Understanding RNA-seq analysis through an example: voting behavior at the French parliament

Hector Roux de Bézieux

May 2018

Introduction

Single-cell RNA-seq (scRNA-seq) is a very potent biological tool used for many applications. A far from exhaustive list would include identifying new cell types, finding differentially expressed (DE) genes, and discovering lineages among cells. However, the usual framework might seem a little daunting for beginners and, while many well-crafted tutorials exists, they all share the same idea: use a biological dataset as an example. Here, we want to use a dataset that would be more understandable to a broader public to explain the usual steps in scRNA-seq analysis.

Contents

1 Context

Our dataset is the voting record of the French delegates of the 14^{th} legislature, going from May 2012 to May 2017. Delegates are characterized by their names and surnames, their department and circonscription, their political group. For each of the 644 votes (which can be laws, amendments, choosing the prime minister, ...), we also have the voting behavior of each delegate (voted yes, no, abstain or did not took part in the vote) and the reason of the vote.

2 Loading the data and building an ExpressionSet object

The usual scRNA-seq data is a matrix of J genes by n cells. Each cell represent the number of copies of the mRNA of a given gene in a given cell. Here, we instead have a J law by n delegates matrix. Each cell represent the vote of the delegate for that low (-1 if against, 1 is for, 0 if not voting or abstained). We store the data in an *ExpressionSet* object, where the *phenodata* is the information about the delegates (the cells) and the *featuredata* is the information about the votes (the genes).

```
voting_record <- read_csv("data/voting_record.csv")
meta <- voting_record %>% select(circo, dept, name, surname, identifiant,
                                 iden, group, NbVote, NbYes, NbNo, NbAbst)
voting_record <- voting_record %>% select(-one_of(colnames(meta))) %>%
                                   t(.)

# We count abstaining and not voting as the same thing for simplification
voting_record[is.na(voting_record)] <- "NA"
voting_record[voting_record == "For"] <- "1"
voting_record[voting_record == "Against"] <- "-1"
voting_record[voting_record == "Abstain"] <- "0"
voting_record[voting_record == "NotVoting"] <- "0"
voting_record[voting_record == "NA"] <- NA
voting_record <- apply(voting_record, 2, as.numeric)


votes <- read_csv("data/votes.csv")
colnames(voting_record) <- rownames(meta) <- meta$identifiant
rownames(voting_record) <- rownames(votes) <- votes$Number
Assay <- ExpressionSet(assayData = voting_record,
                       phenoData = AnnotatedDataFrame(meta) ,
                       featureData = AnnotatedDataFrame(votes))
```

3 Data Cleaning

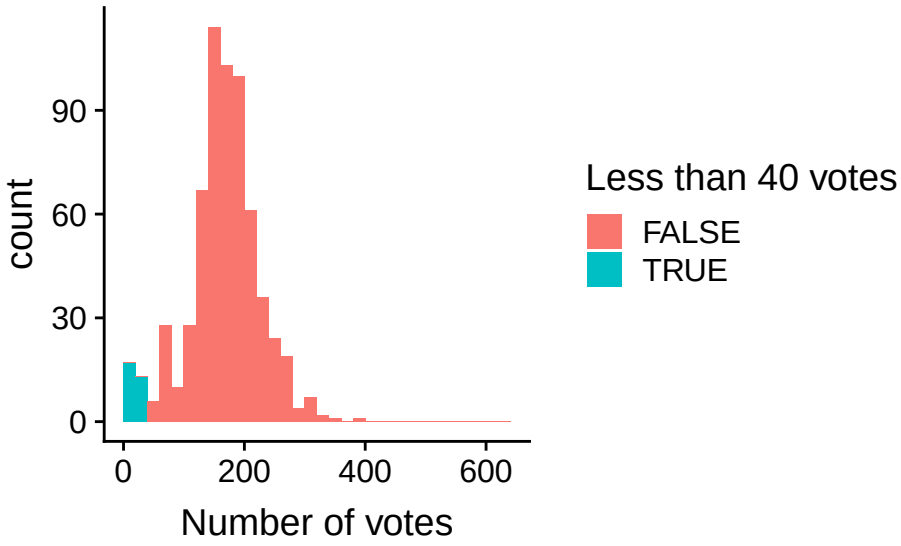
3.1 Filtering the cells (delegate)

In scRNA-seq settings, we want to filter the cells with too few total reads, or reads of too poor quality (that do not match to the genome of reference). Here, we instead filter the delegates with too few votes. This may happen either because a delegate had to leave office before the end of the session, or was appointed to the government, and then its replacement is not around long enough, or because the delegate was really not voting much. A special case is of course the chairman of the Assembly who do not take part in the votes.

We therefore filter delegates whose number of votes is more than 2 SD below the mean (so equal or less to 40).

```
Nb_Votes <- data.frame(Nb_Votes = colSums(!is.na(exprs(Assay))),
                       delegate = rownames(phenoData(Assay)))

ggplot(Nb_Votes, aes(x = Nb_Votes, fill = Nb_Votes <= 40)) +
                 stat_bin(breaks = seq(0, max(Nb_Votes$Nb_Votes), 20)) +
                 labs(x = "Number of votes") +
                 guides(fill = guide_legend(title = "Less than 40 votes"))
```



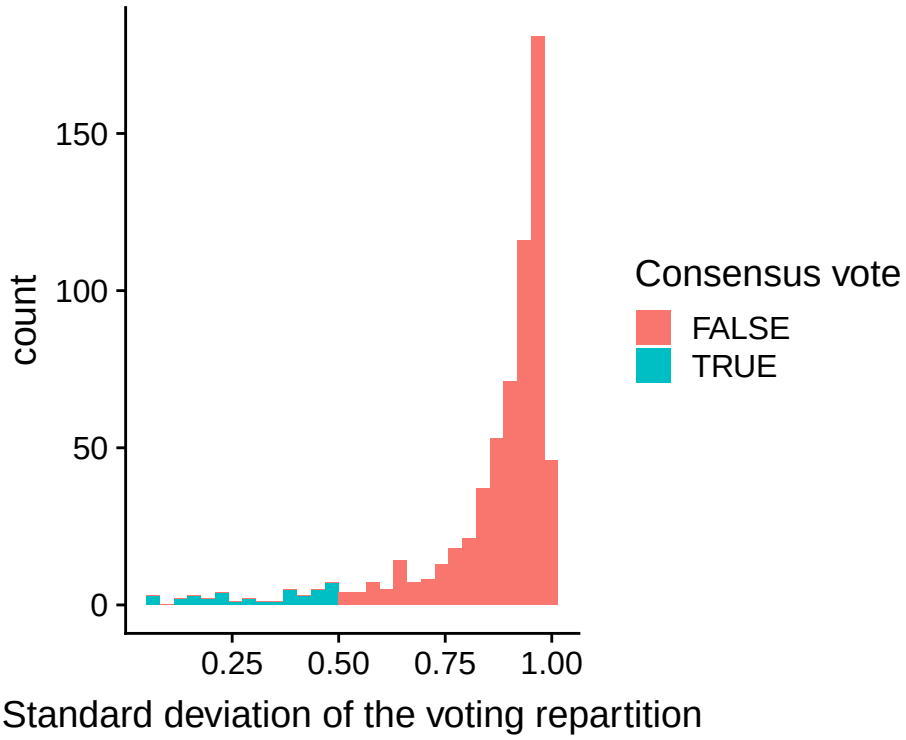

```
Assay <- Assay[, colSums(!is.na(exprs(Assay))) > 40]
```

3.2 Filtering the genes (votes)

In scRNA-seq settings, we want to filter out genes that are expressed in a small number of cells. This is done for two reasons. The first is computational. Most datasets would have dozens of thousands of genes, over dozens or hundreds of cells. Reducing the number of genes to keep speeds up calculations. Secondly, we are usually intersted in a specific tissue or process. Most genes are not being expressed in the samples of interest and just add noise to the analysis. Hence filtration. Simple heuristics in the form of "at least i counts in j cells" are usually quite efficient. The aim is too be quite broad.

In our context, we only have 644 votes so the computational goal does not really hold. But we can filter out votes where an overwehlming majority of delegates voted "yes", or voted "no. We therefore filter out votes with a low standard deviation (< 0.5). This filters out 6.1% of the votes. Note that in scRNA-seq settings, we would usually filter at least 50% of the reads (while keeping more than 90% of the reads).

```
Vote_repartion <- data.frame(sd = rowSds((exprs(Assay)), na.rm = T),
                             votes = rownames(featureData(Assay)))
ggplot(Vote_repartion, aes(x = sd, fill = sd < 0.5)) + geom_histogram() +
  labs(x = "Standard deviation of the voting repartition") +
  guides(fill = guide_legend(title = "Consensus vote"))
```




```
Assay <- Assay[rowSds((exprs(Assay))) >= 0.5, ]
```

Thanks

Special thanks to Vincent Viers for the initial inspiration of this project.

The code used for scraping the data is based on the blog post https://freakonometrics.hypotheses.org/50973.