Grade of Membership Model and Visualization for RNA-seq data using *CountClust*

Kushal K Dey, Chiaowen Joyce Hsiao & Matthew Stephens

Stephens Lab, The University of Chicago

*Correspondending Email: `mstephens@uchicago.edu`

May 15, 2016

Abstract

Grade of membership or GoM models (also known as admixture models or Latent Dirichlet Allocation") are a generalization of cluster models that allow each sample to have membership in multiple clusters. It is widely used to model ancestry of individuals in population genetics based on SNP/ microsatellite data and also in natural language processing for modeling documents [1, 3].

This *R* package implements tools to visualize the clusters obtained from fitting topic models using a Structure plot [2] and extract the top features/genes that distinguish the clusters. In presence of known technical or batch effects, the package also allows for correction of these confounding effects.

CountClust version: 1.0.2 [1]

[1]This document used the vignette from *Bioconductor* package *DESeq2, cellTree* as *knitr* template

Contents

1 Introduction

In the context of RNA-seq expression (bulk or singlecell seq) data, the grade of membership model allows each sample (usually a tissue sample or a single cell) to have some proportion of its RNA-seq reads coming from each cluster. For typical bulk RNA-seq experiments this assumption can be argued as follows: each tissue sample is a mixture of different cell types, and so clusters could represent cell types (which are determined by the expression patterns of the genes), and the membership of a sample in each cluster could represent the proportions of each cell type present in that sample.

Many software packages available for document clustering are applicable to modeling RNA-seq data. Here, we use the R package `maptpx` [4] to fit these models, and add functionality for visualizing the results and annotating clusters by their most distinctive genes to help biological interpretation. We also provide additional functionality to correct for batch effects and also compare the outputs from two different grade of membership model fits to the same set of samples but different in terms of feature description or model assumptions.

2 CountClust Installation

CountClust requires the following CRAN-R packages: *maptpx*, *ggplot2*, *cowplot*, *parallel*, *reshape2*, *RColorBrewer*, *flexmix*, *gtools*, *devtools* along with the *Bioconductor* package: *limma*.

```
source("http://bioconductor.org/biocLite.R")
biocLite("CountClust")
```

For the latest updated version of the package maptpx package (with some bug fixes from the CRAN version), we recommend the user to install the Github version.

```
library(devtools)
install_github("TaddyLab/maptpx")
```

The working version of the *CountClust* package is available on Github, along with the data packages used as examples in this vignette.

```
install_github('kkdey/CountClust')
```

Then load the package with:

```
library(CountClust)
```

3 Data Preparation

We install data packages as `expressionSet` objects for bulk RNA-seq reads data from Brain tissue samples of human donors under GTEx (Genotype Tissue Expression) V6 Project [7] and a singlecell RNA-seq reads data across developmental stages in mouse embryo due to Deng *et al* 2014 [6].

`singleCellRNASeqMouseDeng2014` data package due to Deng *et al* is a processed version of the data publicly available at Gene Expression Omnibus (GEO:GSE45719: see http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE45719).

```
library(devtools)

read.data1 = function() {
  x = tempfile()
  download.file('https://cdn.rawgit.com/kkdey/singleCellRNASeqMouseDeng2014/master/data/De
  z = get(load((x)))
  return(z)
}

Deng2014MouseESC <- read.data1()

## Alternatively
# install_github('kkdey/singleCellRNASeqMouseDeng2014')
```

`GTExV6Brain` The data package for GTEx V6 Brain sample data is again a processed version of the data publicly available at the GTEx Portal (http://www.gtexportal.org/home/, dbGaP accession phs000424.v6.p1, release date: Oct 19, 2015).

```
read.data2 = function() {
  x = tempfile()
  download.file('https://cdn.rawgit.com/kkdey/GTExV6Brain/master/data/GTExV6Brain.rda', des
  z = get(load((x)))
  return(z)
}

GTExV6Brain <- read.data2()

## Alternatively
# install_github('kkdey/GTExV6Brain')
```

3.0.1 Deng et al 2014

Load the scRNA-seq data due to Deng *et al* 2014.

```
deng.counts <- Biobase::exprs(Deng2014MouseESC)
deng.meta_data <- Biobase::pData(Deng2014MouseESC)
deng.gene_names <- rownames(deng.counts)
```

3.0.2 GTEx V6 Brain

Load the bulk-RNA seq data from GTEx V6 brain data.

```
gtex.counts <- Biobase::exprs(GTExV6Brain)
gtex.meta_data <- Biobase::pData(GTExV6Brain)
gtex.gene_names <- rownames(gtex.counts)
```

4 Fitting the GoM Model

We use a wrapper function of the *topics()* function in the *maptpx* due to Matt Taddy [4].

As an example, we fit the topic model for `k`=4 on the GTEx V6 Brain data and save the GoM model output file to user-defined directory.

```
FitGoM(t(gtex.counts),
            K=4, tol=0.1,
            path_rda="../data/GTExV6Brain.FitGoM.rda")
```

One can also input a vector of clusters under `nclus_vec` as we do for a list of cluster numbers from 2 to 7 for Deng *et al* 2014 data.

```
FitGoM(t(deng.counts),
            K=2:7, tol=0.1,
            path_rda="../data/MouseDeng2014.FitGoM.rda")
```

5 Structure plot visualization

Now we perform the visualization for k=6 for the Deng *et al* 2014 data.

We load the GoM fit for k=6.

```
data("MouseDeng2014.FitGoM")
names(MouseDeng2014.FitGoM$clust_6)

## [1] "K"     "theta" "omega" "BF"    "D"     "X"

omega <- MouseDeng2014.FitGoM$clust_6$omega
```

We prepare the annotations for the visualization.

```
annotation <- data.frame(
  sample_id = paste0("X", c(1:NROW(omega))),
  tissue_label = factor(rownames(omega),
                        levels = rev( c("zy", "early2cell",
                                        "mid2cell", "late2cell",
                                        "4cell", "8cell", "16cell",
                                        "earlyblast","midblast",
                                         "lateblast") ) ) )

rownames(omega) <- annotation$sample_id;
```

Now we perform the visualization.

In the above plot, the samples in each batch have been sorted by the proportional memebership of the most representative cluster in that batch. One can also use `order_sample=FALSE` for the un-ordered version, which retains the order as in the data (see Supplementary analysis for example).

Now we perform the Structure plot visualization for k=4 for GTEx V6 data for Brain samples .

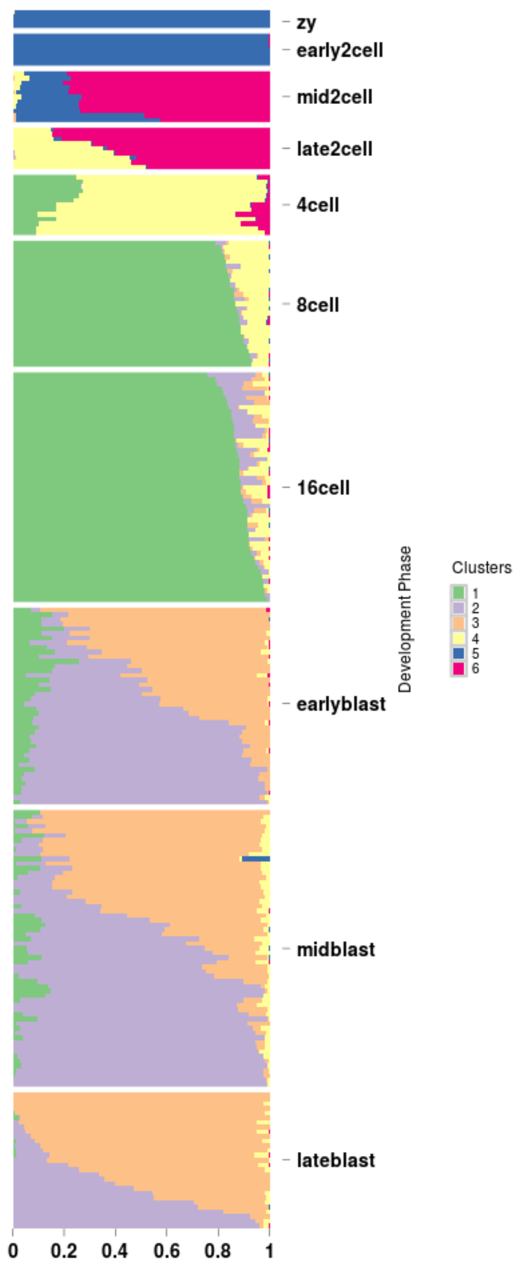
We load the GoM fit for k=4.

```
data("GTExV6Brain.FitGoM")
omega <- GTExV6Brain.FitGoM$omega;
dim(omega)

## [1] 1259    4

colnames(omega) <- c(1:NCOL(omega))
```

```
StructureGGplot(omega = omega,
                annotation = annotation,
                palette = RColorBrewer::brewer.pal(8, "Accent"),
                yaxis_label = "Development Phase",
                order_sample = TRUE,
                axis_tick = list(axis_ticks_length = .1,
                                 axis_ticks_lwd_y = .1,
                                 axis_ticks_lwd_x = .1,
                                 axis_label_size = 7,
                                 axis_label_face = "bold"))
```



We now prepare the annotations for the visualization.

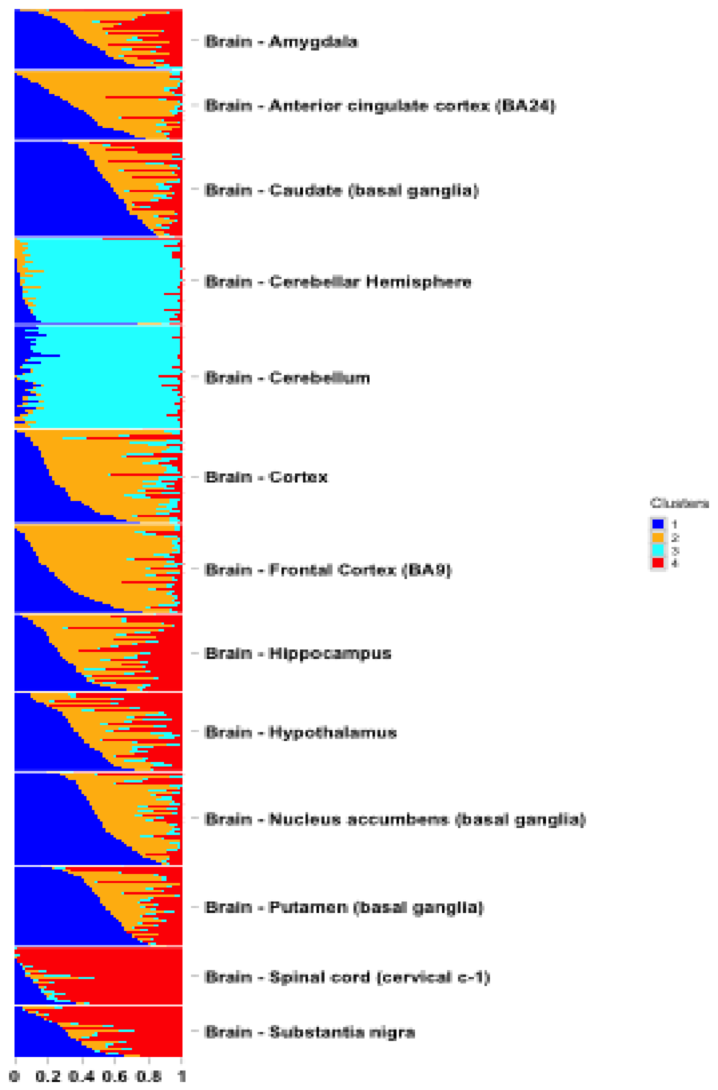
```
tissue_labels <- gtex.meta_data[,3];


annotation <- data.frame(
    sample_id = paste0("X", 1:length(tissue_labels)),
    tissue_label = factor(tissue_labels,
                          levels = rev(unique(tissue_labels) ) ) );

cols <- c("blue", "darkgoldenrod1", "cyan", "red")
```

Now we perform the visualization.

```
StructureGGplot(omega = omega,
                annotation= annotation,
                palette = cols,
                yaxis_label = "",
                order_sample = TRUE,
                split_line = list(split_lwd = .4,
                                  split_col = "white"),
                axis_tick = list(axis_ticks_length = .1,
                                 axis_ticks_lwd_y = .1,
                                 axis_ticks_lwd_x = .1,
                                 axis_label_size = 7,
                                 axis_label_face = "bold"))
```


Brain - Amygdala
Brain - Anterior cingulate cortex (BA24)
Brain - Caudate (basal ganglia)
Brain - Cerebellar Hemisphere
Brain - Cerebellum
Brain - Cortex
Brain - Frontal Cortex (BA9)
Brain - Hippocampus
Brain - Hypothalamus
Brain - Nucleus accumbens (basal ganglia)
Brain - Putamen (basal ganglia)
Brain - Spinal cord (cervical c-1)
Brain - Substantia nigra
0
0.2
0.4
0.6
0.8
1
Clusters
1
2
3
4

6 Cluster Annotations

We extract the top genes driving each cluster using the `ExtractTopFeatures()` functionality of the *CountClust* package. We first perform the cluster annotations from the GoM model fit with k=6 on the single cell RNA-seq data due to Deng *et al*

```
theta_mat <- MouseDeng2014.FitGoM$clust_6$theta;
top_features <- ExtractTopFeatures(theta_mat, top_features=100,
                                   method="poisson", options="min");
gene_list <- do.call(rbind, lapply(1:dim(top_features)[1],
                        function(x) deng.gene_names[top_features[x,]]))
```

We tabulate the top 5 genes for these 6 clusters.

```
xtable::xtable(gene_list[,1:5])

## % latex table generated in R 3.3.0 by xtable 1.8-2 package
## % Sun May 15 23:03:38 2016
## \begin{table}[ht]
## \centering
## \begin{tabular}{rlllll}
##   \hline
##  & 1 & 2 & 3 & 4 & 5 \\
##   \hline
## 1 & Timd2 & Isyna1 & Alppl2 & Pramel5 & Hsp90ab1 \\
##   2 & Upp1 & Tdgf1 & Aqp8 & Fabp5 & Tat \\
##   3 & Actb & Krt18 & Fabp3 & Id2 & Tspan8 \\
##   4 & Rtn2 & Ebna1bp2 & Zfp259 & Nasp & Cenpe \\
##   5 & LOC100502936 & Bcl2l10 & Tcl1 & E330034G19Rik & Oas1d \\
##   6 & Obox3 & Zfp352 & Gm8300 & Usp17l5 & BB287469 \\
##    \hline
## \end{tabular}
## \end{table}
```

	1	2	3	4	5
1	Timd2	Isyna1	Alppl2	Pramel5	Hsp90ab1
2	Upp1	Tdgf1	Aqp8	Fabp5	Tat
3	Actb	Krt18	Fabp3	Id2	Tspan8
4	Rtn2	Ebna1bp2	Zfp259	Nasp	Cenpe
5	LOC100502936	Bcl2l10	Tcl1	E330034G19Rik	Oas1d
6	Obox3	Zfp352	Gm8300	Usp17l5	BB287469

We next perform the same for the topic model fit on the GTEx V6 Brain samples data with k=4 clusters.

```
theta_mat <- GTExV6Brain.FitGoM$theta;
top_features <- ExtractTopFeatures(theta_mat, top_features=100,
                                   method="poisson", options="min");
gene_list <- do.call(rbind, lapply(1:dim(top_features)[1],
                        function(x) gtex.gene_names[top_features[x,]]))
```

The top 3 genes (ensemble IDs) driving these 4 clusters.

```
xtable::xtable(gene_list[,1:3])
```

	1	2	3
1	ENSG00000120885.15	ENSG00000130203.5	ENSG00000131771.9
2	ENSG00000171617.9	ENSG00000160014.12	ENSG00000154146.8
3	ENSG00000112139.10	ENSG00000139899.6	ENSG00000008710.13
4	ENSG00000197971.10	ENSG00000266844.1	ENSG00000237973.1

References

[1] Pritchard, Jonathan K., Matthew Stephens, and Peter Donnelly. Inference of population structure using multilocus genotype data. *Genetics*. 155.2, 945-959, 200.

[2] Rosenberg NA, Pritchard JK, Weber JL, Cann HM, Kidd KK, Zhivotovsky LA, Feldman MW. The genetic structure of human populations. *Science*. 298, 2381-2385, 2002.

[3] Blei DM, Ng AY, Jordan MI. Latent Dirichlet Allocation. *J. Mach. Learn. Res.*. 3, 993-1022, 2003.

[4] Matt Taddy. On Estimation and Selection for Topic Models. *AISTATS 2012, JMLR W&CP 22*.(maptpx R package), 2012.

[5] Jaitin DA, Kenigsberg E et al. Massively Parallel Single-Cell RNA-Seq for Marker-Free Decomposition of Tissues into Cell Types. *Science*. 343 (6172) 776-779, 2014.

[6] Deng Q, Ramskold D, Reinius B, Sandberg R. Single-Cell RNA-Seq Reveals Dynamic, Random Monoallelic Gene Expression in Mammalian Cells. *Science*. 343 (6167) 193-196, 2014.

[7] The GTEx Consortium. The Genotype-Tissue Expression (GTEx) project. *Nature genetics*. 45(6): 580-585. doi:10.1038/ng.2653, 2013.

7 Supplementary analysis

As an additional analysis, we apply the *CountClust* tools on another single-cell RNA-seq data from mouse spleen due to Jaitin *et al* 2014 [5]. The data had technical effects in the form of `amplificationbatch` which the user may want to correct for.

We first install and load the data.

```
read.data3 = function() {
  x = tempfile()
  download.file('https://cdn.rawgit.com/jhsiao999/singleCellRNASeqMouseJaitinSpleen/master/
  z = get(load((x)))
  return(z)
}

MouseJaitinSpleen <- read.data3()

## Alternatively
# devtools::install_github('jhsiao999/singleCellRNASeqMouseJaitinSpleen')
```

```
jaitin.counts <- Biobase::exprs(MouseJaitinSpleen)
jaitin.meta_data <- Biobase::pData(MouseJaitinSpleen)
jaitin.gene_names <- rownames(jaitin.counts)
```

Extracting the non-ERCC genes satisfying some quality measures.

```
ENSG_genes_index <- grep("ERCC", jaitin.gene_names, invert = TRUE)
jaitin.counts_ensg <- jaitin.counts[ENSG_genes_index, ]
filter_genes <- c("M34473","abParts","M13680","Tmsb4x",
                  "S100a4","B2m","Atpase6","Rpl23","Rps18",
                  "Rpl13","Rps19","H2-Ab1","Rplp1","Rpl4",
                  "Rps26","EF437368")
fcounts <- jaitin.counts_ensg[ -match(filter_genes, rownames(jaitin.counts_ensg)), ]
sample_counts <- colSums(fcounts)

filter_sample_index <- which(jaitin.meta_data$number_of_cells == 1 &
                              jaitin.meta_data$group_name == "CD11c+" &
                                 sample_counts > 600)
fcounts.filtered <- fcounts[,filter_sample_index];
```

We filter the metadata likewise

```
jaitin.meta_data_filtered <- jaitin.meta_data[filter_sample_index, ]
```

We fit the GoM model for `k`=7 and plot the Structure plot visualization to show that the amplification batch indeed drives the clustering patterns.

```
StructureObj(t(fcounts),
            nclus_vec=7, tol=0.1,
             path_rda="../data/MouseJaitinSpleen.FitGoM.rda")
```

```
data("MouseJaitinSpleen.FitGoM")
names(MouseJaitinSpleen.FitGoM$clust_7)
```

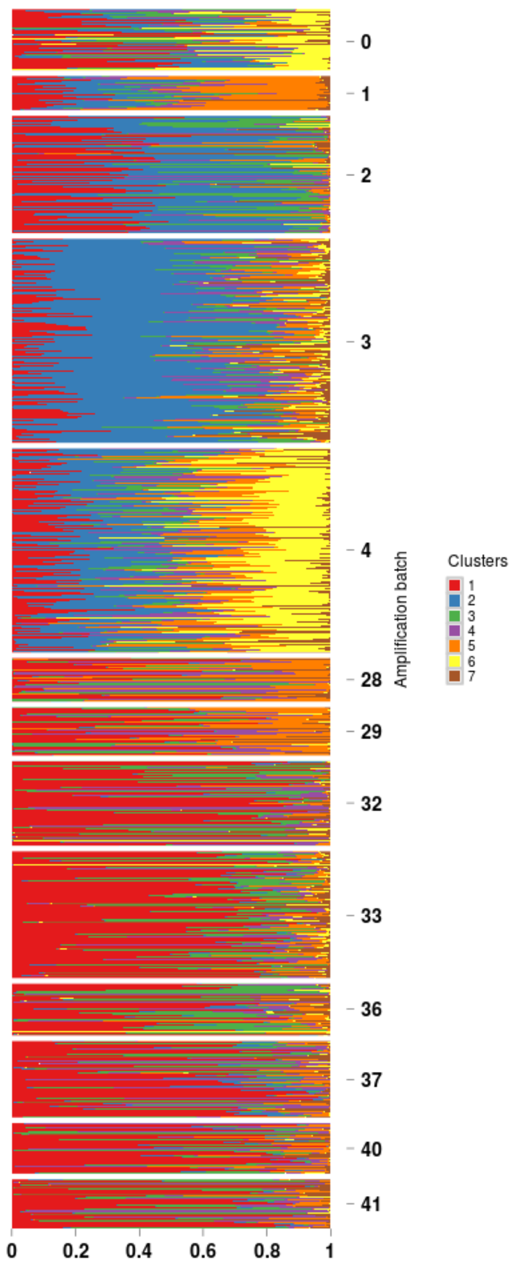
```
## [1] "K"     "theta" "omega" "BF"    "D"     "X"
```

```
omega <- MouseJaitinSpleen.FitGoM$clust_7$omega

amp_batch <- as.numeric(jaitin.meta_data_filtered[ , "amplification_batch"])
annotation <- data.frame(
    sample_id = paste0("X", c(1:NROW(omega)) ),
    tissue_label = factor(amp_batch,
                          levels = rev(sort(unique(amp_batch))) ) )
```

It seems from the above Structure plot that `amplification`batch drives the clusters. To remove the effect of amplification batch, one can use. For this, we use the `BatchCorrectedCounts()` functionality of the package.

```
StructureGGplot(omega = omega,
                annotation = annotation,
                palette = RColorBrewer::brewer.pal(9, "Set1"),
                yaxis_label = "Amplification batch",
                order_sample = FALSE,
                axis_tick = list(axis_ticks_length = .1,
                                 axis_ticks_lwd_y = .1,
                                 axis_ticks_lwd_x = .1,
                                 axis_label_size = 7,
                                 axis_label_face = "bold"))
```



```
batchcorrect.fcounts <- BatchCorrectedCounts(t(fcounts.filtered),
                                          amp_batch, use_parallel = FALSE);
dim(batchcorrect.fcounts)
```

8 Acknowledgements

We would like to thank Deng *et al* and the GTEx Consortium for having making the data publicly available. We would like to thank Matt Taddy, Amos Tanay, Po Yuan Tung and Raman Shah for helpful discussions related to the project and the package.

9 Session Info

```
sessionInfo()

## R version 3.3.0 (2016-05-03)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 14.04.4 LTS
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=en_US.UTF-8        LC_COLLATE=C
##  [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
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] CountClust_1.0.2 ggplot2_2.1.0    devtools_1.11.1  knitr_1.13
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_0.12.5         RColorBrewer_1.1-2  formatR_1.4
##  [4] git2r_0.15.0        plyr_1.8.3          highr_0.6
##  [7] tools_3.3.0         digest_0.6.9        evaluate_0.9
## [10] memoise_1.0.0       gtable_0.2.0        nlme_3.1-128
## [13] lattice_0.20-33     mgcv_1.8-12         Matrix_1.2-6
## [16] curl_0.9.7          parallel_3.3.0      cluster_2.0.4
## [19] withr_1.0.1.9000    httr_1.1.0          stringr_1.0.0
## [22] gtools_3.5.0        stats4_3.3.0        grid_3.3.0
## [25] nnet_7.3-12         cowplot_0.6.2       Biobase_2.32.0
## [28] R6_2.1.2            maptpx_1.9-2        flexmix_2.3-13
## [31] limma_3.28.4        reshape2_1.4.1      magrittr_1.5
## [34] BiocGenerics_0.18.0 scales_0.4.0        modeltools_0.2-21
## [37] MASS_7.3-45         xtable_1.8-2        BiocStyle_2.0.2
## [40] picante_1.6-2       permute_0.9-0       colorspace_1.2-6
## [43] ape_3.4             labeling_0.3        stringi_1.0-1
## [46] munsell_0.4.3       slam_0.1-34         vegan_2.3-5
```