Оглавление

[Перечень условных обозначений 1](#_Toc167489701)

[Введение 2](#_Toc167489702)

[1. Обзор литературы 3](#_Toc167489703)

[2. Теоретическая часть САОМ 6](#_Toc167489704)

[2.1 Постановка задачи приводящей к САОМ 6](#_Toc167489705)

[2.2 Описание САОМ 7](#_Toc167489706)

[2.3 Оценка параметров 13](#_Toc167489707)

[3. Применение модели на реальных данных 15](#_Toc167489708)

[3.1 Объект исследования 15](#_Toc167489709)

[3.2 Применимость САОМ в исследовании научных групп 15](#_Toc167489710)

[3.3 Формирование гипотез о динамике рассматриваемой сети 17](#_Toc167489711)

[3.4 Описание модели 17](#_Toc167489712)

[3.5 Оценка сформированных гипотез 18](#_Toc167489713)

[3.5.1 Оцененные параметры при эффектах 18](#_Toc167489714)

[3.5.2 Интерпретация полученных оценок параметров 18](#_Toc167489715)

[4. Построение имитационной модели 21](#_Toc167489716)

[4.1 Описание модели 21](#_Toc167489717)

[4.2 Допущения 21](#_Toc167489718)

[4.3 Описание работы имитационной модели 22](#_Toc167489719)

[4.4 Оценка работы имитационной модели 22](#_Toc167489720)

[Заключение 26](#_Toc167489721)

[Список использованных источников и литературы 27](#_Toc167489722)

[Приложение А Код загрузчика данных 28](#_Toc167489723)

[Приложение Б Код имитационной модели 43](#_Toc167489724)

[Приложение В Вывод оценки параметров имитационной модели пакетом RSiena 48](#_Toc167489725)

[Приложение Г Вывод оценки пармтров анализа социальной сети научного сообщества ТГУ 49](#_Toc167489726)

# Перечень условных обозначений

Социальная сеть - объединение социальных акторов и их связей.

Актор - действующий субъект, совершающий действие, направленное на других акторов.

Ковариата - частная характеристика актора.

САОМ, SAOM – класс стохастических акторно ориентированных моделей (Stochastic Actor-Oriented Models).

Лонгтитьюдные данные -

Убрать главу, добавить что то еще?

# Введение

В современном мире изучение социальных сетей приобретает всё большую значимость. Это связано с необходимостью анализа и оптимизации различных социальных процессов, будь то в науке, экономике или политике. С развитием сети интернет появилась возможность собирать и анализировать объективные данные о социальных сетях, и их структурах. Это вызвало развитие такой дисциплины, как сетевой анализ. В рамках сетевого анализа существует класс акторно-ориентированных моделей. Стохастическое акторно-ориентированное моделирование является мощным инструментом для анализа и предсказания сетевой динамики.

Объектом данного исследования является класс стохастических акторно-ориентированных моделей, а именно построение этих моделей, их применимость на реальных данных и построение имитационной модели. Предметом исследования выступают факторы, влияющие на динамику сети соавторства в Томском государственном университете.

Целью данной работы является построение и анализ стохастических акторно-ориентированных моделей. Для достижения поставленной цели можно выделить 2 глобальные задачи: Анализ реальных данных на примере сети научного соавторства Томского государственного университета и построение имитационной модели.

Подробнее про задачи данного исследования

1. Применение стохастической акторно-ориентированной модели (САОМ, SAOM) на реальных данных
   1. Выгрузка данных о авторах и их научных публикациях из Тгу. сотрудники.
   2. Предварительная обработка данных. Выделение 3 снимков сети и векторов стажа, обработка некорректных данных и ошибок при загрузке.
   3. Построение модели на основе социологических предположений и проверка гипотез
   4. Анализ результатов
2. Построение имитационной модели САОМ
   1. Описание модели
   2. Разработка алгоритма
   3. Построение модели
   4. Анализ корректности работы модели

# Обзор литературы

В обзорной статье[1] рассматриваются стохастические акторно-ориентированные модели (САОМ) и дискретные экспоненциальные модели случайных графов (ERGM) для анализа динамики социальных сетей.

Эта работа представляет собой обзор двух статистических методов анализа данных, примененных для исследования социальной сети голландских школьников. Статья имеет особую значимость, так как является первой публикацией о САОМ на русском языке, что позволяет исследователям опираться на представленную авторами терминологию в рамках данного исследования.

В статье "Scientific collaboration dynamics in a national scientific system"[3] авторы исследуют взаимодействия исследователей в научной системе Словении в период с 1996 по 2010 годы. Работа охватывает широкий спектр вопросов, связанных с динамикой научного сотрудничества, применяя модели "малого мира" и процесс кумулятивного преимущества.

Модели "малого мира" (Small-World Models) характеризуются высокой кластеризацией и короткими средними путями между узлами сети, что позволяет моделировать эффективность распространения информации и взаимодействий в научном сообществе.

Процесс кумулятивного преимущества (Cumulative Advantage Process) описывает феномен, при котором более успешные и известные исследователи имеют тенденцию к получению большего числа новых коллабораций и ресурсов.

В статье формулируются социологические гипотезы о динамике сети научного сообщества и используются стохастические акторно-ориентированные модели (САОМ) для анализа данных. Социологические гипотезы легли в основу части анализа реальных данных текущего исследования.

Анализ данных подтвердил многие из предложенных гипотез, показав, что структура научной сети в Словении действительно обладает свойствами "малого мира" и подвержена процессу кумулятивного преимущества. Использование САОМ позволило выявить важные закономерности в динамике научного сотрудничества и предложить рекомендации для улучшения управления научной деятельностью и поддержки исследователей.

«Stochastic Actor-Oriented Models for Network Dynamics»[4] является фундаментальной для данного исследования. Данная публикация представляет собой руководство по стохастическим акторно-ориентированным моделям, описывающее основные принципы, методы моделирования, построение моделей, их калибровку и интерпретацию результатов.

Работа Снейдерса и Пикапа является ключевым ресурсом для исследователей, занимающихся анализом динамики социальных сетей. Публикация не только описывает методологические аспекты САОМ, но и предоставляет практические рекомендации по их применению. Эти модели позволяют глубже понять механизмы формирования и развития социальных структур, что имеет важное значение для различных областей, включая социологию, психологию, экономику и информатику.

Руководство по пакету RSiena[7], подготовленное М. Р. Рутом, представляет собой детализированное методическое пособие для пользователей, работающих с программным обеспечением RSiena для анализа динамики социальных сетей. Данное руководство не только предоставляет практические инструкции по использованию RSiena, но и включает описание множества сетевых эффектов, а также интерпретацию результатов моделирования. Оно является незаменимым ресурсом для исследователей, применяющих стохастические акторно-ориентированные модели (САОМ) для анализа сетей.

Работа Тома А. Б. Снайдерса по статистическому оцениванию динамики социальных сетей[8] является базовым текстом в области анализа социальных сетей (SNA). Он предоставляет всестороннюю схему для понимания того, как социальные сети эволюционируют со временем и вводит сложные статистические методы для моделирования и анализа этих изменений.

Снайдерс подчеркивает важность лонгитюдных данных, которые включают повторяющиеся наблюдения за одной и той же социальной сетью во времени. Этот подход позволяет исследователям улавливать динамику эволюции сети и понимать процессы, управляющие этими изменениями.

Значительный вклад Снайдерса состоит в разработке САОМ. Эти модели рассматривают изменения сети как результат решений и действий отдельных акторов, под влиянием их предпочтений и структуры сети.

Статистика и параметры сети: В тексте детально описаны различные сетевые статистики (например, распределение степеней, меры центральности)

# Теоретическая часть САОМ

## Постановка задачи приводящей к САОМ

Рассмотрим некоторую социальную сеть. Изменение сети можно рассматривать как изменение актором своего «положения» в этой сети. Каждое изменение связей актора, а именно создание новой связи с другим актором, или разрушение существующей связи приводит к изменению всей сети.

Для разных сетей могут быть разные требования к этому изменению. Например: для сети дружбы между одногруппниками процесс создания или разрушения связи ( в данном случае связь – считает ли актор другого актора своим другом) происходит в одностороннем порядке, и не требует подтверждения принимающей стороны ( можно считать своим другом того, кто не считает своим другом). Однако в случае сети научного соавторства необходимо подтверждение обеих сторон для создания связи. Так же следует заметить, что в примере с сетью научного соавторства нельзя разрушить связь.

Как при выборе того, с кем актор хочет изменить сеть, так и при выборе какое изменение следует внести (создание или разрушение связи) актор руководствуется некоторыми критериями оптимальности.

При изучении реальных социальных сетей в большинстве случаев не возможно наблюдать за динамикой сети непрерывно. Пусть исследователю доступны лишь несколько снимков состоянии сети.

Для анализа и предсказания динамики социальных сетей существует класс стохастических акторно-ориентированных моделей. Под потянем акторно-ориентированные подразумевается, что все изменения сети происходят из-за принятий решений акторами этой сети.

## Описание САОМ

Модель изменения связи состоит из двух компонент: время и выбор. Время изменения определяется в терминах возможности изменения, а не факта изменения. В случае если актора устраивает его текущее состояние он не будет изменять свои связи. Обозначим , как состояние сети в момент времени .

В любой момент времени в сети происходит не более одного изменения. Этими изменениями могут быть: создание или уничтожение связи. Вероятность изменения состояния сети зависит от её текущего состояния, и не зависит от истории изменений. Таким образом – непрерывная цепь Маркова.

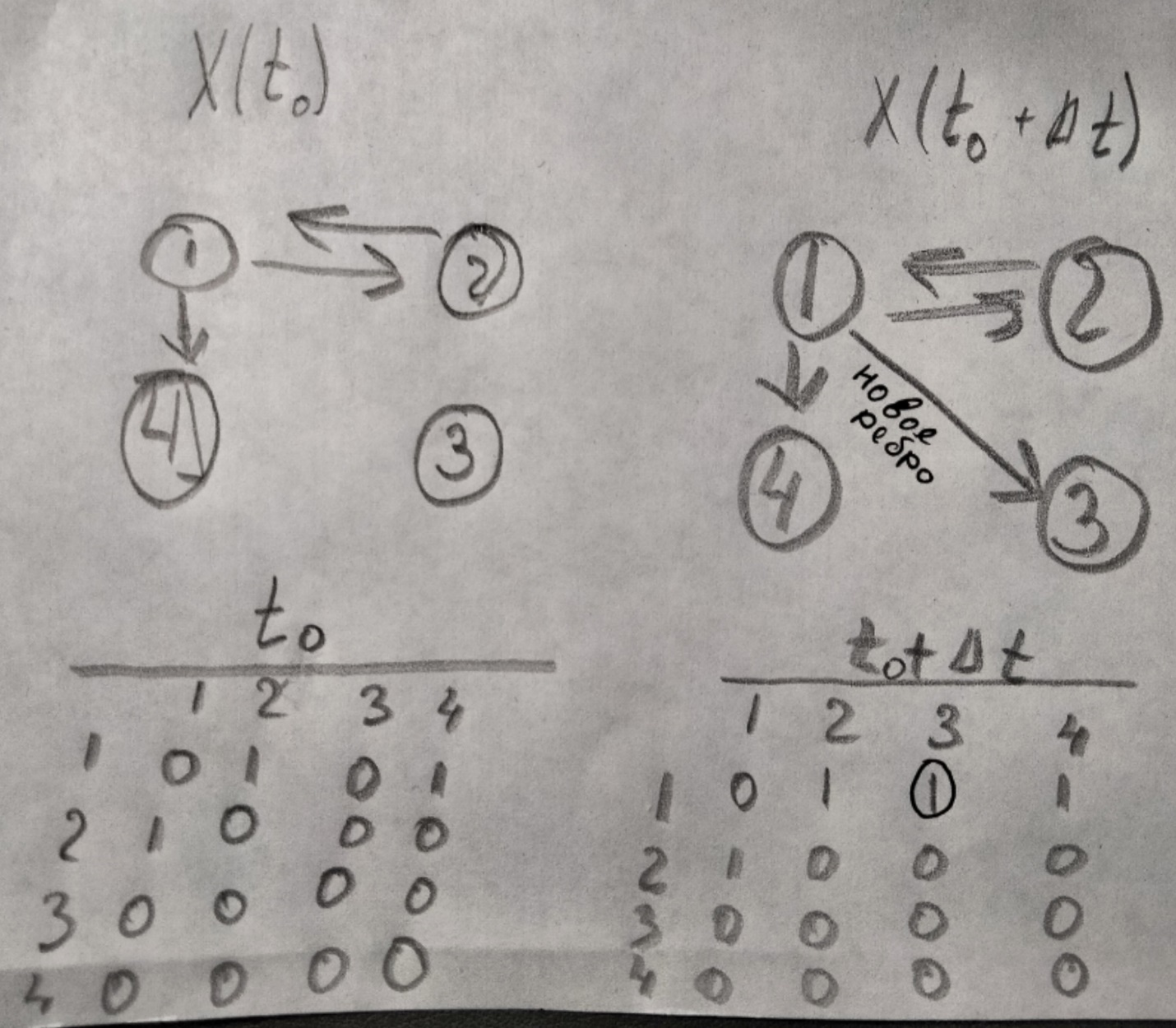


Рисунок 1 – Пример изменения процесса

Частота принятия решений актором *i* зависит от параметра . Обозначим .

|  |  |  |
| --- | --- | --- |
|  |  | (1) |

Моменты времени распределены в соответствии с экспоненциальным распределением.

|  |  |  |
| --- | --- | --- |
|  |  | (2) |

Эта формула соответствует модели «first past the post». Все участники имеют стохастическое время ожидания. Первый получивший возможность произвести изменение делает свой выбор и все начинается с начала, но уже в новом состоянии.

Совокупность выборов *i* актора образуют собой поток Пуассона с параметром .

Пример:

, с интенсивностью



Рисунок 2 – Поток выборов актора 1

, с интенсивностью



Рисунок 3 – Поток выборов актора 2

, с интенсивностью



Рисунок 4 – Поток выборов актора 3

, с интенсивностью



Рисунок 5 – Поток выборов актора n

Образуют собой совокупный поток с интенсивностью .



Рисунок 6 – Совокупный поток выборов всех акторов.

Сумма всех потоков – есть так же поток Пуассона с параметром , где времена между моментами скачков независимы и распределены экспоненциально.

Вероятность того, что *i* актор в выбранный промежуток времени примет решение о изменении сети:

|  |  |  |
| --- | --- | --- |
|  | , | (3) |

где остаточный член высшего порядка.

Вероятность того, что какой то актор в выбранный промежуток времени совокупного потока , где примет решение о изменении сети :

|  |  |  |
| --- | --- | --- |
|  | . | (4) |

У каждого актора есть вариантов выбора, и всего вариантов изменения сети. Построить\разрушить связь с кем то из других вершин, или ничего не делать. Для выбранного актора строится функция полезности для оценки вероятности выбора каждого из вариантов. Функция полезности определяет вероятности того, что при следующей смене связей данный актор перейдёт из состояния в .

Функция полезности:

|  |  |
| --- | --- |
|  | (5) |

где *i* – итератор актора принимающего решение о связи с актором *j*;

– текущее состояние сети, для графа и ,

– граф, который идентичен по всем рёбрам, кроме пары , для которой существование связи является переключателем , причём возможен случай когда это значит, что актор не изменит текущее состояние сети.

определена на множестве всех пар сети и , таких, что. В интерпретации полезности функция полезности может рассматриваться как чистая полезность, которую получает актор *i* от перехода из состояния . .

Предположим, что в момент времени , при текущей сети у актора появилась возможность изменения в состояние

|  |  |
| --- | --- |
|  | (6) |

Это распределение вероятностей выборов актора представляет собой модель множественного выбора при вариантов решений у вершины. Подробнее про построение функции полезности: В рассматриваемой модели функция полезности – есть линейная комбинация параметров и функции сети, где функция сети описывает то, как выглядит изменение сети из текущего состоянии в состояние новое с точки зрения актора *i*.

Функция сети:

|  |  |
| --- | --- |
| , | (7) |

где *k* – индекс параметра модели. Существует несколько реализаций функции сети [6] [4]. Некоторые из реализаций:

* Базовый компонент исходящих степеней:

|  |  |  |
| --- | --- | --- |
|  | (8) | |
|  | |  |

Он подобен постоянному члену в регрессионных моделях и всегда включается. Этот компонент балансирует между созданием и разрушением

связей. Учитывая предыдущее состояние x, следующее состояние либо

имеет на одну связь больше, либо на одну связь меньше, или они идентичны. Функция сети умножается на параметр , он показывает вклад функции полезности в вероятность создания новой связи, вклад в разрушение существующей связи . Таким образом, роль эффекта степени исходящих связей в модели заключается в вкладе в пользу создания

связей по сравнению с их разрушением. Обычно сети разрежены, поэтому

возможностей для создания связей гораздо больше, чем для их прерывания.

* Число взаимных связей:

|  |  |
| --- | --- |
|  | (9) |

Фундаментальный аспект почти всех направленных социальных сетей, потому что почти всегда существует некий обмен или другая взаимная зависимость.

Функция полезности:

|  |  |
| --- | --- |
|  | (10) |

Функция полезности описывает полезность перехода из  
состояния  в для *i* актора.

Вероятность изменения связи :

|  |  |
| --- | --- |
|  | (11) |
|  |  |

где – матрица смежности с элементами представляющая сеть в момент времени *t*.

Создание связи может быть как односторонней  
инициативой, так и совместной, в отличии от разрушения связи. Разрушение  
связи происходит в одностороннем порядке. Рассмотрим разные варианты  
моделирования возникновения связи.

– есть вероятность возникновения связи в разных сценариях.

1. Диктаторский сценарий
   1. Односторонняя инициатива:

|  |  |
| --- | --- |
|  | (12) |

|  |  |
| --- | --- |
| б) Двусторонняя инициатива: | (13) |

1. Взаимный сценарий:

, (14)

|  |  |
| --- | --- |
|  |  |

где позволяет учесть факт существования или отсутствия связи в рассматриваемый момент времени.

|  |  |
| --- | --- |
|  | (15) |

Скорость изменения сети в случае направленной сети:

|  |  |
| --- | --- |
|  | (16) |

Скорость изменения сети с односторонней инициативой, в случае ненаправленной сети:

|  |  |
| --- | --- |
|  | (17) |

Скорость изменения сети с двусторонней инициативой в случае ненаправленной сети:

|  |  |
| --- | --- |
|  | (18) |

– матрица интенсивностей переходов цепи Маркова .

Таким образом Стохастические Акторно-ориентированные модели (САОМ) работают по  
следующему алгоритму:

1. Зададим .
2. Генерируем в соответствии с экспоненциальным временем с параметром .
3. Если установить и остановиться.
4. Выбираем случайный используя распределение вероятностей .
5. Выбираем случайным образом , используя вероятности . - множество всех состояний в которое может измениться сеть  в следствии решения актора *i*.
6. Задать .
7. Задать .
8. Вернуться к шагу (2).

На принятие решения может влияет множество  
эффектов, и параметров при этих эффектах. Дальнейшей задачей является оценка этих параметров.

## Оценка параметров

Для оценки параметров используется имитационное моделирование.  
Реальная система заменяется моделью, и производится множество  
повторений алгоритма. Когда в среднем модель будет хорошо  
согласовываться с данными процесс оценки будет остановлен, и мы будем  
считать, что параметры оценены.  
Оценка параметров может быть произведена метод моментов [5].

Обозначим оцениваемый параметр как

|  |  |
| --- | --- |
| . | (19) |

Для каждого эффекта существует статистика, чувствительная к этому параметру. Для , влияющего на общее количество изменений используется Расстояния Хэмминга,

|  |  |
| --- | --- |
|  | (20) |

Для параметра , обозначающего насколько сильно скорость изменения актора *i* зависит от , существует статистика – ковариата или позиционная характеристика актора, такая как исходящая степень . Имеется в виду что скорость изменения может зависеть от пола, возраста, и тд.

|  |  |
| --- | --- |
| . | (21) |

Для функции полезности, где не зависит от , большие значения будут стремиться к сетям для который значение больше для любого актора *i*. Для оценки статистика имеет вид

|  |  |
| --- | --- |
| . | (22) |

Комбинируя статистики и используя предположение о марковской цепи для наблюдаемых данных оценочные уравнения имеют вид.

|  |  |
| --- | --- |
| ,, | (23) |
| ,, | (24) |
|  |  |
| ,  , | (25) |

где и номера элементов  и .

Для решения этих уравнений используется стохастическая оптимизация на основе алгоритма Роббинса-Монро. Алгоритм применяет многомерную версию алгоритма Роббинса-Монро с улучшениями, предложенными Поляком и Руппертом[10]. Техника "двойного усреднения" также применяется для улучшения результатов. Алгоритм реализован в пакете RSiena языка R.

Он состоит из трёх фаз:

* 1. Определение чувствительности ожидаемых статистик к параметрам.
  2. Обновляет параметры с использованием симуляции динамики сети.
  3. Используется для оценки приближения полученных уравнений и вычисления стандартных ошибок.

Для вычисления производных ожидаемых значений по отношению к параметрам используется метод функции оценки. Этот алгоритм является надёжным, но затратным по времени[10].

# Применение модели на реальных данных

## Объект исследования

Объектом исследования являются научные группы и команды Томского Государственного Университета, а конкретно данные, собранные о сотрудниках ТГУ, их научных публикациях, сетевых связях. Используя библиометрический анализ и стохастическое акторно-ориентированное моделирование, можно выявить факторы, влияющие на формирование научного взаимодействия внутри этих групп и команд.

Данные выгружены с веб сайта «ТГУ.Сотрудники». Используемая информация о авторах: Опыт работы и список публикаций автора, образующий сеть соавторства.

## Применимость САОМ в исследовании научных групп

Научное взаимодействие представляет собой процесс сотрудничества между учёными с целью достижения общей научной цели. Однако, как и любой другой процесс, научное взаимодействие может изменяться под влиянием различных факторов, которые могут способствовать или, наоборот, препятствовать его формированию и развитию.

Основными факторами, влияющими на формирование и разрушение научного взаимодействия, которые можно выделить в рамках взаимодействия в Томском Государственном Университете:

* Общая научная тематика: Учёные, работающие в одной области науки, имеют больше возможностей для сотрудничества, чем те, кто занимаются разными научными направлениями.
* Финансирование: Наличие достаточных финансовых ресурсов может способствовать формированию научных групп и команд. Однако, в случае недостатка финансирования, учёные могут ощутить трудности в формировании и поддержании научных контактов.
* Личностные факторы: Личностные характеристики учёных, такие как общительность, коммуникабельность и т.д., могут оказывать существенное влияние на формирование и поддержание научных связей.
* Технологический прогресс: Технологический прогресс может предоставить новые возможности для научного сотрудничества.

Наличие научной инфраструктуры: Наличие необходимой инфраструктуры, такой как лаборатории, библиотеки, научные журналы, тоже является важным фактором для формирования научных групп и команд.

САОМ - это метод социально-сетевого анализа, который позволяет моделировать и предсказывать эволюцию социальных сетей на основе поведения и взаимодействий индивидуальных акторов в них. САОМ используется для изучения процессов формирования и разрушения социальных связей в группах и командах, а также для оценки влияния различных факторов на эти процессы.

Применительно к исследованию научных групп и команд, САОМ может быть использован для моделирования процессов формирования научных связей между учёными в рамках группы или команды.

Используя САОМ возможно выявить такие факторы, влияющие на формирование или разрушение связей, как: влияние стажа на то, с какой частотой актор будет принимать решения относительно изменения сети, с какой частотой изменяется сеть, как влияет научное подразделение актора на динамику сети, и т.д.

Преимущества САОМ включают возможность моделирования процессов на уровне индивидуальных акторов, учёт динамической природы социальных связей, а также возможность оценки влияния различных факторов на эти процессы. Однако САОМ также имеет некоторые ограничения, такие как необходимость большой вычислительной мощности для моделирования и ограниченная возможность предсказания долгосрочных эффектов.

Таким образом, САОМ является мощным инструментом для исследования научных групп и команд, который может помочь в выявлении факторов, влияющих на формирование научных связей и определении эффективных мер по их сохранению а так же оптимизации научного взаимодействия.

## Формирование гипотез о динамике рассматриваемой сети

Формирование начальных гипотез, а так же формирование социологических выводов – это задача прикладных специалистов. Будут использованы некоторые предположения выдвинутые в подобном исследовании для национальной научной системы Словении [3].

Согласно исследованию Anuška Ferligoj, Luka Kronegger, Franc Mali, Tom A B Snijders, Patrick Doreian в журнале Scientometrics (2015, №104, с. 989–990):«В целом, мы предлагаем следующие механизмы влияния на научное соавторство: встраиваемость в сеть: соавторы соавторов стремятся стать соавторами соавторов; преимущественная привязанность: авторы предпочтительно ищут соавторов, у которых уже много соавторов; институциональная встроенность (принадлежность к одной исследовательской группе и одной научной дисциплине, возрастное сходство также может подпадать под эту категорию, поскольку означает принадлежность к общей когорте учёных, которые взаимодействуют друг с другом больше, чем представители разных когорт) и контрольные переменные, в частности стаж.»

## Описание модели

Необходимо учесть факт, что сеть соавторства является ненаправленной:

|  |  |  |
| --- | --- | --- |
|  | , | (26) |

где является элементом графа и обозначает связь .

В силу ненаправленности сети формирование связи между акторами происходит по одному из двух сценариев:

* 1. Односторонняя инициатива: Выбирается один актор *i*, который получает возможность произвести изменение.
* 2. Двусторонняя инициатива: Выбирается упорядоченная пара акторов , и получает возможность принять новое решение о связи ..

Процесс выбора моделируется по одному из двух сценариев:

D. Диктаторский(Dictatorial): Один актор может навязывать решение о связи другому актору.

M. Взаимный(Mutal): Оба актора дают согласие на существование связи между ними.

Согласно выдвинутым социологическим предположениям[3] формирование связи будет происходить согласно двусторонней инициативы, и оба актора будут давать согласие на создание связи.

Факторы влияющие на оценку полезности создания связи:

* + 1. Влияние количества исходящих связей
    2. Влияние транзитивных троек
    3. Влияние стажа на эффект транзитивных троек
    4. Влияние количества исходящих связей на частоту принятия решений

## 3.5 Оценка сформированных гипотез

### 3.5.1 Оцененные параметры при эффектах

Коэффициент сходимости модели достаточно низок 0.0508, для того, чтобы считать модель сходящейся[7]. Что означает, что выдвинутые предположения о виде модели корректно описывают представленные снимки сети. Так же абсолютное значение *Convergence t-ratio*[13] для каждого параметра меньше , что говорит о значимости каждого из эффектов.

|  |  |  |  |
| --- | --- | --- | --- |
| Эффект | Оценка | Стандартное отклонение | Convergence t-ratio |
| Интенсивность в 1 периоде | 0.2317 | 0.0053 |  |
| Интенсивность в 2 периоде | 0.1678 | 0.0091 |  |
| Влияние количества исходящих связей | -3.2147 | 0.0224 | 0.0157 |
| Влияние транзитивных троек | 0.4268 | 0.0140 | -0.0005 |
| Влияние стажа на эффект транзитивных троек | -0.0018 | 0.0006 | 0.0112 |
| Влияние количества исходящих связей на частоту принятия решений | 1.6070 | 0.0224 | 0.0157 |

Табл 1 Оценённые значения эффектов

### 3.5.2 Интерпретация полученных оценок параметров

Необходимо учитывать, что функция эволюции сети задана формулой:

|  |  |
| --- | --- |
| , | (27) |
| , | (28) |

где не зависит от причём может зависеть от для другого актора *j*.

Она отражает полезность для *i* актора. Используется для сравнения полезности различных изменений. С другой стороны когда *i* актор делает изменение из в , и два возможных результата шага то отношение вероятностей этих вариантов есть: . Следовательно сетевые параметры стоит расценивать как логарифмические.

Для оценки зависимых параметров необходимо учитывать, что возможные изменения в переменной , и оценки не должны выходить за пределы допустимого диапазона. Параметры целевой функции это вклады в логарифмические вероятности увеличения зависимой переменной на  
 единицу при увеличении эффекта на единицу. Где зависимая переменная либо , либо .

Пример:

Рассчитан эффект *среднее число связей того, к кому направлена связь* он равен . Это означает, что при сравнении двух акторов равных во всех отношениях, кроме того, что связи первого в среднем на 1 выше по шкале рассматриваемой ковариаты (Индекс Хирша, и тд), чем у второго, шанс увеличения значения ковариаты по сравнению с отсутствием изменений (в рамках одного шага относительно ковариаты) выше в раза, чем для второго.

Не любой эффект можно трактовать как в приведённом примере.

Пример:

Для эффекта *влияние* *количества исходящих связей* для актора . Пусть число связей, чьи значения меньше, равны или больше значения для *i*, обозначим через *a*, *b* и *c*. Обозначим диапазон (максимальное минус минимальное значение) через . Тогда, в случае минимального шага в отношении поведения, вклады суммарного эффекта сходства в лог-вероятности изменений определяются следующим образом и . Вклады для среднего  
эффекта сходства составляют и . Это показывает, что влияние связей на эффекты сходства зависит только от того, имеют ли они большие или меньшие значения, чем у рассматриваемого актора, а не от того, насколько эти значения больше. Это также показывает, что для эффектов сходства важна дисперсия ценностей связей, а не  
а не только их среднее значение, в то время как для эффекта среднего изменения имеет значение только среднее значение.

Не все эффекты могут быть описаны как изменение некоторой функции полезности. В таких случаях используются элементарные эффекты [8].

Элементарный эффект - это вклад в создание или поддержание связи, определённый напрямую, то есть без выражения его на основе изменения какой-либо функции оценки. Это означает, что элементарные эффекты более общие, чем эффекты оценки, и все эффекты могут быть представлены как элементарные эффекты. Однако для удобства интерпретации предпочтительно использовать формулировку функции оценки.

Элементарные эффекты могут применяться аналогично к созданию и поддержанию связи; или они могут применяться исключительно к созданию связи или исключительно к поддержанию связи.

Пример ситуации, когда невозможно описать как изменение функции полезности - тенденция к созданию связных троек (транзитивные замыкания). Если актор *i* является центральным в тройке, то связи, приводящие к замыканию, представлены как и . Однако первая связь подразумевает создание пути , в то время как вторая связь означает установление связи с актором *h*. Актор *h*, в свою очередь, делает тот же исходящий выбор для третьего актора *j*, что указывает на структурную эквивалентность; таким образом, это различные процессы.

Таким образом полученные оценки можно интерпретировать так:

* *Влияние количества исходящих связей* указывает на то, что с увеличением исходящих степеней снижается вероятность появления новой связи.
* *Зависимость скорости* (параметра для актора ) *от количества исходящих степеней*: c увеличением числа связей актор увеличивает частоту принятия решений.
* *Транзитивные тройки*: показывает количество транзитивных паттернов относительно актора . Для двух акторов, где каждый из них является соавтором с индивидуально, вероятность стать соавторами увеличивается раза.
* *Влияние стажа на эффект транзитивных троек* означает, что с увеличением стажа ослабевает эффект транзитивных троек.

# Построение имитационной модели

## Описание модели

Исследование реальных данных проводилось с использованием пакета RSiena. Пакет RSiena имеет очень большое количество разных реализаций компонентов модели, а так же распространяется с открытым исходный кодом и активна поддержка сообщества при необходимости написания собственных компонент.

Собственной компонентой может быть эффект. К примеру влияние временного ряда научных конференций, на степень кластеризации всей сети. Такой компонент должен включать в себя анализ временного ряда, dummy variable, и классический компонент из RSiena - связные тройки.

Для проверки корректности пользовательских эффектов необходима имитационная модель САОМ.

Была построена модель для направленной сети. Возможные результаты решения актора: создание новой свази, разрушение существующей связи, оставить сеть в том виде, в котором она есть в текущий момент времени. Разрушение связи происходит в одностороннем порядке, создание связи происходит в одностороннем порядке.

Вероятность изменения связи :

|  |  |
| --- | --- |
|  | (29) |

Функция полезности – есть линейная комбинация функций сети и их параметров. Для имитационной модели были взяты такие сетевые эффекты как:

* Влияние количества общих связей
* Влияние количества исходящих связей.

В таком случае функция полезности данной модели имеет вид

|  |  |
| --- | --- |
|  | (30) |

## Допущения

В силу того, что на практике интенсивности имеют зависимости от частных характеристик актора (ковариат), а в данной имитационной модели не используются эффекты связанные с ковариатами, то можно упростить выбор актора принимающего решение. Пусть для каждого актора интенсивности равны. В таком случае распределение вероятностей на шаге выбора актора принимавшего решение – равномерное.

Сеть представлена квадратной матрицей заполненной нулями и единицами. Начальное состояние сети на момент генерируется случайно, используя равномерное распределение.

Длины интервалов времени, между снимками сети одинаковые.

## Описание работы имитационной модели

В начале генерируется начальное состояние сети, снимок сети, матрица смежности, начально созданная нулевая матрица размерностью , где – количество акторов в сети, заполняется единицами, с некоторым, заданным, коэффициентом разреженности; далее стимулируется социальная динамика до момента времени начала следующего снимка сети .

Процесс повторяется дважды, используя каждый раз снимок полученный на предыдущем шаге. Таким образом в результате работы получается 3 снимка сети.

Подробнее про симуляцию сетевой динамики. Положим

задаётся пользователем, вектор параметров при эффектах задаётся пользователем. Генерируем экспоненциально, с параметром . Пока меньше используя распределение акторов выбирается актор принимающий решение о изменении сети в момент . Если меньше то генерируются все возможные состояния сети, в которые можно перейти из текущего состояния. Выбранный актор (актор ) используя распределение вероятностей переходов во все возможные варианты новой сети принимает решение о изменении сети. Процесс заканчивается когда .

Для симуляции принятия решения строится вектор полезностей для каждого актора, при зафиксированном *i* акторе. Каждый элемент этого вектора отражает полезность изменения сети из текущего значения в следующее (минишаг). Считается функция полезности для каждого возможного изменения, где конкретный вид функции полезности задан формулой (30)**.**

## Оценка работы имитационной модели

Для оценки корректности работы имитационной модели необходимо передать полученные снимки сети в пакет RSiena, описав модель. Далее коэффициент сходимости должен быть ниже [7], абсолютное значение Convergence t-ratio[13] для каждого параметра меньше , а так же оценка значения параметра должна находиться в диапазоне , где – среднеквадратическое отклонение.

Был построен алгоритм, запускающий имитационную модель, передающий результат работы (снимки сети) в пакет RSiena и сохраняющий результат оценки параметров, каждого параметра, коэффициент сходимости модели, и *Convergence t-ratio*. На рисунке 7 можно увидеть как возрастает количество связей ( голубых ребер) между акторами (красными вершинами) в результате симуляции сетевой динамики.

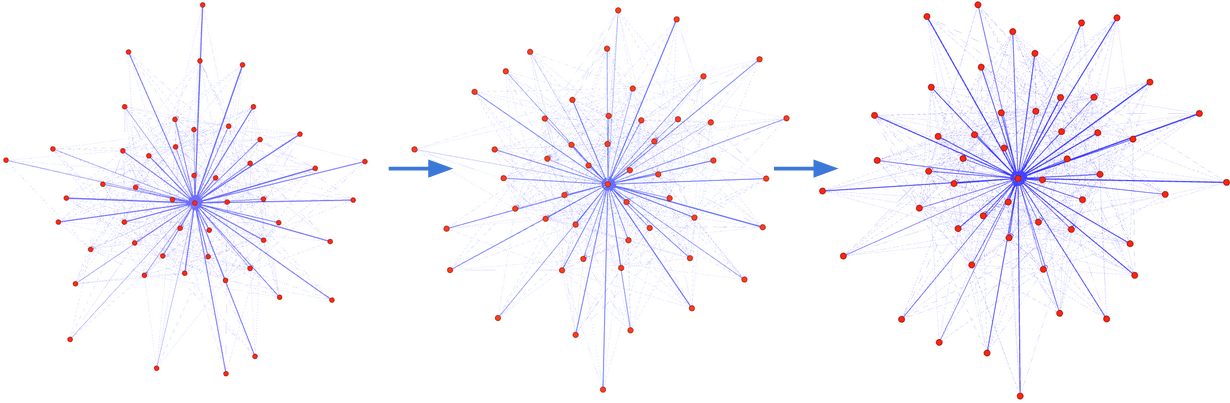


Рисунок 7 – Визуализация работы одной итерации алгоритма

Этот алгоритм был запущен 50 раз, и на каждой итерации выполнялись все необходимые условия, чтобы считать имитационную модель корректно описывающей заданную теоретическую модель САОМ.

Заданные начальные значения: для эффекта *влияние количества общих связей* – 1.5, для эффекта *влияние количества исходящих связей* – 1.8; интервалы времени – 2.5; интенсивности для акторов – 0.8; размерность сети – 50; коэффициент разреженности начального состояния сети – 0.75

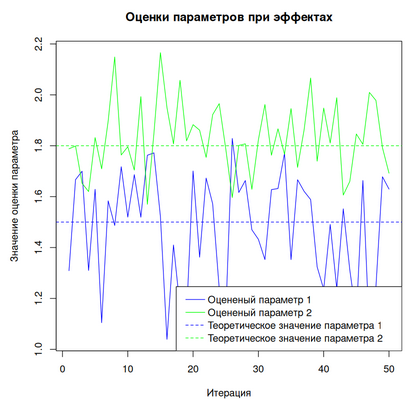


Рисунок 8 – График значений оценок параметров

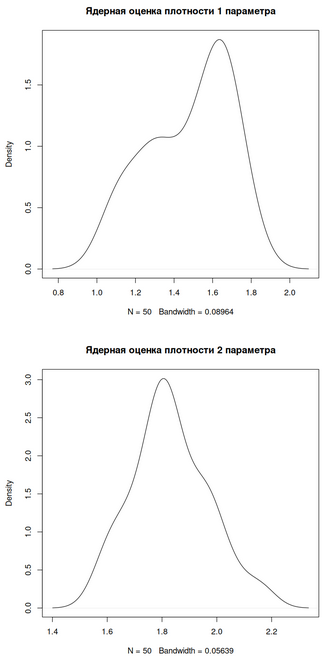


Рисунок 9 – Графики ядерных оценок плотности параметров

Выборочное математическое ожидание оценок параметра 1 эффекта *влияние количества общих связей* – 1.48, параметра 2 эффекта *влияние количества исходящих связей* – 1.83; средняя дисперсия оценок параметра 1 эффекта *влияние количества общих связей* – 0.267, параметра 2 эффекта *влияние количества исходящих связей* – 0.024; выборочная дисперсия оценок параметра 1 эффекта *влияние количества общих связей* – 0.047, параметра 2 эффекта *влияние количества исходящих связей* – 0.02

# Заключение

В рамках этой работы был изучен класс стохастических акторно ориентированных моделей. Поставленная цель, а именно построение и анализ стохастических акторно-ориентированных моделей была достигнута. Каждая из поставленных задач была выполнена.

САОМ, является новым математическим методом, использующий методы и подходи из дисциплин ставших уже классическими. В САОМ используется теория потоков, однако использование классических методов теории массового обслуживания, или теории потоков затруднительно, поскольку процесс изменение которого, моделируется – сеть, и при переходе от одного состояния сети к другому основным фактором является именно структура самой сети, её вид. Что требует использования инструментов теории графов. Так же использование сетевых эффектов очень похоже, на использование линейной регрессии, только вместо предикторов используется не набор данных, а поток принятий решений, что в свою очередь тоже затрудняет анализ сетевых данных, как задачу линейной регрессии.

Сетевые структуры хранят в себе большое количество информации, а динамика этих сетевых структур позволяет узнать многое о виде сети, о акторах, и о том как внешние факторы влияют на рассматриваемую сеть.

Результаты, полученные в ходе анализа социальной сети научного соавторства Томского государственного университета, могут быть положены в основу дальнейшего исследования данной сети, а также создания новых социологических гипотез в отношении сети соавторства ТГУ. Данные исследования могут быть использованы для оптимизации научного сообщества. В наши дни все большую популярность набирают междисциплинарные исследования, и в результате текущего исследования видно, что с увеличением стажа у авторов увеличивается вероятность междисциплинарной связи. Однако для дальнейшего исследования сети необходима разработка социологической теории о динамике сети и анализ поставленной теории. Полученные модели могут послужить не только для оптимизации взаимодействия научного сообщества, но и для оптимизации академического процесса студентов.

# Список использованных источников и литературы

1. Докука C.B., Валеева Д.Р. Статистические модели для анализа динамики социальных сетей в исследованиях образования // Вопросы образования. - 2015. - №1. - С. 201-213.
2. Tom Broekel, Pierre-Alexandre Balland, Martijn Burger, Frank Oort Modeling knowledge networks in economic geography: a discussion of four methods // The Annals of Regional Science. - 2014. - №53. - С. 423-452.
3. Anuška Ferligoj, Luka Kronegger, Franc Mali, Tom A B Snijders, Patrick Doreian Scientific collaboration dynamics in a national scientific system // Scientometrics. - 2015. - №104. - С. 985–1012.
4. Snijders, T.A.B., & Pickup, M. (2016). Stochastic Actor-Oriented Models for Network Dynamics. Retrieved June 10, 2016.
5. Snijders, T. A. B. he Statistical Evaluation of Social Network Dynamics // Sociological Methodology. - 2001. - №31. - С. 361-395.
6. Siena algorithms // Department of Statistics – University of Oxford URL: <https://www.stats.ox.ac.uk/~snijders/siena/Siena_algorithms.pdf> (дата обращения: 06.01.2024).
7. Manual for RSiena / M. R. Ruth. — Текст : электронный // University of Oxford: Department of Statistics; Nuffield College University of Groningen: Department of Sociology : [сайт]. — URL: <https://www.stats.ox.ac.uk/~snijders/siena/RSiena_Manual.pdf> (дата обращения: 14.01.2024).
8. Snijders The statistical evaluation of social network dynamics / Snijders, A T. — Текст : непосредственный // Sociolog- ical Methodology. — 2001. — № 31. — С. 361-395.

# Приложение А Код загрузчика данных

import requests

from bs4 import BeautifulSoup

import re

import csv

import pandas as pd

from scipy.sparse import coo\_matrix, save\_npz

def save\_get\_request\_to\_file(url, filename):

response = requests.get(url)

if response.status\_code == 200:

with open(filename, 'w') as file:

file.write(response.text)

print("Запрос успешно сохранен в файл", filename)

else:

print("Ошибка при выполнении GET-запроса:", response.status\_code)

def extract\_references(response\_text):

pattern = r"\?publicationId=(\d+)"

soup = BeautifulSoup(response\_text, 'html.parser')

references = []

td\_elements = soup.find\_all('td', class\_='py-2')

for td in td\_elements:

link\_element = td.find('a')

if link\_element:

href = link\_element.get('href')

match = re.search(pattern, href)

if match:

publication\_id = match.group(1)

references.append(publication\_id)

return references

def get\_and\_extract\_data(url):

response = requests.get(url)

if response.status\_code == 200:

publication\_titles = extract\_references(response.text)

return publication\_titles

def check\_page\_exist(url):

response = requests.get(url)

if response.status\_code == 200:

soup = BeautifulSoup(response.text, 'html.parser')

object\_div = soup.find('div', class\_='alert alert-primary')

object\_h1 = soup.find('h1', class\_='mb-1 row justify-content-between')

if object\_div and object\_div.get\_text(strip=True) == 'Нет данных':

return False

else:

return True

else:

print("Ошибка при выполнении запроса:", response.status\_code)

def extract\_unique\_publication\_ids(input\_file, output\_file):

unique\_publication\_ids = set()

with open(input\_file, 'r') as csv\_file:

reader = csv.reader(csv\_file, delimiter=';')

next(reader) # Пропускаем заголовок

for row in reader:

publication\_ids = row[1].strip().split(', ')

for publication\_id in publication\_ids:

unique\_publication\_ids.add(publication\_id)

with open(output\_file, 'w', newline='') as csv\_file:

writer = csv.writer(csv\_file, delimiter=';')

writer.writerow(['Publication ID'])

for publication\_id in unique\_publication\_ids:

writer.writerow([publication\_id])

def sort\_csv(input\_file, output\_file, sort\_column):

with open(input\_file, 'r') as csv\_file:

reader = csv.reader(csv\_file)

header = next(reader)

sorted\_rows = sorted(reader, key=lambda row: row[sort\_column])

with open(output\_file, 'w', newline='') as csv\_file:

writer = csv.writer(csv\_file)

writer.writerow(header)

writer.writerows(sorted\_rows)

def build\_adjacency\_matrix\_from\_csv(csv\_file\_path, mtx\_file\_path):

adjacency\_matrix = []

publication\_ids = set()

with open(csv\_file\_path, 'r') as csv\_file:

reader = csv.reader(csv\_file, delimiter=';')

next(reader) # Пропускаем заголовок

for row in reader:

author\_id = int(row[0])

publications = list(map(int, row[1].strip().split(',')))

adjacency\_row = [author\_id, publications]

adjacency\_matrix.append(adjacency\_row)

publication\_ids.update(publications)

publication\_list = sorted(publication\_ids)

num\_authors = len(adjacency\_matrix)

num\_publications = len(publication\_list)

with open(mtx\_file\_path, 'w') as mtx\_file:

mtx\_file.write("%%MatrixMarket matrix coordinate integer general\n")

mtx\_file.write(f"{num\_authors} {num\_publications} {sum(len(row[1]) for row in adjacency\_matrix)}\n")

for i, adjacency\_row in enumerate(adjacency\_matrix):

author\_id = adjacency\_row[0]

publications = adjacency\_row[1]

for publication\_id in publications:

mtx\_file.write(f"{i + 1} {publication\_list.index(publication\_id) + 1} 1\n")

def extract\_year\_from\_publication\_url(url): #

response = requests.get(url)

soup = BeautifulSoup(response.text, 'html.parser')

year\_element = soup.find('td', string='Год')

if year\_element:

year = year\_element.find\_next\_sibling('td').text

return year

else:

print("Год не найден")

def group\_by\_authorID\_and\_append\_publicationID\_by\_coma(csvfile):

df = pd.read\_csv(csvfile, delimiter=';')

# Группировка по 'author\_id' и объединение 'publication\_id' через запятую

grouped\_df = df.groupby('author\_id')['publication\_id'].apply(lambda x: ','.join(x.astype(str))).reset\_index()

# Сохранение результата в новый CSV файл

grouped\_df.to\_csv('new\_' + csvfile, index=False, sep=';')

def find\_year\_by\_publication\_id(publication\_id, ):

with open('Data/PublicationID\_Year.csv', 'r') as file:

reader = csv.DictReader(file, delimiter=';')

for row in reader:

if row['Publication\_ID'] == str(publication\_id):

return row['Year']

return None

#######################################################

# Добавление полей стажа и факультета #

#######################################################

# # Функция для получения значения Experience по AuthorID\_TSU

# def getExp(author\_id):

# url = f'https://persona.tsu.ru/Home/UserProfile/{author\_id}'

# response = requests.get(url)

# soup = BeautifulSoup(response.text, 'html.parser')

# divs = soup.find\_all('div')

# for i, div in enumerate(divs):

# if div.text.strip() == 'Общий стаж' and i + 1 < len(divs):

# return divs[i + 1].text.strip()

# return 0

# def getSubdiv(author\_id):

# url = f'https://persona.tsu.ru/Home/UserProfile/{author\_id}'

# response = requests.get(url)

# soup = BeautifulSoup(response.text, 'html.parser')

# spans = soup.find\_all('span', itemprop='name')

#

# if len(spans) >= 2:

# return spans[1].text.strip()

# else:

# return None

#

# # Чтение исходной таблицы из CSV файла

# df = pd.read\_csv('Data/AuthorID\_NEW\_AuthorID\_TSU\_PublicationsID.csv', delimiter=';')

#

# # Создание новой таблицы с необходимыми полями

# new\_df = pd.DataFrame(columns=['AuthorID\_NEW', 'AuthorID\_TSU', 'Experience', 'Subdivision', 'PublicationsID'])

#

# # Итерация по исходной таблице и заполнение новой таблицы

# for index, row in df.iterrows():

# author\_id\_new = row['AuthorID\_NEW']

# author\_id\_tsu = row['AuthorID\_TSU']

# publications\_id = row['PublicationsID']

# experience = getExp(author\_id\_tsu)

# subdivision = getSubdiv(author\_id\_tsu)

#

# new\_row = {'AuthorID\_NEW': author\_id\_new, 'AuthorID\_TSU': author\_id\_tsu, 'Experience': experience,

# 'Subdivision': subdivision, 'PublicationsID': publications\_id}

# new\_df = new\_df.\_append(new\_row, ignore\_index=True)

#

# # Сохранение новой таблицы в CSV файл

# new\_df.to\_csv('AuthorID\_NEW\_AuthorID\_TSU\_Experience\_Subdivision\_PublicationsID.csv.csv', index=False, sep=';')

# Соединил таблицы в одну

# years\_df = pd.read\_csv('Data/Years\_of\_publication.csv')

# publications\_df = pd.read\_csv('Data/Publication\_ID.csv')

#

# # Combine the columns into a new dataframe

# new\_table\_df = pd.DataFrame({

# 'new\_column': publications\_df['Publication\_ID'].astype(str) + ';' + years\_df['Year\_of\_publication'].astype(str)

# })

#

# # Save the new table as a CSV file

# new\_table\_df.to\_csv('new\_table.csv', index=False)

# with open('Data/AuthorID\_PublicationID.csv', 'w') as file:

# writer = csv.writer(file)

# # Запись заголовков столбцов

# headers = ['Author ID', 'Publication ID']

# writer.writerow(headers)

# file.close()

#

#

# with open('Data/sorted\_Employees\_with\_publications.txt', 'r') as input\_file:

# id\_s = input\_file.readlines()

# for id\_value in id\_s:

# id\_value = id\_value.strip()

# i = 1

# while True:

# url = f"https://persona.tsu.ru/Publications/Index/{id\_value}?page={i}"

# if not check\_page\_exist(url):

# break # Прерываем цикл, если страница не существует

# with open('Data/AuthorID\_PublicationID.csv', 'a') as file:

# writer = csv.writer(file)

# data\_row = [id\_value, get\_and\_extract\_data(url)]

# writer.writerow(data\_row)

# file.close()

# i += 1

# Поиск дат публикаций

# def find\_author\_id(publication\_id, author\_publications):

# for row in author\_publications:

# author\_id = row[0]

# publications = row[1].split(', ')

# if publication\_id in publications:

# return author\_id

# return None

#

# with open('Data/PublicationID(1).csv', 'r') as publication\_file:

# publication\_reader = csv.reader(publication\_file)

# next(publication\_reader) # Пропуск заголовка

# for row in publication\_reader:

# publication\_id = row[0]

#

# # Чтение файла AuthorID\_PublicationsID.csv

# with open('Data/AuthorID\_PublicationsID.csv', 'r') as author\_file:

# author\_reader = csv.reader(author\_file, delimiter=';')

# next(author\_reader) # Пропуск заголовка

# author\_publications = list(author\_reader)

#

# # Поиск Author ID

# author\_id = find\_author\_id(publication\_id, author\_publications)

#

# if author\_id:

# with open('Data/Years\_of\_publication.csv','a') as years\_file:

# writer = csv.writer(years\_file)

# row = extract\_year\_from\_publication\_url(f'https://persona.tsu.ru/Publications/Info/{author\_id}?publicationId={publication\_id}')

# writer.writerow(row)

# years\_file.close()

# else:

# print(f'Publication ID: {publication\_id}, Author ID not found')

#

#######################################################

# Разделение на 3 таблицы по датам публикации #

# до 2016 | 2016-2019 | 2020-2023 #

#######################################################

#

# with open('Data/AuthorID\_NEW\_AuthorID\_TSU\_PublicationsID.csv', 'r', newline='') as input\_file:

# csv\_reader = csv.reader(input\_file, delimiter=';')

# next(csv\_reader) # Пропускаем заголовок

#

# for row in csv\_reader:

# publications = []

# publications.extend(row[2].split(','))

# for publication in publications:

# try:

# if int(find\_year\_by\_publication\_id(publication)) < 2016:

# with open('before\_2016.csv', 'a') as before\_2016\_file:

# writer = csv.writer(before\_2016\_file)

# write\_row = row[0] + ';' + publication

# # print(write\_row)

# writer.writerow(write\_row)

# before\_2016\_file.close()

#

# if int(find\_year\_by\_publication\_id(publication)) >= 2016 and int(

# find\_year\_by\_publication\_id(publication)) <= 2019:

# with open('2016-2019.csv', 'a') as \_2017\_2019\_file:

# writer = csv.writer(\_2017\_2019\_file)

# write\_row = row[0] + ';' + publication

# # print(write\_row)

# writer.writerow(write\_row)

# \_2017\_2019\_file.close()

#

# if int(find\_year\_by\_publication\_id(publication)) > 2019:

# with open('after\_2021.csv', 'a') as after\_2021\_file:

# writer = csv.writer(after\_2021\_file)

# write\_row = row[0] + ';' + publication

# # print(write\_row)

# writer.writerow(write\_row)

# after\_2021\_file.close()

# except Exception:

# # with open('Data/there\_have\_been\_problems.txt', 'a') as exception\_file:

# # txt\_string = f'Publication ID: {publication}\n {row}'

# # exception\_file.write(txt\_string)

# # exception\_file.close()

# print(f'Publication ID: {publication}\n {row}')

# # print(publication)

###########################################################

####### Разбиение на 2 файла с ОДИНАКОВОЙ размерностью ####

###########################################################

# file1\_path = 'Data/PublicationID\_Year.csv'

# data1 = {}

# with open(file1\_path, 'r') as file1:

# reader = csv.DictReader(file1, delimiter=';')

# for row in reader:

# publication\_id = int(row['Publication\_ID'])

# year = int(row['Year'])

# data1[publication\_id] = year

#

# file2\_path = 'Data/AuthorID\_NEW\_AuthorID\_TSU\_PublicationsID.csv'

# data2 = {}

# with open(file2\_path, 'r') as file2:

# reader = csv.DictReader(file2, delimiter=';')

# for row in reader:

# author\_id\_new = int(row['AuthorID\_NEW'])

# publication\_ids = row['PublicationsID'].split(',')

# publication\_ids = [int(publication\_id) for publication\_id in publication\_ids]

# data2[author\_id\_new] = publication\_ids

# # Создание матрицы смежности для периода до 2016 года

# adjacency\_matrix\_before\_2016 = [[0] \* len(data2) for \_ in range(len(data2))]

#

# for author\_id\_new, publication\_ids in data2.items():

# for publication\_id in publication\_ids:

# if publication\_id in data1 and data1[publication\_id] < 2016:

# for other\_author\_id\_new, other\_publication\_ids in data2.items():

# if other\_author\_id\_new != author\_id\_new and publication\_id in other\_publication\_ids:

# adjacency\_matrix\_before\_2016[author\_id\_new-1][other\_author\_id\_new-1] = 1

# adjacency\_matrix\_before\_2016[other\_author\_id\_new-1][author\_id\_new-1] = 1

#

# # Создание матрицы смежности для периода 2016-2019 годы

# adjacency\_matrix\_2016\_2019 = [[0] \* len(data2) for \_ in range(len(data2))]

#

# for author\_id\_new, publication\_ids in data2.items():

# for publication\_id in publication\_ids:

# if publication\_id in data1 and 2016 <= data1[publication\_id] <= 2019:

# for other\_author\_id\_new, other\_publication\_ids in data2.items():

# if other\_author\_id\_new != author\_id\_new and publication\_id in other\_publication\_ids:

# adjacency\_matrix\_2016\_2019[author\_id\_new-1][other\_author\_id\_new-1] = 1

# adjacency\_matrix\_2016\_2019[other\_author\_id\_new-1][author\_id\_new-1] = 1

#

# # Создание матрицы смежности для периода после 2019 года

# adjacency\_matrix\_after\_2019 = [[0] \* len(data2) for \_ in range(len(data2))]

#

# for author\_id\_new, publication\_ids in data2.items():

# for publication\_id in publication\_ids:

# if publication\_id in data1 and data1[publication\_id] > 2019:

# for other\_author\_id\_new, other\_publication\_ids in data2.items():

# if other\_author\_id\_new != author\_id\_new and publication\_id in other\_publication\_ids:

# adjacency\_matrix\_after\_2019[author\_id\_new-1][other\_author\_id\_new-1] = 1

# adjacency\_matrix\_after\_2019[other\_author\_id\_new-1][author\_id\_new-1] = 1

#

# # Сохранение матрицы смежности для периода до 2016 года в файл .dat

# matrix\_before\_2016\_path = 'путь\_к\_файлу\_матрицы\_до\_2016.dat'

# with open(matrix\_before\_2016\_path, 'w') as file:

# for row in adjacency\_matrix\_before\_2016:

# file.write(' '.join(map(str, row)) + '\n')

#

# # Сохранение матрицы смежности для периода 2016-2019 годы в файл .dat

# matrix\_2016\_2019\_path = 'путь\_к\_файлу\_матрицы\_2016\_2019.dat'

# with open(matrix\_2016\_2019\_path, 'w') as file:

# for row in adjacency\_matrix\_2016\_2019:

# file.write(' '.join(map(str, row)) + '\n')

#

# # Сохранение матрицы смежности для периода после 2019 года в файл .dat

# matrix\_after\_2019\_path = 'путь\_к\_файлу\_матрицы\_после\_2019.dat'

# with open(matrix\_after\_2019\_path, 'w') as file:

# for row in adjacency\_matrix\_after\_2019:

# file.write(' '.join(map(str, row)) + '\n')

##################################################################

Сохраниение инфорации о стаже на моменты срезов

##################################################################

# def create\_dat\_file(year, data):

# filename = str(year) + ".dat"

# with open(filename, "w") as file:

# for row in data:

# author\_id\_new = row["AuthorID\_NEW"]

# experience = int(row["Experience"])

# exp\_by\_year = experience - (2023 - year)

# file.write(f"{exp\_by\_year}\n")

#

# # Чтение CSV файла

# with open("Data/AuthorID\_NEW\_AuthorID\_TSU\_Experience\_Subdivision\_PublicationsID.csv", "r") as file:

# csv\_reader = csv.DictReader(file, delimiter=";")

# data = list(csv\_reader)

#

# # Создание .dat файлов

# create\_dat\_file(2016, data)

# create\_dat\_file(2019, data)

# create\_dat\_file(2023, data)

# Приложение Б Код имитационной модели

Generate\_Network\_t0 <- function(size = 40, sparsity = 0.7) {

network <- matrix(0, nrow=size, ncol=size)

for (i in 1:size) {

for (j in 1:size) {

if (runif(1) > sparsity) {

network[i, j] <- 1

}

}

}

return(network)

}

Network\_function\_1 <- function(i, Network) { # outdegree (density)

net <- sum(Network[i, ])

return(net)

}

Network\_function\_2 <- function(i, Network) { # recip

net <- 0

for(j in 1:length(Network[1,])){

if(i != j){

net <- net + Network[i,j]\*Network[j,i]

}

}

return(net)

}

Network\_function\_3 <- function(i, Network) { # transitive triplets (transTrip)

net <- 0

for (j in 1:length(Network[i, ])) {

for(h in 1:length(Network[, j])){

if(i!=j & j!=h & i!=h){

net <- net + (Network[i,j] \* Network[i,h] \* Network[h,j])

}

}

}

return(net)

}

Generate\_Possible\_Ministep\_Matrix\_vector <- function(i, Network) {

Networks <- list()

for (j in 1:length(Network[i, ])) {

if (i != j) {

TMP\_Net <- Network

if ( TMP\_Net[i, j] == 0){ TMP\_Net[i, j] <- 1 }

else if ( TMP\_Net[i, j] == 1){ TMP\_Net[i, j] <- 0 }

Networks[[j]] <- TMP\_Net

}

else{

Networks[[j]] <- Network

}

}

return(Networks)

}

Distribution\_actors <- function(i, Network, beta, Possible\_Ministep\_Matrix\_vector) {

Util <- numeric(length(Network[i, ]))

total\_util <- 0

for (h in 1:length(Network[i, ])) {

total\_util <- total\_util + exp(Utility(Network, beta, i, h, Possible\_Ministep\_Matrix\_vector))

}

Util\_j <- numeric(length(Network[i, ]))

for (j in 1:length(Network[i, ])) {

Util\_j[j] <- exp(Utility(Network, beta, i, j, Possible\_Ministep\_Matrix\_vector))

Util[j] <- Util\_j[j] / total\_util

}

return(Util)

}

Utility <- function(Network, beta, i, j, Possible\_Ministep\_Matrix\_vector) {

NetIJ <- Possible\_Ministep\_Matrix\_vector

util <- (beta[1] \* Network\_function\_1(i, NetIJ[[j]])) + (beta[2] \* Network\_function\_2(i, NetIJ[[j]])) #+ (beta[3] \* Network\_function\_3(i, NetIJ[[j]]))

return(util)

}

Simulator <- function(Network0, t = 0, LambdaV, total\_Lambda, T, beta) {

Network <- Network0

while (t < T) {

deltaT <- rexp(1, rate=total\_Lambda)

# i <- sample(1:length(Network0[1,]), size=1, prob=(LambdaV / total\_Lambda))

i <- sample(1:length(Network0[1,]), size=1)

Possible\_Ministep\_Matrix\_vector <- Generate\_Possible\_Ministep\_Matrix\_vector(i, Network)

prob\_vector <- 0

prob\_vector <- Distribution\_actors(i, Network, beta, Possible\_Ministep\_Matrix\_vector)

j <- sample(1:length(Network0[i, ]), size=1, prob=prob\_vector)

if (j != i ){

if ( Network[i, j] == 0){ Network[i, j] <- 1 }

else if ( Network[i, j] == 1){ Network[i, j] <- 0 }

}

t <- t + deltaT

}

return(Network)

}

# Установка начальных параметров

size <- 50

Network0 <- Generate\_Network\_t0(size = size, sparsity = .75)

LambdaV <- rep(.8, size)

total\_Lambda <- sum(LambdaV)

# beta <- rexp(n = 2)

beta <- c(1.5, 1.8, 2.5)

T <- 2.5

# Использование имитационной модели

Network1 <- Simulator(Network0, t = 2.5, LambdaV, total\_Lambda, 5, beta)

Network2 <- Simulator(Network1, t = 5, LambdaV, total\_Lambda, 7.5, beta)

# Приложение В Вывод оценки параметров имитационной модели пакетом RSiena

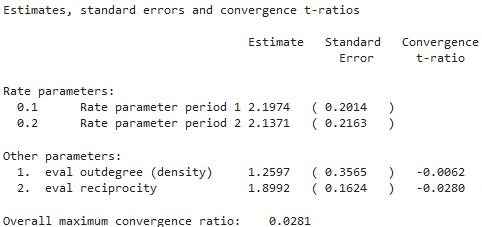


Рисунок 5 - Вывод оценки параметров имитационной модели пакетом RSiena

# Приложение Г Вывод оценки параметров анализа социальной сети научного сообщества ТГУ

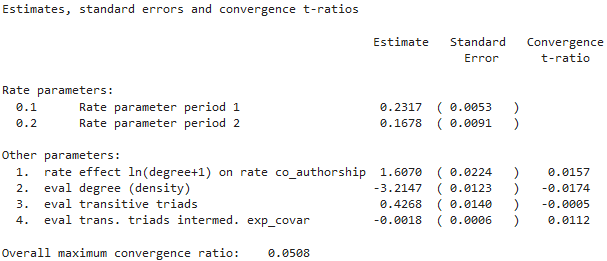


Рисунок 6 - Вывод оценки параметров социальной сети научного соавторства ТГУ