

Zebrafish Dataset Practical 2

All of these exercises will be done on the same dataset as yesterday's practicals. Make sure you've read the document that describes the dataset and make sure you've got the four required files (`Amp.counts.tsv`, `Amp.samples.tsv`, `Oxy.counts.tsv` and `Oxy.samples.tsv`) in your home directory. You will also need the significantly differentially expressed gene lists you made in yesterday's practical and also the files that just contain their Ensembl IDs. (If you haven't finished yesterday's practical then please carry on with it, either before or after today's practical.)

1. Use BioMart to get the sequence of the 1000 bp upstream of each transcript of the top 20 (extracted using `head`) most significantly differentially expressed genes. The header information should include the gene Ensembl ID, the gene name, the transcript Ensembl ID, the transcript type and the transcript length.
2. Use `awk` (not `cut`, which doesn't allow you to change the column order) to make a new file that contains the chromosome, start, end and Ensembl ID (in that order) for all the significant genes. Name the file something like `"Amp.sig.bed"` or `"Oxy.sig.bed"`. Congratulations - you've made a BED file. See <https://genome.ucsc.edu/FAQ/FAQformat.html#format1> for more information about the BED format. Try viewing this file in Ensembl by adding it as a custom track. Have a look at the distribution of the genes in the "Whole genome", "Chromosome summary" and "Region in detail" views.
3. Use BioMart to get the human orthologues of all the significant genes. Your output should include the zebrafish Ensembl ID, the zebrafish gene name, the human Ensembl ID, the human gene name, the human orthology type, the percentage identity (both target to query and query to target) and the human orthology confidence. How many of the zebrafish genes have a human orthologue? How many have a high confidence human orthologue?

This question is trickier than it seems and you'll need to know that the `sort` command has a `"-u"` option that stands for "unique" and will remove any duplicates. Your solution will need to use `cut`, `grep`, `cut` (again) and `sort -u` (in that order).

4. Choose one of your significant genes and go to its Gene page on Ensembl. Click on "External references" in the left-hand menu and then click on the Expression Atlas link. The initial view in Expression Atlas isn't very useful because the stages are ordered alphabetically and not developmentally. If you click on "18 White et al" then you'll get a much more useful view. How does the expression of the gene change over development? Note the option above the heatmap for viewing a boxplot. Add the other 20 most significant genes via the box at the left-hand side. Are there any similarities in the developmental expression of the genes?
5. In the "Region in detail" view, go to "Configure this page" and add the merged RNA-seq models, including the merged intron-spanning reads. Can you find any evidence for alternative splicing in any of the significant genes? <http://www.ensembl.org/Help/Faq?id=472> might be useful.

6. Yesterday Sam mentioned the improved annotation from the Lawson lab. It's described in this paper:

Lawson et al. (2020) "An improved zebrafish transcriptome annotation for sensitive and comprehensive detection of cell type-specific genes" <https://elifesciences.org/articles/55792>

And also described here: <https://www.umassmed.edu/lawson-lab/reagents/zebrafish-transcriptome/>

Try adding <https://www.umassmed.edu/globalassets/lawson-lab/downloadfiles/v4.3.2.gtf> as a custom track. Warning: the file is very large and so Ensembl will be quite slow. Make sure you disable or delete the track when you're done.

Have a look at the Lawson annotation for a selection of the significantly differentially expressed genes. How does it differ from the Ensembl annotation? Can you spot any issues with the Lawson annotation? How does it compare to RefSeq annotation? You'll need to turn on the RefSeq track.

(RefSeq is NCBI's annotated and curated database of reference sequences, including transcripts and proteins. Accessions starting with X are predictions from automatic genome annotation. Those starting with N come from manual annotation.)

7. Now, we're going to take some data from BioMart and plot it in R.

Use BioMart to get all the zebrafish genes annotated to the Gene Ontology (GO) term `GO:0002088`. We'll cover Gene Ontology (GO) terms tomorrow. This particular GO id is for the term 'lens development in camera-type eye'.

The output should include the zebrafish Ensembl ID, Gene name, Chromosome/scaffold name, Gene start, Gene End and Strand. Download the data as a tab-separated values file and load into R.

The column names have spaces and other characters that R doesn't like such as '('. Use the `col_names =` argument to `read_tsv` to set the column names to `c('Gene', 'Name', 'Chr', 'Start', 'End', 'Strand')`

You'll also need to skip over the header row of the file with the `skip = 1` argument

We're going to plot the data by chromosomal location. So we want to filter out any non-numeric chromosomes to simplify this. Use `filter` on the data along with the `%in%` operator and the vector `1:25` to filter the `Chr` column.

`%in%` returns TRUE if the value in the vector on the left-hand side is in the vector on the right-hand side. For example:

```
# letters is a vector of the 26 lowercase letters
# letters[1:6] is the first six letters
# so 'a' and 'd' in this vector evaluate to TRUE
# everything else is FALSE
c('a', 'z', 't', 'd', NA, 'tyu') %in% letters[1:6]
```

```
## [1] TRUE FALSE FALSE TRUE FALSE FALSE
```

Pipe the result of `filter` into `mutate` and make `Chr` a factor with levels `1:25`.

Remember to assign the new `data.frame` created by `mutate` back to the data object.

Plot the data with `Chr` on the x-axis and `Start` on the y. Also use `shape = 21`. Make sure you put `shape = 21` outside of the `aes ()` function.

You may notice that there is a region with multiple points plotted on top of each other. We can make this clearer by using `geom_jitter` to move the points a small random amount along the x-axis so they are more obvious. `geom_jitter` has arguments `width` and `height` to control how much jitter there is and in which direction. The default values for `width` and `height` are 40% of the resolution of the data. The resolution is defined as the smallest non-zero distance between adjacent values in the data.

Play around with different values to see how they affect things.

You may notice other regions of the genome that have more than one gene in a short space.