Mouse genes data analysis

Ousseynou Diallo

19 juillet 2019

Load the data into R

```
dataSet <- read.table("./data.txt", sep="\t")
# head(dataSet)
```

Let's replace the thymys column names by thymus

```
  names(dataSet) <- str_replace(names(dataSet), pattern = 'Thymys',replacement = 'Thymus')
```

Let's find all the organs

To get the right names for the organs we use the `strsplit()` function passing it a `_` to split the names of the columns. It return a list containing for all the column names the different part after the spiting and we use the `sapply()` function on that list and we pass function that will grabe for each column the first element correspoding to the name of the organ.

```
organs <- unique((sapply(strsplit(names(dataSet), "_"), function(x)x[[1]])))
organs
```

```
##  [1] "gene"        "Adrenal"     "Brain"       "Forestomach" "Heart"
##  [6] "Kidney"      "Liver"       "Large"       "Lung"        "Muscle"
## [11] "Ovary"       "Small"       "Spleen"      "Stomach"     "Testis"
## [16] "Thymus"      "Uterus"      "Vescicular"
```

Let's split the dataset into sex specific and non sex specific

To split the data into sex and non-sex specific data set we perform this steps:

- We compute a vector for columns where sex specific columns are TRUE by using the `str_starts()` function passing it the names of the columns in the dataset except the `gene_type` column and and a pattern of string matching the names of sex specific organs

```
sex_specific_organ <- str_starts(names(dataSet[-1]), pattern = "Vescicular|Uterus|Ovary|Testis")
sex_specific_organ
```

```
##  [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [12] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [23] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [34] FALSE FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
## [45] FALSE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE
## [56] FALSE  TRUE  TRUE  TRUE  TRUE
```

- We subset the data using the vector we get above and get all the column that are sex specific and added `TRUE` in `c(TRUE, sex_specific_organ)` to get the gene_type column

```
sex_specific_data <- dataSet[,c(TRUE, sex_specific_organ)]
head(sex_specific_data)
```

```
##            gene_type Ovary_Female_1 Ovary_Female_2 Testis_Male_1
## Gnai3 protein_coding        70.4807        72.9662       31.6262
## Pbsn  protein_coding         0.0000         0.0000        0.7070
## Cdc45 protein_coding         6.8214         7.1763       22.9724
## H19          lincRNA         9.0142        10.4875        1.3826
## Scml2 protein_coding         2.9268         2.0785       19.0269
## Apoh  protein_coding         0.2540         1.9824       37.7230
##       Testis_Male_2 Uterus_Female_1 Uterus_Female_2
## Gnai3       35.2568         72.6195         87.1455
## Pbsn         0.8151          0.0000          0.0000
## Cdc45       24.9343          4.5323         13.7287
## H19          1.3498          2.9863         21.2063
## Scml2       21.2461          1.9175          1.5992
## Apoh        35.5432          0.1215          2.0597
##       Vescicular_Gland_Male_1 Vescicular_Gland_Male_2
## Gnai3                 18.9859                37.0417
## Pbsn                3869.5200              2742.8800
## Cdc45                  0.6882                 2.1766
## H19                    2.4705                 5.0896
## Scml2                  0.5538                 1.3114
## Apoh                   0.6968                 0.2240
```

We subset the data using the the negation of vector`!sex_specific_organ` we get above and get all the column that are sex specific and added `TRUE` in `c(TRUE, sex_specific_organ)` to get the gene_type column

```
non_sex_specific_data <- dataSet[,c(TRUE, !sex_specific_organ)]
```

Get the count of expressed genes for each organs

To get the gene expression of each organ we first write the `get_gene_cout()` function. The function take an organ name and a dataset to return a simple data frame containing in one column the `gene_type` and in another the count of the genes expressed. The second column have the name of the organ passed to the function as column name

- select sample of an organ and the `gene_type` column
- for each row we take every sample check if the value is greate than 0.1 and make the sum of boolean values for each row
- check if the gene is expressed by taking the row sum and see if it is equal to the number of samples for the organ
- use the codition to filter all samples and get only expressed genes
- we use the table function to count the number of `lincRNA` and `coding_protein` in expressed genes
- we get the gene type count as a data frame
- we give the data frame the appropriate names

```
get_gene_count <- function(organ,data){
  all_samples <- data[,str_starts(names(data), pattern = paste("gene", organ, sep = "|"))]
  gene_expressed_row_sum <- rowSums(all_samples[,-1]>0.1)
  gene_expressed_check <- gene_expressed_row_sum == dim(all_samples[,-1])[2]
  gene_expressed <- all_samples[gene_expressed_check,]
  gene_expressed <- table(gene_expressed$gene_type)
  gene_count <- as.data.frame(gene_expressed)
  names(gene_count) <- c("gene_type",organ)
  gene_count
}
```

Get the count of expressed genes for each sex non specific organ passing to `get_gene_count()` function the gene name and the `non_sex_specific_data` data set

```
Adrenal <- get_gene_count("Adrenal",non_sex_specific_data)
Brain <- get_gene_count("Brain", non_sex_specific_data)
Forestomach <- get_gene_count("Forestomach",non_sex_specific_data)
Heart <- get_gene_count("Heart",non_sex_specific_data)
Kidney <- get_gene_count("Kidney",non_sex_specific_data)
Liver <- get_gene_count("Liver",non_sex_specific_data)
Large <- get_gene_count("Large",non_sex_specific_data)
Lung <- get_gene_count("Lung",non_sex_specific_data)
Muscle <- get_gene_count("Muscle",non_sex_specific_data)
Small <- get_gene_count("Small",non_sex_specific_data)
Spleen <- get_gene_count("Spleen",non_sex_specific_data)
Stomach <- get_gene_count("Stomach",non_sex_specific_data)
Thymus <- get_gene_count("Thymus",non_sex_specific_data)
```

bind all the column of non sex specific organs but select `gene_type` column only for the first organ to avoid duplication of the `gene_type` column

```
sex_exp_gene_count <- cbind(Adrenal,Brain[2],Forestomach[2],Heart[2],Kidney[2],Liver[2],Large[2],Lung[2
                        Muscle[2],Small[2],Spleen[2],Stomach[2])
sex_exp_gene_count
```

```
##        gene_type Adrenal Brain Forestomach Heart Kidney Liver Large  Lung
## 1        lincRNA     506   553         422   379    398   271   452   557
## 2 protein_coding   14648 14731       14705 13457  13779 12323 14636 14935
##   Muscle Small Spleen Stomach
## 1    368   364    514     354
## 2  13526 13947  14108   14044
```

Get the count of expressed genes for each sex non specific organ passing to `get_gene_count()` function the gene name and the `sex_specific_data` data set

```
Vescicular <- get_gene_count("Vescicular",sex_specific_data)
Uterus <- get_gene_count("Uterus",sex_specific_data)
Ovary <- get_gene_count("Ovary",sex_specific_data)
Testis <- get_gene_count("Testis",sex_specific_data)
```

Bind all the column sex specific organs but select `gene_type` column only for the first organ to avoid duplication of the `gene_type` column

```
non_sex_exp_gene_count <- cbind(Ovary,Thymus[2],Uterus[2],Vescicular[2])
non_sex_exp_gene_count
```

```
##        gene_type Ovary Thymus Uterus Vescicular
## 1        lincRNA   675    514    556        397
## 2 protein_coding 15893  14473  14875      13908
```

Venn Diagramm for the three organs : *kidney*, *heart* and *Adrenal_gland*

- select sample of an organ and the gene_type column
- for each row we take every sample check if the value is greate than `0.1` and make the sum of boolean values for each row
- check if the gene is expressed by taking the row sum and see if it is equal to the number of samples for the organ
- use the codition to filter all samples and get only expressed genes
- we store the names of the genes in a new column named `gene_names`
- We select the `gene_type` and the `gene_name` column

```
get_expressed_genes <- function(organ){
  all_samples <- dataSet[,str_starts(names(dataSet), pattern = paste("gene", organ, sep = "|"))]
  gene_expressed_row_sum <- rowSums(all_samples[,-1]>0.1)
  gene_expressed_check <- gene_expressed_row_sum == dim(all_samples[,-1])[2]
  gene_expressed <- all_samples[gene_expressed_check,]
  gene_expressed$gene_name <- rownames(gene_expressed)
  gene_expressed <- gene_expressed[c("gene_type", "gene_name")]
  gene_expressed
}
```

get genes expressed in kidney

```
kidney_genes <- get_expressed_genes("Kidney")
head(kidney_genes)
```

```
##            gene_type gene_name
## Gnai3 protein_coding     Gnai3
## Cdc45 protein_coding     Cdc45
## H19          lincRNA       H19
## Scml2 protein_coding     Scml2
## Narf  protein_coding      Narf
## Cav2  protein_coding      Cav2
```

get genes expressed in Heart

```
heart_genes <- get_expressed_genes("Heart")
head(heart_genes)
```

```
##            gene_type gene_name
## Gnai3 protein_coding     Gnai3
## Cdc45 protein_coding     Cdc45
## H19          lincRNA       H19
## Scml2 protein_coding     Scml2
## Narf  protein_coding      Narf
## Cav2  protein_coding      Cav2
```

get genes expressed in Adrenal

```
adrenal_genes <- get_expressed_genes("Adrenal")
head(adrenal_genes)
```

```
##            gene_type gene_name
## Gnai3 protein_coding     Gnai3
## Cdc45 protein_coding     Cdc45
## H19          lincRNA       H19
## Scml2 protein_coding     Scml2
## Apoh  protein_coding      Apoh
## Narf  protein_coding      Narf
```

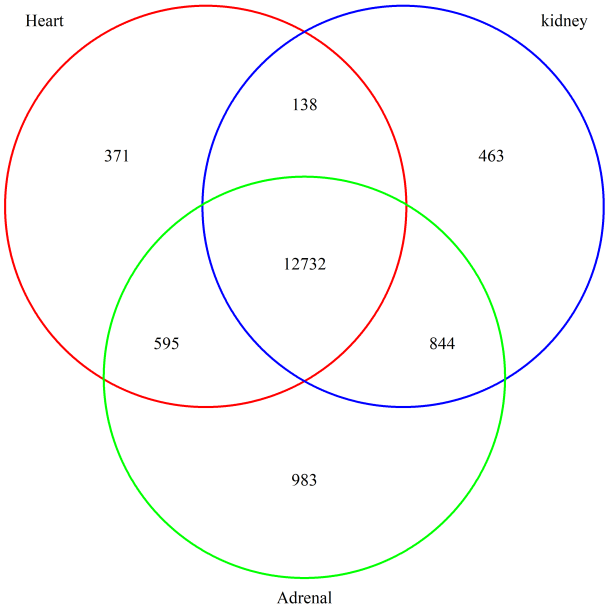
Ploting the venn diagrams

- store the organ names in `venn_titles` variable

```
venn_titles <- c("Heart", "kidney", "Adrenal")
```

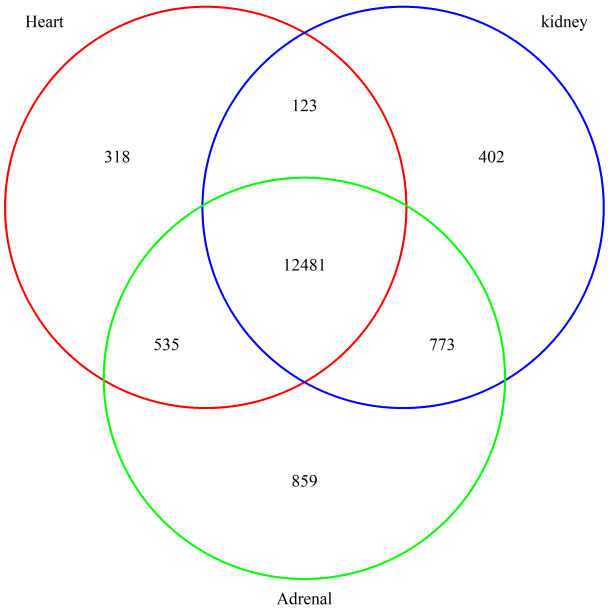
Venn diagram for *heart*, *kidney* and *adrenal* for all expressed genes

```
venn.diagram(x = list(heart_genes$gene_name, kidney_genes$gene_name, adrenal_genes$gene_name),
             category.names = venn_titles,
             imagetype = "png",
             filename = 'Venn_diagram_1.png',
             col = c("red", "blue", "green"))
```



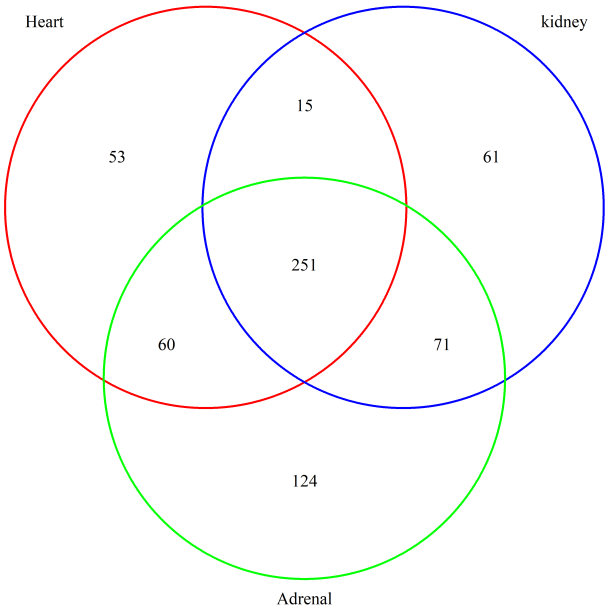
Venn diagram for *heart*, *kidney* and *adrenal* for all expressed genes of type `coding_protein`

```
venn.diagram(x = list(heart_genes[heart_genes$gene_type=="protein_coding","gene_name"],
                      kidney_genes[kidney_genes$gene_type=="protein_coding","gene_name"],
                      adrenal_genes[adrenal_genes$gene_type=="protein_coding","gene_name"]),
             category.names = venn_titles,
             imagetype = "png",
             filename = 'Venn_diagram_2.png',
             col = c("red", "blue", "green"))
```



Venn diagram for *heart*, *kidney* and *adrenal* for all expressed genes of type `lincRNA`

```
venn.diagram(x = list(heart_genes[heart_genes$gene_type=="lincRNA","gene_name"],
                      kidney_genes[kidney_genes$gene_type=="lincRNA","gene_name"],
                      adrenal_genes[adrenal_genes$gene_type=="lincRNA","gene_name"]),
             category.names = venn_titles,
             imagetype = "png",
             filename = 'Venn_diagram_3.png',
             col = c("red", "blue", "green"))
```



We choose the `Liver` and make the barplot for 20 most expressed genes

- select sample of the liver organ and the `gene_type` column

```
liver_samples <- dataSet[,str_starts(names(dataSet), pattern ="Liver")]
```

- for each row we take every sample check if the value is greate than `0.1` and make the sum of boolean values for each row

```
gene_expressed_row_sum <- rowSums(liver_samples>0.1)
```

- check if the gene is expressed by taking the row sum and see if it is equal to the number of samples for the organ

```
gene_expressed_check <- gene_expressed_row_sum == dim(liver_samples)[2]
```

- we filter the data and select expressed genes

```
liver_genes <- liver_samples[gene_expressed_check,]
```

- we conpute the row mean using the for samples

```
liver_genes$row_mean <- rowMeans(liver_genes)
```

- we order the expressed data according to the row_mean

```
liver_genes_ordered <- liver_genes[order(liver_genes$row_mean, decreasing = TRUE),]
```

- select the 20 most expressed genes

```
liver_20_genes <- head(liver_genes_ordered, n=20)
```

Drawing the barplot for each sample of data for the liver organ and also using the `rownames` as the label for the `x axis`

Ploting liver female 1 20 most expressed genes

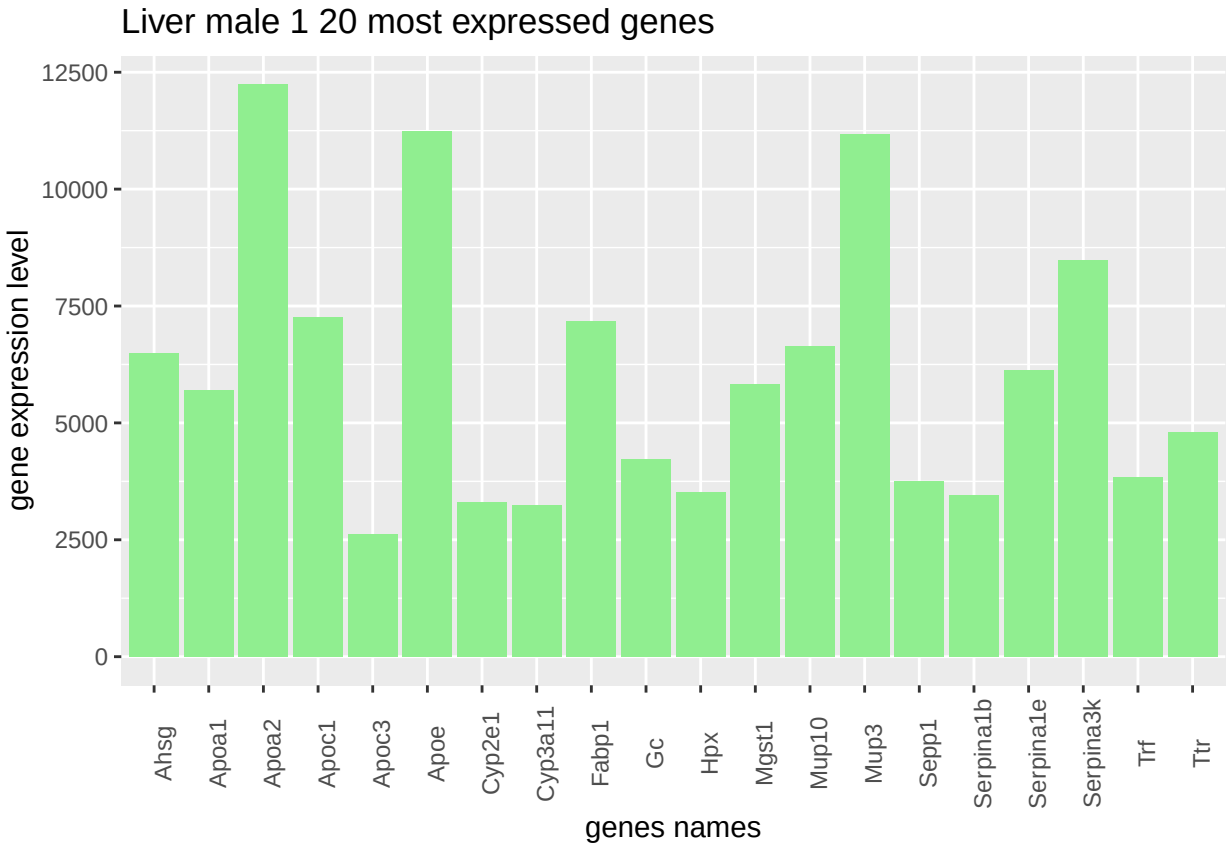
```
ggplot(liver_20_genes, aes(rownames(liver_20_genes), Liver_Female_1)) +
    labs(title = "Liver female 1 20 most expressed genes", x="genes names", y="gene expression level") +
    geom_bar(stat = "identity",fill="pink") + theme(axis.text.x = element_text(angle = 90))
```

Ploting liver female 2 20 most expressed genes

```
ggplot(liver_20_genes, aes(rownames(liver_20_genes), Liver_Female_2)) +
    labs(title = "Liver female 2 20 most expressed genes", x="genes names", y="gene expression level") +
    geom_bar(stat = "identity", fill="red") + theme(axis.text.x = element_text(angle = 90))
```

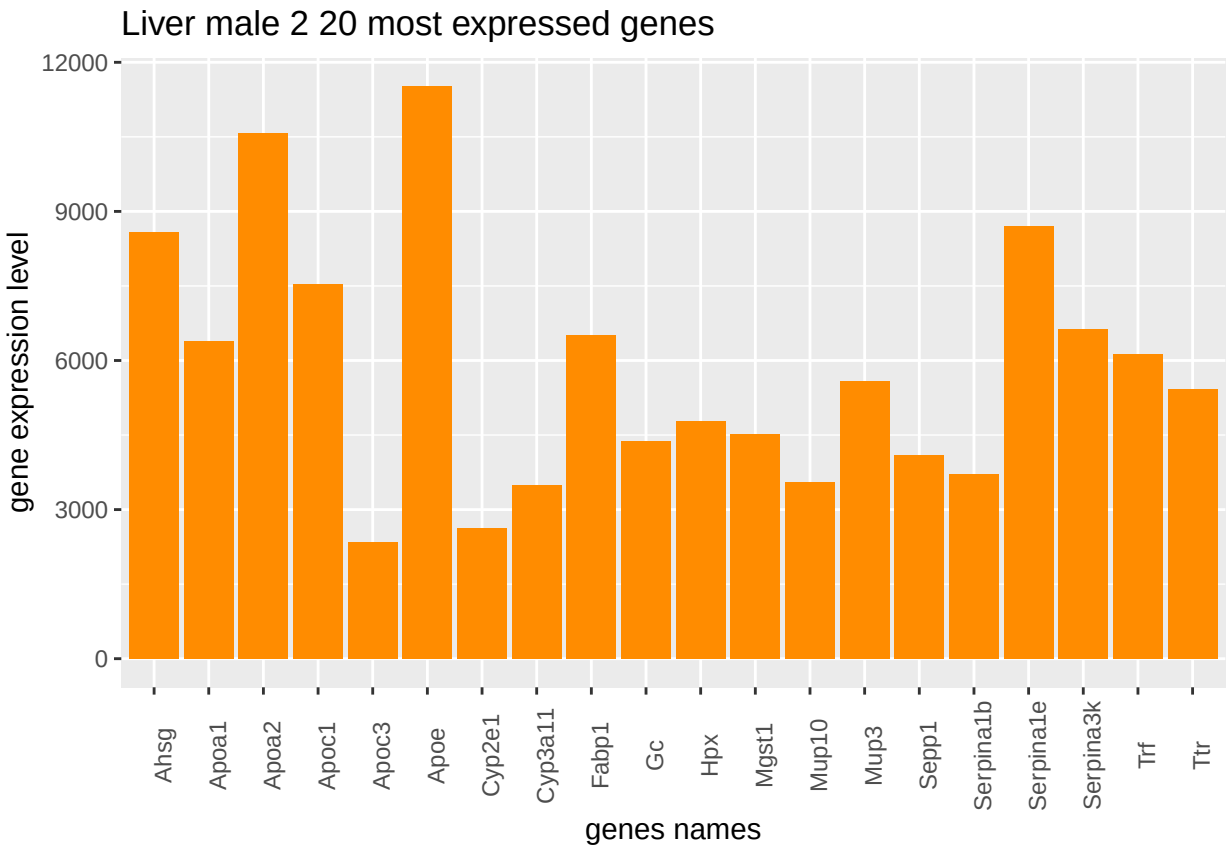
Ploting liver male 2 20 most expressed genes

```
ggplot(liver_20_genes, aes(rownames(liver_20_genes), Liver_Male_1)) +
    labs(title = "Liver male 1 20 most expressed genes", x="genes names", y="gene expression level") +
    geom_bar(stat = "identity", fill="lightgreen") + theme(axis.text.x = element_text(angle = 90))
```



Ploting liver male 2 20 most expressed genes

```
ggplot(liver_20_genes, aes(rownames(liver_20_genes), Liver_Male_2)) +
    labs(title = "Liver male 2 20 most expressed genes", x="genes names", y="gene expression level") +
    geom_bar(stat = "identity", fill="darkorange") + theme(axis.text.x = element_text(angle = 90))
```


Liver male 2 20 most expressed genes
gene expression level
12000
9000
6000
3000
0
Ahsg
Apoa1
Apoa2
Apoc1
Apoc3
Apoe
Cyp2e1
Cyp3a11
Fabp1
Gc
Hpx
Mgst1
Mup10
Mup3
Sepp1
Serpina1b
Serpina1e
Serpina3k
Trf
Ttr
genes names