Apr 26, 2024 Version 1

RNA-Seq protocols for non-model species V.1

This protocol is a draft, published without a DOI.

Chase Donnelly[1]

[1]University of Antwerpen

Chase

Chase Donnelly

University of Antwerpen

Protocol Citation: Chase Donnelly 2024. RNA-Seq protocols for non-model species . **protocols.io** **https://protocols.io/view/rna-seq-protocols-for-non-model-species-dcbm2sk6**

Protocol status: In development
We are still developing and optimizing this protocol

Created: April 18, 2024

Last Modified: April 26, 2024

Protocol Integer ID: 98381

Keywords: RNA-Seq

Disclaimer

This is still under development and not final

Abstract

This is a protocol under development for RNA seq analysis on non model plant species.

RNA Extraction

1 RNA Extraction followed a modified protocols from RNeasy plant mini kit (Qiagen), below we share only the modifications used to extract high qualilty RNA from difficult plant species. https://www.qiagen.com/us/products/discovery-and-translational-research/dna-rna-purification/rna-purification/total-rna/rneasy-plant-mini-kit

2 Note: Amount of Starting Material: the amount of starting material to use can vary between species, we found that for more troublesome species containing a lot of secondary metabolites, such as *Plantago*, using less starting (~30-50mg) material resulted in higher quality RNA output.

3 Note: Grind with mortar and pestle: while instruments to disrupt tissue can work, we found that grinding for an extended period by hand in liquid nitrogen produced higher quality results for RNA extractions on non model plants

4 Note: When adding 450 ul of RLT Buffer perform a 3 min incubation before vortexing, proper mixing here is key to a successful extraction

5 Modification: The wash step with RW1 was performed twice on difficult samples, this helped remove excess metabolites that were still present in plant samples

6 Modification: The wash step with RPE was performed a minimum of three times but more washes were performed if color of the sample was still green or brown. The final sample should be a clear liquid.

Software Setup

7 After sequencing the first step is to set up a clean compute space to analyze the sequencing data to avoid issues with mixing versions of software.

8 There are a few different ways to manage a compute environment, here conda/mamba was used to create a clean environment as it works with all of the downstream programs and has an extensive manual. Mamba is a newer alternative to conda that can install programs with greater speed.

conda doc: https://conda.io/projects/conda/en/latest/user-guide/getting-started.html

mamba doc: **https://mamba.readthedocs.io/en/latest/**

mamba and conda can both be used in the same way, just replace mamba with conda in the codes below.

9 To set up an environment with conda or mamba you can use the following code to create an environment and install the required programs:

```
#First create environment with mamba
mamba create -n rnaseq_env
#Next activate our new environment
mamba activate rnaseq_env
#Install tools needed in future analysis
mamba install fastqc
mamba install trinity
```

Data Quality Assessment

10 When obtaining data from a sequencing facility, it is essential to check the quality of data received for each sample and remove any unwanted (see adaptors) or low-quality reads. FastQC is the standard QC tool for sequencing experiments and other options are mainly command-line-only options that require
multiple steps or tools (Lo and Chain 2014).

Fast qc can be ran through command line or by downloading their GUI interface here:
https://www.bioinformatics.babraham.ac.uk/projects/fastqc/

Read quality tends to decrease as it progresses
to tail ends of the reads, and FastQC should be used to check the quality at
the read tail. If the read quality is low or adapters are still present, then
the reads should be removed through one of multiple available trimming tools

11 Low quality reads should be removed by one of the many available read tools (Table 2), below is an example of how this is done with Trimmomatic (A. M. Bolger, Lohse, and Usadel 2014).

	Tool	Link
	trimmomatic	http://www.usadellab.org/cms/?page=trimmomatic
	cutadapt	https://cutadapt.readthedocs.io/en/stable/
	fastp	https://github.com/OpenGene/fastp
	AdapterRemoval	https://adapterremoval.readthedocs.io/en/stable/

Table 2: : List of possible tools for trimming data.

```
#First Install the program into our env
mamba install bioconda::trimmomatic
#Next we can run trimmomatic setting should all be set around the
parameters found in fastQC results, the ones below can generally
be used, replace all <> with the names of your files
trimmomatic PE -threads <set bases on machine>
<input_forward.fq.gz> <input_reverse.fq.gz>
<output_forward_paired.fq.gz> <output_forward_unpaired.fq.gz>
<output_reverse_paired.fq.gz> <output_reverse_unpaired.fq.gz>
ILLUMINACLIP:$TRIMMOMATIC_DIR/adapters/TruSeq3-PE.fa:2:30:10
SLIDINGWINDOW:4:5 LEADING:5 TRAILING:5 MINLEN:25
```

Transcriptome Assembly

12 When no reference genome is available, or the available genome is of low quality, RNA-Seq reads can be assembled into a de novo transcriptome using a variety of tools. A few of the available tools are listed below.

Tool	Link
Trinity	https://github.com/trinityrnaseq/trinityrnaseq/wiki
SOAP	https://github.com/aquaskyline/SOAPdenovo-Trans
Oases	https://github.com/dzerbino/oases
Trans-ABySS	https://github.com/bcgsc/transabyss
rnaSPAdes	https://github.com/ablab/spades

No tool is perfect for every organism, with specific tools performing better or worse depending on the ploidy level and repetitiveness of the reads and so a comprehensive comparison should be done before starting assembly (Madritsch, Burg, and Sehr 2021; Hölzer and Marz 2019).

13 For this protocol we will be using Trinity due to its high performance and its use in previous publications for similar species to be used in this study (Ponnuchamy et al. 2020; Boehm et al., 2017).

14

```
Trinity \
--seqType fq \
--SS_lib_type RF  \
--left left.fq \
--right right.fq \
--CPU n \ #change n to the number of cpus in use
--max_memory 220G \
--trimmomatic #use this option to run trimmomatic with trinity, can
be removed if trimming was completed already
```

Transcript Quality Assessment

15 Once the de novo assembly is completed, the next step is to check assembly quality. The quality of an assembly will affect multiple downstream analysis such as differential expression analysis, gene structure prediction, and isoform identification (Raghavan et al. 2022). Multiple tools and methods were researched to find the best practices for quality assessment, but no universal protocol is currently used to allow for comparison across studies (Haas et al. 2013; Xie et al. 2014).

A large percentage (70-90%) of a high-quality transcriptome will also map back to the reads, and this is an important step before proceeding with downstream analysis. For this protocol multiple ways to characterize the quality assembly were tested, starting with aligning reads to the assembly to count the number of proper and improper pairs.

Tool	Link
STAR	https://github.com/alexdobin/STAR
Bowtie2	https://github.com/BenLangmead/bowtie2
BWA	https://github.com/lh3/bwa
HISAT2	http://daehwankimlab.github.io/hisat2/
TopHat2	https://github.com/infphilo/tophat

Table 3: List of tools that can be used to map transcripts back to sequencing reads and links to their repositories.

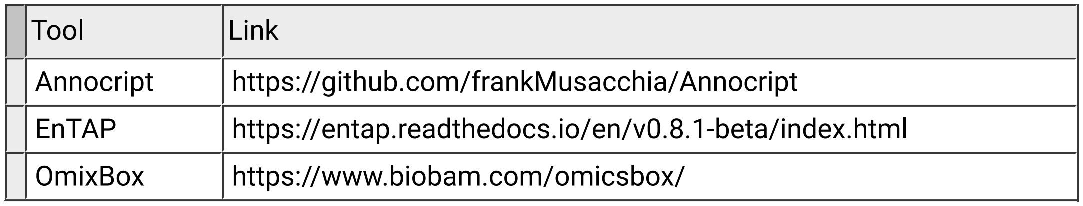
The code for QA can be found below

```
#Check BUSCO scores
#install BUSCO
mamba install -c conda-forge -c bioconda busco=5.7.1 #set lastest
version here
#run BUSCO ; set linage for your species, threads, and name of input
and output files
busco -l <viridiplantae_odb10> -m transcript -c <n> -i
<your_genome.fasta> -o <your_output>

#map genome back to input reads
#install bowtie2
mamba install bioconda::bowtie2
# first build index based on assembled transcriptome
bowtie2 -build <input.fasta> <output.fasta>
#perform read alignment
bowtie2 -p 10 -q --no-unal -k 20 -x Trinity.fasta -1 reads_1.fq -2
reads_2.fq  \
     2>align_stats.txt| samtools view -@10 -Sb -o bowtie2.bam
#combine stats
cat 2>&1 align_stats.txt

#run rnaQUAST
#install
mamba install -c bioconda rnaquast
#run
rnaQUAST \
--transcripts /PATH/TO/transcripts1.fasta
/PATH/TO/ANOTHER/transcripts2.fasta /PATH/TO/MULTIPLE/*.fasta [...] \
--reference /PATH/TO/reference_genome.fasta --gtf
/PATH/TO/gene_coordinates.gtf
#gtf file can be obtained from code that is available in downstream
analysis.
```

Trascriptome Annotation

16 After transcriptome assembly and quality assessment, the putative functionality of the contained sequences can be elucidated. Most tools follow a similar procedure comprised of three steps: homology transfer and assignment via a search algorithm, sequence feature annotation, and gene ontology assignment.

Tool	Link
Trinotate	https://github.com/Trinotate/Trinotate/releases



Tool	Link
Annocript	https://github.com/frankMusacchia/Annocript
EnTAP	https://entap.readthedocs.io/en/v0.8.1-beta/index.html
OmixBox	https://www.biobam.com/omicsbox/

Table 4: List of tools that can be used for functional annotation and links to their repositories.

This protocol will use trinotate as it works directly from trinity, used above for de novo transcriptome assembly, the code to run trinotate is below.

```
#First we run transdecoder to determine likely coding regions
#install
mamba install transdecoder

#run transdecoder
/Pathtotransdecoder/TransDecoder.LongOrfs -t trinity.fasta
/Pathtotransdecoder/TransDecoder.Predict -t trinity.fasta

# the output that we want from this is the transdecoder.pep file
# we can also create a gtf/gff3 file with transdecoder if
needed with gff3_gene_to_gtf_format.pl

# Next we will start the annotation process, this will be done in
trinotate and can be run in parallel with the DE analysis

#make sure environement is activate and install
mamba activate trinityenv
mamba install trinotate

#compile the sequence database, this only needs to be done once
/Path_to_trinotate/Build_Trinotate_Boilerplate_SQLite_db.pl Trinotate

#once completed you will have the following files
Trinotate.sqlite
uniprot_sprot.pep
Pfam-A-hmm.gz

#prepare the protein database for blast searches, set working directory
to file location
makeblastdb -in uniprot_sprot.pep -dbtype prot

#uncompress and prepare the pfam database
gunzip Pfam-A.hmm.gz
hmmpress Pfam-A.hmm

#next we will blast our fasta to the database to find annotations
#this is best to run on an HPC as it will take up to a few days for the
blastx
blastx -query your_transcriptome.fasta -db uniprot_sprot.pep -num_threads
<n> -max_target_seqs 1 -outfmt 6 -evalue 1e-3 > blastx.outfmt6
blastp -query transdecoder.pep -db uniprot_sprot.pep -num_threads <n>-
max_target_seqs 1 -outfmt 6 -evalue 1e-3 > blastp.outfmt6

#identify protein domains (optional)
#run this on HPC as well if possible
```

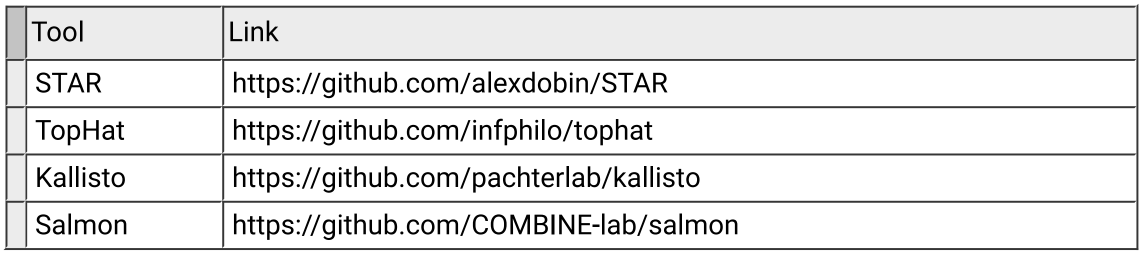
```
hmmscan --cpu 12 --domtblout TrinotatePFAM.out Pfam-A.hmm
trinity.fasta.transdecoder.pep > pfam.log

#load trinotate sqlite database
Trinotate Trinotate.sqlite init --gene_trans_map
Trinity.fasta.gene_trans_map --transcript_fasta Trinity.fasta --
transdecoder_pep trinity.fasta.transdecoder.pep

# load blast homologies
Trinotate Trinotate.sqlite LOAD_swissprot_blastp blastp.outfmt6

Trinotate Trinotate.sqlite LOAD_swissprot_blastx blastx.outfmt6

# load pfam doamain entires
Trinotate Trinotate.sqlite LOAD_pfam TrinotatePFAM.out

#generate output report
Trinotate Trinotate.sqlite report [opts] >
trinotate_annotation_report.xls

#extract go assignments per gene and make into annotations file
/path_to_trinity_install/bin/extract_GO_assignments_from_Trinotate_xls.pl
\
                         --Trinotate_xls trinotate.xls \
                         -G --include_ancestral_terms \
                         > go_annotations.txt

#get a gene lengths file, this can be used for downstream analysis
/path_to_trinity_install/opt/trinity-2.13.2/util/misc/fasta_seq_length.pl
Trinity.fasta > Trinity.fasta.seq_lens

/path_to_trinity_install/opt/trinity-
2.13.2/util/misc/TPM_weighted_gene_length.py  \
         --gene_trans_map
/media/chase/Samsung_T5/Trinotate_Lotus/Trinity.fasta.gene_trans_map \
         --trans_lengths
/media/chase/Samsung_T5/Trinotate_Lotus/Trinity.fasta.seq_lens \
         --TPM_matrix kallisto.isoform.TMM.EXPR.matrix >
Trinity.gene_lengths.txt
```

EnTAP was also performed to compare annotation results across platforms, the code for this is below.

```
#install EnTAP based on the site instructions
(https://entap.readthedocs.io/en/latest/)

#Run database configuration, this can be run for all databases that your
will search against eg: nr, swissprot, etc..
EnTAP --config -d  /Path_to_database_fasta/nr_database.fasta --out-dir
output_folder -t <n>--ini entap_config.ini
#Run EnTAP
EnTAP --runP -i protiens.pep -d /Path_to_database/nr.dmnd -t <n>--ini
Path_to_Entap/EnTAP-v0.10.8-beta/entap_config.ini
```

Transcript Quantification

17 As with the protocols described above, multiple tools are currently available to estimate transcript abundance in the absence of a genome. Recent comparisons of these tools can be found by (Wu et al. 2018; Corchete et al. 2020). These studies found multiple tools perform well and the choice of tool comes down to multiple factors that will differ between experiments.



Tool	Link
STAR	https://github.com/alexdobin/STAR
TopHat	https://github.com/infphilo/tophat
Kallisto	https://github.com/pachterlab/kallisto
Salmon	https://github.com/COMBINE-lab/salmon

Table 5: List of tools that can be used for transcript quantification and links to their repositories.

Kallisto was used for this protocol due to speed and low computational requirments with high accuracy.

```
#install
mamba install kallisto
#create index file
kallisto index -i <index.idx> <transcriptome_file.fasta>

#Can be run with trinity
path_to_trinity/util/align_and_estimate_abundance.pl \
--transcripts transcriptome_file.fasta
--seqType fq \
--left left.fq.gz \
--right right.fq.gz \
--est_method kallisto \
--thread_count <n>\
--gene_trans_map Trinity.fasta.gene_trans_map \
--SS_lib_type RF \
--output_dir kal_out

#create count file for multiple samples with kallisto
path_to_trinity/utils/abundance_estimates_to_matrix.pl \
--est_method kallisto \
--gene_trans_map Trinity.fasta.gene_trans_map \
--out_prefix kallisto \
--name_sample_by_basedir \
--basedir_index -3 \
Sample1/kal_out/abundance.tsv \
Sample2/kal_out/abundance.tsv

#can add all samples, two are given as example
```

If reference is available

18 Some plant species already have a reference genome available, if this is the case, mapping can be done directly to the reference. This can be done with multiple mapping tools (Table 3). A quick example of mapping with STAR is given below.

```
#First create index file
STAR \
--runThreadN <n> \
--runMode genomeGenerate \
--genomeDir Path_to_genomefile/Genome \
--genomeFastaFiles Genome/genome_fasta \
--sjdbGTFfile Genome/genome_gtf \
--sjdbOverhang 99 \
--genomeChrBinNbits 7

#next run mapping
STAR \
--readFilesIn Path_to_forwardread.fastq Path_to_reverseread.fastq
--runThreadN <n> \
--genomeDir Path_to_genomefile/Genome \
--readFilesCommand zcat \
--quantMode GeneCounts \
--outFileNamePrefix ${i/"_Clipped_R1.fastq.gz"/""} \
--sjdbGTFfile $genomedir$genome_gtf  \
--sjdbOverhang 99 \
--outSAMtype BAM SortedByCoordinate
done
```

Differential Expression and GO Analysis

19 Once the transcripts have been quantified, the next step is to investigate the changes in gene expression caused by the experimental stimuli. There are a variety of tools that can be used for differential gene expression analysis with DeSeq2, EdgeR, and Limma currently being the most used (Costa-Silva, Domingues, and Lopes 2017). This protocol uses DeSeq2, which can be run seperately in R or following the trinity analysis protocol.

Tool	Link
DeSeq2	http://bioconductor.org/packages/release/bioc/html/DESeq2.html
edgeR	http://bioconductor.org/packages/release/bioc/html/edgeR.html
Limma	http://bioconductor.org/packages/release/bioc/html/limma.html
ROTS	http://www.btk.fi/research/research-groups/elo/software/rots/

Table 6: List of tools that can be used for differential expression analysis and links to their repositories.

After differential gene analysis, the next step is to place the change in genes to a biological context. GO enrichment analysis is a technique that is widely used to highlight important biological processes in complex outputs of RNA-Seq experiments (Ashburner et al. 2000).

Tool	Link
GSEA	https://www.gsea-msigdb.org/gsea/index.jsp
Panther	https://geneontology.org/
G: Profiler	https://biit.cs.ut.ee/gprofiler/gost
GOseq	https://bioconductor.org/packages/release/bioc/html/goseq.html

Table 7: List of tools that can be used for GO enrichment analysis and links to their repositories.

Example code for DEG and GO enrichment analyisis is included below:

```
# This code is ran following the trinity analysis pipeline, however all
analysis can also be run in R/Rstudio seperately
# setup is based on the sample file and contrasts file, see trinity wiki
for setup
/Path_to_trinityfile/Analysis/DifferentialExpression/analyze_diff_expr.pl
\
--matrix
/media/chase/Samsung_T5/GOA_RNA/Holcus/kallisto.gene.TMM.EXPR.matrix \
-P 0.05 \
-C 1 \
--max_genes_clust 20000 \
--samples /media/chase/Samsung_T5/GOA_RNA/Holcus/sample_file.txt \
--examine_GO_enrichment --GO_annots
/Path_to_annotationfile/go_annotations.txt --gene_lengths
/Path_to_genelengthfile/Trinity.gene_lengths.txt

#In R
library("DESeq2")
#Creating a DESeq2 dataset, here only considering the treatment (control
vs. treated).
dds_treatment <- DESeqDataSetFromMatrix(countData = cts,
                                        colData = metadata_file,
                                        design = ~ Treatment(variable))
#run DeSeq2
dds_treatment <- DESeq(dds_treatment)
# get results
res_trt <- results(dds_treatment)
# Export Results
write.csv(x = as.data.frame(res_trt[order(res_trt$pvalue),]),
          file = "treatment_results.csv")
```