Map reads:

-Map reads to genome:

```
blasr subreadBamFiles.fofn dmel-all-chromosome-r6.12.fasta  --nproc 24 --bam --out dmel.bam  --clipping soft --bestn 1 --minPctIdentity 0.70 --unaligned dmel.unalign.fasta
```

-Parse reads into: 1. Mapped heterochromatin (non-major chromosome arms); 2. Unmapped and then merge to get set of heterochromatin-enriched reads

```
sort and index the bam file, extract reads from heterochromatic regions (given as Heterochromatin.contigs.list.txt), and add unmapped reads:
samtools view -@24 -L Heterochromatin.contigs.list.txt -b dmel.sort.bam | samtools fasta - > dmel_heterochromatin.fasta
cat dmel.unalign.fasta dmel_heterochromatin.fasta > mel_heterochromatin_enriched_reads.fasta
```

Make de novo assemblies:

-*De novo* assemble all reads using Canu and FALCON (see fc_run.cfg in supplementary methods)

-*De novo* assemble heterochromatic reads using Canu and FALCON (see fc_run.cfg in supplementary methods)

```
canu -pacbio-raw all_subreads.fastq -p canu_1.2 -d canu_1.2 genomeSize=160m useGrid=false errorRate=0.035
canu -pacbio-raw mel_heterochromatin_enriched_reads.fasta -p melY_extra -d melY_extra genomeSize=30m  stopOnReadQuality=false  corMinCoverage=0 corOutCoverage=100 ovlMerSize=31 useGrid=false
fc_run.py fc_run.cfg
```

Merge assemblies:

-Merge assemblies sequentially using quickmerge to create the final assembly (order of merging in Table S1)

-*e.g.* Quickmerge commands:

```
python merge_wrapper.py canu2.fasta falcon1.fasta
python merge_wrapper.py merged1.fasta falcon2.fasta
cat merged2.fasta major_arm_R6.fasta > merged3.fasta
python merge_wrapper.py merged3.fasta ISO_merged.fasta
python merge_wrapper.py merged4.fasta  canu1.fasta
```

Polish final assembly:

-Quiver: *e.g.*

```
referenceUploader -c -p./quiver_0530 -n ref -f ./dmel_final.fasta
smrtpipe.py --distribute -D NPROC=24 -D MAX_THREADS=24 --nohtml --params=params.xml xml:input.xml
cmph5tools.py sort --outFile aligned_reads.sorted.cmp.h5 --deep  aligned_reads.cmp.h5
samtools faidx ../ref/sequence/ref.fasta
quiver -v -j 24 aligned_reads.sorted.cmp.h5 -r ../ref/sequence/ref.fasta -o dmel.final_withY.quiver.fasta -o dmel.final_withY.quiver.gff
```

-Pilon x10: *e.g.* see pilon_mel.sh

```
sh pilon_mel.sh ref.fa unused_par1 unused_par2 24 final
```

Manually curate assembly:

-This step is not automated

-Collect external evidence for assemblies from physical or genetic maps (including cytology), PCR, Southern Blots etc.

-Break and correct misassemblies according to empirical evidence

Classify Y-linked contigs:

-Map male and female Illumina reads and filter on Q >10: e.g.

```
bwa mem -t 24 dmel_pilon_final.fasta Female_read_R1.fasta Female_read_R2.fasta |samtools view -@ 24 -bS - |samtools sort -@ 24 -o mel_f_new.bam -
bwa mem -t 24 dmel_pilon_final.fasta Male_read_R1.fasta Male_read_R2.fasta |samtools view -@ 24 -bS - |samtools sort -@ 24 -o mel_m_new.bam -
samtools index mel_f_new.bam
samtools index mel_m_new.bam
samtools depth -Q 10 mel_f_new.bam mel_m_new.bam >mel_new.out
```

-Call average and median female/male depth in 10kb window; see script frame_depth.pl

```
perl frame_depth.pl mel_new.out
```

Scaffold contigs:

-see script generate_scaffold.pl

-final.path states the order and number of N's between contigs

```
perl generate_scaffold.pl dmel_pilon_final.fasta final.path scaffold scaffold
```

Figure S1. Detailed assembly strategy. We show our assembly pipeline with command lines and pointers to code and configuration files in the supplement and/or on github.