Re-analysis of fetal and adult brain raw RNA-seq data from the study “Developmental regulation of human cortex transcription and its clinical relevance at base resolution” (Jaffe et al, 2015 Jan PMID:25501035)

The aim of this project was to analyse the raw RNA-seq data of 24 post-mortem brain samples from the above study (6 adult donors and 6 fetal donors) and find differentially expressed genes, eventually assessing the gene set relationship with histon modification profile from the Roadmap epigenetic project. This can help to better understand human brain development. The pipeline consisted of several steps:

1. Data retrieval
2. Alignment to reference genome
3. Quality control
4. Feature counts
5. Exploratory analysis
6. Differential expression analysis
7. Gene set analysis

Steps 1-4 were performed in Galaxy at https://usegalaxy.org/. (NOTE: on this free public server it’s not possible to run the entire analysis in one continuos workflow, due to memory restrictions).
Steps 5-7 (and some operations in steps 3 and 4) were performed in R 4.0.2 on a Windows machine.

1. Data retrieval

First, with a text editor, I built a tab separated phenotype table with 8 columns: TecReplicate (SRR accession number of the run), Series (1 for runs starting with SRR155, 2 for runs starting with SRR207), Sample (Donor ID), LStage (Adult or Fetal), Age, RIN (RNA Integrity Number), Gender, Prov (biomaterial provider) and Race.

Informations to fill in the table were found at http://www.ncbi.nlm.nih.gov/biosample, searching for the following BioSample numbers: 2999520, 2999521, 2999524, 2999549, 2999550, 2999551, 2999518, 2999519, 2999522, 2999539, 2999544, 2731373. (To get SRR, click on SRA on each of the BioSample pages). The table was saved as PhenoData.txt and put in the R working directory.

Then, I uploaded the data using the *“Faster Download and Extract Reads in FASTQ (Galaxy Version 2.10.8+galaxy0)”* tool, using “SRR accession” as Input type. I ran the jobs with default options, that produce 4 outputs for each run, among which a “Pair-end Data (fasterq-dump)” collection.

2. Alignment to reference genome

I aligned each collection to the built-in human genome hg19 assembly using the spliced alignment tool *“HISAT2 (Galaxy Version 2.1.0+galaxy5)”*, with default parameters, obtaining 24 Dataset collections (with BAM files inside). I stored in a document the informations found in each job summary about number of reads, reads aligned 1 time and reads aligned >1 times.

3. Quality control

I ran the *“FastQC (Galaxy Version 0.72+galaxy1)”* tool with default parameters on each of the Dataset collections produced by HISAT2. Then, I ran 2 times *“MultiQC (Galaxy Version 1.8+galaxy1)”* tool to merge quality data from the 2 groups of samples (Adults and Fetals), setting “FastQC” as the tool used to generate logs and “Raw data” as Type of FastQC output. So, among other outputs, 2 MultiQC Webpages were produced and used to check reads quality. I added a column called “Quality” to the previously created PhenoData.txt file, flagging as “Good” samples shown in green in the Per Base Sequence column of the “Fast QC: Status check” plot, and as “Poor” samples shown in red.

Then I performed some statistical tests in R, after downloading data from the “Per Sequence Quality Scores” plots as 2 files called “FetalQCmeans.csv” and “AdultQCmeans.csv”. Objects “fetal” and “adult” represent total % of reads aligned calculated from summary information saved in step 2.

```
names(adult)<-
row.names(pdata[pdata$LStage=="Adult",])
names(fetal)<-
row.names(pdata[pdata$LStage=="Fetal",])
adult_means<-read.csv("AdultQCmeans.csv")
adult_run_means<-as.numeric()
for(i in 2:13) {
adult_run_means[i]<-
weighted.mean(adult_means[,1],adult_means[,i
])
}
```

```r
names(adult_run_means)<-colnames(adult_means)
fetal_means<-read.csv("FetalQCmeans.csv")
fetal_run_means<-as.numeric()
for(i in 2:13) {
fetal_run_means[i]<-weighted.mean(fetal_means[,1],fetal_means[,i]/1000)
}
names(fetal_run_means)<-colnames(fetal_means)
means<-c(fetal_run_means[-1],adult_run_means[-1])
mappingrates<-c(fetal,adult)
qcmeans<-means[names(mappingrates)]
t.test(fetal,adult,alternative="g")
t.test(adult_run_means,fetal_run_means)
pdata<-read.table("PhenoData.txt", header=T)
row.names(pdata)<-pdata$TecReplicate
t.test(qcmeans[pdata$Series==1],qcmeans[pdata$Series==2],alternative="g")
```

The only significant trend I discovered was a difference in mean Quality Score between the 2 series of replicates (Series 2 lower than Series 1).

4. Feature counts

Counts were calculated on BAM files with *"featureCounts (Galaxy Version 1.6.4+galaxy2)"* tool, choosing Gene annotation file "in your history" and default parameters. I used annotation file uploaded with the "*UCSC Main table Browser*" tool, with the "Send output toGalaxy" option. Search parameters were:

clade: Mammal , genome: Human, assembly: Feb. 2009 (GRCh37/hg19), group: Genes and Gene prediction, track: UCSC Genes, table: knownGene, region: Genome, output format: GTF – gene transfer format (limited), file type returned: plain text

Each of the 24 jobs produced 2 outputs, called Summary and Counts. Counts files were downloaded in the "featurecounts" subdirectory of R working dir, then merged in a table with the following code:

```r
names<-dir("featurecounts",full.names = T)
tables<-list()
for (i in 1:length(names)) {
tables[[i]]<-read.table(names[i],sep = "\t",header=TRUE)
}
counts<-data.frame(row.names=tables[[1]][,1])
for (i in 1:length(tables)) {
counts[,i]<-tables[[i]][,2]
}
samplenames<-character()
for (i in 1:length(tables)) {
samplenames[i]<-names(tables[[i]])[2]
}
names(counts)<-samplenames
counts←counts[,row.names(pdata)]
```

5. Exploratory analysis

To normalize count data, I used the DESeq2 package algorithm; then, I explored the normalized data making some plots, summaries and transformations:

```r
library(DESeq2)
de<-DESeqDataSetFromMatrix(counts,pdata,design=~1)
size<-estimateSizeFactors(de)
normalized<-counts(size,normalized=TRUE)
row.names(normalized)<-row.names(counts)
colnames(normalized)<-names(counts)
summary(normalized[,1:3])
boxplot(normalized)
normlog<-log2(normalized+1)
boxplot(normlog)
summary(normlog[,1:3])
normlogfilt<-normlog[rowMeans(normlog)>0,]
dim(normalized)
dim(normlogfilt)
boxplot(normlogfilt)
summary(normlogfilt[,1:3])
hist(normlogfilt[,1])
```

I used log2 transform filtered for rowMeans>0 for PCA:

```r
centered<-normlogfilt-rowMeans(normlogfilt)
svd<-svd(centered)
plot(svd$d^2/sum(svd$d^2)*100,ylab="% variance explained")
pc<-prcomp(normlogfilt)
par(mfrow=c(2,3))
rin<-ifelse(pdata$RIN<=median(pdata$RIN),"low","high")
```

```
pdata$RINlevel<-rin
par(pch=19)
for(i in c(2,4,5,7,10,11)) {
plot(pc$rotation[,1],pc$rotation[,2],col=as.numeric(as.factor(pdata[,i])),main=names(pdata)[i])
}
corr<-matrix(ncol=2,nrow=11)
for(i in c(2,4,5,7,10,11)) {
  for(j in 1:2) {
    corr[i,j]<-
cor(pc$rotation[,j],as.numeric(as.factor(pdata[,i])))
  }
}
corr<-corr[!is.na(corr[,1]),]
row.names(corr)<-names(pdata)
[c(2,4,5,7,10,11)]
colnames(corr)<-c("First PC","Second PC")
corr
```

Based on the results of this analysis, as Quality appeared to be the main counfounder, I decided to exclude samples labeled as poor quality and re-perform PCA:

```
countgood<-counts[,pdata$Quality=="Good"]
pdatagood<-pdata[pdata$Quality=="Good",]
de1<-
DESeqDataSetFromMatrix(countgood,pdatagood,design=~1)
size<-estimateSizeFactors(de1)
normalized<-counts(size,normalized=TRUE)
row.names(normalized)<-row.names(countgood)
colnames(normalized)<-names(countgood)
normloggood<-log2(normalized+1)
normlogfiltgood<-
normloggood[rowMeans(normloggood)>0,]
pc<-prcomp(normlogfiltgood)
corr<-matrix(ncol=2,nrow=11)
for(i in c(2,4,5,6,7,10,11)) {
for(j in 1:2) {
corr[i,j]<-
cor(pc$rotation[,j],as.numeric(as.factor(pdatagood[,i])))
}
}
corr<-corr[!is.na(corr[,1]),]
row.names(corr)<-names(pdatagood)
[c(2,4,5,6,7,11)]
colnames(corr)<-c("First PC","Second PC")
corr
```

6. Differential expression analysis

I performed this step using Life Stage as the regressor of interest and Series as the only confounder, on count data from good quality samples, filtered by rowMeans>0.

The null hypothesis was: "Mean expression of a feature is equal in Fetal and Adult samples, after adjusting for the set of technical replicates", while the alternative hypothesis was: "A feature is differentially expressed in Fetal and Adult samples, after adjusting for the set of technical replicates".

```
pdatagood$LStage<-
as.factor(pdatagood$LStage)
pdatagood$Series<-
as.factor(pdatagood$Series)
countgoodfilt<-
countgood[rowMeans(countgood)>0,]
pdatagood$LStage<-
relevel(pdatagood$LStage,"Fetal")
de<-DESeqDataSetFromMatrix(countgoodfilt,
pdatagood, design= ~ LStage + Series)
de<-DESeq(de)
resultsNames(de)
res<-
results(de,name="LStage_Adult_vs_Fetal")
res<-res[!is.na(res$padj),]
dim(res)[1]
dim(res[res$padj<0.05,])[1]
sign<-(res[res$padj<0.05,])
dim(sign[sign$log2FoldChange>0,])[1]
dim(sign[sign$log2FoldChange<0,])[1]
hist(res$padj)
```

This histogram reveals a good distribution of adjusted p-values, while results show that (at a max FDR of 0.05), there are 11424 differentially expressed features (out of 82960 in the UCSC database): 6010 are overexpressed in adults and 5414 are overexpressed in fetuses.

Lastly, here is the code used to produce a volcano plot where genes with both higher fold change (>100 times) and a FDR <0.01 are labeled in green.

```
par(pch=19)
plot(res$log2FoldChange,-log(res$padj),
col=ifelse(abs(res$log2FoldChange)>log2(100)
&res$padj<.01,"green","black"))
```

7. Gene set analysis

The goal of this step was to find changes in the histon modification H3K4m3 (involved in expression regulation) in fetal brain, adult brain and liver regarding the DE genes.

Informations about Chip-seq analysis for H3K4me3 and the corresponding donors were found at https://www.ncbi.nlm.nih.gov/geo/roadmap/epigenomics, then the narrow peak data for the chosen samples were accessed via AnnotationHub; GRanges for UCSC known gene transcripts were also retrieved. Then, the dataset was subsetted by gene IDs of differentially expressed transcripts, and a GRanges object with the promoters was created.

```
library(AnnotationHub)
hub<-AnnotationHub()
adult<-
query(hub,"BI.Brain_Mid_Frontal_Lobe.H3K4me3
.112.narrowPeak.gz")
adult<-adult[[1]]
fetal<-query(hub,"UCSF-
UBC.Fetal_Brain.H3K4me3.HuFNSC02.narrowPeak.
gz")
fetal<-fetal[[1]]
liver<-
query(hub,"BI.Adult_Liver.H3K4me3.3.narrowPe
ak.gz")
liver<-liver[[1]]
ucsc<-
query(hub,"TxDb.Hsapiens.UCSC.hg19.knownGene
.sqlite")
ucsc<-ucsc[[1]]
tx<-transcripts(ucsc)
tx<-tx[tx$tx_name%in%row.names(sign),]
names(tx)<-tx$tx_name
tx<-tx[row.names(sign)]
prom<-promoters(tx)
```

To start, I summarized in a matrix the number of overlaps between our DE genes and peaks in tissues of interest:

```
promad<-findOverlaps(prom,adult)
n_prom_olap_ad<-
length(unique(queryHits(promad)))
promfet<-findOverlaps(prom,fetal)
n_prom_olap_fet<-
length(unique(queryHits(promfet)))
promliv<-findOverlaps(prom,liver)
n_prom_olap_liv<-
length(unique(queryHits(promliv)))
m<-matrix(nrow=3,ncol=2)
m[1,1]<-n_prom_olap_fet
m[2,1]<-n_prom_olap_ad
m[3,1]<-n_prom_olap_liv
row.names(m)<-c("Prom H3K4m3 fetal","Prom
H3K4m3 adult","Prom H3K4m3 liver")
colnames(m)<-c("Number","%")
for (i in 1:nrow(m)) {
m[i,2]<-m[i,1]/dim(sign)[1]*100
}
m
```

Then, Fisher's exact tests were run and I could conclude that there are small but definitely significant changes in histon modification patterns:

```
n<-matrix(nrow=2, ncol=2)
colnames(n)<-c("H3K4m3","no H3K4m3")
row.names(n)<-c("Adult brain","Liver")
n[1,1]<-n_prom_olap_ad
n[2,1]<-n_prom_olap_liv
n[1,2]<-dim(sign)[1]-n[1,1]
n[2,2]<-dim(sign)[1]-n[2,1]
n
fisher.test(n)
o<-matrix(nrow=2, ncol=2)
colnames(o)<-c("H3K4m3","no H3K4m3")
row.names(o)<-c("Adult brain","Fetal brain")
o[1,1]<-n_prom_olap_ad
o[2,1]<-n_prom_olap_fet
o[1,2]<-dim(sign)[1]-o[1,1]
o[2,2]<-dim(sign)[1]-o[2,1]
o
fisher.test(o)$p.value
```

Further analysis showed also differences in H3K4m3 between genes overexpressed in fetuses and genes overexpressed in adults, in accord with the hypothesis that at least some of our DE genes are involved in brain development. Finally, studying the correlation between changes in methylation and log2FC gives also biologically consistent results.

```

prom$logfc<-sign$log2FoldChange
prom_over_fet<-prom[prom$logfc<0]
overfetad<-findOverlaps(prom_over_fet,adult)
```

```r
n_overfet_olap_ad<-
length(unique(queryHits(overfetad)))
overfetfet<-
findOverlaps(prom_over_fet,fetal)
n_overfet_olap_fet<-
length(unique(queryHits(overfetfet)))
prom_over_ad<-prom[prom$logfc>0]
overadad<-findOverlaps(prom_over_ad,adult)
n_overad_olap_ad<-
length(unique(queryHits(overadad)))
overadfet<-findOverlaps(prom_over_ad,fetal)
n_overad_olap_fet<-
length(unique(queryHits(overadfet)))
p<-matrix(nrow=2,ncol=2)
row.names(p)<-c("Prom overexpr in fetuses",
"Prom overexpr in adults")
colnames(p)<-c("Fetal peaks","Adult peaks")
p[1,1]<-n_overfet_olap_fet
p[1,2]<-n_overfet_olap_ad
p[2,1]<-n_overad_olap_fet
p[2,2]<-n_overad_olap_ad
p
fisher.test(p)$p.value
t<-table(queryHits(promad))
t<-as.data.frame(t)
ad<-numeric(dim(sign)[1])
for(i in 1:length(ad)) {
if(any(t$Var1==i)) ad[i]<-t[t$Var1==i,2]
}
t<-table(queryHits(promfet))
t<-as.data.frame(t)
fet<-numeric(dim(sign)[1])
for(i in 1:length(fet)) {
if(any(t$Var1==i)) fet[i]<-t[t$Var1==i,2]
}
diff<-ad-fet
boxplot(prom$logfc~diff)
mod<-lm(prom$logfc~diff)
abline(mod$coefficients,col="red")
cor(diff,prom$logfc)
```

Session Info

```
R version 4.0.2 (2020-06-22)

Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows 10 x64 (build 19041)

Matrix products: default
attached base packages:
[1] parallel  stats4    stats     graphics
grDevices utils     datasets  methods   base

other attached packages:
GenomicFeatures_1.40.1, AnnotationDbi_1.50.3
Bsgenome.Hsapiens.UCSC.hg19_1.4.3,
Bsgenome_1.56.0,  Biostrings_2.56.0
Xvector_0.28.0,   rtracklayer_1.48.0,
AnnotationHub_2.20.2,  BiocFileCache_1.12.1,
dbplyr_2.0.0,    DESeq2_1.28.1,
SummarizedExperiment_1.18.2,
DelayedArray_0.14.1, matrixStats_0.57.0,
Biobase_2.48.0,    GenomicRanges_1.40.0,
GenomeInfoDb_1.24.2,    Iranges_2.22.2,
S4Vectors_0.26.1,       BiocGenerics_0.34.0

loaded via a namespace (and not attached):
bitops_1.0-6,bit64_4.0.5,RcolorBrewer_1.1-2,
progress_1.2.2, httr_1.4.2, tools_4.0.2,
R6_2.5.0, DBI_1.1.0, colorspace_2.0-0,
withr_2.3.0, tidyselect_1.1.0,
prettyunits_1.1.1, bit_4.0.4, curl_4.3,
compiler_4.0.2, xml2_1.3.2, scales_1.1.1,
genefilter_1.70.0, askpass_1.1,
rappdirs_0.3.1, stringr_1.4.0,
digest_0.6.25, Rsamtools_2.4.0,
rmarkdown_2.5, pkgconfig_2.0.3,
htmltools_0.5.0, fastmap_1.0.1, rlang_0.4.8
rstudioapi_0.13, RSQLite_2.2.1, shiny_1.5.0
generics_0.1.0, BiocParallel_1.22.0 ,
dplyr_1.0.2, Rcurl_1.98-1.2, magrittr_2.0.1
GenomeInfoDbData_1.2.3, Matrix_1.2-18
Rcpp_1.0.5, munsell_0.5.0, lifecycle_0.2.0
stringi_1.5.3, yaml_2.2.1, zlibbioc_1.34.0
grid_4.0.2, blob_1.2.1, promises_1.1.1
crayon_1.3.4, lattice_0.20-41, splines_4.0.2
annotate_1.66.0, hms_0.5.3, locfit_1.5-9.4
knitr_1.30, pillar_1.4.7, geneplotter_1.66.0
biomaRt_2.44.4, XML_3.99-0.5, glue_1.4.2
BiocVersion_3.11.1, evaluate_0.14
BiocManager_1.30.10,vctrs_0.3.5,
httpuv_1.5.4, gtable_0.3.0, openssl_1.4.3
purrr_0.3.4, assertthat_0.2.1,
ggplot2_3.3.2, xfun_0.19, mime_0.9,
xtable_1.8-4, later_1.1.0.1, survival_3.1-12
tibble_3.0.4, GenomicAlignments_1.24.0,
memoise_1.1.0, ellipsis_0.3.1,
interactiveDisplayBase_1.26.3
```