Assignment 3 group 6

Adham Khaled, Juan Manuel Medina, Antonio Ortega, Isabella Skandorff & Andreas Vincent

May 30, 2017

```
library("ggplot2")
library("cowplot")
library("reshape2")
library("dplyr")
library("gridExtra")
theme_set(theme_bw())
```

Part 1

(1)

Just because IRX3 shows a higher signal than RXR in both microarray experiments, it does not necessarily mean that it is more expressed, as this signal is influenced by methodological artifacts. Th detected signal can be confounded by global effects such as differential sample preparation process, and local effects such as gene specific variation. For instance, a gene with high tendency to dyeing will likely produce a stronger signal than another gene with a feeble one regardless of their actual levels of expression. This could potentially overlook the biological insight, however replicates and normalization techniques can solve the issue.

(2)

- H_0: Mean of expression values across the HIV and the control patients are the same

  ```
  $$\mu_{control} = \mu_{AIDS}$$
  ```

- H_1: Mean of expression values across the HIV and the control patients are different from each other.

  ```
  $$\mu_{control} \neq \mu_{AIDS}$$
  ```

We need to test the difference of means for two distributions of expression values for every gene in the dataset. As there is no evidence showing the values don't follow a normal distribution, we don't reject it and thus assume that the mean of each sample is student's-t distributed. Therefore we decided to use the student's t test.

```
df <- read.table("data_for_part_1/normalized_data.txt", h = F)
names(df) <- paste(rep(c("HIV", "Control"), each = 5), 1:5, sep = "-")
# t-test p-values between HIV and Control
pvals <- apply(df, 1, function (x) t.test(x[1:5], x[6:10])$p.value)
```

(3)

Genes with a p-value less than 0.05:

```
diff_genes <- sum(pvals < 0.05)
diff_genes
```

```
## [1] 1911
```

False positives = 5% of all tests

```
round(0.05 * length(pvals), 0)
```

```
## [1] 1114
```

(4)

Number of genes with p-value < 0.2 with Bonferroni correction

```
# Bonferroni multiple testing correction
bonferroni_pvals <- p.adjust(pvals, method = "bonferroni")

sum(bonferroni_pvals < 0.2)
```

```
## [1] 0
```

```
# BH multiple testing correction
BH_pvals <- p.adjust(pvals, method = "BH")

# Number of genes with p-value < 0.2 with BH correction
sum(BH_pvals < 0.2)
```

```
## [1] 12
```

```
# Expected FP = threshold% used in the t.test -> 0.2% of the t. tests are expected to be FP
round(0.2 * sum(BH_pvals < 0.2), 0)
```

```
## [1] 2
```

The number of significant bonferroni adjusted genes was 0, while the FDR method yielded 12 significant genes. The expected number of false positive genes under FDR was 2, or 20% of the significant FDR adjusted genes (p-value <0.2).

(5)

```
#storing the means in the same dataframe as the original gene expression set
df$D_mean <- rowMeans(df[, c(1:5)])
df$C_mean <- rowMeans(df[, c(6:10)])

my_foldchanges <- log2(df[, 11]) - log2(df[, 12])
```

(6)

```
# Report the fold changes for the genes with a FDR < 0.2 (using the BH method)
genes_FDR_lower_0.2 <- log2(df[which(BH_pvals < 0.2), 11]) -
                       log2(df[which(BH_pvals < 0.2), 12])

genes_FDR_lower_0.2
```

```
##  [1] 0.03680802 0.33677927 0.06387247 0.15993496 0.60119418 0.33087497
##  [7] 0.08063546 0.09060348 0.08200741 0.17300186 0.09105046 0.16164736
```

It is interesting to point out that the 12 genes with a significative fold change are all upregulated in the HIV patients, where the gene with the highest fold change had a 50% higher expression level. ($log_2(FC) = 0.6$). However, some of the genes with the lowest fold changes might not be biologically significant.

Part 2

2.2

Table 1: Average number of lines in the 3 assembled transcripts for both condtions. Mean in the bottom row.

	KD	WT
1	1004	991
2	1011	1011
3	1064	991
mean	1026	998

2.3

Lines of the combined GTF file (combined transcriptome): 1207

The number of lines in the GTF files not only reflects the transcripts but also the exons. Therefore the raw count with wc -l does not really show the size of the transcriptome sets, and a refined version should be used. We can definitely say that not all variants are available in all the local transcriptomes.

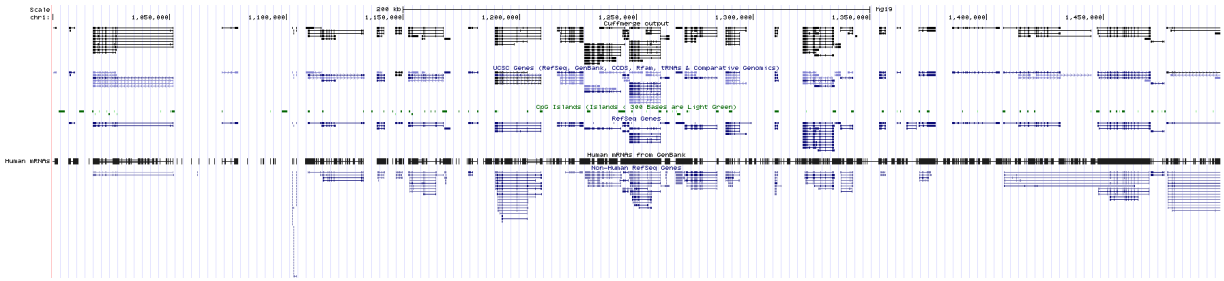

Figure 1: Capture of the chr1:1,000,000-1,500,00 region displaying the regions in the genome where transcripts were assembled by cufflinks (cuffmerge output). The combined or "global" transcriptome is displayed in the top track. **??**

2.4

Extracted from Roberts *et al.*

The biochemistry of RNA-Seq library preparation results in cDNA fragments that are not uniformly distributed within the transcripts they represent. This non-uniformity must be accounted for when estimating expression levels. The randomness inherent in many of the preparation steps for RNA-Seq leads to fragments whose starting points (relative to the transcripts from which they were sequenced) appear to be chosen approximately uniformly at random. This observation has been the basis of assumptions underlying a number of RNA-Seq analysis approaches that, in computer science terms, invert the 'reduction' of transcriptome estimation to DNA sequencing. However, recent careful analysis has revealed both positional and sequence-specific biases in sequenced fragments. Positional bias refers to a local effect in which fragments are preferentially located towards either the beginning or end of transcripts. Sequence-specific bias is a global effect where the sequence surrounding the beginning or end of potential fragments affects their likelihood of being selected for sequencing. These biases can affect expression estimates, and it is therefore important to correct for them during RNA-Seq analysis.

2.5

Based on answer by Merico (see References) .

Cuffidff performs several tests, one of them being "Splicing differential expression". This looks at the distribution of isoforms of each gene and compares it with those found in other conditions. The statistic will sense local isoforms enrichments across conditions.

Part 3

```
gene <- read.table("data_for_part_3/cuffdiff_gene_differential_expression.txt", h = T)
transcript <- read.table("data_for_part_3/cuffdiff_transcript_differential_expression.txt", h = T)
splicing <- read.table("data_for_part_3/cuffdiff_splicing_differential_expression.txt", h = T)
names(transcript)[1] <- "transcript_id" # changing test_id column name to transcript_id
```

3.2

```
# Make two new dataframes from gene and transcript containing all rows with expression in at least one
# of the conditions, that is, with an expression value < 0 in column 8 or 9 (value 1 or value2)
gene_expressed <- gene[gene$value_1 | gene$value_2 != 0, ]
transcript_expressed <- transcript[which(transcript$value_1 | transcript$value_2 != 0), ]
```

3.3

```
# Number of genes and transcripts that were expressed and how many
# were significantly differentially expressed between conditions (column 14 or significant = "yes")
nrow(gene_expressed)
```

```
## [1] 26
```

```
nrow(transcript_expressed)
```

```
## [1] 98
```

```
# dplyr incorporation
gene_expressed %>% filter(significant == "yes") %>% nrow
```

```
## [1] 12
```

```
nrow(transcript_expressed[which(transcript_expressed$significant == "yes"), ])
```

```
## [1] 11
```

3.4

```
# Make two reduced dataframes from gene_expressed and transcript_expressed and merge them based on the
# gene ids
gene_reduced <- select(gene_expressed, gene_id, gene, value_1, value_2)
transcript_reduced <- select(transcript_expressed, transcript_id, gene_id, value_1, value_2)

gen_tr <- merge(gene_reduced, transcript_reduced, by = "gene_id",
                              suffixes = c(".GENE", ".TRANSCRIPT"))

dim(gen_tr)    # 98 rows, 7 columns
```

```
## [1] 98  7
```

3.5

```
# Calculate the IF for each transcript in both conditions
# (IF = isoform_exp / gene_exp) and the dIF (IF2 - IF1)

gen_tr$IF1 <- gen_tr$value_1.TRANSCRIPT / gen_tr$value_1.GENE
gen_tr$IF2 <- gen_tr$value_2.TRANSCRIPT / gen_tr$value_2.GENE

gen_tr$dIF <- gen_tr$IF2 - gen_tr$IF1

# Overwrite the IF1 and IF2 columns of the dataframe
gen_tr <- gen_tr[complete.cases(gen_tr),]

# Recalculate the dIF with the corrected IF1 and IF2 values and overwrite
gen_tr$dIF <- gen_tr$IF2 - gen_tr$IF1
```

If the expression value of a certain gene in condition is 0, when we try to obtain the IF dividing the isoform expression values by it, we obtain a non-sense mathematical result due to the fact that we are dividing by 0, and hence, R shows this non-sense result as NaN. As we are considering only the isoform switching information, these rows can be removed, since no comparison of isoform fractions across the conditions can be performed.

3.6

```
# Calculate the mean and median dIF value and discuss
mean(gen_tr$dIF)
```

```
## [1] 7.468228e-09
```

```
median(gen_tr$dIF)
```

```
## [1] 0.0001255016
```

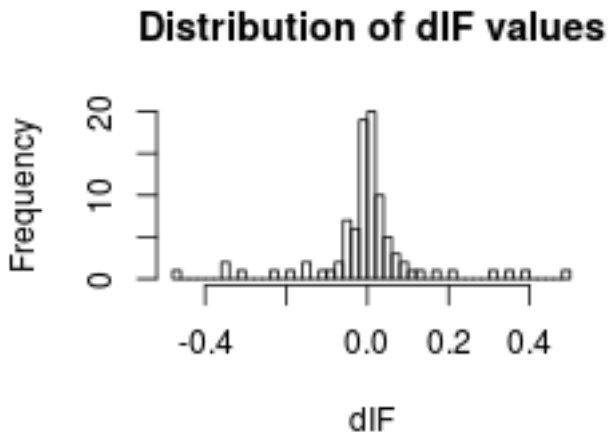

Figure 2: Distribution of the Isoform Fraction differential (dif) across conditions one and two. By definition the average dif is 0 and deviations from this number are due to numerical errors. The median should also equal to 0 if all genes had an even number of isoforms, which is hardly the case. However, this will generate small deviations from 0.

We obtain a mean and a median very close to 0, as seen in the figure 2. This agrees with an expected mean dIF of 0 since adding up all the dIF values for the different isoforms of the same gene is 0 by definition (although the median does not have to be 0). However, we are not obtaining a 0 mean due to numerical errors in R. In conclusion, the mean and the median don't provide any summary information of the data.

3.7

```
# Subset the merged dataframe to only contain genes with dIF > +0.25 and dIF < -0.25
# Then, add the p-value from the splicing data frame using the match() function
gen_tr_pot <- gen_tr[which(gen_tr$dIF > 0.25 | gen_tr$dIF < -0.25), ]
# p-values from splicing dataframe which same gene_id as the ones in gen_tr_potential_
# _switch dataframe
gen_tr_pot$pvals <- splicing[match(gen_tr_pot$gene_id, splicing$gene_id), 12]
```

3.8

```
filter(gen_tr_pot, gene_id == gen_tr_pot %>% arrange(pvals) %>% .$gene_id %>%
         as.character() %>% unique %>% .[1]) %>% select(transcript_id, gene, dIF, pvals)
```

```
##    transcript_id       gene        dIF   pvals
## 1 TCONS_00000021 uc001aeo.3 -0.3426094 0.00075
## 2 TCONS_00000022 uc001aeo.3  0.3426083 0.00075
```

3.9

Gene id is uc001aeo.3. It is named CPTP in the RefSeq track, while in the UCSC genes track it's GLTPD1.

The full name of the gene is ceramide-1-phosphate transfer protein.

From Uniprot (http://www.uniprot.org/uniprot/Q5TA50):

Mediates the intracellular transfer ofceramide-1-phosphate between organelle membranes and the cell membrane. Required for normal structure of the Golgi stacks. Can bind phosphoceramides with a variety of aliphatic chains, but has a preference for lipids with saturated C16:0 or monounsaturated C18:1 aliphatic chains, and is inefficient with phosphoceramides containing lignoceryl (C24:0). Plays a role in the regulation of the cellular levels of ceramide-1-phosphate, and thereby contributes to the regulation of phospholipase PLA2G4A activity and the release of arachidonic acid. Has no activity with galactosylceramide, lactosylceramide, sphingomyelin, phosphatidylcholine, phosphatidic acid and ceramide.

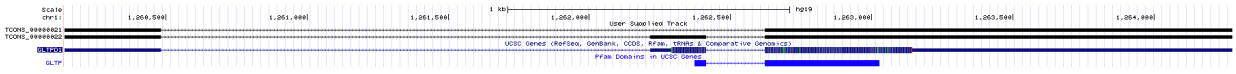

Figure 3: Capture of the UCSC browser at the GLTPD1 gene region. The tracks display the genomic position, a custom gtf file of transcripts, showing the two isoforms of the gene (TCONS_0000021 and TCONS_0000022), the UCSC genes and the PFam domains.

As can be seen in figure 4, this gene lies in the forward strand and comprises 3 exons. 2 of them are homologous to the GLTP PFam domain. This is interesting since one of the isoforms (TCONS_0000021) lacks exon 2, therefore missing the N-terminus residues of this domain. TCONS_0000022 features all 3 exons.

The structure of this protein is available at the PDB (under entry 4K84). It displays an all alpha fold, where most of the residues lie within an alpha helix. In figure 4 one can see that this second exon is spliced in TCONS_0000021, consists of almost two full helices. Only the C-terminus residues of the second helix are encoded in the third exon. This means that the protein product originating from the translation of TCONS_0000021 would lack the Nα helix as well as most of the next helix.

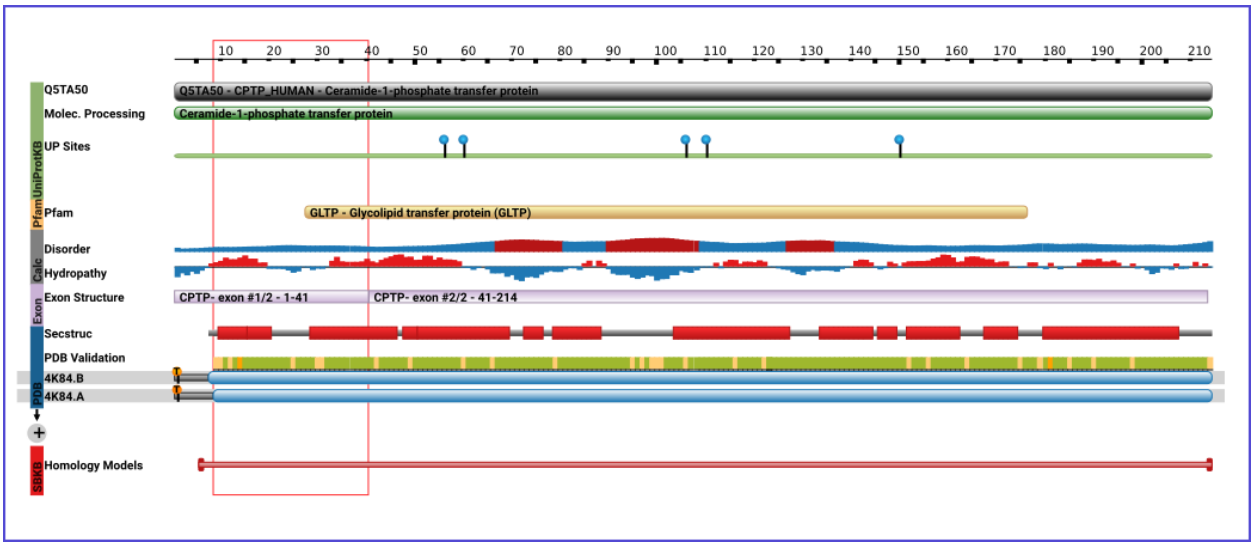

Figure 4: Organization of the GLTP domain in the GLTPD1 structure at the PDB under key 4K84. Several helices are distributed across the domain encompassing 2 exons. The two helices lying in the N-terminus of the domain are almost fully encoded by exon 2, whereas the rest of the domain lies in exon 3.

The CPTP domain consists of a double layer of alpha helices generating a hydrophobic pocket that allows for the transfer of lipids across membranes. This pocket is sealed by two Leu residues within the Nα helix (Simanshu *et al*). According to this, this hydrophobic pocket would be severely affected in protein product encoded by TCONS_0000021 (see figure 5).

Since TCONS_0000021 expression decreases one order of magnitude when transitioning to condition 2, this analysis provides evidence for an inhibition of the expression of its truncated protein in condition 2. This is quite interesting since TCONS_0000022 does not change its expression that much. This means that TCONS_0000021 regulation's does not act at the promoter level but at another downstream process. Even if TCONS_0000022 expression does not change that much, this could either mean that the full protein remains produced in a similar amount, or that there other inhibitory mechanisms acting postranscriptionally.

Table 2: Summary of the collected data. TCONS_0000021 decreases its expression 10 fold in condition 2 and interestingly lacks a proper hydrophobic pocket.

Transcript	Expression condition 1	Expression condition 2	Pocket
TCONS_00000021	3299.43	345.58	No
TCONS_00000022	4623.59	4335.45	Yes

In conclusion, this analysis has revealed that the *gltpd1* gene, producing to isoforms experiences an isoform switch. TCONS_0000021 expression decreases by almost 90%, while TCONS_0000022 remains similar. The structural difference of the proteins encoded by two isoforms, where the first isoforms codes for a protein with a truncated domain, provides an interesting insight in to the biology of lipid transfer across membranes and the function of the *gltpd1* gene.

References

1. Daniele Merico. Cufflinks / Cuffdiff Output - How Are Tests Different? BIOSTARS question.
2. Simanshu, D. K., Kamlekar, R. K., Wijesinghe, D. S., Zou, X., Zhai, X., Mishra, S. K., ... Patel, D. J. (2013). Non-vesicular trafficking by a ceramide-1-phosphate transfer protein regulates eicosanoids. Nature, 500(7463), 463–467. http://doi.org/10.1038/nature12332
3. Roberts, A., Trapnell, C., Donaghey, J., Rinn, J. L., & Pachter, L. (2011). Improving RNA-Seq expression estimates by correcting for fragment bias. Genome Biology, 12(3), R22. http://doi.org/10.1186/gb-2011-12-3-r22

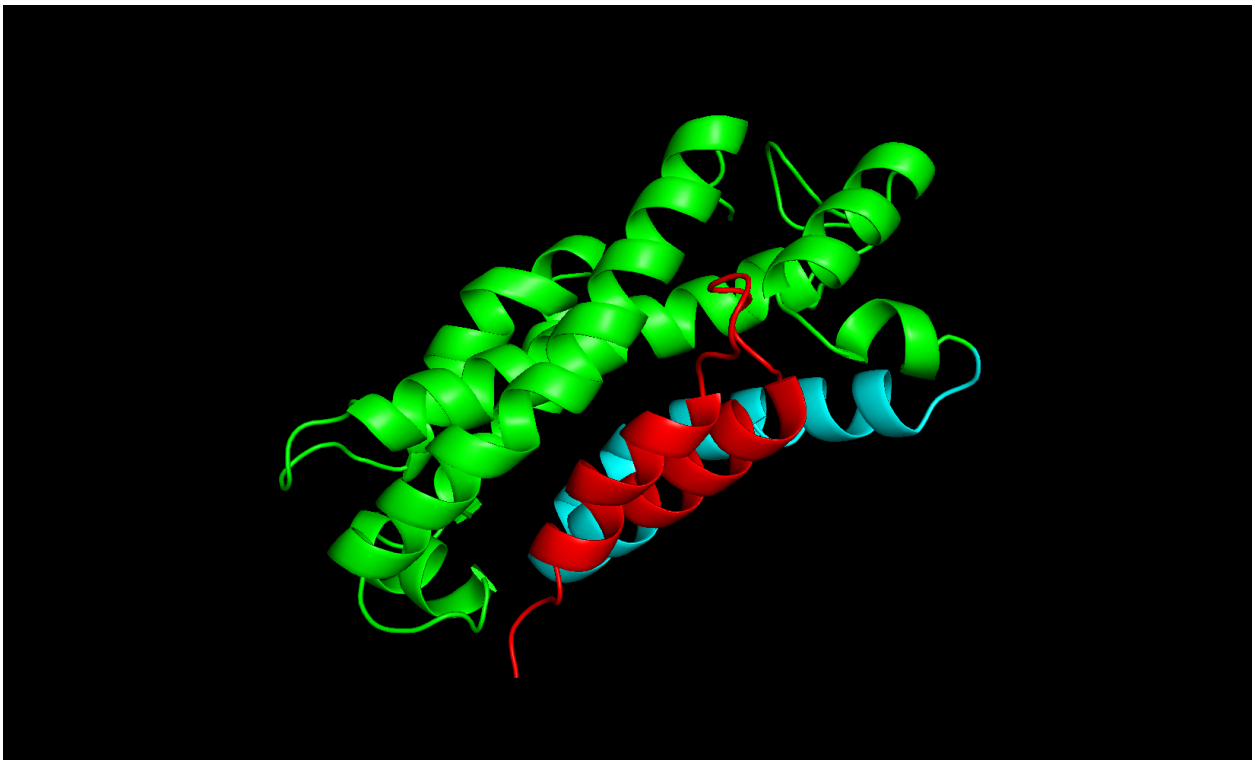

Figure 5: Pocket generated by the alpha helices in the GLTP domain. Green helices shape one side of the pocket. The blue and red helices enclose the pocket on the other side. The red helices are coded in exon 2 and are lost in the protein encoded by the TCONS_0000021. transcript.

```
## [11] LC_MEASUREMENT=es_ES.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] gridExtra_2.2.1 dplyr_0.5.0     reshape2_1.4.2  cowplot_0.7.0
## [5] ggplot2_2.2.1
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_0.12.11     knitr_1.16       magrittr_1.5     munsell_0.4.3
##  [5] colorspace_1.3-2 R6_2.2.1         rlang_0.1.1      highr_0.6
##  [9] stringr_1.2.0    plyr_1.8.4       tools_3.4.0      grid_3.4.0
## [13] gtable_0.2.0     DBI_0.6-1        htmltools_0.3.6  assertthat_0.2.0
## [17] yaml_2.1.14      lazyeval_0.2.0   rprojroot_1.2    digest_0.6.12
## [21] tibble_1.3.1     evaluate_0.10    rmarkdown_1.5    stringi_1.1.5
## [25] compiler_3.4.0   scales_0.4.1     backports_1.1.0
```