Genome Assembly Workshop

Deb Triant
University of Virginia

Programming for Biology
Cold Spring Harbor Labs
October 2024

Genome Assembly Workshop

1. Genome assembly with Canu

2. Python problem set

Genome Assembly Workshop

Genome assembly with PacBio data using Canu

Why are we using this program?

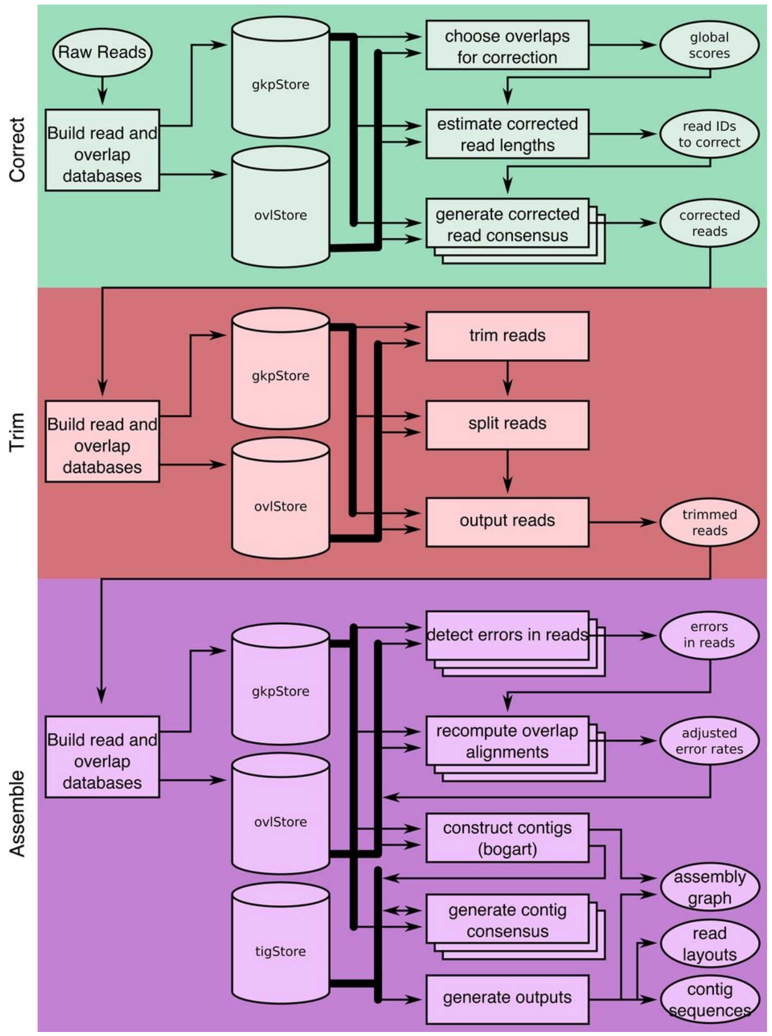
- *Relatively* user-friendly and easy to install
- Not computationally intensive with small data sets
- Well maintained and documented
  - version2.2, released August 2021

Canu assembler

- Canu is a long-read assembler that specializes in PacBio or Oxford Nanopore data
- Derived from Celera assembler
- Corrects reads, trims suspect regions (e.g., adaptors) then assembles the corrected reads
- Can be run with one command to do all steps or each step can be run separately (correcting, trimming and assembling).

A full Canu run includes three stages: correction (green), trimming (red), and assembly (purple).

Generates k-mer histogram and conducts all-vs-all overlaps



Correct:
Select best overlaps to generate corrected reads. Uses longest 40X reads for correction

Trim:
Identifies unsupported regions and trims reads to longest supported range

Assemble:
Identify sequencing errors, constructs best overlap graph and outputs contigs.

Sergey Koren et al. Genome Res. 2017;27:722-736

Using Canu

Outputs contigs. No scaffolding.

The unitig construction task finds sets of overlaps that are consistent, and uses those to place reads into a multialignment layout. The layout is then used to generate a consensus sequence for the unitig or a "high-confidence contig"

For eukaryotic genomes, coverage more than 20x sufficient but 30-60x is recommended minimum. Less complex genomes, can work down to 20X coverage.

More coverage, more long reads for Canu to assemble resulting in better assemblies.

Canu Citations

- Koren S, Walenz BP, Berlin K, Miller JR, Phillippy AM. Canu: scalable and accurate long-read assembly via adaptive k-mer weighting and repeat separation Genome Research. (2017).

- Koren S, Rhie A, Walenz BP, Dilthey AT, Bickhart DM, Kingan SB, Hiendleder S, Williams JL, Smith TPL, Phillippy AM. De Novo assembly of haplotype-resolved genomes with trio binning. Nature Biotechnology. (2018).

- Nurk S, Walenz BP, Rhiea A, Vollger MR, Logsdon GA, Grothe R, Miga KH, Eichler EE, Phillippy AM, Koren S. HiCanu: accurate assembly of segmental duplications, satellites, and allelic variants from high-fidelity long reads. Genome Research. (2020).

Installing Canu

- We will use the sample Pac Bio *E. coli* data found on the canu homepage: http://canu.readthedocs.io

  25x *E.coli* fastq files with quality scores.

Found on canu quickstart tutorial:

https://canu.readthedocs.io/en/latest/quick-start.html#quickstart

Installing Canu

Getting Help

/canu/documentation/source - help docs

$ `canu`   command line options

$ `canu -options`   parameter options

https://canu.readthedocs.io/en/latest/

Installing Canu

To install Canu: package managers not recommended!

Download current release - Canu v2.2

- released August 2021

https://github.com/marbl/canu/releases

- click on green "Latest" tab

```
$ wget https://github.com/marbl/canu/releases/download/
v2.2/canu-2.2.Darwin-amd64.tar.xz
```

#make sure file downloaded 100%!

```
$ tar -xJf canu-2.2.DArwin-amd64.tar.xz pacbio.fastq

    -x extract
     -J specific to .xz files
     -f file
```

Installing Canu

Where is it installed?

- Executables found in:

  canu-2.2./   **README File

- Add the Canu directory to path:
  - export PATH=/Users/student/canu-2.2/bin:$PATH

    ** specific to YOUR pathway!

- You can run the assembler with:
  - ./canu-2.2/bin/canu
- From the command line: canu

Canu Sample Data

Download canu test E.coli dataset:

http://canu.readthedocs.io/en/latest/quick-start.html?highlight=25x

- 25X subset (223Mb)

Where do you want them to download?

-
  ```
  wget -O pacbio.fastq
  https://obj.umiacs.umd.edu/marbl_publications/m
  hap/raw/ecoli_p6_25x.filtered.fastq
  ```

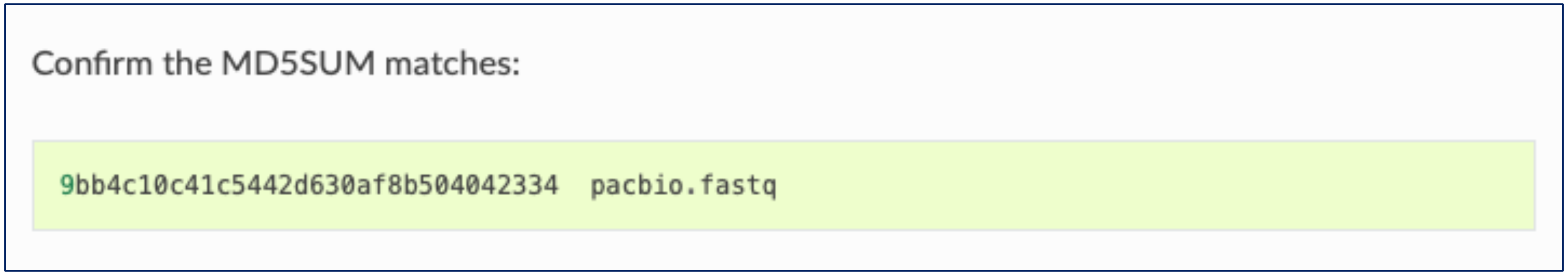
  - O naming downloaded file pacbio.fastq

Canu Sample Data

Check file integrity after downloading

- `md5`

```
>conda install -c conda-forge cms-md5
> md5 pacbio.fastq
```



Confirm the MD5SUM matches:

```
9bb4c10c41c5442d630af8b504042334  pacbio.fastq
```

How many reads within the file?

How many lines?

https://canu.readthedocs.io/en/latest/quick-start.html

FASTQ

```
@m141013_011508_sherri_c100709962550000001823135904221533_s1_p0/5451
9/26233_32397
TCGATCGAGTAACTCGCTGCTGTCAGACTGGTTTTGGTCGATCGACTATTGTTTCAGTCGCAAGAAT
ATTGTGTCCAGTCGATCGACTGAATTCTGCTGTACGGCCACGGCGGATGCACGGTACAGCAGGCTCAG
ACGGATTAAACTGTT
+
5=9=9<=9,-5@<<55>,6+8AC>EE.88AE9CDD7>+7.CC9CD+++5@=-FCCA@EF@+**+*--
55--AA---AA-5A<9C+3+<9)4++=E=+===<D94)00=9)))2@624(/(/2/-
(.(6;9(((((.(.'((6-66<6(///
@m141013_011508_sherri_c100709962550000001823135904221533_s1_p0/5453
2/2817_4395
GTAAAATTGAGGTAAATTGTGCGGAATTTAGCAATACCGTTTTTTTATTATCACCGGATATCTATTC
TGCTGTACGGCCAAGGAGGATGTACGGTACAGCAGGTGCGAACTCACTCCGACGCTCAAGTCAGTGAC
TTAATGATAAGCGTG
+
?????<BBBBBB5<?BFFFFFFECHEFFECCFF?9AAC>7@FHHHHHHFG?EAFGF@EEDEHHDGHHC
BDFFGDFHF)<CCD@F,+3=CFBDFHBD++??DBDEEEDE:):CBEEEBCE68>?))5?**0?:AE*A
*0//:/*:*:**.0)
@m141013_011508_sherri_c100709962550000001823135904221533_s1_p0/5455
1/25910_41116
GCTAGTCTTGTGTTTAGTTTTATGTTTTGCATGTTGTAACGGATTCATAAACATAGGTGTTTGTTTCT
TTTTATGGTTGTACAATTTGGCCCTAAGGCCCTACACTTACTTGTTTGTTTCTTTTATGGTACGACAT
TTGAGTGGTGGTTGA
+
```

Running Canu

- We will run default with correcting, trimming and assembling all in one command:

```
$ canu -p ecoli -d ecoli-pacbio  genomeSize=4.8m  -pacbio-raw
pacbio.fastq saveReads=true > CanuRun_20241023.out \
2>CanuRun_20241023.log  & **What are we saving?
                          **What does the "&" do?

*where are reads....include pathway in your command
-p assembly-prefix
-d output directory
input types:
 -pacbio-raw -pacbio-corrected -nanopore-raw -nanopore-corrected fastq or
fasta format


\ = line continuation – don't need if you have room to write it out!!!
```

****USE THE HELP DOCS!!!!**

Running Canu - Manual assembly

Correct, Trim and Assemble Manually

- You can do the three top-level tasks by hand. This would allow trying multiple construction parameters on the same set of corrected and trimmed reads.

Previous command to correct, trim & assemble:

```
canu  -p ecoli -d ecoli-full  genomeSize=4.8m  -pacbio-raw
pacbio.fastq saveReads=true 2>full_out_20241023.log **to save
output
```

Running Canu - Manual assembly

First correct the raw reads:

```
canu  -correct -p ecoli -d ecoli-manual  genomeSize=4.8m
 -pacbio-raw your_path/canu/pacbio.fastq
 saveReads=true 2>correct_out_20221017.log
```

Then trim the output of the correction:

```
canu -trim -p ecoli -d ecoli-manual  genomeSize=4.8m
-pacbio-corrected ecoli-manual/ecoli.correctedReads.fasta.gz
saveReads=true 2>trim_out_20241023.log
```

*corrected reads created during first step

Running Canu - Manual assembly

Finally, assemble the output of the trimming twice using two error rates (can use as many as you like):

```
canu -assemble -p ecoli -d ecoli-erate-0.013 genomeSize=4.8m
correctedErrorRate=0.013 -pacbio-corrected ecoli-
manual/ecoli.trimmedReads.fasta.gz saveReads=true
2>assemble-0.013_out_20241023.log
```

```
canu -assemble -p ecoli -d ecoli-erate-0.025 genomeSize=4.8m
correctedErrorRate=0.025 -pacbio-corrected ecoli-
manual/ecoli.trimmedReads.fasta.gz saveReads=true
gnuplotImageformat=svg  2>assemble-0.025_out_20241023.log
```

*trimmed reads created during second step

*The error rate specifies the difference in overlap between two corrected reads which is typically <1% (canu default value 0.045) for PacBio data and <2% (canu default 0.144) for Nanopore data (<1% on newest chemistries). Higher rate, more sensitive.

*Notice in the output there are separate directories for each error rate you specify.

Running Canu

- We will run default with correcting, trimming and assembling all in one command:

```
$ canu -p ecoli -d ecoli-pacbio  genomeSize=4.8m  -pacbio-raw
pacbio.fastq saveReads=true > CanuRun_20241023.out \
2>CanuRun_20241023.log **What are we saving?

*where are reads....include pathway in your command
-p assembly-prefix
-d output directory
input types:
 -pacbio-raw -pacbio-corrected -nanopore-raw -nanopore-corrected fastq or
fasta format


\ = line continuation – don't need if you have room to write it out!!!
```

kmer genie: estimates the best k-mer length for genome de novo assembly:
http://kmergenie.bx.psu.edu Chikhi & Medvedev 2014 Bioinformatics

Running Canu - job control

`Ctrl - z` - stops job running at command line

`bg` - moves to background

`jobs` - see what is running

`fg` - job number - move job to foreground

Running Canu

- We will run default with correcting, trimming and assembling all in one command:

```
$ canu -p ecoli -d ecoli-pacbio  genomeSize=4.8m  -pacbio-raw
pacbio.fastq saveReads=true > CanuRun_20221017.out \
2>CanuRun_20241023.log **What are we saving?

*where are reads....include pathway in your command
-p assembly-prefix
-d output directory
input types:
 -pacbio-raw -pacbio-corrected -nanopore-raw -nanopore-corrected fastq or
fasta format


\ = line continuation – don't need if you have room to write it out!!!
```

kmer genie: estimates the best k-mer length for genome de novo assembly:
http://kmergenie.bx.psu.edu   Chikhi & Medvedev 2014 Bioinformatics

Canu output files - all found in helpdocs

- **ecoli*.report -** assembly analysis log **-** histograms of read lengths, kmers and raw/corrected reads. summary of corrected data, overlaps and contig lengths

READS

- **ecoli*.correctedReads.fasta.gz** - The sequences after correction, trimmed and split based on consensus evidence. Typically >99% for PacBio and >98% for Nanopore but it can vary based on your input sequencing quality
- **ecoli*.trimmedReads.fasta.gz -** The sequences after correction and final trimming. The corrected sequences above are overlapped again to identify any missed hairpin adapters or bad sequence that could not be detected in the raw sequences.

SEQUENCE

- **ecoli*.contigs.fasta** - Everything which could be assembled and is part of the primary assembly, including both unique and repetitive elements.
- **ecoli*.unassembled.fasta -** Reads and low-coverage contigs which could not be incorporated into the primary assembly.

How many contigs to we have?

Canu output files - all found in helpdocs

SEQUENCE

- **ecoli.contigs.fasta** - Everything which could be assembled and is part of the primary assembly, including both unique and repetitive elements.
- **ecoli.unassembled.fasta** - Reads and low-coverage contigs which could not be incorporated into the primary assembly.

The header line for each sequence provides some metadata on the sequence:

```
>tig00000001 len=4665109 reads=10220 class=contig suggestRepeat=no
suggestBubble=no suggestCircular=yes
trim=20235-4660825
```

len - Length of the sequence **in** bp.

reads - Number of reads used to form the contig.

class - Type of sequence. Unassembled sequences are primarily low-coverage sequences spanned by a single read.

suggestRepeat - If yes, sequence was detected **as** a repeat based on graph topology **or** read overlaps to other sequences.

suggestBubble - If yes, sequence **is** likely a bubble based on potential placement within longer sequences.

suggestCircular - If yes, sequence **is** likely circular.

trim - X-Y to indicate the non-redundant coordinates

Running Canu

Nanopore data

canu -p ecoli -d **ecoli-oxford** genomeSize=4.8m maxInput**Coverage**=100 -**nanopore** ecolk12mg1655_R10_3_guppy_345_HAC.fastq

PacBio HiFi with HiCanu

canu -p asm -d ecoli_hifi genomeSize=4.8m -**pacbio-hifi** ecoli.fastq

Multiple Technologies & Multiple Files

canu -p ecoli -d ecoli-mix genomeSize=4.8m -pacbio **pacbio**.fastq.gz

    -**nanopore** oxford.fasta.gz

Trio Binning Assembly

- Using parental short-read sequencing

canu -p asm -d ecoliTrio genomeSize=5m -haplotypeK12 K12.parental.fasta -haplotypeO157 O157.parental.fasta -pacbio F1.fasta

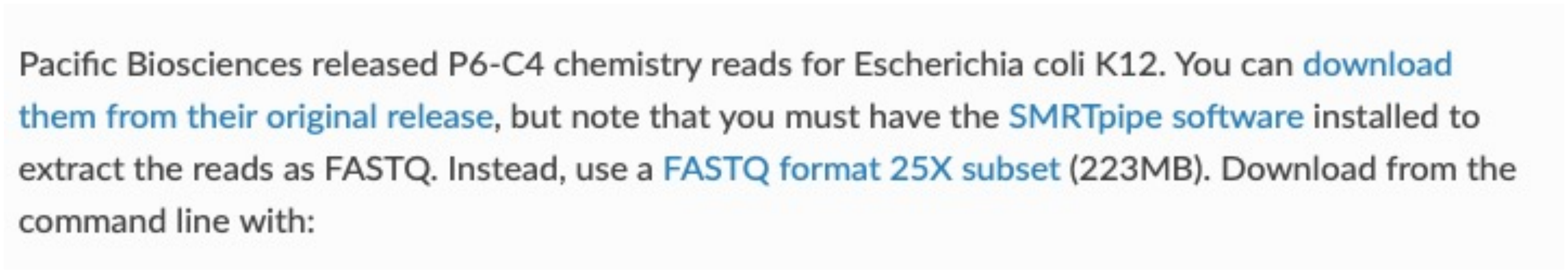
Genome coverage

- How much coverage is enough?

  4.5 Mb - 223Mb fastq reads (25X)

  -13,124 reads

  - 52,496 lines


Pacific Biosciences released P6-C4 chemistry reads for Escherichia coli K12. You can download them from their original release, but note that you must have the SMRTpipe software installed to extract the reads as FASTQ. Instead, use a FASTQ format 25X subset (223MB). Download from the command line with:

Try some assemblies with filtered data sets

Using canu - split reads

- Count and split files:

```
$ wc -l pacbio.fastq  n=52,496
$ split -l
```

  - l create smaller file with n lines

- 60%   31,498 - lines

```
$ split -l 31498 pacbio.fastq
$ mv xaa ecoli_filtered_0.60.fastq
$ rm the rest (xa*)
```

- confirm number of lines and reads

Genome coverage

- How much coverage is enough?

  4.5 Mb - 223Mb fastq reads (25X), 13,124 reads

  52496 lines

Try some assemblies with filtered data sets:

0.80 - 10,499 reads, 180M, ~22X   How many lines?

0.75 - 9,843 reads, 168 Mb, ~18X

0.60 - 7,874 reads, 133Mb, ~15X

0.50 - 6,562 reads, 111Mb, ~12.5X

0.40 - 5250 reads, 89Mb, ~10X

0.25 - 3,281 reads, 56Mb, ~ 6.3X

**Don't forget to split on line counts not reads!

Running Canu

```
$ canu -p ecoli -d ecoli-pacbio  genomeSize=4.8m  -
pacbio-raw  pacbio.fastq saveReads=true >
CanuRun_20241023.out 2>CanuRun_20241023.log
```

****What are we saving?**

```
    -p assembly-prefix
    -d output directory
```

****Make sure you change output file & directory names**

```
canu -p ecoli-60 -d ecoli-pacbio-60  genomeSize=4.8m
-pacbio-raw ecoli_filtered_0.60.fastq saveReads=true
> CanuRun-60_20241023.out
2> CanuRun-60_20241023.log
```

Genome coverage

- How much coverage is enough?

  Full data set:

  4.8 Mb - 223Mb fastq reads (25X), 13,124 reads

Percent Reads	Coverage	Contigs	Unassembled Sequences	Unassembled Length (kb)	Genome Size (Mb)
1	25X				
0.8	22X				
0.75	18X				
0.6	15X				
0.5	12.5X				
0.25	6.3X				

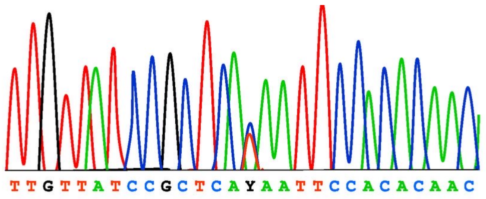
Genome coverage

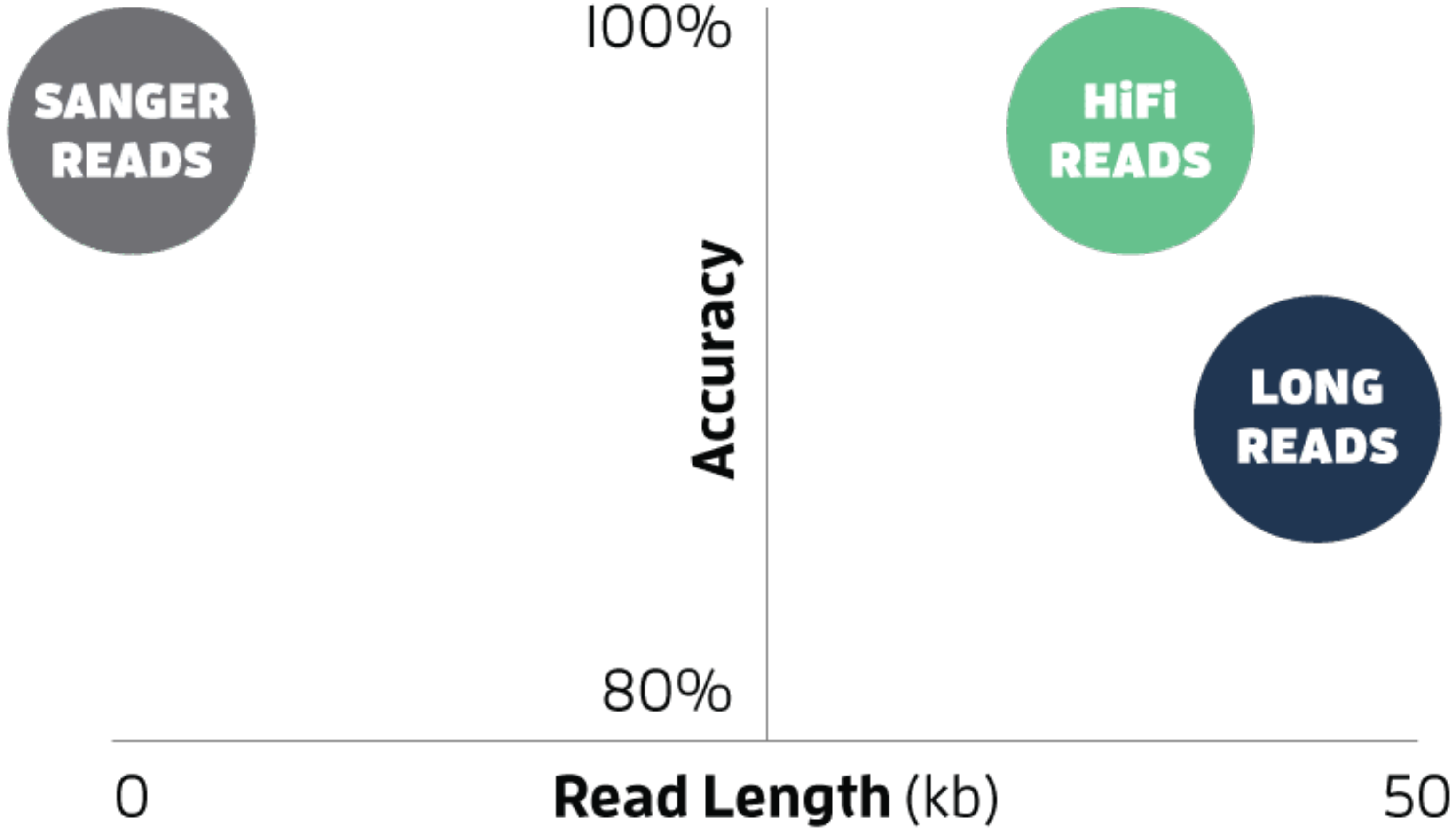
- How much coverage is enough?

  Full data set:

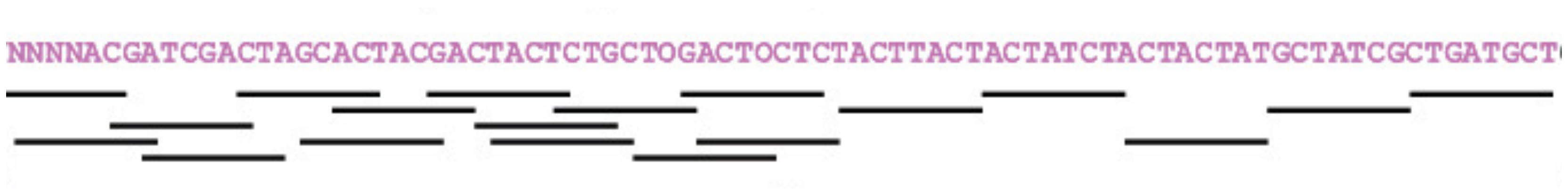
  4.8 Mb - 223Mb fastq reads (25X), 13,124 reads

Percent Reads	Coverage	Contigs	Unassembled Sequences	Unassembled Length (kb)	Genome Size (Mb)
1	25X	1	2	25	4.8
0.8	22X	11	21	258	4.8
0.75	18X	18	37	446	4.6
0.6	15X	48	150	1276	4.5
0.5	12.5X	78	449	3154	4.0
0.25	6.3X	59	1152	6162	1.0


TTGTTATCCGCTCAYAATTCCACACAAC


100%
80%
Accuracy
SANGER READS
HiFi READS
LONG READS
0
Read Length (kb)
50

Genome coverage


NNNNACGATCGACTAGCACTACGACTACTCTGCTOGACTOCTCTACTTACTACTATCTACTACTATGCTATCGCTGATGCT

How many times has genome been sequenced?

1. random distribution of reads
2. overlap detection does not vary across genome
3. number of times base sequenced - Poisson distribution
4. can be used to model any discrete occurrence given an average number of occurrences

Lander ES, Waterman MS. 1988. Genomics

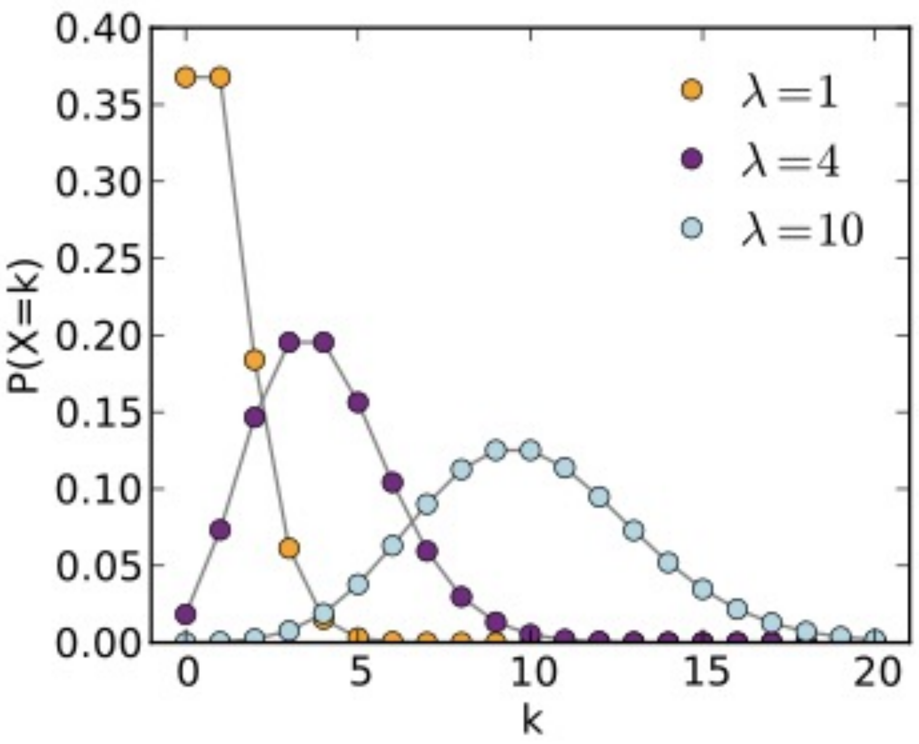
Poisson Distribution

The probability of a given number of events occurring in a fixed interval of time and/or space if these events occur with a known average rate and independently of the time since the last event.

Reads are not distributed evenly over an entire genome, simply because the reads will sample the genome in a random and independent manner

Lander-Waterman Model
Coverage c = L * N / G where L is the read length, N is the number of reads and G is the genome size

Can be modeled by a Poisson distribution where λ = c



https://www.illumina.com/documents/products/technotes/technote_coverage_calculation.pdf

Genome coverage

Coverage	P(x=0)	% sequenced
0.25	0.78	22%
0.5	0.61	39%
0.75	0.47	53%
1	0.37	63%
1.5	0.223	77.7%
2	0.135	87.5%
3	0.05	95%
4	0.018	98.2%
5	0.0067	99.33%
6	0.0025	99.75%
7	0.0009	99.91%
8	0.0003	99.97%
9	0.0001	99.99%
10	0.000045	99.995%

Genome assemblers

Verkko Telomere-to-telomere assemblies

https://github.com/marbl/verkko

FALCON & FALCON-unzip - PacBio

https://pb-falcon.readthedocs.io/en/latest/about.html

MaSuRCA - Illumina, PacBio, Nanopore

https://github.com/alekseyzimin/masurca

SPAdes - Illumina, PacBio, Nanopore

https://github.com/ablab/spades

Hinge - PacBio, Nanopore

https://github.com/HingeAssembler/HINGE

Abyss - Illumina

https://github.com/bcgsc/abyss

Shasta - Nanopore

https://github.com/paoloshasta/shasta

Genome Assembly Problem Set

A. Using ecoli-0.25.contigs.fasta, write a python script that reports:

1. The number of contigs in the file
2. The shortest contig
3. The longest contig
4. Total contig length
5. The L50 size
6. The N50 size

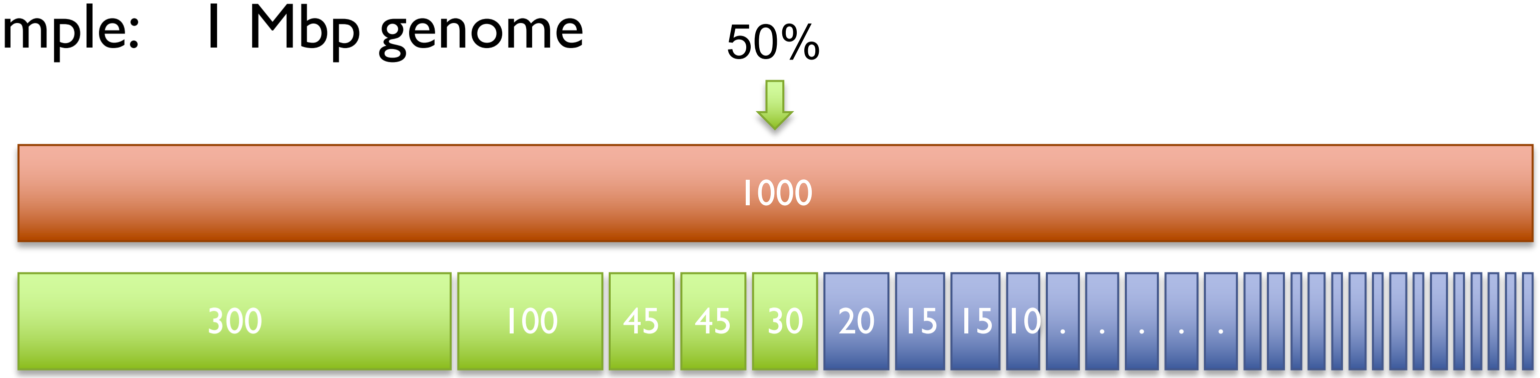
The fasta file for the exercise:

ecoli_0.25.contigs.fasta

N50 size

If we place our contigs from largest to smallest on the genome, 50% of the genome in contigs as long as or larger than N50 value

Example: 1 Mbp genome



N50 size = 30 kb

(300k+100k+45k+45k+30k = 520k >= 500kb)

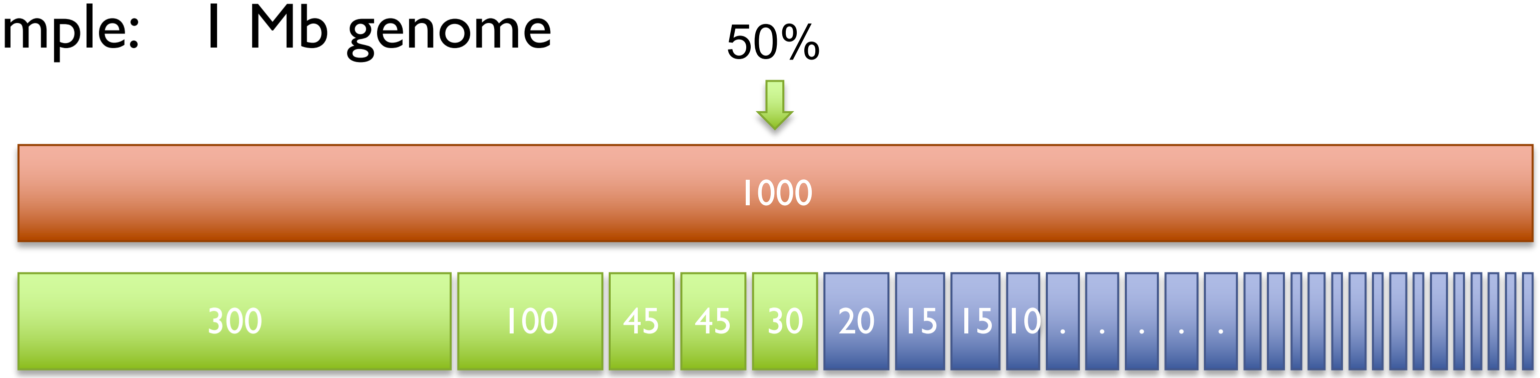
A greater N50 is usually a sign of assembly improvement

- Comparable with genomes of similar size
- Genome composition can bias comparisons
- Low L50 vs High N50

L50 size

Number of contigs that are as long or longer than the N50 value

Example: 1 Mb genome



N50 size = 30 kb L50 - contained in 5 contigs
(300k+100k+45k+45k+30k = 520k >= 500kb)

- Low L50 vs High N50
  - longer sequences and fewer of them….in theory
  - lower stringency can inflate N50

Genome Assembly Problem Set

B. Working with a soft repeat-masked genome assembly with gaps:

When running RepeatMasker or other software that identifies repetitive sequences in a genome, the resulting sequence can be “soft-masked” where the repetitive elements are in lower-case letters. Any gaps present in the genome, where the sequence is unresolved are marked with ”N”s

gap soft-masked repeat

```
NNNNNNNNNNNNcaaacaaatatacaaagacAAAAATTGCCA
CAGCAAAGACAAAGAGATAAATAAAAGGCACAAAATTGTCAC
```

Genome Assembly Problem Set

B. For the following exercise, write a python script that parses a fasta file, *D. melanogaster* chromosome assembly, and identify the following:

1. How many contigs?
2. Nucleotide content:

   - number of each nucleotide both masked (a,c,g,t) and not (A,C,G,T)
3. What proportion of the genome is comprised of gaps (NNN)?

The fasta file for the exercise:
D_melanogaster_genomic.fna

large file, see filtering on next panel

Genome assembly downloaded from:
https://www.ncbi.nlm.nih.gov/datasets/genome/GCF_000001215.2/

Genome Assembly Problem Set

D_melanogaster_genomic.fna

```
wc -l D_melanogaster_genomic.fna
  449842 lines

grep -c ">" D_melanogaster_genomic.fna
448 reads
```

Previously we used Unix split command, which you can use again or write a Python script to output a filtered dataset