#Phytoplasma multi-gene hyb panel

#new algorithm for mapping and assembly 220122

#step 1: trimmomatic/flash2

#step 2: mapping vs protein coding genes (all types)

#step 3: assemble protein coding genes and determine phytoplasma type - eg cpnclassiphyr

#step 4: mapping vs 16S genes (type-specific) plus mitochondrial/chloroplast and nonspecific bacterial 16S sequences to remove junk

#step 5: assemble 16S genes

#step 6: trim all assemblies to retain only contigs >500 bp

#final product: set of fasta files containing long assemblies for each of 6 protein-coding genes/loci plus 16S

#make conda environment

conda create --name hyb

conda activate hyb

conda install -c bioconda trimmomatic

conda install -c bioconda flash2

#Get read names

mkdir -p /path/to/working/directory/trim

mkdir -p /path/to/working/directory/merge

find /path/to/folder/with/illumina/reads/\*.fastq.gz \

-maxdepth 1 \

| sed 's/\/.\*\///' \

| sed 's/\_R.\*$//' \

| sed 's/Undetermined.\*$//' \

| sort | uniq \

> /path/to/working/directory/run-id\_fastq\_files.txt

#step 1A: trimmomatic

#make trimmomatic shell script

nano /path/to/working/directory/trimmomatic.sh

#copy/paste below

#!/bin/bash

#$ -S /bin/bash

#$ -N name\_trim

#$ -j y

#$ -cwd

#$ -pe smp 8

source /path/to/home/directory/miniconda3/etc/profile.d/conda.sh

conda activate hyb

for i in `cat /path/to/working/directory/run-id\_fastq\_files.txt`; do \

trimmomatic PE \

/path/to/folder/with/illumina/reads/$i'\_R1\_001.fastq.gz' \

/path/to/folder/with/illumina/reads/$i'\_R2\_001.fastq.gz'\

/path/to/working/directory/trim/$i.paired.fq \

/path/to/working/directory/trim/$i.unpaired.fq \

/path/to/working/directory/trim/$i.paired.reverse.fq \

/path/to/working/directory/trim/$i.unpaired.reverse.fq \

LEADING:3 TRAILING:20 SLIDINGWINDOW:4:15 MINLEN:36 \

; done

#step 1B: flash2 to merge R1, R2

#make merge shell script

nano /path/to/working/directory/merge.sh

#copy/paste below

#!/bin/bash

#$ -S /bin/bash

#$ -N name\_merge

#$ -j y

#$ -cwd

#$ -pe smp 4

source /path/to/home/directory//miniconda3/etc/profile.d/conda.sh

conda activate hyb

for i in `cat /path/to/working/directory/run-id\_fastq\_files.txt` ; do \

flash2 \

-f 550 \

-s 200 \

-r 300 \

/path/to/working/directory/trim/$i'.paired.fq' \

/path/to/working/directory/trim/$i'.paired.reverse.fq' \

-d -d /path/to/working/directory/merge/ \

-o $i.merged.fq \

; done

#step 3: mapping vs protein-coding genes

#Install bowtie2

conda install -c conda-forge tbb

conda install -c bioconda bowtie2

#make bowtie reference files

#Ensure you are using a compute node (qlogin)

#build one ref file for all protein-coding seqs (cpn60; nusA, secY, secA, rp, tuf - call this file ref\_protein\_seqs.fasta)

#then, build separate ref files containing type-specific 16S genes with junk; eg typeI\_16S\_junk.fasta

bowtie2-build \

-f /path/to/home/directory/ref\_seqs/ref\_protein\_seqs.fasta \

/path/to/home/directory/ref\_seqs/ref\_protein\_seqs.fasta

bowtie2-build \

-f /path/to/home/directory/ref\_seqs/typeI\_16S\_junk.fasta \

/path/to/home/directory/ref\_seqs/typeI\_16S\_junk.fasta

bowtie2-build \

-f /path/to/home/directory/ref\_seqs/typeIII\_16S\_junk.fasta \

/path/to/home/directory/ref\_seqs/typeIII\_16S\_junk.fasta

bowtie2-build \

-f /path/to/home/directory/ref\_seqs/typeX\_16S\_junk.fasta \

/path/to/home/directory/ref\_seqs/typeX\_16S\_junk.fasta

bowtie2-build \

-f /path/to/home/directory/ref\_seqs/typeXII\_16S\_junk.fasta \

/path/to/home/directory/ref\_seqs/typeXII\_16S\_junk.fasta

#make bowtie (mapping) shell script

nano /path/to/working/directory/map\_ref\_genes.sh

#copy/paste below

#!/bin/bash

#$ -S /bin/bash

#$ -N name\_map

#$ -j y

#$ -cwd

#$ -pe smp 8

source /path/to/home/directory/miniconda3/etc/profile.d/conda.sh

conda activate hyb

#have to do this twice; once for the protein-coding genes using all the samples; once for the type-specific 16S genes plus junk

#for the first round, can do all of the samples at the same time - maps protein-coding genes only for all types

#for the second round, map only those samples containing the relevant types

for i in `cat /path/to/working/directory/run-id\_fastq\_files.txt` ; do \

bowtie2 --local \

-p 1 \

-x /path/to/home/directory/ref\_seqs/typeI\_protein\_seqs.fasta \

-U /path/to/working/directory/merge/$i'.merged.fq.extendedFrags.fastq' \

-S /path/to/working/directory/$i.ref\_genes.sam \

; done

#map vs type-specific 16S genes plus junk. First change filenames list to only those samples containing the relevant type

for i in `cat /path/to/working/directory/run-id\_fastq\_files.txt` ; do \

bowtie2 --local \

-p 1 \

-x /path/to/home/directory/ref\_seqs/typeI\_16S\_junk.fasta \

-U /path/to/working/directory/merge/$i'.merged.fq.extendedFrags.fastq' \

-S /path/to/working/directory/$i.ref\_genes.sam \

; done

#Extract mapped protein-coding or 16S reads for all samples, depending on what has been mapped

#Ensure you are using a compute node (qlogin)

for i in `cat /path/to/working/directory/run-id\_fastq\_files.txt` ; do \

samtools view -bS \

/path/to/working/directory/$i'.ref\_genes.sam' \

> /path/to/working/directory/$i.ref\_genes.bam \

; done

for i in `cat /path/to/working/directory/run-id\_fastq\_files.txt` ; do \

samtools sort \

/path/to/working/directory/$i'.ref\_genes.sam' \

> /path/to/working/directory/$i.ref\_genes.bam.sorted \

; done

for i in `cat /path/to/working/directory/run-id\_fastq\_files.txt` ; do \

samtools index \

/path/to/working/directory/$i'.ref\_genes.bam.sorted' \

; done

#Extract mapped reads and corresponding fastq for each protein-coding gene

#can submit as a job to do all at once

nano /path/to/working/directory/mapped\_fastq/mapped\_fastq.sh

#copy/paste below

#!/bin/bash

#$ -S /bin/bash

#$ -N name\_map\_typeI

#$ -j y

#$ -cwd

#$ -pe smp 1

source /path/to/home/directory/miniconda3/etc/profile.d/conda.sh

conda activate hyb

#make sure gene names below (eg "secY\_AY-I" PERFECTLY matches the corresponding gene name in ref\_genes.fasta (after, ">") - or you will get this error: region "secY" specifies an unknown reference name. Continue anyway. And no reads will match. So, need to extract reads vs 1 group at a time - eg secY\_AY-I

#secY

for i in `cat /path/to/working/directory/run-id\_fastq\_files.txt` ; do \

samtools view -bh \

/path/to/working/directory/$i'.ref\_genes.bam.sorted' \

secY\_AY-I \

> /path/to/working/directory/$i.mapped.secY.bam \

; done

for i in `cat //path/to/working/directory/run-id\_fastq\_files.txt` ; do \

samtools fastq \

-0 /path/to/working/directory/mapped\_fastq/$i.mapped.secY.fastq \

/path/to/working/directory/$i'.mapped.secY.bam' \

; done

#cpn60 - skip for type III (no cpn60)

for i in `cat /path/to/working/directory/run-id\_fastq\_files.txt` ; do \

samtools view -bh \

/path/to/working/directory/$i'.ref\_genes.bam.sorted' \

cpn60\_AY-I \

> /path/to/working/directory/$i.mapped.cpn60.bam \

; done

for i in `cat /path/to/working/directory/run-id\_fastq\_files.txt` ; do \

samtools fastq \

-0 /path/to/working/directory/mapped\_fastq/$i.mapped.cpn60.fastq \

/path/to/working/directory/$i'.mapped.cpn60.bam' \

; done

#tuf

for i in `cat /path/to/working/directory/run-id\_fastq\_files.txt` ; do \

samtools view -bh \

/path/to/working/directory/$i'.ref\_genes.bam.sorted' \

tuf\_AY-I \

> /path/to/working/directory/$i.mapped.tuf.bam \

; done

for i in `cat /path/to/working/directory/run-id\_fastq\_files.txt` ; do \

samtools fastq \

-0 /path/to/working/directory/mapped\_fastq/$i.mapped.tuf.fastq \

/path/to/working/directory/$i'.mapped.tuf.bam' \

; done

#rp

for i in `cat /path/to/working/directory/run-id\_fastq\_files.txt` ; do \

samtools view -bh \

/path/to/working/directory/$i'.ref\_genes.bam.sorted' \

rp\_AY-I \

> /path/to/working/directory/$i.mapped.rp.bam \

; done

for i in `cat /path/to/working/directory/run-id\_fastq\_files.txt` ; do \

samtools fastq \

-0 /path/to/working/directory/mapped\_fastq/$i.mapped.rp.fastq \

/path/to/working/directory/$i'.mapped.rp.bam' \

; done

#nusA

for i in `cat /path/to/working/directory/run-id\_fastq\_files.txt` ; do \

samtools view -bh \

/path/to/working/directory/$i'.ref\_genes.bam.sorted' \

nusA\_AY-I \

> /path/to/working/directory/$i.mapped.nusA.bam \

; done

for i in `cat /path/to/working/directory/run-id\_fastq\_files.txt` ; do \

samtools fastq \

-0 /path/to/working/directory/mapped\_fastq/$i.mapped.nusA.fastq \

/path/to/working/directory/$i'.mapped.nusA.bam' \

; done

#secA

for i in `cat /path/to/working/directory/run-id\_fastq\_files.txt` ; do \

samtools view -bh \

/path/to/working/directory/$i'.ref\_genes.bam.sorted' \

secA\_AY-I \

> /path/to/working/directory/$i.mapped.secA.bam \

; done

for i in `cat /path/to/working/directory/run-id\_fastq\_files.txt` ; do \

samtools fastq \

-0 /path/to/working/directory/mapped\_fastq/$i.mapped.secA.fastq \

/path/to/working/directory/$i'.mapped.secA.bam' \

; done

#16S genes; after doing mapping vs type-specific ref seqs plus junk

for i in `cat /path/to/working/directory/run-id\_fastq\_files.txt` ; do \

samtools view -bh \

/path/to/working/directory/$i'.ref\_genes.bam.sorted' \

16S\_AY-I \

> /path/to/working/directory/$i.mapped.16S.bam \

; done

for i in `cat /path/to/working/directory/run-id\_fastq\_files.txt` ; do \

samtools fastq \

-0 /path/to/working/directory/mapped\_fastq/$i.mapped.16S\_AY-I.fastq \

/path/to/working/directory/$i'.mapped.16S.bam' \

; done

##################################

#How many reads mapped to each gene?

grep -c M01666 /path/to/working/directory/all-mapped\_fastq/\*.16S\_AY-I.fastq

#Assembly

mkdir ~/software/

cd ~/software

wget https://www.bcgsc.ca/platform/bioinfo/software/trans-abyss/releases/2.0.1/transabyss-2.0.1.zip

unzip transabyss-2.0.1.zip

#modify .bashrc file

cd ~

nano .bashrc

#add the following:

#added by townj

export PATH="/home/AAFC-AAC/dumonceauxt/software/transabyss-2.0.1/:$PATH"

#log off of server, then log back on

#Add igraph to conda environment

conda activate hyb

conda install -c conda-forge python-igraph

#Assemble

#make sure you are using qlogin session

mkdir -p /path/to/working/directory/assemblies

#cpn60

for i in `cat /path/to/working/directory/run-id\_fastq\_files.txt` ; do \

transabyss --se \

/path/to/working/directory/mapped\_fastq/$i'.mapped.cpn60.fastq' \

-k 32 \

--outdir /path/to/working/directory/assemblies/$i'\_cpn60' \

; done

#secA

for i in `cat /path/to/working/directory/run-id\_fastq\_files.txt` ; do \

transabyss --se \

/path/to/working/directory/mapped\_fastq/$i'.mapped.secA.fastq' \

-k 32 \

--outdir /path/to/working/directory/assemblies/$i'\_secA' \

; done

#secY

for i in `cat /path/to/working/directory/run-id\_fastq\_files.txt` ; do \

transabyss --se \

/path/to/working/directory/mapped\_fastq/$i'.mapped.secY.fastq' \

-k 32 \

--outdir /path/to/working/directory/assemblies/$i'\_secY' \

; done

#tuf

for i in `cat /path/to/working/directory/run-id\_fastq\_files.txt` ; do \

transabyss --se \

/path/to/working/directory/mapped\_fastq/$i'.mapped.tuf.fastq' \

-k 32 \

--outdir /path/to/working/directory/assemblies/$i'\_tuf' \

; done

#nusA

for i in `cat /path/to/working/directory/run-id\_fastq\_files.txt` ; do \

transabyss --se \

/path/to/working/directory/mapped\_fastq/$i'.mapped.nusA.fastq' \

-k 32 \

--outdir /path/to/working/directory/assemblies/$i'\_nusA' \

; done

#rp

for i in `cat /path/to/working/directory/run-id\_fastq\_files.txt` ; do \

transabyss --se \

/path/to/working/directory/mapped\_fastq/$i'.mapped.rp.fastq' \

-k 32 \

--outdir /path/to/working/directory/assemblies/$i'\_rp' \

; done

#16S

for i in `cat /path/to/working/directory/run-id\_fastq\_files.txt` ; do \

transabyss --se \

/path/to/working/directory/mapped\_fastq/$i'.mapped.16S\_AY-I.fastq' \

-k 32 \

--outdir /path/to/working/directory/assemblies/$i'\_16S'\

; done

#step6 - trim assemblies to retain only those >500 bp (code added by Kevin Muirhead)

#if miniconda not there, go get it

wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86\_64.sh

sh Miniconda3-latest-Linux-x86\_64.sh

#... follow miniconda3 installation directions... Mostly pressing enter. Installs in home by default

conda create --name bioawk\_env

conda activate bioawk\_env

conda install -c bioconda bioawk

#this command will now trim transabyss-final.fa to only those with a length >=500 bp and output it to test-seq-l500.fa. Can change names or length as needed.

bioawk -c fastx 'length($seq) >= 500{ print ">"$name" "$comment; print $seq }' transabyss-final.fa > test-seq-l500.fa