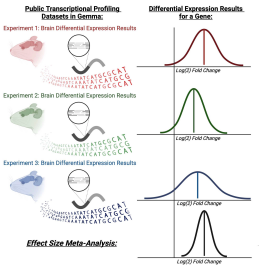
Aug 22, 2024

Brain Data Alchemy Project: Meta-Analysis of Re-Analyzed Public Transcriptional Profiling Data in the Gemma Database



DOI
dx.doi.org/10.17504/protocols.io.j8nlk84jxl5r/v1

Megan Hagenauer[1], Cosette Rhoads[2], Jinglin Xiong[3], Duy Manh Nguyen[4], Erin Hernandez[4], Annaka Saffron[5], Amrita Kondur[1], Elizabeth Flandreau[6]

[1]University of Michigan; [2]NIH/NIMH; [3]Stanford University; [4]Grinnell College; [5]Johns Hopkins University; [6]Grand Valley State University

Megan Hagenauer
University of Michigan

OPEN ACCESS

DOI: dx.doi.org/10.17504/protocols.io.j8nlk84jxl5r/v1

Protocol Citation: Megan Hagenauer, Cosette Rhoads, Jinglin Xiong, Duy Manh Nguyen, Erin Hernandez, Annaka Saffron, Amrita Kondur, Elizabeth Flandreau 2024. Brain Data Alchemy Project: Meta-Analysis of Re-Analyzed Public Transcriptional Profiling Data in the Gemma Database. **protocols.io https://dx.doi.org/10.17504/protocols.io.j8nlk84jxl5r/v1**

Protocol status: Working
We use this protocol and it's working

Created: April 29, 2024

Last Modified: August 22, 2024

Protocol Integer ID: 98973

Keywords: Transcriptional Profiling, RNA-Seq, Microarray, Gemma, Meta-Analysis, Gene Expression, Summer Program, R Programming, PRISMA, Systematic Meta-Analysis

Funders Acknowledgement:
Hope for Depression Research Foundation (HDRF)
Grant ID: Data Center
Pritzker Neuropsychiatric Disorders Research Foundation
Grant ID: NA
National Institute on Drug Abuse (NIDA)
Grant ID: U01DA043098
Grinnell College Center for Careers, Life, and Service
Grant ID: NA
University of Michigan Undergraduate Research Opportunities Program (UROP)
Grant ID: NA

Disclaimer

This protocol provides meta-analysis guidelines and sample code, but not a full reproducible analysis pipeline and coding environment. It was meant to be adapted to the needs of each individual project.

Abstract

Over the past two decades, transcriptional profiling has become an increasingly common tool for investigating the effects of diseases and experimental manipulations on the nervous system. Within transcriptional profiling experiments, microarray or sequencing technologies are used to measure the relative amount of RNA transcript for each of the thousands of genes expressed in a sample. The primary objective of these experiments is to identify genes that are differentially expressed in response to conditions of interest. However, transcriptional profiling experiments have traditionally been conducted with small sample sizes due to expense (e.g., n=3-10/group), resulting in low statistical power. Due to low power, these experiments are prone to capturing large technical artifacts and false positives rather than smaller biological effects of interest.
To address this issue, we developed a 10-week summer research program (*The Brain Data Alchemy Project*) that guides participants through the process of performing a meta-analysis of differential expression effect sizes (Log2 Fold Change or Log2FC) extracted from publicly available transcriptional profiling datasets. To conduct our meta-analyses, we leverage the efforts of the Gemma project, which has curated, preprocessed, and re-analyzed over 19,000 publicly available datasets (https://gemma.msl.ubc.ca/home.html). Participants learn the fundamental principles of systematic review and R programming to conduct the dataset search, result extraction, and whole transcriptome meta-analysis.
This protocol outlines the methods used during the first pilot year of the program in 2022.

Image Attribution

Graphical abstract created with BioRender.com

Guidelines

Any questions regarding this protocol should be directed to Dr. Megan Hagenauer (University of Michigan: hagenaue@umich.edu).

This protocol outlines the meta-analysis methods used by participants within the Brain Data Alchemy Program during the first pilot year of the program (2022). These methods were updated for later years - for updates, see the associated forks for this protocol.

Materials

A computer and internet access.

Safety warnings

N/A

Project Preparation: Environment Set-Up

1 Install R and RStudio

Note

R is a programming language for statistical computing and data visualization. We will be running our analyses using R.
RStudio is an integrated development environment for R (i.e., a software interface that facilitates working in R). We will use the RStudio interface to make writing and editing code easier.

1.1 Install R

Software

R programming language	NAME
The R Foundation	DEVELOPER
Comprehensive R Archive Network	SOURCE LINK

Note

Go to: https://cran.r-project.org
- Choose the precompiled binary distributions of the base system and contributed packages
- If there is an option, use the CRAN “mirror” nearest to you to minimize network load.
- This protocol was designed using the version of R (v.4.2.0) available on 06/01/2022. It is likely to work using other versions.

1.2 Install R Studio

Software

NAME

R Studio Desktop

DEVELOPER

The R Studio, Inc.

Note

Go to: **https://posit.co/download/rstudio-desktop/**
- Install the appropriate version for your operating system
- This protocol was designed using the version of RStudio (v.2022.02.4) available on 06/01/2022. It is likely to work using other versions.

2 Follow the Brain Data Alchemy code repository on Github

Note

Github is a website (and desktop app) that enables easy code sharing, collaboration, and version control. We use Github in its most basic format for this project (easy code sharing).

2.1 Sign up for a free account on Github

Note

Go to: **https://github.com/**

2.2 Follow the Brain Data Alchemy code repository

Note

Go to: https://github.com/hagenaue/BrainDataAlchemy
In the upper right hand corner, click on either "watch" or "star" so that you can easily navigate back. "Watch" means that you will receive updates whenever anything in the repository is changed, whereas "Star" means that you have saved the repository to a list (sort of like a bookmark).

2.3 Optional: Install GitHub Desktop

Software

NAME	DEVELOPER
GitHub Desktop	GitHub

Note

If you want to use Github more fully (for actual version control), GitHub Desktop can make that easier:
Go to: **https://desktop.github.com/**

- Install the version of Github desktop that is compatible with your operating system

Dataset Identification: Pre-specification of Search Strategy

3 We will first plan the search strategy that we will use to identify useful datasets for the meta-analysis.

Note

In order to avoid biasing our meta-analysis, it is important that we pre-specify the search strategy that we will use to identify our datasets *before we learn anything about the results from that search*.
Full guidelines about how to conduct a systematic meta-analysis can be found here:
https://www.prisma-statement.org/

CITATION

Moher D, Liberati A, Tetzlaff J, Altman DG, PRISMA Group (2009). Preferred reporting items for systematic reviews and meta-analyses: the PRISMA statement..
LINK
https://doi.org/10.1136/bmj.b2535

CITATION

Liberati A, Altman DG, Tetzlaff J, Mulrow C, Gøtzsche PC, Ioannidis JP, Clarke M, Devereaux PJ, Kleijnen J, Moher D (2009). The PRISMA statement for reporting systematic reviews and meta-analyses of studies that evaluate healthcare interventions: explanation and elaboration..
LINK
https://doi.org/10.1136/bmj.b2700

3.1 The scope of our meta-analysis search is pre-specified below. The scope will be the same for each meta-analysis conducted using the Brain Data Alchemy analysis pipeline.

Note

For our meta-analysis, we will only be considering:
1) Publicly-available transcriptional profiling datasets (gene expression microarray or RNA-Sequencing) from laboratory rodents (rats, mice).
2) We will only use datasets that profiled RNA extracted from bulk tissue dissections (i.e., not from a particular cell type or small tissue subregion).
3) We will only use datasets that profiled either the full transcriptome or a large representation of the full transcriptome. We will not use datasets targeted at a particular subpopulation of transcripts (e.g., Chip-Seq, miRNA, TRAP-Seq, etc).

3.2 The information source will be the same for all meta-analyses: The Gemma Database.

Note

We will be leveraging the efforts of the Gemma project to curate, preprocess, and re-analyze publicly-available transcriptional profiling datasets:
https://gemma.msl.ubc.ca/home.html
The Gemma database contains >19,000 re-analyzed transcriptional profiling datasets.

Gemma performs the following data preprocessing and analysis steps:
1) Probe alignment or read mapping is redone whenever a new genome assembly is available.
2) Identification and removal of outlier samples.
3) Filtering out genes (rows) with minimal variance (either zero variance or <70% distinct values).
4) Batch correction: When confounded, batches are split into separate analyses.
5) Manual curation of common issues.
6) Differential expression output from omnibus (full dataset) tests examining the effects of the primary variables of interest as well as individual statistical contrasts.

CITATION

Zoubarev A, Hamer KM, Keshav KD, McCarthy EL, Santos JR, Van Rossum T, McDonald C, Hall A, Wan X, Lim R, Gillis J, Pavlidis P (2012). Gemma: a resource for the reuse, sharing and meta-analysis of expression profiling data..
LINK
https://doi.org/10.1093/bioinformatics/bts430

CITATION

Lim N, Tesar S, Belmadani M, Poirier-Morency G, Mancarci BO, Sicherman J, Jacobson M, Leong J, Tan P, Pavlidis P (2021). Curation of over 10 000 transcriptomic studies to enable data reuse..
LINK
https://doi.org/10.1093/database/baab006

3.3 Choose your meta-analysis topic and primary research question.

Note

Not all topics or research questions have been studied extensively using brain transcriptional profiling data. At this point, before pursuing the meta-analysis further, it would be helpful to have a second researcher who is not directly involved in the project (e.g., research mentor) quickly double-check whether there appears to be datasets in the Gemma database related to the chosen topic.

This quick, initial review should only use a handful of the most basic search terms for the topic. It is often advisable to wait to specify the tissue of interest (e.g., brain region) until later after you have determined how many the available datasets will survive the initial inclusion/exclusion criteria (protocol section 5).

The 2022 cohort of the Brain Data Alchemy Project chose to examine the effects of chronic stress, sleep deprivation or sleep restriction, antidepressant treatment, lipopolysaccharide exposure, and viral encephalitis.

3.4 Pre-specify the search terms for your topic.

Note

The search terms should be thoroughly brainstormed to catch any dataset related to the meta-analysis topic. e.g.,

- Topic
- Synonyms for Topic
- Abbreviations for Topic
- Subtypes of Topic
- Umbrella Categories for Topic

Since this is a critical step, feedback on search terms is important. For the 2022 cohort of the Brain Data Alchemy Project, these terms were reviewed by the mentor and whole cohort.

3.5 Examine the search terms more carefully and specify the formal search syntax.

Note

We will be conducting our dataset search using an R coding package that accesses the API (application programming interface) for the Gemma database. Gemma's API uses formal search syntax (i.e., search term construction). This syntax is not as flexible (or smart!) as a Google search, and differs in some ways from more conventional databases such as PubMed or Embase.

Examine your search terms carefully to define your formal syntax:

- Are there a variety of potential endings for any of the terms? (e.g., "stress" vs. "stressful" vs. "stressed" vs. "stressing") If so, you may be able to just include the shared component of the term with an asterisk to indicate term expansion (e.g., "stress*")
- Are any of these terms likely to identify datasets that are not related to your topic? If so, you might want to rule them out with NOT, e.g., "stress* NOT distress")
- If you have a term that is a phrase that includes more than one word, you may need to add quotations around that phrase if the exact phrase is desired/necessary ("chronic social defeat stress"). Or, if you need two or more words in your phrase to be present, but the exact order/spacing is not important, you can use AND (e.g., "chronic AND social AND stress").
- Independent search terms can be combined into a longer list using OR (e.g., "stress* OR advers* ")

Defining search syntax is also a critical step, and feedback is important. For the 2022 cohort of the Brain Data Alchemy Project, these terms were reviewed by the mentor.

Dataset Identification: Coding Steps

4 Dataset Identification: Coding Steps. This coding is all performed within the R environment using Rstudio.

Note

Example code for these steps can be found here: https://github.com/hagenaue/BrainDataAlchemy/blob/main/MetaAnalysis_GemmaDEResults%20/2022_2023_LPS_Code_forGemmaSearch.R

4.1 Install the R package Devtools using: *install.packages("devtools")*

Note

- This protocol was designed using the version of Devtools (v.2.4.3) available on 06/01/2022. It is likely to work using other versions.

Software

R Package devtools

NAME

Wickham, Hester, Chan, and Bryan

DEVELOPER

http://cran.r-project.org/web/packages/devtools/index.html

SOURCE LINK

4.2 Install the gemma.R package.

Software

Gemma.R

NAME

Javier Castillo-Arnemann, Jordan Sicherman, Ogan Mancarci, Guillaume Poirier-Morency

DEVELOPER

http://github.com/PavlidisLab/gemma.R

SOURCE LINK

Command

Example R Code: Install gemma.R

```
if(!requireNamespace("devtools", quietly=T)){
  install.packages("devtools")
}

devtools:: install_github("PavlidisLab/gemma.R", force=T)
```

Note

- This protocol was designed using the version of gemma.R (v.0.99.30) available on 06/01/2022. *It is unlikely to work using later versions.*

4.3 Use the function *search_datasets()* to locate transcriptional profiling datasets within the Gemma database that contain your prespecified keywords.

Command

Example R code for identifying datasets with lipopolysaccharide and hippocampus

```
lps_hipp1 <- gemma.R :: search_datasets("lps hippocamp*",
taxon="rat", limit=100)

lps_hipp2 <- gemma.R :: search_datasets("lps hippocamp*",
taxon="mouse", limit=100)

lps_hippv2_1 <- gemma.R :: search_datasets("lipopolysaccharide
hippocamp*", taxon= "rat", limit= 100)

lps_hippv3_1 <- gemma.R :: search_datasets("lipopolysaccharide
hippocamp*", taxon= "rat", limit= 100, offset=100)

lps_hippv2_2 <- gemma.R :: search_datasets("lipopolysaccharide
hippocamp*", taxon= "mouse", limit= 100)
```

Note

- The first argument in the *search_datasets()* function is your search term syntax. The full syntax for the search terms will need to be placed in quotations as if it were a single string value.
- The argument "taxon" allows you to filter by species.
- Gemma.R can't extract more than 100 records at a time (the "limit")
- If there are more than 100 results, a second search may be necessary using an offset of 100 (to get records #100-200), and so on.

4.4 Compile the metadata records from these identified datasets into a single data frame.

Command

Example R code: Combining search results into a single data frame

```
combined<-rbind.data.frame(lps_hippv2_1, lps_hippv2_2, lps_hipp1,
lps_hipp2)
```

4.5 Eliminate any duplicated records in the data frame.

Command

Example R code: Removing duplicated metadata records

```
unique_combined<- unique(combined)
```

Initial Dataset Filtering Using MetaData

5 The metadata records should be filtered using pre-specified inclusion and exclusion criteria.

5.1 Export the data frame containing the remaining metadata records from R as a .csv file. This data frame of metadata can then be more easily explored and annotated in a spreadsheet software program like Microsoft Excel or Google Sheets.

Command

Example R code: Writing out metadata records as a .csv file

```
write.csv(unique_combined, "Metadata_forDatasets_toReview.csv")
```

5.2 The titles, abstracts, and metadata for the datasets should then be scanned to determine which datasets should be excluded based on our pre-specified criteria.

Note

Pre-specified exclusion criteria:
1) Brain tissue was not collected.
2) The experimental manipulation was unrelated to the meta-analysis topic.
3) The brain tissue used in the experiment was not from a bulk dissection, and instead represented a particular cell type or small subregion.
4) The experiment targeted a particular subpopulation of transcripts (e.g., Chip-Seq, miRNA, TRAP-Seq, etc.).
5) The experiment used subjects from a developmental stage other than the timeframe of interest.
6) The metadata record on Gemma has critical issues, including duplication/overlap with another record or critical missing information (minimal methodological information due to no associated publication and a minimalist metadata record).

5.3 This is a critical step. Following this initial filtering by the individual experimenter, all inclusion/exclusion choices should be reviewed and approved by a second researcher (such as a research mentor).

Specification of Tissue of Interest (If Not Pre-Specified)

6 If a particular tissue or brain region of interest (ROI) had not been pre-specified in the search criteria, the research question should now be narrowed to a particular tissue or brain ROI based on dataset availability, and the dataset records filtered accordingly.

Note

At this point, since the titles and abstracts for the datasets have already been viewed, it is important to run decisions about tissue/ROI definition past a second researcher who is less invested in the project (e.g., a research mentor) to guard against bias. These decisions may include whether to include datasets from samples that only focus on a subregion of the larger ROI, or whether to include tissue cultures as well as fresh-collected tissue.

Secondary Dataset Filtering Using Detailed Methodological Review

7 The remaining dataset metadata records should now be subjected to a detailed review of accompanying experimental methods and available file formats.

Note

For this level of filtering, the publications associated with the datasets need to be referenced in addition to the metadata available on Gemma.

Only the methodological information in the publication and Gemma record should be used to determine study/dataset inclusion/exclusion, because we want our inclusion/exclusion criteria to be independent of the results reported in the original publication as much as possible.

Ideally, this would mean that only the methods section of the publication (or supplementary methods) would be read in depth. However, sometimes it is necessary to reference other sections of the paper in order to interpret or locate methodological information. For example, it may be necessary to read the introduction to understand the terminology and abbreviations used in the paper. Likewise, sometimes the experimental design will be overviewed in a figure, or final quality control decisions (and final sample sizes) discussed at the beginning of the results section or in the figure legends.

7.1 When reviewing the methodological information in the publications and metadata for the datasets, studies should be excluded using pre-specified inclusion/exclusion criteria.

Note

Pre-specified inclusion/exclusion criteria:
1) The study was retracted.
2) The experimental design was confounded or lacked a proper control group for the experimental manipulation of interest.
3) The subjects or protocol used in the study were dramatically different from all other included datasets.

7.2 Depending on the research topic, other inclusion/exclusion criteria may also be applied. Ideally, these criteria should be pre-specified as much as possible. If not pre-specified, this deviation from the planned protocol should be reported.

7.3 Following this secondary filtering by the individual experimenter, the experimental design for the remaining datasets and inclusion/exclusion choices should be reviewed and approved by a second researcher (such as a research mentor).

Note

Since this is a critical step, feedback is important. For the 2022 cohort of the Brain Data Alchemy Project, these inclusion/exclusion decisions were reviewed by the mentor and whole cohort.

7.4 When reviewing the available file formats, studies should be excluded using pre-specified criteria.

Note

Pre-specified exclusion criteria:
1) Differential expression results are not available in Gemma for the experimental manipulation of
interest.
2) Differential expression results are not available in Gemma for the particular tissue or brain ROI.

Post-hoc Filtering: Removing Problematic Externally-Derived Datasets

8 After we ran all of the meta-analyses for the Summer 2022 cohort of the Brain Data Alchemy Project, we realized that datasets that were marked “external” in the Gemma database had sometimes been imported into Gemma in a format that was incompatible with Gemma’s analysis pipeline.

Note

The analysis pipeline in Gemma typically begins with raw data extracted from public databases (e.g., Gene Expression Omnibus (GEO), **https://www.ncbi.nlm.nih.gov/geo/**, (Lim et al., 2021)). After we ran all of the meta-analyses for the Summer 2022 cohort of the Brain Data Alchemy Project, we realized that datasets that were marked “external” in the Gemma database had been subjected to a slightly different analysis pipeline. These datasets lacked available raw data, and were therefore imported into Gemma from external databases as the probe- or gene-level summarized expression data for each subject as generated by the original dataset contributors.

This imported external data sometimes appeared to be in formats that were incompatible with Gemma’s analysis pipeline. This seemed to be particularly true for datasets generated with Agilent transcriptional profiling platforms, which, for proprietary reasons, always lacked raw expression data in the GEO database, and appeared to have summarized gene expression data per sample that was generated by an unstandardized analysis pipeline (*i.e.,* GEO records included a variety of normalization methods and formats). For these Agilent records, we regularly found that the original units in GEO were non-standard or uninterpretable (*e.g.,* just labeled “normalized”) or had something appearing to follow a standard Log2 expression unit format but then had units within the Gemma database that had clearly been log transformed a second time (*e.g.,* the range of Log2 expression units in Gemma was highly restricted, such as units ranging between 2-4).

CITATION

Lim N, Tesar S, Belmadani M, Poirier-Morency G, Mancarci BO, Sicherman J, Jacobson M, Leong J, Tan P, Pavlidis P (2021). Curation of over 10 000 transcriptomic studies to enable data reuse.. LINK
https://doi.org/10.1093/database/baab006

8.1 Due to these irregularities, all "external" datasets should be re-reviewed for evidence of irregular units at import or accidental double log2 transformation (as signaled by units with a highly restricted range - such as only ranging from 2-4). Any questionable externally-derived datasets should be excluded.

Note

For the 2022 Brain Data Alchemy cohort, this exclusion happened *post-hoc.* After discovering these issues related to data import into Gemma, all datasets that were derived from the Agilent microarray platform and any other questionable externally-derived datasets were excluded from the 2022 meta-analyses, and the meta-analyses were re-run.

8.2 Following all filtering, the full search strategy and inclusion/exclusion procedure should be documented using a PRISMA flow diagram.

Note

A template PRISMA flow diagram in an editable Google Slides format can be found here: https://docs.google.com/presentation/d/1Dzw8MQsgo__EkXoIFSFf0ilavvpxX3Zr/edit?usp=sharing&ouid=106595687423493776462&rtpof=true&sd=true

This template should be updated to match the search procedure and the inclusion/exclusion criteria used for your specific project.

8.3 Following all filtering, the essential characteristics of the final datasets that will be included in the meta-analysis should be summarized in a table that can be included with your results.

Note

A template for the table summarizing the characteristics of the included studies in an editable Google Sheets format can be found here: https://docs.google.com/spreadsheets/d/1KptuIeWQH7B6SDk5z7K-Dub2beMSnmSa/edit?usp=sharing&ouid=106595687423493776462&rtpof=true&sd=true

Result Extraction: Gemma's Differential Expression Results

9 Result Extraction: Gemma's Differential Expression Results. This coding is all performed within the R environment using Rstudio.

Note

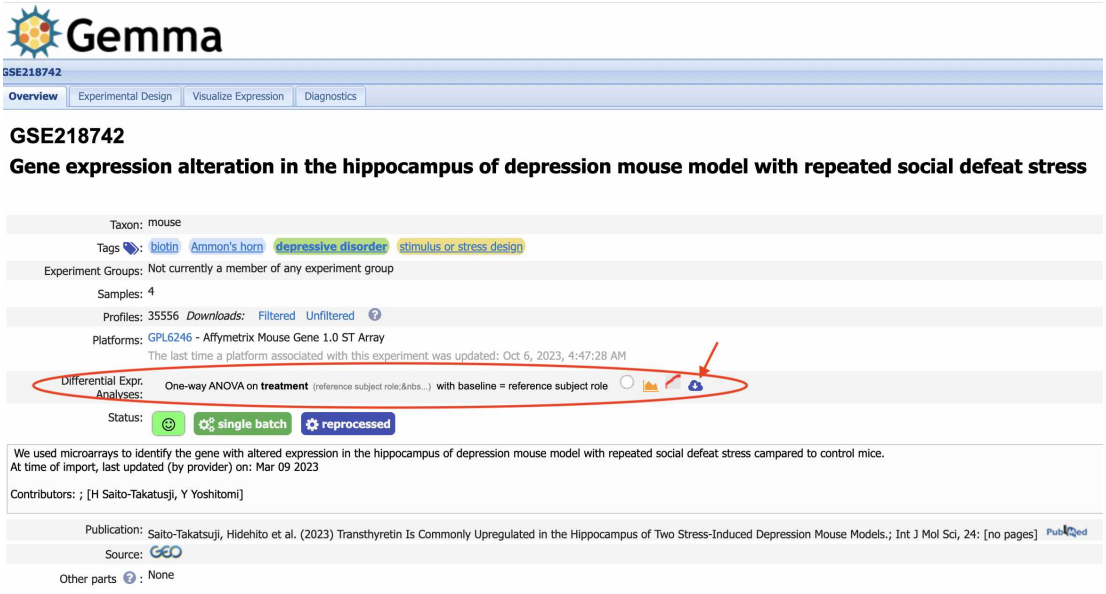
A detailed version of example code can be found here:
https://github.com/hagenaue/BrainDataAlchemy/blob/main/MetaAnalysis_GemmaDEResults%20/2022_Example_MetaAnalysis_GemmaOutput_DetailedReadingInAStudy.R

A version of the code that has been streamlined and functionalized can be found here:
https://github.com/hagenaue/BrainDataAlchemy/blob/main/MetaAnalysis_GemmaDEResults%20/2022_Example_MetaAnalysis_GemmaOutput_StreamlinedReadingInAStudy.R

Another example can be found here:
https://github.com/hagenaue/BrainDataAlchemy/blob/main/MetaAnalysis_GemmaDEResults%20/2022_2023_Antidepressant_Meta_analysis_Rcode_MH.R

Another example can be found here:
https://github.com/hagenaue/BrainDataAlchemy/blob/main/MetaAnalysis_GemmaDEResults%20/2022_LPS_readingandmeta_AS_2022.R

9.1 The Gemma differential expression output relevant to the meta-analysis (e.g., correct brain ROI) for each of the selected datasets should be downloaded and placed into its own folder.



Differential expression results can be downloaded from the Gemma record for a dataset by clicking on the blue cloud with the downward pointing arrow on it.

9.2 Install the *plyr* package.

Note

- This protocol was designed using the version of plyr (v.1.8.7) available on 06/01/2022. *It may not work using later versions.*

Software

R package plyr NAME

Wickham DEVELOPER

http://cran.r-project.org/web/packages/plyr/index.html SOURCE LINK

CITATION

Wickham H (2011). The Split-Apply-Combine Strategy for Data Analysis. Journal of Statistical Software.
LINK
10.18637/jss.v040.i01

9.3 The files in each dataset folder should then be read into R. After reading in the files from a particular dataset folder, all of the result files for that dataset should be joined together by Element_Name using the *join_all()* function from the plyr package.

Note

If this code doesn't work due to version updates, the command *merge_all()* can be used in a similar way.

Command

R Function: Reading the differential expression results for a dataset downloaded from Gemma into the R environment

```
ReadingInGemmaDE<-function(ResultSetFileNames){

#Reading in the analysis results file:
TempAnalysisResults<-read.delim("analysis.results.txt", sep="\t",
stringsAsFactors = FALSE, comment.char = "#")

#I placed these results in a list format:
TempResultsToJoin<-list(TempAnalysisResults)

for(i in c(1:length(ResultSetFileNames))){
#The result set files are the detailed differential expression
results for a particular variable:
TempResultsToJoin[[i]]<-read.delim(ResultSetFileNames[i], sep="\t",
stringsAsFactors = FALSE, comment.char = "#")
}

TempResultsJoined<<-join_all(TempResultsToJoin, by="Element_Name")
#Note: I've heard from other students that the join() and join_all()
functions in the plyr package can cause problems in newer versions of
R - you may need to replace this with merge and merge_all

#Saving the joined results:
write.csv(TempResultsJoined, "TempResultsJoined.csv")

rm(TempAnalysisResults, TempResultsToJoin)

print("Outputted object: TempResultsJoined")
}


#Notes about parameters for function: ReadingInGemmaDE()
#ResultSetFileNames should be a character vector containing the names
of the result files, e.g., "resultset_ID478782.data.txt"
```

Command

Example Usage: Reading the differential expression results for a dataset downloaded from Gemma into the R environment

```
#To start out with, I downloaded the Gemma differential expression
output for the studies of interest.
#I put each study is in its own folder.

#I set working directory to where I downloaded the Gemma differential
expression output for this particular study.


#Example Usage of the function for reading in Gemma differential
expression results:

ReadingInGemmaDE(ResultSetFileNames=c("resultset_ID478782.data.txt"))
#[1] "Outputted object: TempResultsJoined"


#Example Output:

str(TempResultsJoined)
# 'data.frame': 41264 obs. of  13 variables:
# $ Element_Name               : chr  "A_52_P266365" "A_52_P762911"
"A_52_P200244" "A_52_P1029978" ...
# $ Gene_Symbol                : chr  "Rbm27" "" "Sptlc2" "Lasp1" ...
# $ Gene_Name                  : chr  "RNA binding motif protein 27"
"" "serine palmitoyltransferase, long chain base subunit 2" "LIM and
SH3 protein 1" ...
# $ NCBI_ID                    : chr  "225432" "" "20773" "16796" ...
# $ FoldChange_13.d.5.d.of.rest: num  0.03501 -0.1365 -0.000978
0.02177 0.09924 ...
# $ Tstat_13.d.5.d.of.rest     : num  0.371 -0.5066 -0.0117 0.1818
0.2917 ...
# $ PValue_13.d.5.d.of.rest    : num  0.715 0.618 0.991 0.858 0.773
...
# $ FoldChange_8.d             : num  -0.169 -0.09793 0.06481
-0.000861 -0.1216 ...
# $ Tstat_8.d                  : num  -1.791 -0.3635 0.7737 -0.00719
-0.3576 ...
# $ PValue_8.d                 : num  0.0885 0.7201 0.4482 0.9943
0.7244 ...
# $ FoldChange_13.d            : num  -0.07923 -0.1743 -0.01604
```

```
# $ Foldchange_13.d                : num   0.07925  0.1743  0.01604
-0.00859 0.04578 ...
# $ Tstat_13.d                     : num  -0.8396 -0.647 -0.1915 -0.0717
0.1346 ...
# $ PValue_13.d                    : num  0.411 0.525 0.85 0.944 0.894 ...
```

9.4 If gene symbol annotation is missing in the result files, additional annotation information for the transcriptional profiling platform can be downloaded from Gene Expression Omnibus. This information will then need to be read into R, and joined to the differential expression result file.

Note

Gene Expression Omnibus transcriptional profiling platform information:
https://www.ncbi.nlm.nih.gov/geo/browse/?view=platforms

Example code:
https://github.com/hagenaue/BrainDataAlchemy/blob/main/MetaAnalysis_GemmaDEResults%20/2022_Example_FixingGemmaDEResultsWithNoAnnotation.R

9.5 Within the joined file containing the differential expression results for a dataset, rows that lack unambiguous gene symbol annotation (including “”, “null”, '\\|' in the gene symbol) need to be counted and excluded.

Command

R Function: Remove rows of data that have missing or unambiguous gene annotation

```
FilteringDEResults_GoodAnnotation<-function(TempResultsJoined){

print("# of rows with missing NCBI annotation:")
print(sum(TempResultsJoined$NCBI_ID==""|TempResultsJoined$NCBI_ID=="nu
ll"))

print("# of rows with missing Gene Symbol annotation:")
print(sum(TempResultsJoined$Gene_Symbol==""|TempResultsJoined$Gene_Sym
bol=="null"))

print("# of rows mapped to multiple NCBI_IDs:")
print(length(grep('\\|', TempResultsJoined$NCBI_ID)))

print("# of rows mapped to multiple Gene Symbols:")
print(length(grep('\\|', TempResultsJoined$Gene_Symbol)))

#I only want the subset of data which contains rows that do not
contain a Gene Symbol of ""
TempResultsJoined_NoNA<-
TempResultsJoined[(TempResultsJoined$Gene_Symbol==""|TempResultsJoined
$Gene_Symbol=="null")==FALSE,]

if(length(grep('\\|', TempResultsJoined_NoNA$Gene_Symbol))==0){
  TempResultsJoined_NoNA_NoMultimapped<<-TempResultsJoined_NoNA
}else{
#I only want rows annotated with a single Gene Symbol (no pipe):
TempResultsJoined_NoNA_NoMultimapped<<-TempResultsJoined_NoNA[-
(grep('\\|', TempResultsJoined_NoNA$Gene_Symbol)),]
}

print("# of rows with good annotation")
print(nrow(TempResultsJoined_NoNA_NoMultimapped))

#For record keeping (sometimes useful for troubleshooting later)
write.csv(TempResultsJoined_NoNA_NoMultimapped,
"TempResultsJoined_NoNA_NoMultimapped.csv")

rm(TempResultsJoined_NoNA, TempResultsJoined_NoNA_NoMultimapped)
```

```
print("Outputted object: TempResultsJoined_NoNA_NoMultimapped")
}
```

Command

Example Usage: Remove rows of data that have missing or unambiguous gene annotation

```
#Example usage of function:

FilteringDEResults_GoodAnnotation(TempResultsJoined)
# [1] "# of rows with missing NCBI annotation:"
# [1] 11443
# [1] "# of rows with missing Gene Symbol annotation:"
# [1] 11443
# [1] "# of rows mapped to multiple NCBI_IDs:"
# [1] 1838
# [1] "# of rows mapped to multiple Gene Symbols:"
# [1] 1838
# [1] "# of rows with good annotation"
# [1] 27983
# [1] "Outputted object: TempResultsJoined_NoNA_NoMultimapped"

#Example Output:

str(TempResultsJoined_NoNA_NoMultimapped)
# 'data.frame': 27983 obs. of  13 variables:
#   $ Element_Name               : chr  "A_52_P266365" "A_52_P200244"
"A_52_P1029978" "A_52_P674374" ...
# $ Gene_Symbol                : chr  "Rbm27" "Sptlc2" "Lasp1" "Rem1"
...
# $ Gene_Name                  : chr  "RNA binding motif protein 27"
"serine palmitoyltransferase, long chain base subunit 2" "LIM and SH3
protein 1" "rad and gem related GTP binding protein 1" ...
# $ NCBI_ID                    : chr  "225432" "20773" "16796"
"19700" ...
# $ FoldChange_13.d.5.d.of.rest: num  0.03501 -0.000978 0.02177
-0.000332 -0.161 ...
# $ Tstat_13.d.5.d.of.rest     : num  0.371 -0.01167 0.1818 -0.000529
-0.9336 ...
# $ PValue_13.d.5.d.of.rest    : num  0.715 0.991 0.858 1 0.362 ...
# $ FoldChange_8.d             : num  -0.169 0.06481 -0.000861 0.8238
0.03793 ...
# $ Tstat_8.d                  : num  -1.791 0.7737 -0.00719 1.313
0.22 ...
# $ PValue_8.d                 : num  0.0885 0.4482 0.9943 0.2041
0.8281
```

```
0.0281 ...
# $ FoldChange_13.d            : num  -0.07923 -0.01604 -0.00859
-0.00319 -0.2234 ...
# $ Tstat_13.d                 : num  -0.8396 -0.1915 -0.07173
-0.00509 -1.296 ...
# $ PValue_13.d                : num  0.411 0.85 0.944 0.996 0.21 ...
```

9.6 The specific statistical contrasts of interest (group comparisons) that are related to your research question should be identified within the differential expression results.

9.7 The reference group for the statistical comparison should be confirmed to fit expectations (e.g., represent the appropriate control or baseline group).

Note

If the expected reference and treatment groups are reversed, the effect sizes for the differential expression output for the contrast (Log2 Fold Change or Log2FC) can be multiplied by -1.

9.8 The standard error (SE) can be calculated using the Log2FC and T-statistic (Tstat) in the differential expression output for each row. *(This will be performed by the function in step 9.10)*

Note

Log2FC/Tstat=SE

9.9 The sampling variance (SV) for each Log2FC can then be calculated by squaring the standard error (SE^2). *(This will be performed by the function in step 9.10)*

Note

As discussed in the online materials for the *metafor* package: *https://www.metafor-project.org/doku.php/tips:input_to_rma_function*, Viechtbauer 2024.

9.10 If there is more than one row of results representing the same gene symbol, the Log2FC and SE's should be averaged for each gene symbol using the *tapply()* function.

Command

R Function: Calculating SV and Collapsing Differential Expression Results to One Result Per Gene

```
CollapsingDEResults_OneResultPerGene<-function(GSE_ID,
TempResultsJoined_NoNA_NoMultimapped, ComparisonsOfInterest,
NamesOfFoldChangeColumns, NamesOfTstatColumns){

print("Double check that the vectors containing the two fold change
and tstat column names contain the same order as the group
comparisons of interest - otherwise this function won't work
properly!  If the order matches, proceed:")

print("# of rows with unique NCBI IDs:")
print(length(unique(TempResultsJoined_NoNA_NoMultimapped$NCBI_ID)))

print("# of rows with unique Gene Symbols:")
print(length(unique(TempResultsJoined_NoNA_NoMultimapped$Gene_Symbol))
)

#We will need both the Log2FC and T-stats averaged:

TempResultsJoined_NoNA_NoMultimapped_FoldChange_Average<-list()

for(i in c(1:length(NamesOfFoldChangeColumns))){

TempResultsJoined_NoNA_NoMultimapped_FoldChange_Average[[i]]<-
tapply(NamesOfFoldChangeColumns[i][[1]],
TempResultsJoined_NoNA_NoMultimapped$Gene_Symbol, mean)

}

TempResultsJoined_NoNA_NoMultimapped_FoldChange_AveragedByGeneSymbol<-
do.call(cbind,
TempResultsJoined_NoNA_NoMultimapped_FoldChange_Average)

print("Dimensions of Fold Change matrix, averaged by gene symbol:")
print(dim(TempResultsJoined_NoNA_NoMultimapped_FoldChange_AveragedByGe
neSymbol))

colnames(TempResultsJoined_NoNA_NoMultimapped_FoldChange_AveragedByGen
eSymbol)<-ComparisonsOfInterest
```

```
write.csv(TempResultsJoined_NoNA_NoMultimapped_FoldChange_AveragedByGe
neSymbol,
"TempResultsJoined_NoNA_NoMultimapped_FoldChange_AveragedByGeneSymbol.
csv")


TempResultsJoined_NoNA_NoMultimapped_Tstat_Average<-list()

for(i in c(1:length(NamesOfFoldChangeColumns))){

  TempResultsJoined_NoNA_NoMultimapped_Tstat_Average[[i]]<-
tapply(NamesOfTstatColumns[i][[1]],
TempResultsJoined_NoNA_NoMultimapped$Gene_Symbol, mean)

}

TempResultsJoined_NoNA_NoMultimapped_Tstat_AveragedByGeneSymbol<-
do.call(cbind, TempResultsJoined_NoNA_NoMultimapped_Tstat_Average)

colnames(TempResultsJoined_NoNA_NoMultimapped_Tstat_AveragedByGeneSymb
ol)<-ComparisonsOfInterest

write.csv(TempResultsJoined_NoNA_NoMultimapped_Tstat_AveragedByGeneSym
bol,
"TempResultsJoined_NoNA_NoMultimapped_Tstat_AveragedByGeneSymbol.csv")



TempResultsJoined_NoNA_NoMultimapped_SE<-list()

for(i in c(1:length(NamesOfFoldChangeColumns))){
  TempResultsJoined_NoNA_NoMultimapped_SE[[i]]<-
NamesOfFoldChangeColumns[i][[1]]/NamesOfTstatColumns[i][[1]]
}


TempResultsJoined_NoNA_NoMultimapped_SE_Average<-list()

for(i in c(1:length(NamesOfFoldChangeColumns))){

  TempResultsJoined_NoNA_NoMultimapped_SE_Average[[i]]<-
tapply(TempResultsJoined_NoNA_NoMultimapped_SE[[i]],
TempResultsJoined_NoNA_NoMultimapped$Gene_Symbol, mean)

}
```

```
TempResultsJoined_NoNA_NoMultimapped_SE_AveragedByGeneSymbol<-
do.call(cbind, TempResultsJoined_NoNA_NoMultimapped_SE_Average)

colnames(TempResultsJoined_NoNA_NoMultimapped_SE_AveragedByGeneSymbol)
<-ComparisonsOfInterest

write.csv(TempResultsJoined_NoNA_NoMultimapped_SE_AveragedByGeneSymbol
, "TempResultsJoined_NoNA_NoMultimapped_SE_AveragedByGeneSymbol.csv")

#For running our meta-analysis, we are actually going to need the
sampling variance instead of the standard error
#The sampling variance is just the standard error squared.

TempResultsJoined_NoNA_NoMultimapped_SV<-
(TempResultsJoined_NoNA_NoMultimapped_SE_AveragedByGeneSymbol)^2

write.csv(TempResultsJoined_NoNA_NoMultimapped_SV,
"TempResultsJoined_NoNA_NoMultimapped_SV.csv")

TempMasterResults<-
list(Log2FC=TempResultsJoined_NoNA_NoMultimapped_FoldChange_AveragedBy
GeneSymbol,
Tstat=TempResultsJoined_NoNA_NoMultimapped_Tstat_AveragedByGeneSymbol,
 SE=TempResultsJoined_NoNA_NoMultimapped_SE_AveragedByGeneSymbol,
SV=TempResultsJoined_NoNA_NoMultimapped_SV)

assign(paste("DEResults", GSE_ID, sep="_"), TempMasterResults, envir
= as.environment(1))

print(paste("Output: Named DEResults", GSE_ID, sep="_"))

rm(TempMasterResults, TempResultsJoined_NoNA_NoMultimapped_SV,
TempResultsJoined_NoNA_NoMultimapped_SE,
TempResultsJoined_NoNA_NoMultimapped_FoldChange_AveragedByGeneSymbol,
TempResultsJoined_NoNA_NoMultimapped_FoldChange_Average,
TempResultsJoined_NoNA_NoMultimapped_Tstat_AveragedByGeneSymbol,
TempResultsJoined_NoNA_NoMultimapped_Tstat_Average)

}


#Notes about parameters for function
CollapsingDEResults_OneResultPerGene()
#GSE_ID is a string indicating the name of the Gemma dataset
#TempResultsJoined_NoNA_NoMultimapped is the data frame outputted by
our previous function
```

```
#ComparisonsOfInterest is a character vector containing a list of
group comparisons of interest within this dataset. Important: These
group comparisons should be listed in exactly the same order as the
order that you provide the column names for their associated Fold
Change and Tstat output.
#NamesOfFoldChangeColumns is a list containing the columns of
TempResultsJoined_NoNA_NoMultimapped containing the FoldChange
results for your comparisons of interes, in the same order as the
ComparisonsOfInterest vector
#NamesOfTstatColumns is a list containing the columns of
TempResultsJoined_NoNA_NoMultimapped containing the Tstat results for
your comparisons of interest, in the same order as the
ComparisonsOfInterest vector
```

Command

Example Usage: Calculating SV and collapsing differential expression results to one result per gene

```
#Example function usage:

CollapsingDEResults_OneResultPerGene(GSE_ID="GSE59070",
TempResultsJoined_NoNA_NoMultimapped,
ComparisonsOfInterest=c("Stress8days_vs_Acute",
"Stress13days_vs_Acute"),
NamesOfFoldChangeColumns=list(TempResultsJoined_NoNA_NoMultimapped$Fol
dChange_8.d, TempResultsJoined_NoNA_NoMultimapped$FoldChange_13.d),
NamesOfTstatColumns=list(TempResultsJoined_NoNA_NoMultimapped$Tstat_8.
d, TempResultsJoined_NoNA_NoMultimapped$Tstat_13.d))
# [1] "Double check that the vectors containing the two fold change
and tstat column names contain the same order as the group
comparisons of interest - otherwise this function won't work
properly!  If the order matches, proceed:"
# [1] "# of rows with unique NCBI IDs:"
# [1] 18503
# [1] "# of rows with unique Gene Symbols:"
# [1] 18503
# [1] "Dimensions of Fold Change matrix, averaged by gene symbol:"
# [1] 18503     2
# [1] "Output: Named DEResults_GSE59070"


#Example Output:

str(DEResults_GSE59070)
# List of 4
# $ Log2FC: num [1:18503, 1:2] -0.00551 0.16795 -0.09224 -0.05196
-0.02993 ...
# ..- attr(*, "dimnames")=List of 2
# .. ..$ : chr [1:18503] "0610005C13Rik" "0610009B22Rik"
"0610009L18Rik" "0610010F05Rik" ...
# .. ..$ : chr [1:2] "Stress8days_vs_Acute" "Stress13days_vs_Acute"
# $ Tstat : num [1:18503, 1:2] -0.0319 1.6845 -1.311 -0.3 -0.3194 ...
# ..- attr(*, "dimnames")=List of 2
# .. ..$ : chr [1:18503] "0610005C13Rik" "0610009B22Rik"
"0610009L18Rik" "0610010F05Rik" ...
# .. ..$ : chr [1:2] "Stress8days_vs_Acute" "Stress13days_vs_Acute"
# $ SE    : num [1:18503, 1:2] 0.1725 0.0997 0.0704 0.1732 0.0937
```

```
# $ SE    : num [1:18503, 1:2] 0.1725 0.0997 0.0704 0.1732 0.0957 ...
# ..- attr(*, "dimnames")=List of 2
# .. ..$ : chr [1:18503] "0610005C13Rik" "0610009B22Rik"
"0610009L18Rik" "0610010F05Rik" ...
# .. ..$ : chr [1:2] "Stress8days_vs_Acute" "Stress13days_vs_Acute"
# $ SV    : num [1:18503, 1:2] 0.02977 0.00994 0.00495 0.03 0.00878
...
# ..- attr(*, "dimnames")=List of 2
# .. ..$ : chr [1:18503] "0610005C13Rik" "0610009B22Rik"
"0610009L18Rik" "0610010F05Rik" ...
# .. ..$ : chr [1:2] "Stress8days_vs_Acute" "Stress13days_vs_Acute"
```

9.11 The results (Log2FC, SV) from the different statistical contrasts of interest and datasets should then be compiled into a single data frame. To do this, results can be aligned (joined) using official gene symbol.

Note

We are taking advantage of the fact that 75% of official gene symbols are identical in rats and mice (Jackson Labs Mouse ortholog database: https://www.informatics.jax.org/downloads/reports/HOM_AllOrganism.rpt).

Command

R Function: Combining results from different studies into a single data frame

```
AligningDatasets<-function(ListOfDEResults){

  MetaAnalysis_FoldChange_Dfs<-list()

  for(i in c(1:length(ListOfDEResults))){
    MetaAnalysis_FoldChange_Dfs[[i]]<-
data.frame(x=row.names(ListOfDEResults[[i]][[1]]),ListOfDEResults[[i]]
[[1]], stringsAsFactors=FALSE)
  }

  print("MetaAnalysis_FoldChange_Dfs:")
  print(str(MetaAnalysis_FoldChange_Dfs))

  MetaAnalysis_FoldChanges<<-join_all(MetaAnalysis_FoldChange_Dfs,
by="x", type="full")
  #This function could be join_all (if there are more than 2
datasets) or merge/merge_all (if the plyr package isn't working)

  print("MetaAnalysis_FoldChanges:")
  print(str(MetaAnalysis_FoldChanges))

  MetaAnalysis_SV_Dfs<-list()

  for(i in c(1:length(ListOfDEResults))){
    MetaAnalysis_SV_Dfs[[i]]<-
data.frame(x=row.names(ListOfDEResults[[i]][[4]]),ListOfDEResults[[i]]
[[4]], stringsAsFactors=FALSE)
  }

  print("MetaAnalysis_SV_Dfs:")
  print(str(MetaAnalysis_SV_Dfs))

  MetaAnalysis_SV<<-join_all(MetaAnalysis_SV_Dfs, by="x", type="full")
  #This function could be join_all (if there are more than 2
datasets) or merge/merge_all (if the plyr package isn't working)

  print("MetaAnalysis_SV:")
  print(str(MetaAnalysis_SV))

  rm(MetaAnalysis_SV_Dfs, MetaAnalysis_FoldChange_Dfs)
```

```
}

#Notes on using the function:

#This function combines together the relevant results from different
studies into a single data-frame for the effect sizes (Log2FC) and a
single data.frame for the sampling variances.

#The Log2FC values are the first element in the differential
expression result object for each study

#The gene symbols are the row.names - 75% of gene symbols for rats
and mice are the same, so for simplicity sake, instead of using a
gene orthology database to inform equivalency, we're just going to
align the datasets by gene symbol.
```

Command

Example Usage: Combining results from different studies into a single data frame

```
#Example Usage;

ListOfDEResults<-list(DEResults_GSE27532, DEResults_GSE56028,
DEResults_GSE63469)

#The Log2FC values are the first element in the differential
expression result object for each study
#The gene symbols are the row.names
#Make sure that the datasets don't have *comparisons with the same
name*, or those comparisons disappear during the join. :(
#That can be avoided by always including the dataset name (GSE#) in
the column names for the comparisons of interest

AligningDatasets(ListOfDEResults)
# [1] "MetaAnalysis_FoldChange_Dfs:"
# List of 3
# $ :'data.frame':      16378 obs. of  2 variables:
#   ..$ x                           : chr [1:16378] "0610009B22Rik"
"0610010K14Rik" "0610012G03Rik" "0610040J01Rik" ...
# ..$ Desipramine_7mgPerKg_vs_Ctrl: num [1:16378] 0.03548 -0.00465
0.1887 -0.09332 -0.00764 ...
# $ :'data.frame':      15724 obs. of  5 variables:
#   ..$ x                            : chr [1:15724] "A1bg" "A1cf"
"A2m" "A3galt2" ...
# ..$ Tianeptine_10mgPerKg_vs_Ctrl : num [1:15724] 0.1322 0.0188
-0.1332 0.0397 0.0291 ...
# ..$ Agomelatine_40mgPerKg_vs_Ctrl: num [1:15724] 0.121 -0.0832
-0.088 -0.052 -0.0643 ...
# ..$ Fluoxetine_10mgPerKg_vs_Ctrl : num [1:15724] -0.01397 0.1506
-0.04246 0.00956 -0.06863 ...
# ..$ Imipramine_10mgPerKg_vs_Ctrl : num [1:15724] 0.0515 0.0602
-0.2672 0.0789 -0.0039 ...
# $ :'data.frame':      18532 obs. of  3 variables:
#   ..$ x                           : chr [1:18532] "0610005C13Rik"
"0610009B22Rik" "0610009L18Rik" "0610010K14Rik" ...
# ..$ Venlafaxine_30kgDkg_vs_Ctrl : num [1:18532] 0.2384 0.0695
0.0605 0.0921 0.0248 ...
# ..$ Venlafaxine_100kgDkg_vs_Ctrl: num [1:18532] 0.0502 0.0408
0.1478 0.1317 0.0407
```

```
0.1478 0.1317 0.0407 ...
# NULL
# [1] "MetaAnalysis_FoldChanges:"
# 'data.frame': 22772 obs. of  8 variables:
#   $ x                           : chr  "0610009B22Rik"
"0610010K14Rik" "0610012G03Rik" "0610040J01Rik" ...
# $ Desipramine_7mgPerKg_vs_Ctrl : num  0.03548 -0.00465 0.1887
-0.09332 -0.00764 ...
# $ Tianeptine_10mgPerKg_vs_Ctrl : num  NA NA NA NA NA NA NA NA NA NA
...
# $ Agomelatine_40mgPerKg_vs_Ctrl: num  NA NA NA NA NA NA NA NA NA NA
...
# $ Fluoxetine_10mgPerKg_vs_Ctrl : num  NA NA NA NA NA NA NA NA NA NA
...
# $ Imipramine_10mgPerKg_vs_Ctrl : num  NA NA NA NA NA NA NA NA NA NA
...
# $ Venlafaxine_30kgDkg_vs_Ctrl  : num  0.0695 0.0921 0.0248 -0.0467
0.1309 ...
# $ Venlafaxine_100kgDkg_vs_Ctrl : num  0.0408 0.1317 0.0407 0.1255
0.1296 ...
# NULL
# [1] "MetaAnalysis_SV_Dfs:"
# List of 3
# $ :'data.frame':      16378 obs. of  2 variables:
#   ..$ x                           : chr [1:16378] "0610009B22Rik"
"0610010K14Rik" "0610012G03Rik" "0610040J01Rik" ...
# ..$ Desipramine_7mgPerKg_vs_Ctrl: num [1:16378] 0.00518 0.01285
0.01194 0.00369 0.00201 ...
# $ :'data.frame':      15724 obs. of  5 variables:
#   ..$ x                           : chr [1:15724] "A1bg" "A1cf"
"A2m" "A3galt2" ...
# ..$ Tianeptine_10mgPerKg_vs_Ctrl : num [1:15724] 0.01222 0.01078
0.02587 0.00668 0.00968 ...
# ..$ Agomelatine_40mgPerKg_vs_Ctrl: num [1:15724] 0.01221 0.01077
0.02588 0.00668 0.00968 ...
# ..$ Fluoxetine_10mgPerKg_vs_Ctrl : num [1:15724] 0.01222 0.01077
0.02587 0.00668 0.00968 ...
# ..$ Imipramine_10mgPerKg_vs_Ctrl : num [1:15724] 0.01222 0.01077
0.02588 0.00668 0.00968 ...
# $ :'data.frame':      18532 obs. of  3 variables:
#   ..$ x                           : chr [1:18532] "0610005C13Rik"
"0610009B22Rik" "0610009L18Rik" "0610010K14Rik" ...
# ..$ Venlafaxine_30kgDkg_vs_Ctrl : num [1:18532] 0.00133 0.00117
0.00114 0.00315 0.00244 ...
# ..$ Venlafaxine_100kgDkg_vs_Ctrl: num [1:18532] 0.00133 0.00117
0.00114 0.00315 0.00244 ...
# NULL
```

```
# [1] "MetaAnalysis_SV:"
# 'data.frame': 22772 obs. of  8 variables:
```

Meta-Analysis of Differential Expression Results

10 Meta-Analysis of Differential Expression Results: This coding is all performed within the R environment using Rstudio.

Note

Example code can be found here:
https://github.com/hagenaue/BrainDataAlchemy/blob/main/MetaAnalysis_GemmaDEResults%20/2022_2023_Antidepressant_Meta_analysis_Rcode_MH.R

Another example can be found here:
https://github.com/hagenaue/BrainDataAlchemy/blob/main/MetaAnalysis_GemmaDEResults%20/2022_LPS_readingandmeta_AS_2022.R

10.1 Because we are considering differential expression results that are derived from a variety of different transcriptional profiling platforms with varying transcript/probe representation and sensitivity, each dataset will include differential expression results for a slightly different set of genes.

Decide on a minimum number of differential expression results that need to be present to run a meta-analysis for each gene (i.e., a minimum number of Log2FC values that are not NAs).

Note

This decision should be guided by considerations related to sample size and the minimum number of datasets necessary to represent the diversity of subjects and methods available in your datasets. If you only have a small number of datasets included in your meta-analysis (<5 datasets), you may want to require that a gene be represented in all of them to be included in the meta-analysis.

10.2 Install the R package metafor

Software

R package metafor

NAME

Wolfgang Viechtbauer

DEVELOPER

http://CRAN.R-project.org/package=metafor

SOURCE LINK

Note

This protocol was designed using the version of metafor (v.3.4.0) available on 06/01/2022. It is likely to work using other versions of the package.

CITATION

Viechtbauer, W (2010). Conducting meta-analyses in R with the metafor package. Journal of Statistical Software.

LINK

https://doi.org/10.18637/jss.v036.i03

10.3 To run a meta-analysis for every gene meeting our criteria for a minimum number of differential expression results, we use a *for loop* to run the same series of calculations for each row of our Log2FC data frame and accompanying SV data frame.

Note

(This process is performed by the function in 10.5)

10.4 For each row (gene), we first determine whether the gene meets the criteria of having our pre-specified minimum number of differential expression results (Log2FC values). This is done using a conditional *if-else* statement.

Note

(This process is performed by the function in 10.5)

10.5 If the row (gene) meets our criteria, we use the function *rma()* from the metafor package to fit a random effects meta-analysis model to the Log2FC values (y_i) and accompanying SV (v_i).

Note

Due to the relatively small number of differential expression results that are available for most topics, we have chosen to use the simplest model possible for the meta-analysis (an intercept-only model) as our main outcome.

The downside to this approach is an inadequate accounting for the covariance of results derived from the same dataset (*e.g.,* a dataset with multiple drug treatments compared to a control treatment).

This approach also means that we do not attempt to quantify potentially impactful sources of
heterogeneity within the subject characteristics or methodology of the studies. If there is a larger number of datasets (e.g., 4-5 datasets representing each of the potential sources of heterogeneity), it may be worth exploring models including these variables as secondary outcomes.

Command

R Function: Running a meta-analysis on the data frame of differential expression results

```
RunBasicMetaAnalysis<-function(NumberOfComparisons, CutOffForNAs,
MetaAnalysis_FoldChanges, MetaAnalysis_SV){

  MetaAnalysis_FoldChanges_NAsPerRow<-apply(MetaAnalysis_FoldChanges,
1, function(y) sum(is.na(y)))

  print("Table of # of NAs per Row (Gene):")
  print(table(MetaAnalysis_FoldChanges_NAsPerRow))

  MetaAnalysis_FoldChanges_ForMeta<<-
MetaAnalysis_FoldChanges[MetaAnalysis_FoldChanges_NAsPerRow<CutOffForN
As,]
  MetaAnalysis_SV_ForMeta<<-
MetaAnalysis_SV[MetaAnalysis_FoldChanges_NAsPerRow<CutOffForNAs,]

  print("MetaAnalysis_FoldChanges_ForMeta:")
  print(str(MetaAnalysis_FoldChanges_ForMeta))

  #I'm going to make an empty matrix to store the results of my meta-
analysis:
  metaOutput<-matrix(NA, length(MetaAnalysis_FoldChanges_ForMeta$x),
6)

  #And then run a loop that run's a meta-analysis on the differential
expression results (columns 2-10) for each gene (row):
  for(i in c(1:length(MetaAnalysis_FoldChanges_ForMeta$x))){

    effect<-as.numeric(MetaAnalysis_FoldChanges_ForMeta[i,-1])
    var<-as.numeric(MetaAnalysis_SV_ForMeta[i,-1])

    #I added a function tryCatch that double-checks that the meta-
analysis function (rma) doesn't produce errors (which breaks the
loop):
    skip_to_next <- FALSE
    tryCatch(TempMeta<-rma(effect, var), error = function(e)
{skip_to_next <<- TRUE})

    if(skip_to_next){}else{
      TempMeta<-rma(effect, var)
```

```
      TempMeta<- lma(effect, var)
      metaOutput[i, 1]<-TempMeta$b #gives estimate Log2FC
      metaOutput[i, 2]<-TempMeta$se #gives standard error
      metaOutput[i, 3]<-TempMeta$pval #gives pval
      metaOutput[i, 4]<-TempMeta$ci.lb #gives confidence interval
lower bound
      metaOutput[i, 5]<-TempMeta$ci.ub #gives confidence interval
upper bound
      metaOutput[i, 6]<-NumberOfComparisons-sum(is.na(effect))#Number
of comparisons with data
      rm(TempMeta)
    }
    rm(effect, var)
  }

  colnames(metaOutput)<-c("Log2FC_estimate", "SE", "pval", "CI_lb",
"CI_ub", "Number_Of_Comparisons")
  row.names(metaOutput)<-MetaAnalysis_FoldChanges_ForMeta$x

  metaOutput<<-metaOutput
  return(metaOutput)

  print("metaOutput:")
  print(str(metaOutput))

  print("Top of metaOutput:")
  print(head(metaOutput))

  print("Bottom of metaOutput")
  print(tail(metaOutput))

}
```

Command

Example Usage: Running a meta-analysis on the data frame of differential expression results

```
#We can only run a meta-analysis if there are differential expression results from more than one comparison.
#Since I know that the differential expression results from the same study (dataset) are artificially correlated, I would actually prefer that there are results from more than one dataset.

#How many genes satisfy this criteria?

#This code calculates the number of NAs in each row:
MetaAnalysis_FoldChanges_NAsPerRow<-apply(MetaAnalysis_FoldChanges, 1, function(y) sum(is.na(y)))

#I'm going to make a histogram of the results because I'm curious to see how they are distributed
hist(MetaAnalysis_FoldChanges_NAsPerRow)

#Or, since there are a limited number of integer answers (0-11), I could make a table of the results:
table(MetaAnalysis_FoldChanges_NAsPerRow)
#MetaAnalysis_FoldChanges_NAsPerRow
MetaAnalysis_FoldChanges_NAsPerRow
# 0     1     2     3     4     5     6
# 11562  1356   496  2310  2886  2728  1434

#Let's try running a meta-analysis using genes that were found in at least 7 sets of differential expression results
#Since there are 7 sets of differential expression results, that means that the genes that we are including need to have 0 or fewer NAs in their results
#I set this conservatively, because there are so few studies in this meta-analysis.
#1 NAs is too many

#Example Usage:
NumberOfComparisons=7
CutOffForNAs=1
#I want at least 7 comparisons
#1 NA is too many
```

```
metaOutput<-RunBasicMetaAnalysis(NumberOfComparisons, CutOffForNAs,
MetaAnalysis_FoldChanges, MetaAnalysis_SV)
#Note: this function can take a while to run, especially if you have
a lot of data
#Plug in your computer, take a break, grab some coffee...

# [1] "Table of # of NAs per Row (Gene):"
# MetaAnalysis_FoldChanges_NAsPerRow
# 0     1     2     3     4     5     6
# 11562  1356   496  2310  2886  2728  1434
# [1] "MetaAnalysis_FoldChanges_ForMeta:"
# 'data.frame': 11562 obs. of  8 variables:
#   $ x                           : chr  "A4galt" "Aaas" "Aacs"
"Aadac" ...
# $ Desipramine_7mgPerKg_vs_Ctrl : num  0.00882 0.03177 -0.02216
-0.06205 0.08798 ...
# $ Tianeptine_10mgPerKg_vs_Ctrl : num  0.0291 -0.0445 -0.0257
-0.0961 0.2043 ...
# $ Agomelatine_40mgPerKg_vs_Ctrl: num  -0.0643 -0.2562 0.0177
-0.1124 -0.0331 ...
# $ Fluoxetine_10mgPerKg_vs_Ctrl : num  -0.06863 -0.00253 -0.01577
-0.1021 0.1074 ...
# $ Imipramine_10mgPerKg_vs_Ctrl : num  -0.0039 -0.0874 -0.1441
-0.1393 0.00518 ...
# $ Venlafaxine_30kgDkg_vs_Ctrl  : num  0.2085 0.1111 -0.0447 -0.1497
-0.0331 ...
# $ Venlafaxine_100kgDkg_vs_Ctrl : num  0.2465 0.1304 -0.1482 0.0866
-0.1226 ...

#Example Output:

str(metaOutput)
# num [1:11562, 1:6] 0.06022 -0.01543 -0.06567 -0.07498 -0.00186 ...
# - attr(*, "dimnames")=List of 2
# ..$ : chr [1:11562] "A4galt" "Aaas" "Aacs" "Aadac" ...
# ..$ : chr [1:6] "Log2FC_estimate" "SE" "pval" "CI_lb" ...

head(metaOutput)
# Log2FC_estimate         SE       pval       CI_lb       CI_ub
Number_Of_Comparisons
# A4galt     0.060215268 0.04928678 0.22180877 -0.03638505
0.15681559                     7
# Aaas      -0.015428086 0.04761888 0.74594492 -0.10875938
0.07790321                     7
# Aacs      -0.065673071 0.02722289 0.01584698 -0.11902896
-0.01231718                     7
```

```
# Aadac     -0.074981686 0.03104035 0.01570846 -0.13581966
-0.01414371                  7
```

10.6 If a random effects model for the Log2FC values for a gene (row) produces a convergence error (i.e., there is not a stable model fit), that row of results will be recorded as NA.

Applying a False Discovery Rate (FDR) Correction to the Meta-Analysis Results

11 False Discovery Rate Correction: Since we have run meta-analyses for thousands of genes, we correct the p-values for false discovery rate (FDR) using the the Benjamini-Hochberg method. This coding is all performed within the R environment using Rstudio.

Note

To properly interpret the p-values produced by our meta-analyses for each gene, we need to take into account the fact that we have run thousands of statistical tests (i.e., one statistical test for each gene for thousands of genes). Therefore, we are likely to get a large number of results that are "significant" using a traditional p-value threshold (alpha=0.05) just due to random chance. This is called a multiple comparisons correction or p-value adjustment. We use a type of correction called False Discovery Rate (FDR) or q-value. It is also sometimes called a "Benjamini–Hochberg" adjustment after its originators.

11.1 Install the R package *multtest*

Software

R package multtest NAME

Katherine S. Pollard, Houston N. Gilbert, Yongchao Ge, Sandra Taylor, Sandrine Dudoit DEVELOPER

http://bioconductor.org/packages/release/bioc/html/multtest.html SOURCE LINK

Note

This protocol was designed using the version of multtest (v.2.8.0) available on 06/01/2022. It is likely to work using other versions of the package.

CITATION

Pollard K.S., Dudoit S., van der Laan M.J. (2005). Multiple Testing Procedures: R multtest Package and Applications to Genomics, in Bioinformatics and Computational Biology Solutions Using R and Bioconductor. Springer: Bioinformatics and Computational Biology Solutions Using R and Bioconductor .

LINK

10.1007/0-387-29362-0

11.2 We correct the p-values for false discovery rate (FDR) using the the Benjamini-Hochberg method as applied by the *mt.rawp2adjp()* function within the multtest package.

Command

R Function: Applying an FDR correction to meta-analysis p-values

```
FalseDiscoveryCorrection<-function(metaOutput){

  tempPvalAdjMeta<-mt.rawp2adjp(metaOutput[,3], proc=c("BH"))

  metaPvalAdj<-tempPvalAdjMeta$adjp[order(tempPvalAdjMeta$index),]

  metaOutputFDR<-cbind(metaOutput, metaPvalAdj[,2])

  colnames(metaOutputFDR)[7]<-"FDR"

  metaOutputFDR<<-metaOutputFDR

  print("metaOutputFDR:")
  print(str(metaOutputFDR))

  write.csv(metaOutputFDR, "metaOutputFDR.csv")

  #a version of the output in order by p-value:
  metaOutputFDR_OrderbyPval<<-metaOutputFDR[order(metaOutputFDR[,3]),]

  #Let's write out a version of the output in order by p-value:
  write.csv(metaOutputFDR_OrderbyPval,
"metaOutputFDR_orderedByPval_wHDRFData.csv")

  print("Do we have any genes that are statistically significant
following false discovery rate correction?")
  print(sum(metaOutputFDR[,7]<0.10, na.rm=TRUE))

  print("What are the top results?")
  print(head(metaOutputFDR[order(metaOutputFDR[,3]),]))

  rm(tempPvalAdjMeta, metaPvalAdj)

}
```

Command

Example usage: Applying an FDR correction to meta-analysis p-values

```
#Example usage:

FalseDiscoveryCorrection(metaOutput)
# [1] "metaOutputFDR:"
# num [1:11562, 1:7] 0.06022 -0.01543 -0.06567 -0.07498 -0.00186 ...
# - attr(*, "dimnames")=List of 2
# ..$ : chr [1:11562] "A4galt" "Aaas" "Aacs" "Aadac" ...
# ..$ : chr [1:7] "Log2FC_estimate" "SE" "pval" "CI_lb" ...
# NULL
# [1] "Do we have any genes that are statistically significant
following false discovery rate correction?"
# [1] 797
# [1] "What are the top results?"
# Log2FC_estimate         SE         pval      CI_lb       CI_ub
Number_Of_Comparisons          FDR
# Parm1       -0.1798498 0.02389052 5.149049e-14 -0.2266743
-0.13302524                     7 5.953331e-10
# Lpl          0.2138084 0.03015666 1.341870e-12  0.1547024
0.27291434                     7 7.757348e-09
# Gabrr2       0.1965106 0.02859386 6.309657e-12  0.1404677
0.25255358                     7 2.431742e-08
# Pla2g5       0.1593702 0.02375177 1.948623e-11  0.1128176
0.20592280                     7 5.632494e-08
# Akap8l      -0.1174562 0.01848935 2.116479e-10 -0.1536947
-0.08121775                     7 4.894145e-07
# Dusp1       -0.4415588 0.07101670 5.045755e-10 -0.5807489
-0.30236858                     7 9.723171e-07
```

11.3 We will consider results to be statistically significant if they survive a threshold of 5% false discovery (FDR<0.05). If our meta-analyses do not produce any results surviving this threshold, we may consider results at a weaker threshold (FDR<0.10) with the understanding that a higher percent of these results are likely to be false discovery (10%).

Exploring Meta-Analysis Results

12 To explore our meta-analysis results, we first rank the results by p-value and examine the results surviving our false discovery rate correction (FDR<0.05). We will refer to these results as our "top genes".

12.1 Within the results surviving false discovery rate correction, count how many show increased expression ("upregulation") in response to our variable of interest and how many show decreased expression ("down-regulation").

Note

Within the meta-analysis results, a negative "estimate" (estimated Log2FC) means that the gene typically shows lower expression in response to our variable of interest, whereas a positive estimate (estimated Log2FC) means that a gene typically shows higher expression in response to our variable of interest.

12.2 To explore the meta-analysis results for any particular gene in more detail, use the function *forest.rma()* from the *metafor* package to make a forest plot illustrating the effect sizes (Log2FC) and confidence intervals for each study.

Command

Function: Making forest plots to illustrate differential expression results across studies for a gene

```
MakeForestPlots<-function(GeneSymbol){

  pdf(paste("ForestPlot_", GeneSymbol, ".pdf", sep=""), height=5,
width=8)

  effect<-
as.numeric(MetaAnalysis_FoldChanges_ForMeta[MetaAnalysis_FoldChanges_F
orMeta$x==GeneSymbol,-1])
  var<-
as.numeric(MetaAnalysis_SV_ForMeta[MetaAnalysis_FoldChanges_ForMeta$x=
=GeneSymbol,-1])

  forest.rma(rma(effect,
var),slab=colnames(MetaAnalysis_FoldChanges_ForMeta)[-1],  xlim=c(-3,
3))

  mtext(paste(GeneSymbol), line=-1.5, cex=2)
  dev.off()
}
```

Command

Example Usage: Making forest plots to illustrate differential expression results across studies for a gene

```
#Example usage (for gene Parm1):

MakeForestPlots("Parm1")
```

12.3 For the sake of easily including results from the project in future posters, presentations and publications, we have a template for quickly summarizing meta-analysis results.

Note

A Google Docs template for quickly illustrating and summarizing the meta-analysis results can be found here:
https://docs.google.com/presentation/d/1fAirnPu9zdu4uHRIuESeK5RTDxwFqiD8/edit?usp=sharing&ouid=106595687423493776462&rtpof=true&sd=true

12.4 Quick ways to learn about the functions associated with our top genes:

Basic functional summary:
GeneCards: **https://www.genecards.org/**
Rat Genome Database: **https://rgd.mcw.edu/**

Cell types that express the gene in the brain:
DropViz: **http://dropviz.org/**
MouseBrain.Org: **http://www.mousebrain.org/**

Regional distribution of the expression for the gene in the brain:
https://mouse.brain-map.org/search/index

CITATION

Saunders A, Macosko EZ, Wysoker A, Goldman M, Krienen FM, de Rivera H, Bien E, Baum M, Bortolin L, Wang S, Goeva A, Nemesh J, Kamitaki N, Brumbaugh S, Kulp D, McCarroll SA (2018). Molecular Diversity and Specializations among the Cells of the Adult Mouse Brain..
LINK
https://doi.org/10.1016/j.cell.2018.07.028

CITATION

Shimoyama M, De Pons J, Hayman GT, Laulederkind SJ, Liu W, Nigam R, Petri V, Smith JR, Tutaj M, Wang SJ, Worthey E, Dwinell M, Jacob H (2015). The Rat Genome Database 2015: genomic, phenotypic and environmental variations and disease..
LINK
https://doi.org/10.1093/nar/gku1026

CITATION

Zeisel A, Hochgerner H, Lönnerberg P, Johnsson A, Memic F, van der Zwan J, Häring M, Braun E, Borm LE, La Manno G, Codeluppi S, Furlan A, Lee K, Skene N, Harris KD, Hjerling-Leffler J, Arenas E, Ernfors P, Marklund U, Linnarsson S (2018). Molecular Architecture of the Mouse Nervous System..
LINK
https://doi.org/10.1016/j.cell.2018.06.021

CITATION

Lein ES, Hawrylycz MJ, Ao N, Ayres M, Bensinger A, Bernard A, Boe AF, Boguski MS, Brockway KS, Byrnes EJ, Chen L, Chen L, Chen TM, Chin MC, Chong J, Crook BE, Czaplinska A, Dang CN, Datta S, Dee NR, Desaki AL, Desta T, Diep E, Dolbeare TA, Donelan MJ, Dong HW, Dougherty JG, Duncan BJ, Ebbert AJ, Eichele G, Estin LK, Faber C, Facer BA, Fields R, Fischer SR, Fliss TP, Frensley C, Gates SN, Glattfelder KJ, Halverson KR, Hart MR, Hohmann JG, Howell MP, Jeung DP, Johnson RA, Karr PT, Kawal R, Kidney JM, Knapik RH, Kuan CL, Lake JH, Laramee AR, Larsen KD, Lau C, Lemon TA, Liang AJ, Liu Y, Luong LT, Michaels J, Morgan JJ, Morgan RJ, Mortrud MT, Mosqueda NF, Ng LL, Ng R, Orta GJ, Overly CC, Pak TH, Parry SE, Pathak SD, Pearson OC, Puchalski RB, Riley ZL, Rockett HR, Rowland SA, Royall JJ, Ruiz MJ, Sarno NR, Schaffnit K, Shapovalova NV, Sivisay T, Slaughterbeck CR, Smith SC, Smith KA, Smith BI, Sodt AJ, Stewart NN, Stumpf KR, Sunkin SM, Sutram M, Tam A, Teemer CD, Thaller C, Thompson CL, Varnam LR, Visel A, Whitlock RM, Wohnoutka PE, Wolkey CK, Wong VY, Wood M, Yaylaoglu MB, Young RC, Youngstrom BL, Yuan XF, Zhang B, Zwingman TA, Jones AR (2007). Genome-wide atlas of gene expression in the adult mouse brain..

LINK

https://doi.org/

12.5 *Optional:* To learn about the functions associated with our entire collection of differential expression results, we can use a functional ontology analysis. There are many different varieties of functional ontology analysis tools; I find that these two are an easy place to start out:

GOrilla:
https://cbl-gorilla.cs.technion.ac.il/

EnrichR:
https://maayanlab.cloud/Enrichr/

Note

When running functional ontology analyses on differential expression results from brain tissue, it is particularly important to either use a tool that either considers the full ranked list of results (i.e., all significant (FDR<0.05) and non-significant results) or that compares the significant results (FDR<0.05) to a "background" of all genes contained within the results. Do not use the full genome as the "background" for your functional comparison. Only a subset of the genome is expressed in brain tissue - therefore, by definition, the genes included in your results are already enriched for brain functions, regardless of their relationship to your variable of interest.

CITATION

Eden E, Navon R, Steinfeld I, Lipson D, Yakhini Z (2009). GOrilla: a tool for discovery and visualization of enriched GO terms in ranked gene lists..
LINK
https://doi.org/10.1186/1471-2105-10-48

CITATION

Kuleshov MV, Jones MR, Rouillard AD, Fernandez NF, Duan Q, Wang Z, Koplev S, Jenkins SL, Jagodnik KM, Lachmann A, McDermott MG, Monteiro CD, Gundersen GW, Ma'ayan A (2016). Enrichr: a comprehensive gene set enrichment analysis web server 2016 update..
LINK
https://doi.org/10.1093/nar/gkw377

Wrapping Up the Project

13 In order to wrap up the meta-analysis project in a manner that can be easily followed up on later by yourself or others, it is important to preserve input, output, code, and workspaces.

13.1 First, make sure that you save both your code and workspace in R.

Note

The names for the code files will end with the file extension ".R".
The names for the workspace files will end with the file extension ".Rdata".

13.2 To preserve and share code, it is useful to upload the final R code and workspaces to a Github repository.

Note

Here are instructions for initiating a Github repository: https://docs.github.com/en/repositories/creating-and-managing-repositories/quickstart-for-repositories

(if you haven't already been using Github for version control throughout the duration of the project)

13.3 To make your analyses reproducible, make sure that you save the folders containing the Gemma differential expression results for each dataset that you used as input for your meta-analysis.

Note

Gemma updates gene annotation each time a new genome assembly is released, therefore a scientist following the same extraction and analysis procedure a year from now might get different results if they do not have your exact input.

13.4 In addition to your PRISMA search diagram, table of final datasets, and results summary, make sure to save the full meta-analysis results for all genes. The full results can be provided as a supplementary table in a publication.

13.5 A README describing the organization for all of the meta-analysis files can help you (or others) navigate the files later.

Protocol references

Eden, E., Navon, R., Steinfeld, I., Lipson, D., Yakhini, Z., 2009. GOrilla: a tool for discovery and visualization of enriched GO terms in ranked gene lists. BMC Bioinformatics 10, 48. https://doi.org/10.1186/1471-2105-10-48

Kuleshov, M.V., Jones, M.R., Rouillard, A.D., Fernandez, N.F., Duan, Q., Wang, Z., Koplev, S., Jenkins, S.L., Jagodnik, K.M., Lachmann, A., McDermott, M.G., Monteiro, C.D., Gundersen, G.W., Ma'ayan, A., 2016. Enrichr: a comprehensive gene set enrichment analysis web server 2016 update. Nucleic Acids Res 44, W90-97. https://doi.org/10.1093/nar/gkw377

Lein, E.S., Hawrylycz, M.J., Ao, N., Ayres, M., Bensinger, A., Bernard, A., Boe, A.F., Boguski, M.S., Brockway, K.S., Byrnes, E.J., Chen, Lin, Chen, Li, Chen, T.-M., Chin, M.C., Chong, J., Crook, B.E., Czaplinska, A., Dang, C.N., Datta, S., Dee, N.R., Desaki, A.L., Desta, T., Diep, E., Dolbeare, T.A., Donelan, M.J., Dong, H.-W., Dougherty, J.G., Duncan, B.J., Ebbert, A.J., Eichele, G., Estin, L.K., Faber, C., Facer, B.A., Fields, R., Fischer, S.R., Fliss, T.P., Frensley, C., Gates, S.N., Glattfelder, K.J., Halverson, K.R., Hart, M.R., Hohmann, J.G., Howell, M.P., Jeung, D.P., Johnson, R.A., Karr, P.T., Kawal, R., Kidney, J.M., Knapik, R.H., Kuan, C.L., Lake, J.H., Laramee, A.R., Larsen, K.D., Lau, C., Lemon, T.A., Liang, A.J., Liu, Y., Luong, L.T., Michaels, J., Morgan, J.J., Morgan, R.J., Mortrud, M.T., Mosqueda, N.F., Ng, L.L., Ng, R., Orta, G.J., Overly, C.C., Pak, T.H., Parry, S.E., Pathak, S.D., Pearson, O.C., Puchalski, R.B., Riley, Z.L., Rockett, H.R., Rowland, S.A., Royall, J.J., Ruiz, M.J., Sarno, N.R., Schaffnit, K., Shapovalova, N.V., Sivisay, T., Slaughterbeck, C.R., Smith, S.C., Smith, K.A., Smith, B.I., Sodt, A.J., Stewart, N.N., Stumpf, K.-R., Sunkin, S.M., Sutram, M., Tam, A., Teemer, C.D., Thaller, C., Thompson, C.L., Varnam, L.R., Visel, A., Whitlock, R.M., Wohnoutka, P.E., Wolkey, C.K., Wong, V.Y., Wood, M., Yaylaoglu, M.B., Young, R.C., Youngstrom, B.L., Yuan, X.F., Zhang, B., Zwingman, T.A., Jones, A.R., 2007. Genome-wide atlas of gene expression in the adult mouse brain. Nature 445, 168–176. https://doi.org/10.1038/nature05453

Liberati, A., Altman, D.G., Tetzlaff, J., Mulrow, C., Gøtzsche, P.C., Ioannidis, J.P.A., Clarke, M., Devereaux, P.J., Kleijnen, J., Moher, D., 2009. The PRISMA Statement for Reporting Systematic Reviews and Meta-Analyses of Studies That Evaluate Health Care Interventions: Explanation and Elaboration. PLOS Medicine 6, e1000100. https://doi.org/10.1371/journal.pmed.1000100

Lim, N., Tesar, S., Belmadani, M., Poirier-Morency, G., Mancarci, B.O., Sicherman, J., Jacobson, M., Leong, J., Tan, P., Pavlidis, P., 2021. Curation of over 10 000 transcriptomic studies to enable data reuse. Database (Oxford) 2021, baab006. https://doi.org/10.1093/database/baab006

Moher, D., Liberati, A., Tetzlaff, J., Altman, D.G., 2010. Preferred reporting items for systematic reviews and meta-analyses: The PRISMA statement. International Journal of Surgery 8, 336–341. https://doi.org/10.1016/j.ijsu.2010.02.007

Pollard, K.S., Dudoit, S., Laan, M.J. van der, 2005. Multiple Testing Procedures: the multtest Package and Applications to Genomics, in: Bioinformatics and Computational Biology Solutions Using R and Bioconductor, Statistics for Biology and Health. Springer, New York, NY, pp. 249–271. https://doi.org/10.1007/0-387-29362-0_15

Ritchie, M.E., Phipson, B., Wu, D., Hu, Y., Law, C.W., Shi, W., Smyth, G.K., 2015. limma powers differential expression analyses for RNA-sequencing and microarray studies. Nucleic Acids Res. 43, e47. https://doi.org/10.1093/nar/gkv007

Saunders, A., Macosko, E.Z., Wysoker, A., Goldman, M., Krienen, F.M., de Rivera, H., Bien, E., Baum, M., Bortolin, L., Wang, S., Goeva, A., Nemesh, J., Kamitaki, N., Brumbaugh, S., Kulp, D., McCarroll, S.A., 2018. Molecular Diversity and Specializations among the Cells of the Adult Mouse Brain. Cell 174, 1015-1030.e16. https://doi.org/10.1016/j.cell.2018.07.028

Shimoyama, M., De Pons, J., Hayman, G.T., Laulederkind, S.J.F., Liu, W., Nigam, R., Petri, V., Smith, J.R., Tutaj, M., Wang, S.-J., Worthey, E., Dwinell, M., Jacob, H., 2015. The Rat Genome Database 2015: genomic, phenotypic and environmental variations and disease. Nucleic Acids Res. 43, D743-750.
https://doi.org/10.1093/nar/gku1026

Viechtbauer, W., 2010. Conducting Meta-Analyses in R with The metafor Package. Journal of Statistical Software 36.
https://doi.org/10.18637/jss.v036.i03

Wickham, H., 2023. plyr: Tools for Splitting, Applying and Combining Data.

Wickham, H., Hester, J., Chang, W., Bryan, J., RStudio, 2022. devtools: Tools to Make Developing R Packages Easier.

Zeisel, A., Hochgerner, H., Lönnerberg, P., Johnsson, A., Memic, F., van der Zwan, J., Häring, M., Braun, E., Borm, L.E., La Manno, G., Codeluppi, S., Furlan, A., Lee, K., Skene, N., Harris, K.D., Hjerling-Leffler, J., Arenas, E., Ernfors, P., Marklund, U., Linnarsson, S., 2018. Molecular Architecture of the Mouse Nervous System. Cell 174, 999-1014.e22.
https://doi.org/10.1016/j.cell.2018.06.021

Zoubarev, A., Hamer, K.M., Keshav, K.D., McCarthy, E.L., Santos, J.R.C., Van Rossum, T., McDonald, C., Hall, A., Wan, X., Lim, R., Gillis, J., Pavlidis, P., 2012. Gemma: a resource for the reuse, sharing and meta-analysis of expression profiling data. Bioinformatics 28, 2272–2273. https://doi.org/10.1093/bioinformatics/bts430

Citations

Step 10.2

Viechtbauer, W. Conducting meta-analyses in R with the metafor package
https://doi.org/10.18637/jss.v036.i03

Step 11.1

Pollard K.S., Dudoit S., van der Laan M.J.. Multiple Testing Procedures: R multtest Package and Applications to Genomics, in Bioinformatics and Computational Biology Solutions Using R and Bioconductor
10.1007/0-387-29362-0

Step 12.4

Saunders A, Macosko EZ, Wysoker A, Goldman M, Krienen FM, de Rivera H, Bien E, Baum M, Bortolin L, Wang S, Goeva A, Nemesh J, Kamitaki N, Brumbaugh S, Kulp D, McCarroll SA. Molecular Diversity and Specializations among the Cells of the Adult Mouse Brain.
https://doi.org/10.1016/j.cell.2018.07.028

Step 12.4

Shimoyama M, De Pons J, Hayman GT, Laulederkind SJ, Liu W, Nigam R, Petri V, Smith JR, Tutaj M, Wang SJ, Worthey E, Dwinell M, Jacob H. The Rat Genome Database 2015: genomic, phenotypic and environmental variations and disease.
https://doi.org/10.1093/nar/gku1026

Step 12.4

Lein ES, Hawrylycz MJ, Ao N, Ayres M, Bensinger A, Bernard A, Boe AF, Boguski MS, Brockway KS, Byrnes EJ, Chen L, Chen L, Chen TM, Chin MC, Chong J, Crook BE, Czaplinska A, Dang CN, Datta S, Dee NR, Desaki AL, Desta T, Diep E, Dolbeare TA, Donelan MJ, Dong HW, Dougherty JG, Duncan BJ, Ebbert AJ, Eichele G, Estin LK, Faber C, Facer BA, Fields R, Fischer SR, Fliss TP, Frensley C, Gates SN, Glattfelder KJ, Halverson KR, Hart MR, Hohmann JG, Howell MP, Jeung DP, Johnson RA, Karr PT, Kawal R, Kidney JM, Knapik RH, Kuan CL, Lake JH, Laramee AR, Larsen KD, Lau C, Lemon TA, Liang AJ, Liu Y, Luong LT, Michaels J, Morgan JJ, Morgan RJ, Mortrud MT, Mosqueda NF, Ng LL, Ng R, Orta GJ, Overly CC, Pak TH, Parry SE, Pathak SD, Pearson OC, Puchalski RB, Riley ZL, Rockett HR, Rowland SA, Royall JJ, Ruiz MJ, Sarno NR, Schaffnit K, Shapovalova NV, Sivisay T, Slaughterbeck CR, Smith SC, Smith KA, Smith BI, Sodt AJ, Stewart NN, Stumpf KR, Sunkin SM, Sutram M, Tam A, Teemer CD, Thaller C, Thompson CL, Varnam LR, Visel A, Whitlock RM, Wohnoutka PE, Wolkey CK, Wong VY, Wood M, Yaylaoglu MB, Young RC, Youngstrom BL, Yuan XF, Zhang B, Zwingman TA, Jones AR. Genome-wide atlas of gene expression in the adult mouse brain.
https://doi.org/

Step 12.4

Zeisel A, Hochgerner H, Lönnerberg P, Johnsson A, Memic F, van der Zwan J, Häring M, Braun E, Borm LE, La Manno G, Codeluppi S, Furlan A, Lee K, Skene N, Harris KD, Hjerling-Leffler J, Arenas E, Ernfors P, Marklund U, Linnarsson S. Molecular Architecture of the Mouse Nervous System.
https://doi.org/10.1016/j.cell.2018.06.021

Step 12.5

Eden E, Navon R, Steinfeld I, Lipson D, Yakhini Z. GOrilla: a tool for discovery and visualization of enriched GO terms in ranked gene lists.
https://doi.org/10.1186/1471-2105-10-48

Step 12.5

Kuleshov MV, Jones MR, Rouillard AD, Fernandez NF, Duan Q, Wang Z, Koplev S, Jenkins SL, Jagodnik KM, Lachmann A, McDermott MG, Monteiro CD, Gundersen GW, Ma'ayan A. Enrichr: a comprehensive gene set enrichment analysis web server 2016 update.
https://doi.org/10.1093/nar/gkw377

Step 3

Moher D, Liberati A, Tetzlaff J, Altman DG, PRISMA Group. Preferred reporting items for systematic reviews and meta-analyses: the PRISMA statement.
https://doi.org/10.1136/bmj.b2535

Step 3

Liberati A, Altman DG, Tetzlaff J, Mulrow C, Gøtzsche PC, Ioannidis JP, Clarke M, Devereaux PJ, Kleijnen J, Moher D. The PRISMA statement for reporting systematic reviews and meta-analyses of studies that evaluate healthcare interventions: explanation and elaboration.
https://doi.org/10.1136/bmj.b2700

Step 3.2

Zoubarev A, Hamer KM, Keshav KD, McCarthy EL, Santos JR, Van Rossum T, McDonald C, Hall A, Wan X, Lim R, Gillis J, Pavlidis P. Gemma: a resource for the reuse, sharing and meta-analysis of expression profiling data.
https://doi.org/10.1093/bioinformatics/bts430

Step 3.2

Lim N, Tesar S, Belmadani M, Poirier-Morency G, Mancarci BO, Sicherman J, Jacobson M, Leong J, Tan P, Pavlidis P. Curation of over 10 000 transcriptomic studies to enable data reuse.
https://doi.org/10.1093/database/baab006

Step 8

Lim N, Tesar S, Belmadani M, Poirier-Morency G, Mancarci BO, Sicherman J, Jacobson M, Leong J, Tan P, Pavlidis P. Curation of over 10 000 transcriptomic studies to enable data reuse.
https://doi.org/10.1093/database/baab006

Step 9.2

Wickham H. The Split-Apply-Combine Strategy for Data Analysis

10.18637/jss.v040.i01