MG_HW7: Taxonomic Classification Using Centrifuge Version 5

James Thornton

Abstract

This protocol provides a procedure to generate taxonomic data from assembled contigs using centrifuge.

Citation: James Thornton MG_HW7: Taxonomic Classification Using Centrifuge. **protocols.io** dx.doi.org/10.17504/protocols.io.f8ibrue

Published: 28 Oct 2016

Guidelines

Centrifuge documentation

Protocol

Step 1.

Log in to the HPC cluster (ICE)

cmd COMMAND

```
$ ssh hpc
$ ice
```

NOTES

James Thornton Jr 28 Oct 2016

Option 3 for those with menu enabled.

Step 2.

Move into your class directory.

cmd COMMAND

```
$ cd /rsgrps/bh_class/username
```

Use YOUR username

Step 3.

Create a directory called "original" and move all of your current directories into this directory. We will use this directory to store your original work.

cmd COMMAND

```
mkdir original
mv * original
```

■ ANNOTATIONS

Bonnie Hurwitz 28 Oct 2016

Ignore the warning that: "

mv: cannot move `original' to a subdirectory of itself, `original/original'"

Step 4.

Create directories for running the steps in this protocol. We will download the paired reads from the SRA, run quality control, pair up the reads follow QC, run bowtie2 to remove human contamination, and centrifuge for taxonomic analysis of the individual reads.

cmd COMMAND

```
mkdir fastq
mkdir unmapped
mkdir taxonomy
```

Step 5.

Go into the fastq direectory

cmd COMMAND

```
cd fastq
```

Step 6.

Create a file called "list" with all of your SRR file names.

cmd COMMAND

```
nano list
```

■ ANNOTATIONS

Bonnie Hurwitz 28 Oct 2016

My file has the following:

% cat list

SRR1647045

SRR1647046

SRR1647047

SRR1647048

SRR1647049

SRR1647141

SRR1647142

SRR1647143

Step 7.

Now we will download all of the SRR files again from the SRA, but this time we will use a few new tricks to download the files as separate paired end read files (_1 and _2 files).

Create the following script called get-fastq.sh:

cmd COMMAND

```
#!/bin/bash

#PBS -W group_list=bh_class
#PBS -q windfall
#PBS -l jobtype=cluster_only
#PBS -l select=1:ncpus=2:mem=4gb
#PBS -l walltime=24:00:00
#PBS -l cput=24:00:00
#PBS -l place=pack:shared
#PBS -M netid@email.arizona.edu
#PBS -m bea

module load sratools
echo “my job_id is: ${PBS_JOBID}”

FASTQ_DIR="/rsgrps/bh_class/username/fastq"
cd $FASTQ_DIR

for file in `cat list`; do
   fastq-dump --outdir $FASTQ_DIR --gzip --skip-technical --readids --dumpbase --split-
files --clip $file;
done
```

ANNOTATIONS

Bonnie Hurwitz 28 Oct 2016

--clip

removes adapter sequences

Bonnie Hurwitz 28 Oct 2016

--skip-technical

Only output the biological, not technical reads

Bonnie Hurwitz 28 Oct 2016

--readids

Gives a unique name for each read (for forward and reverse read ids), so they don't end up with the same read id in the R1 and R2 file and break downstream processes.

Bonnie Hurwitz 28 Oct 2016

--gzip

compresses your files with gzip

Bonnie Hurwitz 28 Oct 2016

--dumpbase ensures the bases are A, T, G, C not in color space (as in SOLiD sequencing)

Bonnie Hurwitz 28 Oct 2016

--split-files

outputs paired ends as _1 and _2 files for the forward and reverse read.

Step 8.

Run the script on the cluster to get all of the fastq paired-end files

cmd COMMAND

```
mkdir std-err std-out
chmod 755 get-fastq.sh
qsub -e std-err/ -o std-out/ get-fastq.sh
```

Step 9.

You can look at the quality of the read files using fastqc as you did before.

But, because we already have an idea of the issues, we are going to run a streamlined QC process in the next step.

cmd COMMAND

```
module load fastqc/0.11.2
fastqc ./*.fastq.gz
```

Step 10.

To speed things up, we are going to apply a general set of quality control parameters. Create a script called run-qualityctrl.sh.

cmd COMMAND

```
#!/bin/bash

#PBS -W group_list=bh_class
#PBS -q windfall
#PBS -l jobtype=cluster_only
#PBS -l select=1:ncpus=2:mem=4gb
#PBS -l walltime=24:00:00
#PBS -l cput=24:00:00
#PBS -l place=pack:shared
#PBS -M netid@email.arizona.edu
#PBS -m bea

module load fastx/0.0.14

echo “my job_id is: ${PBS_JOBID}”
```

```
FASTQ_DIR="/rsgrps/bh_class/userid/fastq"
export $FASTQ_DIR
cd $FASTQ_DIR

for file in `cat list`; do
   R1=$file"_1.fastq.gz"
   R2=$file"_2.fastq.gz"
   R1OUT=$file"_1.fastq"
   R2OUT=$file"_2.fastq"
   gzip -dc $R1 | fastx_trimmer -f 12 | fastq_quality_filter -q 20 -p 80 | fastx_clipper -
l 50 > $R1OUT
   gzip -dc $R2 | fastx_trimmer -f 12 | fastq_quality_filter -q 20 -p 80 | fastx_clipper -
l 50 > $R2OUT
done
```

ANNOTATIONS

Bonnie Hurwitz 28 Oct 2016

make sure you don't split the lines in the script below, or you will get an error.

Step 11.

Run the quality control script on the cluster.

cmd COMMAND

```
chmod 755 run-qualityctrl.sh
qsub -e std-err/ -o std-err/ run-qualityctrl.sh
```

Step 12.

After quality control, some of the paired-end sequences are lost and they become singletons. We need to create new files, where the pairs are in R1 (forward) and R2 (reverse), and singletons are in a separate file. We will create a perl script to do this. Please create a script called get-paired.pl and paste in the script from below.

cmd COMMAND

```
#! /uaopt/perl/5.14.2/bin/perl
use strict;
use Bio::SeqIO;

my $file1 = shift @ARGV;
my $file2 = shift @ARGV;
my $out = shift @ARGV;

my $seq_in1 = Bio::SeqIO->new( -format => 'fastq',
                               -file   => $file1,
                             );
my $seq_in2 = Bio::SeqIO->new( -format => 'fastq',
                               -file   => $file2,
                             );
my $seq_out1 = Bio::SeqIO->new( -format => 'fastq',
                                -file   => ">$out.R1.fastq"
                             );
my $seq_out2 = Bio::SeqIO->new( -format => 'fastq',
                                -file   => ">$out.R2.fastq"
                             );
my $seq_out3 = Bio::SeqIO->new( -format => 'fastq',
                                -file   => ">$out.singletons.fastq"
                             );

my %read_to_count;
while ( my $seq1 = $seq_in1->next_seq() ) {
   my $id = $seq1->id();
   my ($srr, $ct, $end) = split(/\./, $id);
   my $read = $srr . "." . $ct;
   $read_to_count{$read}++;
}

while ( my $seq2 = $seq_in2->next_seq() ) {
   my $id = $seq2->id();
   my ($srr, $ct, $end) = split(/\./, $id);
   my $read = $srr . "." . $ct;
   $read_to_count{$read}++;
}

my $seq_in3 = Bio::SeqIO->new( -format => 'fastq',
                               -file   => $file1,
                             );
my $seq_in4 = Bio::SeqIO->new( -format => 'fastq',
                               -file   => $file2,
                             );

while ( my $seq1 = $seq_in3->next_seq() ) {
   my $id = $seq1->id();
   my ($srr, $ct, $end) = split(/\./, $id);
   my $read = $srr . "." . $ct;
   if ($read_to_count{$read} == 2) {
      $seq_out1->write_seq($seq1);
   }
   else {
      $seq_out3->write_seq($seq1);
   }
}

while ( my $seq2 = $seq_in4->next_seq() ) {
   my $id = $seq2->id();
   my ($srr, $ct, $end) = split(/\./, $id);
   my $read = $srr . "." . $ct;
   if ($read_to_count{$read} == 2) {
      $seq_out2->write_seq($seq2);
   }
   else {
      $seq_out3->write_seq($seq2);
   }
}
```

ANNOTATIONS

Bonnie Hurwitz 28 Oct 2016

note that I am defining an explicit path here to the version of perl that has BioPerl installed. You can also "module load perl" and use the perl from your environment

Step 13.

Now we need to create a bash script to run the paired end script on the cluster.

create a script called:

run-pairedends.sh

cmd COMMAND

```
#!/bin/bash

#PBS -W group_list=bh_class
#PBS -q windfall
#PBS -l jobtype=cluster_only
#PBS -l select=1:ncpus=2:mem=4gb
#PBS -l walltime=24:00:00
#PBS -l cput=24:00:00
#PBS -l place=pack:shared
#PBS -M netid@email.arizona.edu
#PBS -m bea

module load perl

echo “my job_id is: ${PBS_JOBID}”

FASTQ_DIR="/rsgrps/bh_class/uername/fastq"
export $FASTQ_DIR
cd $FASTQ_DIR

for file in `cat list`; do
   R1=$file"_1.trim.fastq"
   R2=$file"_2.trim.fastq"
   OUT=$file
   ./get-paired.pl $R1 $R2 $OUT
done
```

ANNOTATIONS

Bonnie Hurwitz 28 Oct 2016

Be sure to replace the "netid" and "username" in this script with your own.

Step 14.

Run the paired ends script on the cluster.

cmd COMMAND

```
chmod 755 get-paired.pl
chmod 755 run-pairedends.sh
qsub -e std-err/ std-out/ run-pairedends.sh
```

ANNOTATIONS

James Thornton Jr 29 Oct 2016

qsub -e std-err/ -o std-out/ run-pairedends.sh

Step 15.

Now we are ready to run the comparison against the human genome to remove contaminants, and then run the remaining sequences through centrifuge to look at the taxonomy.

change into the taxonomy directory.

cmd COMMAND

```
cd ../taxonomy
```

Step 16.

Copy the following into a new script named centrifuge_tax.sh:

cmd COMMAND

```
#!/bin/bash

#PBS -W group_list=bh_class
#PBS -q windfall
#PBS -l jobtype=cluster_only
#PBS -l select=1:ncpus=12:mem=23gb
#PBS -l pvmem=22gb
#PBS -l walltime=24:00:00
#PBS -l cput=24:00:00
#PBS -M netid@email.arizona.edu
#PBS -m bea

#--------------EDIT THESE---------------
FASTQ_DIR="/rsgrps/bh_class/username/fastq"
OUT_DIR="/rsgrps/bh_class/username/taxonomy"
BT2_OUT_DIR="/rsgrps/bh_class/username/unmapped"
#---------------------------------------

CENT_DB="/rsgrps/bh_class/b_compressed+h+v/b_compressed+h+v"
BT2_INDEX="/rsgrps/bh_class/bowtie2_index/human_index"

module load bowtie2/2.2.5

cd $FASTQ_DIR
for file in `cat list`; do
   # unfiltered
   R1=$FASTQ_DIR/$file".R1.fastq"
   R2=$FASTQ_DIR/$file".R2.fastq"
   S=$FASTQ_DIR/$file".singletons.fastq"

   # no human
   NH_R1=$BT2_OUT_DIR/$file".paired.1.fastq"
   NH_R2=$BT2_OUT_DIR/$file".paired.2.fastq"
   NH_S=$BT2_OUT_DIR/$file".singletons.fastq"

   bowtie2 -x $BT2_INDEX -1 $R1 -2 $R2 -U $S -q --very-sensitive-local -p 12 --un $BT2_OUT_DIR/$file.singletons.fastq --un-conc $BT2_OUT_DIR/$file.paired.fastq
   centrifuge -x $CENT_DB -1 $NH_R1 -2 $NH_R2 -U $NH_S -S $OUT_DIR/$file-classout --report-file $OUT_DIR/$file-centrifuge_report.tsv -q
done
```

NOTES

James Thornton Jr 28 Oct 2016

Important: For this to work you Fasta files must end with the extension .fasta

Step 17.

Submit centrifuge_tax.sh using qsub:

cmd COMMAND

```
mkdir std-err std-out
qsub -o std-out/ -e std-err/ centrifuge_tax.sh
```

Step 18.

Once the job is running it will loop through all of your Fastq files, remove human reads from the Fastq files, then run Centrifuge on unmapped files to generate taxonomic data. This will take about 1 hour to generate reports for all of your fastq files. You can use qstat to check the status of your job.

cmd COMMAND

```
$ qstat -u username
```

Use YOUR username Under S (Status) 'Q' means queued, 'R' means running

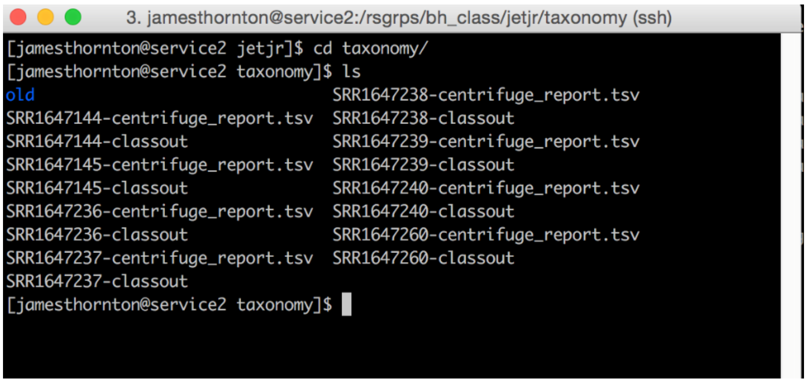
Step 19.

Once the job is complete move into your taxonomy directory and ensure all output files are there. If the job was successful there should be a total of 6 "classout" files and 6 "centrifuge_report.tsv" files.

cmd COMMAND

```
$ cd taxonomy
$ ls
```

EXPECTED RESULTS



3. jamesthornton@service2:/rsgrps/bh_class/jetjr/taxonomy (ssh)

```
[jamesthornton@service2 jetjr]$ cd taxonomy/
[jamesthornton@service2 taxonomy]$ ls
old                                SRR1647238-centrifuge_report.tsv
SRR1647144-centrifuge_report.tsv  SRR1647238-classout
SRR1647144-classout                SRR1647239-centrifuge_report.tsv
SRR1647145-centrifuge_report.tsv  SRR1647239-classout
SRR1647145-classout                SRR1647240-centrifuge_report.tsv
SRR1647236-centrifuge_report.tsv  SRR1647240-classout
SRR1647236-classout                SRR1647260-centrifuge_report.tsv
SRR1647237-centrifuge_report.tsv  SRR1647260-classout
SRR1647237-classout
[jamesthornton@service2 taxonomy]$
```

Step 20.

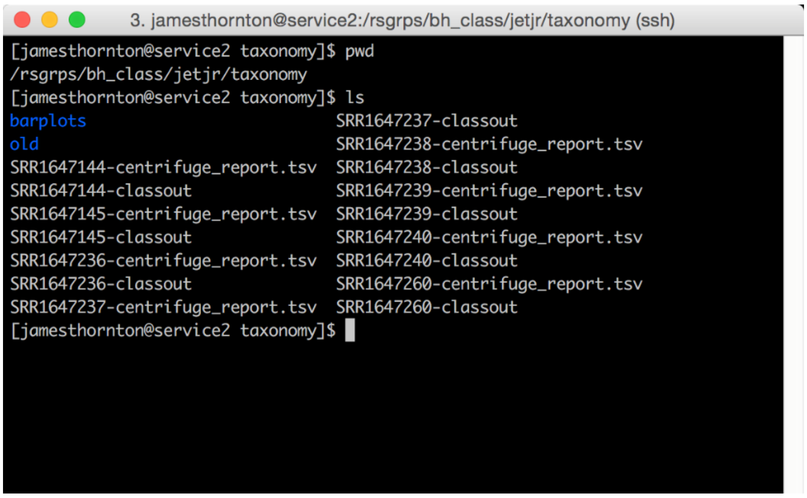
In your taxonomy directory make a new directory called barplots

cmd COMMAND

```
$ mkdir barplots
```

Make sure you are in /rsgrps/bh_class/username/taxonomy for this to work correctly

EXPECTED RESULTS



3. jamesthornton@service2:/rsgrps/bh_class/jetjr/taxonomy (ssh)

```
[jamesthornton@service2 taxonomy]$ pwd
/rsgrps/bh_class/jetjr/taxonomy
[jamesthornton@service2 taxonomy]$ ls
barplots                           SRR1647237-classout
old                                SRR1647238-centrifuge_report.tsv
SRR1647144-centrifuge_report.tsv  SRR1647238-classout
SRR1647144-classout                SRR1647239-centrifuge_report.tsv
SRR1647145-centrifuge_report.tsv  SRR1647239-classout
SRR1647145-classout                SRR1647240-centrifuge_report.tsv
SRR1647236-centrifuge_report.tsv  SRR1647240-classout
SRR1647236-classout                SRR1647260-centrifuge_report.tsv
SRR1647237-centrifuge_report.tsv  SRR1647260-classout
[jamesthornton@service2 taxonomy]$
```

Step 21.

Copy + Paste the following into a script called cent_barplots.R

Important: Edit cent.dir and out.dir to include the correct paths

- Edit cent.dir to include the path to your taxonomy directory (/rsgrps/bh_class/username/taxonomy/)
- Edit out.dir to include the path to your barplots diretory (/rsgrps/bh_class/username/taxonomy/barplots/)

cmd COMMAND

```
#!/usr/bin/env Rscript

#----------EDIT HERE----------
cent.dir <- "/rsgrps/bh_class/username/taxonomy/"
out.dir <- "/rsgrps/bh_class/username/taxonomy/barplots/"
#--------------------------------

file.names <- dir(cent.dir, pattern="-centrifuge_report.tsv")

gen_barplot <- function (data) {
  data_title <- gsub("-centrifuge_report.tsv", "", data)
  data <- read.delim(paste0(i, data))
  total_reads <- sum(data$numReads)
  proportion_classified <- data$numReads / total_reads
  data["proportion_classified"] <- proportion_classified
  read_subset <-
 subset(data, proportion_classified > 0.005, select = c("name", "numReads", "proportion_cla
ssified"))
  read_subset$numReads <- as.numeric(read_subset$numReads)
  png(filename=paste0(out.dir,data_title,"_taxonomy.png"), width = 600, height = 600)
  op <- par(mar=c(15, 8, 4, 2) + 0.1, mgp = c(10, 1, 0))
  p1 <-
 barplot(read_subset$proportion_classified, main=paste0("Read Proportional Classification:
",data_title), names.arg = read_subset$name, las=2, cex.names = 1, cex.axis = 1, ylab="Prop
ortion Classified", ylim = c(0, 0.90))
  grid(nx=NA, ny=NULL)
  print(p1)
  dev.off()
}

for (i in cent.dir) {
  lapply(file.names, gen_barplot)
}
```

Make sure to edit username in cent.dir and out.dir to include YOUR path. Also ensure that both cent.dir and out.dir end with the slash

NOTES

James Thornton Jr 28 Oct 2016

This R script will calculate the total number of reads and then divide the classified reads by the total for each hit generating a proportion classified statistic. Only hits with a proportion of 0.5% of reads classified will be plotted.

Step 22.

Once you have edited cent.dir and out.dir save and close the file. Make cent_barplots.R executable.

cmd COMMAND

```
$ chmod +x cent_barplots.R
```

Step 23.

Load the module R:

cmd COMMAND

```
$ module load R
```

Step 24.

Execute cent_barplots.R

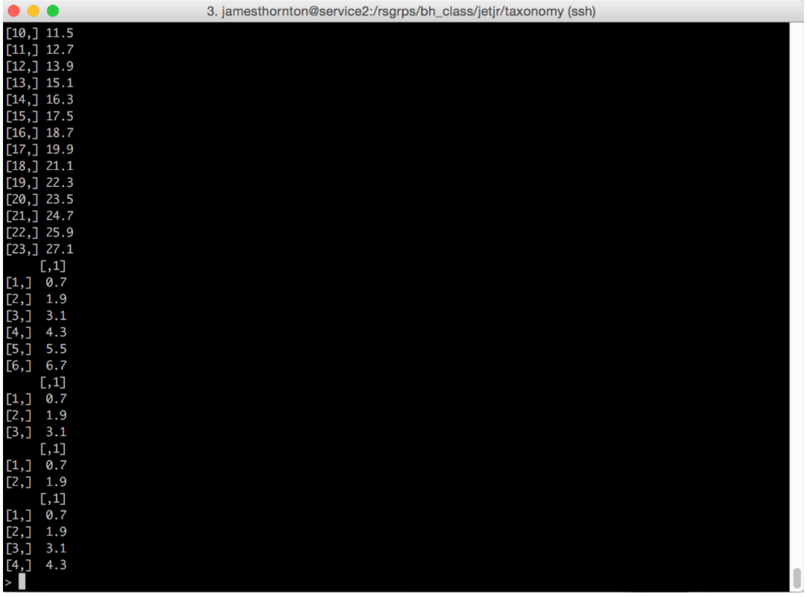
cmd COMMAND

```
$ ./cent_barplots.R
```

Step 25.

You should see something similar to what is shown below.

EXPECTED RESULTS


3. jamesthornton@service2:/rsgrps/bh_class/jetjr/taxonomy (ssh)

```
[10,] 11.5
[11,] 12.7
[12,] 13.9
[13,] 15.1
[14,] 16.3
[15,] 17.5
[16,] 18.7
[17,] 19.9
[18,] 21.1
[19,] 22.3
[20,] 23.5
[21,] 24.7
[22,] 25.9
[23,] 27.1
     [,1]
[1,]  0.7
[2,]  1.9
[3,]  3.1
[4,]  4.3
[5,]  5.5
[6,]  6.7
     [,1]
[1,]  0.7
[2,]  1.9
[3,]  3.1
     [,1]
[1,]  0.7
[2,]  1.9
     [,1]
[1,]  0.7
[2,]  1.9
[3,]  3.1
[4,]  4.3
>
```

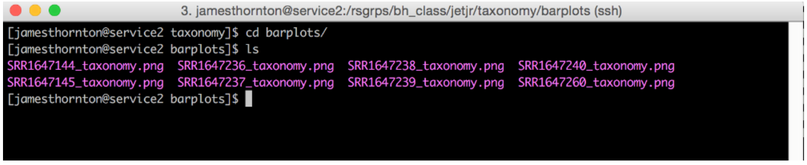
Step 26.

Move into your barplots directory and make sure you have 6 .png images.

cmd COMMAND

```
$ cd /rsgrps/bh_class/username/taxonomy/barplots
$ ls
```

EXPECTED RESULTS


3. jamesthornton@service2:/rsgrps/bh_class/jetjr/taxonomy/barplots (ssh)

```
[jamesthornton@service2 taxonomy]$ cd barplots/
[jamesthornton@service2 barplots]$ ls
SRR1647144_taxonomy.png  SRR1647236_taxonomy.png  SRR1647238_taxonomy.png  SRR1647240_taxonomy.png
SRR1647145_taxonomy.png  SRR1647237_taxonomy.png  SRR1647239_taxonomy.png  SRR1647260_taxonomy.png
[jamesthornton@service2 barplots]$
```

Step 27.

To view the images you must scp to your local machine. Open a new terminal (don't log into hpc). Determine where you want to store the files on your local machine and move into that directory.

NOTES

James Thornton Jr 28 Oct 2016

Windows users using Cygwin, your file will be stored in C:/cygwin64/home/USER. Just open a new terminal window and proceed to next step (you can't move to a specific local directory).

Step 28.

Execute the following command to scp the .png files to your local machine:

cmd COMMAND

```
$ scp netid@hpc.arizona.edu:/rsgrps/bh_class/username/taxonomy/barplots/*.png .
```

Replace netid and username. (They may be different).

Step 29.

You can now open the images on your local machine. Reminder that windows users will have their images in C:/cygwin64/home/USER.

Step 30.

Report on what you've found for each sample. Make sure to state the method used to obtain these results.