Exercise 2-2: Post-assembly analysis

- **Some steps in this exercise could take several hours. We will skip these steps during the hands-on session. These steps are marked with "DO IT AT HOME". You can find results of this exercise in the directory /shared_data/Trinity_workshop_2018/part2_results/ . During the hands-on session, we will do Part 5 and 6 only.**
- **During the workshop, the CPU (threads) were set at 2 for each command. When analyzing your real data, you should set threads based on the the CPU of the server you are using.**

In this exercise, you will not run Trinity assembly. Instead you will focus on post-assembly data analysis. You are provided with a Trinity assembly result file Trinity.fasta, together with RNA-seq data from four samples: tis1rep1, tis1rep2, tis2rep1 and tis2rep2 (Source: NCBI SRA SRP021207, species *Drosophila yakuba*). These are paired-end sequencing data, so there are two "fastq.gz" files per sample. The four samples are from two different tissues (tis1 and tis2), with two biological replicates for each tissue (rep1 and rep2).

Part 1. Prepare the work directory.

1. **(DO IT AT HOME)** Create a work directory and copy all data files required for this workshop into the work directory.

```
mkdir /workdir/$USER
cd /workdir/$USER
cp /shared_data/Trinity_workshop_2018/part2/* ./
```

Part 2. Evaluate the assembly with BUSCO

1. **(DO IT AT HOME)** Run BUSCO to evaluate the assembly.

This test tells you what percentage of genes are missing in your assembly. Instructions of running BUSCO is also available on BioHPC software page: https://biohpc.cornell.edu/lab/userguide.aspx?a=software&i=255#c

BUSCO could take hours. Run it in "screen".

```
cd /workdir/$USER

cp /programs/busco-3.1.0/config/config.ini ./
export BUSCO_CONFIG_FILE=/workdir/$USER/config.ini
export PYTHONPATH=/programs/busco-3.1.0/lib/python3.6/site-packages
export PATH=/programs/busco-3.1.0/scripts:$PATH

tar xvfz insecta_odb9.tar.gz

run_BUSCO.py --in ./Trinity.fasta --lineage_path ./insecta_odb9 --mode
transcriptome --out trinityBUSCO --cpu 2

generate_plot.py -wd run_trinityBUSCO
```

After it is done, you should see a new directory /workdir/$USER/run_trinityBUSCO/. In this directory, there is a summary text file and an image file in .png format. You can use FileZilla to download these files and open them in your laptop.

- short_summary_trinityBUSCO.txt
- busco_figure.png

Note: In this exercise, you will use the BUSCO lineage-specific database file insecta_odb9.tar.gz, which is the closest to *Drosophila*. If you work on other species, the database file can be downloaded from BUSCO web site: https://busco.ezlab.org/ .

Part 3. Evaluate assembled transcript by comparing with known proteins

(DO IT AT HOME) The Trinity package provides a tool **analyze_blastPlus_topHit_coverage.pl** to evaluate the assembled transcripts by comparing them with known proteins. This test would tell you what percentage of the transcripts in your assembly are full length or close to full length. In this example, we will compare the assembly with the annotated *Drosophila melanogaster* proteins. A fasta file of all *melanogaster* proteins (**Drosophila_melanogaster.BDGP5.pep.all.fa**) is included among the exercise data files. If there is no closely related species, you can also use the **Uniprot** sequences for evaluation. (ftp://ftp.uniprot.org/pub/databases/uniprot/current_release/knowledgebase/complete/uniprot_sprot.fasta.gz)

Use "screen":

```
makeblastdb -in Drosophila_melanogaster.BDGP5.pep.all.fa -dbtype prot

blastx -query Trinity.fasta \
-db Drosophila_melanogaster.BDGP5.pep.all.fa \
-out blastx.outfmt6 -evalue 1e-20 -num_threads 2 \
-max_target_seqs 1 -outfmt 6

export TRINITY_HOME=/programs/trinityrnaseq-v2.8.6

$TRINITY_HOME/util/analyze_blastPlus_topHit_coverage.pl \
blastx.outfmt6 Trinity.fasta Drosophila_melanogaster.BDGP5.pep.all.fa
```

The three commands executed by the script are:

1. **makeblastdb**: create a blast database using the D. melanogaster protein sequences;
2. **blastx**: run blastx against the *D. melanogaster* protein database;
3. **analyze_blastPlus_topHit_coverage.pl**: summarize the blast results, and check the how many full length proteins are covered in the assembly.

The output is the file **blastx.outfmt6.hist**. The interpretation of this file can be found at https://github.com/trinityrnaseq/trinityrnaseq/wiki/Counting-Full-Length-Trinity-Transcripts.

Part 4. Abundance Estimation using RSEM.

1. **(DO IT AT HOME)** Create the following shell script. You can create this file with **Notepad++** (Windows) or **BBEdit** (Mac), and save the script as **run_rsem.sh**. Upload the files to the directory /workdir/$USER. The explanation of this shell script is in the note below.

```
export TRINITY_HOME=/programs/trinityrnaseq-v2.8.6

$TRINITY_HOME/util/align_and_estimate_abundance.pl \
--transcripts Trinity.fasta --est_method RSEM \
--aln_method bowtie2 --prep_reference

$TRINITY_HOME/util/align_and_estimate_abundance.pl \
--thread_count 2 \
--aln_method bowtie2 \
--transcripts Trinity.fasta --seqType fq \
--est_method RSEM --SS_lib_type RF \
--trinity_mode --samples_file mysamples
```

2. **(DO IT AT HOME)** This step could take several hours. Run it in "screen" session. Run the shell script in screen.

```
sh run_rsem.sh
```

3. **(DO IT AT HOME)** After it is done, you would find one new directory per sample: tissue1_rep1, tissue1_rep2, tissue2_rep1, tissue1_rep2. Within each directory, there is a file RSEM.genes.results. The "expected_count" column is the gene level count. Use the following command to combine all samples into one data table. Use the parameter "cross_sample_norm none" to specify that no-normalization will be performed at this step. The result file name is: mystudy.isoform.counts.matrix.

```
$TRINITY_HOME/util/abundance_estimates_to_matrix.pl --est_method RSEM \
--gene_trans_map none \
--cross_sample_norm none \
--out_prefix mystudy \
--name_sample_by_basedir \
   tis1rep1/RSEM.genes.results \
   tis1rep2/RSEM.genes.results \
   tis2rep1/RSEM.genes.results \
   tis2rep2/RSEM.genes.results
```

Note:

a) The first command (**export**) is to set up a Linux environment variable **TRINITY_HOME**. After it is set, you can use **$TRINITY_HOME** to replace the string **/programs/trinityrnaseq-v2.8.6**.

b) The first command (align_and_estimate_abundance.pl --prep_reference) in this script will index the transcriptome sequence file **Trinity.fasta**, which is the assembled transcriptome and serves as reference for the transcript quantification. After indexing is done, fastq files from each sample can be aligned to the reference transcriptome.

c) The second command (align_and_estimate_abundance.pl) would run **bowtie2** to align reads from each sample to the reference, and run RSEM to quantify read counts for each gene/isoform. The relationship between sample and fastq data file names is specified in the file mysamples. The file format of the sample file is defined in the web page: https://github.com/trinityrnaseq/trinityrnaseq/wiki/Trinity-Transcript-Quantification. The "--trinity_mode" parameter specifies that the assembly is done through Trinity, so that the sequence title in the fasta file is in Trinity format, and represent the gene-to-isoform relationship.

Part 5. Sample QC.

For details, read this page: https://github.com/trinityrnaseq/trinityrnaseq/wiki/QC-Samples-and-Biological-Replicates

As Part 4 takes several hours to finish, you will do Part 4 at home. The result from Part 4 is a read count matrix file from 4 RNA-seq samples. You can copy this file (**mystudy.isoform.counts.matrix**) from the directory "/shared_data/Trinity_workshop_2018/part2_results/". You will also need a text file "mysamples", which provides the sample information.

```
cd /workdir/$USER/

cp /shared_data/Trinity_workshop_2018/part2_results/mysamples ./

cp
/shared_data/Trinity_workshop_2018/part2_results/mystudy.isoform.counts.matrix
./

cat mysamples
```

The Trinity package provides many scripts to process the read count matrix. We will use some of them.

1. Compare samples within replicate group.

   The command "export PATH=/programs/R-3.5.0s/bin:$PATH" change the path of the default R programs. As the default R program has some issues with R packages called by these scripts.

```
export TRINITY_HOME=/programs/trinityrnaseq-v2.8.6
export PATH=/programs/R-3.5.0s/bin:$PATH

$TRINITY_HOME/Analysis/DifferentialExpression/PtR \
  --matrix mystudy.isoform.counts.matrix \
  --samples mysamples \
  --log2 \
  --min_rowSums 10 \
  --compare_replicates
```

The output is one PDF file per condition group. You will find two PDF files, one for tissue 1, and one for tissue 2. Each PDF file have multiple pages.

2. Compare samples between replicate groups

```
$TRINITY_HOME/Analysis/DifferentialExpression/PtR \
          --matrix mystudy.isoform.counts.matrix \
          --min_rowSums 10 \
          -s mysamples --log2 --CPM --sample_cor_matrix
```

Two output files:

mystudy.isoform.counts.matrix.minRow10.CPM.log2.sample_cor_matrix.pdf: plots

mystudy.isoform.counts.matrix.minRow10.CPM.log2.sample_cor.dat: data used for generating plots.

3. PCA plot

```
$TRINITY_HOME/Analysis/DifferentialExpression/PtR \
    --matrix mystudy.isoform.counts.matrix \
    -s mysamples --min_rowSums 10 --log2 \
    --CPM --center_rows \
    --prin_comp 2
```

Output file:

mystudy.isoform.counts.matrix.minRow10.CPM.log2.centered.prcomp.principal_components.pdf

Part 6. Differentially expressed genes

For details, read this page: https://github.com/trinityrnaseq/trinityrnaseq/wiki/Trinity-Differential-Expression

1. The Trinity package provides a script to identify differentially expressed genes.

```
$TRINITY_HOME/Analysis/DifferentialExpression/run_DE_analysis.pl \
          --matrix mystudy.isoform.counts.matrix \
          --method voom \
          --samples_file mysamples
```

Output:

A new directory: voom.*.dir. Within this directory you will find two files:

- mystudy.isoform.counts.matrix.tis1_vs_tis2.voom.DE_results: a table with differentially expressed genes
- mystudy.isoform.counts.matrix.tis1_vs_tis2.voom.DE_results.MA_n_Volcano.pdf: volcano plot

2. Extracting and clustering differentially expressed transcripts.

   Two filtering criteria:

   - FDR < 0.05
   - log(FoldChange) > 1 or <-1 (fold change greater than 2)

```
cd voom.*.dir

$TRINITY_HOME/Analysis/DifferentialExpression/analyze_diff_expr.pl \
  --matrix ../mystudy.isoform.counts.matrix \
  -s ../mysamples \
  -P 0.05 -C 2
```

Output: two pdf files with heatmap of the clustering results:

- diffExpr.P0.05_C2.matrix.log2.centered.sample_cor_matrix.pdf

- diffExpr.P0.05_C2.matrix.log2.centered.genes_vs_samples_heatmap.pdf

3. Partitioning Genes into Expression Clusters

```
$TRINITY_HOME/Analysis/DifferentialExpression/define_clusters_by_cutting_tree.pl
-R  diffExpr.P0.05_C2.matrix.RData --Ptree 60
```

Output:

A new directory diffExpr.P0.05_C2.matrix.RData.clusters_fixed_P_60. Files:

my_cluster_plots.pdf

subcluster_2_log2_medianCentered_fpkm.matrix (1 file per cluster)