Universidade Federal de São Carlos
Centro de Ciências Exatas e de Tecnologia
Departamento de Estatística

Correções de Bartlett baseadas em bootstrap no modelo de regressão Dirichlet com parametrização baseada no vetor de médias

Vinícius Morgado Rosa Vianna

Trabalho de Conclusão de Curso

Universidade Federal de São Carlos
Centro de Ciências Exatas e de Tecnologia
Departamento de Estatística

Correções de Bartlett baseadas em bootstrap no modelo de regressão Dirichlet com parametrização baseada no vetor de médias

Vinícius Morgado Rosa Vianna
Orientador: Prof. Dr. Gustavo Henrique de Araujo Pereira

Trabalho de Conclusão de Curso apresentado como parte dos requisitos para obtenção do título de Bacharel em Estatística.

São Carlos
Janeiro de 2025

Federal University of São Carlos
Exact and Technology Sciences Center
Department of Statistics

Bootstrap-based Bartlett corrections in the Dirichlet regression model with mean vector parametrization

Vinícius Morgado Rosa Vianna
Advisor: Gustavo Henrique de Araujo Pereira

Bachelors dissertation submitted to the Department of Statistics, Federal University of São Carlos - DEs-UFSCar, in partial fulfillment of the requirements for the degree of Bachelor in Statistics.

São Carlos
January 2025

Vinícius Morgado Rosa Vianna

Correções de Bartlett baseadas em bootstrap no modelo de regressão Dirichlet com parametrização baseada no vetor de médias

Este exemplar corresponde à redação final do trabalho de conclusão de curso devidamente corrigido e defendido por Vinícius Morgado Rosa Vianna e aprovado pela banca examinadora.

Aprovado em 30 de janeiro de 2025

Banca Examinadora:

- Gustavo Henrique de Araujo Pereira (Orientador)
- Maria Silvia de Assis Moura
- Pedro Ferreira Filho

Dedicatória

Dedico este trabalho aos meus pais, Roberto Rosa Vianna e Neusa Rodrigues Morgado Rosa Vianna e à minha avó, Maria Terezinha Scarelli.

Agradecimentos

Agradeço em primeiro lugar aos seres de luz que me acompanham e protegem, especialmente ao meu Preto Velho e ao meu Guardião, por sempre me auxiliarem com foco, sabedoria, luz, saúde e paciência ao longo de toda a graduação.

Agradeço em especial a minha mãe Neusa Rodrigues Morgado Rosa Vianna por sempre estar ao meu lado me apoiando, incentivando e me ouvindo falar sobre as coisas da graduação. Agradeço ao meu pai Roberto Rosa Vianna por sempre me apoiar nas minhas conquistas e à minha avó Maria Terezinha Scarelli por todo apoio incondicional nos momentos em que mais precisei, sendo minha companhia na maior parte do tempo em que volto para minha cidade natal. Obrigado por sempre acreditarem em mim e me ajudarem a evoluir cada vez mais.

Agradeço a todos os meus professores, em especial à professora Andressa Cerqueira com quem fiz diversas matérias da minha graduação e com certeza foi a melhor professora que tive e ao professor Ricardo Felipe Ferreira que também foi um professor excelente e essencial para todo o processo. Agradeco ao professor Pedro Ferreira Filho e à professora Maria Silvia de Assis Moura por terem feito parte da banca examinadora deste trabalho e por todas as contribuições para o mesmo.

Agradeço a todos os meus amigos que com certeza foram os responsáveis pelo ambiente da minha graduação ser tão bom, pois estiveram comigo nos melhores e piores momentos. Em especial, ao Pedro Lana Cordeiro e ao Matheus Yuiti Moriy Miata que foram meus companheiros de apartamento durante todo o período e ao Felipe Baptistão Durante Molina que foi um dos amigos que mais me inspirei.

Agradeço à música que é meu trabalho, meu porto seguro e sempre alivou minhas rotinas de estudo.

Por fim, agradeço ao meu orientador Gustavo Henrique de Araujo Pereira que me guiou e proporcionou uma estrutura ótima para que eu pudesse desenvolver tudo da melhor forma possível, aliviando meu maior ponto fraco que são as inseguranças e sendo a pessoa que eu sempre quis e precisava ter para meu desenvolvimento acadêmico.

“Não há nada que eu não saiba que eu não seja capaz de aprender.”

(Autoria Própria)

Resumo

Em modelos de regressão paramétrica, é comum termos interesse de testar hipóteses em relação ao valor de seus parâmetros. Neste quesito, o teste mais utilizado quando temos interesse em realizar um teste envolvendo mais de um parâmetro é o Teste de Razão de Verossimilhança (TRV). Esse teste pode ser usado por exemplo para realizar inferência sobre os parâmetros do modelo de regressão multivarado de Dirichlet, modelo esse muito usado na modelagem de dados composicionais, ou seja, observações positivas multivariadas somando um. Um problema desse teste é que a distribuição da sua estatística sob H_0 pode não ser bem aproximada pela distribuição qui-quadrado em pequenas amostras, tornando o teste muito liberal. Assim, utilizaremos um modelo com uma parametrização específica da distribuição Dirichlet em que os parâmetros são função do vetor de médias e aplicaremos um fator de correção de Bartlett baseada em bootstrap na estatística do TRV. Dessa forma, o teste corrigido deve ter estatística do teste com distribuição sob H_0 mais próxima da qui-quadrado independente do tamanho amostral. O objetivo principal deste trabalho é comparar a performance dos testes resultantes das correções bootstrap com o TRV usual. Os estudos de simulação de Monte Carlo que foram realizados mostraram que os testes com as correções aplicadas obtiveram taxas de rejeição nula bem próximas dos níveis nominais em tamanhos amostrais pequenos e moderados, tendo também um bom poder. A aplicação realizada em um conjunto de dados sobre estágios do sono ilustra a utilidade desses testes corrigidos.

Palavras-chave: *bootstrap, correção de Bartlett, distribuição Dirichlet, modelo de regressão Dirichlet, simulação de Monte Carlo, Teste de Razão de Verossimilhança (TRV).*

Abstract

In parametric regression models, it is common to be interested in testing hypotheses about the values of their parameters. In this regard, the most commonly used test when testing more than one parameter is the Likelihood Ratio Test (LRT). This test can be used, for example, to infer the parameters of the Dirichlet multivariate regression model, which is widely applied in modeling compositional data, i.e., multivariate positive observations summing to one. A problem with this test is that the distribution of its statistic under H_0 may not be well approximated by the chi-square distribution in small samples, making the test too liberal. Thus, we consider a model with a specific parameterization of the Dirichlet distribution, where the parameters are a function of the mean vector, and apply a Bartlett correction factor based on bootstrap to the LRT statistic. In this way, the corrected test is expected to have a test statistic whose distribution under H_0 is closer to the chi-square distribution, regardless of the sample size. The main objective of this study is to compare the performance of the tests resulting from the bootstrap corrections to the usual LRT. Monte Carlo simulation studies showed that the tests with the applied corrections achieved null rejection rates close to nominal levels in small and moderate sample sizes, while also demonstrating good power. An application on sleep stages illustrates the usefulness of these corrected tests.

Keywords: *bootstrap, Bartlett correction, Dirichlet distribution, Dirichlet regression model, Monte Carlo simulation, Likelihood Ratio Test (LRT).*

Lista de Figuras

Lista de Tabelas

Sumário

Capítulo 1

Introdução

A distribuição de probabilidade Dirichlet é amplamente utilizada para modelar dados composicionais, os quais são na forma de proporções, onde cada observação é um vetor de números positivos somando um. Esse tipo de dados é muito comum e surge naturalmente na prática em diversas áreas da ciência, como por exemplo ao analisarmos a composição corporal de um atleta, podemos ter três componentes como massa muscular, gordura corporal e água no corpo, sendo os três representados por porcentagens que somam 100%. Dados de composição podem ser modelados de diferentes formas, dentre elas a regressão de Dirichlet, a qual é frequentemente utilizada por capturar a correlação natural entre as proporções (isto é, se uma proporção aumenta, outra deve diminuir para manter a soma constante). Este modelo permite grande flexibilidade de modelagem, como pode ser visto em Morais *et al.* (2018) e Hanretty (2021) que são algumas das contribuições mais recentes, além de ser uma generalização da regressão Beta pelo fato de cada componente ter distribuição marginal Beta (Ferrari e Cribari-Neto, 2004). Existem duas maneiras de definir os parâmetros da distribuição Dirichlet: uma é conhecida como parametrização comum e a outra como parametrização alternativa (Morais *et al.*, 2018). Neste caso, optamos pela abordagem alternativa, na qual os parâmetros são expressos como funções do vetor de médias, tornando-os mais interpretáveis e adequados para ajustar o modelo de regressão em questão.

Em modelos de regressão paramétrica como o modelo Dirichlet (Hijazi e Jernigan, 2009), frequentemente temos interesse em testar hipóteses em relação ao valor de seus parâmetros, ou melhor, testar quais parâmetros são significativos para o modelo. Neste quesito, o teste mais utilizado quando há interesse em testar mais de um parâmetro simultaneamente é o Teste de Razão de Verossimilhanças (TRV) (Lawley, 1956). Nesse

teste, a distribuição de referência qui-quadrado não fornece, em pequenas amostras, uma boa aproximação da distribuição verdadeira da estatística sob H_0, sendo necessária alguma correção para obter um bom resultado.

Em geral, os resultados obtidos pela teoria estatística de grandes amostras se preocupa com o comportamento dos procedimentos estatísticos à medida que o tamanho da amostra aumenta. No TRV, o valor crítico utilizado é aproximado, obtido a partir da distribuição nula limitante da estatística. Dessa forma, podem ocorrer distorções de tamanho em amostras pequenas, resultando em um valor muito discrepante do esperado (Botter, 1993). Nesse âmbito, um fator de correção de Bartlett é proposto, o qual, na prática, pode ser obtido a partir dos cumulantes conjuntos da derivação da função log-verossimilhança, sendo incorporado à estatística usual do TRV. Além desse fator de correção analítica, há também dois tipos de correções de Bartlett baseadas em bootstrap. Essas correções representam uma importante área de pesquisa na teoria assintótica devido à sua ampla aplicabilidade, tratando as limitações presentes na maioria dos resultados baseados em grandes amostras (Cordeiro e Cribari-Neto, 2014).

Melo *et al.* (2022) realizaram um estudo de refinamentos assintóticos de ordem superior (melhorias) no modelo de regressão multivariado de Dirichlet, aplicando diversas correções na estatística do Teste de Razão de Verossimilhança, entre elas as correções de Bartlett, verificando a efetividade dessas correções em diferentes tamanhos amostrais. Dessa forma, este trabalho será complementar à pesquisa citada, utilizando outra parametrização da distribuição Dirichlet, além de aplicar uma segunda correção de Bartlett baseada em bootstrap. Nosso principal objetivo é verificar, com enfoque nessa parametrização alternativa, a performance dessas duas correções de Bartlett por bootstrap em comparação com o TRV usual.

Este trabalho está organizado da seguinte forma: no Capítulo 2, apresentamos a estrutura e os componentes do modelo de regressão Dirichlet, destacando também algumas características da distribuição Dirichlet. No Capítulo 3, abordamos os diferentes testes da razão de verossimilhanças que serão comparados neste trabalho, definindo tanto o teste usual como os testes com as correções propostas. Em seguida, com a implementação de toda metodologia estudada no software R, no Capítulo 4 realizamos estudos de simulação de Monte Carlo para comparar a performance dos testes mencionados na regressão Dirichlet. No Capítulo 5, realizamos uma aplicação em um conjunto de dados sobre estágios do sono, mostrando como as correções desempenham em dados reais. Por fim, no Capítulo

6 temos uma síntese final sobre as conclusões que foram obtidas a cerca do objetivo em estudo com base na metodologia proposta.

Capítulo 2

Modelo de Regressão Dirichlet

Neste capítulo são discutidos os principais aspectos do modelo de regressão Dirichlet, com exceção do teste de hipótese, que será apresentado no capítulo seguinte.

2.1 Distribuição Dirichlet

Servindo como base para esta pesquisa, a distribuição de probabilidade Dirichlet é amplamente utilizada para modelar dados composicionais, ou seja, dados na forma de proporções em que cada observação é um vetor de números positivos somando um. Tal distribuição possui duas parametrizações mais conhecidas: a comum e a alternativa. Na primeira, temos k parâmetros de forma, sendo k o número de componentes da variável aleatória composicional, ou seja, k é a dimensão do vetor aleatório de cada elemento da variável resposta. Já sua parametrização alternativa, a qual apresenta os parâmetros como funções do vetor de médias, possui um formato mais adequado para o modelo que iremos ajustar, pois fornece maior interpretabilidade.

Seja W uma variável aleatória contínua composicional, isto é, $W = (W_1, W_2, ..., W_k)^T$, $0 < W_j < 1$ e $\sum_{j=1}^{k} W_j = 1$. A variável aleatória W tem distribuição de probabilidade Dirichlet com parâmetros $(\mu_1, \mu_2, ..., \mu_k, \phi)^T$ (parametrização alternativa), sendo $\mu_1 = 1 - \sum_{j=2}^{k} \mu_j$, $0 < \mu_j < 1$, $\forall j \in 1, 2, ..., k$ e $\phi > 0$, se sua função densidade de probabilidade é dada por (Da-Silva e Rodrigues, 2015):

$$f(w_1, w_2, ..., w_k; \mu_1, \mu_2, ..., \mu_k, \phi) = \frac{\Gamma(\phi)}{\prod_{j=1}^{k} \Gamma(\phi\mu_j)} \prod_{j=1}^{k} w_j^{\phi\mu_j - 1}. \tag{2.1}$$

A partir desta parametrização, os momentos da distribuição são (Pereira e Cai, 2024):

$$E(W_j|\mu_j,\phi)=\mu_j \quad e \quad Var(W_j|\mu_j,\phi)=\frac{\mu_j(1-\mu_j)}{1+\phi} \tag{2.2}$$

O que torna ela interessante é o fato da esperança de cada componente de W ser diretamente um dos parâmetros do modelo, sendo assim atrativa para o ajuste de modelos de regressão, pois podemos interpretar o efeito das covariáveis diretamente na média, isto é, o parâmetro do modelo já representa a grandeza de interesse, dispensando interpretações mais complexas. Já ϕ é um parâmetro de precisão, ou seja, quanto maior seu valor, menor é a variabilidade de W, sendo comprovado pela fórmula apresentada para o segundo momento da distribuição (variância).

2.2 Modelo de Regressão Dirichlet

Dados composicionais são comuns em diversas áreas e podem ser modelados de diferentes maneiras (Prado, 2017), sendo a regressão Dirichlet um modelo interessante para esse tipo de dados, a qual também pode ser vista como uma generalização da regressão Beta (Ferrari e Cribari-Neto, 2004).

Dando continuidade à parametrização Dirichlet apresentada na seção anterior, consideramos n observações de dados composicionais, $Y_1, Y_2, \ldots, Y_n$, em que cada Y_i é um vetor $(Y_{i1}, Y_{i2}, \ldots, Y_{ik})^T$ que segue uma distribuição Dirichlet com parametrização alternativa, isto é, para cada i, $Y_i \sim \text{Dirichlet}(\mu_{i1}, \mu_{i2}, \ldots, \mu_{ik}, \phi_i)$.

Para modelar a dependência de μ_{ij} e ϕ_i em relação a uma variável explicativa, definimos o modelo de regressão logística de Dirichlet (Pereira e Cai, 2024):

$$log\Big(\frac{\mu_{ij}}{\mu_{i1}}\Big) = x_{ij1}\beta_{j1} + x_{ij2}\beta_{j2} + ... + x_{ijp_j}\beta_{jp_j}, \quad 2 \leq j \leq k, \tag{2.3}$$

$$log(\phi_i) = d_{i1}\gamma_1 + d_{i2}\gamma_2 + ... + d_{ip_\phi}\gamma_{p_\phi}, \tag{2.4}$$

em que $(x_{ij1}, x_{ij2}, \ldots, x_{ijp_j}, d_{i1}, d_{i2}, \ldots, d_{ip_\phi})^T$ são constantes que representam os valores das covariáveis e $(\beta_{j1}, \beta_{j2}, \ldots, \beta_{jp_j}, \gamma_1, \gamma_2, \ldots, \gamma_{p_\phi})^T$ são os parâmetros desconhecidos. O modelo utiliza a função de ligação logito para as expressões relacionadas ao parâmetro da média e o logaritmo para a expressão relacionada ao parâmetro de precisão. Vale ressaltar que outras funções de ligação poderiam ser utilizadas neste modelo com resposta Dirichlet, porém, combinando a parametrização utilizada com a função de ligação logito

obtemos parâmetros interpretáveis, isto é, temos a interpretação em termos de log-odds, a qual já é bastante difundida em modelos de regressão logística no geral (Norton e Dowd, 2018).

Os coeficientes do modelo (2.3) e (2.4) podem ser estimados através do método da máxima verossimilhança, sendo necessário utilizar métodos numéricos para a obtenção desses estimadores (EMVs), assim como na maioria dos modelos de regressão. O ajuste do modelo pode ser realizado utilizando o pacote *DirichletReg* (Maier, 2014) do software *R*. Este pacote será empregado em nossos estudos de simulação e na aplicação do Capítulo 5.

Em modelos de regressão paramétrica, nosso principal objetivo é compreender como as variáveis em estudo se relacionam e avaliar a intensidade dessa influência, de modo a viabilizar previsões futuras para a variável resposta. Entretanto, para termos inferências razoáveis a respeito do modelo utilizado, é comum testar hipóteses em relação ao valor dos parâmetros a fim de identificar quais variáveis são mais adequadas para o ajuste e, assim, ter um modelo parcimonioso. Os testes mais conhecidos na literatura são o teste de Wald, escore e da razão de verossimilhanças (Cordeiro *et al.*, 2024). Este último será mais explorado neste estudo e utilizado para alcançar o objetivo desejado.

2.3 Análise de Diagnóstico

Ao realizar o ajuste de um modelo de regressão, uma importante etapa é verificar a adequação do modelo aos dados, verificando também se há valores discrepantes ou influentes interferindo no ajuste. Para isso, é interessante que uma análise de diagnóstico seja realizada com base em resíduos cuja distribuição seja bem aproximada pela distribuição normal padrão. Caso contrário, os gráficos de resíduos podem ser difíceis de serem interpretados e modelos bem ajustados talvez sejam descartados.

Na literatura, há vários resíduos que são utilizados em diversos modelos de regressão, dentre eles o resíduo quantílico, o qual pode ser calculado de duas maneiras diferentes para dados composicionais: univariado (ou marginal) e multivariado. A abordagem univariada apresenta certa desvantagem na identificação de outliers e na verificação da adequabilidade do modelo, pois cada observação possui vários resíduos. Já a abordagem multivariada também pode não ser tão boa, pois ao calcular um único resíduo para cada observação, este talvez não seja uma boa medida de discrepância entre o valor observado

e o valor ajustado pelo modelo. Além desse resíduo, outros foram propostos para dados composicionais (Gueorguieva *et al.*, 2008), mas eles apresentam as mesmas desvantagens citadas. Recentemente foi proposta uma classe de resíduos multivariados baseados em bootstrap para dados composicionais que têm distribuição assintótica normal padrão, sendo também bem aproximada por essa distribuição em pequenas amostras (Pereira e Cai, 2024). Usaremos esse resíduo na aplicação deste trabalho.

Capítulo 3

Testes da Razão de Verossimilhanças

O Teste da Razão de Verossimilhanças (TRV) é um teste de hipóteses que compara dois modelos para determinar se o modelo mais complexo oferece uma melhora significativa em relação ao modelo mais simples, sendo um modelo irrestrito com todos os parâmetros livres, e seu modelo correspondente restrito pela hipótese nula em que alguns parâmetros assumem um valor fixado (em geral zero). Por exemplo, podemos usar o TRV para comparar a qualidade do ajuste de um modelo de regressão com um parâmetro (restrito) com o mesmo modelo de regressão com dois parâmetros (irrestrito). No TRV, a estatística de teste sob a hipótese nula segue uma distribuição assintótica qui-quadrado (χ^2) com um número de graus de liberdade igual ao número de restrições impostas pela hipótese nula, ou seja, a diferença na quantidade de parâmetros do modelo irrestrito para o restrito. Assim, se o valor-p do TRV for menor do que o nível de significância, rejeitamos a hipótese nula e concluímos que o modelo para dois parâmetros oferece uma qualidade do ajuste significativamente melhor do que o modelo para um parâmetro.

3.1 Teste da Razão de Verossimilhanças Usual

Seja um modelo paramétrico $f(y;\theta)$, com função de probabilidade ou densidade definida com o vetor de parâmetros $\theta = (\beta_a, \beta_b, \phi)$, sendo β_a o vetor de parâmetros que não estão sendo testados e β_b o vetor de parâmetros a serem testados. Queremos testar $H_0 : \beta_b = \beta_b^{(0)} \times H_1 : \beta_b \neq \beta_b^{(0)}$ em que $\beta_b^{(0)} = \underset{\sim}{0}$ de dimensão q. A estatística de teste do TRV é dada por:

$$Q_{rv} = 2[log(\hat{\beta}_a, \hat{\beta}_b, \hat{\phi}) - log(\tilde{\beta}_a, \tilde{\beta}_b, \tilde{\phi})], \tag{3.1}$$

em que $log(\hat{\beta}_a, \hat{\beta}_b, \hat{\phi})$ é o logaritmo da função de verossimilhança do modelo aplicado no EMV irrestrito dos parâmetros e $log(\tilde{\beta}_a, \tilde{\beta}_b, \tilde{\phi})$ é o logaritmo da função de verossimilhança do modelo aplicado no EMV restrito dos parâmetros (sob H_0). A definição dessa estatística de teste é baseada em medir quanto a inclusão de parâmetros adicionais (no modelo completo) melhora o ajuste em comparação ao modelo mais simples (reduzido), ou seja, valores maiores indicam maior diferença entre as verossimilhanças, sugerindo que o modelo completo pode fornecer um ajuste melhor; caso contrário, a vantagem do modelo mais complexo pode não ser estatisticamente relevante.

O TRV, conforme especificado acima, é o teste mais simples para avaliar mais de um parâmetro ao mesmo tempo, especialmente no nosso caso da regressão Dirichlet, em que é comum querermos investigar se uma ou mais covariáveis afetam o vetor de médias.

Por outro lado, quando trabalhamos com tamanhos amostrais pequenos, o TRV não apresenta um desempenho razoável, isto é, a distribuição nula exata da estatística de teste não é muito bem aproximada pela distribuição qui-quadrado (Cordeiro e Cribari-Neto, 2014). Dessa maneira, como forma de melhorar a inferência do TRV e superar este problema, utilizaremos duas abordagens: correção de Bartlett e correção no valor crítico, ambas baseadas no método de Bootstrap.

3.2 Correções para o Teste da Razão de Verossimilhanças por Bootstrap

O método bootstrap foi originalmente proposto por Efron (1979) num contexto de estimação. Porém, esse método também vem sendo bastante utilizado para a condução de testes de hipótese, no nosso caso, iremos considerar versões da estatística do TRV que são corrigidas diretamente pelo bootstrap (estimativa do respectivo quantil crítico) ou que utilizam a técnica na correção (Bartlett-bootstrap). Maiores detalhes dessas correções podem ser vistos em Cordeiro e Cribari-Neto (2014).

3.2.1 Correção de Bartlett por Bootstrap

Inicialmente, Rocke (1989) apresenta uma forma para calcular o fator de correção de Bartlett baseada em bootstrap. A proposta principal é utilizar o método de reamostragem para calcular o valor esperado da estatística do TRV, o qual é utilizado para correção da

estatística usual. A estatística corrigida é definida como:

$$Q_{boot1} = \frac{q \cdot Q_{rv}}{\overline{Q_{rv}}}, \tag{3.2}$$

sendo Q_{rv} a estatística de teste do TRV usual (da amostra original) definida em (3.1), q o número de restrições impostas pela hipótese nula (ou número de parâmetros a serem testados simultaneamente) e $\overline{Q_{rv}}$ a média do valor das estatísticas do TRV usual $\overset{*}{Q}_{rv_b}$ calculadas em cada uma das B amostras bootstrap, isto é:

$$\overline{Q_{rv}} = \frac{1}{B}\sum_{b=1}^{B} \overset{*}{Q}_{rv_b}, \tag{3.3}$$

Dessa forma, a correção é realizada da seguinte forma:

1. Ajusta-se os modelos completo e reduzido utilizando as covariáveis presentes no banco de dados e calcula-se a estatística de teste do TRV usual Q_{rv};

2. Calcula-se os parâmetros da distribuição Dirichlet para as n observações com base nos coeficientes estimados do modelo reduzido ajustado e nas covariáveis;

3. Para cada iteração b, com $b = 1, ..., B$, gera-se uma nova amostra a partir da distribuição Dirichlet utilizando os parâmetros calculados no passo 2, estima-se o modelo completo e o modelo reduzido e calcula-se $\overset{*}{Q}_{rv_b}$;

4. Calcula-se a estatística corrigida Q_{boot1} com base nas amostras geradas;

5. Calcula-se o valor-p para Q_{boot1} considerando a distribuição de referência qui-quadrado.

3.2.2 Correção no Valor Crítico por Bootstrap

Outra estratégia de correção do TRV é utilizar o bootstrap com a finalidade de estimar diretamente a distribuição da estatística de teste. Note que essa correção não é aplicada diretamente na estatística de teste, pois em vez de recorrer à aproximação assintótica usual (distribuição qui-quadrado), utilizamos a distribuição empírica do teste para calcular o valor-p. Assim, os três primeiros passos para a aplicação da correção são iguais à correção anterior, alterando apenas a forma como as estatísticas calculadas para cada reamostra de bootstrap são utilizadas, ou seja:

1. a 3. Semelhantes à correção anterior;

4. Calcula-se o valor-p = $(k+1)/(B+1)$ considerando a distribuição estimada, em que $k = \#\overset{*}{Q}_{rv_b} > Q_{rv}$, isto é, k é o número de estatísticas de teste calculadas nas reamostras que são maiores que a estatística de teste calculada utilizando a amostra original. Com base nessa correção, a hipótese nula do TRV é rejeitada se o valor-p calculado for menor do que o nível de significância α.

Capítulo 4

Estudos de Simulação

Para avaliar se as correções bootstrap descritas melhoram o desempenho do TRV em amostras pequenas, utilizaremos estudos de simulação de Monte Carlo com os seguintes testes: TRV usual (TRV), TRV com correção de Bartlett por bootstrap (Boot1) e TRV com correção no valor crítico por bootstrap (Boot2). Vale ressaltar que o ajuste do modelo de regressão Dirichlet será realizado utilizando o pacote *DirichletReg* (Maier, 2014) do software *R*, apresentado no Apêndice A.

4.1 Paralelização em Simulações

Ao desenvolver algoritmos de simulações no geral, principalmente envolvendo bootstrap, uma dificuldade inevitável é a alta demanda de tempo para rodar os códigos. Neste sentido, para superar esta complicação, utilizamos mecanismos de programação em paralelo que possibilitaram um ganho significativo no tempo de execução dos *scripts*. A paralelização de simulações depende de um bom controle das sementes utilizadas por cada núcleo do paralelo, a fim de garantir a reprodutibilidade do algoritmo. Além disso, o pacote utilizado para construção do ambiente paralelo foi o $future\_apply$ (Bengtsson, 2020), o qual fornece implementações paralelas das funções *apply* do *R* base. (Apêndice B)

4.2 Estudos de Simulação

Para cada cenário construído para as simulações, utilizamos 5000 replicações de Monte Carlo e $B = 1000$ réplicas de bootstrap. Esse valor de B é recomendado principalmente

por conta da segunda correção (Boot2), pois calculamos um quantil com base em uma distribuição empírica, o que é consideravelmente difícil de estimar e, portanto, exige mais replicações do que a primeira correção, que com apenas 200 réplicas já apresenta um resultado razoável (Cordeiro e Cribari-Neto, 2014).

Primeiramente, avaliamos as taxas de rejeição nula de cada teste considerando três níveis de significância (1%, 5% e 10%) e cinco tamanhos amostrais (20, 30, 40, 50 e 100).

O seguinte modelo de regressão Dirichlet foi utilizado nas simulações:

$$log\left(\frac{\mu_{i2}}{\mu_{i1}}\right) = \beta_{11} + x_{i12}\beta_{12} + x_{i13}\beta_{13}, \tag{4.1}$$

$$log\left(\frac{\mu_{i3}}{\mu_{i1}}\right) = \beta_{21} + x_{i22}\beta_{22} + x_{i23}\beta_{23}, \tag{4.2}$$

$$log(\phi_i) = \gamma_1 + d_{i2}\gamma_2 + d_{i3}\gamma_3, \tag{4.3}$$

considerando $x_{i12} = x_{i22} = d_{i2}$ e $x_{i13} = x_{i23} = d_{i3}$. Avaliaremos alguns cenários para testar a hipótese:

$$H_0 : \beta_{13} = 0, \beta_{23} = 0 \tag{4.4}$$

$$H_1 : \text{Violação de pelo menos uma igualdade.}$$

Ou seja, testaremos se o vetor de médias da variável resposta varia com a covariável $x_{i13} = x_{i23} = d_{i3}$.

No cenário 1, definimos $\beta_{11} = -0,5$, $\beta_{12} = 1,1$, $\beta_{13} = 0$, $\beta_{21} = -0,5$, $\beta_{22} = 1,1$, $\beta_{23} = 0$, $\gamma_1 = 4,6$, $\gamma_2 = 0$ e $\gamma_3 = 0$. Esses valores de parâmetros foram atribuídos de forma que $\mu_1 \in (0{,}22; 0{,}45)$, $\mu_2 \in (0{,}27; 0{,}39)$ e $\mu_3 \in (0{,}27; 0{,}39)$, isto é, substituindo os valores dos parâmetros nos modelos (4.1) e (4.2) e isolando os componentes de médias μ_1, μ_2 e μ_3, obtemos as fórmulas para cada componente e, assim, calculamos o limite superior do intervalo para quando as covariáveis têm valor um e o limite inferior para quando elas têm valor zero, tendo assim as médias dos intervalos de variação de cada componente todas próximas de 0,333 para os parâmetros estipulados. Além disso, os valores das covariáveis foram gerados aleatoriamente da distribuição uniforme padrão e mantidos fixos em todas as réplicas de Monte Carlo.

Para o cenário 2, os parâmetros β_{11} e β_{21} foram alterados para -1 e $-1,5$ respectivamente, explorando assim outra área do espaço paramétrico com $\mu_1 \in (0{,}36; 0{,}63)$, $\mu_2 \in (0{,}23; 0{,}40)$ e $\mu_3 \in (0{,}14; 0{,}24)$, ou seja, o ponto médio dos intervalos de variação de

cada componente iguais a 0,49, 0,32 e 0,19, respectivamente. Para o cenário 3, apenas o parâmetro γ_1 foi reduzido para $2,3$, aumentando a variabilidade da variável resposta. Para o cenário 4, alteramos a distribuição das covariáveis, sendo x_1 com distribuição Normal e x_2 com distribuição Gama, ambas com média e variância iguais a da distribuição uniforme padrão, isto é, média igual a 1/2 e variância igual a 1/12. Por fim, para o cenário 5, o parâmetro γ_2 foi alterado para $-2,5$ e os testes passaram a serem realizados considerando as covariáveis no modelo do parâmetro de precisão, isto é, a hipótese nula do teste (4.4) passou a ser $H_0 : \beta_{13} = 0, \beta_{23} = 0, \gamma_3 = 0$.

4.2.1 Taxa de Rejeição Nula

A Tabela 4.1 apresenta as taxas de rejeição nula para o teste de hipóteses (4.4) no cenário 1. Notamos claramente que o TRV usual é liberal, com porcentagens de rejeição da hipótese nula bem mais altas do que o nível de significância esperado, como por exemplo quando $\alpha = 1\%$ e $n = 20$ em que a taxa de rejeição nula é de $3,84\%$, ou seja, mais que o triplo do nível nominal do teste, comprovando que, para amostras pequenas, o TRV usual não é adequado. Por outro lado, os testes corrigidos apresentam resultados bem mais próximos dos níveis α, como por exemplo no mesmo caso citado anteriormente em que as taxas são $0,94\%$ (Boot1) e $0,96\%$ (Boot2), ou seja, uma performance significativamente melhor. Em suma, o TRV usual apresenta taxas distorcidas para tamanhos amostrais pequenos e vai melhorando seu desempenho conforme este tamanho aumenta. Já os dois testes corrigidos apresentam um desempenho ideal, tendo taxa de rejeição nula próxima do nível nominal em todos os tamanhos amostrais considerados. Mesmo em tamanhos amostrais moderados ($n = 100$), as correções também apresentam desempenho melhor em relação ao TRV usual.

As Tabelas 4.2 a 4.5 apresentam as taxas de rejeição nula para os cenários 2, 3, 4 e 5, respectivamente. Podemos observar que o mesmo padrão obtido no cenário 1 se repete de maneira similar: no cenário 2, por exemplo, com $\alpha = 10\%$ e $n = 20$, a taxa de rejeição nula do TRV chega a $13,52\%$, enquanto as correções ficam em torno de $10,78\%$. No cenário 3, para $\alpha = 5\%$ e $n = 20$, o TRV é de $7,72\%$ contra cerca de $4,7\%$ nos testes Boot1 e Boot2. Já no cenário 4, com $\alpha = 1\%$ e $n = 20$, o TRV apresenta taxa de $2,26\%$, mais que o dobro do nível nominal, ao passo que as correções se mantêm próximas a $1,4\%$ e $1,3\%$. Por fim, no cenário 5, também em $\alpha = 1\%$ e $n = 20$, o TRV ultrapassa $3,5\%$, enquanto Boot1 e Boot2 apontam taxas não tão próximas como nos cenários anteriores (em torno de $0,6\%$),

Tabela 4.1: Taxa de rejeição nula no cenário 1.

	$\alpha = 10\%$				
Teste	$n = 20$	$n = 30$	$n = 40$	$n = 50$	$n = 100$
TRV	13,70	13,44	13,10	12,56	12,10
Boot1	9,72	9,81	10,15	10,17	10,20
Boot2	9,62	9,37	9,68	9,91	10,15
	$\alpha = 5\%$				
Teste	$n = 20$	$n = 30$	$n = 40$	$n = 50$	$n = 100$
TRV	8,56	7,56	6,92	6,74	6,11
Boot1	4,74	4,89	5,22	5,25	5,10
Boot2	4,62	4,54	4,70	5,21	4,91
	$\alpha = 1\%$				
Teste	$n = 20$	$n = 30$	$n = 40$	$n = 50$	$n = 100$
TRV	3,84	2,90	1,92	1,53	1,21
Boot1	0,94	1,20	1,11	0,96	1,02
Boot2	0,96	0,91	1,02	1,04	1,02

mas ainda assim bem melhores em comparação com o TRV, ressaltando que os testes corrigidos também melhoram conforme o tamanho amostral cresce. Portanto, em todos os cenários, o TRV usual segue mais liberal para tamanhos amostrais pequenos, ao passo que os testes corrigidos (Boot1 e Boot2) mantêm suas taxas de rejeição nula muito mais alinhadas ao nível nominal, evidenciando maior adequação, a qual ocorre desde amostras reduzidas até amostras maiores, superando o desempenho do TRV em todos os casos.

Tabela 4.2: Taxa de rejeição nula no cenário 2.

	$\alpha = 10\%$				
Teste	$n = 20$	$n = 30$	$n = 40$	$n = 50$	$n = 100$
TRV	13,52	12,60	11,26	12,01	11,56
Boot1	10,77	10,42	10,21	10,15	10,04
Boot2	10,79	10,50	10,22	10,21	10,11
	$\alpha = 5\%$				
Teste	$n = 20$	$n = 30$	$n = 40$	$n = 50$	$n = 100$
TRV	7,24	6,90	5,76	6,22	6,16
Boot1	5,72	5,92	5,57	5,31	5,15
Boot2	5,62	5,61	5,62	5,28	5,12
	$\alpha = 1\%$				
Teste	$n = 20$	$n = 30$	$n = 40$	$n = 50$	$n = 100$
TRV	1,82	1,68	1,30	1,80	1,54
Boot1	1,15	1,15	1,12	1,07	1,03
Boot2	1,14	1,12	1,09	1,08	1,02

Tabela 4.3: Taxa de rejeição nula no cenário 3.

	$\alpha = 10\%$				
Teste	$n = 20$	$n = 30$	$n = 40$	$n = 50$	$n = 100$
TRV	14,46	12,84	12,14	11,92	10,88
Boot1	9,21	9,38	9,82	10,15	10,21
Boot2	9,37	9,39	9,77	9,86	10,12
	$\alpha = 5\%$				
Teste	$n = 20$	$n = 30$	$n = 40$	$n = 50$	$n = 100$
TRV	7,72	6,70	6,72	6,40	5,66
Boot1	4,76	4,81	4,85	5,21	5,08
Boot2	4,71	4,79	4,86	5,16	5,12
	$\alpha = 1\%$				
Teste	$n = 20$	$n = 30$	$n = 40$	$n = 50$	$n = 100$
TRV	1,88	1,54	1,64	1,44	1,18
Boot1	1,32	1,19	0,96	0,96	0,99
Boot2	1,10	1,05	0,97	0,96	0,98

Tabela 4.4: Taxa de rejeição nula no cenário 4.

	$\alpha = 10\%$				
Teste	$n = 20$	$n = 30$	$n = 40$	$n = 50$	$n = 100$
TRV	14,20	13,26	12,80	11,52	10,58
Boot1	11,00	10,89	10,23	10,12	10,02
Boot2	10,98	10,91	10,41	10,20	10,11
	$\alpha = 5\%$				
Teste	$n = 20$	$n = 30$	$n = 40$	$n = 50$	$n = 100$
TRV	8,14	7,44	6,68	5,82	5,24
Boot1	5,74	5,88	5,34	5,27	5,13
Boot2	5,97	5,90	5,81	5,35	5,19
	$\alpha = 1\%$				
Teste	$n = 20$	$n = 30$	$n = 40$	$n = 50$	$n = 100$
TRV	2,26	1,78	1,36	1,54	1,21
Boot1	1,41	1,36	1,35	1,35	1,03
Boot2	1,27	1,25	1,22	1,14	1,02

Tabela 4.5: Taxa de rejeição nula no cenário 5.

	$\alpha = 10\%$				
Teste	$n = 20$	$n = 30$	$n = 40$	$n = 50$	$n = 100$
TRV	18,26	15,54	13,00	13,40	11,04
Boot1	9,50	9,71	9,53	9,84	9,90
Boot2	9,53	9,55	9,61	9,69	9,87
	$\alpha = 5\%$				
Teste	$n = 20$	$n = 30$	$n = 40$	$n = 50$	$n = 100$
TRV	10,86	8,76	7,12	7,02	5,22
Boot1	4,68	4,71	4,81	4,84	4,96
Boot2	4,61	4,64	4,72	4,83	4,89
	$\alpha = 1\%$				
Teste	$n = 20$	$n = 30$	$n = 40$	$n = 50$	$n = 100$
TRV	3,54	2,08	1,74	1,62	1,26
Boot1	0,57	0,62	0,79	0,88	1,02
Boot2	0,55	0,59	0,71	0,85	0,99

4.2.2 Taxa de Rejeição Não-Nula (Poder do Teste)

Para todos os cenários, também realizamos um estudo comparando a taxa de rejeição sob a hipótese alternativa (poder) ao invés da taxa de rejeição sob a hipótese nula, variando apenas os valores dos coeficientes $\beta_{13} = \beta_{23} = \tau$ para os quatro primeiros cenários e $\gamma_3 = \tau$ para o cenário 5, sendo $\tau = -0,50, -0,25, 0,25$ e $0,50$. Nesse estudo, consideramos dois níveis de significância (5% e 10%) e dois tamanhos amostrais (50 e 100).

Nas cinco tabelas a seguir (4.6 a 4.10) apresentamos os resultados para avaliar as taxas de rejeição não nulas (poder) do teste (4.4) para todos os cenários com base nos testes corrigidos Boot1 e Boot2. A estatística do TRV não foi incluída por apresentar taxa de rejeição não nula muito superior ao nível nominal. Conforme esperado, o poder do teste aumentou a medida que o tamanho amostral e o valor absoluto de τ aumentaram, ou seja, amostras maiores e efeitos mais acentuados tendem a aumentar a taxa de rejeição sob a hipótese alternativa, resultando em um maior poder dos testes. Além disso, ao comparar o desempenho das duas estatísticas, notamos que as taxas de rejeição não nula para Boot1 e para Boot2 são muito próximas em todos os cenários. Note que para $n = 50$ o poder do teste para $\tau = j$ e $\tau = -j$ não apresenta valores muito simétricos nos cenários 1 e 3, o que também ocorre em outros trabalhos similares (Magalhães *et al.*, 2024). Já para $n = 100$, o comportamento é mais próximo de simétrico em todos os cenários, conforme previsto.

Tabela 4.6: Taxa de rejeição não-nula no cenário 1.

	$n = 50$			
	$\alpha = 5\%$		$\alpha = 10\%$	
τ	Boot1	Boot2	Boot1	Boot2
$-0,50$	$75,8$	$74,9$	$88,3$	$88,0$
$-0,25$	$26,4$	$26,2$	$39,1$	$38,5$
$0,25$	$30,6$	$29,8$	$40,2$	$39,7$
$0,50$	$87,2$	$87,1$	$91,9$	$91,5$
	$n = 100$			
	$\alpha = 5\%$		$\alpha = 10\%$	
τ	Boot1	Boot2	Boot1	Boot2
$-0,50$	$97,2$	$97,1$	$99,1$	$99,0$
$-0,25$	$50,1$	$49,4$	$61,7$	$61,5$
$0,25$	$51,3$	$49,8$	$64,5$	$64,1$
$0,50$	$98,9$	$98,6$	$99,5$	$99,5$

Tabela 4.7: Taxa de rejeição não-nula no cenário 2.

	$n = 50$			
	$\alpha = 5\%$		$\alpha = 10\%$	
τ	Boot1	Boot2	Boot1	Boot2
$-0,50$	$80,1$	$79,9$	$87,0$	$87,0$
$-0,25$	$27,0$	$27,9$	$41,5$	$41,7$
$0,25$	$28,4$	$28,8$	$40,9$	$41,4$
$0,50$	$86,2$	$85,1$	$90,9$	$90,4$
	$n = 100$			
	$\alpha = 5\%$		$\alpha = 10\%$	
τ	Boot1	Boot2	Boot1	Boot2
$-0,50$	$97,9$	$97,2$	$98,9$	$99,1$
$-0,25$	$49,2$	$49,4$	$65,3$	$65,5$
$0,25$	$51,9$	$50,8$	$65,9$	$66,2$
$0,50$	$97,4$	$98,0$	$99,0$	$99,5$

Tabela 4.8: Taxa de rejeição não-nula no cenário 3.

	$n = 50$			
	$\alpha = 5\%$		$\alpha = 10\%$	
τ	Boot1	Boot2	Boot1	Boot2
$-0,50$	$79,1$	$78,5$	$88,0$	$88,0$
$-0,25$	$26,1$	$26,0$	$40,6$	$40,0$
$0,25$	$29,9$	$29,1$	$40,9$	$41,1$
$0,50$	$87,2$	$86,7$	$91,0$	$90,4$
	$n = 100$			
	$\alpha = 5\%$		$\alpha = 10\%$	
τ	Boot1	Boot2	Boot1	Boot2
$-0,50$	$97,4$	$97,2$	$99,1$	$99,1$
$-0,25$	$48,9$	$48,9$	$61,1$	$61,0$
$0,25$	$52,4$	$51,8$	$64,8$	$64,9$
$0,50$	$98,6$	$98,2$	$99,5$	$99,1$

Tabela 4.9: Taxa de rejeição não-nula no cenário 4.

	$n = 50$			
	$\alpha = 5\%$		$\alpha = 10\%$	
τ	Boot1	Boot2	Boot1	Boot2
$-0,50$	$87,2$	$87,5$	$92,0$	$92,5$
$-0,25$	$36,9$	$37,5$	$50,7$	$50,8$
$0,25$	$36,5$	$36,8$	$49,8$	$50,3$
$0,50$	$87,9$	$87,1$	$92,7$	$92,7$
	$n = 100$			
	$\alpha = 5\%$		$\alpha = 10\%$	
τ	Boot1	Boot2	Boot1	Boot2
$-0,50$	$99,4$	$99,5$	$99,9$	$99,9$
$-0,25$	$64,9$	$64,6$	$75,9$	$75,8$
$0,25$	$71,1$	$71,3$	$81,2$	$81,4$
$0,50$	$99,8$	$99,9$	$99,9$	$99,9$

Tabela 4.10: Taxa de rejeição não-nula no cenário 5.

	$n = 50$			
	$\alpha = 5\%$		$\alpha = 10\%$	
τ	Boot1	Boot2	Boot1	Boot2
$-0,50$	$78,5$	$78,6$	$87,2$	$87,3$
$-0,25$	$27,5$	$27,8$	$39,0$	$39,4$
$0,25$	$28,1$	$28,6$	$42,1$	$42,9$
$0,50$	$83,8$	$84,7$	$90,9$	$91,0$
	$n = 100$			
	$\alpha = 5\%$		$\alpha = 10\%$	
τ	Boot1	Boot2	Boot1	Boot2
$-0,50$	$95,8$	$95,9$	$98,8$	$98,7$
$-0,25$	$52,8$	$54,2$	$67,4$	$67,6$
$0,25$	$55,2$	$55,3$	$69,0$	$69,5$
$0,50$	$98,8$	$98,8$	$99,8$	$99,8$

Capítulo 5

Aplicação em dados sobre estágios do sono

5.1 Introdução ao tema

O sono é uma atividade biológica essencial que influencia diretamente o desempenho físico e mental, contribuindo para a saúde e a qualidade de vida. Durante o sono, o corpo passa por quatro estágios principais: N1, N2, N3 e REM, cada um com funções específicas e complementares para a recuperação e manutenção do organismo. Os estágios N1 e N2 correspondem ao sono leve, em que ocorre uma desaceleração gradual de sinais vitais, preparando o corpo para um sono mais profundo (Hussain *et al.*, 2022). O estágio N3 é caracterizado pelo sono profundo, também chamado de sono de ondas lentas, essencial para a recuperação física, pois nesse momento ocorre a regeneração muscular e de tecidos, fortalecendo o sistema imunológico (Hussain *et al.*, 2022).

A fase REM, por outro lado, é marcada por uma atividade cerebral intensa, acompanhada de movimentos rápidos dos olhos e relaxamento muscular. Essa fase não apenas facilita a consolidação da memória, mas também desempenha um papel crucial no processamento emocional. Estudos recentes demonstram que o sono REM está intimamente ligado à regulação e à memória emocional, pois sua privação pode impactar negativamente a reatividade emocional e a capacidade de processar e consolidar memórias emocionais (Tempesta *et al.*, 2018).

Para um ciclo de sono saudável, é necessário que cada estágio ocorra em proporções equilibradas. O sono NREM, que inclui os estágios N1, N2 e N3, representa aproximadamente 75% do tempo total de sono, enquanto o sono REM compõe cerca de 25% desse

tempo (Hussain *et al.*, 2022). A distribuição ideal dos estágios do sono é de 3-8% para N1, 45-55% para N2, 15-20% para N3 e 20-25% para REM, o que garante um equilíbrio adequado entre recuperação física e processamento emocional (González-Naranjo *et al.*, 2019). Esse equilíbrio é fundamental para que o sono desempenhe seu papel protetor e restaurador, tanto para o corpo quanto para a mente.

Veje *et al.* (2021) coletaram dados de sono de 42 adultos, dos quais 22 eram pacientes com encefalite transmitida por carrapatos (TBE) e 20 controles. O seu principal objetivo foi investigar se o TBE afeta a qualidade do sono medida por um questionário. No nosso caso, utilizaremos um modelo de regressão Dirichlet para estudar se a proporção de tempo gasto em cada estágio do sono muda com o tempo de sono total (TST) e com pacientes com TBE para adultos saudáveis. Ao final, avaliaremos como as correções propostas para o TRV desempenham no modelo ajustado final. Esses dados foram considerados por Pereira e Cai (2024) para ilustrar o resíduo proposto por eles.

5.2 Análise descritiva dos dados

Para termos indicativos sobre o comportamento das variáveis em estudo e da relação entre elas, realizamos uma análise descritiva. Como vamos ajustar um modelo de regressão Dirichlet aos dados, o foco dessa análise é ter indícios da significância das covariáveis, o que será testado posteriormente.

Como temos duas covariáveis, sendo uma contínua e outra categórica, dividiremos a análise em duas frentes, iniciando pela variável contínua. Na Figura 5.1, notamos que o componente N2 apresenta uma concentração de percentuais maiores em relação aos outros três componentes da variável resposta, porém não conseguimos ter um indicativo de tendência ou relação entre os percentuais de qualidade do sono e o tempo de sono total.

Assim, buscando entender melhor essa relação, categorizamos a variável contínua para construir um gráfico de média por categoria (Figura 5.2). Foram criadas quatro categorias a partir dos quartis da covariável, sendo a categoria tempo de sono total baixo de 3,8 até 5,6 horas, médio-baixo de 5,7 até 6,4 horas, médio-alto de 6,5 até 7,2 horas e alto de 7,3 até 8,7 horas. Por este gráfico conseguimos notar que, para os componentes N1 e N2, conforme aumentamos a categoria do tempo de sono total, o percentual médio desses componentes decresce. Já para os componentes N3 e REM, notamos um comportamento contrário ao N1 e N2, isto é, conforme aumentamos a categoria do tempo de sono total,

o percentual médio desses componentes cresce, nos dando um indicativo de que quanto maior a duração do sono, maior o percentual médio desses estágios.

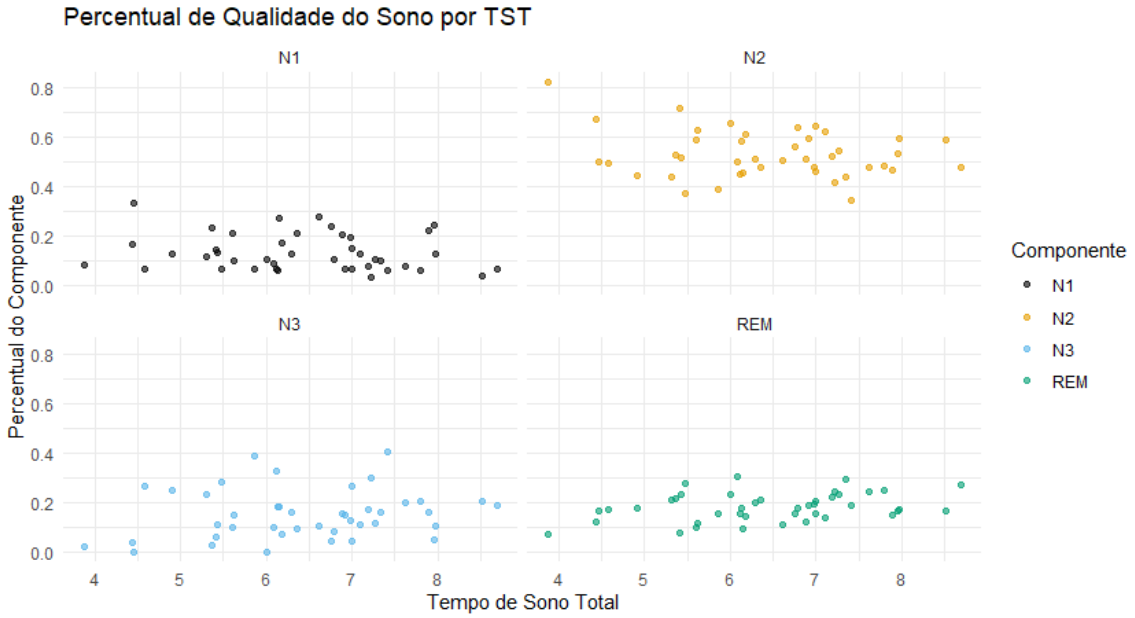

Figura 5.1: Gráfico de dispersão do percentual de cada componente da variável resposta pelo tempo de sono total.

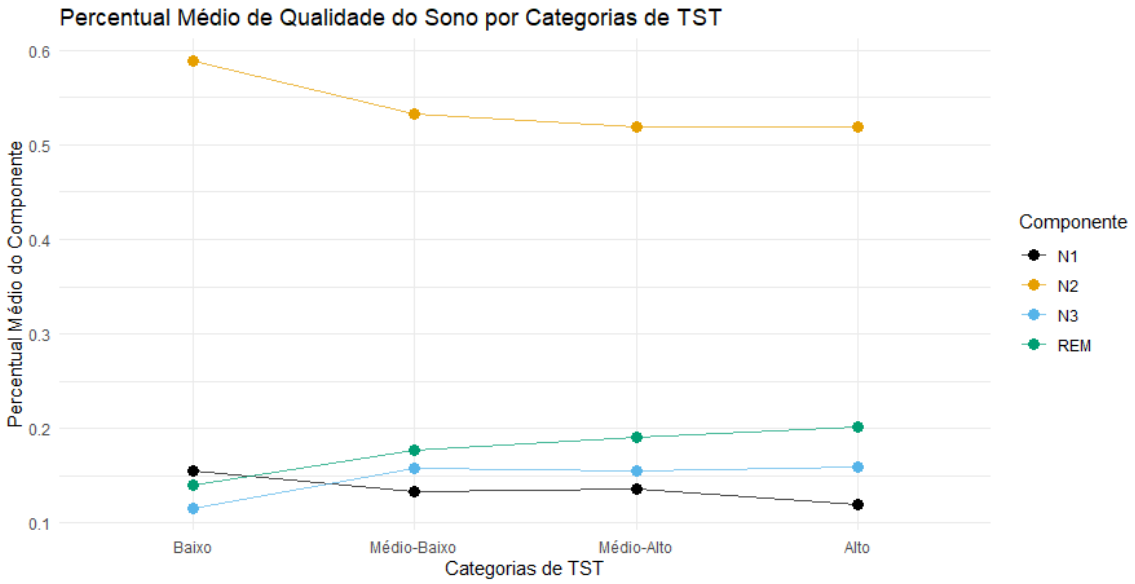

Figura 5.2: Gráfico de dispersão do percentual médio de cada componente da variável resposta por categorias do tempo de sono total.

Agora, partindo para a análise da variável categórica indicadora do grupo caso (adultos com encefalite transmitida por carrapatos) ou grupo controle (adultos saudáveis), podemos observar a partir dos boxplots na Figura 5.3 que não há grandes diferenças entre os gráficos de um mesmo componente ao comparar os grupos caso-controle, visto que em todos esses casos as medianas e o tamanho das caixas de cada grupo estão bem próximos,

isto é, ao passar do grupo portador da doença para o grupo saudável, o percentual médio gasto em cada estágio do sono não parece sofrer alterações significativas.

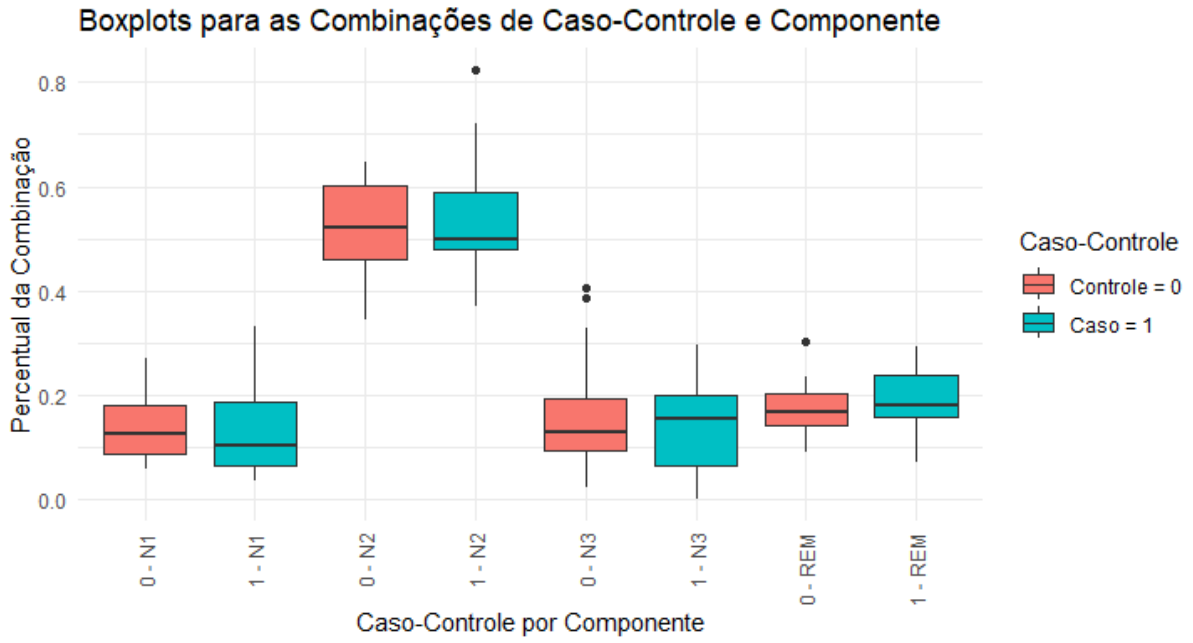

Figura 5.3: Boxplots para cada combinação entre os grupos caso-controle e os componentes da resposta.

Em resumo, essa análise nos fornece indícios de que aumentar o tempo de sono total tem efeitos diferentes para cada estágio do sono, sendo que quanto maior esse tempo, mais os percentuais de cada estágio vão se aproximando das proporções ideais para um ciclo de sono saudável (conforme citado na introdução). Além disso, a divisão dos indivíduos em grupos caso e controle não parece ter efeito relevante, pois ambos os grupos apresentam comportamentos semelhantes dentro dos quatro estágios do sono.

5.3 Testes de Hipóteses

Feita a análise descritiva, agora iremos realizar testes de hipóteses para ter evidências a respeito dos indicativos obtidos, testando a significância de cada covariável no modelo de regressão Dirichlet ajustado.

Inicialmente, definimos o seguinte modelo completo que será utilizado para testar a

significância da covariável categórica (TBE):

$$
\begin{aligned}
log\left(\frac{\mu_{i2}}{\mu_{i1}}\right) &= \beta_{11} + \beta_{12} \cdot TST + \beta_{13} \cdot TBE && (5.1)\\
log\left(\frac{\mu_{i3}}{\mu_{i1}}\right) &= \beta_{21} + \beta_{22} \cdot TST + \beta_{23} \cdot TBE && (5.2)\\
log\left(\frac{\mu_{i4}}{\mu_{i1}}\right) &= \beta_{31} + \beta_{32} \cdot TST + \beta_{33} \cdot TBE && (5.3)\\
log(\phi_i) &= \gamma_1 + \gamma_2 \cdot TST + \gamma_3 \cdot TBE && (5.4)
\end{aligned}
$$

Para cada covariável, realizaremos três testes diferentes. O teste para todos os parâmetros relacionados à covariável ($k = 2$ ou $k = 3$):

$$
\begin{aligned}
&H_0 : \beta_{1k} = 0, \beta_{2k} = 0, \beta_{3k} = 0, \gamma_k = 0 && (5.5)\\
&H_1 : \text{Violação de pelo menos uma igualdade.}
\end{aligned}
$$

O teste para os parâmetros dos modelos das médias relacionados à covariável ($k = 2$ ou $k = 3$):

$$
\begin{aligned}
&H_0 : \beta_{1k} = 0, \beta_{2k} = 0, \beta_{3k} = 0 && (5.6)\\
&H_1 : \text{Violação de pelo menos uma igualdade.}
\end{aligned}
$$

O teste para o parâmetro do modelo de precisão relacionado à covariável ($k = 2$ ou $k = 3$):

$$
\begin{aligned}
&H_0 : \gamma_k = 0 && (5.7)\\
&H_1 : \gamma_k \neq 0
\end{aligned}
$$

Definidos os testes, aplicamos o Teste da Razão de Verossimilhanças (TRV) em cada um dos cenários e, com base na Tabela 5.1, temos evidências ao nível de significância de 5% de que a covariável TBE não é significativa para o modelo, ou seja, há evidências a respeito do indicativo obtido na análise descritiva de que a divisão dos indivíduos em grupos caso e controle não tem efeito relevante para o estudo.

Tabela 5.1: Resultados do TRV para os três testes realizados.

Testes	Valor-p TRV
Teste (5.5)	0,74692
Teste (5.6)	0,65575
Teste (5.7)	0,63265

Agora, realizaremos os testes para a covariável tempo de sono total (TST) considerando o mesmo modelo completo definido anteriormente, porém sem a presença da covariável TBE. Para essa covariável, além do TRV, também aplicaremos as duas correções baseadas em bootstrap, sendo a correção de Bartlett (Boot1) e a correção no valor crítico (Boot2).

Com base na Tabela 5.2, temos evidências ao nível de significância de 5% de que a covariável TST é significativa para o modelo, ou seja, há evidências de que o vetor de médias e o parâmetro de precisão da variável resposta qualidade do sono são funções do tempo total de sono, mais uma vez corroborando com o indicativo obtido na análise descritiva de que o TST tem efeitos diferentes para cada estágio do sono.

Tabela 5.2: Resultados do TRV e das duas correções para os três testes realizados.

Testes	Valor-p TRV	Valor-p Boot1	Valor-p Boot2
Teste (5.5)	0,00008	0,00014	0,00099
Teste (5.6)	0,00918	0,01191	0,01298
Teste (5.7)	0,00299	0,00523	0,00499

Como os valores-p obtidos nesse teste foram muito baixos, não foi possível observar diferença entre os resultados do teste TRV e dos testes corrigidos baseados em bootstrap. Por isso, faremos uma análise para demonstrar que na prática os resultados desses testes podem ser diferentes.

Nessa análise, em cada iteração sorteamos aleatoriamente uma amostra de tamanho 15 (sem reposição) do conjunto de dados original e aplicamos os três testes (TRV, Boot1 e Boot2) aos dados da amostra. Repetindo esse processo 500 vezes, obtemos as taxas de rejeição da hipótese nula nos níveis de 1%, 5% e 10%. A Tabela 5.3 apresenta o percentual de amostras da análise em que a hipótese nula foi rejeitada para os três testes considerados neste trabalho. Note que esses percentuais são bem maiores para o TRV do que para os testes baseados em bootstrap. Isso indica que, se o tamanho da amostra fosse igual a 15, seria bem provável que o TRV rejeitasse a hipótese nula enquanto os demais não rejeitassem H_0. Vale ressaltar também que as hipóteses testadas dizem respeito aos parâmetros de média do modelo Dirichlet, conforme definido em (5.6).

Tabela 5.3: Percentual de amostras da análise em que H_0 foi rejeitada.

	1%	5%	10%
TRV	32,0	65,6	86,0
Boot1	8,0	25,2	34,8
Boot2	8,2	25,6	34,2

5.4 Análise de Diagnóstico

Por fim, para avaliar se o modelo ajustado está adequado aos dados, reproduzimos a análise de diagnóstico realizada em Pereira e Cai (2024) utilizando o resíduo proposto por esses autores.

Com base nesse resíduo, construímos um gráfico de probabilidade normal com envelope simulado proposto por Atkinson (1981), apresentado na Figura 5.4. Observamos que todos os pontos estão dentro da banda simulada, sugerindo não haver indícios de que os resíduos tenham distribuição diferente da normal padrão. Assim, o comportamento do gráfico indica consistência do ajuste do modelo aos dados, reforçando a adequação da distribuição escolhida para a variável resposta.

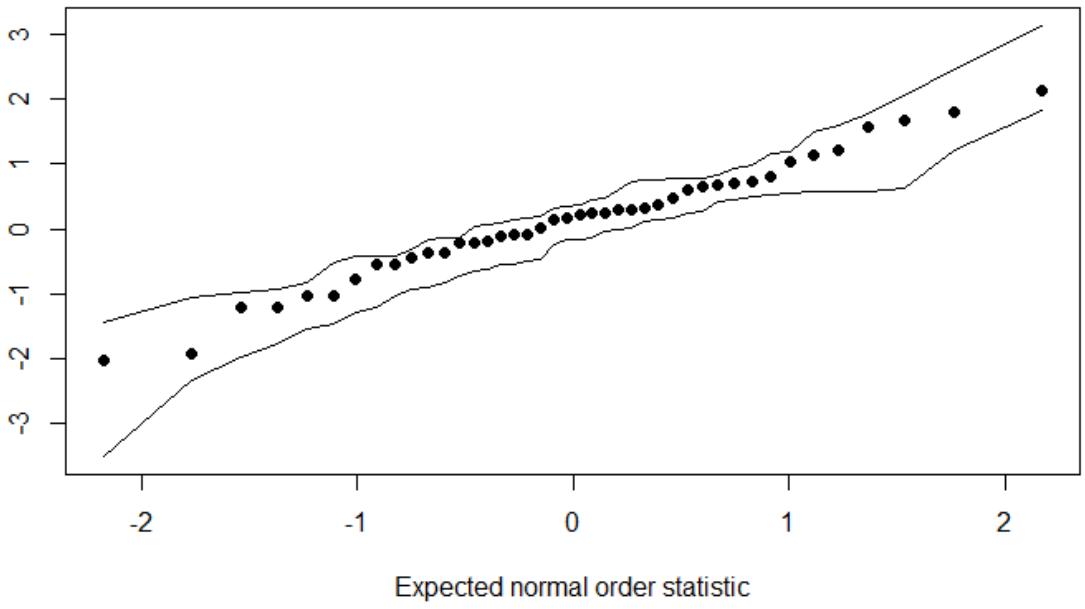

Figura 5.4: Gráfico de probabilidade normal com envelope simulado considerando o modelo final para os dados sobre estágios do sono.

Capítulo 6

Considerações Finais

Dados composicionais em forma de porcentagens ou proporções são muito comuns em diversas áreas da ciência e uma das principais formas de modelar esse tipo de dado é usando um modelo de regressão Dirichlet. Geralmente, na grande maioria dos modelos de regressão, o Teste da Razão de Verossimilhanças (TRV) é o mais utilizado para realizar testes de hipóteses, especialmente quando a hipótese nula envolve mais de um parâmetro simultaneamente. Entretanto, o TRV é consideravelmente liberal nesse tipo de modelagem quando o tamanho da amostra é pequeno. Neste trabalho, propomos duas correções para o TRV baseadas em bootstrap aplicadas ao modelo de regressão Dirichlet com parametrização alternativa, uma utilizando a técnica computacional para corrigir diretamente a fórmula da estatística do teste e outra focando em calcular o valor-p diretamente da distribuição empírica das estatísticas estimadas. Foram desenvolvidos estudos de simulação de Monte Carlo em cinco diferentes cenários, buscando explorar diferentes variações tanto nos parâmetros do modelo, como na distribuição das covariáveis, mostrando que as taxas de rejeição nula dos testes corrigidos estão bem próximas dos níveis nominais para tamanhos amostrais pequenos e também observando um desempenho superior em tamanhos amostrais moderados. Também concluímos que os dois testes de razão de verossimilhanças corrigidos têm poder semelhante. Por fim, uma aplicação mostrou a utilidade dos testes corrigidos em um cenário real.

Modelos de regressão Dirichlet e correções Bartlett-Bootstrap estão presentes em diversas contribuições da literatura e um dos objetivos atingidos por este trabalho foi mostrar que os testes de razão de verossimilhanças corrigidos têm um desempenho superior quando comparados ao TRV usual, sendo úteis quando se deseja testar a significância de parâmetros em uma modelagem envolvendo a distribuição Dirichlet com parametrização

alternativa, características essas que são exclusivas deste trabalho.

Com base nos resultados dos estudos de simulação e da aplicação realizada, sugerimos que o teste corrigido definido como Boot1 seja utilizado quando o tamanho da amostra disponível for pequeno e houver o desejo de realizar testes de hipóteses envolvendo modelos de regressão Dirichlet. Essa recomendação é baseada no fato de que ambos os testes corrigidos obtiveram desempenhos semelhantes em relação à taxa de rejeição sob a hipótese nula e também sob a hipótese alternativa. Porém, a forma como a técnica de bootstrap é utilizada para calcular a correção de Bartlett em Boot1 exige menos réplicas do que a correção definida em Boot2, tendo assim um ganho computacional significativo, mostrado por Rocke (1989).

Para trabalhos futuros, podem ser desenvolvidos estudos para investigar, por exemplo, de que forma a correção de Bartlett analítica se compara às correções baseadas em bootstrap no contexto do modelo de regressão Dirichlet com parametrização alternativa. Além disso, pode-se estender o presente estudo para outros modelos de regressão apropriados a dados composicionais, bem como para modelos de regressão multivariados que lidem com respostas não composicionais, a fim de avaliar o desempenho e a aplicabilidade dessas correções nesses diferentes cenários.

Referências Bibliográficas

Atkinson, A. C. (1981). Two graphical displays for outlying and influential observations in regression. *Biometrika*, **68**(1), 13–20.

Bengtsson, H. (2020). A unifying framework for parallel and distributed processing in r using futures. *arXiv preprint arXiv:2008.00553*.

Botter, D. A. (1993). *Correcoes de bartlett e poderes de testes em algumas classes de modelos de regressao*. Tese de doutorado, Instituto de Matemática e Estatística, Universidade de São Paulo.

Cordeiro, G. M. e Cribari-Neto, F. (2014). *An introduction to Bartlett correction and bias reduction*. Springer, Berlin.

Cordeiro, G. M., Demétrio, C. G. B. e Moral, R. d. A. (2024). *Modelos lineares generalizados e aplicações*. Blucher.

Da-Silva, C. Q. e Rodrigues, G. S. (2015). Bayesian dynamic dirichlet models. *Communications in Statistics-Simulation and Computation*, **44**(3), 787–818.

Efron, B. (1979). Bootstrap methods: Another look at the jackknife. *The Annals of Statistics*, **7**(1), 1–26.

Ferrari, S. e Cribari-Neto, F. (2004). Beta regression for modelling rates and proportions. *Journal of applied statistics*, **31**(7), 799–815.

González-Naranjo, J. E., Alfonso-Alfonso, M., Grass-Fernandez, D., Morales-Chacón, L. M., Pedroso-Ibáñez, I., Ricardo-de la Fe, Y. e Padrón-Sánchez, A. (2019). Analysis of sleep macrostructure in patients diagnosed with parkinson's disease. *Behavioral Sciences*, **9**(1), 6.

Gueorguieva, R., Rosenheck, R. e Zelterman, D. (2008). Dirichlet component regression and its applications to psychiatric data. *Computational statistics & data analysis*, **52**(12), 5344–5355.

Hanretty, C. (2021). Forecasting multiparty by-elections using dirichlet regression. *International Journal of Forecasting*, **37**(4), 1666–1676.

Hijazi, R. H. e Jernigan, R. W. (2009). Modelling compositional data using dirichlet regression models. *Journal of Applied Probability & Statistics*, **4**(1), 77–91.

Hussain, I., Hossain, M. A., Jany, R., Bari, M. A., Uddin, M., Kamal, A. R. M., Ku, Y. e Kim, J.-S. (2022). Quantitative evaluation of eeg-biomarkers for prediction of sleep stages. *Sensors*, **22**(8), 3079.

Lawley, D. N. (1956). A general method for approximating to the distribution of likelihood ratio criteria. *Biometrika*, **43**(3/4), 295–303.

Magalhães, T. M., Pereira, G. H., Botter, D. A. e Sandoval, M. C. (2024). Bartlett corrections for zero-adjusted generalized linear models. *Statistical Papers*, **65**(4), 2191–2209.

Maier, M. J. (2014). Dirichletreg: Dirichlet regression for compositional data in r. *Research Report Series / Department of Statistics and Mathematics*, **125**.

Melo, T. F., Vargas, T. M., Lemonte, A. J. e Moreno-Arenas, G. (2022). Higher-order asymptotic refinements in the multivariate dirichlet regression model. *Communications in Statistics-Simulation and Computation*, **51**(1), 53–71.

Morais, J., Thomas-Agnan, C. e Simioni, M. (2018). Using compositional and dirichlet models for market share regression. *Journal of Applied Statistics*, **45**(9), 1670–1689.

Norton, E. C. e Dowd, B. E. (2018). Log odds and the interpretation of logit models. *Health services research*, **53**(2), 859–878.

Pereira, G. H. A. e Cai, J. (2024). A class of bootstrap based residuals for compositional data. *arXiv*, **2403.13544**.

Prado, N. V. d. (2017). *Abordagens para análise de dados composicionais*. Tese de doutorado, Instituto de Matemática e Estatística, Universidade de São Paulo.

Rocke, D. M. (1989). Bootstrap bartlett adjustment in seemingly unrelated regression. *Journal of the American Statistical Association*, **84**(406), 598–601.

Tempesta, D., Socci, V., De Gennaro, L. e Ferrara, M. (2018). Sleep and emotional processing. *Sleep medicine reviews*, **40**, 183–195.

Veje, M., Studahl, M., Thunström, E., Stentoft, E., Nolskog, P., Celik, Y. e Peker, Y. (2021). Sleep architecture, obstructive sleep apnea and functional outcomes in adults with a history of tick-borne encephalitis. *PLoS One*, **16**(2), e0246767.

Apêndice A

Pacote *DirichletReg* no *R*

O pacote DirichletReg que foi utilizado neste trabalho fornece funções para analisar dados composicionais usando métodos de regressão Dirichlet.

A utilização do pacote parte do princípio inicial de que temos um banco de dados com as variáveis composicionais e as covariáveis a serem consideradas no modelo de regressão, como por exemplo com as colunas y_1, y_2 e y_3 que são as variáveis composicionais e x_1 e x_2 que são as covariáveis. Para realizar o ajuste do modelo, basta seguir os seguintes passos:

1. Utilizar a função auxiliar *DR_data* para preparar as variáveis composicionais para o processamento posterior. Essa função recebe um parâmetro Y que consiste em uma matriz ou um DataFrame com as variáveis composicionais (y_1, y_2 e y_3 no nosso exemplo). Caso a soma das linhas de Y não forem exatamente iguais a um, essa função realiza uma normalização nos dados para suprir essa condição;

2. Ajustar o modelo com a função principal *DirichReg*, passando como parâmetros uma fórmula comum de regressão ($Y \sim x_1 + x_2$, por exemplo), o banco de dados conforme definido anteriormente e qual parametrização da distribuição Dirichlet que será utilizada, a comum ou a alternativa.

Feito o ajuste, há várias particularidades do modelo que podem ser obtidas (ver *?DirichReg* no *R*), dentre elas a log-verossimilhança, a qual é utilizada para realizar o teste da razão de verossimilhanças.

Abaixo, segue uma implementação com base no exemplo que foi dado durante a explicação do pacote, realizando a aplicação de um TRV:

```r
# Carregando o pacote
library(DirichletReg)

# Preparando as variaveis composicionais
dados$Y = DR_data(dados[, 1:3])

# Ajustando o modelo completo
ajuste1 <- DirichReg(Y ~ x1 + x2, data = dados, model = 'alternative')
# Ajustando o modelo reduzido
ajuste2 <- DirichReg(Y ~ x1, data = dados, model = 'alternative')

# Obtendo a log-verossimilhanca dos modelos ajustados
lveros_completo <- ajuste1$logLik
lveros_reduzido <- ajuste2$logLik

# Calculando estatistica de teste e valor-p
estat_teste <- 2 * (lveros_completo - lveros_reduzido)
valorp_TRV <- 1 - pchisq(estat_teste, df = 2)
```

Apêndice B

Códigos

Todos os códigos que foram desenvolvidos para esse trabalho (simulações e aplicação) estão disponíveis no GitHub https://github.com/ViniMorgado1/Codigos_TCC.git.

Para facilitar o acesso e a compreensão dos códigos apresentados neste relatório, eles também serão disponibilizados a seguir.

B.1 Simulações

Para as simulações, foi desenvolvida uma função que recebe como parâmetros o tamanho amostral, o número de réplicas de Monte Carlo e o número de réplicas de bootstrap. O código apresentado abaixo refere-se ao Cenário 1, enquanto os demais cenários estão indicados como comentários no próprio código. Para calcular a taxa de rejeição não nula (ou o poder do teste), basta ajustar os valores dos betas no começo da função.

```
library(future.apply)
library(DirichletReg)

# Funcao a ser executada para cada tamanho amostral
simulacao <- function(n, rep=5000, B=1000){
  # Definindo a semente principal
  set.seed(8261)

  # Definindo o ambiente de paralelizacao
  future::plan(multisession)

  # Definindo os betas arbitrarios (muda de acordo com o cenario)
  ## Cenarios 1 e 4
```

```r
betas <- matrix(data=c(-0.5,1.1,0,
                       -0.5,1.1,0,
                       4.6,0,0),
                nrow = 3,ncol = 3,byrow = TRUE)
## Cenario 2
# betas <- matrix(data=c(-1,1.1,0,
#                        -1.5,1.1,0,
#                        4.6,0,0),
#                 nrow = 3,ncol = 3,byrow = TRUE)
## Cenario 3
# betas <- matrix(data=c(-0.5,1.1,0,
#                        -0.5,1.1,0,
#                        2.3,0,0),
#                 nrow = 3,ncol = 3,byrow = TRUE)
## Cenario 5
# betas <- matrix(data=c(-0.5,1.1,0,
#                        -0.5,1.1,0,
#                        4.6,-2.5,0),
#                 nrow = 3,ncol = 3,byrow = TRUE)

# Definindo a matriz X com as covariaveis (muda de acordo com o
   cenario)
## Cenarios 1, 2, 3 e 5
X <- cbind(1, runif(n), runif(n))
## Cenario 4
# X <- cbind(1, rnorm(n, 1/2, 1/12), rgamma(n, 3, 6))
colnames(X) <- c('x11', 'x12', 'x13')

# Calculando etas do modelo
etas <- X %*% t(betas)
colnames(etas) <- c('eta1', 'eta2', 'eta3')
# Calculando as medias de cada componente
denom <- 1 + exp(etas[, "eta1"]) + exp(etas[, "eta2"])
medias <- cbind(1 / denom,
                exp(etas[, "eta1"]) / denom,
                exp(etas[, "eta2"]) / denom)
# Calculando o parametro de precisao
phi <- exp(etas[,"eta3"])
# Calculando os parametros alpha
alphas <- medias * phi
```

```r

##########################################################
## Loop externo paralelizado (replicas de Monte Carlo) ##
##########################################################
resultados <- future_lapply(1:rep, function(i) {
  # Guia para ver se o codigo esta rodando ou travou
  if (i %% 500 == 0) {
    print(paste("Simulacao", i))
  }

  # Gerando a variavel resposta
  y <- rdirichlet(n, alphas)
  # Criando um dataframe com a resposta e as covariaveis
  dados <- data.frame(y, x12 = X[, "x12"], x13 = X[, "x13"])
  # Tratando a resposta para o ajuste
  dados$y <- DR_data(dados[, 1:3])

  # Ajustando modelos completo e reduzido (muda de acordo com o
     cenario)
  ## Cenarios 1, 2, 3 e 4
  mod_completo <- DirichReg(y ~ x12 + x13, data = dados, model =
     'alternative')
  mod_reduzido <- DirichReg(y ~ x12, data = dados, model =
     'alternative')
  ## Cenario 5
  # mod_completo <- DirichReg(y ~ x12 + x13 | x12 + x13, data =
     dados, model = 'alternative')
  # mod_reduzido <- DirichReg(y ~ x12 | x12, data = dados, model =
     'alternative')

  # Calculando valor-p do TRV
  veros_reduzido <- mod_reduzido$logLik
  veros_completo <- mod_completo$logLik
  estat_teste <- 2 * (veros_completo - veros_reduzido)
  # Para cenario 5 mudar: df = 3
  valorp_trv <- 1 - pchisq(estat_teste, df = 2)

  # Obtendo betas do modelo reduzido (muda de acordo com o cenario)
  b <- as.vector(mod_reduzido$coefficients)
  ## Cenarios 1, 2, 3 e 4
```

```
88      b <- c(b[1:2], 0, b[3:4], 0, b[5], 0, 0)
89      ## Cenario 5
90      # b <- c(b[1:2], 0, b[3:4], 0, b[5:6], 0)
91      # Definindo os betas para bootstrap
92      betas_boot <- matrix(data=b, nrow = 3, ncol = 3, byrow = TRUE)
93
94      # Calculando etas para bootstrap
95      etas_boot <- X %*% t(betas_boot)
96      colnames(etas_boot) <- c('eta1', 'eta2', 'eta3')
97      # Calculando as medias para bootstrap
98      denom_boot <- 1 + exp(etas_boot[, "eta1"]) + exp(etas_boot[,
            "eta2"])
99      medias_boot <- cbind(1 / denom_boot,
100                           exp(etas_boot[, "eta1"]) / denom_boot,
101                           exp(etas_boot[, "eta2"]) / denom_boot)
102     # Calculando o parametro de precisao para bootstrap
103     phi_boot <- exp(etas_boot[,"eta3"])
104     # Calculando os parametros alpha para bootstrap
105     alphas_boot <- medias_boot * phi_boot
106
107     ##########################################
108     ## Loop interno (replicas de Bootstrap) ##
109     ##########################################
110     estatisticas_boot <- sapply(1:B, function(j) {
111       # Gerando a variavel resposta
112       y_boot <- rdirichlet(n, alphas_boot)
113       # Criando um dataframe com a resposta e as covariaveis
114       dados_boot <- data.frame(y_boot, x12 = X[, "x12"], x13 = X[,
              "x13"])
115       # Tratando a resposta para o ajuste
116       dados_boot$y_boot <- DR_data(dados_boot[, 1:3])
117
118       # Ajustando modelos completo e reduzido (muda de acordo com o
              cenario)
119       ## Cenarios 1, 2, 3 e 4
120       mod_completo_boot <- DirichReg(y_boot ~ x12 + x13, data =
              dados_boot, model = 'alternative')
121       mod_reduzido_boot <- DirichReg(y_boot ~ x12, data = dados_boot,
              model = 'alternative')
122       ## Cenario 5
```

```
123       # mod_completo_boot <- DirichReg(y_boot ~ x12 + x13 | x12 + x13,
              data = dados_boot, model = 'alternative')
124       # mod_reduzido_boot <- DirichReg(y_boot ~ x12 | x12, data =
              dados_boot, model = 'alternative')
125
126       # Calculando a estatistica de teste do TRV e armazenando
127       veros_reduzido_boot <- mod_reduzido_boot$logLik
128       veros_completo_boot <- mod_completo_boot$logLik
129
130       2 * (veros_completo_boot - veros_reduzido_boot)
131     })
132     # Calculando valor-p do Boot1 (Para cenario 5 mudar: df = 3)
133     estat_corrigida <- (estat_teste * 2) / mean(estatisticas_boot)
134     valorp_boot1 <- 1 - pchisq(estat_corrigida, df = 2)
135
136     # Calculando valor-p do Boot2
137     k <- sum(estatisticas_boot > estat_teste)
138     valorp_boot2 <- (k + 1) / (B + 1)
139
140     # Retornando uma lista para cada iteracao de Monte Carlo
141     list(valorp_trv = valorp_trv,
142          valorp_boot1 = valorp_boot1,
143          valorp_boot2 = valorp_boot2)
144   },
145   # Definindo parametro que controla as sementes dentro dos nucleos
        paralelos
146   future.seed = TRUE)
147
148   # Juntando todos os resultados das iteracoes paralelizadas
149   valoresp_trv   <- sapply(resultados, `[[`, "valorp_trv")
150   valoresp_boot1 <- sapply(resultados, `[[`, "valorp_boot1")
151   valoresp_boot2 <- sapply(resultados, `[[`, "valorp_boot2")
152
153   # Calculando taxas de rejeicao
154   niveis_rejeicao_trv <- c(mean(valoresp_trv   <= 0.01) * 100,
155                            mean(valoresp_trv   <= 0.05) * 100,
156                            mean(valoresp_trv   <= 0.1) * 100)
157   niveis_rejeicao_boot1 <- c(mean(valoresp_boot1 <= 0.01) * 100,
158                              mean(valoresp_boot1 <= 0.05) * 100,
159                              mean(valoresp_boot1 <= 0.1) * 100)
```

```
  niveis_rejeicao_boot2 <- c(mean(valoresp_boot2 <= 0.01) * 100,
                             mean(valoresp_boot2 <= 0.05) * 100,
                             mean(valoresp_boot2 <= 0.1) * 100)

  # Organizando resultados finais em uma tabela
  resultados_finais <- data.frame(rbind(niveis_rejeicao_trv,
                                        niveis_rejeicao_boot1,
                                        niveis_rejeicao_boot2))
  colnames(resultados_finais) <- c("1%", "5%", "10%")
  rownames(resultados_finais) <- c("TRV", "Boot1", "Boot2")
  return(resultados_finais)
}

# Exemplo para cenario 1
simulacao(20,5000,1000)
simulacao(30,5000,1000)
simulacao(40,5000,1000)
simulacao(50,5000,1000)
simulacao(100,5000,1000)
```

B.2 Aplicação

```
library(tidyverse)
library(DirichletReg)
library(gamlss)

# Carregando o banco de dados
dados <- read_table("C:/Users/vini-/OneDrive/Area de
   Trabalho/TCC/dados_aplicacao.txt")

# Removendo a coluna de identificacao
dados <- dados %>%
  select(-ID)

#########################
## Analise Descritiva ##
#########################
# Transformando os dados para o formato long
```

```r
dados_long <- dados %>%
  pivot_longer(cols = c("N1", "N2", "N3", "REM"), names_to =
     "Componente", values_to = "Valor")
# Grafico de dispersao para variavel TST
ggplot(dados_long, aes(x = TST, y = Valor, color = Componente)) +
  geom_point(alpha = 0.6) +
  facet_wrap(~ Componente) +
  theme_minimal() +
  labs(x = "Tempo de Sono Total", y = "Percentual do Componente",
       title = "Percentual de Qualidade do Sono por TST") +
  scale_color_manual(values = c("#000000", "#E69F00", "#56B4E9",
     "#009E73"))

# Categorizando TST em 4 classes
dados_long <- dados_long %>%
  mutate(
    TST_cat = cut(
      TST, breaks = 4,
      labels = c("Baixo", "Medio-Baixo", "Medio-Alto", "Alto"),
      include.lowest = TRUE))
# Calculando as medias para cada categoria
media_categorias <- dados_long %>%
  group_by(TST_cat, Componente) %>%
  summarise(Media_Valor = mean(Valor, na.rm = TRUE)) %>%
  ungroup()
# Grafico de perfil para variavel TST categorizada
ggplot(media_categorias, aes(x = TST_cat, y = Media_Valor, color =
   Componente, group = Componente)) +
  geom_point(size = 3) +
  geom_line(alpha = 0.7) +
  theme_minimal() +
  labs(
    x = "Categorias de TST",
    y = "Percentual Medio do Componente",
    color = "Componente",
    title = "Percentual Medio de Qualidade do Sono por Categorias de
       TST"
  ) +
  scale_color_manual(values = c("#000000", "#E69F00", "#56B4E9",
     "#009E73"))
```

```
51
52 # Boxplots para variavel Caso-Controle por componente da resposta
53 ggplot(dados_long,
54        aes(x = interaction(CaseControl, Componente),
55            y = Valor, fill = factor(CaseControl))) +
56   geom_boxplot() +
57   theme_minimal() +
58   labs(x = "Caso-Controle por Componente",
59        y = "Percentual da Combinacao",
60        fill = "Caso-Controle",
61        title = "Boxplots para as Combinacoes de Caso-Controle e
           Componente") +
62   scale_x_discrete(labels = function(x) gsub("\\.", " - ", x)) +
63   theme(axis.text.x = element_text(angle = 90, hjust = 1, vjust = 0.5))
     +
64   scale_fill_discrete(labels = c("0" = "Controle = 0", "1" = "Caso =
     1"))
65
66
67 ##########################
68 ## Testes de Hipoteses ##
69 ##########################
70 # Preparando os dados para ajustar o modelo
71 y = dados[, 3:6]
72 x12 = dados$TST
73 x13 = dados$CaseControl
74 dados$y = DR_data(dados[, 3:6])
75 colnames(dados$y) = c(c("y1","y2","y3","y4"))
76
77 ### Testando covariavel Caso-Controle
78 # Semente para reprodutibilidade
79 set.seed(8261)
80
81 # Aplicando o TRV para testar todos os betas (media e precisao)
82 mod_completo <- DirichReg(y ~ x12 + x13 | x12 + x13, data = dados,
     model = 'alternative')
83 mod_reduzido <- DirichReg(y ~ x12 | x12, data = dados, model =
     'alternative')
84
85 veros_reduzido <- mod_reduzido$logLik
```

```r
veros_completo <- mod_completo$logLik
estat_teste <- 2 * (veros_completo - veros_reduzido)
valorp_trv1 <- 1 - pchisq(estat_teste, df = 4)

# Aplicando o TRV para testar os betas relacionados as medias
mod_completo <- DirichReg(y ~ x12 + x13 | x12 + x13, data = dados,
    model = 'alternative')
mod_reduzido <- DirichReg(y ~ x12 | x12 + x13, data = dados, model =
    'alternative')

veros_reduzido <- mod_reduzido$logLik
veros_completo <- mod_completo$logLik
estat_teste <- 2 * (veros_completo - veros_reduzido)
valorp_trv2 <- 1 - pchisq(estat_teste, df = 3)

# Aplicando o TRV para testar o beta do parametro de precisao
mod_completo <- DirichReg(y ~ x12 + x13 | x12 + x13, data = dados,
    model = 'alternative')
mod_reduzido <- DirichReg(y ~ x12 + x13 | x12, data = dados, model =
    'alternative')

veros_reduzido <- mod_reduzido$logLik
veros_completo <- mod_completo$logLik
estat_teste <- 2 * (veros_completo - veros_reduzido)
valorp_trv3 <- 1 - pchisq(estat_teste, df = 1)

# Tabela com os resultados
data.frame(TRV_x13 = c(valorp_trv1, valorp_trv2, valorp_trv3))


### Testando covariavel TST
## Teste para todos os betas (media e precisao)
# Semente para reprodutibilidade
set.seed(8261)

# Definindo tamanho amostral e quantidade de replicas de bootstrap
n <- nrow(dados)
B = 1000

# Aplicando o TRV para testar todos os betas (media e precisao)
```

```
mod_completo <- DirichReg(y ~ x12 | x12, data = dados, model =
    'alternative')
mod_reduzido <- DirichReg(y ~ 1, data = dados, model = 'alternative')

veros_reduzido <- mod_reduzido$logLik
veros_completo <- mod_completo$logLik
estat_teste <- 2 * (veros_completo - veros_reduzido)
valorp_trv <- 1 - pchisq(estat_teste, df = 4)

# Obtendo betas do modelo reduzido
b <- as.vector(mod_reduzido$coefficients)
b <- c(b[1], 0, b[2], 0, b[3], 0, b[4], 0)
# Definindo os betas para os testes via bootstrap
betas_boot <- matrix(data=b,
                     nrow = 4,ncol = 2,byrow = TRUE)
# Definindo a matriz X com as covariaveis
X <- cbind(1, dados$TST)
colnames(X) <- c('x11', 'x12')

# Calculando etas para os testes via bootstrap
etas_boot <- X %*% t(betas_boot)
colnames(etas_boot) <- c('eta1', 'eta2', 'eta3', 'eta4')
# Calculando as medias para os testes via bootstrap
denom_boot <- 1 + exp(etas_boot[, "eta1"]) + exp(etas_boot[, "eta2"]) +
    exp(etas_boot[, "eta3"])
medias_boot <- cbind(1 / denom_boot,
                     exp(etas_boot[, "eta1"]) / denom_boot,
                     exp(etas_boot[, "eta2"]) / denom_boot,
                     exp(etas_boot[, "eta3"]) / denom_boot)
# Calculando o parametro de precisao para os testes via bootstrap
phi_boot <- exp(etas_boot[,"eta4"])
# Calculando os parametros alpha para os testes via bootstrap
alphas_boot <- medias_boot * phi_boot

# Criando vetor para armazenar as replicas de bootstrap
estatisticas_boot <- numeric(B)
for (j in 1:B){
  if (j %% 100 == 0) {
    print(paste("Simulacao", j))
  }
```

```
160   # Gerando a variavel resposta
161   y_boot <- rdirichlet(n, alphas_boot)
162   # Criando um dataframe com a resposta e as covariaveis
163   dados_boot <- data.frame(y_boot, x12 = X[, "x12"])
164   # Tratando a resposta para o ajuste
165   dados_boot$y_boot <- DR_data(dados_boot[, 1:4])
166
167   # Ajustando modelos completo e reduzido
168   mod_completo_boot <- DirichReg(y_boot ~ x12 | x12, data = dados_boot,
          model = 'alternative')
169   mod_reduzido_boot <- DirichReg(y_boot ~ 1, data = dados_boot, model =
          'alternative')
170   # Calculando a estatistica de teste do TRV e armazenando
171   veros_reduzido_boot <- mod_reduzido_boot$logLik
172   veros_completo_boot <- mod_completo_boot$logLik
173   estatisticas_boot[j] <- 2 * (veros_completo_boot -
          veros_reduzido_boot)
174 }
175 # Calculando valor-p do Boot1
176 estat_corrigida <- (estat_teste*4) / mean(estatisticas_boot)
177 valorp_boot1 <- 1 - pchisq(estat_corrigida, df = 4)
178
179 # Calculando valor-p do Boot2
180 k <- sum(estatisticas_boot > estat_teste)
181 valorp_boot2 <- (k+1) / (B+1)
182
183 # Organizando resultados em uma tabela
184 todos <- data.frame(Valores_p = c(valorp_trv, valorp_boot1,
      valorp_boot2))
185
186
187 ## Teste para os betas relacionados as medias
188 # Semente para reprodutibilidade
189 set.seed(8261)
190
191 # Aplicando o TRV para testar os betas relacionados as medias
192 mod_completo <- DirichReg(y ~ x12 | x12, data = dados, model =
      'alternative')
193 mod_reduzido <- DirichReg(y ~ 1 | x12, data = dados, model =
      'alternative')
```

```r

veros_reduzido <- mod_reduzido$logLik
veros_completo <- mod_completo$logLik
estat_teste <- 2 * (veros_completo - veros_reduzido)
valorp_trv <- 1 - pchisq(estat_teste, df = 3)

# Obtendo betas do modelo reduzido
b <- as.vector(mod_reduzido$coefficients)
b <- c(b[1], 0, b[2], 0, b[3], 0, b[4:5])
# Definindo os betas para os testes via bootstrap
betas_boot <- matrix(data=b,
                     nrow = 4,ncol = 2,byrow = TRUE)

# Calculando etas para os testes via bootstrap
etas_boot <- X %*% t(betas_boot)
colnames(etas_boot) <- c('eta1', 'eta2', 'eta3', 'eta4')
# Calculando as medias para os testes via bootstrap
denom_boot <- 1 + exp(etas_boot[, "eta1"]) + exp(etas_boot[, "eta2"]) +
    exp(etas_boot[, "eta3"])
medias_boot <- cbind(1 / denom_boot,
                     exp(etas_boot[, "eta1"]) / denom_boot,
                     exp(etas_boot[, "eta2"]) / denom_boot,
                     exp(etas_boot[, "eta3"]) / denom_boot)
# Calculando o parametro de precisao para os testes via bootstrap
phi_boot <- exp(etas_boot[,"eta4"])
# Calculando os parametros alpha para os testes via bootstrap
alphas_boot <- medias_boot * phi_boot

# Criando vetor para armazenar as replicas de bootstrap
estatisticas_boot <- numeric(B)
for (j in 1:B){
  if (j %% 100 == 0) {
    print(paste("Simulacao", j))
  }
  # Gerando a variavel resposta
  y_boot <- rdirichlet(n, alphas_boot)
  # Criando um dataframe com a resposta e as covariaveis
  dados_boot <- data.frame(y_boot, x12 = X[, "x12"])
  # Tratando a resposta para o ajuste
  dados_boot$y_boot <- DR_data(dados_boot[, 1:4])
```

```
233
234   # Ajustando modelos completo e reduzido
235   mod_completo_boot <- DirichReg(y_boot ~ x12 | x12, data = dados_boot,
          model = 'alternative')
236   mod_reduzido_boot <- DirichReg(y_boot ~ 1 | x12, data = dados_boot,
          model = 'alternative')
237   # Calculando a estatistica de teste do TRV e armazenando
238   veros_reduzido_boot <- mod_reduzido_boot$logLik
239   veros_completo_boot <- mod_completo_boot$logLik
240   estatisticas_boot[j] <- 2 * (veros_completo_boot -
          veros_reduzido_boot)
241 }
242 # Calculando valor-p do Boot1
243 estat_corrigida <- (estat_teste*3) / mean(estatisticas_boot)
244 valorp_boot1 <- 1 - pchisq(estat_corrigida, df = 3)
245
246 # Calculando valor-p do Boot2
247 k <- sum(estatisticas_boot > estat_teste)
248 valorp_boot2 <- (k+1) / (B+1)
249
250 # Organizando resultados em uma tabela
251 mus <- data.frame(Valores_p = c(valorp_trv, valorp_boot1, valorp_boot2))
252
253
254 ## Teste para os betas relacionados as medias
255 # Semente para reprodutibilidade
256 set.seed(8261)
257
258 # Aplicando o TRV para testar o beta do parametro de precisao
259 mod_completo <- DirichReg(y ~ x12 | x12, data = dados, model =
      'alternative')
260 mod_reduzido <- DirichReg(y ~ x12 | 1, data = dados, model =
      'alternative')
261
262 veros_reduzido <- mod_reduzido$logLik
263 veros_completo <- mod_completo$logLik
264 estat_teste <- 2 * (veros_completo - veros_reduzido)
265 valorp_trv <- 1 - pchisq(estat_teste, df = 1)
266
267 # Obtendo betas do modelo reduzido
```

```
b <- as.vector(mod_reduzido$coefficients)
b <- c(b, 0)
# Definindo os betas para os testes via bootstrap
betas_boot <- matrix(data=b,
                     nrow = 4,ncol = 2,byrow = TRUE)

# Calculando etas para os testes via bootstrap
etas_boot <- X %*% t(betas_boot)
colnames(etas_boot) <- c('eta1', 'eta2', 'eta3', 'eta4')
# Calculando as medias para os testes via bootstrap
denom_boot <- 1 + exp(etas_boot[, "eta1"]) + exp(etas_boot[, "eta2"]) +
    exp(etas_boot[, "eta3"])
medias_boot <- cbind(1 / denom_boot,
                     exp(etas_boot[, "eta1"]) / denom_boot,
                     exp(etas_boot[, "eta2"]) / denom_boot,
                     exp(etas_boot[, "eta3"]) / denom_boot)
# Calculando o parametro de precisao para os testes via bootstrap
phi_boot <- exp(etas_boot[,"eta4"])
# Calculando os parametros alpha para os testes via bootstrap
alphas_boot <- medias_boot * phi_boot

# Criando vetor para armazenar as replicas de bootstrap
estatisticas_boot <- numeric(B)
for (j in 1:B){
  if (j %% 100 == 0) {
    print(paste("Simulacao", j))
  }
  # Gerando a variavel resposta
  y_boot <- rdirichlet(n, alphas_boot)
  # Criando um dataframe com a resposta e as covariaveis
  dados_boot <- data.frame(y_boot, x12 = X[, "x12"])
  # Tratando a resposta para o ajuste
  dados_boot$y_boot <- DR_data(dados_boot[, 1:4])

  # Ajustando modelos completo e reduzido
  mod_completo_boot <- DirichReg(y_boot ~ x12 | x12, data = dados_boot,
      model = 'alternative')
  mod_reduzido_boot <- DirichReg(y_boot ~ x12 | 1, data = dados_boot,
      model = 'alternative')
  # Calculando a estatistica de teste do TRV e armazenando
```

```
305   veros_reduzido_boot <- mod_reduzido_boot$logLik
306   veros_completo_boot <- mod_completo_boot$logLik
307   estatisticas_boot[j] <- 2 * (veros_completo_boot -
          veros_reduzido_boot)
308 }
309 # Calculando valor-p do Boot1
310 estat_corrigida <- (estat_teste*1) / mean(estatisticas_boot)
311 valorp_boot1 <- 1 - pchisq(estat_corrigida, df = 1)
312
313 # Calculando valor-p do Boot2
314 k <- sum(estatisticas_boot > estat_teste)
315 valorp_boot2 <- (k+1) / (B+1)
316
317 # Organizando resultados em uma tabela
318 phis <- data.frame(Valores_p = c(valorp_trv, valorp_boot1,
        valorp_boot2))
319
320
321 ### Analise adicional
322 # Semente para reprodutibilidade
323 set.seed(8261)
324
325 # Criando vetores para armazenar os valores-p da analise
326 valoresp_trv <- numeric(500)
327 valoresp_boot1 <- numeric(500)
328 valoresp_boot2 <- numeric(500)
329
330 # Analise: taxas de rejeicao se amostra tivesse tamanho 15
331 for(i in 1:500){
332   # Tirando uma amostra sem reposicao de tamanho 15 do banco de dados
          original
333   amostra <- dados[sample(1:nrow(dados), size = 15, replace = FALSE), ]
334
335   # Preparando os dados da amostra para ajustar o modelo
336   y = amostra[, 3:6]
337   x12 = amostra$TST
338   amostra$y = DR_data(amostra[, 3:6])
339   colnames(amostra$y) = c(c("y1","y2","y3","y4"))
340
341   # Definindo tamanho amostral e quantidade de replicas de bootstrap
```

```
342 n <- nrow(amostra)
343 B = 1000
344
345 # Aplicando o TRV para testar os betas relacionados as medias
346 mod_completo <- DirichReg(y ~ x12 | x12, data = amostra, model =
        'alternative')
347 mod_reduzido <- DirichReg(y ~ 1 | x12, data = amostra, model =
        'alternative')
348
349 veros_reduzido <- mod_reduzido$logLik
350 veros_completo <- mod_completo$logLik
351 estat_teste <- 2 * (veros_completo - veros_reduzido)
352 valoresp_trv <- 1 - pchisq(estat_teste, df = 3)
353
354 # Obtendo betas do modelo reduzido
355 b <- as.vector(mod_reduzido$coefficients)
356 b <- c(b[1], 0, b[2], 0, b[3], 0, b[4:5])
357 # Definindo os betas para os testes via bootstrap
358 betas_boot <- matrix(data=b,
359                      nrow = 4,ncol = 2,byrow = TRUE)
360 # Definindo a matriz X com as covariaveis da amostra
361 X <- cbind(1, amostra$TST)
362 colnames(X) <- c('x11', 'x12')
363
364 # Calculando etas para os testes via bootstrap
365 etas_boot <- X %*% t(betas_boot)
366 colnames(etas_boot) <- c('eta1', 'eta2', 'eta3', 'eta4')
367 # Calculando as medias para os testes via bootstrap
368 denom_boot <- 1 + exp(etas_boot[, "eta1"]) + exp(etas_boot[, "eta2"])
        + exp(etas_boot[, "eta3"])
369 medias_boot <- cbind(1 / denom_boot,
370                      exp(etas_boot[, "eta1"]) / denom_boot,
371                      exp(etas_boot[, "eta2"]) / denom_boot,
372                      exp(etas_boot[, "eta3"]) / denom_boot)
373 # Calculando o parametro de precisao para os testes via bootstrap
374 phi_boot <- exp(etas_boot[,"eta4"])
375 # Calculando os parametros alpha para os testes via bootstrap
376 alphas_boot <- medias_boot * phi_boot
377
378 # Criando vetor para armazenar as replicas de bootstrap
```

```
  estatisticas_boot <- numeric(B)
  for (j in 1:B){
    if (j %% 100 == 0) {
      print(paste("Simulacao", j))
    }
    # Gerando a variavel resposta
    y_boot <- rdirichlet(n, alphas_boot)
    # Criando um dataframe com a resposta e as covariaveis
    dados_boot <- data.frame(y_boot, x12 = X[, "x12"])
    # Tratando a resposta para o ajuste
    dados_boot$y_boot <- DR_data(dados_boot[, 1:4])

    # Ajustando modelos completo e reduzido
    mod_completo_boot <- DirichReg(y_boot ~ x12 | x12, data =
       dados_boot, model = 'alternative')
    mod_reduzido_boot <- DirichReg(y_boot ~ 1 | x12, data = dados_boot,
       model = 'alternative')
    # Calculando a estatistica de teste do TRV e armazenando
    veros_reduzido_boot <- mod_reduzido_boot$logLik
    veros_completo_boot <- mod_completo_boot$logLik
    estatisticas_boot[j] <- 2 * (veros_completo_boot -
       veros_reduzido_boot)
  }
  # Calculando valor-p do Boot1
  estat_corrigida <- (estat_teste*3) / mean(estatisticas_boot)
  valoresp_boot1 <- 1 - pchisq(estat_corrigida, df = 3)

  # Calculando valor-p do Boot2
  k <- sum(estatisticas_boot > estat_teste)
  valoresp_boot2 <- (k+1) / (B+1)
}
# Calculando taxas de rejeicao
niveis_rejeicao_trv <- c(mean(valoresp_trv <= 0.01) * 100,
   mean(valoresp_trv <= 0.05) * 100,
                         mean(valoresp_trv <= 0.1) * 100)
niveis_rejeicao_boot1 <- c(mean(valoresp_boot1 <= 0.01) * 100,
   mean(valoresp_boot1 <= 0.05) * 100,
                           mean(valoresp_boot1 <= 0.1) * 100)
niveis_rejeicao_boot2 <- c(mean(valoresp_boot2 <= 0.01) * 100,
   mean(valoresp_boot2 <= 0.05) * 100,
```

```
                              mean(valoresp_boot2 <= 0.1) * 100)

# Organizando resultados em uma tabela
resultados <- data.frame(rbind(niveis_rejeicao_trv,
   niveis_rejeicao_boot1, niveis_rejeicao_boot2))
colnames(resultados) <- c("1%", "5%", "10%")

##################
## Diagnostico ##
##################
# Semente para reprodutibilidade
set.seed(8261)

# Definindo tamanho amostral e quantidade de replicas de bootstrap
n <- nrow(dados)
B = 1000

# Preparando os dados para ajustar o modelo
y = dados[, 2:5]
x12 = dados$TST
dados$y = DR_data(dados[, 2:5])
colnames(dados$y) = c(c("y1","y2","y3","y4"))
dados$x12 = dados$TST

# Ajustando o modelo final
ajuste <- DirichReg(y ~ x12 | x12, data = dados, model = 'alternative')

# Criando uma matriz para guardar os residuos de cada componente
residuos = matrix(0,n,4)
# Calculando residuos quantilicos aleatorizados
residuos[,1] = qnorm(pBE(unlist(y[,1]), ajuste$fitted.values$mu[,1],
   1/(1+ajuste$fitted.values$phi)^0.5), 0, 1)
residuos[,2] = qnorm(pBE(unlist(y[,2]), ajuste$fitted.values$mu[,2],
   1/(1+ajuste$fitted.values$phi)^0.5), 0, 1)
residuos[,3] = qnorm(pBE(unlist(y[,3]), ajuste$fitted.values$mu[,3],
   1/(1+ajuste$fitted.values$phi)^0.5), 0, 1)
residuos[,4] = qnorm(pBE(unlist(y[,4]), ajuste$fitted.values$mu[,4],
   1/(1+ajuste$fitted.values$phi)^0.5), 0, 1)

# Tirando o valor absoluto dos residuos
```

```r
abs_residuos <- abs(residuos)
# Identificando qual coluna tem o residuo de maior valor absoluto e
    retornando o indice dessa coluna
max_indices <- max.col(abs_residuos, ties.method = "first")
# Extraindo o residuo (com sinal) correspondente ao indice maximo
max_residuos <- residuos[cbind(1:n, max_indices)]
# Calculando a funcao sinal para os residuos dominantes
hi <- sign(max_residuos)
# Calculando li: somando os valores absolutos de todos os residuos de
    cada obs., mas com sinal definido pelo residuo dominante
li <- hi * rowSums(abs_residuos)

# Obtendo os parametros alpha com base no modelo final ajustado
alphas_boot <- ajuste$fitted.values$alpha
# Criando uma matriz para armazenar os li para cada replica
li_boot <- matrix(NA, nrow = n, ncol = B)
for(i in 1:B){
  # Gerando nova amostra com parametros do modelo ajustado
  y_boot <- rdirichlet(n, alphas_boot)
  # Criando um dataframe com a resposta e as covariaveis
  dados_boot <- as.data.frame(y_boot)
  colnames(dados_boot) <- c("y1", "y2", "y3", "y4")
  dados_boot$x12 <- x12
  # Tratando a resposta para o ajuste
  dados_boot$y_boot <- DR_data(dados_boot[, 1:4])

  # Ajustando um modelo com a nova amostra bootstrap
  ajuste_boot <- DirichReg(y_boot ~ x12 | x12, data = dados_boot, model
      = 'alternative')

  # Calculando e armazenando os residuos para a nova amostra
  residuos_boot = matrix(0,n,4)
  residuos_boot[,1] = qnorm(pBE(unlist(y_boot[,1]),
      ajuste_boot$fitted.values$mu[,1],
      1/(1+ajuste_boot$fitted.values$phi)^0.5), 0, 1)
  residuos_boot[,2] = qnorm(pBE(unlist(y_boot[,2]),
      ajuste_boot$fitted.values$mu[,2],
      1/(1+ajuste_boot$fitted.values$phi)^0.5), 0, 1)
  residuos_boot[,3] = qnorm(pBE(unlist(y_boot[,3]),
      ajuste_boot$fitted.values$mu[,3],
```

```
      1/(1+ajuste_boot$fitted.values$phi)^0.5), 0, 1)
  residuos_boot[,4] = qnorm(pBE(unlist(y_boot[,4]),
      ajuste_boot$fitted.values$mu[,4],
      1/(1+ajuste_boot$fitted.values$phi)^0.5), 0, 1)

  # Mesmo procedimento descrito anteriormente ate calcular li
  abs_residuos_boot <- abs(residuos_boot)
  max_indices_boot <- max.col(abs_residuos_boot, ties.method = "first")
  max_residuos_boot <- residuos_boot[cbind(1:n, max_indices_boot)]
  hi_boot <- sign(max_residuos_boot)
  li_boot[,i] <- hi_boot * rowSums(abs_residuos_boot)
}

# Criando vetores ai e si a serem preenchidos
ai <- rep(-1, n)
si <- rep(-1, n)
for(j in 1:n){
  # Calculando ai: quantas replicas li_boot sao menores que li_original
  ai[j] <- sum(li_boot[j,] < li[j])
  # Residuo final proposto por Pereira e Cai (2024)
  si[j] = qnorm(runif(1, ai[j]/(B+1), (ai[j]+1)/(B+1)), 0, 1)
}


### Repeticao do processo acima 19 vezes para criar envelope simulado
n_sim <- 19

# Matriz para armazenar os residuos simulados
residuos_simulados <- matrix(NA, nrow = n, ncol = n_sim)
# Obtendo os parametros alpha com base no modelo final ajustado
alphas_env <- ajuste$fitted.values$alpha
for(e in 1:n_sim){
  # Verificacao de qual simulacao esta
  print(e)

  y_env <- rdirichlet(n, alphas_env)
  dados_env <- as.data.frame(y_env)
  colnames(dados_env) <- c("y1", "y2", "y3", "y4")
  dados_env$x12 <- x12
  dados_env$y_env <- DR_data(dados_env[, 1:4])
```

```
517
518 ajuste_env <- DirichReg(y_env ~ x12 | x12, data = dados_env, model =
        'alternative')
519
520 residuos_env = matrix(0,n,4)
521 residuos_env[,1] = qnorm(pBE(unlist(y_boot[,1]),
        ajuste_env$fitted.values$mu[,1],
        1/(1+ajuste_env$fitted.values$phi)^0.5), 0, 1)
522 residuos_env[,2] = qnorm(pBE(unlist(y_boot[,2]),
        ajuste_env$fitted.values$mu[,2],
        1/(1+ajuste_env$fitted.values$phi)^0.5), 0, 1)
523 residuos_env[,3] = qnorm(pBE(unlist(y_boot[,3]),
        ajuste_env$fitted.values$mu[,3],
        1/(1+ajuste_env$fitted.values$phi)^0.5), 0, 1)
524 residuos_env[,4] = qnorm(pBE(unlist(y_boot[,4]),
        ajuste_env$fitted.values$mu[,4],
        1/(1+ajuste_env$fitted.values$phi)^0.5), 0, 1)
525
526 abs_residuos_env <- abs(residuos_env)
527 max_indices_env <- max.col(abs_residuos_env, ties.method = "first")
528 max_residuos_env <- residuos_env[cbind(1:n, max_indices_env)]
529 hi_env <- sign(max_residuos_env)
530 li_env <- hi_env * rowSums(abs_residuos_env)
531
532 alphas_boot_env <- ajuste_env$fitted.values$alpha
533 li_boot_env <- matrix(NA, nrow = n, ncol = B)
534 for(ii in 1:B){
535   y_boot_env <- rdirichlet(n, alphas_boot_env)
536   dados_boot_env <- as.data.frame(y_boot_env)
537   colnames(dados_boot_env) <- c("y1", "y2", "y3", "y4")
538   dados_boot_env$x12 <- x12
539   dados_boot_env$y_boot_env <- DR_data(dados_boot_env[, 1:4])
540
541   ajuste_boot_env <- DirichReg(y_boot_env ~ x12 | x12, data =
        dados_boot_env, model = 'alternative')
542
543   residuos_boot_env = matrix(0,n,4)
544   residuos_boot_env[,1] = qnorm(pBE(unlist(y_boot_env[,1]),
        ajuste_boot_env$fitted.values$mu[,1],
        1/(1+ajuste_boot_env$fitted.values$phi)^0.5), 0, 1)
```

```
545     residuos_boot_env[,2] = qnorm(pBE(unlist(y_boot_env[,2]),
            ajuste_boot_env$fitted.values$mu[,2],
            1/(1+ajuste_boot_env$fitted.values$phi)^0.5), 0, 1)
546     residuos_boot_env[,3] = qnorm(pBE(unlist(y_boot_env[,3]),
            ajuste_boot_env$fitted.values$mu[,3],
            1/(1+ajuste_boot_env$fitted.values$phi)^0.5), 0, 1)
547     residuos_boot_env[,4] = qnorm(pBE(unlist(y_boot_env[,4]),
            ajuste_boot_env$fitted.values$mu[,4],
            1/(1+ajuste_boot_env$fitted.values$phi)^0.5), 0, 1)
548
549     abs_residuos_boot_env <- abs(residuos_boot_env)
550     max_indices_boot_env <- max.col(abs_residuos_boot_env, ties.method
            = "first")
551     max_residuos_boot_env <- residuos_boot_env[cbind(1:n,
            max_indices_boot_env)]
552     hi_boot_env <- sign(max_residuos_boot_env)
553     li_boot_env[,ii] <- hi_boot_env * rowSums(abs_residuos_boot_env)
554   }
555
556   ai_env <- rep(-1, n)
557   si_env <- rep(-1, n)
558   for(jj in 1:n){
559     ai_env[jj] <- sum(li_boot[jj,] < li_env[jj])
560     si_env[jj] = qnorm(runif(1, ai_env[jj]/(B+1),
            (ai_env[jj]+1)/(B+1)), 0, 1)
561   }
562   residuos_simulados[,e] <- si_env
563 }
564 # Combinando os residuos originais com os simulados
565 residuos_completos <- cbind(si, residuos_simulados)
566 # Ordenando cada coluna individualmente
567 residuos_ordenados <- apply(residuos_completos, 2, sort)
568
569 # Calculando o minimo de cada linha desconsiderando a primeira coluna
570 minimos <- apply(residuos_ordenados[, -1], 1, min)
571 # Calculando o maximo de cada linha desconsiderando a primeira coluna
572 maximos <- apply(residuos_ordenados[, -1], 1, max)
573
574 # Montando eixo x com quantis teoricos
575 x_env = rep(0, n)
```

```
576 for (a in 1:n){
577   x_env[a] = qnorm((a-3/8)/(n+1/4),0,1)
578 }
579
580 # Definindo as bandas de confianca combinando min e max de todas as
      colunas
581 faixa <- range(residuos_ordenados[,1],minimos,maximos)
582 # Grafico de envelope simulado
583 plot(x_env,residuos_ordenados[,1],xlab="Expected normal order
      statistic",
584       ylab="", ylim=faixa, pch=16)
585 par(new=T)
586 # Linha inferior do envelope
587 lines(x_env,minimos)
588 # Linha superior do envelope
589 lines(x_env,maximos)
```