КАЗАНСКИЙ (ПРИВОЛЖСКИЙ) ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ
Институт фундаментальной медицины и биологии
Кафедра биохимии, биотехнологии и фармакологии

Власенкова Р.А., Савенкова Д.В., Нургалиева А.К., Киямова Р.Г.

АНАЛИЗ ВЗВЕШЕННЫХ СЕТЕЙ КО-ЭКСПРЕССИИ ГЕНОВ

Учебно-методическое пособие

Казань - 2024

УДК 51-76; 573
ББК 28.00

Печатается по решению Учебно-методической комиссии Института фундаментальной медицины и биологии Казанского (Приволжского) федерального университета
Протокол №19 от 19.06.2024 г.
Заседания кафедры биохимии, биотехнологии и фармакологии
Протокол №14 от 05.06.2024 г.

Рецензенты:
канд. биол. наук, ст. науч. сотр. Козлова О.С.
PhD, вед. науч. сотр. Булатов Э.Р.

Анализ взвешенных сетей ко-экспрессии генов: учебно-методическое пособие / Р.А. Власенкова, Д.В. Савенкова, Нургалиева А.К., Киямова Р.Г. – Казань: Изд-во Казан. ун-та, 2024. – 41 с.

В учебно-методическом пособии систематизированы методы построения взвешенных сетей ко-экспрессии генов, анализа сетей и их визуализации с использованием свободной программной среды вычислений R и графической оболочки Posit. В пособие включены базы данных, содержащие данные об экспрессии генов, методы определения корреляции экспрессии генов, построение взвешенных сетей ко-экспрессии, анализ центральности и нахождение модулей сетей, а также сравнения сетей на основе метода иерархической кластеризации и нахождения коэффициента кофенетической корреляции.

Каждый раздел пособия снабжён примерами реализации методов на языке R и заданиями с комментариями и для самостоятельного решения.

Пособие предназначено для использования в курсе магистратуры «Системная биология», а также для студентов и аспирантов медико-биологического профиля, имеющий опыт в программировании на языке R, при подготовке курсовых и выпускных квалификационных работ и проведении научных исследований.

УДК 51-76; 573
ББК 28.00

СОДЕРЖАНИЕ

Введение

Понимание функций и регуляторных механизмов генов является одной из центральных проблем биологии. Основой для их предсказания может служить ко-экспрессия генов, поэтому она является важной концепцией в системной биологии и биоинформатике. Одним из наиболее удобных способов представления такой информации является сеть, поскольку она показывает как объекты, так и их взаимодействия.

Сеть состоит из набора узлов и рёбер, обозначающих отношения между узлами. Узлами могут быть гены, РНК, белки, метаболиты и другие молекулы; рёбра же указывают на то, что между биомолекулами существуют химические или физические взаимодействия, химические реакции или отношения совместной экспрессии. В зависимости от типов узлов и значения рёбер сеть может быть направленной или ненаправленной, взвешенной или невзвешенной. Сеть является направленной, когда ребро между узлами имеет направление от одного узла к другому и представлено стрелкой. Их можно использовать для графического представления направленных отношений между объектами, например, ингибирования или активации. В ненаправленных же сетях рёбра указывают на неориентированные отношения между узлами. В невзвешенных сетях взаимодействие между генами бинарное (т.е. гены либо связаны, либо нет). Во взвешенной сети связи между генами имеют значения веса, которые указывают на силу взаимодействия. Одними из самых распространённых видов сетей являются сети ко-экспрессии, белок-белковых взаимодействий, регуляции

генов, а также сигнальные и метаболические сети.

Сеть ко-экспрессии позволяет изучать существующие закономерности экспрессии генов и определяет, какие гены показывают скоординированный паттерн в группе образцов. Её построение и анализ происходит в несколько этапов. На первом этапе построения сети определяются индивидуальные отношения между генами. Такие взаимодействия определяются с помощью расчёта коэффициентов корреляции. Наиболее популярными способами расчёта коэффициентов корреляции являются метод Пирсона и метод Спирмена. Коэффициент корреляции Пирсона более распространён и имеет большую статистическую мощность, однако подходит только для линейной зависимости и чувствителен к выбросам. Его альтернативой является коэффициент корреляции Спирмена, который можно применять к данным с нелинейной зависимостью.

На втором этапе значения используются для построения сети, в которой каждый узел представляет ген, а каждое ребро показывает наличие и силу ко-экспрессии. Последним этапом становится обнаружение генетических модулей (групп совместно экспрессируемых генов) и проведение кластеризации.

Сети ко-экспрессии могут быть взвешенными и невзвешенными, а также знаковыми и беззнаковыми. Корреляция принимает значения от -1 до 1. В беззнаковой сети используются абсолютные значения корреляции, что означает, что два гена с отрицательной корреляцией будут считаться коэкспрессированными. Знаковая сеть

масштабирует значения корреляции между 0 и 1, и значения <0,5 указывают на отрицательную корреляцию.

Анализ сети ко-экспрессии может служить инструментом для идентификации молекулярных мишеней в качестве потенциальных молекулряных маркеров. Благодаря ему можно выяснить изменения в паттернах совместной экспрессии генов. Подобный анализ может расширить понимание патогенеза того или иного заболевания и дать ценную информацию для исследования потенциальных биомаркеров, которые можно использовать для диагностики и терапии различных заболеваний.

Код, написанный для этого пособия и все иллюстрации можно найти по ссылке:
https://github.com/RamiliaV/weighted-gene-co-expression-network-analysis-study-guide
Или можете использовать данный QR-код:

В репозитории на сайте GitHub можно найти все

скрипты, указанные в заданиях. В скриптах указаны комментарии с номерами шагов для лучшей навигации по командам.

Код можно запускать построчно, выделяя его и нажимая комбинацию клавиш CTRL+Enter. Либо запустить каждый скрипт полностью, как указано в заданиях.

Тема 1. Базы данных, содержащие информацию об экспрессии генов

Базы данных по экспрессии генов становятся важными информационными ресурсами, способными хранить данные безопасным и в то же время легко извлекаемым способом. В основном, подобные хранилища содержат количественные данные об экспрессии генов, полученные методами ДНК-микрочипов и секвенирования РНК.

Анализ экспрессии генов методом ДНК-микрочипов выполняется путём гибридизации различных ДНК — той, которая нанесена на микрочип, и содержащейся в образцах, дополнительно наносимых на микрочип. Эти образцы представляют собой исследуемый биоматериал, который содержит флуоресцентные метки. Далее микрочип сканируют конфокальным лазером и фиксируют величины флуоресцентных сигналов. Технология позволяет сразу сохранять данные на электронный носитель для дальнейшей обработки.

Метод секвенирования РНК (RNA-Seq) основан на секвенировании нового поколения с использованием фрагментов кДНК. Данный метод совершил прорыв в изучении транскриптомов благодаря ряду преимуществ перед ранее предложенными методами. Во-первых, он позволяет детектировать транскрипты как с установленной, так и с неизвестной последовательностью. Во-вторых, RNA-seq имеет очень широкий динамический диапазон (6 порядков). По сравнению с микрочипами он имеет низкий уровень фонового сигнала. Благодаря методу глубокого

секвенирования RNA-seq способен различать индивидуальные однонуклеотидные полиморфизмы (SNPs). Кроме того, с его помощью можно количественно сравнивать экспрессию генов в разных образцах. RNA-seq позволяет получать как качественную, так и количественную информацию.

Базы данных могут предоставлять информацию о наборах данных или об уровнях экспрессии генов в индивидуальных образцах. Также базы данных содержат информацию об образцах: организм, тип ткани, заболевание. Помимо этого, в интерфейс базы могут быть интегрированы инструменты анализа и визуализации данных.

Примеры баз данных с откртым доступом онлайн:

- **Gene Expression Omnibus** − это общедоступное функциональное хранилище геномных данных, поддерживающее передачу данных, совместимых со стандартами MIAME. В рекомендациях MIAME (Minimum Information About a Microarray Experiment − Минимальная информация об эксперименте с микрочипами) описывается минимальная информация, которая должна быть включена при описании исследования с использованием микрочипов или секвенирования. Хранилище предоставляет инструменты, помогающие пользователям запрашивать и загружать эксперименты и профили экспрессии генов, а также анализировать данные.
- **ArrayExpress** − хранилище данных, полученных в результате высокопроизводительных экспериментов по функциональной геномике, и предоставляет данные для повторного использования

исследовательскому сообществу. В соответствии с руководящими принципами сообщества, исследование обычно содержит метаданные, такие как подробные аннотации к образцам, протоколы, обработанные данные и необработанные исходные данные.

- **cBioPortal** − портал по геномике рака обеспечивает визуализацию, анализ и загрузку крупномасштабных наборов данных по геномике злокачественных опухолевых заболеваний. Портал предоставляет данные о мутациях и уровне экспрессии генов в образцах различных типов опухолей.

В этом пособии рассмотрена работа с базой данных Gene Expression Omnibus.

Алгоритм поиска наборов данных следующий:

1. Перейдите на страницу https://www.ncbi.nlm.nih.gov/geo/
2. В строке поиска укажите ключевые слова на английском языке. Перейдите по ссылке с результатами.
3. К выдаче результатов можно применить фильтр: организм − Organism, тип исследования – Study type (метод ДНК-микрочипов или секвенирования РНК, профиль экспрессии или метилирования), время публикации набора данных − Publication dates.
4. После применения фильтров можно найти нужный вам набор данных для анализа, перейдя по ссылке.
5. После перехода по ссылке вы можете найти описание набора данных, публикацию и авторов, контактную информацию и информацию об образцах и файлы, содержащие данные и доступные для скачивания.

Задание 1

Найти и скачать набор данных в базе данных Gene Expression Omnibus для построения взвешенных сетей ко-экспрессии молекулярно-генетических маркеров болезни Паркинсона

1. Перейдите на страницу https://www.ncbi.nlm.nih.gov/geo/ (рис.1)

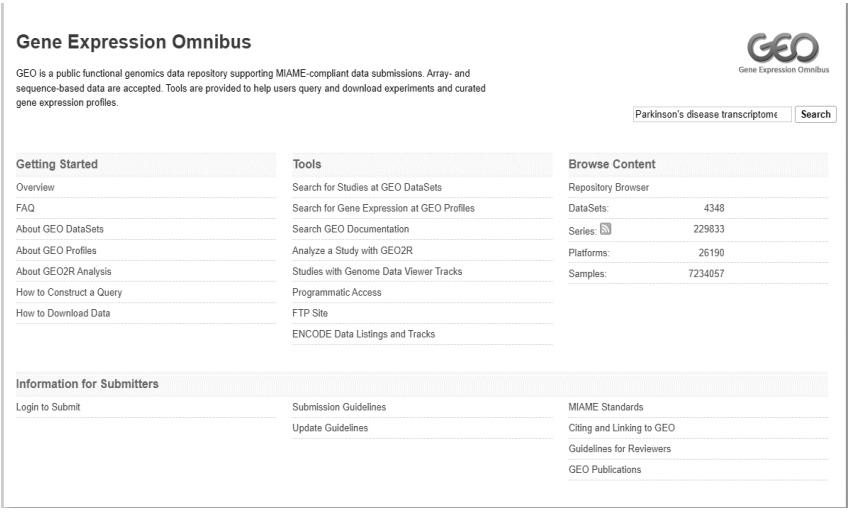


Рисунок 1. База данных Gene Expression Omnibus

2. Произведите поиск по следующим ключевым словам: «Parkinson's disease transcriptome datasets» – наборы данных транскриптома болезни Паркинсона (рис.2)

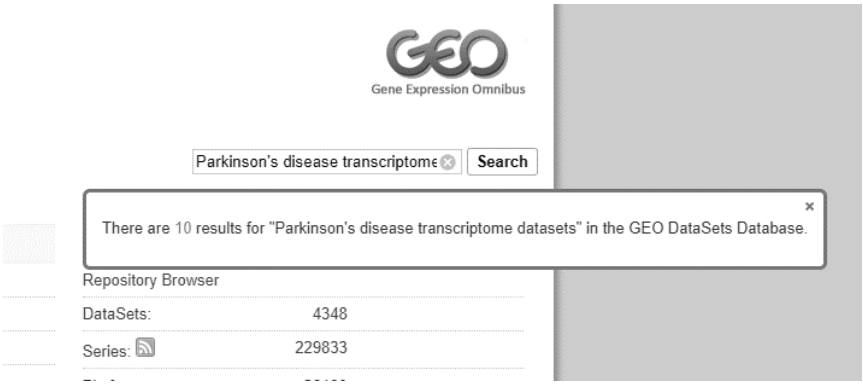


Рисунок 2. Строка поиска в базе данных Gene Expression Omnibus

3. Примените следующий фильтр: организм – Homo sapiens, тип исследования – Expression profiling by high throughput sequencing (рис.3)

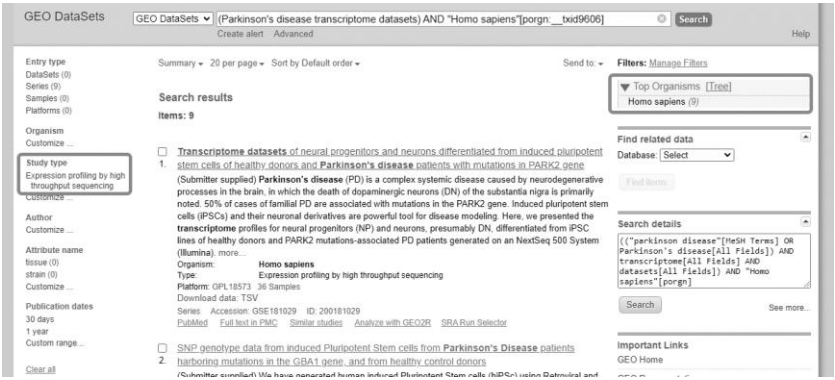
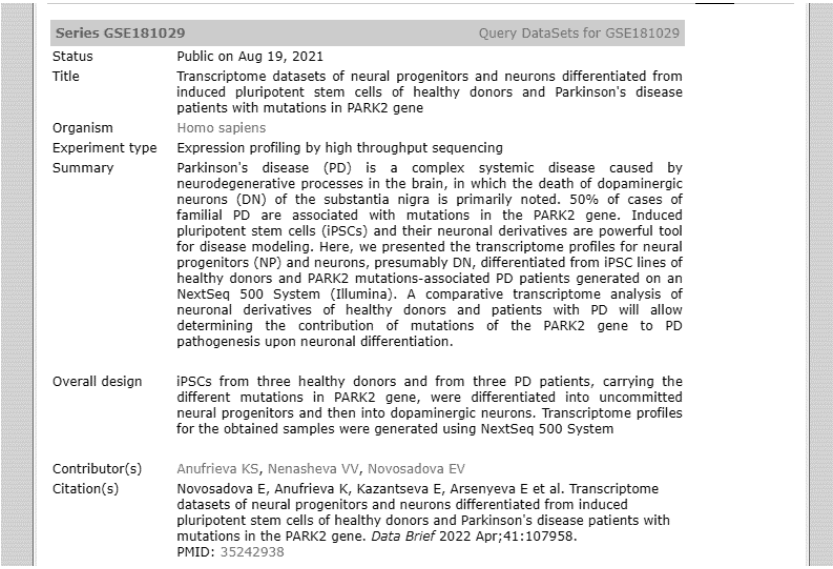


Рисунок 3. Применение фильтров в выдаче базы данных Gene Expression Omnibus

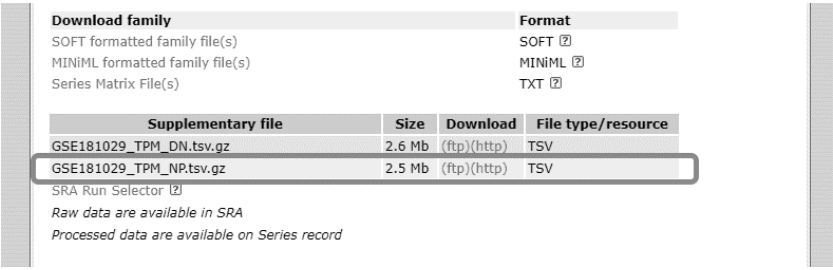
4. Найдите и скачайте набор данных под названием «Transcriptome datasets of neural progenitors and neurons differentiated from induced pluripotent stem cells of healthy donors and Parkinson's disease patients with mutations

in PARK2 gene» (рис.4): файл под названием GSE181029_TPM_NP.tsv.gz (рис.5)



Series GSE181029	Query DataSets for GSE181029
Status	Public on Aug 19, 2021
Title	Transcriptome datasets of neural progenitors and neurons differentiated from induced pluripotent stem cells of healthy donors and Parkinson's disease patients with mutations in PARK2 gene
Organism	Homo sapiens
Experiment type	Expression profiling by high throughput sequencing
Summary	Parkinson's disease (PD) is a complex systemic disease caused by neurodegenerative processes in the brain, in which the death of dopaminergic neurons (DN) of the substantia nigra is primarily noted. 50% of cases of familial PD are associated with mutations in the PARK2 gene. Induced pluripotent stem cells (iPSCs) and their neuronal derivatives are powerful tool for disease modeling. Here, we presented the transcriptome profiles for neural progenitors (NP) and neurons, presumably DN, differentiated from iPSC lines of healthy donors and PARK2 mutations-associated PD patients generated on an NextSeq 500 System (Illumina). A comparative transcriptome analysis of neuronal derivatives of healthy donors and patients with PD will allow determining the contribution of mutations of the PARK2 gene to PD pathogenesis upon neuronal differentiation.
Overall design	iPSCs from three healthy donors and from three PD patients, carrying the different mutations in PARK2 gene, were differentiated into uncommitted neural progenitors and then into dopaminergic neurons. Transcriptome profiles for the obtained samples were generated using NextSeq 500 System
Contributor(s)	Anufrieva KS, Nenasheva VV, Novosadova EV
Citation(s)	Novosadova E, Anufrieva K, Kazantseva E, Arsenyeva E et al. Transcriptome datasets of neural progenitors and neurons differentiated from induced pluripotent stem cells of healthy donors and Parkinson's disease patients with mutations in the PARK2 gene. *Data Brief* 2022 Apr;41:107958. PMID: 35242938

Рисунок 4. Набор данных под названием «Transcriptome datasets of neural progenitors and neurons differentiated from induced pluripotent stem cells of healthy donors and Parkinson's disease patients with mutations in PARK2 gene»



Download family	Format
SOFT formatted family file(s)	SOFT
MINiML formatted family file(s)	MINiML
Series Matrix File(s)	TXT

Supplementary file	Size	Download	File type/resource
GSE181029_TPM_DN.tsv.gz	2.6 Mb	(ftp)(http)	TSV
GSE181029_TPM_NP.tsv.gz	2.5 Mb	(ftp)(http)	TSV

SRA Run Selector

Raw data are available in SRA

Processed data are available on Series record

Рисунок 5. Ссылка для скачивания набора данных

В конечном результате с помощью данного пособия вы сможете построить, проанализировать и сравнить взвешенные сети ко-экспрессии молекулярно-генетических маркеров болезни Паркинсона на основе данного набора.

Тема 2. Анализ корреляции экспрессии генов

Для дальнейшего анализа рассмотрена работа в онлайн-программе **Posit Cloud**, использующую свободную программную среду вычислений и язык программирования R. Можно использовать и стационарную программу **RStudio** или **Posit**.

Для регистрации на портале необходимо:

1. Перейти на страницу https://posit.cloud/
2. Нажать кнопку «Get Started»
3. Выбрать «Learn More» в графе «Cloud Free»
4. Нажать кнопку «Sign Up»
5. Заполнить форму регистрации или использовать аккаунт Google или GitHub
6. Если заполнили форму, нужно перейти по ссылке подтверждения, которая приходит на указанную электронную почту

После вы можете зайти в собственный аккаунт и создать проект для работы. Для создания проекта нажмите кнопку «New Project» и выберите «New RStudio Project». После можете дать название собственному проекту в заголовке страницы и создать скрипт нажав меню «File», выбрав «New File» и «R Script».

Либо вы можете использовать ссылку, представленную во введении, чтобы добавить все готовые скрипты в ваш аккаунт на сайте. Для этого нажмите кнопку «New Project», выберите «New Project from Git Repository», вставьте скопированную ссылку на репозиторий в появившееся окно и нажмите «ОК». Все файлы – скрипты и таблицы − автоматически перенесутся в ваш проект.

Чтение и обработку данных можно произвести с помощью пакетов **readr**, **dplyr**, **tidyr** и **janitor**.

Функции **read_table()** и **read_csv()** из пакета **readr** используются для чтения файлов форматов **.tsv** и **.csv** соответственно. Для чтения файлов достаточно указать только адрес файла, либо только название файла, если он находится в рабочей директории.

```
table_tsv <- read_table("file_1.tsv")

table_csv <- read_csv("folder/file_2.csv")
```

Функция **clean_names()** из пакета **janitor** используется для обработки названий столбцов и удобства их чтения. Функция изменяет регистр текста на нижний и заменяет пробелы на нижние подчеркивания. В функции нужно указать только название таблицы.

```
clean_names(table)
```

Функция **select()** из пакета **dplyr** необходима для выбора столбцов из общей таблицы. Аргументами функции являются название таблицы и названия столбцов. Можно указать знак минуса перед названием столбца, чтобы его удалить. Также можно использовать функцию **starts_with()** для выбора столбцов по первым буквам названий.

```
select(table, column)
select(table, -column)
select(table, starts_with("a")
```

Функции **gather()** и **spread()** из пакета **tidyr** необходимы для транспонирования таблиц. Для того, чтобы использовать их в связке применяется оператор **pipe** (**%>%**), который позволяет проводить несколько манипуляций с

таблицами, не сохраняя промежуточные результаты. Функция **gather()** принимает несколько столбцов и объединяет их в пары «ключ-значение». Первым аргументов является «ключ», в этот столбец перейдут названия столбцов, второй аргумент – «значение», в этот столбец перейдут значения в указанных выше столбцах. И нужно указать какие столбцы будут участвовать в обработке. Их можно указать через знак двоеточия, если они идут друг за другом, обозначив только первый и последний столбец. Функция **spread()** распределяет пары «ключ-значение» по нескольким столбцам. В нашем случае, первой парой «ключ-значение» будут столбцы с названиями образцов и значения экспрессии, а второй парой – названия генов и значения экспрессии. Две функции в подобной связке позволят нам транспонировать таблицу, это необходимо для корреляционного анализа.

```
norm_table %>%
  tidyr::gather(key_1, value_2, column_1:
column_n) %>%
  tidyr::spread(key_2, value_2)
```

Функция **inner_join()** из пакета **dplyr** служит для объединения таблиц. Аргументами функции являются названия объединяемых таблиц и названия столбцов-«ключей», по которым будет происходить объединение. Последний аргумент указывается как **by**, на него подается вектор, представляющий собой названия столбцов-«ключей» в первой и второй таблицах. Между названиями ставится знак равенства.

```
inner_join(table_1, table_2,
by=c('key_1'='key_2'))
```

Для нахождения коэффициентов корреляции используется функция **cor_test()** из пакета **rstatix**. Функции данного пакета можно использовать с оператором pipe (**%>%**). На основе подсчитанной между парами генов корреляции Спирмена (указывается в аргументе **method**) создается таблица, из которой затем отбираются только значимые корреляции (p <0,05) с помощью функции **filter()** из пакета **dplyr**.

```
corr_table <- table %>%
   cor_test(method="spearman")
corr_table_signif <- filter(corr_table,
p<0.05)
```

Задание 2

Обработать скачанные таблицы, выбрать необходимые гены молекулярно-генетических маркеров болезни Паркинсона и получить коэффициенты корреляции экспрессии данных генов в образцах здоровых людей и пациентов − theme_2_script.R

1. Создайте проект в вашем аккаунте Posit Cloud, нажмите кнопку «New Project» и выберите «New RStudio Project»

2. Загрузите файл из набора данных GSE181029_TPM_DN.tsv.gz в папку в проекте Posit Cloud с помощью кнопки Upload в области Files

3. Тем же способом загрузите файл со списком молекулярно-генетических маркеров болезни Паркинсона.

4. *Создайте скрипт нажав меню «File», выбрав «New File» и «R Script». Шаги 1-4 можно пропустить, если вы загрузили скрипты из репозитория GitHub – откройте скрипт под названием theme_2_script.R*

5. *Установите необходимые библиотеки. Код можно скопировать в консоль и нажать Enter*

```
install.packages(c("dplyr", "janitor",
"tidyr", "rstatix", "readr"))
```

6. *В заголовке скрипта напишите необходимые для работы библиотеки*

```
library(dplyr)
library(janitor)
library(tidyr)
library(rstatix)
library(readr)
```

7. *Далее используйте функции чтения и обработки названий столбцов*

```
tableNP <-
read_table("GSE181029_TPM_NP.tsv.gz")
tableNP <- clean_names(tableNP)

t_gene <- read_csv("parkinsons.csv")
```

8. *Объедините таблицы*

```
table_gene <- inner_join(t_gene, tableNP,
by=c('genes'='gene_name'))
table_gene <- select(table_gene, -gene_id)
```

9. *Выберите столбцы, начинающиеся с буквы «g» и с буквы «n» – столбец «genes» и столбцы с уровнем экспрессии генов в образцах здоровых людей (начинаются со слова «norma»)*

```
table_norm <- select(table_gene,
starts_with("g"), starts_with("n"))
```

10. Транспонируйте таблицы для корреляционного анализа и удалите первый столбец

```
table_norm2 <- table_norm %>%
  tidyr::gather(Samples, Expression,
norma1np_1:norma2np_3) %>%
  tidyr::spread(genes, Expression) %>%
  select(-1)
```

11. Проведите корреляционный анализ, найдите только значимые результаты по столбцу p (уровень значимости) и удалите все коэффициенты корреляции, равные единице. Эти коэффициенты обозначают корреляцию уровня экспрессии одного и того же гена

```
corr_norm1 <- table_norm2 %>%
cor_test(method="spearman")
corr_norm <- filter(corr_norm1, p<0.05 & cor
!= 1)
```

12. ***Проведите ту же обработку данных для образцов пациентов с болезнью Паркинсона, шаги 9-11. Столбцы с образцами пациентов, начинаются с буквы «p» («park»)***

13. Исполните команды скрипта, выделив весь код и нажав комбинацию клавиш CTRL+Enter

В итоге мы получили 24 коэффициента корреляции уровней экспрессии 15 молекулярно-генетических маркеров болезни Паркинсона в образцах здоровых людей и 50 коэффициентов корреляции в образцах

пациентов. Далее на основе этих данных построим сети ко-экспрессии.

Тема 3. Построение и визуализация сетей ко-экспрессии генов

Построение сетей взаимодействия генов осуществляется с использованием пакетов **tidygraph**. Гены в сети ко-экспрессии являются узлами в сети, а полученные данные ко-экспрессии – рёбрами. Пакет **tidygraph** используется для построения и анализа любых сетей.

Для построения сети нам необходима таблица с коэффициентами корреляции, полученная на прошлом этапе. С помощью нее и функции **as_tbl_graph()** из пакета **tidygraph** мы можем построить сеть ко-экспрессии генов. Расположение столбцов в таблице, полученной при помощи функции **cor_test()**, позволяет избежать дополнительной обработки полученных данных. Первые два столбца таблицы представляют собой пары генов, между уровнями экспрессии которых есть значимая корреляция, а функция **as_tbl_graph()** определяет взаимодействия между узлами, обозначенными в первых двух столбцах таблицы. Также указываем аргумент **directed** равным *FALSE*, так как взаимодействия в сети ко-экспрессии не имеют направления.

```
as_tbl_graph(corr_table, directed = FALSE)
```

Для визуализации используется пакет **ggraph**. Функции данного пакета имеют такую же структуру, как и функции пакета **ggplot2**. Основная функция **ggraph()** принимает как аргумент полученную сеть (граф). Далее все дополнительные функции добавляются как слои с помощью оператор «+». Это могут быть функции обозначения цвета и ширины ребер − **geom_edge_link()** через аргументы **color** и

width, функция общей темы рисунка – **theme_graph()**, а также функция палитры цвета ребер в виде градиента – **scale_edge_color_gradient2()**. Аргументами последней функции являются **low** – цвет для самого низкого значения, **high** – цвет для самого высокого значения, **mid** – цвет для медианного значения. Цвет переходят один в другой градиентом.

Для обозначения узлов используется аргумент **size** в функции **geom_node_point()** и для обозначения названий узлов используется функция **geom_node_text().** Также можно использовать аргумент **repel** в этой функции, чтобы избежать наложения текста.

```
ggraph(graph) +
  geom_edge_link(aes(color = cor, width =
cor)) +
  theme_graph() +
  geom_node_text(aes(label = name), repel =
TRUE) +
  geom_node_point(size = 5) +
  scale_edge_color_gradient2(low = "blue",
high = "red", mid = "white")
```

Для наглядности используется следующий градиент: положительная корреляция между маркерами обозначалась красным, отрицательная – синим, и чем более значение корреляции было приближено к 0, тем более бледный цвет был у ребра (близким к белому). В зависимости от силы корреляции менялась и толщина рёбер.

Для сохранения полученных иллюстраций используется функции **ggsave()**. Аргументы функции следующие: название файла с указанием необходимого расширения (pdf, jpg, png и другие), название переменной с

сетью, ширина и длина иллюстрации.

```
ggsave("graph.png", graph, width = 10,
height = 15, units="cm")
```

Задание 3

Построить и визуализировать сети ко-экспрессии молекулярно-генетических маркеров болезни Паркинсона по данным образцов здоровых людей и пациентов – theme_3_script.R

1. Создайте скрипт нажав меню «File», выбрав «New File» и «R Script». Шаг 1 можно пропустить, если вы загрузили скрипты из репозитория GitHub – откройте скрипт под названием theme_3_script.R

2. Установите необходимые библиотеки. Код можно скопировать в консоль и нажать Enter

```
install.packages(c("tidygraph", "ggraph"))
```

3. Напишите в заголовке необходимые для работы библиотеки

```
library(tidygraph)
library(ggraph)
```

4. С помощью функции as_tbl_graph() постройте сети ко-экспрессии молекулярно-генетических маркеров болезни Паркинсона по данным образцов здоровых людей

```
norm_graph <- as_tbl_graph(corr_norm,
directed = FALSE)
```

5. Визуализируйте сети ко-экспрессии с помощью функции ***ggraph()***

```
network_norm <- ggraph(norm_graph) +
  geom_edge_link(aes(color = cor, width =
cor)) +
  theme_graph() +
  scale_edge_color_gradient2(low = "blue",
high = "red", mid = "white")
```

6. *Сохраните иллюстрации с помощью функции* ***ggsave()*** *– рисунок 6А*

```
ggsave("corr_network_norm.png",
network_norm, width = 50, height = 50,
units="cm")
```

7. ***Постройте и визуализируйте сети ко-экспрессии молекулярно-генетических маркеров болезни Паркинсона по данным образцов пациентов с болезнью Паркинсона, шаги 4-6.***

8. *Исполните команды скрипта, выделив весь код и нажав комбинацию клавиш CTRL+Enter*

В итоге мы построили и визуализировали сети ко-экспрессии молекулярно-генетических маркеров болезни Паркинсона по данным образцов здоровых людей и пациентов (рис.6). Далее проведем анализ построенных сетей и определим ключевые гены и модули генов в сетях.

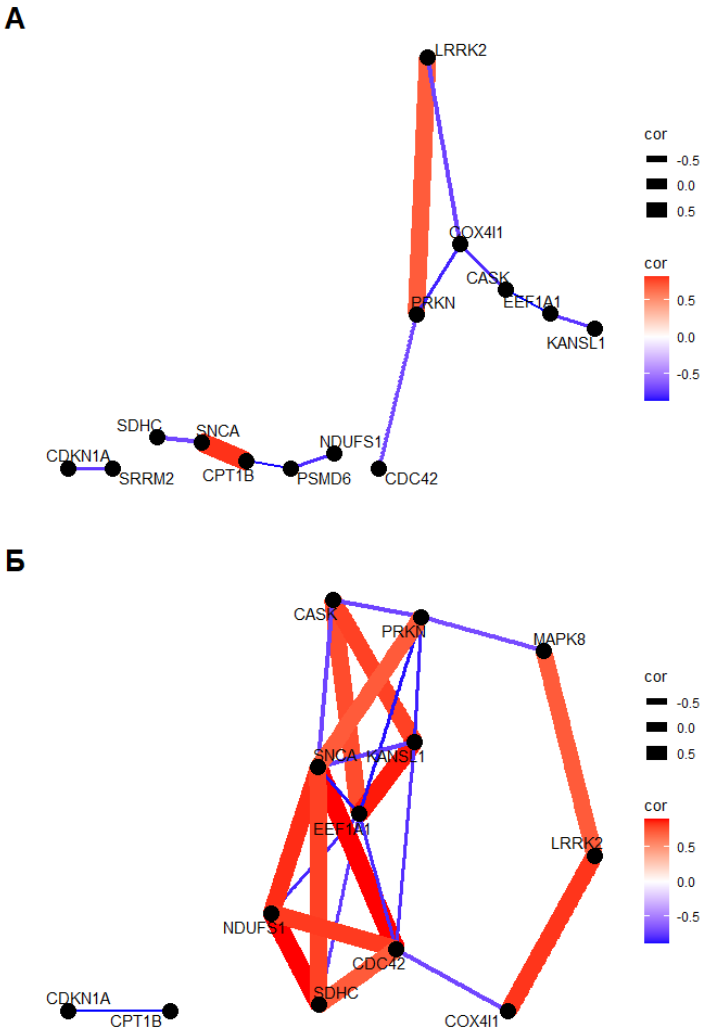


Рисунок 6. Сети ко-экспрессии молекулярно-генетических маркеров болезни Паркинсона по данным образцов здоровых людей (А) и пациентов (Б)

Тема 4. Анализ взвешенных сетей ко-экспрессии генов

Для выявления ключевых узлов и модулей, а также их дальнейшей визуализации используются пакеты **tidygraph**, **ggraph** и **dplyr**.

В основе каждой сети можно выделить 2 таблицы. Таблица узлов (**nodes**) представляет собой порядковые номера узлов, таблица ребер (**edges**) состоит из столбцов **from** – первый взаимодействующий узел, **to** – второй взаимодействующий узел, и еще вес взаимодействия, который будет учитываться при определении ключевых узлов и модулей. С помощью функции **activate()** можно обратится к той или иной таблице. При этом можно использовать оператор **pipe** (**%>%**).

```
graph %>%
  activate(nodes)
graph %>%
  activate(edges)
```

При определении параметров узлов и нахождении модулей должен использоваться вес ребер сети. Чем больше вес ребра, тем важнее взаимодействие двух узлов. Вес ребра не может быть отрицательным, поэтому коэффициенты корреляции будут обозначены по модулю. Для этого можно использовать функцию **abs()** и **mutate()**. Функция **abs()** изменяет все значения столбца на модуль числа. Функция **mutate()** из пакета **dplyr** позволяет создавать новые столбцы на основе данных в имеющихся столбцах.

```
graph %>%
  activate(edges) %>%
  mutate(weight = abs(cor))
```

С помощью функции **centrality_hub()** можно найти параметр концентрирования узлов, который указывает на количество связей этого узла с другими в сети. Для нахождения параметра промежуточности, который указывает на связность маркера, то есть то, как часто он является соединителем кратчайшего пути между узлами сети, не соединёнными напрямую, была использована функция **centrality_betweenness()**. Так как эти параметры относятся к узлам сети обращаемся к таблице узлов с помощью функции **activate()** и **mutate()**. Аргументами функций **centrality_hub()** и **centrality_betweenness()** является вес ребра в аргументе **weights**, указывается только название столбца из таблицы **edges**.

```
norm_graph %>%
  activate(nodes) %>%
  mutate(hub = centrality_hub(weights =
weight),
         betweenness =
centrality_betweenness(weights = weight))
```

С помощью функции **group_edge_betweenness()** можно найти модули узлов в сетях. Модули представляют собой группы наиболее связанных узлов в сетях. Так же обращаемся к таблице узлов с помощью функции **activate()** и **mutate()**. Еще указывается вес ребра в аргументе **weights** в виде названия столбца из таблицы **edges**.

```
norm_graph %>%
  activate(nodes) %>%
  mutate(group =
group_edge_betweenness(weights = weight))
```

Затем сети взаимодействия могут быть вновь визуализированы. Для обозначения найденных параметров

узлов можно использовать различные свойства визуализации: размер узлов в сетях может меняться в зависимости от параметра концентрирования, а непрозрачность названий генов в узлах сети – в зависимости от значений параметра промежуточности. Могут быть отмечены и модули в сети − цвет узлов указывает на номер модуля.

Для изменения размера и цвета узлов используется аргумент **size** и **color** в функции **geom_node_point()**, им приравниваются названия столбцов с параметрами концентрирования и промежуточности. Для изменения непрозрачности названий узлов используется функция **geom_node_text()** и аргумент **alpha**. Данному аргументу приравнивается столбец с указанием модулей узлов. Все остальные функции и аргументы указываются как в теме 3, вместе с функцией **ggraph**.

```
ggraph(graph) +
  geom_edge_link(aes(color = cor, width =
cor)) +
  geom_node_point(aes(color = nodes$group,
size = nodes$hub)) +
  geom_node_text(aes(label = name, alpha =
nodes$betweenness), repel = TRUE) +
  theme_graph() +
  scale_edge_color_gradient2(low = "blue",
high = "red", mid = "white")
```

Задание 4

Определить ключевые гены по параметрам промежуточности и концентрирования в сетях ко-экспрессии молекулярно-генетических маркеров болезни Паркинсона по данным образцов здоровых людей и пациентов, а также определите ключевые модули в данных сетях − theme_4_script.R

1. Создайте скрипт нажав меню «File», выбрав «New File» и «R Script». Шаг 1 можно пропустить, если вы загрузили скрипты из репозитория GitHub – откройте скрипт под названием theme_4_script.R

2. Напишите в заголовке необходимые для работы библиотеки

```
library(tidygraph)
library(ggraph)
library(readr)
library(dplyr)
```

*3. С помощью функций **activate()**, **mutate()** и **abs()** сохраните коэффициенты корреляции по модулю в столбец weight*

```
norm_graph_2 <- norm_graph %>%
  activate(edges) %>%
  mutate(weight = abs(cor))
```

*4. С помощью функций **activate()**, **mutate()**, **centrality_hub()**, **centrality_betweenness()** и **group_edge_betweenness()** сохраните параметры концентрирования, промежуточности и номера модулей в таблице (функция **as.data.frame()**)*

```
norm_graph_parameters <- norm_graph_2 %>%
  activate(nodes) %>%
  mutate(hub = centrality_hub(weights =
weight),
         betweenness =
centrality_betweenness(weight)) %>%
  mutate(group =
group_edge_betweenness(weights = weight))
%>%
  as.data.frame()
```

5. *Визуализируйте сети ко-экспрессии* с *помощью функции* ***ggraph()****, При этом используйте функцию* ***log()*** *– логарифмирование – для того, чтобы параметры отображались на иллюстрации корректно*

```
network_norm_2 <- ggraph(norm_graph_2) +
  geom_edge_link(aes(color = cor, width =
cor)) +
  geom_node_point(aes(color =
factor(norm_graph_parameters$group), size =
log(norm_graph_parameters$hub))) +
  geom_node_text(aes(label = name, size =
log(norm_graph_parameters$betweenness)),
repel = TRUE) +
  theme_graph() +
  scale_edge_color_gradient2(low = "blue",
high = "red", mid = "white")
```

6. *Сохраните иллюстрации* с *помощью функции* ***ggsave()*** *– рисунок 7А*

```
ggsave("corr_network_norm_key_genes.png",
network_norm_2, width = 70, height = 70,
units="cm")
```

7. ***Определите ключевые гены и модули в сетях ко-экспрессии молекулярно-генетических маркеров болезни Паркинсона по данным образцов пациентов, а также визуализируйте их, шаги 3-6.***

8. *Исполните команды скрипта, выделив весь код и нажав комбинацию клавиш CTRL+Enter*

Мы проанализировали сети ко-экспрессии по данным образцов здоровых людей и пациентов, и визуализировали их с учетом полученных параметров (рис.7).

Давайте подведем итог. Ключевые гены будем отбирать по следующим порогам: параметр концентрирования − 0.75, параметр промежуточности – 6, а также выберем самый крупный модуль генов.

Мы нашли, что для сетей, построенных на образцах здоровых людей, ключевыми генами по параметру концентрирования являются COX4I1, PRKN и LRRK2; а по параметру промежуточности − COX4I1 и CASK. Самый крупный модуль генов состоит из следующих генов: CPT1B, PSMD6, SNCA, NDUFS1 и SDHC.

А для сетей, построенных на образцах пациентов, ключевыми генами по параметру концентрирования являются EEF1A1, SNCA, CDC42, KANSL1; а по параметру промежуточности – PRKN, CDC42, SNCA и MAPK8 (порог − 6). Самый крупный модуль генов состоит из следующих генов: PRKN, SNCA, KANSL1 и CASK.

Таким образом, мы можем сказать, что ключевые гены и модули генов в сетях ко-экспрессии по данным образцов здоровых людей и пациентов отличаются. Далее проведем количественную оценку отличий данных сетей ко-экспрессии.

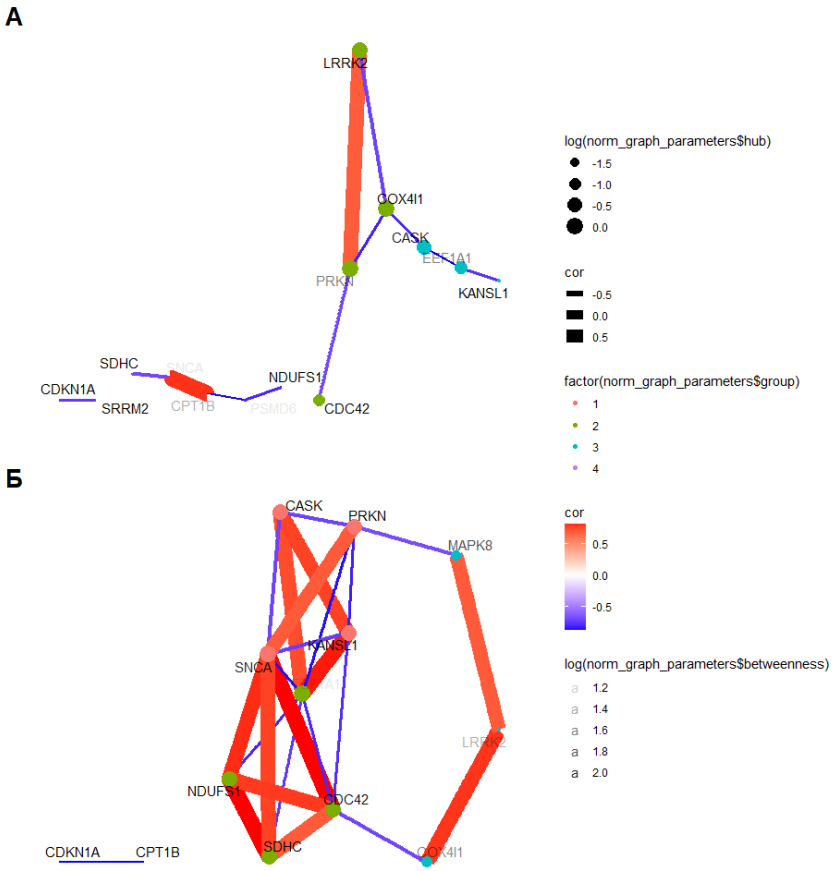


Рисунок 7. Сети ко-экспрессии молекулярно-генетических маркеров болезни Паркинсона по данным образцов здоровых людей (А) и пациентов (Б) с обозначением параметров концентрирования и промежуточности, а также обозначением модулей

Тема 5. Сравнение взвешенных сетей ко-экспрессии генов

Сравнение сетей взаимодействия молекулярно-генетических маркеров болезни Паркинсона производилось с помощью построения и сравнения дедрограмм, для этого используются пакеты **cluster** и **dendextend**. Пакет **cluster** используется для кластерного анализа различными методами, а функции из пакета **dendextend** позволяют визуализировать и сравнивать деревья иерархических кластеризации - дендрограммы.

Для того, чтобы использовать алгоритм иерархической кластеризации, нам необходимо получить коэффициенты корреляции в виде матрицы. Для этого, мы можем использовать функцию **cor_mat()** из пакета **rstatix**. Переносим названия генов в подписи строк с помощью функции **column_to_rownames()** из пакета **tibble**.

```
matrix <- table %>%
cor_mat(method="spearman")
matrix <- column_to_rownames(matrix, var =
"rowname")
```

Далее к полученной таблице применяется алгоритм агломеративной иерархической кластеризации (AGNES). Агломеративная или объединяющая кластеризация — иерархический метод формирования кластеров, при котором каждый объект сначала находится в отдельном кластере, затем объекты группируются в значительно более крупные кластеры. В данном пособии рассмотрена агломеративная иерархическая кластеризация методом полной связи и с использованием Евклидова расстояния – функция **agnes()** из

пакета **cluster**. Сохраняем результаты кластеризации как дендрограмму с помощью функции **as.dendrogram()**.

```
hier_clust <- agnes(matrix, metric  =
"euclidean", method = "complete")
dendrogram <- as.dendrogram(hier_clust)
```

После построения двух дендрограмм проводим их сравнение с использованием функции **cor_cophenetic()** из пакета **dendextend**. Полученный коэффициент кофенетической корреляции может принимать значения от -1 до 1. Чем ближе значение коэффициента к единице по модулю, тем более схожи дендрограммы.

```
cor_cophenetic(dendrogram_1, dendrogram_2)
```

Для визуализации различий между дендрограммами используется график танглграммы. Танглграмма (tanglegram, от англ. tangle – переплетение) показывает две дендрограммы и соединяют линиями одни и те же узлы в двух дендрограммах. Для построения танглграммы используется функция **tanglegram()** из пакета **dendextend**. Для того, чтобы названия генов отображались корректно можно использовать следующие аргументы: **lab.cex** – размер шрифта, **margin_inner** – расстояние между дендрограммами.

```
tanglegram(dendrogram_1, dendrogram_2,
lab.cex = 1.5, margin_inner = 10)
```

Задание 5

Сравнить взвешенные сети ко-экспрессии молекулярно-генетических маркеров болезни Паркинсона по данным образцов здоровых людей и пациентов − theme_5_script.R

1. *Создайте скрипт нажав меню «File», выбрав «New File» и «R Script». Шаг 1 можно пропустить, если вы загрузили скрипты из репозитория GitHub – откройте скрипт под названием theme_5_script.R*

2. *Установите необходимые библиотеки. Код можно скопировать в консоль и нажать Enter*

```
install.packages(c("tibble", "cluster",
"dendextend"))
```

3. *Напишите в заголовке необходимые для работы библиотеки*

```
library(tibble)
library(cluster)
library(dendextend)
```

4. *С помощью функций* ***cor_mat()*** *сохраните корреляционную матрицу и переносим названия генов в названия строк с помощью функции* ***column_to_rownames()***

```
cor_mat_norm <- table_norm2 %>%
  cor_mat(method="spearman")
cor_mat_norm <-
    column_to_rownames(cor_mat_norm, var =
    "rowname")
```

5. *С помощью функций* ***agnes()*** *и* ***as.dendrogram()*** *сохраните результаты кластеризации и дедрограмму*

```
hc_norm <- agnes(cor_mat_norm, metric  =
"euclidean", method = "complete")
dend_norm <- as.dendrogram(hc_norm)
```

6. ***Постройте корреляционную матрицу сети ко-экспрессии молекулярно-генетических маркеров болезни Паркинсона по данным образцов пациентов и сохраните результаты кластеризации и дедрограмму,***

шаги 3-6. Сохраните дедрограмму в переменную под именем dend_disease

7. Рассчитайте коэффициент кофенетической корреляции для двух дендрограмм

```
denr_cor <- cor_cophenetic(dend_norm,
dend_disease)
```

8. Создайте график танглграммы. В аргументе ***main*** *добавьте на иллюстрацию заголовок указывающий коэффициент кофенетической корреляции. С помощью функции* ***paste0()*** *соедините слово «Корреляция» и значение корреляции, полученной на шаге 7 и округленной с помощью функции* ***round()****. И в аргументы* ***main_left*** *и* ***main_ right*** *добавьте подписи к дендрограммам*

```
tanglegram(dend_norm, dend_disease, lab.cex
= 1.5, margin_inner = 10, main_left =
"Здоровые образцы", main_right = "Образцы
пациентов", main = paste0("Корреляция =",
round(denr_cor, 3)))
```

9. Если вы хотите сохранить танглграмму, используйте функции ***png()*** *и* ***dev.off()****. В функции* ***png()*** *нужно указать название, ширину, длину и разрешение файла. Вместо функции* ***png()*** *можно использовать другие функции, соответствующие необходимому разрешению:* ***jpg()****,* ***pdf()*** *и другие.*

```
png("tanglegram.png", width = 25, height =
20, res = 300, units = "cm")
tanglegram(dend_norm, dend_disease, lab.cex
= 1.5, margin_inner = 10, main_left =
"Здоровые образцы", main_right = "Образцы
пациентов", main = paste0("Корреляция =",
round(denr_cor, 3)))
dev.off()
```

10. *Исполните команды скрипта, выделив весь код и нажав комбинацию клавиш CTRL+Enter*

В итоге мы сравнили и визуализировали дендрограммы сетей ко-экспрессии по данным образцов здоровых людей и пациентов (рис.8).

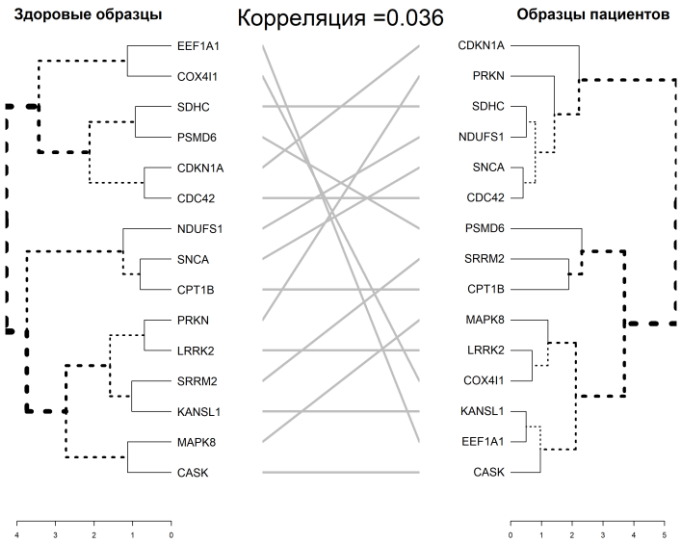


Рисунок 8. Танглграмма, построенная по сетям ко-экспрессии молекулярно-генетических маркеров болезни Паркинсона по данным образцов здоровых людей и пациентов

Коэффициент кофенетической корреляции равен 0.036. На иллюстрации видно, что дендрограммы значительно отличаются по кластерам и положению узлов. Таким образом, мы можем сказать, что сети ко-экспрессии

значительно отличаются. То есть возможные взаимодействия молекулярно-генетических маркеров болезни Паркинсона на уровне экспрессии отличаются. Подобные результаты могут пролить свет на процессы, происходящие в клетках, связанных с тем или иным заболеванием.

Заключение

Анализ взвешенных сетей ко-экспрессии генов облегчает исследования, ведущиеся с целью идентификации потенциальных биомаркеров или терапевтических мишеней. Методы данного анализа успешно применяются в различных областях биологии, например, исследования наследственных заболеваний и злокачественных опухолей. Разработанное учебно-методическое пособие представляет собой надежный путеводитель для построения взвешенных сетей ко-экспрессии генов, анализа сетей, а также их визуализации.

Пособие предназначено для студентов и аспирантов медико-биологического профиля, имеющий опыт в программировании на языке R, при проведении научных исследований в области биомедицинских исследований.

Использованная литература

1. Акберова Н.И. Основы анализа данных и программирования в R: учебно-методическое пособие / Н.И. Акберова, О.С. Козлова. – Казань: Альянс, 2017. – 33 с
2. Кабаков Р.И. R в действии. Анализ и визуализация данных на языке R / Р.И. Кабаков. – М.: ДМК Пресс, 2014. – 588 с.
3. Posit Cloud (formerly RStudio Cloud). Режим доступа https://posit.cloud/
4. R: Анализ и визуализация данных. Режим доступа http://r-analytics.blogspot.ru/
5. STHDA: Statistical tools for high-throughput data analysis. Режим доступа http://www.sthda.com/english/
6. tidyverse: set of packages that work in harmony because they share common data representations and API design. Режим доступа https://tidyverse.tidyverse.org/
7. tidygraph: a tidy API for graph/network manipulation. Режим доступа https://tidygraph.data-imaginist.com/
8. ggraph: An Implementation of Grammar of Graphics for Graphs and Networks. Режим доступа https://ggraph.data-imaginist.com/