Clustering high-throughput sequencing data based on patterns of co-expression

Thomas J. Hardcastle

May 29, 2026

1 Introduction

This vignette outlines two possibilities for clustering gene expression based on patterns of co-expression between samples, optionally accounting for the replicate structure of the data.

The package can be installed as

```
> if (!requireNamespace("BiocManager", quietly=TRUE))
+     install.packages("BiocManager")
> BiocManager::install("clusterSeq")
```

```
> library(clusterSeq)
```

We demonstrate these analyses on a set of time series data in female rat thymus tissues taken from the Rat Bodymap project [1]. By identifying clusters of genes that demonstrate similar patterns of expression over time we can identify patterns of time dependence within the data.

We first load in the processed data of observed read counts at each gene for each sample.

```
> data(ratThymus, package = "clusterSeq")
> head(ratThymus)
```

We define the replicate structure of the data in a vector whose members correspond to the columns of the data matrix.

```
> replicates <- c("2week","2week","2week","2week",
+                 "6week","6week","6week","6week",
+                 "21week","21week","21week","21week",
+                 "104week","104week","104week","104week")
```

Library scaling factors are acquired here using the `baySeq::getLibsizes` function but might be acquired through any other means.

```
>   library(baySeq)
>   libsizes <- getLibsizes(data = ratThymus)
```

2 K-means based clustering

For k-means based clustering, we assume an approximately log-normal distribution. We adjust the data to remove zeros and rescale by the library scaling factors.

```
> ratThymus[ratThymus == 0] <- 1
> normRT <- log2(ratThymus %*% diag(1/libsizes) * mean(libsizes))
```

For speed purposes, we will consider only the first 1000 genes in the data.

```
> normRT <- normRT[1:1000,]
```

K-means based clustering compares two genes by separately clustering the expression values of each gene using all possible values of k (or, if the sample size is exceptionally large, all values of k from 1 to some large maximum permitted value.). The genes are then compared to find the maximum k for which the average expression of the identified clusters is monotonic between the two genes. For this value of k, the the maximum difference between expression levels observed within a cluster of either gene is reported as a measure of the dissimilarity between the two genes.

However, a problem arises for genes that are non-differentially expressed across all samples. For these genes, the appropriate number of clusters is 1; however, choices of $k > 1$ often lead to false assignment of genes to clusters exhibiting differential expression. To resolve this, the clustering incorporates a bootstrapping stage based on Tibshirani's gap statistic. Bootstrapping uniformly distributed data on the same range as the observed data, we calculate the the dissimilarity score as above, and find those cases for which the gap between the bootstrapped mean dissimilarity and the observed dissimilarity for k $= 1$ exceeds that for k $= 2$ by more than some multiple of the standard error of the bootstrapped dissimilarities of k $= 2$. These cases are forced to be treated as non-differentially expressed by discarding all dissimilarity data for k > 1.

We generate a matrix of dissimilarity scores between each pair of genes using the `kCluster` function. The full matrix can be reported to (gzipped) file; the function returns a data.frame which for each gene defines its nearest neighbour of higher row index, and the dissimilarity with that neighbour. This is sufficient for a clustering based on singleton agglomeration (see `hclust`).

```
> kClust <- kCluster(normRT, matrixFile = "kclust_matrix.txt.gz", B = 1000)
> head(kClust)
```

Using the output from 'kCluster' we construct a clustering of genes by singleton agglomeration of those genes with dissimilarity scores lower than a specified threshold.

```
> mkClust <- makeClusters(kClust, normRT, threshold = 1)
```

We can repeat this analysis forcing members of the same replicate groups to cluster together at the k-means stage.

```
> kClustR <- kCluster(normRT, replicates = replicates, matrixFile = "kclust_matrix_newForceReps.txt.gz", B
> mkClustR <- makeClusters(kClustR, normRT, threshold = 1)
```

As an alternative to singleton agglomeration, we can use the full matrix and any agglomerative method defined in the `hclust` heirarchical analysis function.

```
> mkClustRC <- makeClustersFF("kclust_matrix_newForceReps.txt.gz", method = "complete", cut.height = 5)
```

3 Posterior likelihood analyses

As an alternative to the K-means based clustering, we can use empirical Bayesian likelihoods on all possible models for the structure of the data at each gene, and from these estimate the likelihood that two genes share the same structure. We estimate these likelihoods using the baySeq package.

We first create a `countData` object to contain the data and find all possible models that preserve the replicate structure.

```
> library(baySeq)
> cD.ratThymus <- new("countData", data = ratThymus, replicates = c(1,1,1,1,2,2,2,2,3,3,3,3,4,4,4,4))
> densityFunction(cD.ratThymus) <- nbinomDensity
> libsizes(cD.ratThymus) <- getLibsizes(cD.ratThymus)
> cD.ratThymus <- allModels(cD.ratThymus)
```

The commented out code is for the analysis of the data with baySeq. Since this takes some time, this has already been performed and can be loaded in with the analysis already performed.

```
> #cl <- makeCluster(4)
> #cD.ratThymus <- getPriors(cD.ratThymus, consensus = TRUE, cl = cl)
> #cD.ratThymus <- getLikelihoods(cD.ratThymus, cl = cl)
>
> data(cD.ratThymus, package = "clusterSeq")
```

Given the estimated posteriors for each gene to have each model, we estimate the likelihood that two genes have identical models, and that the ordering of average expression in the groups defined by those models is monotonic between the two genes. As before, the function `associatePosteriors` returns a data.frame which for each gene defines its nearest neighbour of higher row index, and the dissimilarity with that neighbour. This is sufficient for a clustering based on singleton agglomeration using the `makeClusters` function.

```
> cD.ratThymus <- cD.ratThymus[1:1000,]
> aM <- associatePosteriors(cD.ratThymus)
> sX <- makeClusters(aM, cD.ratThymus, threshold = 0.5)
```

Likelihoods of a pair being clustered together in one clustering given that they are clustered in that way in the other.

```
> wallace(sX, mkClustRC)

P.v1_given_v2 P.v2_given_v1
    1.0000000     0.2787628
```

```
> par(mfrow = c(2,3))
> plotCluster(sX[1:6], cD.ratThymus)
```

«label=plotClusterBS-fig, fig=TRUE,echo=FALSE»= «plotClusterBS» @

Figure 1: Profiles of gene expression in top six clusters acquired from baySeq analysis.

Session Info

```
> sessionInfo()

R version 4.6.0 (2026-04-24)
Platform: x86_64-pc-linux-gnu
Running under: Ubuntu 24.04.4 LTS

Matrix products: default
BLAS:   /usr/lib/x86_64-linux-gnu/openblas-pthread/libblas.so.3
LAPACK: /usr/lib/x86_64-linux-gnu/openblas-pthread/libopenblasp-r0.3.26.so;  LAPACK version 3.12.0

locale:
 [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
 [3] LC_TIME=en_US.UTF-8        LC_COLLATE=en_US.UTF-8
 [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
 [9] LC_ADDRESS=C               LC_TELEPHONE=C
[11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C

time zone: Etc/UTC
tzcode source: system (glibc)

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] clusterSeq_1.36.0   baySeq_2.46.0       BiocParallel_1.46.0

loaded via a namespace (and not attached):
 [1] cli_3.6.6            knitr_1.51           rlang_1.2.0
 [4] xfun_0.57            generics_0.1.4       statmod_1.5.2
 [7] S4Vectors_0.50.1     buildtools_1.0.0     BiocStyle_2.40.0
[10] htmltools_0.5.9      maketools_1.3.2      sys_3.4.3
[13] stats4_4.6.0         locfit_1.5-9.12      rmarkdown_2.31
[16] grid_4.6.0           Seqinfo_1.2.0        evaluate_1.0.5
[19] abind_1.4-8          fastmap_1.2.0        yaml_2.3.12
[22] IRanges_2.46.0       BiocManager_1.30.27  compiler_4.6.0
[25] codetools_0.2-20     limma_3.68.3         edgeR_4.10.1
[28] lattice_0.22-9       digest_0.6.39        parallel_4.6.0
[31] GenomicRanges_1.64.0 tools_4.6.0          BiocGenerics_0.58.1
```

References

[1] Yu, Ying and Fuscoe, James C and Zhao, Chen and Guo, Chao and Jia, Meiwen and Qing, Tao and Bannon, Desmond I and Lancashire, Lee and Bao, Wenjun and Du, Tingting and Luo, Heng and Su, Zhenqiang and Jones, Wendell D and Moland, Carrie L and Branham, William S and Qian, Feng and Ning, Baitang and Li, Yan and Hong, Huixiao and Guo, Lei and Mei, Nan and Shi, Tieliu and Wang, Kevin Y and Wolfinger,

Russell D and Nikolsky, Yuri and Walker, Stephen J and Duerksen-Hughes, Penelope and Mason, Christopher E and Tong, Weida and Thierry-Mieg, Jean and Thierry-Mieg, Danielle and Shi, Leming and Wang, Charles *A rat RNA-Seq transcriptomic BodyMap across 11 organs and 4 developmental stages.* Nature Communications (2014).