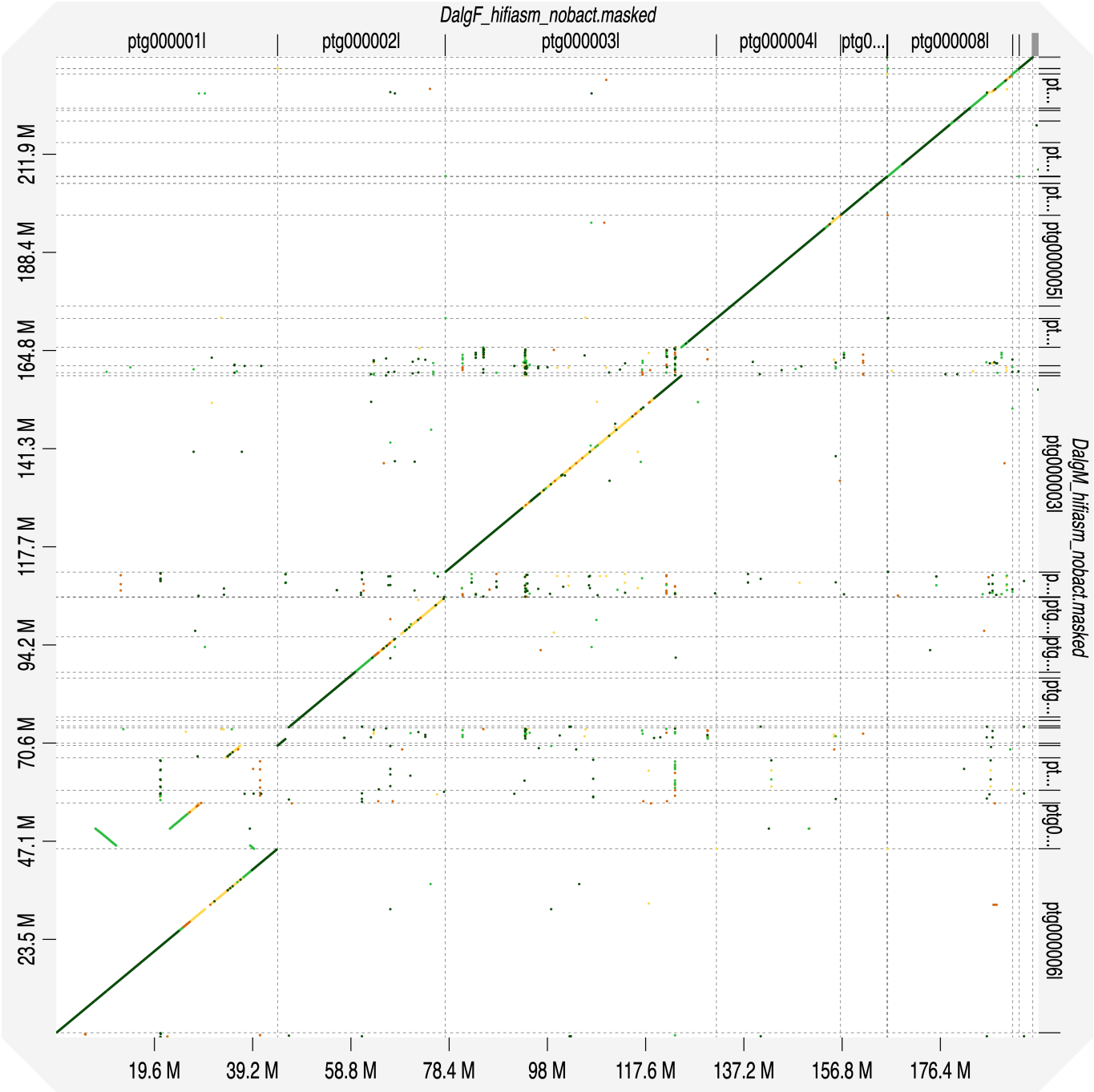
D. algonquin Strain: NH2

Assembly

```
#!/bin/bash
#SBATCH --job-name=hifiasm  # Job name
#SBATCH --partition=kucg      # Partition Name (Required)
#SBATCH --mail-type=END,FAIL,BEGIN     # Mail events (NONE, BEGIN, END,
FAIL, ALL)
#SBATCH --mail-user=tconway@ku.edu   # Where to send mail
#SBATCH --ntasks=1          # Run on a single CPU
#SBATCH --mem=100gb           # Job memory request
#SBATCH --time=7-24:00:00        # Time limit days-hrs:min:sec
#SBATCH --output=hifiasm_M%j.log  # Standard output and error log


module load conda
conda activate hifiasm

hifiasm -o /home/t043c581/scratch/data/Dalg_downsampled.asm -t 32
/home/t043c581/scratch/data/DalgM_hifi.downsampled.fastq
```



Remove bacterial contamination

```
blastn -task megablast -query foo.fa -remote -db nt -outfmt '6 qseqid
staxids bitscore std' -max_target_seqs 1 -max_hsps 1 -evalue 1e-25 -out
foo.megablast
```

- When this is finished, go through the file and remove bacteria/virus using NCBI

quast, busco, and other data I will undoubtedly be asked for...

```
# busco
nohup busco -i /hdd/Taylor/data/foo.fa -o /hdd/Taylor/data/BUSCO.foo -l
/hdd/Taylor/data/diptera_odb10 -m genome --auto-lineage-euk -f &
```

```
# quast
python3 ../software/quast-5.2.0/quast.py /hdd/Taylor/data/foo.fa
```

Species (sex)	Length	# of Scaffolds	N50	BUSCO (complete)	BUSCO (single)	BUSCO (dup)
D. algonquin(male)	~235Mb	42	10.9Mb	98.6%	91.1%	7.5%
D. algonquin(female)	~196Mb	34	44.1Mb	99.3%	98.4%	0.9%

Align fasta to reference via mummer to rename scaffolds locally

```
nucmer --maxgap=500 -mincluster=100 reference.fasta query.fasta
delta-filter -q -r out.delta > foo.filter
show-coords -B foo.filter > foo.coords
```

- Using the Rscript "chrom_mapping.R", check PID for each alignment and rename accordingly using:

```
sed 's/scaffold_to_be_renamed/rename_it_here/g' foo.fa >temp1
sed 's/scaffold_to_be_renamed/rename_it_here/g' temp1 >temp2
sed 's/scaffold_to_be_renamed/rename_it_here/g' temp2 >temp1

and so on...
```

Coverage

Method 1

Make .bam files with minimap and plot coverage using chromosome quotient method

Code:

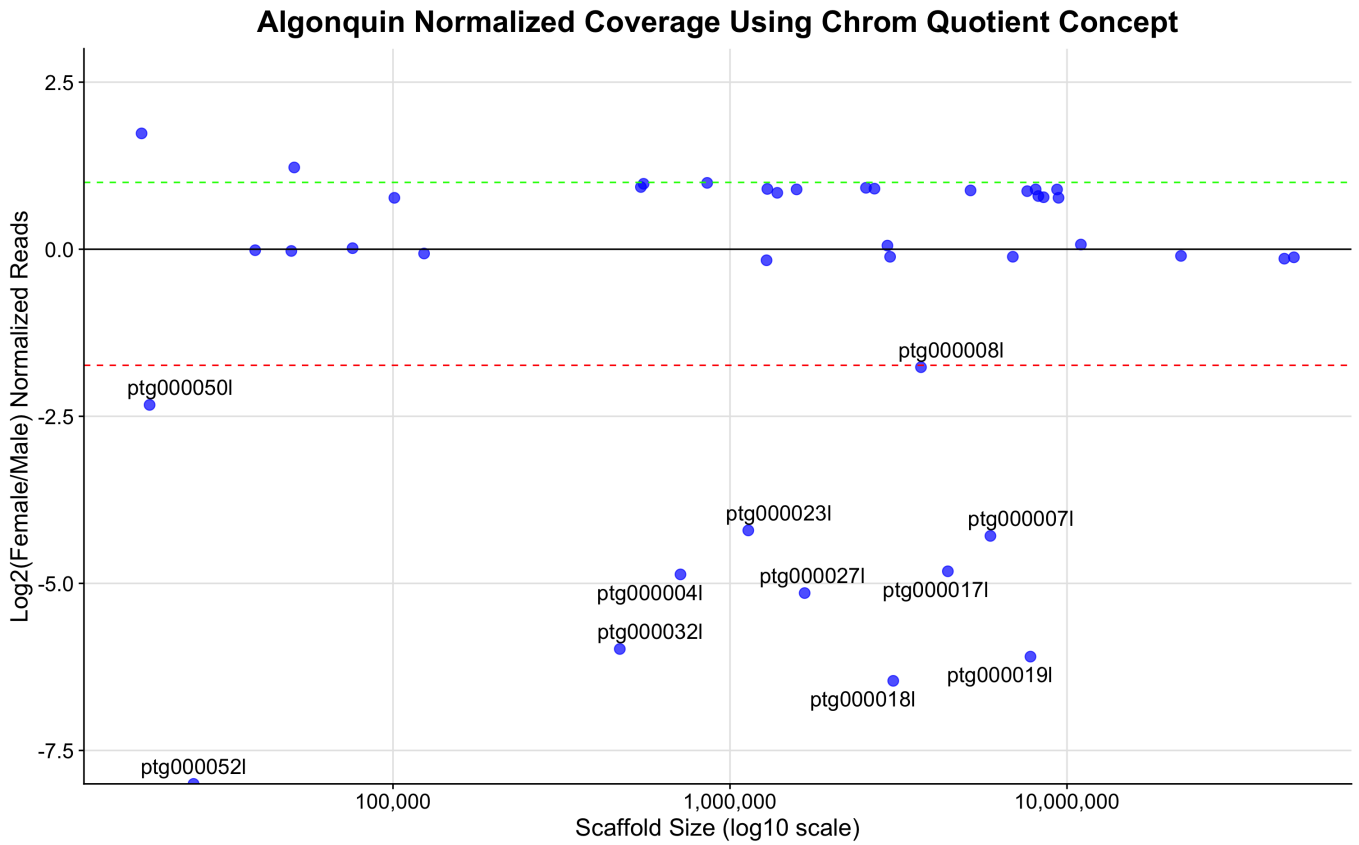
```
# on the linux
# align female longreads to male reference
minimap2 -t 8 -ax map-hifi /hdd/Taylor/data/DalgM_hifi_nobact_v2.fa
/hdd/Taylor/data/D-algonquin_F_HiFi.fastq.gz |samtools view -bS >
/hdd/Taylor/data/DalgF_hifi.bam

# align male longreads to male reference
minimap2 -t 8 -ax map-hifi /hdd/Taylor/data/DalgM_hifi_nobact_v2.fa
/hdd/Taylor/data/D-algonquin_M_HiFi.fastq.gz |samtools view -bS >
/hdd/Taylor/data/DaztM_hifi.bam
```

```
# sort bams and get coverage
samtools sort DalgM_hifi.bam >DalgM_hifi_sorted.bam
samtools coverage DalgM_hifi_sorted.bam >DalgM_hifi.cov
samtools sort DalgF_hifi.bam >DalgF_hifi_sorted.bam
samtools coverage DalgF_hifi_sorted.bam >DalgF_hifi.cov
```

Using RStudio:

```
# Load required libraries
library(ggplot2)
library(cowplot)
library(ggrepel)

# Load data
aztmale <-
read.delim("/Users/conway/Desktop/CurrentWorkingDatasets/DalgM_hifi.cov",
header = TRUE)
aztfem <-
read.delim("/Users/conway/Desktop/CurrentWorkingDatasets/DalgF_hifi.cov",
header = TRUE)

# Rename columns for clarity
colnames(algmale) <- c("scaffold", "startpos", "endpos", "numreads",
"covbases", "coverage", "meandepth", "meanbaseq", "meanmapq")
colnames(algfem) <- c("scaffold", "startpos", "endpos", "numreads",
"covbases", "coverage", "meandepth", "meanbaseq", "meanmapq")

# Normalize data
algmale$normalized <- algmale$numreads / sum(algmale$numreads)
algfem$normalized <- algfem$numreads / sum(algfem$numreads)

# Calculate scaffold sizes
algmale$size <- algmale$endpos - algmale$startpos
algfem$size <- algfem$endpos - algfem$startpos

# Compute log2 ratio of female/male normalized reads
algmale$log2fem_male <- log2(algfem$normalized / algmale$normalized)

# List of scaffolds to highlight
#highlight_scaffolds <- c("ptg000009l", "ptg000018l", "ptg000019l",
#                         "ptg000022l", "ptg000028l", "ptg000029l",
#                         "ptg000038l", "ptg000046l", "ptg000098l")

# Add a column to indicate if a scaffold should be highlighted
#algmale$highlight <- ifelse(algmale$scaffold %in% highlight_scaffolds,
"yes", "no")

ggplot(algmale[algmale$size > 10000, ], aes(x = size, y = log2fem_male)) +
  # Add points with conditional coloring
  geom_point(aes(color = highlight), size = 3, alpha = 0.7) +
  # Use a log10 scale for the x-axis
  scale_x_continuous(trans = "log10", labels = scales::comma_format()) +
```

```
  # Add horizontal reference lines
  geom_hline(yintercept = 0, color = "black", linetype = "solid") +
  geom_hline(yintercept = log2(0.3), color = "red", linetype = "dashed") +
  geom_hline(yintercept = log2(2), color = "green", linetype = "dashed") +
  # Label only points below the red line
  geom_text_repel(aes(label = ifelse(log2fem_male < log2(0.3), scaffold,
""")),
                  size = 5,
                  box.padding = 0.4,
                  point.padding = 0.4,
                  max.overlaps = Inf) +
  # Customize color scale
  scale_color_manual(values = c("no" = "blue", "yes" = "blue"), guide =
"none") +
  # Axis labels
  labs(
    x = "Scaffold Size (log10 scale)",
    y = "Log2(Female/Male) Normalized Reads",
    title = "Algonquin Normalized Coverage Using Chrom Quotient Concept"
  ) +
  # Set y-axis limits
  ylim(-7.5, 2.5) +
  # Apply clean theme
  theme_cowplot(font_size = 14) +
  theme(
    plot.title = element_text(size = 20, face = "bold", hjust = 0.5),  #
Center and bold title
    axis.title = element_text(size = 16),  # Larger axis titles
    axis.text = element_text(size = 14),   # Larger axis text
    panel.grid.major = element_line(color = "grey90", size = 0.5),  #
Subtle grid lines
    legend.position = "none"               # Remove legend
  )
```



Points aligning around 1 should be X-linked, and points aligning around 0 should be autosomal. Anything under the red line is putative Y-linked. This is because females have 2 Xs when males have 1, and males have 1 Y while females have zero. You expect (when log2 tranformed) for the autosomal reads to cancel out and end up around 0.

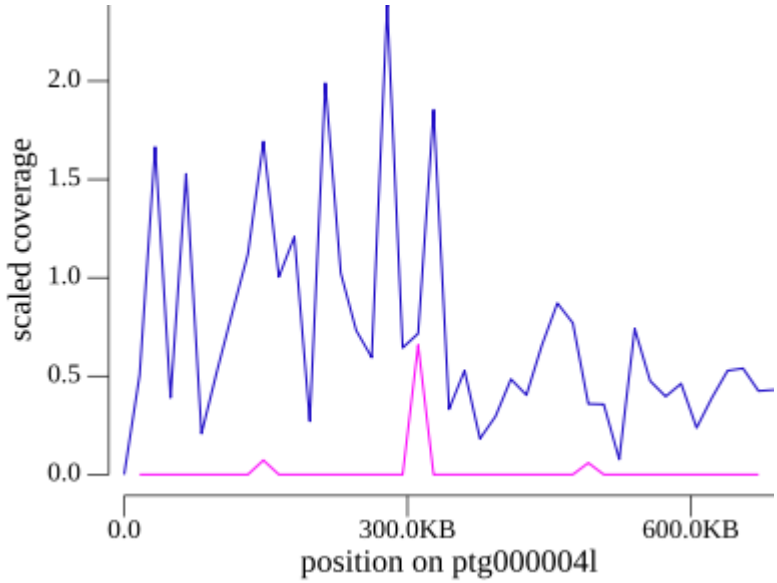
Method 2

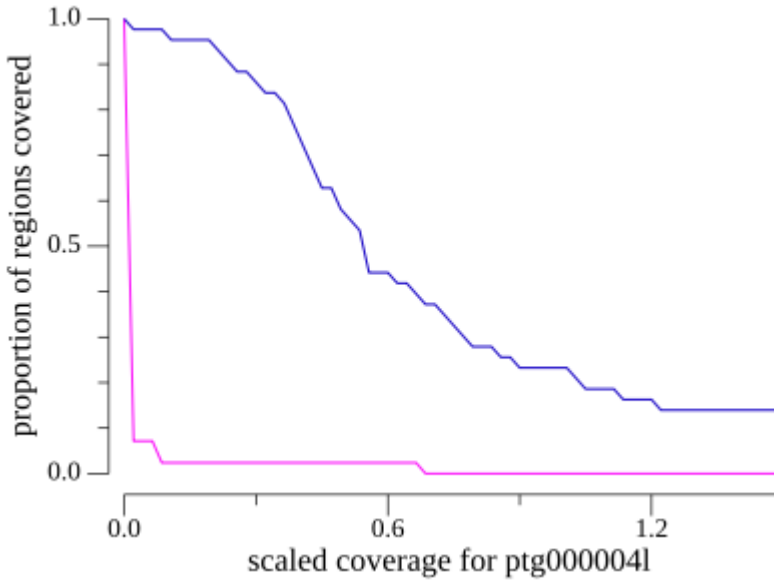
Indexcov

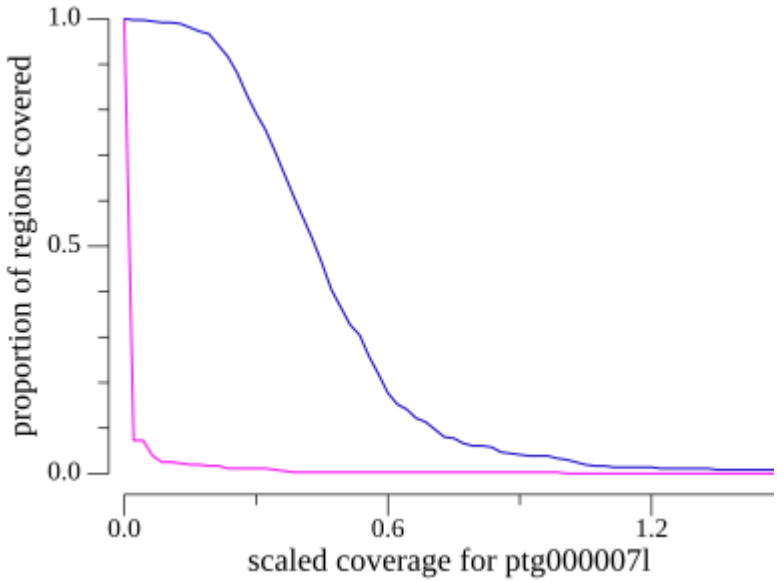
Code:

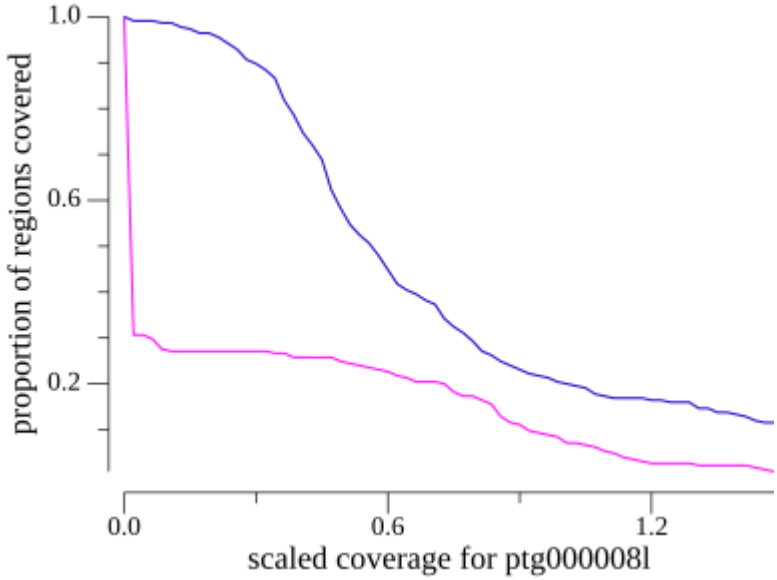
```
goleft indexcov --directory ../indexcov_Dalg *.bam
```

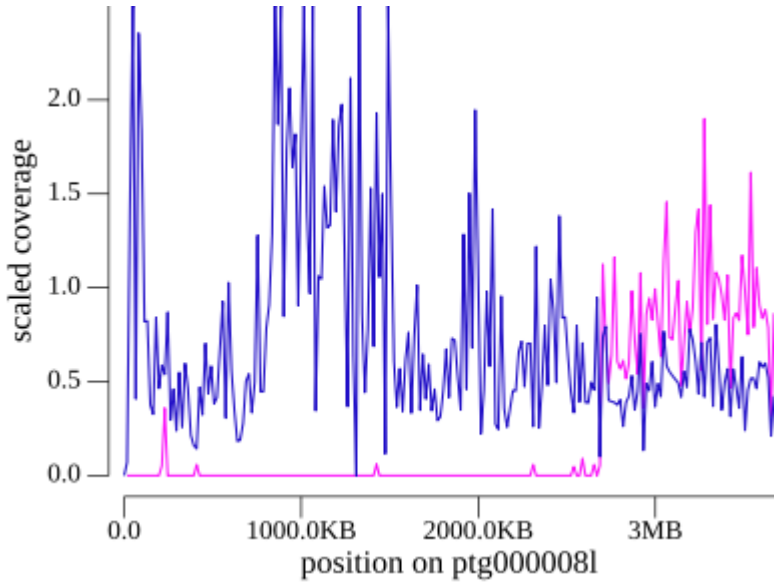
Putative Y scaffolds

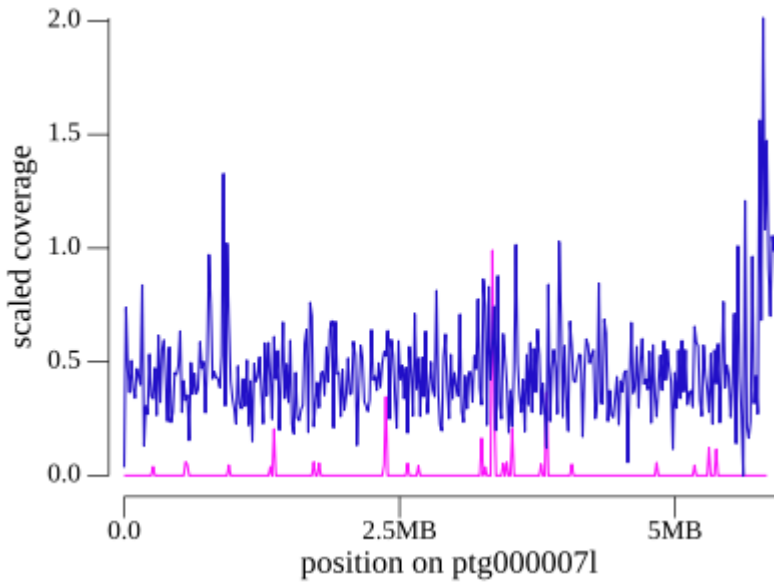

scaled coverage
2.0
1.5
1.0
0.5
0.0
0.0
300.0KB
600.0KB
position on ptg000004l

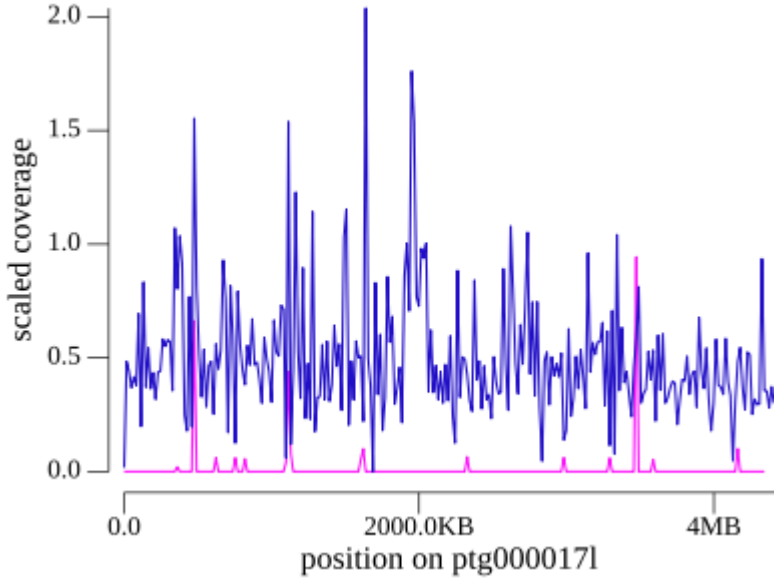

proportion of regions covered
1.0
0.5
0.0
0.0
0.6
1.2
scaled coverage for ptg000004l

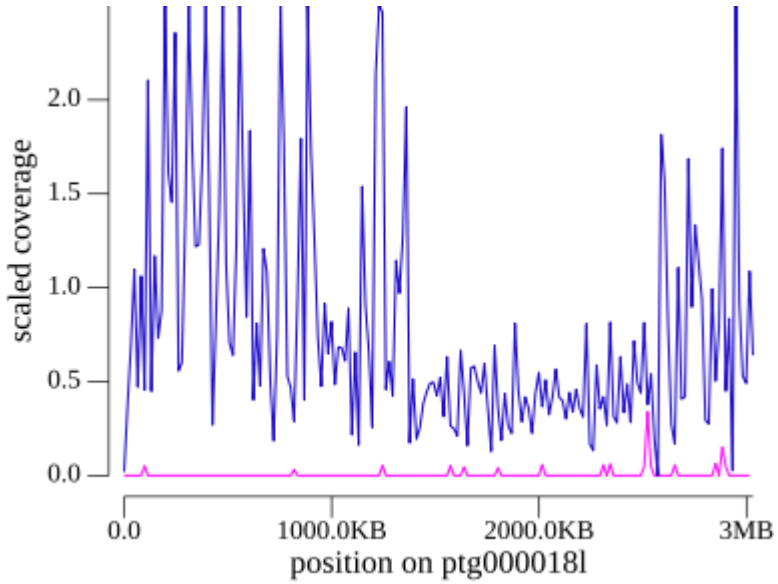

proportion of regions covered
1.0
0.5
0.0
0.0
0.6
1.2
scaled coverage for ptg000007l

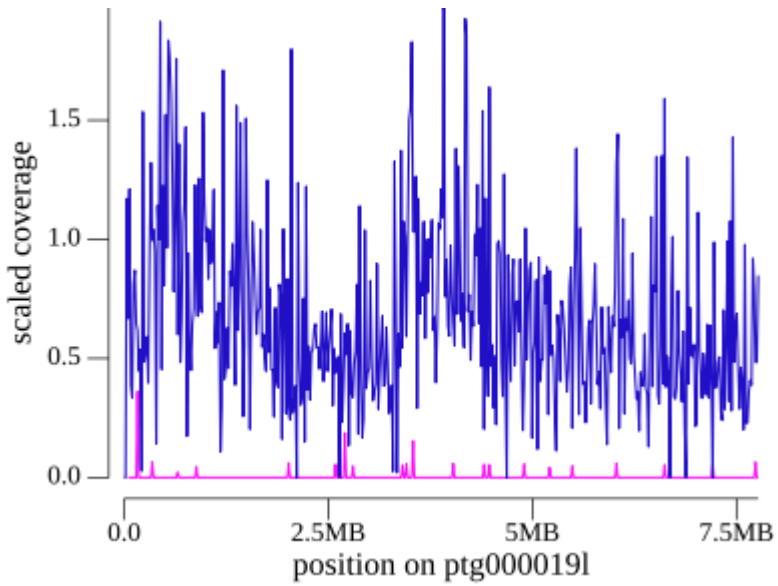

proportion of regions covered
1.0
0.6
0.2
0.0
0.6
1.2
scaled coverage for ptg000008l

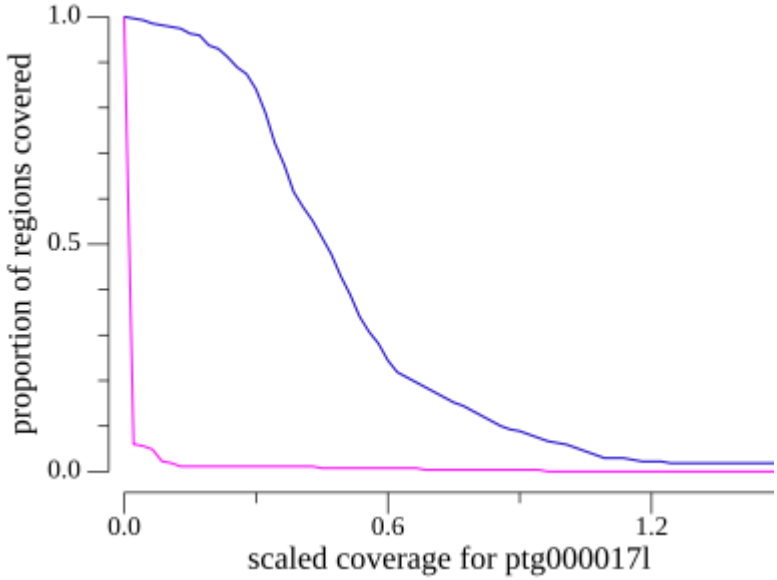

scaled coverage
2.0
1.5
1.0
0.5
0.0
0.0
1000.0KB
2000.0KB
3MB
position on ptg000008l

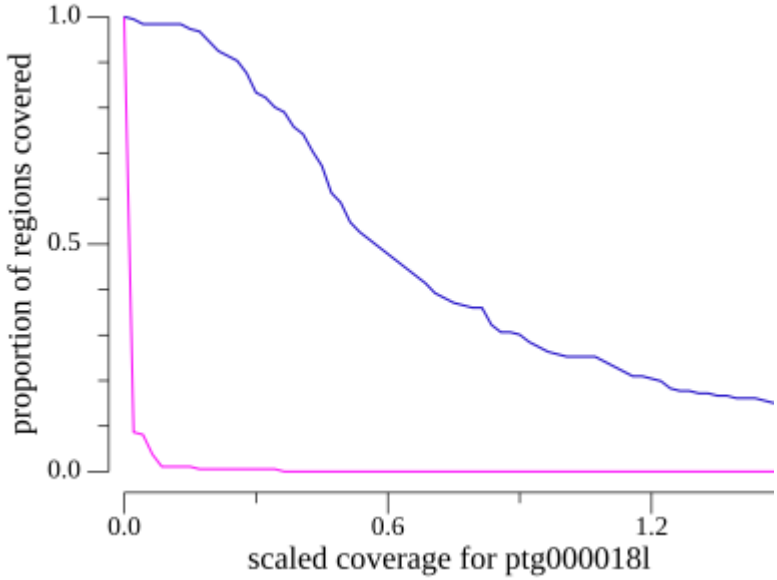

scaled coverage
2.0
1.5
1.0
0.5
0.0
0.0
2.5MB
5MB
position on ptg000007l

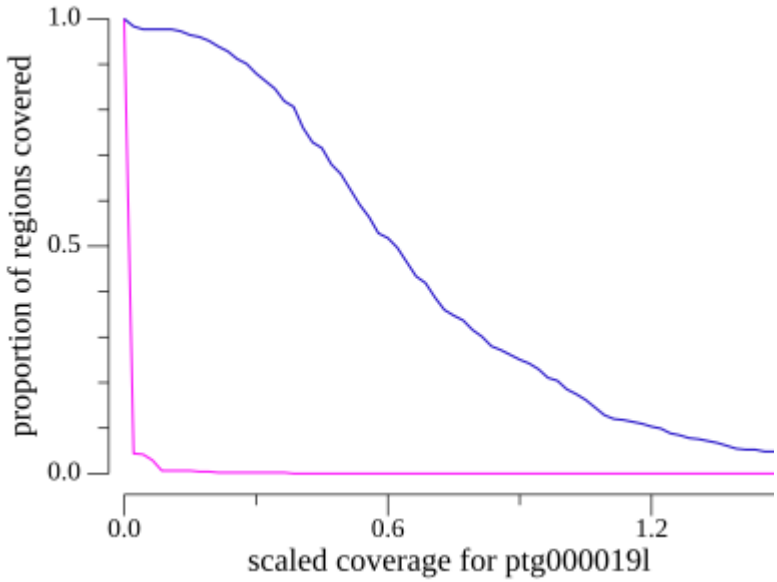

scaled coverage
2.0
1.5
1.0
0.5
0.0
0.0
2000.0KB
4MB
position on ptg000017l

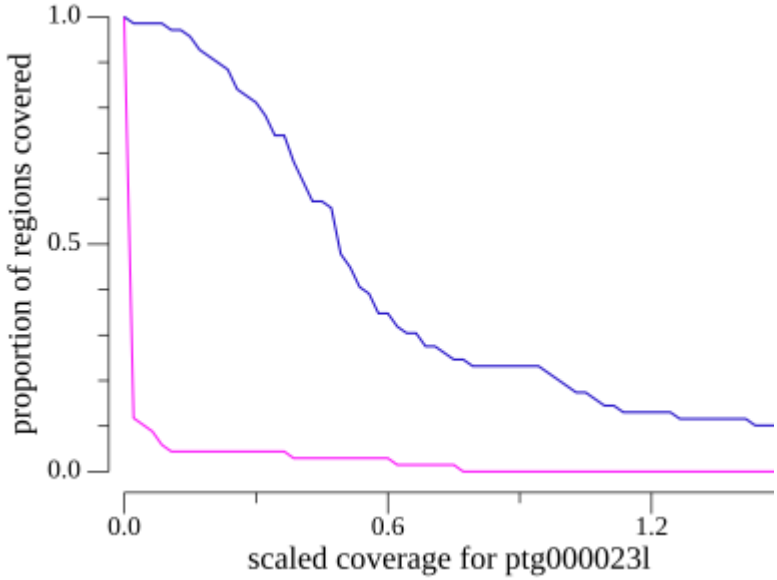

scaled coverage
2.0
1.5
1.0
0.5
0.0
0.0
1000.0KB
2000.0KB
3MB
position on ptg000018l

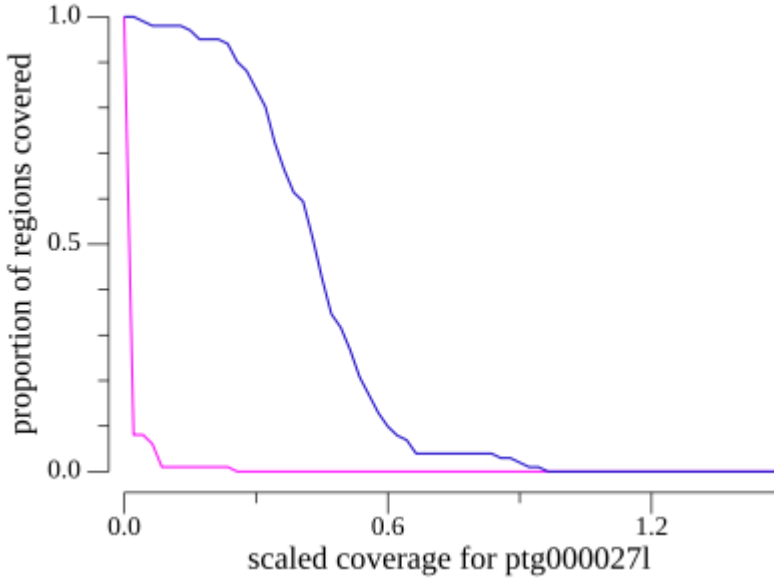

scaled coverage
1.5
1.0
0.5
0.0
0.0
2.5MB
5MB
7.5MB
position on ptg000019l

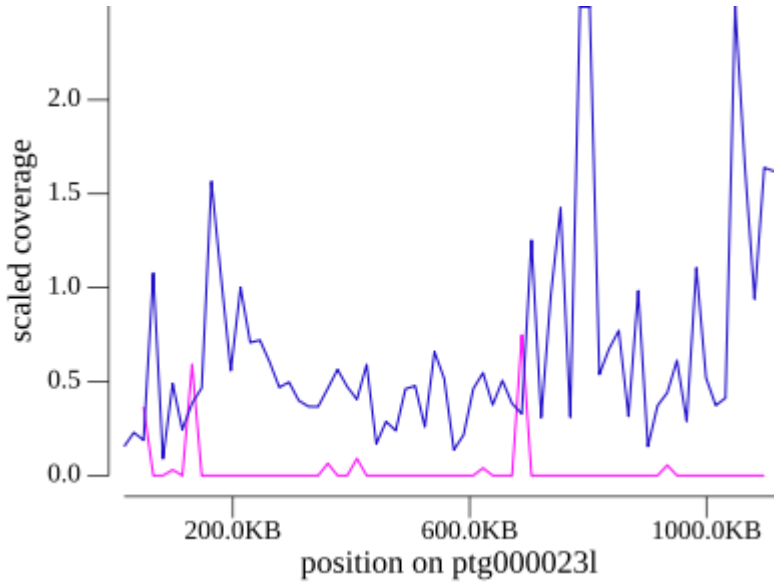

1.0
0.5
0.0
proportion of regions covered
0.0
0.6
1.2
scaled coverage for ptg000017l

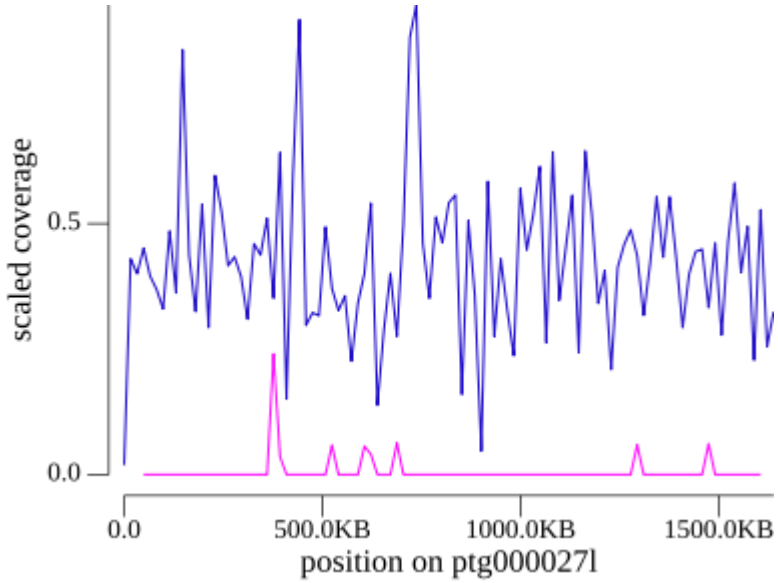

1.0
0.5
0.0
proportion of regions covered
0.0
0.6
1.2
scaled coverage for ptg000018l

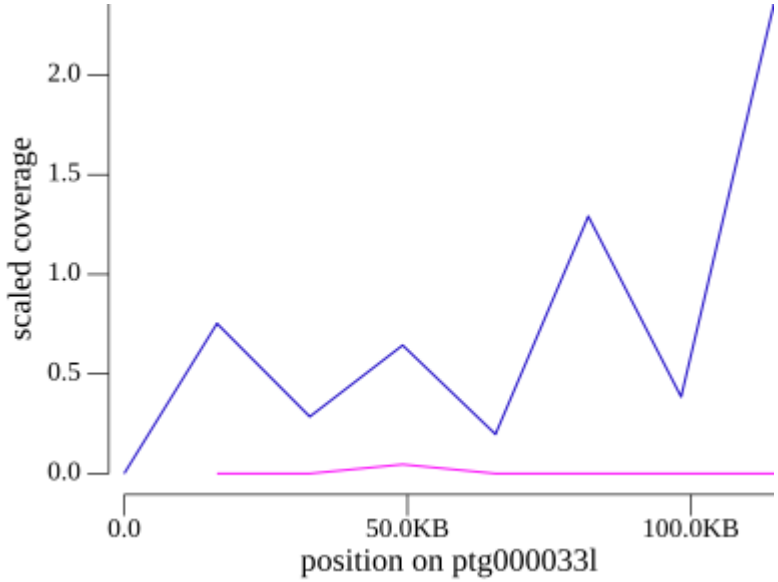

1.0
0.5
0.0
proportion of regions covered
0.0
0.6
1.2
scaled coverage for ptg000019l

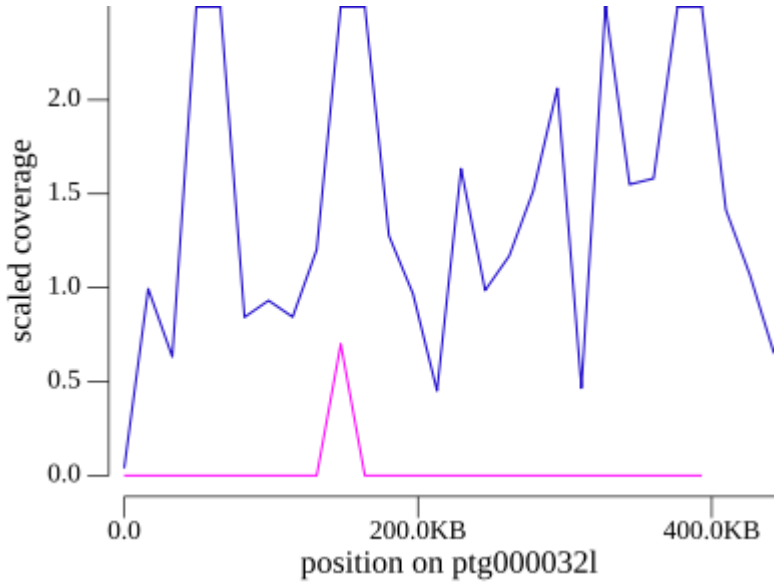

1.0
0.5
0.0
proportion of regions covered
0.0
0.6
1.2
scaled coverage for ptg000023l

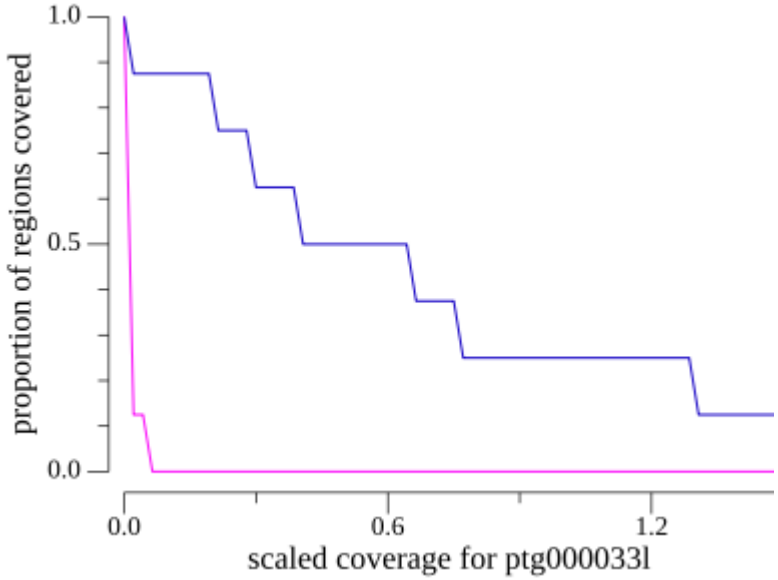

1.0
0.5
0.0
proportion of regions covered
0.0
0.6
1.2
scaled coverage for ptg000027l

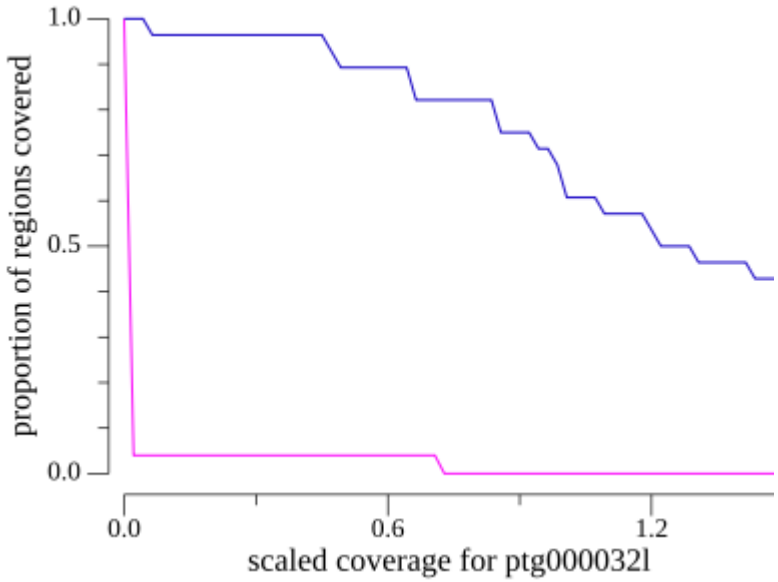

2.0
1.5
1.0
0.5
0.0
scaled coverage
200.0KB
600.0KB
1000.0KB
position on ptg000023l


scaled coverage
0.5
0.0
0.0
500.0KB
1000.0KB
1500.0KB
position on ptg000027l


scaled coverage
2.0
1.5
1.0
0.5
0.0
0.0
50.0KB
100.0KB
position on ptg000033l


scaled coverage
2.0
1.5
1.0
0.5
0.0
0.0
200.0KB
400.0KB
position on ptg000032l





Confirmation of putative Y scaffolds

Scaffold	Length	# of genes (unmasked)	# of genes (masked)	samtools coverage	indexcov	PCR
ptg000050l	18957			y	n	
ptg000004l	712952			y	y	
ptg000032l	470959			y	y	
ptg000023l	1133167			y	y	
ptg000027l	1665441			y	y	
ptg000008l	3687293			y	y	
ptg000017l	4430452			y	y	
ptg000007l	5926166			y	y	
ptg000018l	3050678			y	y	
ptg000019l	7791240			y	y	

Scaffold	Length	# of genes (unmasked)	# of genes (masked)	samtools coverage	indexcov	PCR
ptg000033l	143607			n	y	

Locate and mask repeats with repeatmodelor and repeatmasker on the cluster

```
#!/bin/bash
#SBATCH --job-name=RMaffM  # Job name
#SBATCH --partition=kucg      # Partition Name (Required)
#SBATCH --mail-type=END,FAIL,BEGIN     # Mail events (NONE, BEGIN, END,
FAIL, ALL)
#SBATCH --mail-user=tconway@ku.edu   # Where to send mail
#SBATCH --ntasks=8
#SBATCH --cpus-per-task=1          # Run on a single CPU
#SBATCH --mem=64gb           # Job memory request
#SBATCH --time=4-00:00:00        # Time limit days-hrs:min:sec
#SBATCH --output=RMaffM_%j.log  # Standard output and error log

module load repeatmodeler
module load repeatmasker/4.0.9

#usage: sbatch RepeatMasker.args.job <fasta> <prefix>

cd $SCRATCH
mkdir RMaffinisM_pilon2

echo "STARTING"
cd RMaffinisM_pilon2
cp $HOME/$1 .

BuildDatabase -name $2 -engine ncbi $1

RepeatModeler -engine ncbi -pa 8 -database $2

RepeatMasker -pa 8 -gff -lib $2-families.fa -dir MaskerOutput$2 $1

echo done
```

- With this data, you can look at Y-linked repeat families.

Annotate with helixer

- Go to https://www.plabipd.de/helixer_main.html
- Input fasta
- Change "Select Lineage-specific mode" to invertebrate
- Enter GFF label name and email address
- Submit job and wait
- grep gene foo.gff > genes.txt

- Import genes.txt into spreadsheet
- Convert gff to fasta using gffread (see below for code)
- blastx Y_transcripts.fa
- look up each gene on flybase and fill out spreadsheet

```
# gffread
gffread your_transcripts.gff -g genomic_reference.fasta -w
your_transcripts.fasta
```

Renaming Transcripts

Using Nilanjan's method

```
makeblastdb -in ../Dmel_translation_clean.fasta -dbtype prot -out
dmel_protein_database

blastx -query DalgM_masked_fixednames_transcripts.fa -db
dmel_protein_database -outfmt 6 -evalue 1e-5 -max_target_seqs 1 -
num_threads 4 -out blast_algM_transcripts.txt

cut -f2 blast_algM_transcripts.txt | sort | uniq >
algM_best_hit_proteins.list

seqtk subseq ../Dmel_translation_clean.fasta algM_best_hit_proteins.list >
algM_best_hit_proteins.fa

makeblastdb -in DalgM_masked_fixednames_transcripts.fa -dbtype nucl -out
DalgM_transcripts_db

tblastn -query algM_best_hit_proteins.fa -db DalgM_transcripts_db -outfmt
6 -evalue 1e-5 -max_target_seqs 1 -out algM_blast_reciprocal.txt

awk '{print $1"\t"$2}' algM_blast_reciprocal.txt > algM_forward_hits.txt

awk '{print $2"\t"$1}' algM_blast_reciprocal.txt >
algM_reciprocal_hits.txt

sort algM_forward_hits.txt algM_reciprocal_hits.txt | sed  's/-
P[ABCDEFGHIJKLMNOPQRSTUVWXYZ]//g'  | uniq > reciprocal_best_hits.txt

awk '{if(a[$2]++){print $1"\t"$2"."a[$2]}else{print $0}}'
reciprocal_best_hits.txt > algM_RBH.txt

awk '{print $0 ".1"}' algM_RBH.txt > algM_RBH2.txt

gawk 'NR==FNR { mapping[$1] = $2; next } { for (key in mapping) gsub(key,
mapping[key]) } 1' algM_RBH2.txt ../DalgM_masked_helixer.gff >
algM_temp.gff
```

```
gawk 'NR==FNR { mapping[$1] = $2; next } { for (key in mapping) gsub(key,
mapping[key]) } 1' algM_RBH.txt algM_temp.gff > algM_renamed.gff
```

Primers

ptg 27

alg_ARY_F - CTGCTTGACTACTTGCGATGA alg_ARY_R - AGGTAGTGCTTTAGTGAGTCAA

ptg 23

alg_ptg23_F - CAGAGTCTGTTCAGTCGAGTT alg_ptg23_R - GCAGTTTTCTGTCGACATGCA

alg_Bl1_F - GCTGGTACTGGGACTGCATTT alg_Bl1_R - TGCGATGCTTACACTCTCAGA

ptg 8

alg_mael_F - TTCCGCTCCTGATGGCACTG alg_mael_R - TCGCACAACAAGTCTTCTGGAAT