**CODES AND TUTORIALS FOR CLUSTERING AND TRAJECTORY INFERENCE**

1. For **Seurat**: Follow the tutorial in the link below, step by step, to find the differential markers in top 25 PCA loadings using the RNA-Seq count matrix as your input data file.

<https://satijalab.org/seurat/articles/pbmc3k_tutorial.html#finding-differentially-expressed-features-cluster-biomarkers->

1. For **BigSCale** Clustering: Follow the instructions in the tutorial below with the RNA-Seq counts matrix as your input data file.

<https://github.com/iaconogi/bigSCale2#bigscale-2-gene-regulatory-networks-tutorial>

You have to create a SingleCellExperiment (sce) class with your input matrix prior to processing the clustering tutorial with BigSCale. Below are some useful codes to create the gene expression matrix for a sce.

1. For **ScEpath** Trajectory Analysis: Download the zip folder from the Github link below, unzip folder and run MATLAB code with counts matrix as input data .txt (text) file. The hyperparameter-tuned, modified code has been attached as zip file to this page.

Extract the top most differentially co-expressed markers (in all patient groups, and in all cluster populations) and subject it to the PID Network Inference as explained below.

**TO CREATE A SINGLE CELL EXPERIMENT:**

Link: <https://bioconductor.org/books/3.13/OSCA.intro/the-singlecellexperiment-class.html#storing-primary-experimental-data>

library(SingleCellExperiment)

mat <- read.csv(file = "C:/Users/15145/Desktop/GBM.csv", header=TRUE, row.names=1, check.names=FALSE)

mat <- read.csv(file = "C:/Users/15145/Desktop/GBM.csv", header=TRUE, row.names=1, check.names=FALSE)

gene.length <- mat[,1]

mat <- **as.matrix**(mat[,-1])

**dim**(mat)

sce <- **SingleCellExperiment**(assays = **list**(counts = mat))

library(bigSCale)

#Here GBM.csv is the scRNA-Seq count matrix, with genes as rows and cells as columns.

OTHER USEFUL CODE FOR BIGSCALE

GRNs: **To construct the gene expression dgcMatrix (in R):**

library(Matrix)

setwd("C:/Users/15145/Desktop") #set Directory

meta<- read.csv(file = "expr.data.csv")

mat1 <- as.matrix(meta)

gene.names<-read.csv("gene.names.csv") #make a CSV file with V1 as header and list of gene names column

mat2<- matrix(gene.names, ncol = 1) #save gene names as a matrix

library(bigSCale)

results=compute.network(expr.data = mat1,gene.names = mat2)

Option #2:

meta<- read.csv(file = "GBM.csv")

A <- as.matrix(meta, nrow=23684, ncol=292, sparse=TRUE) (removed the gene names column from expression matrix)

gene.names<-read.csv("gene.names.csv")

mat2<- matrix(gene.names, ncol = 1)

library(bigSCale)

results=compute.network(expr.data = A,gene.names = mat2)

**MATRIX NORMALIZATION**

raw\_counts <- read.csv(file = "C:/Users/15145/Desktop/GBM.csv")

library(BBmisc)

normdata <-normalize(raw\_counts, method = "standardize", range = c(0, 1), margin = 1L, on.constant = "quiet")

**MATRIX BIANRIZATION (for OACC)**

Install.packages(“Binarize”)

Or

Install.packages(“Biclust”)

Library(Binarize)

binarize<- binarizeMatrix(raw\_counts)

write.csv(logdata, file = "binary.csv")

#Alternately, use an online binary conversion tool/calculator.

**CODE FOR RECONSTRUCTING PIDC NETWORKS**

Source Code: <https://github.com/Tchanders/NetworkInference.jl#scope>

#Prepare a matrix of differentially expressed gene markers where the genes are the rows and cell samples are the columns. Save as ‘matrix.txt’ file.

#Open Julia 1.5.4 and execute following code:

using NetworkInference

infer\_network("/Users/15145/Desktop/matrix.txt", PIDCNetworkInference())

nodes = get\_nodes("/Users/15145/Desktop/matrix.txt")

inferred\_network = InferredNetwork(PIDCNetworkInference(), nodes)

write\_network\_file("/Users/15145/Desktop/Network.txt",inferred\_network)

#Set Directory to wherever the matrix file is located. Here ‘Users/15145..’ is used as an example directory.

#Once the Network.txt is obtained, you have your PID Network and this can be now processed through the Julia LightGraph code (see PIDC\_Ped.GBM.ipynb) in the folder. The code is also provided as a PDF of the Jupyter notebook. Alternately, use the R-package igraph and organize your PID network to three columns: ‘from’, ‘to’ and ‘weight’, where from are the starting nodes (genes or TFs) and to are the nodes to which they connect, weight denotes the PID score. Then, follow the tutorial in the links below and centrality measures can be used from the igraph package.

<https://www.r-bloggers.com/2020/03/community-detection-with-louvain-and-infomap/>

<https://www.r-bloggers.com/2018/12/network-centrality-in-r-an-introduction/>