Microarray vs RNA-seq data for chronic pancreatitis in humans: a comparison of differential gene expression results

Video tutorial:
https://youtu.be/S4-i-sPLI2M

GitHub repository:
https://github.com/lalagkaspn/chronic_pancreatitis_omics.git

Panagiotis N. Lalagkas

December 2023

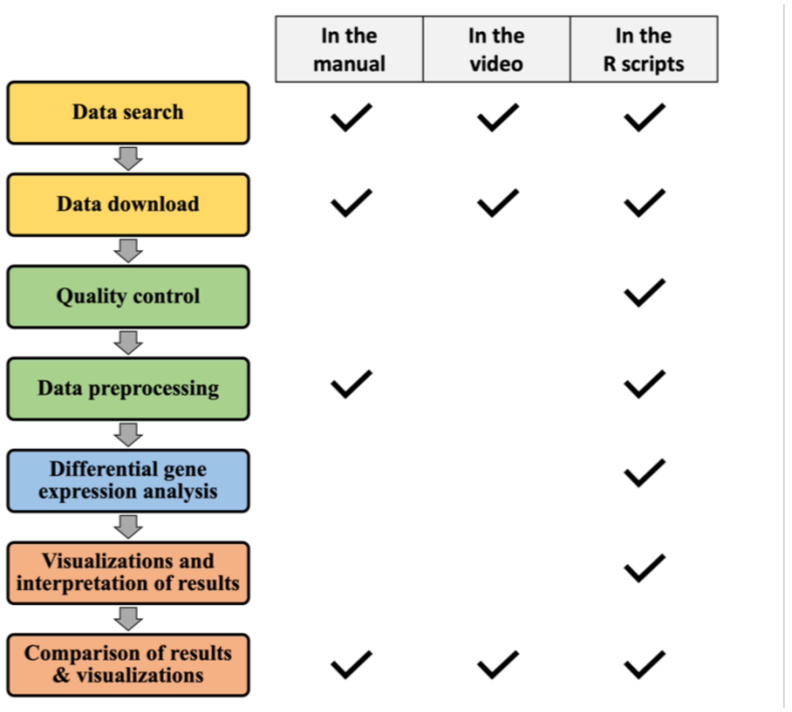
TABLE OF CONTENTS

1. Purpose statement

This manual is divided into three parts. First, it demonstrates how to search the Gene Expression Omnibus (GEO) database for publicly available microarray and RNA-sequencing (RNA-seq) data and how to download it in R. Second, it emphasizes the importance of data preprocessing through specific examples in R. Third, it provides a recommended workflow and visualization techniques to compare two lists of significantly differentially expressed genes obtained after the analysis of microarray or RNA-seq data for the same disease.

Other intermediate steps, such as differential gene expression analysis, are also of high importance. However, they are not the primary focus of this manual and are not mentioned in detail. But, to build this manual, all these steps had to be performed. Therefore, if you want to explore them, you can find all the R scripts used for downloading and preprocessing the gene expression data, conducting the differential gene expression analysis and comparing two lists of differentially expressed genes in this GitHub repository.

The following graphical abstract provides an overview of all the steps typically involved in a differential gene expression analysis. Its purpose is to give you an overall perspective before delving deeper into the present manual. In this illustration, you can also see which of these steps are covered in the manual, the tutorial video, or the R scripts.



	In the manual	In the video	In the R scripts
Data search	✓	✓	✓
Data download	✓	✓	✓
Quality control			✓
Data preprocessing	✓		✓
Differential gene expression analysis			✓
Visualizations and interpretation of results			✓
Comparison of results & visualizations	✓	✓	✓

2. Approach

We will search the GEO database for publicly available microarray and RNA-seq data for pancreatitis and normal samples. Then, we will download this data in R, an open-source, powerful programming language designed for statistical computing and data analysis. R offers a wide variety of packages, available in both CRAN and Bioconductor repositories, which we can use for data manipulation, gene expression analysis and visualization.

After downloading the microarray and RNA-seq data, we will preprocess it and perform a differential gene expression analysis by comparing pancreatitis samples to normal samples. Eventually, we will compare two lists of differentially expressed genes derived from the analysis of either microarray or RNA-seq data, create insightful figures and run simple statistical tests.

3. Background

The analysis of gene expression data has significantly helped scientists to gain a better understanding of the underlying biology of various diseases[1]. Two of the most widely used techniques for quantifying the expression of genes are microarray and RNA-seq, with the latter being the more modern approach. Consequently, knowing how to search, download, preprocess and analyze these two types of data is a valuable skill for every bioinformatician.

Both microarray and RNA-seq techniques can simultaneously measure the expression of thousands of genes in a single experiment. However, there is one important difference between them. Microarrays rely on pre-designed probes, which can introduce bias to the results as only sequences that hybridize with the probes can be detected. On the other hand, RNA-seq is a sequence-agnostic technique. This means that it can capture the expression of a broader range of genes, and it can detect low-abundance transcripts and novel isoforms[2].

Nevertheless, both microarrays and RNA-seq techniques have been extensively used to improve our understanding of biological processes and the biology of complex diseases. Thus, a plethora of microarray and RNA-seq data is publicly available for many complex diseases. Given that both techniques measure the expression of genes, logical questions are raised: do they give the same results or not? If not, are the differences important and can they be traced back to their potential cause? Previous studies have shown a high correlation between the gene expression profiles generated by the two techniques. Additionally, RNA-seq was found to be better at detecting a greater number of differentially expressed genes with higher fold-change[3]. But this might differ based on the question under study and the analyzed datasets.

Therefore, the purpose of this manual is to analyze microarray and RNA-seq data of pancreatitis and healthy samples to identify statistically significantly dysregulated genes and compare the results obtained from analyzing microarray and RNA-seq data.

4. Installation of R packages

Throughout this manual, we will be using various R packages (from either CRAN or Bioconductor repositories). To install all the necessary packages, please refer to the "packages_installation.R" script on this GitHub repository. Below, you find a list of these packages, their intended use, and hyperlinks to their respective websites.

Packages annotated with "*" are CRAN packages.

Packages annotated with "†" are Bioconductor packages.

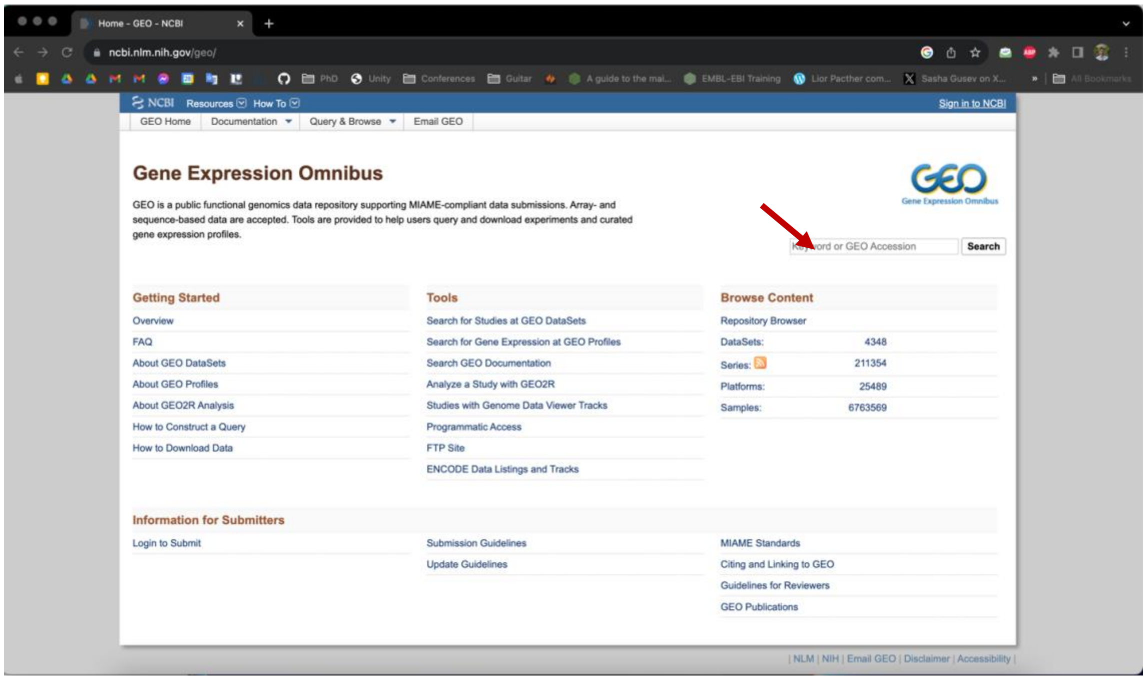
- Data manipulation
    - * dplyr: https://cran.r-project.org/web/packages/dplyr/index.html
    - * matrixStats: https://cran.rstudio.com/web/packages/matrixStats/index.html
    - * reshape2: https://cran.r-project.org/web/packages/reshape2/index.html
- Save data into excel file
    - * openxlsx: https://cran.r-project.org/web/packages/openxlsx/index.html
- Download microarray and RNA-seq data from the GEO database
    - † GEOquery: https://bioconductor.org/packages/release/bioc/html/GEOquery.html
- Imputation of missing values in gene expression datasets
    - † impute: https://bioconductor.org/packages/release/bioc/html/impute.html
- Conversion of human gene IDs to Entrez IDs:
    - † org.Hs.eg.db: https://bioconductor.org/packages/release/data/annotation/html/org.Hs.eg.db.html
- Differential gene expression analysis:
    - † limma: https://bioconductor.org/packages/release/bioc/html/limma.html
- Visualizations:
    - * ggplot2 (various plots): https://ggplot2.tidyverse.org/

- * pheatmap (heatmap): https://cran.r-project.org/web/packages/pheatmap/index.html
- † EnhancedVolcano (volcano plot): https://bioconductor.org/packages/release/bioc/html/EnhancedVolcano.html

5. Gene expression data search in GEO

GEO is a repository for gene expression data from array- and sequencing-based experiments. All the data in this database is publicly available. Below, you can see a screenshot of the GEO landing page (https://www.ncbi.nlm.nih.gov/geo/).

To search for gene expression data, you must provide keywords in the white rectangle next to the "Search" button (as indicated by the red arrow in the screenshot). Examples of keywords that can be used include specific disease terms, the species of interest or the desired method used to generate the gene expression data. In our case, we are interested in searching for gene expression data related to "pancreatitis" in humans and generated using either microarray or RNA-seq method.



Gene Expression Omnibus

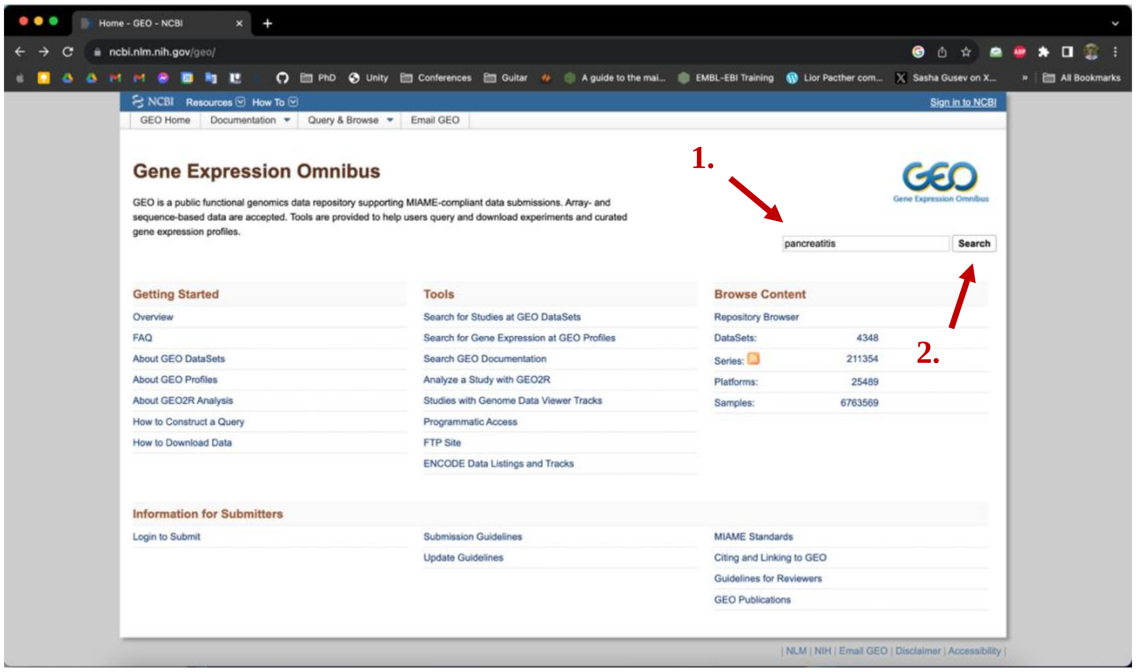
GEO is a public functional genomics data repository supporting MIAME-compliant data submissions. Array- and sequence-based data are accepted. Tools are provided to help users query and download experiments and curated gene expression profiles.

Getting Started	Tools	Browse Content	
Overview	Search for Studies at GEO DataSets	Repository Browser	
FAQ	Search for Gene Expression at GEO Profiles	DataSets:	4348
About GEO DataSets	Search GEO Documentation	Series:	211354
About GEO Profiles	Analyze a Study with GEO2R	Platforms:	25489
About GEO2R Analysis	Studies with Genome Data Viewer Tracks	Samples:	6763569
How to Construct a Query	Programmatic Access		
How to Download Data	FTP Site		
	ENCODE Data Listings and Tracks		

Information for Submitters

Login to Submit	Submission Guidelines	MIAME Standards
	Update Guidelines	Citing and Linking to GEO
		Guidelines for Reviewers
		GEO Publications

Let’s begin by searching for “pancreatitis” data. We enter “pancreatitis in the white rectangle box and click “Search”.



Gene Expression Omnibus

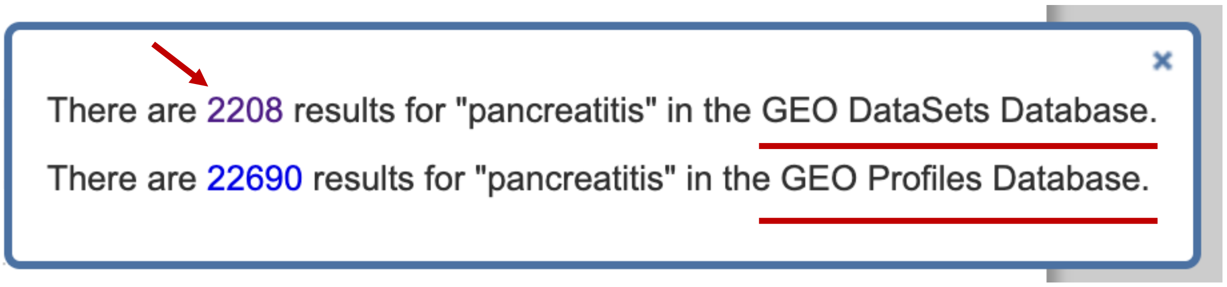
GEO is a public functional genomics data repository supporting MIAME-compliant data submissions. Array- and sequence-based data are accepted. Tools are provided to help users query and download experiments and curated gene expression profiles.

Getting Started	Tools	Browse Content	
Overview	Search for Studies at GEO DataSets	Repository Browser	
FAQ	Search for Gene Expression at GEO Profiles	DataSets:	4348
About GEO DataSets	Search GEO Documentation	Series:	211354
About GEO Profiles	Analyze a Study with GEO2R	Platforms:	25489
About GEO2R Analysis	Studies with Genome Data Viewer Tracks	Samples:	6763569
How to Construct a Query	Programmatic Access		
How to Download Data	FTP Site		
	ENCODE Data Listings and Tracks		

Information for Submitters

Login to Submit	Submission Guidelines	MIAME Standards
	Update Guidelines	Citing and Linking to GEO
		Guidelines for Reviewers
		GEO Publications

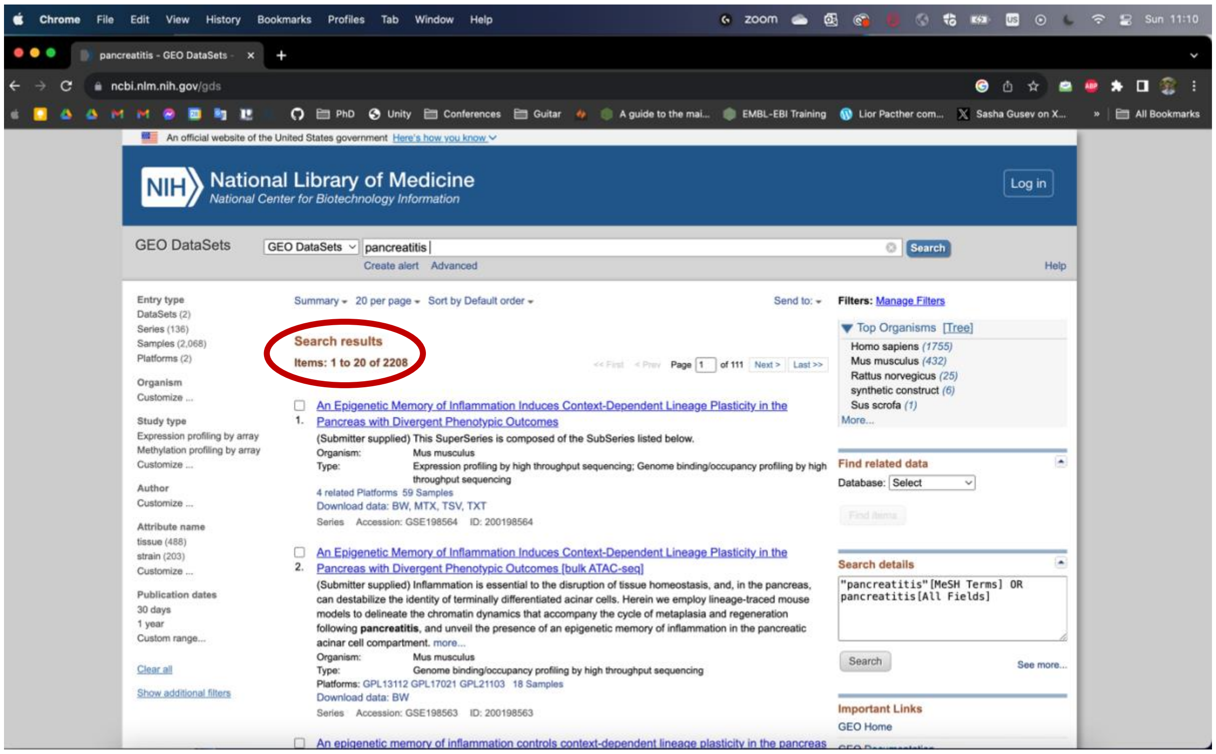
A white box appears (see screenshot below). As you can see, we have the option to search the GEO DataSets or GEO Profiles database. A GEO DataSet is a collection of biologically and statistically comparable samples that were analyzed using the same process and contains the expression of many genes. In contrast, a GEO Profile includes expression measurements for an individual gene across many samples in a GEO DataSet. In our case, we want to look for GEO DataSets. So, we click on the hyperlink for the GEO DataSets (the hyperlink is usually the number of results; so, we click on the “2208”). Note that the number of returned results might change in the future as more datasets will be added from future studies.



There are 2208 results for "pancreatitis" in the GEO DataSets Database.

There are 22690 results for "pancreatitis" in the GEO Profiles Database.

Now, we can see all the GEO DataSets available in the GEO database and are related to pancreatitis. We have 2,208 results (see red circle below).


Chrome File Edit View History Bookmarks Profiles Tab Window Help
pancreatitis - GEO DataSets
ncbi.nlm.nih.gov/gds
An official website of the United States government Here's how you know
National Library of Medicine
National Center for Biotechnology Information
Log in
GEO DataSets
Search
Create alert Advanced
Help

GEO DataSets | pancreatitis

Entry type
DataSets (2)
Series (136)
Samples (2,068)
Platforms (2)

Organism
Customize ...

Study type
Expression profiling by array
Methylation profiling by array
Customize ...

Author
Customize ...

Attribute name
tissue (488)
strain (203)
Customize ...

Publication dates
30 days
1 year
Custom range...

Clear all
Show additional filters

Summary ▾ 20 per page ▾ Sort by Default order ▾

Search results
Items: 1 to 20 of 2208

<< First < Prev Page 1 of 111 Next > Last >>

1. An Epigenetic Memory of Inflammation Induces Context-Dependent Lineage Plasticity in the Pancreas with Divergent Phenotypic Outcomes
(Submitter supplied) This SuperSeries is composed of the SubSeries listed below.
Organism: Mus musculus
Type: Expression profiling by high throughput sequencing; Genome binding/occupancy profiling by high throughput sequencing
4 related Platforms 59 Samples
Download data: BW, MTX, TSV, TXT
Series Accession: GSE198564 ID: 200198564

2. An Epigenetic Memory of Inflammation Induces Context-Dependent Lineage Plasticity in the Pancreas with Divergent Phenotypic Outcomes [bulk ATAC-seq]
(Submitter supplied) Inflammation is essential to the disruption of tissue homeostasis, and, in the pancreas, can destabilize the identity of terminally differentiated acinar cells. Herein we employ lineage-traced mouse models to delineate the chromatin dynamics that accompany the cycle of metaplasia and regeneration following **pancreatitis**, and unveil the presence of an epigenetic memory of inflammation in the pancreatic acinar cell compartment. more...
Organism: Mus musculus
Type: Genome binding/occupancy profiling by high throughput sequencing
Platforms: GPL13112 GPL17021 GPL21103 18 Samples
Download data: BW
Series Accession: GSE198563 ID: 200198563

An epigenetic memory of inflammation controls context-dependent lineage plasticity in the pancreas

Send to: ▾

Filters: Manage Filters

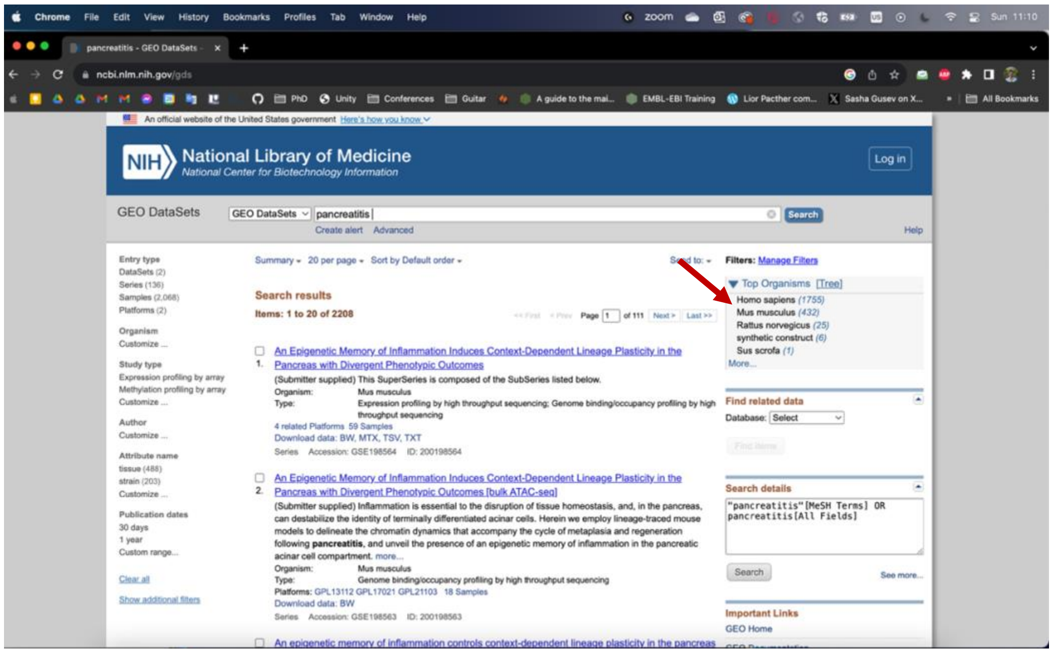
Top Organisms [Tree]
Homo sapiens (1755)
Mus musculus (432)
Rattus norvegicus (25)
synthetic construct (6)
Sus scrofa (1)
More...

Find related data
Database: Select
Find items

Search details
```
"pancreatitis"[MeSH Terms] OR
pancreatitis[All Fields]
```
Search
See more...

Important Links
GEO Home

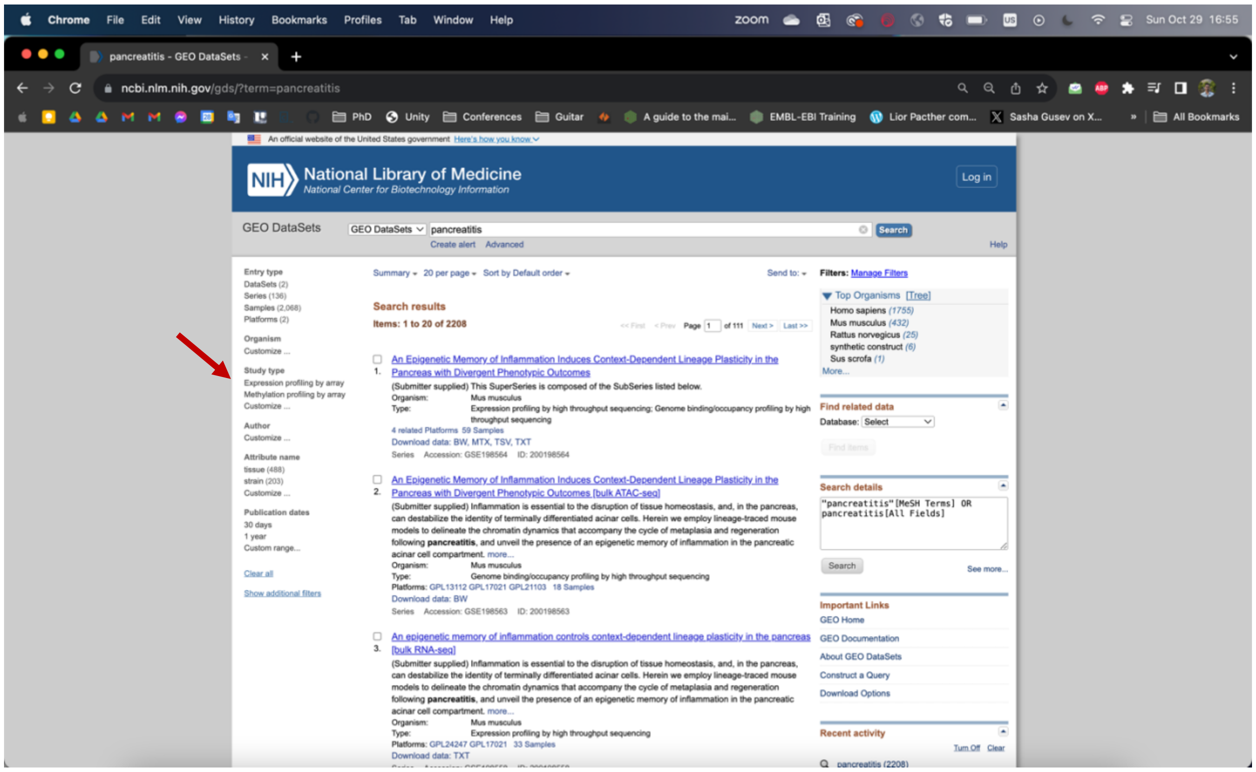
Next, we filter the results for those related to humans (Top Organisms = Homo sapiens; see right red arrow below).


GEO DataSets | pancreatitis

Search results
Items: 1 to 20 of 2208

Top Organisms [Tree]
Homo sapiens (1755)
Mus musculus (432)
Rattus norvegicus (25)
synthetic construct (6)
Sus scrofa (1)
More...

To search for **microarray data**, we select the study type to be "Expression profiling by array" (see red arrow below).



Entry type
DataSets (2)
Series (136)
Samples (2,068)
Platforms (2)

Organism
Customize ...

Study type
Expression profiling by array
Methylation profiling by array
Customize ...

Author
Customize ...

Attribute name
tissue (488)
strain (203)
Customize ...

Publication dates
30 days
1 year
Custom range...

Clear all

Show additional filters

Search results

Items: 1 to 20 of 2208

<< First < Prev Page 1 of 111 Next > Last >>

1. An Epigenetic Memory of Inflammation Induces Context-Dependent Lineage Plasticity in the Pancreas with Divergent Phenotypic Outcomes
(Submitter supplied) This SuperSeries is composed of the SubSeries listed below.
Organism: Mus musculus
Type: Expression profiling by high throughput sequencing; Genome binding/occupancy profiling by high throughput sequencing
4 related Platforms 59 Samples
Download data: BW, MTX, TSV, TXT
Series Accession: GSE198564 ID: 200198564

2. An Epigenetic Memory of Inflammation Induces Context-Dependent Lineage Plasticity in the Pancreas with Divergent Phenotypic Outcomes [bulk ATAC-seq]
(Submitter supplied) Inflammation is essential to the disruption of tissue homeostasis, and, in the pancreas, can destabilize the identity of terminally differentiated acinar cells. Herein we employ lineage-traced mouse models to delineate the chromatin dynamics that accompany the cycle of metaplasia and regeneration following **pancreatitis**, and unveil the presence of an epigenetic memory of inflammation in the pancreatic acinar cell compartment. more...
Organism: Mus musculus
Type: Genome binding/occupancy profiling by high throughput sequencing
Platforms: GPL13112 GPL17021 GPL21103 18 Samples
Download data: BW
Series Accession: GSE198563 ID: 200198563

3. An epigenetic memory of inflammation controls context-dependent lineage plasticity in the pancreas [bulk RNA-seq]
(Submitter supplied) Inflammation is essential to the disruption of tissue homeostasis, and, in the pancreas, can destabilize the identity of terminally differentiated acinar cells. Herein we employ lineage-traced mouse models to delineate the chromatin dynamics that accompany the cycle of metaplasia and regeneration following **pancreatitis**, and unveil the presence of an epigenetic memory of inflammation in the pancreatic acinar cell compartment. more...
Organism: Mus musculus
Type: Expression profiling by high throughput sequencing
Platforms: GPL24247 GPL17021 33 Samples
Download data: TXT

Top Organisms [Tree]
Homo sapiens (1755)
Mus musculus (432)
Rattus norvegicus (25)
synthetic construct (6)
Sus scrofa (1)
More...

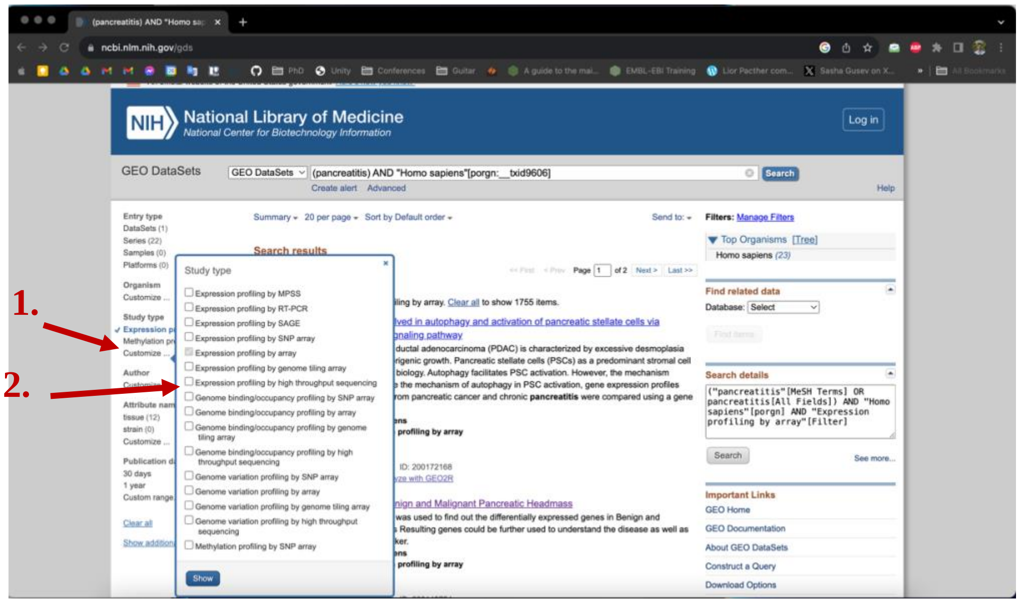
Find related data
Database: Select
Find items

Search details
```
"pancreatitis"[MeSH Terms] OR
pancreatitis[All Fields]
```
Search
See more...

Important Links
GEO Home
GEO Documentation
About GEO DataSets
Construct a Query
Download Options

Recent activity
Turn Off Clear
pancreatitis (2208)

To search for **RNA-seq data**, we select the "Study type" to be "Expression profiling by high throughput sequencing" instead. We find this option if we click on "Customize" under the Study type.



(pancreatitis) AND "Homo sapiens"[porgn:__txid9606]

Entry type
DataSets (1)
Series (22)
Samples (0)
Platforms (0)

Study type
- Expression profiling by MPSS
- Expression profiling by RT-PCR
- Expression profiling by SAGE
- Expression profiling by SNP array
- Expression profiling by array
- Expression profiling by genome tiling array
- Expression profiling by high throughput sequencing
- Genome binding/occupancy profiling by SNP array
- Genome binding/occupancy profiling by array
- Genome binding/occupancy profiling by genome tiling array
- Genome binding/occupancy profiling by high throughput sequencing
- Genome variation profiling by SNP array
- Genome variation profiling by array
- Genome variation profiling by genome tiling array
- Genome variation profiling by high throughput sequencing
- Methylation profiling by SNP array

Page 1 of 2

Top Organisms [Tree]
Homo sapiens (23)

Search details
```
("pancreatitis"[MeSH Terms] OR
pancreatitis[All Fields]) AND "Homo
sapiens"[porgn] AND "Expression
profiling by array"[Filter]
```

Important Links
GEO Home
GEO Documentation
About GEO DataSets
Construct a Query
Download Options

Eventually, we find 23 GEO DataSets for pancreatitis in humans generated by microarrays. However, it is possible that some of the returned GEO DataSets do not contain pancreatitis or normal samples. After manual inspection, we keep the following 5 microarray datasets:

- GSE143754: https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE143754
- GSE61166: https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE61166
- GSE71989: https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE71989
- GSE101462: https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE101462
- GSE77858: https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE77858

Similar to microarrays, we find 59 GEO DataSets for pancreatitis in humans generated by RNA-seq. Again, after manual inspection, we keep the following 2 RNA-seq datasets that contain gene expression data from both pancreatitis patients and normal samples:

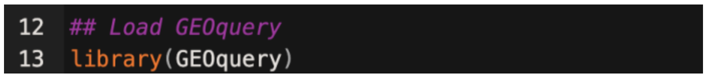
- GSE194331: https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE194331
- GSE133684: https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE133684

At this point, it is important to acknowledge that our upcoming comparison involves samples from 5 microarray studies and samples from 2 RNA-seq studies. This difference in the number of studies will potentially introduce bias to our analysis. Additionally, these samples stem from different individuals, which might introduce biological variation. These factors are essential to consider when interpreting our analysis results at the end of this manual. Ideally, we would like to have same samples from the same individuals processed using both microarray and RNA-seq methods to minimize bias.

6. Gene expression data download in R

6.1. Microarray data

To download GEO microarray expression data in R, we use the R package "GEOquery". Given that you have already installed it by following the "package_installation.R" script from this GitHub repository, we load it into R.



```r
## Load GEOquery
library(GEOquery)
```

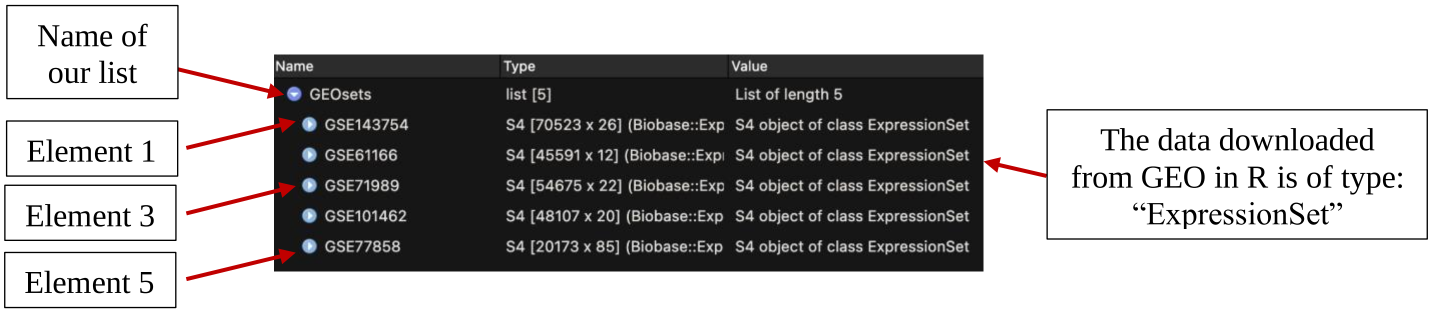
We create a vector containing the accession numbers of the GEO microarray DataSets we want to download (the 5 datasets mentioned earlier).

```r
# Vector with datasets to download from GEO
datasets = c("GSE143754", "GSE61166", "GSE71989", "GSE101462", "GSE77858")
```

As we want to download more than one GEO DataSets, we create an empty list named "GEOsets" (row 20), which we populate later with the downloaded data. Using the *getGEO()* function in a for loop, we download all the available data for each GEO DataSet (rows 21-24). Eventually, we name the elements of the list using the corresponding GEO accession numbers (row 26).

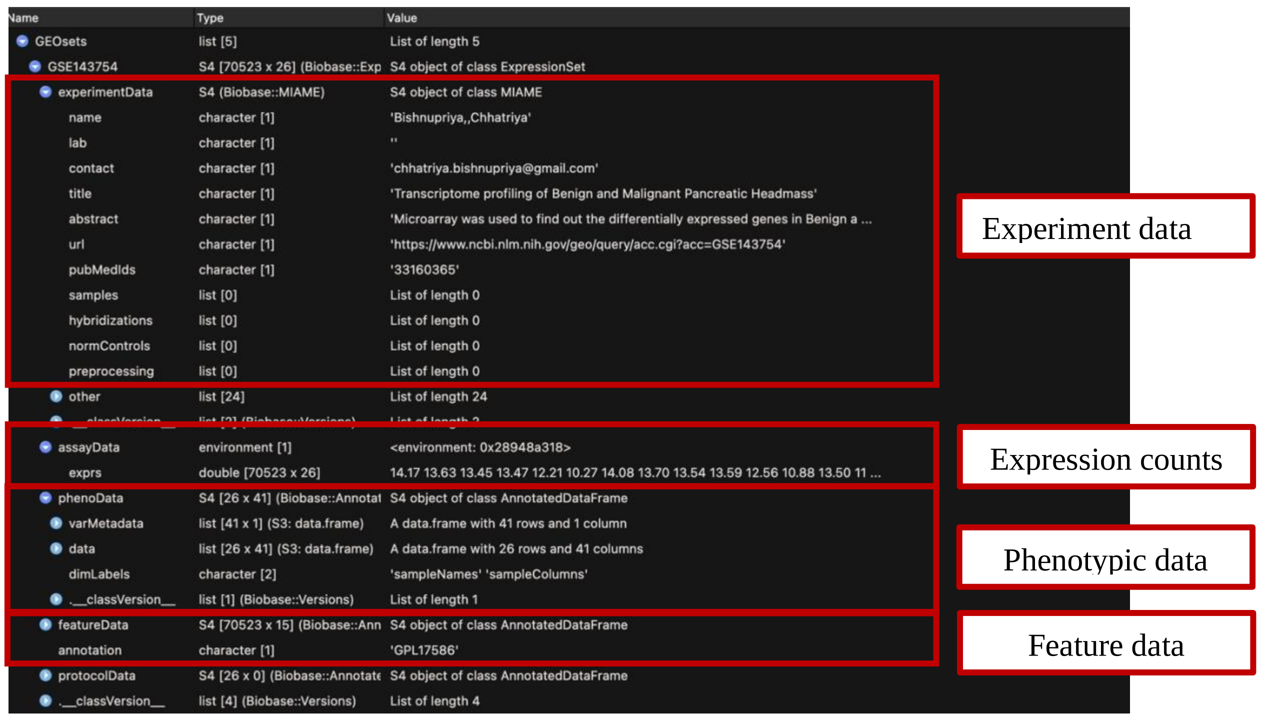
```r
19  # Download data
20  GEOsets = list()
21  for (i in 1:length(datasets)){
22    GEOsets[[i]] = getGEO(datasets[i])
23  }; rm(i)
24  GEOsets = unlist(GEOsets)
25  # Name the elements of the list with the GEO accession numbers
26  names(GEOsets) = datasets; rm(datasets)
```

After downloading the data, we can click on the "GEOSets" variable in our R environment to inspect it. We can see that our list has 5 elements, each named after a GEO accession number.


Name of our list
Element 1
Element 3
Element 5
The data downloaded from GEO in R is of type: "ExpressionSet"

Name	Type	Value
GEOsets	list [5]	List of length 5
GSE143754	S4 [70523 x 26] (Biobase::Exp	S4 object of class ExpressionSet
GSE61166	S4 [45591 x 12] (Biobase::Expı	S4 object of class ExpressionSet
GSE71989	S4 [54675 x 22] (Biobase::Exp	S4 object of class ExpressionSet
GSE101462	S4 [48107 x 20] (Biobase::Exp	S4 object of class ExpressionSet
GSE77858	S4 [20173 x 85] (Biobase::Exp	S4 object of class ExpressionSet

Let's investigate one element to see what information is included. As you can see below, it contains information about the experiment (experimentData), the expression counts per probe (assayData), and metadata about the samples (phenoData) and each probe (featureData). Feel free to further explore the data.



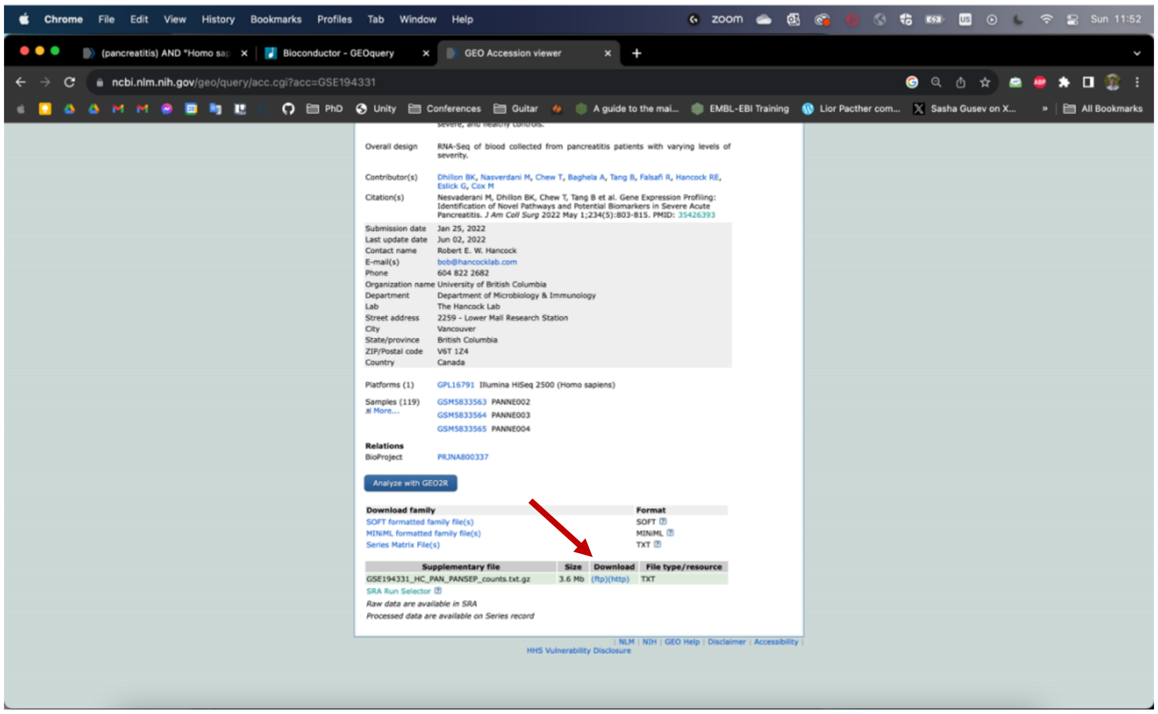
```
Name                 Type                            Value
GEOsets              list [5]                        List of length 5
  GSE143754          S4 [70523 x 26] (Biobase::Exp   S4 object of class ExpressionSet
    experimentData   S4 (Biobase::MIAME)             S4 object of class MIAME
      name           character [1]                   'Bishnupriya,,Chhatriya'
      lab            character [1]                   ''
      contact        character [1]                   'chhatriya.bishnupriya@gmail.com'
      title          character [1]                   'Transcriptome profiling of Benign and Malignant Pancreatic Headmass'
      abstract       character [1]                   'Microarray was used to find out the differentially expressed genes in Benign a ...
      url            character [1]                   'https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE143754'
      pubMedIds      character [1]                   '33160365'
      samples        list [0]                        List of length 0
      hybridizations list [0]                        List of length 0
      normControls   list [0]                        List of length 0
      preprocessing  list [0]                        List of length 0
      other          list [24]                       List of length 24
      .__classVersion__ list [2] (Biobase::Versions) List of length 2
    assayData        environment [1]                 <environment: 0x28948a318>
      exprs          double [70523 x 26]             14.17 13.63 13.45 13.47 12.21 10.27 14.08 13.70 13.54 13.59 12.56 10.88 13.50 11 ...
    phenoData        S4 [26 x 41] (Biobase::Annotat  S4 object of class AnnotatedDataFrame
      varMetadata    list [41 x 1] (S3: data.frame)  A data.frame with 41 rows and 1 column
      data           list [26 x 41] (S3: data.frame) A data.frame with 26 rows and 41 columns
      dimLabels      character [2]                   'sampleNames' 'sampleColumns'
      .__classVersion__ list [1] (Biobase::Versions) List of length 1
    featureData      S4 [70523 x 15] (Biobase::Ann   S4 object of class AnnotatedDataFrame
    annotation       character [1]                   'GPL17586'
    protocolData     S4 [26 x 0] (Biobase::Annotate  S4 object of class AnnotatedDataFrame
    .__classVersion__ list [4] (Biobase::Versions)   List of length 4
```

TIP: to access an element of a list in R, you can use the list name followed by double square brackets and the name (or index) of the desired element. To access a specific component of an element, you can use the "@". For example, if you want to access the expression data for the GEO DataSet "GSE143754", you can use the following command:

```
GEOsets[["GSE143754"]]@assayData
```

6.2. RNA-seq data

If we follow the same process as above to download gene expression data for RNA-seq experiments in R, we will notice that the assayData element, which contains the expression values, is empty. This happens because the getGEO function is designed to download expression data from microarrays but not from RNA-seq experiments. Instead, we can download RNA-seq expression data directly from the webpage of each GEO DataSet. For example, to download the RNA-seq expression data for the GEO DataSet "GSE194331", we search the GEO database for "GSE194331" (at the same white box that we searched for pancreatitis earlier). After clicking search, we arrive at the page shown below. This is the webpage of the study we searched for. If we scroll to the bottom of this webpage, we find the expression data in a “txt” file. We download and save it locally in our laptop by clicking on the hyperlink at the third column (see red arrow).



Chrome File Edit View History Bookmarks Profiles Tab Window Help
(pancreatitis) AND "Homo sa
Bioconductor - GEOquery
GEO Accession viewer
ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE194331
PhD Unity Conferences Guitar A guide to the mai... EMBL-EBI Training Lior Pacther com... Sasha Gusev on X... All Bookmarks
Analyze with GEO2R

Overall design	RNA-Seq of blood collected from pancreatitis patients with varying levels of severity.
Contributor(s)	Dhillon BK, Nasverdani M, Chew T, Baghela A, Tang B, Falsafi R, Hancock RE, Eslick G, Cox M
Citation(s)	Nesvaderani M, Dhillon BK, Chew T, Tang B et al. Gene Expression Profiling: Identification of Novel Pathways and Potential Biomarkers in Severe Acute Pancreatitis. *J Am Coll Surg* 2022 May 1;234(5):803-815. PMID: 35426393
Submission date	Jan 25, 2022
Last update date	Jun 02, 2022
Contact name	Robert E. W. Hancock
E-mail(s)	bob@hancocklab.com
Phone	604 822 2682
Organization name	University of British Columbia
Department	Department of Microbiology & Immunology
Lab	The Hancock Lab
Street address	2259 - Lower Mall Research Station
City	Vancouver
State/province	British Columbia
ZIP/Postal code	V6T 1Z4
Country	Canada
Platforms (1)	GPL16791 Illumina HiSeq 2500 (Homo sapiens)
Samples (119) More...	GSM5833563 PANNE002
	GSM5833564 PANNE003
	GSM5833565 PANNE004

Relations

BioProject	PRJNA800337

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
GSE194331_HC_PAN_PANSEP_counts.txt.gz	3.6 Mb	(ftp)(http)	TXT

SRA Run Selector

Raw data are available in SRA

Processed data are available on Series record

| NLM | NIH | GEO Help | Disclaimer | Accessibility |
HHS Vulnerability Disclosure

NOTE: every GEO DataSet has a dedicated webpage similar to the one shown above. This webpage includes all the information related to the study/dataset. All this information is also included in the list we previously viewed about the microarray data (the one we downloaded in R using the getGEO() function). For example, experimentData contains the data presented above the “gray” box in the screenshot. phenoData contains all the information found under the “Samples” section of the webpage. featureData contains all the data provided under the “Platforms” section of the webpage.

7. Gene expression data preprocessing in R

Data preprocessing is a crucial step in any data analysis workflow. However, the specific preprocessing steps to apply depend on the data at hand and may vary from case to case. In the context of the present manual, all the preprocessing steps applied to the data can be found in detail in the corresponding github scripts (GSE_microarrays.R rows 44-597; GSE_rnaseq.R rows 49-346). Below, the most important ones are briefly mentioned.

7.1. Microarray data

7.1.1. Dealing with missing expression data

Some of the probes in microarray DataSets have missing expression data for some samples (due to technical issues). In such cases, we can impute the missing values by leveraging information from the expression of a gene in other samples. However, when a gene has missing values for many samples, imputation may not be accurate. Therefore, we first remove the probes with missing values in more than 25% of the samples. The 25% threshold was chosen arbitrarily, as no gold-standard guidelines exist. After filtering them out, we use a method called k-NN imputation to impute the missing values in the remaining probes (GSE_microarrays.R rows 301-330).

7.1.2. Dealing with multiple probes matching to the same gene

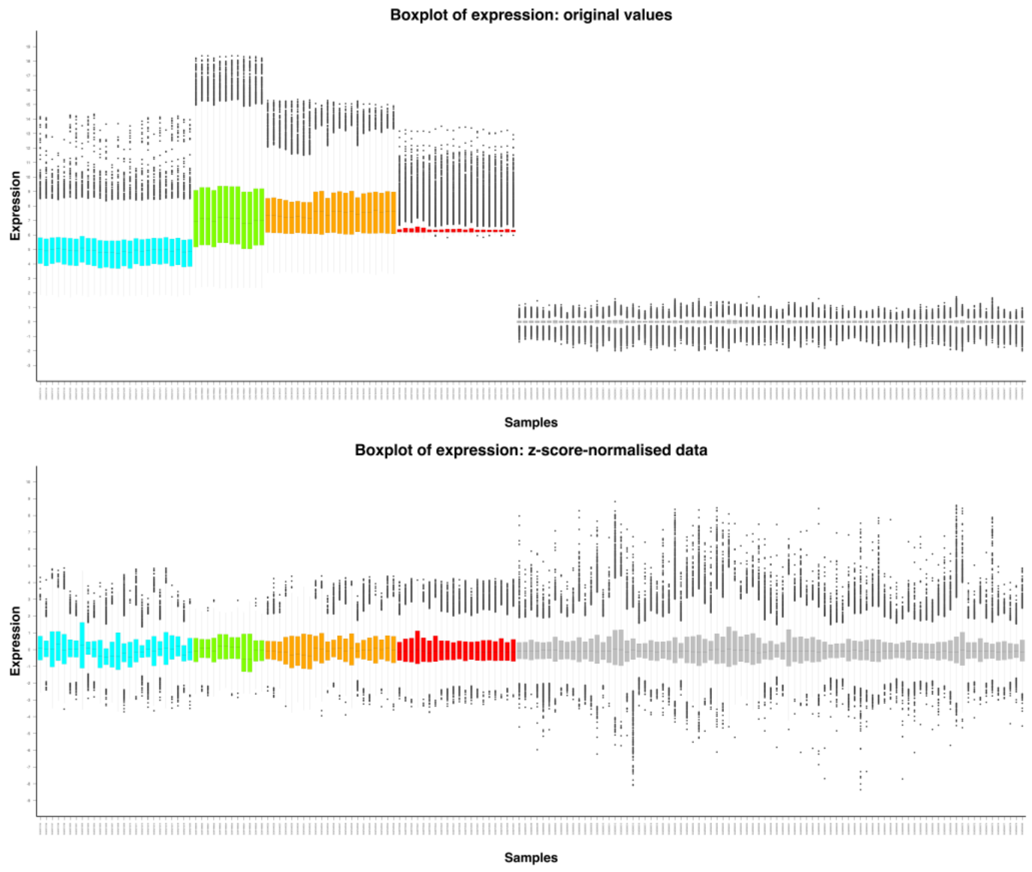
Some microarrays include multiple probes that match the same gene. This is intentional and done to capture different parts of a gene as it improves coverage and the accuracy of the calls. However, in downstream analysis we only need one expression value per gene per sample. To address this, we need to combine the information from all the probes that match the same gene. Therefore, for each gene, we find all the matched probes and calculate the variance of their expression values. Eventually, for each gene, we keep the matched probe with the highest variance (GSE_microarrays.R: rows 345-357, 450-522). Please note that various methods have been proposed in the past for this preprocessing step, but this is out of the scope of this manual.

7.1.3. Matching microarray probe sequences to genes

In some cases, such as for the "GSE61166" dataset, we might only have access to the sequence of the probes instead of the gene IDs matching to each probe. To obtain the gene IDs for each probe, so that we know the identity of genes and their expression, we used the BLASTN online tool (https://blast.ncbi.nlm.nih.gov/Blast.cgi). This tool accepts nucleotide sequences as input and outputs RefSeq IDs of genes with sequences that match the input sequences. We only keep RefSeq IDs of genes with sequences that had a 100% match with the provided nucleotide sequences. You can find the code for preparing the BLASTN input data and processing of the returned outputs from BLASTN in the rows 359-449 of the GSE_microarrays.R.

7.1.4. Normalization of gene expression counts

The final preprocessing step, before conducting the differential gene expression analysis, typically involves the normalization of the expression data. This step is crucial to ensure that all expression data, as obtained from various experiments, are on the same scale. While there are several normalization methods available, here we employ one of the most widely used: the z-normalization method. This method scales the expression data of all genes per sample to have a mean value of zero and a standard deviation of one. The code for applying the z-normalization method can be found in the rows 539-553 of the GSE_microarrays.R script. To get a better understanding of why normalization of gene expression counts is important, please see below two boxplot figures that show the expression data for each gene per sample before and after the z-normalization method:



As you can see, after the z-normalization (bottom boxplot), the expression values of all genes per sample are centered around zero (same scale), which was not the case before the normalization (top boxplot). This plot, along with other quality control plots can be found in the

QC/GSE_microarrays subdirectory on GitHub. All these plots were created using the GSE_microarrays.R script, rows 555-903.

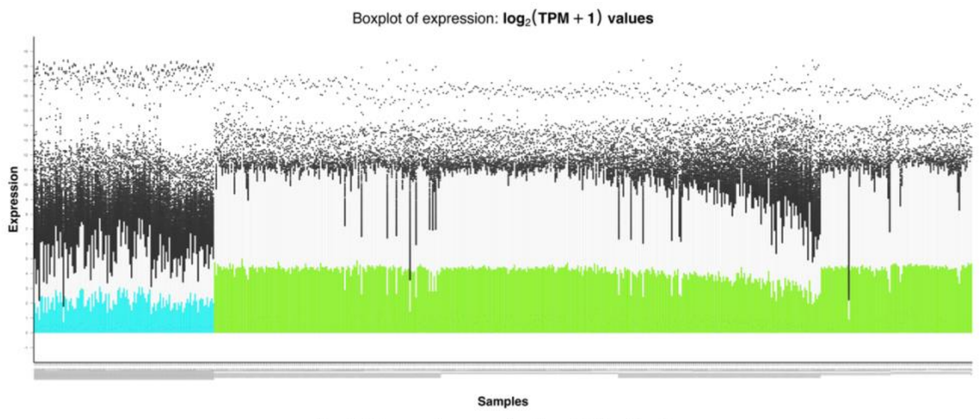
7.2. RNA-seq data

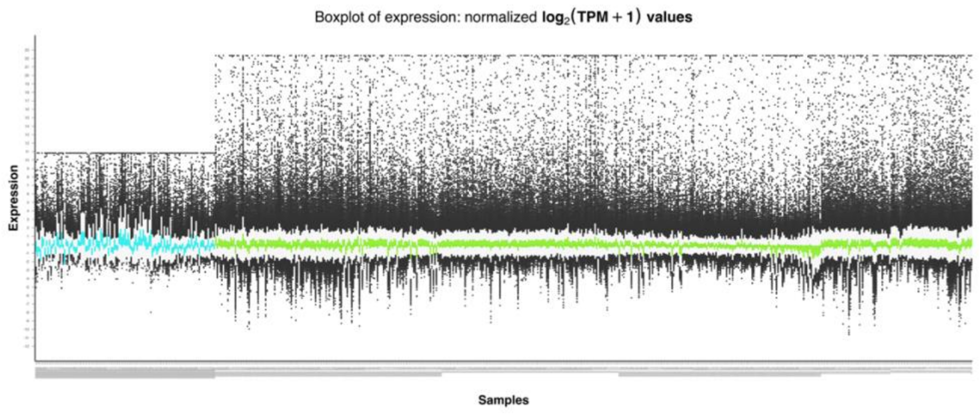
7.2.1. Convert raw counts to TPM

“GSE133684” provided gene expression data in transcripts per million (TPM), while “GSE194331” made available the raw gene expression data (raw counts). Consequently, we need to convert raw counts to TPM, so both datasets are on the same scale. To accomplish this, we first compute the Reads Per Kilobase (RPK) for each gene. This involves obtaining information about gene length (GSE_rnaseq.R rows: 159-221). Then, we calculate the RPK values for each gene by dividing the expression counts by the transcript length (in kilobases). Using the calculated RPK values, we compute the TPM. For those interested, the whole process is implemented and described in detail in rows 223-242 of the GSE_rnaseq.R script.

7.2.2. Normalization of gene expression counts

As with microarray data, the final preprocessing step before conducting the differential gene expression analysis in RNA-seq data is the normalization of the expression data. However, in the case of RNA-seq data, we are working with TPM values rather than counts. Therefore, we first log2 transform the TPM values and then apply z-normalization (GSE_rnaseq.R rows: 280-326). If you want to learn about the rationale behind the log2 transformation, you can read this thread. Below, you can see two boxplots showing the expression data of genes per sample before (top boxplot) and after (bottom boxplot) the log2 transformation and z-normalization:

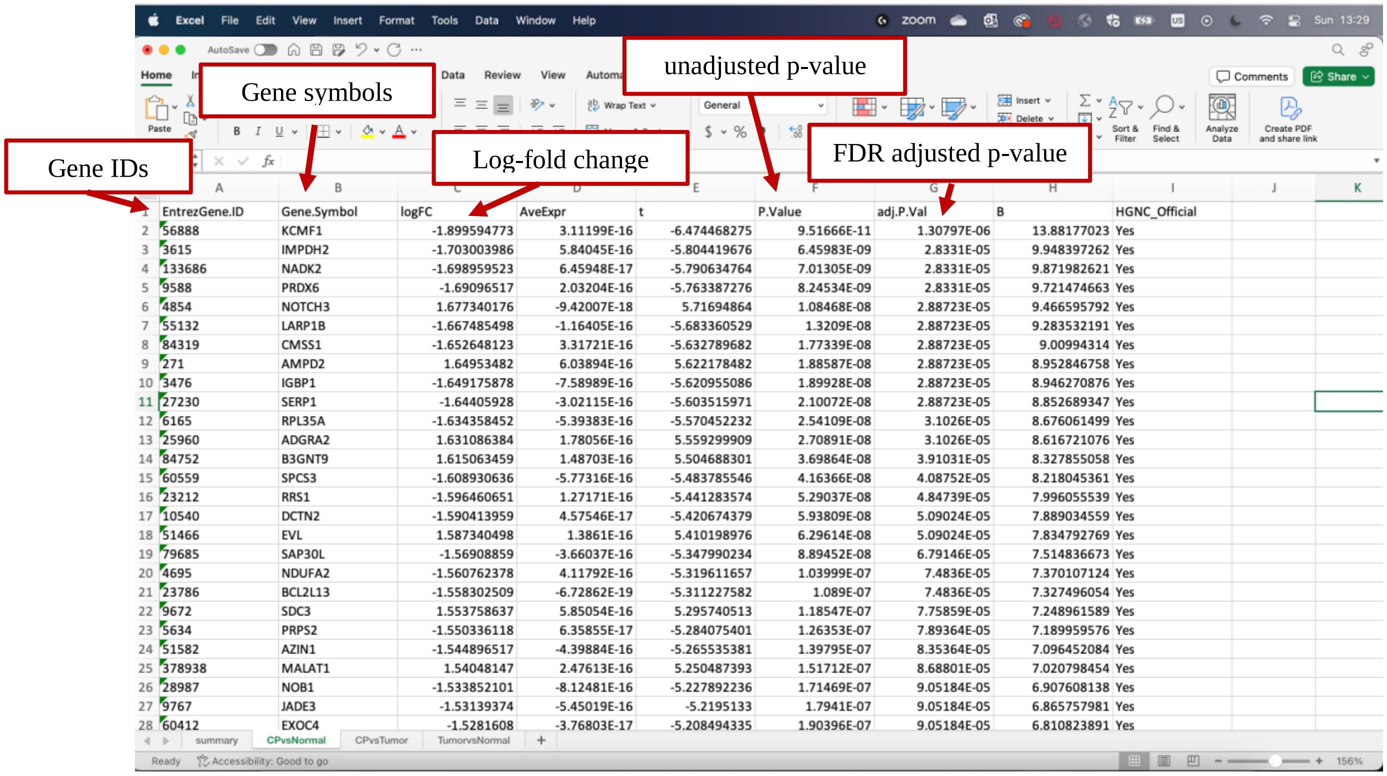




Again, we can see that after the normalization (bottom boxplot), the expression values of all genes per sample are centered around zero, which is not the case before normalization (top boxplot). This plot, along with other quality control plots, can be found in the QC/GSE_RNAseq subdirectory on GitHub. All the plots can be generated using the GSE_rnaseq.R script rows 348-644.

8. Differential gene expression analysis

After finishing the data preprocessing for both microarray and RNA-seq data, we are ready to conduct the differential gene expression analysis. To do so, we use the R package "limma". Limma employs a linear model for each gene and assesses whether its expression is statistically significantly different between two groups, such as pancreatitis and normal control groups. All the steps to perform the differential gene expression analysis using microarray data can be found in the GSE_microarrays.R rows 905-1012, and using RNA-seq data in the GSE_rnaseq.R rows 647-752. The output of these scripts is two excel files containing different statistics per gene. You can access them here. Below, we see the first rows of the one for the microarray data, but the same columns are found in the file for the RNA-seq data.



EntrezGene.ID	Gene.Symbol	logFC	AveExpr	t	P.Value	adj.P.Val	B	HGNC_Official
56888	KCMF1	-1.899594773	3.11199E-16	-6.474468275	9.51666E-11	1.30797E-06	13.88177023	Yes
3615	IMPDH2	-1.703003986	5.84045E-16	-5.804419676	6.45983E-09	2.8331E-05	9.948397262	Yes
133686	NADK2	-1.698959523	6.45948E-17	-5.790634764	7.01305E-09	2.8331E-05	9.871982621	Yes
9588	PRDX6	-1.69096517	2.03204E-16	-5.763387276	8.24534E-09	2.8331E-05	9.721474663	Yes
4854	NOTCH3	1.677340176	-9.42007E-18	5.71694864	1.08468E-08	2.88723E-05	9.466595792	Yes
55132	LARP1B	-1.667485498	-1.16405E-16	-5.683360529	1.3209E-08	2.88723E-05	9.283532191	Yes
84319	CMSS1	-1.652648123	3.31721E-16	-5.632789682	1.77339E-08	2.88723E-05	9.00994314	Yes
271	AMPD2	1.64953482	6.03894E-16	5.622178482	1.88587E-08	2.88723E-05	8.952846758	Yes
3476	IGBP1	-1.649175878	-7.58989E-16	-5.620955086	1.89928E-08	2.88723E-05	8.946270876	Yes
27230	SERP1	-1.64405928	-3.02115E-16	-5.603515971	2.10072E-08	2.88723E-05	8.852689347	Yes
6165	RPL35A	-1.634358452	-5.39383E-16	-5.570452232	2.54109E-08	3.1026E-05	8.676061499	Yes
25960	ADGRA2	1.631086384	1.78056E-16	5.559299909	2.70891E-08	3.1026E-05	8.616721076	Yes
84752	B3GNT9	1.615063459	1.48703E-16	5.504688301	3.69864E-08	3.91031E-05	8.327855058	Yes
60559	SPCS3	-1.608930636	-5.77316E-16	-5.483785546	4.16366E-08	4.08752E-05	8.218045361	Yes
23212	RRS1	-1.596460651	1.27171E-16	-5.441283574	5.29037E-08	4.84739E-05	7.996055539	Yes
10540	DCTN2	-1.590413959	4.57546E-17	-5.420674379	5.93809E-08	5.09024E-05	7.889034559	Yes
51466	EVL	1.587340498	1.3861E-16	5.410198976	6.29614E-08	5.09024E-05	7.834792769	Yes
79685	SAP30L	-1.56908859	-3.66037E-16	-5.347990234	8.89452E-08	6.79146E-05	7.514836673	Yes
4695	NDUFA2	-1.560762378	4.11792E-16	-5.319611657	1.03999E-07	7.4836E-05	7.370107124	Yes
23786	BCL2L13	-1.558302509	-6.72862E-19	-5.311227582	1.089E-07	7.4836E-05	7.327496054	Yes
9672	SDC3	1.553758637	5.85054E-16	5.295740513	1.18547E-07	7.75859E-05	7.248961589	Yes
5634	PRPS2	-1.550336118	6.35855E-17	-5.284075401	1.26353E-07	7.89364E-05	7.189959576	Yes
51582	AZIN1	-1.544896517	-4.39884E-16	-5.265535381	1.39795E-07	8.35364E-05	7.096452084	Yes
378938	MALAT1	1.54048147	2.47613E-16	5.250487393	1.51712E-07	8.68801E-05	7.020798454	Yes
28987	NOB1	-1.533852101	-8.12481E-16	-5.227892236	1.71469E-07	9.05184E-05	6.907608138	Yes
9767	JADE3	-1.53139374	-5.45019E-16	-5.2195133	1.7941E-07	9.05184E-05	6.865757981	Yes
60412	EXOC4	-1.5281608	-3.76803E-17	-5.208494335	1.90396E-07	9.05184E-05	6.810823891	Yes

Next, we use these two excel files to compare the results obtained by analyzing microarray and RNA-seq data.

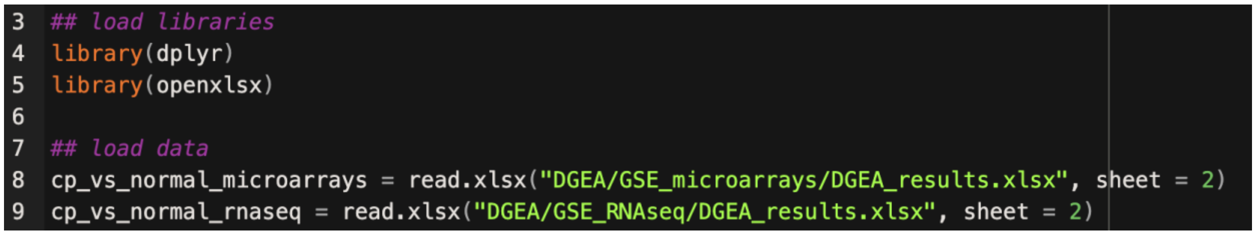
9. Comparison of results

9.1. Expected results

Before conducting any analysis, it is useful to think about what you expect the results to be. In our experiment, we anticipate observing differences in the two lists of differentially expressed genes for several reasons. First, the microarray and RNA-seq data we analyzed do not originate from the same individuals and different number of samples were included in each data. Ideally, we would want to have gene expression microarray and RNA-seq data from the same individuals to reduce the inter-individual variance. Second, microarray uses pre-defined probes, while RNA-seq does not. Consequently, we anticipate finding more differentially expressed genes in RNA-seq data compared to microarray data. Third, according to existing literature, although we expect most of the genes to be found dysregulated (downregulated or upregulated) in the same direction, we expect to observe higher fold-changes in the differentially expressed genes from RNA-seq data compared to microarray data. Having all these in mind, let's now compare the two lists of differentially expressed genes we have.

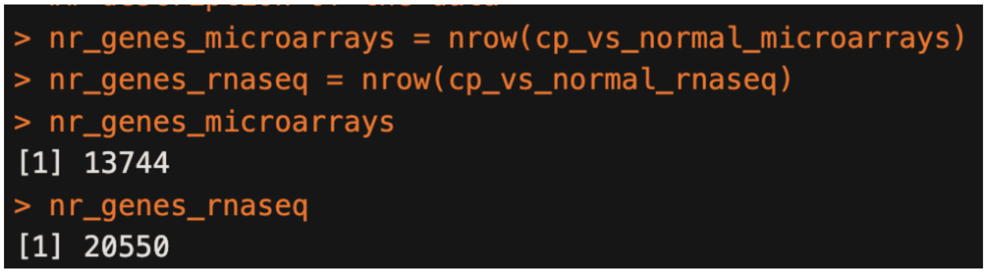
9.2. Description of the data

First, we load the two excel files containing the results from the differential gene expression analysis in R. As they are excel files, we use the function read.xlsx() from the "openxlsx" R package to load them. With the function head(), we can inspect the files.



```r
## load libraries
library(dplyr)
library(openxlsx)

## load data
cp_vs_normal_microarrays = read.xlsx("DGEA/GSE_microarrays/DGEA_results.xlsx", sheet = 2)
cp_vs_normal_rnaseq = read.xlsx("DGEA/GSE_RNAseq/DGEA_results.xlsx", sheet = 2)
```

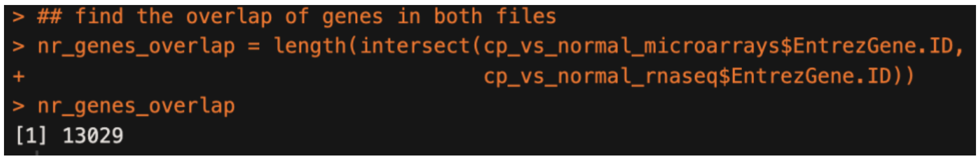
In both files, each row represents a gene. Therefore, the number of rows in each file tells us how many genes were tested for differential gene expression in each case.



```
> nr_genes_microarrays = nrow(cp_vs_normal_microarrays)
> nr_genes_rnaseq = nrow(cp_vs_normal_rnaseq)
> nr_genes_microarrays
[1] 13744
> nr_genes_rnaseq
[1] 20550
```

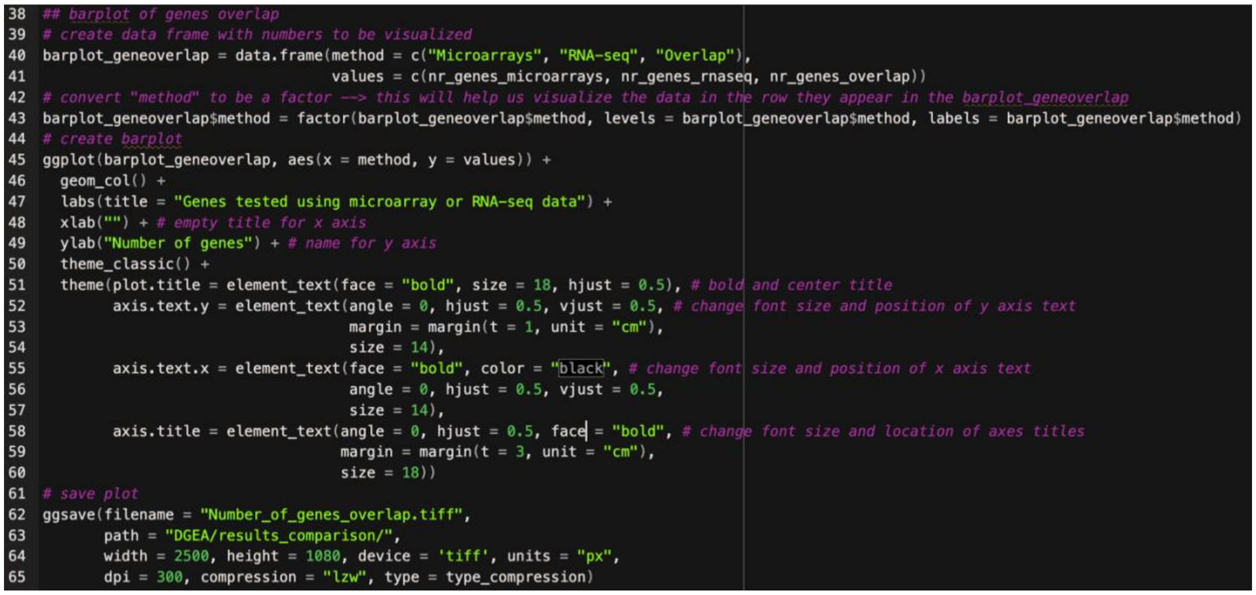
We have 13,744 genes for microarray and 20,550 genes for RNA-seq. This larger number of genes tested in the RNA-seq data compared to microarray data aligns with our expectations. As mentioned earlier, this happens because RNA-seq does not rely on pre-defined probes and can capture a larger number of genes compared to microarrays.

Next, we aim to determine how many genes were tested in both cases. In other words, we want to find the overlap of the genes in the two files. To do this, we use the intersect() function.



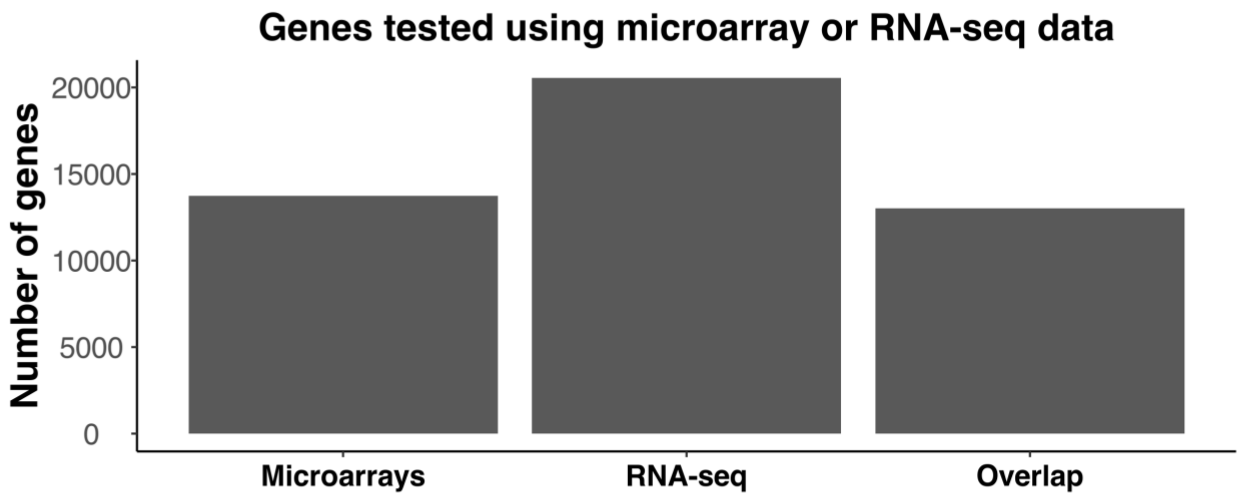
```
> ## find the overlap of genes in both files
> nr_genes_overlap = length(intersect(cp_vs_normal_microarrays$EntrezGene.ID,
+                                    cp_vs_normal_rnaseq$EntrezGene.ID))
> nr_genes_overlap
[1] 13029
```

We find that 13,029 genes appear in both files. Let’s create a barplot to visualize this information using ggplot2 and the following code:



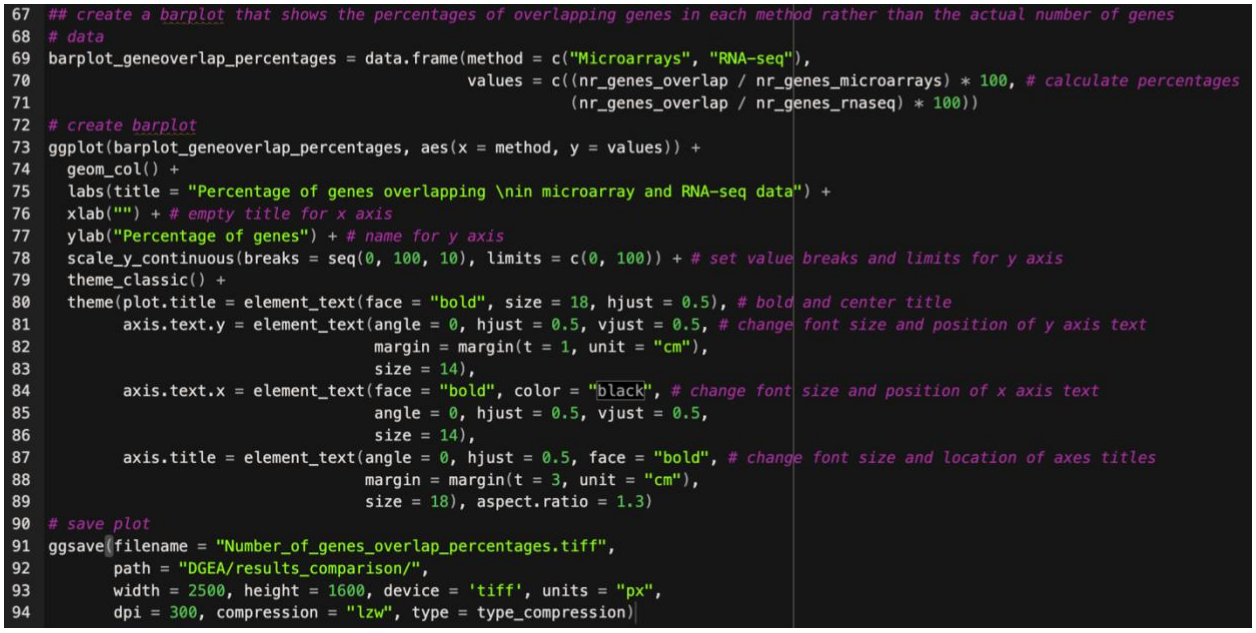
```
## barplot of genes overlap
# create data frame with numbers to be visualized
barplot_geneoverlap = data.frame(method = c("Microarrays", "RNA-seq", "Overlap"),
                                 values = c(nr_genes_microarrays, nr_genes_rnaseq, nr_genes_overlap))
# convert "method" to be a factor --> this will help us visualize the data in the row they appear in the barplot_geneoverlap
barplot_geneoverlap$method = factor(barplot_geneoverlap$method, levels = barplot_geneoverlap$method, labels = barplot_geneoverlap$method)
# create barplot
ggplot(barplot_geneoverlap, aes(x = method, y = values)) +
  geom_col() +
  labs(title = "Genes tested using microarray or RNA-seq data") +
  xlab("") + # empty title for x axis
  ylab("Number of genes") + # name for y axis
  theme_classic() +
  theme(plot.title = element_text(face = "bold", size = 18, hjust = 0.5), # bold and center title
        axis.text.y = element_text(angle = 0, hjust = 0.5, vjust = 0.5, # change font size and position of y axis text
                                   margin = margin(t = 1, unit = "cm"),
                                   size = 14),
        axis.text.x = element_text(face = "bold", color = "black", # change font size and position of x axis text
                                   angle = 0, hjust = 0.5, vjust = 0.5,
                                   size = 14),
        axis.title = element_text(angle = 0, hjust = 0.5, face = "bold", # change font size and location of axes titles
                                  margin = margin(t = 3, unit = "cm"),
                                  size = 18))
# save plot
ggsave(filename = "Number_of_genes_overlap.tiff",
       path = "DGEA/results_comparison/",
       width = 2500, height = 1080, device = 'tiff', units = "px",
       dpi = 300, compression = "lzw", type = type_compression)
```

The output is this:



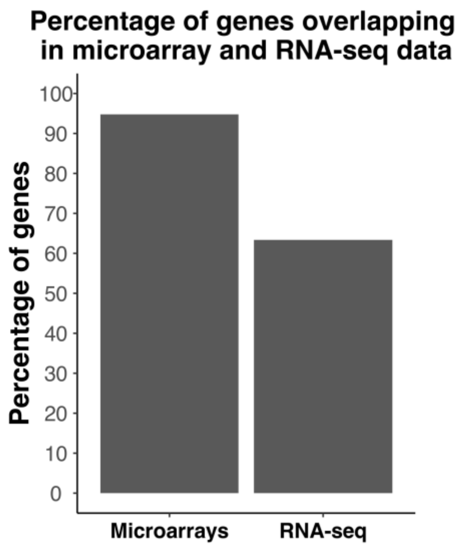
We can see that almost all the genes tested in microarrays were also tested in RNA-seq. The few genes that were not tested in RNA-seq might be genes with low expression values that happened not to be captured by RNA-seq, just by chance. This is attributed to the bias of both methods, which tend to capture transcripts that are more abundant in samples. Let’s now create a barplot that shows the percentages of overlapping genes in each method, instead of the number of genes.

The code is:



```
## create a barplot that shows the percentages of overlapping genes in each method rather than the actual number of genes
# data
barplot_geneoverlap_percentages = data.frame(method = c("Microarrays", "RNA-seq"),
                                             values = c((nr_genes_overlap / nr_genes_microarrays) * 100, # calculate percentages
                                                        (nr_genes_overlap / nr_genes_rnaseq) * 100))
# create barplot
ggplot(barplot_geneoverlap_percentages, aes(x = method, y = values)) +
  geom_col() +
  labs(title = "Percentage of genes overlapping \nin microarray and RNA-seq data") +
  xlab("") + # empty title for x axis
  ylab("Percentage of genes") + # name for y axis
  scale_y_continuous(breaks = seq(0, 100, 10), limits = c(0, 100)) + # set value breaks and limits for y axis
  theme_classic() +
  theme(plot.title = element_text(face = "bold", size = 18, hjust = 0.5), # bold and center title
        axis.text.y = element_text(angle = 0, hjust = 0.5, vjust = 0.5, # change font size and position of y axis text
                                   margin = margin(t = 1, unit = "cm"),
                                   size = 14),
        axis.text.x = element_text(face = "bold", color = "black", # change font size and position of x axis text
                                   angle = 0, hjust = 0.5, vjust = 0.5,
                                   size = 14),
        axis.title = element_text(angle = 0, hjust = 0.5, face = "bold", # change font size and location of axes titles
                                  margin = margin(t = 3, unit = "cm"),
                                  size = 18), aspect.ratio = 1.3)
# save plot
ggsave(filename = "Number_of_genes_overlap_percentages.tiff",
       path = "DGEA/results_comparison/",
       width = 2500, height = 1600, device = 'tiff', units = "px",
       dpi = 300, compression = "lzw", type = type_compression)
```

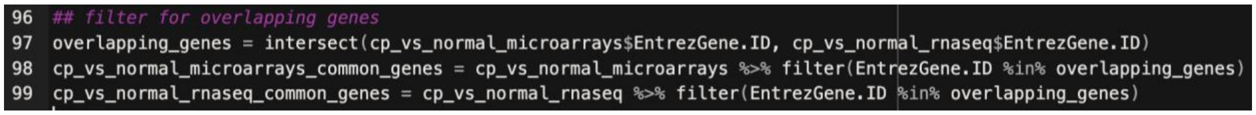
The output is:



Again, we can see that almost all the microarray genes are present in the RNA-seq data. However, approximately 35% of the RNA-seq genes were not found in the microarray data.

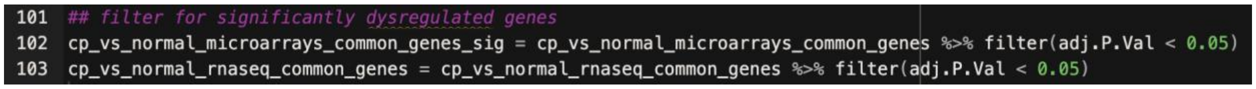
9.3. Overlap of significantly dysregulated genes

Let's now test if both methods identify genes to be dysregulated in the same direction. To do so, we first need to filter for the overlapping genes. We can use the intersect() function to find the overlapping genes and then use dplyr syntax to filter both variables for the overlapping genes:



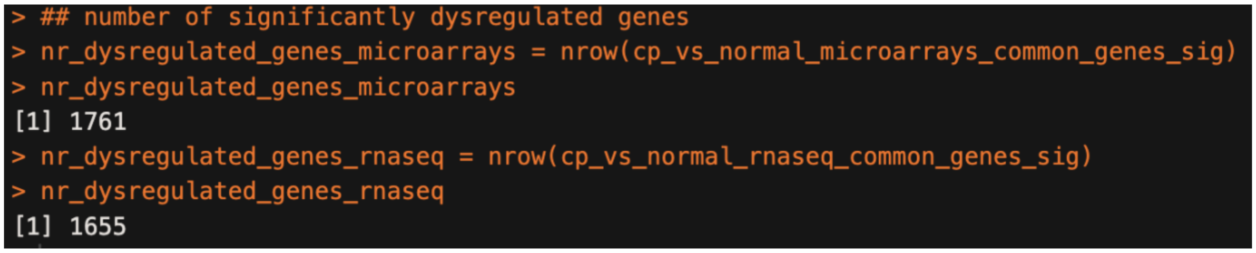
```
## filter for overlapping genes
overlapping_genes = intersect(cp_vs_normal_microarrays$EntrezGene.ID, cp_vs_normal_rnaseq$EntrezGene.ID)
cp_vs_normal_microarrays_common_genes = cp_vs_normal_microarrays %>% filter(EntrezGene.ID %in% overlapping_genes)
cp_vs_normal_rnaseq_common_genes = cp_vs_normal_rnaseq %>% filter(EntrezGene.ID %in% overlapping_genes)
```

We then filter for the statistically significantly dysregulated genes. We do this by filtering both data frames for adjusted p.value < 0.05 (this is a standard threshold of significance used in research):



```
## filter for significantly dysregulated genes
cp_vs_normal_microarrays_common_genes_sig = cp_vs_normal_microarrays_common_genes %>% filter(adj.P.Val < 0.05)
cp_vs_normal_rnaseq_common_genes = cp_vs_normal_rnaseq_common_genes %>% filter(adj.P.Val < 0.05)
```

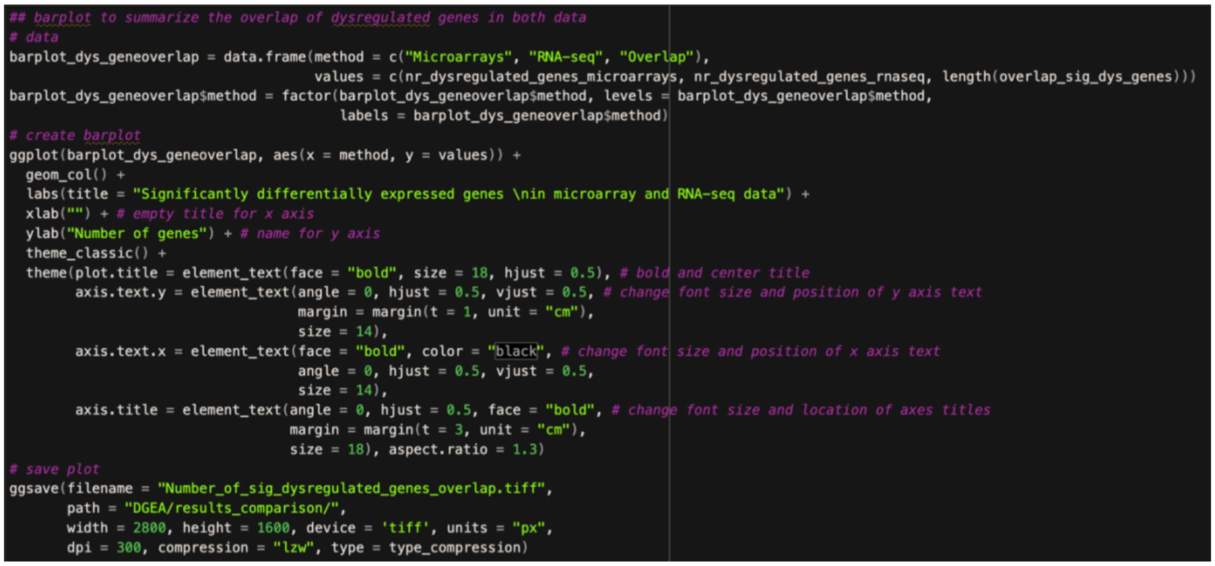
Let's see how many genes were found to be dysregulated in each case:



```
> ## number of significantly dysregulated genes
> nr_dysregulated_genes_microarrays = nrow(cp_vs_normal_microarrays_common_genes_sig)
> nr_dysregulated_genes_microarrays
[1] 1761
> nr_dysregulated_genes_rnaseq = nrow(cp_vs_normal_rnaseq_common_genes_sig)
> nr_dysregulated_genes_rnaseq
[1] 1655
```

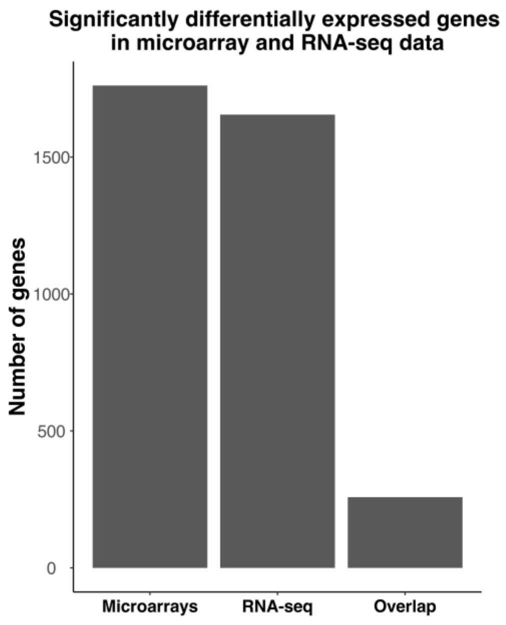
A total of 1,761 genes were found to be dysregulated using microarray data, while 1,655 genes were found to be dysregulated using RNA-seq data. Out of these, only 258 were significantly dysregulated in both experiments. This is a surprisingly low number, as we had expected most significantly dysregulated genes to be identified by both methods. If we apply a second filter requiring the absolute log fold-change to be greater than one (|logFC| > 1), we find 829 differentially expressed genes for microarray data but 0 for the RNA-seq data. Several reasons could be responsible for this discrepancy. Inter-individual variability might be one of them since we did not have microarray and RNA-seq data from the same individuals, or the fewer replicates for the RNA-seq data. Another potential reason could be related to the normalization method we employed for RNA-seq. It is important to note that different normalization methods can yield different results. Additionally, the imputation step with the microarray data could also play a role. To gain further insights, we could repeat the analysis by changing the aforementioned preprocessing methods. However, this is outside of the scope of this manual.

As previously done, let's create a barplot using ggplot2 that summarizes these results. The code is the following:

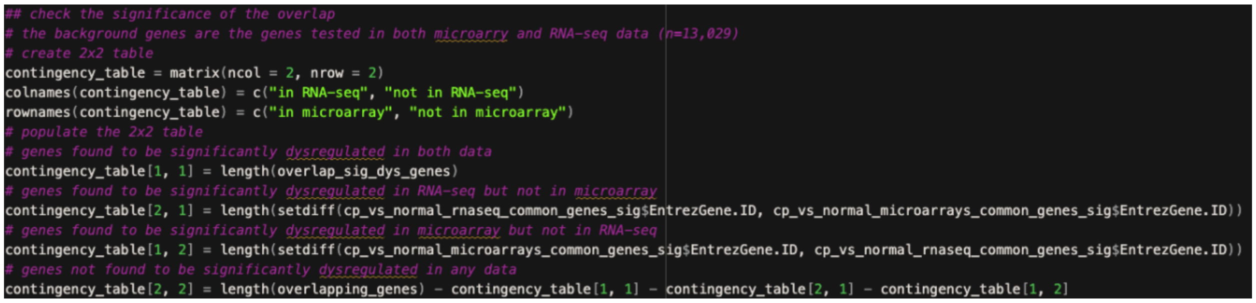


```r
## barplot to summarize the overlap of dysregulated genes in both data
# data
barplot_dys_geneoverlap = data.frame(method = c("Microarrays", "RNA-seq", "Overlap"),
                                     values = c(nr_dysregulated_genes_microarrays, nr_dysregulated_genes_rnaseq, length(overlap_sig_dys_genes)))
barplot_dys_geneoverlap$method = factor(barplot_dys_geneoverlap$method, levels = barplot_dys_geneoverlap$method,
                                        labels = barplot_dys_geneoverlap$method)
# create barplot
ggplot(barplot_dys_geneoverlap, aes(x = method, y = values)) +
  geom_col() +
  labs(title = "Significantly differentially expressed genes \nin microarray and RNA-seq data") +
  xlab("") + # empty title for x axis
  ylab("Number of genes") + # name for y axis
  theme_classic() +
  theme(plot.title = element_text(face = "bold", size = 18, hjust = 0.5), # bold and center title
        axis.text.y = element_text(angle = 0, hjust = 0.5, vjust = 0.5, # change font size and position of y axis text
                                   margin = margin(t = 1, unit = "cm"),
                                   size = 14),
        axis.text.x = element_text(face = "bold", color = "black", # change font size and position of x axis text
                                   angle = 0, hjust = 0.5, vjust = 0.5,
                                   size = 14),
        axis.title = element_text(angle = 0, hjust = 0.5, face = "bold", # change font size and location of axes titles
                                  margin = margin(t = 3, unit = "cm"),
                                  size = 18), aspect.ratio = 1.3)
# save plot
ggsave(filename = "Number_of_sig_dysregulated_genes_overlap.tiff",
       path = "DGEA/results_comparison/",
       width = 2800, height = 1600, device = 'tiff', units = "px",
       dpi = 300, compression = "lzw", type = type_compression)
```

The output is:

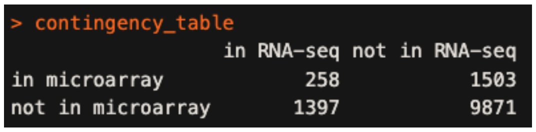


We see that the number of commonly significantly dysregulated genes in the two datasets is very low (most right bar). However, from this visualization we cannot tell if the overlap is significant or not. To test this, we will run a Fisher's exact test (hypergeometric test). In this test, we will ask if the overlap we observe is greater than expected by chance. To do so, we need to create a 2x2 table which will contain the information about how many genes were found to be significantly dysregulated in both studies, only in microarray, only in RNA-seq or in none of them. The code we will use is this:



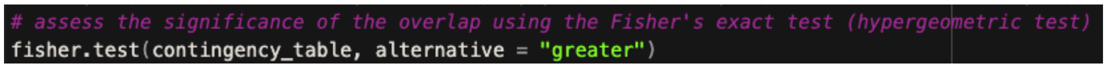
```
## check the significance of the overlap
# the background genes are the genes tested in both microarry and RNA-seq data (n=13,029)
# create 2x2 table
contingency_table = matrix(ncol = 2, nrow = 2)
colnames(contingency_table) = c("in RNA-seq", "not in RNA-seq")
rownames(contingency_table) = c("in microarray", "not in microarray")
# populate the 2x2 table
# genes found to be significantly dysregulated in both data
contingency_table[1, 1] = length(overlap_sig_dys_genes)
# genes found to be significantly dysregulated in RNA-seq but not in microarray
contingency_table[2, 1] = length(setdiff(cp_vs_normal_rnaseq_common_genes_sig$EntrezGene.ID, cp_vs_normal_microarrays_common_genes_sig$EntrezGene.ID))
# genes found to be significantly dysregulated in microarray but not in RNA-seq
contingency_table[1, 2] = length(setdiff(cp_vs_normal_microarrays_common_genes_sig$EntrezGene.ID, cp_vs_normal_rnaseq_common_genes_sig$EntrezGene.ID))
# genes not found to be significantly dysregulated in any data
contingency_table[2, 2] = length(overlapping_genes) - contingency_table[1, 1] - contingency_table[2, 1] - contingency_table[1, 2]
```

The created 2x2 table should look like this:

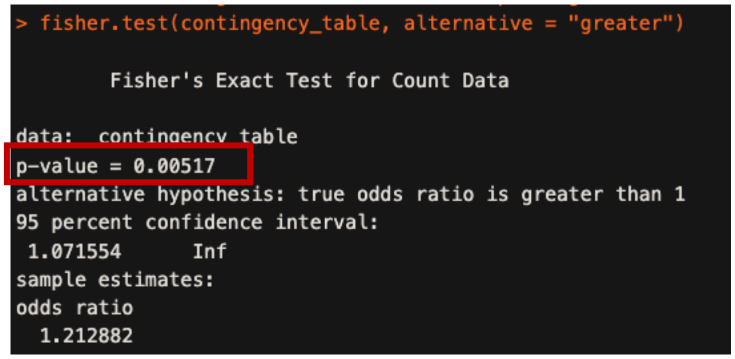


```
> contingency_table
                  in RNA-seq not in RNA-seq
in microarray            258           1503
not in microarray       1397           9871
```

Then, we run the Fisher's exact test using the fisher.test() function:



```
# assess the significance of the overlap using the Fisher's exact test (hypergeometric test)
fisher.test(contingency_table, alternative = "greater")
```

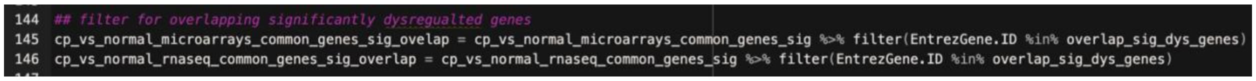


```
> fisher.test(contingency_table, alternative = "greater")

	Fisher's Exact Test for Count Data

data:  contingency_table
p-value = 0.00517
alternative hypothesis: true odds ratio is greater than 1
95 percent confidence interval:
 1.071554      Inf
sample estimates:
odds ratio
  1.212882
```

As we can see, the overlap seems to be significant (p = 0.005 < 0.05).

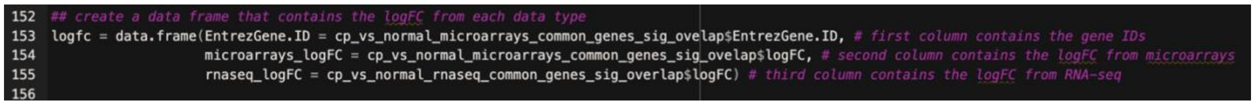
9.4. Direction of dysregulation in overlapping genes

Next, let’s investigate whether the overlapping significantly differentially expressed genes exhibit dysregulation in the same direction in both microarray and RNA-seq data. To accomplish this, we first need to filter our two gene lists for these genes:



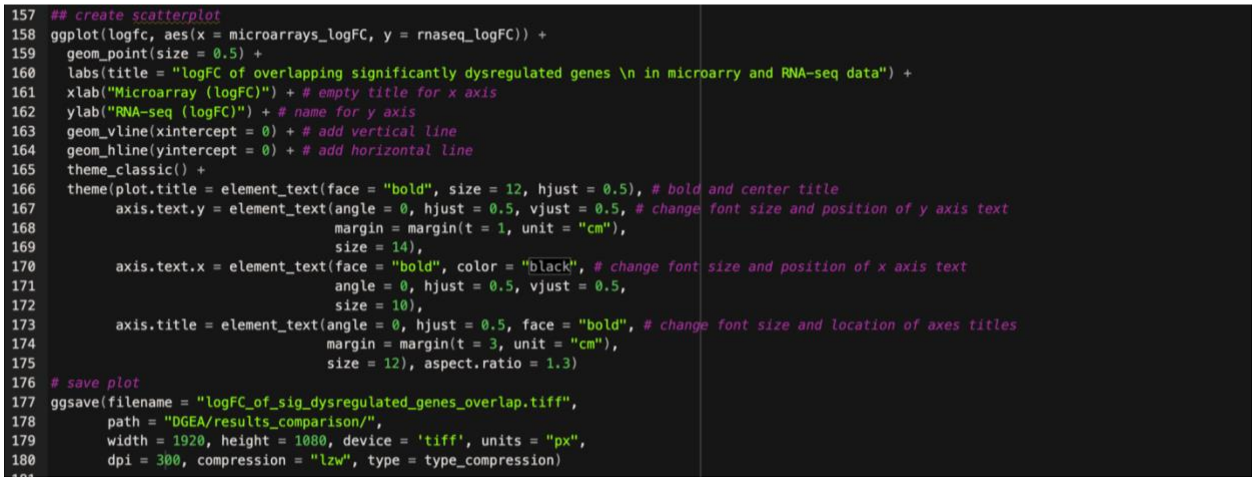
```
## filter for overlapping significantly dysregualted genes
cp_vs_normal_microarrays_common_genes_sig_ovelap = cp_vs_normal_microarrays_common_genes_sig %>% filter(EntrezGene.ID %in% overlap_sig_dys_genes)
cp_vs_normal_rnaseq_common_genes_sig_overlap = cp_vs_normal_rnaseq_common_genes_sig %>% filter(EntrezGene.ID %in% overlap_sig_dys_genes)
```

Now, we use the “logFC” column (log-fold change) to create a scatterplot. In this scatterplot, each point represents a gene. The x-axis depicts the log fold-change of each gene as found by the microarray data, while the y-axis represents the log fold-change of each gene as found by the RNA-seq data. To do this, we first should create a data frame with three columns. The first column should be the gene identifiers (EntrezID), the second the log-fold change in microarrays and the third the log-fold change in the RNA-seq data.



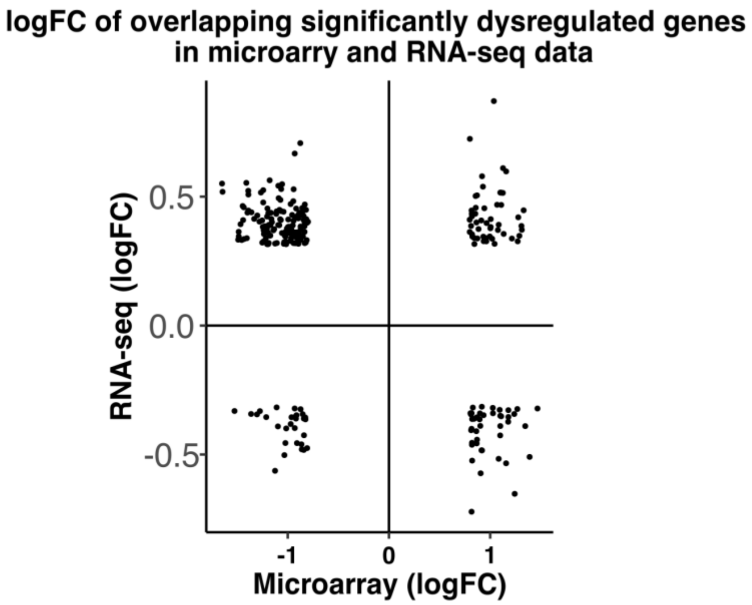
```
## create a data frame that contains the logFC from each data type
logfc = data.frame(EntrezGene.ID = cp_vs_normal_microarrays_common_genes_sig_ovelap$EntrezGene.ID, # first column contains the gene IDs
                   microarrays_logFC = cp_vs_normal_microarrays_common_genes_sig_ovelap$logFC, # second column contains the logFC from microarrays
                   rnaseq_logFC = cp_vs_normal_rnaseq_common_genes_sig_overlap$logFC) # third column contains the logFC from RNA-seq
```

Next, we create the scatterplot using ggplot2. The code is:



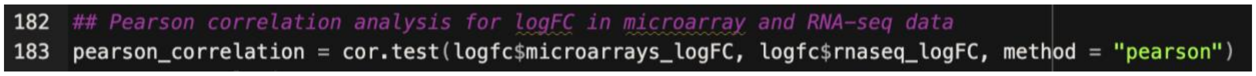
```
## create scatterplot
ggplot(logfc, aes(x = microarrays_logFC, y = rnaseq_logFC)) +
  geom_point(size = 0.5) +
  labs(title = "logFC of overlapping significantly dysregulated genes \n in microarry and RNA-seq data") +
  xlab("Microarray (logFC)") + # empty title for x axis
  ylab("RNA-seq (logFC)") + # name for y axis
  geom_vline(xintercept = 0) + # add vertical line
  geom_hline(yintercept = 0) + # add horizontal line
  theme_classic() +
  theme(plot.title = element_text(face = "bold", size = 12, hjust = 0.5), # bold and center title
        axis.text.y = element_text(angle = 0, hjust = 0.5, vjust = 0.5, # change font size and position of y axis text
                                   margin = margin(t = 1, unit = "cm"),
                                   size = 14),
        axis.text.x = element_text(face = "bold", color = "black", # change font size and position of x axis text
                                   angle = 0, hjust = 0.5, vjust = 0.5,
                                   size = 10),
        axis.title = element_text(angle = 0, hjust = 0.5, face = "bold", # change font size and location of axes titles
                                  margin = margin(t = 3, unit = "cm"),
                                  size = 12), aspect.ratio = 1.3)
# save plot
ggsave(filename = "logFC_of_sig_dysregulated_genes_overlap.tiff",
       path = "DGEA/results_comparison/",
       width = 1920, height = 1080, device = 'tiff', units = "px",
       dpi = 300, compression = "lzw", type = type_compression)
```

The output is:



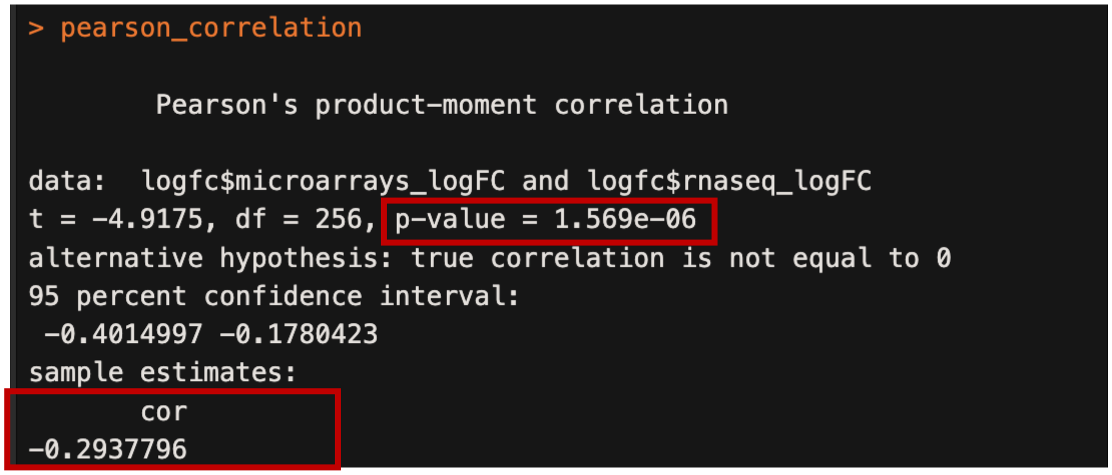
If there was complete concordance between the results from microarray and RNA-seq data, we would expect all the points/genes to fall in the top-right or bottom-left quartiles. However, we observe many points in the top-left and bottom-right quartiles, indicating that there are numerous genes found to have opposite log-fold change values in microarrays compared to RNA-seq data.

With this in mind, let's test if there is a significant correlation between the log-fold change of dysregulated genes in microarray and RNA-seq data. To do this, we can run a Pearson correlation test using the cor.test() function and defining the parameter method to be "pearson":



```
## Pearson correlation analysis for logFC in microarray and RNA-seq data
pearson_correlation = cor.test(logfc$microarrays_logFC, logfc$rnaseq_logFC, method = "pearson")
```

The output:



```
> pearson_correlation

	Pearson's product-moment correlation

data:  logfc$microarrays_logFC and logfc$rnaseq_logFC
t = -4.9175, df = 256, p-value = 1.569e-06
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.4014997 -0.1780423
sample estimates:
       cor
-0.2937796
```

We observe a negative correlation (cor = -0.29). This suggests that the majority of significantly differentially expressed genes between chronic pancreatitis and normal samples were found to be dysregulated in opposite directions in microarray and RNA-seq data. Additionally, we find that this correlation is statistically significant (p-value = 1.569e-06 < 0.05). This finding is surprising, as we initially expected the log-fold change to be in the same direction for the same genes in both datasets.

10. Conclusion

While there is a significant overlap in the significantly dysregulated genes identified from the analysis of both microarray and RNA-seq data, there are significant discrepancies in the direction of dysregulation. However, given that the analyzed microarray and RNA-seq data were not obtained from the same individuals, we cannot draw definitive conclusions. Furthermore, some of the decisions made during the preprocessing steps, such as selection of imputation and normalization methods, may impact the results. Nevertheless, it appears that the method of analysis and expression inference can have an impact on the findings and researchers should keep this in mind when they compare differential gene expression results from different methods and studies.

11. References

1. Kaklamani, V.G. and Gradishar, W.J. (2006) ‘Gene expression in breast cancer’, *Current Treatment Options in Oncology*, 7(2), pp. 123–128. doi:10.1007/s11864-006-0047-0.
2. Wang, C. *et al.* (2014) ‘The concordance between RNA-seq and microarray data depends on chemical treatment and transcript abundance’, *Nature Biotechnology*, 32(9), pp. 926 932. doi:10.1038/nbt.3001.
3. Zhao, S. *et al.* (2014) ‘Comparison of RNA-seq and microarray in transcriptome profiling of activated T cells’, *PLoS ONE*, 9(1). doi:10.1371/journal.pone.0078644