

Module 5.2: Learn - Coding

Overview

In this section, we will be using results generated by the whole class to look across a range of trimming parameter combinations. Code to get you started is in:

```
Week_5/FullComparison.Rmd
```

We expect that you will add to this code to perform specific analyses based on the results that you get. We don't guarantee that it will be the most optimized code, but the code is meant to give you a few examples of how to pull out data in R. We want you to try new techniques and visualizations. Don't be afraid to ask for help if you run into trouble programming what you want to do.

Just like you saw in Module 4, for the development of this template code Rmd, we generated fake results to stand in for trimming results that your class was going to generate. To make sure you understand how this code works, run it once using the fake result files in the Week_5 subdirectory in your home directory:

```
/home/<ASUrite_id>/CURE_2022_Scripts
```

Once you understand how the upset plot and other variables and figures are generated, you can change the path to the directory of trimmed data we have collected from your turned in assignments.

Path to differential gene expression results from trimmed data sets

[*SetDirectory* chunk]

The instructors will collect results from differential expression analysis of trimmed data sets and put them here for you to use:

```
/data/mwilsons/GenomicsResearchExperience/Placenta_SexDiff/female_vs_male/
```

The differential expression pipeline () writes out the female versus male statistics/metrics for all expressed genes

```
untrimmed_dge_femalevs_male_all_expressed_genes.csv
```

We will have the same output from the trimmed data sets so you will see the name of the trimming parameter directly in:

```
female_vs_male/
```

For example, this file:

```
trimq10_minlen10_dge_femalevs_male_all_expressed_genes.csv
```

represents the differential expression pipeline output for the data set trimmed with `trimq` set to 10 and `minlen`

Load differentially expressed genes [*Untrimmeddeg* and *Trimmeddeg* chunk]

In this chunk, you will load the tables for all expressed genes that are given as results from the differential expression filter these to get differentially expressed gene lists with our threshold of adjusted p-value < 0.05 and separate into female and higher expression in male just as we did in the previous module. All of this data will be stored in a list that we can easily look through all our data at once when it is helpful to do so. We will store all of the differential expression results in as a nested list so that we can pass these lists as input to the function that creates an upset plot.

Make an upset plot [*UpsetPlot* chunk]

This chunk simply passes the list of lists we just created into the `upset` function in the `UpSetR` package to create our differentially expressed gene lists. It makes one for all of the differentially expressed genes and then two separate plots for higher expression in females and higher expression in males. From here, you can start to interpret how trimming affects the differentially expressed genes identified. You can see which conditions give you the most identified genes, most overlap in identified genes, which tend to give unique results.


Get overlapping (intersection) gene lists [*Membership* chunk]

Once you see the larger pattern of overlap, you might want to list the genes that are giving you the amounts you graph portion of the upset plot. Sadly there is no function within the UpSetR package to do this, but we can get the lists as we have done in previous modules. The comments help you to work through an example in the fake results understanding to probe the real results.

Plotting features of differentially expressed genes that are affected by trimming parameters [*CharacterizingGenes* chunk]

Just as we did after overlap analysis in Module 4, we might like to plot features of differentially expressed genes in the data sets to compare with the untrimmed data case. As an example, we have provided code in this chunk to look at a specific set of intersecting genes across the untrimmed and (fake) trimmed data sets. We use the same visualization graph, but you can use parts and pieces of this code to look at other features and other visualizations. Please build any other coding solutions you learn from your own abilities and creativity, other modules, and internet searches.

Module 5.2 Additional Resources

- [Paper presenting different ways to view differential gene expression results if you are curious](https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-019-2968-1) 
(<https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-019-2968-1>)