Transposons and Repetitive Elements Identification

Indentification of repetitive elements and transposable elements (TE) is a key step in annotating genomes. By identifying these repetitive regions and masking them from gene prediction it prevents calling TEs genes.

In addition, discovery and annotation of transposable elements is supervise

Running Repeat Masker

The downloaded and installed RepBase on the UCR HPCC is from 2017 (RepBaseRepeatMaskerEdition-20170127.tar.gz). RepBase converted to a subscription model so availability of this an more recent versions of RepBase are only accessible with a subscription.

The Dfam consortium has constructed a database and approach for masking repeats using DNA HMMs to model repeat families. This includes both bulk Repeats found through *de novo* clustering and a curated set of families. The work is more heavily skewed towards model species include vertebrates (human, mouse, zebrafish), worm and fly. There are partial libraries for seven additional species. This resource has less utility for Fungi but could provide a reasonable repository for curation of fungal families in the future.

To run RepeatMasker the following script could be used: The scripts shown here are in TE_repeats github.

```
#SBATCH -N 1 -n 16 --out RepeatMasker.%A.log -J RepMsk

CPU=$SLURM_CPUS_ON_NODE # set the CPUS dynamicall for the job
if [ -z $CPU ]; then # unless this is not really a slurm job
 CPU=2 # set the number of CPUs to 2
fi
module load RepeatMasker  # note current version at time of this writing was 4-1-1 to load t
#module load RepeatMasker/4-1-1

# run with NCBI-RMBLAST
RepeatMasker -pa $CPU -e rmblast -species Fungi Genome.fa > Genome.RepeatMasker.log
```

To also generate the repeats as GFF format use the `-gff` option

```
RepeatMasker -pa $CPU -e rmblast -species Fungi -gff Genome.fa > Genome.RepeatMasker.log
```

I usually prefer to run a separate script to process the RepeatMasker output and convert that to GFF.

Developing a *de novo* Repeat library with RepeatModeler

```
#SBATCH -N 1 -n 16 --out RepeatMasker.%A.log -J RepMsk

CPU=$SLURM_CPUS_ON_NODE # set the CPUS dynamicall for the job
if [ -z $CPU ]; then # unless this is not really a slurm job
 CPU=2 # set the number of CPUs to 2
fi
module load RepeatModeler/2.0.1
SPECIES=AfumigatusAf293
if [ ! -f $SPECIES.translation ]; then # assumes using rmblast as default
  BuildDatabase -name $SPECIES $SPECIES.genome.fasta
fi
# this can take 2-48 hrs depending on genome size - you may need to set job running
RepeatModeler -database $SPECIES -LTRStruct -pa $CPU
```

Classifying *de novo* repeats

Masking Repeats with REPET pipeline

Identifying LTR Elements

Finding MITE elements with MITE_Hunter