UNIVERSIDADE ESTADUAL DE CAMPINAS
INSTITUTO DE MATEMÁTICA, ESTATÍSTICA E COMPUTAÇÃO
DEPARTAMENTO DE ESTATÍSTICA

Algoritmos de estimação para Cadeias de Markov de Alcance Variável - aplicações a detecção do ritmo em textos escritos

David Henriques da Matta

Dissertação de Mestrado orientada pela

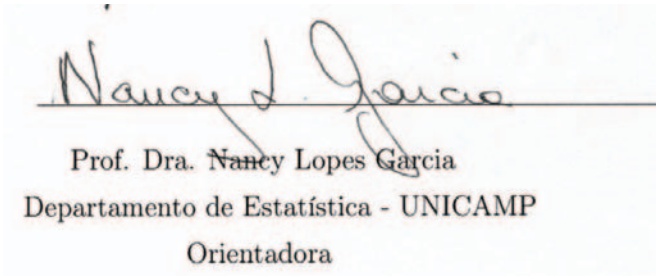
Profa. Dra. Nancy Lopes Garcia

Projeto Fapesp: 05/03434-5

Algoritmos de estimação para Cadeias de Markov de Alcance Variável - aplicações a detecção do ritmo em textos escritos

Este exemplar correponde à redação final da dissertação devidamente corrrigida e defendida por ALUNO e aprovada pela comissão julgadora.

Campinas, 03 de abril de 2008.



Prof. Dra. ~~Nancy~~ Lopes Garcia
Departamento de Estatística - UNICAMP
Orientadora

Banca Examinadora:

1 Prof. Dr. Jefferson Antonio Galves - IME-USP.

2 Prof. Dr. Jesus Enrique Garcia - IMECC-UNICAMP.

3 Profa. Dra. Nancy Lopes Garcia (orientadora) - IMECC-UNICAMP.

Dissertação apresentada ao Instituto de Matemática, Estatística e Computação Científica, UNICAMP, como requesito parcial para obtenção do Título de MESTRE em Estatística.

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA DO IMECC DA UNICAMP
Bibliotecária: Maria Júlia Milani Rodrigues

Matta, David Henriques da
M429a Algoritmos de estimação para cadeias de Markov de alcance variável - aplicações a detecção do ritmo em textos escritos / David Henriques da Matta -- Campinas, [S.P. :s.n.], 2008.

Orientador : Nancy Lopes Garcia
Dissertação (mestrado) - Universidade Estadual de Campinas, Instituto de Matemática, Estatística e Cómputação Científica.

1. Cadeias de Markov de alcance variável. 2. Algoritmo contexto. 3. Bootstrap (Estatística) . I. Garcia, Nancy Lopes. II. Universidade Estadual de Campinas. Instituto de Matemática, Estatística e Computação Científica. III. Título.

Título em inglês: Estimation of algorithms for variable length Markov chains – applications in the detection of rhythm in written texts.

Palavras-chave em inglês (Keywords): 1. Variable length Markov chain. 2. Context algorithm. 3. Bootstrap (Statistical).

Área de concentração: Estatística e Probabilidade

Titulação: Mestre em Estatística

Banca examinadora:

Prof. Dr. Antonio Galves (IME-USP)
Prof. Dr. Jesus Enrique Garcia (IMECC-UNICAMP)
Prof. Dr. Nancy Lopes Garcia (IMECC-UNICAMP)

Data da defesa: 25/03/008

Programa de pós-graduação: Mestrado em Estatística

Dissertação de Mestrado defendida em 25 de março de 2008 e aprovada

Pela Banca Examinadora composta pelos Profs. Drs.

Prof. (a). Dr (a). NANCY LOPES GARCIA

Prof. (a). Dr (a). JEFFERSON ANTONIO GALVES

Prof. (a). Dr (a). JESUS ENRIQUE GARCIA

Dedico este trabalho à Janaina, à minha
mãe Teresinha, à mémoria de meu pai
Ruy e às minhas irmãs Danielle e Danusa.

AGRADECIMENTOS

Gostaria de agradecer primeiramente a Deus e ao meu mentor que tanto me ajudaram nesta etapa.

Agradeço à Prof. Dra. Nancy Lopes Garcia pela sua orientação e incentivo durante estes dois últimos anos de estudo.

Sou grato à Dra. Flaviane e ao Vinicius por terem coloborado na obtenção do conjunto de dados utilizado neste trabalho, bem como a Prof. Dra. Ana Georgina Flesia por ter cedido um dos algoritmos que utilizamos no decorrer do estudo.

Agradeço em especial à minha família pelo apoio incondicional em todas as horas. Não poderia deixar de agradecer também aos meus amigos pelo apoio e incentivo. Agradeço ao Marley, um grande amigo que fiz neste mestrado, e que tanto me ajudou.

Agradeço à Fapesp pelo apoio financeiro a este projeto, e ao grupo do projeto temático "Comportamento estocástico, fenômenos críticos e identificação de padrões rítmicos nas línguas naturais" (Projeto Fapesp: 03/09930-9) pela oportunidade de estar presente neste trabalho.

RESUMO

No presente trabalho, direcionamos nossos estudos à questão de se encontrar evidências estatísticas na detecção de ritmos em textos escritos, apresentando para isso ferramentas probabilísticas que nos permitam discriminar textos brasileiros e portugueses.

Para alcançarmos tais objetivos, abordamos alguns resultados teóricos e práticos em modelagem, reamostragem e estimação das cadeias de Markov de alcance variável. Sendo que na parte de reamostragem, propomos um novo método para conjuntos de dados com um ponto de renovação.

ABSTRACT

In this project, we focus our studies on the question of finding statistical evidences in detecting rhythm in written texts by presenting probabilistic tools that allow us to discriminate Brazilian and Portuguese texts.

To achieve such goals, we some present theoretical and practical results in modeling, resampling and estimation of variable length Markov Chains. More over in the part, we propose a new method of resampling for data sets with a renewal point.

Sumário

Lista de Figuras

Lista de Tabelas

1 Introdução

Existem muitos estudos na área da Lingüística que tem como interesse analisar as diferenças entre o Português Brasileiro e o Português Europeu (abreviados aqui por PB e PE respectivamente). A língua portuguesa moderna é uma língua particularmente interessante para ser estudada pois tanto o PB quanto o PE, apresentam o mesmo conjunto de palavras em sua estrutura (léxico). No entanto, estas línguas apresentam diferentes sintaxes e diferentes prosódias, isto é, não só ordenam as suas palavras de maneiras diferentes como também implementam sentenças com ritmos diferenciados.

O objetivo principal deste trabalho, está relacionado à questão de se encontrar evidências estatísticas na detecção de ritmos em textos escritos, apresentando para isso ferramentas probabilísticas que permitam discriminar textos brasileiros e portugueses modernos, com base na codificação das sílabas em relação às suas propriedades prosódicas (Seção 4.1). Serão analisadas 80 reportagens de jornais contemporâneos, 40 reportagens do jornal brasileiro Folha de São Paulo, e 40 reportagens do jornal português Público, dos anos 1994 e 1995.

Mais especificamente, dada $X_1, X_2, ...$ uma seqüência de variáveis aleatórias definidas como função dos textos escritos, tomando valores em um alfabeto finito A (Seção 2.1), tentamos predizer cada novo símbolo X_n como função do passado $X_1, ..., X_{n-1}$. Dentro deste contexto teórico, pode-se assumir que o passado relevante tem um tamanho finito e fixo k ($k \in \mathbb{N}$), representado nas probabilidades de transição por

$$P(X_n = x_n | X_{n-1} = x_{n-1}, ...X_1 = x_1) = P(X_n = x_n | X_{n-1} = x_{n-1}, ...X_{n-k} = x_{n-k}),$$

para qualquer $n \geq 1$ e $x_1, ..., x_n \in$ A. Os processos que apresentam esta propriedade são denominados cadeias Markov de ordem k.

Um aspecto inerente a este modelo, porém desvantajoso, é que o número de parâmetros a serem estimados cresce exponencialmente com a ordem da cadeia. Utilizando do fato de que a ordem da cadeia pode não ser um valor fixo k, Rissanen (1983) propôs então um modelo cuja a ordem da cadeia varia de acordo com o passado (contexto), isto é, as cadeias de Markov de alcance variável. O nome contexto se refere à porção do passado que influencia a probabilidade de transição do próximo símbolo.

Um procedimento de estimação para as cadeias de Markov de alcance variável, chamado de algoritmo Contexto, foi proposto inicialmente por Rissanen (1983). Posteriormente, trabalhos abordando a convergência de tal algoritmo, popularizaram tal procedimento na literatura estatística. Estes trabalhos se devem a Bühlmann & Wyner (1999) (caso limitado) e Ferrari & Wyner (2003) (caso não limitado). Recentemente, Duarte et al. (2006) deram uma majoração para a velocidade de convergência do algoritmo Contexto para cadeias de alcance variável não limitadas. O interessante desta classe de modelos, é que para se decidir sobre o próximo estado da cadeia, em vez de considerarmos todo o passado, consideramos apenas a parte do passado que é relevante, chamado por Rissanen de contexto.

A teoria e resultados que serão apresentados neste trabalho, foram organizados e serão relatados a seguir.

Iniciamos o Capítulo 2 introduzindo os conceitos básicos de cadeia de Markov de alcance variável, ressaltando o modelo probabilístico e abordando algumas definições importantes. O problema mais interessante desta parte, é relacionado à estimação das árvores de contexto. Existem basicamente dois tipos de algoritmos locais para se solucionar tal problema, sendo eles: o algoritmo Contexto (Rissanen(1983)) e o PST (Ron et al.(1996)). Apresentaremos também os critérios "AIC" e "BIC", que são importantes e bem conhecidos na literatura estatística e utilizados na seleção de modelos, ou seja, uma ferramenta de estimação global. Finalizaremos este capítulo, simulando ambos os algoritmos buscando obter o melhor método para a estimação dos contextos relevantes em nosso conjunto de dados.

Em seguida, no Capítulo 3, apresentamos algumas maneiras presentes na literatura para se fazer reamostragem para Cadeias de Markov. Abordaremos também o método que está sendo proposto neste trabalho, para reamostragem de um conjunto de dados que apresenta um ponto de renovação. Trabalharemos com algumas estimações estatísticas, afim de estudar o comportamento de nosso método de reamostragem perante a alguns métodos já conhecidos.

Logo após, já no Capítulo 4, mostramos toda a parte de coleta dados, abordando a maneira com a qual estes foram coletados. Apresentamos também as árvores estimadas pelo algoritmo Contexto com critérios de seleção AIC e BIC, referentes ao nosso conjunto de dados. Ainda neste capítulo, abordamos o teste proposto por Busch, Ferrari, Flesia, Fraiman, Grynberg e Leonardi (2007), no qual a estrutura média que caracteriza duas amostras de populações distintas de árvores de PB e PE, são comparadas buscando-se testar se a diferença entre essas estruturas são estatisticamente significantes para rejeitar-se uma igualdade de distribuição.

Prosseguimos, abordando os resultados obtidos no teste de hipótese realizado em nosso conjunto de dados, quando ajustamos as árvores utilizando o algoritmo Contexto com critérios de seleção AIC e BIC. Finalizamos este capítulo apresentando um teste de hipótese para uma cadeia de Markov de alcance variável sob reamostragem, utilizando o método proposto neste trabalho.

Finalmente, reservamos para o Capítulo 5 algumas conclusões importantes obtidas na realização desse estudo.

2 *Cadeias de Markov de Alcance Variável*

A noção de cadeia de Markov de alcance variável, foi introduzido por Rissanen (1983) para referir-se a cadeias de Markov de ordem finita, onde a memória da cadeia é uma função dos valores passados. No entanto, não há necessidade de restringir a ordem da cadeia, pois as definições tomam sentido perfeito para cadeias de ordem ilimitada, chamadas por "cadeia de ordem infinita". Neste capítulo, para melhor compreensão deste contexto teórico, abordaremos alguns conceitos básicos sobre cadeias de Markov de alcance variável, bem como alguns métodos de estimação.

2.1 Modelo Probabilístico

Primeiramente, consideraremos aqui um alfabeto A como sendo qualquer conjunto finito formado por símbolos. Por exemplo, A pode ser um espaço de estado discreto e finito de um processo estocástico.

Definição 2.1.1. *Seja $(X_t)_{t\in\mathbb{Z}}$ um processo estocástico tomando valores em um alfabeto finito* A*. O processo $(X_t)_{t\in\mathbb{Z}}$ é uma cadeia de Markov de ordem k, se existir um elemento $k \in \mathbb{N}\cup\{\infty\}$ tal que para todo $t \in \mathbb{Z}$ temos*

$$\begin{aligned} &P(X_t = x_t | X_{t-1} = x_{t-1}, X_{t-2} = x_{t-2}, X_{t-3} = x_{t-3}, ...) = \\ &\quad P(X_t = x_t | X_{t-1} = x_{t-1}, X_{t-2} = x_{t-2}, ..., X_{t-k} = x_{t-k}), \end{aligned} \tag{2.1.1}$$

para toda seqüência $x_t, x_{t-1}, ... \in$ A. A cadeia será dita ainda estacionária se para todo $n \in \mathbb{Z}$ tivermos $P(X_n = a) = \Pi(a)$, onde Π é a medida estacionária da cadeia.

Observação 2.1.1. *Note que na Definição* 2.1.1*, quando $k = \infty$ teremos uma cadeia de ordem infinita, mais precisamente, o símbolo "∞" indica a não possibilidade de se limitar a ordem cadeia.*

Considere uma cadeia de Markov estacionária $(X_t)_{t\in\mathbb{Z}}$ de ordem k ($k \in \mathbb{N} \cup \{\infty\}$) com valores em um alfabeto A com $|\mathrm{A}| < \infty$. Nós denotaremos as variáveis aleatórias por letras maiúsculas, os valores determinísticos fixados por letra minúscula e x_i^j com $\{i, j \in \mathbb{Z}\cup\{-\infty, \infty\} | i < j\}$, como sendo o vetor $(x_j, x_{j-1}, ..., x_i)$ de tamanho $|x_i^j| = j-i+1$. Deste modo, as probabilidades de transição podem ser escritas da seguinte forma:

$$P(X_t = x_t | X_{t-1} = x_{t-1}, ..., X_{t-k} = x_{t-k}) = P(X_t = x_t | X_{t-k}^{t-1} = x_{t-k}^{t-1}). \tag{2.1.2}$$

Para introduzir a idéia de memória de tamanho variável, para uma cadeia de Markov de alcance k, primeiramente apresentaremos a função alcance $\varrho : \mathrm{A}^k \to \{0, 1, ..., k\}$ que para cada $x_{t-k}^{t-1} \in \mathrm{A}^k$ é definida como

$$\varrho(x_{t-k}^{t-1}) = \min\{\varrho \leq k | P(X_t = x_t | X_{t-k}^{t-1} = x_{t-k}^{t-1}) = P(X_t = x_t | X_{t-\varrho}^{t-1} = x_{t-\varrho}^{t-1})\}. \tag{2.1.3}$$

A interpretação de $\varrho(x_{t-k}^{t-1})$, dado um passado x_{t-k}^{t-1}, é representar o número de passos anteriores que devemos observar para escolher o próximo símbolo. Esta definição exclue o caso independente, no qual teríamos $k = 0$ e $\varrho \equiv 0$.

A função $C : \mathrm{A}^k \to \cup_{m=1}^{k} \mathrm{A}^m$, dada por

$$C : x_{t-k}^{t-1} \mapsto x_{t-\varrho(x_{t-k}^{t-1})}^{t-1}, \tag{2.1.4}$$

é denominada *função contexto* e o vetor resultante $C(x_{t-k}^{t-1}) = (x_{t-k}, ..., x_{t-1})$ é chamado contexto. Note que a ordem em que são escritos os símbolos no contexto é inversa a ordem em que aparecem na expressão 2.1.2. O nome contexto se refere à porção do passado que influencia a probabilidade de transição do próximo símbolo.

Temos ainda que a definição de ϱ implica que $\mid C(x_{t-k}^{t-1}) \mid = \varrho(x_{t-k}^{t-1}) \leq k$ para todo $x_{t-k}^{t-1} \in \mathrm{A}^k$. De agora em diante, k denotara também a ordem da função contexto.

Nós formalizaremos a seguir, o conceito de uma cadeia de Markov de alcance variável.

Definição 2.1.2. *Seja $(X_t)_{t\in\mathbb{Z}}$ uma cadeia de Markov estacionária, tomando valores* A*, $|\mathrm{A}| < \infty$, e $C(.)$ sua função contexto correspondente definida por* (2.1.4)*. Seja $k \in \mathbb{N}\cup\{\infty\}$ o menor valor tal que*

$$|C(x_{-\infty}^{t})| = \varrho(x_{-\infty}^{t}) \leq k \text{ para todo } x_{-\infty}^{t} \in \mathrm{A}^{\infty}. \tag{2.1.5}$$

Então $(X_t)_{t\in\mathbb{Z}}$ será denominada cadeia de Markov de alcance variável de ordem k.

Observação 2.1.2. *Claramente, uma cadeia de Markov de alcance variável de ordem k é uma cadeia de Markov estacionária de ordem k.*

Observação 2.1.3. *Para efeito de notação, vamos denotar o conjunto de todas as probabilidades de transição de um símbolo $x_t \in \mathrm{A}$ por $P(X_t = x_t|C(x_{-\infty}^{t-1}))$, onde os valores obtidos pela função contexto $C(.)$ representam todos os estados que determinam estas probabilidades.*

Pela exigência de estacionariedade, a distribuição de probabilidade P de uma cadeia de Markov de alcance variável é completamente especificada pelas probabilidades de transição $P(X_0 = x_0|C(x_{-\infty}^{-1}))$. Desta forma, um modo conveniente de representar estes estados, o espaço de estado minimal, é a representação por árvores (árvores de contexto). Note que a função contexto satisfaz a propriedade do sufixo, isto é, nenhum contexto é um sufixo de outro contexto. Portanto, se $(x_l, ..., x_1)$ é um contexto então nenhuma das subseqüências $(x_j, ..., x_1)$, com $1 \leq j \leq l-1$, é um contexto. Sendo assim, devido a esta propriedade, a função contexto pode ser representada por uma árvore construída da seguinte maneira:

- Raízes no topo;
- Ramos crescem para baixo;
- Todo nó interno tem no máximo | A | descendentes.
- O contexto $u = C(x_{-\infty}^{-1})$ é representado por um ramo, cujo o sub-ramo do topo é determinado por x_{-1}, o próximo sub-ramo é determinado por x_{-2} e assim sucessivamente.

O último pedaço de cada ramo é chamado terminal. Ressaltamos ainda que as árvores de contexto não necessitam ser completas, isto é, seus nódulos internos não necessitam ter necessariamente | A | sub-ramos.

De maneira mais formal, definiremos a seguir o que será uma árvore no contexto em que estamos trabalhando, bem como algumas de suas propriedades.

Definição 2.1.3. *Uma árvore com finitos galhos será definida como um subconjunto contável* τ *de* $\Gamma = \cup_{k=1}^{\infty} \mathrm{A}^{\{1,...,k\}}$*, que satisfaça a propriedade do sufixo, isto é, para nenhum* $w_1^s \in \tau$ *com* $s \in \{1, 2, ..., k\}$ *existe* $u_1^j \in \tau$ *com* $j < s$ *tal que* $w_i = u_i$ *para* $i = 1, ..., j$*.*

Definição 2.1.4. *O tamanho de uma árvore* τ *será dada por* $|\tau| = \max\{|w||w \in \tau\}$ $(w = w_1^s | s \in \{1, 2, 3, ...\})$.

Definição 2.1.5. *Uma árvore com número finitos de galhos* τ *será dita completa se* τ *define uma partição de* $\mathrm{A}^{\{1,2,...\}}$*. Cada elemento da partição coincide com o conjunto das seqüências em* $\mathrm{A}^{\{1,2,...\}}$ *tendo* w_1^k *como sufixo, para algum* $w_1^k \in \tau$*.*

Definição 2.1.6. *Dizemos que uma árvore é ilimitada se o conjunto* τ *é contável e infinito, portanto teremos que a função alcance é ilimitada.*

2.2 Algoritmo Contexto

Esta seção tem como objetivo a descrição do *algoritmo Contexto*, proposto inicialmente por Rissanen em 1983. Dada uma amostra de uma cadeia de Markov de alcance variável, a intenção aqui é a de estimar a função contexto, assim como as probabilidades de transição correspondentes. Cabe lembrar que Bühlmann & Wyner (1999) provaram a consistência de uma versão do algoritmo Contexto (versão que será exemplificada nesta seção e utilizada neste trabalho), e que este resultado foi posteriormente generalizado, em Ferrari & Wyner(1999) e em Duarte et al. (2006), para o caso de cadeias de memória ilimitada .

A estratégia do algoritmo é a seguinte. Primeiro, uma árvore maximal (completa) é produzida, cuja construção considera todos os ramos que possuem um comprimento pré-estabelecido e que aparecem um número mínimo de vezes na amostra (uma condição inicial

a ser determinada para execução do algoritmo). A árvore obtida nesse primeiro processo representa um modelo de cadeia de Markov de alcance variável super dimensionada. Como o espaço A é finito, não há nenhum problema sofisticado em construir tal árvore, tornando-se assim em um problema simples e computacionalmente rápido.

Em segundo lugar, o algoritmo utiliza um procedimento passo a trás ("backward", de baixo para cima) na poda da árvore, usando para isso um critério de decisão de poda. Esse procedimento de poda é efetuado em cada coordenada do contexto testado.

Definição 2.2.1. *Seja $C(.)$ uma função contexto de uma cadeia de Markov de alcance variável tomando valores em um alfabeto* A*, a árvore de contexto e a árvore de contexto de nó terminal serão definidas respectivamente por*

$$\tau = \tau_C = \{w : w = C(x_{-\infty}^{-1}), x_{-\infty}^{-1} \in \mathrm{A}^{\infty}\},$$
$$\tau^T = \tau_C^T = \{w : w \in \tau_C \text{ e } wu \notin \tau_C \text{ para todo } u \in \mathrm{A}\}.$$

De modo geral, a definição acima diz que somente nós terminais da árvore representada por τ serão considerados elementos da árvore de contexto de nó terminal τ^T e os estados $w \in \tau$ não necessitam ser nós terminais.

Exemplo 2.2.1. *Considere uma cadeia de Markov estacionária de ordem* 2*, tomando valores no alfabeto* $\{1, 2, 3, 4\}$*. Um modelo que podemos sugerir para as probabilidades de transição é:*

$$P(X_n = x_n | X_{n-1} = x_{n-1}, X_{n-2} = x_{n-2}) =$$
$$= \begin{cases} P(X_n = x_n | X_{n-1} = 1), & se \quad x_{n-1} = 1, x_{n-2} \in \mathrm{A} \\ P(X_n = x_n | X_{n-1} = 2), & se \quad x_{n-1} = 2, x_{n-2} \in \mathrm{A} \\ P(X_n = x_n | X_{n-1} = 3), & se \quad x_{n-1} = 3, x_{n-2} \in \mathrm{A} \\ P(X_n = x_n | X_{n-1} = 4), & se \quad x_{n-1} = 3, x_{n-2} \in \{1, 2, 3\} \\ P(X_n = x_n | X_{n-1} = 4, X_{n-2} = 4), & se \quad x_{n-1} = 4, x_{n-2} = 4. \end{cases}$$

A função contexto da cadeia de Markov de alcance variável acima pode ser representada por:

$$C(x_{-\infty}^{n-1}) = \begin{cases} (1) & se \quad x_{n-1} = 1, & x_{-\infty}^{n-2} \ arbitrário \\ (2) & se \quad x_{n-1} = 2, & x_{-\infty}^{n-2} \ arbitrário \\ (3) & se \quad x_{n-1} = 3, & x_{-\infty}^{n-2} \ arbitrário \\ (4) & se \quad x_{n-1} = 4, & x_{n-2} \in \{1,2,3\}, x_{-\infty}^{n-3} \ arbitrário \\ (4,4) & se \quad x_{n-1} = 4, & x_{n-2} = 4, x_{-\infty}^{n-3} \ arbitrário. \end{cases}$$

As Figuras 2.2.1 e 2.2.2, representam respectivamente a árvore τ e a árvore de contexto de nó terminal τ^T, da função contexto do Exemplo 2.2.1.

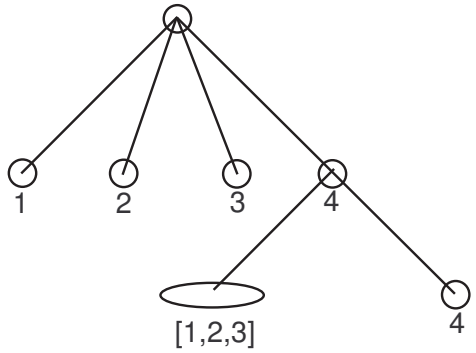


Figura 2.2.1: Árvore de contexto.

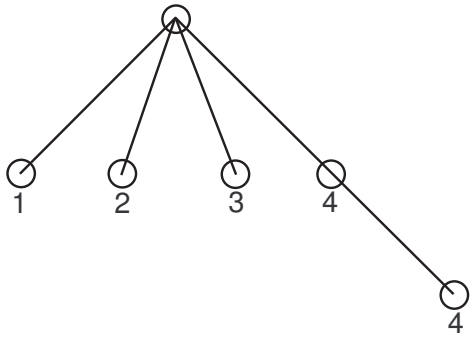


Figura 2.2.2: Árvore de contexto de nó terminal.

Exemplo 2.2.2. *Apresentaremos aqui a função contexto de uma árvore esparsa de nó terminal (definida na Seção 2.4), tomando valores em um alfabeto* $\mathrm{A} = \{0,1\}$. *Como poderemos notar, a árvore esparsa é um exemplo de uma cadeia de Markov de alcance variável de ordem infinita, com função contexto dada por:*

$$C(x_{-\infty}^{n-1}) = \begin{cases} (1) & se & x_{n-1} = 1, & x_{-\infty}^{n-2} \ arbitrário & \\ (1,0) & se & x_{n-1} = 0, & x_{n-2} = 1 & x_{-\infty}^{n-2} \ arbitrário \\ (1,0,0) & se & x_{n-1}^{n-2} = (0,0), & x_{n-3} = 1 & x_{-\infty}^{n-4} \ arbitrário \\ (1,0,0,0) & se & x_{n-1}^{n-3} = (0,0,0), & x_{n-4} = 1 & x_{-\infty}^{n-5} \ arbitrário \\ \ldots & & & & \\ \ldots. & & & & \end{cases}$$

A Figura 2.2.3 representa a árvore esparsa referente a função contexto do Exemplo 2.2.2.

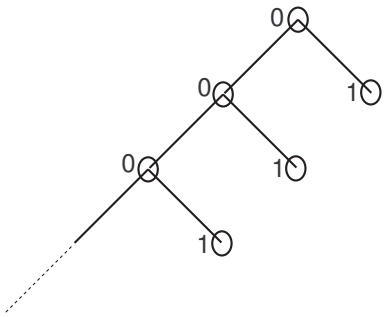


Figura 2.2.3: Árvore esparsa

Definição 2.2.2. *Se $X_1, X_2, ..., X_n$ é uma amostra consistente com a distribuição de probabilidade de um processo estocástico, diremos então que esta é um realização de tal processo.*

Consideremos agora uma realização de uma cadeia de Markov de alcance variável $(X_t)_{t\in\mathbb{Z}}$, ou seja, uma seqüência $x_1, x_2, ..., x_n$ tomando valores no alfabeto A. Seja w_1^j um vetor tal que $w_1^j \in \bigcup_{m=1}^{n} \mathrm{A}^m$. Defina

$$N(w_1^j) = \sum_{t=1}^{n-j+1} I\{x_t^{t+j-1} = w_1^j\} \quad \text{e} \quad \hat{P}(w_1^j) = \frac{N(w_1^j)}{n} \tag{2.2.1}$$

que indicam respectivamente o número de ocorrências, e o estimador da probabilidade do vetor w_1^j na amostra. Se $N(w_1^r.) = \sum_{a\in\mathrm{A}} N(w_1^r a) > 0$, defina os estimadores das probabilidades de transição do vetor w_1^r por

$$\hat{P}(v|w_1^r) = \frac{N(w_1^r v)}{N(w_1^r.)}, \qquad \text{para } v \in \mathrm{A}, w_1^r \in \bigcup_{m=1}^{n-1} \mathrm{A}^m, \tag{2.2.2}$$

onde $w_1^r v$ é a seqüência obtida pela concatenação das duas sequëncias w_1^r e v com tamanho $\mid w_1^r \mid + \mid v \mid$ dada por $(w_r, ..., w_1, v_{|v|}, ..., v_{|1|})$. Se $N(w_1^r.) = 0$, teremos então que $\hat{P}(v|w_1^r) = \dfrac{1}{|A|}$.

A árvore de contexto estimada $\hat{\tau}$ é a maior árvore tal que para todo $w_1^r v \in \hat{\tau}^T$, $(v \in \mathrm{A})$.

$$\Delta_{w_1^r v} = \sum_{x \in \mathrm{A}} \hat{P}(x|w_1^r v) \log \frac{\hat{P}(x w_1^r v)}{\hat{P}(x|w_1^r)} N(w_1^r v) \geq D_n, \tag{2.2.3}$$

onde $D_n = Dlog(n), D > \chi^2_{\nu,0.01}$, $\nu = |\mathrm{A}| - 1$ é um valor de corte escolhido de forma adequada.

A seguir descreveremos o método através de quatro passos, e em seguida observaremos alguns aspectos inerentes ao método que julgamos serem importantes.

Passo(1). Dado o conjunto de dados $x_1, ..., x_n \in \mathrm{A}$, ajustamos uma árvore de contexto maximal τ^T_{max}, que consiste da maior árvore tal que todo elemento (um nó terminal) foi observado no mínimo um número s de vezes nos dados (no modelo proposto originalmente temos que $s = 2$). Sendo assim, considere então $C_{max}(.)$ a função contexto correspondente a esta árvore de contexto maximal. Isto pode ser formalizado da seguinte forma:

- se $w_1^j \in \tau^T_{max}$ implica $N(w_1^j) \geq s$,
- para toda τ^T, árvore de nó terminal que pode ser gerada pelos conjunto de dados, temos que se $w_1^j \in \tau^T$ implica em $N(w_1^j) \geq s$, e que $\tau^T \preceq \tau^T_{max}$. Aqui, $\tau_1^T \preceq \tau_2^T$ significa $w_1^j \in \tau_1 \Rightarrow w_1^j u \in \tau_2$ para algum $u \in \mathrm{A}$.

Passo(2). Tome agora $\tau^T_{(0)} = \tau^T_{max}$, e examine todo elemento (nó terminal) de $\tau^T_{(0)}$ da seguinte forma. Seja $C_{(0)}(.)$ a função contexto correspondente de $\tau^T_{(0)}$ e considere

$$w_1^r u = C_{(0)}(x_1^n), \qquad u \in \mathrm{A}, \quad w_1^r \in \bigcup_{m=1}^{n-1} \mathrm{A}^m$$

onde $w_1^r u$ é um elemento de $\tau^T_{(0)}$ que nós comparamos com a sua versão podada w_1^r. Note que se $r = 1$, a versão podada é o ramo vazio $\emptyset$, isto é, nó da raiz.

Podamos $w_1^r u$ para w_1^r se

$$\Delta_{w_1^r u} = \sum_{x \in \mathrm{A}} \hat{P}(x|w_1^r u) \log \frac{\hat{P}(x|w_1^r u)}{\hat{P}(x|w_1^r)} N(w_1^r u) < D_n.$$

A decisão sobre podar ou não os nós terminais em $\tau_{(0)}^T$ gera (possivelmente) uma árvore menor $\tau_1 \preceq \tau_{(0)}^T$. Construímos então a árvore de nó terminal $\tau_{(1)}^T$ com função contexto $\hat{C}_1(.)$.

Passo(3). Repita o passo(2) com $\tau_{(i)}^T$ ($i \in \{1, 2, 3, .., n\}$), gerando $\tau_{(i+1)}^T$ e assim sucessivamente, até que a poda não seja mais possível. Denote a árvore de contexto gerada no final desse processo por árvore de contexto de poda máxima ($\hat{\tau}$), e sua respectiva função contexto por $\hat{C}(.)$.

Passo(4). Se for de interesse estimar as probabilidades de transição, estime os valores de $P(X_1 = x_1|C(x_1^n))$ por $\hat{P}(x_1|\hat{C}(x_1^n))$, onde $\hat{P}(.|.)$ é definido em (2.2.2).

Observação 2.2.1. *A decisão de corte descrita no passo(2) está relacionada à distância de Kullback-Leibler e ao teste de razão de verossimilhança.*

De fato, pela definição temos que

$$\Delta_{w_1^r u} = \sum_{x \in \mathrm{A}} \hat{P}(x|w_1^r u) \log \frac{\hat{P}(x|w_1^r u)}{\hat{P}(x|w_1^r)} N(w_1^r u) = D(\hat{P}(.|w_1^r u) \parallel \hat{P}(.|w_1^r)) N(w_1^r u) \quad (2.2.4)$$

onde $D(P \parallel Q) = \sum_{x \in \mathrm{A}} P(X = x) \log(\frac{P(X=x)}{Q(X=x)})$ *é a distância de Kullback-Leibler entre duas medidas de probabilidade* P *e* Q *em* A. *Aqui,* $\sum_{x \in \mathrm{A}} P(X = x) \log(\frac{P(X=x)}{Q(X=x)})$ *é definido como zero se* $P(X = x) = 0$ *e* $+\infty$ *se* $P(X = x) > Q(X = x) = 0$. *Uma propriedade interessante desse operador é que ele é sempre não negativo e é zero se e somente se* $P = Q$

Denote agora a função de verossimilhança estimada (condicionada no primeiro estado), baseada em uma função contexto $C(.)$ *por*

$$\hat{P}(x_1^n) = \prod_{t=k+1}^{n} \hat{P}(x_t|C(x_1^{t-1})), \quad (2.2.5)$$

onde k *é a ordem de* $C(.)$. *Considere ainda* $C(.)$ *a função contexto de uma árvore de contexto não podada e por* $C'(.)$ *a função contexto de uma sub-árvore, podada em um nó terminal* $w_1^r u$, *substituído por* w_1^r. *Pela estrutura multiplicativa de* (2.2.5), *alguns*

termos se cancelam na estatística de razão de verossimilhança restando somente os que são considerados nós terminais pela poda. Sendo assim temos:

$$\Delta_{w_1^r u} = \log(\frac{\hat{P}_C(x_1^n)}{\hat{P}_{C'}(x_1^n)}). \tag{2.2.6}$$

Portanto, a formula (2.2.6) *nós diz que o nosso critério de poda é nada mais que um teste de razão de verossimilhança, com região de aceitação para a poda da sub-árvore dada por* $[0, D_n]$.

Observação 2.2.2. *O valor da interrupção* D_n *no passo(2), que é o ponto escolhido para a decisão da poda, é escolhido com base em uma consideração assintótica (Mächler and Bühlmann (2004)).*

Observação 2.2.3. *Para toda árvore* $\tau_{(i)}$*, a ordem na qual são testados os nós terminais nos passos (2) e (3) é irrelevante.*

Se nós interpretarmos o passo (2), visto anteriormente, como um teste de razão de verossimilhança, teremos claramente que pontos de corte pequenos resultarão em árvores de contexto maiores e a ocorrência de um ajuste superestimado. Como este é um parâmetro unidimensional, a otimização com relação a interrupção (do corte) é um problema relativamente fácil.

Com o objetivo de resolver tal problema, indicaremos dois métodos de seleção de modelos muito conhecidos na literatura. Os critérios AIC e BIC serão utilizados com o intuito de escolher o modelo que melhor se ajusta aos dados. Os critérios podem ser descritos para o problema de seleção de árvores de contexto pela seguinte função:

$$G(\gamma, D_n) = -2\text{log-verossimilhança}_{(D_n)} + \gamma(|\mathrm{A}| - 1)|\tau_{\hat{c}_{D_n}}|, \tag{2.2.7}$$

onde $\gamma = 2$ ou $log(n)$ para o AIC e BIC respectivamente, $|\tau_{\hat{c}_{D_n}}|$ é o número de parâmetros livres (probabilidades de transição) da árvore estimada, e finalmente log-verossimilhança$_{(D_n)}$ é o valor da verossimilhança estimada dada por (2.2.5), quando o ponto de corte é igual a D_n.

O objetivo destes critérios ao estimar o ponto de corte é minimizar a divergência de Kullback-Leibler, portanto, o melhor modelo a ser considerado é aquele que apresentar menor valor na função AIC ou BIC conforme a escolha do critério a ser utilizado. Cabe ressaltar ainda, que no critério BIC a penalização (γ) não é uma constante, mas variável de acordo com o tamanho da amostra. Temos também, que utilizando o BIC no conjunto de todas as árvores possíveis, obtemos um estimador consistente para a ordem da cadeia (Csiszár & Talata (2006)).

Na Seção 2.4 trabalharemos com algumas simulações do algoritmo de contexto, apresentado alguns fatos importantes inerentes ao mesmo. Maiores detalhes e motivações a respeito do algoritmo Contexto podem ser encontrados em Bühlman & Wyner (1999).

2.3 Algoritmo PST

Nesta seção descreveremos o método PST (do inglês *Probabilistc suffix trees*) de estimação de árvores introduzido por Bejerano & Yona (2001), que mesmo tendo os mesmos traços do algoritmo de contexto (quando utilizamos um alfabeto A finito e contendo elementos uni-dimensionais), pode ser diferenciado por dois fatos importantes. O primeiro é que o algoritmo PST constroi a árvore e vai realiza os testes entre as distribuições de maneira simultânea. Porém essa diferença não é muito relevante, já que ambas as maneiras, utilizando uma mesma decisão de corte, devolvem a mesma árvore estimada.

A outra diferença, é que o critério utilizado na comparação das distribuições está baseado em uma razão entre medidas e um critério suavizador, e não em um divergente (Kullback-Leibler) ou no teste da razão de verossimilhança como era feito anteriormente.

O algoritmo PST é composto por cinco parâmetros externos que devem ser definidos pelo usuário: L o comprimento da memoria (isto é, o tamanho máximo de um vetor na árvore); $P_{\min}$ a probabilidade mínima em que uma seqüência de dados deve ocorrer na amostra, r que será uma simples medida de diferença entre o candidato predito e seu "Pai", $\gamma_{\min}$ que é um fator suavizante do corte e α que junto com o fator suavizante definem um ponto inicial para a condição de aparecimento do símbolo.

Considere que $\tilde{T}$ denote uma árvore de contexto constituída somente da raiz e $\tilde{S}$ o conjunto de todos os vetores possíveis que devemos testar (vetores que começam com comprimento 1 e variam até o comprimento L). O procedimento de construção da árvore de contexto estimada consiste na seguinte rotina. Enquanto $\tilde{S} \neq \emptyset$, tomamos $w \in \tilde{S}$ (começando pelos de comprimento 1 e aumentando o tamanho gradualmente), e diremos que w será considerado elemento da árvore gerada pelo algoritmo se existir um $\sigma \in$ A tal que

$$\begin{aligned} &\hat{P}(\sigma|w) \geq (1+\alpha)\gamma_{\min} \qquad \text{e} \\ &\frac{\hat{P}(\sigma|w)}{\hat{P}(\sigma|\text{suf(w)})} \geq r, \end{aligned} \tag{2.3.1}$$

onde $\hat{P}$ é definido por (2.2.2) e se $w = w_i^j$, definimos suf(w) como sendo o menor sufixo de w, isto é, w_{i+1}^j. Assim vamos sobrepondo os vetores e construindo a árvore de contexto estimada.

2.4 Simulação

O objetivo principal desta seção, é estudar através de simulações o comportamento dos dois algoritmos apresentados neste capítulo para estimação de árvores de contexto. Para tal análise utilizaremos basicamente dois tipos de conjuntos de dados. O primeiro tipo de conjunto de dados, que chamaremos de Conjunto 1, é uma parte da amostra a ser considerada no Capítulo 4, com alfabeto dado pelo conjunto A $= \{0, 1, 2, 3, 4\}$. O interessante de tal conjunto de dados é que nem todas as transições são possíveis, como por exemplo a do símbolo 4 para o 1.

O outro tipo de conjunto de dados utilizado (Conjunto 2) refere-se a dados gerados de uma cadeia esparsa. Considere uma cadeia de alcance variável $(X_n)_{n\in\mathbb{Z}}$ tomando valores no alfabeto $\{0, 1\}$ uma cadeia esparsa, se suas probabilidades de transição são dadas por

$$P(X_{n+1} = 1 | X_n = 0, ..., X_2 = 0, X_1 = 1, X_{-\infty}^0 = x_{-\infty}^0) = 1 - b^n, \tag{2.4.1}$$

onde $b < 1$ (em nossa amostra $b = 0.7$). O processo perde memória cada vez que alcança o símbolo 1 e recomeça novamente, assim esta cadeia é composta por blocos independentes de zeros delimitados por 1's.

O interessante desse conjunto de dados, é que todas as transições são possíveis e que sabemos qual é a estrutura da árvore gerada, que será dada pela Figura 2.2.3.

Começaremos nosso estudo analisando as árvores estimadas pelo algoritmo Contexto, quando aplicamos os critérios de seleção de modelos (AIC e BIC). O uso desses critérios procederá da seguinte maneira. Variamos de 0.01 em 0.01 um dos parâmetros de corte (D em D_n (2.2.3)), começando pelo valor $\chi^2_{\nu,0.01} + 0.01$ ($\nu = |\mathrm{A}| - 1$) e encerrando quando este atingir um acréscimo de vinte unidades, isto é, para cada novo valor de D uma árvore é estimada pelo algoritmo de contexto e o valor do critério de seleção é calculado. Ao final deste processo foram estimadas 2000 árvores, e conseqüentemente, a escolhida será aquela que apresentar o menor valor no critério de seleção (AIC ou BIC).

No decorrer das simulações que realizaremos, temos que no algoritmo de contexto, a quantidade mínima de vezes em que uma seqüência de dados deve aparecer (para ser testada quanto ao fato de ser um contexto), será dada pelo valor 5. Esse valor será tomado para todas as realizações do algoritmo Contexto nesta seção.

Na Figura 2.4.1, apresentamos a árvore estimada pelo algoritmo de contexto em um conjunto de dados do "Conjunto 1" (tamanho n=2500), utilizando ambos os critérios de seleção de modelos. Ou seja, obtivemos a mesma árvore estimada na adoção dos critérios. Cabe ressaltar, que ao estimarmos todo o Conjunto1, obtivemos algumas poucas situações em que as árvores estimadas variavam de acordo com o critério utilizado. Sendo que, o critério AIC acaba por construir árvores maiores, este fato pode ser presenciado com maior clareza na estimação de todo o nosso conjunto de dados, que será realizada na Seção 4.2.

Seguindo ainda nesta análise, utilizaremos agora dois conjuntos de dados simulados de uma cadeia esparsa, um de tamanho $n = 3469$ e o outro $n = 2586$. As árvores estimadas

são dadas nas Figuras 2.4.2 - 2.4.5. Como poderemos ver, no caso em que a amostra é de tamanho $n = 3469$ (Figuras 2.4.2 e 2.4.3) obtivemos duas árvores de mesma profundidade, ambas tendo como raiz somente o símbolo 0, porém com o critério AIC, o símbolo 1 se ramificou, o que não deveria ocorrer em nossas estimativas.

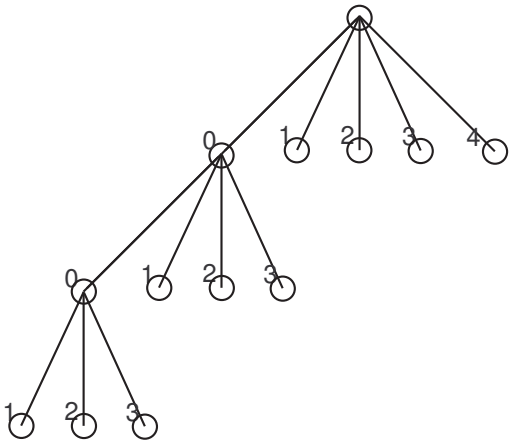


Figura 2.4.1: Árvore estimada pelo algoritmo Contexto (critérios AIC e BIC)

Com o conjunto de dados esparsa de tamanho $n = 2586$ (Figuras 2.4.4 e 2.4.5), obtivemos o mesmo problema encontrado anteriormente, ou seja, o símbolo 1 se ramifica quando usamos o critério AIC. Além disso, a profundidade da árvore estimada pelo critério AIC é maior em uma unidade (o que não é um problema). Portanto, com os dados esparsa, os resultados obtidos com o critério de seleção BIC se mostraram melhores, devido ao fato de conhecermos a verdadeira estrutura da árvore esparsa, isto é, o símbolo 1 não se ramifica.

Nossa próxima análise se dará, de maneira geral, da seguinte forma. Tomamos primeiramente uma árvore (τ_C) qualquer, juntamente com seu alfabeto e o conjunto de todas suas probabilidades de transição. Utilizando da função "simulate.vlmc" já estabelecida no pacote "VLMC" do programa "R", simulamos através da árvore (τ_C) e de suas probabilidades de transição, um conjunto de dados (dados simulados) referente a esta. Desta forma, podemos comparar os algoritmo propostos, já que sabemos qual será a "verdadeira" árvore referente aos dados utilizados.

Na prática, as variáveis $X_1, ...X_n$ geradas pela função "simulate", não são nada mais

que uma reamostra de uma cadeia de Markov de alcance variável ajusta. Descreveremos tal método com maiores detalhes na Seção 3.4.

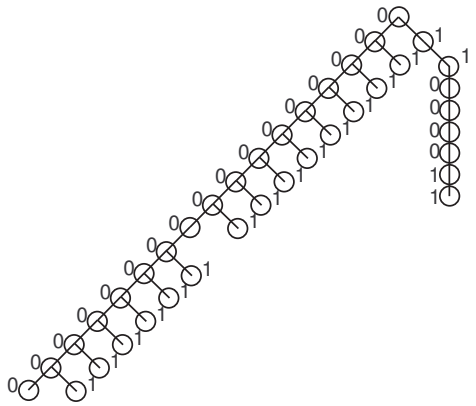


Figura 2.4.2: Árvore estimada pelo algoritmo Contexto, dados esparsa(AIC, n=3496).

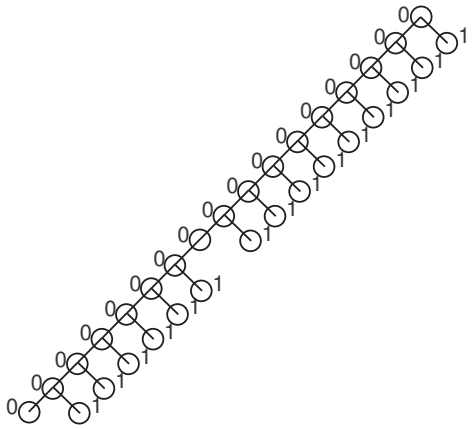


Figura 2.4.3: Árvore estimada pelo algoritmo Contexto, dados esparsa (BIC, n=3496).

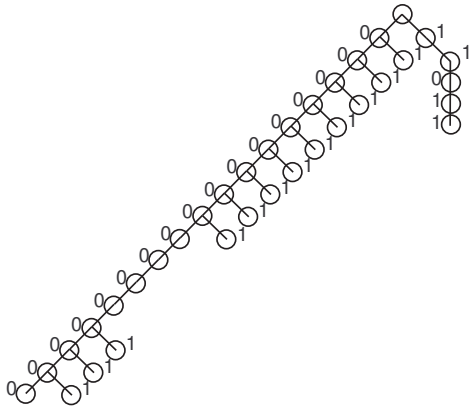


Figura 2.4.4: Árvore estimada pelo algoritmo Contexto, dados esparsa (AIC, n=2586).

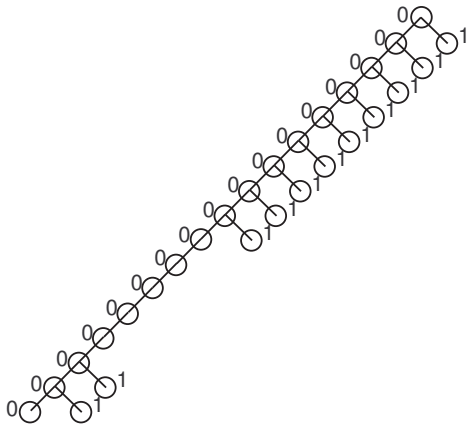


Figura 2.4.5: Árvore estimada pelo algoritmo Contexto, dados esparsa (BIC, n=2586).

A partir do Conjunto 1, ajustaremos cerca de 240 árvores (variando D em D_n (2.2.3)) com suas respectivas probabilidades de transição. Utilizando do processo descrito acima que utiliza da função "simulate.vlmc", simularemos os conjuntos de dados referentes a estas 240 árvores, com tamanho igual ao do conjunto de dados que gerou a árvore na qual os dados foram simulados. A Tabela 2.4.1 apresenta os ajustes realizados pelo algoritmo

Contexto, utilizando os critérios AIC e BIC, nos dados simulados. Para efeito de notação, iremos utilizar as seguintes situações:

- Situação A := A árvore estimada com o critério AIC foi idêntica a árvore "verdadeira";
- Situação B := A árvore estimada com o critério BIC foi idêntica a árvore "verdadeira";
- Situação C := Ambos os ajustes (critérios AIC e BIC) foram idênticos a árvore "verdadeira";
- Situação D := Ambos os ajustes erraram, porém estimaram árvores idênticas;
- Situação E := Ambos os ajustes erraram, porém a estimativa do critério AIC foi melhor, onde "ser melhor" implica que o ajuste se aproximou mais da árvore "verdadeira";
- Situação F := Ambos os ajustes erraram, porém a estimativa do critério BIC foi melhor.

	Quantidade
Situação A	124
Situação B	92
Situação C	89
Situação D	68
Situação E	30
Situação F	15

Tabela 2.4.1: Ajuste com o algoritmo Contexto utilizando AIC e BIC

Um fato importante que observamos durante a construção dos ajuste da Tabela 2.4.1, foi que critério AIC se mostrou um pouco melhor, principalmente quando trata-se de árvores maiores (grande número de parâmetro a serem estimados). Esse fato pode ser justificado pois o critério BIC devido a sua estrutura, acaba penalizando "muito" árvores maiores que provém de amostras pequenas. Porém, temos a grande vantagem que este é

consistente para a ordem da cadeia, quando trabalhos sobre o espaço de todas as árvores possíveis (Csiszár & Talata (2006)). Por outro lado, devido ao fato da penalização do critério AIC ser sempre constante, teremos então uma tendência a superestimar as árvores. Ou seja, o melhor critério depende do tamanho da árvore se a amostra for pequena.

Abordaremos a seguir as estimativas obtidas pelo algoritmo Contexto (critérios AIC e BIC) e PST, referente ao conjunto de dados simulado proveniente da Figura 2.4.1 e de suas probabilidades de transição, gerado pela função "simulate.vlmc". Com relação ao algoritmo PST, várias simulações foram realizadas variando os valores dos parâmetros (r, γ_{min}, α), para chegar ao melhor ajuste. Os parâmetros (L, P_{min}) foram escolhidos (em toda seção) de modo a obtermos equivalência ao algoritmo de contexto usado aqui, são eles: $P_{\min} = 5/n$ onde n é o tamanho da amostra e $L \cong log_{|\mathrm{A}|}n$.

A árvore estimada pelo algoritmo Contexto (critérios AIC e BIC) foi a mesma da Figura 2.4.1. O algoritmo PST com $r = 3.3$, $\gamma_{\min} = 0.1$ e α=0.1, estimou a árvore apresentada na Figura 2.4.6. As escolhas dos parâmetros ($r = 3.3$, $\gamma_{\min} = 0.1$, $\alpha = 0.1$) se deu pela escolha da árvore estimada pelo algoritmo PST que mais se aproximou da árvore "verdadeira".

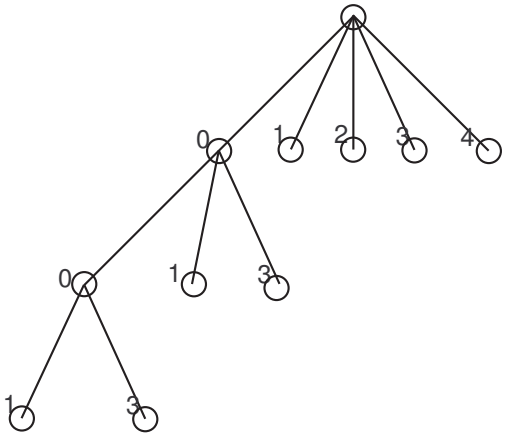


Figura 2.4.6: Árvore estimada pelo algoritmo PST (dados simulados).

Como podemos notar, dentre os métodos utilizados nesta situação, o que obteve melhor estimativa foi o algoritmo de contexto. Ressaltamos, que para diferentes conjuntos de dados, os parâmetros (r, $\gamma_{\min}$, α) que otimizam as simulações no PST podem variar.

Temos ainda, que na estimativa da Figura 2.4.6 L teve que tomar o valor 3, pois para $L = 4$ (quando $n = 2500$) não obtivemos bons resultados. Além do resultado apresentado acima, outras simulações em conjuntos de dados do "Conjunto 1" foram realizadas. Porém, em todas elas o PST se mostrou pior.

Utilizando novamente o conjunto de dados simulado, iremos introduzir uma pequena pertubação neste conjunto e verificar o desempenho e robustez de ambos os algoritmos. Esta pertubação será feita através do acréscimo, de forma aleatória, de cinco seqüências $(4, 1)$ no conjunto de dados simulados, lembrando que esta é uma seqüência que não acontece nos dados. O interessante dessa análise e ver como se comporta a medida de corte quando inserimos esses erros, já que nesse caso a $P_{\min}$ ou a quantidade mínima (dependendo do algoritmo utilizado), esta sendo verificada. Vale lembrar, que estas seqüências foram inseridas em qualquer parte do conjunto de dados, independentemente de quem será o seu antecessor ou sucessor.

As árvores ajustadas com os dados simulados e perturbados, foram exatamente as mesmas que obtivemos sem as pertubações. Ou seja, tanto as estimações do algoritmo Contexto (critérios AIC e BIC) e PST, não detectaram a inserção de um erro no conjunto de dados. Esse fato é muito importante, pois temos agora um pequeno indício que tais algoritmos são robustos a erros que possam vir a ocorrer, como por exemplo em um processo de digitação ou transposição de dados.

Os resultados que apresentaremos seguir são referentes as estimações obtidas pelo algoritmo PST, quando aplicado aos conjuntos de dados da árvore esparsa. O conjunto de parâmetros no qual obtivemos melhor resultado foi $r = 3.3$, $\gamma_{\min} = 0.1$ e α=0.1 (exatamente os mesmos que no exemplo anterior). A escolha destes parâmetros foi baseada no fato em que o símbolo 1 não se ramifica, isto é, buscamos a árvore em que houve menos ramificações deste símbolo. Porém, ainda nestas estimativas que apresentaremos, por diversas vezes o símbolo 1 se ramificou, mostrando novamente que as estimativas obtidas pelo algoritmo de contexto (Figuras 2.4.3 e 2.4.5) são melhores.

Como a dimensão da árvore estimada é muito grande, não é possível desenhar as árvores estimadas pelo PST. As Tabelas 2.4.2 e 2.4.3, apresentam os conjuntos de contextos estimados (a raiz nessas estimativas pode assumir os valores 0 ou 1).

Contextos (n=3469)	
(1,1,0,0)	(1,1,0,0,0)
(1,0,0,0,0,1,1)	(1,1,0,0,0,0)
(1,1,0,0,0,0,1)	(1,0,0,0,0,0,0)
(1,1,0,0,0,0,1,1)	(1,0,0,0,0,0,0,0)
(1,0,0,0,0,0,0,0)	(1,1,0,0,0,0,1,1,0)
(0,0,0,0,0,0,0,0,0,0)	(1,0,0,0,0,0,0,0,0,0)

Tabela 2.4.2: Contextos estimados pelo algoritmo PST

Contextos (n=2586)	
(0,0,1,0)	(1,1,1,1)
(0,0,1,0,1)	(1,1,1,1,0)
(0,0,1,0,1,0)	(1,1,1,1,0,1)
(1,1,1,0,1,0,0)	(1,1,1,1,0,1,0)
(0,0,1,0,1,0,1)	(0,0,1,0,1,0,1,0)
(1,1,1,1,0,1,0,0,0)	(0,0,1,0,1,0,1,0,1)
(1,1,1,1,0,1,0,0,0,0)	(0,0,1,0,1,0,1,0,1,0)
(1,1,1,1,0,1,0,0,0,0,1)	

Tabela 2.4.3: Contextos estimados pelo algoritmo PST

Agregando todos resultados obtidos nessa seção, ao fato de que não conhecemos a verdadeira estrutura das árvores do conjunto de dados que iremos utilizar, daqui por diante iremos adotar o algoritmo Contexto com critério AIC como sendo o nosso principal algoritmo de ajuste de árvores. Porém, para efeito de completude, iremos realizar um estudo conjunto com o algoritmo de contexto utizando ambos os critérios (AIC e BIC).

3 Reamostragem para Cadeias de Markov de Alcance Variável

A idéia principal deste capítulo é a de sugerir um método de reamostragem para cadeias de Markov de alcance variável. Para tanto, iniciaremos com uma breve introdução do primeiro método de reamostragem (introduzido por Efron (1979)), especificando algumas utilidades e maneiras nas quais este é mais utilizado. Depois descreveremos também alguns métodos apresentados na literatura de se fazer reamostragem para uma cadeia de Markov e para uma cadeia de Markov de alcance variável. Finalizaremos sugerindo um método de reamostragem que será adequado à análise de dados reais a ser feita no Capítulo 4.

3.1 Método Reamostragem Independente

O método de reamostragem independente foi introduzido primeiramente por Efron (1979), tendo como intuito apresentar técnicas para se estimar quantidades desconhecidas associadas à modelos estatísticos. Dentre suas utilidades, as mais freqüentes são os cálculos de erro padrão para estimadores, intervalos de confiança para parâmetros desconhecidos e ainda valores descritivios para estatísticas de teste sob a hipótese nula. Portanto, este método é tipicamente usado para estimar quantidades associadas com a distribuição amostral de estimadores e estatísticas de teste.

A respeito de um modelo estatístico, podemos relembrar que este é essencialmente um conjunto de distribuição de probabilidade que tenta descrever o verdadeiro estado natural, tendo a inferência estatística a meta de inferir algo sobre uma população desco-

nhecida baseado nos dados recolhidos da mesma. O reamostragem é uma ferramenta importante para se fazer tal inferência, especialmente quando tratamos de exemplos mais complicados.

Para exemplificar o que sugere o método, iremos supor uma situação hipotética de estimação de erro padrão com supostos dados analíticos. Para tanto, considere uma amostra aleatória observada $\boldsymbol{x} = (x_1, ..., x_n)$ de uma distribuição de probabilidade desconhecida F. Seja $\hat{F}$ a distribuição empírica, atribuindo probabilidade $\frac{1}{n}$ em cada um dos valores x_i observados, $i = 1, 2, ..., n$. Assim, para A evento do espaço amostral temos:

$$\hat{P}(A) = \frac{\sharp\{x_i \in A\}}{n}, \tag{3.1.1}$$

isto é, a freqüência relativa que elementos da amostra x_i ocorre em A.

Uma reamostra é definida como sendo uma amostra aleatória de tamanho n extraída de $\hat{F}$, simbolizada por $\boldsymbol{x}^\star = (x_1^\star, ..., x_n^\star)$, onde a notação estrela indica que $\boldsymbol{x}^\star$ não é o conjunto atual de dados, mas uma aleatorização, ou reamostragem, da versão $\boldsymbol{x}$.

$$\hat{F} \rightarrow (x_1^\star, ..., x_n^\star). \tag{3.1.2}$$

Esta amostra consiste de elementos do conjunto de dados original $(x_1, ..., x_n)$, onde alguns não aparecem, alguns aparecem uma vez, outros duas e assim por diante. Isto é, um número aleatório projeta inteiros selecionados $i_1, i_2, ...i_n$, cada qual igual a qualquer valor entre 1 e n com probabilidade $\frac{1}{n}$, e portanto, a reamostra correspondente é dada por

$$x_1^\star = x_{i_1}, x_2^\star = x_{i_2}, ..., x_n^\star = x_{i_n}. \tag{3.1.3}$$

O algoritmo acaba por extrair diversas amostras independentes, $\boldsymbol{x}_1^\star, ..., \boldsymbol{x}_B^\star$.

Correspondente ao conjunto de dados obtido pela reamostragem, temos a versão pela reamostra de $\hat{\theta}$,

$$\hat{\theta}^\star(b) = s(\boldsymbol{x}_b^\star), \quad b = 1, ...B. \tag{3.1.4}$$

Assim, o erro padrão ($se_F(\hat{\theta})$) da estatística $\hat{\theta}$, é calculado a partir da função de distribuição empírica $\hat{F}$ no lugar da distribuição F desconhecida. Portanto a estimativa dada pela reamostragem é $se_{\hat{F}}(\hat{\theta}^\star)$.

Temos então, que a estimativa do erro padrão ($se_F(\hat{\theta})$) será dada pelos seguintes cálculos

$$\hat{se}_B = \{\sum_{b=1}^{B} \frac{[\hat{\theta}^\star(b) - \hat{\theta}^\star(.)]}{(B-1)}\}^{1/2}, \tag{3.1.5}$$

onde B é o número de reamostragens feitas e $\hat{\theta}^\star(.) = \sum_{b=1}^{B} \frac{\hat{\theta}^\star(b)}{B}$.

Finalmente, o limite de $\hat{se}_B$ quando B tende para o finito será o nosso estimador ideal de ($se_F(\hat{\theta})$),

$$\lim_{B\to\infty} \hat{se}_B = \hat{se}_{\hat{F}} = \hat{se}_{\hat{F}}(\hat{\theta}^\star). \tag{3.1.6}$$

O fato que $\hat{se}_B$ aproxima de $\hat{se}_{\hat{F}}$ quando B tende para o infinito quer dizer que o desvio padrão empírico se aproxima do desvio padrão populacional com um número grande de aplicações por reamostragem. Isso pode ser explicado, pois quando consideramos amostras independentes, a distribuição empírica dos dados converge para a distribuição verdadeira.

A estimador ideal $\hat{se}_{\hat{F}}(\hat{\theta}^\star)$ e sua aproximação $\hat{se}_B$ são às vezes chamados de reamostragem não-paramétrica porque eles são baseados em $\hat{F}$, que é a estimativa não paramétrica da função de distribuição acumulada (f.d.a.) F.

3.2 Reamostragem para Cadeias de Markov

Descreveremos aqui a reamostragem para cadeias de Markov com espaço de estado finito, proposto por Kulperger (1989), e que tem como intuito estimar de modo consistente, para uma cadeia de Markov ergódica homogênea, a distribuição de alguns estimadores, que definiremos posteriormente.

Consideremos então que $(X_n)_{n\in\mathbb{Z}}$ seja uma cadeia de Markov ergódica homogênea, com matriz de transição dada por $\mathcal{P} = (p_{ij})$ e espaço de estado A $= \{1, 2, ..., \eta\}$ com $\eta \geq 2$. Assumimos ainda que $p_{ij} > 0$ para todo $i, j \in S$, isto é, todos os estados se comunicam. Esta suposição pode ser dispensada para cadeias de Markov regulares pela ampliação do espaço de estado se necessário, desde que exista um inteiro $N \geq 1$ tal que $\mathcal{P}_N = \mathcal{P}_{ij}^{(N)}$ (matriz de transição a N passos) são todas estritamente positivas.

Seja agora $\boldsymbol{x} = (x_0, .., x_n)$ uma realização observada no tempo n. Estimamos $\mathcal{P}$ por $\hat{\mathcal{P}}_n = (\hat{p}_{ij})$, onde

$$\hat{p}_{ij} = \begin{cases} \dfrac{n_{ij}}{n_i} & \text{se} \quad n_i > 0 \\ \delta_{ij} & \text{se} \quad n_i = 0 \end{cases} \tag{3.2.1}$$

com n_{ij} sendo o número de transições observadas do estado i para o j, e $n_i = \sum_{j=1}^{\eta} n_{ij}$.

Sabe-se (Billingsley (1961), Basawa e Prakasa Rao (1980)) que

$$n^{\frac{1}{2}}(\hat{p}_{ij} - p_{ij}) \to Z_{ij} \quad 1 \leq i, j \leq \eta \tag{3.2.2}$$

em distribuição, onde Z_{ij}, $i, j \in S$ tem distribuição conjunta normal multivariada com média 0 e covariância que é contínua em $\mathcal{P}$. Temos ainda que a taxa de convergência em (3.2.2) é uniforme em uma vizinhança de $\mathcal{P}$ (Sirazhdinov e Formann (1983), Lifshits (1978), ou Nagaev (1957)).

Para esse processo, sem perda de generalidade, assumiremos que o estado inicial da cadeia de Markov é 1, isto é, $x_0 = 1$. Denotaremos a variável "primeiro tempo de parada ao estado η ($\eta \in$ A)" como sendo

$$T = \begin{cases} \inf\{n \geq 0 : X_n = \eta\}, & \text{se } X_n \text{ atinge } \eta \text{ alguma vez} \\ \infty, & \text{caso contrário.} \end{cases} \tag{3.2.3}$$

Denote por $P(t, Q) = P(T \leq t | X_0 = 1; Q)$, ou seja, a probabilidade de que $T \leq t$, para $t \in \{2, 3, 4...\}$, para uma cadeia de Markov com matriz de transição Q e valor inicial

$X_0 = 1$.

O método de reamostragem para estimação da distribuição do primeiro tempo de parada T, pode ser explicado como segue. $P(t, \hat{\mathcal{P}}_n)$ estima $P(t, \mathcal{P})$, e conseqüentemente uma questão de interesse é encontrar a distribuição de

$$\sqrt{n}(P_r(t, \hat{\mathcal{P}}_n) - P_r(t, \mathcal{P})), \quad t \geq 1. \tag{3.2.4}$$

Quando é $\mathcal{P}$ é conhecida, a distribuição de (3.2.4) pode ser obtida. Um método de Monte Carlo pode ser usado, se a distribuição for uma função complicada de $\mathcal{P}$. Por outro lado, quando $\mathcal{P}$ é desconhecida, um procedimento de reamostragem pode aproximar a distribuição de (3.2.4) pela distribuição de

$$G_n(t; Q) = \sqrt{n}(\hat{P}_r(t; \hat{Q}_n) - P_r(t; Q)), \tag{3.2.5}$$

onde $\hat{Q}_n$ é dada pela formula (3.2.1), estimada através de uma sub-amostra gerada por uma cadeia de Markov (realização observada) com matriz de transição $\hat{P}_n$, e Q é a matriz de transição próxima de $\mathcal{P}$, $Q = \hat{P}_n$. A distribuição de (3.2.5) pode ser calculada, desde que $\hat{\mathcal{P}}_n$ seja uma matriz conhecida, por exemplo, através do método de Monte Carlo. O problema constituiu em justificar que as distribuições de (3.2.4) e (3.2.5) são próximas, isto é, obter um método consistente.

Adaptando o mesmo tipo de análise, é possível estimar $m_1(\mathcal{P}) = E_{\mathcal{P}}(T|X_0 = 1; \mathcal{P})$, que é a esperança do primeiro tempo de chegada, ou estudar também a estimação de $\pi(\mathcal{P})$, onde $\pi(\mathcal{P})$ é a distribuição estacionária do processo.

3.3 Reamostragem para Processos Estacionários

O método que apresentaremos nesta seção foi proposto por Künsch (1989). De modo geral, Temos ainda, que de um modo geral, este método é utilizado para processos estacionários arbitrários com pequeno domínio de dependência expressado.

Künsch (1989) ressalta também, que se a estatística utilizada sobre os dados de

reamostragem gerados por blocos, que veremos nesta seção, não for função simétrica das observações, observações no meio ou até mesmo blocos selecionados aleatoriamente que forem deixados de fora, podem causar problemas. Outro fato importante é que deve ser tomado cuidados com a junção dos blocos selecionados aleatoriamente.

Uma maneira de cuidar destes problemas, é restringir a classe de estatísticas a funcionais de uma distribuição marginal empírica, com dimensão fixada. Cabe notar ainda que este procedimento é consistente no caso do estimador ser a média aritmética, onde podemos obter assintoticamente o viés e a variância.

Definição 3.3.1. *Para observações $x_1, ..., x_n$ de um processo estacionário, a distribuição marginal m-dimensional empírica será*

$$\rho_n^m = (n - m + 1)^{-1} \sum_{t=0}^{n-m} I(X_{t+1}^{t+m} = x_{t+1}^{t+m}). \tag{3.3.1}$$

Nesta seção, sempre serão consideradas estatíticas T_n da forma $T_n = T(\rho_n^m)$, com algum m fixado e algum funcional T com valores em $\mathbb{R}$, definido no conjunto de todas as medidas de probabilidade em $\mathbb{R}^m$ (ou ainda um subconjunto próprio deste). Para efeito de notação, considere blocos de observações Y_t como sendo $(X_{t+1}, ..., X_{t+l-1})$ e o valor $q = n - l + 1$. Desta forma nossa distribuição marginal m-dimensional será denotada por

$$\rho_n^m = q^{-1} \sum_{t=1}^{q} I(Y_t = y_t), \tag{3.3.2}$$

onde $y_t = (x_{t+1}, ..., x_{t+l-1})$.

Em analogia ao caso de amostras i.i.d., seleciona-se blocos de tamanho l aleatoriamente. Assumindo que $q = kl$ com $l \in \mathbb{N}$, a reamostragem de distribuição marginal m-dimensional é

$$\rho_n^{l\,*} = q^{-1} \sum_{j=1}^{k} \sum_{t=S_j+1}^{S_j+l} I(Y_t = y_t), \tag{3.3.3}$$

onde $S_1, ..., S_k$ são variaveis uniformes i.i.d. em $\{0, 1, ...q - l\}$. A expressão (3.3.3) pode

ser reescrita como

$$\rho_n^{l\,*} = q^{-1} \sum_{t=1}^{q} f_t I(Y_t = y_t) \text{ com } f_t = \sharp\{j; t-l \leq S_j \leq t-1\} \tag{3.3.4}$$

ou ainda

$$\rho_n^{l\,*} = q^{-1} \sum_{t=1}^{q} I^*(Y_t = y_t), \tag{3.3.5}$$

onde os k blocos (k=ordem da cadeia) $(Y_1^*, ..., Y_l^*)$, $(Y_{l+1}^*, ..., Y_{2l}^*), ..., (Y_{n-l+1}^*, ..., Y_n^*)$ são i.i.d. com distribuição $(q-l+1)^{-1} \sum_{t=0}^{Q-l} I(Y_{t+1} = y_{t+1}, ..., Y_{t+l} = y_{t+l}) = \rho_{Y,q}^l$.

A partir dos dados gerados tem-se então, para efeitos de estudos estatísticos, uma estatística $T_n^* = T(\rho_n^m)$ e aproximamos a distribuição de $T_n - T(F^m)$ que é desconhecida, pela distribuição de $T_n^* - T_n$, onde F^m é a verdadeira distribuição dos blocos de tamanho m com $Y_1, ...Y_n$ fixados, sendo que $S_1, ..., S_k$ variam.

Em particular, pode-se usar $\sigma_{Boot}^2 = var^*(T_n^*) = E^*[(T_n^* - E^*[T_n^*])^2]$. Onde E^* denota a esperança com respeito a $S_1, ..., S_k$. Similarmente podem ser estimados quantis e ordens quantílicas de interesse. Em geral, tem-se também que σ_{Boot}^2 e a distribuição de $T_n^* - T_n$ são avaliadas através de uma simulação.

Usando independência, o estimador obtido pela reamostragem de Efron estima F^n por $(\rho_n^1)^{\otimes\, n}$, onde $\otimes$ denota o produto de medidas. Para $m = 1$ a proposta apresentada aqui, estima F^n por $(\rho_n^l)^{\otimes\, k}$, e portanto coincide com o modelo de Efron se $l = 1$. Entretanto, será deixado que $l \to \infty$ quando $q \to \infty$, pois desta maneira, assintoticamente, obtêm-se todas distribuições marginais corretamente.

Para $m > 1$ podemos notar que a proposta não nos conduz a um estimador de F^n. Enquanto reescrevemos reamostras $(Y_1^*, ..., Y_q^*)$ em termos de observações originais de (X_t), nós obtemos $q + k(l-1)$ dados pontuais. A razão para este método é que não deseja-se usar observações de diferentes blocos independentes no cálculo de $\rho_n^{l\,*}$, pois desta forma é possível reduzir o efeito de independência conjunta de blocos juntos.

3.4 Reamostragem para Cadeias de Markov de Alcance Variável

Abordaremos brevemente nesta seção o método de reamostragem para uma cadeia de Markov de alcance variável, proposto por Bühlmann & Wyner (1999). Para alcançar tal objetivo, é necessário a introdução de alguns conceitos e definições.

Consideremos uma seqüência de cadeias de Markov de alcance variável $(P_n)_{n\in\mathbb{N}}$, $P_n \in \mathfrak{P}$, com árvore de contexto $\tau_n = \tau_{c_n}$ e funções contexto $C_n(.)$, onde $\mathfrak{P} = \bigcup_{k=0}^{\infty} \mathfrak{P}_k$ e $\mathfrak{P}_k$ é o conjunto de todas as cadeias de Markov de alcance variável de ordem k. Seja também, $X_{1,n}, ..., X_{n,n}$ realizações finitas de P_n, sendo que as probabilidades de transição correspondente a P_n, serão denotadas aqui por $p_n(.|.)$.

Com o intuito de comparar medidas de probabilidade, defina a seguinte métrica para duas medidas $P, P' \in \mathrm{A}^\infty$, que será dada por:

$$d(P, P') = \sum_{m=1}^{\infty} 2^{-m} d_m(P \circ \pi_{1,...,m}^{-1}, P' \circ \pi_{1,...,m}^{-1}), \tag{3.4.1}$$

$$d_m(P \circ \pi_{1,...,m}^{-1}, P' \circ \pi_{1,...,m}^{-1}) = \sup_{x_1^m \in \mathrm{A}^m} |P(x_1^m) - P'(x_1^m)|,$$

onde $\pi_{1,...,m} : \boldsymbol{x} \to x_1, ..., x_m$, $(\boldsymbol{x} \in \mathrm{A}^\infty)$ é uma função coordenada.

O teorema que enunciaremos a seguir é um dos principais resultados para este tipo de reamostragem, ele mostra porque o estimador $\hat{P}_n$ de P_n pode ser utilizado para construir uma amostra. Cabe ressaltar aqui, que tanto a prova deste teorema, quanto maiores informações a respeito deste assunto podem ser encontrados de forma mais detalhada em Bühlmann & Wyner (1999).

Teorema 3.4.1. *Considere dados $X_{1,n}, ..., X_{n,n}$ com P_n satisfazendo certas condições de regularidade (Bühlmann & Wyner (1999)). Então:*

(i) *para $\hat{P}(.|.)$ como em (2.2.2) e $\hat{C}(.)$ estimado no Passo(3) no algoritmo Contexto,* $\lim_{n\to\infty} P(\text{conjunto } \hat{P}(.|\hat{C}(x_1^\infty)); x_1^\infty \in \mathrm{A}^\infty\}$ *gerar uma única medida de probabilidade* $\hat{P}_n \in \mathfrak{P}) = 1$.

(ii) *Para* $\hat{P}_n$ *em* (i) *e* $d(.,.)$ *como em* (3.4.1), $d(\hat{P}_n, P_n) = o_P(1)$ *quando* $n \to \infty$.

(iii) *O processo* $\hat{P}_n$ *descrito no item* (i) *acima, satisfaz a seguinte convergência:*

$P(\hat{P}_n$ *ser um* ϕ*-mixing com coeficientes satisfazendo* $\phi_{\hat{P}_n}(j) \leq (1-\kappa)^j$ *para todo* $j \in \mathbb{N}$*) converge para* 1*, quando* $n \to \infty$.

Cabe ressaltar aqui, que a proposta apresentada nesta seção é utilizada em séries temporais categóricas estacionária (Bühlmann & Wyner (1999)), porém este método de reamostragem ainda pode ser utilizado de maneira bem mais geral.

Descreveremos a seguir o método nos seguintes passos:

Passo 1 Dados $X_1, ...X_n$ tomando valores em A ajustamos uma cadeia de Markov de alcance variável, criamos então uma medida estacionária de probabilidade $\hat{P}_n$ em $\mathrm{A}^{\mathbb{N}}$, veja Teorema 3.4.1.

Passo 2 Crie uma realização finita $X_1^*, ..., X_n^* \ \tilde{} \ \hat{P}_n \circ \pi_{1,...,n}^{-1}$.

Para entendermos um pouco melhor o processo realizado por esse procedimento, temos que a reamostragem de uma cadeia de Markov de alcance variável de ordem "k" é dada através do conjunto de todas as suas probabilidades de transição que pode ser denotado por $\{P(X_n = x_n | C(x_{n-k}^{n-1})); x_{n-k}^{n-1} \in \mathrm{A}^k\}$, começando com um vetor inicial $X_1^k \in \mathrm{A}^k$, onde k é a ordem da cadeia de Markov de alcance variável. Então simulamos

$$X_t \sim P(.|C(X_{t-k}^{t-1})), \quad t = k+1, k+2, ... \tag{3.4.2}$$

Sob condições de regularidade, o fato de termos que inicializar com k valores, torna-se esquecido exponecialmente rápido no decorrer da simulação. Portanto, para simular uma amostra n-dimensional de uma distribuição estacionária de uma cadeia de Markov de alcance variável, nós simulamos $X_1, X_2, ..., X_{m+n}$ como em (3.4.2) e escolhemos $X_{m+1}, ..., X_{m+n}$ como sendo a nossa amostra de tamanho n. Aqui m é um número maior que 10^3 ou 10^4, e é dado por $64 card(\tau_C)$.

Cabe ressaltar, que sob certas condições de regularidade, Bühlmann e Wyner provaram que esta reamostragem para cadeia de Markov de alcance variável é consistente sobre dimensionalidade crescente.

3.5 Reamostragem para Conjunto de Dados com um ponto de Renovação

Nesta seção estamos propondo um método de realização de reamostragem por blocos para cadeias de Markov de alcance variável que possuem um ponto de renovação, isto é, existe $a \in \mathrm{A}$, tal que

$$P(X_0 = \cdot | X_{-l}^{-1} = x_{-l}^{-1}, X_{-\infty}^{-l-1} = x_{-\infty}^{-l-1}) = P(X_0 = \cdot | X_{-l}^{-1} = x_{-l}^{-1}), \tag{3.5.1}$$

para qualquer $l \geq 1$ sempre que $x_{-l} = a$. Como veremos na Seção 4.2, na aplicação em dados lingüísticos temos a presença de um ponto de renovação em todas as árvores estimadas do nosso conjunto de dados.

Considere então $(X_n)_{n\in\mathbb{Z}}$ uma cadeia de Markov de alcance variável estacionária de ordem k, tomando valores em um alfabeto finito $\mathrm{A} = \{a_1, a_2, ..., a_n\}$. Denote por "$a$" o nosso ponto de renovação.

O método de reamostragem proposto procederá da seguinte maneira. Primeiramente, a partir do conjunto de dados, montamos uma lista que contém todas as seqüências encontradas, que terminam no ponto de renovação "a". De maneira mais geral, o algoritmo que constrói a lista, começa lendo a amostra de nossa cadeia de Markov de alcance variável e vai construindo a primeira seqüência, terminando essa no primeiro aparecimento do símbolo de renovação (a). A partir daí, começa-se a construir a próxima seqüência terminando-a no aparecimento do segundo ponto de renovação. Este procedimento acaba ao final da leitura de todo o conjunto de dados. Vale lembrar que uma mesma seqüência pode aparecer mais de uma vez em nossa lista.

Para efeito de notação, considere χ_i com sendo a i-ésima seqüência encontrada na lista, e número total de seqüências por ϖ. Assim, nossa lista será composta por $\{\chi_1, \chi_2, ..., \chi_\varpi\}$.

Dando seguimento, geramos uma variável aleatória (u_i) uniformemente distribuída em $W = \{1, 2, 3, ..., \varpi\}$, e a cada valor encontrado a seqüência $\chi_i^* = \chi_{u_i}$ é inserida na nossa reamostragem. Esse processo é efetuado para $i \in \mathbb{N}$ variando de 1 até $q \in W$, onde q determina o número de seqüências na reamostra. Assim montamos a nossa reamostra que será dada por $\chi_1^*, \chi_2^*, ..., \chi_q^*$.

A seguir, com o intuito de apresentar uma breve análise a respeito do método que acabamos de descrever, iremos propor o estudo da variância (estimada) da seguinte estatística $T_n = \hat{P}(1|0)$ dada por (2.2.2), utilizando o método de reamostragem proposto no trabalho de Bühlmann & Wyner (1999) e apresentado na Seção 3.4 (aqui chamaremos de método I), e o método de reamostragem proposto neste trabalho que foi mostrado nesta secção (que chamaremos por método II). Cabe ressaltar que já é conhecido que o método I é consistente para este caso (Bühlmann & Wyner, 1999).

A amostra que será utilizada para esta análise será uma cadeia esparsa (a mesma utilizada no Capítulo 1 na seção "Simulações", de tamanho 3469), com reamostras contendo o mesmo número de seqüências da amostra original. A variância estimada será

$$\hat{\sigma}_n^2 = nVar^*(T_n^*) \text{ para } \sigma_n^2 = nVar(T_n), \tag{3.5.2}$$

baseada em 500 reamostras.

Os momentos da variância obtida por reamostragem $\hat{\sigma}_n^2$ serão estimados com 200 reamostragens sob diferentes modelos (modelos simulados de uma mesma cadeia esparsa), já o verdadeiro valor de σ_n^2 foi estimado sob 1000 modelos diferentes.

Os resultados estão apresentados na Tabela 3.5.1.

	$E(\hat{\sigma}_n^2)$	$E(\hat{\sigma}_n^2) - \sigma_n^2$	$Var(\sigma_n^2)$	EQMR$(\hat{\sigma}_n^2)$
Método I	0.008512874	-0.03676231	1.180982e-06	0.6598773
Método II	0.008433874	-0.03684131	1.045146e-06	0.662648

Tabela 3.5.1: Momentos da Variância $\hat{\sigma}_n^2$

Como podemos notar na tabela acima, ambos os métodos se comportaram de maneira

similar, porém com uma pequena vantagem para o método I por apresentar menor EQMR. Vale lembrar que o EQMR($\hat{\sigma}_n^2$) é o erro quadrático médio relativo dado pela fórmula, $E|\hat{\sigma}_n^2 - \sigma_n^2|/\sigma_n^4$.

Utilizando ainda do mesmo conjunto de dados esparsa, realizaremos o mesmo estudo efetuado acima, porém utilizando agora a estatística $T_n = \overline{X} = \dfrac{1}{n}\sum_{i=1}^{n} X_t$, que nesse conjunto de dados retrata a freqüência relativa do símbolo 1. Os resultados são apresentados na Tabela 3.5.2.

	$E(\hat{\sigma^*}_n^2)$	$E(\hat{\sigma^*}_n^2) - \sigma_n^2$	$Var(\sigma_n^2)$	EQMR($\hat{\sigma^*}_n^2$)
Método I	0.00585543	-0.02474463	4.76726e-07	0.6544165
Método II	0.005798472	-0.02480159	4.295882e-07	0.6573802

Tabela 3.5.2: Momentos da Variância $\hat{\sigma^*}_n^2$

Nesta segunda análise, ambos os métodos se mostraram novamente muito parecidos, e novamente o método I se mostrou um pouco melhor devido ao seu menor EQMR. Cabe salientar ainda, que se trabalharmos em um processo onde as estimativas das probabilidades de transição apresentarem algum erro, o método II tende a ser mais robusto, pelo fato de não trabalhar com as estimativas.

4 Aplicações aos Dados Lingüísticos

O português moderno é uma língua particulamente interessante para ser estudada na nossa abordagem, sendo que tanto o português do Brasil quanto o português Europeu (abreviados aqui por PB e PE respectivamente), apresentam o mesmo conjunto de palavras em sua estrutura (léxico). No entanto estas linguas apresentam diferentes sintaxes e diferentes prosódias, isto é, não só ordenam as suas palvras de maneiras diferentes como também implementam sentenças com ritmos diferenciados. Neste trabalho, analisamos textos jornalísticos de ambas as línguas usando o algoritmo Contexto.

4.1 Questões Linguísticas

Nesta seção detalharemos de que forma os dados foram obtidos e codificados, para que estes retratassem evidências estatísticas na detecção de ritmos nos textos escritos. A amostra na qual trabalharemos, foi selecionada aleatoriamente de todas as reportagens dos jornais "Folha de São Paulo" (PB) e "Público" (PE) nos anos de 1994 e 1995, Sendo 40 textos de PB e 40 textos PE (20 de cada língua por ano).

Os textos foram amostrados através de um processo de amostragem aleatória simples, que consiste em fazer um sorteio dentre todos os jornais com a mesma probabilidade de serem selecionados. Os textos fazem parte dos Corpus mantido pelo projeto AC/DC (Acesso a Corpora/Disponibilização de Corpora) disponibilizado no endereço eletrônico http://lusiadas.linguateca.pt/acesso/corpus.php?corpus=acesso a Corpora/Disponibilização de Corpora, onde podem ser encontradas as 365 edições dos jornais Público e Folha de São Paulo dos anos de 1994 e 1995. Dentro de cada edição, foram

selecionados textos com mais de 1000 palavras (visto que essa era uma condição para estimar árvores com ordem 3 (4 talvez)) e escolhido apenas 1 texto dentre todos.

Após a escolha dos textos, foi analisado o tipo do texto (reportagem) que se tratava. Devido ao objetivo ser o de construir um modelo que retrate como é falado o português através de textos escritos, certos tipos de reportagens não foram utilizadas. Por exemplo, entrevistas, sinopses de filmes, peças de teatros, ou promulgação de leis e coletâneas de textos. Desta forma eles foram retirados da amostra, e só permaneceram reportagens gerais, das diversas seções dos jornais.

Na codificação dos textos escritos, tentou-se retratar a idéia de que as conjecturas das classes rítmicas são caracterizadas pelo fato de que elas designam relevância para diferentes domínios prosódicos. A questão então é como reaver em termos estatísticos domínios relevantes para a codificação de uma amostra de um texto escrito (Galves, Galves, Garcia, Peixoto,Lacerda e Leonardi (2005)).

A codificação virá pela atribuição de dois símbolos para cada sílaba do texto, de acordo com o seguinte regra:

- se esta é tônica ou não;
- se é início de palavra prosódica ou não,

onde a palavra prosódica é definida como sendo uma palavra léxica juntamente com as palavras não acentuadas que a precedem (Vigário (2003)).

Usaremos inicialmente o conjunto $\{0,1\}^2$ como o conjunto de símbolos adotados, onde o primeiro símbolo indicará início ou não de palavra prosódica e o segundo símbolo indicará se a sílaba é tônica ou não. Como um método de simplificação nós usaremos uma expansão binária identificando $(0,0) = 0$, $(0,1) = 1$, $(1,0) = 2$, $(1,1) = 3$. Adicionalmente nós iremos atribuir um símbolo extra (4) para o final de cada sentença, marcado por ponto final, ponto interrogação, ponto exclamação etc. Portanto, obtemos para o nosso processo o alfabeto, $\mathrm{A} = \{0,1,2,3,4\}$.

Para melhor compreensão da codificação considere a sentença "O menino já comeu o doce". Esta sentença é divida em quatro palavras prosódicas e será codificada como segue,

Sentença	O	me	ni	no	já	co	meu	o	do	ce	.
Início de palavra prosódica	1	0	0	0	1	1	0	1	0	0	
Sílaba tônica	0	0	1	0	1	0	1	0	1	0	
Código	2	0	1	0	3	2	1	2	1	0	4

Podemos notar que os elementos formados pelos dígitos binários 0 e 1 indicam se as condições "início de palavra prosódica" ou " silaba tônica" são satisfeitos ou não, sendo que na linha código temos o par binário de maneira sintetizada. O símbolo 4 indica final de sentença.

As palavras prosódicas, formam um conjunto finito e seu número de sílabas é limitado por uma constante M, que para propósitos práticos pode ser tomada como $M = 15$. Temos ainda que de acordo com a definição por nós adotada, qualquer palavra prosódica deve conter uma e somente uma sílaba tônica (codificada por 1 ou 3). Além disso, o português somente permite que uma sílaba tônica possa ser seguida, no máximo, por três sílabas atônas dentro da palavra prosódica.

Finalmente, podemos ressaltar ainda que as sentenças são formadas pelas concatenações das palavras prosódicas, e pelos símbolos que seguem aos finais de sentenças (codificado pelo símbolo 4), que podem somente ser seguido pelos símbolos (2 ou 3).

O programa utilizado para a codificação dos dados, é sensível com relação a alguns símbolos, por exemplo, é preciso retirar aspas, escrever todos os numerais por extenso, escrever "como se lê" as siglas, e retirar frases com palavras estrangeiras. O programa segue a norma gramatical da língua portuguesa e os códigos poderiam simbolizar sinais errados.

Os nomes próprios não foram alterados. Foram também retirados os hifens de todas as palavras, com exceção de verbos como "falaram-se". Palavras com o prefixo "ex" como "ex-deputado" foram reescritas como "exdeputado", reticências foram substituídas por apenas um ponto, assim como o ponto de exclamação e interrogação. Foram também retirados todos os parênteses nos textos e o símbolo % foi escrito como "porcento". Todas estas questões referentes a marcas lingüísticas foram supervisionadas por profissionais da área que revisaram todos os textos que geraram posteriormente as árvores.

Um último aspecto importante, é que a nossa sugestão para a reamostragem de uma cadeia de Markov de alcance variável deve conter, devido ao nosso método de codificação, certas condições algébricas, sendo elas:

- Os símbolos 1 e 3 devem ser seguidos por no máximo três 0‘s e os símbolos 2, 3 ou 4.
- O símbolo 2 deve ser seguido por qualquer números de 0‘s e o símbolo 1.
- O símbolo 4 deve ser seguido pelos símbolos 2 ou 3.

Vale lembrar, que o programa para codificação, pode ser gratuitamente adquirido para propósitos acadêmicos em URL *www.ime.usp.br/∼ tycho/prosody/vlmc/tools /silaba.pl*.

4.2 Estimação

Apresentaremos nesta seção, o conjunto das 80 árvores obtidas na simulação pelo algoritmo Contexto (critérios AIC e BIC). Visando ainda obter algum conhecimento a respeito do comportamento do conjunto de dados, realizaremos uma breve análise descritiva. Cabe ressaltar, que a escolha do algoritmo Contexto se deu pelos resultados apresentados na Seção 2.4. Temos também que, para efeito de notação, iremos classificar as árvores obtidas por Árvore1, Árvore2 e assim por diante.

Usando primeiramente o algoritmo Contexto com critério BIC, foram observados em ambos os idiomas basicamente 6 tipos de árvores, que são dadas pelas Figuras 4.2.1- 4.2.6, respectivamente. A Tabela 4.2.2 expressa a freqüência dessas árvores para cada um dos idiomas em estudo.

Em todas as 80 árvores que foram geradas pelo critério BIC, apenas os contextos 0 e (0,0) apresentaram alguma ramificação. Todas as árvores também apresentam profundidade 3, ou seja, no máximo deve-se voltar 2 passos no passado para predizer o próximo símbolo. Em geral a estrutura das árvores é semelhante, as diferenças apareceram apenas na terceira geração de ramos. Não há nenhuma informação que as diferem, de certo modo seguem um padrão.

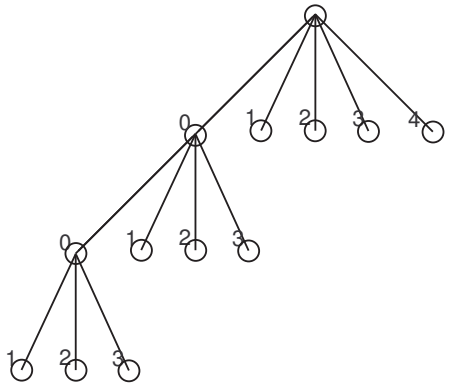


Figura 4.2.1: Árvore-1

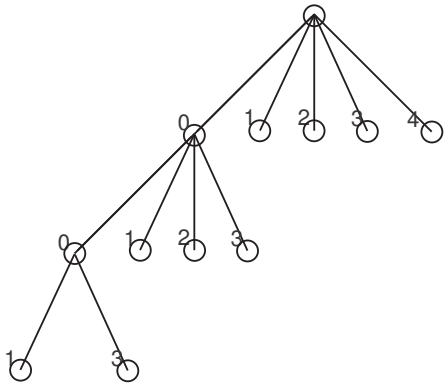


Figura 4.2.2: Árvore-2

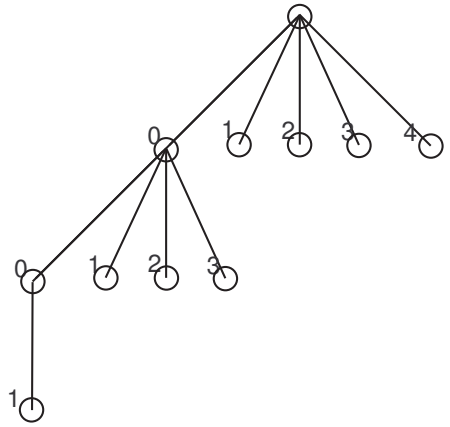


Figura 4.2.3: Árvore-3

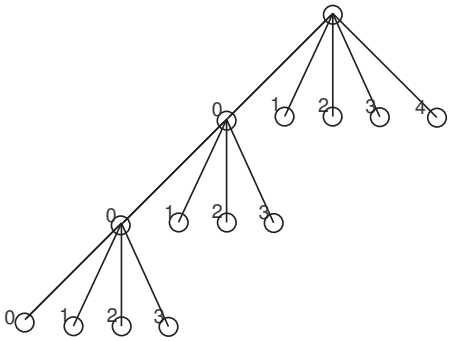


Figura 4.2.4: Árvore-4

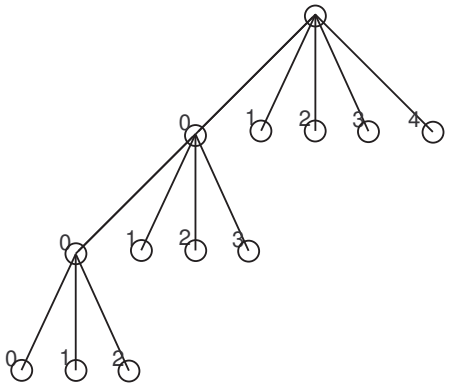


Figura 4.2.5: Árvore-5

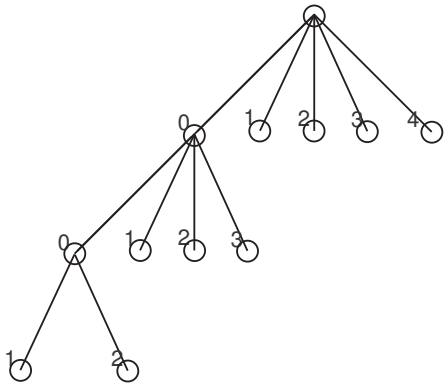


Figura 4.2.6: Árvore-6

Árvores	PB	PE
Árvore-1	12	13
Árvore-2	15	10
Árvore-3	11	13
Árvore-4	1	0
Árvore-5	1	3
Árvore-6	0	1
Total	40	40

Tabela 4.2.1: Freqüência das árvores geradas pelo algoritmo Contexto via "BIC".

As Figuras 4.2.1 - 4.2.18, apresentam respectivamente as 18 árvores estimadas pelo algoritmo Contexto usando o critério AIC.

A Tabela 4.2.2 apresenta as freqüências relativas das árvores obtidas pelo algoritmo Contexto, utilizando o critério AIC.

Como podemos obervar, com critério AIC obtivemos árvores maiores, algumas com profundidade 4, bem como uma maior discrepância entre os ajustes. Porém, as estimadas em ambos os processos continuam concentradas em árvores com a mesma estrutura (Árvore-1 à Árvore-6).

Finalizaremos esta seção, apresentando alguns aspectos interessantes com relação a freqüência dos contextos, em ambos os processos.

- Nas árvores do português brasileiro e do português europeu, os contextos (1,0,0), (1,0), (2,0), (3,0), 1, 2, 3 e 4 aparecem em 100% dos ajustes;
- O contexto (0,0,0), apareceu em média $12,5\%$ das reportagens da Folha de São Paulo e em $18,75\%$ do jornal português Público;
- O contexto (2,0,0) aparece em média $51,25\%$ das árvores do português brasileiro e em 60% do português europeu;
- O contexto (3,0,0) foi observado em média $76,25\%$ dos textos da "Folha de São Paulo" e em $67,5\%$ do "Público".

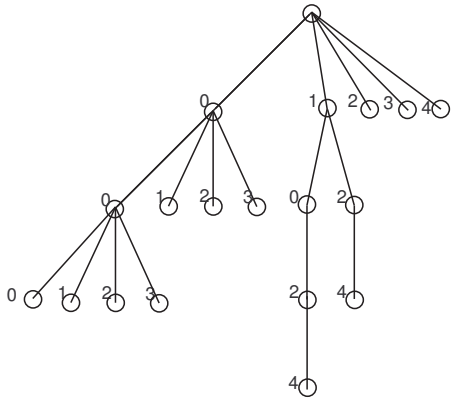


Figura 4.2.7: Árvore-7

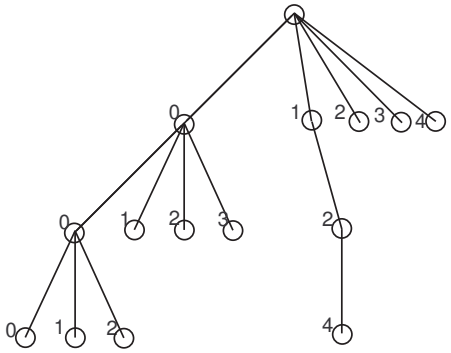


Figura 4.2.8: Árvore-8

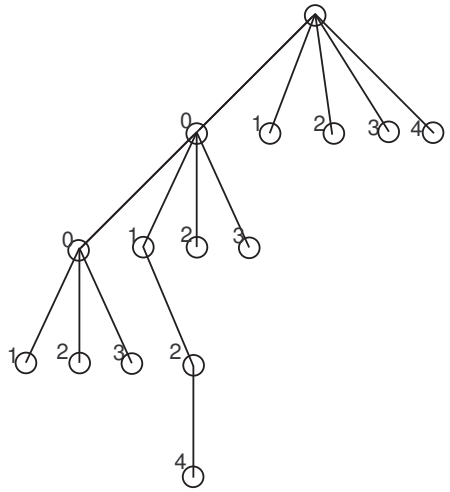


Figura 4.2.9: Árvore-9

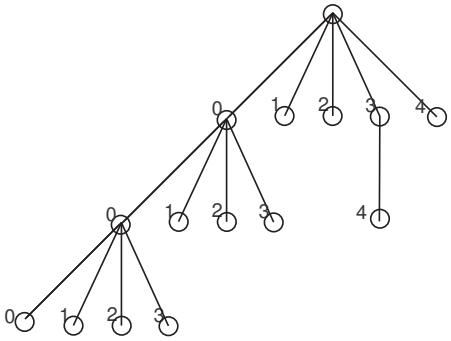


Figura 4.2.10: Árvore-10

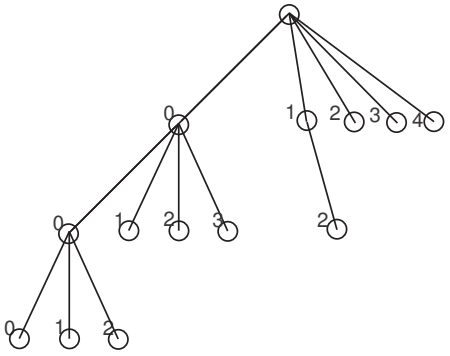


Figura 4.2.11: Árvore-11

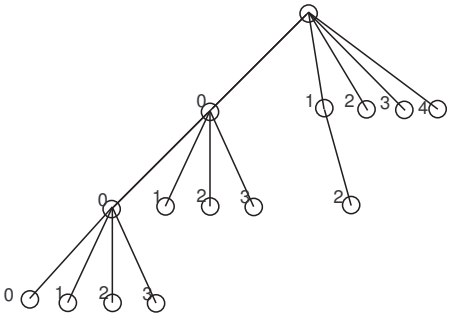


Figura 4.2.12: Árvore-12

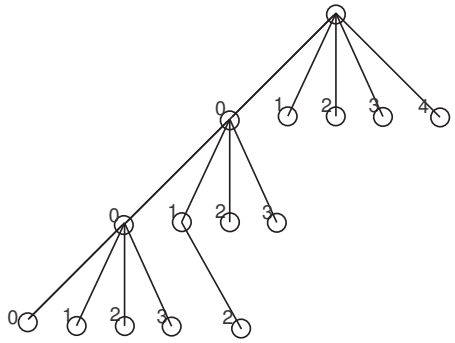


Figura 4.2.13: Árvore-13

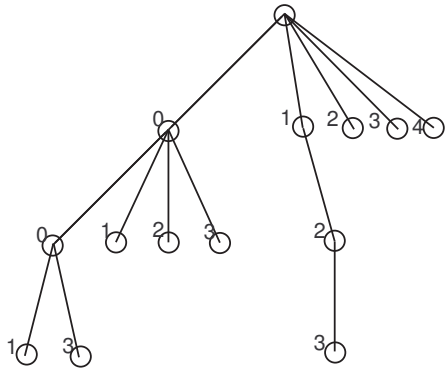


Figura 4.2.14: Árvore-14

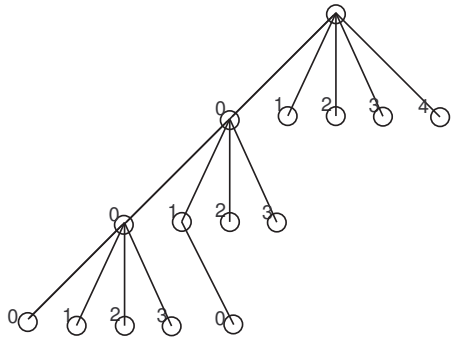


Figura 4.2.15: Árvore-15

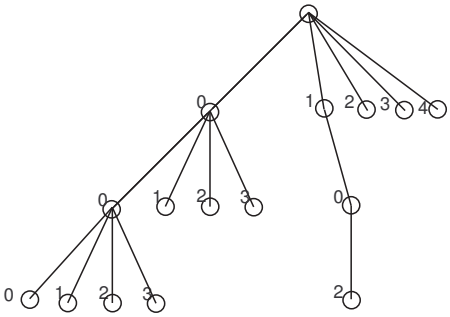


Figura 4.2.16: Árvore-16

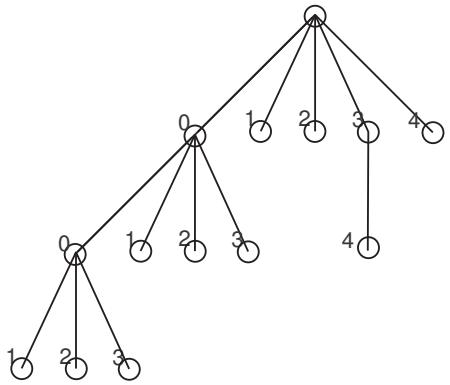


Figura 4.2.17: Árvore-17

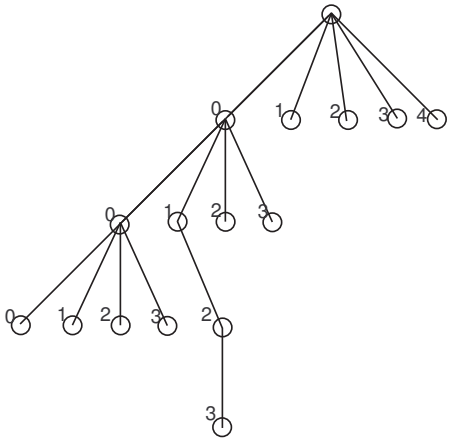


Figura 4.2.18: Árvore-18

Árvores	PB	PE
Árvore-1	17	15
Árvore-2	11	8
Árvore-3	2	0
Árvore-4	1	0
Árvore-5	2	8
Árvore-6	1	1
Árvore-7	1	0
Árvore-8	1	0
Árvore-9	1	0
Árvore-10	1	0
Árvore-11	1	0
Árvore-12	1	0
Árvore-13	0	2
Árvore-14	0	1
Árvore-15	0	1
Árvore-16	0	2
Árvore-17	0	1
Árvore-18	0	1
Total	40	40

Tabela 4.2.2: Freqüência das árvores geradas pelo algoritmo Contexto via "AIC".

4.3 Teste de hipótese para Árvores Aleatórias

Nesta seção, apresentaremos o teste de hipótese proposto por Busch, Ferrari, Flesia, Fraiman, Grynberg e Leonardi (2007) para testar a hipótese da igualdade das distribuições de árvores aleatórias, que evoluem no tempo com gerações discretas. O teste é baseado na comparação entre as estruturas médias que caracterizam duas amostras de populações distintas de árvores. Busca-se então testar se a diferença (distância) entre essas estruturas são estatisticamente significantes para rejeitar uma igualdade de distribuições.

Considere um alfabeto finito A $= \{1, 2, .., m\}$, com $m \geq 2$, representando o número máximo de descendentes que um nó pode ter. Denote também o conjunto das finitas seqüências de elementos em A por $V = \{1, 2, ..., m, 11, 21, ..., m1, 12, 22, 32, ...\}$, representando todos os contextos possíveis de uma árvore. Dado $w = (a_k, ..., a_1)$ um contexto da árvore, diremos que este fará parte da geração $k+1$, que será denotada por $gen(w) = k+1$.

Desta forma, somente a raiz será considerada da geração 1.

Seja $\Gamma = \cup_{k=1}^{\infty} \mathrm{A}^{\{1,...,k\}}$ (Definição 2.1.3) o conjunto de todas as árvores. Para cada árvore τ subconjunto de Γ, defina uma $t : V \to \{0, 1\}$ satisfazendo

$$t(v) \geq t(va) \tag{4.3.1}$$

com $v \in V$ e $a \in \mathrm{A}$. Para efeito de notação denotaremos por função identificadora de τ. Temos portanto que $t(v) = 1$ se $v \in \tau$ e zero caso contrário. Podemos notar ainda que os contextos de uma árvore estarão bem caracterizados por esta função.

Considere agora, $\phi : V \to \mathbb{R}^+$ como sendo uma função estritamente positiva tal que $\sum_{v \in V} \phi(v) < \infty$ e defina a distância entre duas árvores τ_1 e τ_2 por

$$d(\tau_1, \tau_2) = \sum_{v \in \Gamma} \phi(v)(t_1(v) - t_2(v))^2, \tag{4.3.2}$$

onde t_1 e t_2 são as respectivas funções identificadoras de τ_1 e τ_2. A $\phi(.)$ pode ser por exemplo a função $\phi(v) = Z^{gen(v)}$ com $1 < Z < \dfrac{1}{m}$, que não penaliza muito a primeira geração.

Com esta distância têm-se então que (Γ, d) é um espaço métrico compacto. Denote $\mathbf{B}$ como sendo a sigma-álgebra de Borel formada pelos subconjuntos de Γ.

Definição 4.3.1. *Dado* $\mathbf{T} = (T_1, T_2, ...T_n)$ *uma amostra aleatória de* T *(árvores independentes com a mesma lei de* T*). A distância média empírica da amostra para a árvore* τ *é definida por*

$$g_{\mathbf{T}}(\tau) := \frac{1}{n} \sum_{i=1}^{n} d(T_i, \tau).$$

Seja ν, $\nu^{'}$ distribuições em $\mathcal{Q}_f$ (o espaço de medidas de probabilidades que concentra massa em árvores com um número finito de nós). A objetivo então é testar

$$H_0 : \nu = \nu^{'} \qquad H_1 : \nu \neq \nu^{'} \tag{4.3.3}$$

usando amostras aleatórias i.i.d. $\mathbf{T} = (T_1, ..., T_n)$ e $\mathbf{T}' = (T_1^{'}, ..., T_m^{'})$ com distribuições ν e $\nu^{'}$ respectivamente.

O teste **BFFS** introduzido por Balding, Ferrari, Fraiman e Sued (2004) propõe a seguinte estatística

$$\sup_{\tau\in\Gamma}|W_{\mathbf{T},\mathbf{T}'}(\tau)| = \sup_{\tau\in\Gamma}\sqrt{n}|g_{\mathbf{T}}(\tau) - g_{\mathbf{T}'}(\tau)|.$$

A hipótese nula será rejeitada a um nível α quando $\sup_{\tau\in\Gamma}|W_{\mathbf{T},\mathbf{T}'}(\tau)| > q_\alpha$.

Na prática, como a distribuição da estatística de teste sob a hipótese nula é desconhecida, utiliza-se dos seguintes passos para obter o quantil sob a hipótese nula.

- Usa-se da reordenação da amostra agrupada para obter cada novo par de amostra, isto é, desta forma reamostramos árvores de ambas as leis. Em seguida, utiliza-se deste par de amostra para se calcular a estatística de teste
- O passo relatado acima é repetido um número fixo de vezes N, e os valores obtidos das estatísticas são ordenados de forma crescente.
- Defina o quantil $q_{(1-\alpha)}$ como sendo o valor que ocupa a posição $(1-\alpha)N$ no vetor formado pelas estatísticas ordenadas.

Como podemos notar, no cálculo do quantil sob a hipótese nula, o teste supõe que a variedade de árvores encontrada na cadeia de Markov de alcance variável (sob a hipótese nula) é a mesma que obtivemos na estimação do nosso conjunto de dados (PB e PE). Isso se deve ao fato de que a reamostragem é feita a partir da reordenação dos textos (árvores).

Para o cálculo do valor descritivo (empírico), definimos primeiramente $\mathbf{L}$ como sendo um vetor contendo todas os valores obtidos pelo teste nas amostras reordenadas, que para efeito de notação diremos que é de tamanho N. Considere também l como sendo o valor do teste nas amostras originais $\mathbf{T}, \mathbf{T}'$. Assim o valor descritivo será dado por:

$$\text{Valor-Descritivo} = \frac{\sum_{j=1}^{N} I\{l < L_j\}}{r},$$

onde L_j é a j-ésima coordenada de $\mathbf{L}$.

Quando o teste foi aplicado nos ajustes obtidos pelo algoritmo Contexto com critério BIC, obtivemos um valor descritivo de aproximadamente 0.35, ou seja, não temos indícios estatísticos para rejeitar a hipótese nula. Assim, segundo o resultado do teste, ambas as línguas (PB e PE) apresentam a mesma distribuição. Esse resultado era esperado, já que as árvores ajustadas em ambas as amostras com o critério BIC, são muito parecidas. Enfim, provavelmente a grande diferença apresentada nestes ajustes está relacionada às probabilidades de transição (o que não é detectado neste teste).

Realizando agora o mesmo teste nos ajustes obtidos pelo algoritmo Contexto com critério AIC, o valor descritivo obtido foi de aproximadamente 0.37, isto é, aqui também não rejeitamos a hipótese nula. Portanto, segundo o teste realizado nestas amostras (critério AIC) o PB e PE também provém da mesma distribuição que atribui massa a árvores com um número finito de nós. Esse resultado pode ser interligado com o fato ressaltado na Seção 4.2, isto é, apesar desse critério fornecer uma maior variedade de árvores, ambas as amostras (PB e PE) continuam concentrando suas estimativas em árvores idênticas.

4.4 Teste de Hipótese para Cadeias de Makov de Alcance Variável sob Reamostragem

Nesta seção, abordaremos um estudo de teste de hipótese utilizando da estatística de teste **BFFS** apresentada na Seção 4.3, e do método de reamostragem proposto neste trabalho (Seção 3.5).

O objetivo então é testar a seguinte hipótese:

$$H_0 : (\tau, P) = (\tau', P')$$

onde (τ, P) é a cadeia de Markov de alcance variável que gera textos de PB, e (τ', P') é a cadeia de Markov de alcance variável que gera textos de PE.

Ressaltamos que os ajustes realizados aqui, serão efetuados através do algoritmo Contexto com critérios de seleção AIC e BIC.

Esta análise procederá da seguinte maneira:

- Primeiramente, supomos que ambas as línguas PB e PE provenham de uma mesma cadeia de Markov de alcance variável, isto é, trabalhamos sobre a hipótese nula.
- Utilizando de cada conjunto de dados (PB e PE), cria-se uma única lista contendo todas as seqüências encontradas nos 80 textos codificados (Seção 4.1). Vale lembrar que seqüências que ocorrem por mais de uma vez nesses textos, aparecem de forma repetida na lista. Após esse passo, temos então uma lista de seqüências.
- Realizaremos 500 reamostragens do nosso conjunto de dados (textos de PB e PE), isto é, cada texto (de PB e PE) foi reamostrado (respeitando o número de seqüências do texto original), por 500 vezes utilizando a técnica proposta neste trabalho. Portanto após esta reamostragem, obtivemos então 500 conjuntos de dados de PB e 500 de PE, contendo 40 textos em cada conjunto;
- A partir dessas reamostragens a estatística **BFFS** do teste foi executada, e o valor descritivo (empírico) foi calculado.

As Figuras 4.4.1 - 4.4.3 apresentam respectivamente o histograma, a densidade (estimada) e o diagrama de dispersão dos resultados obtidos nos testes realizados nas reamostras, quando utilizamos do critério AIC no ajuste das árvores. A curva da densidade que aparece na Figura 4.4.2 representa uma estimativa não paramétrica da densidade obtida pelo método Kernel (Kernel Gaussiano), utilizando "software R".

Como podemos notar, quando trabalhamos com o critério AIC os valores obtidos pelo teste são mais freqüêntes entre 0.015 e 0.02, com mediana de aproximadamente 0.018, e alguns valores extremos acima de 0.035.

O valor descritivo obtido com esse ajuste (AIC) foi de aproximadamente 0.039. A grande diferença encontrada aqui, é que na nossa reamostragem obtivemos cerca de 250 árvores distintas, diferentemente das 18 que o teste anterior utiliza na reordenação dos dados para o cálculo do seu valor descritivo.

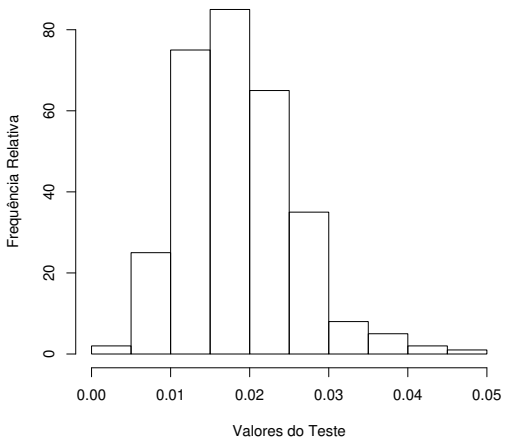


Figura 4.4.1: Histograma (AIC)

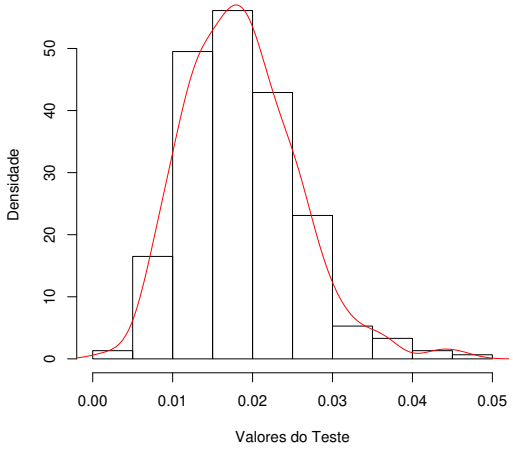


Figura 4.4.2: Densidade (AIC)

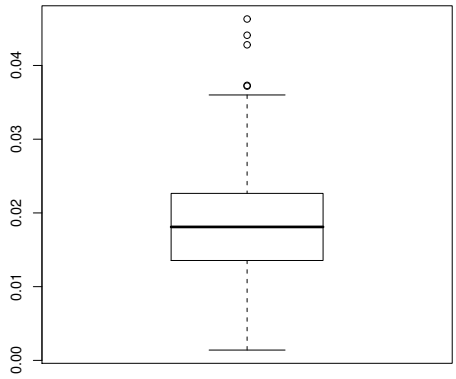


Figura 4.4.3: Diagrama de Dispersão (AIC)

Com relação aos ajustes realizados com o critério BIC, as Figuras 4.4.4 - 4.4.6 apresentam respectivamente o histograma, a densidade (estimada) e o diagrama de dispersão dos resultados obtidos nos testes realizados nas reamostras.

Neste caso (critério BIC), os valores obtidos pelo teste são mais freqüêntes entre 0.005 e 0.01, com mediana de aproximadamente 0.008, e alguns valores extremos acima de 0.02.

Diferentemente do resultado apresentado com o critério AIC, o valor descritivo obtido com esse ajuste (BIC) foi de aproximadamente 0.37. Lebramos que neste caso, nas reamostras, obtivemos aproximadamente 18 tipos de árvores distintas.

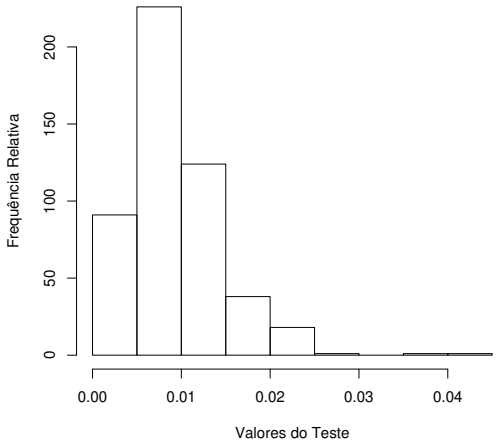


Figura 4.4.4: Histograma (BIC)

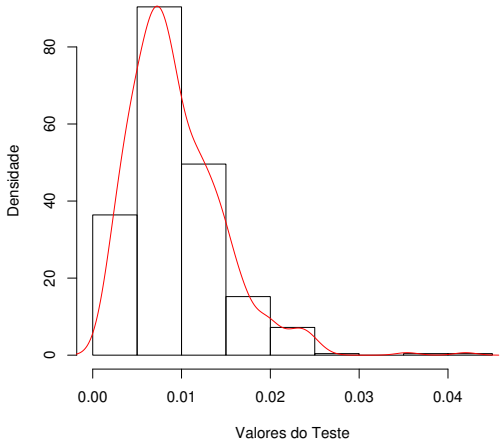


Figura 4.4.5: Densidade (BIC)

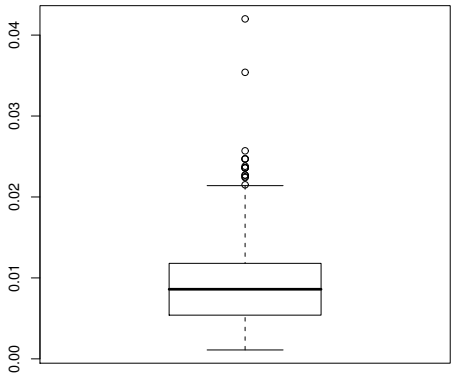


Figura 4.4.6: Diagrama de Dispersão (BIC)

5 Conclusões

Neste capítulo apresentaremos alguns aspectos interessantes inerentes ao estudo realizado neste trabalho.

Com relação aos algoritmos de estimação de árvores adotados neste estudo, algoritmo de contexto com critérios de seleção AIC e BIC, obtivemos as seguintes características apresentadas em nossas estimações (Seção 2.4).

Quando trabalhamos com os conjuntos de dados esparsa, fica bem claro que o algoritmo Contexto com critério BIC estima melhor as árvores. Essa característica pode ser justificada pelo fato de que nestas árvores estimadas o símbolo 1 nunca se ramifica, o que não ocorreu com a estimação utilizando o critério AIC.

Por outro lado, como foi apresentado na Tabela 2.4.1, obtivemos em simulações realizadas em dados "simulados" (através das probabilidades de transição) de árvores do conjunto de dados da Seção 4.2, que na maioria das vezes o critério AIC (neste caso) acaba por ajustar melhor as árvores estimadas. Esse fato pode ser explicado seguindo do princípio de que para amostras pequenas com grande número de parâmetros a ser estimado, o critério AIC penaliza menos os ajustes, isto é, tende a estimar menores valores de ponto de corte (2.2.3).

Como não temos indícios de como é a distribuição das árvores que provém tanto de PB quanto de PE, acabamos optando então pelo algoritmo Contexto com critério AIC.

A respeito do nosso método de reamostragem (Seção 3.5), quando aplicado na estimação da variância de alguns estimadores, vimos que este tem um comportamento bem próximo ao encontrado por um método de reamostragem consistente para esses esti-

madores, ou seja, para aquele caso (que é um conjunto de dados com ponto de renovação) obtivemos um bom resultado.

Tratando agora dos resultados obtidos pelo teste de hipótese para cadeias de Makov de alcance variável sob reamostragem, vimos que para o caso em que o critério AIC é utilizando nos ajustes das árvores, o valor descritivo obtido nos leva a concluir que existe evidências estatísticas que para se rejeitar a hipótese nula. Esse fato pode ser explicado, pois quando utilizamos este critério, a quantidade de árvores geradas nas reamostras é absurdamente maior que as 18 apresentadas nos ajustes das amostras. Ou seja, para o cálculo do valor descritivo com as reamostras utilizamos uma maior variedade de árvores, que acabaram resgatando a diferença entre as cadeias de Markov de alcance variável.

Já com o critério BIC obtivemos valor que nos leva a concluir que as línguas provém de uma mesma cadeia de Markov de alcance variável, isto é, não existe evidências estatísticas que nos leve a rejeitar a hipótese nula. O fato de que o algoritmo Contexto com esse critério penaliza muito os ajustes, pôde também ser observada aqui, pois em todo nosso conjunto de reamostras obtivemos apenas 18 tipos de árvores distintas.

Enfim, como acabamos optando pelo algoritmo Contexto com o critério AIC, temos então que com o valor descritivo (0.039) obtido neste caso rejeitamos a hipótese nula, isto é, segundo nosso método de reamostragem temos evidências estatísticas de que PB e PE provém de cadeias de Markov de alcance variável distintas.

Referências

Balding, D. Ferrari, P., Fraiman, R., Sued, M. (2004). Limit theorems for sequences of random trees. ArXiv: math.PR/0406280v2.

Busch, J.R. Ferrari, P. Flesia, G. Fraiman, R. Grynberg, S. Leonardi, F. (2007). Testing statistical hypothesis on Random Trees. ArXiv: math.ST/0603378v3.

Basawa, I. V., and Prakasa Rao B. L. S. (1980). Statistical Inference for Stochastic Process, Academic Press, Londom.

Bejerano, G. & Yona, G. (2001). Variations on probabilistic suffix trees: statistical modeling and prediction of protein families, *Bioinformatics*, **17**(1): 23-43.

Billingsley, P. (1961). *Statistical Inference for Markov Processes*, The University of Chicago Press.

Bühlmann, P. & Wyner, A.J. (1999). Variable length Markov chains. *Annals of Statistics* **27**, 480-513.

Csiszár, I. & Talata, Z. (2006). Context tree estimation for not necessarily finite memory processes, via bic and mdl, *Information Theory, IEEE Transactions on* **52**(3): 1007-1016.

Duarte, D., Galves, A. & Garcia, N. (2006). Markov approximation and consistent estimation of unbounded probabilistic suffix trees, *Bull, Braz. Math. Soc.* p. Aceito.

Efron, B. (1979). Bootstrap methods: another look at the jackknife. *Ann. Statist.*, **7**, 1-26.

Ferrari, F. & Wyner, A. (2003). Estimation of general stationary processes by variable length Markov chains, *Scand. J. Statist.*, **30**(3): 459-480.

Kulperger, R. J., Prakasa Rao, B. L. S. (1989). Bootstrapping a finite state markov chain, *The Indian Journal of Statistics*, Vol. 51, pp. 178-191.

Künsch, H. R. (1989), The Jackknife and the Bootstrap for General Stationary Observations, *The Annals of Statistics*, Vol. 17, No. 3, pp. 1277-1241.

Lifshits, B. A. (1978). On the central limit theorem for Markov chains. *Th. Prob. and Appl.,* **2**, 279-296.

Galves, A., Galves, C., Garcia, N.L., Peixoto, C., Lacerda, D. & Leonardi, F. (2005). Alternatives to the Context Algoritm for estimating VLMC with an application to linguistic. Preprint.

Mächler, M., The VLMC package, (2005). Can be downloaded from http://cran.r-project.org/doc/packages/VLMC. Pdf.

Mächler, M. and Bühlmann, P. (2004). Variable length Markov chains: methodology, computing, and software, *J. Comput. Graph. Statist.*, Vol. 13, pp. 435-455.

Nagaev, S. V. (1957). Some limit theorems for Markov chains. *Th. Prob. and Appl.,* **2**, 378-406.

Rissanen, J. (1983). A universal data compression system, *IEEE Trans. Inform. Theory*, **29**(5): 656-664.

Ron, D., Singer, Y., & Tishby, N. (1996). The power of amnesia: Learning probabilistic automata with variable memory length, *Machine Learning*, **25**(2-3): 117-149.

Sirazhdinov, S. Kh., and Formann Sh. K. (1983). On estimates of the rate of convergence in the central limit theorem for homogeneous Markov chain. *Th. Prob. and Appl.*, **28**, 229-239.

The R Project for Statistical Computing, http://www.rproject.org.

Vigário, M. (2003). The prosodic word in European Portuguese, Mouton de Gruyter.

6 Apêndice

Apresentaremos nos Apêndices a seguir os códigos fonte do programas executados, bem como algumas características dos conjuntos de dados utilizados neste trabalho. Vale lembrar que tais códigos são executáveis no software "R".

6.1 Apêndice A

Apresentaremos primeiramente os dois códigos fonte dos programas que ajustam as árvores (conjunto de dados em um mesmo diretório) utilizando o pacote VLMC, com critérios de seleção AIC e BIC respectivamente (Seção 2.4). Estes programas tem como saída as árvores ajustadas e plotadas em formato "ps".

```
#####################################################################
# Ajuste de uma cadeia de Markov de alcance variável usando  AIC ####
#####################################################################
## primeiramente deve-se colocar o diretório do R, para onde se localizam
## os conjunto de dados
library(VLMC)
textos <- list.files(path = ".", pattern = "txt", all.files = FALSE,
full.names = FALSE, recursive = FALSE)
alfabeto<-5
for(u in 1:length(textos))
{
arquivo <-  paste(textos[u], sep = " ", collapse = "")
```

```
texto <- scan(arquivo)
n<-length(texto)
alfa<-0.01
k<-0.5*qchisq(1-alfa,(alfabeto-1))
aic<-1:2
c<-1:2
j<-k+20
q<-k+0.00001
i<-1
while(q <= j)
      {
       c[i]<-q
       vc<-vlmc(texto,cutoff=c[i],threshold.gen = 5) #realiza ajuste
       aic[i]<-as.numeric(AIC(vc))##guarda valores de AIC já calcu-
       ## lados no pacote VLMC
       i<-i+1
       q<-q + 0.01
      }
posicao<-which.min(aic)## Encontra a posição do menor valor de AIC
ajuste<-vlmc(texto,cutoff=c[posicao],threshold.gen = 5)
draw(ajuste)
denden1sp<- as.dendrogram(ajuste)## desenha árvore com menor AIC
plot(denden1sp,type="tr",nodePar=list(pch=c(1,16)),center=TRUE,
main=format(ajuste$call))
savePlot(filename=textos[u],type=c("ps"))
rm(arquivo,texto,n,alfa,k,c,j,q,i,vc,aic,posicao,ajuste,denden1sp)
}##Fim do for u
rm(u,alfabeto,textos)

#########################################################################
# Ajuste de uma cadeia de Markov de alcance variável usando BIC #####
#########################################################################
```

```
## primeiramente deve-se colocar o diretório do R, para onde se localizam
## os conjunto de dados
textos <- list.files(path = ".", pattern = "txt", all.files = FALSE,
full.names = FALSE, recursive = FALSE)
alfabeto<-5
for(u in 1:length(textos))
{
arquivo <-  paste(textos[u], sep = " ", collapse = "")
texto <- scan(arquivo)
n<-length(texto)
alfa<-0.01
k<-0.5*qchisq(1-alfa,(alfabeto-1))
bic<-1:2
c<-1:2
j<-k+20
q<-k+0.00001
i<-1
while(q <= j)
        {
        c[i]<-q ##guarda valores do cuttoff
        vc<-vlmc(texto,cutoff=c[i],threshold.gen = 5)#realiza ajuste
        card<-as.numeric(vc$size[2])
        bic[i]<- as.numeric(-2*logLik(vc)+(alfabeto-1)*log(n)*card)
        ## Calcula e guarda valores do BIC, o valor da log-verrossi-
        ##lhança é dado no pacote VLMC
        i<-i+1
        q<-q + 0.01
        }
posicao<-which.min(bic) ## Encontra a posição do menor valor de BIC
ajuste<-vlmc(texto,cutoff=c[posicao],threshold.gen = 5)
draw(ajuste)
```

```
denden1sp<- as.dendrogram(ajuste)## desenha árvore com menor BIC
plot(denden1sp,type="tr",nodePar=list(pch=c(1,16)),center=TRUE,
main=format(ajuste$call))
savePlot(filename=textos[u],type=c("ps"))
rm(arquivo,texto,n,alfa,k,c,j,q,i,vc,bic,posicao,ajuste,denden1sp,card)
}
rm(u,alfabeto,textos)
```

6.2 Apêndice B

O código fonte que apresentaremos agora foi utilizado na reamostragem feita sob a hipótese nula na Seção 4.4. Isto é, dado o nosso conjunto de dados (PB e PE) ele cria pastas "bootstrap" que contém as reamostras.

```
#########################################################################
# Colocar o diretório onde se encontra os dados de português brasileiro #
#########################################################################

pb<-list()
brasileiro<-list.files(path = ".", pattern = "bin", all.files = FALSE,
full.names = FALSE, recursive = FALSE)
i<-1
for(i in 1:length(brasileiro))
{
arquivo <- as.vector(scan(brasileiro[i]))
pb[[i]]<-as.vector(arquivo)
rm(arquivo)
}
rm(i)
#########################################################################
```

```
# Colocar o diretório onde se encontra os dados de português europeu #
#####################################################################
pe<-list()
europeu<-list.files(path = ".", pattern = "txt", all.files = FALSE,
full.names = FALSE, recursive = FALSE)
i<-1
for(i in 1:length(europeu))
{
arquivo<- as.vector(scan(europeu[i]))
pe[[i]]<-as.vector(arquivo)
rm(arquivo)
}
rm(i)
#####################################################################
#mudar para um diretório de trabalho e contruir as pastas "folha"#
#e "publico", estas pastas conterão as reamostras                  #
#####################################################################
##na pasta folha serão guardados as reamostras de PB e
##consequêntemente os de PE na pasta publico
reamostras<-500   ####número de reamostras desejadas
nome_jornal<-1:2
nome_jornal[1]<-as.character("folha")
nome_jornal[2]<-as.character("publico")
diretorio_trabalho <- " ##colocar aqui diretório de trabalho"
###nomear o diretórios dos jornais
caminho_jornal<-1:2
i<-1
##nomear os caminhos dos textos PB e PE
for(i in 1:2)
{
caminho_jornal[i]<-paste(diretorio_trabalho,nome_jornal[i],"/", sep="")
```

```
}
rm(i)
dir<-as.character("bootstrap")
indice<-1:2
nome_pasta<-1:2
j<-1
for(j in 1:reamostras)
{
indice[j]<-as.character(j)
nome_pasta[j]<-paste(dir,indice[j], sep="")   ## nomeia as pastas bootstrap
}
rm(j,indice,dir)
#########################################################################
##############Carregar a lista de sequências#############################
#########################################################################
lista_de_amostra<-list()
tam<-1
quantidade_texto_pb<-length(pb)
tamanho_texto_pb<-1:2
renovacao<-4
A<-1
for(A in 1:quantidade_texto_pb)
{
auxiliar<-as.vector(pb[[A]])
tamanho<-length(auxiliar)
q<-1
l<-0
while(q <= tamanho)
        {
        y<-1:1
        i<-1
```

```
        while(auxiliar[q] != renovacao)
          {
          y[i]<-as.numeric(auxiliar[q])
          i<-i+1
          q<-q+1
          }
        y[i]<-as.numeric(auxiliar[q])
        q<-q+1
        lista_de_amostra[[tam]]<-as.vector(y)
        tam<-tam+1
        l<-l+1
        rm(y,i)
       }
tamanho_texto_pb[A]<-l
rm(tamanho,q,auxiliar,l)
}
rm(A)
quantidade_texto_pe<-length(pe)
tamanho_texto_pe<-1:2
renovacao<-4
A<-1
j<-1
for(A in 1:quantidade_texto_pe)
{
auxiliar<-as.vector(pe[[A]])
tamanho<-length(auxiliar)
q<-1
l<-0
while(q <= tamanho)
{
y<-1:1
```

```
i<-1
while(auxiliar[q] != renovacao)
     {
     y[i]<-as.numeric(auxiliar[q])
     i<-i+1
     q<-q+1
     }
y[i]<-as.numeric(auxiliar[q])
q<-q+1
lista_de_amostra[[tam]]<-as.vector(y)
tam<-tam+1
l<-l+1
rm(y,i)
}
tamanho_texto_pe[A]<-l
rm(tamanho,q,auxiliar,l)
}
rm(A,tam,pb,pe)

########## construindo amostras de PB################################
setwd(caminho_jornal[1])
a<-1
##reamostra o primeiro texto (mesmo número de sequências)de PB
## por 500 vezes
for(a in 1:reamostras)
   {
   write(dir.create(nome_pasta[a]))
   caminho<-paste(caminho_jornal[1],nome_pasta[a], sep="")
   setwd(caminho)
   ##gerar a amostra Bootstrap
   r<-1
```

```
   bootstrap<-1:2
   k<-1
   for(k in 1:tamanho_texto_pb[1])
     {
     u<-runif(1,min=1 , max= length(lista_de_amostra))
     n<-length(lista_de_amostra[[u]])
     auxiliar2<-as.vector(lista_de_amostra[[u]])
     for(m in 1:n)
       {
       bootstrap[r]<-as.numeric(auxiliar2[m])
       r<-r+1
       }
     rm(u,n,m,auxiliar2)
    }
    write(bootstrap,file = brasileiro[1],ncolumns = 50,sep = " " )
    setwd(caminho_jornal[1])
    rm(caminho,r,bootstrap,k)
}
rm(a)

###preencher as pastas
for(l in 2: length(brasileiro))
{
a<-1
for(a in 1:reamostras)
   {
   caminho<-paste(caminho_jornal[1],nome_pasta[a], sep="")
   setwd(caminho)
   ##gerar a amostra Bootstrap
   r<-1
   bootstrap<-1:2
```

```
   k<-1
   for(k in 1:tamanho_texto_pb[l])
      {
      u<-runif(1,min=1 , max= length(lista_de_amostra))
      n<-length(lista_de_amostra[[u]])
      auxiliar2<-as.vector(lista_de_amostra[[u]])
      for(m in 1:n)
             {
              bootstrap[r]<-as.numeric(auxiliar2[m])
              r<-r+1
             }
      rm(u,n,m,auxiliar2)
      }
    write(bootstrap,file = brasileiro[l],ncolumns = 50,sep = " " )
    setwd(caminho_jornal[1])
    rm(caminho,r,bootstrap,k)
   }
rm(a)
}
rm(quantidade_texto_pb,tamanho_texto_pb,l)
########## construindo amostras de PE################################
setwd(caminho_jornal[2])
a<-1
for(a in 1:reamostras)
{
write(dir.create(nome_pasta[a]))
caminho<-paste(caminho_jornal[2],nome_pasta[a], sep="")
setwd(caminho)
##gerar a amostra Bootstrap
r<-1
bootstrap<-1:2
```

```
k<-1
for(k in 1:tamanho_texto_pe[1])
   {
   u<-runif(1,min=1 , max= length(lista_de_amostra))
   n<-length(lista_de_amostra[[u]])
   auxiliar2<-as.vector(lista_de_amostra[[u]])
   for(m in 1:n)
         {
         bootstrap[r]<-as.numeric(auxiliar2[m])
         r<-r+1
         }
   rm(u,n,m,auxiliar2)
   }
write(bootstrap,file = europeu[1],ncolumns = 50,sep = " " )
setwd(caminho_jornal[2])
rm(caminho,r,bootstrap,k)
}
rm(a)
###preencher as pastas
for(l in 2: length(europeu))
{
a<-1
for(a in 1:reamostras)
      {
      caminho<-paste(caminho_jornal[2],nome_pasta[a], sep="")
      setwd(caminho)
‘     ##gerar a amostra Bootstrap
      r<-1
      bootstrap<-1:2
      k<-1
      for(k in 1:tamanho_texto_pe[l])
```

```
        {
        u<-runif(1,min=1 , max= length(lista_de_amostra))
        n<-length(lista_de_amostra[[u]])
        auxiliar2<-as.vector(lista_de_amostra[[u]])
        for(m in 1:n)
             {
              bootstrap[r]<-as.numeric(auxiliar2[m])
              r<-r+1
             }
        rm(u,n,m,auxiliar2)
        }
      write(bootstrap,file = europeu[l],ncolumns = 50,sep = " " )
      setwd(caminho_jornal[2])
      rm(caminho,r,bootstrap,k)
      }
rm(a)
}
\rm(l,lista_de_amostra,quantidade_texto_pe,tamanho_texto_pe)
```

6.3 Apêndice C

Nesta seção abordaremos o código fonte que ajusta os dados gerados pela reamostragem, utilizando do pacote VLMC, do critério de seleção BIC, e descrito na Seção 4.4. A saída do programa são árvores ajustadas e organizadas em pastas, juntamente com um arquivo (Quantidades) que fornece as quantidades de cada tipo de árvore encontrada em cada pasta "bootstrap".

```
#####################################################################
#mudar para diretório que contenha as pastas "folha"  e "publico"#
#####################################################################
reamostras<-500   ####número de reamostras desejadas
```

```
REAMOSTRAS<-3000  ###Variável auxiliar
nome_jornal<-1:2
nome_jornal[1]<-as.character("folha")
nome_jornal[2]<-as.character("publico")
diretorio_trabalho <- " ##colocar aqui diretório de trabalho"
###nomear o diretórios dos jornais
caminho_jornal<-1:2
i<-1
##nomear os caminhos dos textos PB e PE
for(i in 1:2)
{
caminho_jornal[i]<-paste(diretorio_trabalho,nome_jornal[i],"/", sep="")
}
rm(i)
dir<-as.character("bootstrap")
indice<-1:2
nome_pasta<-1:2
j<-1
for(j in 1:REAMOSTRAS)
{
indice[j]<-as.character(j)
nome_pasta[j]<-paste(dir,indice[j],sep="")   ## nomeia as pastas bootstrap
}
rm(j,dir)
############################################################
#################### Fazer ajustes  ########################
############################################################
##colocar em diretótio de trabalho que contenha as duas
##pasta (folha e publico) com os bootstrap
library(VLMC)
conj_arvores<-1
```

```
conj<-as.character("arvore")
lista<-list()
local_pasta_arvore<-list()
contador<-1
A<-1
for(A in 1:2)
{
B<-1
for(B in 1:reamostras)
 {
 local<-paste(caminho_jornal[A],nome_pasta[B],"/",sep="")
 setwd(local)
 textos <- list.files(path = ".", pattern = "txt", all.files = FALSE,
 full.names = FALSE, recursive = FALSE)
 alfabeto<-5
 u<-1
 for(u in 1:length(textos))
  {
  arquivo <-  paste(textos[u], sep = " ", collapse = "")
  texto <- scan(arquivo)
  n<-length(texto)
  alfa<-0.01
  k<-0.5*qchisq(1-alfa,(alfabeto-1))
  bic<-1:2
  c<-1:2
  j<-k+20
  q<-k+0.00001
  i<-1
  while(q <= j)
    {
    c[i]<-q
```

```
  vc<-vlmc(texto,cutoff=c[i],threshold.gen = 5)
  card<-as.numeric(vc$size[2])
  bic[i]<- as.numeric(-2*logLik(vc)+(alfabeto-1)*log(n)*card)
  i<-i+1
  q<-q + 0.01
  }
posicao<-which.min(bic)
ajuste<-vlmc(texto,cutoff=c[posicao],threshold.gen = 5)
draw(ajuste)
ajuste_vetor<-ajuste$vlmc.vec
seq<-alfabeto + 1
vetor_auxiliar<-1:1
tamanho_ajuste_vetor<-length(ajuste_vetor)
H<-seq+2
J<- 1
while(H <= tamanho_ajuste_vetor)
  {
   M<-0
   if(ajuste_vetor[H]==-1)
     {
     vetor_auxiliar[J]<-ajuste_vetor[H]
     J<-J+1
     M<-1
     }
  if(ajuste_vetor[H] != -1)
     {
      vetor_auxiliar[J] <- ajuste_vetor[H]
      J<-J+1
     }
  if(M==0){H<-H+seq} else {H<-H+1}
  }
```

```
rm(ajuste_vetor,seq,tamanho_ajuste_vetor,H,J,M)
if(contador==1)
 {
 lista[[conj_arvores]]<-as.vector(vetor_auxiliar)
 pasta_arvore<-paste(conj,indice[conj_arvores],sep ="")
 local_pasta_arvore[conj_arvores]<-as.character(pasta_arvore)
 conj_arvores<-conj_arvores+1
 write(dir.create(pasta_arvore))
 caminho_pasta_arvore<-paste(local,pasta_arvore,sep ="")
 setwd(caminho_pasta_arvore)
 denden1sp<- as.dendrogram(ajuste)
 plot(denden1sp,type="tr",nodePar=list(pch=c(1,16)),
 center=TRUE,main=format(ajuste$call))
 savePlot(filename=u,type=c("ps"))
 setwd(local)
 }
if(contador>=2)
 {
 indicador<-0
 N<-1
 while(N <= as.numeric(length(lista)))
   {
   aux<-as.vector(lista[[N]])
   if(length(aux)==length(vetor_auxiliar))
    {
    soma<-0
    P<-1
    for(P in 1:length(aux))
     {
     soma<-soma + abs(aux[P]-vetor_auxiliar[P])
     }
```

```
if(soma==0)
 {
 lugar<-as.numeric(N)
 indicador<-indicador+1
 aux2<-local_pasta_arvore[[lugar]]
 diretorios<-dir()
 numero_diretorios<-as.numeric(length(diretorios))
 T<-1
 existe<-0
 for(T in 1:numero_diretorios)
   {
   if(diretorios[T]==aux2){existe<-1}
   }
 if(existe==1)
   {
   aux3<-paste(local,aux2,sep ="")
   setwd(aux3)
   denden1sp<- as.dendrogram(ajuste)
   plot(denden1sp,type="tr",nodePar=list(pch=c(1,16)),
   center=TRUE,main=format(ajuste$call))
   savePlot(filename=u,type=c("ps"))
   }
 if(existe ==0)
   {
   write(dir.create(aux2))
   aux3<-paste(local,aux2,sep ="")
   setwd(aux3)
   denden1sp<- as.dendrogram(ajuste)
   plot(denden1sp,type="tr",nodePar=list(pch=c(1,16)),
   center=TRUE,main=format(ajuste$call))
   savePlot(filename=u,type=c("ps"))
```

```
        }
      setwd(local)
      N<-as.numeric(length(lista))
      rm(aux2,aux3,diretorios,numero_diretorios,existe,T)
      }
    rm(soma,P)
    }
  N<-N+1
}
rm(aux,N)
if(indicador==0)
 {
 lista[[conj_arvores]]<-as.vector(vetor_auxiliar)
 pasta_arvore<-paste(conj,indice[conj_arvores],sep ="")
 local_pasta_arvore[conj_arvores]<-as.character(pasta_arvore)
 conj_arvores<-conj_arvores+1
 write(dir.create(pasta_arvore))
 caminho_pasta_arvore<-paste(local,pasta_arvore,sep ="")
 setwd(caminho_pasta_arvore)
 denden1sp<- as.dendrogram(ajuste)
 plot(denden1sp,type="tr",nodePar=list(pch=c(1,16)),
 center=TRUE,main=format(ajuste$call))
 savePlot(filename= u,type=c("ps"))
 setwd(local)
 }
}
contador<-contador+1
rm(arquivo,texto,n,alfa,k,c,j,q,i,vc,bic,posicao,
ajuste,denden1sp,card,vetor_auxiliar)
} ###FOR DO u
P<-1
```

```
G<-1
numero<-1:1
for(P in 1: (conj_arvores-1))
   {
   pasta_arvore<-paste(conj,indice[P],sep ="")
   aux2<-as.character(pasta_arvore)
   diretorios<-dir()
   numero_diretorios<-as.numeric(length(diretorios))
   T<-1
   existe<-0
   for(T in 1:numero_diretorios)
    {
    if(diretorios[T]==aux2){existe<-1}
    }
   if(existe==1)
    {
    caminho_pasta_arvore<-paste(local,pasta_arvore,sep ="")
    setwd(caminho_pasta_arvore)
    arvores<-list.files(path = ".", pattern = "ps",
    all.files = FALSE,full.names = FALSE, recursive = FALSE)
    numero[G]<- paste(pasta_arvore,"=",length(arvores),sep ="")
    G<-G+1
    setwd(local)
    }
  }
write(numero, file="Quantidades")
rm(P,numero,T,existe,diretorios,numero_diretorios,aux2,G)
setwd(caminho_jornal[A])
rm(u,alfabeto,textos)
}###FOR DO B
rm(B)
```

```
}####FOR DO A
```

6.4 Apêndice D

As Tabelas 6.4.1 e 6.4.2 apresentam respectivamente as caracteríticas dos conjutos de dados de PB e PE.

	Ano	Mês	Dia	Número de Sílabas		Ano	Mês	Dia	Número de Sílabas
1	1994	1	19	1650	21	1995	2	5	3202
2	1994	1	20	1805	22	1995	2	14	2055
3	1994	1	26	1997	23	1995	3	12	3192
4	1994	2	16	2133	24	1995	3	16	1925
5	1994	2	24	1942	25	1995	3	20	3133
6	1994	2	25	2252	26	1995	4	22	2025
7	1994	3	12	1319	27	1995	6	9	2438
8	1994	3	20	2640	28	1995	6	25	2518
9	1994	3	21	2473	29	1995	7	11	1317
10	1994	3	27	2601	30	1995	7	15	2353
11	1994	7	28	1849	31	1995	8	3	2874
12	1994	8	31	2313	32	1995	8	13	2237
13	1994	9	4	2162	33	1995	8	18	2643
14	1994	10	9	2301	34	1995	9	23	2017
15	1994	11	8	2500	35	1995	10	29	3025
16	1994	11	16	2276	36	1995	10	31	3631
17	1994	11	18	4499	37	1995	11	3	2464
18	1994	12	16	2152	38	1995	11	29	2131
19	1994	12	20	2372	39	1995	12	2	1959
20	1994	12	22	1789	40	1995	12	11	2489

Tabela 6.4.1: Dados de PB

	Ano	Mês	Dia	Número de Sílabas		Ano	Mês	Dia	Número de Sílabas
1	1994	1	4	2977	21	1995	1	2	3038
2	1994	2	18	2351	22	1995	1	11	2659
3	1994	2	21	4045	23	1995	1	14	3704
4	1994	3	16	2338	24	1995	2	13	2468
5	1994	3	17	2391	25	1995	2	26	2061
6	1994	4	9	2264	26	1995	3	3	2875
7	1994	4	24	2945	27	1995	3	15	2686
8	1994	5	27	2592	28	1995	3	27	2017
9	1994	5	30	3552	29	1995	4	27	2655
10	1994	7	9	2377	30	1995	5	12	2236
11	1994	7	23	1982	31	1995	5	24	2743
12	1994	8	9	3049	32	1995	6	25	3471
13	1994	8	27	2318	33	1995	7	16	2329
14	1994	9	8	1722	34	1995	7	24	2955
15	1994	9	10	2590	35	1995	9	17	2307
16	1994	9	17	2956	36	1995	9	26	2067
17	1994	11	12	2244	37	1995	9	28	2879
18	1994	11	15	2554	38	1995	10	4	3026
19	1994	11	28	2581	39	1995	12	1	2563
20	1994	12	17	2649	40	1995	12	22	1992

Tabela 6.4.2: Dados de PE