

# Assignment 2 group 6

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

May 15, 2017

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

## Part 1

(a) What are the first five genomic nucleotides from the first exon of this transcript?

AAAGG

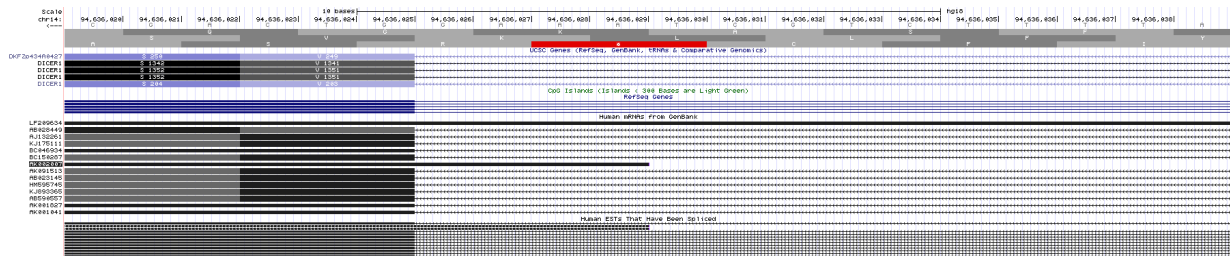
The DICER1 mRNA molecule should have the same sequence as the DNA genomic sequences in the sense strand (substituting Ts by Us). As AK002007 is transcribed on the reverse strand and the default genomic sequence presented by the browser is the antisense one, we have to reverse it. Therefore, the first five nucleotides of the exon in the AK002007 cDNA are AAAGG.

(b) Look at the raw mRNA sequence of AK002007, from the database it actually comes from. What are the first five nucleotides?

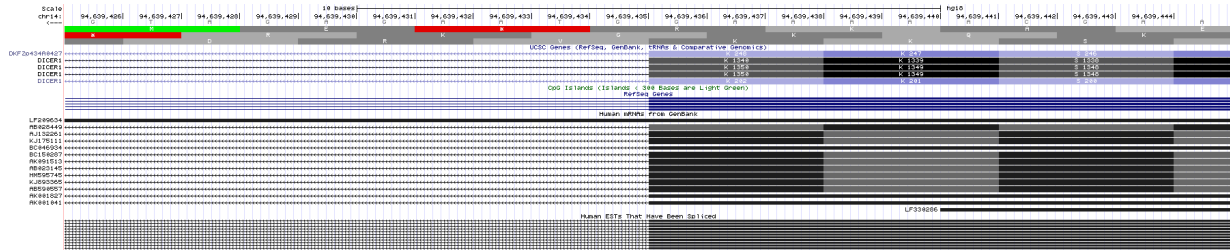
When we check the raw mRNA sequence of AK002007, it can be seen that the first five nucleotides are GAAGC.

(c) How do you explain the discrepancy (maximum 5 lines)?

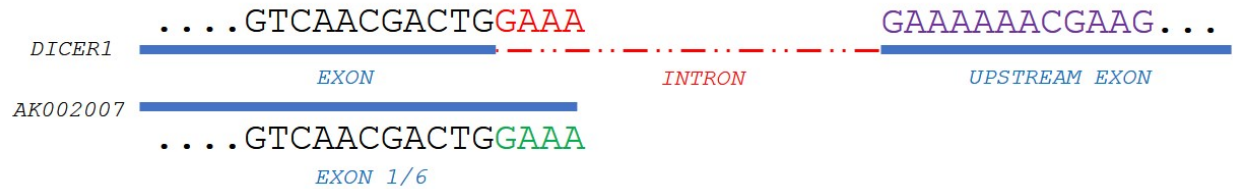
AK002007 is possibly a truncated version of the DICER1 mRNA found in Genbank and refseq. The first 11 nt of AK002007 are found in the last 11 nt of the previous exons in the other mrna sequences. The first 4 nt of these are predicted to be part of the 5' UTR of AK002007 by the aligner, confirming a misalignment. Otherwise, those 11 nt would be aligned as a separate exon in alignment with the other mrna sequences.



**Figure 1:** Screenshot of the start of AK002007. The first 7 letters are discarded by the aligner, while the following 4 are aligned to the end of the intron.



**Figure 2:** Screenshot of the upstream exon. As we can see, the last letters of the exon match the letters aligned to the intron.



**Figure 3:** Shows the mRNA sequence of AK002007 and the general DICER1. These mRNA sequences are read from the right to the left, because they are transcribed from the minus strand. Remarkably, the final 4 letters of the upstream exon and the intron are the exact same GAAA

## Part 2

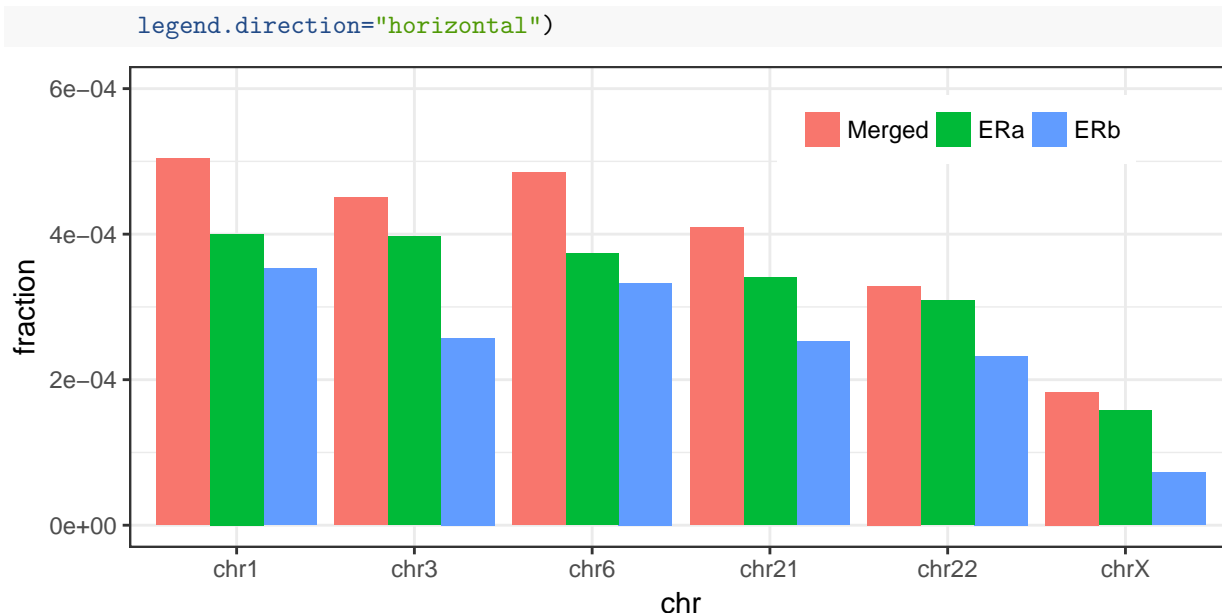
(a)

Bedtools commands to preprocess data before R

```
cat Era_hg18.bed ERb_hg18.bed > full.bed
bedtools sort -i full.bed > full.sorted.bed
bedtools merge -i full.sorted.bed > merged.bed
# Calculates the percent coverage by chromosome
bedtools genomecov -i merged.bed -g hg18_chrom_sizes.txt > coverage.txt
bedtools genomecov -i Era_hg18.bed -g hg18_chrom_sizes.txt > ERa.txt
bedtools genomecov -i ERb_hg18.bed -g hg18_chrom_sizes.txt > ERb.txt
```

R

```
df <- rbind(cbind(read.table(file = "coverage.txt"), protein = "Merged"),
            cbind(read.table(file = "ERa.txt"), protein = "ERa"),
            cbind(read.table(file = "ERb.txt"), protein = "ERb"))
colnames(df) <- c("chr", "bit", "start", "end", "fraction", "protein")
df$chr <- factor(as.character(df$chr),
                levels = c(paste("chr", c(as.character(1:22), "X", "Y", "M"), sep = ""), "genome"))
sorted.index <- order(as.numeric(substring(df$chr, 4)))
df <- df[sorted.index,]
ggplot(data = df[(df$bit == 1 & df$chr != "genome"),],
        mapping = aes(x = chr, y = fraction, fill = protein)) +
  geom_bar(stat = "identity", position = "dodge") +
  scale_y_continuous(limits = c(0, 6e-4)) +
  theme(legend.title = element_blank(),
        legend.position = c(0.75, 0.87),
        legend.key.size = unit(0.5, "cm"),
```



**Figure 4.** Barplot of the coverage across the genome for ERA, ERb and the intervals resulting for a total merge of both.

We also considered running a Wilcoxon test on the distribution of the fractions to determine whether there is a significant differences between the coverages of ERA and ERb, given an apparent decline in coverage in ERb in relation to ERA, but there was not ( $p > 0.05$ ).

```
wilcox.test(df[df$bit == 1 & df$chr != "genome" & df$protein == "ERA", 5], df[df$bit == 1 & df$chr != "g

##
## Wilcoxon rank sum test
##
## data: df[df$bit == 1 & df$chr != "genome" & df$protein == "ERA", 5] and df[df$bit == 1 & df$chr !=
## W = 28, p-value = 0.132
## alternative hypothesis: true location shift is not equal to 0
```

The most noticeable result is that the intervals highly overlap. This is not surprising at all once we notice that both proteins are known to interact forming heterodimers. Moreover, remarkably not just the absolute size of the intervals, but that the coverage decreases with chromosome length. This way, the shorter the chromosome, the smaller the fraction of the chromosome covered by the intervals. Also chromosome X coverage is notably shorter than the rest possibly due to only one chromosome being available regardless of the gender (males only have one X and women have 2 but one of them is inactive as a Barr body).

(b)

Bedtools commands to preprocess data before R

```
# Calculates number of interval overlaps between ERA and ERb
# Reports multiple overlaps as a single occurrence
bedtools intersect -a ERA_hg18.bed -b ERb_hg18.bed -c > AtoBoverlap.bed
bedtools intersect -a ERb_hg18.bed -b ERA_hg18.bed -c > BtoAoverlap.bed
```

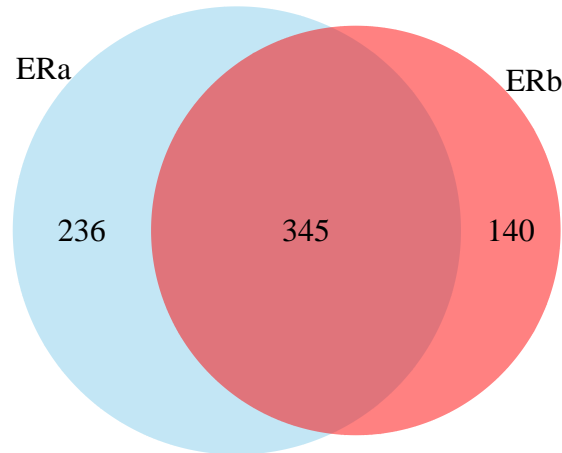
R

```
df2 <- read.table(file = "AtoBoverlap.bed")
df3 <- read.table(file = "BtoAoverlap.bed")
```

```

overlap <- sum(df3$V4)
draw.pairwise.venn(nrow(df2), nrow(df3),
  cross.area = overlap,
  category = c("ERa", "ERb"),
  lty = rep("blank", 2),
  fill = c("skyblue", "red"),
  alpha = rep(0.5, 2))

```



```
## (polygon[GRID.polygon.60], polygon[GRID.polygon.61], polygon[GRID.polygon.62], polygon[GRID.polygon.63])
```

**Figure 5.** Venn diagram of the *ERa* and *Erb* interval sets, showing in the cross area how many of them overlap at least one base.

### Part 3.

The region chr9:24,851,809-24,851,889 from mm8 is a pseudogene of *rpl41*. There are other 5 exact copies in the mouse genome apart from the main *rpl41* gene and several other copies with less identity (around 25 in total). They are 81.5 % identical to one subset of NM\_001014551, which is the *rpl41* gene in the fly. More concretely, this subset of the fly *rpl41* spans the 2 first exons of the gene (out of 3, and not completely). The reason why blating the fly sequence in the fly genome returns sth different from the fly seq in the mouse genome is that the identity is in the exons only, and just a part of them (not the whole cds). Therefore, the fly sequences comprises 3 full exons and 2 introns, and not just a subset of the first 2 exons.

So the main differences are that I) The mouse version does not have exons. II) It is shorter than the fly cds.

Conclusion: is a fucking pseudogene generated by retrotransposition of the mRNA molecule. Why are there 5 exact copies: most probable it was retrotransposed once and then it was duplicated 5 times generating 5 copies in the genome. In that case the sequences surrounding the pseudogene should also be identical in the 5 copies. It's also possible that the same sequence got retrotransposed 5 times in different places of the genome independently. But there are many copies with varying degrees of conservation, so I wouldn't argue that the conservation is surprising: it's just that there's 5 copies that were somehow generated very recently and they haven't had the time to mutate. This reinforces a) cause if they had been generated independently, they would show varying conservation.

### GOOGLE DOC

**Figure 6.** Captures of UCSC browser of the mouse genome (upper) and fly genome (bottom). Upper browser capture shows the chr9:24,851,809-24,851,889 region in the mouse genome. Bottom browser capture shows the conserved fly mRNA sequence BLATed to the fly genome. This sequence corresponds to the *rpl41* gene, encoding a ribosomal protein. The capture in the mouse genome shows that a 78 bp region in the fly genome aligns to the mouse genome at chr9. This corresponds to a subset of exons 2 and 3 in the fly *rpl41* gene



The fly mRNA sequence, which aligns to the mouse genome at chr9:24,851,809-24,851,889 (defined as “input region”) looks different from the mRNA sequence that we get when we BLAT this to the fly genome because the gene in the fly genome includes the full fly’s `_rpl41_exome` as well as introns (as seen in figures UCSC1 and UCSC2)

Is the mouse gene likely to be real?

We don’t find it likely that the mouse input region is a real gene, based on the following three arguments:

- First of all, we find no support from the UCSC track or the Refseq track that there is any transcription of the input region, which makes us doubt that it represents a real gene.
- Secondly, if we compare the input region/conservation track to the functional `Rpl41` gene in mouse (chr10:127,951,059-127,952,117), it can be seen that the functional gene includes introns and the input region/conservation track does not. Most eukaryotic genes include introns, so this indicates that the input region is not a real functional gene.
- Thirdly, we looked at the fly mRNA sequence in the conservation track from the input region and saw that there are 5 other exact copies of it in the mouse genome. These are 81.5 % identical to the `NM_001014551` gene in the fly. We doubt that it is likely for a gene to have 5 identical and functional copies in a genome. Furthermore, our input region appears to be the complete last exon of the official `rlp41` gene and part of the intronic region as well (FIGURE). It is more likely that these are remnants of an original gene that was once retrotransposed and then duplicated 5 times generating 5 copies in the genome (if they had been generated independently, they would show some variation between them, and they are 100% identical). If these copies are pseudogenes, one would often expect them to have acquired mutations, but it could be that the 5 copies were generated relatively recently and haven’t mutated yet.

Therefore, we conclude that the mouse input region isn’t likely to represent a real gene.

```
sessionInfo()
```

```
## R version 3.4.0 (2017-04-21)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 16.04.2 LTS
##
## Matrix products: default
## BLAS: /usr/lib/openblas-base/libblas.so.3
## LAPACK: /usr/lib/libopenblas-r0.2.18.so
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8      LC_NUMERIC=C
##  [3] LC_TIME=es_ES.UTF-8      LC_COLLATE=en_US.UTF-8
##  [5] LC_MONETARY=es_ES.UTF-8  LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=es_ES.UTF-8     LC_NAME=C
##  [9] LC_ADDRESS=C             LC_TELEPHONE=C
## [11] LC_MEASUREMENT=es_ES.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] grid      stats      graphics  grDevices  utils      datasets  methods
## [8] base
##
## other attached packages:
## [1] VennDiagram_1.6.17  futile.logger_1.4.3  gridExtra_2.2.1
## [4] dplyr_0.5.0         reshape2_1.4.2      cowplot_0.7.0
## [7] ggplot2_2.2.1
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_0.12.10      knitr_1.16        magrittr_1.5
```

## [4] munsell_0.4.3	colorspace_1.3-2	R6_2.2.1
## [7] rlang_0.1.1	stringr_1.2.0	plyr_1.8.4
## [10] tools_3.4.0	gtable_0.2.0	DBI_0.6-1
## [13] lambda.r_1.1.9	htmltools_0.3.6	assertthat_0.2.0
## [16] yaml_2.1.14	lazyeval_0.2.0	rprojroot_1.2
## [19] digest_0.6.12	tibble_1.3.1	futile.options_1.0.0
## [22] evaluate_0.10	rmarkdown_1.5	labeling_0.3
## [25] stringi_1.1.5	compiler_3.4.0	scales_0.4.1
## [28] backports_1.0.5		