Semi supervised Grade of Membership models for RNA-seq data using *classtpx*

Kushal K Dey, Chiaowen Joyce Hsiao & Matthew Stephens

Stephens Lab, The University of Chicago

*Correspondending Email: `mstephens@uchicago.edu`

April 1, 2016

Contents

1 Introduction

The Grade of Membership (GoM) model, as fitted by the `topics()` function of the package *maptpx* or the `FitGoM()` in the package CountClust, suffers from the issue of identifiability, the reason being clusters that are determined in a completely unsupervised way. Oftentimes, we may have information about expression patterns of the biological clusters of interest. To cite an example, suppose we have RNA-seq data on blood samples (bulk or single cell) and also RNA-seq data on some FACS sorted single cells from Blood identifying different Blood cell types. Sequencing these FACS sorted cells gives us information of the expression patterns of the underlying cell types. One can view these cell types as potential clusters in clustering of RNA-seq data (non FACS sorted) from Blood. It is this prior information about the clusters that can ultimately lead to more distinct patterns of expression compared to unsupervised models and give a better sense about the mixing proportions of different cell types of interest in the samples.

2 Installation

To install the current working version of the *classtpx* package, make sure FORTRAN compatibility.

```
#curl -O http://r.research.att.com/libs/gfortran-4.8.2-darwin13.tar.bz2
#sudo tar fvxz gfortran-4.8.2-darwin13.tar.bz2 -C /
```

To install the Github working version,

```
devtools::install_github("kkdey/classtpx")
```

To load the package

```
library("classtpx")
```

3 Data Preparation

Recently a number of studies have been published on FACS sorted RNA-sequqnicng data with the aim of identifying distinct cell types of cell cycle phases. We are trying to build a library of data packages that seem relevant to our research interest and have them available as ExpressionSet objects (integrating the expression data succinctly with metadata information on samples and features). You can find a number of these data packages hosted on our Github pages https://github.com/kkdey?tab=repositories and https://github.com/jhsiao999?tab=repositories.

We present an example of the Leng et al 2015 data [paper site: http://www.nature.com/nmeth/journal/v12/n10/full/nmeth.3549.html] of RNA-sequencing data on human embryonic stem cells. Total 213 H1 single cells and 247 H1-Fucci single cells were sequenced. The 213 H1 cells were used to evaluate Oscope in identifying oscillatory genes. The H1-Fucci cells were used to confirm the cell cycle gene cluster identified by Oscope in the H1 hESCs. In the dataset, we had cells labeled H1 (213), G1 (91), S (80) and G2 (76).

```
devtools::install_github("kkdey/singleCellRNASeqHumanLengESC", force=TRUE)
```

```
library(singleCellRNASeqHumanLengESC)
data("HumanLengESC")
leng_gene_names <- Biobase::featureNames(HumanLengESC);

leng_data <- t(Biobase::exprs(HumanLengESC));
leng_metadata <- Biobase::pData(HumanLengESC)
leng_cell_state <- leng_metadata$cell_state;

table(leng_cell_state)
```

```
## leng_cell_state
##  G1  G2  H1   S
##  91  76 213  80
```

4 Methods and Materials

The general framework for Grade of Membership (GoM) models is as follows.

suppose c_{ng} represents the read counts for sample n and gene g. Then we assume the model

$$(c_{n1}, c_{n2}, \cdots, c_{nG}) \sim Mult\left(c_{n+}, p_{n1}, p_{n2}, \cdots, p_{nG}\right) \tag{1}$$

where p_{ng} represents the probability of observing a read mapping to gene g from sample n. We write this probability as

$$p_{ng} = \sum_{k=1}^{K} \omega_{nk}\theta_{kg} \qquad \sum_{k=1}^{K} \omega_{nk} = 1 \quad \forall n \qquad \sum_{g=1}^{G} \theta_{kg} = 1 \quad \forall k \tag{2}$$

where ω_{nk} represents membership probability of k th cluster in the n th sample and θ_{kg} represents cluster mass at gene g for cluster k.

In standard GoM models, the priors on ω and θ are non-informative.

$$(\omega_{n1}, \omega_{n2}, \cdots, \omega_{nK}) \sim Dir\left(\frac{1}{K}, \frac{1}{K}, \cdots, \frac{1}{K}\right) \tag{3}$$

$$(\theta_{k1}, \theta_{k2}, \cdots, \theta_{kG}) \sim Dir\left(\frac{1}{KG}, \frac{1}{KG}, \cdots, \frac{1}{KG}\right) \tag{4}$$

Now in *classtpx*, we assume that for some samples, the class labels or cluster labels are known. This information is used to either drive the θ matrix or the ω matrix. For instance, in the Leng et al 2015 data, the three classes may be considered to be the G1, S and G2 phases. There are three methods we propose

- **omega.fix**: We fix the ω vector for the samples for which the class labels are known. For instance, in the Leng et al 2015 data, for a classtpx model with K=3 representing the clusters due to G1, S and G2 phases, if the sample n comes from the G1 phase, we fix $\omega_{n.} = (1, 0, 0)$. Similarly, if the sample comes from S or G2 phase, we fix $\omega_{n.}$ to be $(0, 1, 0)$ and $(0, 0, 1)$ respectively. For the cells corresponding to H1 phase, the ω vector is not known and estimated from the data. In mathematical terms, we can write the model for ω as follows

$$\begin{aligned} &(\omega_{n1}, \omega_{n2}, \cdots, \omega_{nK}) = e_k \qquad if \quad class(n) = k \\ &(\omega_{n1}, \omega_{n2}, \cdots, \omega_{nK}) \sim Dir\left(\frac{1}{K}, \frac{1}{K}, \cdots, \frac{1}{K}\right) \qquad if \quad class(n) = NULL \end{aligned}$$

  where $class(n)$ represents the class label of the sample (it is NULL if the class label is not known). e_k is the vector with 1 at position k and 0 at all other positions of the vector. We then perform updates on the ω on the NULL class label samples and the θ matrix and the updating scheme is similar to the `topics()` as in *maptpx* package due to Matt Taddy.
- **theta.prior**: For each class label k, we pool all samples n with $class(n) = k$, and then normalize the counts data to determine the prior θ matrix.

$$
\begin{aligned}
\theta_{kg} &= \frac{\sum_{n:class(n)=k} c_{ng}}{\sum_{g} \sum_{n:class(n)=k} c_{ng}} \quad \forall g, \quad if \quad card\{n : class(n) = k\} \neq 0 \\
(\theta_{k1}, \theta_{k2}, \cdots, \theta_{kG}) &\sim Dir\left(\frac{1}{KG}, \frac{1}{KG}, \cdots, \frac{1}{KG}\right) \quad if \quad card\{n : class(n) = k\} = 0
\end{aligned} \tag{5}
$$

One can also apply adaptive shrinkage on the $\theta_{k.}$ values. In that case, define

$$
\beta_{kg} = \frac{\sum_{n:class(n)=k} c_{ng}}{N_k} - \frac{\sum_{g} \sum_{n:class(n)=k} c_{ng}}{N} \quad card\{n : class(n) = k\} = N_k \tag{6}
$$

We assume

$$
s_{kg} = \frac{1}{N_k(N_k - 1)} \sum_{class(n)=k} \left(c_{ng} - \frac{\sum_{n:class(n)=k} c_{ng}}{N_k} \right)^2 \quad card\{n : class(n) = k\} = N_k \tag{7}
$$

Then we perform `ash` on the vector (β_{kg}, s_{kg}) over all genes g for each k and then obtain the posterior mean of β, say $\beta_{post}(kg)$.
We then fix the θ values as

$$
\theta^{\star}_{kg} = \frac{\sum_{g} \sum_{n:class(n)=k} c_{ng}}{N} + \beta_{post}(kg) \tag{8}
$$

So, in other words we assume

$$
\begin{aligned}
\theta_{kg} &= \theta^{\star}_{kg} \quad \forall g, \quad if \quad card\{n : class(n) = k\} \neq 0 \\
(\theta_{k1}, \theta_{k2}, \cdots, \theta_{kG}) &\sim Dir\left(\frac{1}{KG}, \frac{1}{KG}, \cdots, \frac{1}{KG}\right) \quad if \quad card\{n : class(n) = k\} = 0
\end{aligned} \tag{9}
$$

These $\theta_{k.}$ as in Eqn (5) or Eqn (9) (depending on whether we use shrinkage or not) are then input into the GoM model framework as prior cluster probability vectors and we update them based on the data to find the posterior estimates at each stage of iteration (the updating scheme similar to the `topics()` as in *maptpx* package due to Matt Taddy).

- **theta.fix**: In this method, instead of setting $\theta_{k.}$ matrix for k with $N_k \neq 0$ as prior, we fix them at these values and do not update them during the iterative steps. The only updates correspond to those k for which $N_k = 0$, and the updating scheme is similar to the `topics()` as in *maptpx* package due to Matt Taddy.

5 Results

We first cite the example of the Leng et al data and fit `classtpx()` model on the data.

5.1 Leng et al (2015)

We fix the class labels and the sample indices as follows.

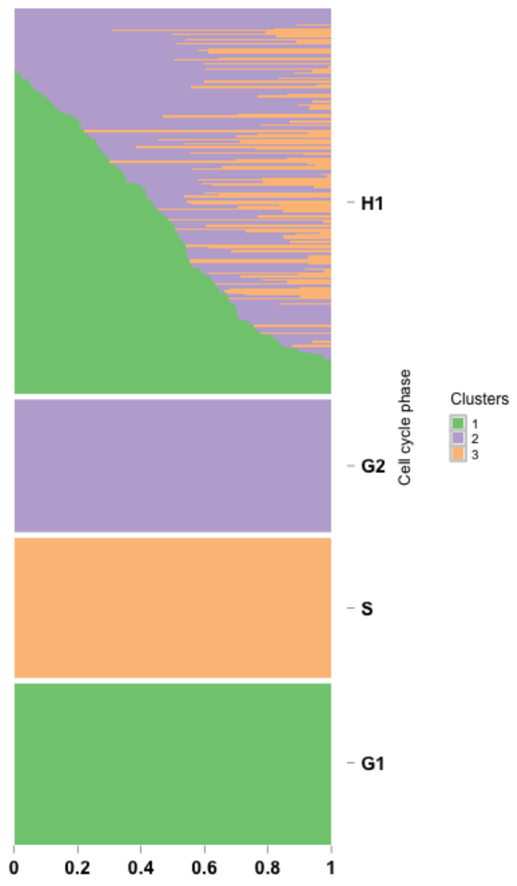
```
index_1 <- which(leng_cell_state=="G1");
index_2 <- which(leng_cell_state=="S");
index_3 <- which(leng_cell_state=="G2");

known_samples <- c(index_1, index_2, index_3);
class_labs <- c(rep("G1", length(index_1)),
                rep("S", length(index_2)),
                rep("G2", length(index_3)));
```

Then we can perform `classtpx()` model for `omega.fix()` method as follows

```
Topic_clus <- classtpx::class_topics(
    leng_data,
    K=3,
    known_samples = known_samples,
    class_labs = class_labs,
    method="omega.fix",
    tol=0.01)

save(Topic_clus, file="../data/leng_topic_fit_3_classtpx_omega_fix.rda")
```

We can perform Structure plot visualization of the results.

```
Topic_clus <- get(load(file="../data/leng_topic_fit_3_classtpx_omega_fix.rda"))

omega <- Topic_clus$omega;

annotation <- data.frame(
  sample_id = paste0("X", c(1:NROW(omega))),
  tissue_label = factor(leng_cell_state,
                        levels = c("G1", "S", "G2", "H1") ) )


rownames(omega) <- annotation$sample_id;


CountClust::StructureGGplot(omega = omega,
                annotation = annotation,
                palette = RColorBrewer::brewer.pal(8, "Accent"),
                yaxis_label = "Cell cycle 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"))
```

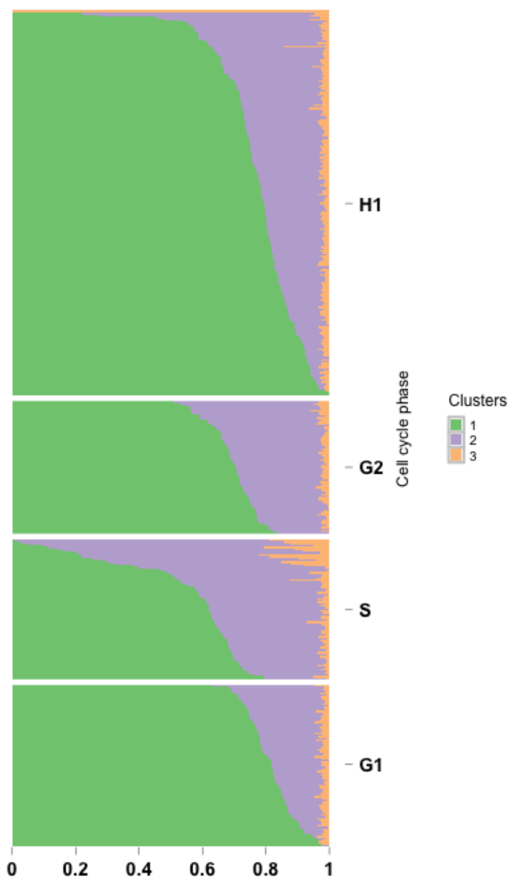


We now perform `classtpx()` model for `theta.prior()` method.

```
Topic_clus <- classtpx::class_topics(
    leng_data,
    K=3,
    known_samples = known_samples,
    class_labs = class_labs,
    method="theta.prior",
    tol=0.01,
    shrink=TRUE)

save(Topic_clus, file="../data/leng_topic_fit_3_classtpx_theta_prior.rda")
```

We can perform Structure plot visualization of the results.

```r
Topic_clus <- get(load(file="../data/leng_topic_fit_3_classtpx_theta_prior.rda"))

omega <- Topic_clus$omega;

annotation <- data.frame(
  sample_id = paste0("X", c(1:NROW(omega))),
  tissue_label = factor(leng_cell_state,
                        levels = c("G1", "S", "G2", "H1") ) )


rownames(omega) <- annotation$sample_id;


CountClust::StructureGGplot(omega = omega,
                annotation = annotation,
                palette = RColorBrewer::brewer.pal(8, "Accent"),
                yaxis_label = "Cell cycle 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"))
```

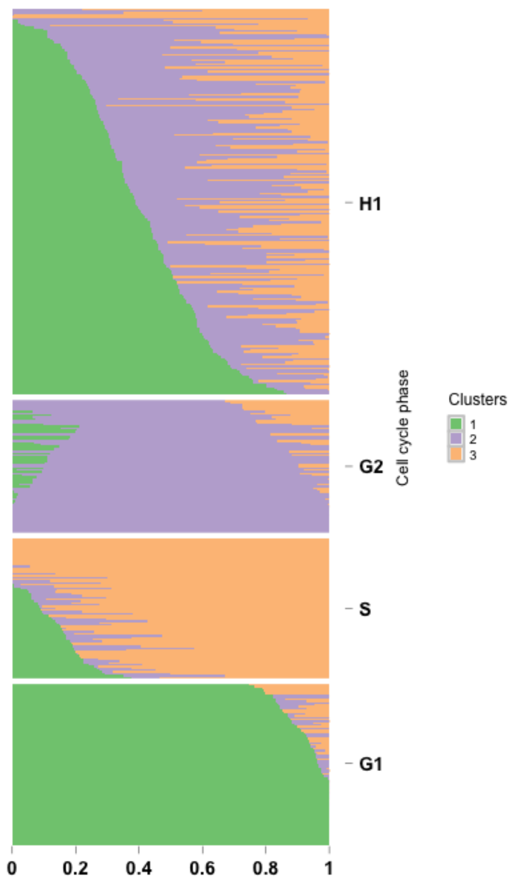


Finally we apply the `theta.fix()` method.

```
Topic_clus <- classtpx::class_topics(
    leng_data,
    K=3,
    known_samples = known_samples,
    class_labs = class_labs,
    method="theta.fix",
    tol=0.01,
    shrink=FALSE)

save(Topic_clus, file="../data/leng_topic_fit_3_classtpx_theta_fix.rda")
```

We can perform Structure plot visualization of the results.

```
Topic_clus <- get(load(file="../data/leng_topic_fit_3_classtpx_theta_fix.rda"))

omega <- Topic_clus$omega;

annotation <- data.frame(
  sample_id = paste0("X", c(1:NROW(omega))),
  tissue_label = factor(leng_cell_state,
                        levels = c("G1", "S", "G2", "H1") ) )


rownames(omega) <- annotation$sample_id;


CountClust::StructureGGplot(omega = omega,
                annotation = annotation,
                palette = RColorBrewer::brewer.pal(8, "Accent"),
                yaxis_label = "Cell cycle 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"))
```



5.2 Treutlin et al (2014)

Treutlin et al 2014 sequenced single cell transcriptome data from mouse lung epithelium. The cells were collected at various stages E14.5, E16.5, E18.5 and some adult replicates. We performed both `maptpx()` model and `classtpx` model fitting on this data.

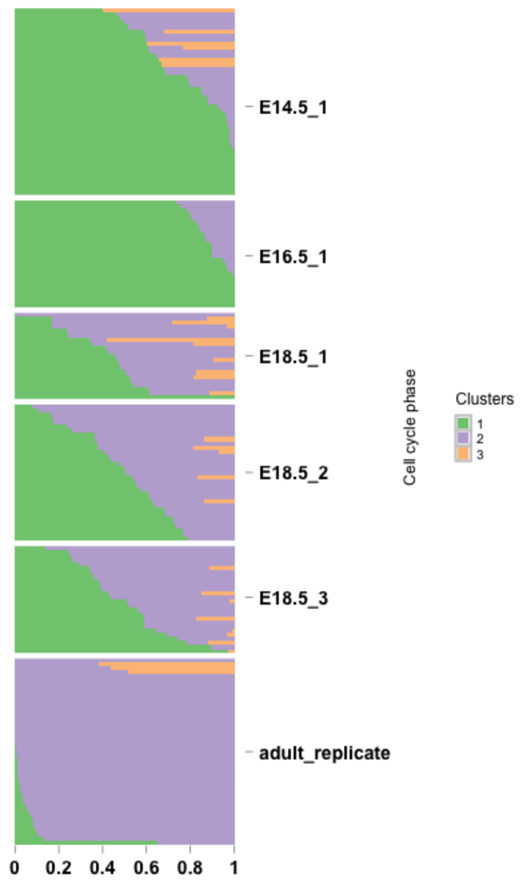
```
devtools::install_github("jhsiao999/singleCellRNASeqMouseTreutleinLung", force=TRUE)
```

```
library(singleCellRNASeqMouseTreutleinLung)
data("MouseTreutleinLung")
leng_gene_names <- Biobase::featureNames(HumanLengESC);

counts_data <- t(Biobase::exprs(MouseTreutleinLung));
pheno_metadata <- pData(MouseTreutleinLung);
table(pheno_metadata[,1])

##
## adult E14.5 E16.5 E18.5
##    46    45    27    83
```

We first apply the maptpx model for $K = 3$.

```
Topic_clus <- maptpx::topics(counts_data, 3, tol=0.1);
save(Topic_clus, file="../data/treutlin_topic_fit_3_maptpx.rda")
```

```
Topic_clus <- get(load(file="../data/treutlin_topic_fit_3_maptpx.rda"))

omega <- Topic_clus$omega;

annotation <- data.frame(
  sample_id = paste0("X", c(1:NROW(omega))),
  tissue_label = factor(paste0(pheno_metadata$cell_type, "_",
                               pheno_metadata$replicate),
                        levels=rev(c("E14.5_1", "E16.5_1",
                                     "E18.5_1","E18.5_2",
                                     "E18.5_3","adult_replicate"))
))

rownames(omega) <- annotation$sample_id;


CountClust::StructureGGplot(omega = omega,
                annotation = annotation,
                palette = RColorBrewer::brewer.pal(8, "Accent"),
                yaxis_label = "Cell cycle 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"))
```

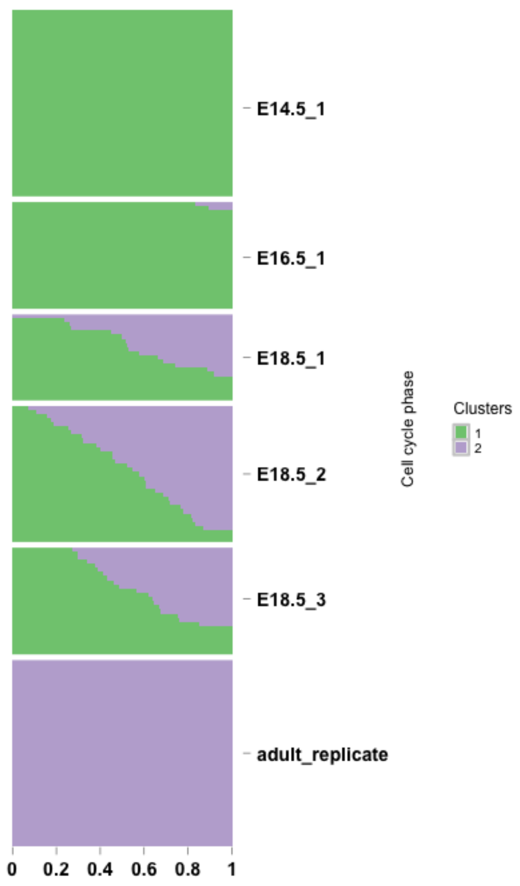


We next performed classtpx model for K=2 with `omega.fix()` method. We chose E14.5 as one group and adult replicates as another group in defining class labels and assumed we do not have class label information for E16.5 and E18.5 phases.

```
known_samples <- c(which(pheno_metadata$cell_type=="E14.5"),
                         which(pheno_metadata$cell_type=="adult"));
class_labs <- c(rep(1, length(which(pheno_metadata$cell_type=="E14.5"))),
                     rep(2,length(which(pheno_metadata$cell_type=="adult"))));
```

```
Topic_clus <- classtpx::class_topics(
    counts_data,
    K=2,
    known_samples = known_samples,
    class_labs = class_labs,
    method="omega.fix",
    tol=0.01,
    shrink=FALSE)

save(Topic_clus, file="../rdas/treutlin_topic_fit_2_classtpx_omega_fix.rda")
```

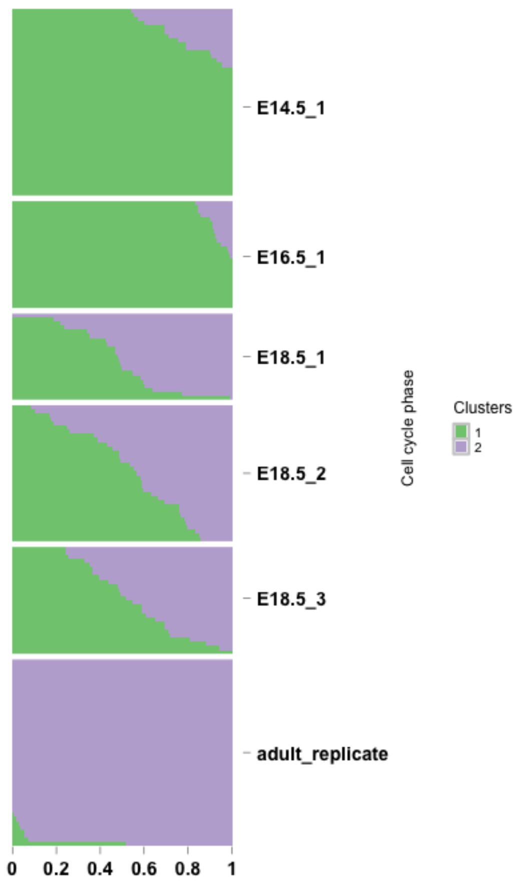
```
Topic_clus <- get(load(file="../data/treutlin_topic_fit_2_classtpx_omega_fix.rda"))

omega <- Topic_clus$omega;

annotation <- data.frame(
  sample_id = paste0("X", c(1:NROW(omega))),
  tissue_label = factor(paste0(pheno_metadata$cell_type, "_",
                               pheno_metadata$replicate),
                        levels=rev(c("E14.5_1", "E16.5_1",
                                     "E18.5_1","E18.5_2",
                                     "E18.5_3","adult_replicate"))
))

rownames(omega) <- annotation$sample_id;


CountClust::StructureGGplot(omega = omega,
                annotation = annotation,
                palette = RColorBrewer::brewer.pal(8, "Accent"),
                yaxis_label = "Cell cycle 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 perform the method with `theta.fix()` for K=2 and K=3.

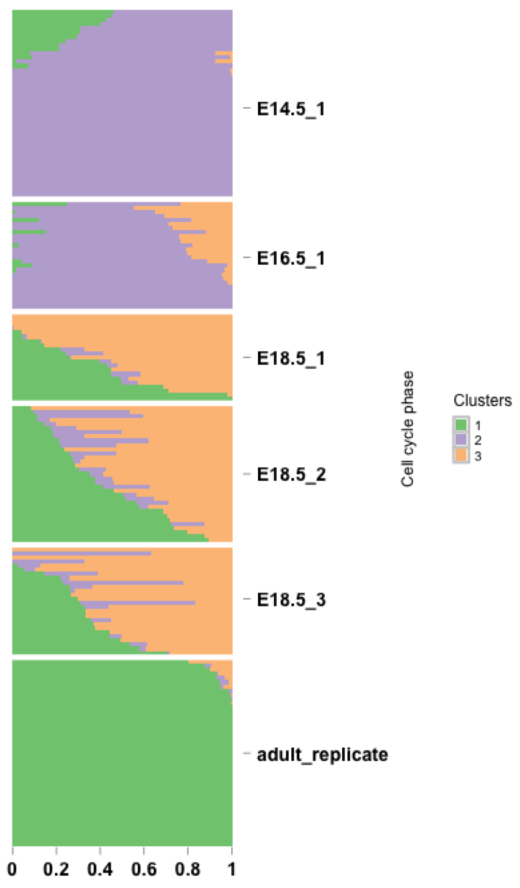
```
Topic_clus <- classtpx::class_topics(
    counts_data,
    K=2,
    known_samples = known_samples,
    class_labs = class_labs,
    method="theta.fix",
    tol=0.01,
    shrink=FALSE)

save(Topic_clus, file="../rdas/treutlin_topic_fit_2_classtpx_theta_fix.rda")
```

```
Topic_clus <- get(load(file="../data/treutlin_topic_fit_2_classtpx_theta_fix.rda"))

omega <- Topic_clus$omega;

annotation <- data.frame(
  sample_id = paste0("X", c(1:NROW(omega))),
  tissue_label = factor(paste0(pheno_metadata$cell_type, "_",
                                   pheno_metadata$replicate),
                           levels=rev(c("E14.5_1", "E16.5_1",
                                         "E18.5_1","E18.5_2",
                                         "E18.5_3","adult_replicate"))
))

rownames(omega) <- annotation$sample_id;


CountClust::StructureGGplot(omega = omega,
                  annotation = annotation,
                  palette = RColorBrewer::brewer.pal(8, "Accent"),
                  yaxis_label = "Cell cycle 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"))
```



```
Topic_clus <- classtpx::class_topics(
    counts_data,
    K=2,
    known_samples = known_samples,
    class_labs = class_labs,
    method="theta.fix",
    tol=0.01,
    shrink=FALSE)

save(Topic_clus, file="../data/treutlin_topic_fit_3_classtpx_theta_fix.rda")
```

```
Topic_clus <- get(load(file="../data/treutlin_topic_fit_3_classtpx_theta_fix.rda"))

omega <- Topic_clus$omega;

annotation <- data.frame(
  sample_id = paste0("X", c(1:NROW(omega))),
  tissue_label = factor(paste0(pheno_metadata$cell_type, "_",
                               pheno_metadata$replicate),
                        levels=rev(c("E14.5_1", "E16.5_1",
                                     "E18.5_1","E18.5_2",
                                     "E18.5_3","adult_replicate"))
))

rownames(omega) <- annotation$sample_id;


CountClust::StructureGGplot(omega = omega,
                annotation = annotation,
                palette = RColorBrewer::brewer.pal(8, "Accent"),
                yaxis_label = "Cell cycle 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"))
```



5.3 Scialdone et al (2015)

Scialdone et al sequenced bulk-FACS sorted cells from phases G1, S and G2M phases. Also they sequenced a number of liver cells and they showed from PCA analysis that most of these liver cells seemed close to the G1 phase data. This was intuitive as most of these differentiated liver cells are in G1 phase. We perform classtpx here on the liver cells taking the bulk-RNA FACS sorted data from G1, S and G2M phases to represent three classes or clusters.

```
devtools::install_github("jhsiao999/singleCellRNASeqMouseScialdoneLiver",
                         force=TRUE);
```

```
library(singleCellRNASeqMouseScialdoneLiver)
data("MouseScialdoneLiver")
liver_data <- t(Biobase::exprs(MouseScialdoneLiver));
```

Now we install the bulk FACS sorted cell cycle data from Scialdone et al (2015).

```
devtools::install_github("jhsiao999/bulkRNASeqMouseScialdoneESC", force=TRUE);
```

```
library(bulkRNASeqMouseScialdoneESC)
data("MouseScialdoneESC")
facs_data <- t(Biobase::exprs(MouseScialdoneESC));
facs_pheno_metadata <- pData(MouseScialdoneESC);
```

```
indices_intersect <- intersect(featureNames(MouseScialdoneESC),
                               featureNames(MouseScialdoneLiver))

matched_liver <-  match(indices_intersect, featureNames(MouseScialdoneLiver))
liver_data_mod <- liver_data[,matched_liver];
colnames(liver_data_mod) <- indices_intersect;

matched_facs <- match(indices_intersect, featureNames(MouseScialdoneESC))
facs_data_mod <- facs_data[,matched_facs];
colnames(facs_data_mod) <-  indices_intersect;

counts <- rbind(facs_data_mod, liver_data_mod);
```

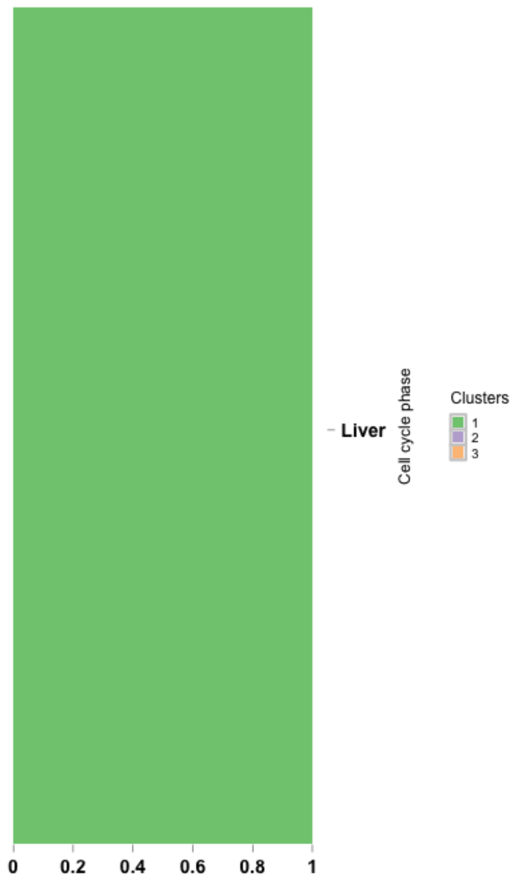
We now fit the classtpx model with the first three samples (bulk FACS sorted data representing cell cycle classes G1, S and G2M respectively) assumed to be from known classes.

```
known_samples <- 1:3;
class_labs <-  1:3;
```

```
Topic_clus <- classtpx::class_topics(
    counts,
    K=3,
    known_samples = known_samples,
    class_labs = class_labs,
    method="omega.fix",
    tol=0.01,
    shrink=FALSE)

save(Topic_clus, file="../data/scialdone_topic_fit_3_classtpx_omega_fix.rda")
```

```
Topic_clus <- get(load("../data/scialdone_topic_fit_3_classtpx_omega_fix.rda"))
omega <- Topic_clus$omega[-(1:3),];

annotation <- data.frame(
  sample_id = paste0("X", c(1:NROW(omega))),
  tissue_label = factor(as.character(rep("Liver",96)))
)

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

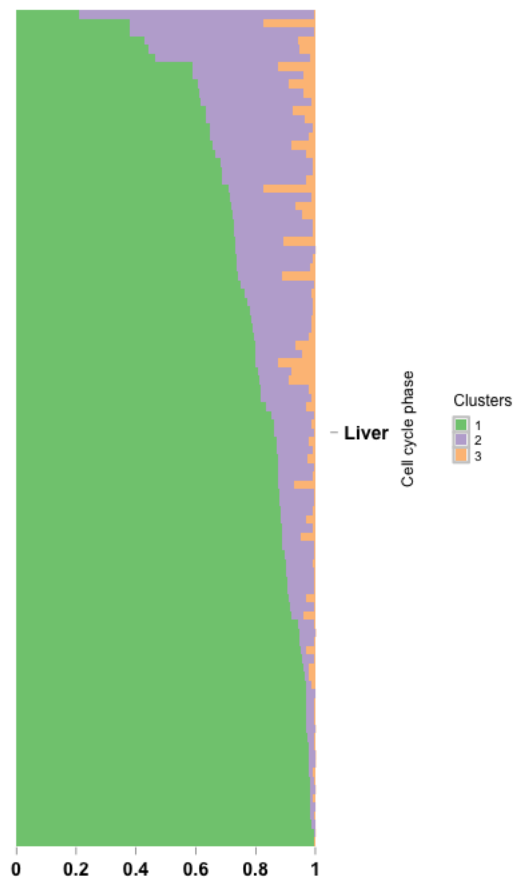
```
CountClust::StructureGGplot(omega = omega,
                annotation = annotation,
                palette = RColorBrewer::brewer.pal(8, "Accent"),
                yaxis_label = "Cell cycle phase",
                order_sample = TRUE,
                axis_tick = list(axis_ticks_length = .1,
   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



Note that all the cells seem to be represented by the first class which corresponds to the G1 phase. We perform the same analysis with `theta.fix()` method.

```
Topic_clus <- classtpx::class_topics(
    counts,
    K=3,
    known_samples = known_samples,
    class_labs = class_labs,
    method="theta.fix",
    tol=0.01,
    shrink=FALSE)

save(Topic_clus, file="../data/scialdone_topic_fit_3_classtpx_theta_fix.rda")
```

```
Topic_clus <- get(load("../data/scialdone_topic_fit_3_classtpx_theta_fix.rda"))
omega <- Topic_clus$omega[-(1:3),];

annotation <- data.frame(
  sample_id = paste0("X", c(1:NROW(omega))),
  tissue_label = factor(as.character(rep("Liver",96)))
)

rownames(omega) <- annotation$sample_id;


CountClust::StructureGGplot(omega = omega,
                annotation = annotation,
                palette = RColorBrewer::brewer.pal(8, "Accent"),
                yaxis_label = "Cell cycle 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"))
```



Here also maximum representativeness is due to the cluster from G1 phase.

5.4 Buettner et al 2015 + Leng et al 2015

Buettner et al 2015 collected single cell RNA-seq FACS sorted data from cell cycle phases G1, G2 and S from mouse. Leng et al 2015 collected FACS sorted data from G1, G2 and S for humans. The idea was to see if we can use the FACS sorted mouse data coming from three cell cycle phases G1, G2 and S, to learn about human data and predict the cell cycle phases of the human data correctly.

The one barrier to doing so is that the genes in humans and those in mouse are different. But there are many ortholog genes which are common between the two species and we extract such genes out using BioMart.

First we install and load the mouse single cell RNA-seq data due to Buettner et al 2015.

```
devtools::install_github("jhsiao999/singleCellRNASeqMouseBuettnerESC", force=TRUE)
```

```
library(singleCellRNASeqMouseBuettnerESC)
data("MouseBuettnerESC")
buettner_gene_names <- Biobase::featureNames(MouseBuettnerESC);
```

```
buettner_data <- Biobase::exprs(MouseBuettnerESC);
buettner_metadata <- Biobase::pData(MouseBuettnerESC)
buettner_cell_state <- buettner_metadata$cell_cycle;

table(buettner_cell_state)
```

```
## buettner_cell_state
##  G1 G2M   S
##  96  96  96
```

We now extract the ortholog genes common between the Leng et al 2015 data and the Buettner et al 2015 data.

```
library(biomaRt)
human = useMart("ensembl", dataset = "hsapiens_gene_ensembl");
attributes = c("ensembl_gene_id","mmusculus_homolog_ensembl_gene",
               "mmusculus_homolog_perc_id_r1")
attributes=c(attributes,"mmusculus_homolog_orthology_type",
"mmusculus_homolog_subtype", "mmusculus_homolog_perc_id")
orth.mouse.human = getBM(attributes,
filters="with_homolog_mmus",values=TRUE, mart = human,
uniqueRows=TRUE)
```

We determine those ortholog human genes obtained above which have valid HGNC symbol which has been used to report feature names in the Leng et al 2015 data

```
filter_indices <- match(buettner_gene_names, orth.mouse.human[,2]);
filter_indices <- filter_indices[!is.na(filter_indices)];
ortholog_buettner_mouse_ids <- orth.mouse.human[filter_indices,2];
ortholog_buettner_human_ids <- orth.mouse.human[filter_indices,1];

human = useMart("ensembl", dataset = "hsapiens_gene_ensembl")
mySymbols <- ortholog_buettner_human_ids; # mySymbols is a vector of MGI symbols.
gene_list_human <- getBM( attributes=c("ensembl_gene_id", "hgnc_symbol") , filters=
"ensembl_gene_id", values =mySymbols ,mart=human)

gene_names_human <- gene_list_human[,2];
```

We filter the Leng et al data

```
matched_human_gene_indices_leng <- match(gene_names_human, leng_gene_names)
matched_human_gene_indices_leng <-
  matched_human_gene_indices_leng[!is.na(matched_human_gene_indices_leng)];



matched_genes_leng <- leng_gene_names[matched_human_gene_indices_leng];
```

```r
filter_indices <- match(matched_genes_leng, leng_gene_names);

leng_filtered_data <- leng_data[,filter_indices];
```

Next we filter the Buettner et al data

```r
filter_indices <- match(gene_names_human, leng_gene_names);
gene_list_human_filtered <- gene_list_human[which(!is.na(filter_indices)),];
gene_names_human_filtered <- gene_list_human_filtered[,1];
human_ids_buettner <-
  ortholog_buettner_human_ids[match(gene_names_human_filtered,
                                    ortholog_buettner_human_ids)];
mouse_ortholog_ids_buettner <-
  ortholog_buettner_mouse_ids[match(gene_names_human_filtered,
                                    ortholog_buettner_human_ids)];

filter_genes_buettner <- match(mouse_ortholog_ids_buettner, buettner_gene_names);

buettner_filtered_data <- t(buettner_data[filter_genes_buettner,]);
```

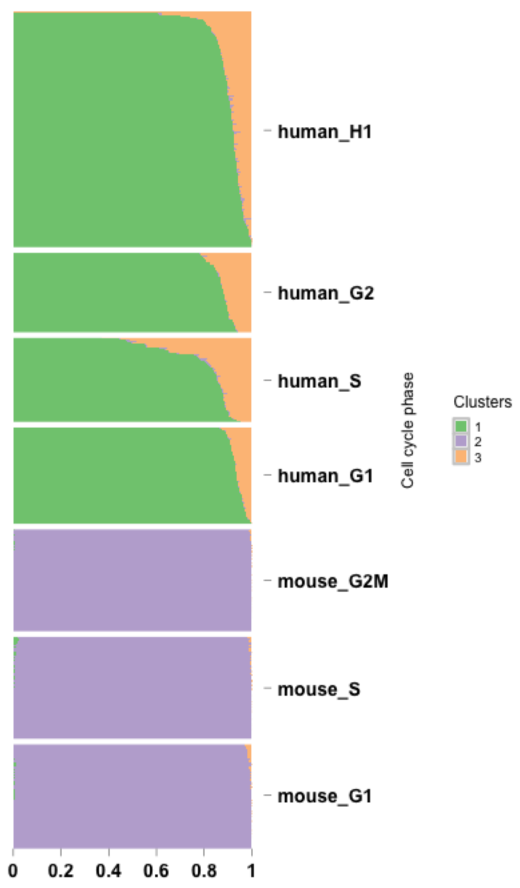
We pool the filtered data from Leng et al and Buettner et al containing expression patterns of ortholog matched genes.

```r
pooled_data <- rbind(buettner_filtered_data, leng_filtered_data);
```

We first apply the `topics()` of the `maptpx()` package to detrmine the unsupervised clustering patterns.

```r
Topic_clus <- maptpx::topics(
    pooled_data,
    K=3,
    tol=0.1
    )

save(Topic_clus, file="../data/leng_topic_fit_3_maptpx_buettner.rda")
```

```r
Topic_clus <- get(load(file="../data/leng_topic_fit_3_maptpx_buettner.rda"))

omega <- Topic_clus$omega;

leng_cell_state_human <- paste0("human", "_", leng_cell_state);
buettner_cell_state_mouse <- paste0("mouse","_", buettner_cell_state);

annotation <- data.frame(
  sample_id = paste0("X", c(1:NROW(omega))),
  tissue_label = factor(c(as.character(buettner_cell_state_mouse),
                          as.character(leng_cell_state_human)),
                        levels = c("mouse_G1", "mouse_S", "mouse_G2M",
                                        "human_G1","human_S",
                                         "human_G2", "human_H1") ) )


rownames(omega) <- annotation$sample_id;


CountClust::StructureGGplot(omega = omega,
                annotation = annotation,
                palette = RColorBrewer::brewer.pal(8, "Accent"),
                yaxis_label = "Cell cycle 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"))
```



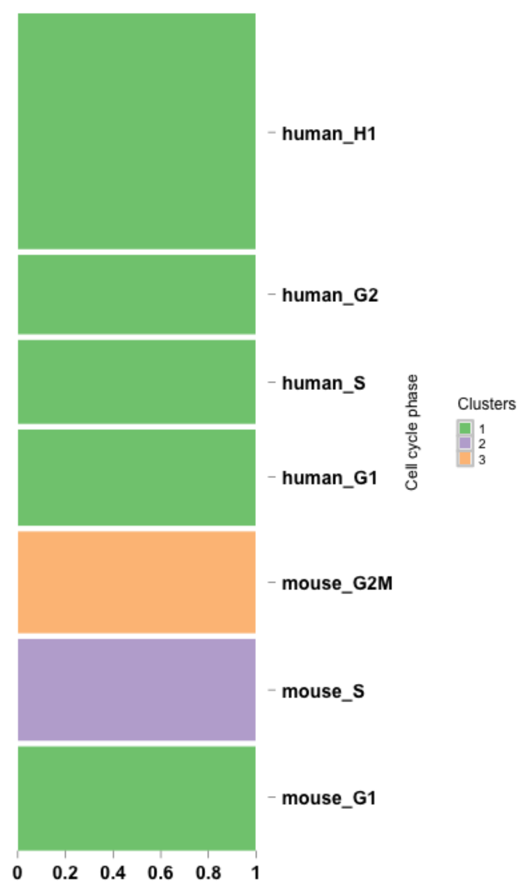
Note that when the clustering is performed over mouse and human ortholog genes together, it seems to pick up those genes that separate the two species rather than finding out cell-cycle related structure. This is again more intuitive I guess as the inter species differences are way stronger than the cell cycle effect.

We next performed classtpx model by using the mouse data as training sample and we fit $K = 3$ model with both `omega.fix()` and `theta.fix()` methods.

```
known_samples <- 1:288;
class_labs <- c(rep(1,96), rep(2,96), rep(3,96));
```

```
Topic_clus <- classtpx::class_topics(
    counts,
    K=3,
    known_samples = known_samples,
    class_labs = class_labs,
    method="omega.fix",
    tol=0.01,
    shrink=FALSE)

save(Topic_clus, file="../data/leng_topic_fit_classtpx_omega_fix_buettner.rda")
```

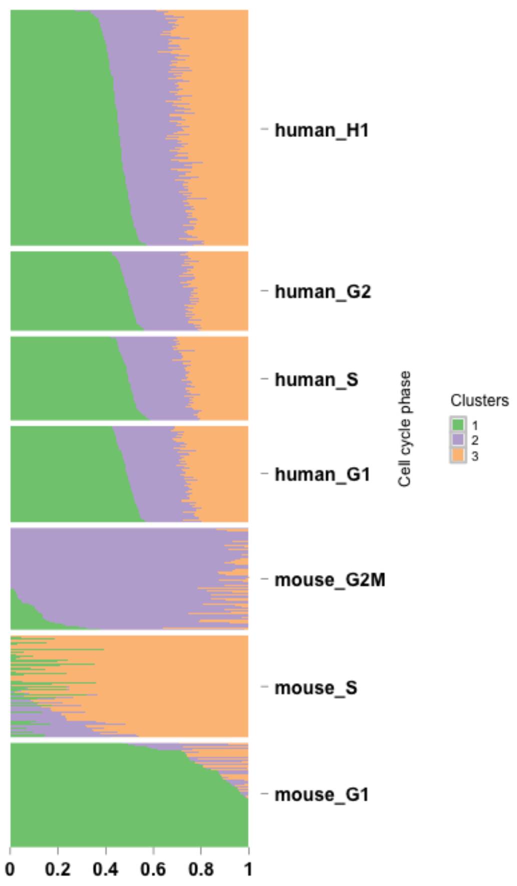
```
Topic_clus <- get(load("../data/leng_topic_fit_classtpx_omega_fix_buettner.rda"))
omega <- Topic_clus$omega;

leng_cell_state_human <- paste0("human", "_", leng_cell_state);
buettner_cell_state_mouse <- paste0("mouse","_", buettner_cell_state);

annotation <- data.frame(
  sample_id = paste0("X", c(1:NROW(omega))),
  tissue_label = factor(c(as.character(buettner_cell_state_mouse),
                          as.character(leng_cell_state_human)),
                        levels = c("mouse_G1", "mouse_S", "mouse_G2M",
                                   "human_G1","human_S",
                                    "human_G2", "human_H1") ) )



rownames(omega) <- annotation$sample_id;



CountClust::StructureGGplot(omega = omega,
                annotation = annotation,
                palette = RColorBrewer::brewer.pal(8, "Accent"),
                yaxis_label = "Cell cycle 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"))
```



```
Topic_clus <- classtpx::class_topics(
    pooled_data,
    K=3,
    known_samples = known_samples,
    class_labs = class_labs,
    method="theta.fix",
    tol=0.01,
    shrink=FALSE)

save(Topic_clus, file="../data/leng_topic_fit_classtpx_theta_fix_buettner.rda")
```

```
Topic_clus <- get(load("../data/leng_topic_fit_classtpx_theta_fix_buettner.rda"))
omega <- Topic_clus$omega;

leng_cell_state_human <- paste0("human", "_", leng_cell_state);
```

```
buettner_cell_state_mouse <- paste0("mouse","_", buettner_cell_state);

annotation <- data.frame(
  sample_id = paste0("X", c(1:NROW(omega))),
  tissue_label = factor(c(as.character(buettner_cell_state_mouse),
                          as.character(leng_cell_state_human)),
                        levels = c("mouse_G1", "mouse_S", "mouse_G2M",
                                        "human_G1","human_S",
                                         "human_G2", "human_H1") ) )


rownames(omega) <- annotation$sample_id;


CountClust::StructureGGplot(omega = omega,
                annotation = annotation,
                palette = RColorBrewer::brewer.pal(8, "Accent"),
                yaxis_label = "Cell cycle 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"))
```



The above analysis (especially for the `theta.fix` approach) shows that the cluster patterns obtained from cell cycle phases of mouse does not quite explain the patterns in the human data. One reason could be because we have pooled all the ortholog genes in this analysis and not just the cell-cycle related genes. It could also mean the orthologs of cell-cycle genes for mouse may not be cell-cycle related genes for humans.

6 Discussion

- Since the number of features (genes) is usually way larger than the number of samples in RNA-seq data, it is important to protect the analysis from noisy genes which are not informative biologically but incorporate a lot of variation that can compromise the results. We use the adaptive shrinkage method (check Methods and Materials) through `shrink=TRUE` to effectively reweigh the variables based on their within-class variation. The shrinkage method adopted may be improved by using FASH due to Mengyin Lu https://github.com/mengyin/ashlar-fash. Her model shrinks the F-statistics derived from the following linear model (which may be applied on voom transformed data)

$$Y_{ng} = \mu_g + \beta_{cl(n),g} + e_{ng} \qquad \beta_{cl(n),g} \sim N(0, \sigma^2) \qquad e_{ng} \sim N(0, s^2) \tag{10}$$

However I am not sure how I can go from shrunk F-statistics in this model to shrunk estimates of β values, which I think is what I may need ultimately to obtain refined estimates of θ.

- We may want to indicate in the Structure plot visualization, which samples have known class labels and what that class label is, compared to the ones for which we have no information about class labels. This would be particularly handy in case of `theta.fix()` or `theta.prior` models where it is not clear just from viewing the Structure plot, which of the samples have known cluster memberships.
- So far the attempt has been to take data from multiple experiments across labs and then check how far the knowledge of patterns from experiment in one lab can explain the patterns in the other. But there seems to be differences probably resulting from lab effects or batch or plate effects (cases in point: Buettner + Scialdone 2015 data analysis, Deng et al 2014 + Blakeley et al 2015 data analysis).
- For additional simulated data and real data applications of the *classtpx* package, please check *Cell classification using classtpx* section in http://jhsiao999.github.io/singleCell-method/ and for codes and source files, please check: https://github.com/kkdey/classtpx.

7 Session Info

```
sessionInfo()

## R version 3.2.4 (2016-03-10)
## Platform: x86_64-apple-darwin13.4.0 (64-bit)
## Running under: OS X 10.10.5 (Yosemite)
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] parallel  stats     graphics  grDevices utils     datasets  methods
## [8] base
##
## other attached packages:
##  [1] biomaRt_2.26.1
##  [2] singleCellRNASeqMouseBuettnerESC_0.99.0
##  [3] bulkRNASeqMouseScialdoneESC_0.99.0
##  [4] singleCellRNASeqMouseScialdoneLiver_0.99.0
##  [5] singleCellRNASeqMouseTreutleinLung_0.99.0
##  [6] singleCellRNASeqHumanLengESC_0.99.0
##  [7] Biobase_2.30.0
##  [8] BiocGenerics_0.16.1
##  [9] classtpx_0.0.1
## [10] slam_0.1-32
## [11] knitr_1.12.3
##
```

```
## loaded via a namespace (and not attached):
##  [1] Rcpp_0.12.3          formatR_1.2.1        RColorBrewer_1.1-2
##  [4] plyr_1.8.3           highr_0.5.1          bitops_1.0-6
##  [7] tools_3.2.4          digest_0.6.9         RSQLite_1.0.0
## [10] evaluate_0.8         gtable_0.2.0         nlme_3.1-125
## [13] lattice_0.20-33      mgcv_1.8-12          Matrix_1.2-4
## [16] DBI_0.3.1            stringr_1.0.0        cluster_2.0.3
## [19] IRanges_2.4.6        S4Vectors_0.8.7      gtools_3.5.0
## [22] CountClust_0.99.3    stats4_3.2.4         grid_3.2.4
## [25] nnet_7.3-12          cowplot_0.6.1        maptpx_1.9-2
## [28] flexmix_2.3-13       AnnotationDbi_1.32.3 XML_3.98-1.3
## [31] limma_3.26.8         ggplot2_2.1.0        reshape2_1.4.1
## [34] magrittr_1.5         scales_0.4.0         modeltools_0.2-21
## [37] MASS_7.3-45          BiocStyle_1.8.0      picante_1.6-2
## [40] permute_0.9-0        colorspace_1.2-6     ape_3.4
## [43] labeling_0.3         stringi_1.0-1        RCurl_1.95-4.7
## [46] munsell_0.4.3        vegan_2.3-4
```