Script P12: Antibiotic Resistance

HANNIGAN GD, GRICE EA, ET AL.

Abstract

This protocol provides a method for analysis of the relative abundance of potential antibiotic resistance genes found in the virome. We use the Comprehensive Antibiotic Resistance Database (CARD) as our reference for potential antibiotic resistance genes.

Citation: HANNIGAN GD, GRICE EA, ET AL. Script P12: Antibiotic Resistance. **protocols.io** dx.doi.org/10.17504/protocols.io.ehwbb7e
Published: 10 Mar 2016

Guidelines

Required Software:

- CARD
- VFDB
- NCBI's BLAST+ v2.2.0
- Bowtie2-2.1.0

Relevant Files
Output:

- CARD_abx_resistance/CARD_annotated_orfs_in_otu_table.tsv
- CARD_abx_resistance/orf_otu_table.tsv

Perl Script: calculate_abundance_from_sam.pl
R Script: R14

Protocol

Step 1.
Download antibiotic resistance ontology reference info to use with the database.

cmd COMMAND
```
wget http://arpcard.mcmaster.ca/obo-download/aro.obo
```

NOTES
Geoffrey Hannigan 04 Feb 2016
Working directory for this reference modification is './references/CARD/'

Step 2.
Download the antibiotic resistance protein and gene database.

cmd COMMAND
```
wget http://arpcard.mcmaster.ca/blast/db/protein/AR-polypeptides.fa.gz
```

Step 3.
Unzip the zipped files that were downloaded.

cmd COMMAND

```
gunzip ./*.gz
```

Step 4.

Build blast databases from the antibiotic resistance databases. Make the protein reference databases.

cmd COMMAND

```
makeblastdb -dbtype prot -in AR-polypeptides.fa -out AR-polypeptides-db
```

Step 5.

We also need a usable reference table so that we can annotate the antibiotic resistance gene hits with antibiotic resistance gene category names, which will especially aid the visualization of the relative abundance profiles.

cmd COMMAND

```
cat ./aro.obo | tr "\n" "@" | sed 's/@@/\n/g' | grep -v format-version | grep -
v Typedef | sed 's/\[Term\]@id\:\s//g' | sed 's/@.*@is_a/\tis_a/' | grep is_a | sed 's/@rel
ationship.*//' | sed 's/is_a.*\!\s//' | sed 's/ /_/g' > ./ARO_numbers_and_AR_groups.tsv
```

Step 6.

Get a list of ARO numbers with their corresponding gene ID numbers and taxonomic associations from fasta. The fasta is annotated as a heirarchy so all ARO numbers should be taken.

cmd COMMAND

```
grep '>' AR-
polypeptides.fa | sed 's/>//' | sed 's/ARO:1000001//g' |sed 's/\s.*ARO/\tARO/' | sed 's/\ .
*\[/\t[/' | sed 's/ /_/g' > ./gene_IDs_and_ARO_numbers_and_AR_groups.tsv
```

Step 7.

Next, merge the files (using awk) into a single reference database

cmd COMMAND

```
awk 'FNR==NR { a[$1]=$2; next } $2 in a { print a[$2]"\t"$1"\t"$2"\t"$3 }' ./ARO_numbers_an
d_AR_groups.tsv ./gene_IDs_and_ARO_numbers_and_AR_groups.tsv > ./CARD_annotation_reference.
tsv
```

Step 8.

From here we can annotate the virome open reading frames with the CARD reference database. We also want to get the CARD reference gene length so that we can calculate the RPKM values below. Once we have this information, we will use Bowtie to map each sample's individual, non-assembled reads to the annotated ORFs. This information can be used to generate a relative abundance table (like an OTU table), like the relative abundance table we described in our taxonomic analysis.

Step 9.

We want to calculate the relative abundances as RPKM, since there can be a skew in the lengths of certain antibiotic resistance gene lengths.

cmd COMMAND

```
# Now back to the usual top level working directory

#Make directory for the results
mkdir ./CARD_taxonomy_using_orfs
```

Step 10.

Perform blastx of the predicted ORFs against the CARD database reference. Use gene fasta file (from glimmer3) from the predict_orfs_using_glimmer.sh script.

cmd COMMAND

```
blastx -query ./glimmer3/output/Contigs_no_block_with_names_glimmer_output_final.fa -
out ./CARD_taxonomy_using_orfs/blastx_CARD_glimmer_total_orfs.txt -
db /project/egricelab/references/CARD/AR-polypeptides-db -outfmt 6 -num_threads 16 -
max_target_seqs 1 -evalue 1e-5
```

Step 11.

Filter by percent identity this way since the functionality is not built into blastx.

cmd COMMAND

```
mv ./CARD_taxonomy_using_orfs/blastx_CARD_glimmer_total_orfs.txt ./CARD_taxonomy_using_orfs
/blastx_CARD_glimmer_total_orfs_all.txt
awk '!($3
```

Step 12.

Also calculate lengths of the orfs.

cmd COMMAND

```
mkdir ./glimmer3/orf_stats
awk 'NR % 2 {printf $0"\t"} !(NR % 2) {print length($0)}' ./glimmer3/output/Contigs_no_bloc
k_with_names_glimmer_output_final.fa > ./glimmer3/orf_stats/orf_length.txt
sed 's/>.*\(orf\)/\1/g' ./glimmer3/orf_stats/orf_length.txt | sed 's/___[^\t]\+//' > ./glim
mer3/orf_stats/orf_length_formatted.txt
```

Step 13.

Get abundance of sequences that map to the variosu ORFs. Run bowtie2 of the negative cleaned samples against the contig reference database. First build bowtie reference of the ORFs.

cmd COMMAND

```
mkdir ./CARD_taxonomy_using_orfs/bowtie2_neg_clean_against_unanot_orfs
bowtie2-build -
f ./glimmer3/output/Contigs_no_block_with_names_glimmer_output_final.fa ./CARD_taxonomy_usi
ng_orfs/bowtie2_neg_clean_against_unanot_orfs/bowtie2_orf_build
```

Step 14.

Map the sequences to the ORFs.

cmd COMMAND

```
mkdir ./CARD_taxonomy_using_orfs/bowtie2_neg_cleaned_hits
run_bowtie2_against_orfs () {
if [ -f /etc/profile.d/modules.sh ]; then
       source /etc/profile.d/modules.sh
fi
module load bowtie2-2.1.0
bowtie2 -
x ./CARD_taxonomy_using_orfs/bowtie2_neg_clean_against_unanot_orfs/bowtie2_orf_build -
f ./negative_clean_seqs/$1 -S ./CARD_taxonomy_using_orfs/bowtie2_neg_cleaned_hits/$1 -
L 25 -N 1
}
export -f run_bowtie2_against_orfs
ls ./negative_clean_seqs/ | xargs -I {} --max-procs=64 sh -c 'bsub -K -
q max_mem64 "run_bowtie2_against_orfs {}"'
```

Step 15.

Rename the files to be .sam files.

cmd COMMAND

```
for file in $(ls ./CARD_taxonomy_using_orfs/bowtie2_neg_cleaned_hits); do
    mv ./CARD_taxonomy_using_orfs/bowtie2_neg_cleaned_hits/"${file}" ./CARD_taxonomy_using_
orfs/bowtie2_neg_cleaned_hits/"${file/%.fa/.sam}"
done
```

Step 16.

Calculate the ORF hit abundances form bowtie2 using Qi's estimation perl script.

cmd COMMAND

```
mkdir ./CARD_taxonomy_using_orfs/abundance_from_sam
for file in $(ls ./CARD_taxonomy_using_orfs/bowtie2_neg_cleaned_hits); do
    perl calculate_abundance_from_sam.pl ./CARD_taxonomy_using_orfs/bowtie2_neg_cleaned_hit
s/${file} ./CARD_taxonomy_using_orfs/abundance_from_sam/${file}
done
```

NOTES

Geoffrey Hannigan 04 Feb 2016

Perl script available here.

Step 17.

Rename the sam files to text files.

cmd COMMAND

```
for file in $(ls ./CARD_taxonomy_using_orfs/abundance_from_sam); do
    mv ./CARD_taxonomy_using_orfs/abundance_from_sam/"${file}" ./CARD_taxonomy_using_orfs/a
bundance_from_sam/"${file/%.sam/.txt}"
done
```

Step 18.

Add in ORF length information using awk.

cmd COMMAND

```
mkdir ./CARD_taxonomy_using_orfs/CARD_abundance_with_length
for file in $(ls ./CARD_taxonomy_using_orfs/abundance_from_sam); do
    sed 's/__len\=/\t/' ./CARD_taxonomy_using_orfs/abundance_from_sam/${file} | tail -
n +2 > ./CARD_taxonomy_using_orfs/CARD_abundance_with_length/${file}
done
```

Step 19.

Generate rpkm information for hits to the ORFs.

cmd COMMAND

```
mkdir ./CARD_taxonomy_using_orfs/CARD_rpkm
for file in $(ls ./CARD_taxonomy_using_orfs/CARD_abundance_with_length); do
    export SUM=$(awk '{ SUM += $3 } END { print SUM }' ./CARD_taxonomy_using_orfs/CARD_abun
dance_with_length/${file})
```

Step 20.

Add an echo of the sum value to confirm that the sum is being calculated.

cmd COMMAND

```
echo Sum is $SUM for ${file}
    awk --
assign sum=$SUM '{ print $1"\t"$2"\t"$3"\t"($3*1000000000/($2*sum)) }' ./CARD_taxonomy_usin
g_orfs/CARD_abundance_with_length/${file} | sed "1 s/^ORF_ID\tORF_Length\tHit_Count\t${fil
e}\n/" > ./CARD_taxonomy_using_orfs/CARD_rpkm/${file}
done
```

Step 21.

Match the rpkm values to an ORF master list for generating a complete ORF OTU table. First generate a master list of the ORFs.

cmd COMMAND

```
egrep '>' ./glimmer3/output/Contigs_no_block_with_names_glimmer_output_final.fa | sed 's/__
len.*//' | sed 's/>//' > ./glimmer3/master_orf_list.txt
```

Step 22.

Add a header to the master list for later.

cmd COMMAND

```
sed '1 s/^/ORF_ID\n/' ./glimmer3/master_orf_list.txt > ./glimmer3/master_orf_list_with_head
er.txt
```

Step 23.

Align each individual relative abundance chart to the master ORF list.

cmd COMMAND

```
mkdir ./CARD_taxonomy_using_orfs/CARD_rpkm_smpls_to_master_orf_list
for file in $(ls ./CARD_taxonomy_using_orfs/CARD_rpkm); do
    awk 'FNR==NR {a[$1]=$4;next}{ print $1"\t"a[$1] }' ./CARD_taxonomy_using_orfs/CARD_rpkm
/${file} ./glimmer3/master_orf_list.txt | sed '/[0-9]\t[0-9]/!s/$/0/' | sed "1 s/^ORF_ID\t
```

```
${file}\n/" > ./CARD_taxonomy_using_orfs/CARD_rpkm_smpls_to_master_orf_list/${file}
done
```

Step 24.

Get lists of the rpkm information only so that they can be merged with the master file.

cmd COMMAND

```
mkdir ./CARD_taxonomy_using_orfs/CARD_rpkm_smpls_to_master_orf_list_only_rpkm
for file in $(ls ./CARD_taxonomy_using_orfs/CARD_rpkm_smpls_to_master_orf_list); do
    cut -
f 2 ./CARD_taxonomy_using_orfs/CARD_rpkm_smpls_to_master_orf_list/${file} > ./CARD_taxonomy
_using_orfs/CARD_rpkm_smpls_to_master_orf_list_only_rpkm/${file}
done
paste ./glimmer3/master_orf_list_with_header.txt ./CARD_taxonomy_using_orfs/CARD_rpkm_smpls
_to_master_orf_list_only_rpkm/* > ./CARD_taxonomy_using_orfs/orf_otu_table.tsv
```

Step 25.

Create a reference table with the ORFs (annotated with blastx in annotate_orfs_CARD_VFDB.sh) and their corresponding CARD hits.

cmd COMMAND

```
awk 'FNR==NR { a[$2]=$1"\t"$4; next } $2 in a { print $1"\t"a[$2] }' /project/egricelab/ref
erences/CARD/CARD_annotation_reference.tsv ./CARD_taxonomy_using_orfs/blastx_CARD_glimmer_t
otal_orfs.txt | sed 's/__len=[^\t]\+\t/\t/' > ./CARD_taxonomy_using_orfs/CARD_annotated_orf
s.tsv
```

Step 26.

Substitute the ORFs in the orf_otu_table.tsv with CARD annotations.

cmd COMMAND

```
awk 'FNR==NR { a[$1]=$0; next } $1 in a { print $2"\t"a[$1] }' ./CARD_taxonomy_using_orfs/o
rf_otu_table.tsv ./CARD_taxonomy_using_orfs/CARD_annotated_orfs.tsv | cut -f '1,3-
' > ./CARD_taxonomy_using_orfs/CARD_annotated_orfs_in_otu_table_without_header.tsv
```

Step 27.

Get header from original OTU table.

cmd COMMAND

```
head -
n 1 ./CARD_taxonomy_using_orfs/orf_otu_table.tsv > ./CARD_taxonomy_using_orfs/orf_otu_table
_header.tsv
cat ./CARD_taxonomy_using_orfs/orf_otu_table_header.tsv ./CARD_taxonomy_using_orfs/CARD_ann
otated_orfs_in_otu_table_without_header.tsv | sed 's/_R1\.txt//g' > ./CARD_taxonomy_using_o
rfs/CARD_annotated_orfs_in_otu_table.tsv
```

NOTES

Geoffrey Hannigan 04 Feb 2016

This table can be brought into R for further processing.

Step 28.

We can then determine how many CARD ORFs are found on phage contigs, the taxonomic IDs of those phages, and visualize those contigs. The visualization information (contig fastas and glimmer annotations as .gff3 files) can be used in the program Geneious.

cmd COMMAND

```
# Get a list of the contigs that contain CARD ORFs
cut -
f 1 ./CARD_taxonomy_using_orfs/CARD_annotated_orfs.tsv | sed 's/_\.orf.*$//' | sort | uniq
| sed 's/^/\>/' | sed 's/$/_/' > ./CARD_taxonomy_using_orfs/CARD_ORFs_for_comparing_to_phag
es.tsv
```

NOTES

Geoffrey Hannigan 04 Feb 2016

In this section we are going to determine how many ORFs are found on contigs that are annotated

as bacteriophages. In other words, calculate the number of phage contigs that contain CARD ORFs

Step 29.

Get a list of the contigs that contain phage ORFs that were generated in predict_temperate_phages.sh

cmd COMMAND

```
cp ./phage_lifecycle/integrase/phage_contigs_no_negs_uniq.txt ./CARD_taxonomy_using_orfs/ph
age_contigs_no_negs_uniq.txt

awk 'FNR==NR { a[$1]=$1; next } $1 in a { print $1"\t"a[$1] }' ./CARD_taxonomy_using_orfs/C
ARD_ORFs_for_comparing_to_phages.tsv ./CARD_taxonomy_using_orfs/phage_contigs_no_negs_uniq.
txt > ./CARD_taxonomy_using_orfs/phage_contigs_containing_CARD_ORFs.tsv
```

Step 30.

To easily visualize the contigs, run custom perl scripts and visualize in geneous.

cmd COMMAND

```
mkdir ./CARD_contig_visualization/overallContigAnnotation
perl annotateGlimmerORFs.pl ./glimmer3/output/Contigs_no_block_with_names_glimmer_output.pr
edict ./uniprot_taxonomy_using_orfs/blastx_raw_results/blastx_trembl_glimmer_total_orfs.txt
 ./CARD_contig_visualization/overallContigAnnotation/ContigsFromGlimmer.txt
```

Step 31.

Then replace the uniprot accession numbers with human readable gene names.

cmd COMMAND

```
perl annotateGlimerUniprotAcc.pl ./CARD_contig_visualization/overallContigAnnotation/Contig
sFromGlimmer.txt ./references/uniprot_gene_function_and_acc_reference_no_space.tsv ./CARD_c
ontig_visualization/overallContigAnnotation/ContigsFromGlimmerAnnotated.txt
```

Step 32.

Finally convert the glimmer .predict format to gff3 so it can be used in geneous and other genome browsers (Geneious). Here you need to specify the contig ID number you want, so it might be best to run it through a loop. Here is an example of how it works for contig number 1.

cmd COMMAND

```
perl GlimmerPredict2Gff3.pl ./CARD_contig_visualization/overallContigAnnotation/ContigsFrom
GlimmerAnnotated.txt 1 ./CARD_contig_visualization/overallContigAnnotation/ContigsFromGlimm
erAnnotated.gff3
```

Step 33.

To get all of the contigs with CARD genes and hits to phage genes (or are themselves phage genes), first get togehter a list of those contigs.

cmd COMMAND

```
mkdir ./CARD_contig_visualization/PhageCardGenesAnnotations/
mkdir ./CARD_contig_visualization/PhageCardGenesAnnotations/gff3Files
mkdir ./CARD_contig_visualization/PhageCardGenesAnnotations/contigFastaFiles
cut -
f 1 ./CARD_contig_visualization/PhageCardGenes/list_CARD_and_phage_contigs.tsv | sed 's/>//
' | sed 's/_//' > ./CARD_contig_visualization/PhageCardGenesAnnotations/ContigAccList.tsv
for i in $(cat ./CARD_contig_visualization/PhageCardGenesAnnotations/ContigAccList.tsv); do
    perl GlimmerPredict2Gff3.pl ./CARD_contig_visualization/overallContigAnnotation/Contigs
FromGlimmerAnnotated.txt ${i} ./CARD_contig_visualization/PhageCardGenesAnnotations/gff3Fil
es/${i}.gff3
    grep -
A 1 \>${i}_ ./ray_contigs_from_total_cat_pairs/Contigs_no_block_with_names.fasta > ./CARD_c
ontig_visualization/PhageCardGenesAnnotations/contigFastaFiles/${i}.fa
done
```

Step 34.

Access what taxa the CARD annotate ORFs are co-localizing with. Make directory for output.

cmd COMMAND

```
mkdir ./CARD_contig_visualization/PhageCardGenes
```

Step 35.
Test whether these contigs are also contigs with phage genes present.

cmd COMMAND
```
awk 'FNR==NR { a[$1]=$1; next } $1 in a { print $1"\t"a[$1] }' ./CARD_contig_visualization/
PropPhageCardGenes/listCardContigs.tsv ./CARD_taxonomy_using_orfs/phage_contigs_no_negs_uni
q.txt > ./CARD_contig_visualization/PhageCardGenes/list_CARD_and_phage_contigs.tsv
```
Step 36.
As mentioned earlier in analysis, there are 119 CARD + phage contigs. Now get the predicted IDs for those contigs that contain phage and card ORFs.

cmd COMMAND
```
wk 'FNR==NR { a[$1]=$1"\t"$2"\t"$3"\t"$4"\t"$5"\t"$6; next } $1 in a { print $1"\t"a[$1] }'
 ./CARD_contig_visualization/contig_id_reference_table_format_for_CARD.tsv ./CARD_contig_vi
sualization/PhageCardGenes/list_CARD_and_phage_contigs.tsv > ./CARD_contig_visualization/Ph
ageCardGenes/list_CARD_contig_phageIDs.tsv
```

NOTES
Geoffrey Hannigan 04 Feb 2016
They are mostly within bacillus phage.

Step 37.
Get counts of the IDs.

cmd COMMAND
```
cut -
f 7 ./CARD_contig_visualization/PhageCardGenes/list_CARD_contig_phageIDs.tsv | sort | uniq
-c > ./CARD_contig_visualization/PhageCardGenes/count_CARD_contig_phageIDs.tsv
```
Step 38.
Additionally annotate ORFs as viruelence factors, determine how many are found on phage contigs, and visualize those contigs. First we need to download the virulence factor database and create a blast formatted reference database. Make a new dir and move working directory to location where the CARD reference seqs are being stored.

cmd COMMAND
```
mkdir /project/egricelab/references/VFDB
cd ./references/VFDB
```
Step 39.
Download the VFDB (this will be the most up-to-date version which was updated Fri Sep 5 10:06:01 2014 according to the website; accessed September 15, 2014). This will be the fasta file of the virulence factor polypeptides.

cmd COMMAND
```
wget http://www.mgc.ac.cn/VFs/Down/VFs.faa.gz
```
Step 40.
Download the protein sequences for comparitive studies.

cmd COMMAND
```
wget http://www.mgc.ac.cn/VFs/Down/CP_VFs.faa.gz
```
Step 41.
Gunzip and untar the files.

cmd COMMAND
```
gunzip VFs.faa.gz
gunzip CP_VFs.faa.gz
```
Step 42.
Remove the block format of the fasta files.

cmd COMMAND

```
perl remove_block_fasta_format.pl VFs.faa VFs_no_block.fa
perl remove_block_fasta_format.pl CP_VFs.faa CP_VFs_no_block.fa
```

Step 43.

Make directory for the results.

cmd COMMAND

```
mkdir ./VFDB_taxonomy_using_orfs
```

Step 44.

Make blast reference.

cmd COMMAND

```
makeblastdb -dbtype prot -in ./references/VFDB/VFs_no_block.fa -out ./references/VFDB/VF_db
```

Step 45.

Perform blastx of the predicted ORFs against the VFDB database reference. Use gene fasta file (from glimmer3) from the predict_orfs_using_glimmer.sh script.

cmd COMMAND

```
blastx -query ./glimmer3/output/Contigs_no_block_with_names_glimmer_output_final.fa -
out ./VFDB_taxonomy_using_orfs/blastx_VFDB_glimmer_total_orfs.txt -
db ./references/VFDB/VF_db -outfmt 6 -num_threads 16 -max_target_seqs 1 -evalue 1e-5
```

Step 46.

Filter by percent ID.

cmd COMMAND

```
mv ./VFDB_taxonomy_using_orfs/blastx_VFDB_glimmer_total_orfs.txt ./VFDB_taxonomy_using_orfs
/blastx_VFDB_glimmer_total_orfs_all.txt
awk '!($3
```

Step 47.

All of the ORF relative abundance table information was already done using the script calculate_orf_CARD_VFDB_rel_abund.sh, so all we have to do here is annotate the ORFs with the VFDB information so that it can be used in R for determining the presence of high quality reads.

cmd COMMAND

```
grep '>' ./references/VFDB/VFs_no_block.fa | sed 's/^>\(.*\) (gi.*) \(.*\) \[\(.*\)\].*/\1\
t\2\t\3/' | sed 's/ /_/g' > ./references/VFDB/VFDB_annotation_reference.tsv
```

Step 48.

Create a reference table with the ORFs (annotated with blastx in annotate_orfs_VFDB_VFDB.sh) and their corresponding VFDB hits.

cmd COMMAND

```
awk 'FNR==NR { a[$1]=$0; next } $2 in a { print $1"\t"$2"\t"a[$2] }' ./references/VFDB/VFDB
_annotation_reference.tsv ./VFDB_taxonomy_using_orfs/blastx_VFDB_glimmer_total_orfs.txt | s
ed 's/__len=[^\t]\+\t/\t/' | sed 's/ //g'> ./VFDB_taxonomy_using_orfs/VFDB_annotated_orfs.t
sv
```

Step 49.

Substitute the orfs in the orf_otu_table.tsv with VFDB annotations.

cmd COMMAND

```
awk 'FNR==NR { a[$1]=$0; next } $1 in a { print $2"\t"a[$1] }' ./CARD_taxonomy_using_orfs/o
rf_otu_table.tsv ./VFDB_taxonomy_using_orfs/VFDB_annotated_orfs.tsv | cut -f '1,3-
' > ./VFDB_taxonomy_using_orfs/VFDB_annotated_orfs_in_otu_table_without_header.tsv
```

Step 50.

Get header from original OTU table.

cmd COMMAND

```
head -
n 1 ./CARD_taxonomy_using_orfs/orf_otu_table.tsv > ./VFDB_taxonomy_using_orfs/orf_otu_table
_header.tsv
cat ./VFDB_taxonomy_using_orfs/orf_otu_table_header.tsv ./VFDB_taxonomy_using_orfs/VFDB_ann
```

```
otated_orfs_in_otu_table_without_header.tsv | sed 's/_R1\.txt//g' > ./VFDB_taxonomy_using_o
rfs/VFDB_annotated_orfs_in_otu_table.tsv
```

NOTES

Geoffrey Hannigan 09 Feb 2016

This table can be brought into R for futher processing.

Step 51.

In this section I am going to determine how many ORFs are found on contigs that are annotated as bacteriophages. In other words: calculate the number of phage contigs that contain VFDB ORFs.

Step 52.

Get a list of the contigs that contain VFDB ORFs. This shows there are 189 uniq contigs that contain VFDB ORFs.

cmd COMMAND

```
cut -
f 1 ./VFDB_taxonomy_using_orfs/VFDB_annotated_orfs.tsv | sed 's/_\.orf.*$//' | sort | uniq
| sed 's/^/\>/' | sed 's/$/_/' > ./VFDB_taxonomy_using_orfs/VFDB_ORFs_for_comparing_to_phag
es.tsv
```

Step 53.

Get a list of the contigs that contain phage ORFs that were generated in predict_temperate_phages.sh.

cmd COMMAND

```
cp ./phage_lifecycle/integrase/phage_contigs_no_negs_uniq.txt ./VFDB_taxonomy_using_orfs/ph
age_contigs_no_negs_uniq.txt

awk 'FNR==NR { a[$1]=$1; next } $1 in a { print $1"\t"a[$1] }' ./VFDB_taxonomy_using_orfs/V
FDB_ORFs_for_comparing_to_phages.tsv ./VFDB_taxonomy_using_orfs/phage_contigs_no_negs_uniq.
txt > ./VFDB_taxonomy_using_orfs/phage_contigs_containing_VFDB_ORFs.tsv
```

Step 54.

Assess what taxa the VFDB annotate ORFs are co-localizing with. Make directory for output.

cmd COMMAND

```
mkdir ./VFDB_contig_visualization
mkdir ./VFDB_contig_visualization/PhageVFDBGenes
```

Step 55.

Test whether these contigs are also contigs with phage genes present. Use the contig list from calculate_orf_VFDB_rel_abund.sh which has contigs with both phages and VFDB orfs.

cmd COMMAND

```
cut -
f 1 ./VFDB_taxonomy_using_orfs/phage_contigs_containing_VFDB_ORFs.tsv > ./VFDB_contig_visua
lization/PhageVFDBGenes/phage_contigs_containing_VFDB_ORFs_single_col.tsv
```

Step 56.

As mentioned earlier in analysis, there are 68 VFDB + phage contigs. Now get the predicted IDs for those contigs that contain phage and VFDB orfs.

cmd COMMAND

```
awk 'FNR==NR { a[$1]=$1"\t"$2"\t"$3"\t"$4"\t"$5"\t"$6; next } $1 in a { print $1"\t"a[$1] }
' ./CARD_contig_visualization/contig_id_reference_table_format_for_CARD.tsv ./VFDB_contig_v
isualization/PhageVFDBGenes/phage_contigs_containing_VFDB_ORFs_single_col.tsv > ./VFDB_cont
ig_visualization/PhageVFDBGenes/list_VFDB_contig_phageIDs.tsv
```

Step 57.

They are mostly within bacillus phages. Get counts of the IDs.

cmd COMMAND

```
cut -
f 7 ./VFDB_contig_visualization/PhageVFDBGenes/list_VFDB_contig_phageIDs.tsv | sort | uniq
```

```
-c > ./VFDB_contig_visualization/PhageVFDBGenes/count_VFDB_contig_phageIDs.tsv
```

Step 58.

To easily visualize the contigs, run my perl scripts and visualize in geneous.

cmd COMMAND

```
mkdir ./VFDB_contig_visualization/overallContigAnnotation
perl annotateGlimmerORFs.pl ./glimmer3/output/Contigs_no_block_with_names_glimmer_output.pr
edict ./uniprot_taxonomy_using_orfs/blastx_raw_results/blastx_trembl_glimmer_total_orfs.txt
 ./VFDB_contig_visualization/overallContigAnnotation/ContigsFromGlimmer.txt
```

Step 59.

Then replace the uniprot accession numbers with human readable gene names.

cmd COMMAND

```
perl annotateGlimerUniprotAcc.pl ./VFDB_contig_visualization/overallContigAnnotation/Contig
sFromGlimmer.txt /project/egricelab/references/UniProt-Virus-
Phage/uniprot_gene_function_and_acc_reference_no_space.tsv ./VFDB_contig_visualization/over
allContigAnnotation/ContigsFromGlimmerAnnotated.txt
```

Step 60.

Finally convert the glimmer .predict format to gff3 so it can be used in geneous and other genome browsers. Here you need to specify the contig ID number that you want, so it might be best to run it through a loop. Here is an example of how it works with contig 1.

cmd COMMAND

```
perl GlimmerPredict2Gff3.pl ./VFDB_contig_visualization/overallContigAnnotation/ContigsFrom
GlimmerAnnotated.txt 1 ./VFDB_contig_visualization/overallContigAnnotation/ContigsFromGlimm
erAnnotated.gff3
```

Step 61.

To get all of the contigs with VFDB genes and hits to phage genes (or are themselves phage genes), first get together a list of those contigs.

cmd COMMAND

```
mkdir ./VFDB_contig_visualization/PhageVFDBGenesAnnotations/
mkdir ./VFDB_contig_visualization/PhageVFDBGenesAnnotations/gff3Files
mkdir ./VFDB_contig_visualization/PhageVFDBGenesAnnotations/contigFastaFiles
cut -
f 1 ./VFDB_contig_visualization/PhageVFDBGenes/list_VFDB_contig_phageIDs.tsv | sed 's/>//'
| sed 's/_//' > ./VFDB_contig_visualization/PhageVFDBGenesAnnotations/ContigAccList.tsv
for i in $(cat ./VFDB_contig_visualization/PhageVFDBGenesAnnotations/ContigAccList.tsv); do
    perl GlimmerPredict2Gff3.pl ./VFDB_contig_visualization/overallContigAnnotation/Contigs
FromGlimmerAnnotated.txt ${i} ./VFDB_contig_visualization/PhageVFDBGenesAnnotations/gff3Fil
es/${i}.gff3
    grep -
A 1 \>${i}_ ./ray_contigs_from_total_cat_pairs/Contigs_no_block_with_names.fasta > ./VFDB_c
ontig_visualization/PhageVFDBGenesAnnotations/contigFastaFiles/${i}.fa
done
```