**General Bioinformatics Procedures**

***Initial notes on instructions pertaining to the Command line interface:***

* As a matter of convention in the following text any command that you type into the computer is in Courier New.
* You do not need to type the '$' at the beginning of each line - that is there to indicate the command line prompt.
* Sometimes the text has to wrap to the next line because the command is too long. In some cases the “\” is used to explicitly break a long line without disrupting the ability to for the command line to correctly interpret it.
* Be careful of “-” and “–”, “” and "" . if copying directly from Word to the command line and make sure you have spaces between the different symbols unless otherwise indicated.
* Many of custom PERL scripts will require editing of file paths and/or additional hard coded variables.

**Step 1: Retrieve fastq reads**

Retrieve rawpaired genome reads (R1, R2) from sequencing center into a local directory.

wget or curl

**Step 2: Check sequence quality**

Make an output directory for the quality results

$ mkdir fastqc\_results

Then check the quality using FastQC:

$ fastqc -t 4 Hi49\_14MC\_f1\_7A\_S3\_R1\_001.fastq.gz Hi49\_14MC\_f1\_7A\_S3\_R2\_001.fastq.gz -o fastqc\_results

Check the reports (.html) files for read counts, overall quality and read lengths. Download them to your computer and use your favorite browser (e.g., Chrome, Safari).

**Step 3: Trim reads**

Remove adapter sequences and trim based read quality with Trimmomatic (v0.27)

$ trimmomatic PE -phred33 \

Hi49\_14MC\_f1\_7A\_S3\_R1\_001.fastq.gz \

Hi49\_14MC\_f1\_7A\_S3\_R2\_001.fastq.gz \

Hi49\_R1\_paired.fastq.gz \

Hi49\_R1\_unpaired.fastq.gz \

Hi49\_R2\_paired.fastq.gz \

Hi49\_R2\_unpaired.fastq.gz \

ILLUMINACLIP:TruSeq3\_PE.fa:2:30:10 LEADING:3 TRAILING:3 \

SLIDINGWINDOW:4:15 MINLEN:50

There will be 4 output files 1P & 2P contain paired reads where both reads survived trimming; 1U & 2U contain reads where the other read in the pair was lost.

Record read trimming results and remaining read counts in each file.

**Step 4: Check the effect of quality trimming on the read data**

Then check the quality using FastQC:

$ fastqc -t 4 Hi49\_R1\_paired.fastq.gz Hi49\_R2\_paired.fastq.gz \

Hi49\_R1\_unpaired.fastq.gz Hi49\_R2\_unpaired.fastq.gz \

-o fastqc\_results

Check the reports (.html) files for read counts, overall quality and read lengths. Download them to your computer and use your favorite browser (e.g., Chrome, Safari).

**Step 5: Assemble reads into contigs with MEGAHIT**

There are many excellent genome assembly tools available, largely using overlap-layout-consensus (OLC) or de Brujin graph approaches.

Assembly time will be uncertain – dependent on # of processors, memory available and # of reads.

$ megahit -t 256 -1 Hi49\_R1\_paired.fastq.gz \

-2 Hi49\_R2\_paired.fastq.gz -r Hi49\_R1\_unpaired.fastq.gz,\

Hi49\_R2\_unpaired.fastq.gz -o Hi49\_megahit\_v1

When complete, fasta files of contigs and scaffolds will appear in the output directory, along with other output files.

Generate a quick summary of the assembly using QUAST:

$ quast.py --threads 256 -o Hi49\_quast\_v1/ \ Hi49\_megahit\_v1/final.contigs.fa

Results will appear in the directory specified after -o . There are several formats available; try looking at report.txt or report.tsv.

**Step 6: Read mapping**

This approach to map reads can be used on entire or partial genomes, metagenomes… any genomic DNA region really. Most genome assembly algorithms produce just the contigs (contiguous stretches of DNA sequence) and/or scaffolds (contigs determined to be adjacent to one another, but lacking DNA sequences that physically overlap). Therefore, after assembling a meta/genome it is common to re-map the reads to the contigs in order to understand how the reads actually constitute the contigs. Read coverage can be indicative of copy number and possibly biology.

*Note: This step can be performed on the entire metagenome, or it can be done on contig subsets after* ***Step 9*** *is completed. SAM and BAM files are large and if space is a constraint, only analyzing contig subsets may be preferable.*

First generate an index file of your contigs/genome sequence to accelerate read mapping:

$ bowtie2-build --threads 20 final.contigs.fa final.contigs

Now map the paired reads to the genome sequence index:

$ bowtie2 --threads 20 --local -x final.contigs \

--sensitive-local \

-1 Hi49\_R1\_paired.fastq.gz \

-2 Hi49\_R2\_paired.fastq.gz \

-U Hi49\_R1\_unpaired.fastq.gz \

-U Hi49\_R2\_unpaired.fastq.gz \

-S final.contigs.sam --no-unal

Your main output will be in <outputfile.sam>; bowtie2 will also print some useful statistics to your terminal regarding how many reads mapped. Record these data as needed.

**Step 7: Prepare sorted BAM file for Visualization**

To visualize read mapping in tablet or artemis requires \*.bam files.

$ samtools view --threads 20 -b final.contigs.sam > final.contigs.bam

$ samtools sort --threads 20 -o final.contigs\_sorted.bam \

final.contigs\_v1.bam

$ samtools index -@ 20 final.contigs\_sorted.bam

$ gzip final.contigs.sam

**Step 8: Visualize short read alignments in Tablet**

Open Tablet (<https://ics.hutton.ac.uk/tablet/download-tablet/>).

Download your scaffolds or reference genome (\*fasta) and sorted .bam file and index file .bai to your laptop.

Examine patterns of read coverage among contigs (even, uneven within and between contigs).

Export read coverage data from Tablet for later use.

**Step 9: Identification of contigs of interest**

Use BlastN similarity to identify potential contigs of interest when related genomes are available.

$ blastn -out final.contigs.fa -query gracilis\_carsonella.fa -subject final.contigs.fa -evalue 1e-40 -max\_target\_seqs 15 -outfmt "6 std qlen slen"

Evaluate BlastN results in Spreadsheet application (Excel, Google Sheets).

Use available tools like EMBOSS, Snapgene, Artemis, etc. to calculate base composition.

Additional taxonomiv information can be determined by BlastN similarity to the NT database or the use of Centrifuge.

Combine results with exported read coverage data from Tablet if **Steps 6-8** are already performed.

Use above information to select contigs of interest.

Manually or with available fasta parsing tools make new file with contigs of interest.

Hi49\_carsonella.fna

If potentially a circular element or there is expectation of physical linkages between contigs use BlastN to compare contig ends to one another.

$ blastn -query Hi49\_carsonella.fna -subject Hi49\_carsonella.fna -evalue 1e-10 -outfmt "6 std"

If physical overlap detected, manually merge contigs then return to **Step 6-8** to re-map reads to newly defined contig merger. Reject or accept based on coverage results observed in Tablet.

If no overlap detected examine similarity of contig edges to related genomes, use alignment/BlastN to predict junction sequence. Then return to **Step 6-8** to re-map reads to newly defined contig merger. Reject or accept based on coverage results observed in Tablet.

If circularization/merger of contigs remains uncertain – leave as independent contigs/unclosed circular elements until additional sequence data is generated (e.g., long reads, Sanger Sequencing of PCR products).

**Step 10: Genome annotation**

Carry out structural and functional annotation of contigs of interest. Multiple tools are available. Prokka (v1.14.5) is one option.

For bacteria:

$ prokka --outdir Hi49\_prokka\_1 --locustag Hi49 --metagenome --cpus 8 --rfam Hi49\_carsonella.fna

For mitochondria:

prokka --kingdom Mitochondria --cpus 8 --genus Pariaconus \

--species dorsostriatus --strain 4A \

--outdir dorsostriatus\_mito --compliant --rfam \

--prefix dorsostriatus\_mito --locustag PDORS \

dorsostriatus\_mito.fna

View output GFF or GBK file in Artemis.

Manually inspect gene predictions.

*Note:* *Prokka often does a poor job of identifying pseudogenes. Depending on likely frequency of pseudogenes, manual inspection of long intergenic spacers and/or evaluating curiously short looking ORFs can be performed. For high frequency cases try PseudoFinder.*

*Note:* *Prokka performed poorly on mitochondrial gene annotations. Manual curation was required to conform to NCBI standards.*

Use Hidden Markov Models to predict similarity of protein query to a database of protein family domains (<http://pfam.xfam.org>):

$ hmmscan --acc --cut\_tc --cpu 1 Pfam-A.hmm Hi49\_carsonella.faa > Hi49\_carsonella.pfam

Pfam is for protein domains – *TIGRFAMs* is for whole gene models:

<https://www.jcvi.org/cgi-bin/tigrfams/index.cgi>

$ hmmscan --acc --cut\_tc --cpu 1 TIGRFAMs.HMM Hi49\_carsonella.faa > Hi49\_carsonella.tfam

See updated HMM database information from InterPro ( <https://ebi.ac.uk/interpro> )

The output can be converted into a BLAST-like report and a simple feature table using a custom script:

$ pfam2tsv.pl -p Hi49\_carsonella.pfam -o Hi49\_carsonella\_pfam.txt

Combine results in spreadsheet application (Excel, Google Sheets) with existing Prokka annotations and improve/confirm functional annotations.

**Step 11: OrthoVenn3 Analysis**

Use NCBI datasets to download predicted proteins for genomes of interest.

$ datasets download genome accession GCA\_044442985.1 --include gbff,genome,cds,protein

This will default save the information to ncbi\_dataset.zip, which can be decompressed:

$ unzip -o ncbi\_dataset.zip

Unpacks data entries to:

ncbi\_dataset/data/GCA\_044442985.1/

Upload the data for the protein.faa files from the genomes of interest to <https://orthovenn3.bioinfotoolkits.net/> (max = 12 genomes) and select the run option.

Once complete take note of cluster analysis and view the phylogeny in the “Phylogenetic Analysis” tab.

Then download the phylogeny and the alignment it is based on from the “Downloads” tab.

Also download the table of “Orthogroups” for subsequent analyses.

**Step 12: Whole genome alignment**

Use Mauve (v snapshot\_2015-02-25) to align the gbk files of your annotated genomes with the gbff files downloaded from NCBI.

Available here <https://darlinglab.org/mauve/mauve.html>

Follow GUI instructions to load genome files. Use default parameters.

Some genomes will align well using the preferred progressiveMauve algorithm. However, on occasion compositionally biased genomes require using the original mauveAligner algorithm.

For fragmented genomes (e.g., >1 contig/scaffold) consider using the “Move Contigs” function to better order/orient the fragments.

*Note: GBK/GBFF formatted files from some applications (e.g., Snapgene) may not be properly interpreted by Mauve. These files may require minor edits in order to work properly. Compare formats between one that is and isn’t working to identify possible issues – usually it is the header.*

*Note: Multisequence GBK/GBFF files are NOT recognized by Mauve. However, multisequence FASTA files are.*

**Step 13: Identification of insect host core genes**

Retrieve cDNA/mRNA sequences of BUSCO core genes of interest

triozidae\_genes.faa

Make a NT Blast database of the insect metagenome contigs:

$ makeblastdb -dbtype nucl -in final.contigs.fa -out Hi49 -title "Hi49 megahit assembly v1"

Search BUSCO core genes against the NT Blast database:

$ tblastn -num\_threads 20 -query triozidae\_genes.faa -db Hi49 -out Hi49\_triozidae.txt -evalue 1e-10 -max\_target\_seqs 15 -outfmt "6 std qlen slen"

Evaluate tBlastN results in Spreadsheet application (Excel, Google Sheets).

Sort and filter results for matches ≥50 percent of query length aligned on a single contig and ≥60% nucleotide identity.

Compare set of genomes being analyzed to identify list of BUSCO/contig targets to be examined further and generate tab separated lists:

busco1\tcontig1\n

busco2\tcontig15\n

Use available fasta parsing tools to run Exonerate (2.4.0) on each BUSCO to contig pair:

$ exonerate -c 8 --refine full --model protein2genome --showquerygff -r TRUE --percent 25 -q busco1.fa -t contig1.fa

Write(use) custom code to read the tab separated list from above wrapping the parsing, exonerate, compiling results into a single command:

$ ./wrap\_exon.pl hi49\_list.txt

Then use cufflinks (v2.2.1) gffread tool to extract DNA regions

$ gffread hi49\_exonerate.gff -g final.contigs.fa -x hi49\_cds.fa

Write(use) custom code assemble multi-exon CDS:

$ ./merge\_exons.pl hi49\_cds.fa hi49\_merge.fa

The CDS can then be used as needed or translated using EMBOSS tools (e.g., transeq).

**Step 14.** **General steps for running dN/dS analyses with PAML**

1. Generate multi-sequence fasta files (\*fna).

*Note:* *Mesquite and PAML do not like sequence names that are exclusively numbers.*

*Note: if running a large number of files be sure to submit a Batch job or use an interactive job*

$ build\_fastas.pl cds\_list.txt genomeA.ffn genomeB.ffn genomeC.ffn

cds\_list.txt should be a tab separated file with locus IDs that match genes in \*ffn files. Column order needs to match the order the files are given as command line arguments. No header expected/allowed. Resulting alignment files will use the first locus number as the filename prefix. Program can accept any number of columns/files.

cds\_list.txt

A1 B1 C1

A2 B2 C2

A3 B3 C3

…

*Note: The OrthoVenn3 output file for* “Orthogroups” *can be used to generate the* cds\_list.txt *file.*

Input fasta files should be the nucleotide sequences of protein coding genes:

genomeA.ffn

>A1

ATGAAA…

>A2

ATGGAT…

Output:

A1.fna

A2.fna

A3.fna

…

1. Align files using command line program that aligns by codons (optional algorithms include: clustalw, mafft, muscle) – for one file:

$ clustalw2nexus.pl -c Y -a mafft -n A1.fna

This will produce an aligned fasta file A1.aln and a nexus file A1.nex

For multiple files:

$ Wrapper.pl -p "clustalw2nexus.pl -c Y -a mafft -n" -f "\*fna"

clustalw2nexus.pl -c Y -a mafft -n A1.fna

clustalw2nexus.pl -c Y -a mafft -n A2.fna

...

1. **<optional step>** Manually edit files– necessary if dealing with pseudogenes , or other problematic sequences **<optional step>**
   1. Script above generates a Nexus files (\*nex)
   2. Open, edit and Save in Mesquite or similar alignment editor
   3. Convert back to a Fasta file (\*fa, \*ffn, \*faa, \*aln)

$ Wrapper.pl -p "nex2fa.pl -g N -w N -n" -f "\*nex"

1. Trim alignments of gap containing columns & stop codons. Script works on all files in the local directory if a wild card is used. Script overwrites input files. You will have to regenerate alignments if you want the gaps and stop codons back.

$ trim\_align.pl codon \*aln

1. Convert trimmed Fasta alignment file to Phylip file (\*phy). *Note: 18 refers to the number of character spaces allowed for the sequence names. If your sequence names are longer than 17 characters, increase the number accordingly. Watch out - Some programs use the original Phylip format specification of only 11 characters. PAML isn’t one of them though.*

$ Wrapper.pl -p "fa2phylip.pl -l 18 -f" -f "\*fa"

1. Run PAML’s codeML program (pairwise)

$ RUNcodeml\_p.pl -p "\*phy"

1. **<optional step>** Run codeML (with tree – more complicated) **<optional step>**
   1. Generate trees… (generate custom script to suit purpose)
   2. Then run codeML

$ RUNcodeml.pl -p "\*phy" -o OUTFILE\_PREFIX

*Note: See Álvarez-Carretero et al. 2023 for details on testing for site and branch specific positive selection.*

1. Summarize output (generate custom script to suit purpose). An example is below. Generally programs have to be re-written to ID correct locus ID name patterns.

$ ps4.pl prefixes.txt \*txt >codeml\_results.txt

**Step 15. Absolute rates of change**

Use a best estimated phylogeny that includes nodes of known age to estimate remaining node ages with r8s (v1.81). Use downloaded tree from OrthoVenn3 (**Step 11**).

Generate NEXUS file r8s\_ctl\_1.txt with parameters for r8s:

#NEXUS

begin trees;

tree nj\_tree = [&R] (rooted tree w/ branch lengths);

End;

begin rates;

blformat nsites=XXXXX lengths=persite ultrametric=no;

collapse;

mrca ROOT Genome1 Genome10;

fixage taxon= ROOT age=YYY;

divtime method=pl algorithm=tn cvStart=0 c vInc=0.5 cvNum=8 crossv=yes;

describe plot=chronogram;

describe plot=tree\_description;

end;

User modified values:

(rooted tree w/ branch lengths)

XXXXX = number of sites in alignment used to generate tree

ROOT = adding name and defining node that represents the root/or node with date estimate. Node defined by Genome1 and Genome10 that span share that node.

YYY = point age estimate

Run r8s:

$ r8s -b -f r8s\_ctl\_1.txt > r8s\_ctl\_1.txt

Manually view and extract ages.

Modify nexus control file to change age to upper or lower bound estimate YYY as needed. Re-run r8s.

Date estimates can then be combined with Summarized PAML output results in spreadsheet application (Excel, Google Sheets). Then exported to stats program as needed (e.g., JMP, Prism)

**Step 16. Additional Phylogenetic Estimation methods**

OrthoVenn3 provides a concatenated protein sequence alignment for all single copy conserved core genes in a multisequence fasta file format. Use online or custom code tool to convert the fasta file to a phylip file.

Use ModetTest-ng (v0.1.7) to determine initial starting model of molecular evolution that is appropriate for the phylip alignment.

$ modeltest-ng --threads 20 -t ml -d aa -i Multiple\_sequence\_alignment.phy

Different selection criteria available – BIC, AIC, AICc. View screen output for results. Choose criterion to follow, and select corresponding model.

Start RAxML tree search using selected model.

$ raxml-ng --msa Multiple\_sequence\_alignment.phy --threads 20 \

--model JTT+I+G4+F --all --tree pars{10} --bs-trees 200

The best tree with the bootstrap values for the assigned nodes can be found in the output file ending with the extension \*raxml.support . Open tree in tree viewer software to visualize (e.g. Figtree).