Commands/Workflow for Binning, Annotations, Anvio, Bin Curation of 2017 Lost City Metagenome

##Binning with ABAWACA

```
srun prepare_esom_files.pl abawaca_prep_files H08.contigs.renamed.fa

srun abawaca abawaca_prep_files/esom.names abawaca_prep_files/esom.lrn
H08.contigs.renamed.fa abawaca_bins &
```

H08 bins: 13, 15, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 6, 8

##Prodigal Protein Prediction for each Bin

```
---------------------------------------------------------------------------
prodigal.sh

#! /bin/sh

#SBATCH --output prodigal.out
#SBATCH --error prodigal.err
#SBATCH --ntasks 19
#SBATCH --cpus-per-task 3
#SBATCH --nodes 1-2
#SBATCH --partition highmem
#SBATCH --mem-per-cpu 1G

srun prodigal -q -p meta -f gff -i 13.fasta -o 13.gff -a 13.faa &
srun prodigal -q -p meta -f gff -i 15.fasta -o 15.gff -a 15.faa &
srun prodigal -q -p meta -f gff -i 28.fasta -o 28.gff -a 28.faa &
srun prodigal -q -p meta -f gff -i 29.fasta -o 29.gff -a 29.faa &
srun prodigal -q -p meta -f gff -i 30.fasta -o 30.gff -a 30.faa &
srun prodigal -q -p meta -f gff -i 31.fasta -o 31.gff -a 31.faa &
srun prodigal -q -p meta -f gff -i 32.fasta -o 32.gff -a 32.faa &
srun prodigal -q -p meta -f gff -i 33.fasta -o 33.gff -a 33.faa &
srun prodigal -q -p meta -f gff -i 34.fasta -o 34.gff -a 34.faa &
srun prodigal -q -p meta -f gff -i 35.fasta -o 35.gff -a 35.faa &
srun prodigal -q -p meta -f gff -i 36.fasta -o 36.gff -a 36.faa &
srun prodigal -q -p meta -f gff -i 37.fasta -o 37.gff -a 37.faa &
srun prodigal -q -p meta -f gff -i 6.fasta -o 6.gff -a 6.faa &
srun prodigal -q -p meta -f gff -i 8.fasta -o 8.gff -a 8.faa &
---------------------------------------------------------------------------------------------------
```

##Annotations with Diamond for each protein prediction (done 2 times, with both T1000 and prokaryotes databases)

annotate.sh

```
#! /bin/sh

#SBATCH --output diamond.out
#SBATCH --error diamond.err
#SBATCH --ntasks 1
#SBATCH --cpus-per-task 5
#SBATCH -w node2
#SBATCH --partition batch
#SBATCH --mem-per-cpu 10G

diamond blastp --threads 5 --tmpdir /tmp --sensitive --evalue 0.000001 --outfmt 6 --out 13.KO.b6
--db /srv/databases/internal/diamond/KEGG-T10000.dmnd --query 13.faa
diamond blastp --threads 5 --tmpdir /tmp --sensitive --evalue 0.000001 --outfmt 6 --out 15.KO.b6
--db /srv/databases/internal/diamond/KEGG-T10000.dmnd --query 15.faa
diamond blastp --threads 5 --tmpdir /tmp --sensitive --evalue 0.000001 --outfmt 6 --out 28.KO.b6
--db /srv/databases/internal/diamond/KEGG-T10000.dmnd --query 28.faa
diamond blastp --threads 5 --tmpdir /tmp --sensitive --evalue 0.000001 --outfmt 6 --out 29.KO.b6
--db /srv/databases/internal/diamond/KEGG-T10000.dmnd --query 29.faa
diamond blastp --threads 5 --tmpdir /tmp --sensitive --evalue 0.000001 --outfmt 6 --out 30.KO.b6
--db /srv/databases/internal/diamond/KEGG-T10000.dmnd --query 30.faa
diamond blastp --threads 5 --tmpdir /tmp --sensitive --evalue 0.000001 --outfmt 6 --out 31.KO.b6
--db /srv/databases/internal/diamond/KEGG-T10000.dmnd --query 31.faa
diamond blastp --threads 5 --tmpdir /tmp --sensitive --evalue 0.000001 --outfmt 6 --out 32.KO.b6
--db /srv/databases/internal/diamond/KEGG-T10000.dmnd --query 32.faa
diamond blastp --threads 5 --tmpdir /tmp --sensitive --evalue 0.000001 --outfmt 6 --out 33.KO.b6
--db /srv/databases/internal/diamond/KEGG-T10000.dmnd --query 33.faa
diamond blastp --threads 5 --tmpdir /tmp --sensitive --evalue 0.000001 --outfmt 6 --out 34.KO.b6
--db /srv/databases/internal/diamond/KEGG-T10000.dmnd --query 34.faa
diamond blastp --threads 5 --tmpdir /tmp --sensitive --evalue 0.000001 --outfmt 6 --out 35.KO.b6
--db /srv/databases/internal/diamond/KEGG-T10000.dmnd --query 35.faa
diamond blastp --threads 5 --tmpdir /tmp --sensitive --evalue 0.000001 --outfmt 6 --out 36.KO.b6
--db /srv/databases/internal/diamond/KEGG-T10000.dmnd --query 36.faa
diamond blastp --threads 5 --tmpdir /tmp --sensitive --evalue 0.000001 --outfmt 6 --out 37.KO.b6
--db /srv/databases/internal/diamond/KEGG-T10000.dmnd --query 37.faa
diamond blastp --threads 5 --tmpdir /tmp --sensitive --evalue 0.000001 --outfmt 6 --out 6.KO.b6 -
-db /srv/databases/internal/diamond/KEGG-T10000.dmnd --query 6.faa
diamond blastp --threads 5 --tmpdir /tmp --sensitive --evalue 0.000001 --outfmt 6 --out 8.KO.b6 -
-db /srv/databases/internal/diamond/KEGG-T10000.dmnd --query 8.faa
diamond blastp --threads 5 --tmpdir /tmp --sensitive --evalue 0.000001 --outfmt 6 --out 14.KO.b6
--db /srv/databases/internal/diamond/KEGG-T10000.dmnd --query 14.faa
diamond blastp --threads 5 --tmpdir /tmp --sensitive --evalue 0.000001 --outfmt 6 --out 20.KO.b6
--db /srv/databases/internal/diamond/KEGG-T10000.dmnd --query 20.faa
```

```
diamond blastp --threads 5 --tmpdir /tmp --sensitive --evalue 0.000001 --outfmt 6 --out 23.KO.b6
--db /srv/databases/internal/diamond/KEGG-T10000.dmnd --query 23.faa
diamond blastp --threads 5 --tmpdir /tmp --sensitive --evalue 0.000001 --outfmt 6 --out 26.KO.b6
--db /srv/databases/internal/diamond/KEGG-T10000.dmnd --query 26.faa
diamond blastp --threads 5 --tmpdir /tmp --sensitive --evalue 0.000001 --outfmt 6 --out 27.KO.b6
--db /srv/databases/internal/diamond/KEGG-T10000.dmnd --query 27.faa
```

```
#! /bin/sh

#SBATCH --output diamond.proks.out
#SBATCH --error diamond.proks.err
#SBATCH --ntasks 1
#SBATCH --cpus-per-task 5
#SBATCH -w node2
#SBATCH --partition batch
#SBATCH --mem-per-cpu 10G

bins=("13" "15" "28" "29" "30" "31" "32" "33" "34" "35" "36" "37" "6" "8")
for b in "${bins[@]}"; do
  diamond blastp --threads 5 --tmpdir /tmp --sensitive --evalue 0.000001 --outfmt 6 --out
$b.KO.prok.b6 --db /srv/databases/internal/diamond/KEGG-prokaryotes.dmnd --query $b.faa
done
```

##CheckM to asses bin quality

```
srun --ntasks 1 --cpus-per-task 4 --partition highmem --mem 200G --out checkm.faa.out --err
checkm.faa.err checkm lineage_wf --genes -f H08_checkm_table.tsv --tab_table -t 4 --
pplacer_threads 4 -x faa prodigal_proteins H08_checkm_faa.results &
```

##Chris's script annotate_features.py makes these next steps obsolete now. But that was not written yet when I did the next steps to get geneCalls for Anvio

```
for sample in "13" "14" "15" "28" "29" "30" "31" "32" "33" "34" "35" "36" "37" "6" "8"
do
srun -p single python get_koID.py ${sample}.KO.prok.b6
/srv/databases/proteins/kegg/prokaryotes ${sample}.koID.proks.out &
done

for sample in "13" "14" "20" "23" "26" "27" "15" "28" "29" "30" "31" "32" "33" "34" "35" "36" "37"
"6" "8"
do
srun python get_koID.py ${sample}.KO.b6 /srv/databases/proteins/kegg/KO
${sample}.koID.KO.out &
done
```

##Annotate Features on metagenome (NOT BIN) from all .gffs from prodigal and b6 files from diamond

```
srun cat *.gff > H08.gff
```

```
awk -F'\t' '/^#/{print $0; next} {sub(/ID\=[^_]*_/, "ID="$1"_", $9); print
$1"\t"$2"\t"$3"\t"$4"\t"$5"\t"$6"\t"$7"\t"$8"\t"$9}' H08.gff > H08.clean.gff
```

```
srun cat annotations/diamond_output/* > all.KO.b6
```

```
srun -p highmem --mem 70G annotate_features --type CDS --mapping
/srv/databases/internal/json/KEGG-T10000.mapping.json,/srv/databases/internal/json/KEGG-
prokaryotes.mapping.json --fields
gene,gene_family,product,organism,database,md5,Ontology_term,dummy --conflict quality --out
H08.annotated.gff --b6 annotations/diamond_output/all.KO.b6 binning/abawaca_bins/final-
clusters/H08.clean.gff > H08.annotate.log &
```

##Count features to calculate gene coverage for metagenome

```
srun -p highmem --mem 70G count_features --attr gene --sorting name --units fpk,fpkm --out
H08.gene.abundance data/H08.mapped.sorted.bam H08.annotated.gff 2> H08.fpk.gene.log &
```

##Bin Refining

Pull out all contigs with same PhyloPythia id in bin, make new .fasta and .faa files just this list of contigs:

```
srun python ~/myScripts/makeFastaBin.py -c <list of contigs in .txt file> -f <bin .fasta file from
abawaca> -o <new .fasta file>
```

```
srun python ~/makeFaaBin.py -c <list of contigs in .txt file> -f <.faa file to be modified> -o <new
.faa file>
```

##CheckM on new bin with all same PhyloPythia id

```
srun --ntasks 1 --cpus-per-task 4 --partition highmem --mem 100G --out <outfile name> --err
<errorfile name> checkm lineage_wf --genes -f <sample. checkM table name> --tab_table -t 4 --
pplacer_threads 4 -x <new .faa file> <directory of that file> <results_name> &
```

##Gene Call for Anvio (Alex's script anvi-converter.py might work with Chris's new annotate_features.py created annotated .gff and be more sophisticated than my script)

```
srun -p single python ~/myScripts/geneCall.py -f <new .fasta_file> -k <get_koID_outfile> -a
<new .faa file> -g <geneCall outfile> -o <function.outfile> -s <database_name> &
```

##Generate the Anvio DB Files, <geneCall outfile> from geneCall.py, the gene calls and profile never really worked right. But it grouped the contigs and I could get coverage information and I just used that information to split the bin

```
srun -p single --mem 10G --err <error.out> --out <out.out> anvi-gen-contigs-database -f <new
.fasta file> -o anvio/$sample.contigs.db --external-gene-calls <geneCall outfile> &

srun -p single --mem 10G --out <out.out> --err <error.err> anvi-run-hmms -c
anvio/$sample.contigs.db &
```

##<function.outfile> from geneCall.py

```
srun -p single --mem 10G --err <error.err> --out <out.out> anvi-import-functions -c
anvio/$sample.contigs.db -i <function.outfile> &
```

##Map

```
srun -p batch --out <out.out> --err <error.err> --mem 20G bowtie2-build <new .fasta>
<new.bin.name> &

sbatch bt.sh
---------------------------------------------------
bt.sh
#! /bin/sh

#SBATCH --output bt2.out
#SBATCH --error bt2.err
#SBATCH --ntasks 1
#SBATCH --cpus-per-task 4
#SBATCH --partition batch
#SBATCH --mem 20G

bowtie2 -p 4 --very-sensitive -I 130 -X 174 -x <> --interleaved
<sample>.interleaved.atrim.decontam.qtrim.derep.fq.gz 2> bowtie.mapping.log | samtools sort -
@ 1 -m 10G -l 9 -O bam -T <new.bin.name> -o <new.bin.name>.mapped.sorted.bam
```

##Generate the PROFILE file for anvio

```
srun -p batch --mem 20G --error <error.err> --output <out.out> anvi-profile -i
<mapped.sorted.filt.derep.bam for .fasta file anvio db made with> -c anvio/$sample.contigs.db --
min-contig-length 3000 --sample-name $sample --output-dir anvio/$sample --cluster-contigs &

anvi-interactive -p $sample.PROFILE.db -c $sample.contigs.db
```

##Visually split bin by basal branching, rerun checkM

##Annotate features

```
awk -F'\t' '/^#/{print $0; next} {sub(/ID\=[^_]*_/, "ID="$1"_", $9); print
$1"\t"$2"\t"$3"\t"$4"\t"$5"\t"$6"\t"$7"\t"$8"\t"$9}' <OG bin gff file>.gff > <bin>.clean.gff
```

##Combine both T1000 and prokaryotes databases together to make one .b6 file

```
cat annotations/diamond_output/<bin>.KO.* > <bin>.all.b6
```

```
srun -p single --mem 50G annotate_features --type CDS --mapping
/srv/databases/internal/json/KEGG-T10000.mapping.json,/srv/databases/internal/json/KEGG-
prokaryotes.mapping.json --fields
gene,gene_family,product,organism,database,md5,Ontology_term,dummy --conflict quality --out
<bin>.annotated.gff --b6 <bin>.all.b6 <bin>.clean.gff 2> <bin>.annotate.log
```

##Get annotated .gff for refined bin

```
srun python ~/myScripts/getgffBin.py -f <anvio split .fasta bin file> -g <bin>.annotated.gff -o
<new bin>.annotated.gff
```

##Make mapping file for KEGG Mapper Online Tool

```
srun ~/myScripts/genesKO_from_gff.py <new bin>.annotated.gff <new bin>.forMapping.txt
```