DECNEO Documentation

Release 1.0.0

S. Domanskyi, A. Hakansson, M. Meng, J. S. Graff Zivin, C. Pierma

Jan 05, 2021

CONTENTS

In silico detection of transcriptional regulation genes from single cell transcriptomics data.

CHAPTER

ONE

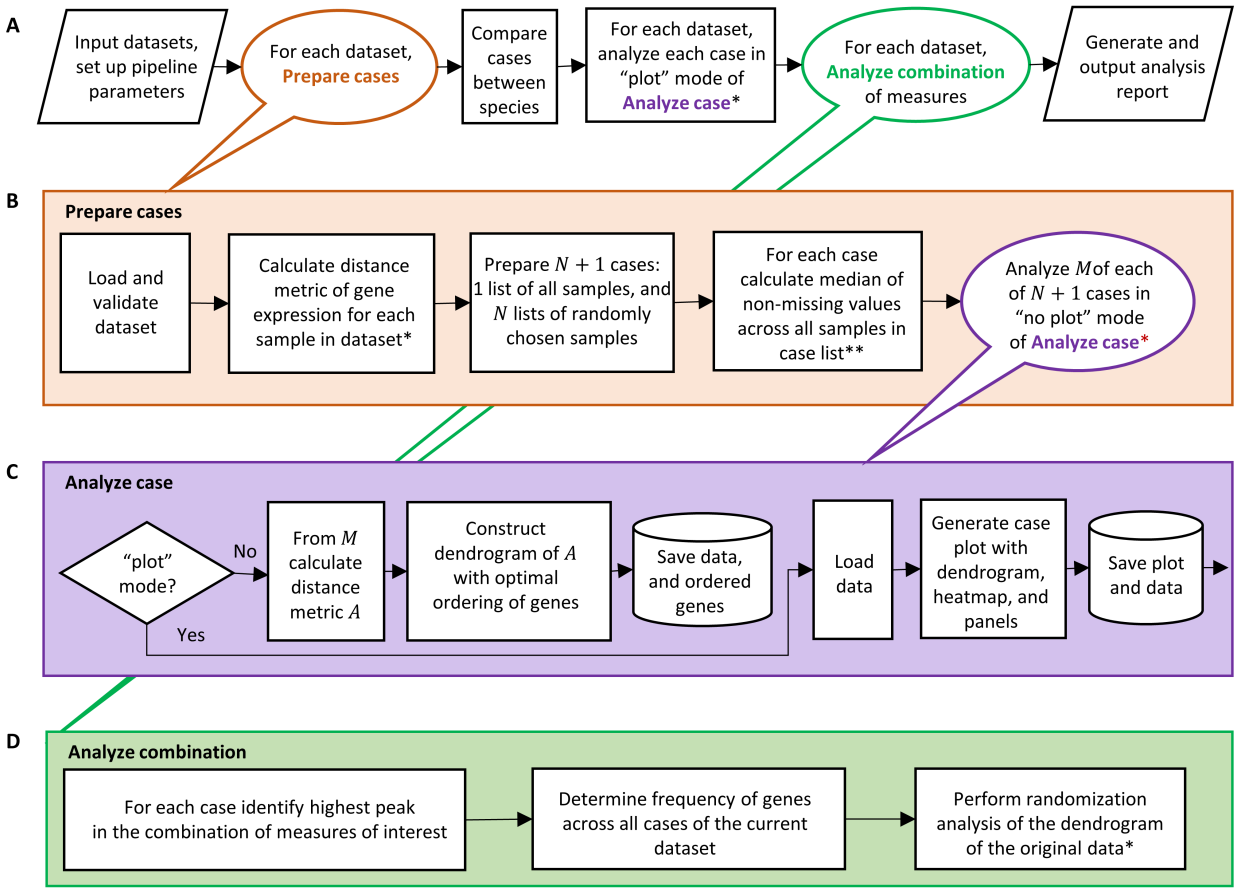
OVERVIEW

In silico detection of transcriptional regulation genes from single cell transcriptomics.

1.1 Description of the package functionality

Single cell expression datasets in the correct *Input data format* are required as input. Normalized single cell transcriptomics datasets are then processed and principle steps: preparing cases, analyzing cases, and analyzing combinations are performed. An analysis report along with *Output data* directories and files are generated at the conclusion of the pipeline. For a more in-depth summary, reference the figure below.

Summary of Analysis Pipeline



*Parallel CPU processing
**Parallel CPU processing for large datasets requires large RAM

1.2 Versions change log

1.0.1 Setting up the documentation at ReadTheDocs

1.0.0 Initial release

CHAPTER

TWO

GETTING STARTED

These instructions will get you a copy of the project up and running on your machine for data analysis, development or testing purposes.

2.1 Installation

Install of the latest release of `decneo`:

```
$ pip install decneo
```

For detailed instructions and other ways to install `decneo` as well as list of optional packages and instructions on how to install them see **Prerequisites** section at https://github.com/sdomanskyi/decneo

2.2 Loading the package

In your script import the package:

```
from decneo.analysisPipeline import Analysis
```

Create an instance of `class decneo`. Here, for simplicity, we use Default parameter values:

```
an = Analysis()
```

CHAPTER

THREE

CORE CLASS API

Core module

Description of the package functionality

Module holding class that implements the analysis pipeline

analysisPipeline.**process**(*df1other, df2main, df2other, dir1, dir2, genesOfInterest=None, knownRegulators=None, nCPUs=4, panels=['fraction', 'binomial', 'top50', 'markers', 'combo3avgs', 'combo4avgs'], parallelBootstrap=False, exprCutoff1=0.05, exprCutoff2=0.05, perEachOtherCase=True, doScramble=False, part1=True, part2=True, part3=True, **kwargs*)

Main workflow programmed in two scenaria depending on parameter "perEachOtherCase".

Parameters:

df1main: pandas.DataFrame Expression data of main group of cells of the first species

df1other: pandas.DataFrame Expression data of other cells of the first species

df2main: pandas.DataFrame Expression data of main group of cells of the second species

df2other: pandas.DataFrame Expression data of other cells of the second species

dir1: str Path to the first species working directory

dir2: str Path to the second species working directory

genesOfInterest: list, Default None Particular genes to analyze, e.g. receptors

knownRegulators: list, Default None Known marker genes

nCPUs: int, Default 1 Number of CPUs to use for multiprocessing, recommended 10-20

panels: list, Default None Particular measurements to include in the analysis

parallelBootstrap: boolean, Default False Whether to generate bootstrap experiments in parallel mode

exprCutoff1: float, Default 0.05 Per-batch expression cutoff for the first dataset

exprCutoff2: float, Default 0.05 Per-batch expression cutoff for the second dataset

perEachOtherCase: boolean, Default True Scenario of comparison

Any other parameters that class "Analysis" can take

Returns:

Analysis First class Analysis instance

Analysis Second class Analysis instance

class Analysis(*workingDir=''*, *otherCaseDir=''*, *genesOfInterest=None*, *knownRegulators=None*, *nCPUs=1*, *panels=None*, *nBootstrap=100*, *majorMetric='correlation'*, *perEachOtherCase=False*, *metricsFile='metricsFile.h5'*, *seed=None*, *PCNpath='data/'*, *minBatches=5*, *pseudoBatches=10*, *dendrogramMetric='euclidean'*, *dendrogramLinkageMethod='ward'*, *methodForDEG='ttest'*)

Bases: `object`

Class of analysis and visualization functions for DECNEO

Parameters:

workingDir: str, Default ‘’ Directory to retrieve and save files and results to

otherCaseDir: str, Default ‘’ Directory holding comparison (other species) data

genesOfInterest: list, Default None Particular genes to analyze, e.g. receptors

knownRegulators: list, Default None Known marker genes

nCPUs: int, Default 1 Number of CPUs to use for multiprocessing, recommended 10-20

panels: list, Default None Particular measurements to include in the analysis

nBootstrap: int, Default 100 Number of bootstrap experiments to perform

majorMetric: str, Default ‘correlation’ Metric name (e.g. ‘correlation’, ‘cosine’, ‘euclidean’, ‘spearman’)

methodForDEG: str, Default ‘ttest’ Possible options: {‘ttest’, ‘mannwhitneyu’}

perEachOtherCase: boolean, Default False Whether to perform comparisons of bootstrap experiments with other bootstrap experiments or with a single case

metricsFile: str, ‘metricsFile.h5’ Name of file where gene expression distance data is saved for specified metric

seed: int, None Used to set randomness deterministic

PCNpath: str, Default ‘data/’ Path to PCN file

Methods:

analyzeAllPeaksOfCombinationVariant(variant)	Find all peaks and their frequency from the bootstrap experiments
analyzeBootstrapExperiments()	Analyze all bootstrap experiments
analyzeCase(df_expr[, ...])	Analyze, calculate, and generate plots for individual experiment
analyzeCombinationVariant(variant)	Analyze a combination of measures (same as in panels)
bootstrapMaxpeakPlot(variant)	Bootstrap max-peak plot
compareTwoCases(saveDir1, saveDir2[, name1, ...])	Compare gene measurements between two cases for each bootstrap experiment
generateAnalysisReport()	Generate analysis report.
prepareBootstrapExperiments([allDataToo, ...])	Prepare bootstrap experiments data and calculating gene statistics for each experiment
prepareDEG(dfa, dfb[, pvalueLimit])	Save gene expression data of cell type of interest.

Continued on next page

Table 1 – continued from previous page

preparePerBatchCase(**kwargs)	Process gene expression data to generate per-batch distance measure and save to file.
reanalyzeMain([case])	Reanalyze case
runPairOfExperiments(args)	Analyze the case, compare it with comparison case, find the conserved genes between the cases, analyze case again
scramble(measures[, subDir, case, N, M, ...])	Run control analysis for the dendrogram order

Attributes:

combinationPanels
combinationPanelsDict
deprecatedPanels
standardPanels

```
standardPanels = ['fraction', 'binomial', 'top50', 'markers']
deprecatedPanels = ['PubMedHits', 'gAbove50_PanglaoMouse', 'gAbove50_PanglaoHuman', 'G
combinationPanels = ['combo3avgs', 'combo4avgs']
combinationPanelsDict = {'combo2avgs':  ['fraction', 'binomial'], 'combo3avgs':  ['fra
```

prepareDEG(*dfa*, *dfb*, *pvalueLimit=0.001*)

Save gene expression data of cell type of interest. Create rank dataframe (df_ranks) with genes ranked by differential expression

Parameters:

dfa: pandas.Dataframe Dataframe containing expression data for cell type of interest Has genes as rows and (batches, cells) as columns

dfb: pandas.Dataframe Dataframe containing expression data for cells of type other than cell type of interest Has genes as rows and (batches, cells) as columns

pvalueLimit: float, Default 0.001 Maximum possible p-value to include

Returns: None

Usage: prepareDEG(dfa, dfb)

preparePerBatchCase(**kwargs*)

Process gene expression data to generate per-batch distance measure and save to file. No plots are generated

Parameters: Any parameters that function 'analyzeCase' can accept

Returns: None

Usage: an = Analysis()

an.preparePerBatchCase()

prepareBootstrapExperiments(*allDataToo=True*, *df_ranks=None*, *parallel=False*)

Prepare bootstrap experiments data and calculating gene statistics for each experiment

Parameters:

allDataToo: boolean, Default True Whether to prepare experiment for all data as well

df_ranks: pd.DataFrame, Default None Genes ranked by differential expression If None function will use rank dataframe from working directory

Returns: None

Usage: an = Analysis()

an.prepareBootstrapExperiments()

compareTwoCases(*saveDir1*, *saveDir2*, *name1='N1'*, *name2='N2'*, *saveName='saveName'*)
Compare gene measurements between two cases for each bootstrap experiment

Parameters:

saveDir1: str Directory storing gene measurement data for case 1

saveDir2: str Directory storing gene measurement data for case 2

name1: str, Default 'N1' Phrase to append to keys of the resulting dataframe for case 1

name2: str, Default 'N2' Phrase to append to keys of the resulting dataframe for case 2

saveName: str, Default 'saveName' Name of file to save result dataframe to

Returns: None

Usage: an = Analysis()

an.compareTwoCases(saveDir1, saveDir2, name1, name2, saveName)

runPairOfExperiments(*args*)
Analyze the case, compare it with comparison case, find the conserved genes between the cases, analyze case again

Parameters:

saveDir: str Directory with all bootstrap experiments

saveSubDir: str Subdirectory for a bootstrap experiment

otherCaseDir: str Directory holding comparison data

Returns: None

Usage: For internal use only

analyzeBootstrapExperiments()
Analyze all bootstrap experiments

Parameters: None

Returns: None

Usage: an = Analysis()

an.analyzeBootstrapExperiments()

analyzeCombinationVariant(*variant*)
Analyze a combination of measures (same as in panels)

Parameters:

variant: str Name of combination variant (e.g. 'Avg combo4avgs', 'Avg combo3avgs')

Returns:

pandas.DataFrame Analysis result

Usage: an = Analysis()

an.analyzeCombinationVariant(variant)

scramble(*measures*, *subDir=''*, *case='All'*, *N=10000*, *M=20*, *getMax=False*, *maxSuff=''*)
Run control analysis for the dendrogram order

Parameters:

measures: list Measures (e.g: [Markers', 'Binomial -log(pvalue)', 'Top50 overlap'])

subDir: str, Default '' Subdirectory to save dataframe to

N: int Chunk size

M: int Number of chunks

Returns: None

Usage: an = Analysis()

an.scramble (measures)

analyzeCase(*df_expr*, *toggleCalculateMajorMetric=True*, *exprCutoff=0.05*, *toggleExportFigureData=True*, *toggleCalculateMeasures=True*, *suffix=''*, *saveDir=''*, *toggleGroupBatches=True*, *dpi=300*, *toggleAdjustText=True*, *markersLabelsRepelForce=1.5*, *figureSize=(8, 22)*, *toggleAdjustFigureHeight=True*, *noPlot=False*, *halfWindowSize=10*, *printStages=True*, *externalPanelsData=None*, *toggleIncludeHeatmap=True*, *addDeprecatedPanels=False*, *includeClusterNumber=True*, *togglePublicationFigure=False*)
Analyze, calculate, and generate plots for individual experiment

Parameters:

df_expr: pandas.Dataframe Gene expression data

toggleCalculateMajorMatric: boolean, Default True Whether to calculate cdist of major metric. This is a legacy parameter

exprCutoff: float, Default 0.05 Cutoff for percent expression in a batch of input data

toggleExportFigureData: boolean, Default True Whether to export figure data

toggleCalculateMeasures: boolean, Default True Whether to calculate measures

suffix: str, Default '' Name of experiment

saveDir: str, Default '' Exerything is exported to this directory, should be unique for each dataset

toggleGroupBatches: boolean, Default True Whether to group batches or save per-batch distance measure

dpi: int or 'figure', Default 300 Resolution in dots per inch, if 'float' use figures dpi value

toggleAdjustText: boolean, Default True Whether to use (external) module to minimize text overlap in figure

figure_size: tuple, Default (8, 20) Width, height in inches

toggleAdjustFigureHeight: boolean, Default True Whether to adjust figure height

noPlot: boolean, Default False Whether to generate plot

halfWindowSize: int, Default 10 Moving average half-window size

printStages: boolean, Default True Whether to print stage status to output

externalPanelsData: dict, Default None Dictionary containing additional panels data

toggleIncludeHeatmap: boolean, Default True Whether to include heatmap in figure

addDeprecatedPanels: boolean, Default False Whether to include deprecated panels

Returns: None

Usage: self.analyzeCase(df_expr)

reanalyzeMain(*case='All'*, ***kwargs*)
Reanalyze case

Parameters: Any parameters that function 'analyzeCase' can accept

Returns: None

Usage: an = Analyze()

an.reanalyzeMain()

analyzeAllPeaksOfCombinationVariant(*variant*, *nG=8*, *nE=30*, *fcutoff=0.5*, *width=50*)
Find all peaks and their frequency from the bootstrap experiments

Parameters:

variant: str Name of combination variant (e.g. 'Avg combo4avgs', 'Avg combo3avgs')

nG: int, Default 8 Number of clusters of genes

nE: int, Default 30 Number of clusters of bootstrap experiments

fcutoff: float, Default 0.5 Lower peak height cutoff

width: int, Default 50 Width of peak

Returns: None

Usage: an = Analyze()

an.analyzeAllPeaksOfCombinationVariant('Avg combo4avgs', nG=8, nE=30, fcutoff=0.5, width=50)

bootstrapMaxpeakPlot(*variant*)
Bootstrap max-peak plot

generateAnalysisReport()
Generate analysis report.

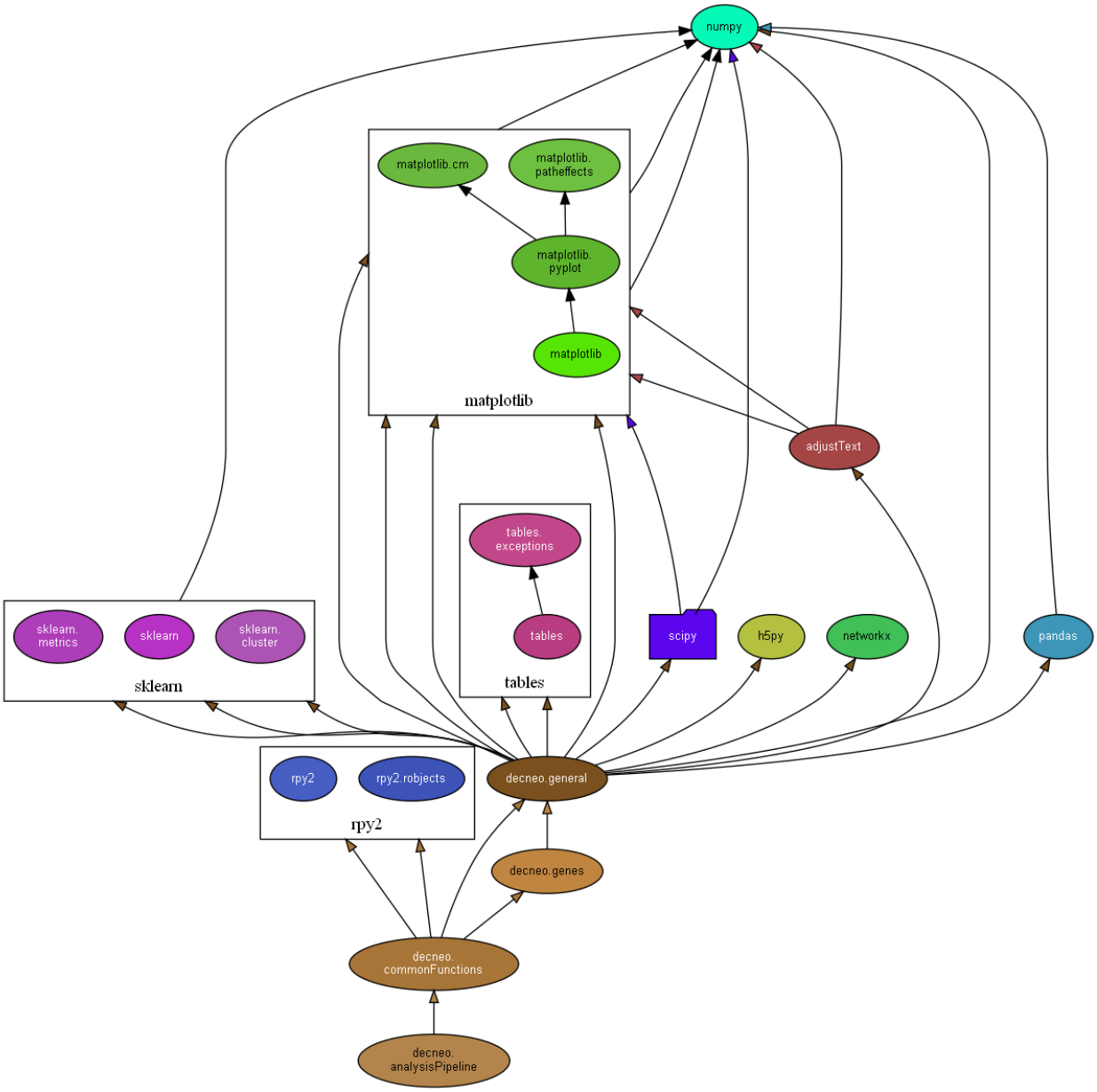
CHAPTER

FOUR

DEPENDENCY GRAPH

This graph was generated with Python module dependency visualization tool pydeps by running the following (after installation of the necessary components, i.e. Graphviz etc.):

```
pydeps decneo --reverse --max-bacon=2 --cluster --max-cluster-size=6 --min-cluster-
↪size=2 -T=png -o=docs/examples/dependency.png
```


numpy
matplotlib.cm
matplotlib.
patheffects
matplotlib.
pyplot
matplotlib
matplotlib
adjustText
tables.
exceptions
tables
tables
sklearn.
metrics
sklearn
sklearn.
cluster
sklearn
scipy
h5py
networkx
pandas
rpy2
rpy2.robjects
rpy2
decneo.general
decneo.genes
decneo.
commonFunctions
decneo.
analysisPipeline

CHAPTER

FIVE

INPUT DATA FORMAT

Expression data for **two different species** for comparison is required. For each of these species provide the input gene expression data is expected in one of the following formats:

1. Spreadsheet of comma-separated values `csv` where rows are genes, columns are cells with gene expression counts, this should be accompanied by another dataframe with two columns with one specifying batches and the other specifying corresponding cells. Alternatively, the first row of the dataframe should be `'batch'` and the second `'cell'`.

Cell vs Genes

cell	C1	C2	C3	C4
G1		3	1	7
G2	2	2		2
G3	3	1		5
G4	10		5	4
…	…	…	…	…

Batches and Cells

batch	cell
B1	C1
B1	C2
B2	C3
B3	C4
…	…

or:

batch	batch0	batch0	batch1	batch1
cell	C1	C2	C3	C4
G1		3	1	7
G2	2	2		2
G3	3	1		5
G4	10		5	4
…	…	…	…	…

2. `Pandas DataFrame` where `axis 0` is genes and `axis 1` are cells. If the are batched in the data then the index of `axis 1` should have two levels, e.g. `('batch', 'cell')`, with the first level indicating patient, batch or expreriment where that cell was sequenced, and the second level containing cell barcodes for identification.

```
df = pd.DataFrame(data=[[2,np.nan],[3,8],[3,5],[np.nan,1]],
                  index=['G1','G2','G3','G4'],
                  columns=pd.MultiIndex.from_arrays([['batch0','batch1'],['C1','C2']],
↪ names=['batch', 'cell']))
```

CHAPTER

SIX

EXAMPLES

Download file `VoightChoroid4567RemappedData.h5` (456.7 Mb) from https://zenodo.org/ which contains normalized gene expression of **27504** genes of **7996** endothelial cells from **8** batches, and **5704** non-endothelial cells from **8** batches. Genes that are not expressed in endothelial cells are removed from non-endothelial cells dataset.

Save the downloaded data file to `demo/`, or otherwise modify path in `demoData` of `demo.py`:

```
import pandas as pd
from decneo.analysisPipeline import process

demoData = '/mnt/home/domansk6/Projects/Endothelial/scripts/demo/
↪VoightChoroid4567RemappedData.h5'

if __name__ == '__main__':

    wdir = '/mnt/scratch/domansk6/DECNEOdemo/'

    process(pd.read_hdf(demoData, key='dfa'),    # Endothelial cells
            pd.read_hdf(demoData, key='dfb'),    # Non-endothelial cells
            None, None,                          # Comparison dataset is provided
            wdir,                                # Working directory
            wdir+'fromPanglaoDBmouseAllbyDCS/', # Comparison dataset
            parallelBootstrap=True,              # Set False if RAM is limited
            exprCutoff1=0.01,                    # Gene expression cutoff
            perEachOtherCase=False)              # Comparison mode setting
```

CHAPTER

SEVEN

OUTPUT DATA

Outputs all resulting directories, files, and figures to directory specified as the `workingDir` when creating an instance of class `Analysis`. It will also output an analysis report detailing all results and figures.

7.1 Directories

These subdirectories are created and:

bootstrap/

Contains a folder for each individual bootstrap experiment which includes:

Files	Description
batches.txt	List of batches used in the analysis
comparison.xlsx	Evolutionary conservation file
dendrogram-heatmap-correlation-data.(h5/xlsx)	For each gene holds all measurement data (e.g. cluster, Fraction, Top50 overlap, etc.). Contains correlation distance of expression measure.
dendrogram-heatmap-correlation.png	Saved dendrogram, heatmap, and bargraphs, see example below
metricFile.h5	Gene expression data for specified metric
per-gene-measures-correlation.(h5 & xlsx)	Intra-measures from each gene
perGeneStats.h5	For each gene holds fraction of cells expressing it, median expression, and per batch counts
size.txt	Size of input data (number of cells and genes)

byBatches/

Gene expression distance is calculated for each batch, and results are not grouped, and are further used in bootstrap analysis.

Files	Description
dendrogram-heatmap-correlation-data.(h5/xlsx)	For each gene holds all measurement data (e.g. cluster, Fraction, Top50 overlap, etc.). Contains correlation distance of expression measure.
metricsFile.h5	Gene expression data for specified metric
per-gene-measures-correlation.(h5/xlsx)	Intra-measures from each gene
perGeneStats.h5	For each gene holds fraction of cells expressing it, median expression, and per batch counts
size.txt	Size of input data (number of cells and genes)

random/

Contains files saved from Scramble function

Files	Description
combined_M_aligned.h5	Temporary file that holds distribution data; removed when analysis is completed.
se_distribution.png	Plotted counts distribution
se_distribution.xlsx	Counts distribution data

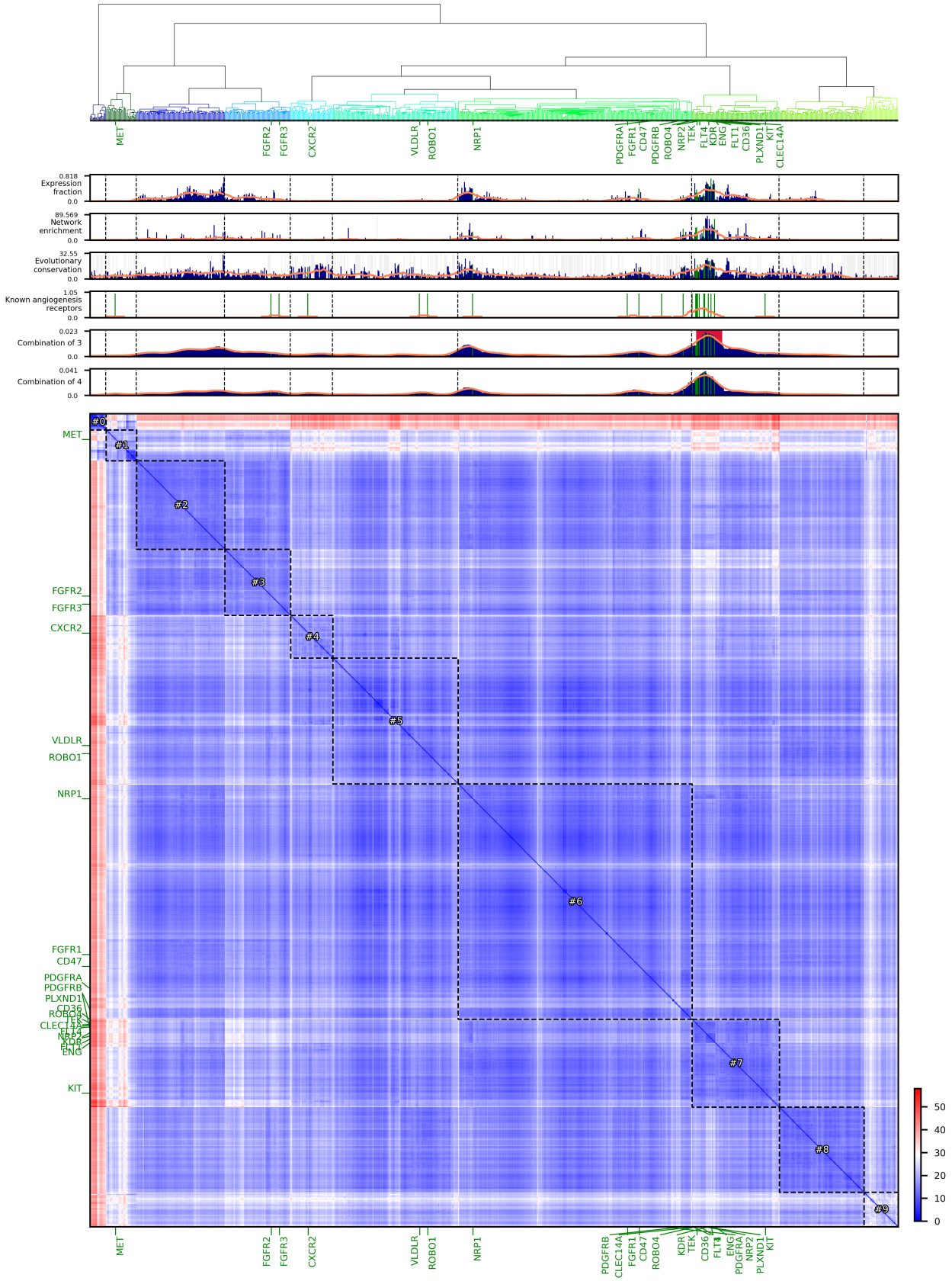
7.2 Files

Files	Description
bootstrap_experiments_dendro_data.h5	Aggregation of all gene measurement data and correlation distance of expression measure for all bootstrap experiments
data.h5	Output file from `prepareDEG` which saved expression data and ranking data of genes
results.png	Saved dendrogram, heatmap, and bargraphs, see example below

For each combination of measurements (variant) of interest each of these files are outputted:

Files	Description
bootstrap_in-peak_genes_SD.xlsx	Aggregation of all gene measurement data and correlation distance of expression measure for all bootstrap experiments
variant.xlsx	For each gene gives the percentage of bootstrap experiments in which it apppears in a peak along with mean, standard deviation, and covariance calculations. For each bootstrap experiment lists genes in the peak.
variability.xlsx	Holds mean, standard deviation, and covariance calculations

Example of result figures using mouse PangLaoDB DCS-annotated endothelial cells


MET
FGFR2
FGFR3
CXCR2
VLDLR
ROBO1
NRP1
PDGFRA
FGFR1
CD47
PDGFRB
ROBO4
NRP2
TEK
FLT4
KDR
ENG
FLT1
CD36
PLXND1
KIT
CLEC14A
0.818
Expression fraction
0.0
89.569
Network enrichment
0.0
32.55
Evolutionary conservation
0.0
1.05
Known angiogenesis receptors
0.0
0.023
Combination of 3
0.0
0.041
Combination of 4
0.0
#0
#1
#2
#3
#4
#5
#6
#7
#8
#9
50
40
30
20
10
0

CHAPTER

EIGHT

INDICES AND TABLES

- genindex
- modindex
- search

INDEX