The genome of a tardigrade -
Horizontal gene transfer or bacterial contamination?

Felix Bemm, Clemens Weiß, Jörg Schultz, Frank Förster

1 Figures

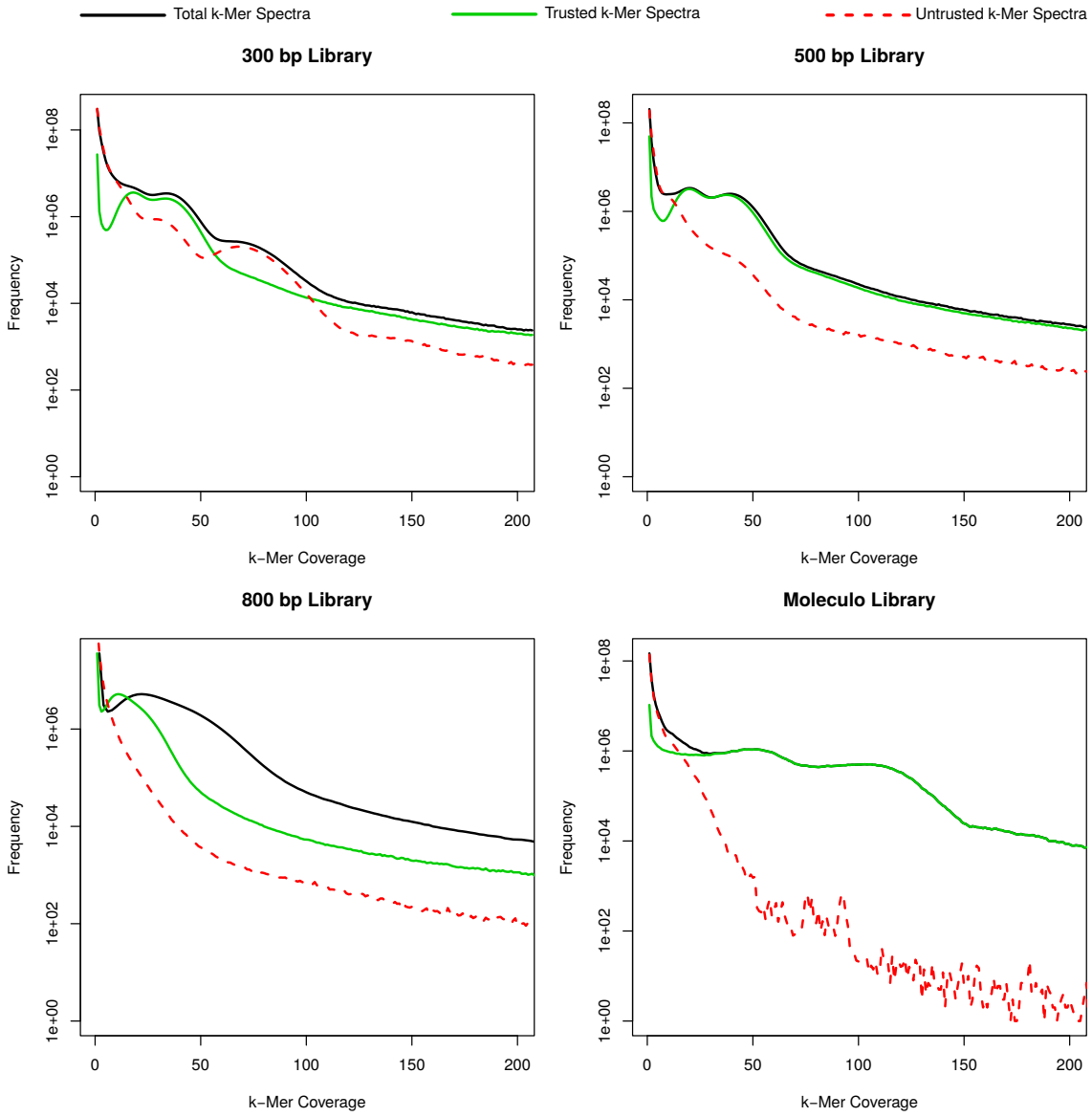

Figure 1: The plots depict the kmer distribution for each library before (black line) and after classification into "trusted" (green line) and "untrusted" kmers (red line).

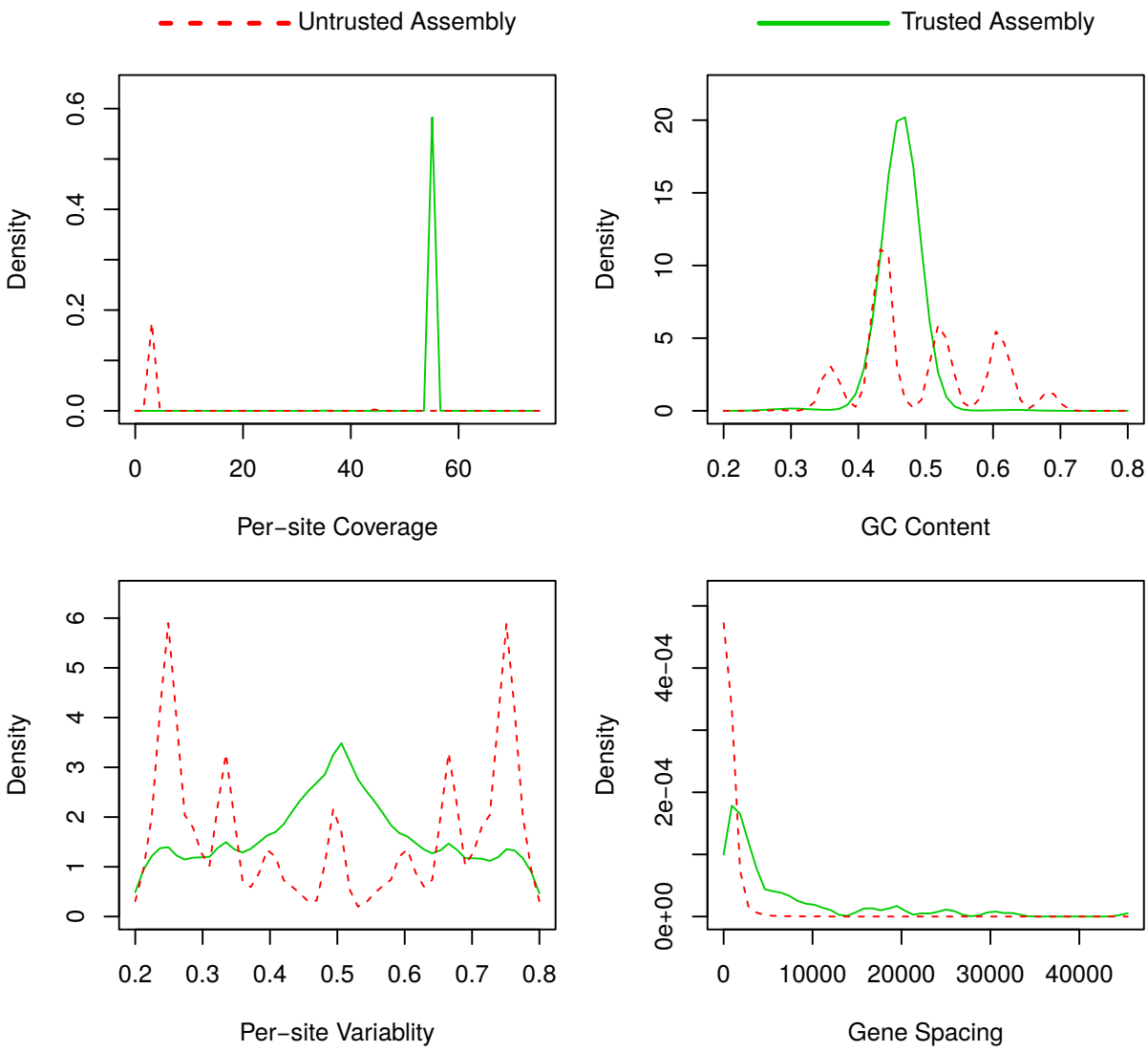

Figure 2: Assembly Feature Comparisons

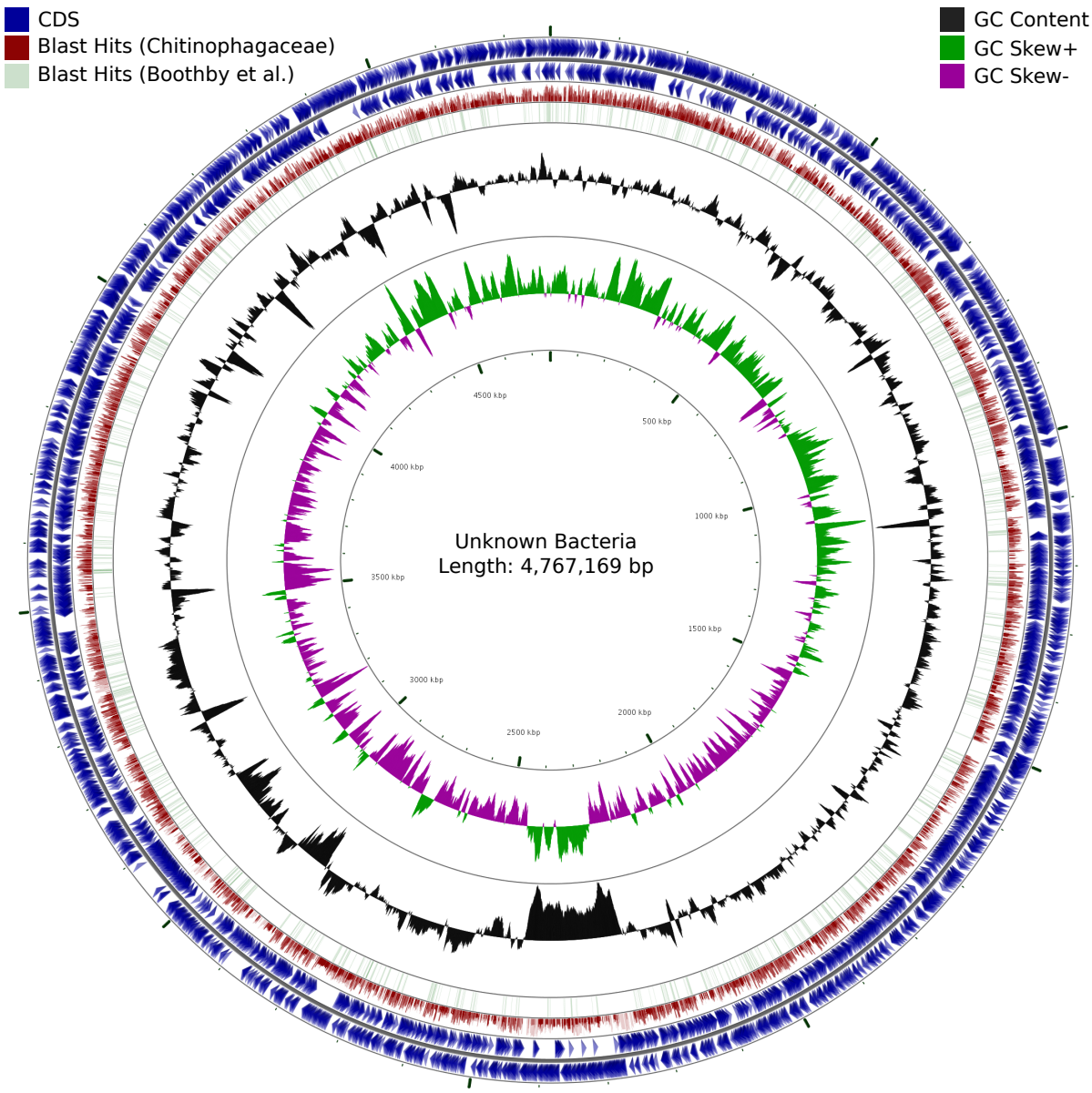

Figure 3: Unknown Bacterial Genome

2 Methods

GitHub repository

All script files are available from our GitHub repository (`https://github.com/greatfireball/hypsibius_genome_revised/`).

Data set

We used the data set provided by **Boothby2015** and downloaded the data from `http://weatherby.genetics.utah.edu/seq_transf/`. A complete list of the used input files are given in **??**.

Programs

Table 2: List of all programs including the version numbers and references to publications or websites used for the data processing and analysis

Programname	Version	Reference
Allpath-LG	v XXXX	**Gnerre2011**, **Ribeiro2012**
BEDTools	v 2.20.1	**Quinlan2010**
bioperl	v 1.69.1	**Stajich2002**
bowtie2	v 2.2.2	**Langmead2012**
bwa	v 0.7.10	**Li2009a**, **Li2010**
CGView	v XXXX	**Grin2011** ???
Falcon	v XXXX	`https://github.com/PacificBiosciences/falcon`
GenemarkS	v XXXX	**Besemer2001**
GenemarkET	v XXXX	**Lomsadze2014**
Jellyfish	v 2.2.4	**Marcais2011**
Perl	v 5.14.2	`https://www.perl.org/`
samtools	v 1.1	**Li2009b**, **Li2011a**, **Li2011b**

Trimming of the input data

Estimation of the genomes size

Counting and Filtering bases on kmers

The kmers of all libraries where counted using the software jellyfish [**Marcais2011**]:

```
# counting the kmers inside the trimmed sequence libraries
mkdir kmer
cd kmer
jellyfish count -t 32 -m 19 -s 30G -C \
   -o HD_gen.il_L300.trimmed_mer_19 \
   ../trimmed/HD_gen.il_L300.trimmed_P1.fastq \
   ../trimmed/HD_gen.il_L300.trimmed_P2.fastq
jellyfish count -t 32 -m 19 -s 30G -C \
   -o HD_gen.il_L500.trimmed_mer_19 \
   ../trimmed/HD_gen.il_L500.trimmed_P1.fastq \
   ../trimmed/HD_gen.il_L500.trimmed_P2.fastq
jellyfish count -t 32 -m 19 -s 30G -C \
   -o HD_gen.il_L800.trimmed_mer_19 \
   ../trimmed/HD_gen.il_L800.trimmed_P1.fastq \
   ../trimmed/HD_gen.il_L800.trimmed_P2.fastq
```

Table 1: Bla

Filename and location	Modification time	Size in Bytes	MD5 check sum	MD5 check sum decompressed
tg.genome.fsa.gz	2015-11-25T01:34:44Z	72,215,266	b8bd39390ef35dd43d1cda1ca6944d5a	77be374d28b91232c0810cc4d3cd37b9
tg.default.maker.proteins.final.fasta.gz	2015-12-02T23:43:44Z	12,359,873	2de12e5d28d6dba121973db2071565d9	1ad17cfa9e6c26e552fa8048c6ee90af
short_reads/TG-300-SIPE_1_sequence.txt	2015-11-30T21:48:51Z	11,526,955,725	c16b5442c9893b6feaa3aa81a39eefcd	c16b5442c9893b6feaa3aa81a39eefcd
short_reads/TG-300-SIPE_2_sequence.txt.gz	2015-11-30T21:52:41Z	3,920,224,257	3bea43d66d71926fb620966d281598c6	bc8423d4fe4275863e0809445ffd21ce
short_reads/TG-500-SIPE_1_sequence.txt.gz	2015-12-01T05:32:05Z	2,738,243,219	da8b15d388961938584343f8926f7b24	eee7363557ccb1fb0fa75ebe55ae7ee5
short_reads/TG-500-SIPE_2_sequence.txt.gz	2015-12-01T05:35:15Z	2,805,269,168	aa8c2c345484b9464d272e0993d6968b	325d74bbafd9b6019609e2fd33eca260
short_reads/TG-800-SIPE_1_sequence.txt.gz	2015-12-01T05:36:55Z	2,155,735,304	6e9cce1a27000ae2b4f87181a976df92	a85568ef53979c367870eee6390f2ced
short_reads/TG-800-SIPE_2_sequence.txt.gz	2015-12-01T05:37:46Z	2,058,207,374	ccf097cf4f13bb5cbc5a8e002250093d	4a4cc02c2f289d59c300810fb621eb28
moleculo_reads/LR6000049-DNA_A01-LRAAD-01_LongRead.fastq.gz	2015-11-30T17:50:17Z	825,877,986	86e75544f2d6ef5185bae419bbd2a4b2	bace73ed4750b33fc144e56c155454ab
moleculo_reads/LR6000049-DNA_A01-LRAAD-02_LongRead.fastq.gz	2015-11-30T17:51:34Z	835,283,315	4dea3e39a7a25059a6ebbd5588e845b2	cb83c39f9a385f0b4fd1e507cfe40ff1
moleculo_reads/LR6000049-DNA_A01-LRAAD-03_LongRead.fastq.gz	2015-11-30T17:52:51Z	847,867,943	16276b6ef8dea90721eb67ac21d616e6	51d4ce37668684b4aa25e061fb95b4ef
moleculo_reads/LR6000049-DNA_A01-LRAAD-04_LongRead.fastq.gz	2015-11-30T17:56:08Z	859,746,540	3364040445c7377c9323f82d98a2258c	dbe06ec4248199f416bb1d02ff1e65f5
moleculo_reads/LR6000049-DNA_A01-LRAAD-05_LongRead.fastq.gz	2015-11-30T17:56:51Z	854,266,597	7995559df803ef0de0250f1bfac71f1a	98d30f3ceb813d9f53c6df2ed1fa2239

```
for i in $(find ../trimmed/ -name HD_gen.mo_L[12345]*.trimmed.formatted.fastq)
do
    jellyfish count -t 32 -m 19 -s 30G -C -o $(basename "$i" .fastq)_mer_19 "$i"
done

# merging of the moleculo hashes
jellyfish merge \
   -o HD_gen.mo_L1-5.trimmed.formatted_mer_19 \
   HD_gen.mo_L[12345].trimmed.formatted_mer_19
```

The resulting kmer hashes need to be dumped and converted to a hash utilized later during the filtering step. This step and the following required > 200 GB of memory and was performed by the perl script `prepare_filter_fastq_by_valid_kmers.pl`.

```
cd kmer

# dumping the kmer hashes
for i in *_mer_19
do
   jellyfish dump --column --tab -o $(basename "$i").dump "$i"
done

# merging kmer hash information into a single perl hash
prepare_filter_fastq_by_valid_kmers.pl \
   --output kmer_hash.bin \
   --kmerlib 300=HD_gen.il_L300.trimmed_mer_19.dump \
   --kmerlib 500=HD_gen.il_L500.trimmed_mer_19.dump \
   --kmerlib 800=HD_gen.il_L800.trimmed_mer_19.dump \
   --kmerlib Moleculo=HD_gen.mo_L1-5.trimmed.formatted_mer_19.dump
```

The generated hash was used to filter individual libraries by the perl script `filter_fastq_by_valid_k`

```
mkdir kmer_filtered
cd kmer_filtered
../scripts/filter_fastq_by_valid_kmers_reduced.pl \
   --infile ../trimmed/HD_gen.il_L300.trimmed_P1.fastq,../trimmed/HD_gen.il_L300.tr
   --kmerhash ../kmer/kmers_hash.bin \
   --out HD_gen.il_L300.trimmed_P12.fastq.interleaved.filtered \
   --paired

../scripts/filter_fastq_by_valid_kmers_reduced.pl \
   --infile ../trimmed/HD_gen.il_L500.trimmed_P1.fastq,../trimmed/HD_gen.il_L500.tr
   --kmerhash ../kmer/kmers_hash.bin \
   --out HD_gen.il_L500.trimmed_P12.fastq.interleaved.filtered \
   --paired

../scripts/filter_fastq_by_valid_kmers_reduced.pl \
   --infile ../trimmed/HD_gen.il_L800.trimmed_P1.fastq,../trimmed/HD_gen.il_L800.tr
   --kmerhash ../kmer/kmers_hash.bin \
   --out HD_gen.il_L800.trimmed_P12.fastq.interleaved.filtered \
   --paired

for i in $(find ../trimmed/ -name HD_gen.mo_L[12345]*.trimmed.formatted.fastq)
do
```

```
   ../scripts/filter_fastq_by_valid_kmers_reduced.pl \
      --infile "$i" \
      --kmerhash ../kmer/kmers_hash.bin \
      --out $(basename "$i").filtered
done
```

The filtered data sets are classified as "trusted" or "untrusted" based on the "trusted" kmer content. Reads with at least 95 % "trusted" kmers content are called "trusted" while reads below that threshold are classified as "untrusted".

```
pv HD_gen.il_L300.trimmed_P12.interleave.kmerfiltered.fastq | \
   perl -ne '
      unless (/percent_valid:([\d.]+)/) { die "Fehler"; }

      if ($1 < 0.95) {
        print STDERR $_, scalar <>,
                     scalar <>, scalar <>,
                     scalar <>, scalar <>,
                     scalar <>, scalar <>;
      } else {
        print $_,scalar <>,
              scalar <>, scalar <>,
              scalar <>, scalar <>,
              scalar <>, scalar <>;
      }' 2> HD_gen.il_L300.trimmed_P12.interleave.kmerfiltered.untrusted.fastq \
          > HD_gen.il_L300.trimmed_P12.interleave.kmerfiltered.trusted.fastq

pv HD_gen.il_L500.trimmed_P12.interleave.kmerfiltered.fastq | \
   perl -ne '
      unless (/percent_valid:([\d.]+)/) { die "Fehler"; }

      if ($1 < 0.95) {
        print STDERR $_, scalar <>,
                     scalar <>, scalar <>,
                     scalar <>, scalar <>,
                     scalar <>, scalar <>;
      } else {
        print $_,scalar <>,
              scalar <>, scalar <>,
              scalar <>, scalar <>,
              scalar <>, scalar <>;
      }' 2> HD_gen.il_L500.trimmed_P12.interleave.kmerfiltered.untrusted.fastq \
          > HD_gen.il_L500.trimmed_P12.interleave.kmerfiltered.trusted.fastq

pv HD_gen.il_L800.trimmed_P12.interleave.kmerfiltered.fastq | \
   perl -ne '
      unless (/percent_valid:([\d.]+)/) { die "Fehler"; }

      if ($1 < 0.95) {
        print STDERR $_, scalar <>,
                     scalar <>, scalar <>,
                     scalar <>, scalar <>,
                     scalar <>, scalar <>;
```

```
    } else {
      print $_,scalar <>,
            scalar <>, scalar <>,
            scalar <>, scalar <>,
            scalar <>, scalar <>;
    }' 2> HD_gen.il_L800.trimmed_P12.interleave.kmerfiltered.untrusted.fastq \
        > HD_gen.il_L800.trimmed_P12.interleave.kmerfiltered.trusted.fastq
```

Long Read Assembly

Falcon

Assembly Annotation

GeneMark-S and GeneMark-ES CGView -> Visualizatin

Assembly Comparison

GC content The GC content was determined for all contigs ≥ 1 kbp using a sliding window of 1 kbp and a stepsize of 100 bp by the perl script `sliding_window_gc.pl`.

```
mkdir cg
cd cg

for i in ../assemblies/HD*.fasta
do
   ../scripts/sliding_window_gc/sliding_window_gc.pl
      --in "$i" \
      --min-length 1000
      > $(basename "$i").sliding_gc.tsv
done
```

Mapping Coverage The mapping coverage was determinded by remapping of the short or longreads onto the assembled contig. For the short read libraries, we used `bowtie2` as mapper. Long read libraries were mapped by `bwa`. The per-base coverage was determined by bedtools.

```
mkdir mapping
cd mapping

BWA=bwa
SAMTOOLS=samtools

ln -s ../assemblies/HD*.fasta ./

# prepare mapping indices for bowtie2 and bwa
for i in *.fasta
do
   # bowtie2 preparation
```

```
   bowtie2-build "$i" \
      $(basename "$i" .fasta) 2>&1 | \
      tee bowtie2-build-$(basename "$i" .fasta).log
   # bwa preparation
   bwa index "$i" 2>&1 | \
      tee bwa-index-$(basename "$i" .fasta).log
done

# mapping of short reads
for REF in HD_gen.supported.fasta HD_gen.unsupported.fasta
do
   # 300 bp library
   bowtie2 \
        -x "$REF" \
        -1 ../trimmed/HD_gen.il_L300.trimmed_P1.fastq \
        -2 ../trimmed/HD_gen.il_L300.trimmed_P2.fastq \
        -p 32 \
        --minins 0 \
        --maxins 900 | \
        samtools view -uS - | \
        samtools sort -@32 - "$REF"-il.L300

   # 500 bp library
   bowtie2 \
        -x "$REF" \
        -1 ../trimmed/HD_gen.il_L500.trimmed_P1.fastq \
        -2 ../trimmed/HD_gen.il_L500.trimmed_P2.fastq \
        -p 32 \
        --minins 0 \
        --maxins 1500 | \
        samtools view -uS - | \
        samtools sort -@32 - "$REF"-il.L500

   # 800 bp library
   bowtie2 \
        -x "$REF" \
        -1 ../trimmed/HD_gen.il_L800.trimmed_P1.fastq \
        -2 ../trimmed/HD_gen.il_L800.trimmed_P2.fastq \
        -p 32 \
        --minins 0 \
        --maxins 2400 | \
        samtools view -uS - | \
        samtools sort -@32 - "$REF"-il.L800
done

# mapping of long reads
# combine all long reads
find ../trimmed/ -name "HD_gen.mo_L[12345].trimmed.formatted.fastq" | \
   xargs cat > ../trimmed/HD_gen.mo_L12345.trimmed.formatted.fastq

# map the longreads
for REF in HD_gen.supported.fasta HD_gen.unsupported.fasta
```

```
do
   for SEQ in ../trimmed/HD_gen.mo_L12345.trimmed.formatted.fastq
      OUT=$(basename "$REF")_$(basename "$SEQ")

      $BWA mem -t 32 "$REF" "$SEQ" | \
      $SAMTOOLS view -uS - | \
      $SAMTOOLS sort -@32 - "$OUT"
   done
done

# extraction of per base coverage
for REF in HD_gen.supported.fasta HD_gen.unsupported.fasta
do
   for BAM in "$REF"*.bam
   do
      OUT=$(basename "$BAM" .bam).cov
      bedtools genomecov \
         -ibam "$BAM" -d -g "$REF" > "$OUT"
   done
done
```

Per-site Variability ? sm-Packages to compare distributions