DNA sequencing, FASTQ format, tools.

BFX Workshop Week 2

Chris Miller

Parts adapted from:
Applied Computational Genomics, Lecture 6
https://github.com/quinlan-lab/applied-computational-genomics
Aaron Quinlan

How to sequence a human genome: Sanger method

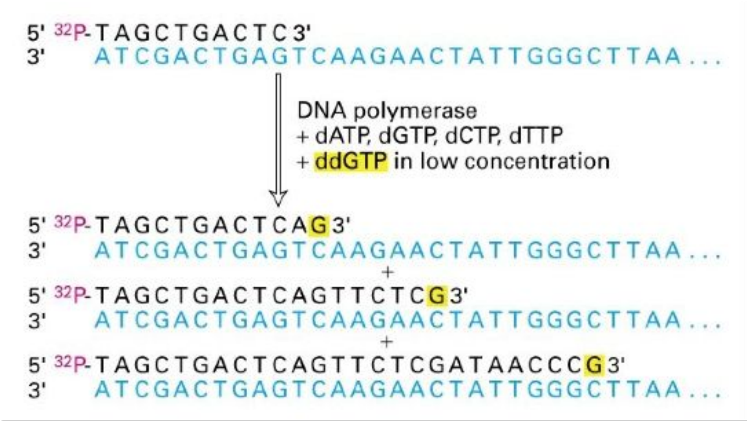
Key points:

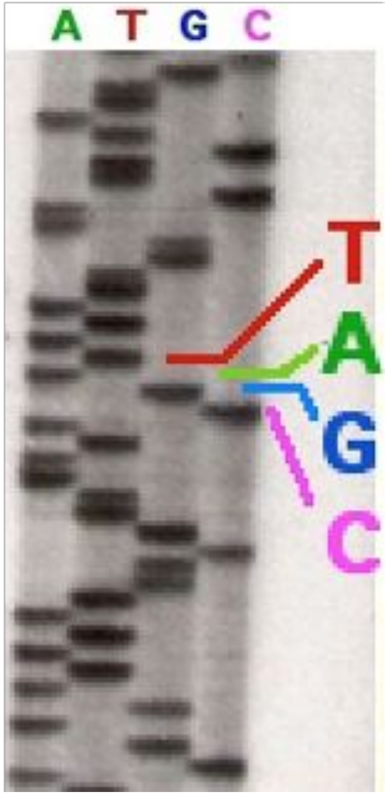
1) sequencing by synthesis (not degradation)

2) primers hybridize to DNA

3) polymerase + dNTPS + ddNTP terminators at low concentration

4) 1 lane per base, visually interpret ladder





Sanger sequencing: technological advances

1977: Fred Sanger → 1 hard working technician
= 700 bases per day
= 118,000 years to sequence the human genome

1985: ABI 370 (first automated sequencer) → 5000 bases per day
= 16,000 years

1995: ABI 377 (Bigger gels, better chemistry & optics, more sensitive dyes, faster computers) → 19,000 bases per day
= 4,400 years

1999: ABI 3700 (96 capillaries, 96 well plates, fluid handling robots) → 400,000 bases per day
= 205 years

Slide from Ira Hall

The next wave of DNA sequencing technologies

whiz-bang terms

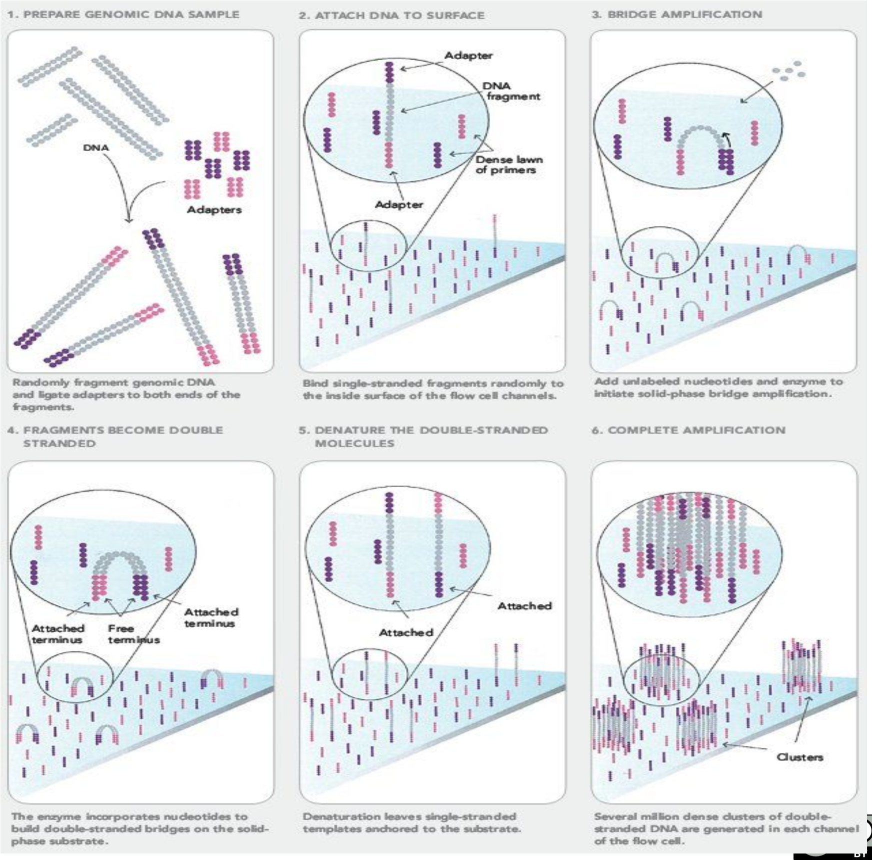
- "Massively parallel" sequencing
- "High-throughput" sequencing
- "Ultra high-throughput" sequencing
- "Next generation" sequencing (NGS)
- "Second generation" sequencing

- **2005: 454 (Roche)**
- **2006: Solexa (Illumina)**
- **2007: ABI/SOLiD (Life Technologies)**
- **2010: Complete Genomics**
- **2011: Pacific Biosciences**
- **2010: Ion Torrent (Life Technologies)**
- **2015: Oxford Nanopore Technologies**

Slide from Ira Hall

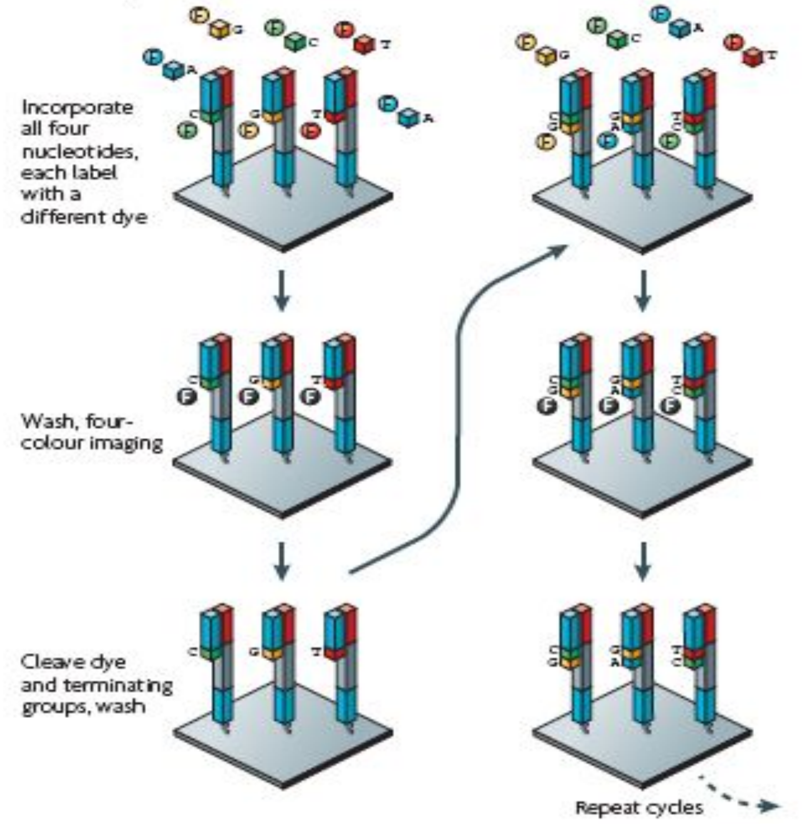
Solexa (Illumina) sequencing (2006)

- **PCR amplify sample (opt.)**
- **Immobilize and amplify single molecules on a solid surface**
- **<u>Reversible</u> terminator sequencing with 4 color dye-labelled nucleotides**

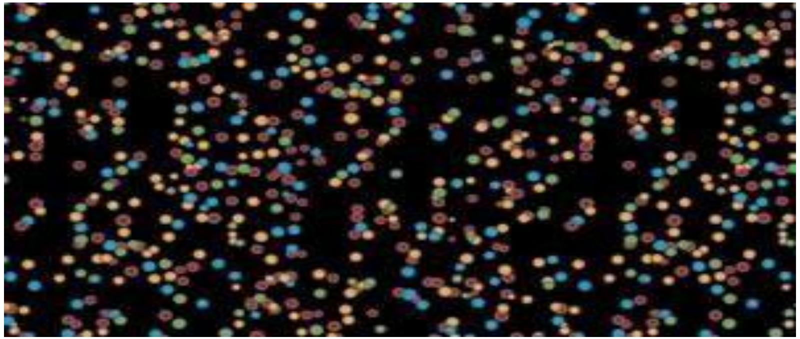


Slide from Ira Hall

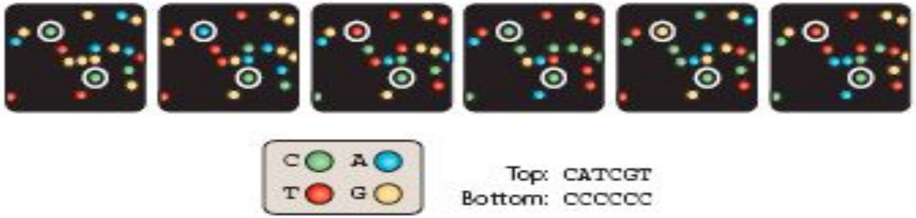
Cluster amplification by "bridge" PCR



4 different images merged



6 cycles w/ base-calling



Slide from Ira Hall

Illumina sequencing summary

Advantages:

- Best throughput, accuracy and read length for any 2nd gen. sequencer
- Fast & robust library preparation

Disadvantages:

- Inherent limits to read length (practically, 150bp)
- Some runs are error prone

<u>Illumina HiSeq</u>
~3 billion paired 100bp reads
~600Gb, $10K, 8 days
(or “rapid run” ~90Gb in 1-2 days)

<u>Illumina X Ten</u>
~6 billion paired 150bp reads
1.8Tb, <3 days, ~1000 / genome($$)
(or “rapid run” ~90Gb in 1-2 days)

<u>Illumina NextSeq</u>
One human genome in **<u><30 hours</u>**

Slide from Ira Hall

Paired-end sequencing:
A molecular hack to sequence longer fragments

genomic DNA

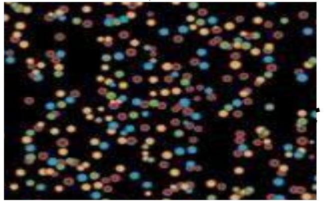
Shear to desired length (~400bp)

DNA fragments

ligate adapters, size select

sequencing library

Illumina GA2



clusters on a flow-cell

millions to billions of paired-end reads (readpairs)

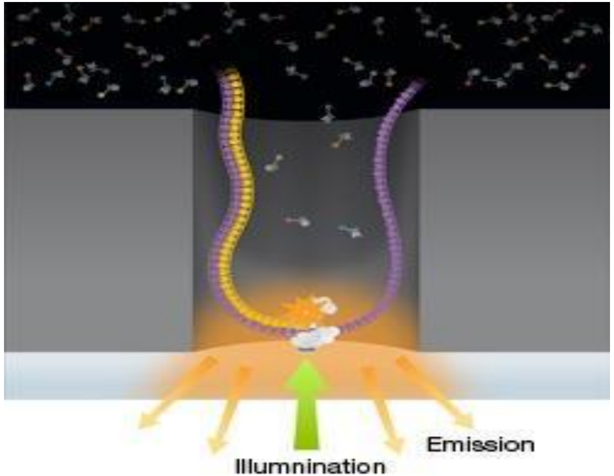
~150bp ~200bp ~150bp

5’ GGTGTACGAATAGTTTCCTTTTACACTCCTTGACCATCCTAGC --------------------------------//---------------------------------

---------------------------------//--------------------------------- GGACTGAAACTTCATCTGTCTTTATAGATATGCGTGCAGCAGC 5’

Slide from Ira Hall

Pacific Biosciences



Key Points:

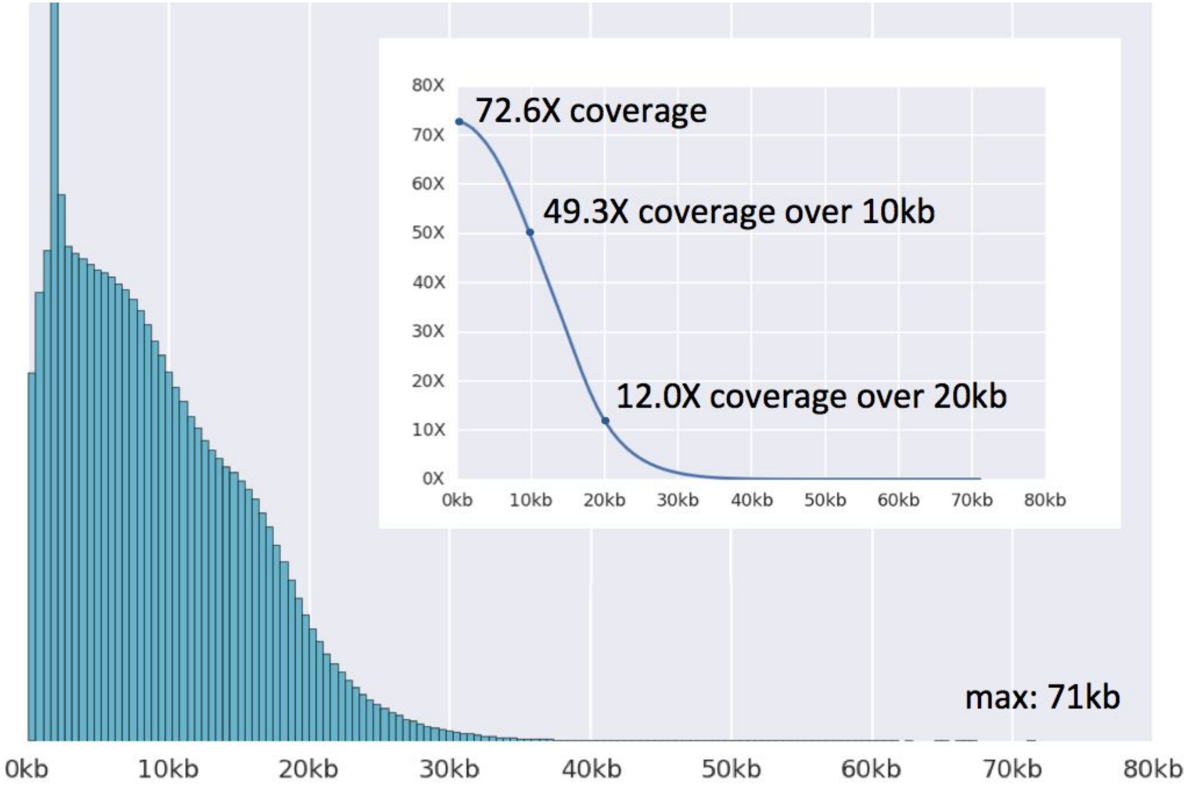
- 1 DNA molecule and 1 polymerase in each well (zero-mode waveguide)
- 4 colors flash in real time as polymerase acts
- Methylated cytosine has distinct pattern
- No _theoretical_ limit to DNA fragment length

Caveats:

- higher error rate (1-2%)
- lower throughput : roughly 5 gigabases per run

Slide from Ira Hall

Pacific Biosciences: long reads. Great for genome assembly



http://schatzlab.cshl.edu/data/skbr3/

Pacific Biosciences: long reads. Great for genome assembly

Sequence and assemble a bacterial genome in 24 hours for ~ $500

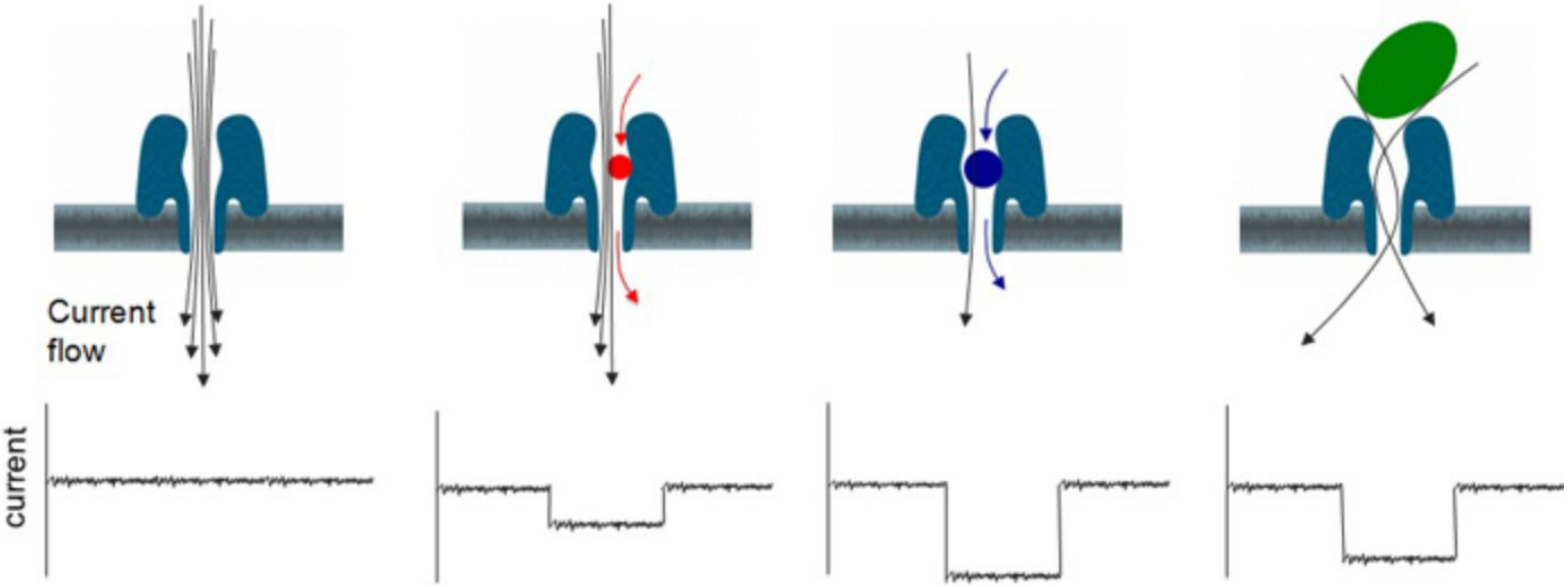
Human genome for ~$10,000

Single haplotype assembly of the human genome from a hydatidiform mole

Karyn Meltz Steinberg,[1] Valerie A. Schneider,[2] Tina A. Graves-Lindsay,[1] Robert S. Fulton,[1] Richa Agarwala,[2] John Huddleston,[3,4] Sergey A. Shiryev,[2] Aleksandr Morgulis,[2] Urvashi Surti,[5] Wesley C. Warren,[1] Deanna M. Church,[6] Evan E. Eichler,[3,4] and Richard K. Wilson[1]

[1]The Genome Institute at Washington University, St. Louis, Missouri 63108, USA; [2]National Center for Biotechnology Information, National Library of Medicine, National Institutes of Health, Bethesda, Maryland 20894, USA; [3]Department of Genome Sciences, University of Washington, Seattle, Washington 98195, USA; [4]Howard Hughes Medical Institute, University of Washington, Seattle, Washington 98195, USA; [5]Department of Pathology and Human Genetics, University of Pittsburgh, Pittsburgh, Pennsylvania 15260, USA; [6]Personalis, Inc., Menlo Park, California 94025, USA

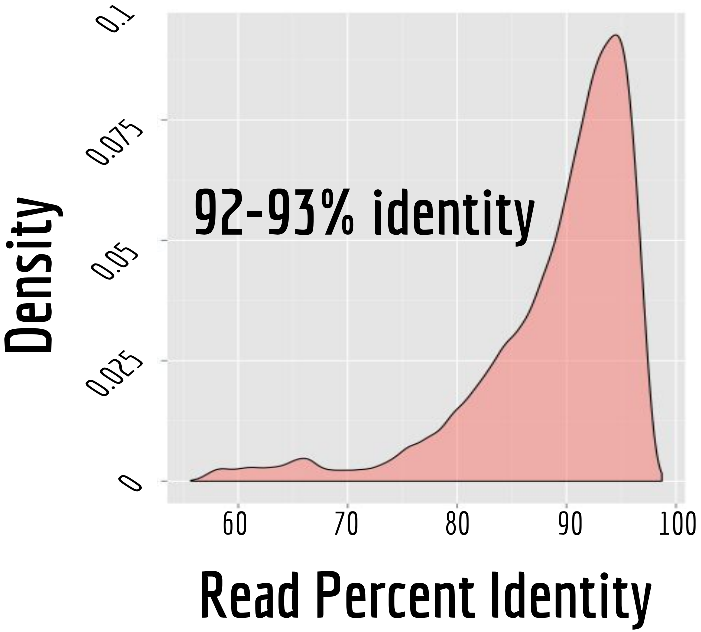
Oxford Nanopore Technologies



Nanopore sequencing with ONT is accurate and reliabl

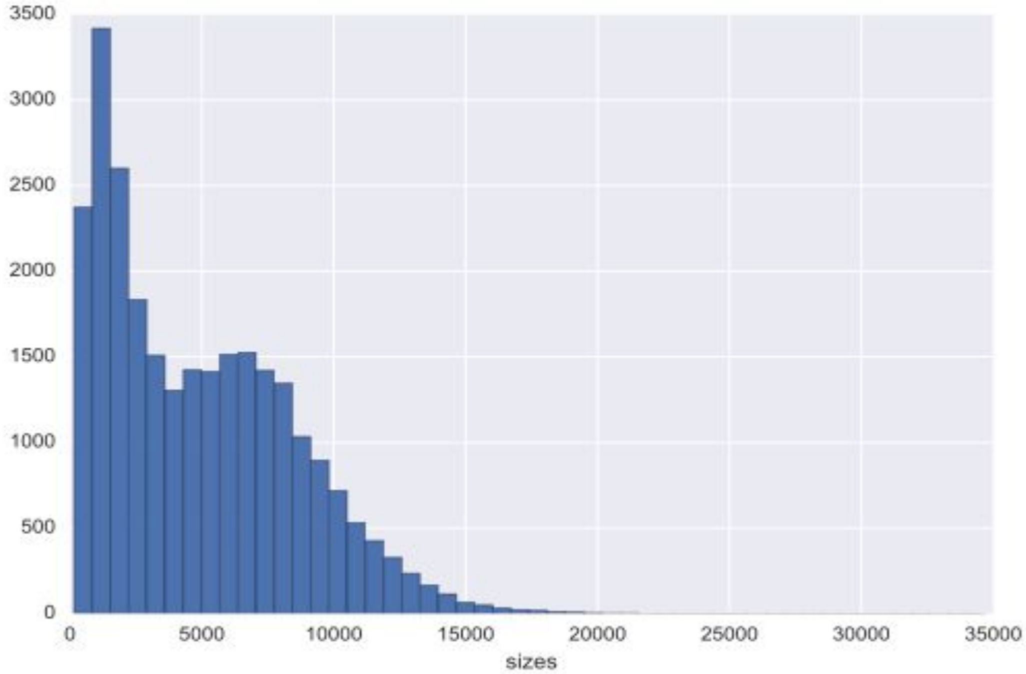
load libraries onto
array of 2,048 nanopores

2015 yield (R9 chemistry): ~500 Mbp
mean read size: 4-6 kbp



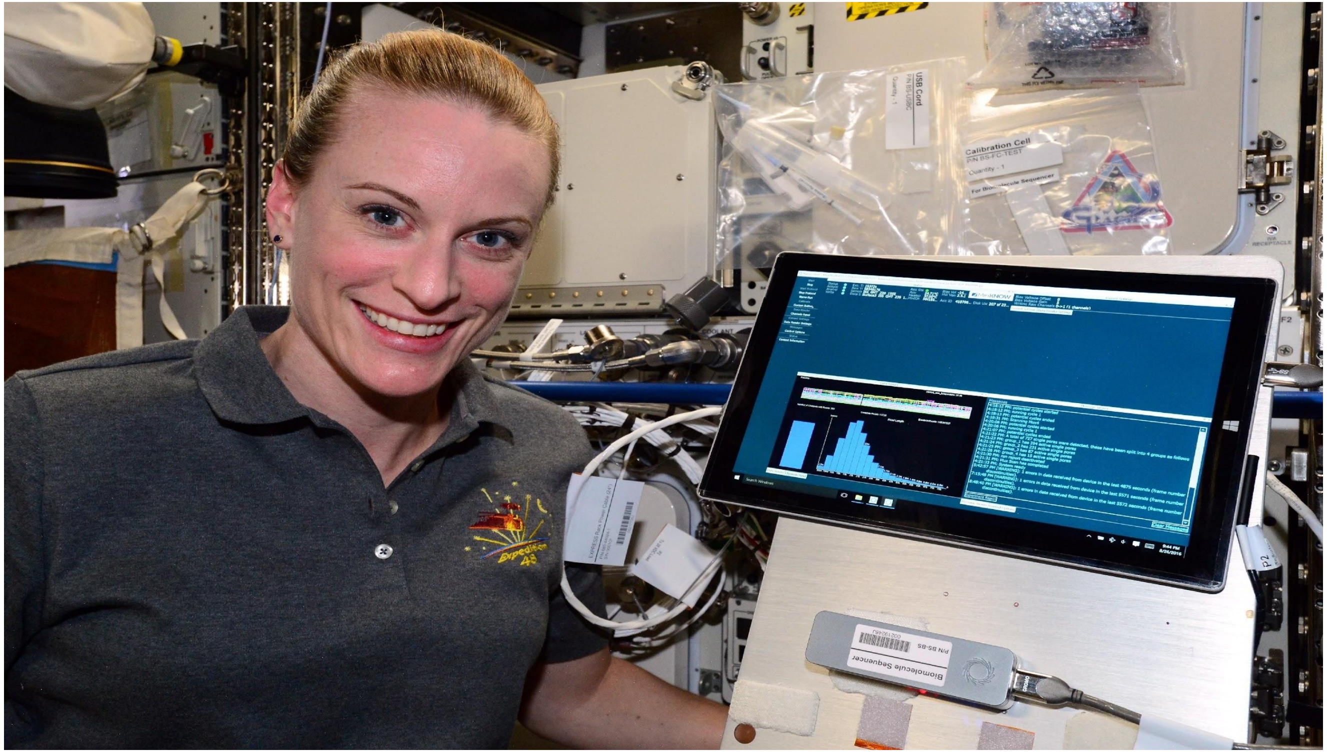
Current yield per run ("R9.4" chemistry):
~5 Gbp, 97% identity (i.e., 3% error rate)

ONT sequence length distribution



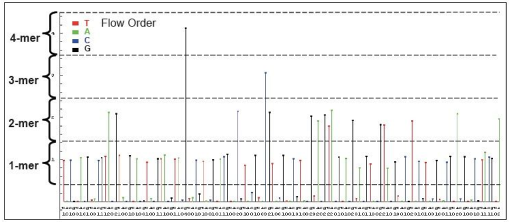
This is out of date - they're even longer now!

Nanopore sequencing is *extremely* portable



Kate Rubins sequencing DNA on the ISS

Base calling: the conversion of signal to a nucleotide sequence



Raw signal
(e.g., 454 Life Sciences)

Base calling algorithms

Errors happen.
Hopefully infrequently

ACCTTCGAACGGCGGGGGGTTACAA

(Mostly) all technologies yield DNA sequences in FASTQ format

DNA

```
@seq1
ACCTTCGAACGGCGGGGGGTTACAA
+
!''*((((***+))%%%++).1***
@seq2
TGGAACCGAACGGCCCCGGTTACAT
+
!''*!!!!***+))+++++).1***
And so on...
```

The FASTQ format. Welcome to a minor hell.

A “standard” format for storing and defining sequences from next-generation sequencing technologies.

```
 Sequence ID @SEQ_ID
    Sequence GATTTGGGGTTCAAAGCAGTATCGATCAAATAGTAAATCCATTTGTTCAACTCACAGTTT
 <separator> +
Quality scores !''*((((***+))%%%++)(%%%%).1***-+*''))**55CCF>>>>>>CCCCCCC65
```

http://en.wikipedia.org/wiki/FASTQ_format

The FASTQ format's sequence identifier (first line of each record)

Old format

```
@HWUSI-EAS100R:6:73:941:1973#0/1
```

HWUSI-EAS100R	the unique instrument name
6	flowcell lane
73	tile number within the flowcell lane
941	'x'-coordinate of the cluster within the tile
1973	'y'-coordinate of the cluster within the tile
#0	index number for a multiplexed sample (0 for no indexing)
/1	the member of a pair, /1 or /2 *(paired-end or mate-pair reads only)*

New format

```
@EAS139:136:FC706VJ:2:2104:15343:197393 1:Y:18:ATCACG
```

EAS139	the unique instrument name
136	the run id
FC706VJ	the flowcell id
2	flowcell lane
2104	tile number within the flowcell lane
15343	'x'-coordinate of the cluster within the tile
197393	'y'-coordinate of the cluster within the tile
1	the member of a pair, 1 or 2 *(paired-end or mate-pair reads only)*
Y	Y if the read is filtered, N otherwise
18	0 when none of the control bits are on, otherwise it is an even number
ATCACG	index sequence

FASTQ quality scores: estimate of confidence in each base
(sequencing technologies make errors!)

Sequence ID	`@SEQ_ID`
Sequence	`GATTTGGGGTTCAAAGCAGTATCGATCAAATAGTAAATCCATTTGTTCAACTCACAGTTT`
‹separator›	`+`
Quality scores	`!''*((((***+))%%%++)(%%%%).1***-+*''))**55CCF>>>>>>CCCCCCC65`

FASTQ quality scores: estimate of confidence in each base

(sequencing technologies make errors!)

Sequence ID	`@SEQ_ID`
Sequence	`GATTTGGGGTTCAAAGCAGTATCGATCAAATAGTAAATCCATTTGTTCAACTCACAGTTT`
‹separator›	`+`
Quality scores	`!''*((((***+))%%%++)(%%%%).1***-+*''))**55CCF>>>>>>CCCCCCC65`

↓

Qualities are based on the Phred scale and are *encoded*

$$Q = -10 * \log_{10}(P_{err})$$

Phred quality score calculation

$$Q = -10*\log_{10}(P_{err})$$

Error probability (P_{err})	$\log_{10}(P_{err})$	Phred quality score
1	0	0
0.1	-1	10
0.01	-2	20
0.001	-3	30
0.0001	-4	40

FASTQ quality scores: estimate of confidence in each base
(sequencing technologies make errors!)

Sequence ID	`@SEQ_ID`
Sequence	`GATTTGGGGTTCAAAGCAGTATCGATCAAATAGTAAATCCATTTGTTCAACTCACAGTTT`
‹separator›	`+`
Quality scores	`!''*((((***+))%%%++)(%%%%).1***-+*''))**55CCF>>>>>>CCCCCCC65`

Quality score encoding based on ASCII table

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	00	Null	32	20	Space	64	40	@	96	60	`
1	01	Start of heading	33	21	!	65	41	A	97	61	a
2	02	Start of text	34	22	"	66	42	B	98	62	b
3	03	End of text	35	23	#	67	43	C	99	63	c
4	04	End of transmit	36	24	$	68	44	D	100	64	d
5	05	Enquiry	37	25	%	69	45	E	101	65	e
6	06	Acknowledge	38	26	&	70	46	F	102	66	f
7	07	Audible bell	39	27	'	71	47	G	103	67	g
8	08	Backspace	40	28	(	72	48	H	104	68	h
9	09	Horizontal tab	41	29	)	73	49	I	105	69	i
10	0A	Line feed	42	2A	*	74	4A	J	106	6A	j
11	0B	Vertical tab	43	2B	+	75	4B	K	107	6B	k
12	0C	Form feed	44	2C	,	76	4C	L	108	6C	l
13	0D	Carriage return	45	2D	-	77	4D	M	109	6D	m
14	0E	Shift out	46	2E	.	78	4E	N	110	6E	n
15	0F	Shift in	47	2F	/	79	4F	O	111	6F	o
16	10	Data link escape	48	30	0	80	50	P	112	70	p
17	11	Device control 1	49	31	1	81	51	Q	113	71	q
18	12	Device control 2	50	32	2	82	52	R	114	72	r
19	13	Device control 3	51	33	3	83	53	S	115	73	s
20	14	Device control 4	52	34	4	84	54	T	116	74	t
21	15	Neg. acknowledge	53	35	5	85	55	U	117	75	u
22	16	Synchronous idle	54	36	6	86	56	V	118	76	v
23	17	End trans. block	55	37	7	87	57	W	119	77	w
24	18	Cancel	56	38	8	88	58	X	120	78	x
25	19	End of medium	57	39	9	89	59	Y	121	79	y
26	1A	Substitution	58	3A	:	90	5A	Z	122	7A	z
27	1B	Escape	59	3B	;	91	5B	[	123	7B	{
28	1C	File separator	60	3C	<	92	5C	\	124	7C	|
29	1D	Group separator	61	3D	=	93	5D	]	125	7D	}
30	1E	Record separator	62	3E	>	94	5E	^	126	7E	~
31	1F	Unit separator	63	3F	?	95	5F	_	127	7F	□

Formula for getting PHRED quality from encoded quality:

Q = ascii(char) - 33

Example:
!+EJ

ASCII	33	43	69	74
-33	0	10	36	41

Historically, FASTQ has had different encoding schemes for encoding PHRED quality scores. Ouch.

```
SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS.....................................................
..........................XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX......................
...............................IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII......................
.................................JJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJ......................
LLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLL....................................................
!"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|}~
|                         |    |        |                              |                     |
33                        59   64       73                            104                   126
0........................26...31.......40
                         -5....0........9..............................40
                               0........9..............................40
                                  3.....9..............................40
0.2......................26...31........41

S - Sanger        Phred+33,  raw reads typically (0, 40)
X - Solexa        Solexa+64, raw reads typically (-5, 40)
I - Illumina 1.3+ Phred+64,  raw reads typically (0, 40)
J - Illumina 1.5+ Phred+64,  raw reads typically (3, 40)
    with 0=unused, 1=unused, 2=Read Segment Quality Control Indicator (bold)
    (Note: See discussion above).
L - Illumina 1.8+ Phred+33,  raw reads typically (0, 41)
```

Current encoding:
! = quality 0
J = quality 41

FASTQE

https://github.com/fastqe/fastqe

```
$ fastqe  example.fastq  --min --max --bin
example.fastq   max (binned)     😎 😎 😎 😎 😎 😎 😎
example.fastq   mean (binned)    😁 😁 😁 😁 😁 😁 😁
example.fastq   min (binned)     💩 💀 💀 💀 💩 💀 💩
```

Quality score binning

Table 1: Q-Score Bins for an Optimized 8-Level Mapping

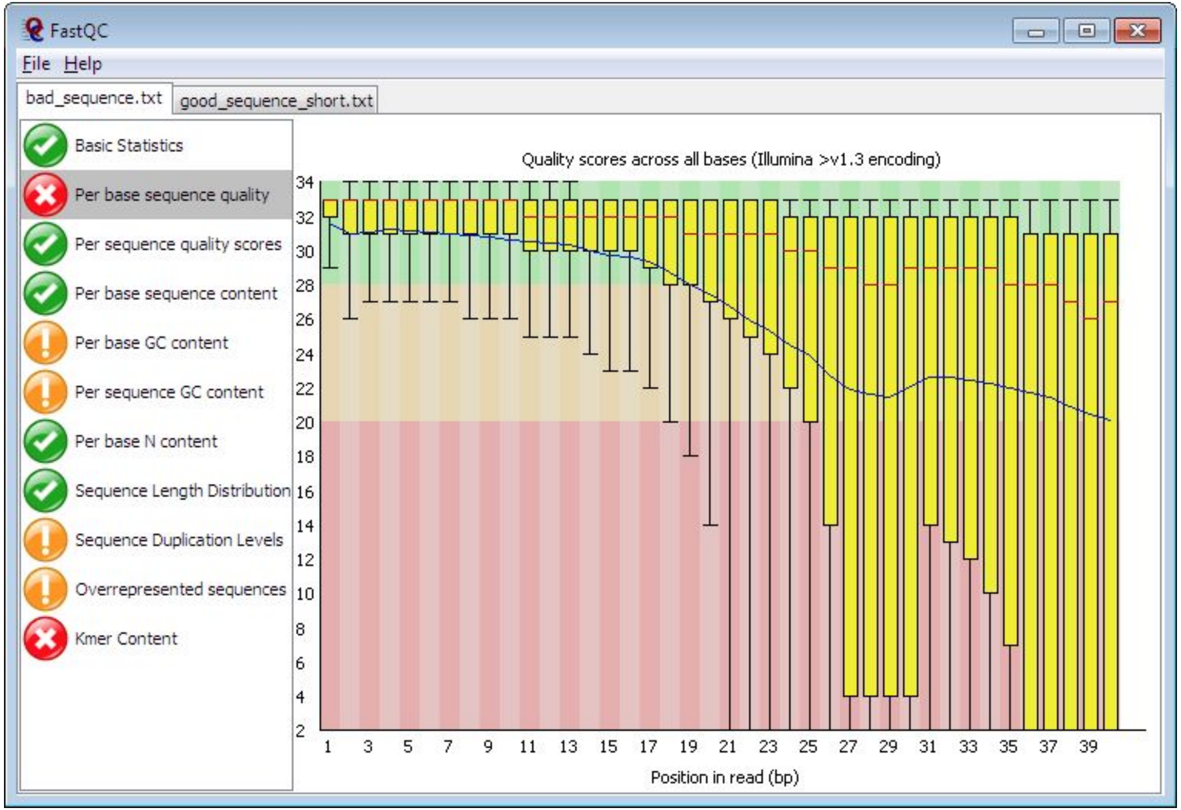
Quality Score Bins	Example of Empirically Mapped Quality Scores*
N (no call)	N (no call)
2–9	6
10–19	15
20–24	22
25–29	27
30–34	33
35–39	37
≥ 40	40

https://www.illumina.com/documents/products/whitepapers/whitepaper_datacompression.pdf

FASTQ vs FASTA

```
>seq1
ACGACGAGACCTTCATCAAAAACATCATCATCCAGGACTGTATGTGGAGCGGCTTCTCGG
CCGCCGCCAAGCTCGTCTCAGAGAAGCTGGCCTCCTACCAGGCTGCGCGCAAAGACAGCG
GCAGCCCGAACCCCGCCCGCGGCCACAGCGTCTGCTCCACCTCCAGCTTGTACCTGCAGG
ATCTGAGCGCCGCCGCCTCAGAGTGCATCGACCCCTCGGTGGTCTTCCCCTACCCTCTCA
ACGACAGCAGCTCGCCCAAGTCCTGCGCCTCGCAAGACTCCAGCGCCTTCTCTCCGTCCT
CGGATTCTCTGCTCTCCTCGACGGAGTCCTCCCCGCAGGGCAGCCCCGAGC
>seq2
TCCATGAGGAGACACCGCCCACCACCAGCAGCGACTCTGGTAAGCGAAGCCCGCCCAGGC
CTGTCAAAAGTGGGCGGCTGGATACCTTTCCCATTTTCATTGGCAGCTTATTTAACGGGC
CACTCTTATTAGGAAGGAGAGATAGCAGATCTGGAGAGATTTGGGAGCTCATCACCTCTG
AAACCTTGGGCTTTAGCGTTTCCTCCCATCCCTTCCCCTTAGACTGCCCATGTTTGCAGC
CCCCCTCCCCGTTTGTCTCCCACCCCTCAGGAATTTCATTTAGGTTTTTAAACCTTCTGG
CTTATCTTACAACTCAATCCACTTCTTCTTACCTCCCGTTAACATTTTAATTGCCCTGGG
GCGGGGTGGCAGGGAGTGTATGAATGAGGATAAGAGAGGATTGATCTCTGAGAGTGAATG
AATTGCTTCCCTCTTAACTTCCGAGAAGTGGTGGGATTTAATGAACTATCTACAAAAATG
```

FASTQC: Is my sequence data any good?



http://www.bioinformatics.babraham.ac.uk/projects/fastqc/

LSF on compute1

- If you are not set up on compute1, do that ASAP
  - If you need help, let us know!

- Example command for an interactive job:

```
bsub -Is -M 2G -G compute-timley -n 1 -R 'select[mem>2G] rusage[mem=2G]'
-q general-interactive -a 'docker(ubuntu:focal)' /bin/bash
```

LSF on compute1

- If you are not set up on compute1, do that ASAP
  - If you need help, let us know!

- Example command for an interactive job:

```
bsub -Is -M 2G -G compute-timley -n 1 -R 'select[mem>2G] rusage[mem=2G]'
-q general-interactive -a 'docker(ubuntu:focal)' /bin/bash
```

Some useful UNIX commands

- **`head`** print the first 10 lines of a file
- **`tail`** print the last 10 lines of a file
  getting fancy: **`tail -n +2`**
- **`wc`** count the number of characters/words/lines in a file
  `wc -l` for only lines
- **`less`** because you don't want 3 million lines scrolling through your terminal
  `q` to exit, **`-S`** to wrap lines (lots more useful options here)
- **`grep`** to search through a file (**`-v`** to search for lines *without* pattern)

| (pipes)

You cannot be a productive command line user until you really understand the power of pipes

```
grep TP53 genes.txt | grep “exon” | wc -l
```

This kind of construction allows you to get answers quickly!

Working with compressed data

- **`tar`** work with a "bundle" of data

  create: **tar -cvf output.tar infile1 infile2**
  extract: **tar -xvf output.tar**

- **`gzip`** compress a single file

  create: **gzip mydata.txt** (creates mydata.txt.gz)
  extract: **gunzip mydata.txt.gz** (creates mydata.txt)

Often these operations are combined

```
tar -czvf myfile.tar.gz <list of files>
tar -xzvf myfile.tar.gz
```

Sed (**s**tream **ed**itor)

Most commonly used for find and replace operations:

```
cat file.txt | sed ‘s/foo/bar/g’ >file_fixed.txt
```

Also useful in some other contexts, like printing every 2nd line of a file:

```
sed ‘0~2p’ file.txt
```

Schedule