Melissa Lorena Araújo Pinho

Modelos Exponenciais para Grafos Aleatórios Valorados

Belo Horizonte

2018

Melissa Lorena Araújo Pinho

Modelos Exponenciais para Grafos Aleatórios Valorados

Dissertação apresentada ao Departamento de Estatística do Instituto de Ciências Exatas da Universidade Federal de Minas Gerais, para a obtenção do Título de Mestre em Estatística.

Orientadora: Prof[a]. Dr[a]. Lourdes Coral Contreras Montenegro
Co-Orientador: Prof. Dr. Adrian Pablo Hinojosa Luna

Belo Horizonte
2018

Agradecimentos

Agradeço primeiramente ao meu companheiro Paulo César, pelo seu amor, compreensão, respeito e tolerância.

À minha orientadora, Professora Doutora Lourdes C. C. Montenegro e ao co-orientador Professor Doutor Adrian P. H. Luna, pelo emprenho com que sempre me orientaram nesse trabalho. Muito obrigada pelas correções necessárias.

Agradeço também à minha família e amigos, pelo apoio e carinho. Em especial, agradeço aos meus pais, Nilceu e Consolação, pelo amor incondicional.

Finalmente, agradeço aos colegas da equipe DataViva, que tanto colaboraram para a realização deste trabalho. Em particular, ao Elton Freitas pela oportunidade de trabalharmos juntos e por sempre me incentivar. À FAPEMIG, pela disponibilização dos dados utilizados.

Resumo

Modelos Exponenciais para Grafos Aleatórios (ERGM) são modelos estatísticos para estrutura de redes que nos permitem fazer inferência sobre o processo gerador de tais estruturas. São baseados em três principais estatísticas: arestas, k-estrelas e triângulos.

Hunter e Handcock (2006) apresentam um método para estimação dos parâmetros do modelo ERGM para grafos simples através de simulações MCMC, e também a estimação da matriz de covariância dos parâmetros estimados. O objetivo dessa dissertação é estender esse método para os grafos aleatórios valorados, cujos valores das arestas não estão restritos a zero ou um. Nós estendemos o algoritmo proposto por Hunter e Handcock (2006) para o Modelo Exponencial para Grafos Aleatórios Valorados (ERGM-V), proposto por Krivitsky (2012), e o implementamos para o modelo onde os valores das arestas possuem distribuição Poisson. Implementamos também o algoritmo proposto por Krivitsky (2012) para simulação de grafos aleatórios valorados.

Os resultados do estudo de simulação são satisfatórios. Para o modelo uniparamétrico, com arestas independentes, todas as simulações convergiram. Ao inserir uma medida de correlação entre as observações, a convergência depende imensamente do vetor de parâmetros inicial fixado para as simulações. Contudo, em todos os casos em que houve convergência do algoritmo, tanto uniparamétrico como biparamétrico, observamos que este foi eficiente para estimar os parâmetros do modelo.

Palavras-chave: ERGM, ERGM-V, Espaço de Produtos, Grafos Aleatórios Exponenciais, Grafos Valorados, MCMC.

Abstract

Exponential Random Graph Models (ERGM) are statistical models for network structure, which allows us to make inferences about the generating process of such structures. They are based on three main statistics: edges, k-stars and triangles.

Hunter and Handcock (2006) present a method for estimation of the parameters of the ERGM model for simple graphs through MCMC simulations, as well as the covariance matrix of the estimated parameters. The main objective of this work is to extend this method to valued random graphs, whose edge values are not constrained to zero or one. We extend the algorithm proposed by Hunter and Handcock (2006) to Exponential Random Graph Model for valued networks (ERGM-V) proposed by Krivitsky (2012) and implement it to the model where the values of the edges are Poisson. We also implemented the algorithm proposed by Krivitsky (2012) for simulation of valued random graphs.

The results for simulation studies are satisfactory. For the uniparametric model, with independent edges, all of the simulations converged. By inserting a correlation measure between the observations, the convergence depends greatly on the initial parameter vector set for the simulations. However, in all of the cases where there was convergence of the algorithm, both uniparametric and biparametric, we observed that it was efficient to estimate the parameters of the model.

Keywords: ERGM, ERGM-V, Exponential Random Graphs, MCMC, Product Space, Valued Graphs.

Lista de Tabelas

Lista de Figuras

Conteúdo

Capítulo 1

Introdução

Atualmente estamos vivendo em um mundo conectado, onde as diferentes características dos objetos que percebemos em nosso entorno aparecem como relações entre estes objetos, e para o estudo destes objetos são utilizados grafos ou redes. Podemos dizer que uma rede, ou grafo, é uma coleção de elementos e suas inter-relações. Os modelos de grafos têm sido aplicados nas mais diversas áreas do conhecimento. Por exemplo, na medicina, Ayla et al. (2006) utilizam grafos para identificar genes associados ao câncer de próstata. Na epidemiologia Gopal (2007) utiliza grafos para relacionar blogs com a temática AIDS. Ainda, podem ser utilizados no estudo das redes sociais, como o Facebook ou Twitter, e na Comunicação são estudados as redes de computadores e celulares.

O estudo de modelos de grafos aleatórios pode ser considerado recente na estatística, uma vez que a função de verossimilhança, que é a função utilizada para inferir sobre os parâmetros do modelo estatístico, dos mais simples modelos para grafos aleatórios não são deriváveis analiticamente ou são modelos muito complexos cujo comportamento não é totalmente compreendido.

Um dos fatores que mais contribuíram para o desenvolvimento dos modelos estatísticos de estimação ou inferência para redes aleatórias é o recente uso dos métodos computacionais intensivos, como o MCMC (Cadeias de Markov Monte Carlo), Método de amostragem de Gibbs e o Metropolis Hastings, que são métodos de simulação de amostras aleatórias para uma distribuição de probabilidades. Ainda, a recente facilidade de obter dados em diferentes tipos de redes gera a necessidade de se obter adaptações dos modelos estatísticos para tais redes.

O modelo exponencial para grafos aleatórios (ERGM) é um dos principais modelos paramétricos utilizado para fazer inferência sobre grafos aleatórios. Uma revisão geral deste modelo pode ser encontrada em Lusher et al. (2012). O estudo dos modelos exponenciais com um enfoque mais matemático pode ser visto em Chatterjee e Diaconis (2013). O estimador de máxima verossimilhança foi proposto por Besag (1974), para o modelo de grafo que possui interações com os vizinhos próximos, chamado modelo "autologístico". Robins et al. (2007) desenvolveram uma aproximação do estimador de máxima verossimilhança, o método ficou conhecido como estimadores de Pseudo-máxima verosimilhança. O estimador de máxima verosimilhança para este modelo foi desenvolvido posteriormente em uma série de trabalhos, dos quais destacamos o trabalho de Hunter e Handcock (2006), onde é desenvolvido um algoritmo para estimar a variância do estimador de máxima verossimilhança proposto.

Este trabalho visa estudar uma modificação do modelo de grafo aleatório, chamado grafo aleatório exponencial valorado ('Valued Exponential Random Graph Model', ERGM-V), que foi introduzido por Krivitsky (2012)

para incluir valores aleatórios inteiros positivos nas arestas. Um estudo mais matemático e detalhado deste modelo pode ser encontrado em Yin et al. (2016). O modelo valorado nos permite considerar modelos longitudinais para grafos valorados. Para este modelo de grafo valorado, iremos estender o método de estimação de máxima verosimilhança (EMV) proposto por Hunter e Handcock (2006), que usa a aproximação Monte-Carlo para a função de Verossimilhança, para o modelo apresentado por Krivitsky (2012). Ainda, implementamos no software R Core Team (2017) o algoritmo para a estimação da variância do EMV. Como mencionado em Krivitsky (2012) e na implementação no R deste modelo no Capítulo 4 desta dissertação, o algoritmo apresenta sérios problemas de convergência.

Como aplicação dos modelos exponenciais para grafos valorados, estudaremos nesta dissertação um modelo de grafo chamado 'Espaço de Produtos' (Product Space). Este modelo econômico foi proposto por Hidalgo et al. (2007) como explicação do fenômeno da complexidade associada aos produtos que são exportados por um país. O Espaço de Produtos é uma rede cujos vértices são produtos e onde dois produtos são conectados se possuem alta probabilidade de serem exportados conjuntamente. Neste trabalho utilizaremos o espaço de produtos para os anos de 2015, 2010, 2005, 2000, 1995, 1990, 1985 e 1980, como observações do Espaço de Produtos em diferentes tempos e representamos estes grafos em um único grafo valorado, composto pela soma dos valores do Espaço de Produtos de cada ano. Para obter o Espaço de Produtos, utilizamos o conjunto de dados de comércio internacional fornecido pelo UN Comtrade (2017) e disponibilizado para este trabalho pela Fundação de Amparo à Pesquisa do Estado de Minas Gerais (FAPEMIG).

Esta dissertação inclui cinco capítulos. No Capítulo 2 apresentamos as principais definições de grafos. No Capítulo 3 introduzimos os grafos aleatórios e o modelo ERGM, e descrevemos a estimação dos parâmetros por máxima verosimilhança para o modelo em estudo. No Capítulo 4, apresentamos o modelo exponencial valorado para grafos, ERGM-V e estendemos a estimação proposta no Capítulo 3 para esse modelo. No Capítulo 5 apresentamos como aplicação o espaço de produtos valorado.

Capítulo 2

Grafos

A teoria dos grafos tem origem a partir de um problema histórico, que ficou conhecido como *as sete pontes de Königsberg*. Cortada pelo rio Prególia, a cidade de Königsberg, no território da Prússia, possuía duas grandes ilhas que, juntas, formavam um complexo que continha sete pontes. Por muito tempo se discutia a possibilidade de atravessar todas as pontes sem repetir nenhuma delas. Em 1736, o matemático Leonard Euler provou que não existe um caminho que permita tal travessia, representando os caminhos como arestas e suas interseções como vértices (Barabási, 2016).

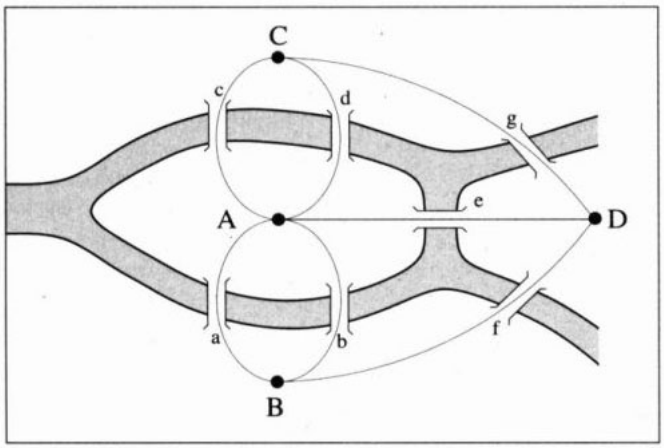

Figura 2.1: As sete pontes de Königsberg.

Em muitos problemas práticos, como o exemplo citado, um grafo é a melhor maneira de representar a relação entre os objetos de um determinado conjunto, onde alguns pares de objetos, chamados vértices, estão conectados por arestas.

2.1 Definição

Definição 2.1.1. *Feofiloff et al. (2011). Considere o conjunto de vértices* $V = 1, \ldots, n$*, finito com* n *elementos. Denotamos* $V^{(2)}$ *o conjunto de pares não ordenados de elementos de* V*. Os elementos de* $V^{(2)}$ *são identificados com os subconjuntos de* V *que possuem cardinalidade* 2*, e cada elemento de* $V^{(2)}$ *terá a forma* $\{v, w\}$*, sendo* v *e* w *dois elementos distintos de* V*. Um grafo é um par* (V, E) *em que* E *é um subconjunto de* $V^{(2)}$*. Os elementos de* V *são chamados de vértices e os elementos de* E *são chamados arestas.*

Considere, por exemplo, o conjunto de vértices $V = \{1, \ldots, 7\}$ e as arestas $E = \{(1,4), (1,5), (2,3), (2,7), (3,6), (4,7), (5,6), (5,3)\}$, então, o grafo $G = (V, E)$ pode ser representado pela Figura 2.2.

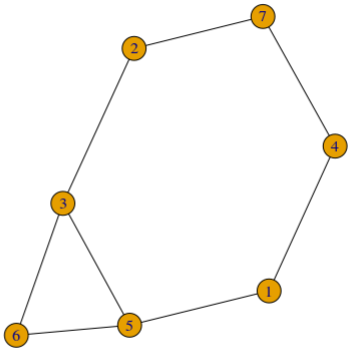

Figura 2.2: Exemplo de grafo com $n = 7$ vértices.

Kolaczyk e Csárdi (2014) define o tamanho de um grafo como o número de arestas e a sua ordem como o número de vértices. Assim, no exemplo anterior, o tamanho do grafo é $|E| = 8$ e a ordem é $|V| = 7$.

2.2 Características de Grafos

Duas importantes características de um grafo são:

- Laços - Um laço é uma aresta que conecta um vértice a ele mesmo. Também são chamados de *loops*.
- Multigrafo - Um multigrafo pode apresentar múltiplas arestas entre um par de vértices.

Define-se um grafo simples como aquele grafo que não possui laços ou arestas múltiplas. Segundo Lusher et al. (2012), muitos trabalhos têm sido desenvolvidos para este tipo particular de grafos, e mais especificamente para

aqueles não dirigidos. A Figura 2.2 representa um grafo simples não dirigido.

Uma importante estatística de um grafo é a distribuição dos graus de seus vértices. Kolaczyk e Csárdi (2014) define o grau $d(i)$ de um vértice $i \in V$ como o número de outros vértices conectados ao vértice i por meio de uma aresta que pertence ao grafo:

$$d(i) = \sum_{j:(i,j)\in E(G)} 1,$$

onde $E(G)$ é o conjunto das arestas de G. Assim, no grafo da Figura 2.2 temos que $d(1) = 2$, $d(2) = 2$, etc. Dois vértices são ditos adjacentes se o grafo contém uma aresta que conecta ambos. Utilizando este conceito, pode-se representar um grafo através de uma **matriz de adjacência** $\boldsymbol{A} = (a_{i,j})$, na qual as linhas e as colunas são indexadas por vértices de um grafo, e os coeficientes $a_{i,j}$ assumem o valor 1 se os vértices i e j são adjacentes ou 0 caso contrário. A matriz de adjacência $\boldsymbol{A}$ do grafo representado na Figura 2.2 seria

$$\boldsymbol{A} = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 \end{pmatrix}.$$

Observe que, quando o grafo G é não-dirigido, os elementos $a_{i,j}$ e $a_{j,i}$ são

idênticos e portanto a matriz de adjacência $\boldsymbol{A}$ é simétrica.

Capítulo 3

Grafos Aleatórios

O primeiro modelo para grafos aleatórios foi introduzido por Erdös e Rényi (1959). Neste modelo, cada aresta era incluída em um grafo simples de maneira independente e com igual probabilidade. Neste capítulo serão apresentados os principais modelos estatísticos para grafos aleatórios simples.

Segundo Robins (2013), o Modelo Exponencial para Grafos Aleatórios (ERGM) é o modelo paramétrico mais usado nos estudos de redes sociais. É um modelo flexível e que nos permite introduzir observações das características estruturais dos grafos em sua distribuição de probabilidade. Em outras palavras, a probabilidade de um grafo depende de características tais como o número de arestas, triângulos e estrelas que estão presentes no grafo. Lusher et al. (2012) apresentam uma descrição detalhada do desenvolvimento destes modelos paramétricos.

3.1 Definição

Seja $\mathcal{G}_n$ o conjunto de todos os grafos simples com n vértices, chamado espaço amostral e seja Θ é um conjunto contido em $\mathbb{R}^k$, chamado espaço

paramétrico. Um modelo para grafos aleatórios está definido pela coleção

$$\{\mathcal{G}_n, \mathbb{P}_{\boldsymbol{\theta}}(G)\}.$$

Para cada $\boldsymbol{\theta} \in \Theta$ a distribuição de probabilidade em $\mathcal{G}_n$ é dada por

$$\mathbb{P}_{\boldsymbol{\theta}} : \mathcal{G}_n \to [0, 1].$$

A principal dificuldade em trabalhar com modelos aleatórios para grafos consiste em definir o valor da probabilidade $\mathbb{P}$ para cada grafo $G = (V, E)$ do espaço amostral $\mathcal{G}_n$, pois embora o número de grafos deste espaço seja finito ele é muito grande quando o número de vértices n é grande, pois $|\mathcal{G}_n| = 2^{\binom{n}{2}}$ (Kolaczyk, 2009).

3.2 Alguns Modelos de Grafos Aleatórios Simples

A seguir apresentaremos três modelos de grafos aleatórios simples: Uniforme, Erdös-Rényi e Exponencial, sendo o modelo exponencial objeto de estudo principal desse trabalho.

3.2.1 Modelo Uniforme $G(n, m)$

Neste modelo todos os grafos $G \in \mathcal{G}_n$ possuem igual probabilidade, definida como

$$\mathbb{P}(G) = \frac{1}{|\mathcal{G}_n|}.$$

Em geral, suponha que o número de vértices é fixo, $|V| = n$ bem como o número de arestas, $|E| = m$. Definimos o Modelo Uniforme com m arestas, e

$G(n, m)$, como a distribuição de probabilidade que atribui igual probabilidade aos grafos pertencentes a $\mathcal{G}_{n,m} = \{G = (E, V) : |V| = n, |E| = m\}$, isto é

$$\mathbb{P}(G) = \frac{1}{\binom{N}{m}}, \quad G \in \mathcal{G}_{n,m},$$

com $N = \binom{n}{2}$ o número de arestas possíveis em um grafo com n vértices.

3.2.2 Modelo Erdös-Rényi

O modelo de Erdös-Rényi foi introduzido em 1959 por Paul Erdös. A construção do modelo consiste em atribuir uma probabilidade p a todas as arestas, em que cada aresta pertença ao grafo, independentemente das outras arestas: $\mathbb{P}((u, v) \in E(G)) = p$. A probabilidade para $G = (E, V) \in \mathcal{G}_n$ no modelo de Erdös-Rényi, $G(n, p)$, é dada por

$$\mathbb{P}_\theta(G) = p^{|E|}(1 - p)^{\binom{n}{2} - |E|}.$$

Nesse caso temos que $\theta = \log\left(\frac{p}{1-p}\right)$ Note que a probabilidade de cada grafo G com m arestas é $\mathbb{P}(G) = p^m(1 - p)^{\binom{n}{2} - m}$. Segundo Robins (2013), a hipótese de independência entre as arestas não é esperada nos grafos aleatórios. No entanto, o modelo independente é amplamente utilizado como modelo base para comparações.

3.2.3 Modelo Exponencial Aleatório ERGM

Considere um conjunto de estatísticas $Z_1, \ldots, Z_k$ de um grafo, que podem ser o número de arestas, de triângulos, estrelas, etc. e suponha que associamos parâmetros: $\theta_1, \ldots, \theta_k$ a cada uma delas. Assim, definimos a distribuição

de probabilidades deste modelo como

$$\mathbb{P}_\theta(G) = \frac{e^{\sum_{i=1}^{k} \theta_i Z_i(G)}}{\kappa(\boldsymbol{\theta})}, \quad G \in \mathcal{G}_n,$$

onde $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_k) \in \Theta$ é o vetor de parâmetros e $\kappa(\boldsymbol{\theta})$ é uma constante normalizadora chamada função de partição,

$$\kappa(\boldsymbol{\theta}) = \sum_{G \in \mathcal{G}_n} e^{\sum_{i=1}^{k} \theta_i Z_i(G)}.$$

O modelo de Erdös-Rényi é um caso particular do modelo ERGM, correspondente ao caso de usar somente o número de vértices dos grafos no modelo

$$\mathbb{P}_\theta(G) = \frac{e^{\theta |E(G)|}}{\kappa(\theta)}, \quad G \in \mathcal{G}_n.$$

Observe que $Z(G) = |E(G)|$. Neste caso é fácil mostrar o valor da constante normalizadora $\kappa(\theta) = (1 + e^\theta)^{\binom{n}{2}}$, em que a probabilidade de ter uma aresta é $p = \frac{e^\theta}{1+e^\theta}$. Em geral a grande dificuldade em estimar os parâmetros do modelo ERGM consiste em encontrar o valor de $\kappa(\boldsymbol{\theta})$ e, por essa razão, são utilizados métodos numéricos para sua aproximação. Nas Seções 3.3 e 3.4 será apresentado o método de estimação para o modelo ERGM proposto por Hunter e Handcock (2006).

3.3 Estimadores de Máxima Verosimilhança para o modelo ERGM

Para o modelo ERGM temos uma única observação de um grafo G, que será denotado por $\boldsymbol{g_o}$ com função de verossimilhança dada por

$$\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{g_o}) = \frac{e^{\boldsymbol{\theta Z}(\boldsymbol{g_o})}}{\kappa(\boldsymbol{\theta})}. \tag{3.3.1}$$

O vetor $\boldsymbol{Z(g_o)} = (Z_1(\boldsymbol{g_o}), \cdots, Z_k(\boldsymbol{g_o}))$ contém estatísticas do grafo $\boldsymbol{g_o}$, ou seja, são funções de $\boldsymbol{g_o}$. Conforme descrito em Robins (2013), os modelos ERGM são completamente caracterizados pelas estatísticas suficientes deste modelo (Casella e Berger, 2002).

O Estimador de Máxima Verossimilhança (EMV) do vetor de parâmetros $\boldsymbol{\theta}$ é definido por

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} \ell(\boldsymbol{\theta}, \boldsymbol{g_o}),$$

onde $\ell(\boldsymbol{\theta}, \boldsymbol{g_o}) = \log \mathbb{P}_{\theta}(\boldsymbol{g_o})$.

Por exemplo, se considerarmos $V = \{1, 2, 3, 4, 5\}$, $Z = |E(V)|$ e o grafo $\boldsymbol{g_o}$ apresentado na Figura 3.1, temos que $Z(\boldsymbol{g_o}) = 5$.

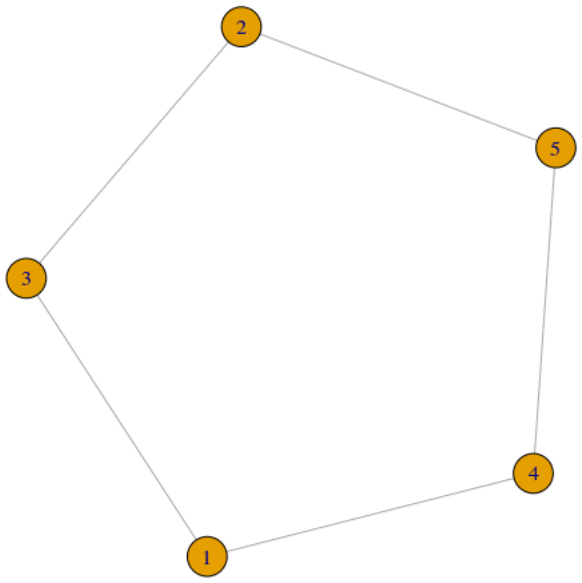

Figura 3.1: Exemplo de ciclo com $n = 5$ vértices.

A função de verossimilhança para o grafo da Figura 3.1 fica dada por

$$\mathbb{P}_\theta(\boldsymbol{g_o}) = \frac{e^{5\theta}}{(1+e^\theta)^{\binom{5}{2}}}.$$

A função de log-verossimilhança $\ell(\theta, \boldsymbol{g_o})$

$$\begin{aligned}\ell(\theta, \boldsymbol{g_o}) &= \log\left[\frac{e^{5\theta}}{(1+e^\theta)^{10}}\right] \\ &= 5\theta - 10\log(1+e^\theta).\end{aligned}$$

A primeira derivada da função de log-verossimilhança fica dada por

$$\frac{\partial \ell(\theta, \boldsymbol{g_o})}{\partial \theta} = 5 - \frac{10e^{\theta}}{\log(1+e^{\theta})}.$$

No ponto de máximo da função de log-verossimilhança, temos

$$\begin{aligned}
&\frac{\partial \ell(\hat{\theta}, \boldsymbol{g_o})}{\partial \hat{\theta}} = 0 \\
&\Rightarrow 5 - \frac{10e^{\hat{\theta}}}{\log(1+e^{\hat{\theta})}} = 0 \\
&\Rightarrow \hat{\theta} = 0
\end{aligned}$$

A maximização da função $\ell(\boldsymbol{\theta}, \boldsymbol{g_o})$ não é trivial devido à dificuldade de aproximar a constante normalizadora $\kappa(\boldsymbol{\theta})$. Nesta seção será apresentado um método de estimação via aproximação de Monte Carlo proposto por Hunter e Handcock (2006).

Seja $\boldsymbol{\theta_0} \in \boldsymbol{\Theta}$ um vetor de parâmetros conhecido, e suponha que podemos simular grafos aleatórios da distribuição que possui parâmetro $\boldsymbol{\theta_0}$. Denotemos $r(\boldsymbol{\theta}, \boldsymbol{\theta_0}) = \ell(\boldsymbol{\theta}, \boldsymbol{g_o}) - \ell(\boldsymbol{\theta_0}, \boldsymbol{g_o})$, em que $\ell(\boldsymbol{\theta_0}, \boldsymbol{g_o})$ é a função de log-verossimilhança avaliada em $\boldsymbol{\theta_0}$.

Observamos que

$$\begin{aligned}
\ell(\boldsymbol{\theta}, \boldsymbol{g_o}) - \ell(\boldsymbol{\theta_0}, \boldsymbol{g_o}) &= \log\left(\frac{e^{\boldsymbol{\theta} \cdot \boldsymbol{Z}(\boldsymbol{g_o})}}{\kappa(\boldsymbol{\theta})}\right) - \log\left(\frac{e^{\boldsymbol{\theta_0} \cdot \boldsymbol{Z}(\boldsymbol{g_o})}}{\kappa(\boldsymbol{\theta_0})}\right) \\
&= (\boldsymbol{\theta} - \boldsymbol{\theta_0}) \cdot \boldsymbol{Z}(\boldsymbol{g_o}) - \log\left(\frac{\kappa(\boldsymbol{\theta})}{\kappa(\boldsymbol{\theta_0})}\right).
\end{aligned}$$

A razão $\frac{\kappa(\boldsymbol{\theta})}{\kappa(\boldsymbol{\theta_0})}$ pode ser reescrita como

$$\begin{aligned}\frac{\kappa(\boldsymbol{\theta})}{\kappa(\boldsymbol{\theta_0})} &= \frac{\sum_{\boldsymbol{G}\in\mathcal{G}_n} e^{\boldsymbol{\theta Z(G)}}}{\kappa(\boldsymbol{\theta_0})} \\ &= \sum_{\boldsymbol{G}\in\mathcal{G}_n} e^{\boldsymbol{(\theta-\theta_0)Z(G)}} \frac{e^{\boldsymbol{\theta_0\cdot Z(G)}}}{\kappa(\boldsymbol{\theta_0})} \\ &= \mathbb{E}_{\boldsymbol{\theta_0}}\left[e^{\boldsymbol{(\theta-\theta_0)\cdot Z(G)}}\right],\end{aligned}$$

onde $\boldsymbol{G}$ é um grafo aleatório com função de probabilidade $P_{\theta_0}(\boldsymbol{G})$. Considerando uma amostra $\boldsymbol{g_1}, \cdots, \boldsymbol{g_m}$ de grafos aleatórios com distribuição $P_{\boldsymbol{\theta_0}(\boldsymbol{G})}$ e utilizando a utilizando a Lei dos Grandes Números (James, 2015), podemos aproximar $\log\left(\frac{\kappa(\boldsymbol{\theta})}{\kappa(\boldsymbol{\theta_0})}\right)$ por

$$\log\left(\frac{\kappa(\boldsymbol{\theta})}{\kappa(\boldsymbol{\theta_0})}\right) \approx \log\left(\frac{1}{m}\sum_{i=1}^{m} e^{\boldsymbol{(\theta-\theta_0)\cdot Z(g_i)}}\right).$$

Assim, utilizamos esta aproximação para definir

$$\hat{r}_m(\boldsymbol{\theta, \theta_0}) = \boldsymbol{(\theta-\theta_0)\cdot Z(g_o)} - \log\left(\frac{1}{m}\sum_{i=1}^{m} e^{\boldsymbol{(\theta-\theta_0)\cdot Z(g_i)}}\right). \tag{3.3.2}$$

Portanto, o EMV aproximado para o modelo ERGM é dado por

$$\begin{aligned}\hat{\boldsymbol{\theta}}_m &= \arg\max_{\boldsymbol{\theta}\in\Theta} \hat{l}_m(\boldsymbol{\theta}, \boldsymbol{\theta_0}) \\ &= \arg\max_{\boldsymbol{\theta}\in\Theta} \hat{r}_m(\boldsymbol{\theta}, \boldsymbol{\theta_0}).\end{aligned}$$

A aproximação descrita na Equação 3.3.2 possibilita a estimação do vetor de parâmetros $\boldsymbol{\theta}$. No entanto, para a estimação da variância de $\hat{\boldsymbol{\theta}}_m$ utilizamos o método escore de Fisher, onde a estimativa de $\boldsymbol{\theta}$ na t-ésima interação é dada

por

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} + \left[I(\boldsymbol{\theta}^t)\right]^{-1} \nabla \ell(\boldsymbol{\theta}^{(\boldsymbol{t})}), \tag{3.3.3}$$

onde $I(\boldsymbol{\theta})$ representa a matriz de Informação de Fisher de $\boldsymbol{\theta}$ e $\nabla \ell(\boldsymbol{\theta}^{(\boldsymbol{t})})$ representa a primeira derivada da função de log-verossimilhança $\ell(\boldsymbol{\theta})$

$$\nabla \ell(\boldsymbol{\theta}^{(\boldsymbol{t})}) = \boldsymbol{Z(g_o)} - \mathbb{E}_{\boldsymbol{\theta}^{(t)}}\left[\boldsymbol{Z(g)}\right]. \tag{3.3.4}$$

Para estimar a variância do estimador $\hat{\boldsymbol{\theta}}_m$, descrevemos a seguir o algoritmo proposto por Hunter e Handcock (2006). Defina $\boldsymbol{\theta_0}$ um vetor de parâmetros conhecido. A escolha de $\boldsymbol{\theta_0}$ é muito importante pois implica diretamente em um bom ajuste e a convergência do algoritmo. Quanto menor a distância entre $\boldsymbol{\theta_0}$ e $\boldsymbol{\theta}$, melhor a convergência.

Para gerar a amostra de grafos aleatórios com a distribuição dada em 3.3.1, foi utilizado o pacote **ergm** do software R. Para cada simulação i e em cada passo t da estimação, calcula-se o vetor de estatísticas de interesse $\boldsymbol{Z_i}$, para $i = 1, \cdots, m$ e o seu peso $w_i^{(t)}$ é dado por

$$w_i^{(t)} = \frac{\exp\{[\boldsymbol{\theta^{(t)}} - \boldsymbol{\theta_0}]Z_i\}}{\sum_{j=1}^{m} \exp\{[\boldsymbol{\theta^{(t)}} - \boldsymbol{\theta_0}]Z_i\}}, i = 1, \ldots, m.$$

A matriz de Informação de Fisher pode ser aproximada por

$$\hat{I}(\boldsymbol{\theta})^{(t)} = \sum_{i=1}^{m} w_i^{(t)} \boldsymbol{Z_i Z_i^T} - \left(\sum_{i=1}^{m} w_i^{(t)} \boldsymbol{Z_i}\right)\left(\sum_{i=1}^{m} w_i^{(t)} \boldsymbol{Z_i}\right)^T.$$

Por outro lado, a expressão 3.3.4 pode ser aproximada por

$$\nabla \ell(\boldsymbol{\theta^{(t)}}) = \boldsymbol{Z(g_o)} - \sum_{\boldsymbol{i=1}}^{\boldsymbol{m}} \boldsymbol{w_i^{(t)} Z_i}. \tag{3.3.5}$$

Portanto, a expressão 3.3.3 pode ser aproximada por

$$\hat{\boldsymbol{\theta}}^{(t+1)} = \hat{\boldsymbol{\theta}}^{(t)} + \left[\hat{I}(\hat{\boldsymbol{\theta}}^{(\boldsymbol{t})})\right]^{-1} \left[\boldsymbol{Z(g_o)} - \sum_{\boldsymbol{i=1}}^{\boldsymbol{m}} \boldsymbol{w}_{\boldsymbol{i}}^{(\boldsymbol{t})} \boldsymbol{Z_i}\right]. \tag{3.3.6}$$

A convergência do algoritmo 3.3.3 pode ser verificada pela diferença absoluta entre as estimativas $\hat{\boldsymbol{\theta}}^{(\boldsymbol{t+1})}$ e $\hat{\boldsymbol{\theta}}^{(\boldsymbol{t})}$: $|\hat{\boldsymbol{\theta}}^{(\boldsymbol{t+1})} - \hat{\boldsymbol{\theta}}^{(\boldsymbol{t})}| < \epsilon$. Seja $\tilde{\boldsymbol{\theta}}$ o estimador obtido após convergência de 3.3.6. Deve-se recomeçar a estimação com um novo $\boldsymbol{\theta_0}$ se o seguinte critério de parada não for atingido

$$\sqrt{\hat{\mathrm{V}}_{MC}[\hat{r}_m]} \leq k\hat{\ell}(\tilde{\boldsymbol{\theta}}), \tag{3.3.7}$$

para algum k. Para aproximar a função de log-verossimilhança $\hat{\ell}(\tilde{\boldsymbol{\theta}})$ utilizada no critério de parada do algoritmo, utilizamos que

$$\begin{aligned}
\ell(\mathbf{0}) &= \log\left[\frac{e^{\mathbf{0}\cdot\boldsymbol{Z(g_o)}}}{\kappa(\mathbf{0})}\right] \\
&= \log\left[\frac{1}{\sum_{G\in\mathcal{G}_n} e^{\mathbf{0}\cdot\boldsymbol{Z(G)}}}\right] \\
&= \log\left[\frac{1}{\sum_{G\in\mathcal{G}_n} 1}\right] \\
&= \log\left[\frac{1}{|\mathcal{G}_n|}\right] \\
&= \log\left[2^{-\binom{n}{2}}\right] \\
&= -\binom{n}{2}\log(2)
\end{aligned}$$

e definimos

$$\begin{aligned}\hat{r}_m(\tilde{\boldsymbol{\theta}}, \boldsymbol{\theta_0}) - \hat{r}_m(\mathbf{0}, \boldsymbol{\theta_0}) &= \hat{\ell}(\tilde{\boldsymbol{\theta}}) - \hat{\ell}(\boldsymbol{\theta_0}) - \hat{\ell}(\mathbf{0}) + \hat{\ell}(\boldsymbol{\theta_0}) \\ &= \hat{\ell}(\tilde{\boldsymbol{\theta}}) - \hat{\ell}(\mathbf{0}) \\ &= \hat{\ell}(\tilde{\boldsymbol{\theta}}) + \binom{n}{2} \log 2,\end{aligned}$$

Portanto, temos que

$$\hat{\ell}(\tilde{\boldsymbol{\theta}}) = \hat{r}_m(\tilde{\boldsymbol{\theta}}, \boldsymbol{\theta_0}) - \hat{r}_m(\mathbf{0}, \boldsymbol{\theta_0}) - \binom{n}{2} \log 2.$$

Considere $U_i = \exp\{[\tilde{\boldsymbol{\theta}} - \boldsymbol{\theta_0}] \cdot Z_i\}$, para $i = 1, \ldots, m$, os pesos não padronizados das simulações Monte Carlo e $\bar{U} = \frac{1}{m}\sum_{i=1}^{m} U_i$. A variância da função $\hat{r}_m$ é dada por

$$\hat{\mathrm{V}}_{MC}[\hat{r}_m] = \frac{1}{m^2 \bar{U}^2} \sum_{k=-K}^{K} (m - |k|)\hat{\gamma}_k, \tag{3.3.8}$$

onde $\hat{\gamma}_k = Cov(U_j, U_{j+k})$ denota a função de autocovariância estimada da sequência $U_1, U_2, \ldots$ e K representa o número de auto-covariâncias consideradas para a estimação de $\hat{\mathrm{V}}_{MC}$.

3.4 Desvio-Padrão para o Estimador de Máxima Verossimilhança do Modelo ERGM

Para estimar a variância do estimador $\tilde{\boldsymbol{\theta}}$ definido na Seção 3.3, apresentaremos o método proposto por Hunter e Handcock (2006), que é baseado no artigo de Geyer (1994). Este artigo é teórico e estuda a aproximação

via Cadeias de Markov Monte Carlo (MCMC) da função de verossimilhança em distribuições exponenciais. Os resultados teóricos utilizados neste artigo encontram-se no Apêndice A.

Para calcular a matriz de covariância do estimador $\tilde{\boldsymbol{\theta}}$ devemos corrigir a matriz de Informação de Fisher aproximada. Para isso, considere $W_i = (\boldsymbol{Z(g_o)} - \boldsymbol{Z_i}) \exp\{[\tilde{\boldsymbol{\theta}} - \boldsymbol{\theta^0}]^t \boldsymbol{Z_i}\}$ o peso MCMC do grafo g_i simulado, $i = 1, \cdots, m$. Como em A.0.4, a matriz $\boldsymbol{A}$ é aproximada por

$$\tilde{\boldsymbol{A}} = \frac{1}{m^2} \left[\sum_{i=1}^{m} \exp\{[\boldsymbol{\theta_0} - \tilde{\boldsymbol{\theta}}]^t \boldsymbol{Z_i}\} \right]^2 \sum_{k=-K}^{K} \hat{\xi}_k,$$

onde $\tilde{\boldsymbol{\theta}}$ é o estimador proposto em 3.3 e $\hat{\xi}_k$ é a matriz de autocovariância de $W_1, \ldots, W_m$ de lag k, ou seja, $\hat{\xi}_k = \mathrm{Cov}(W_i, W_{i+k})$. Assim, a matriz de covariância MCMC para $\tilde{\boldsymbol{\theta}}$ fica dada por

$$\frac{1}{m} \left[\hat{I}(\tilde{\boldsymbol{\theta}}) \right]^{-1} \tilde{\boldsymbol{A}} \left[\hat{I}(\tilde{\boldsymbol{\theta}}) \right]^{-1}.$$

3.5 Estudo de Simulação

Apresentamos a seguir alguns resultados de simulação para avaliar o desempenho dos métodos de estimação descritos nas Seções 3.3 e 3.4. Todas as simulações foram implementadas no software R Core Team (2017) e o código encontra-se no Apêndice B desta dissertação.

Para avaliar o algoritmo desenvolvido na Seção 3.3, foram implementados dois modelos: o primeiro modelo é o mais simples, que é o modelo uniparamétrico, com função Z_1 igual ao número de arestas $|E|$ do grafo. Posteriormente, foi implementado o modelo bi-paramétrico, que considera duas características: número de arestas e número de triângulos. Para verificar a convergência

da sequência $\theta^{(t)}$, foi utilizado o seguinte critério: $\theta^{(t)} - \theta^{(t-1)} < 0,0001$, e a constante k da expressão 3.3.7 foi fixada em $k = 4$. O número de autocovariâncias K consideradas para o cálculo da variância da função $\hat{\mathrm{V}}_{MC}[\hat{r}_m]$ foi definido como $K = 10$. Foi realizada uma análise de sensibilidade para tais valores e foi observado que eles não interferem no tempo de convergência ou nos valores dos parâmetros estimados.

3.5.1 Modelo Uniparamétrico

Como apresentado na Seção 3.2.3, ao considerarmos o modelo uniparamétrico e a estatística $Z(\boldsymbol{g})$ como o número de arestas, o modelo ERGM corresponde ao modelo de Erdös-Rényi descrito na Subseção 3.2.2. Assim, para gerar um grafo $\boldsymbol{g}$ com $n = 15$ vértices, cada uma das $\binom{15}{2}$ arestas foi inserida aleatoriamente com probabilidade $p = 0,2$, de maneira independente. A Figura 3.2 apresenta o grafo simulado sob essas condições.

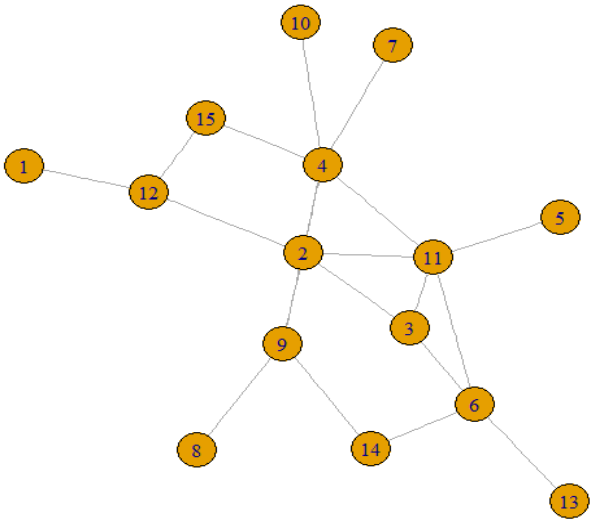

Figura 3.2: Grafo G

O modelo contém apenas um parâmetro θ, que pode ser calculado como definido na Subseção 3.2.3

$$p = \frac{e^{\theta}}{1+e^{\theta}} \Rightarrow \theta = \log\left(\frac{p}{1-p}\right) = \log\left(\frac{0,2}{0,8}\right) \approx -1,386.$$

Aplicando os algoritmos descritos na Seção 3.3, o valor de θ_0 estimado foi $-1,447$ e $\tilde{\theta} = -1,340$, com erro-padrão igual a $0,535$, para $m = 100$ simulações MCMC. Note que o algoritmo gerou uma boa aproximação para o parâmetro θ.

Para verificar se o algoritmo gera boas estimativas para os demais valores de θ foi realizada uma análise de sensibilidade. A Tabela 3.1 apresenta o resultado da análise de sensibilidade do algoritmo para diferentes valores

para o parâmetro θ. O tempo computacional para cada simulação foi aproximadamente $1,7$ segundos para os modelos com $m = 100$ simulações Monte Carlo e 9 segundos para $m = 1000$ simulações.

Tabela 3.1: Análise de sensibilidade para diferentes valores de θ, $n = 20$.

p	θ	$m = 100$ simulações MC			$m = 1000$ simulações MC		
		θ_0	$\tilde{\theta}$	$V(\tilde{\theta})$	θ_0	$\tilde{\theta}$	$V(\tilde{\theta})$
0,1	-2,197	-2,257	-2,250	0,430	-2,257	-2,254	0,020
0,2	-1,386	-1,322	-1,340	0,286	-1,322	-1,324	0,052
0,3	-0,847	-0,872	-0,887	0,968	-0,872	-0,877	0,087
0,4	-0,405	-0,427	-0,427	1,195	-0,427	-0,431	0,164
0,5	0.000	0,126	0,105	1,656	0,126	0,129	0,225
0,6	0,405	0,427	0,420	4,248	0,427	0,428	0,249
0,7	0,847	0,872	0,857	5,130	0,872	0,882	0,395
0,8	1,386	1,260	1,246	6,703	1,322	1,324	0,407
0,9	2,197	2,257	2,247	13,852	2,257	2,243	1,844

Na Tabela 3.1 n representa o número de vértices do grafo simulado, p é a probabilidade de existir uma aresta entre um par de vértices, θ é o valor verdadeiro do parâmetro, $\theta_0 = \log\left(\frac{d}{1-d}\right)$ é o valor inicial do algoritmo do qual as simulações são realizadas, d é a densidade observada no grafo simulado, $\tilde{\theta}$ é o parâmetro estimado pelo algoritmo e $V(\tilde{\theta})$ a sua respectiva variância. Podemos observar que os resultados $\tilde{\theta}$ são próximos ao valor verdadeiro θ, tanto para $m = 100$ quanto para $m = 1000$ simulações MCMC, para todos os valores de p testados. A variância foi estimada entre $0,286$ e $13,852$ para $m = 100$ simulações e entre $0,020$ e $1,844$ para $m = 1000$ simulações. Observe também que a variância é maior para $m = 100$ simulações do que para $m = 1000$, além disso, observamos que quando o valor de p aumenta o erro padrão também aumenta, embora não conhecemos uma explicação teórica para tal comportamento. Observamos também que, em alguns casos a estimativa θ_0 se aproxima mais do valor verdadeiro do que a estimativa $\tilde{\theta}$.

Ainda, foi realizada uma análise de sensibilidade do algoritmo para diferentes valores do número de vértices n do grafo. Na Tabela 3.2, fixamos o valor de p em $0,2$ e variamos o valor de n entre 10 e 90. Com aumento do número de vértices n, as variâncias diminuem, para $m = 1000$ simulações apresenta erro padrão menor do que para $m = 100$.

Tabela 3.2: Análise de Sensibilidade para diferentes valores de n, $p = 0,2$ e $\theta = -1,386$

n	θ_0	$m = 100$ simulações		$m = 1000$ simulações	
		$\tilde{\theta}$	EP	$\tilde{\theta}$	EP
10	-2.079	-1.997	0,038	-2,064	0,005
20	-1,260	-1,290	1,034	-1,266	0,063
30	-1,415	-1,427	1,190	-1,416	0,110
40	-1,555	-1,555	1,132	-1,557	0,116
50	-1,326	-1,336	2,326	-1,328	0,295
60	-1,358	-1,372	5,915	-1,362	0,438
70	-1,402	-1,399	2,579	-1,346	0,591
80	-1,374	-1,380	17,098	-1,374	0,792
90	-1,363	-1,360	6,975	-1,363	1,113

Variando o n entre 10 e 90 e fixando $p = 0,2$ e $\theta = -1,386$ observamos que, em geral foram realizadas boas estimações e o método consegue obter um valor estimado próximo ao parâmetro verdadeiro. Assim como na Tabela 3.1, observamos que os valores de $\tilde{\theta}$ são próximos de θ_0 e, maiores valores de n apresentam variâncias menores e para $m = 1000$ simulações apresentam menor variância que para $m = 100$ simulações. Para maiores valores de n, as estimativas estão mais próximas do θ verdadeiro do que para os menores valores de n.

A Tabela 3.5.1 apresenta os resultados da análise de sensibilidade, para a realização de 100 repetições em cada simulação. Observamos que θ_0 médio é muito próximo de $\tilde{\theta}$ médio. Esta análise nos mostra que o algoritmo possui uma forte dependência de θ_0.

Tabela 3.3: Análise de Sensibilidade para diferentes valores de p, $n = 12$, $m = 100$ simulações, 100 repetições.

p	θ	θ_0 Médio	$\tilde{\theta}$ Médio	$Var(\tilde{\theta})$ Média
0.1	-2.197	-2.265	-2.267	0.238
0.3	-0.847	-0.865	-0.866	0.474
0.5	0.000	0.003	0.004	0.708
0.7	0.847	0.856	0.857	1.105
0.9	2.197	2.268	2.268	2.294

3.5.2 Caso Multiparamétrico

Nesta subseção será considerado o caso multiparamétrico, ou seja, com um vetor de estatísticas $\boldsymbol{Z(g)} = (Z_1(\boldsymbol{g}); Z_2(\boldsymbol{g}))$ em que $Z_1(\boldsymbol{g})$ representa o número de arestas e $Z_2(\boldsymbol{g})$ o número de triângulos do grafo $\boldsymbol{g}$. Para o grafo da Figura 3.2 os valores estimados são $\tilde{\theta}_1 = -1,684$ e $\tilde{\theta}_2 = -0,005$. Considerando que este grafo foi gerado sob a distribuição exponencial com parâmetros $\theta_1 = -1,386$ e $\theta_2 = 0$, observamos que os valores estimados pelo algoritmo foram próximos aos valores verdadeiros dos parâmetros, no entanto a variância de θ_1 é muito grande: $1,0310^{-4}$ em relação à variância de θ_2: $3,0210^{-4}$, e a covariância estimada entre $\tilde{\theta}_1$ e $\tilde{\theta}_2$ foi 10^{-4}.

Posteriormente, foi realizada uma análise de sensibilidade para os parâmetros θ_1 e θ_2. A Tabela 3.4 apresenta os resultados dessa análise.

Nesse caso, foi utilizado $\boldsymbol{\theta_0} = \boldsymbol{\theta}$ e portanto, não conseguimos estimar a variância de $\tilde{\boldsymbol{\theta}}$ em quase todas as simulações.

Tabela 3.4: Análise de sensibilidade para o caso biparamétrico, com $n = 20$ e $m = 100$.

θ_1	θ_2	$\widetilde{\theta_1}$	V($\widetilde{\theta_1}$)	$\widetilde{\theta_2}$	$VP(\widetilde{\theta_1})$
-2.0	1	-1,999	0,000	1,001	0,000
-1,5	1	-1,499	0,000	1,001	0,000
-1.0	1	-0,999	0,000	1,001	0,000
-0,5	1	-0,499	0,000	1,001	0,000
0	-1	-0,723	0,000	-1,236	0,000
0	1	0,001	0,000	1,001	0,000
0,5	-1	0,116	0,000	-0,658	0,000
0,5	1	0,501	0,000	1,001	0,000
1.0	1	1,001	0,000	1,001	0,000
1,5	-1	1,539	0,057	-1,066	0,115
1,5	0,5	1,501	0,000	0,501	0,000
1,5	1	1,501	0,000	1,001	0,000
2.0	-0,5	0,562	0,000	-0,169	0,000
2.0	0,5	2,001	0,000	0,501	0,000
2.0	1	2,001	0,000	1,001	0,000
2,5	-1	2,439	0,000	-1,700	0,000
2,5	0,5	2,501	0,000	0,501	0,000
2,5	1	2,501	0,000	1,001	0,000
3.0	-0,5	2,272	0,000	-0,350	0,000
3.0	0,5	3,001	0,000	0,501	0,000
3.0	1	3,001	0,000	1,001	0,000
3,5	0,5	3,501	0,000	0,501	0,000
3,5	1	3,501	0,000	1,001	0,000
4.0	-0,5	3,377	0,000	-0,452	0,000
4.0	0,5	4,001	0,000	0,501	0,000
4.0	1	4,001	0,000	1,001	0,000

Capítulo 4

Modelo Exponencial para Grafos Aleatórios Valorados

Em várias situações, as relações de interesse entre os vértices de um grafo não são necessariamente iguais entre si. Nesses casos, não é adequado representá-las através de um grafo estritamente dicotômico. Tais relações podem aparecer de diferentes formas, como por exemplo, valores de contagens, ordenações ou até mesmo variáveis contínuas. Robins (2013) apresenta diversos modelos que comportam esse tipo de informação, e neste Capítulo apresentaremos o modelo para dados de contagem proposto por Krivitsky (2012). Tais modelos podem ser vistos como uma extensão do modelo ERGM apresentado no Capítulo 3 para o caso de grafos valorados. O objetivo neste Capítulo é implementar o método de estimação apresentado no Capítulo 3 para o modelo exponencial para grafos aleatórios valorados. A Figura 4.1 apresenta um exemplo de grafo valorado com 5 vértices.

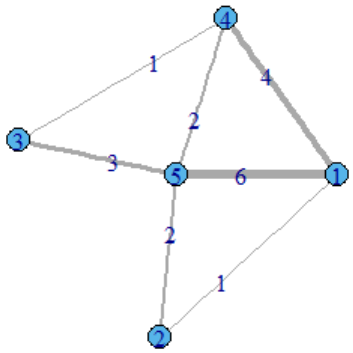

Figura 4.1: Representação gráfica para um grafo valorado.

4.1 Definição do Modelo

Sejam V e $V^{(2)}$ definidos no Capítulo 2, $n = |V|$ o número de vértices, $\mathbb{N}_0$ o conjunto dos números naturais incluindo o zero e $h(G)$ uma medida de referência. O espaço amostral é denotado por $\mathcal{G}_n \subset \mathbb{N}_0^{V^{(2)}}$ e a ele associamos a seguinte medida de probabilidade

$$\mathbb{P}_\theta(G) = h(G)\frac{e^{\boldsymbol{\theta}\cdot\boldsymbol{Z}(\boldsymbol{G})}}{\kappa(\boldsymbol{\theta})}, \tag{4.1.1}$$

onde $\kappa(\boldsymbol{\theta})$ é a constante normalizadora dada por

$$\kappa(\boldsymbol{\theta}) = \sum_{G\in\mathcal{G}_n} h(G)e^{\boldsymbol{\theta}\cdot\boldsymbol{Z}(\boldsymbol{G})},$$

onde $\boldsymbol{Z}(G)$ representa o vetor de estatísticas do grafo G. Por exemplo, se consideramos o caso $Z(G) = \sum_{(i,j)\in E(G)} G_{ij}$, onde G_{ij} representa o valor

associado à aresta (i,j) e tomamos $h(G) = 1$, temos que

$$\begin{aligned}\mathbb{P}_\theta(G) &\propto e^{\theta\dot{\sum}_{(i,j)\in E(G)} G_{ij}} \\ &\propto \prod_{(i,j)\in E(G)} e^{\theta G_{ij}}\end{aligned}$$

Assim, a função de probabilidade fica dada por

$$\prod_{(i,j)\in E(G)} e^{\theta G_{ij}}(1-e^{\theta}),$$

que corresponde a um modelo onde cada aresta G_{ij} pode ser representada de forma independente por uma distribuição geométrica com probabilidade de sucesso igual a $(1-e^{\theta})$. Por outro lado, se tomamos $h(G) = \prod_{(i,j)\in E(G)} \frac{1}{G_{ij}!}$ temos que

$$\begin{aligned}\mathbb{P}_\theta(G) &\propto \frac{e^{\theta\dot{\sum}_{(i,j)\in E(G)} G_{ij}}}{\prod_{(i,j)\in E(G)} G_{ij}!} \\ &\propto \prod_{(i,j)\in E(G)} \frac{e^{\theta G_{ij}}}{G_{ij}!},\end{aligned}$$

e a sua função de probabilidade fica dada por

$$\mathbb{P}_\theta(G) = \prod_{(i,j)\in E(G)} \frac{e^{\theta G_{ij}}}{G_{ij}! e^{e^{\theta}}}, \tag{4.1.2}$$

que corresponde ao modelo onde os valores das arestas possuem distribuição Poisson com parâmetro e^{θ}. Este Capítulo tem o objetivo de obter estimadores para o parâmetro $\boldsymbol{\theta}$ da distribuição com a função $h(G) = \prod_{(i,j)\in E(G)} \frac{1}{G_{ij}!}$, pois este corresponde ao modelo Poisson, que é um dos modelos estatísticos mais utilizados para dados de contagem.

4.2 Inferência para Grafos Aleatórios Valorados

A função de log-verossimilhança do modelo proposto na Seção 4.1, para um grafo observado denotado por g_o é dada por

$$\begin{aligned}\ell(\boldsymbol{\theta}) &= \log\left[h(\boldsymbol{g_o})\frac{e^{\boldsymbol{\theta}\cdot Z(\boldsymbol{g_o})}}{\kappa(\boldsymbol{\theta})}\right] \\ &= \log\left[h(\boldsymbol{g_o})\right] + \boldsymbol{\theta}\cdot Z(\boldsymbol{g_o}) - \log\left[\kappa(\boldsymbol{\theta})\right].\end{aligned}$$

A principal dificuldade para a maximização da função acima, assim como visto no Capítulo 3, é a estimação da constante normalizadora $\kappa(\boldsymbol{\theta})$. No entanto, a mesma aproximação utilizada anteriormente pode ser adaptada para este caso

$$\frac{\kappa(\boldsymbol{\theta})}{\kappa(\boldsymbol{\theta_0})} = \mathbb{E}_{\theta_0}\left[e^{(\boldsymbol{\theta}-\boldsymbol{\theta_0})\cdot\boldsymbol{Z(G)}}\right],$$

onde $\boldsymbol{\theta_0} \in \Theta$ é um vetor de parâmetros conhecido e G é um grafo aleatório com função de probabilidade indexada por $\boldsymbol{\theta_0}$. Utilizando a Lei dos Grandes Números, pode-se aproximar a razão acima por

$$\frac{\kappa(\boldsymbol{\theta})}{\kappa(\boldsymbol{\theta_0})} \approx \frac{1}{m}\sum_{i=1}^{m} e^{(\boldsymbol{\theta}-\boldsymbol{\theta_0})\cdot\boldsymbol{Z(g_i)}},$$

onde $\boldsymbol{g_i}, i = 1, \cdots, m$ é uma amostra de m grafos aleatórios valorados com distribuição dada por $\mathbb{P}_{\theta_0}$. Assim como no Capítulo 3, podemos aproximar $r_m(\boldsymbol{\theta}, \boldsymbol{\theta_0}) = \ell(\boldsymbol{\theta}) - \ell(\boldsymbol{\theta_0})$ por

$$\hat{r}_m(\boldsymbol{\theta}, \boldsymbol{\theta_0}) = (\boldsymbol{\theta} - \boldsymbol{\theta_0})\cdot\boldsymbol{Z(g_o)} - \log\left[\frac{1}{m}\sum_{i=1}^{m} e^{(\boldsymbol{\theta}-\boldsymbol{\theta_0})\cdot\boldsymbol{Z(g_i)}}\right],$$

O Estimador de Máxima Verossimilhança de $\boldsymbol{\theta}$ será denotado $\hat{\boldsymbol{\theta}}$ e maximiza

a função $\hat{r}_m(\boldsymbol{\theta}, \boldsymbol{\theta_0})$.

No entanto, obter uma amostra de grafos aleatórios gerados sob a distribuição $\mathbb{P}_{\theta_0}$ será um desafio maior para o caso de grafos valorados. Para isso, iremos implementar o algoritmo proposto por Krivitsky (2012) e que está explicado na Seção 4.3. Este algoritmo foi proposto para a geração de grafos aleatórios com a distribuição dada na Equação 4.1.1.

4.3 Simulação Metropolis-Hastings de Grafos Aleatórios Exponenciais com Distribuição Poisson

Nesta Seção, apresentaremos o algoritmo proposto por Krivitsky (2012) para simular um grafo aleatório com distribuição Poisson. Definimos $U(0,1)$ o valor obtido após a simulação de uma variável aleatória com distribuição uniforme no intervalo $(0,1)$ e $Poisson_{\neq g^{(t-1)}}(g_{ij}^{(t-1)})$ o valor aleatório gerado sob distribuição Poisson com média $g_{ij}^{(t-1)}$, condicional a ser diferente de $g_{ij}^{(t-1)}$.

Algoritmo 1: V-ERGM Poisson

Entrada: $g^{(0)}, T, \pi_0, \boldsymbol{\theta}$

Saída: V-ERGM simulado sob a distribuição $\boldsymbol{\theta}$

início

  para $t = 1, \cdots, T$ **faça**

    Gere um par $(i, j) \in V^{(2)}$ aleatoriamente

    Se $g_{ij}^{(t-1)} \neq 0$ e $U(0,1) < \pi_0$, então $g_{ij}^{*} \leftarrow 0$

    Caso contrário $g_{ij}^{*} \leftarrow Poisson_{\neq g^{(t-1)}}(g_{ij}^{(t-1)})$

    Calcule $q \leftarrow \begin{cases} \frac{\pi_0 + (1-\pi_0)p(0; g_{ij}^{*})}{p(g_{ij}^{*}, 0)}, & \text{se } g_{ij}^{(t-1)} = 0, \\ \frac{p(g_{ij}^{t-1}; 0)}{\pi_0 + (1-\pi_0)p(0; g_{ij}^{t-1})}, & \text{se } g_{ij}^{(t-1)} \neq 0 \text{ e } g_{ij}^{*} = 0, \\ \frac{(1-\pi_0)p(g_{ij}^{t-1}; g_{ij}^{*})}{(1-\pi_0)p(g_{ij}^{*}, g_{ij}^{t-1})}, & \text{caso contrário.} \end{cases}$

    Calcule $r \leftarrow q \times \frac{g_{ij}^{(t-1)}!}{g_{ij}^{*}!} \exp\left(\boldsymbol{\theta} \boldsymbol{\Delta}_{\boldsymbol{ij}}^{\boldsymbol{g_{ij}^{(t-1)}}} \boldsymbol{Z}(g^{(t-1)})\right)$

    Se $U(0,1) \leq r$ então $g_{ij}^{(t)} \leftarrow g_{ij}^{*}$ Caso contrário $g_{ij}^{(t)} \leftarrow g_{ij}^{(t-1)}$

  fim

fim

retorna $g^{(T)}$

4.4 Algoritmo para calcular o estimador EMV

O método de estimação utilizando o método escore de Fisher será muito parecido com o caso dos grafos simples apresentado no Capítulo 3. A estimativa de $\boldsymbol{\theta}$ na iteração $t+1$ fica dada por

$$\hat{\boldsymbol{\theta}}^{(t+1)} = \hat{\boldsymbol{\theta}}^{(t)} + \left[\hat{I}(\boldsymbol{\theta}^{(t)})\right]^{-1} \left[\boldsymbol{Z}_{(\boldsymbol{g_o})} - \sum_{\boldsymbol{i=1}}^{\boldsymbol{m}} \boldsymbol{w}_{\boldsymbol{i}}^{(\boldsymbol{t})} \boldsymbol{Z}_{\boldsymbol{i}}\right], \tag{4.4.3}$$

sendo $\hat{I}(\boldsymbol{\theta^{(t)}})^{-1}$ a matriz de informação de Fisher inversa aproximada na iteração t. No entanto, com a inclusão da função $h(\cdot)$, os pesos podem ser escritos como

$$w_i^{(t)} = \frac{h(g_i)\exp\left[\boldsymbol{\theta^{(t)}} - \boldsymbol{\theta_0}\right]\cdot Z_i}{\sum_{j=1}^{m} h(g_i)\exp\left[\boldsymbol{\theta^{(t)}} - \boldsymbol{\theta_0}\right] Z_i}, i = 1, \ldots, m.$$

Utilizando o mesmo critério de parada do Capítulo 3, nesse caso temos que

$$\hat{\ell}(\tilde{\boldsymbol{\theta}}) = \hat{r}_m(\tilde{\boldsymbol{\theta}}, \boldsymbol{\theta_0}) - \hat{r}_m(\mathbf{0}, \boldsymbol{\theta_0}) - \frac{1}{e^{\binom{n}{2}}} \prod_{(i,j)\in E(g_o)} \frac{1}{g_{ij}!}.$$

A matriz de covariância de $\tilde{\boldsymbol{\theta}}$ é calculada de maneira idêntica ao apresentado na Seção 3.4, adaptada para o caso valorado, e é dada por

$$\frac{1}{m}\left[\hat{I}(\tilde{\boldsymbol{\theta}})\right]^{-1} \tilde{A} \left[\hat{I}(\tilde{\boldsymbol{\theta}})\right]^{-1},$$

onde $\tilde{A} = \frac{1}{m^2}\left[\sum_{i=1}^{m} \exp\{[\boldsymbol{\theta_0} - \tilde{\boldsymbol{\theta}}]^t \boldsymbol{Z_i}\}\right]^2 \sum_{k=-K}^{K} \hat{\xi}_k$ e $\hat{\xi}_k$ é a matriz de autocovariância dos pesos $W_1, \ldots, W_m$ de lag k.

4.5 Estudo de Simulação

Nessa seção serão apresentados alguns resultados de simulações realizadas para avaliação do método de estimação proposto na Seção 4.4. Foram implementados dois modelos: o primeiro, apresentado na Seção 4.5.1 considera o modelo apresentado em 4.1.2 em que há apenas um parâmetro para a distribuição de referência Poisson, com arestas independentes. Na Seção 4.5.2 estão apresentados os resultados do modelo biparamétrico, que considera a distribuição Poisson e uma medida de correlação entre os valores das arestas.

Durante a implementação do algoritmo, foi observado que o algoritmo

converge e gera bons resultados apenas quando o vetor de parâmetros inicial do algoritmo $\boldsymbol{\theta_0}$ é próximo ao vetor de parâmetros verdadeiro $\boldsymbol{\theta}$, o que implica que a escolha $\boldsymbol{\theta_0}$ é essencial para a convergência do algoritmo. No entanto, Hunter e Handcock (2006) não apresentam um critério para a escolha desse vetor de parâmetros inicial. No modelo uniparamétrico, utilizamos a relação entre o parâmetro da distribuição de referência Poisson e^{θ} para estabelecer o seguinte critério:

$$\theta_0 = \log\left[\frac{\sum_{(i,j)\in E(\boldsymbol{G})} G_{ij}}{\binom{n}{2}}\right]$$

Para o modelo multiparamétrico, Krivitsky e Butts (2013) sugerem dois métodos diferentes para definição de $\boldsymbol{\theta_0}$: o primeiro consiste em definir $\boldsymbol{\theta_0} = 0$ e o segundo método consiste em utilizar $\theta_{01} = \log\left[\frac{\sum_{(i,j)\in \boldsymbol{g}} g_{ij}}{\binom{n}{2}}\right]$ e definir $\theta_{0i} = 0$, para $i > 1$. O primeiro método não foi implementado nesse trabalho, pois inicialmente verificamos que apenas valores de $\boldsymbol{\theta_0}$ próximos a $\boldsymbol{\theta}$ poderiam gerar bons resultados. Na Seção 4.5.2 implementamos o segundo método proposto.

Assim como no Capítulo 3, utilizamos $\theta^{(t)} - \theta^{(t-1)} < 0,0001$ para verificar a convergência da sequência $\theta^{(t)}$; e fixamos $k = 4$ e $K = 10$. Para o modelo uniparamétrico, o tempo computacional para cada ajuste foi de aproximadamente 2 minutos para o modelo com 100 simulaçoes Monte Carlo e 16 minutos para 1000 simulações, enquanto o modelo multiparamétrico com $m = 100$ simulações demora aproximadamente 1 hora e 15 minutos para cada simulação, com $n = 12$ vértices.

4.5.1 Modelo Uniparamétrico

A distribuição de referência nesse caso é dada por

$$\mathbb{P}_\theta(g) = h(\boldsymbol{g})\frac{e^{\boldsymbol{\theta} Z(\boldsymbol{g})}}{\kappa(\boldsymbol{\theta})},$$

onde $h(\boldsymbol{g}) = \prod_{(i,j)\in \boldsymbol{g}} \frac{1}{g_{ij}!}$ e $Z(\boldsymbol{g}) = \sum_{(i,j)\in \boldsymbol{g}} g_{ij}$. Esse modelo corresponde ao caso onde os valores atribuídos a cada aresta são independentes, com distribuição Poisson com taxa e^θ. A Tabela 4.1 apresenta os resultados de estimação quando o número de vértices $n = 5$, para diferentes valores de θ e utilizamos $m = 100$ e $m = 1000$ simulações MCMC. Note que, quando consideramos θ entre -2 e 4, as estimativas $\tilde{\theta}$ foram próximas aos valores verdadeiros de θ. As variâncias estimadas apresentam valores muito pequenos, entre 0,0000012 e 0,57. Em geral, os modelos que utilizam $m = 1000$ simulações apresentam variância estimada menores que os modelos que utilizam $m = 100$ simulações. Ainda verificamos que $m = 100$ simulações MCMC é suficiente para gerar boas estimativas de θ.

Tabela 4.1: Resultados para $n = 5$ vértices.

θ	θ_0	$m = 100$ simulações		$m = 1000$ simulações	
		$\tilde{\theta}$	$Var(\tilde{\theta})$	$\tilde{\theta}$	$Var(\tilde{\theta})$
-2	-2,303	-2,285	0,0060457	-1,593	0,0006381
-1	-0,693	-0,613	0,0177808	-0,927	0,0002120
0	0,336	0,327	0,0006115	0,269	0,0000965
1	1,030	1,046	0,0032495	1,031	0,0000665
2	1,872	1,863	0,0000053	2,081	0,0000142
3	3,073	3,068	0,0000012	2,921	0,0000010
4	4,050	4,055	0,5706785	4,038	0,0000033

A Tabela 4.2 apresenta os resultados da análise de sensibilidade para $\theta = 1$, para diferentes valores de n. Para n maior que 40, o algoritmo não converge. Observamos que, em todos os casos foram realizadas boas estimações para θ, com as estimativas $\tilde{\theta}$ variando entre 0,894 e 1,025. Ainda, podemos notar que, assim como no caso anterior, os valores das variâncias estimadas são muito próximos a zero.

Tabela 4.2: Análise de sensibilidade para $\theta = 1$ e $m = 100$ simulações.

n	θ_0	$\tilde{\theta}$	$Var(\tilde{\theta})$
6	1,030	1,021	0,0000088
7	0,887	0,894	0,0007780
8	1,025	1,025	0,0000359
9	0,991	0,988	0,0000148
10	0,912	0,907	0,0000171
15	0,933	0,944	1,0603290
20	1,018	1,023	0,4531279
30	0,951	0,953	0,3851729
40	0,989	0,972	9,72e-70

Fixando $n = 12$, foram ajustados 100 modelos para cada um dos seguintes valores de $\theta : 0, 1, 2$ e 3. A Tabela 4.3 apresenta as estimativas médias dos resultados obtidos, onde podemos observar novamente que a estimativa θ_0 fornece uma boa aproximação para θ. A Figura 4.2 apresenta os valores estimados de θ_0 e $\tilde{\theta}$, para $\theta = 1$ em cada uma das 100 repetições. Observamos que há uma forte relação linear entre estes dois valores, ou seja, para os valores de θ_0 mais distantes de 1, temos $\tilde{\theta}$ também mais distantes de 1, ou seja, o algoritmo não consegue melhorar a estimativa inicial θ_0.

Tabela 4.3: Análise de sensibilidade para 100 repetições em cada modelo, $n = 12$, $m = 100$ simulações MC e 100 repetições.

θ	θ_0 Médio	$\tilde{\theta}$ Médio	$Var(\tilde{\theta})$ Média
0	0.00066	0.00016	0.0000643
1	0.97802	0.97708	0.0000521
2	1.99558	1.99564	0.0000152
3	3.00178	3.00189	0.0000029

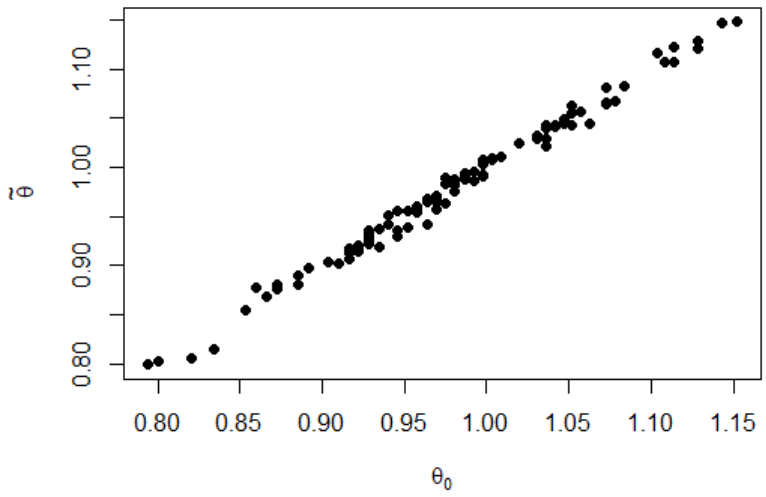

Figura 4.2: θ_0 e $\tilde{\theta}$ estimados,$\theta = 1$, $n = 12$, $m = 100$.

4.5.2 Modelo Biparamétrico

Nesta Seção iremos considerar o modelo com função de probabilidade

$$\mathbb{P}_\theta(g) = h(\boldsymbol{g})\frac{e^{\boldsymbol{\theta} Z(\boldsymbol{g})}}{\kappa(\boldsymbol{\theta})},$$

onde $h(\boldsymbol{g}) = \prod_{(i,j)\in \boldsymbol{g}} \frac{1}{g_{ij}!}$ e $\boldsymbol{Z}(\boldsymbol{g}) = (Z_1(\boldsymbol{g}), Z_2(\boldsymbol{g}))$ para $Z_1(\boldsymbol{g}) = \sum_{(i,j)\in \boldsymbol{g}} g_{ij}$ e $Z_2(\boldsymbol{g}) = \sum_{(i,j)\in \boldsymbol{E(g)}, k\in V(g)} g_{ik}g_{kj}$. Esse modelo é uma extensão do modelo Poisson apresentado na Seção 4.5.1, considerando uma correlação entre as arestas. A Tabela 4.4 apresenta os resultados de estimação para o número de vértices $n = 8$ e para diferentes valores de θ. Nesse caso, fixamos θ_0 igual a θ, o vetor de parâmetros verdadeiro. Verificamos que houve convergência do algoritmo em quase todas as simulações. Nesse caso, utilizar $m = 100$ simulações MCMC não foi suficiente para obter a convergência do algoritmo, portanto foram utilizadas $m = 1000$ simulações MCMC por modelo. Obser-

vamos que as estimativas de $\tilde{\boldsymbol{\theta}}$ estão próximas aos respectivos valores de $\boldsymbol{\theta}$ e, em todos os casos, a variância estimada de $\tilde{\theta_1}$ é maior que a variância de $\tilde{\theta_2}$. Ainda, podemos observar que a correlação é aproximadamente igual a -1 ou 1 em todos os casos, sendo negativa na grande maioria das simulações.

Tabela 4.4: Análise de sensibilidade para $n = 8$ e $m = 1000$ simulações.

θ_1	θ_2	$\tilde{\theta_1}$	$\tilde{\theta_2}$	$Var(\tilde{\theta_1})$	$Var(\tilde{\theta_2})$	$Cor(\tilde{\theta_1}, \tilde{\theta_2})$
0,00	-1,00	1,3318	-0,7505	0,00588	0,00004	0,9999
0,00	-0,50	0,1585	-0,6663	0,00752	0,00018	-0,9999
0,00	0,00	1,9292	-0,1290	8,26484	0,03875	-0,9999
0,00	0,50	0,2358	0,4998	9287,23200	0,01379	-1,0000
0,00	1,00	0,3944	0,9995	243417,70000	0,64107	-1,0000
0,00	2,00	0,0059	1,9999	6009,21400	0,05177	-0,9999
0,00	3,00	0,4598	2,9983	1655,82800	0,03028	-1,0000
0,00	4,00	-1,8132	4,0108	21731,82000	0,72131	-0,9999
0,00	5,00	-0,3617	5,0028	3041,75300	0,15949	-0,9999
1,00	-0,50	1,13410	-0,56852	0,00643	0,00009	-1,0000
1,00	0,50	0,96603	0,50009	32459,60000	0,04671	-0,9999
1,00	1,50	1,31288	1,49986	6678,48100	0,03159	-1,0000
1,00	3,00	2,06633	2,99603	2537,11500	0,04180	-0,9999
1,00	4,00	2,04360	3,99567	5566,64400	0,14201	-1,0000
1,00	5,00	0,63830	5,00281	3041,75300	0,15949	-0,9999
2,00	-0,50	2,07406	-0,48245	0,00015	0,00003	0,9999
2,00	0,00	2,08120	-0,00094	1344,10300	0,17035	-0,9999
2,00	0,50	2,09768	0,49994	20359,62000	0,02971	-0,9999
2,00	1,00	1,93121	1,00019	363800,60000	0,97414	-0,9999
2,00	2,25	2,01059	2,24994	8608,54900	0,09381	-0,9999
2,00	3,00	3,21412	2,99551	2879,15800	0,04605	-1,0000
2,00	4,00	0,67955	4,00741	14272,90000	0,47636	-1,0000
2,00	5,00	1,66351	5,00265	3040,92800	0,15955	-0,9999
3,00	0,50	2,72885	0,50041	54778,08640	0,07697	-1,0000
3,00	1,00	3,25464	0,99963	308632,40420	0,83712	-0,9999
3,00	2,00	2,31944	2,00252	12477,05820	0,10838	-1,0000
3,00	3,00	2,68219	3,00151	8252,74020	0,15664	-1,0000
3,00	4,00	3,75616	3,99655	310,90860	0,00896	-0,9999
3,00	5,00	2,66406	5,00264	3086,64810	0,16204	-0,9999
4,00	0,00	3,84497	0,00026	356685,83440	0,80613	-1,0000
4,00	1,00	4,11278	0,99990	332433,86230	0,89247	-1,0000
4,00	2,00	3,92428	2,00029	5628,38020	0,04949	-1,0000
4,00	4,00	4,75616	3,99655	310,90860	0,00896	-1,0000
4,00	5,00	3,66406	5,00264	3086,64810	0,16204	-0,9999

Posteriormente, utilizamos $\boldsymbol{\theta_0}$ proposto por Krivitsky e Butts (2013)

$$\theta_{01} = \log\left[\frac{\sum_{(i,j)\in \boldsymbol{g}} g_{ij}}{\binom{n}{2}}\right], \theta_{02} = 0.$$

Podemos observar os resultados para esta análise na Tabela 4.5. Observamos que o algoritmo convergiu apenas para o caso em que $\theta_2 \leq 0$ e, mesmo fixando $\theta_{02} = 0$ existe convergência para valores próximos ao verdadeiro $\theta_2 = 0$. A covariância entre esses estimadores também é negativa, ou seja, durante a convergência do algoritmo eles se relacionam de maneira inversa.

Tabela 4.5: Análise de sensibilidade para $n = 8$, $m = 1000$ simulações e $\boldsymbol{\theta_0}$ estimado.

θ_1	θ_2	$\tilde{\theta_1}$	$\tilde{\theta_2}$	$Var(\tilde{\theta_1})$	$Var(\tilde{\theta_2})$	$Cov(\tilde{\theta_1}, \tilde{\theta_2})$
0	-1	-1,554	-0,075	0,0189	0,0020	-0,0061
0	-0,5	-0,245	-0,486	0,0197	0,0001	-0,0016
0	0	0,578	-0,029	12,7788	0,0596	-0,8726
1	-1	0,763	-0,689	0,1030	0,0005	-0,0069
1	0	1,354	-0,003	148,5260	0,0889	-3,6334
3	0	3,659	-0,003	49092,7400	0,8462	-203,8165

Krivitsky (2012) discute a dificuldade de convergência do modelo Poisson ao considerarmos o modelo com estatística $Z_2(\boldsymbol{g}) = \sum_{(i,j)\in \boldsymbol{E}, k\in V} g_{ik}g_{kj}$:

> "Nós consideramos outras variantes, incluindo a versão não centralizada, em que cada termo é apenas $\sqrt{g_{i,j}g_{i,k}}$. Descobrimos que, particularmente em redes não direcionadas, tal termo pode induzir uma distribuição de probabilidade bimodal."(Krivitsky, 2012)

Nesse caso, uma distribuição bimodal justifica a não convergência do algoritmo.

Capítulo 5

Aplicação

Neste Capítulo apresentaremos uma aplicação dos modelos para grafos exponenciais ERGM apresentados nos Capítulos 3 e 4, no contexto econômico de Complexidade dos Produtos proposto por Hidalgo et al. (2007). Buscando responder perguntas clássicas da economia, do tipo: "Qual a consequência do tipo de produto que um país exporta para a sua performance econômica?", os autores propuseram um modelo econômico através da teoria dos grafos. Enquanto a teoria econômica clássica geralmente utiliza os fatores produtivos para responder tal pergunta, Hidalgo et al. (2007) definem o produto exportado como uma materialização da capacidade produtiva de uma região. Assim, cada produto possui um conjunto de capacidades e, produtos que utilizam as mesmas capacidades possuem uma alta probabilidade de serem exportados conjuntamente.

Conforme será apresentado na Seção 5.1, podemos criar uma medida de similaridade entre os produtos e, representar tais produtos e suas relações em um grafo, que será chamado Espaço de Produtos. Na Seção 5.2 será definida uma extensão do modelo proposto por Hausmann et al. (2011) e a base de dados utilizada. Na Seção 5.3 serão apresentados os resultados obtidos e

iremos discutir tais resultados e seu contexto.

5.1 O Espaço de Produtos

Hidalgo et al. (2007) utilizam a ideia inicial que, se dois produtos são relacionados, por utilizarem matéria-prima, infraestrutura ou tecnologia similares, então eles tendem a serem produzidos conjuntamente. Dessa forma, podemos estabelecer uma relação de semelhança entre dois produtos i e j se os países que se especializam no produto i também produzem o produto j, e vice-versa. Em outras palavras, podemos dizer que dois produtos estão relacionados se eles requerem insumos similares e então tendem a ser produzidos conjuntamente, enquanto os bens dissimilares são pouco prováveis de serem produzidos conjuntamente. Para criar então uma medida de similaridade entre produtos, utiliza-se o conceito de vantagem comparativa.

A vantagem comparativa revelada (RCA) é um índice utilizado para se calcular a vantagem ou desvantagem de uma certa região na produção de um bem ou serviço (Mankiw (2012)). O RCA_{ik} de uma região k em um produto i é dado por

$$RCA_{ik} = \frac{\frac{X_i^k}{\sum_i X_i^k}}{\frac{\sum_k X_i^k}{\sum_i \sum_k X_i^k}},$$

onde X_i^k representa o valor total de exportações do produto i pela região k. Em outras palavras, a vantagem comparativa é uma comparação entre a concentração do produto i na região k e a concentração desse produto na economia global. Se $RCA_{ik} > 1$, dizemos que a região k possui vantagem comparativa na produção do bem i, caso contrário, a região não possui vantagem comparativa.

Uma vez definido o conceito de vantagem comparativa, podemos agora

definir a similaridade entre dois produtos

$$\phi_{ij} = \min \{P(RCA_{ik} > 1 | RCA_{jk} > 1), P(RCA_{jk} > 1 | RCA_{ik} > 1)\}.$$

Utilizando a medida acima, podemos criar uma matriz $\boldsymbol{M}$ de similaridades, onde o elemento $\boldsymbol{M}_{i,j}$ é composto pela similaridade ϕ_{ij} entre os produtos i e j. Tal matriz pode ser representada por um grafo, que chamaremos de Espaço de Produtos.

Para que o grafo produzido seja conexo, utiliza-se o algoritmo proposto por Cormen et al. (2009) para gerar a Árvore Geradora Máxima (AGM) da matriz de similaridade $\boldsymbol{M}$. O MST é o conjunto de arestas que conecta todos os vértices da rede utilizando um número mínimo de arestas e maximiza a soma das similaridades. Portanto, a AGM inclui todos os produtos, criando assim um grafo conexo, com o número de arestas mínimo. O segundo passo para criação do Espaço de Produtos é adicionar arestas entre os produtos similares que não foram adicionadas pelo método MST. Hausmann et al. (2011) incluem todas as similaridades maiores ou iguais a $0,55$.

Assim, utilizando o banco de dados de exportação de um ano, pode-se estimar a matriz de similaridades e produzir o espaço de produtos daquele ano. Neste trabalho, iremos também utilizar a ideia de um espaço de produtos valorado, contendo informações das relações entre os produtos por vários anos. Assim, o Espaço de Produtos Valorado é composto pela soma dos valores do espaço de produtos de vários anos.

5.2 O Banco de Dados

Para a estimação das matrizes de similaridades e consequentemente, criação dos grafos, foram utilizados os dados de comércio internacional fornecido

pelo UN Comtrade (2017). O UN Comtrade é um repositório de estatísticas oficiais de comércio global, e são de direitos autoriais da Organização das Nações Unidas. Os dados foram disponibilizados pela Fundação de Amparo à Pesquisa do Estado de Minas Gerais (FAPEMIG) através do projeto DataViva. A base de dados contém as seguintes informações:

- Ano de referência,
- Código do país exportador,
- Código SITC (Standard International Trade Classification) do produto exportado,
- Valor da exportação correspondente, em US$.

Para esta dissertação foram utilizados os dados referentes aos seguintes anos: 2015, 2010, 2005, 2000, 1995, 1990, 1985 e 1980. Ainda, seguindo a metodologia proposta por Hausmann et al. (2011), foram retirados da análise os dados dos países cujo valor médio de exportação no período entre 2013 e 2015 foi inferior a 1 bilhão de dólares, e também as exportações dos países que possuem população menor que $1,2$ milhões de pessoas no ano de 2015.

Após o tratamento dos dados, são apresentados dados de 109 países. O Espaço de Produtos proposto por Hidalgo et al. (2007) contém todos os 1467 produtos codificados, divididos em 9 seções, ou grupos de produtos. Neste trabalho iremos utilizar apenas os produtos da Seção 1, que representa Bebidas e Tabacos. Esta seção contém 12 produtos, que estão descritos na Tabela 5.1.

Tabela 5.1: Produtos que Compõem a Seção 1 - SITC-Rev.2

Código	**Descrição**
11101	Águas (incluindo águas de spa e águas gaseificadas); gelo e neve
11102	Limonada, águas aromatizadas, águas gaseificadas aromatizadas e outras bebidas não alcoólicas
11211	Vinho suave, em fermentação ou com fermentação interrompida, exceto pela adição de álcool
11212	Vinho de uvas frescas; vinho de uva com fermentação interrompida pela adição de álcool
11213	Vermouths e outros vinhos de uvas frescas aromatizados com extratos aromáticos
11241	Uísque
11242	Álcool obtido com vinho destilado ou bagaço de uva
11249	Álcool e bebidas alcoólicas destiladas, preparações alcoólicas para fabricação de bebidas
12111	Tabaco, não descascado, curado do tipo Virginia
12119	Tabaco, não descascado, exceto o tipo curado da Virgínia
12121	Tabaco, totalmente ou parcialmente descascado, curado, do tipo Virginia
12129	Tabaco, total ou parcialmente descascado, exceto do tipo Virgínia

Este subconjunto de produtos, chamado Seção 1, apresenta uma média de $1,17\%$ da exportação mundial no período considerado. Aplicando o algoritmo obtemos a seguinte matriz de adjacência valorada, representada na Tabela 5.2.

Tabela 5.2: Matriz de Adjacência Observada - Seção 2 - Espaço de Produtos

Produto	11101	11102	11211	11212	11213	11241	11242	11249	12111	12119	12121	12129
11101	0	8	0	2	1	0	0	0	1	0	0	1
11102	8	0	0	0	1	1	1	8	0	0	0	0
11211	0	0	0	5	6	0	2	0	0	1	0	0
11212	2	0	5	0	5	0	1	1	0	0	0	0
11213	1	1	6	5	0	1	0	3	0	0	0	0
11241	0	1	0	0	1	0	3	4	1	0	0	0
11242	0	1	2	1	0	3	0	1	0	1	0	1
11249	0	8	0	1	3	4	1	0	0	0	0	0
12111	1	0	0	0	0	1	0	0	0	2	3	2
12119	0	0	1	0	0	0	1	0	2	0	2	2
12121	0	0	0	0	0	0	0	0	3	2	0	2
12129	1	0	0	0	0	0	1	0	2	2	2	0

A Figura 5.1 apresenta o Grafo Espaço de Produtos restrito à Seção 1, apresentado na Tabela 5.2.

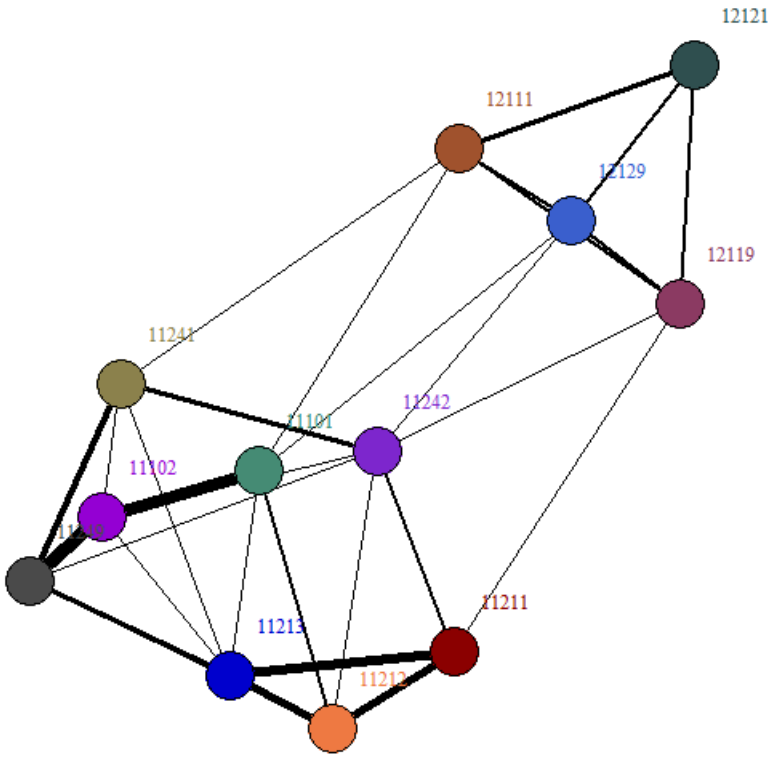

Figura 5.1: Espaço de Produtos Valorado para a Seção 1.

Podemos observar que o produto 11102 - Limonada, águas aromatizadas, águas gaseificadas e outras bebidas não alcoólicas - apresenta as relações de maiores valores estabelecidas no grafo observado, com os produtos 11101 - Águas, gelo e neve e 11249 - Álcool e bebidas alcoólicas destiladas, preparações alcoólicas para fabricação de bebidas, sendo que ambos foram considerados similares ao produto 11102 em todas as oito versões dos grafos

estimados. Os produtos 1121 - Vinho Suave,... e 11213 - Vermouths e outros vinhos ... apresentam a segunda maior relação, sendo que esses produtos foram similares em seis dos oito casos analisados. Ainda, o produto 11212 - Vinho de uvas frescas,... apresenta uma forte similaridade com os dois citados anteriormente, sendo o valor das arestas iguais a 5. O valor médio das arestas observado é igual a 1,106 e densidade igual a 0,455, ou seja, 55% das arestas apresentam valor nulo.

5.3 Estimação VERGM e Resultados

As estatísticas observadas para o Espaço de Produtos apresentado na Figura 5.1 são $Z_1 = 73$ e $Z_2 = 680$. Para o modelo uniparamétrico, temos $\theta_0 = 0,1008047$ e o valor estimado do parâmetro θ é $\tilde{\theta} = 0.0913$ e a variância estimada é $Var(\tilde{\theta}) = 1.05e-05$, utilizando m=100 simulações. Se considerarmos 1000 simulações, o parâmetro estimado é igual a $\tilde{\theta} = 0.1013706$ e a sua variância é $Var(\tilde{\theta}) = 1.21e-05$. Não houve convergência para o modelo biparamétrico.

Para verificar se o modelo Poisson independente é adequado para representar o grafo observado, foi realizado um estudo de simulação para verificar a qualidade do ajuste de tal modelo. Foram simulados 1000 grafos com distribuição indexada por $\theta_1 = 0,1008047$ e $\theta_2 = 0$, e para cada simulação foram calculadas as estatísticas Z_1 e Z_2. A distribuição das estatísticas simuladas estão representadas na Figura 5.2. Verificamos que, um grafo com distribuição $\boldsymbol{\theta} = (0,101; 0)$ tem alta probabilidade de gerar o grafo observado, ou seja, podemos concluir que o modelo Poisson independente é adequado para representar o Espaço de Produtos - Seção 1.

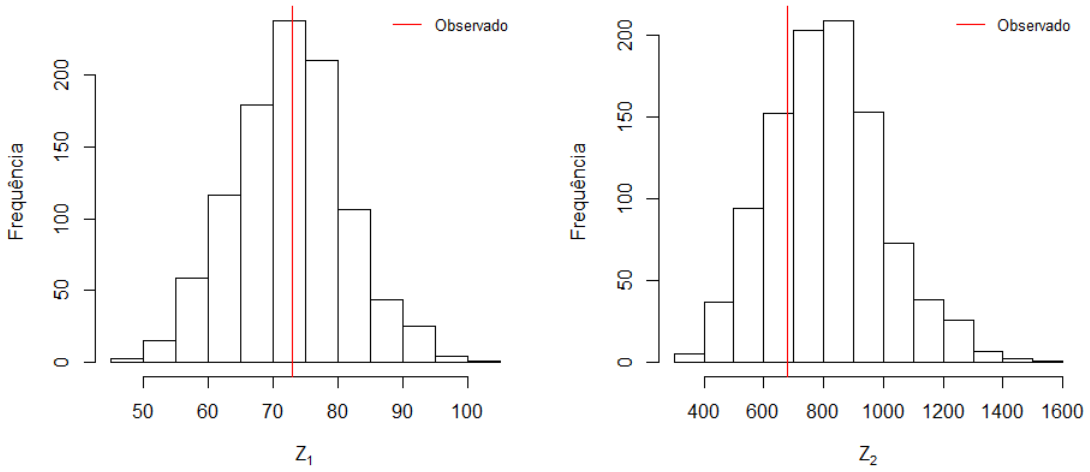

Figura 5.2: Distribuição das estatísticas observadas para $\theta_1 = 0,101$ e $\theta_2 = 0$.

Capítulo 6

Conclusão e Trabalhos Futuros

Nesse trabalho foram realizadas duas contribuições principais para o estudo de grafos aleatórios: primeiramente, estendemos e implementamos o método de estimação por máxima verossimilhança via MCMC para o modelo de grafos aleatórios exponenciais valorados com distribuição de referência Poisson e com correlação entre os valores das arestas. Até o final desta dissertação não identificamos nenhum trabalho que tenha implementado tal modelo. Vimos que tal medida de correlação pode ocasionar a não-convergência do método proposto, mas para os modelos em que houve convergência observamos que frequentemente o valor inicial do algoritmo θ_0 é uma estimativa para θ melhor que o $\tilde{\theta}$. Implementamos também o algoritmo para simulação de grafos aleatórios valorados com distribuição Poisson.

Posteriormente, aplicamos os métodos citados acima em um problema prático da área econômica, chamado Espaço de Produtos, e observamos que o modelo Poisson independente é adequado para representar tal modelo.

Como possíveis trabalhos futuros, podemos apontar

- Aplicação do método de estimação por máxima verossimilhança via

MCMC para outras estatísticas, inclusive as variações da medida de correlação Z_2 sugeridas por Krivitsky (2012);

- Aplicação do algoritmo de simulação de grafos aleatórios para outras distribuições, particularmente para a distribuição Geométrica;
- Estender também o método de estimação por máxima verossimilhança via MCMC para a distribuição geométrica;
- Implementar medidas de qualidade de ajuste para as estimações e
- Otimizar o tempo computacional das simulações.

Bibliografia

Barabási, A.-L. (2016), *Network Science*, Cambridge University Press.

Besag, J. (1974), "Spatial Interaction and the Statistical Analysis of Lattice Systems," *Journal of the Royal Statistical Society Series B (Methodological)*, 36.

Casella, G. e Berger, R. L. (2002), *Statistical Inference*, Duxbury Advanced Series.

Chatterjee, S. e Diaconis, P. (2013), "Estimating and understanding exponential random graph models," *The Annals of Statistics*, 41.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., e Stein, C. (2009), *Introduction to Algorithms*, Massachusetts Institute of Technology.

Feofiloff, P., Kohayakawa, Y., e Wakabayashi, Y. (2011), *Uma Introdução Sucinta à Teoria dos Grafos*.

Geyer, C. J. (1994), "On the Convergence of Monte Carlo Maximum Likelihood Calculations," *Journal of the Royal Statistical Society*, pp. 261–274.

Gopal, S. (2007), *The envolving social geography of blogs*, Springer.

Hausmann, R., Hidalgo, C. A., Bustos, S., Coscia, M., Chung, S., Jimenez, J., Simoes, A., e Yildirim, M. A. (2011), *The Atlas of Economic Complexity: Mapping Paths to Prosperity*, Puritan Press.

Hidalgo, C. A., Klinger, B., Barabási, A.-L., e Hausmann, R. (2007), "The Product Space Conditions the Development of Nations," *Science*, 317, 482–

487.

Hunter, D. R. e Handcock, M. S. (2006), "Inference in Curved Exponential Family Models for Networks," *Journal of Computational and Graphical Statistics*, 15, 565–583.

James, B. R. (ed.) (2015), *Probabilidade: um curso em nível intermediário*, IMPA.

Kolaczyk, E. D. (2009), *Statistical Analysis of Network Data: Methods and Models*, Springer.

Kolaczyk, E. D. e Csárdi, G. (2014), *Statistical Analysis of Network Data with R*, Springer Series in Statistics, Springer.

Krivitsky, P. N. (2012), "Exponential-family random graph models for valued networks," *Electron. J. Statist.*, 6, 1100–1128.

Krivitsky, P. N. e Butts, C. T. (2013), "Modeling Valued Networks with statnet," .

Lusher, D., Koskinen, J., e Robins, G. (eds.) (2012), *Exponential Random Graph Models for Social Networks: Theory, Methods, and Applications (Structural Analysis in the Social Sciences)*, Cambridge University Press.

Mankiw, N. G. (2012), *Principles of Economics*, South-Western Cengage Learning.

R Core Team (2017), *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria.

Robins, G. (2013), "A tutorial on methods for the modeling and analysis of social network data," *Journal of Mathematical Psychology*, 57, 261–274.

Robins, G., Pattison, P., Kalish, Y., e Lusher, D. (2007), "An introduction to exponential random graph (p*) models for social networks," *Social Networks*, 29.

UN Comtrade (2017), "UN Comtrade International Trade Statistics Data-

base," https://comtrade.un.org/, acessado em 03/07/2017.

Varadhan, C. K. S. R. S. (1986), "Central limit theorem for additive functionals of reversible Markov processes and applications to simple exclusions," *Communications in Mathematical Physics*, 104.

Yin, M., DeMuse, R., e Larcomb, D. (2016), "Phase transitions in edge-weighted exponential random graphs," *Journal of Statistical Physics*.

Apêndice A

Estimação

Neste apêndice, iremos apresentar o resultado proposto por Geyer (1994) para calcular a variância (Teorema A.0.1) dos estimadores obtidos via aproximação MCMC da função de verossimilhança para distribuições da família exponencial, através da convergência assintótica desses estimadores. Este resultado foi utilizado por Hunter e Handcock (2006) para obter a variância dos estimadores de máxima verossimilhança via MCMC para grafos aleatórios, e também foi utilizado nessa dissertação para obter a variância do estimador para o modelo de grafos aleatórios valorados.

Teorema A.0.1. *Suponha que sejam válidas as seguintes condições de regularidade*

1. *O EMV* $\hat{\boldsymbol{\theta}}$ *é único e o espaço paramétrico* $\boldsymbol{\Theta}$ *contém uma vizinhança aberta de* $\hat{\boldsymbol{\theta}} \in \mathbb{R}^d$*;*

2. *O MCMCMLE* $\hat{\boldsymbol{\theta}}_{\boldsymbol{m}}$ *converge em probabilidade para o EMV* $\hat{\boldsymbol{\theta}}$*;*

3. $\kappa(\boldsymbol{\theta})$ *é duas vezes diferenciável sob o sinal de integração com relação a* $\boldsymbol{\theta}$*;*

4. $n^{\frac{1}{2}} \nabla \ell_m(\hat{\boldsymbol{\theta}}) \longrightarrow N(0, \boldsymbol{A})$, *para alguma matriz de covariância* $\boldsymbol{A}$*;*

5. $\boldsymbol{B} = -\nabla^2 \ell_m(\hat{\boldsymbol{\theta}})$ *é positiva definida;*

6. $\nabla^3 \ell_m(\boldsymbol{\theta})$ *é limitado em probabilidade uniformemente em uma vizinhança de* $\hat{\boldsymbol{\theta}}$*.*

Então

$$\nabla^2 \ell_m(\hat{\boldsymbol{\theta}}_{\boldsymbol{m}}) \longrightarrow \boldsymbol{B} \tag{A.0.1}$$

em probabilidade e,

$$n^{\frac{1}{2}}(\hat{\boldsymbol{\theta}}_{\boldsymbol{m}} - \hat{\boldsymbol{\theta}}) \longrightarrow N(0, \boldsymbol{B^{-1}AB^{-1}}) \tag{A.0.2}$$

A parte A.0.1 do teorema pode ser demonstrada através da Proposição 6.7(b) de James (2015), enquanto a parte A.0.2 é uma adaptação do método delta para o caso multiparamétrico e pode ser provado através da expansão em série de Taylor (Ver Geyer (1994)).

As condições de regularidade definidas no Teorema A.0.1 são satisfeitas sob algumas condições. A condição 3 é verdadeira quando não há Transição de Fase, ou seja, quando a função de verossimilhança é unimodal. Yin et al. (2016) mostra que há Transição de Fase apenas no subconjunto de Θ onde $\theta_1 < 0$. Ou seja, se $\theta_1 \geq 0$, a condição 3 é satisfeita. A condição 4 apresenta uma versão do Teorema Central do Limite para o método MCMC. Para o caso em que o vetor de parâmetros θ pertence a um espaço contínuo, o método MC é reversível e a soma das autocovariâncias da cadeia é finita, então, a condição 4 é verdadeira (Ver Geyer (1994)). Nos artigos de Hunter e Handcock (2006) e Geyer (1994), tais condições de regularidades são assumidas para todo o espaço paramétrico e os resultados do Teorema A.0.1 são utilizados.

Para encontrar a variância assintótica $\boldsymbol{B^{-1}AB^{-1}}$ do estimador $\tilde{\boldsymbol{\theta}}$, precisamos encontrar a matriz $\boldsymbol{A}$. Seguindo Geyer (1994), para obter a variância podemos escrever $\nabla \hat{r}_m(\theta, \theta_0)$ como

$$\nabla \hat{r}_m(\theta, \theta_0) = \frac{\mathbb{E}_{m,\theta_0}\left[(t_\theta(g_{obs}) - t_\theta(G))\,\frac{H_\theta G)}{H_{\theta_0}(G)}\right]}{\mathbb{E}_{m,\theta_0}\left[\frac{H_\theta(G)}{H_{\theta_0}(G)}\right]}, \tag{A.0.3}$$

onde G é um grafo sobre a distribuição $\mathbb{P}_{\theta_0}$ e $\mathbb{E}_{m,\theta_0}[f(G)] = \frac{1}{m}\sum_{k=1}^{m} f(G_k)$ é o estimador de Monte Carlo de $\mathbb{E}_{\theta_0}[f(G)] = \sum_{g\in\mathcal{G}_n} f(g)\cdot\mathbb{P}_{\theta_0}(G=g)$.

Podemos escrever a verossimilhança como

$$\mathbb{P}_\theta(g_{obs}) = \frac{h(g_{obs})e^{\theta Z(g_{obs})}}{\kappa(\theta)} = \frac{H_\theta(g_{obs})}{\kappa(\theta)},$$

tal que $H_\theta(g_{obs}) = h(g_{obs})e^{\theta Z(g_{obs})}$.

Portanto, temos

$$t_\theta(g) = \frac{\nabla H_\theta(g)}{H_\theta(g)} = \frac{h(g)e^{\theta Z(g)}g(g)}{h(g)e^{\theta Z(g)}} = Z(g).$$

Quando m vai para infinito, o numerador da equação A.0.3 converge para 0

$$\lim_{m\to\infty}\mathbb{E}_{m,\theta_0}\left[(t_\theta(z_{obs}) - t_\theta(Z))\,\frac{H_\theta(Z)}{H_{\theta_0}(Z)}\right] = 0$$

e o denominador:

$$\lim_{m\to\infty}\mathbb{E}_{m,\theta_0}\left[\frac{H_\theta(Z)}{H_{\theta_0}(Z)}\right] = \frac{\kappa_{n,g}(\theta)}{\kappa_{n,g}(\theta_0)}.$$

Agora definimos $W_t = W_{\theta_0}(G_t) := (t_\theta(g_{obs}) - t_\theta(G_t))\,\frac{H_\theta(G_t)}{H_{\theta_0}(G_t)}$, onde G_t é a simulação MCMC de grafos valorados, utilizando Metropolis Hastings, esta

sequencia W_t converge para G sob a distribuição $\mathbb{P}_{\theta_0}$, e seja $W = \lim_{t\to\infty} W_t = \lim_{t\to\infty} W_{\theta_0}(G_t)$ Temos que $\mathbb{E}(W) = 0$, assim, temos o seguinte Teorema Central do Limite

$$\frac{1}{\sqrt{m}}\left(\sum_{i=1}^{m} W_i - \mathbb{E}(W)\right) \to \frac{\kappa(\theta_0)}{\kappa(\theta)} N(0, A),$$

Para cadeias reversíveis (Varadhan (1986)), a matriz A pode ser representada como

$$A = \left(\frac{\kappa(\theta_0)}{\kappa(\tilde{\theta})}\right)^2 \sum_{t=-\infty}^{\infty} \gamma_t, \tag{A.0.4}$$

onde $\gamma_t = \mathrm{Cov}(W_t, W_0)$.

Note que utilizando a expressão da distribuição $\mathbb{P}_\theta$ para os grafos valorados, podemos expressar W_t como

$$\begin{aligned} W_t &= (t_\theta(g_{obs}) - t_\theta(G_t))\,\frac{H_\theta(G_t)}{H_{\theta_0}(G_t)} \\ &= (Z(g_{obs}) - Z(G_t))\,\frac{h(G_t)e^{\theta Z(G_t)}}{h(G_t)e^{\theta_0 Z(G_t)}} \\ &= (Z(g_{obs}) - Z(G_t))\exp{(\theta - \theta_0)Z(G_t)}. \end{aligned}$$

Apêndice B

Código R

```
### CAPITULO 3: GRAFOS ALEATORIOS SIMPLES

my.ergm <- function(n,p,m,max.times){
  eta.true <- log(p/(1-p))
  x <- rbinom(n^2,1,p) #gera vetor aleatorio
  A <- matrix(x,nrow=n,ncol=n)
  A[lower.tri(A,diag=T)] = 0
  A = A + t(A) # Deixa a matriz simetrica
  diag(A) <- 0 # Fixa a diagonal princial em 0
  g <- graph.adjacency(A,mode="undirected")

  # Transformando o grafo g em um objeto network
  gnet <- asNetwork(g)
  z.obs <- network.edgecount(gnet) #Numero de arestas
  d <- z.obs/choose(n,2) # densidade do grafo
  eta0 <- NULL; eta0[1] <- log(d/(1-d)) # valor inicial
```

```
# Algoritmo 3.4 (Hunter 2006)
log.l <- NULL; # log-verossimilhanca
hats <- NULL # estimativas
K <- 10 # numero de lags utilizado
k <- 4   # Constante passo 4 do algoritmo

teste <- FALSE
i <- 1

while(teste==FALSE & i <= max.times){
  eta <- NULL; denominador <- NULL; f.inf <- NULL
  eta[1] <- eta0[1]
  w <- matrix(NA, ncol=max.times, nrow=m) # pesos das simulacoes
  w.f <- matrix(NA, ncol=max.times, nrow=m) # W_i
  z.sim <- rep(NA,m)
  for(cont in 1:m){
    sim <- simulate(g ~ edges, nsim=1, coef=eta0, basis=gnet)
    z.sim[cont] <- network.edgecount(sim)
  }
  t <- 1
  denominador[t] <- sum(exp((eta[t]-eta0)*(z.sim)))
  for(j in 1:m){
    w[j,t] <- exp((eta[t]-eta0)*z.sim[j])/denominador[t]
    w.f[j,t] <- (z.obs-z.sim[j])*(exp((eta[t]-eta0)*z.sim[j]))
  }
  f.inf[t] <- w[,t]%*%(z.sim^2) -
      (w[,t]\%*\%z.sim)*(w[,t]%*%z.sim)
```

```
eta[t+1] <- eta[t] + (1/f.inf[t])%*%
    (z.obs-(w[,t]%*%z.sim))
t <- t+1
while(abs(eta[t]-eta[t-1])>=0.0001 & t<=max.times){
  denominador[t] <- sum(exp((eta[t]-eta0)*(z.sim)))
  for(j in 1:m){
    w[j,t] <- exp((eta[t]-eta0)*z.sim[j])/denominador[t]
    w.f[j,t] <- (z.obs-z.sim[j])*(exp((eta[t]-eta0)*z.sim[j]))
  }
  f.inf[t] <- w[,t]%*%(z.sim^2) -
    (w[,t]%*%z.sim)*(w[,t]%*%z.sim)
  eta[t+1] <- eta[t] + (1/f.inf[t])%*%
    (z.obs-(w[,t]%*%z.sim))
  t <- t+1
}
hats[i] <- eta[t]
# Calculando a log-verossimilhanca estimada em eta:
r_function <- function(eta){
  dif_exp <- exp((eta-eta0)*z.sim)
  r <- (eta-eta0)*z.obs-log(1/m*sum(dif_exp))
  return(as.double(r))
}
l_function <-function(eta){
  return(r_function(eta)-r_function(0)-
    (choose(n,2)*log(2)))
}
log.l[i] <- l_function(hats[i])
```

```
    # Calculando var(rm_hat):
    u <- exp((hats[i]-eta0)*z.sim)
    u.barra <- mean(u)
    gama <- c(acf(u,plot=FALSE,type="covariance",
                  lag.max=K)$acf)
    k.sum0 <- m*gama[1] + sum((m-1:10)*(gama[2:(K+1)]))
    var.mc <- (1/m^2)*(1/u.barra^2)*k.sum0
    teste <- sqrt(var.mc) < k*abs(log.l[i])
    eta0[i+1] <- hats[i]
    i <- i+1
  }
  eta_til <- hats[length(hats)]
  eta0 <- log(d/(1-d)) # valor inicial para estimativa
  # Calculando o erro padrao estimado:
  soma0 <- sum(exp(eta0-eta_til)*z.sim)^2
  epsilon <- c(acf(w.f[,1],plot=FALSE,type="covariance",
               lag.max=K)$acf)
  soma1 <- epsilon[1] + sum(2*(epsilon[2:(K+1)]))
  V <- 1/(m^2)*soma0*soma1

  var.mcmc <- 1/m*V*(1/f.inf[1])^2
  se.mcmc <- sqrt(var.mcmc)
  se.mcmc
  return(list(result=c(n,p,eta.true,m,max.times,
              eta_til,se.mcmc,eta0),g=g))
}
```

```
### CAPITULO 4: GRAFOS ALEATORIOS VALORADOS
# Funcao gera poisson
r.pois <- function(y){
  x <- y
  while(x==y){x <- rpois(1,y+0.5)}
  return(x)
}

# Funcao p
p.function <- function(a,b){
  x <- ((exp(-(b+0.5))*((b+0.5)^a)/factorial(a)))/
       ((1-exp(-(b+0.5))*((b+0.5)^b)/factorial(b)))
  return(x)
}

# Funcao g
g.function <- function(y){
  g1 <- sum(y)/2
  g2 <- 0
  for(i in 1:(dim(y)[1]-1)){
    for(j in (i+1):dim(y)[1]){
      for(k in 1:dim(y)[1]){
        g2 <- g2 + y[i,k]*y[k,j]
      }
    }
  }
  return(c(g1,g2))
```

```
}

#Funcao delta
delta.function <- function(y1,y2){
  dif <- g.function(y2)-g.function(y1)
  return(dif)
}

# Simula grafo aleatorio valorado uniparametrico
simulate.vergm <- function(n,m,T,pi0,eta1,eta2){
  x <- NULL
  for(i in 1:(n^2)){x[i] <- rpois(1,m)}
  A <- matrix(x,nrow=n,ncol=n)
  A[lower.tri(A,diag=T)]=0
  A=A+t(A)
  diag(A) <- 0

  # Implementacao do algoritmo
  N <- 1:n # Conjunto de vertices
  eta <- c(eta1,eta2); delta <- NULL; y.list <- NULL;
  y.list[[1]] <- A; q <- NULL; r <- NULL
  for(i in 1:T){
  ij <- sample(N,2,replace=FALSE)
  y.star <- ifelse(y.list[[i]][ij[1],ij[2]] != 0 &
  runif(1)<pi0,0,r.pois(y=y.list[[i]][ij[1],ij[2]]))

  # razao de probabilidades de transi   o
```

```
  q[i] <- ifelse(y.list[[i]][ij[1],ij[2]]==0,(pi0+(1-pi0)*
          p.function(0,y.star))/(p.function(y.star,0+0.01)),
          ifelse(y.list[[i]][ij[1],ij[2]] != 0 & y.star==0,
          p.function(y.list[[i]][ij[1],ij[2]],0)/(pi0+(1-pi0)*
          p.function(0,y.list[[i]][ij[1],ij[2]])),
          ((1-pi0)*p.function(y.list[[i]][ij[1],ij[2]],y.star)/
          ((1-pi0)*p.function(y.star,y.list[[i]][ij[1],ij[2]]))))

    y2 <- y.list[[i]]
    y2[ij[1],ij[2]] <- y2[ij[2],ij[1]] <- y.star
    delta <- delta.function(y.list[[i]],y2)

    # taxa de aceitacao do MH
    r[i] <- q[i]*(factorial(y.list[[i]][ij[1],ij[2]])/
        factorial(y.star))*(exp(t(eta)%*%delta))
    if(r[i]>runif(1)){
      y.list[[i+1]] <- y2
    } else { y.list[[i+1]] <- y.list[[i]] }
  }
  return(y.list[[T+1]])
}

# Simula grafo aleatorio valorado biparametrico
simulate.vergm2 <- function(n,m,T,pi0,eta1,eta2){
  x <- NULL
  for(i in 1:(n^2)){x[i] <- rpois(1,m)}
  A <- matrix(x,nrow=n,ncol=n)
```

```
A[lower.tri(A,diag=T)]=0
A=A+t(A)
diag(A) <- 0

N <- 1:n #Conjunto de v rtices
eta <- c(eta1,eta2); delta <- NULL; y.list <- NULL;
y.list[[1]] <- A; q <- NULL; r <- list()
for(i in 1:T){
   ij <- sample(N,2,replace=FALSE)
   y.star <- ifelse(y.list[[i]][ij[1],ij[2]] != 0 &
   runif(1)<pi0,0,r.pois(y=y.list[[i]][ij[1],ij[2]]))

   # razao de probabilidades de transi   o
   q[i] <- ifelse(y.list[[i]][ij[1],ij[2]]==0,(pi0+(1-pi0)*
           p.function(0,y.star))/(p.function(y.star,0)),
           ifelse(y.list[[i]][ij[1],ij[2]] != 0 & y.star==0,
           p.function(y.list[[i]][ij[1],ij[2]],0)/
           (pi0+(1-pi0)*p.function(0,y.list[[i]][ij[1],
           ij[2]])),
           ifelse(((1-pi0)*p.function(y.star,y.list[[i]]
           [ij[1],ij[2]]))==Inf,0,
           ((1-pi0)*p.function(y.list[[i]][ij[1],ij[2]],
           y.star))/((1-pi0)*p.function(y.star,
           y.list[[i]][ij[1],ij[2]])))))

   y2 <- y.list[[i]]
   y2[ij[1],ij[2]] <- y2[ij[2],ij[1]] <- y.star
```

```
    delta <- delta.function(y.list[[i]],y2)

    # taxa de aceitacao do MH (fixar menor que 1)
    r[[i]] <- asNumeric(q[i][[1]])*
              (factorialZ(y.list[[i]][ij[1],ij[2]])/
              factorial(y.star))*(exp(t(eta)%*%delta))
    r[[i]] <- ifelse(is.na(r[[i]])==TRUE,0,asNumeric(r[[i]]))

    if(r[[i]]>runif(1)){
      y.list[[i+1]] <- y2
    } else { y.list[[i+1]] <- y.list[[i]] }

  }
  return(y.list[[T+1]])
}

# Estimacao modelo Uniparametrico
my.vergm <- function(m.times,g1,max.times,K,k){
  n <- dim(g1)[1] # N mero de vertices
  eta0 <- log(sum(g1)/(2*choose(n,2)))
  z.obs <- g.function(g1)[[1]]
  z.sim <- matrix(NA,ncol=1,nrow=m.times)
  h=NULL; eta <- eta0; w <- list(); hats <- NULL

  i=1;teste=FALSE
  while(teste==FALSE & i<max.times){
    for(c0 in 1:m.times){
```

```
  print(c0)
  sim <- simulate.vergm(n=n,m=4,pi0=0,eta1=eta0[1],T=10000)
  z.sim[c0] <- c(sum(sim)/2)
  h[[c0]] <- prod(factorialZ(sim[lower.tri
             (sim,diag=F)])^(-1))
}
t <- 1 # t: conta interacoes
# Estima os pesos w's das simulacoes (equacao 3.4)
soma <- 0; w.f <- rep(NA,m.times)
for(count in 1:m.times){
  w[[count]] <- exp(mpfr((eta[t]-eta0)*
                z.sim[count],128))*h[[c0]]
  soma <- soma+w[[count]]
  w.f[count] <- t(z.obs-z.sim[count,])%*%
                (exp(t(t((eta[[t]])-eta0))%*%
                z.sim[count,]))
}
for(j in 1:m.times){w[[j]] <- w[[j]]/soma}
# Calculo da informacao de fisher
soma2 <- soma3 <- mpfr(0,128)
for(c1 in 1:m.times){
  soma2 <- soma2+w[[c1]]*(z.sim[c1]^2)
  soma3 <- soma3+w[[c1]]*(z.sim[c1])
}
f.inf <- asNumeric(soma2-soma3^2)
# Atualiza o valor de eta estimado
eta[t+1] <- asNumeric(eta[t][[1]]) +
```

```
                f.inf^(-1)%*%(z.obs-asNumeric(soma3))
# Repete os passos para as demais simulacoes
t <- t+1
while(abs(eta[t]-eta[t-1])>=0.001 & t<=max.times){
  soma <- 0
  for(count in 1:m.times){
    w[[count]] <- exp(mpfr((eta[t]-eta0)*
    z.sim[count],128))*h[[c0]]
    soma <- soma+w[[count]]
    w.f[count] <- t(z.obs-z.sim[count,])%*%
    (exp(t(t((eta[[t]])-eta0))%*%z.sim[count,]))
  }
  for(j in 1:m.times){w[[j]] <- w[[j]]/soma}
  soma2 <- soma3 <- mpfr(0,128)
  for(c1 in 1:m.times){
    soma2 <- soma2+w[[c1]]*(z.sim[c1]^2)
    soma3 <- soma3+w[[c1]]*(z.sim[c1])
  }
  f.inf <- asNumeric(soma2-soma3^2)
  eta[t+1] <- asNumeric(eta[t][[1]]) + f.inf^(-1)%*%
  (z.obs-asNumeric(soma3))
  t <- t+1
}
hats[i] <- eta[t]
# Calculando a log-verossimilhanca estimada em eta:
r_function <- function(eta){
  dif_exp <- ifelse(exp((eta-eta0)%*%t(z.sim))==0,0.001,
```

```
                exp((eta-eta0)%*%t(z.sim)))
    soma <- 0
    for(i in 1:m.times){soma <- soma+h[[i]]*dif_exp[i]}
    r <- (eta-eta0)%*%z.obs-log(mpfr(1/m.times*soma))
    return(as.numeric(r))
  }
  h.obs <- prod(factorialZ(g1[lower.tri
  (g1,diag=F)])^(-1))*exp(-1)^(choose(n,2))
  l_function <-function(eta){
    return(asNumeric(r_function(eta))-
    asNumeric(r_function(0))-log(asNumeric(h.obs)))
  }
  log.l <- l_function(hats[i]) # Verossimilhanca
  # Calculando var(rm_hat):
  u <- exp((hats[i]-eta0)*z.sim)
  u.barra <- mean(u)
  gama <- c(acf(u,plot=FALSE,type="covariance",
  lag.max=K)$acf)
  k.sum0 <- m.times*gama[1] + sum((m.times-1:10)*
  (gama[2:(K+1)]))
  var.mc <- (1/m.times^2)*(1/u.barra^2)*k.sum0
  teste <- sqrt(var.mc) < k*abs(log.l)
  eta0 <- ifelse(teste==TRUE,eta0,hats[i])
}
# Calculando o erro padrao estimado:
epsilon11 <- c(acf(w.f,plot=FALSE,type="covariance",
          lag.max=K)$acf)
```

```
  soma11 <- epsilon11[1] + sum(2*(epsilon11[2:(K+1)]))
  denominador <- sum(exp(t(hats[i]-eta0)\%*\%t(z.sim)))
  V <- 1/(m.times^2)*(denominador^2)*soma11
  var.mcmc <- 1/m.times*(solve(f.inf))%*%
              V%*%(solve(f.inf))
  return(list(eta=eta, var.mcmc=var.mcmc,
              teste=teste, hats=hats, i=i))
}

# simula um grafo e estima os parametros simulados
simest.vergm <- function(n,m,pi0,eta1,T,m.times,max.times,K,k){
  g1 <- simulate.vergm(n=n,m=m,pi0=0,eta1=eta1,T=10000)
  mod <- my.vergm(m.times=m.times,g1=g1,
              max.times=max.times,K=K,k=k)
  return(mod)
}

# Estimacao modelo Biparametrico
my.vergm2 <- function(m.times,g1,max.times,K,k,eta0){
  z.obs <- g.function(g1)
  n <- dim(g1)[1]
  z.sim <- matrix(NA,ncol=2,nrow=m.times)
  h=NULL; eta <- list(); eta[[1]] <- eta0;
  w <- list(); hats <- matrix(NA,nrow=100,ncol=2)

  i=1;teste=FALSE
  while(teste==FALSE & i<100){
```

```
for(c0 in 1:m.times){
  print(c0)
  sim <- simulate.vergm(n=n,m=4,pi0=0,eta1=eta0[1],
  eta2=eta0[2],T=10000)
  z.sim[c0,] <- g.function(sim)
  h[[c0]] <- prod(factorialZ(sim[lower.tri
             (sim,diag=F)])^(-1))
}
t <- 1 # t: conta interacoes
# Estima os pesos w's das simulacoes (equacao 3.4)
soma <- 0
for(count in 1:m.times){
  w[[count]] <- exp(mpfr(t(eta[[t]]-eta0)%*%
                z.sim[count,],128))
  soma <- soma+w[[count]]
}
for(j in 1:m.times){w[[j]] <- w[[j]]/soma}
# Calculo da informacao de fisher
soma2 <- matrix(0,ncol=2,nrow=2);
soma3 <- matrix(0,ncol=2,nrow=1)
for(c1 in 1:m.times){
  soma2 <- soma2+w[[c1]][1,1]*(z.sim[c1,]
           %*%t(z.sim[c1,]))
  soma3 <- soma3+w[[c1]][1,1]*(z.sim[c1,])
}
soma3 <- asNumeric(soma3)
f.inf <- asNumeric(soma2-t(soma3)%*%soma3)
```

```
# Atualiza o valor de eta estimado
eta[[t+1]] <- asNumeric(eta[[t]]) + solve(f.inf)%*%
                    t(z.obs-asNumeric(soma3))
# Repete os passos para as demais simulacoes
t <- t+1
while(sum(abs(eta[[t]]-eta[[t-1]])>=0.0001) & t<=max.times){
  soma <- 0
  for(count in 1:m.times){
    w[[count]] <- exp(mpfr(t(eta[[t]]-eta0)%*%
                    z.sim[count,],128))*h[[c0]]
    soma <- soma+w[[count]]
  }
  for(j in 1:m.times){w[[j]] <- w[[j]]/soma}
  # Calculo da informacao de fisher
  soma2 <- matrix(0,ncol=2,nrow=2);
  soma3 <- matrix(0,ncol=2,nrow=1)
  for(c1 in 1:m.times){
    soma2 <- soma2+w[[c1]][1,1]*(z.sim[c1,]%*%
                    t(z.sim[c1,]))
    soma3 <- soma3+w[[c1]][1,1]*(z.sim[c1,])
  }
  soma3 <- asNumeric(soma3)
  f.inf <- asNumeric(soma2-t(soma3)%*%soma3)
  # Atualiza o valor de eta estimado
  eta[[t+1]] <- asNumeric(eta[[t]]) + solve(f.inf)%*%
  t(z.obs-asNumeric(soma3))
  t <- t+1
```

```
}
hats[i,] <- eta[[t]]
# Calculando a log-verossimilhanca estimada em eta:
r_function <- function(eta){
  dif_exp <- ifelse(exp((eta-eta0)%*%t(z.sim))==0,0.001,
                    exp((eta-eta0)%*%t(z.sim)))
  soma <- 0
  for(i in 1:m.times){soma <- soma+h[[i]]*dif_exp[i]}
  r <- (eta-eta0)%*%z.obs-log(mpfr(1/m.times*soma))
  return(as.numeric(r))
}
h.obs <- prod(factorialZ(g1[lower.tri(g1,diag=F)])
              ^(-1))*exp(-1)^(choose(n,2))
l_function <-function(eta){
  return(r_function(eta)-r_function(c(0,eta0))-
         log(mpfr(h.obs)))
}
log.l <- l_function(hats[i,]) # Verossimilhanca
# Calculando var(rm_hat):
u <- exp(t(hats[i,]-eta0)%*%t(z.sim))
u.barra <- mean(u)
gama <- c(acf(u,plot=FALSE,type="covariance",
              lag.max=K)$acf)
k.sum0 <- m.times*gama[1] + sum((m.times-1:10)*
              (gama[2:(K+1)]))
var.mc <- (1/m.times^2)*(1/u.barra^2)*k.sum0
teste <- sqrt(var.mc) < k*abs(log.l)
```

```
    eta0 <- hats[i,]
  }
  # Calculando o erro padrao estimado:
  w.f <- matrix(NA, ncol=2,nrow=m.times) # W_i
  for(count in 1:m.times){
    w.f[count,] <- t(z.obs-z.sim[count,])%*%
          (exp(t(t((eta[[t]])-eta0))%*%z.sim[count,]))
  }
  epsilon11 <- c(acf(w.f[,1], plot=FALSE,
                     type="covariance",lag.max=10)$acf)
  epsilon12 <- c(c(ccf(w.f[,1],w.f[,2], plot=FALSE,
                     type="covariance",lag.max=10))$acf)
  epsilon22 <- c(acf(w.f[,2], plot=FALSE,
                     type="covariance",lag.max=10)$acf)

  soma11 <- epsilon11[1] + sum(2*(epsilon11[2:(K+1)]))
  soma22 <- epsilon22[1] + sum(2*(epsilon22[2:(K+1)]))
  soma12 <- sum(epsilon12)

  soma1 <- cbind(c(soma11,soma12),c(soma12,soma22))
  denominador <- sum(exp(t(eta[[t]]-eta0)%*%t(z.sim)))
  V <- 1/(m.times^2)*(denominador^2)*soma1
  var.mcmc <- 1/m.times*(solve(f.inf))%*%
                    V%*%(solve(f.inf))
  return(list(eta=eta, var.mcmc=var.mcmc,
              teste=teste, hats=hats, i=i))
}
```

```
# simula um grafo e estima os parametros simulados
simest.vergm2 <-
   function(n,m, pi0 , eta1 , eta2 ,T,m.times ,max.times ,K, k){
   g1 <- simulate.vergm(n=n,m=m, pi0=0,eta1=eta1 ,
                eta2=eta2 ,T=10000)
   #eta0 <- c(eta1 , eta2)
   eta0 <- c(log(sum(g1)/(2*choose(n,2))) ,0)
   mod <- my.vergm(m.times=m.times , g1=g1 ,
     max.times=max.times ,K=K, k=k , eta0=eta0)
   return(mod)
}
```