A brief instruction of NCG

Contents

1.Introduction

NCG is a single-cell potency model to predict single-cell differentiation and discriminate pluripotent and non-pluripotent cells. NCG distinguishes pluripotent from non-pluripotent cells with high accuracy, correctly ranks different cell types by their differentiation potency, tracks changes during the differentiation process, and constructs the lineage trajectory from human myoblasts into skeletal muscle cells.

NCG was proposed by following steps:

(1) Computation of Edge Clustering Coefficient (ECC) based on PPI network,

(2) Computation of gene-to-gene functional similarities using GO terms,

(3) Assignment of gene-to-gene functional similarities as weights to ECC,

(4) Combination of weighted ECC with scRNA-Seq data.

This document gives an instruction of how to predict single-cell differentiation by using NCG. The following environments are required:

- R 3.6.0
- packages: mclust, igraph, isva, cluster, corpcor， preprocessCore， AnnotationDbi， org.Hs.eg.db

Some packages need to be installed using Bioconductor such as preprocessCore， AnnotationDbi and org.Hs.eg.db.

2.Data processing

In order to pre-process the data, a PPI network and a scRNA-seq data matrix with rows labeling genes and columns labeling single cells should be provided first. We can then use function `Processdata` to pre-process the data, this function will processes all datasets except Yao1 including quantile normalization, log2-transformation and other steps, and performs a gene ID conversion on PPI network.

```
source('Processdata.R');
load("hprdAsigH-13Jun12.Rd");
counts <- read.table("GSE75748_sc_cell_type_ec.csv",sep=",",header=T);
data <- Processdata(counts,hprdAsigH.m);
```

After preprocessing, we will be able to get two values, one for the scRNA-seq data matrix after preprocessing and one for the adjacency matrix of PPI network after gene ID conversion. The following code shows the dimension of the scRNA-seq data matrix after preprocessing.

```
print(dim(data$exp));
```

```
## [1] 19097  1018
```

The following code shows the adjacency matrix of PPI network after gene ID conversion.

```
print(dim(data$adj));
```

```
## [1] 8434 8434
```

The following code shows the names of the first five genes in the adjacency matrix of PPI network.

```
print(rownames(data$adj)[1:5]);
```

```
## [1] "CTSE" "EMG1" "BAG6" "MCM4" "MCM5"
```

3.Calculation

In this section, we will calculate NCG. NCG is designed by combining network topology property with edge clustering coefficient, and gene function information using gene ontology function similarity scores.

The first step is to integrate the genes in the scRNA-Seq data matrix with those in the PPI network. we use function `DoIntegPPI` to find the common genes between the scRNA-Seq data matrix and the genes present in the PPI network, and constructs the maximally connected subnetwork and reduced expression matrix for the computation of signaling entropy.

```
source('DoIntegPPI.R');
int.o <- DoIntegPPI(data$exp,data$adj);
str(int.o);
```

```
## List of 2
##  $ expMC: num [1:8027, 1:1018] 0.138 9.374 4.464 11.055 10.152 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:8027] "CTSE" "EMG1" "BAG6" "MCM4" ...
##   .. ..$ : chr [1:1018] "H1_Exp1.001" "H1_Exp1.002" "H1_Exp1.003"
"H1_Exp1.004" ...
##  $ adjMC: num [1:8027, 1:8027] 0 1 1 0 0 0 0 0 0 0 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:8027] "CTSE" "EMG1" "BAG6" "MCM4" ...
##   .. ..$ : chr [1:8027] "CTSE" "EMG1" "BAG6" "MCM4" ...
```

With the function `DoIntegPPI`, we can get two output arguments, where `expMC` represents reduced expression matrix with genes in the maximally connected subnetwork and `adjMC` represents adjacency matrix of the maximally connected subnetwork.(待讨论)

With the output object `int.o`, we can calculate the edge clustering coefficient which is necessary for the calculation of NCG using function `CompECC`. It requires one argument as input:

- PPI: The adjacency matrix of a user-given PPI network with row names and column names labeling a gene ID.

```
source('CompECC.R');
ECC <- CompECC(int.o$adjMC);
print(dim(ECC));
```

```
## [1] 8027 8027
```

The function `CompECC` will compute the Edge Clustering Coefficient matrix, whose size is the same as the PPI network matrix.

Then we can calculate NCG using function `CompNCG`, the function requires three arguments as inputs:

- Edge clustering coefficient(ECC): The output value of function `CompECC`.
- The scRNA-seq data matrix : The `expMC` which is from the output value of function `DoIntegPPI`.
- `km`: The pre-compiled pairwise Kappa similarity matrix on Gene Ontology of human genes.

This function finds the common genes between the scRNA-Seq data matrix, ECC matrix and gene-to-gene functional similarity matrix, and computes NCG value for each cell. The following code shows the calculation of NCG and the maximum and minimum values of NCG on the Chu1 dataset.

```
source('CompNCG.R');
load("hs_km.Rda");
NCG <- CompNCG(ECC,int.o$expMC,km);
print(range(NCG));
```

```
## [1] 0.6070033 1.0000000
```

4.Session information

```
sessionInfo()
```

```
## R version 3.6.0 (2019-04-26)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 19041)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=Chinese (Simplified)_China.936  LC_CTYPE=Chinese
(Simplified)_China.936
##     LC_MONETARY=Chinese (Simplified)_China.936
## [4] LC_NUMERIC=C                               LC_TIME=Chinese
(Simplified)_China.936
##
## attached base packages:
## [1] parallel  stats4    stats     graphics  grDevices utils     datasets
methods   base
##
## other attached packages:
## [1] igraph_1.2.6          org.Hs.eg.db_3.10.0   AnnotationDbi_1.48.0
IRanges_2.20.2
```

```
##     S4Vectors_0.24.4      Biobase_2.46.0
## [7] BiocGenerics_0.32.0   preprocessCore_1.48.0
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_1.0.6      lattice_0.20-44 grid_3.6.0      DBI_1.1.1
magrittr_2.0.1  RSQLite_2.2.7
##      cachem_1.0.4    rlang_0.4.11
##  [9] blob_1.2.1      Matrix_1.3-3    vctrs_0.3.8     tools_3.6.0
bit64_4.0.5     bit_4.0.4
##      fastmap_1.1.0   compiler_3.6.0
## [17] pkgconfig_2.0.3 memoise_2.0.0
```