自回归动态网络模型

胡洁琼*, 王一博†, 吴征‡, 张林逍§

(中国科学技术大学 管理学院, 安徽 合肥 230026)

摘要: 动态网络是指网络结构随时间变化的网络系统, 常用于分析复杂系统中的演变过程。本文是对现有文献提出的自回归动态网络模型的学习与复现, 在本文的模型下, 允许各边的转移概率依赖于其他边的历史时刻, 但在取定历史时刻时是条件独立的, 同时, 本模型的极大似然估计具有良好的渐进性质。本文通过模拟实验验证了参数估计的有效性, 并将模型应用于真实数据集进行数据分析, 结果表明自回归动态网络模型在实际问题中具有较好的应用效果。

关键词: 时间序列; 网络数据; 动态网络; 自回归

1 引言

传统的时间序列分析主要考虑对一维数据的建模分析, 但随着大数据时代的到来, 越来越多可以被连接成网络的数据被人们获取并保存下来, 例如电子邮件的记录、社交网络等等, 这些网络通常随着时间发生变化, 学会分析这类网络数据可以帮助我们在杂乱的连接关系中找到所需要的信息。

动态网络 (Dynamic Network) 是指网络结构随时间变化的网络系统。与静态网络不同, 动态网络不仅考虑了网络结构, 还考虑了网络随时间变化的情况, 它常用于分析复杂系统中的演变过程, 可以帮助我们理解时间变化对系统整体行为的影响。如今, 动态网络的应用越来越广泛, 例如监控互联网流量的异常、预测电力供应网络的需求及设定价格、了解新闻和观点在社交网络的传播机制等。因此, 理解动态网络随时间变化的规律, 并建立相应的网络模型, 对现代复杂问题的处理具有重要的意义。

本文是对 Chang et al. (2024) 和 Jiang et al. (2023) 提出的自回归动态网络模型 (Autoregressive Networks) 的学习与复现, 介绍了一种基于 Markov 链的动态网络模型。现有的动态网络模型主要

作者按姓名首字母排序

*PB22151759: 真实数据分析, 第 4 节撰写

†PB22051070: 任务分配, 模拟实验, 辅助真实数据分析, 其余部分撰写

‡PB22020515: 真实数据预处理, 辅助模拟实验

§PB22151770: 真实数据收集, 辅助论文撰写

考虑网络中各边相互独立的情形, 而在本文的模型下, 我们允许各边的转移概率依赖于其他边的历史时刻, 这个使得本模型具有充足的灵活性和普适性, 以更好地适应不同类型的网络。为便于进行参数估计, 我们假设各边在取定历史时刻时是条件独立的。在条件独立性的基础上, 本文证明了本模型的极大似然估计具有良好的渐进性质。

本文其余内容将按照如下结构展开: 在第 2 节, 本文将介绍一般的自回归动态网络模型, 并给出两种特殊模型, 同时说明其参数估计问题; 在第 3 节, 本文将通过模拟实验, 验证参数估计的有效性; 在第 4 节, 本文将利用真实数据集, 讨论自回归动态网络模型在实际问题中的适用性与应用效果; 在第 5 节, 本文将进行总结讨论; 在附录, 本文将给出模拟实验与真实数据集的所有代码。

2 自回归网络模型

2.1 AR(m) 网络

考虑一个定义在 p 个节点上的动态网络过程, 其节点集合为 $V = \{1, \dots, p\}$, 定义 $\mathbf{X}_t = (X_{i,j}^t)_{p\times p}$ 表示其位于 t 时刻的邻接矩阵, 其中, 若 $X_{i,j}^t = 1$, 则表示在 t 时刻有一条连接节点 i 与节点 j 的边, 若 $X_{i,j}^t = 0$, 则表示无边连接节点 i 与节点 j。为了处理方便, 本文只考虑无向边的情形, 且不允许节点的自相连, 即 $X_{i,i}^t \equiv 0$, $X_{i,j}^t = X_{j,i}^t$, $\forall i, j \in V$, $\forall t \geqslant 1$, 但本文的结论可以较为容易地推广至有向边的情形。

定义 1 (AR(m) 网络) *在给定 $\{\mathbf{X}_s\}_{s\leqslant t-1}$ 的条件下, 各边 $\{X_{i,j}^t\}_{1\leqslant i<j\leqslant p}$ 相互独立, 且*

$$\begin{aligned}
\alpha_{i,j}^{t-1} &\equiv \mathbb{P}(X_{i,j}^t = 1 \,|\, X_{i,j}^{t-1} = 0, \mathbf{X}_{t-1}\backslash X_{i,j}^{t-1}, \mathbf{X}_{t-k} \text{ for } k \geqslant 2) \\
&= \mathbb{P}(X_{i,j}^t = 1 \,|\, X_{i,j}^{t-1} = 0, \mathbf{X}_{t-1}\backslash X_{i,j}^{t-1}, \mathbf{X}_{t-2}, \dots, \mathbf{X}_{t-m}), \qquad (2.1) \\
\beta_{i,j}^{t-1} &\equiv \mathbb{P}(X_{i,j}^t = 0 \,|\, X_{i,j}^{t-1} = 1, \mathbf{X}_{t-1}\backslash X_{i,j}^{t-1}, \mathbf{X}_{t-k} \text{ for } k \geqslant 2) \\
&= \mathbb{P}(X_{i,j}^t = 0 \,|\, X_{i,j}^{t-1} = 1, \mathbf{X}_{t-1}\backslash X_{i,j}^{t-1}, \mathbf{X}_{t-2}, \dots, \mathbf{X}_{t-m}), \qquad (2.2)
\end{aligned}$$

其中 m 为一固定正整数, 记号 $\mathbf{X}_t\backslash X_{i,j}^t$ 表示差集 $\mathbf{X}_t\backslash\{X_{i,j}^t\}$。

不难看出, 如上定义的 AR(m) 网络模型是一个 m 阶 Markov 链。根据 (2.1) 和 (2.2), 可以得出

$$\mathbb{P}(X_{i,j}^t = 1 \mid \mathbf{X}_{t-1}, \dots, \mathbf{X}_{t-m}) = \alpha_{i,j}^{t-1} + X_{i,j}^{t-1}(1 - \alpha_{i,j}^{t-1} - \beta_{i,j}^{t-1}) \equiv \gamma_{i,j}^{t-1}, \qquad (2.3)$$

即

$$X_{i,j}^t \mid \mathbf{X}_{t-1}, \dots, \mathbf{X}_{t-m} \sim \text{Bernoulli}(\gamma_{i,j}^{t-1}), \quad \forall 1 \leqslant i < j \leqslant p。 \qquad (2.4)$$

需要注意的是, 在不给定条件时, 各边 $\{X_{i,j}^t\}_{1\leqslant i<j\leqslant p}$ 不一定相互独立。

为了适应不同类型的网络, 需要对转移概率 $\alpha_{i,j}^{t-1}$ 和 $\beta_{i,j}^{t-1}$ 施加不同的形式, 即对此进行参数化处理

$$\alpha_{i,j}^{t-1}(\boldsymbol{\theta}) = f_{i,j}(\mathbf{X}\backslash X_{i,j}^{t-1}, \mathbf{X}_{t-2}, \ldots, \mathbf{X}_{t-k}; \boldsymbol{\theta}), \tag{2.5}$$

$$\beta_{i,j}^{t-1}(\boldsymbol{\theta}) = g_{i,j}(\mathbf{X}\backslash X_{i,j}^{t-1}, \mathbf{X}_{t-2}, \ldots, \mathbf{X}_{t-k}; \boldsymbol{\theta}), \tag{2.6}$$

其中 $f_{i,j}$ 和 $g_{i,j}$ 为任意非负函数, $\boldsymbol{\theta} \in \boldsymbol{\Theta} \subset \mathbb{R}^q$ 为参数向量。对真实值, 令

$$\alpha_{i,j}^{t-1} = f_{i,j}(\mathbf{X}\backslash X_{i,j}^{t-1}, \mathbf{X}_{t-2}, \ldots, \mathbf{X}_{t-k}; \boldsymbol{\theta}_0) = \alpha_{i,j}^{t-1}(\boldsymbol{\theta}_0), \tag{2.7}$$

$$\beta_{i,j}^{t-1} = g_{i,j}(\mathbf{X}\backslash X_{i,j}^{t-1}, \mathbf{X}_{t-2}, \ldots, \mathbf{X}_{t-k}; \boldsymbol{\theta}_0) = \beta_{i,j}^{t-1}(\boldsymbol{\theta}_0), \tag{2.8}$$

即 $\boldsymbol{\theta}_0$ 为真实参数。特别地, 可以得到以下两种 AR(1) 网络模型, 后文也将主要围绕这两种模型展开讨论。

定义 2 (标准 AR(1) 网络) 若 AR 网络的转移概率满足

$$\alpha_{i,j}^{t-1}(\boldsymbol{\theta}) \equiv \alpha_{i,j}, \quad \beta_{i,j}^{t-1}(\boldsymbol{\theta}) \equiv \beta_{i,j}, \tag{2.9}$$

其中 $\boldsymbol{\theta} = \{\alpha_{i,j}, \beta_{i,j} \mid \alpha_{i,j}, \beta_{i,j} > 0,\ \alpha_{i,j} + \beta_{i,j} \leqslant 1,\ \forall 0 \leqslant i < j \leqslant p\}$, 则称该网络为标准 AR(1) 网络。

注 1 此时每条边的转移概率不依赖于其他边的历史时刻, 即

$$\alpha_{i,j}^{t-1} = \mathbb{P}(X_{i,j}^t = 1 \,|\, X_{i,j}^{t-1} = 0) \equiv \alpha_{i,j}, \quad \beta_{i,j}^{t-1} = \mathbb{P}(X_{i,j}^t = 0 \,|\, X_{i,j}^{t-1} = 1) \equiv \beta_{i,j}。 \tag{2.10}$$

更进一步, 若初值

$$\mathbb{P}(X_{i,j}^1 = 1) = \frac{\alpha_{i,j}}{\alpha_{i,j} + \beta_{i,j}} = 1 - \mathbb{P}(X_{i,j}^1 = 0), \tag{2.11}$$

则有

$$\mathbb{P}(X_{i,j}^t = 1) = \frac{\alpha_{i,j}}{\alpha_{i,j} + \beta_{i,j}} = 1 - \mathbb{P}(X_{i,j}^t = 0),\ \forall t \geqslant 1, \tag{2.12}$$

此时为严平稳序列。

注 2 不难发现, 若标准 AR(1) 网络严平稳, 则

$$\mathbb{E}(X_{i,j}^t) = \frac{\alpha_{i,j}}{\alpha_{i,j} + \beta_{i,j}}, \quad \mathrm{Var}(X_{i,j}^t) = \frac{\alpha_{i,j}\beta_{i,j}}{(\alpha_{i,j} + \beta_{i,j})^2}, \tag{2.13}$$

$$\mathrm{Corr}(X_{i,j}^t, X_{l,m}^s) = \begin{cases} (1 - \alpha_{i,j} - \beta_{i,j})^{|t-s|}, & 若 (i,j) = (l,m), \\ 0, & 其他, \end{cases} \tag{2.14}$$

故也是宽平稳的。

定义 3 (传递 AR(1) 网络) 若 AR 网络的转移概率满足

$$\alpha_{i,j}^{t-1}(\boldsymbol{\theta}) = \frac{\xi_i \xi_j \exp(aU_{i,j}^{t-1})}{1+\exp(aU_{i,j}^{t-1})+\exp(bV_{i,j}^{t-1})}, \tag{2.15}$$

$$\beta_{i,j}^{t-1}(\boldsymbol{\theta}) = \frac{\eta_i \eta_j \exp(bV_{i,j}^{t-1})}{1+\exp(aU_{i,j}^{t-1})+\exp(bV_{i,j}^{t-1})}, \tag{2.16}$$

其中, $\boldsymbol{\theta} = (a, b, \xi_1, \ldots, \xi_p, \eta_1, \ldots, \eta_p)^\top \in \mathbb{R}_+^{2p+1}$,

$$U_{i,j}^{t-1} = \frac{1}{p-2} \sum_{k:k\neq i,j} X_{i,k}^{t-1} X_{j,k}^{t-1}, \tag{2.17}$$

$$V_{i,j}^{t-1} = \frac{1}{p-2} \sum_{k:k\neq i,j} \{X_{i,k}^{t-1}(1-X_{j,k}^{t-1}) + (1-X_{i,k}^{t-1})X_{j,k}^{t-1}\}, \tag{2.18}$$

则称该网络为传递 AR(1) 网络。

注 3 变量 $U_{i,j}^{t-1}$ 表示 $t-1$ 时刻同时连接节点 i 与节点 j 的节点的密度, 变量 $V_{i,j}^{t-1}$ 表示 $t-1$ 时刻只连接节点 i 与节点 j 其中之一的节点的密度。从表达式中可以看出, 若 $U_{i,j}^{t-1}$ 越大, 则 $X_{i,j}^{t}=1$ 的概率越大, 若 $V_{i,j}^{t-1}$ 越大, 则 $X_{i,j}^{t}=0$ 的概率越大。故上述模型体现了网络节点的传递性特征, 适合对朋友关系网络、贸易关系网络等类型的网络进行建模。以朋友关系网络为例, 若两个人的共同朋友越多, 则他们越倾向于成为朋友。

2.2 参数估计

以下将不加证明地罗列一些重要的定理, 具体细节请见 Chang et al. (2024) 和 Jiang et al. (2023), 这些定理对模型的参数估计具有重要意义。利用前述转移概率的参数表达, 将式 (2.3) 改写为参数形式

$$\gamma_{i,j}^{t-1}(\boldsymbol{\theta}) = \alpha_{i,j}^{t-1}(\boldsymbol{\theta}) + X_{i,j}^{t-1}\{1-\alpha_{i,j}^{t-1}(\boldsymbol{\theta}) - \beta_{i,j}^{t-1}(\boldsymbol{\theta})\}, \tag{2.19}$$

则有 $\gamma_{i,j}^{t-1} = \gamma_{i,j}^{t-1}(\boldsymbol{\theta}_0)$。首先将参数进行分类, 定义全局参数和局部参数。

定义 4 参数 $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_q)^\top$, 其中正整数 $q \geqslant 1$ 代表参数的总数。定义

$$\mathcal{G} = \{l \in [q] \mid \gamma_{i,j}^{t-1}(\boldsymbol{\theta}) \text{ 的表达式包含参数 } \theta_l,\ \forall 1 \leqslant i < j \leqslant p,\ \forall t \in [n]\backslash[m]\}, \tag{2.20}$$

其中记号 $[n]$ 表示集合 $\{1, \ldots, n\}$。称 $\mathcal{G}$ 为全局参数下标向量, 对应地, 称 $\mathcal{G}^{\mathrm{c}}$ 为局部参数下标向量。

由上述定义, 在传递 AR(1) 网络模型中, 参数 a 和 b 为全局参数, 而 ξ_i 和 η_i 为局部参数; 在标准 AR(1) 网络模型中, 所有参数均为局部参数。

令 $\mathcal{F}_t$ 表示 $\{\mathbf{X}_1, \ldots, \mathbf{X}_t\}$ 生成的 σ-域。对任一 $l \in [q]$, 定义

$$\mathcal{S}_l = \{(i,j) \mid 1 \leqslant i < j \leqslant p, \text{ 且}\gamma_{i,j}^{t-1}(\boldsymbol{\theta}) \text{ 的表达式包含参数 } \theta_l, \ \forall t \in [n]\backslash[m]\}。 \tag{2.21}$$

不难注意到, 若 θ_l 是全局参数, 则 $\mathcal{S}_l = \{(i,j) \mid 1 \leqslant i < j \leqslant p\}$。由上述定义, 在估计参数 θ_l 时, 只需考虑 $\mathcal{S}_l$ 中的边, 因为只有这些边写出的表达式才包含该参数。考虑

$$\ell_{n,p}^{(l)}(\boldsymbol{\theta}) = \frac{1}{(n-m)|\mathcal{S}_l|} \sum_{t=m+1}^{n} \sum_{(i,j)\in\mathcal{S}_l} \mathbb{E}_{\mathcal{F}_{t-1}}\{\log[\{\gamma_{i,j}^{t-1}(\boldsymbol{\theta})\}^{X_{i,j}^t}\{1-\gamma_{i,j}^{t-1}(\boldsymbol{\theta})\}^{1-X_{i,j}^t}]\}, \tag{2.22}$$

其中 $\mathbb{E}_{\mathcal{F}_{t-1}}(\cdot)$ 代表给定 $\mathcal{F}_{t-1}$ 时在真实参数 θ_0 下的条件期望, 记号 $|\cdot|$ 表示集合的大小。可以证明,

$$\ell_{n,p}^{(l)}(\boldsymbol{\theta}) \leqslant \ell_{n,p}^{(l)}(\boldsymbol{\theta}_0), \quad \forall \boldsymbol{\theta} \in \boldsymbol{\Theta}, \tag{2.23}$$

且对任意的 $\boldsymbol{\theta} \in \boldsymbol{\Theta}\backslash\{\boldsymbol{\theta}_0\}$,

$$\ell_{n,p}^{(l)}(\boldsymbol{\theta}) = \ell_{n,p}^{(l)}(\boldsymbol{\theta}_0) \Leftrightarrow \gamma_{i,j}^{t-1}(\boldsymbol{\theta}) = \gamma_{i,j}^{t-1}(\boldsymbol{\theta}_0), \ \forall t \in [n]\backslash[m], \ \forall (i,j) \in \mathcal{S}_l。 \tag{2.24}$$

定义

$$\mathcal{I}_{i,j} = \{l \in [q] \mid \gamma_{i,j}^{t-1}(\boldsymbol{\theta}) \text{ 的表达式包含参数 } \theta_l, \ \forall t \in [n]\backslash[m]\}。 \tag{2.25}$$

为了保证参数的可识别性, 我们还需要如下命题。

命题 1 *在满足某些条件时, 当 $n \to \infty$, 以概率为 1 地, 有*

$$\ell_{n,p}^{(l)}(\boldsymbol{\theta}_0) - \ell_{n,p}^{(l)}(\boldsymbol{\theta}) \geqslant \frac{C}{|\mathcal{S}_l|} \sum_{(i,j)\in\mathcal{S}_l} \|\boldsymbol{\theta}_{\mathcal{I}_{i,j}} - \boldsymbol{\theta}_{0,\mathcal{I}_{i,j}}\|_2^2, \tag{2.26}$$

$\forall \boldsymbol{\theta} \in \boldsymbol{\Theta}$, $\forall l \in [q]$, 其中 $C > 0$ 为常数。

上述命题表明, 当 $\ell_{n,p}^{(l)}(\boldsymbol{\theta})$ 的值接近真实值 $\ell_{n,p}^{(l)}(\boldsymbol{\theta}_0)$ 时, 参数 $\boldsymbol{\theta}$ 也接近真实参数 $\boldsymbol{\theta}_0$, 这就得到了使用 $\ell_{n,p}^{(l)}(\boldsymbol{\theta})$ 估计参数时的可识别性。

当我们观察到 $\mathbf{X}_1, \ldots, \mathbf{X}_n$ 时, 由 Markov 链性质, 可以写出条件似然函数

$$\mathcal{L}_{n,p}(\mathbf{X}_n, \ldots, \mathbf{X}_{m+1} \mid \mathbf{X}_m, \ldots, \mathbf{X}_1; \boldsymbol{\theta}) = \prod_{t=m+1}^{n} L_{t,p}(\mathbf{X}_t \mid \mathbf{X}_{t-1}, \ldots, \mathbf{X}_{t-m}; \boldsymbol{\theta}), \tag{2.27}$$

其中 $L_{t,p}(\mathbf{X}_t \mid \mathbf{X}_{t-1}, \ldots, \mathbf{X}_{t-m}; \boldsymbol{\theta})$ 是在给定 $\mathbf{X}_{t-1}, \ldots, \mathbf{X}_{t-m}$ 时 $\mathbf{X}_t$ 的转移概率。根据式 (2.3) 及其参数化表达, 可写出 (标准化的) 对数似然函数

$$\begin{aligned} &\frac{2}{(n-m)p(p-1)} \log \mathcal{L}_{n,p}(\mathbf{X}_n, \ldots, \mathbf{X}_{m+1} \mid \mathbf{X}_m, \ldots, \mathbf{X}_1; \boldsymbol{\theta}) \\ &\qquad = \frac{2}{(n-m)p(p-1)} \sum_{t=m+1}^{n} \sum_{i,j:1\leqslant i<j\leqslant p} \log[\{\gamma_{i,j}^{t-1}(\boldsymbol{\theta})\}^{X_{i,j}^t}\{1-\gamma_{i,j}^{t-1}(\boldsymbol{\theta})\}^{1-X_{i,j}^t}], \end{aligned} \tag{2.28}$$

这是也当 $l \in \mathcal{G}$ 时, $\ell_{n,p}^{(l)}(\boldsymbol{\theta})$ 的样本形式。类似地, 对任意 $l \in [q]$, 定义

$$\hat{\ell}_{n,p}^{(l)}(\boldsymbol{\theta}) = \frac{1}{(n-m)|\mathcal{S}_l|} \sum_{t=m+1}^{n} \sum_{(i,j)\in\mathcal{S}_l} \log[\{\gamma_{i,j}^{t-1}(\boldsymbol{\theta})\}^{X_{i,j}^t} \{1-\gamma_{i,j}^{t-1}(\boldsymbol{\theta})\}^{1-X_{i,j}^t}]。 \tag{2.29}$$

可以证明, 对任意的 $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, $\hat{\ell}_{n,p}^{(l)}(\boldsymbol{\theta})$ 一致依概率收敛于 $\ell_{n,p}^{(l)}(\boldsymbol{\theta})$, 结合命题 1, 我们可以通过最大化 $\hat{\ell}_{n,p}^{(l)}(\boldsymbol{\theta})$ 的方式估计参数 $\boldsymbol{\theta}$。

对每个 $l \in [q]$, 令 $(\hat{\theta}_{*,1}^{(l)}, \ldots, \hat{\theta}_{*,q}^{(l)})^\top = \arg\max_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \hat{\ell}_{n,p}^{(l)}(\boldsymbol{\theta})$, 定义 $\tilde{\boldsymbol{\theta}} = (\tilde{\boldsymbol{\theta}}_{\mathcal{G}}^\top, \tilde{\boldsymbol{\theta}}_{\mathcal{G}^c}^\top)^\top$ 为 $\boldsymbol{\theta}_0$ 的估计, 其中

$$\tilde{\boldsymbol{\theta}}_{\mathcal{G}} = (\hat{\theta}_{*,l}^{(l')})_{l\in\mathcal{G}}, \quad \tilde{\boldsymbol{\theta}}_{\mathcal{G}^c} = (\hat{\theta}_{*,l}^{(l)})_{l\in\mathcal{G}^c}, \tag{2.30}$$

l' 为 $\mathcal{G}$ 中任一元素。容易发现, $\hat{\ell}_{n,p}^{(l_1)}(\boldsymbol{\theta}) = \hat{\ell}_{n,p}^{(l_2)}(\boldsymbol{\theta})$, $\forall l_1, l_2 \in \mathcal{G}$, 故 $\tilde{\boldsymbol{\theta}}_{\mathcal{G}}$ 并不依赖 l' 的选取。

定理 1 *在满足某些条件时,*

$$\|\tilde{\boldsymbol{\theta}}_{\mathcal{G}} - \boldsymbol{\theta}_{0,\mathcal{G}}\|_2 = o_p(1), \quad \|\tilde{\boldsymbol{\theta}}_{\mathcal{G}^c} - \boldsymbol{\theta}_{0,\mathcal{G}^c}\|_\infty = o_p(1)。 \tag{2.31}$$

还可以通过其他方式得到对 $\boldsymbol{\theta}_0$ 效果更好的估计, 下文将不再具体介绍。

特别地, 对标准 AR(1) 网络模型, 上述估计具有简单的形式。写出参数估计的显式表达

$$\hat{\alpha}_{i,j} = \frac{\sum_{t=1}^{n-1} X_{i,j}^{t+1}(1-X_{i,j}^t)}{\sum_{t=1}^{n-1}(1-X_{i,j}^t)}, \quad \hat{\beta}_{i,j} = \frac{\sum_{t=1}^{n-1}(1-X_{i,j}^{t+1})X_{i,j}^t}{\sum_{t=1}^{n-1} X_{i,j}^t}。 \tag{2.32}$$

令

$$J_1 = \{(i_1,j_1), \ldots, (i_{m_1}, j_{m_2})\}, \quad J_2 = \{(k_1,l_1), \ldots, (k_{m_2}, l_{m_2})\}, \tag{2.33}$$

其中 m_1, m_2 为任意正整数, 对应地, 令

$$\boldsymbol{\Theta}_{J_1,J_2} = (\alpha_{i_1,j_1}, \ldots, \alpha_{i_{m_1},j_{m_1}}, \beta_{k_1,l_1}, \ldots, \beta_{k_{m_2},l_{m_2}})^\top, \tag{2.34}$$

及相应的估计

$$\hat{\boldsymbol{\Theta}}_{J_1,J_2} = (\hat{\alpha}_{i_1,j_1}, \ldots, \hat{\alpha}_{i_{m_1},j_{m_1}}, \hat{\beta}_{k_1,l_1}, \ldots, \hat{\beta}_{k_{m_2},l_{m_2}})^\top。 \tag{2.35}$$

下述命题说明, 此参数估计具有渐进正态性。

命题 2 *在满足某些条件时,*

$$\sqrt{n}(\hat{\boldsymbol{\Theta}}_{J_1,J_2} - \boldsymbol{\Theta}_{J_1,J_2}) \xrightarrow{d} N(0, \boldsymbol{\Sigma}_{J_1,J_2}), \tag{2.36}$$

其中 $\boldsymbol{\Sigma}_{J_1,J_2} = \mathrm{diag}(\sigma_{1,1}, \ldots, \sigma_{m_1+m_2,m_1+m_2})$ *为对角阵,*

$$\sigma_{r,r} = \frac{\alpha_{i_r,j_r}(1-\alpha_{i_r,j_r})(\alpha_{i_r,j_r}+\beta_{i_r,j_r})}{\beta_{i_r,j_r}}, \quad 1 \leqslant r \leqslant m_1 \tag{2.37}$$

$$\sigma_{r,r} = \frac{\beta_{k_r,l_r}(1-\beta_{k_r,l_r})(\alpha_{k_r,l_r}+\beta_{k_r,l_r})}{\alpha_{k_r,l_r}}, \quad m_1+1 \leqslant r \leqslant m_1+m_2。 \tag{2.38}$$

3 模拟实验

本节将通过模拟实验验证参数估计的有效性, 选用标准 AR(1) 网络和传递 AR(1) 网络两种模型进行模拟实验。

3.1 标准 AR(1) 网络

首先说明标准 AR(1) 网络模型的模拟实验设计, 固定节点总数 p 和序列长度 n, 对每个 $1 \leqslant i < j \leqslant p$, 参数 $\alpha_{i,j}$ 和 $\beta_{i,j}$ 的真实值从 $U(0.1, 0.5)$ 中独立生成, 初值 $\mathbf{X}_1$ 的每个分量均从 Bernoulli(0.5) 中独立生成。依据式 (2.32) 进行参数估计, 并计算参数估计的 MSE, 依据命题 2 构造渐进 95% 置信区间, 计算其覆盖真实参数的比例。对不同的 n 和 p, 分别重复上述实验 50 次, 最终结果取均值。

		$\hat{\alpha}_{i,j}$		$\hat{\beta}_{i,j}$	
n	p	MSE	覆盖率	MSE	覆盖率
5	100	0.2037	78.85%	0.2041	78.85%
5	200	0.2039	79.02%	0.2042	78.95%
20	100	0.0407	89.21%	0.0408	89.22%
20	200	0.0405	89.19%	0.0409	89.12%
50	100	0.0116	92.86%	0.0116	92.91%
50	200	0.0116	92.90%	0.0117	92.93%
100	100	0.0050	94.01%	0.0051	94.00%
100	200	0.0051	93.97%	0.0051	93.91%
200	100	0.0024	94.44%	0.0024	94.48%
200	200	0.0024	94.52%	0.0024	94.46%

表 1: 标准 AR(1) 网络模拟结果

模拟结果如表 1 所示, 结果表明随着序列长度 n 的增加, 参数估计的 MSE 稳步降低, 置信区间的覆盖率稳步提升并接近 95%, 但估计效果不随节点总数 p 发生明显变化。该结果与直观判断相符, 越多的观察值越能反应某时间序列的整体信息, 基于此得出的统计推断也自然具有越小的不确定度。

3.2 传递 AR(1) 网络

接下来进行传递 AR(1) 网络模型的模拟实验, 由于在此模型下较难写出参数估计的显式表达, 只能通过数值方法得到参数估计。在试验中, 固定节点总数 p 和序列长度 n, 全局参数 a 的真实值从 $U(29,31)$ 中独立生成, b 的真实值从 $U(14,16)$ 中独立生成, 对每个 $i \in [p]$, 局部参数 ξ_i 的真实值从 $U(0.65,0.75)$ 中独立生成, η_i 的真实值从 $U(0.75,0.85)$ 中独立生成, 利用 R 包 `arnetworks` 独立生成对应的网络序列并进行参数估计, 该 R 包可于 https://github.com/peterwmacd/arnetworks 处下载。对不同的 n 和 p, 分别重复上述实验 50 次, 得到参数估计的 MSE 均值。

n	p	MSE($\hat{a}$)	MSE($\hat{b}$)	MSE($\hat{\xi}$)	MSE($\hat{\eta}$)
20	50	9.4297	1.3959	0.0276	0.0226
20	100	7.3535	0.3263	0.0079	0.0036
50	50	3.0584	0.4257	0.0145	0.0147
50	100	1.9777	0.0969	0.0034	0.0014

表 2: 传递 AR(1) 网络模拟结果

模拟结果如表 2 所示, 结果表明在节点总数 p 不变时, 随着序列长度 n 的增加, 参数估计的 MSE 均有所降低, 这符合增加样本量可以减小估计的不确定性的直观解释。并且在序列长度 n 不变时, 各个参数估计的 MSE 随着节点总数 p 的增加而降低, 这是由于在传递模型中, 更多的节点数量可以表现出更为明显的传递效应, 进而使得参数估计更为准确。

4 真实数据分析

有别于原文的真实数据集, 我们将传递 AR(1) 网络模型应用于 Enron 电子邮件数据集上, 来探究其适用性与应用效果。该数据集记录了 Enron 公司员工的来往邮件, 可于 https://www.kaggle.com/datasets/wcukierski/enron-email-dataset 处获取。本文选取处理时间段为 2000 年 1 月 1 日至 2002 年 3 月 31 日, 并选取收发邮件最活跃的 64 名员工进行分析, 考虑到收发邮件的周效应以及延迟性, 以两周为一个时间单位, 即节点总数 $p = 64$, 序列长度 $n = 117$。若第 i 位员工和第 j 位员工在第 t 个时间单位内进行过至少一封邮件的通讯, 则记 $X_{ij}^t = 1$。

为了更好地利用传递 AR(1) 网络模型拟合数据, 首先考虑数据的平稳性, 定义三种节点密度

$$D_t = \frac{\sum_{i,j:i<j} X_{i,j}^t}{p(p-1)/2}, \quad D_{1,t} = \frac{\sum_{i,j:i<j}(1-X_{i,j}^{t-1})X_{i,j}^t}{p(p-1)/2}, \quad D_{0,t} = \frac{\sum_{i,j:i<j} X_{i,j}^{t-1}(1-X_{i,j}^t)}{p(p-1)/2}, \tag{4.1}$$

其中 D_t 代表 t 时刻的边密度, 表示 t 时刻平均建立联系的人数, $D_{1,t}, D_{2,t}$ 分别是 t 时刻新生长和新消失的边的密度, 表示 t 时刻新建立和新解除联系的人数比例, 这可用于说明网络的动态变化情况。图 1 揭示了边密度 D_t 随时间的变化关系, 图 2 说明生长边密度 $D_{1,t}$ 和消失边密度 $D_{2,t}$ 随时间的变化关系, 从两张图都能看出, 在 $t \in [17, 37]$ 时间段内数据较为平稳, 在 $t = 16, 38$ 处发生了明显变化, 故选取 $t \in [17, 36]$ 的数据进行模型拟合, 取 $t = 37$ 进行模型的预测和检验。

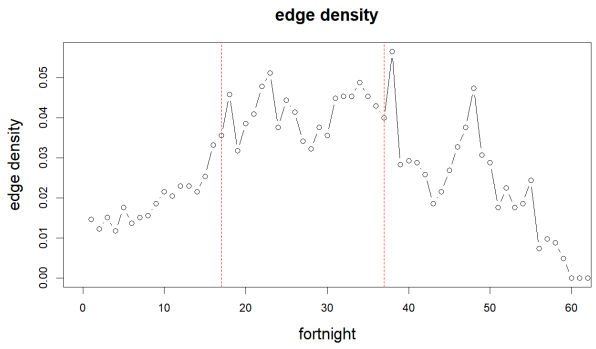

图 1: 边密度

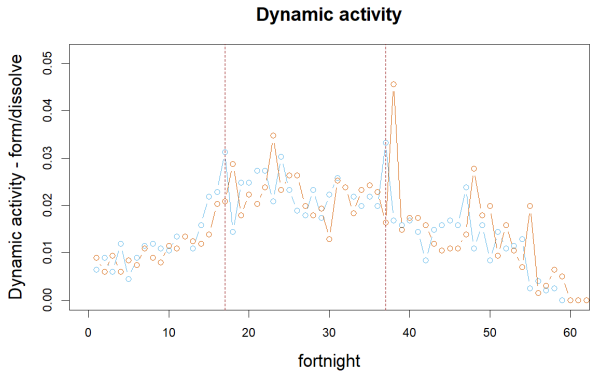

图 2: 生长边密度 (蓝色), 消失边密度 (橙色)

接下来说明该真实数据集中表现出的传递效应, 对每个整数 $\ell \geqslant 0$, 定义

$$\mathcal{U}_\ell = \left\{(i,j,t) : 1 \leqslant i < j \leqslant p,\ t \in [n]\backslash\{1\},\ X_{i,j}^{t-1} = 0,\ U_{i,j}^{t-1} = \ell/(p-2)\right\}, \tag{4.2}$$

$$\mathcal{V}_\ell = \left\{(i,j,t) : 1 \leqslant i < j \leqslant p,\ t \in [n]\backslash\{1\},\ X_{i,j}^{t-1} = 1,\ V_{i,j}^{t-1} = \ell/(p-2)\right\}, \tag{4.3}$$

$$\mathcal{U}_\ell^1 = \left\{(i,j,t) \in \mathcal{U}_\ell,\ X_{i,j}^t = 1\right\}, \tag{4.4}$$

$$\mathcal{V}_\ell^0 = \left\{(i,j,t) \in \mathcal{V}_\ell,\ X_{i,j}^t = 0\right\}。 \tag{4.5}$$

$|\mathcal{U}_\ell^1|/|\mathcal{U}_\ell|$ 表示固定共同朋友 (同时连接节点 i 与节点 j 的节点) 总数时, 对应节点的生长边相对频率, 本数据集生长边相对频率如图 3 所示, 可以发现, 共同朋友总数越多, 对应节点的生长边相对频率 $|\mathcal{U}_\ell^1|/|\mathcal{U}_\ell|$ 越高, 这揭示了拥有更多共同朋友的两人更倾向于进行邮件交流。图 4 展示了消失边相对频率 $|\mathcal{V}_\ell^0|/|\mathcal{V}_\ell|$ 与非共同朋友 (仅连接节点 i 或节点 j 之一的节点) 总数的关系, 可以看出, 消失边相对频率随着非共同朋友总数的增长先保持基本稳定, 而在非共同朋友总数增长至较大数值时, 消失边相对频率表现出了上升趋势, 这表明了拥有较多非共同朋友的两人进行邮件交流的倾向较低, 但表现并不明显。

同时也可以通过拟合的模型参数映证传递效应。对于时间段 $t \in [17, 36]$, 使用 R 包 `arnetworks` 进行参数估计, 得到全局参数的估计 $\hat{a} = 27.5643, \hat{b} = 14.4803$, 从传递 AR(1) 网络模型的定义 3 可

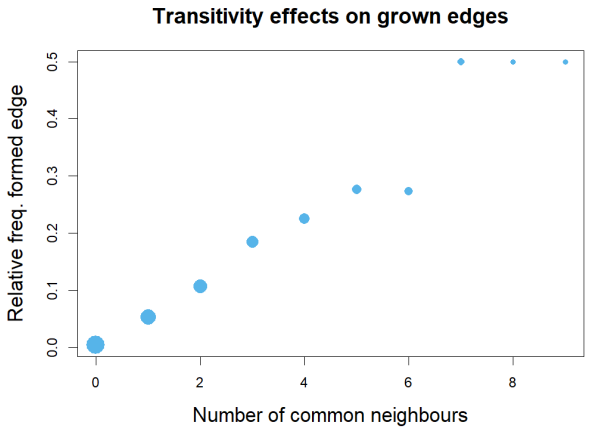

图 3: 生长边相对频率

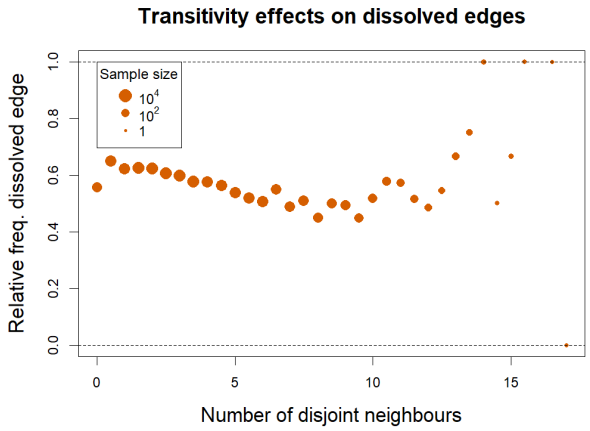

图 4: 消失边相对频率

以看出, 全局参数 $\hat{a}$ 较大意味着拥有更多共同朋友时边生长的倾向较强, 而全局参数 $\hat{b}$ 较小则表明拥有更多非共同朋友时边消失的倾向并不明显, 这与上文两者的趋势一致。

利用图 5 分析局部参数的估计值。局部参数 $\hat{\xi}$ 的均值为 0.3767, 向靠近 0 处倾斜, $\hat{\eta}$ 的均值为 1.0549, 向远离 0 处倾斜, 局部参数的不同取值表明了每个节点的异质性。可以明显看出, 局部参数 $\hat{\xi}$ 和 $\hat{\eta}$ 之间存在负相关关系, 这表明倾向于生长边的节点同时倾向于保持现有的边, 即该公司员工在新增邮件联系人的同时保持与原有联系人的交流。

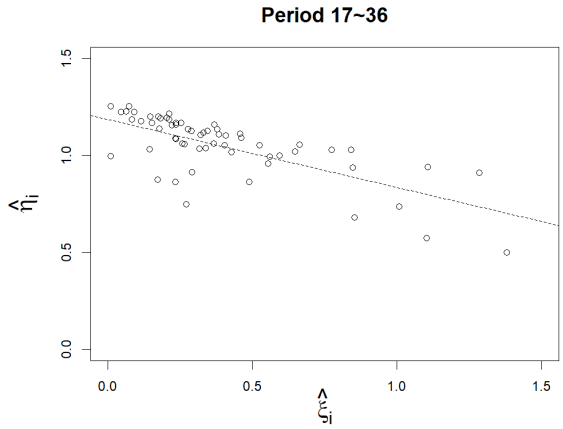

图 5: 局部参数估计值 $\hat{\xi}$ 和 $\hat{\eta}$

根据式 (2.3), 按如下步骤进行一步预测,

$$\hat{X}_{i,j}^{t+1} = \begin{cases} 0, & \text{若 } \hat{\alpha}_{i,j}^t + X_{i,j}^t(1 - \hat{\alpha}_{i,j}^t - \hat{\beta}_{i,j}^t) \in [0, 0.5), \\ 1, & \text{若 } \hat{\alpha}_{i,j}^t + X_{i,j}^t(1 - \hat{\alpha}_{i,j}^t - \hat{\beta}_{i,j}^t) \in [0.5, 1], \end{cases} \tag{4.6}$$

其中 $\hat{\alpha}_{i,j}^t$ 和 $\hat{\beta}_{i,j}^t$ 由估计参数、式 (2.15) 和式 (2.16) 得到。为了检验一步预测的效果, 使用 $\frac{\|X_{ij}^{37}-\hat{X}_{ij}^{37}\|_F}{p(p-1)}$ 度量预测值和真实值的差异, 即估计错误的边在总边数中所占比例, 其中记号 $\|\cdot\|_F$ 为 Frobenius 范数, 得到结果为 0.0352, 这说明传递 AR(1) 网络模型的一步预测在本数据集下具有良好的准确性。

为了更好地说明传递 AR(1) 网络模型在本数据集下的适用性, 我们将其与标准 AR(1) 网络模型进行比较, 对比它们的 AIC 与 BIC 信息准则, 结果如表 3 所示。可以发现, 传递 AR(1) 网络模型的 AIC 值和 BIC 值均小于标准 AR(1) 网络模型, 说明从信息准则的角度考虑, 传递 AR(1) 网络模型更优。

模型	AIC	BIC
传递 AR(1) 网络	**9260**	**10372**
标准 AR(1) 网络	13372	47859

表 3: 模型比较

5 结论

本文通过对 Chang et al. (2024) 和 Jiang et al. (2023) 提出的自回归动态网络模型的学习与复现, 介绍了一般的 AR(m) 网络模型及其极大似然估计, 并对标准 AR(1) 网络和传递 AR(1) 网络模型进行了模拟实验和真实数据集的应用。

在标准 AR(1) 网络的模拟实验中, 随着序列长度的增加, 参数估计的均方误差稳步降低且置信区间的覆盖率稳步提升, 而估计的 MSE 与置信区间的覆盖率并没有随节点总数的变化而发生明显改变。在传递 AR(1) 网络的模拟实验中, 固定节点总数时, 参数估计的 MSE 随序列长度的增加均有所降低; 固定序列长度时, 参数估计的 MSE 随序列长度的增加也均有所降低。通过以上两个模拟实验, 本文成功验证了自回归动态网络模型极大似然估计的有效性。

本文将传递 AR(1) 网络模型应用于 Enron 电子邮件数据集之上, 该真实数据集表现出良好的传递效应, 得到的参数估计也映证了这一性质, 同时基于参数估计进行的一步预测也具有较高的准确率。通过比较 AIC 与 BIC 信息准则, 传递 AR(1) 网络模型在此数据集下的适用性优于标准 AR(1) 网络模型。该真实数据集的结论表明自回归动态网络模型在实际问题中具有较好的应用效果。

参考文献

Chang, J., Fang, Q., Kolaczyk, E. D., MacDonald, P. W. and Yao, Q. (2024). Autoregressive networks with dependent edges.
URL: *https://arxiv.org/abs/2404.15654*

Jiang, B., Li, J. and Yao, Q. (2023). Autoregressive networks, *Journal of Machine Learning Research* **24**(227): 1–69.

A 附录：代码

标准 AR(1) 网络模拟实验代码:

```
EstAlpha <- function(data, length, size) { #估计alpha
  ans <- matrix(0, nrow = size, ncol = size)
  for(i in 1:size) {
    for(j in 1:size) {
      num <- den <- 0
      for(k in 1:(length - 1)) {
        num <- num + data[[k + 1]][i, j] * (1 - data[[k]][i, j])
        den <- den + (1 - data[[k]][i, j])
      }
      ans[i, j] <- ifelse(num == 0 && den == 0, 1, num / den)
    }
  }
  return(ans)
}

EstBeta <- function(data, length, size) { #估计beta
  ans <- matrix(0, nrow = size, ncol = size)
  for(i in 1:size) {
    for(j in 1:size) {
      num <- den <- 0
      for(k in 1:(length - 1)) {
        num <- num + (1 - data[[k + 1]][i, j]) * data[[k]][i, j]
        den <- den + data[[k]][i, j]
      }
      ans[i, j] <- ifelse(num == 0 && den == 0, 1, num / den)
    }
  }
  return(ans)
}

library(snowfall)
sfInit(parallel = TRUE, cpus = 4)

set.seed(123)
n <- 100 #序列长度
p <- 100 #网络节点总数
times <- 50 #重复实验次数

MSEa <- MSEb <- numeric(times) #MSE
CONa <- CONb <- numeric(times) #在置信域的比例

for(l in 1:times) {
  alpha <- matrix(0, nrow = p, ncol = p)
  beta <- matrix(1, nrow = p, ncol = p)
  x <- list(matrix(0, nrow = p, ncol = p))
  for(i in 1:p) {
```

```
    for(j in (i + 1):p) {
      if(j == p + 1) break
      alpha[j, i] <- alpha[i, j] <- runif(1, 0.1, 0.5)
      beta[j, i] <- beta[i, j] <- runif(1, 0.1, 0.5)
      x[[1]][j, i] <- x[[1]][i, j] <- rbinom(1, 1, 0.5)
    }
  }
  epsilon <- vector(mode = "list", length = n)
  for(k in 2:n) { #生成随机数据
    x[[k]] <- epsilon[[k]] <- matrix(0, nrow = p, ncol = p)
    for(i in 1:p) {
      for(j in (i + 1):p) {
        if(j == p + 1) break
        e <- sample(-1:1, size = 1, replace = TRUE,
          prob = c(beta[i, j], 1 - alpha[i, j] - beta[i, j], alpha[i, j]))
        epsilon[[k]][i, j] <- e
        x[[k]][j, i] <- x[[k]][i, j] <- x[[k - 1]][i, j] * (e == 0) + (e == 1)
      }
    }
  }

  alpha_hat <- EstAlpha(data = x, length = n, size = p)
  beta_hat <- EstBeta(data = x, length = n, size = p)

  MSEa[l] <- mean((alpha_hat - alpha) ^ 2)
  MSEb[l] <- mean((beta_hat - beta) ^ 2)

  SigmaA <- alpha * (1 - alpha) * (alpha + beta) / beta
  SigmaB <- beta * (1 - beta) * (alpha + beta) / alpha
  BA <- qnorm(0.975) * sqrt(SigmaA / n) #计算置信界
  BB <- qnorm(0.975) * sqrt(SigmaB / n)
  for(i in 1:p) {
    for(j in (i + 1):p) {
      if(j == p + 1) break
      if(abs(alpha[i, j] - alpha_hat[i, j]) <= BA[i, j]) CONa[l] <- CONa[l] + 1
      if(abs(beta[i, j] - beta_hat[i, j]) <= BB[i, j]) CONb[l] <- CONb[l] + 1
    }
  }
  CONa[l] <- CONa[l] / (p * (p - 1) / 2)
  CONb[l] <- CONb[l] / (p * (p - 1) / 2)
}

sfStop()

mean(MSEa)
mean(CONa)
mean(MSEb)
mean(CONb)
```

传递 AR(1) 网络模拟实验代码:

```
library(snowfall)
sfInit(parallel = TRUE, cpus = 4)

library(arnetworks)

set.seed(123)
n <- 50
p <- 100
times <- 50
MSEa <- numeric(times)
MSEb <- numeric(times)
MSEx <- numeric(times)
MSEe <- numeric(times)

for(l in 1:times) {
  xi <- runif(p, 0.65, 0.75)
  eta <- runif(p, 0.75, 0.85)
  a <- runif(1, 29, 31)
  b <- runif(1, 14, 16)

  data <- simulateTransitivity(p, n, xi, eta, a, b)
  X = data$X
  U = data$U
  V = data$V

  fit = estTransitivity(X, U, V, verbose = TRUE)

  MSEa[l] <- (fit$gVal[1] - a) ^ 2
  MSEb[l] <- (fit$gVal[2] - b) ^ 2
  MSEx[l] <- mean((fit$xi - xi) ^ 2)
  MSEe[l] <- mean((fit$eta - eta) ^ 2)
}

sfStop()

MSEa
mean(MSEa)
MSEb
mean(MSEb)
MSEx
mean(MSEx)
MSEe
mean(MSEe)
```

真实数据预处理代码:

```
library(dplyr)
library(GGally)
library(gender)
library(readr)
```

```
library(ggplot2)
library(stringr)
library(tidyr)
data <- read_csv("emails.csv", na=c("NA"))
data %>%
  select(file) %>%
  sample_n(10)
data <- data %>%
    separate(message, into=c("header", "body"), sep="\n\n",
             extra="merge", remove=TRUE)
parseHeader <- function(header){
        MessageID <- str_sub(str_extract(header, "^Message-ID:.*"), start = 12)
        Date <- str_sub(str_extract(header,"Date:.*"), start = 7)
        From <- str_sub(str_extract(header,"From:.*"), start = 7)
        To <- str_sub(str_extract(header,"To:.*"), start = 5)
        Subject <- str_sub(str_extract(header,"Subject:.*"), start = 10)

        headerParsed <- data.frame(cbind(MessageID, Date, From, To, Subject),
                                   stringsAsFactors = FALSE)
        return(headerParsed)
}
headerParsed <- parseHeader(data$header)
head(headerParsed)
data <- cbind(data, headerParsed)
emails_df <- data
Sys.setlocale("LC_TIME", "C")
emails_df<-emails_df[,c("From","To","Date")]
emails_df$Date <- sub("\\s\\d{2}:\\d{2}:\\d{2}.*", "", emails_df$Date)
emails_df$Date <- sub("^\\w{3},\\s", "", emails_df$Date)
emails_df$Date <-as.character(emails_df$Date)
emails_df$Date <- as.Date(emails_df$Date,format = "%d %b %Y")
start_date <- as.Date("2000-01-01")
end_date <- as.Date("2002-03-31")
emails_df <- emails_df[emails_df$Date >= start_date & emails_df$Date <= end_date, ]
emails_df_split <- do.call(rbind, lapply(1:nrow(emails_df), function(i) {
  recipients <- strsplit(emails_df$To[i], ",\\s*")[[1]]
  data.frame(From = rep(emails_df$From[i], length(recipients)),
             To = recipients,
             Date = rep(emails_df$Date[i], length(recipients)),
             stringsAsFactors = FALSE)
}))
write.csv(emails_df_split,"final.csv")
```

传递 AR(1) 网络真实数据分析代码:

```
library(arnetworks)
library(dplyr)
library(lubridate)
library(latex2exp)
```

```r
# 预处理得到网络数据X

# email_data是包含邮件数据的数据框,具有以下列:
# From: 发件人邮箱
# To: 收件人邮箱
# Date: 邮件发送日期

email_data <- read.csv(file = 'final-split.csv',header = T,
                       na.strings = '',encoding = 'UTF-8')
email_data$Date <- as.Date(email_data$Date)

#自己发给自己的删去
k <- which(email_data$From==email_data$To)
email_data <- email_data[-k,]

# 获取所有唯一的参与者,创建参与者ID映射
participants <- unique(c(email_data$From, email_data$To))
participant_ids <- setNames(as.integer(1:length(participants)), participants)
email_data$From_id <- email_data$From %in% names(participant_ids)
email_data$To_id <- email_data$To %in% names(participant_ids)
email_data$From_id <- participant_ids[email_data$From]
email_data$To_id <- participant_ids[email_data$To]
email_data <- na.omit(email_data)

# 选取收发邮件最活跃的id

id1_counts <- table(email_data$From_id)
id2_counts <- table(email_data$To_id)
# 将结果转换为数据框,并按出现次数降序排序
id1_counts_df <- as.data.frame(id1_counts) %>%
  mutate(count = as.integer(Freq)) %>%
  arrange(desc(count))
id2_counts_df <- as.data.frame(id2_counts) %>%
  mutate(count = as.integer(Freq)) %>%
  arrange(desc(count))

# 选择出现次数最多的前1000个ID
id1 <- head(id1_counts_df, 1000)
id2 <- head(id2_counts_df, 1000)

id <- id1[id1$Var1 %in% id2$Var1,]$Var1
id <- sort(id)
active <- email_data[email_data$From_id %in% id & email_data$To_id %in% id, ]

# 填充X

# 计算时间窗口(两周)
active$Date <- floor_date(active$Date, unit = "week")
unique_dates <- unique(active$Date)
```

```r
max_t <- length(unique_dates)
X <- array(0, dim = c(length(id), length(id), max_t))

for (t in 1:max_t) {

  current_date <- min(active$Date) + days(14*(t-1))
  subset_data <- filter(active, Date == current_date)

  for (i in 1:nrow(subset_data)) {
    j <- which(id==subset_data$From_id[i])
    k <- which(id==subset_data$To_id[i])
    X[j, k, t] <- 1
    X[k, j, t] <- 1
  }
}

grow <- X[,,-1]*(1 - X[,,-max_t]) # 多少条边从0变成1
diss <- X[,,-max_t]*(1 - X[,,-1]) # 多少条边从1变成0
dynamic_count <- apply(grow+diss, 1, sum) # 每个节点变化的总边数

node_keep <- which(dynamic_count >= 200) # 门槛,只保留变化多的节点
X <- X[node_keep,node_keep,]

# 用平稳部分拟合传递AR模型
X_ <- X[,,17:36]
fit <- arnetworks::estTransitivity(X_, verbose=TRUE)
UV <- arnetworks::statsTransitivity(X_)

# 预测
n <- dim(X_)[3]
p <- dim(X_)[1]
Xnew = X[,,36]
pred1 = predictTransitivity(fit,Xnew,nStep=1)
dist =  sum(abs(X[,,37]-round(pred1, 0)))/(p*(p-1))

# 绘图
# 边密度图
plot(dens, type = 'b', main = 'edge density',
     ylab = 'edge density', xlab = 'Week', cex.main = 1.5, cex.lab = 1.4)
abline(v = 17, col = "red", lty = 2)
abline(v = 37, col = "red", lty = 2)
dev.off()

# 生长边密度和消失边密度图
grow_sub <- grow[node_keep,node_keep,]
diss_sub <- diss[node_keep,node_keep,]

grt <- apply(grow_sub,3,sum)/p
dst <- apply(diss_sub,3,sum)/p
```

```
grt_p2 <- apply(grow_sub,3,sum)/(p*(p-1))
dst_p2 <- apply(diss_sub,3,sum)/(p*(p-1))
cbp <- c("#000000", "#E69F00", "#56B4E9", "#009E73",
         "#F0E442", "#0072B2", "#D55E00", "#CC79A7")

plot(grt_p2,type='b',col=cbp[3],main='Dynamic activity',
     ylim=c(0,0.02),ylab='Dynamic activity - form/dissolve',xlab='Week',
     cex.main=1.5,cex.lab=1.4)
lines(dst_p2,type='b',col=cbp[7])
abline(h=mean(grt_p2),col=cbp[3],lty=3)
abline(h=mean(dst_p2),col=cbp[7],lty=3)
abline(v = 17, col = "darkred", lty = 2)
abline(v = 37, col = "darkred", lty = 2)
dev.off()

# 传递效应图
hollowize <- function(M){
  M - diag(diag(M))
}
gr_cn <- NULL
for(tt in 2:n){
  prev_cn <- X[,,tt-1]%*%X[,,tt-1]
  # growing edges
  grow_ind <- as.logical((1-X[,,tt-1])*X[,,tt])
  if(sum(grow_ind) > 0){
    gr_cn <- rbind(gr_cn,cbind(1,prev_cn[grow_ind]))
  }
  # non-growing edges
  nongrow_ind <- as.logical(hollowize((1-X[,,tt-1])*(1-X[,,tt])))
  if(sum(nongrow_ind)>0){
    gr_cn <- rbind(gr_cn,cbind(0,prev_cn[nongrow_ind]))
  }
}

gr_cn_summary <- table(gr_cn[,1],gr_cn[,2])

plot(as.numeric(colnames(gr_cn_summary)),
     gr_cn_summary[2,]/colSums(gr_cn_summary),
     type='p',pch=16,cex=(3+log(colSums(gr_cn_summary)))/6,col=cbp[3],
     main='Transitivity effects on grown edges',cex.main=1.5,
     xlab='Number of common neighbours',ylab='Relative freq. formed edge',
     cex.lab=1.4)
legend(x=17.5,y=0.25,pch=16,col=cbp[3],title='Sample size',
       pt.cex=(3+log(c(1e4,1e2,1)))/6,
       legend=c(TeX('$10^4$'),TeX('$10^2$'),TeX('$1$')))
dev.off()

ds_ncn <- NULL
```

```
for(tt in 2:n){
  prev_ncn <- X[,,tt-1] %*% hollowize(1-X[,,tt-1])
  # symmetrize
  prev_ncn <- .5*(prev_ncn + t(prev_ncn))
  # dissolving edges
  dissolve_ind <- as.logical(X[,,tt-1]*(1-X[,,tt]))
  if(sum(dissolve_ind) > 0){
    ds_ncn <- rbind(ds_ncn,cbind(1,prev_ncn[dissolve_ind]))
  }
  # non-growing edges
  nondissolve_ind <- as.logical(X[,,tt-1]*X[,,tt])
  if(sum(nondissolve_ind)>0){
    ds_ncn <- rbind(ds_ncn,cbind(0,prev_ncn[nondissolve_ind]))
  }
}

ds_ncn_summary <- table(ds_ncn[,1],ds_ncn[,2])

plot(as.numeric(colnames(ds_ncn_summary)),
     ds_ncn_summary[2,]/colSums(ds_ncn_summary),
     type='p',pch=16,cex=(3+log(colSums(ds_ncn_summary)))/6,col=cbp[7],
     main='Transitivity effects on dissolved edges',cex.main=1.5,
     xlab='Number of disjoint neighbours',ylab='Relative freq. dissolved edge',
     cex.lab=1.4)
legend(x=0,y=1,pch=16,col=cbp[7],title='Sample size',
       pt.cex=(3+log(c(1e4,1e2,1)))/6,
       legend=c(TeX('$10^4$'),TeX('$10^2$'),TeX('$1$')))
abline(h=0,lty=2)
abline(h=1,lty=2)
dev.off()

# 局部参数估计图
par(mar=c(4,5.5,4,4))
plot(fit$xi,fit$eta,
     main='Period 17~36',xlab=TeX('$\\hat{\\xi}_i'),ylab=TeX('$\\hat{\\eta}_i'),
     xlim=c(0,1.5),ylim=c(0,1.5),cex.lab=1.8,cex.main=1.5)
abline(lm(fit$eta~fit$xi),lty=2)
dev.off()


# 模型比较
logadj = function(x){
  log(pmax(1e-5,x))
}

model_probs <- function(fit,stats,X){
  p <- dim(X)[1]
  n <- dim(X)[3]
  alpha <- beta <- gamma <- array(NA,c(p,p,n-1))
```

```
  Xxi <- tcrossprod(fit$xi)
  Eeta <- tcrossprod(fit$eta)
  Xxi <- Xxi - diag(diag(Xxi))
  Eeta <- Eeta - diag(diag(Eeta))
  for(t in 1:(n-1)){
    eaU <- exp(fit$gVal[1] * stats$U[,,t])
    ebV <- exp(fit$gVal[2] * stats$V[,,t])
    alpha[,,t] <- (Xxi * eaU) / (1 + eaU + ebV)
    beta[,,t] <- (Eeta * ebV) / (1 + eaU + ebV)
    gamma[,,t] <- alpha[,,t] + X[,,t]*(1 - alpha[,,t] - beta[,,t])
  }
  return(list(alpha=alpha,beta=beta,gamma=gamma))
}

edge_ar_fit <- function(X){
  n <- dim(X)[3]
  p <- dim(X)[1]
  a1 <- apply(X[,,-1]*(1-X[,,-n]),c(1,2),sum)
  a2 <- apply(1-X[,,-n],c(1,2),sum)
  alpha_hat <- a1/a2
  alpha_hat[is.nan(alpha_hat)] <- 1
  b1 <- apply((1-X[,,-1])*X[,,-n],c(1,2),sum)
  b2 <- apply(X[,,-n],c(1,2),sum)
  beta_hat <- b1/b2
  beta_hat[is.nan(beta_hat)] <- 1
  return(list(A=alpha_hat,B=beta_hat))
}


ut <- function(M){
  c(M[upper.tri(M,diag=FALSE)])
}

ar_loglike <- function(X,gamma){
  n1 <- dim(X)[3]
  temp <- (1-X)*logadj(1-gamma) + X*logadj(gamma)
  ll_vec <- apply(temp,3,function(M){sum(ut(M))})
  ll <- sum(ll_vec)
  return(ll)
}

n_bic <- (n-1)*choose(p,2)

gamma_tar <- model_probs(fit,UV,X_)$gamma
ll_tar <- ar_loglike(X_[,,-1],gamma_tar)
AIC1 <- 2*(2*p + 2) - 2*ll_tar
BIC1 <- log(n_bic)*(2*p + 2) - 2*ll_tar

fit_ear <- edge_ar_fit(X_)
```

```
gamma_ear <- array(rep(fit_ear$A,n-1),c(p,p,n-1)) +
             X_[,,-n]*(1 - array(rep(fit_ear$A+fit_ear$B,n-1),c(p,p,n-1)))
ll_ear <- ar_loglike(X_[,,-1],gamma_ear)
AIC2 <- 2*(2*choose(p,2)) - 2*ll_ear
BIC2 <- log(n_bic)*(2*choose(p,2)) - 2*ll_ear
```