Gene expression and transcript annotation in *Tiliqua adelaidensis* kidney tissue

Carmel M

2022

Analysis Workflow

Gene annotation and seasonal expression in kidney tissue of the Australian pygmy bluetongue skink, *Tiliqua adelaidensis* Supplementary Information "***"

This document outlines the pipeline methods for assembly of poly-a selected transcripts extrracted from kidney tissues of three male and five female Australian pygmy bluetongue skinks collected at the beginning and end of the dry season. Long read transcripts were sequenced from one individual "G6" using Pacific Biosciences' IsoSeq Sequel platform, and short reads for all eight individuals were sequenced on an Illumina HiSeq.

Sample ID	Experimental factors
G1	September Female
G2	September Female
G3	March/April Female
G4	March/April Male
G5	September Male
G6	September Male
G7	March/April Female
G8	March/April Female

There are three main sections to this document, which are referenced in three separate chapters of the thesis:

Assembly (Chapter 4): Assembly, cleaning and clustering of transcript isoforms from **both** sequencing formats to form a pseudo-reference for *T. adelaidensis* and compare methods.

Annotation (Chapter 5): Annotation and further analysis of the full length transcript isoforms derived from the long read analysis.

Gene expression (Chapter 6): Gene expression analysis using the short read data aligned to the long read pseudo reference produced in Chapter 4, to assess seasonal variation of gene expression.

Analysis was done using a combination of computing and HPC machines with specifications as below:

- Long read cleaning, read frame prediction and blast searches were run on the Australian National eResearch Collaboration Tools and Resources project (NECTAR) cloud, using an 'm1.xlarge flavour' allocation with 8 AMD Opteron 63xx class CPUs, 32GB RAM, 10GB root disk, and 240GB of secondary disk storage.

- The ANGEL program which required a higher memory allocation was run on a Dell PowerEdge server with 40 cores and 512G RAM.
- Short read QC, trimming, initial trinity assembly, and alignments were completed using Flinders University's high-performance computer "Deep thought" with 16 x 256Gb (4TB RAM), 1024 AMD x86 CPU cores at 2.0GHz across 14 standard compute and 2 data science nodes, and 100 TiB of usable storage via the Dell EMC Network Storage reference architecture.
- Downstream gene expression analysis and manual manipulation of annotation files was completed on a local Intel core i7 machine using RStudio or in-house bash scripts applied in Git Bash for Windows.

All larger scripts included in this document have also been uploaded to Github separately and are linked here as relevant.
All scripts and commands in this pipeline depend on specific directory structure, and may show inconsistencies in this structure between steps where files have been transferred between computing resources. The input and output files of note are highlighted in the text between steps to bridge these gaps.

Assembly (Chapter 4)

A genome from a closely related species is unavailable for these data. Gene expression analysis of short read data was initially intended to be carried out using a *de-novo* assembly of this same data as a pseudo-reference for expression counts.
When long read data became available both datasets have been assembled *de-novo* and these assemblies compared. The long read assembly then provided the reference for expression counts in Chapter 6.

This section is divided into the two methods of assembly; using the long read IsoSeq data, and using the short read HiSeq data. Both these sections contain titles referencing the main program in use at each step.

Long read data

De-novo assembly of long-read IsoSeq data sequenced from poly-a selected RNA extracted from a single kidney from september male G6. These sequences were be used to annotate genes found in kidney tissue of T. adelaidensis and create a set of reference transcripts for gene expression analysis conducted in later chapters.

Sequences were downloaded from Pacific Biosciences & checksumms verified.

SMRT Tools - IsoSeq3

SMRT Tools provides various programs for appropriate treatment of Pacific Biosciences sequencing outputs. IsoSeq3 was used to collapse circular sequences, initial polishing, and to characterise full length transcript reads. This analysis was conducted in an environment on the eRSA NECTAR research cloud.

A primer reference file was created manually for reference in the Isoseq3 script using the primers listed in the Clontech SMARTer cDNA Library prep kit which was used during the synthesis of cDNA for these samples.

primers.fasta:

```
>primer_5p
AAGCAGTGGTATCAACGCAGAGTACATGGG
>primer_3p
GTACTCTGCGTTGATACCACTGCTT
```

A single script was used to complete the Pacific Biosciences Isoseq3 pipeline isoseq3.sh:

```
#!/bin/bash
#
# This script is used to characterise full length transcripts de novo using
↪  smrttools and bioconda as specified by PacBio workflow
# see: https://github.com/PacificBiosciences/IsoSeq3/blob/master/README_v3.0.md and
↪  follow links for bioconda references
#
# The paths in the script assume that a specific directory structure has been set
↪  up.
# This directory structure is for the Nectar cloud VM

# open a screen
# usage from /mnt
# bash isoseq3.sh <ctrl-a ctrl-d>
#
# Carmel Maher
# April 2019


#---------Installed Programs---------

# smrttools               -> smrttools-release_6.0.0.47835
  # help:
  #
  ↪  /pacbio/smrtlink/install/smrtlink-release_6.0.0.47841/bundles/smrttools/install/smrttools-r
  ↪  -h
  #
  ↪  /pacbio/smrtlink/install/smrtlink-release_6.0.0.47841/bundles/smrttools/install/smrttools-r
  ↪  --version
  # Version: isoseq3 3.0.0 (commit v3.0.0-7-gcc6cddd)

# python 2.7          -> to run miniconda - automatically installed on nectar
# miniconda           -> recommended python 2.7 compatible version by PacBio
↪  workflow. Miniconda2 | VER: 4.6.14
# ccs                    -> installed through conda, inside pbccs package | VER:
↪  pbccs-3.4.1
# lima                   -> installed through conda | VER: lima-1.9.0


#------adjust these for your run------

#as per the local installation, the smrttools program directories are (in /mnt).
↪  Notable directories listed in installation:
SMRT_ROOT=/mnt/pacbio/smrtlink

# Isoseq3 executeable location (from /mnt)
```

```
↪  Iso3_DIR=$SMRT_ROOT/install/smrtlink-release_6.0.0.47841/bundles/smrttools/install/smrttools-

DATA=/mnt/data
MOVIE="MAH6260A1_m54196_190204_223227"
THREADS=0     #0 for the Iseseq3 tools =autodetection of threads and is the default


#-------------------------------------

function error_exit
{
    # Exit function due to fatal error
    # Accepts 1 arg:
    # string - descriptive error message

    echo "${PROGNAME}: ${1:-"Unknown error"}" 1>&2
    exit 1
}

# go to the working data directory
cd $DATA || error_exit "$LINENO: directory error"

# Generate consensus sequnces (ccs file) from raw subread data
# NOTE: this automatically runs with default autodetection of threads =  8
ccs $MOVIE.subreads.bam $MOVIE.ccs.bam --noPolish --minPasses 1 || error_exit
↪  "$LINENO: consensus sequence ccs error"

echo " - - - consensus sequnces generated"

# The primers.fasta as per the recommended Clontech SMARTer cDNA library prep - no
↪  barcodes used for this sample
# primers.fasta has been uploaded to /mnt/data and does not need to be created
↪  here:
      # primers.fasta
      # >primer_5p
      # AAGCAGTGGTATCAACGCAGAGTACATGGG
      # >primer_3p
      # GTACTCTGCGTTGATACCACTGCTT

# Remove primers and demultiplex:
lima --isoseq --dump-clips --no-pbi $MOVIE.ccs.bam primers.fasta demux.bam ||
↪  error_exit "$LINENO: Primer demultiplex lima error"

echo " - - - primers removed and demultiplexed"

             #!#!#!#!#
             # CHECK #  Note: A search using Git bash on the output files for any
             ↪  of the primers listed for Isoseq by Pacific biosciences returns
             ↪  no results after this script
             #!#!#!#!#

# *********************
```

```
# From here on, execute the following steps for each output BAM file. -- only one
↪  in this case.
# isoseq3 tool usage: isoseq3 <tool>
# due to nectar permissions and installation location, usage: $Iso3_DIR/isoseq3
↪  <tool>
# ********************

### cluster Tool: Cluster CCS reads and generate unpolished transcripts.
# recommended to give this step as many cores as possible
# Usage
# isoseq3 cluster [options] input output
# Example
# isoseq3 cluster <two types of potential inputfile> demux.bam <OR>
↪  $MOVIE.consensusreadset.xml  unpolished.bam

# Cluster consensus sequences to generate unpolished transcripts:
# note the demux.bam is named after the headers used in primers.fasta
$Iso3_DIR/isoseq3 cluster --verbose -j $THREADS demux.primer_5p--primer_3p.bam
↪  unpolished.bam || error_exit "$LINENO: Isoseq3 cluster error"

echo " - - - css reads clustered"

### polish Tool: Polish transcripts using subreads.
# Usage
# isoseq3 polish [options] input_1 input_2 output

# Polish transcripts using subreads:
$Iso3_DIR/isoseq3 polish --verbose -j $THREADS unpolished.bam $MOVIE.subreads.bam
↪  polished.bam || error_exit "$LINENO: Isoseq3 polish error"

echo " - - - transcripts polished"

### summarize Tool: Create a .csv-format barcode overview from transcripts.
# Usage
# isoseq3 summarize [options] input output
$Iso3_DIR/isoseq3 summarize --verbose polished.bam summary.csv || error_exit
↪  "$LINENO: Isoseq3 summary error"

echo " - - - summary file created"
echo " - - - done"
```

All of the above output was moved into a folder ~/data/2019-04-30_isoseq3_out

The output from this script, the polished.hq.fasta.gz file was unzipped and then used as the input for the program Cogent.

Cogent

Transcript clustering and collapse of redundant isoforms. Line by line instructions for running Cogent for clustering of transcript isoforms in a screen on the NECTAR machine as below. Includes instructions for Cupcake and minimap to collapse redundant transcript isoforms.

Some notes and commentary are based on notes and personal communication with Tessa Bradford & Terry Bertozzi at the SA Museum, as well as the Cogent GitHub Tutorial page

Environments were used for installations (most often achieved with mini conda) on the NEXTAR machine. Activate screen, activate the environment, make Cogent output directory, and go to analysis working directory

```
screen -s cogent
conda activate anaCogent
mkdir /mnt/IsoSeq-analysis/data/Cogent
cd /mnt/IsoSeq-analysis/data/Cogent
```

From the analysis directory the input data directory is ../2019-04-30_out/polished.hq.fasta. The first output will be in this original data directory. Check paths (During installation, all paths were put into .profile)

Family Finding Running Family Finding for a small dataset

Create a k-mer profile of the input and calculate pairwise distances
Note: default K-mer sixe =30

```
python /mnt/Prog/Cogent/Cogent/run_mash.py --cpus=7
↪  /mnt/IsoSeq-analysis/data/2019-04-30_out/polished.hq.fasta
```

The output will be <fasta_filename>.k<sketch_size>.dist
Output written to: /mnt/IsoSeq-analysis/data/2019-04-30_out/polished.hq.fasta.s1000k30.dist

Process the distance file and create the partitions for each gene family

```
process_kmer_to_graph.py ../2019-04-30_out/polished.hq.fasta
↪  ../2019-04-30_out/polished.hq.fasta.s1000k30.dist ./hq hq
```

This generates an output log file hq.partition.txt and for each partition ("gene" family or isoform set), a subdirectory called hq/<partition_number> which contains the subset of fasta seqeuences belonging to that family. Note that sequences that don't belong to any partition (because it has no similarity with other sequences) will be "unassigned" and noted in the partition log file.

Coding Genome Reconstruction Each "gene" family must be done individually for coding region. Reconstructed contigs will contain the whole coding region. Reconstructed file is called cogent2.renamed.fasta

The script for an individual folder would be:

```
reconstruct_contig.py -S T.adelaidensis hq/hq_0
```

However there are hundreds of folders so this must be looped. This script was written to serve this purpose reconstructContig_1.sh:

```
#!/bin/bash
#
## this script is for coding region reconstruction of Cogent gene families as these
↪  must be completed individually.
# This script assumes you are following Running-Cogent.txt and using the same
↪  directory structure
# This script to be used after the process_kmer_to_graph.py step
```

```
# script in /mnt/IsoSeq-analysis/src
# usage from /mnt/IsoSeq-analysis/data/Cogent/hq
## bash ../../../src/reconstructContig.sh

# Carmel Maher
# August 2019

#Terry's error exit
function error_exit
{
    # Exit function due to fatal error
    # Accepts 1 arg:
    # string - descriptive error message
    echo "${PROGNAME}: ${1:-"Unknown error"}" 1>&2
    exit 1
}

cd ~/mnt/IsoSeq-analysis/data/Cogent/hq

for d in */ ; do

  DIR=${d%*}
  echo $DIR

  reconstruct_contig.py -S T.adelaidensis ./$DIR || error_exit "$LINENO: Error
  ↪  "$DIR""

done
echo compete
```

An error message prompted further checking of folder completion as below

Number of gene families:

```
ls ./hq/*/cogent2.renamed.fasta | wc -1
4454
```

Finding failed reconstruction jobs (pers. comms Terry Bertozzi, SA Museum)

Number of hq gene folders:

```
ls hq | wc -l #number of folders
4459
```

Number of /hq folders completed:

```
ls ./hq/*/COGENT.DONE | wc -l
4454
```

This matches the total gene family count so all outputs have been recorded correctly, but does not match the total /hq folder count = 5 folders not completed.

The following will list the folders that have not finished:

```
comm -3 <(find /mnt/IsoSeq-analysis/data/Cogent/hq -iname 'COGENT.DONE' -printf
↪  '%h\n' | sort -u) <(find /mnt/IsoSeq-analysis/data/Cogent/hq -maxdepth 1
↪  -mindepth 1 -type d | sort) | sed -e 's/^.*hq\///'

hq_2605
hq_2738
hq_320
hq_6391
hq_9030
```

Some testing required to create a modified script with a loop for failed folders, in order to include 5 folders which failed to run with kmer length 30 informed by this issue.
Due to how the loop functions inputting kmer size 27 on execution will actually run all at kmer size 30.

```
bash /mnt/IsoSeq-analysis/src/TBreconstructContig-edit.sh
↪  /mnt/IsoSeq-analysis/data/Cogent/hq 27 T.adelaidensis
```

The modified script TBreconstructContig-edit.sh:

```
#!/bin/bash

# This script runs reconstruct_contig.py on every subdirectory contained in the top
↪  directory. reruns with a kmer parameter of +3 (to a max of 99) for failed
↪  directories.
# This script assumes directories and files exist as up until the Coding Genome
↪  Reconstruction step in the Running Cogent GitHub wiki
#
# usage:  bash <path/script> <full directory containing hq_folders> <starting kmer
↪  value> <Species Name>
# e.g.        bash /mnt/IsoSeq-analysis/src/TBreconstructContig-edit.sh
↪  /mnt/IsoSeq-analysis/data/Cogent/hq 27 T.adelaidensis

# DIRNAME=$1
# species=$3

kmerSize=$2

find $1 -mindepth 1 -maxdepth 1 -type d | while read line; do #lists off all the
↪  /hq_* folders within the input directory and passes them through the script one
↪  by one (by line)
  FILE=$line/cogent2.fa
  if [ ! -f $FILE ]; then
      echo "failed no cogent2.fa Increase K-mer size" #Note all jobs will "fail" on
      ↪  first attempt unless you are running this in a previously reconstructed
      ↪  directory. In the latter case only the failed directories will continue
      ↪  to rerun.
      while [ ! -f "$FILE" ];do
          kmerSize=$((kmerSize + 3))
          rerunCMD="reconstruct_contig.py -k ${kmerSize} -S $3 $line"
          ↪  #reconstruction script increasing kmer parameter by 3 -note because
          ↪  all directories fail initially the input kmer value should be 3 below
          ↪  the desired actual value.
          echo ${rerunCMD}
```

```
            eval ${rerunCMD}
            if [ -f $FILE ]; then
                touch $1/hq_kmerSize.txt #This text file will give a list of the used
                ↪  kmerSize per hq_* folder
                echo "SUCCESS! The increased Kmer size $kmerSize was successful for
                ↪  $line." >> $1/hq_kmerSize.txt
            fi
            if [ ${kmerSize} -gt 99 ];then
                cp $line/in.fa $line/cogent2.fa # if a folder reruns until 99kmer
                ↪  parameter length is reached the input is simply copied into
                ↪  cogent2.fa
                touch $1/failed-jobs.txt #This text file will remain empty or give a
                ↪  list of all failed folders
                echo "Failed to succeed reconstruction with largest Kmer on ${line}.
                ↪  Copied input transcripts to cogent2.fa as output." >>
                ↪  $1/failed-jobs.txt
            fi
        done
        kmerSize=$2
    fi
done

if [ -s $1/failed-jobs.txt ]; then
    echo "failed-jobs.txt not empty. Some jobs failed. Please re-run them."
else
    echo "failed-jobs.txt empty or doesnt exist. All jobs completed."
fi
```

Re-check output

Number of completed gene families:

```
ls ./hq/*/COGENT.DONE | wc -l
4459
```

OR:

```
ls ./hq/*/cogent2.renamed.fasta | wc -l
4459
```

Finding failed reconstruction jobs

Number of items in ./hq:

```
ls hq | wc -l
4460
```

The added 1 is the text file reporting the successful kmer size for each "gene" family folder. Therefore all have completed.

List the names of folders that have not finished:

```
comm -3 <(find /mnt/IsoSeq-analysis/data/Cogent/hq -iname 'COGENT.DONE' -printf
↪  '%h\n' | sort -u) <(find /mnt/IsoSeq-analysis/data/Cogent/hq -maxdepth 1
↪  -mindepth 1 -type d | sort) | sed -e 's/^.*hq\///'
```

(blank, as it should be)

See failed-jobs.txt and/or hq_kmerSize.txt
Note: all completed at kmer size 30 this time, and filed-jobs.txt does not exist as none failed.

Using Cogent to collapse redundant transcripts in absence of genome Creating the "fake genome"

List and number of unassigned sequences
Make sure you are in the correct Cogent directory and can see the file hq.partition.txt:

```
tail -n 1 hq.partition.txt |tr ',' '\n' > unassigned.list
```

Create the unassigned.list file in the same directory:

```
tail -n 1 hq.partition.txt |tr ',' '\n' | wc -l
3193
```

Note: this number did not change with the addition of the 5 failed reconstruction folders. i.e. they failed completely and did not end up in the unassigned dataset. Now that they are fixed this dataset is unaffected.

Make the unassigned hq sequences into a fasta file. Make sure you are in the anaCogent environment so Cupcake works. Make sure the .py directory for cupcake is in the PATH.

```
export PATH=$PATH:/mnt/Prog/cDNA_Cupcake/sequence
get_seqs_from_list.py ../2019-04-30_out/polished.hq.fasta unassigned.list >
↪  unassigned.fasta
```

Concatenate unassigned with Cogent contigs by putting the reconstructed genes plus the unassigned single hq isoforms into a single fasta file:

```
mkdir collected
cd collected
cat ../hq/*/cogent2.renamed.fasta ../unassigned.fasta > cogent_fake_genome.fasta
```

Collapsing redundant isoforms

Create a SAM alignment. This can be done either with Minimap2 or GMAP - We have used minimap2.
Obtain a final set of unique (non-redundant) transcript Isoforms that can be used as a reference gene set.
As there is natural 5' degradation in RNA some sequences will represent identical isoforms which may not all be identified in clustering.
These parameters are default or suggested in the Cogent tutorial page.

Create an aligned, sorted SAM file:

```
export PATH=$PATH:/mnt/Prog/minimap2
minimap2 -ax splice -t 30  -uf --secondary=no cogent_fake_genome.fasta
↪  ../../2019-04-30_out/polished.hq.fasta > hq.fasta.sam
```

Output:

```
      [M::mm_idx_gen::2.392*1.00] collected minimizers
      [M::mm_idx_gen::3.526*1.32] sorted minimizers
      [M::main::3.528*1.32] loaded/built the index for 9960 target sequence(s)
      [M::mm_mapopt_update::3.708*1.30] mid_occ = 31
      [M::mm_idx_stat] kmer size: 15; skip: 5; is_hpc: 0; #seq: 9960
      [M::mm_idx_stat::3.810*1.29] distinct minimizers: 6267388 (82.12% are singletons);
↪  average occurrences: 1.346; average spacing: 2.846
      [M::worker_pipeline::13.901*5.65] mapped 25117 sequences
      [M::main] Version: 2.11-r797
      [M::main] CMD:
↪  /mnt/Prog/pacbio/smrtlink/install/smrtlink-release_6.0.0.47841/bundles/smrttools/install/smrttools-
↪  -ax splice -t 30 -uf --secondary=no cogent_fake_genome.fasta
↪  ../../2019-04-30_out/polished.hq.fasta
      [M::main] Real time: 13.942 sec; CPU: 78.558 sec
```

Then follow the collapse tutorial from Cupcake
There is another cupcake page with information but the example scripts are provided on the cogent tutorial page.

Sort the SAM file:

```
      sort -k 3,3 -k 4,4n hq.fasta.sam > hq.fasta.sorted.sam
```

Colapse identical isoforms to obtain a list of full length, unique, hq isoforms to use as reference transcripts:

```
      cd /mnt/Prog/cDNA_Cupcake/sequence
      which collapse_isoforms_by_sam.py
      /home/ubuntu/miniconda2/envs/anaCogent/bin/collapse_isoforms_by_sam.py

      cd /mnt/IsoSeq-analysis/data/Cogent/collected
```

Usage is:

```
      # usage: collapse_isoforms_by_sam.py [-h] [--input INPUT] [--fq] -s SAM -o
      #                                    PREFIX [-c MIN_ALN_COVERAGE]
      #                                    [-i MIN_ALN_IDENTITY]
      #                                    [--max_fuzzy_junction MAX_FUZZY_JUNCTION]
      #                                    [--flnc_coverage FLNC_COVERAGE]
      #                                    [--dun-merge-5-shorter]
```

Collapse:

```
      collapse_isoforms_by_sam.py --input ../../2019-04-30_out/polished.hq.fasta -s
      ↪  hq.fasta.sorted.sam -c 0.94 -i 0.85 --dun-merge-5-shorter -o hq.fasta.no5merge
```

These parameters taken from pers. comm Tessa Bradford, SA Museum.
Output is the name 'stem' the collapsed.rep.fa is for annotation.
The output files are <-o>.collapsed.gff, <-o>.collapsed.rep.fq, and <-o>.collapsed.group.txt i.e. hq.fasta.no5merge.collapsed.g
hq.fasta.no5merge.collapsed.rep.fq, and hq.fasta.no5merge.collapsed.group.txt

The naming system for the post-collapse isoform is PB.<loci_index>.<isoform_index>

More terminal output from the above has been excluded. This is an example:

```
Ignored IDs written to: hq.fasta.no5merge.ignored_ids.txt
Output written to:
hq.fasta.no5merge.collapsed.gff
hq.fasta.no5merge.collapsed.group.txt
hq.fasta.no5merge.collapsed.rep.fa
Namespace(allow_extra_5exon=False, flnc_coverage=-1, fq=False,
↪  input='../../2019-04-30_out/polished.hq.fasta', max_3_diff=100,
↪  max_5_diff=1000, max_fuzzy_junction=5, min_aln_coverage=0.94,
↪  min_aln_identity=0.85, prefix='hq.fasta.no5merge', sam='hq.fasta.sorted.sam')
```

Check outputs:

```
ls

cogent_fake_genome.fasta                        hq.fasta.no5merge.collapsed.rep.fa
hq.fasta.no5merge.collapsed.gff                 hq.fasta.no5merge.ignored_ids.txt
hq.fasta.no5merge.collapsed.gff.unfuzzy         hq.fasta.sam
hq.fasta.no5merge.collapsed.group.txt           hq.fasta.sorted.sam
hq.fasta.no5merge.collapsed.group.txt.unfuzzy
```

The file hq.fasta.no5merge.collapsed.group.txt names the isoforms as PB.<loci_index>.<isoform_index> and lists the collapsed identical isoforms.
Each 'locus' consists of a strand-specific locus with isoforms that overlap by at least 1 bp. So PB.11.1 and PB.11.2 means this locus has two isoforms.

Count the 'loci' (transcripts) found:

```
wc -l hq.fasta.no5merge.collapsed.group.txt
15729 hq.fasta.no5merge.collapsed.group.txt
```

Write file statistics:

```
get_abundance_post_collapse.py hq.fasta.no5merge.collapsed
↪  /mnt/IsoSeq-analysis/data/2019-04-30_out/polished.cluster_report.csv

WARNING: isoseq3 format detected. Output `length` column will be `NA`.

Read stat file written to hq.fasta.no5merge.collapsed.read_stat.txt
Abundance file written to hq.fasta.no5merge.collapsed.abundance.txt
```

Note: this is true, hq.fasta.no5merge.collapsed.read_stat.txt has no length information and a whole column of NA values, however the headers of hq.fasta.no5merge.collapsed.rep.fa do have length information. This is how IsoSeq3 format data is processed and does not affect other aspects of this script.

Get count information from the abundance and group .txt files

Add the headers:

```
echo -e "pbid\tcount_fl" > output.collapsed.abundance.txt
```

Add the PB.loci_index.isoform_index information and number of isoforms:

```
paste <(awk '{print $1}' hq.fasta.no5merge.collapsed.group.txt) <(awk -F , '{print
↪  NF}' hq.fasta.no5merge.collapsed.group.txt) >> output.collapsed.abundance.txt
```

All these commands except reconstructContig_1.sh and TBreconstructContig.edit.sh are provided in the one document Running-Cogent-2.sh on Github.

The longest transcript for each set of isoforms is in hq.fasta.no5merge.collapsed.rep.fa. This file is used in the following step and also provides the full transcripts to be sub-set as a reference based on further clustering in Chapter 5.

ANGEL

ANGEL: Robust Open Reading Frame prediction

Line by line instructions for running ANGEL in order to predict open reading frame and peptide sequences of transcript Isoforms. Also outputs 5' and 3' untranslated regions and scores transcript completeness.

This analysis was tested on an environment on the eRSA NECTAR research cloud, final data run was completed by Terry Bertozzi on a Dell PowerEdge server with 40 cores and 512G RAM due to memory limitations of the NECTAR allocation.

Script parameters and file names are accurate but directories are an example consistent with the resultant data, as this analysis was run on a different HPC machine. All results were then immediately placed in ~IsoSeq-analysis/data/ANGEL

```
#The python dependencies for ANGEL are:

numpy
Biopython
scikit-learn
CD-HIT version 4.8.1
conda install -n anaCogent scikit-learn

mkdir /mnt/IsoSeq-analysis/data/ANGEL
cd /mnt/IsoSeq-analysis/data/ANGEL
```

Dumb ORF prediction dumb_predict.py takes as input a FASTA file. It outputs the longest ORF in all frames - use to create a top training dataset.

Usage:

```
dumb_predict.py <fasta_filename> <output_prefix>
       [--min_aa_length MIN_AA_LENGTH]
       [--use_firstORF]
       [--use_rev_strand] [--cpus CPUS]
```

By default, only the forward strand is used. This is especially true for PacBio transcriptome sequencing output.

```
dumb_predict.py
↪  /mnt/IsoSeq-analysis/data/Cogent/collected/hq.fasta.no5merge.collapsed.rep.fa
↪  pygmy.dumb --min_aa_length 100 --cpus 24

# ------
13335  finished       7611  clusters
```

Creating a non-redundant training dataset ANGEL classifier training:

```
angel_make_training_set.py pygmy.dumb.final pygmy.dumb.final.training --random
↪  --cpus 24

angel_train.py pygmy.dumb.final.training.cds pygmy.dumb.final.training.utr
↪  pygmy.dumb.final.classifier.pickle --cpus 12
```

Robust ORF prediction Based on both the ANGEL Classifier training and the dumb ORF prediction.

Sequences are output as fasta files with whether they are complete, 5' partial, 3' partial, or internal in the header information, which also includes the aa length.
Preidctions are tagged as confident, likely, or suspicious, and dumb ORF predictions as dumb.
The proportion of each isoform that is untranslated, as well as the position of the cds sequence is also output.

```
angel_predict.py ../Cogent/collected/hq.fasta.no5merge.collapsed.rep.fa
↪  pygmy.dumb.final.classifier.pickle pygmy --output_mode=best
↪  --min_angel_aa_length 100 --min_dumb_aa_length 100 --cpus 48
```

Output is written to:

- pygmy.ANGEL.cds
- pygmy.ANGEL.pep
- pygmy.ANGEL.utr

These outputs are used further in Chapter 5 for additional clustering and BLASTx searches.

Note: The reverse strand options within ANGEL are not necessary on Isoseq data.

All these commands are provided in the one document Running_ANGEL.sh on Github.

CD-HIT

Clustering of transcript isoforms. Notes and output recorded for CD-HIT analysis of peptide sequences output from ANGEL ORF prediction. This analysis was conducted on the CD-HIT web server.

cd-hit was run by uploading pygmy.ANGEL.pep to the server and selecting a cutoff at 0.99.
CD-HIT notes and recorded output are in CDHit notes.pep99.bash and below.

```
cd-hit - run on pygmy.ANGEL.pep

separate run cutoff at .99

***
```

```
Your job 1598936109 is finished.
Program you ran: cd-hit
You input file is pygmy.ANGEL.pep and we named it as 1598936109.fas.0
Summary information for 1598936109.fas.0 included in 1598936109.fas.0.stat
You required 1 runs for sequence clustering
     1. Fasta file for representative sequences at 99% identity is 1598936109.fas.1
         Summary information for 1598936109.fas.1 included in 1598936109.fas.1.stat
         Corresponding cluster file is1598936109.fas.1.clstr
         Sorted cluster file by size is 1598936109.fas.1.clstr.sorted
Generated shell script is run-1598936109.sh

faa_stat.pl 1598936109.fas.0
faa_stat.pl 1598936109.fas.1
/data5/data/NGS-ann-project/apps/cd-hit/clstr_sort_by.pl no <
↪  1598936109.fas.1.clstr > 1598936109.fas.1.clstr.sorted
/data5/data/NGS-ann-project/apps/cd-hit/clstr_list.pl 1598936109.fas.1.clstr
↪  1598936109.clstr.dump
gnuplot1.pl < 1598936109.fas.1.clstr > 1598936109.fas.1.clstr.1; gnuplot2.pl
↪  1598936109.fas.1.clstr.1 1598936109.fas.1.clstr.1.png
/data5/data/NGS-ann-project/apps/cd-hit/clstr_list_sort.pl 1598936109.clstr.dump
↪  1598936109.clstr_no.dump
/data5/data/NGS-ann-project/apps/cd-hit/clstr_list_sort.pl 1598936109.clstr.dump
↪  1598936109.clstr_len.dump len
/data5/data/NGS-ann-project/apps/cd-hit/clstr_list_sort.pl 1598936109.clstr.dump
↪  1598936109.clstr_des.dump des
```

The generated CD-HIT script is uploaded to github as run-1598936109.sh

The full output metadata file is uploaded to github as 1598936109.out and the summary data provided below:

```
================================================================
Program: CD-HIT, V4.7 (+OpenMP), Sep 06 2018, 09:36:42
Command: /data5/data/NGS-ann-project/apps/cd-hit/cd-hit -i
         1598936109.fas.0 -d 0 -o 1598936109.fas.1 -c 0.99 -n 5
         -G 1 -g 1 -b 20 -l 10 -s 0.0 -aL 0.0 -aS 0.0 -T 4 -M
         32000

Started: Mon Aug 31 21:55:20 2020
================================================================
                            Output
----------------------------------------------------------------
total seq: 13882
longest and shortest : 2513 and 99
Total letters: 4952779
Sequences have been sorted

...


13882  finished       9907  clusters
```

The output 1598936109.fas.1.clstr.sorted contains the sorted cluster groups generated by CD-Hit and indicates the longest representative transcript for each cluster.

Reading CD-HIT output in R

dplyr was used to manipulate the CD-Hit output to create a list of unique transcript IDs of the longest representative transcript for each cluster. This list file is required in the following step to subset from the full length transcript file which contains the untranslated regions.

Load libraries and set the working directory

Import 1598936109.fas.1.clstr.sorted, the CD-Hit output file with all transcript IDs assigned to a sorted cluster (note: ./ReferenceClusters/ has been continually included due to setwd errors).

```
clstr <- read.csv("./ReferenceClusters/1598936109.fas.1.clstr.sorted",
    sep = "\t", row.names = NULL, header = FALSE, stringsAsFactors = FALSE)
head(clstr)
```

Initial file had a blank line with ">Cluster #" in between new clusters. Extend the >Cluster # down the column so that all transcripts are directly associated with a cluster number in their row.

```
clstr2 <- clstr
n = nrow(clstr)
x = 0
numbers_only <- function(x) !grepl("\\D", x)
for (row in c(1:n)) {
    if (numbers_only(clstr2[row, 1]) == TRUE) {
        clstr2[row, 1] <- x
    } else {
        NULL
    }
    x <- clstr2[row, 1]
}
head(clstr2, 20)
```

Count the number of transcript members for each cluster.

```
clstr.sums <- data.frame(dplyr::count(clstr2, V1))
write.csv(clstr.sums, "~/@Uni/@Flinders University
↪  PhD/RWorkingDir/Workflow-readthedown/ReferenceClusters/clstr.sums.csv",
    quote = FALSE, row.names = FALSE, col.names = FALSE)
head(clstr.sums)
```

Remove the additional blank row without a transcript name from between each cluster group.

```
switch <- clstr.sums[1, 2]
clstr3 <- cbind(clstr2[1], clstr)
clstr3[c((switch - 5):(switch + 5)), ]
```

```
clstr4 <- clstr2[-which(clstr2$V2 == ""), ]
clstr4[c(1:5, (switch - 5):(switch + 5)), ]
```

Separate CD-Hit information into columns by delimiters:

- Remove > symbol from transcript names

- Move number of amino acids into a new column by separating "aa"
- Move statistics of % match into a new column byt separating "..."
- Give the columns headers

```
library(stringr)
clstr5 <- clstr4
clstr5[] <- lapply(clstr5, gsub, pattern = ">", replacement = "")
clstr5.2 <- data.frame(str_split_fixed(clstr5$V2, "aa, ", 2))
clstr5.3 <- data.frame(str_split_fixed(clstr5.2$X2, "... ", 2))
clstr6 <- cbind(clstr5[1], clstr5.2[1], clstr5.3[1:2])
colnames(clstr6) <- c("cluster", "aa", "TranscriptID", "stat")
head(clstr6)
```

Filter based on rows that have a "*" in the stat column. These indicate the representative sequences for each cluster determined by CD-Hit and will act as the main reference.

```
clstr7 <- filter(clstr6, stat == "*")
head(clstr7, 50)
```

Pull out the column "TranscriptID" and export into a txt file so it can be used as a list to extract these sequences.

```
clstrTranscriptID <- pull(clstr7, var = TranscriptID)

write.csv(clstrTranscriptID, "~/@Uni/@Flinders University
↪  PhD/RWorkingDir/Workflow-readthedown/ReferenceClusters/clstrTranscriptID.csv",
    quote = FALSE, row.names = FALSE, col.names = FALSE)
head(clstrTranscriptID)
```

Pull out the first segment before the first | from column "TranscriptID" and export into a txt file. this reduced the ID to the initial PB.<loci_index>.<isoform_index> identifier.

```
WholeTranscriptID <- word(clstrTranscriptID, 1, sep = "\\|")
write.csv(WholeTranscriptID, "~/@Uni/@Flinders University
↪  PhD/RWorkingDir/Workflow-readthedown/ReferenceClusters/WholeTranscriptID.csv",
    quote = FALSE, row.names = FALSE, col.names = FALSE)
head(WholeTranscriptID)
```

Note there are "duplicates" if the name is truncated all the way back to the PB.<loci_index>.<isoform_index> identifier due to multiple isoforms that are retained. A longer, but still truncated version of this ID (excluding the information appended by ANGEL) is required to subset from the hq.fasta.no5merge.collapsed.rep.fa file.

The clstrTranscriptID.csv is now the full list of representative transcripts based on clustering of the translated proteins in predicted read frame. This file was then used in NECTAR to extract a .fasta of all the reference transcripts.

Seqtk

Seqtk was used on the NECTAR machine to manipulate the fasta files and subset based on list files of transcript names.

Installation:

```
#Carmel July 2020

#Used on NECTAR research cloud

#Relies on hard-coded paths & filenames
#run line by line in shell

----------------

cd /mnt/Prog
conda install -c bioconda seqtk

#___
The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    certifi-2020.6.20          |   py37hc8dfbb8_0         151 KB  conda-forge
    conda-4.8.3                |   py37hc8dfbb8_1         3.0 MB  conda-forge
    openssl-1.1.1g             |       h516909a_1         2.1 MB  conda-forge
    python_abi-3.7             |          1_cp37m           4 KB  conda-forge
    seqtk-1.3                  |       hed695b0_2          39 KB  bioconda
    ------------------------------------------------------------
                                           Total:         5.3 MB

The following NEW packages will be INSTALLED:

  python_abi         conda-forge/linux-64::python_abi-3.7-1_cp37m
  seqtk              bioconda/linux-64::seqtk-1.3-hed695b0_2
#___

Conda packages are located in /mnt/Prog/miniconda3/bin/ if you dont set a path
----------------
```

Manually convert clstrTranscriptID.csv to an clstrTranscriptID.lst file.

Seqtk was used to subset the pygmy.ANGEL.cds fasta file based on the R cluster list outputs: Moved files into /mnt/IsoSeq-analysis/data/Seqtk

```
screen        #to be safe
cd /mnt/IsoSeq-analysis/data/Seqtk
/mnt/Prog/miniconda3/bin/seqtk subseq ../ANGEL/pygmy.ANGEL.cds
↪  clstrTranscriptID.lst > clstrTranscriptID.fa
```

check:

```
wc -l clstrTranscriptID.lst
# 9907
wc -l clstrTranscriptID.fa
# 19814

# 19814/2 = 9907. So it is correct
```

The hq.fasta.no5merge.collapsed.rep.fa file created in the program ANGEL in Chapter 4 is a fasta file where sequence information ran over multiple "lines" within the file.

Seqtk was used to convert this file into a .fasta with sequence on a single line.

```
17/11/2020
# just converting hq.fasta.no5merge.collapsed.rep.fa into a fasta with sequence all
↪  on one line
# command seq is basic fasta/q conversion

cd /mnt/IsoSeq-analysis/data/Seqtk
/mnt/Prog/miniconda3/bin/seqtk seq
↪  ../Cogent/collected/hq.fasta.no5merge.collapsed.rep.fa >
↪  hq.fasta.no5merge.collapsed.rep.1L.fa
```

Checked that the file still has the correct number of sequences:

```
$ grep -c ">" hq.fasta.no5merge.collapsed.rep.fa
15729

grep -c ">" hq.fasta.no5merge.collapsed.rep.1L.fa
15729
```

Seqtk was also used to search for transcripts which returned BLAST results for some genes of interest as data exploration. The reference list files for this were extracted from the BLASTx output were created using find and sort functions in notepad++ and excel, and then applied to the pygmy.ANGEL.cds fasta file to extract sequences.

```
#Usage from /mnt/IsoSeq-analysis/data/ANGEL

cd /mnt/IsoSeq-analysis/data/Seqtk
/mnt/Prog/miniconda3/bin/seqtk subseq ../ANGEL/pygmy.ANGEL.cds AQP.lst > AQP.fa

/mnt/Prog/miniconda3/bin/seqtk subseq ../ANGEL/pygmy.ANGEL.cds HSP.lst > HSP.fa

/mnt/Prog/miniconda3/bin/seqtk subseq ../ANGEL/pygmy.ANGEL.cds SLC.lst > SLC.fa

/mnt/Prog/miniconda3/bin/seqtk subseq ../ANGEL/pygmy.ANGEL.cds MAPK.lst > MAPK.fa
```

Subsetting full length transcripts which represent each cluster

Creation of a subset of the longest representative transcripts for each cluster as per CD-HIT and the previous two to form the reference for counts in the following Chapter 6.

As only the peptide coding sequence of transcripts was used in the BLASTx search and the CD-HIT clustering in order to cluster similarly expressed transcripts, the Untranslated regions of transcripts are absent. Our goal is to include these untranslated regions in the reference to be used for gene expression counts.
Rather than attempting to piece pygmy.ANGEL.cds and pygmy.ANGEL.utr back together, a truncated version of the transcript names was used to pull the full length unaltered transcript out of the earlier hq.fasta.no5merge.collapsed.rep.fa file.

Initial checks of clstrTranscriptID.csv were carried out. Files are continually checked for transcript ID or sequence counts to ensure manual manipulation did not remove any data that was not being targeted. Exploration of discrepancies below identified a number of transcripts that with clustering of multiple isoforms, have

seeded more than one cluster. This means that there were duplicates in the list file once coding region identifiers were removed and sequences had to be extracted manually and not by applying the clstrTranscriptID list to hq.fasta.no5merge.collapsed.rep.1L.fa in seqtk.

The following was done on a local Intel core i7 machine using RStudio or in-house bash scripts applied in Git Bash for Windows. Some later steps were carried out on the NECTAR allocation.

All files copied to "~/IsoSeq-analysis/data/Seqtk/reference COPIES 2020-11" and contents checked

The following file contains all of the below:

```
Carmel@CarmelASUS MINGW64 ~
cd "E:/@DATA/@@PBT_KidneyIsoSeq_Working
↪  Data_C.Maher/IsoSeq-analysis/data/Seqtk/reference COPIES 2020-11"

2020-11-17

cd "C:\Users\Carmel\Desktop\reference COPIES 2020-11"

-----

19814 clstrTranscriptID.fa                    /2 = 9907
9907 clstrTranscriptID.lst

hq.fasta.no5merge.collapsed.rep.fa
hq.fasta.no5merge.collapsed.rep.1L.fa
pygmy.ANGEL.cds
pygmy.ANGEL.pep
pygmy.ANGEL.utr
```

Sort and filter clstrTranscriptID.lst by unique values to make sure there are no dupicates

```
sort clstrTranscriptID.lst | uniq > clstrUniq.txt
wc -l clstrUniq.txt
9907 clstrUniq.txt
```

List duplicates if there are any found (there are not)

```
sort clstrTranscriptID.lst | uniq -d > clstrUniqD.txt
wc -l clstrUniqD.txt
0 clstrUniqD.txt
```

```
Carmel@CarmelASUS MINGW64 /e/@DATA/@@PBT_KidneyIsoSeq_Working
↪  Data_C.Maher/IsoSeq-analysis/data/Seqtk/reference COPIES 2020-11
```

---> **1. Isoseq HQ output file: hq.fasta.no5merge.collapsed.rep.fa**

header format

```
head -n 2 hq.fasta.no5merge.collapsed.rep.fa
>PB.1.1|000643|path0:1-1879(+)|transcript/14742 transcript/14742
↪  full_length_coverage=3;length=1887;num_subreads=52
ATAGAGTACAAGTACTAGAGTTTTAGTGTGGTCGAAGAGAAATCCAAGATCCACCATGGG
```

no of sequences

```
grep -c ">" hq.fasta.no5merge.collapsed.rep.fa
15729
```

---> **2. Angel CDS output from Isoseq HQ file: pygmy.ANGEL.cds**

header format

```
head -n 2 pygmy.ANGEL.cds
>PB.2.1|002537|path0:1-1624(+)|transcript/18304|m.1 type:likely-NA len:135 strand:+
↪  pos:282-686

↪  ATGTCCTACTGCAGGCAAGAAGGAAAGGACAGAATTATATTTGTGACAAAAGAGGACCATGAAACTCCAAGTAGTGCTGAGTTAGTAGCGGATC
```

Note Sequence is on a single line

no of sequences

```
grep -c ">" pygmy.ANGEL.cds
13882
```

---> Angel CDS file clusterd with cdHit: 1598936109.fas.1.clstr

header format

```
cd cd-hit\ 99
head 1598936109.fas.1.clstr
>Cluster 0
0       2513aa, >PB.715.4|1b393a|path6:6-8031(+)|transcript/8|m.1216... *
1       2250aa, >PB.715.5|1b393a|path6:278-8030(+)|transcript/13|m.1217... at
↪  99.56%
2       2369aa, >PB.715.6|1b393a|path6:524-8029(+)|transcript/20|m.1218... at
↪  99.58%
3       2148aa, >PB.715.7|1b393a|path6:1187-8030(+)|transcript/49|m.1219... at
↪  99.53%
4       1913aa, >PB.715.9|1b393a|path6:1893-8028(+)|transcript/118|m.1221... at
↪  99.79%
5       1688aa, >PB.715.11|1b393a|path6:2596-8054(+)|transcript/279|m.1223... at
↪  99.59%
6       1452aa, >PB.715.14|1b393a|path6:3304-8029(+)|transcript/591|m.1226... at
↪  100.00%
7       1333aa, >PB.715.15|1b393a|path6:3629-8031(+)|transcript/935|m.1227... at
↪  99.77%
8       1314aa, >PB.715.16|1b393a|path6:3916-8078(+)|transcript/962|m.1228... at
↪  100.00%
```

File contains truncated headers from the Angel CDS file, clustered acoording to similarity. Truncation appears to be at the first whitespace no of sequences

```
grep -c ', >' 1598936109.fas.1.clstr     #search pattern changed to avoid the ">"
↪  in front of each cluster label eg >Cluster 0
13882
```

Sequence number matches Angel CDS file

no of clusters

```
grep -c '>C' 1598936109.fas.1.clstr
9907
```

```
grep -c '>C' 1598936109.fas.1.clstr.sorted
9907
```

both match

```
tail 1598936109.fas.1.clstr
>Cluster 9902
0       99aa, >PB.8346.1|transcript/22300:1-1143(+)|transcript/22300|m.14112... *
>Cluster 9903
0       99aa, >PB.8372.1|transcript/22508:1-1103(+)|transcript/22508|m.14137... *
>Cluster 9904
0       99aa, >PB.8489.1|transcript/23315:1-855(+)|transcript/23315|m.14244... *
>Cluster 9905
0       99aa, >PB.8706.1|transcript/25054:1-372(+)|transcript/25054|m.14469... *
>Cluster 9906
0       99aa, >PB.9028.1|transcript/4242:1-2955(+)|transcript/4242|m.14776... *
```

```
tail 1598936109.fas.1.clstr.sorted
>Cluster 9902
0       99aa, >PB.8346.1|transcript/22300:1-1143(+)|transcript/22300|m.14112... *
>Cluster 9903
0       99aa, >PB.8372.1|transcript/22508:1-1103(+)|transcript/22508|m.14137... *
>Cluster 9904
0       99aa, >PB.8489.1|transcript/23315:1-855(+)|transcript/23315|m.14244... *
>Cluster 9905
0       99aa, >PB.8706.1|transcript/25054:1-372(+)|transcript/25054|m.14469... *
>Cluster 9906
0       99aa, >PB.9028.1|transcript/4242:1-2955(+)|transcript/4242|m.14776... *
```

Cluster numbers also match both files:Cluster 9906 + Cluster 0 = 9907

How has the header changed from Isoseq HQ after processing with Angel?

```
cd ../
grep '>PB.715.4|1b393a|path6:6-8031(+)|transcript/8'
↪  hq.fasta.no5merge.collapsed.rep.fa
>PB.715.4|1b393a|path6:6-8031(+)|transcript/8 transcript/8
↪  full_length_coverage=43;length=7837;num_subreads=60
```

the "|m.xxxxx" suffix in the ANGEL output file is added at the first whitespace

---> **3. The longest seed sequence from the cdHit output: clstrUniq.txt**

header format

```
head clstrUniq.txt
PB.10.1|004815|path1:5-1713(+)|transcript/17534|m.3
PB.100.1|049515|path5:1-1563(+)|transcript/19243|m.132
PB.1000.1|26d4b2|path0:1-2889(+)|transcript/4521|m.1797
PB.1000.2|26d4b2|path0:1287-2889(+)|transcript/18232|m.1798
PB.1002.1|26ea71|path2:1-2673(+)|transcript/6271|m.1799
PB.1002.2|26ea71|path2:11-2691(+)|transcript/8089|m.1800
PB.1003.1|26fc93|path2:1-3987(+)|transcript/1197|m.1801
PB.1005.1|2718e2|path0:1-5294(+)|transcript/258|m.1804
PB.1006.1|272202|path0:1-2713(+)|transcript/5814|m.1805
PB.1007.1|272202|path1:1-3370(+)|transcript/2554|m.1806
```

Prefix ">" and suffix "... *" have been removed in R Studio

no of strings

```
grep -c 'PB.' clstrUniq.txt
9907
```

matches number of clusters above

strip the "|m.xxxx" from the strings in clstrUniq.txt

```
sed 's/|m.*//' clstrUniq.txt > new_cluster.txt
```

check

```
head new_cluster.txt
PB.10.1|004815|path1:5-1713(+)|transcript/17534
PB.100.1|049515|path5:1-1563(+)|transcript/19243
PB.1000.1|26d4b2|path0:1-2889(+)|transcript/4521
PB.1000.2|26d4b2|path0:1287-2889(+)|transcript/18232
PB.1002.1|26ea71|path2:1-2673(+)|transcript/6271
PB.1002.2|26ea71|path2:11-2691(+)|transcript/8089
PB.1003.1|26fc93|path2:1-3987(+)|transcript/1197
PB.1005.1|2718e2|path0:1-5294(+)|transcript/258
PB.1006.1|272202|path0:1-2713(+)|transcript/5814
PB.1007.1|272202|path1:1-3370(+)|transcript/2554
```

```
grep -c 'PB.' new_cluster.txt
9907
```

How many of thes strings are in the original ONE LINE isoseq file?

```
grep -c -Ff new_cluster.txt hq.fasta.no5merge.collapsed.rep.1L.fa
9813
```

94 strings are missing

---> **4. find the missing strings**

convert the isoform hq file into the same format as new_cluster.txt

```
grep '>' hq.fasta.no5merge.collapsed.rep.1L.fa| sed s'/ tra.*//' | sed s'/>//' >
↪  hq_headers
head hq_headers
PB.1.1|000643|path0:1-1879(+)|transcript/14742
PB.2.1|002537|path0:1-1624(+)|transcript/18304
PB.3.2|00361c|path0:43-2240(+)|transcript/10564
PB.4.1|004079|path1:6-745(+)|transcript/23781
PB.4.2|004079|path1:6-612(+)|transcript/24311
PB.5.1|004079|path2:209-642(+)|transcript/24981
PB.6.1|004079|path3:273-616(+)|transcript/25193
PB.7.1|004079|path5:6-664(+)|transcript/24242
PB.8.1|004079|path6:1-607(+)|transcript/24477
PB.9.1|004079|path7:1-725(+)|transcript/23873
```

```
cat hq_headers |wc -l
15729
```

Check for the same number of matches

```
grep -Ff new_cluster.txt hq_headers > matching_strings
cat matching_strings |wc -l
9813
```

Check for duplicates

```
cat hq_headers | sort |uniq -cd
0

cat new_cluster.txt | sort |uniq -cd
      2 PB.1206.1|2e8181|path5:1-4617(+)|transcript/551
      2 PB.1261.1|30ba06|path3:633-1922(+)|transcript/21324
      2 PB.1570.2|3bd5a8|path2:20-1934(+)|transcript/13977
      2 PB.1719.1|404d72|path0:1-3429(+)|transcript/2407
      2 PB.1764.3|429969|path8:44-2062(+)|transcript/12739
      2 PB.1813.1|446aed|path0:1-2871(+)|transcript/4320
      2 PB.1865.1|46bd4e|path0:1-2536(+)|transcript/6726
      2 PB.1869.1|46d663|path2:1-1320(+)|transcript/21375
      2 PB.220.1|08884f|path0:1-4120(+)|transcript/991
      2 PB.2360.1|5843a6|path32:1-2062(+)|transcript/11326
      2 PB.2541.3|5e93d1|path0:5-2495(+)|transcript/7381
      2 PB.2560.1|5f09e2|path0:1-1196(+)|transcript/21957
      2 PB.2618.1|612418|path1:1-2013(+)|transcript/12200
      2 PB.2713.3|65abaa|path0:37-2495(+)|transcript/7693
      2 PB.2892.1|6b359a|path12:1-4597(+)|transcript/526
      2 PB.3037.2|702e9b|path5:2-3018(+)|transcript/3923
      2 PB.3243.1|784f79|path0:1-2683(+)|transcript/5903
      2 PB.3259.1|792211|path3:1-5233(+)|transcript/268
```

```
2 PB.3286.3|7a2645|path3:103-3087(+)|transcript/3780
2 PB.3405.1|7f2f96|path0:5-2081(+)|transcript/11903
3 PB.3430.1|8055ae|path8:1-3589(+)|transcript/2043
2 PB.3461.1|81f072|path5:1-2492(+)|transcript/7285
2 PB.351.4|0cccb8|path5:27-4293(+)|transcript/775
2 PB.3562.1|864d1b|path0:1-3492(+)|transcript/2174
2 PB.3615.1|883e51|path3:1-4044(+)|transcript/1347
2 PB.3836.3|8ff0e9|path0:4-3972(+)|transcript/946
2 PB.3934.1|93b547|path0:1-3353(+)|transcript/2676
2 PB.4042.2|97fba8|path2:34-2096(+)|transcript/13603
2 PB.4219.2|9d8e09|path3:9-3566(+)|transcript/2045
2 PB.4442.1|a6620e|path0:1-2906(+)|transcript/4555
2 PB.4458.1|a6e2a5|path0:1-3673(+)|transcript/1739
2 PB.4822.1|b4950f|path1:1-3315(+)|transcript/2177
2 PB.4822.2|b4950f|path1:1240-3315(+)|transcript/10982
2 PB.4919.1|b7da5b|path0:1-3327(+)|transcript/2652
2 PB.4923.1|b7fed9|path0:1-2394(+)|transcript/8643
2 PB.5080.1|bde542|path3:1-3060(+)|transcript/3777
2 PB.5096.2|be8c26|path4:70-2754(+)|transcript/5782
2 PB.5149.1|c15864|path0:1-1628(+)|transcript/17934
2 PB.5308.1|c8b5b5|path2:35-3805(+)|transcript/1487
2 PB.5395.1|cbd93f|path0:1-1699(+)|transcript/17107
2 PB.5483.2|cf1828|path1:23-2344(+)|transcript/9126
2 PB.5619.1|d4d2ea|path0:1-2677(+)|transcript/6673
2 PB.5663.1|d5e71b|path0:1-5101(+)|transcript/303
2 PB.5939.1|e23f0e|path0:1-2960(+)|transcript/3907
2 PB.5939.2|e23f0e|path0:5-4542(+)|transcript/622
2 PB.613.1|16d05c|path0:1-1558(+)|transcript/18673
2 PB.6133.1|ea032e|path9:1-2052(+)|transcript/12270
2 PB.6276.1|ee992b|path0:1-1487(+)|transcript/19762
2 PB.6283.1|eee48e|path1:1-3454(+)|transcript/2331
2 PB.6286.1|ef0b10|path12:1-2744(+)|transcript/5222
2 PB.6303.1|efc514|path0:1-3222(+)|transcript/2889
2 PB.6378.3|f31894|path9:80-2754(+)|transcript/5975
2 PB.6385.1|f32c31|path0:1-1500(+)|transcript/12840
2 PB.6426.1|f4ded6|path0:1-1796(+)|transcript/16785
2 PB.6576.2|facceb|path2:1-3314(+)|transcript/3157
2 PB.659.1|19437d|path1:1-3483(+)|transcript/2210
2 PB.6763.1|transcript/10186:1-2230(+)|transcript/10186
2 PB.6877.1|transcript/1099:1-4113(+)|transcript/1099
2 PB.6966.1|transcript/11636:1-2103(+)|transcript/11636
2 PB.713.1|1b043a|path0:1-965(+)|transcript/23154
2 PB.7227.1|transcript/138:1-5869(+)|transcript/138
2 PB.7287.1|transcript/14227:1-1898(+)|transcript/14227
3 PB.7342.1|transcript/14710:1-1879(+)|transcript/14710
2 PB.7349.1|transcript/14730:1-1903(+)|transcript/14730
2 PB.7353.1|transcript/14772:1-1860(+)|transcript/14772
2 PB.74.1|02ebe4|path0:1-2606(+)|transcript/6537
2 PB.7501.1|transcript/1596:1-3759(+)|transcript/1596
2 PB.7528.1|transcript/16151:1-1797(+)|transcript/16151
2 PB.7678.1|transcript/1728:1-3672(+)|transcript/1728
2 PB.7707.1|transcript/17496:1-1683(+)|transcript/17496
2 PB.7868.1|transcript/188:1-5558(+)|transcript/188
```

```
      2 PB.7973.1|transcript/19628:1-1476(+)|transcript/19628
      2 PB.8082.1|transcript/20433:1-1412(+)|transcript/20433
      2 PB.816.1|1e334e|path4:1-2040(+)|transcript/12316
      2 PB.8325.1|transcript/22158:1-1191(+)|transcript/22158
      2 PB.834.2|1ec273|path2:1291-2750(+)|transcript/19945
      2 PB.8389.1|transcript/22625:1-1060(+)|transcript/22625
      2 PB.8709.1|transcript/252:1-5299(+)|transcript/252
      2 PB.8883.1|transcript/3393:1-3152(+)|transcript/3393
      2 PB.8886.1|transcript/3402:1-3146(+)|transcript/3402
      2 PB.8920.1|transcript/3588:1-3076(+)|transcript/3588
      2 PB.9068.1|transcript/4442:1-2918(+)|transcript/4442
      2 PB.9162.1|transcript/4983:1-2815(+)|transcript/4983
      2 PB.9209.1|transcript/5238:1-2771(+)|transcript/5238
      2 PB.924.1|22d3cf|path3:1-1494(+)|transcript/19025
      2 PB.9277.1|transcript/5650:1-2707(+)|transcript/5650
      2 PB.944.1|241290|path4:2-2629(+)|transcript/6557
      2 PB.9535.1|transcript/7343:1-2508(+)|transcript/7343
      2 PB.9581.1|transcript/7647:1-2463(+)|transcript/7647
      2 PB.9683.1|transcript/8355:1-2402(+)|transcript/8355
      2 PB.9766.1|transcript/9024:1-2324(+)|transcript/9024
      2 PB.9784.1|transcript/915:1-4197(+)|transcript/915
```

During the clustering it appears that several of the sequences have been made seed sequences (denoted with a *) for more than one cluster.

The counts above equate to how many strings

```
cat new_cluster.txt | sort |uniq -cd | cut -d' ' -f7 | awk '{sum+=$1-1}
↪  END{printf("%d\n",sum)}'
94
```

Which is the discrepancy noted in section 3 above. Both hq.fasta.no5merge.collapsed.rep.fa and hq.fasta.no5merge.collapsed.rep.1L.fa. were checked in this way but the single line file is required in later steps so only those commands have been included.

Final pulling out of transcripts sequences into a .fasta on a single line, also removes the skipped lines that grep outputs as '--'.
note: NOT using seqtk anymore:

```
grep -Ff new_cluster.txt -A 1 hq.fasta.no5merge.collapsed.rep.1L.fa | grep -v ^-- >
↪  reference_transcripts.1Lv.fasta

grep -c ">" reference_transcripts.1Lv.fasta
9813
```

---> **next step**

Trim some persisting poly-a tails from this file:

```
reference_transcripts.1Lv.fasta
```

Note: The Trim polya command in bbduk leaves many tails intact due to single non 'A' bases and does not facilitate hamming distance command at the same time (?), however specifying Ktrim=r literal or "trimming twice" (poly-a command and THEN ktrim) sequences become very short even with different K values.

Terry Bertozzi provided perl script as below to manually trim:

```
# Terry Bertozzi
#command line to remove poly A tails

#for use on /mnt/IsoSeq-analysis/data/Seqtk/reference_transcripts.1Lv.fasta

cd /mnt/IsoSeq-analysis/data/Seqtk

perl -pe 's/(A{5,}.{0,2})+$//gm' reference_transcripts.1L.fasta >
↪  reference_transcripts.1L.clean.fasta

perl -pe 's/(A{5,}.{0,2})+$//gm' reference_transcripts.1Lv.fasta >
↪  reference_transcripts.1Lv.clean.fasta

grep -c ">" reference_transcripts.1Lv.clean.fasta
9813

head reference_transcripts.1Lv.clean.fasta
#also looks good.
```

reference_transcripts.1Lv.clean.fasta - IS THE REFERENCE

Short Read data

De-novo assembly of short-read Transcripts:
Short read transcripts used for gene expression analysis in Chater 6 were initially intended for de-novo assembly and analysis due to a lack of closely related genomic references for T. adelaidensis.

This section outlines the methods for assembly of short read datasets in the absence of a reference or genome for the purposes of comparing these final transcript datasets to the assembly generated from the long reads as above.

Other notes: Initial plans for Chapter 6 did not include the availability of long-read sequencing to supplement a reference transcript list. Therefore a pooled transcript assembly was created by concatenating reads from all eight samples in order to use this larger dataset as a pseudo-reference for other programs.
This longer assembly is included in the BUSCO analysis below to illustrate assembly completeness and duplication.

Sequences were downloaded from the Australian Genomics Research Facility server & checksumms verified. This analysis was conducted on the Flinders University 'Deep Thought' HPC.

FASTQC quality assessment

Quality of sequencing output was quantified using the following script and FastQC v0.11.8 (note there are two commands because there were two sequencing run data folders AGRF_CAGRF13871_HCGYYBCXY and AGRF_CAGRF20022_CDNJBANXX).

The FastQC Reports were then combined for visualisation using NGSReports The full FastQC output summary .html file of all eight samples PRIOR to trimming is available here.

Samples G6 and G7 were run on a second sequencing run, and have different overall lengths. This is the main source of variation in sequencing scores.

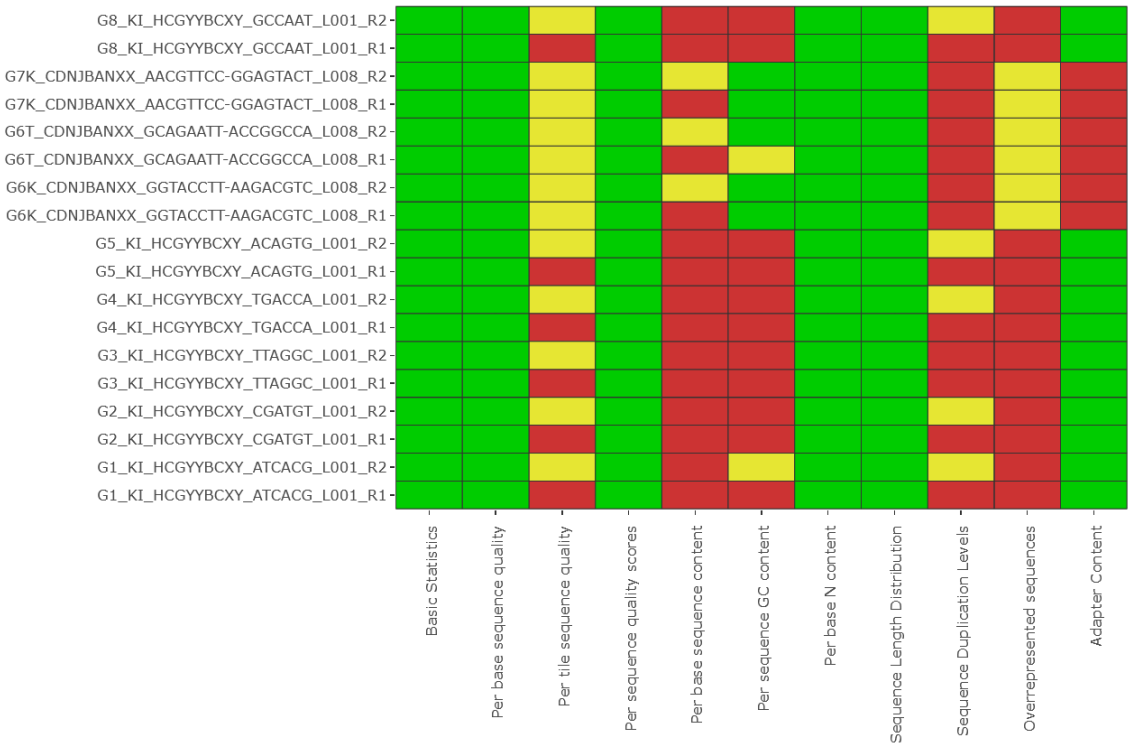

Figure 1: FastQC Summary of scores for paired reads of 8 samples prior to any adapter or quality trimming

BBduk adapter trimming

Trimming to remove Illumina TruSeq adapters and SMARTer PCR tags as well as trimming based on quality scores was done using bbduk, in in bbmap v38.46 tools and the following script [removeAdapters2.sh] (https://github.com/Carmel-src/T.adelaidensis_SuppInfo/blob/main/Chapter%204%20-%20Assembly/removeAdapters2.sh).

```
!/bin/bash

# November 2019
# Carmel Maher
# This script is to submit a BBduk script to cut adapters and quality on raw reads
↪  through Flinders' DeepThought HPC Slurm
# This script therefore assumes submission to Flinders Deepthought with an assumed
↪  directory structure and use of available modules


#-------------------------------------
# Required Modules:
module add bbmap #/38.46
module add fastqc #/0.11.8

#adapters location :ref=/cm/shared/apps/bbmap/38.46/resources/adapters.fa

THREADS=8
```

```
#-------------------------------------
# Terry's error exit

function error_exit
{
    # Exit function due to fatal error
    # Accepts 1 arg:
    # string - descriptive error message

    echo "${PROGNAME}: ${1:-"Unknown error"}" 1>&2
    exit 1
}

# in = /scratch/user/mahe0050/DE-analysis/0_rawData/
# out = /scratch/user/mahe0050/DE-analysis/1_trimmedData

#-------------------------------------

#change directories into raw sequencing directory 1
cd /scratch/user/mahe0050/DE-analysis/0_rawData/AGRF_CAGRF13871_HCGYYBCXY ||
↪  error_exit "$LINENO: directory error 1"
mkdir ../../1_trimmedData/temp

for file in *R1.fastq.gz
do
  echo $file
  FILESTEM=${file%_*}
  echo $FILESTEM
  FILESTEM2=`echo $file | cut -d "_" -f1,2 --output-delimiter="_"`
  echo $FILESTEM2

  #remove Illumina adapters from all paired data using bbduk
  bbduk.sh in=$file in2=$FILESTEM"_R2.fastq.gz"
  ↪  out=../../1_trimmedData/temp/$FILESTEM2"_R1_Iclean.fq.gz"
  ↪  out2=../../1_trimmedData/temp/$FILESTEM2"_R2_Iclean.fq.gz"
  ↪  outs=../../1_trimmedData/temp/$FILESTEM2"_singletons_Iclean.fq.gz"
  ↪  literal=GATCGGAAGAGCACA,AGATCGGAAGAGCGT ktrim=r k=15 mink=15 hdist=1
  ↪  threads=$THREADS || error_exit "$LINENO: Error removing Illumina adapters
  ↪  from R1 or R2 - directory 1"

  #remove SMARTer PCR tags from all paired data using bbduk
  bbduk.sh in=../../1_trimmedData/temp/$FILESTEM2"_R1_Iclean.fq.gz"
  ↪  in2=../../1_trimmedData/temp/$FILESTEM2"_R2_Iclean.fq.gz"
  ↪  out=../../1_trimmedData/$FILESTEM2"_R1_clean.fq.gz"
  ↪  out2=../../1_trimmedData/$FILESTEM2"_R2_clean.fq.gz"
  ↪  outs=../../1_trimmedData/$FILESTEM2"_singletons_clean.fq.gz"
  ↪  literal=AAGCAGTGGTATCAACGCAGAGTACNNNNN,GTACTCTGCGTTGATACCACTGCTT ktrim=r k=15
  ↪  mink=15 hdist=1 qtrim=r trimq=10 minlength=30 threads=$THREADS || error_exit
  ↪  "$LINENO: Error removing SMARTer adapters from R1 or R2 - directory 1"

  #remove SMARTer PCR tags from the singleton file using bbduk
  bbduk.sh in=../../1_trimmedData/temp/$FILESTEM2"_singletons_Iclean.fq.gz"
  ↪  out=../../1_trimmedData/$FILESTEM2"_singletons1.fq.gz"
  ↪  literal=AAGCAGTGGTATCAACGCAGAGTACNNNNN,GTACTCTGCGTTGATACCACTGCTT ktrim=r k=15
  ↪  mink=15 hdist=1 qtrim=r trimq=20 minlength=30 threads=$THREADS || error_exit
  ↪  "$LINENO: Error removing SMARTer adapters from singletons - directory 1"
```

```
  #concatenate the singleton files
  cat ../../1_trimmedData/$FILESTEM2"_singletons1.fq.gz" >>
  ↪  ../../1_trimmedData/$FILESTEM2"_singletons_clean.fq.gz" || error_exit
  ↪  "$LINENO: Error concatenating singletons - directory 1"


  #clean up: i.e. remove temporary directories
  rm ../../1_trimmedData/$FILESTEM2"_singletons1.fq.gz" || error_exit "$LINENO:
  ↪  Error deleting singleton intermediate - directory 1"

done


#change directories into raw sequencing directory 2 and do it again for these
↪  samples (output to same trimmed directory for further analysis together)
cd /scratch/user/mahe0050/DE-analysis/0_rawData/AGRF_CAGRF20022_CDNJBANXX ||
↪  error_exit "$LINENO: directory error 2"

for file in *R1.fastq.gz
do
  echo $file
  FILESTEM=${file%_*}
  echo $FILESTEM
  FILESTEM2=`echo $file | cut -d "_" -f1,2 --output-delimiter="_"`
  echo $FILESTEM2

#(repeat of the above but edit directories & names: same output folder as all
↪  samples have unique names)
  #remove Illumina adapters from all paired data using bbduk
  bbduk.sh in=$file in2=$FILESTEM"_R2.fastq.gz"
  ↪  out=../../1_trimmedData/temp/$FILESTEM2"_R1_Iclean.fq.gz"
  ↪  out2=../../1_trimmedData/temp/$FILESTEM2"_R2_Iclean.fq.gz"
  ↪  outs=../../1_trimmedData/temp/$FILESTEM2"_singletons_Iclean.fq.gz"
  ↪  literal=GATCGGAAGAGCACA,AGATCGGAAGAGCGT ktrim=r k=15 mink=15 hdist=1
  ↪  threads=$THREADS || error_exit "$LINENO: Error removing Illumina adapters
  ↪  from R1 or R2 - directory 2"

  #remove SMARTer PCR tags from all paired data using bbduk
  bbduk.sh in=../../1_trimmedData/temp/$FILESTEM2"_R1_Iclean.fq.gz"
  ↪  in2=../../1_trimmedData/temp/$FILESTEM2"_R2_Iclean.fq.gz"
  ↪  out=../../1_trimmedData/$FILESTEM2"_R1_clean.fq.gz"
  ↪  out2=../../1_trimmedData/$FILESTEM2"_R2_clean.fq.gz"
  ↪  outs=../../1_trimmedData/$FILESTEM2"_singletons_clean.fq.gz"
  ↪  literal=AAGCAGTGGTATCAACGCAGAGTACNNNNN,GTACTCTGCGTTGATACCACTGCTT ktrim=r k=15
  ↪  mink=15 hdist=1 qtrim=r trimq=10 minlength=30 threads=$THREADS || error_exit
  ↪  "$LINENO: Error removing SMARTer adapters from R1 or R2 - directory 2"

  #remove SMARTer PCR tags from the singleton file using bbduk
  bbduk.sh in=../../1_trimmedData/temp/$FILESTEM2"_singletons_Iclean.fq.gz"
  ↪  out=../../1_trimmedData/$FILESTEM2"_singletons1.fq.gz"
  ↪  literal=AAGCAGTGGTATCAACGCAGAGTACNNNNN,GTACTCTGCGTTGATACCACTGCTT ktrim=r k=15
  ↪  mink=15 hdist=1 qtrim=r trimq=20 minlength=30 threads=$THREADS || error_exit
  ↪  "$LINENO: Error removing SMARTer adapters from singletons - directory 2"
```

```
  #concatenate the singleton files
  cat ../../1_trimmedData/$FILESTEM2"_singletons1.fq.gz" >>
  ↪  ../../1_trimmedData/$FILESTEM2"_singletons_clean.fq.gz" || error_exit
  ↪  "$LINENO: Error concatenating singletons - directory 2"


  #clean up: i.e. remove temporary directories
  rm ../../1_trimmedData/$FILESTEM2"_singletons1.fq.gz" || error_exit "$LINENO:
  ↪  Error deleting singleton intermediate - directory 2"
done

cd /scratch/user/mahe0050/DE-analysis/1_trimmedData
rm -rf temp || error_exit "$LINENO: Error removing temp directory"
```

Quality of trimmed data was again assessed again using FastQC v0.11.8.

```
#run cleaned data through fastqc
mkdir -p /scratch/user/mahe0050/DE-analysis/1_trimmedData/FastQC-clean

for file in *R1_clean.fq.gz
do
  echo $file
  FILESTEM2=`echo $file | cut -d "_" -f1,2 --output-delimiter="_"`
  echo "FILESTEM2" $FILESTEM2

  fastqc --noextract --threads $THREADS -o ./FastQC-clean
  ↪  ./$FILESTEM2"_R1_clean.fq.gz" ./$FILESTEM2"_R2_clean.fq.gz"
  ↪  ./$FILESTEM2"_singletons_clean.fq.gz" || error_exit "$LINENO: Error with
  ↪  FastQC at "$FILESTEM2""
done


#primers check
#zcat G1_KI_HCGYYBCXY_ATCACG_L001_R1.fastq.gz | grep "AGATCGGAAGAGCAC" -m10
#zcat G1_KI_HCGYYBCXY_ATCACG_L001_R1.fastq.gz | grep "GATCGGAAGAGCACA" -m10
#zcat G1_KI_HCGYYBCXY_ATCACG_L001_R2.fastq.gz | grep "AGATCGGAAGAGCGT" -m10
```

The output html file of all eight samples FastQC reports AFTER trimming is available here.

De-novo assembly with TRINITY

De-novo assembly of transcripts.
Short reads were assembled *de novo* using the program TRINITY v2.8.4. Sequencing run AGRF_CAGRF13871_HCGYYBCX involved the synthesis of a non-stranded cDNA library prior to sequencing, while sequencing run AGRF_CAGRF20022_CDNJBANXX was completed with a different kit. All samples were treated as non stranded for consistency. Singleton reads (where a paired read was eliminated in quality trimming) were included for all TRINITY assembly runs.

Assembly was run on each sample individually as well as on a pooled (concatenated) file of all reads.

The script trinity2019.sh

Single samples were run through Trinity as per the linked script above using the following:

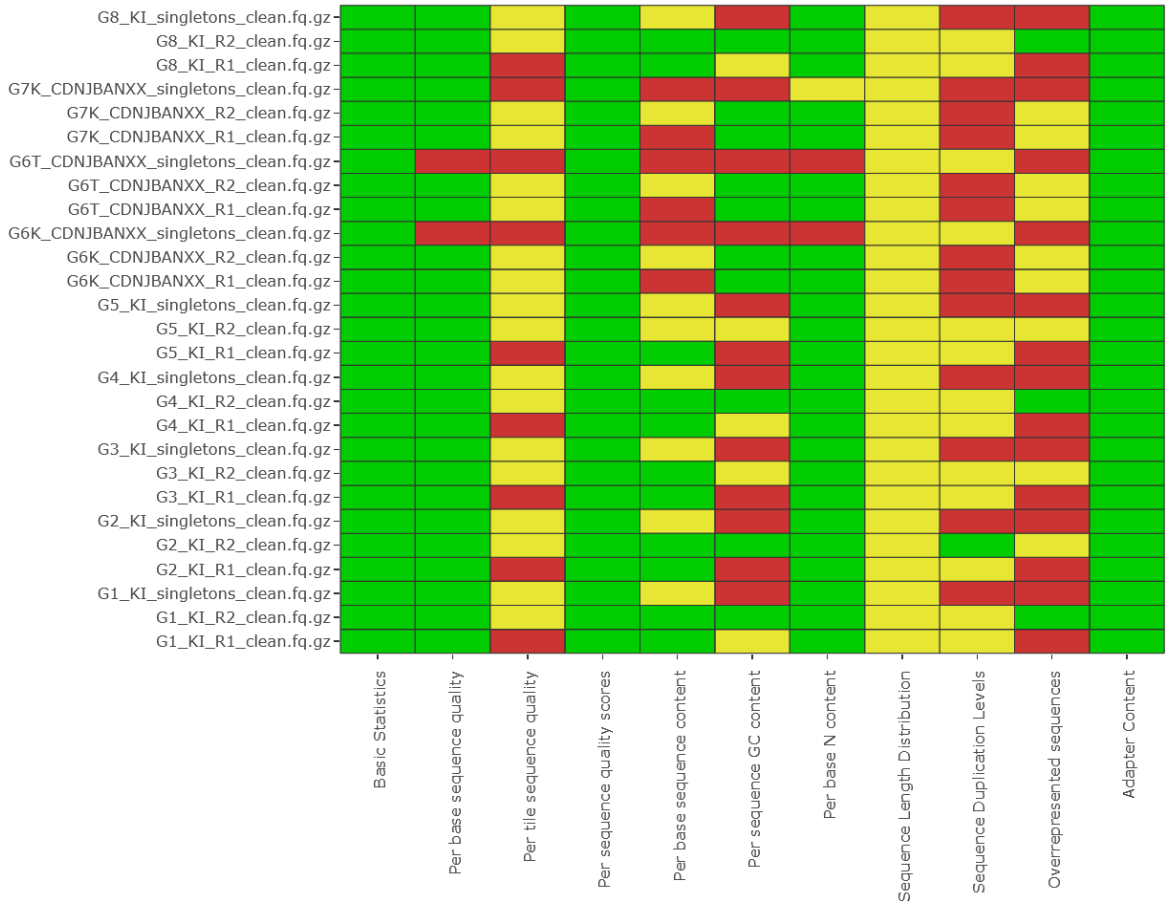

Figure 2: FastQC Summary of scores for paired reads of 8 samples and resulting 'singleton' files for trimmed reads with no pair, after adapter and quality trimming in BBduk

```
THREADS=16
MEM="64G"

cd /scratch/user/mahe0050/DE-analysis/1_trimmedData
```

To run trinity separately on each sample

```
mkdir -p ./trinity-sep
mkdir -p ./trinity-sep/temp

for file in *R1_clean.fq.gz
do
  FILESTEM=${file%_*}
  #this FILESTEM only cuts to _clean, the _R1 is included
  FILESTEM=${FILESTEM/R1/}
  #removes R1 from FILESTEM (FILESTEM ends in _ therefore not needed in "text"
  ↪  names)
```

concatenate relevant singleton file to R1 as library is not stranded?

```
  cat ./$file ./$FILESTEM"singletons_clean.fq.gz" >
  ↪  ./trinity-sep/$FILESTEM"concat_R1.fq.gz" || error_exit "$LINENO: Error
  ↪  concatenating R1 and singletons"
```

Run TRINITY to assemble transcripts from a single paired sample file. Note trinity requires output directory with "trinity" in name.

```
  Trinity --seqType fq --max_memory $MEM --bflyCalculateCPU --left
  ↪  ./trinity-sep/$FILESTEM"concat_R1.fq.gz" --right ./$FILESTEM"R2_clean.fq.gz"
  ↪  --CPU $THREADS --output ./trinity-sep/$FILESTEM"trinity"/trinity ||
  ↪  error_exit "$LINENO: Error running trinity-sep at $FILESTEM"
```

clean up

```
  rm -rf ./trinity-sep/$FILESTEM"concat_R1.fq.gz" || error_exit "$LINENO: Error
  ↪  removing trinity-sep concat at $FILESTEM"

done
echo "Trinity separate samples complete"
```

All kidney samples were run through Trinity together for a single large assembly as per the linked script above above using the following:

Sample files to be concatenated and named appropriately

```
# create temp & working dir
mkdir -p ./trinity-all
mkdir -p ./trinity-all/temp
```

concatenate relevant sample files for one transcriptome - add singletons to R1 as its not stranded

```
    cat ./*_R1_clean.fq.gz ./*_singletons_clean.fq.gz >>
    ↪  ./trinity-all/temp/PBTKI_cat_R1.fq.gz || error_exit "$LINENO: Error
    ↪  concatenating R1 and singletons"
    cat ./*_R2_clean.fq.gz >> ./trinity-all/temp/PBTKI_cat_R2.fq.gz || error_exit
    ↪  "$LINENO: Error concatenating R2"
```

Run TRINITY

```
    Trinity --seqType fq --max_memory $MEM --bflyCalculateCPU --left
    ↪  ./trinity-all/temp/PBTKI_cat_R1.fq.gz --right
    ↪  ./trinity-all/temp/PBTKI_cat_R2.fq.gz --CPU $THREADS --output
    ↪  ./trinity-all/trinity || error_exit "$LINENO: Error running Trinity-all"
```

clean up

```
    rm -rf ./trinity-all/temp || error_exit "$LINENO: Error removing trinity-all temp
    ↪  directory"
  echo "Trinity concatenated kidney samples complete"
```

TRINITY outputs each samples assembly as Trinity.fasta, within a /trinity output folder, within the folder assigned with a sample name. In order to compare different Trinity.fasta results in one directory the following script was used to copy and rename these final files and place them in a common folder TrinityName.sh.

This was done to avoid any errors in manual handling of files and it is these files which are used later in a BUSCO comparison of assemblies.

Summary / Quantification

Long read sequences data summary

15,729 Total transcript isoforms retained in initial cleaning and clustering of redundant isoforms.

13,882 sequences in a predicted open reading frame.

9,907 clusters identified based on translated proteins of predicted open reading frame.

9,813 full length transcripts have been subset into a reference file as the longest representatives of one or more transcript clusters.

Short read sequences data summary

Total Transcript count of each TRINITY generated dataset was simply conducted using shell commands. Total transcript count for long read files are included in the BUSCO usage file below as headers required manipulation for input into BUSCO and totals were checked at this step.

Sample ID	Experimental factors	File name	Total transcripts/sequences assembled
G1	September Female	G1_KI_trinity	46538
G2	September Female	G2_KI_trinity	43283
G3	March/April Female	G3_KI_trinity	35991

Sample ID	Experimental factors	File name	Total transcripts/sequences assembled
G4	March/April Male	G4_KI_trinity	39239
G5	September Male	G5_KI_trinity	37379
G6	September Male	G6K_CDNJBANXX_trinity	214799
G7	March/April Female	G7K_CDNJBANXX_trinity	266539
G8	March/April Female	G8_KI_trinity	43050
All Samples combined	n/a	trinity-all	373287

BUSCO

Completeness of assemblies were assessed using Benchmarking Universal Single-Copy Orthologs BUSCO version 5.0 (BUSCO) and the dataset vertebrata_odb10.

Moving finalised assembly files into the BUSCO Directory:
TRINITY automatically names the final outputs "trinity.fasta" within folders which have the sample ID. Therefore for all these files to be moved into one directory for BUSCO to reference, these sequence IDs from the folders must be appended to the file names.

Conda Version 4.8.3 has been used for all analyses up to this point. Updated conda as version 4.8.4 or higher is recommended for BUSCO. This analysis was conducted on eRSA's NECTAR research cloud

Working directory of full (single line) hq transcript Isoseq dataset, and trimmed, non-redundant file of transcripts identified as cluster representatives.

```
/mnt/IsoSeq-analysis/data/Seqtk/hq.fasta.no5merge.collapsed.rep.1L.fa
/mnt/IsoSeq-analysis/data/Seqtk/reference_transcripts.1Lv.clean.fasta
```

Working directory of assembled (using TRINITY) short read transcript files

```
/mnt/DE-analysis/2_alignedData/trinity
```

concatenated short reads all assembled using trinity into largest transcript file/mnt/DE-analysis/2_alignedData/trinity/trinity-all/trinity

```
/mnt/DE-analysis/2_alignedData/trinity/trinity-all/trinity
# Then under
# /trinity-sep/<sample_ID_trinity>/trinity

  # Sample IDs folders:
  #   G1_KI_trinity
  #   G2_KI_trinity
  #   G3_KI_trinity
  #   G4_KI_trinity
  #   G5_KI_trinity
  #   G6K_CDNJBANXX_trinity
  #   G7K_CDNJBANXX_trinity
  #   G8_KI_trinity
```

This was done using the script output-BUSCO_Name2020.sh which also moved the long-read assemblies into the correct path:

```
#!/bin/bash
#
#
# This script is used to rename and copy trinity output files with their input file
↪  "filestem" into the /BUSCO directory so all files have similar naming
↪  convention for input into the BUSCO script.
#
# The paths in the script assume that a specific directory structure has been set
↪  up.
#
# Modules required: none
#
# usage:  bash Output-BUSCO_Name2020.sh
# location:           cd /scratch/user/mahe0050/BUSCO/

# Carmel Maher
# August 2020


#------adjust these for your run-----

#-------------------------------------


function error_exit
{
    # Exit function due to fatal error
    # Accepts 1 arg:
    # string - descriptive error message

    echo "${PROGNAME}: ${1:-"Unknown error"}" 1>&2
    exit 1
}


# go to the working directory - working from /clean
cd /scratch/user/mahe0050/DE-analysis/1_trimmedData  || error_exit "$LINENO:
↪  Directory Error 1"

for file in *R1_clean.fq.gz
  do

      FILESTEM=${file%_*}
      #this FILESTEM only cuts to _clean, the _R1 is included in stem
      FILESTEM=${FILESTEM/R1/}
      #removes R1 from FILESTEM (FILESTEM ends in _ therefore not needed in "text"
      ↪  names)

      cp -i
      ↪  /scratch/user/mahe0050/DE-analysis/2_alignedData/trinity-sep/$FILESTEM"trinity"/trinity.
      ↪  /scratch/user/mahe0050/BUSCO/$FILESTEM"t_assembled".fasta  || error_exit
      ↪  "$LINENO: Error copying separate trinity outputs"
```

```
            # -> all files should end up in /scratch/user/mahe0050/BUSCO named
            ↪  G#_XX.t_assembled.fasta

    done

    echo "Trinity-sep done"

cp -i
↪  /scratch/user/mahe0050/DE-analysis/2_alignedData/trinity-all/trinity/Trinity.fasta
↪  /scratch/user/mahe0050/BUSCO/trinity-all_assembled.fasta  || error_exit
↪  "$LINENO: Error copying concatenated trinity-all output"

      # -> one ANGEL file should end up in /scratch/user/mahe0050/BUSCO named
      ↪  trinity-all_assembled.fasta

    echo "Trinity-all done"

cp -i
↪  /scratch/user/mahe0050/IsoSeq-analysis/data/Cogent/collected/hq.fasta.no5merge.collapsed.rep.
↪  /scratch/user/mahe0050/BUSCO/hq.fasta.no5merge.collapsed.rep_assembled.fasta
↪  || error_exit "$LINENO: Error copying Cogent output"

      # -> one Cogent file should end up in /scratch/user/mahe0050/BUSCO named
      ↪  hq.fasta.no5merge.collapsed.rep_assembled.fasta

    echo "Cogent done"

cp -i /scratch/user/mahe0050/IsoSeq-analysis/data/ANGEL/pygmy.ANGEL.cds
↪  /scratch/user/mahe0050/BUSCO/pygmy.ANGEL_assembled.fasta  || error_exit
↪  "$LINENO: Error copying ANGEL output"

      # -> one ANGEL file should end up in /scratch/user/mahe0050/BUSCO named
      ↪  pygmy.ANGEL_assembled.fasta

    echo "ANGEL done"


# all file sizes checked after run = matches
# Trinity outputs removed from scratch 20/12/2020 to save space
```

Editing .fasta headers so that BUSCO can input the files

This BUSCO_all_v5.0.sh document contains notes and single lines run to run BUSCO 5.0 installed using the conda package on all short read and long read assembled transcript files. It outlines how to format the input files in preparation for BUSCO and the usage of the BUSCO script.

Sample format compatibility

```
cd /mnt/IsoSeq-analysis/BUSCO/

# NOTE the Isoseq files have symbols in fasta headers before transcript IDs which
↪  need to be removed for BUSCO
```

Header formats:

```
head /mnt/IsoSeq-analysis/data/Seqtk/hq.fasta.no5merge.collapsed.rep.1L.fa
>PB.4.2|004079|path1:6-612(+)|transcript/24311 transcript/24311
↪  full_length_coverage=2;length=640;num_subreads=60

head /mnt/IsoSeq-analysis/data/Seqtk/reference_transcripts.1Lv.clean.fasta
>PB.11.2|004815|path2:1047-3035(+)|transcript/13401 transcript/13401
↪  full_length_coverage=2;length=1995;num_subreads=8

head /mnt/IsoSeq-analysis/data/ANGEL/pygmy.ANGEL.cds
>PB.11.2|004815|path2:1047-3035(+)|transcript/13401|m.5 type:dumb-complete len:311
↪  strand:+ pos:214-1146
```

Total number of sequences:

```
grep -c ">" /mnt/IsoSeq-analysis/data/Seqtk/hq.fasta.no5merge.collapsed.rep.1L.fa
15729
grep -c ">" /mnt/IsoSeq-analysis/data/Seqtk/reference_transcripts.1Lv.clean.fasta
9813
grep -c ">" /mnt/IsoSeq-analysis/data/ANGEL/pygmy.ANGEL.cds
13882
```

after sed to remove all "/" resulted in an error at metaeuk headers "ValueError: could not convert string to float: '+' ", after sed to remove all "+" same error.

Goal is a %score of alignments to BUSCOs and which transcripts match is no explored further. Therefore all headers were grossly simplified. The files created here are ONLY used for BUSCO.

```
cut -d '|' -f1
↪  /mnt/IsoSeq-analysis/data/Seqtk/reference_transcripts.1Lv.clean.fasta >
↪  reference_transcripts.1Lv.clean_.fasta

cut -d '|' -f1 /mnt/IsoSeq-analysis/data/ANGEL/pygmy.ANGEL.cds > pygmy.ANGEL_.cds

cut -d '|' -f1
↪  /mnt/IsoSeq-analysis/data/Seqtk/hq.fasta.no5merge.collapsed.rep.1L.fa >
↪  hq.fasta.no5merge.collapsed.rep.1L_.fa
```

Total number of sequences retained as counted above

```
grep -c ">" reference_transcripts.1Lv.clean_.fasta
9813
grep -c ">" pygmy.ANGEL_.cds
13882
grep -c ">" hq.fasta.no5merge.collapsed.rep.1L_.fa
15729
```

Two of these files will have "duplicate" headers in this simplified format. Numbers were simply appended as below so that all values are unique. These files are for use in BUSCO *only*:

```
awk '/^>/{$0=$0"_"(++i)}1'  pygmy.ANGEL_.cds > pygmy.ANGEL__.cds
awk '/^>/{$0=$0"_"(++i)}1'  hq.fasta.no5merge.collapsed.rep.1L_.fa >
↪  hq.fasta.no5merge.collapsed.rep.1L__.fa
```

```
grep -c ">" pygmy.ANGEL__.cds
13882
grep -c ">" hq.fasta.no5merge.collapsed.rep.1L__.fa
15729
```

BUSCO Installation & usage:

The script run-conda-BUSCO.bash was used to run the program BUSCO on all samples on the NECTAR machine.

```
cd /mnt/IsoSeq-analysis/BUSCO

#Create a conda environment for the BUSCO Install

conda create -n conda-BUSCO -c bioconda -c conda-forge busco=5.0.0
conda activate conda-BUSCO
```

The basic repeated usage of BUSCO for all samples listed below will be:

```
busco -i [SEQUENCE_FILE] -l [LINEAGE] -o [OUTPUT_NAME] -m transcriptome


# note: E-value cutoff for BLAST searches Default: 1e-03

# busco --list-datasets
#check $LINEAGE in script file
#     - vertebrata_odb10
#     may also --auto-lineage-euk

# script currently set to threads (-c) = 12
```

Run BUSCO -> usage

```
# location of usage doesnt matter as paths are specified within script.
# use from within /mnt/IsoSeq-analysis/BUSCO/ in case of log files.
# Usage:
          conda activate conda-BUSCO
          screen
          bash run-conda-BUSCO.bash 2>&1 | tee BUSCO-out.txt
```

run-conda-BUSCO.bash:

```
#!/bin/bash
#
#
# This script is used to assess transcriptome outputs against BUSCO lineages
# The paths in the script assume that a specific directory structure has been set
↪  up.
# Analysis run on eRSA NECTAR machine
#
```

```
#Usage:
#             conda activate conda-BUSCO
#         screen
#             bash run-conda-BUSCO.bash 2>&1 | tee BUSCO-out.txt

# Carmel Maher
# February 2021


# See notes file associated with usage: BUSCO_all_v5.0.sh
# Inputs include:
#     Short-read Trinity outputs for all 8 Kidney samples assembled from short reads
#     Short-read Trinity output for one file of all 8 kidney sequencing concatenated and assembled into a single transcript file
#     Isoseq3 long-read de novo assembly full list of hq transcripts
#     Isoseq3 long-read de novo assembly list of non-redundant representative transcripts of peptide clusters, with additional poly-a tail trimming


#------adjust these for your run-----

LINEAGE="vertebrata_odb10"

#-------------------------------------

function error_exit
{
    echo "${PROGNAME}: ${1:-"Unknown error"}" 1>&2
    exit 1
}

# (commented out because it failed after "done" - rerun with # )

#go to the directory containing trimmed files to pull ID names
cd /mnt/DE-analysis/1_trimmedData

for file in *R1_clean.fq.gz
  do
      FILESTEM=${file%_*}
      #this FILESTEM only cuts to _clean, the _R1 is included in stem
      FILESTEM=${FILESTEM/R1/}
      #removes R1 from FILESTEM (FILESTEM ends in _ therefore not needed in "text" names)

      mkdir /mnt/IsoSeq-analysis/BUSCO/$FILESTEM"BUSCOout" || error_exit "$LINENO: Error creating trinity-sep output directory at $FILESTEM"

      cd /mnt/IsoSeq-analysis/BUSCO/

      busco -i /mnt/DE-analysis/2_alignedData/trinity/trinity-sep/$FILESTEM"trinity"/trinity/Trinity.fa
      -l $LINEAGE -f -c 12 -o $FILESTEM"BUSCOout" -m transcriptome ||
      error_exit "$LINENO: Error running BUSCO at $FILESTEM"
```

```
      cd /mnt/DE-analysis/1_trimmedData
  done

#

cd /mnt/IsoSeq-analysis/BUSCO/

mkdir ./trinity-all_BUSCOout  || error_exit "$LINENO: directory error at
↪  trinity-all"

busco -i /mnt/DE-analysis/2_alignedData/trinity/trinity-all/trinity/Trinity.fasta
↪  -l $LINEAGE -f -c 12 -o trinity-all_BUSCOout -m transcriptome || error_exit
↪  "$LINENO: Error running BUSCO at trinity-all"

#

cd /mnt/IsoSeq-analysis/BUSCO/

mkdir ./reference-transcripts_BUSCOout || error_exit "$LINENO: directory error at
↪  reference-transcripts"

busco -i reference_transcripts.1Lv.clean_.fasta -l $LINEAGE -f -c 12 -o
↪  reference-transcripts_BUSCOout -m transcriptome || error_exit "$LINENO: Error
↪  running BUSCO at reference-transcripts"

#

mkdir ./ANGEL.cds_BUSCOout || error_exit "$LINENO: directory error at
↪  reference-transcripts"

busco -i pygmy.ANGEL__.cds -l $LINEAGE -f -c 12 -o ANGEL.cds_BUSCOout -m
↪  transcriptome || error_exit "$LINENO: Error running BUSCO at ANGEL.cds"

#

mkdir ./hq-fasta_BUSCOout || error_exit "$LINENO: directory error at hq-fasta"

busco -i hq.fasta.no5merge.collapsed.rep.1L__.fa -l $LINEAGE -f -c 12 -o
↪  hq-fasta_BUSCOout -m transcriptome || error_exit "$LINENO: Error running BUSCO
↪  at hq-fasta"

#

echo "all BUSCOs complete"
```

Visualisation

BUSCO outputs a file named short_summary.specific.vertebrata_odb10.<-o>.txt. These are used to plot a summary figure of all samples.

generate plot.py and r script of all sample BUSCO summaries

```
mkdir BUSCO_summaries

cp G1_KI_BUSCOout/short_summary.*_odb10.G1_KI_BUSCOout.txt BUSCO_summaries/.
cp G2_KI_BUSCOout/short_summary.*_odb10.G2_KI_BUSCOout.txt BUSCO_summaries/.
cp G3_KI_BUSCOout/short_summary.*_odb10.G3_KI_BUSCOout.txt BUSCO_summaries/.
cp G4_KI_BUSCOout/short_summary.*_odb10.G4_KI_BUSCOout.txt BUSCO_summaries/.
cp G5_KI_BUSCOout/short_summary.*_odb10.G5_KI_BUSCOout.txt BUSCO_summaries/.
cp G6K_CDNJBANXX_BUSCOout/short_summary.*_odb10.G6K_CDNJBANXX_BUSCOout.txt
↪  BUSCO_summaries/.
cp G7K_CDNJBANXX_BUSCOout/short_summary.*_odb10.G7K_CDNJBANXX_BUSCOout.txt
↪  BUSCO_summaries/.
cp G8_KI_BUSCOout/short_summary.*_odb10.G8_KI_BUSCOout.txt BUSCO_summaries/.
cp
↪  reference-transcripts_BUSCOout/short_summary.*_odb10.reference-transcripts_BUSCOout.txt
↪  BUSCO_summaries/.
cp ANGEL.cds_BUSCOout/short_summary.*_odb10.ANGEL.cds_BUSCOout.txt
↪  BUSCO_summaries/.
cp hq-fasta_BUSCOout/short_summary.*_odb10.hq-fasta_BUSCOout.txt BUSCO_summaries/.
cp trinity-all_BUSCOout/short_summary.*_odb10.trinity-all_BUSCOout.txt
↪  BUSCO_summaries/.

#(The path below is where the config files for conda-BUSCO environment are located)

python3 /mnt/Prog/miniconda3/envs/conda-BUSCO/bin/generate_plot.py -wd
↪  /mnt/IsoSeq-analysis/BUSCO/BUSCO_summaries
```

Short Read summaries:

```
mkdir BUSCO_SR_summaries

cp G1_KI_BUSCOout/short_summary.*_odb10.G1_KI_BUSCOout.txt BUSCO_SR_summaries/.
cp G2_KI_BUSCOout/short_summary.*_odb10.G2_KI_BUSCOout.txt BUSCO_SR_summaries/.
cp G3_KI_BUSCOout/short_summary.*_odb10.G3_KI_BUSCOout.txt BUSCO_SR_summaries/.
cp G4_KI_BUSCOout/short_summary.*_odb10.G4_KI_BUSCOout.txt BUSCO_SR_summaries/.
cp G5_KI_BUSCOout/short_summary.*_odb10.G5_KI_BUSCOout.txt BUSCO_SR_summaries/.
cp G6K_CDNJBANXX_BUSCOout/short_summary.*_odb10.G6K_CDNJBANXX_SR_BUSCOout.txt
↪  BUSCO_summaries/.
cp G7K_CDNJBANXX_BUSCOout/short_summary.*_odb10.G7K_CDNJBANXX_SR_BUSCOout.txt
↪  BUSCO_summaries/.
cp G8_KI_BUSCOout/short_summary.*_odb10.G8_KI_BUSCOout.txt BUSCO_SR_summaries/.
cp trinity-all_BUSCOout/short_summary.*_odb10.trinity-all_BUSCOout.txt
↪  BUSCO_SR_summaries/.

python3 /mnt/Prog/miniconda3/envs/conda-BUSCO/bin/generate_plot.py -wd
↪  /mnt/IsoSeq-analysis/BUSCO/BUSCO_SR_summaries
```

Long Read summaries:

```
mkdir BUSCO_LR_summaries

cp
↪  reference-transcripts_BUSCOout/short_summary.*_odb10.reference-transcripts_BUSCOout.txt
↪  BUSCO_LR_summaries/.
```

```
cp ANGEL.cds_BUSCOout/short_summary.*_odb10.ANGEL.cds_BUSCOout.txt
↪  BUSCO_LR_summaries/.
cp hq-fasta_BUSCOout/short_summary.*_odb10.hq-fasta_BUSCOout.txt
↪  BUSCO_LR_summaries/.

python3 /mnt/Prog/miniconda3/envs/conda-BUSCO/bin/generate_plot.py -wd
↪  /mnt/IsoSeq-analysis/BUSCO/BUSCO_LR_summaries
```

These files for each sample were also moved into an R working directory to allow the generated figure to be edited.
The R script produced by the python command above was edited to allow for local directory structure and streamline the labelling and order of sample names.

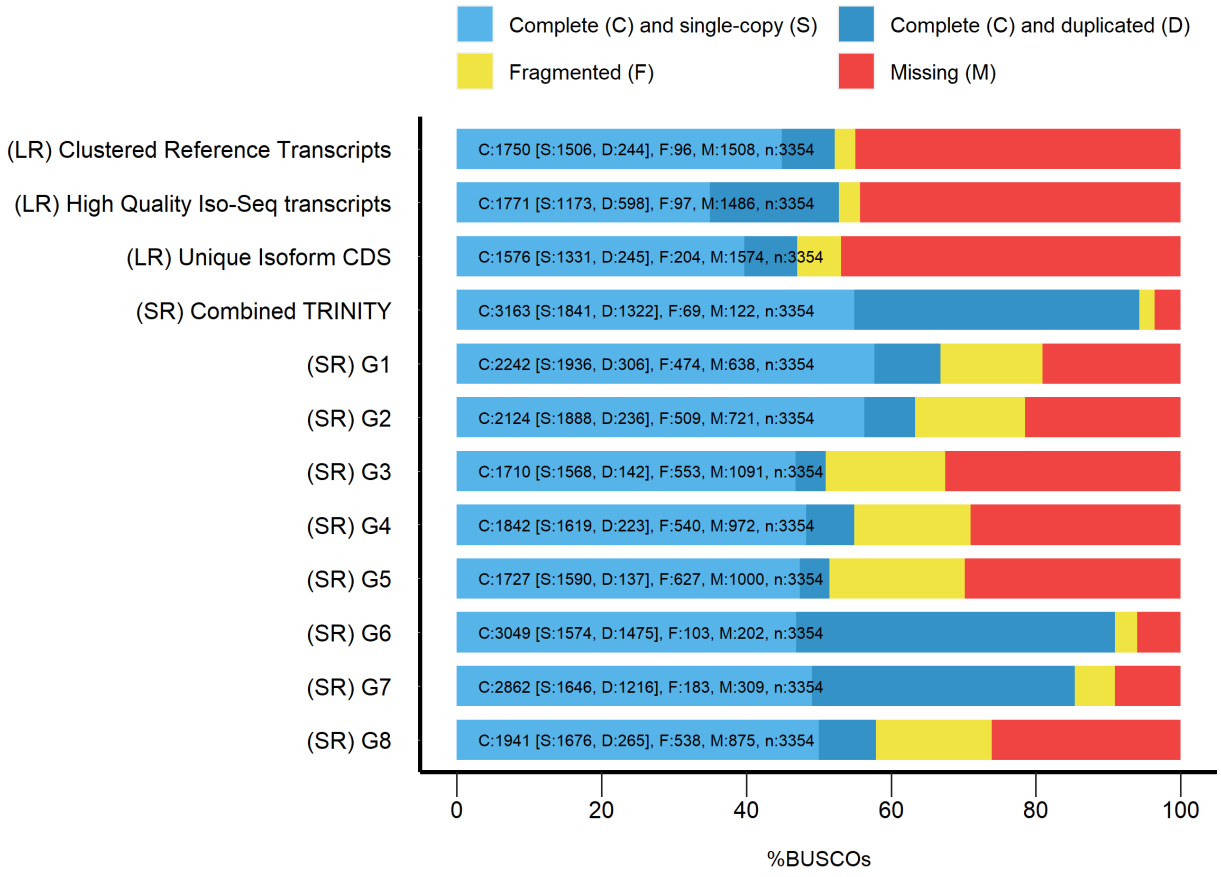

Figure 3: Busco Summary Statistics (BUSCO v5.0.0 run on database vertebrata_odb10)

Annotation (Chapter 5)

Completed on the long read data from Chapter 4

- pygmy.ANGEL.cds
- pygmy.ANGEL.pep
- pygmy.ANGEL.utr

BLASTx

BLASTx searches were conducted on the dataset at this stage to include isoforms that are collapsed later on.

BLASTx was run on the pygmy.ANGEL.cds file as it contains the predicted open reading frame sequence of nucleotides for translateable proteins. This was completed on the NECTAR machine.

```
# Package: blast 2.9.0, build Mar 11 2019 15:20:05
# Uniprot sprot & trembl databases downloaded: 28/5/19     (note: trembl database
↪  not indexed or used)
# Anolis .pep fasta files downloaded 12/11/19
      Anolis_carolinensis.AnoCar2.0.pep.all.fa
      Anolis_carolinensis.AnoCar2.0.pep.abinitio.fa

# To format a database: #
# makeblastdb -in mydb.fsa -dbtype nucl -parse_seqids
```

UniProt SwissProt database:

```
/mnt/Prog/blast/ncbi-blast-2.9.0+/bin/makeblastdb -in uniprot_sprot.fasta
↪  -parse_seqids -blastdb_version 5 -title "sprot" -dbtype prot -out sprot
```

Anolis carolinensis proteins database:

```
#AnoCar2.0.pep.all
/mnt/Prog/blast/ncbi-blast-2.9.0+/bin/makeblastdb -in
↪  Anolis_carolinensis.AnoCar2.0.pep.all.fa -parse_seqids -blastdb_version 5 -title
↪  "sprot" -dbtype prot -out AnoCar2.0.pep.all
```

Anolis Carolinensis database *ab initio*:

```
#AnoCar2.0.pep.abinitio
/mnt/Prog/blast/ncbi-blast-2.9.0+/bin/makeblastdb -in
↪  Anolis_carolinensis.AnoCar2.0.pep.abinitio.fa -parse_seqids -blastdb_version 5
↪  -title "sprot" -dbtype prot -out AnoCar2.0.pep.abinitio
```

Various searches were run in a screen to examine the effects of parameters and databases. Final searches are as listed below.

Searches included all databases listed above and variations of the parameters: -max_target_seqs 5 or -max_target_seqs 1 -max_hsps 5 or -max_hsps 1 -evalue 0.00001 or -evalue 1e-10

The NCBI Blast results most often referenced in the thesis are final parameters blastx -fmt6 -maxtarget1 -maxhsps1 -eval0.00001 against the UniProt SwissProt database.

```
# uniprot sprot
/mnt/Prog/blast/ncbi-blast-2.9.0+/bin/blastx -db
↪ /mnt/Prog/blast/blastdb/sprot/sprot -query
↪ /mnt/IsoSeq-analysis/data/ANGEL/pygmy.ANGEL.cds -out
↪ pygmy.ANGEL_blastx_sprot-maxtarg1-maxhsp1.out -outfmt 6 -max_target_seqs 1
↪ -max_hsps 1 -evalue 0.00001 -num_threads 3
```

```
#AnoCar2.0.pep.all
/mnt/Prog/blast/ncbi-blast-2.9.0+/bin/blastx -db
↪ /mnt/Prog/blast/blastdb/AcarProt/AnoCar2.0.pep.all/AnoCar2.0.pep.all -query
↪ /mnt/IsoSeq-analysis/data/ANGEL/pygmy.ANGEL.cds -out
↪ pygmy.ANGEL_blastx_AnoCar.pep.all-maxtarg1-maxhsp1.out -outfmt 6
↪ -max_target_seqs 1 -max_hsps 1 -evalue 0.00001 -num_threads 3
```

All searches except the below were output in tab separated -outfmt 6. Due to import requirements an -outfmt 5 an .xml file was required for results to be visualised in the program BLAST2GO (-max_target_seqs 5 -max_hsps 5 -evalue 0.00001).

```
/mnt/Prog/blast/ncbi-blast-2.9.0+/bin/blastx -db
↪ /mnt/Prog/blast/blastdb/AcarProt/AnoCar2.0.pep.all/AnoCar2.0.pep.all -query
↪ /mnt/IsoSeq-analysis/data/ANGEL/pygmy.ANGEL.cds -out
↪ pygmy.ANGEL_blastx_AnoCar.pep.all-BLAST2GO -outfmt 5 -max_target_seqs 5
↪ -max_hsps 5 -evalue 0.00001 -num_threads 4
```

The above and other trial searches are all listed in the file IsoSeq analysis post-ANGEL blast notes.sh

BLAST 2 GO

BLAST2GO (Within OmicsBox – Bioinformatics Made Easy, BioBam Bioinformatics, March 3, 2019) was used to perform a gene ontology analysis of the above BLASTx results.

Götz S., Garcia-Gomez JM., Terol J., Williams TD., Nagaraj SH., Nueda MJ., Robles M., Talon M., Dopazo J. and Conesa A. (2008). High-throughput functional annotation and data mining with the Blast2GO suite. Nucleic acids research, 36(10), 3420-35.

For import into the program a BLASTx search with the output set to .xml was performed using the fasta file containing the predicted open reading frame compared to the Anolis carolinensis proteins database (-max_target_seqs 5 -max_hsps 5 -evalue 0.00001).

Comparison to GO databases and visualisation was all completed within a Windows x64 v1.2.4 build of the OmicsBox program and figures were exported as below.

Summary

15,729 Total transcript isoforms retained in initial cleaning and clustering of redundant isoforms.

13,882 sequences in a predicted open reading frame.

9,907 clusters identified based on translated proteins of predicted open reading frame.

9,813 full length transcripts have been subset into a reference file as the longest representatives of one or more transcript clusters.

12,602 of the 13,882 sequences in a predicted open reading frame were successfully annotated to a BLASTx result, protein ID, and gene ontology category.

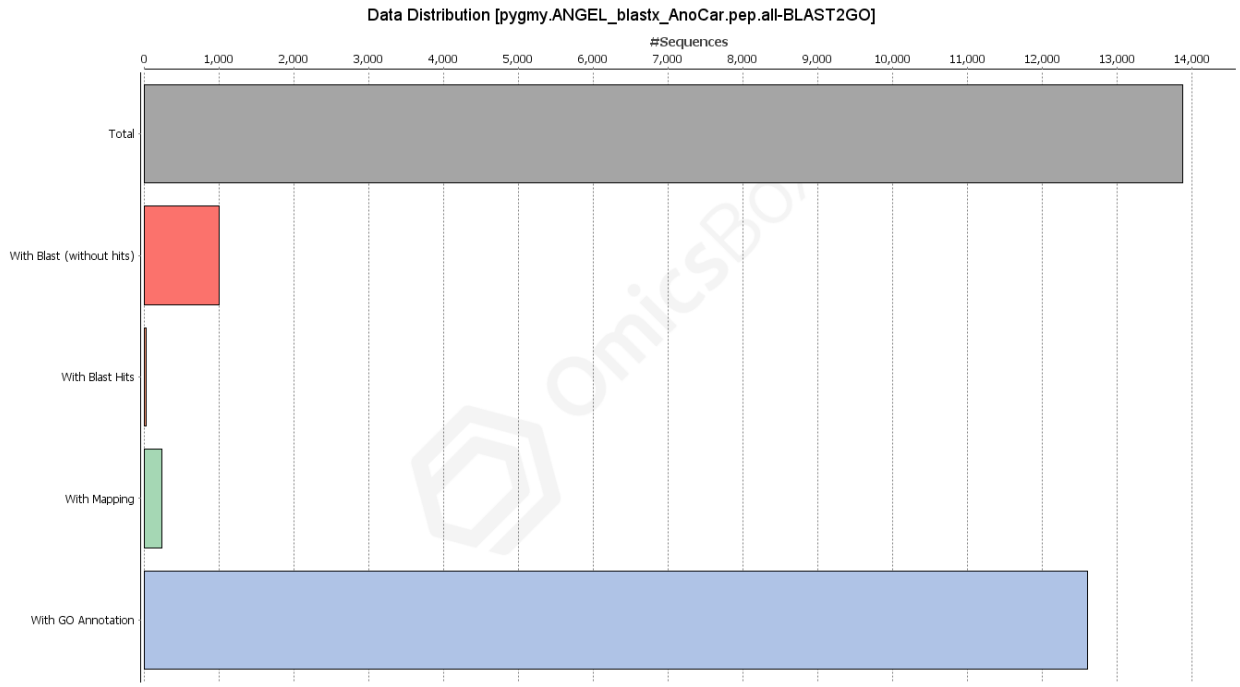

Figure 4: Summary data distribution. pygmy.ANGEL contains 13,882 unique sequences representing predicted open reading frame of transcripts: 12,602 of these sequences produced >1 BLASTx hit with Mapping and GO Annotation, 243 produced a BLASTx hit and mapping only, 32 produced a BLAST hit only, and 1005 did not produce BLASTx hits

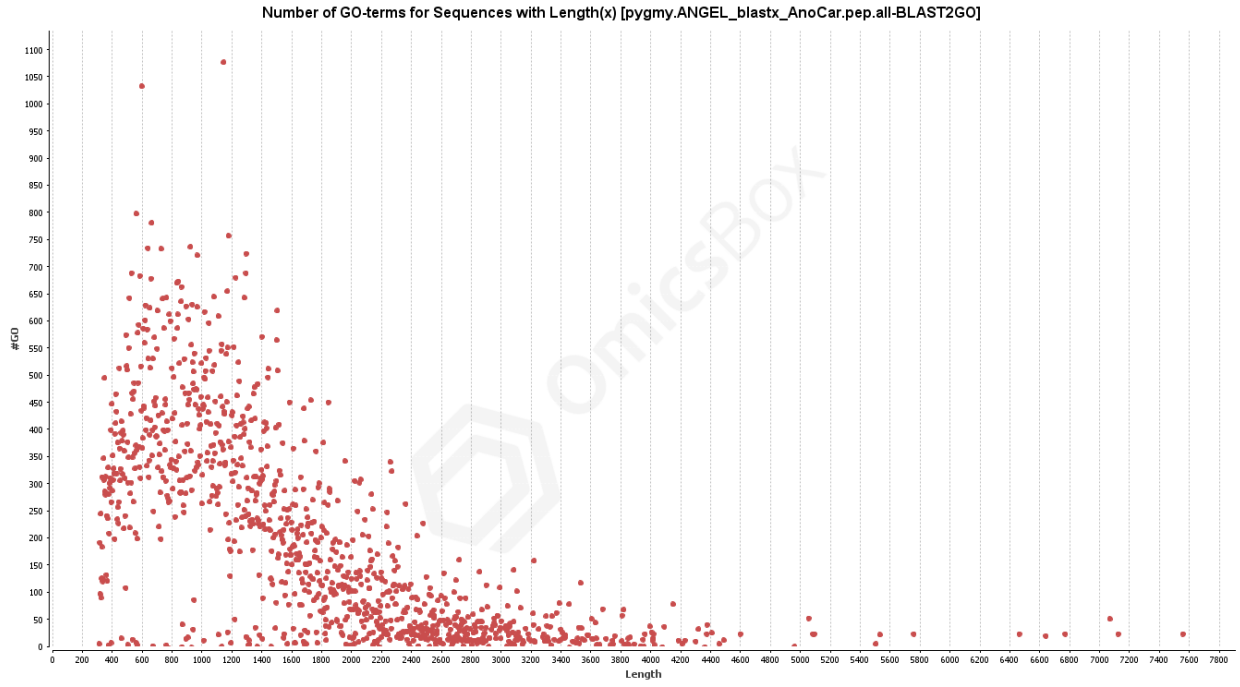

Figure 5: Number of GO-terms identified for sequences with length (x). Length is based on nucleotide sequence input into the BLASTx search

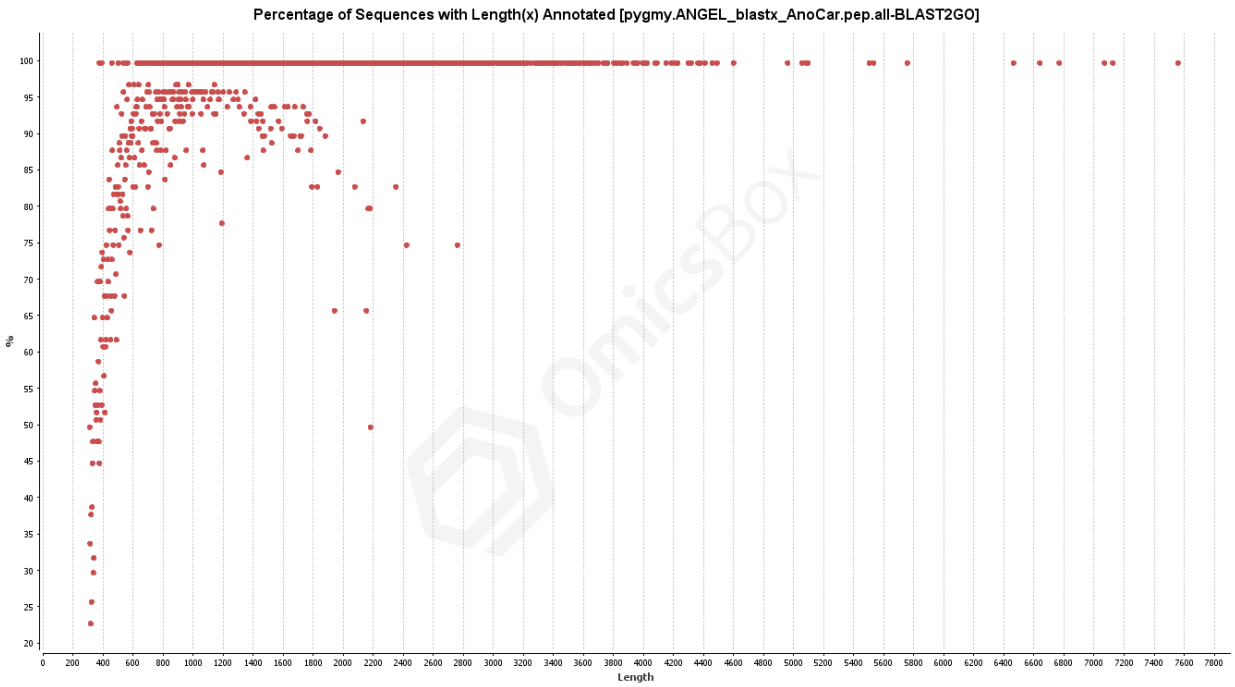

Figure 6: Percentage of sequences with length (x) that were annotated

---

Gene expression analysis (Chapter 6)

Long reads (reference index) - Previous analysis:

- Isoseq3 pipe
- Collapsing of isoforms & generating fasta of longest transcripts in predicted ORFs (Cogent/ANGEL)
- BLASTx search of coding sequence file for the longest transcripts in predicted ORFs
- Grouping of transcript clusters/putative gene families (CD-HIT)
- Longest representative transcript collapsed into a reference .fasta (R & Shell scripts)
- As this longest transcript was taken from earlier in the pipeline persisting poly-a tails trimmed from reference .fasta file (Shell)

Short reads - Previous analysis:

- FASTQC quality assessment
- BBduk trimming
- FASTQC quality assessment (note: TRINITY assembly not used for counts)

Both datasets are combined in this analysis:

- Transcript counts per million estimated using Kallisto
  - Clean short reads (above) counted against reference index generated from long read .fasta (above)
- Kallisto output imported into R for analysis using EdgeR

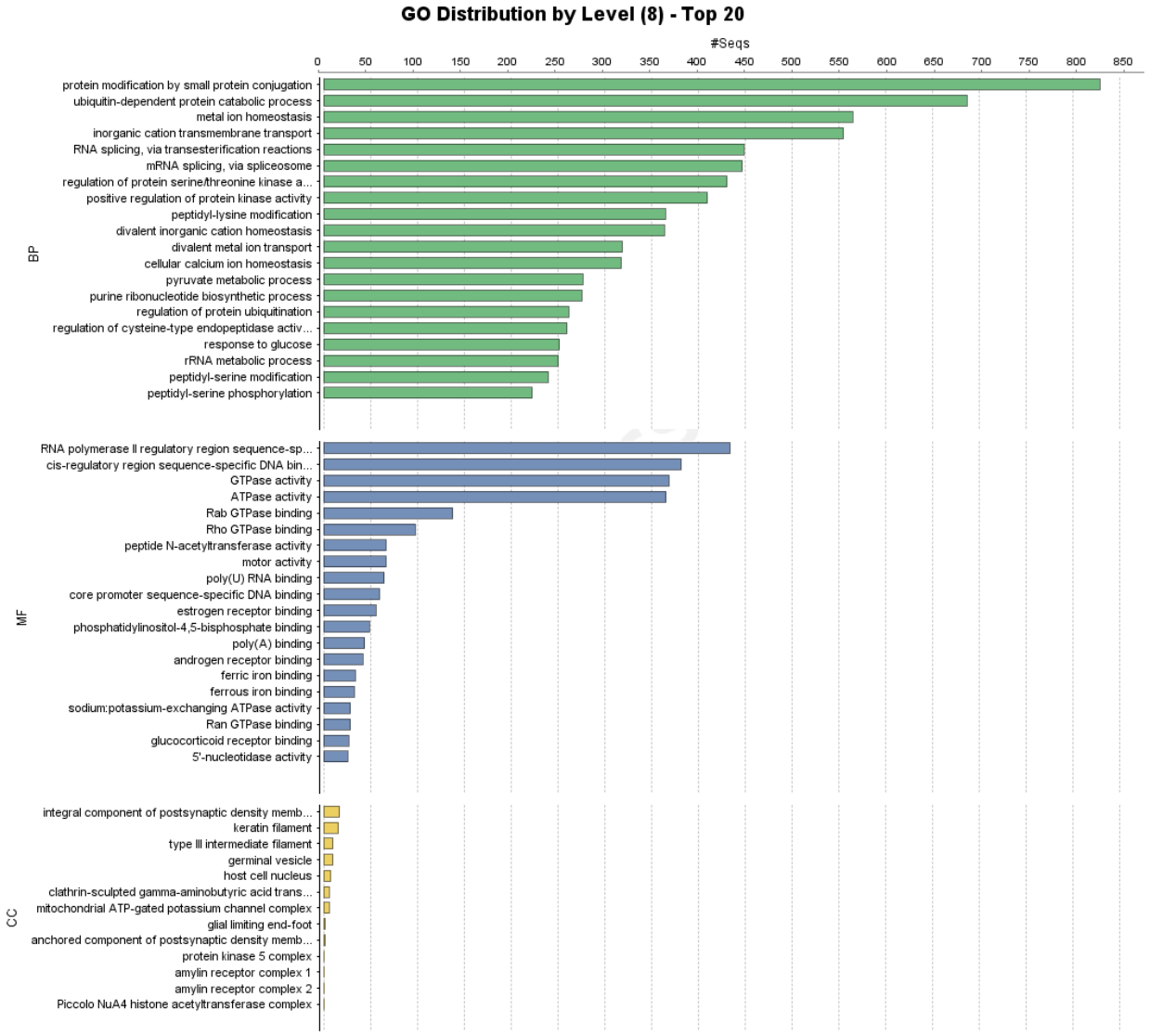

Figure 7: Top 20 annotation results (at level 8) for each category BP - Biological Process, MF - Molecular Function, and CC - Celular Component

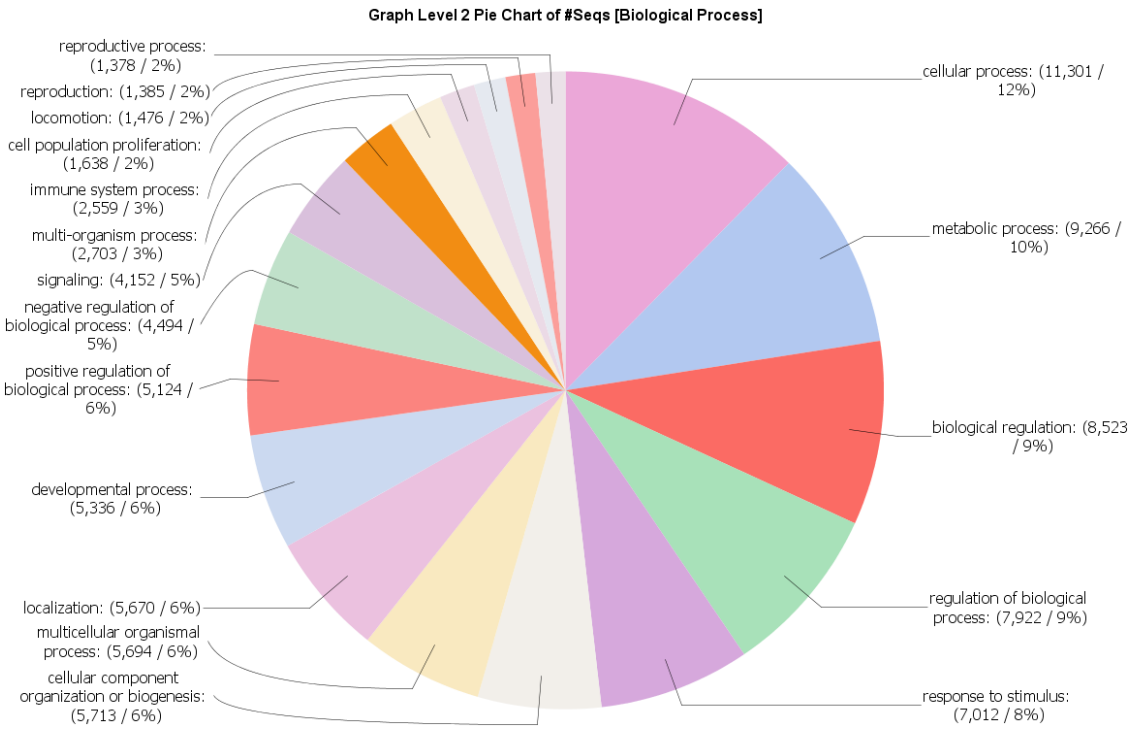

Figure 8: Pie chart of proportion of sequences annotated to Level 2 of the Biological Process category

Kallisto

Short reads have already been trimmed using BBDuk as per Chapter 4. These cleaned sequences were imported directly into Kallisto.

As TRINITY was run on the Flinders' HPC Deep Thought that is where trimming was completed, Kallisto however was run on the eRSA NECTAR cloud machine and files were moved accordingly, keeping directories relative locations intact.

The file reference_transcripts.1Lv.clean.fasta created in the subsetting section of Chapter 4's methods was used as the index.
All of this was achieved using the following kallisto-clstr-clean.sh script summarised below.

create the index the below contains the representative transcript for clustered isoforms with (most) of the poly-a tails trimmed:

```
kallisto index -i reference_transcripts.1Lv.clean.idx
↪  /mnt/IsoSeq-analysis/data/Seqtk/reference_transcripts.1Lv.clean.fasta
```

Output:

```
# [build] loading fasta file
↪  /mnt/IsoSeq-analysis/data/Seqtk/reference_transcripts.1Lv.clean.fasta
# [build] k-mer length: 31
# [build] counting k-mers ... done.
# [build] building target de Bruijn graph ...  done
# [build] creating equivalence classes ...  done
# [build] target de Bruijn graph has 59186 contigs and contains 17770125 k-mers
```

Run kallisto:

```
KallistoDIR=/mnt/DE-analysis/2_alignedData/kallisto-clstr-clean
ReadDIR=/mnt/DE-analysis/1_trimmedData #contains trimmed R1, R2, and singletons

# need to be in the input directory for for file
cd $ReadDIR

for file in *R1_clean.fq.gz
do
  FILESTEM=${file%_*}
  #this FILESTEM only cuts to _clean, the _R1 is included
  FILESTEM=${FILESTEM/R1/}
  #removes R1 from FILESTEM (FILESTEM ends in _ therefore not needed in "text"
  ↪  names)

echo $FILESTEM

kallisto quant -i $KallistoDIR/reference_transcripts.1Lv.clean.idx -o
↪  $KallistoDIR/$FILESTEM"kallisto-out" -b, --bootstrap-samples=100 --threads=8
↪  --pseudobam $file $FILESTEM"R2_clean.fq.gz" || error_exit "$LINENO: kallisto
↪  error at $FILESTEM"
```

The full kallisto terminal output is here kallisto-clstr-clean-log.txt.

EdgeR Expression Analysis

Preliminary data exploration for these data has been included twice for different groupings. The first analysis below includes all original eight samples.
Samples G6 and G7 have ultimately been removed due to batch effects caused by sequencing runs.

Eight *T. adelaidensis* individuals collected between two seasonal periods.

Six *T. adelaidensis* individuals collected between two seasonal periods. (These methods exclude samples G6 and G7).

This document separates samples by Seasonal group factors AND Sex Group Factors to explore the effects of sample G5 on the data.