i) = 1;
2817     }else
2818         ncobesq_5_99_t_percentil_duplo(i) = 0;
2819
2820     if(beta(1)>ls_0_99){
2821         ncobdi_0_99_t_percentil_duplo(i) = 1;
2822     }else
2823         ncobdi_0_99_t_percentil_duplo(i) = 0;
2824
2825     ampl_0_99_t_percentil_duplo(i) = ls_0_99-li_0_99;
2826
2827     if(beta(1)>ls_2_99){
2828         ncobdi_2_99_t_percentil_duplo(i) = 1;
2829     }else
2830         ncobdi_2_99_t_percentil_duplo(i) = 0;
2831
2832     ampl_2_99_t_percentil_duplo(i) = ls_2_99-li_2_99;
2833
2834     if(beta(1)>ls_3_99){
2835         ncobdi_3_99_t_percentil_duplo(i) = 1;
2836     }else
2837         ncobdi_3_99_t_percentil_duplo(i) = 0;
2838
2839     ampl_3_99_t_percentil_duplo(i) = ls_3_99-li_0_99;
2840

```

```

2841         if(beta(1)>ls_4_99){
2842             ncobdi_4_99_t_percentil_duplo(i) = 1;
2843         }else
2844             ncobdi_4_99_t_percentil_duplo(i) = 0;
2845
2846         ampl_4_99_t_percentil_duplo(i) = ls_4_99-li_4_99;
2847
2848         if(beta(1)>ls_5_99){
2849             ncobdi_5_99_t_percentil_duplo(i) = 1;
2850         }else
2851             ncobdi_5_99_t_percentil_duplo(i) = 0;
2852
2853         ampl_5_99_t_percentil_duplo(i) = ls_5_99-li_5_99;
2854
2855         // CONTANDO CONVERGENCIAS PARA O BOOTSTRAP T
2856         // (AQUI NAO E O BOOSTRAP DUPL0.)
2857
2858         quantil0_inferior99 = myfunctions::quantil1(z_estrela0,
2859             0.995,nrep_boot);
2860         quantil0_superior99 = myfunctions::quantil1(z_estrela0,
2861             0.005,nrep_boot);
2862
2863         li_0_99 = temp(1,0)-quantil0_inferior99*sqrt(HC0(1,1));
2864         ls_0_99 = temp(1,0)-quantil0_superior99*sqrt(HC0(1,1));
2865
2866         quantil2_inferior99 = myfunctions::quantil1(z_estrela2,
2867             0.995,nrep_boot);
2868         quantil2_superior99 = myfunctions::quantil1(z_estrela2,
2869             0.005,nrep_boot);
2870
2871         li_2_99 = temp(1,0)-quantil2_inferior99*sqrt(HC2(1,1));
2872         ls_2_99 = temp(1,0)-quantil2_superior99*sqrt(HC2(1,1));
2873
2874         quantil3_inferior99 = myfunctions::quantil1(z_estrela3,
2875             0.995,nrep_boot);
2876         quantil3_superior99 = myfunctions::quantil1(z_estrela3,
2877             0.005,nrep_boot);
2878
2879         li_3_99 = temp(1,0)-quantil3_inferior99*sqrt(HC3(1,1));
2880         ls_3_99 = temp(1,0)-quantil3_superior99*sqrt(HC3(1,1));
2881
2882         quantil4_inferior99 = myfunctions::quantil1(z_estrela4,
2883             0.995,nrep_boot);
2884         quantil4_superior99 = myfunctions::quantil1(z_estrela4,
2885             0.005,nrep_boot);
2886
2887         li_4_99 = temp(1,0)-quantil4_inferior99*sqrt(HC4(1,1));
2888         ls_4_99 = temp(1,0)-quantil4_superior99*sqrt(HC4(1,1));
2889
2890         quantil5_inferior99 = myfunctions::quantil1(z_estrela5,
2891             0.995,nrep_boot);
2892         quantil5_superior99 = myfunctions::quantil1(z_estrela5,
2893             0.005,nrep_boot);
2894
2895         li_5_99 = temp(1,0)-quantil5_inferior99*sqrt(HC5(1,1));
2896         ls_5_99 = temp(1,0)-quantil5_superior99*sqrt(HC5(1,1));
2897
2898         if(beta(1)>=li_0_99 && beta(1)<=ls_0_99){

```

```
2899         cob_0_99_t_percentil(i) = 1;
2900     }else
2901         cob_0_99_t_percentil(i) = 0;
2902
2903     if(beta(1)>=li_2_99 && beta(1)<=ls_2_99){
2904         cob_2_99_t_percentil(i) = 1;
2905     }else
2906         cob_2_99_t_percentil(i) = 0;
2907
2908     if(beta(1)>=li_3_99 && beta(1)<=ls_3_99){
2909         cob_3_99_t_percentil(i) = 1;
2910     }else
2911         cob_3_99_t_percentil(i) = 0;
2912
2913     if(beta(1)>=li_4_99 && beta(1)<=ls_4_99){
2914         cob_4_99_t_percentil(i) = 1;
2915     }else
2916         cob_4_99_t_percentil(i) = 0;
2917
2918     if(beta(1)>=li_5_99 && beta(1)<=ls_5_99){
2919         cob_5_99_t_percentil(i) = 1;
2920     }else
2921         cob_5_99_t_percentil(i) = 0;
2922
2923     if(beta(1)<li_0_99){
2924         ncobesq_0_99_t_percentil(i) = 1;
2925     }else
2926         ncobesq_0_99_t_percentil(i) = 0;
2927
2928     if(beta(1)<li_2_99){
2929         ncobesq_2_99_t_percentil(i) = 1;
2930     }else
2931         ncobesq_2_99_t_percentil(i) = 0;
2932
2933     if(beta(1)<li_3_99){
2934         ncobesq_3_99_t_percentil(i) = 1;
2935     }else
2936         ncobesq_3_99_t_percentil(i) = 0;
2937
2938     if(beta(1)<li_4_99){
2939         ncobesq_4_99_t_percentil(i) = 1;
2940     }else
2941         ncobesq_4_99_t_percentil(i) = 0;
2942
2943     if(beta(1)<li_5_99){
2944         ncobesq_5_99_t_percentil(i) = 1;
2945     }else
2946         ncobesq_5_99_t_percentil(i) = 0;
2947
2948     if(beta(1)>ls_0_99){
2949         ncobdi_0_99_t_percentil(i) = 1;
2950     }else
2951         ncobdi_0_99_t_percentil(i) = 0;
2952
2953     ampl_0_99_t_percentil(i) = ls_0_99-li_0_99;
2954
2955     if(beta(1)>ls_2_99){
2956         ncobdi_2_99_t_percentil(i) = 1;
```



```

2957         }else
2958             ncobdi_2_99_t_percentil(i) = 0;
2959
2960         ampl_2_99_t_percentil(i) = ls_2_99-li_2_99;
2961
2962         if(beta(1)>ls_3_99){
2963             ncobdi_3_99_t_percentil(i) = 1;
2964         }else
2965             ncobdi_3_99_t_percentil(i) = 0;
2966
2967         ampl_3_99_t_percentil(i) = ls_3_99-li_3_99;
2968
2969         if(beta(1)>ls_4_99){
2970             ncobdi_4_99_t_percentil(i) = 1;
2971         }else
2972             ncobdi_4_99_t_percentil(i) = 0;
2973
2974         ampl_4_99_t_percentil(i) = ls_4_99-li_4_99;
2975
2976         if(beta(1)>ls_5_99){
2977             ncobdi_5_99_t_percentil(i) = 1;
2978         }else
2979             ncobdi_5_99_t_percentil(i) = 0;
2980
2981         ampl_5_99_t_percentil(i) = ls_5_99-li_5_99;
2982
2983
2984         // INTERVALO BOOTSTRAP PERCENTIL.
2985         double li99 = myfunctions::quantil1(beta2,0.005,nrep_boot);
2986         double ls99 = myfunctions::quantil1(beta2,0.995,nrep_boot);
2987         if(beta(1)>=li99 && beta(1)<=ls99)
2988             cob99_percentil(i) = 1;
2989         else
2990             cob99_percentil(i) = 0;
2991
2992         ampl99_percentil(i) = ls99-li99;
2993
2994         if(beta(1)<li99)
2995             ncobesq99_percentil(i) = 1;
2996         else
2997             ncobesq99_percentil(i) = 0;
2998
2999         if(beta(1)>ls99)
3000             ncobdi99_percentil(i) = 1;
3001         else
3002             ncobdi99_percentil(i) = 0;
3003
3004         // INTERVALO PERCENTIL BOOTSTRAP DUPL0 - 99% (BOOTSTRAP EXTERIOR).
3005         double hat_ql99 = myfunctions::quantil1(u_estrela,0.005,nrep_boot);
3006         double hat_qu99 = myfunctions::quantil1(u_estrela,0.995,nrep_boot);
3007         ls99 = myfunctions::quantil1(beta2,hat_qu99,nrep_boot);
3008         li99 = myfunctions::quantil1(beta2,hat_ql99,nrep_boot);
3009
3010         // ls99 = temp(1)-myfunctions::quantil1(betaj_estrela_menos_betaj,
3011         // hat_ql99,nrep_boot);
3012         // li99 = temp(1)-myfunctions::quantil1(betaj_estrela_menos_betaj,
3013         // hat_qu99,nrep_boot);
3014

```

```

3015         ampl99_percentil_duplo(i) = ls99-li99;
3016
3017         if(li99<=beta(1) && beta(1)<=ls99)
3018             cob99_percentil_duplo(i) = 1;
3019         else
3020             cob99_percentil_duplo(i) = 0;
3021
3022         if(beta(1)<li99)
3023             ncobesq99_percentil_duplo(i) = 1;
3024         else
3025             ncobesq99_percentil_duplo(i) = 0;
3026
3027         if(beta(1)>ls99)
3028             ncobdi99_percentil_duplo(i) = 1;
3029         else
3030             ncobdi99_percentil_duplo(i) = 0;
3031
3032         // INTERVALO BOORSTRAP T DUPLO (CORRETO). BASEADO NO ALGORITMO
3033         // DAS PAGINAS 84-85 DO ARTIGO: IMPLEMENTING THE DOUBLE BOOTSTRAP,
3034         // MCCULLOUGH AND VINOD, COMPUTATIONAL ECONOMICS, 1998.
3035
3036         quantil0_inferior99 = myfunctions::quantil1(z_estrela0,
3037             myfunctions::quantil1(Z0_j,0.995,
3038                 nrep_boot),nrep_boot);
3039         quantil0_superior99 = myfunctions::quantil1(z_estrela0,
3040             myfunctions::quantil1(Z0_j,0.005,
3041                 nrep_boot),nrep_boot);
3042
3043         li_0_99 = temp(1,0)-quantil0_inferior99*sqrt(HC0(1,1));
3044         ls_0_99 = temp(1,0)-quantil0_superior99*sqrt(HC0(1,1));
3045
3046         quantil2_inferior99 = myfunctions::quantil1(z_estrela2,
3047             myfunctions::quantil1(Z2_j,0.995,
3048                 nrep_boot),nrep_boot);
3049         quantil2_superior99 = myfunctions::quantil1(z_estrela2,
3050             myfunctions::quantil1(Z2_j,
3051                 0.005,nrep_boot),nrep_boot);
3052
3053         li_2_99 = temp(1,0)-quantil2_inferior99*sqrt(HC2(1,1));
3054         ls_2_99 = temp(1,0)-quantil2_superior99*sqrt(HC2(1,1));
3055
3056         quantil3_inferior99 = myfunctions::quantil1(z_estrela3,
3057             myfunctions::quantil1(Z3_j,0.995,
3058                 nrep_boot),nrep_boot);
3059         quantil3_superior99 = myfunctions::quantil1(z_estrela3,
3060             myfunctions::quantil1(Z3_j,0.005,
3061                 nrep_boot),nrep_boot);
3062
3063         li_3_99 = temp(1,0)-quantil3_inferior99*sqrt(HC3(1,1));
3064         ls_3_99 = temp(1,0)-quantil3_superior99*sqrt(HC3(1,1));
3065
3066         quantil4_inferior99 = myfunctions::quantil1(z_estrela4,
3067             myfunctions::quantil1(Z4_j,0.995,
3068                 nrep_boot),nrep_boot);
3069         quantil4_superior99 = myfunctions::quantil1(z_estrela4,
3070             myfunctions::quantil1(Z4_j,
3071                 0.005,nrep_boot),nrep_boot);
3072

```

```

3073     li_4_99 = temp(1,0)-quantil4_inferior99*sqrt(HC4(1,1));
3074     ls_4_99 = temp(1,0)-quantil4_superior99*sqrt(HC4(1,1));
3075
3076     quantil5_inferior99 = myfunctions::quantil1(z_estrela5,
3077                                           myfunctions::quantil1(Z5_j,
3078                                           0.995,nrep_boot),nrep_boot);
3079
3080     quantil5_superior99 = myfunctions::quantil1(z_estrela5,
3081                                           myfunctions::quantil1(Z5_j,
3082                                           0.005,nrep_boot),nrep_boot);
3083
3084     li_5_99 = temp(1,0)-quantil5_inferior99*sqrt(HC5(1,1));
3085     ls_5_99 = temp(1,0)-quantil5_superior99*sqrt(HC5(1,1));
3086
3087     if(beta(1)>=li_0_99 && beta(1)<=ls_0_99){
3088         cob_0_99_t_percentil_duplo1(i) = 1;
3089     }else
3090         cob_0_99_t_percentil_duplo1(i) = 0;
3091
3092     if(beta(1)>=li_2_99 && beta(1)<=ls_2_99){
3093         cob_2_99_t_percentil_duplo1(i) = 1;
3094     }else
3095         cob_2_99_t_percentil_duplo1(i) = 0;
3096
3097     if(beta(1)>=li_3_99 && beta(1)<=ls_3_99){
3098         cob_3_99_t_percentil_duplo1(i) = 1;
3099     }else
3100         cob_3_99_t_percentil_duplo1(i) = 0;
3101
3102     if(beta(1)>=li_4_99 && beta(1)<=ls_4_99){
3103         cob_4_99_t_percentil_duplo1(i) = 1;
3104     }else
3105         cob_4_99_t_percentil_duplo1(i) = 0;
3106
3107     if(beta(1)>=li_5_99 && beta(1)<=ls_5_99){
3108         cob_5_99_t_percentil_duplo1(i) = 1;
3109     }else
3110         cob_5_99_t_percentil_duplo1(i) = 0;
3111
3112     if(beta(1)<li_0_99){
3113         ncobesq_0_99_t_percentil_duplo1(i) = 1;
3114     }else
3115         ncobesq_0_99_t_percentil_duplo1(i) = 0;
3116
3117     if(beta(1)<li_2_99){
3118         ncobesq_2_99_t_percentil_duplo1(i) = 1;
3119     }else
3120         ncobesq_2_99_t_percentil_duplo1(i) = 0;
3121
3122     if(beta(1)<li_3_99){
3123         ncobesq_3_99_t_percentil_duplo1(i) = 1;
3124     }else
3125         ncobesq_3_99_t_percentil_duplo1(i) = 0;
3126
3127     if(beta(1)<li_4_99){
3128         ncobesq_4_99_t_percentil_duplo1(i) = 1;
3129     }else
3130         ncobesq_4_99_t_percentil_duplo1(i) = 0;

```

```

3131
3132         if(beta(1)<li_5_99){
3133             ncobesq_5_99_t_percentil_duplo1(i) = 1;
3134         }else
3135             ncobesq_5_99_t_percentil_duplo1(i) = 0;
3136
3137         if(beta(1)>ls_0_99){
3138             ncobdi_0_99_t_percentil_duplo1(i) = 1;
3139         }else
3140             ncobdi_0_99_t_percentil_duplo1(i) = 0;
3141
3142         ampl_0_99_t_percentil_duplo1(i) = ls_0_99-li_0_99;
3143
3144         if(beta(1)>ls_2_99){
3145             ncobdi_2_99_t_percentil_duplo1(i) = 1;
3146         }else
3147             ncobdi_2_99_t_percentil_duplo1(i) = 0;
3148
3149         ampl_2_99_t_percentil_duplo1(i) = ls_2_99-li_2_99;
3150
3151         if(beta(1)>ls_3_99){
3152             ncobdi_3_99_t_percentil_duplo1(i) = 1;
3153         }else
3154             ncobdi_3_99_t_percentil_duplo1(i) = 0;
3155
3156         ampl_3_99_t_percentil_duplo1(i) = ls_3_99-li_3_99;
3157
3158         if(beta(1)>ls_4_99){
3159             ncobdi_4_99_t_percentil_duplo1(i) = 1;
3160         }else
3161             ncobdi_4_99_t_percentil_duplo1(i) = 0;
3162
3163         ampl_4_99_t_percentil_duplo1(i) = ls_4_99-li_4_99;
3164
3165         if(beta(1)>ls_5_99){
3166             ncobdi_5_99_t_percentil_duplo1(i) = 1;
3167         }else
3168             ncobdi_5_99_t_percentil_duplo1(i) = 0;
3169
3170         ampl_5_99_t_percentil_duplo1(i) = ls_5_99-li_5_99;
3171 } // AQUI TERMINA O LACO MONTE CARLO
3172
3173 time_t rawtime_1;
3174 time (&rawtime_1);
3175 timeinfo = localtime (&rawtime_1);
3176 saida << ">> [*] Horário de termino da simulacao: " << asctime(timeinfo) << endl;
3177 saida << "(*) TEMPO DE EXECUCAO: " << float(clock() -
3178     tempo_inicial)/CLOCKS_PER_SEC << " segundos / " <<
3179     (float(clock() - tempo_inicial)/CLOCKS_PER_SEC)/60 << " minutos / " <<
3180     ((float(clock() - tempo_inicial)/CLOCKS_PER_SEC)/60)/60 << " horas / " <<
3181     (((float(clock() - tempo_inicial)/CLOCKS_PER_SEC)/60)/60)/24 << " dias."
3182     << endl << endl;
3183
3184 saida << "-----" << endl;
3185 saida << "                INTERVALOS SEM BOOTSTRAP                " << endl;
3186 saida << "-----" << "\n"
3187     << endl;
3188 saida << "..... INTERVALO T" << endl;

```

```

3189   saida << "..... NIVEL DE CONFIANCA: 90%" << endl;
3190   saida << "COBERTURA:      "
3191       << "OLS = " << (sum(cob90_t_ols)/nrep)*100 << ", "
3192       << "HC0 = " << (sum(cob90_t_hc0)/nrep)*100 << ", "
3193       << "HC2 = " << (sum(cob90_t_hc2)/nrep)*100 << ", "
3194       << "HC3 = " << (sum(cob90_t_hc3)/nrep)*100 << ", "
3195       << "HC4 = " << (sum(cob90_t_hc4)/nrep)*100 << ", "
3196       << "HC5 = " << (sum(cob90_t_hc5)/nrep)*100 << endl;
3197   saida << "AMPLITUDE:      "
3198       << "OLS = " << sum(ampl90_t_ols)/nrep << ", "
3199       << "HC0 = " << sum(ampl90_t_hc0)/nrep << ", "
3200       << "HC2 = " << sum(ampl90_t_hc2)/nrep << ", "
3201       << "HC3 = " << sum(ampl90_t_hc3)/nrep << ", "
3202       << "HC4 = " << sum(ampl90_t_hc4)/nrep << ", "
3203       << "HC5 = " << sum(ampl90_t_hc5)/nrep << endl;
3204   saida << "NAO COB. ESQ.:  "
3205       << "OLS = " << (sum(ncobesq90_t_ols)/nrep)*100 << ", "
3206       << "HC0 = " << (sum(ncobesq90_t_hc0)/nrep)*100 << ", "
3207       << "HC2 = " << (sum(ncobesq90_t_hc2)/nrep)*100 << ", "
3208       << "HC3 = " << (sum(ncobesq90_t_hc3)/nrep)*100 << ", "
3209       << "HC4 = " << (sum(ncobesq90_t_hc4)/nrep)*100 << ", "
3210       << "HC5 = " << (sum(ncobesq90_t_hc5)/nrep)*100 << endl;
3211   saida << "NAO COB. DIR.:  "
3212       << "OLS = " << (sum(ncobdi90_t_ols)/nrep)*100 << ", "
3213       << "HC0 = " << (sum(ncobdi90_t_hc0)/nrep)*100 << ", "
3214       << "HC2 = " << (sum(ncobdi90_t_hc2)/nrep)*100 << ", "
3215       << "HC3 = " << (sum(ncobdi90_t_hc3)/nrep)*100 << ", "
3216       << "HC4 = " << (sum(ncobdi90_t_hc4)/nrep)*100 << ", "
3217       << "HC5 = " << (sum(ncobdi90_t_hc5)/nrep)*100 << "\n" << endl;
3218
3219   saida << "..... INTERVALO T" << endl;
3220   saida << "..... NIVEL DE CONFIANCA: 95%" << endl;
3221   saida << "COBERTURA:      "
3222       << "OLS = " << (sum(cob95_t_ols)/nrep)*100 << ", "
3223       << "HC0 = " << (sum(cob95_t_hc0)/nrep)*100 << ", "
3224       << "HC2 = " << (sum(cob95_t_hc2)/nrep)*100 << ", "
3225       << "HC3 = " << (sum(cob95_t_hc3)/nrep)*100 << ", "
3226       << "HC4 = " << (sum(cob95_t_hc4)/nrep)*100 << ", "
3227       << "HC5 = " << (sum(cob95_t_hc5)/nrep)*100 << endl;
3228   saida << "AMPLITUDE:      "
3229       << "OLS = " << sum(ampl95_t_ols)/nrep << ", "
3230       << "HC0 = " << sum(ampl95_t_hc0)/nrep << ", "
3231       << "HC2 = " << sum(ampl95_t_hc2)/nrep << ", "
3232       << "HC3 = " << sum(ampl95_t_hc3)/nrep << ", "
3233       << "HC4 = " << sum(ampl95_t_hc4)/nrep << ", "
3234       << "HC5 = " << sum(ampl95_t_hc5)/nrep << endl;
3235   saida << "NAO COB. ESQ.:  "
3236       << "OLS = " << (sum(ncobesq95_t_ols)/nrep)*100 << ", "
3237       << "HC0 = " << (sum(ncobesq95_t_hc0)/nrep)*100 << ", "
3238       << "HC2 = " << (sum(ncobesq95_t_hc2)/nrep)*100 << ", "
3239       << "HC3 = " << (sum(ncobesq95_t_hc3)/nrep)*100 << ", "
3240       << "HC4 = " << (sum(ncobesq95_t_hc4)/nrep)*100 << ", "
3241       << "HC5 = " << (sum(ncobesq95_t_hc5)/nrep)*100 << endl;
3242   saida << "NAO COB. DIR.:  "
3243       << "OLS = " << (sum(ncobdi95_t_ols)/nrep)*100 << ", "
3244       << "HC0 = " << (sum(ncobdi95_t_hc0)/nrep)*100 << ", "
3245       << "HC2 = " << (sum(ncobdi95_t_hc2)/nrep)*100 << ", "
3246       << "HC3 = " << (sum(ncobdi95_t_hc3)/nrep)*100 << ", "

```

```

3247         << "HC4 = " << (sum(ncobdi95_t_hc4)/nrep)*100 << ", "
3248         << "HC5 = " << (sum(ncobdi95_t_hc5)/nrep)*100 << "\n" << endl;
3249
3250     saida << "..... INTERVALO T" << endl;
3251     saida << "..... NIVEL DE CONFIANCA: 99%" << endl;
3252     saida << "COBERTURA:      "
3253         << "OLS = " << (sum(cob99_t_ols)/nrep)*100 << ", "
3254         << "HC0 = " << (sum(cob99_t_hc0)/nrep)*100 << ", "
3255         << "HC2 = " << (sum(cob99_t_hc2)/nrep)*100 << ", "
3256         << "HC3 = " << (sum(cob99_t_hc3)/nrep)*100 << ", "
3257         << "HC4 = " << (sum(cob99_t_hc4)/nrep)*100 << ", "
3258         << "HC5 = " << (sum(cob99_t_hc5)/nrep)*100 << endl;
3259     saida << "AMPLITUDE:      "
3260         << "OLS = " << sum(ampl99_t_ols)/nrep << ", "
3261         << "HC0 = " << sum(ampl99_t_hc0)/nrep << ", "
3262         << "HC2 = " << sum(ampl99_t_hc2)/nrep << ", "
3263         << "HC3 = " << sum(ampl99_t_hc3)/nrep << ", "
3264         << "HC4 = " << sum(ampl99_t_hc4)/nrep << ", "
3265         << "HC5 = " << sum(ampl99_t_hc5)/nrep << endl;
3266     saida << "NAO COB. ESQ.:  "
3267         << "OLS = " << (sum(ncobesq99_t_ols)/nrep)*100 << ", "
3268         << "HC0 = " << (sum(ncobesq99_t_hc0)/nrep)*100 << ", "
3269         << "HC2 = " << (sum(ncobesq99_t_hc2)/nrep)*100 << ", "
3270         << "HC3 = " << (sum(ncobesq99_t_hc3)/nrep)*100 << ", "
3271         << "HC4 = " << (sum(ncobesq99_t_hc4)/nrep)*100 << ", "
3272         << "HC5 = " << (sum(ncobesq99_t_hc5)/nrep)*100 << endl;
3273     saida << "NAO COB. DIR.:  "
3274         << "OLS = " << (sum(ncobdi99_t_ols)/nrep)*100 << ", "
3275         << "HC0 = " << (sum(ncobdi99_t_hc0)/nrep)*100 << ", "
3276         << "HC2 = " << (sum(ncobdi99_t_hc2)/nrep)*100 << ", "
3277         << "HC3 = " << (sum(ncobdi99_t_hc3)/nrep)*100 << ", "
3278         << "HC4 = " << (sum(ncobdi99_t_hc4)/nrep)*100 << ", "
3279         << "HC5 = " << (sum(ncobdi99_t_hc5)/nrep)*100 << "\n" << endl;
3280
3281     saida << "..... INTERVALO Z" << endl;
3282     saida << "..... NIVEL DE CONFIANCA: 90%" << endl;
3283     saida << "COBERTURA:      "
3284         << "OLS = " << (sum(cob90_z_ols)/nrep)*100 << ", "
3285         << "HC0 = " << (sum(cob90_z_hc0)/nrep)*100 << ", "
3286         << "HC2 = " << (sum(cob90_z_hc2)/nrep)*100 << ", "
3287         << "HC3 = " << (sum(cob90_z_hc3)/nrep)*100 << ", "
3288         << "HC4 = " << (sum(cob90_z_hc4)/nrep)*100 << ", "
3289         << "HC5 = " << (sum(cob90_z_hc5)/nrep)*100 << endl;
3290     saida << "AMPLITUDE:      "
3291         << "OLS = " << sum(ampl90_z_ols)/nrep << ", "
3292         << "HC0 = " << sum(ampl90_z_hc0)/nrep << ", "
3293         << "HC2 = " << sum(ampl90_z_hc2)/nrep << ", "
3294         << "HC3 = " << sum(ampl90_z_hc3)/nrep << ", "
3295         << "HC4 = " << sum(ampl90_z_hc4)/nrep << ", "
3296         << "HC5 = " << sum(ampl90_z_hc5)/nrep << endl;
3297     saida << "NAO COB. ESQ.:  "
3298         << "OLS = " << (sum(ncobesq90_z_ols)/nrep)*100 << ", "
3299         << "HC0 = " << (sum(ncobesq90_z_hc0)/nrep)*100 << ", "
3300         << "HC2 = " << (sum(ncobesq90_z_hc2)/nrep)*100 << ", "
3301         << "HC3 = " << (sum(ncobesq90_z_hc3)/nrep)*100 << ", "
3302         << "HC4 = " << (sum(ncobesq90_z_hc4)/nrep)*100 << ", "
3303         << "HC5 = " << (sum(ncobesq90_z_hc5)/nrep)*100 << endl;
3304     saida << "NAO COB. DIR.:  "

```

```

3305         << "OLS = " << (sum(ncobdi90_z_ols)/nrep)*100 << ", "
3306         << "HC0 = " << (sum(ncobdi90_z_hc0)/nrep)*100 << ", "
3307         << "HC2 = " << (sum(ncobdi90_z_hc2)/nrep)*100 << ", "
3308         << "HC3 = " << (sum(ncobdi90_z_hc3)/nrep)*100 << ", "
3309         << "HC4 = " << (sum(ncobdi90_z_hc4)/nrep)*100 << ", "
3310         << "HC5 = " << (sum(ncobdi90_z_hc5)/nrep)*100 << "\n" << endl;
3311
3312 saida << "..... INTERVALO Z" << endl;
3313 saida << "..... NIVEL DE CONFIANCA: 95%" << endl;
3314 saida << "COBERTURA: "
3315         << "OLS = " << (sum(cob95_z_ols)/nrep)*100 << ", "
3316         << "HC0 = " << (sum(cob95_z_hc0)/nrep)*100 << ", "
3317         << "HC2 = " << (sum(cob95_z_hc2)/nrep)*100 << ", "
3318         << "HC3 = " << (sum(cob95_z_hc3)/nrep)*100 << ", "
3319         << "HC4 = " << (sum(cob95_z_hc4)/nrep)*100 << ", "
3320         << "HC5 = " << (sum(cob95_z_hc5)/nrep)*100 << endl;
3321 saida << "AMPLITUDE: "
3322         << "OLS = " << sum(aml95_z_ols)/nrep << ", "
3323         << "HC0 = " << sum(aml95_z_hc0)/nrep << ", "
3324         << "HC2 = " << sum(aml95_z_hc2)/nrep << ", "
3325         << "HC3 = " << sum(aml95_z_hc3)/nrep << ", "
3326         << "HC4 = " << sum(aml95_z_hc4)/nrep << ", "
3327         << "HC5 = " << sum(aml95_z_hc5)/nrep << endl;
3328 saida << "NAO COB. ESQ.: "
3329         << "OLS = " << (sum(ncobesq95_z_ols)/nrep)*100 << ", "
3330         << "HC0 = " << (sum(ncobesq95_z_hc0)/nrep)*100 << ", "
3331         << "HC2 = " << (sum(ncobesq95_z_hc2)/nrep)*100 << ", "
3332         << "HC3 = " << (sum(ncobesq95_z_hc3)/nrep)*100 << ", "
3333         << "HC4 = " << (sum(ncobesq95_z_hc4)/nrep)*100 << ", "
3334         << "HC5 = " << (sum(ncobesq95_z_hc5)/nrep)*100 << endl;
3335 saida << "NAO COB. DIR.: "
3336         << "OLS = " << (sum(ncobdi95_z_ols)/nrep)*100 << ", "
3337         << "HC0 = " << (sum(ncobdi95_z_hc0)/nrep)*100 << ", "
3338         << "HC2 = " << (sum(ncobdi95_z_hc2)/nrep)*100 << ", "
3339         << "HC3 = " << (sum(ncobdi95_z_hc3)/nrep)*100 << ", "
3340         << "HC4 = " << (sum(ncobdi95_z_hc4)/nrep)*100 << ", "
3341         << "HC5 = " << (sum(ncobdi95_z_hc5)/nrep)*100 << "\n" << endl;
3342
3343 saida << "..... INTERVALO Z" << endl;
3344 saida << "..... NIVEL DE CONFIANCA: 99%" << endl;
3345 saida << "COBERTURA: "
3346         << "OLS = " << (sum(cob99_z_ols)/nrep)*100 << ", "
3347         << "HC0 = " << (sum(cob99_z_hc0)/nrep)*100 << ", "
3348         << "HC2 = " << (sum(cob99_z_hc2)/nrep)*100 << ", "
3349         << "HC3 = " << (sum(cob99_z_hc3)/nrep)*100 << ", "
3350         << "HC4 = " << (sum(cob99_z_hc4)/nrep)*100 << ", "
3351         << "HC5 = " << (sum(cob99_z_hc5)/nrep)*100 << endl;
3352 saida << "AMPLITUDE: "
3353         << "OLS = " << sum(aml99_z_ols)/nrep << ", "
3354         << "HC0 = " << sum(aml99_z_hc0)/nrep << ", "
3355         << "HC2 = " << sum(aml99_z_hc2)/nrep << ", "
3356         << "HC3 = " << sum(aml99_z_hc3)/nrep << ", "
3357         << "HC4 = " << sum(aml99_z_hc4)/nrep << ", "
3358         << "HC5 = " << sum(aml99_z_hc5)/nrep << endl;
3359 saida << "NAO COB. ESQ.: "
3360         << "OLS = " << (sum(ncobesq99_z_ols)/nrep)*100 << ", "
3361         << "HC0 = " << (sum(ncobesq99_z_hc0)/nrep)*100 << ", "
3362         << "HC2 = " << (sum(ncobesq99_z_hc2)/nrep)*100 << ", "

```

```

3363         << "HC3 = " << (sum(ncobesq99_z_hc3)/nrep)*100 << ", "
3364         << "HC4 = " << (sum(ncobesq99_z_hc4)/nrep)*100 << ", "
3365         << "HC5 = " << (sum(ncobesq99_z_hc5)/nrep)*100 << endl;
3366 saida << "NAO COB. DIR.: "
3367         << "OLS = " << (sum(ncobdi99_z_ols)/nrep)*100 << ", "
3368         << "HC0 = " << (sum(ncobdi99_z_hc0)/nrep)*100 << ", "
3369         << "HC2 = " << (sum(ncobdi99_z_hc2)/nrep)*100 << ", "
3370         << "HC3 = " << (sum(ncobdi99_z_hc3)/nrep)*100 << ", "
3371         << "HC4 = " << (sum(ncobdi99_z_hc4)/nrep)*100 << ", "
3372         << "HC5 = " << (sum(ncobdi99_z_hc5)/nrep)*100 << "\n" << endl;
3373
3374 saida << "-----" << endl;
3375 saida << "                INTERVALOS USANDO BOOTSTRAP                " << endl;
3376 saida << "-----" << "\n"
3377         << endl;
3378 saida << "..... BOOTSTRAP PERCENTIL" << endl;
3379 saida << "..... NIVEL DE CONFIANCA: 90%" << "\n" << endl;
3380 saida << "COBERTURA:      " << (sum(cob90_percentil)/nrep)*100 << endl;
3381 saida << "APLITUDE:        " << (sum(ampl90_percentil)/nrep) << endl;
3382 saida << "NAO COB. ESQ.:    " << (sum(ncobesq90_percentil)/nrep)*100 << endl;
3383 saida << "NAO COB. DIR.:    " << (sum(ncobdi90_percentil)/nrep)*100 << "\n"
3384         << endl;
3385
3386 saida << "..... BOOTSTRAP PERCENTIL DUPL0" << endl;
3387 saida << "..... NIVEL DE CONFIANCA: 90%" << "\n" << endl;
3388 saida << "COBERTURA:      " << (sum(cob90_percentil_duplo)/nrep)*100 << endl;
3389 saida << "APLITUDE:        " << (sum(ampl90_percentil_duplo)/nrep) << endl;
3390 saida << "NAO COB. ESQ.:    " << (sum(ncobesq90_percentil_duplo)/nrep)*100 << endl;
3391 saida << "NAO COB. DIR.:    " << (sum(ncobdi90_percentil_duplo)/nrep)*100 << "\n"
3392         << endl;
3393
3394 saida << "..... BOOTSTRAP T" << endl;
3395 saida << "..... NIVEL DE CONFIANCA: 90%" << "\n" << endl;
3396 saida << "COBERTURA:      "
3397         << "HC0 = " << (sum(cob_0_90_t_percentil)/nrep)*100 << ", "
3398         << "HC2 = " << (sum(cob_2_90_t_percentil)/nrep)*100 << ", "
3399         << "HC3 = " << (sum(cob_3_90_t_percentil)/nrep)*100 << ", "
3400         << "HC4 = " << (sum(cob_4_90_t_percentil)/nrep)*100 << ", "
3401         << "HC5 = " << (sum(cob_5_90_t_percentil)/nrep)*100 << endl;
3402 saida << "APLITUDE:      "
3403         << "HC0 = " << (sum(ampl_0_90_t_percentil)/nrep) << ", "
3404         << "HC2 = " << (sum(ampl_2_90_t_percentil)/nrep) << ", "
3405         << "HC3 = " << (sum(ampl_3_90_t_percentil)/nrep) << ", "
3406         << "HC4 = " << (sum(ampl_4_90_t_percentil)/nrep) << ", "
3407         << "HC5 = " << (sum(ampl_5_90_t_percentil)/nrep) << endl;
3408 saida << "NAO COB. ESQ.:    "
3409         << "HC0 = " << (sum(ncobesq_0_90_t_percentil)/nrep)*100 << ", "
3410         << "HC2 = " << (sum(ncobesq_2_90_t_percentil)/nrep)*100 << ", "
3411         << "HC3 = " << (sum(ncobesq_3_90_t_percentil)/nrep)*100 << ", "
3412         << "HC4 = " << (sum(ncobesq_4_90_t_percentil)/nrep)*100 << ", "
3413         << "HC5 = " << (sum(ncobesq_5_90_t_percentil)/nrep)*100 << endl;
3414 saida << "NAO COB. DIR.:    "
3415         << "HC0 = " << (sum(ncobdi_0_90_t_percentil)/nrep)*100 << ", "
3416         << "HC2 = " << (sum(ncobdi_2_90_t_percentil)/nrep)*100 << ", "
3417         << "HC3 = " << (sum(ncobdi_3_90_t_percentil)/nrep)*100 << ", "
3418         << "HC4 = " << (sum(ncobdi_4_90_t_percentil)/nrep)*100 << ", "
3419         << "HC5 = " << (sum(ncobdi_5_90_t_percentil)/nrep)*100 << "\n" << endl;
3420

```



```

3421      saida << "..... BOOTSTRAP T DUPLO" << endl;
3422      saida << "..... NIVEL DE CONFIANCA: 90%" << "\n" << endl;
3423      saida << "COBERTURA:      "
3424          << "HC0 = " << (sum(cob_0_90_t_percentil_duplo1)/nrep)*100 << ", "
3425          << "HC2 = " << (sum(cob_2_90_t_percentil_duplo1)/nrep)*100 << ", "
3426          << "HC3 = " << (sum(cob_3_90_t_percentil_duplo1)/nrep)*100 << ", "
3427          << "HC4 = " << (sum(cob_4_90_t_percentil_duplo1)/nrep)*100 << ", "
3428          << "HC5 = " << (sum(cob_5_90_t_percentil_duplo1)/nrep)*100 << endl;
3429      saida << "APLITUDE:      "
3430          << "HC0 = " << (sum(ampl_0_90_t_percentil_duplo1)/nrep) << ", "
3431          << "HC2 = " << (sum(ampl_2_90_t_percentil_duplo1)/nrep) << ", "
3432          << "HC3 = " << (sum(ampl_3_90_t_percentil_duplo1)/nrep) << ", "
3433          << "HC4 = " << (sum(ampl_4_90_t_percentil_duplo1)/nrep) << ", "
3434          << "HC5 = " << (sum(ampl_5_90_t_percentil_duplo1)/nrep) << endl;
3435      saida << "NAO COB. ESQ.:  "
3436          << "HC0 = " << (sum(ncobesq_0_90_t_percentil_duplo1)/nrep)*100 << ", "
3437          << "HC2 = " << (sum(ncobesq_2_90_t_percentil_duplo1)/nrep)*100 << ", "
3438          << "HC3 = " << (sum(ncobesq_3_90_t_percentil_duplo1)/nrep)*100 << ", "
3439          << "HC4 = " << (sum(ncobesq_4_90_t_percentil_duplo1)/nrep)*100 << ", "
3440          << "HC5 = " << (sum(ncobesq_5_90_t_percentil_duplo1)/nrep)*100 << endl;
3441      saida << "NAO COB. DIR.:  "
3442          << "HC0 = " << (sum(ncobdi_0_90_t_percentil_duplo1)/nrep)*100 << ", "
3443          << "HC2 = " << (sum(ncobdi_2_90_t_percentil_duplo1)/nrep)*100 << ", "
3444          << "HC3 = " << (sum(ncobdi_3_90_t_percentil_duplo1)/nrep)*100 << ", "
3445          << "HC4 = " << (sum(ncobdi_4_90_t_percentil_duplo1)/nrep)*100 << ", "
3446          << "HC4 = " << (sum(ncobdi_5_90_t_percentil_duplo1)/nrep)*100 << "\n"
3447          << endl;
3448
3449      saida << "..... BOOTSTRAP T DUPLO (ESQUEMA ERRADO)" << endl;
3450      saida << "..... NIVEL DE CONFIANCA: 90%" << "\n" << endl;
3451      saida << "COBERTURA:      "
3452          << "HC0 = " << (sum(cob_0_90_t_percentil_duplo)/nrep)*100 << ", "
3453          << "HC2 = " << (sum(cob_2_90_t_percentil_duplo)/nrep)*100 << ", "
3454          << "HC3 = " << (sum(cob_3_90_t_percentil_duplo)/nrep)*100 << ", "
3455          << "HC4 = " << (sum(cob_4_90_t_percentil_duplo)/nrep)*100 << ", "
3456          << "HC5 = " << (sum(cob_5_90_t_percentil_duplo)/nrep)*100 << endl;
3457      saida << "APLITUDE:      "
3458          << "HC0 = " << (sum(ampl_0_90_t_percentil_duplo)/nrep) << ", "
3459          << "HC2 = " << (sum(ampl_2_90_t_percentil_duplo)/nrep) << ", "
3460          << "HC3 = " << (sum(ampl_3_90_t_percentil_duplo)/nrep) << ", "
3461          << "HC4 = " << (sum(ampl_4_90_t_percentil_duplo)/nrep) << ", "
3462          << "HC5 = " << (sum(ampl_5_90_t_percentil_duplo)/nrep) << endl;
3463      saida << "NAO COB. ESQ.:  "
3464          << "HC0 = " << (sum(ncobesq_0_90_t_percentil_duplo)/nrep)*100 << ", "
3465          << "HC2 = " << (sum(ncobesq_2_90_t_percentil_duplo)/nrep)*100 << ", "
3466          << "HC3 = " << (sum(ncobesq_3_90_t_percentil_duplo)/nrep)*100 << ", "
3467          << "HC4 = " << (sum(ncobesq_4_90_t_percentil_duplo)/nrep)*100 << ", "
3468          << "HC5 = " << (sum(ncobesq_5_90_t_percentil_duplo)/nrep)*100 << endl;
3469      saida << "NAO COB. DIR.:  "
3470          << "HC0 = " << (sum(ncobdi_0_90_t_percentil_duplo)/nrep)*100 << ", "
3471          << "HC2 = " << (sum(ncobdi_2_90_t_percentil_duplo)/nrep)*100 << ", "
3472          << "HC3 = " << (sum(ncobdi_3_90_t_percentil_duplo)/nrep)*100 << ", "
3473          << "HC4 = " << (sum(ncobdi_4_90_t_percentil_duplo)/nrep)*100 << ", "
3474          << "HC5 = " << (sum(ncobdi_5_90_t_percentil_duplo)/nrep)*100 << "\n"
3475          << endl;
3476
3477      saida << "..... BOOTSTRAP PERCENTIL" << endl;
3478      saida << "..... NIVEL DE CONFIANCA: 95%" << "\n" << endl;

```

```

3479      saida << "COBERTURA:      " << (sum(cob95_percentil)/nrep)*100 << endl;
3480      saida << "APLITUDE:      " << (sum(aml95_percentil)/nrep) << endl;
3481      saida << "NAO COB. ESQ.:  " << (sum(ncobesq95_percentil)/nrep)*100 << endl;
3482      saida << "NAO COB. DIR.:  " << (sum(ncobdi95_percentil)/nrep)*100 << "\n"
3483      << endl;
3484
3485      saida << "..... BOOTSTRAP PERCENTIL DUPL0" << endl;
3486      saida << "..... NIVEL DE CONFIANCA: 95%" << "\n" << endl;
3487      saida << "COBERTURA:      " << (sum(cob95_percentil_duplo)/nrep)*100 << endl;
3488      saida << "APLITUDE:      " << (sum(aml95_percentil_duplo)/nrep) << endl;
3489      saida << "NAO COB. ESQ.:  " << (sum(ncobesq95_percentil_duplo)/nrep)*100
3490      << endl;
3491      saida << "NAO COB. DIR.:  " << (sum(ncobdi95_percentil_duplo)/nrep)*100 << "\n"
3492      << endl;
3493
3494      saida << "..... BOOTSTRAP T" << endl;
3495      saida << "..... NIVEL DE CONFIANCA: 95%" << "\n" << endl;
3496      saida << "COBERTURA:      "
3497      << "HC0 = " << (sum(cob_0_95_t_percentil)/nrep)*100 << ", "
3498      << "HC2 = " << (sum(cob_2_95_t_percentil)/nrep)*100 << ", "
3499      << "HC3 = " << (sum(cob_3_95_t_percentil)/nrep)*100 << ", "
3500      << "HC4 = " << (sum(cob_4_95_t_percentil)/nrep)*100 << ", "
3501      << "HC5 = " << (sum(cob_5_95_t_percentil)/nrep)*100 << endl;
3502      saida << "APLITUDE:      "
3503      << "HC0 = " << (sum(aml_0_95_t_percentil)/nrep) << ", "
3504      << "HC2 = " << (sum(aml_2_95_t_percentil)/nrep) << ", "
3505      << "HC3 = " << (sum(aml_3_95_t_percentil)/nrep) << ", "
3506      << "HC4 = " << (sum(aml_4_95_t_percentil)/nrep) << ", "
3507      << "HC5 = " << (sum(aml_5_95_t_percentil)/nrep) << endl;
3508      saida << "NAO COB. ESQ.:  "
3509      << "HC0 = " << (sum(ncobesq_0_95_t_percentil)/nrep)*100 << ", "
3510      << "HC2 = " << (sum(ncobesq_2_95_t_percentil)/nrep)*100 << ", "
3511      << "HC3 = " << (sum(ncobesq_3_95_t_percentil)/nrep)*100 << ", "
3512      << "HC4 = " << (sum(ncobesq_4_95_t_percentil)/nrep)*100 << ", "
3513      << "HC5 = " << (sum(ncobesq_5_95_t_percentil)/nrep)*100 << endl;
3514      saida << "NAO COB. DIR.:  "
3515      << "HC0 = " << (sum(ncobdi_0_95_t_percentil)/nrep)*100 << ", "
3516      << "HC2 = " << (sum(ncobdi_2_95_t_percentil)/nrep)*100 << ", "
3517      << "HC3 = " << (sum(ncobdi_3_95_t_percentil)/nrep)*100 << ", "
3518      << "HC4 = " << (sum(ncobdi_4_95_t_percentil)/nrep)*100 << ", "
3519      << "HC5 = " << (sum(ncobdi_5_95_t_percentil)/nrep)*100 << "\n"
3520      << endl;
3521
3522      saida << "..... BOOTSTRAP T DUPL0" << endl;
3523      saida << "..... NIVEL DE CONFIANCA: 95%" << "\n" << endl;
3524      saida << "COBERTURA:      "
3525      << "HC0 = " << (sum(cob_0_95_t_percentil_duplo1)/nrep)*100 << ", "
3526      << "HC2 = " << (sum(cob_2_95_t_percentil_duplo1)/nrep)*100 << ", "
3527      << "HC3 = " << (sum(cob_3_95_t_percentil_duplo1)/nrep)*100 << ", "
3528      << "HC4 = " << (sum(cob_4_95_t_percentil_duplo1)/nrep)*100 << ", "
3529      << "HC5 = " << (sum(cob_5_95_t_percentil_duplo1)/nrep)*100 << endl;
3530      saida << "APLITUDE:      "
3531      << "HC0 = " << (sum(aml_0_95_t_percentil_duplo1)/nrep) << ", "
3532      << "HC2 = " << (sum(aml_2_95_t_percentil_duplo1)/nrep) << ", "
3533      << "HC3 = " << (sum(aml_3_95_t_percentil_duplo1)/nrep) << ", "
3534      << "HC4 = " << (sum(aml_4_95_t_percentil_duplo1)/nrep) << ", "
3535      << "HC5 = " << (sum(aml_5_95_t_percentil_duplo1)/nrep) << endl;
3536      saida << "NAO COB. ESQ.:  "

```

```

3537         << "HC0 = " << (sum(ncobesq_0_95_t_percentil_duplo1)/nrep)*100 << ", "
3538         << "HC2 = " << (sum(ncobesq_2_95_t_percentil_duplo1)/nrep)*100 << ", "
3539         << "HC3 = " << (sum(ncobesq_3_95_t_percentil_duplo1)/nrep)*100 << ", "
3540         << "HC4 = " << (sum(ncobesq_4_95_t_percentil_duplo1)/nrep)*100 << ", "
3541         << "HC5 = " << (sum(ncobesq_5_95_t_percentil_duplo1)/nrep)*100 << endl;
3542 saida << "NAO COB. DIR.: "
3543         << "HC0 = " << (sum(ncobdi_0_95_t_percentil_duplo1)/nrep)*100 << ", "
3544         << "HC2 = " << (sum(ncobdi_2_95_t_percentil_duplo1)/nrep)*100 << ", "
3545         << "HC3 = " << (sum(ncobdi_3_95_t_percentil_duplo1)/nrep)*100 << ", "
3546         << "HC4 = " << (sum(ncobdi_4_95_t_percentil_duplo1)/nrep)*100 << ", "
3547         << "HC4 = " << (sum(ncobdi_5_95_t_percentil_duplo1)/nrep)*100 << "\n"
3548         << endl;
3549
3550 saida << "..... BOOTSTRAP T DUPLO (ESQUEMA ERRADO)" << endl;
3551 saida << "..... NIVEL DE CONFIANCA: 95%" << "\n" << endl;
3552 saida << "COBERTURA: "
3553         << "HC0 = " << (sum(cob_0_95_t_percentil_duplo)/nrep)*100 << ", "
3554         << "HC2 = " << (sum(cob_2_95_t_percentil_duplo)/nrep)*100 << ", "
3555         << "HC3 = " << (sum(cob_3_95_t_percentil_duplo)/nrep)*100 << ", "
3556         << "HC4 = " << (sum(cob_4_95_t_percentil_duplo)/nrep)*100 << ", "
3557         << "HC5 = " << (sum(cob_5_95_t_percentil_duplo)/nrep)*100 << endl;
3558 saida << "APLITUDE: "
3559         << "HC0 = " << (sum(aml_0_95_t_percentil_duplo)/nrep) << ", "
3560         << "HC2 = " << (sum(aml_2_95_t_percentil_duplo)/nrep) << ", "
3561         << "HC3 = " << (sum(aml_3_95_t_percentil_duplo)/nrep) << ", "
3562         << "HC4 = " << (sum(aml_4_95_t_percentil_duplo)/nrep) << ", "
3563         << "HC5 = " << (sum(aml_5_95_t_percentil_duplo)/nrep) << endl;
3564 saida << "NAO COB. ESQ.: "
3565         << "HC0 = " << (sum(ncobesq_0_95_t_percentil_duplo)/nrep)*100 << ", "
3566         << "HC2 = " << (sum(ncobesq_2_95_t_percentil_duplo)/nrep)*100 << ", "
3567         << "HC3 = " << (sum(ncobesq_3_95_t_percentil_duplo)/nrep)*100 << ", "
3568         << "HC4 = " << (sum(ncobesq_4_95_t_percentil_duplo)/nrep)*100 << ", "
3569         << "HC5 = " << (sum(ncobesq_5_95_t_percentil_duplo)/nrep)*100 << endl;
3570 saida << "NAO COB. DIR.: "
3571         << "HC0 = " << (sum(ncobdi_0_95_t_percentil_duplo)/nrep)*100 << ", "
3572         << "HC2 = " << (sum(ncobdi_2_95_t_percentil_duplo)/nrep)*100 << ", "
3573         << "HC3 = " << (sum(ncobdi_3_95_t_percentil_duplo)/nrep)*100 << ", "
3574         << "HC4 = " << (sum(ncobdi_4_95_t_percentil_duplo)/nrep)*100 << ", "
3575         << "HC5 = " << (sum(ncobdi_5_95_t_percentil_duplo)/nrep)*100 << "\n"
3576         << endl;
3577
3578 saida << "..... BOOTSTRAP PERCENTIL" << endl;
3579 saida << "..... NIVEL DE CONFIANCA: 99%" << "\n" << endl;
3580 saida << "COBERTURA: " << (sum(cob99_percentil)/nrep)*100 << endl;
3581 saida << "APLITUDE: " << (sum(aml99_percentil)/nrep) << endl;
3582 saida << "NAO COB. ESQ.: " << (sum(ncobesq99_percentil)/nrep)*100 << endl;
3583 saida << "NAO COB. DIR.: " << (sum(ncobdi99_percentil)/nrep)*100 << "\n"
3584         << endl;
3585
3586 saida << "..... BOOTSTRAP PERCENTIL DUPLO" << endl;
3587 saida << "..... NIVEL DE CONFIANCA: 99%" << "\n" << endl;
3588 saida << "COBERTURA: " << (sum(cob99_percentil_duplo)/nrep)*100 << endl;
3589 saida << "APLITUDE: " << (sum(aml99_percentil_duplo)/nrep) << endl;
3590 saida << "NAO COB. ESQ.: " << (sum(ncobesq99_percentil_duplo)/nrep)*100
3591         << endl;
3592 saida << "NAO COB. DIR.: " << (sum(ncobdi99_percentil_duplo)/nrep)*100 << "\n"
3593         << endl;
3594

```

```

3595      saida << "..... BOOTSTRAP T" << endl;
3596      saida << "..... NIVEL DE CONFIANCA: 99%" << "\n" << endl;
3597      saida << "COBERTURA:      "
3598          << "HC0 = " << (sum(cob_0_99_t_percentil)/nrep)*100 << ", "
3599          << "HC2 = " << (sum(cob_2_99_t_percentil)/nrep)*100 << ", "
3600          << "HC3 = " << (sum(cob_3_99_t_percentil)/nrep)*100 << ", "
3601          << "HC4 = " << (sum(cob_4_99_t_percentil)/nrep)*100 << ", "
3602          << "HC5 = " << (sum(cob_5_99_t_percentil)/nrep)*100 << endl;
3603      saida << "APLITUDE:      "
3604          << "HC0 = " << (sum(ampl_0_99_t_percentil)/nrep) << ", "
3605          << "HC2 = " << (sum(ampl_2_99_t_percentil)/nrep) << ", "
3606          << "HC3 = " << (sum(ampl_3_99_t_percentil)/nrep) << ", "
3607          << "HC4 = " << (sum(ampl_4_99_t_percentil)/nrep) << ", "
3608          << "HC5 = " << (sum(ampl_5_99_t_percentil)/nrep) << endl;
3609      saida << "NAO COB. ESQ.:      "
3610          << "HC0 = " << (sum(ncobesq_0_99_t_percentil)/nrep)*100 << ", "
3611          << "HC2 = " << (sum(ncobesq_2_99_t_percentil)/nrep)*100 << ", "
3612          << "HC3 = " << (sum(ncobesq_3_99_t_percentil)/nrep)*100 << ", "
3613          << "HC4 = " << (sum(ncobesq_4_99_t_percentil)/nrep)*100 << ", "
3614          << "HC5 = " << (sum(ncobesq_5_99_t_percentil)/nrep)*100 << endl;
3615      saida << "NAO COB. DIR.:      "
3616          << "HC0 = " << (sum(ncobdi_0_99_t_percentil)/nrep)*100 << ", "
3617          << "HC2 = " << (sum(ncobdi_2_99_t_percentil)/nrep)*100 << ", "
3618          << "HC3 = " << (sum(ncobdi_3_99_t_percentil)/nrep)*100 << ", "
3619          << "HC4 = " << (sum(ncobdi_4_99_t_percentil)/nrep)*100 << ", "
3620          << "HC5 = " << (sum(ncobdi_5_99_t_percentil)/nrep)*100 << "\n"
3621          << endl;
3622
3623      saida << "..... BOOTSTRAP T DUPLO" << endl;
3624      saida << "..... NIVEL DE CONFIANCA: 99%" << "\n" << endl;
3625      saida << "COBERTURA:      "
3626          << "HC0 = " << (sum(cob_0_99_t_percentil_duplo1)/nrep)*100 << ", "
3627          << "HC2 = " << (sum(cob_2_99_t_percentil_duplo1)/nrep)*100 << ", "
3628          << "HC3 = " << (sum(cob_3_99_t_percentil_duplo1)/nrep)*100 << ", "
3629          << "HC4 = " << (sum(cob_4_99_t_percentil_duplo1)/nrep)*100 << ", "
3630          << "HC5 = " << (sum(cob_5_99_t_percentil_duplo1)/nrep)*100 << endl;
3631      saida << "APLITUDE:      "
3632          << "HC0 = " << (sum(ampl_0_99_t_percentil_duplo1)/nrep) << ", "
3633          << "HC2 = " << (sum(ampl_2_99_t_percentil_duplo1)/nrep) << ", "
3634          << "HC3 = " << (sum(ampl_3_99_t_percentil_duplo1)/nrep) << ", "
3635          << "HC4 = " << (sum(ampl_4_99_t_percentil_duplo1)/nrep) << ", "
3636          << "HC5 = " << (sum(ampl_5_99_t_percentil_duplo1)/nrep) << endl;
3637      saida << "NAO COB. ESQ.:      "
3638          << "HC0 = " << (sum(ncobesq_0_99_t_percentil_duplo1)/nrep)*100 << ", "
3639          << "HC2 = " << (sum(ncobesq_2_99_t_percentil_duplo1)/nrep)*100 << ", "
3640          << "HC3 = " << (sum(ncobesq_3_99_t_percentil_duplo1)/nrep)*100 << ", "
3641          << "HC4 = " << (sum(ncobesq_4_99_t_percentil_duplo1)/nrep)*100 << ", "
3642          << "HC5 = " << (sum(ncobesq_5_99_t_percentil_duplo1)/nrep)*100 << endl;
3643      saida << "NAO COB. DIR.:      "
3644          << "HC0 = " << (sum(ncobdi_0_99_t_percentil_duplo1)/nrep)*100 << ", "
3645          << "HC2 = " << (sum(ncobdi_2_99_t_percentil_duplo1)/nrep)*100 << ", "
3646          << "HC3 = " << (sum(ncobdi_3_99_t_percentil_duplo1)/nrep)*100 << ", "
3647          << "HC4 = " << (sum(ncobdi_4_99_t_percentil_duplo1)/nrep)*100 << ", "
3648          << "HC4 = " << (sum(ncobdi_5_99_t_percentil_duplo1)/nrep)*100 << "\n"
3649          << endl;
3650
3651      saida << "..... BOOTSTRAP T DUPLO (ESQUEMA ERRADO)" << endl;
3652      saida << "..... NIVEL DE CONFIANCA: 99%" << "\n" << endl;

```

```

3653      saida << "COBERTURA:      "
3654          << "HC0 = " << (sum(cob_0_99_t_percentil_duplo)/nrep)*100 << ", "
3655          << "HC2 = " << (sum(cob_2_99_t_percentil_duplo)/nrep)*100 << ", "
3656          << "HC3 = " << (sum(cob_3_99_t_percentil_duplo)/nrep)*100 << ", "
3657          << "HC4 = " << (sum(cob_4_99_t_percentil_duplo)/nrep)*100 << ", "
3658          << "HC5 = " << (sum(cob_5_99_t_percentil_duplo)/nrep)*100 << endl;
3659      saida << "APLITUDE:      "
3660          << "HC0 = " << (sum(ampl_0_99_t_percentil_duplo)/nrep) << ", "
3661          << "HC2 = " << (sum(ampl_2_99_t_percentil_duplo)/nrep) << ", "
3662          << "HC3 = " << (sum(ampl_3_99_t_percentil_duplo)/nrep) << ", "
3663          << "HC4 = " << (sum(ampl_4_99_t_percentil_duplo)/nrep) << ", "
3664          << "HC5 = " << (sum(ampl_5_99_t_percentil_duplo)/nrep) << endl;
3665      saida << "NAO COB. ESQ.:  "
3666          << "HC0 = " << (sum(ncobesq_0_99_t_percentil_duplo)/nrep)*100 << ", "
3667          << "HC2 = " << (sum(ncobesq_2_99_t_percentil_duplo)/nrep)*100 << ", "
3668          << "HC3 = " << (sum(ncobesq_3_99_t_percentil_duplo)/nrep)*100 << ", "
3669          << "HC4 = " << (sum(ncobesq_4_99_t_percentil_duplo)/nrep)*100 << ", "
3670          << "HC5 = " << (sum(ncobesq_5_99_t_percentil_duplo)/nrep)*100 << endl;
3671      saida << "NAO COB. DIR.:  "
3672          << "HC0 = " << (sum(ncobdi_0_99_t_percentil_duplo)/nrep)*100 << ", "
3673          << "HC2 = " << (sum(ncobdi_2_99_t_percentil_duplo)/nrep)*100 << ", "
3674          << "HC3 = " << (sum(ncobdi_3_99_t_percentil_duplo)/nrep)*100 << ", "
3675          << "HC4 = " << (sum(ncobdi_4_99_t_percentil_duplo)/nrep)*100 << ", "
3676          << "HC5 = " << (sum(ncobdi_5_99_t_percentil_duplo)/nrep)*100 << "\n"
3677          << endl;
3678
3679      saida.close();
3680      //cout << "\a" << endl; // ALERTA SONORO.
3681      return 0;
3682  } // AQUI TERMINA A FUNCAO main().

```

Programa - Funções em R para o cálculo das estimativas intervalares bootstrap simples e duplo

Esse apêndice apresenta os códigos de funções que calculam algumas estimativas intervalares para os parâmetros que indexam o modelo linear de regressão com heteroscedasticidade de forma desconhecida. Essas funções (Pboot, Tboot) foram escritas utilizando a linguagem R e estão presentes no pacote `hcci`. Também é apresentada a função `HC` que estima a matriz de covariância de $\hat{\beta}$. Maiores informações de como utilizar essas funções foram apresentadas no Capítulo 5. Outras informações poderão ser encontradas nos manuais do pacote `hcci` disponível em <http://cran.r-project.org/>.

C.1 Função HC

```

1 HC <- function(model, method=4, k=0.7){
2
3   if(class(model)!="lm") stop("The argument model must have class lm.")
4   if(method%in%c(0,2,3,4,5) == FALSE){
5     warning("The argument method should be 0, 2, 3, 4 or 5. How did you
6       choose method = ", method, " that is not an option, the
7       calculation is considering method = 4.")
8     method=4L
9   }
10
11   X = as.matrix(cbind(1,model$model[, -1]))
12   bread_1 = solve(t(X)%*%X)%*%t(X)
13   bread_2 = X%*%solve(t(X)%*%X)
14   error_hat = as.vector(model$residuals)
15   h = as.vector(hatvalues.lm(model))
16
17   if(method==0L){
18     omega = diag(error_hat^2)
19     hc = bread_1%*%omega%*%bread_2

```

```

20   }
21
22   if(method==2L){
23     omega = diag((error_hat^2)/(1-h))
24     hc = bread_1%%omega%%bread_2
25   }
26
27   if(method==3L){
28     omega = diag((error_hat^2)/((1-h)^2))
29     hc = bread_1%%omega%%bread_2
30   }
31
32   if(method==4L){
33     delta = pmin(4,h/mean(h))
34     omega = diag((error_hat^2)/((1-h)^delta))
35     hc = bread_1%%omega%%bread_2
36   }
37
38   if(method==5L){
39     alpha = pmin(h/mean(h), max(4,k*max(h)/mean(h)))
40     omega = diag((error_hat^2)/sqrt((1-h)^alpha))
41     hc = bread_1%%omega%%bread_2
42   }
43   hc
44 }

```

C.2 Função Pboot

```

1  Pboot <- function(model, significance=0.05,
2     double=FALSE, J=NULL, K=NULL, distribution="rademacher"){
3
4     if(class(model)!="lm") stop("The argument model must have class lm.")
5     if(significance>=1 || significance<=0){
6       stop("The significance level should belong to the open interval (0,1).")
7     }
8
9     # Booth, J.G. and Hall, P. (1994). Monte Carlo approximation and the
10    # iterated bootstrap. Biometrika, 81, 331-340.
11
12    if(is.null(J)==TRUE || is.null(K)==TRUE){
13      L = length(model$residuals)^3
14      gamma2 = ((1/2) * (1-significance)^(-2) * (5/4-significance))^(1/3)
15      J = as.integer(gamma2*L^(2/3))
16      K = as.integer(gamma2^(-1)*L^(1/3))
17    }
18
19    is.wholenumber <- function(x, tol = .Machine$double.eps^0.5)
20      abs(x - round(x)) < tol
21
22    while(is.wholenumber(K/2)==FALSE && is.wholenumber((J+1)*significance)==FALSE){
23      while(is.wholenumber(K/2)==FALSE){
24        K = K+1
25      }
26      while(is.wholenumber((J+1)*significance)==FALSE){
27        J = J+1
28      }
29      while(is.wholenumber((J+1)/K)==FALSE){

```

```

30     K = K+1
31   }
32 }
33
34 number_parameters = length(model$coefficients)
35 X = as.matrix(cbind(1,model$model[, -1]))
36 n = dim(X)[1]
37 beta = as.vector(model$coefficients)
38 h = as.vector(hatvalues.lm(model))
39 Xbeta = X%*%beta
40 error_hat = as.vector(model$residuals)
41 root_1_less_h = sqrt(1-h)
42 U = matrix_beta_star = matrix(,nrow=J,ncol=number_parameters)
43 matrix_u = matrix(,nrow=K,ncol=number_parameters)
44
45 # Aqui inicia o primeiro bootstrap.
46 #browser()
47
48 for(j in 1:J){
49
50   if(distribution=="rademacher"){
51     t_star = as.vector(sample(c(-1,1),size=length(model$fitted.values),
52                               replace=TRUE))
53   }else t_star = rnorm(length(model$fitted.values),0,1)
54
55   y_star = as.vector(Xbeta + t_star*error_hat/root_1_less_h)
56   model_string = as.character(model$call$formula)
57   model_star = lm(formula=as.formula(paste("y_star~",
58                                           as.character(model_string[3]),sep="")))
59   error_hat_star = as.vector(model_star$residuals)
60   beta_star = as.vector(model_star$coefficients)
61   matrix_beta_star[j,] = as.vector(beta_star)
62
63   Xbeta_star = X%*%beta_star
64
65   # Aqui começa o bootstrap duplo.
66   if(double==TRUE){
67     for(k in 1:K){
68       if(distribution=="rademacher"){
69         t_star_star = as.vector(sample(c(-1,1),
70                                         size=length(model$fitted.values),
71                                         replace=TRUE))
72       }else t_star_star = rnorm(length(model$fitted.values),0,1)
73
74       y_star_star = as.vector(Xbeta_star
75                               + t_star_star*error_hat_star/root_1_less_h)
76       model_string = as.character(model$call$formula)
77       model_star_star = lm(formula=as.formula(paste("y_star_star~",
78                                                   as.character(model_string[3]),sep="")))
79       beta_star_star = as.vector(model_star_star$coefficients)
80
81       for(m in 1:number_parameters){
82         if(beta_star_star[m]<=2*beta_star[m]-beta[m]){
83           matrix_u[k,m] = 1
84         }else matrix_u[k,m] = 0
85       }
86     } # Aqui termina o segundo bootstrap.
87

```



```

88     U[j,] = as.vector(apply(matrix_u,2,mean))
89   } # Aqui termina o IF
90 } # Aqui termina o primeiro bootstrap.
91
92 ic_inf_simple = ic_sup_simple = ic_inf_double = ic_sup_double <- vector()
93
94 for(m in 1:number_parameters){
95   ic_inf_simple[m] = quantile(as.vector(matrix_beta_star[,m]), significance/2)
96   ic_sup_simple[m] = quantile(as.vector(matrix_beta_star[,m]), 1-significance/2)
97
98   if(double==TRUE){
99     ic_inf_double[m] = quantile(as.vector(matrix_beta_star[,m]),
100                                quantile(as.vector(U[,m]), significance/2))
101     ic_sup_double[m] = quantile(as.vector(matrix_beta_star[,m]),
102                                quantile(as.vector(U[,m]), 1-significance/2))
103   }
104 }
105
106 result = list("beta" = beta, "ci_lower_simple" = ic_inf_simple,
107              "ci_upper_simple" = ic_sup_simple, "ci_lower_double" = ic_inf_double,
108              "ci_upper_double" = ic_sup_double)
109 class(result) <- "list"
110 return(result)
111 }

```

C.3 Função Tboot

```

1  Tboot <-
2  function(model, significance=0.05, hc=4,
3           double=FALSE, J=NULL, K=NULL, distribution="rademacher"){
4
5    if(class(model)!="lm") stop("The argument model must have class lm.")
6    if(significance>=1 || significance<=0){
7      stop("The significance level should belong to the open interval (0,1).")
8    }
9    if(hc%in%c(0,2,3,4,5) == FALSE){
10     warning("The argument hc should be 0, 2, 3, 4 or 5. How did you choose hc = ",
11            hc, " that is not an option, the calculation is considering hc = 4.")
12     hc=4
13   }
14
15   # Booth, J.G. and Hall, P. (1994). Monte Carlo approximation
16   # and the iterated bootstrap. Biometrika, 81, 331-340.
17
18   if(is.null(J)==TRUE || is.null(K)==TRUE){
19     L = length(model$residuals)^3
20     gamma2 = ((1/2) * (1-significance)^(-2) * (5/4-significance))^(1/3)
21     J = as.integer(gamma2*L^(2/3))
22     K = as.integer(gamma2^(-1)*L^(1/3))
23   }
24
25   is.wholenumber <- function(x, tol = .Machine$double.eps^0.5)
26     abs(x - round(x)) < tol
27
28   while(is.wholenumber(K/2)==FALSE && is.wholenumber((J+1)*significance)==FALSE){
29     while(is.wholenumber(K/2)==FALSE){
30       K = K+1

```

```

31     }
32     while(is.wholenumber((J+1)*significance)==FALSE){
33         J = J+1
34     }
35     while(is.wholenumber((J+1)/K)==FALSE){
36         K = K+1
37     }
38 }
39 number_parameters = length(model$coefficients)
40 X = as.matrix(cbind(1,model$model[, -1]))
41 n = dim(X)[1]
42 beta = as.vector(model$coefficients)
43 h = as.vector(hatvalues.lm(model))
44 Xbeta = X%*%beta
45 error_hat = as.vector(model$residuals)
46 root_1_less_h = sqrt(1-h)
47 standard_error = as.vector(sqrt(diag(HC(model, method=4))))
48 Z = z_star = matrix(,nrow=J,ncol=number_parameters)
49 Z_temp = matrix(,nrow=K,ncol=number_parameters)
50
51 # Aqui inicia o primeiro bootstrap.
52 for(j in 1:J){
53
54     if(distribution=="rademacher"){
55         t_star = as.vector(sample(c(-1,1),size=length(model$fitted.values),
56                                 replace=TRUE))
57     }else t_star = rnorm(length(model$fitted.values),0,1)
58
59     y_star = as.vector(Xbeta + t_star*error_hat/root_1_less_h)
60     model_string = as.character(model$call$formula)
61     model_star = lm(formula=as.formula(paste("y_star~",
62                                             as.character(model_string[3]),sep="")))
63     error_hat_star = as.vector(model_star$residuals)
64     beta_star = as.vector(model_star$coefficients)
65     standard_error_star = sqrt(diag(HC(model_star, method=hc)))
66
67     for(m in 1:number_parameters){
68         z_star[j,m] = as.numeric((beta_star[m] - beta[m])/standard_error_star[m])
69     }
70
71     Xbeta_star = X%*%beta_star
72
73     if(double==TRUE){
74         # Aqui comeca o bootstrap duplo
75         for(k in 1:K){
76             if(distribution=="rademacher"){
77                 t_star_star = as.vector(sample(c(-1,1),size=length(model$fitted.values),
78                                                 replace=TRUE))
79             }else t_star_star = rnorm(length(model$fitted.values),0,1)
80             y_star_star = as.vector(Xbeta_star + t_star_star*error_hat_star/root_1_less_h)
81             model_string = as.character(model$call$formula)
82             model_star_star = lm(formula=as.formula(paste("y_star_star~",
83                                                         as.character(model_string[3]),sep="")))
84             beta_star_star = as.vector(model_star_star$coefficients)
85             standard_error_star_star = sqrt(diag(HC(model_star_star, method=hc)))
86
87             for(l in 1:number_parameters){
88                 z_star_star = as.numeric((beta_star_star[l]

```

```

89             - beta_star[1])/standard_error_star_star[1])
90         if(z_star_star<=as.numeric(z_star[j,1])) Z_temp[k,1] = 1
91         else Z_temp[k,1] = 0
92     }
93     } # Aqui termina o bootstrap duplo
94     Z[j,] = as.vector(apply(Z_temp,2,mean))
95 } # AQUI TERMINA O PRIMEIRO IF DO BOOTSTRAP.
96 } # Aqui termina o primeiro bootstrap
97
98 ic_inf_simple <- vector()
99 ic_sup_simple <- vector()
100 ic_inf_double <- vector()
101 ic_sup_double <- vector()
102
103 for(m in 1:number_parameters){
104     ic_inf_simple[m] = beta[m] - as.numeric(quantile(as.vector(z_star[,m]),
105                                                     1-significance/2))*standard_error[m]
106     ic_sup_simple[m] = beta[m] - as.numeric(quantile(as.vector(z_star[,m]),
107                                                     significance/2))*standard_error[m]
108
109     if(double==TRUE){
110         ic_inf_double[m] = beta[m] -
111             as.numeric(quantile(as.vector(z_star[,m]),
112                                 as.numeric(quantile(Z[,m],1-significance/2)))*standard_error[m]
113         ic_sup_double[m] = beta[m] -
114             as.numeric(quantile(as.vector(z_star[,m]),
115                                 as.numeric(quantile(Z[,m],significance/2)))*standard_error[m]
116     }
117 }
118 result = list("beta" = beta,"ci_lower_simple" = ic_inf_simple,
119              "ci_upper_simple" = ic_sup_simple,
120              "ci_lower_double" = ic_inf_double,
121              "ci_upper_double" = ic_sup_double, "J" = J,
122              "K" = K)
123 class(result) <- "list"
124 return(result)
125 }
```