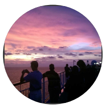
Clustering of differentially expressed genes

Ahmad Husaini AHS
Suhaimi[1]

[1]Institute of Biological Sciences, Faculty of Science, Universiti Malaya

MP2Lab



Ahmad Husaini AHS Suhaimi

JUL 06, 2023

OPEN ACCESS

DOI:
dx.doi.org/10.17504/protocols.io.rm7vzx82rgx1/v1

Protocol Citation: Ahmad Husaini AHS Suhaimi 2023. Clustering of differentially expressed genes. **protocols.io** https://dx.doi.org/10.17504/protocols.io.rm7vzx82rgx1/v1

License: This is an open access protocol distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited

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

Created: Jul 04, 2023

Last Modified: Jul 06, 2023

PROTOCOL integer ID:
84450

ABSTRACT

This differentially expressed genes clustering pipeline utilizes coseq v3.17 package (Rau & Maugis-Rabusseau, 2018) in R.

Clustering of differentially expressed genes (DEG) using Co...

1 Load the package (coseq).

Command

```
library(coseq)
library(matrixStats)
```

2 Run Coseq on transformed and normalized counts.
Example:
Performing clustering on bud data with expected clusters, K=5-16.
Clustering process is repeated for 10x.

Command

```
coseq_bud_logclr_1 = coseq(tcounts_logclr_exp_bud_ORF_scTMM[,1:15], K=5:16,
normFactor = "none", transformation = "none")
coseq_bud_logclr_2 = coseq(tcounts_logclr_exp_bud_ORF_scTMM[,1:15], K=5:16,
normFactor = "none", transformation = "none")
coseq_bud_logclr_3 = coseq(tcounts_logclr_exp_bud_ORF_scTMM[,1:15], K=5:16,
normFactor = "none", transformation = "none")
coseq_bud_logclr_4 = coseq(tcounts_logclr_exp_bud_ORF_scTMM[,1:15], K=5:16,
normFactor = "none", transformation = "none")
coseq_bud_logclr_5 = coseq(tcounts_logclr_exp_bud_ORF_scTMM[,1:15], K=5:16,
normFactor = "none", transformation = "none")
coseq_bud_logclr_6 = coseq(tcounts_logclr_exp_bud_ORF_scTMM[,1:15], K=5:16,
normFactor = "none", transformation = "none")
coseq_bud_logclr_7 = coseq(tcounts_logclr_exp_bud_ORF_scTMM[,1:15], K=5:16,
normFactor = "none", transformation = "none")
coseq_bud_logclr_8 = coseq(tcounts_logclr_exp_bud_ORF_scTMM[,1:15], K=5:16,
normFactor = "none", transformation = "none")
coseq_bud_logclr_9 = coseq(tcounts_logclr_exp_bud_ORF_scTMM[,1:15], K=5:16,
normFactor = "none", transformation = "none")
coseq_bud_logclr_10 = coseq(tcounts_logclr_exp_bud_ORF_scTMM[,1:15], K=5:16,
normFactor = "none", transformation = "none")
```

3 Manually inspect the results and decide on the average number of clusters
Choose one clustering result to proceed with the subsequent steps

Command

```
summary(coseq_bud_logclr_1)
summary(coseq_bud_logclr_2)
summary(coseq_bud_logclr_3)
summary(coseq_bud_logclr_4)
summary(coseq_bud_logclr_5)
summary(coseq_bud_logclr_6)
summary(coseq_bud_logclr_7)
summary(coseq_bud_logclr_8)
summary(coseq_bud_logclr_9)
summary(coseq_bud_logclr_10)
```

Assigning clusters to transcripts

4 Retrieve and tabulate the clustering information based on the chose clustering from the previous step.
Example:
coseq_bud_logclr_1 is chosen as the best clustering
results_coseq_bud_logclr: the new table/vector.

Command

```
results_coseq_bud_logclr = clusters(coseq_bud_logclr_1)
```

5 Convert the vector into a data frame.

Command

```
results_coseq_bud_logclr = data.frame(results_coseq_bud_logclr)
```

6 Create a column containing the assigned cluster number for each transcript in the read count data frame.
Example:
the new column: bud_logclr
the data frame with read counts: tcounts_logclr_exp_bud_ORF_scTMM

Command

```
tcounts_logclr_exp_bud_ORF_scTMM$bud_logclr =
results_coseq_bud_logclr_1$results_coseq_bud_logclr
```