Introduction to second-generation sequencing

Review: DNA sequencing

- Technologies to determine the nucleotide sequences of a DNA molecule.
- Motivation: decipher the genetic codes hidden in DNA sequences for different biological processes.
- **Genome projects**: determine DNA sequences for different species, e.g., human genome project.
- **Genomic research** (in a nutshell): study the functions of DNA sequences and related components.

Sequencing technologies

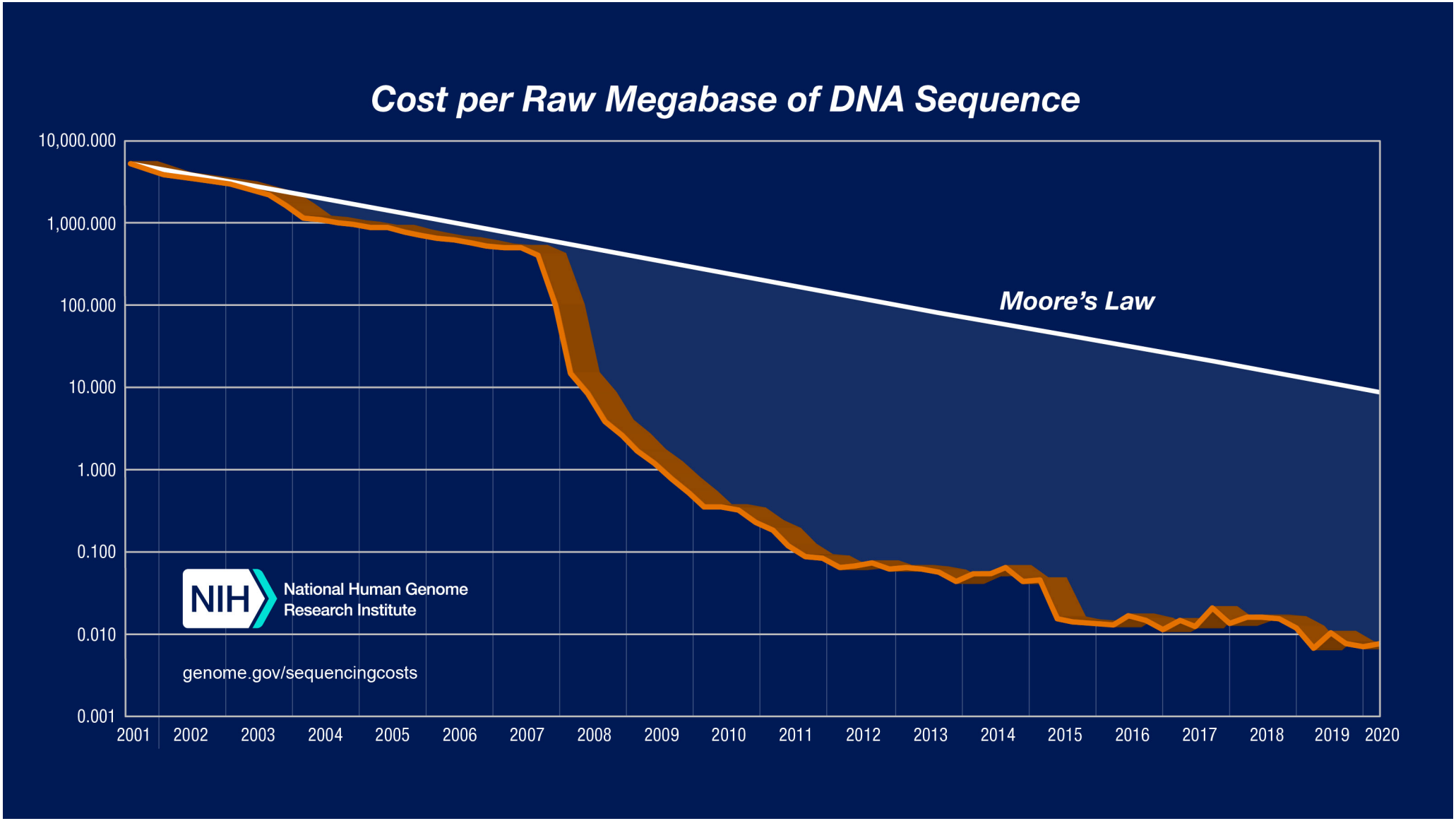
- Traditional technology: **Sanger sequencing**.
  - Slow (low throughput) and expensive: it took Human Genome Project (HGP) 13 years and $3 billion to sequence the entire human genome.
  - Relatively accurate.
- New technology: different types of **high-throughput sequencing**.

Second generation sequencing

- Aka: high-throughput sequencing, next generation sequencing (NGS).
- Able to sequence large amount of short sequence segments in a short period:
  - high throughput: billions of sequences in a run.
  - Cheap: sequence entire human genome with good coverage costs below one thousand dollars now.
  - short read length: up to several hundred bps.

HiSeq X Instrument Performance Parameters*

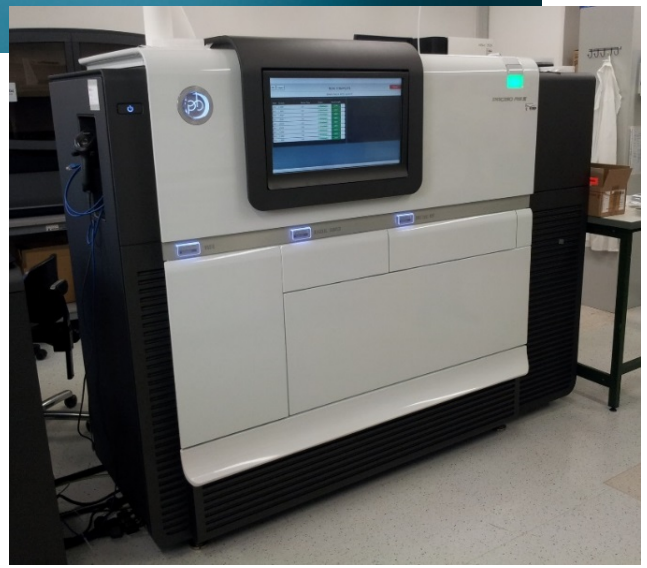
	Dual Flow Cell	Single Flow Cell
Output per Run	1.6-1.8 Tb	800-900 Gb
Reads Passing Filter	5.3-6 billion	2.6-3 billion
Supported Read Length	2 × 150 bp	
Run Time	< 3 days	
Quality Scores	≥ 75% of bases above Q30 at 2 × 150 bp	
Supported Library Preparation	TruSeq DNA PCR-Free Library Prep Kit, TruSeq Nano DNA Library Prep Kit	


Cost per Raw Megabase of DNA Sequence
10,000.000
1,000.000
100.000
10.000
1.000
0.100
0.010
0.001
Moore's Law
NIH
National Human Genome Research Institute
genome.gov/sequencingcosts
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020

Available platforms

- Major player:
  - Illumina: HiSeq, MiSeq.
  - LifeTech (now ThermoFisher): SOLiD, Ion Torrent.
- Others (third-generation sequencing):
  - Pacific Bioscience (SMRT)
  - Oxford Nanopore

High-throughput sequencers



Second-generation sequencing technologies

Second-generation sequencing technologies

- Complicated and involves a lot of biochemical reactions.
  - Sequencing by synthesis: https://www.youtube.com/watch?v=fCd6B5HRaZ8.
  - Sequencing by ligation.
  - Pyrosequencing.
- In a nutshell:
  - Cut the long DNA into smaller segments (several hundreds to several thousand bases).
  - Sequence each segment: start from one end and sequence along the chain, base by base.
  - The process stops after a while because the noise level is too high.
  - Results from sequencing are many sequence pieces. The lengths vary, usually a few thousands from Sanger, and several hundreds from NGS.
  - The sequence pieces are called “reads” for NGS data.

Single-end vs. paired-end sequencing

- Sequence one or both ends of the DNA segments.
- **Single-end** sequencing: sequence one end of the DNA segment.
- **Paired-end** sequencing: sequence both ends of a DNA segments.
  - Result reads are “paired”, separated by certain length (the length of the DNA segments, usually a few hundred bps).
  - Paired-end data can be used as single-end, but contain extra information which is useful in some cases, e.g., detecting structural variations in the genome.
  - Modeling technique is more complicated.

Applications of Second-generation sequencing

Applications

- NGS has a wide range of applications.
  - DNA-seq: sequence genomic DNA.
  - RNA-seq: sequence RNA products.
  - ChIP-seq: detect protein-DNA interaction sites.
  - Bisulfite sequencing (BS-seq): measure DNA methylation strengths.
  - A lot of others.
- Basically replaced microarrays with better data: greater dynamic range and higher signal-to-noise ratios.

Technology	Brief description
ChIP-seq	Locate protein-DNA interaction or histone modification sites.
CLIP-seq	Map protein-RNA binding sites
RNA-seq	Quantify expression
SAGE-seq	Quantify expression
RIP-seq	capture TF-bound transcripts
GRO-seq	evaluate promoter-proximal pausing
BS-seq	Profile DNA methylation patterns
MeDIP-seq	Profile DNA methylation patterns
TAB-seq	Profile DNA hydroxyl-methylation patterns
MIRA-seq	Profile DNA methylation patterns
ChiRP-seq	Map lncRNA occupancy
DNase-seq	Identify regulatory regions
FAIRE-seq	Identify regulatory regions
FRT-seq	Quantify expression
Repli-seq	Assess DNA replication timing
MNase-seq	Identify nucleosome position
Hi-C	Infer 3D genome organization
ChIA-PET	Detect long distance chromosome interactions
4C-seq	Detect long distance chromosome interaction
Sono-seq	Map open-chromatin sites
NET-seq	determine *in vivo* position of all active RNAP complexes.
NA-seq	Map Nuclease-Accessible Sites

DNA-seq

- Sequence the untreated genomic DNA.
  - Obtain DNA from cells, cut into small pieces then sequence the segments.
- Goals:
  - Compare with the reference genome and look for genetic variants:
    - Single nucleotide polymorphisms (SNPs)
    - Insertions/deletions (indels),
    - Copy number variations (CNVs)
    - Other structural variations (gene fusion, etc.).
  - De novo assembly of a new genome.

Variations of DNA-seq

- Targeted sequencing, e.g., exome sequencing.
  - Sequence the genomic DNA at targeted genomic regions.
  - Cheaper than whole genome DNA-seq, so that money can be spent to get bigger sample size (more individuals).
  - The targeted genomic regions need to be "captured" first using technologies such as microarrays.
- Metagenomic sequencing.
  - Sequence the DNA of a mixture of species, mostly microbes, in order to understand the microbial environments.
  - The goal is to determine number of species, and their proportions in the population.
  - The proportion can be associated to outcomes.

RNA-seq

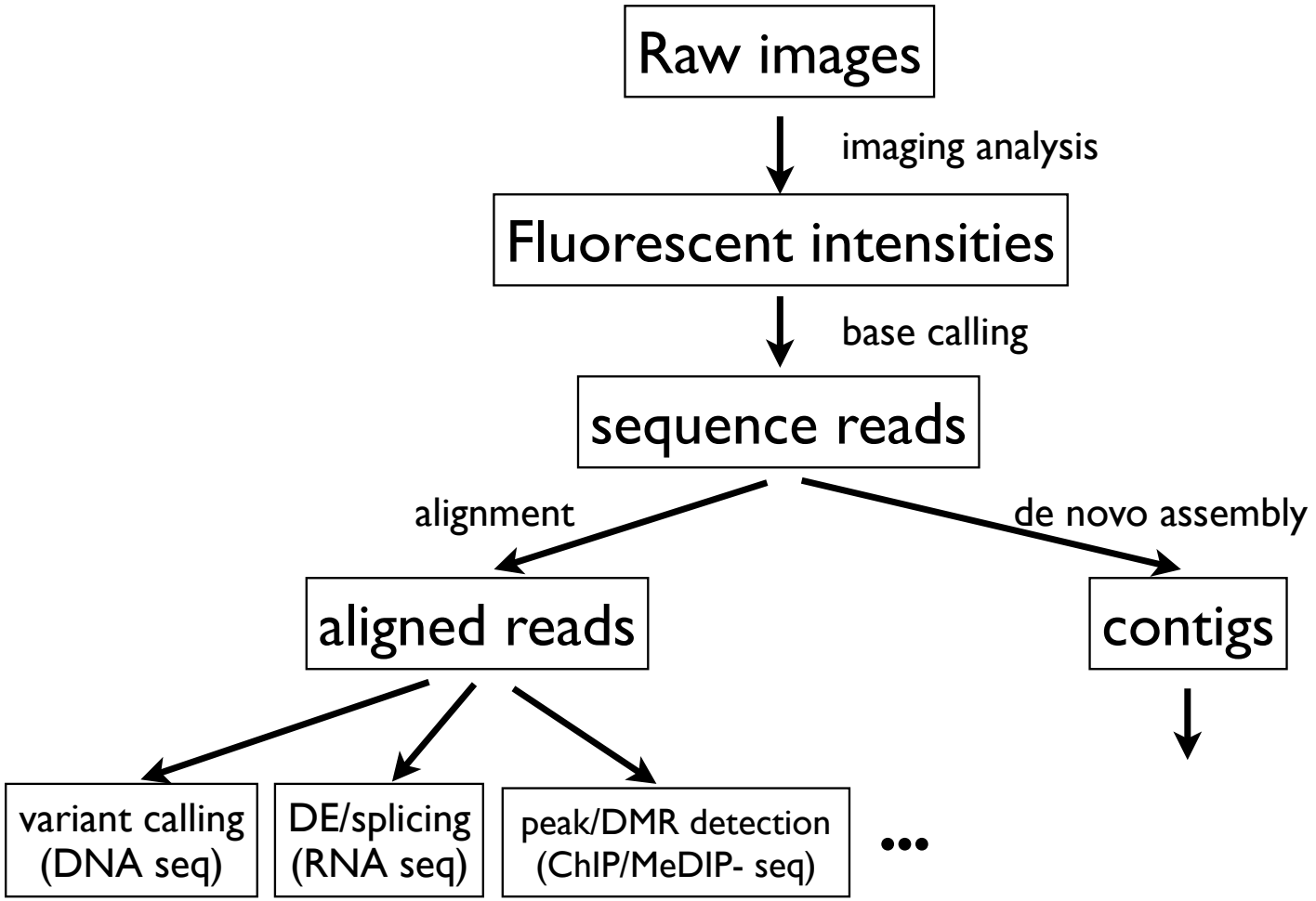
- Sequence the “transcriptome”: the set of RNA molecules.
- Goals:
  - Catalogue RNA products.
  - Determine transcriptional structures: alternative splicing, gene fusion, etc.
  - Quantify gene expression: the sequencing version of gene expression microarray.

ChIP-seq

- Chromatin-Immunoprecipitation (ChIP) followed by sequencing (seq): sequencing version of ChIP-chip.
- Used to detect locations of certain “events” on the genome:
  - Transcription factor binding.
  - DNA methylations and histone modifications.
- A type of “captured” sequencing. ChIP step is to capture genomic regions of interest.
- Similar technologies: MeDIP-seq, hMe-seal, etc.

Second-generation sequencing data analyses

Workflow of second generation sequencing data analysis



Imaging analysis

- Extract intensity values from images.
  - There are four images per sequencing cycle, one for a nucleotide.
- Similar to that in microarrays.
- Involves many statistical methods to extract signals from noisy data.
- Results of the imaging analysis: a 3-dimensional matrix: nreads x 4 x nbases.

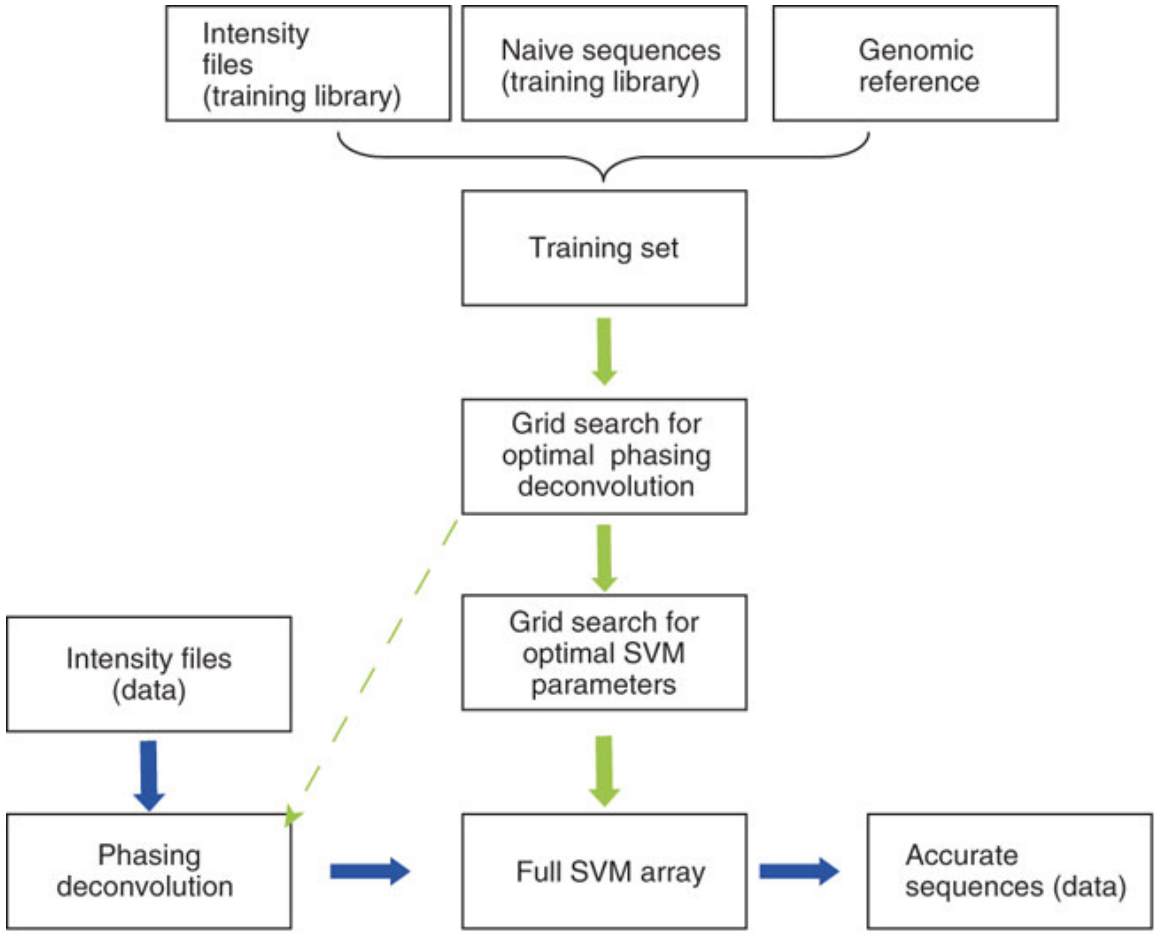
Base calling

- For each read, at each position, convert four fluorescent intensities (continuous) into a base or color (categorical).
- It's a classification problem.

	Base1	Base2	Base3	Base4	Base5	
A	0.290	0.046	0.014	0.026	0.010	
C	0.014	0.654	0.132	0.803	0.006	...
G	0.062	0.009	0.001	0.016	0.712	
T	0.016	0.010	0.455	0.046	0.768	

ACTCT…

Example of base calling method: Alta-Cyclic for Illumina data



The training process (green arrows) starts with creation of the training set, beginning with sequences generated by the standard Illumina pipeline, by linking intensity reads and a corresponding genome sequence (the 'correct' sequence). Then, two grid searches are used to optimize the parameters to call the bases. After optimization, a final SVM array is created, each of which corresponds to a cycle. In the base-calling stage (blue arrows), the intensity files of the desired library undergo deconvolution to correct for phasing noise using the optimized values and are sent for classification with the SVM array. The output is processed, and sequences and quality scores are reported.

Erlich et al. Nature Methods 5: 679-682 (2008)

Raw sequence reads from second generation sequencing after base calling

- Large text file (millions of lines) with simple format.
  - Most frequently used: fastq format

```
@HWI-EAS165:1:1:50:908:1
CTGCGGTCTCTAAAGTGCCATCTCATTGTGCTTTGTATCAGTCAGTGCTGGA
+
BCCBCB8ABBBBBBBB:BC=8@BBA:@BB@BBBCBB<9BBAC;A<C?BAAB<#
@HWI-EAS165:1:1:50:0:1
NCAACCCCCACAGTAATATGTAAAACAAAAACTAAAACCAGGAGCTGAAGGG
+
#BABABBBBBB@08<@?A@7:A@CCBCCCCBBBCCBB=?BBBB@7@B=A>:2
@HWI-EAS165:1:1:50:708:1
GGTCAGCATGTCTTCTGTTAAGTGCTTGCACAAGCTAGCCTCTGCCTATGGG
+
BB@A;B>@A@@=BB=BB?A>@@>B?ABBA=A?@@>@@A:=?>?A@=B8@@AB
@HWI-EAS165:1:1:50:1494:1
CTGGTGTCACACAAGCAGGTCTCCTGTGTTGACTTCACCAGACACTGTCATT
+
BCBB@AB@1ABBBBBBAAB?BBBBAB<A?AA>BB@?1ABBA@BBBA@;B>>:
```

read name
read sequence
separator
quality scores

Sequence alignment and assembly

- Sequence a known genome --- Alignment
  - Use the known genome (called “reference genome”) as a blue print.
  - Determine where each read is located in the reference genome.
- Sequence a whole new genome --- Assembly
  - New genome: a species with unknown genome, or the genome is believed to be very different from reference (e.g., cancer).
  - Basically the short reads are “stitched” together to form long sequences called “contigs”.
  - Overlaps among sequence reads are required, so it needs a lot of reads (deep coverage).
  - More computationally intensive.

Alignment

- Need: sequence reads file and a reference genome.
- It is basically a string search problem: where is the short (50-letter) string located within the reference string of 3 billion letters.
- Brute-force searching is okay for a single read, but computationally infeasible to alignment millions of reads.
- Clever algorithms are needed to preprocess the reference genome (indexing), which is beyond the scope of this class.

Popular general alignment software

- Bowtie: fast, but less accurate.
- BWA (Burrows-Wheeler Aligner): same algorithm as bowtie, but allow gaps in alignments.
  - about 5-10 times slower than bowtie, but provide better results especially for paired end data.
- Maq (Mapping and Assembly with Qualities): with SNP calling capabilities.
- ELAND: Illumina's commercial software.
- A lot of others. See http://en.wikipedia.org/wiki/List_of_sequence_alignment_software for more details.

Other technology-specific alignment software

- RNA-seq:
  - Tophat, STAR, HISAT
  - Pseudoaligner*:* Salmon, kallisto.
- Bisulfite sequencing:
  - Bismark
  - BSMAP

bowtie

Bowtie is an ultrafast, memory-efficient short read aligner. It aligns short DNA sequences (reads) to the human genome at a rate of over 25 million 35-bp reads per hour. Bowtie indexes the genome with a Burrows-Wheeler index to keep its memory footprint small: typically about 2.2 GB for the human genome (2.9 GB for paired-end).

Use bowtie: build alignment index

- Alignment index files are built based on reference genome (can be download as text files from UCSC).
- Note that pre-built indexes for many genomes are available from bowtie page, check that before building your own index.
- Command example for Human hg18 genome. Assume we have the hg18 sequence file ready called hg18.fa:

```
bowtie-build hg18.fa hg18
```

- Results: several ebwt files.
- Tips: the index files can be stored in a common place and shared among colleagues.

Use bowtie: alignment

- Test whether it works:

```
bowtie -c hg18 GGTATATGCACAAAATGAGATGCTTGCTTA
```

- Align a read files

```
bowtie -v 3 -f hg18 reads.fa reads.map
```

bowtie: commonly used parameters

- Input file format:
  - -q: query input files are FASTQ .fq/.fastq (default)
  - -f: query input files are (multi-)FASTA .fa/.mfa
  - -r: query input files are raw one-sequence-per-line
- Aligment:
  - -v: allowing v mismatches.
  - -5: ignoring some based from 5’ end.
  - -3: ignoring some based from 3’ end.
- Output format:
  - -S: output in SAM (sequence alignment map)format.
- Example: input is a single fa file, allowing 3 mismatches, ignore 5 bases from 3’ end, output in SAM format:

```
bowtie -v 3 -3 5 -S hg18 reads.fa reads.sam
```

Output from bowtie

- SAM format

```
:@HD     VN:1.0  SO:unsorted
@SQ     SN:phage        LN:5386
@PG     ID:Bowtie       VN:0.12.7       CL:"bowtie -v 3 -f -S phage reads.fa reads.sam"
5_143_428_832   4       *       0       0       *       *       0       0       GATATTGTAGCATAACGCAACTTGGGAGGTGAGCTT    IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII    XM:i:0
5_143_984_487   0       phage   3948    255     36M     *       0       0       GTTTTCATGCCTCCAAATCTTGGAGGCTTTTTTATG    IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII    XA:i:0  MD:Z:36 NM:i:0
5_143_963_690   0       phage   3503    255     36M     *       0       0       GGTATATGCACAAAATGAGATGCTTGCTTATCAACA    IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII    XA:i:0  MD:Z:36 NM:i:0
5_143_957_461   16      phage   3903    255     36M     *       0       0       TTGTCTAGGAAATAACCGTCAGGATTGACACCCTCC    IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII    XA:i:0  MD:Z:36 NM:i:0
5_143_808_403   0       phage   4122    255     36M     *       0       0       GATAACCGCATCAAGCTCTTGGAAGAGATTCTGTCT    IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII    XA:i:0  MD:Z:36 NM:i:0
5_143_986_385   4       *       0       0       *       *       0       0       GATGCTGAAGGAACTTGGTAAAATTTATCTGGAGAA    IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII    XM:i:0
5_143_981_626   16      phage   1522    255     36M     *       0       0       TCCTCCTGAGACTGAGCTTTCTCGCCAAATGACGAC    IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII    XA:i:0  MD:Z:36 NM:i:0
5_143_470_717   16      phage   2061    255     36M     *       0       0       ATGCGCCTTCGTATGTTTCTCCTGCTTATCACCTTC    IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII    XA:i:0  MD:Z:36 NM:i:0
5_143_992_626   4       *       0       0       *       *       0       0       GCCCAGAAGGGGCGGTTAAATGGTTTTTGGAGAAAG    IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII    XM:i:0
5_143_400_771   0       phage   3816    255     36M     *       0       0       GATATTTTTCATGGAATTGATAAAGCTGTTGCCGAT    IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII    XA:i:1  MD:Z:14T21
NM:i:1
5_143_962_110   16      phage   5074    255     36M     *       0       0       AATGGAACAACTCACTAAAAACCAAGCTGTCGCTAC    IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII    XA:i:0  MD:Z:36 NM:i:0
5_143_774_100   0       phage   3810    255     36M     *       0       0       GTGGTTGATATTTTTCATGGTATTGATAAAGCTGTT    IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII    XA:i:0  MD:Z:36 NM:i:0
```

- Bowtie format

```
5_143_961_681   +       phage   4397    GCTGCTGAACGCCCTCTTAAGGATATTCGCGATGAG    IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII    0
5_143_996_500   +       phage   2537    GGTTAATGCTGGTAATGGTGGTTTTCTTCATTTCAT    IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII    0       32:G>T
5_143_468_916   +       phage   339     GGATTACTATCTGAGTCCGATGCTGTTCAACCACTA    IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII    0
5_143_972_467   +       phage   3021    GTGGCATTCAAGGTGATGTGCTTGCTACCGATAACA    IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII    0
5_143_953_471   -       phage   5017    ATACGTTAACAAAAAGTCAGATATGGACCTTGCTGC    IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII    0
5_143_687_97    +       phage   1287    GACTGTTAACACTACTGGTTATATTGACCATGCCAC    IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII    0       34:G>A
5_143_620_93    +       phage   4463    GATGAGTGTTCAAGATTGCTGGAGGCCTCCACTATG    IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII    0
5_143_766_307   +       phage   3024    GCATTCTAGGCGATGTGCTTGCTACCGTTAACAATA    IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII    0       6:A>T,10:T>C,27:A>T
```

Bioconductor alignment packages

- Rsubread, Rbowtie/Rbowtie2, QuasR, etc.
- Example code for Rsubread:

```
library('Rsubread')

## build reference
buildindex(basename="reference_index",
           reference=”ref.fa")
## align
align.stat <- align(index="reference_index",
                    readfile1="reads.fa",
                    output_file="alignResults.BAM",
                    type="dna")
```

Once the reads are aligned

- Downstream analyses depend on purpose.
  - We will cover the analyses for RNA-seq, ChIP-seq, and BS-seq in next several lectures.
- Often one wants to manipulating and visualizing the alignment results. There are several useful tools:
  - file manipulating (format conversion, counting, etc.): samtools/Rsamtools, BEDTools, bamtools, IGV tools.
  - Visualizing: samtools (text version), IGV (Java GUI).

samtools

- samtools provide various utilities for manipulating alignments in the SAM format, including sorting, merging, indexing and generating alignments in a per-position format.
- Command line driven, meaning one needs to type command in a terminal window.
  - Installation could be tricky. Needs to install extra tools on Windows or Mac, such as Cygwin and perl on Windows and Xcode on Mac.
- Main functionalities:
  - view: SAM<->BAM conversion
  - sort: sort alignment file
  - mpileup: multi-way pileup
  - depth: compute the coverage depth
  - tview: text alignment viewer
  - index: index alignment

samtools: generate sorted, indexed bam files

- BAM file: binary SAM. Smaller file sizes and faster operations.
- To convert from sam to bam:

```
samtools view -bS reads.sam > reads.bam
```

- Sort and index bam file. This sorts the reads by chromosome and position and makes subsequence analysis easier.

```
samtools sort reads.bam reads.sorted
samtools index reads.sorted.bam
```

Another useful software: BEDTools

- A set of commands to manipulate BED/GFF/VCF files.
- Conversion tools: `pairToBed(BAM), bamToBed, bedToBam`, etc.
- Counting tools: `coverageBed(BAM), windowBed (BAM)`
- Others: `sortBed, overlap`, etc.

Bioconductor package: `Rsamtools`

- Provide functions to import BAM files to R.
- There are many tools (samtools, BEDTools, bamtools) available to convert different formats (BED, SAM, fasta, fastq, etc.) to BAM.
- Read alignment results should always be saved in BAM format because they are smaller and faster.

Read in a BAM file

```
> bamFile="reads.sorted.bam"
> bam <- scanBam(bamFile)
> names(bam[[1]])
 [1] "qname"  "flag"   "rname"  "strand" "pos"    "qwidth" "mapq"   "cigar"
 [9] "mrnm"   "mpos"   "isize"  "seq"    "qual"
```

This gives the available information in the BAM file. One can specify what to read in (to save time and memory):

```
> what <-c("rname", "strand", "pos", "qwidth") ## fields to read in
> param <- ScanBamParam(what = what)
> bam <- scanBam(bamFile, param=param)[[1]]
> names(bam)
[1] "rname"  "strand" "pos" "qwidth"
> bam$pos[1:10]
 [1] 1 2 3 3 4 4 4 4 4 5
> bam$strand[1:10]
 [1] + + + + - - - - - +
Levels: + - *
```

Summarize the read counts

- Remember each aligned read can be treated as a genomic interval. So the results from scanBam can be used to construct an GRanges object (of millions of intervals):

```
> GRange.reads=GRanges(seqnames=Rle(bam$rname),
 ranges=IRanges(bam$pos, width=bam$qwidth))
```

- Then it becomes very handy, for example, we can:
  - compute genome coverage:

```
> cc=coverage(IRange.reads)
```

  - count number of reads in intervals (such as genes):

```
> countOverlaps(genes, GRange.reads)
```

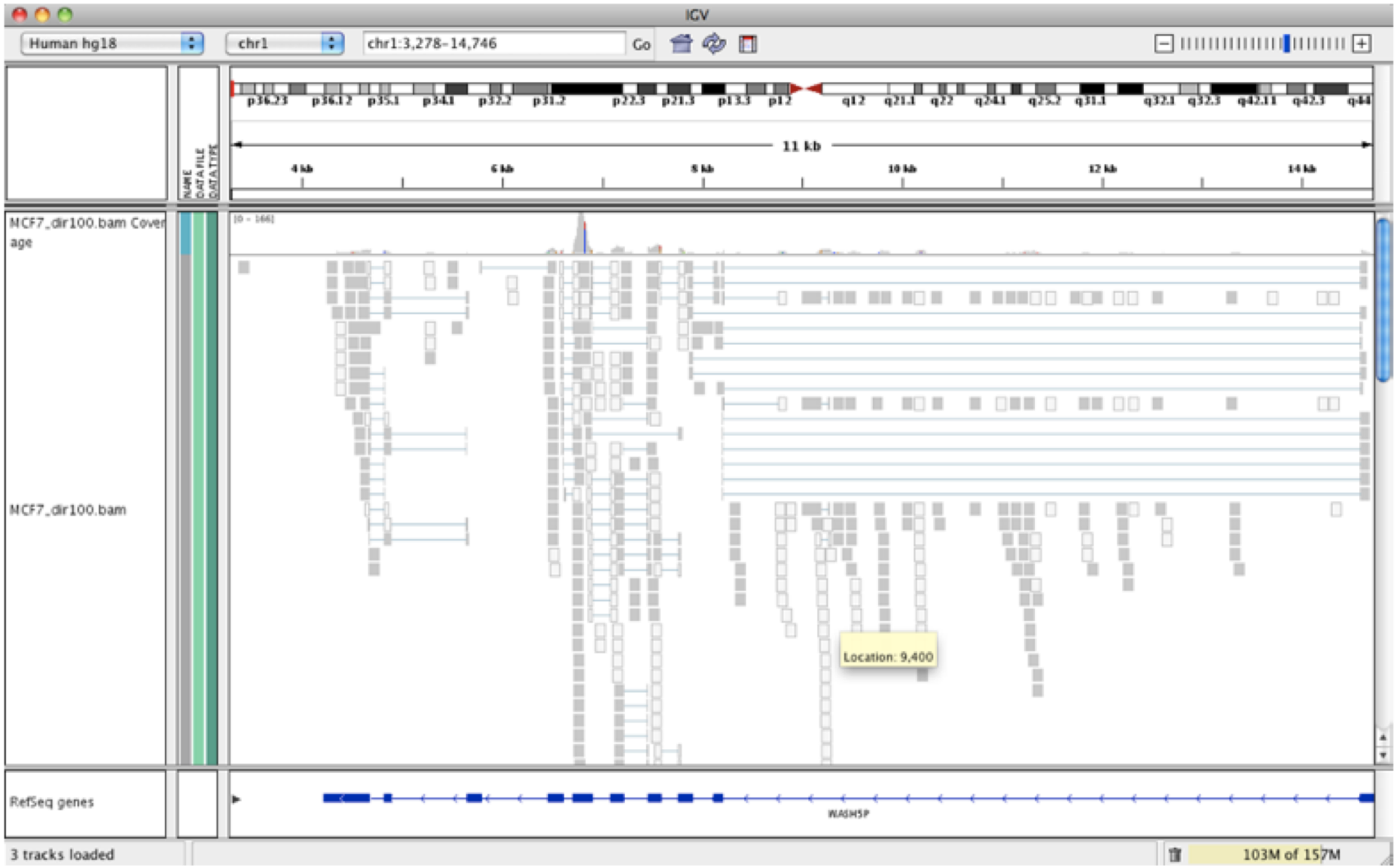
An example: obtaining RNA-seq reads mapped to exons and introns

```
library(GenomicRanges)
library(GenomicFeatures)
library(Rsamtools)
## get gene annotation, and extract exons/introns
refGene.hg18=makeTranscriptDbFromUCSC(genom="hg18",tablename="refGene")
ex=exonsBy(refGene.hg18, "tx")
intr=intronsByTranscript(refGene.hg18)

## read in RNA-seq BAM file
what=c("rname", "strand", "pos", "qwidth")
TSS.counts=NULL
param=ScanBamParam(what = what)
bam=scanBam("RNA-seq.bam", param=param)[[1]]
IRange.reads=GRanges(seqnames=Rle(bam$rname),
    ranges=IRanges(bam$pos, width=bam$qwidth))

## obtain counts
counts.exon=countOverlaps(ex, IRange.reads)
counts.intron=countOverlaps(intr, IRange.reads)
```

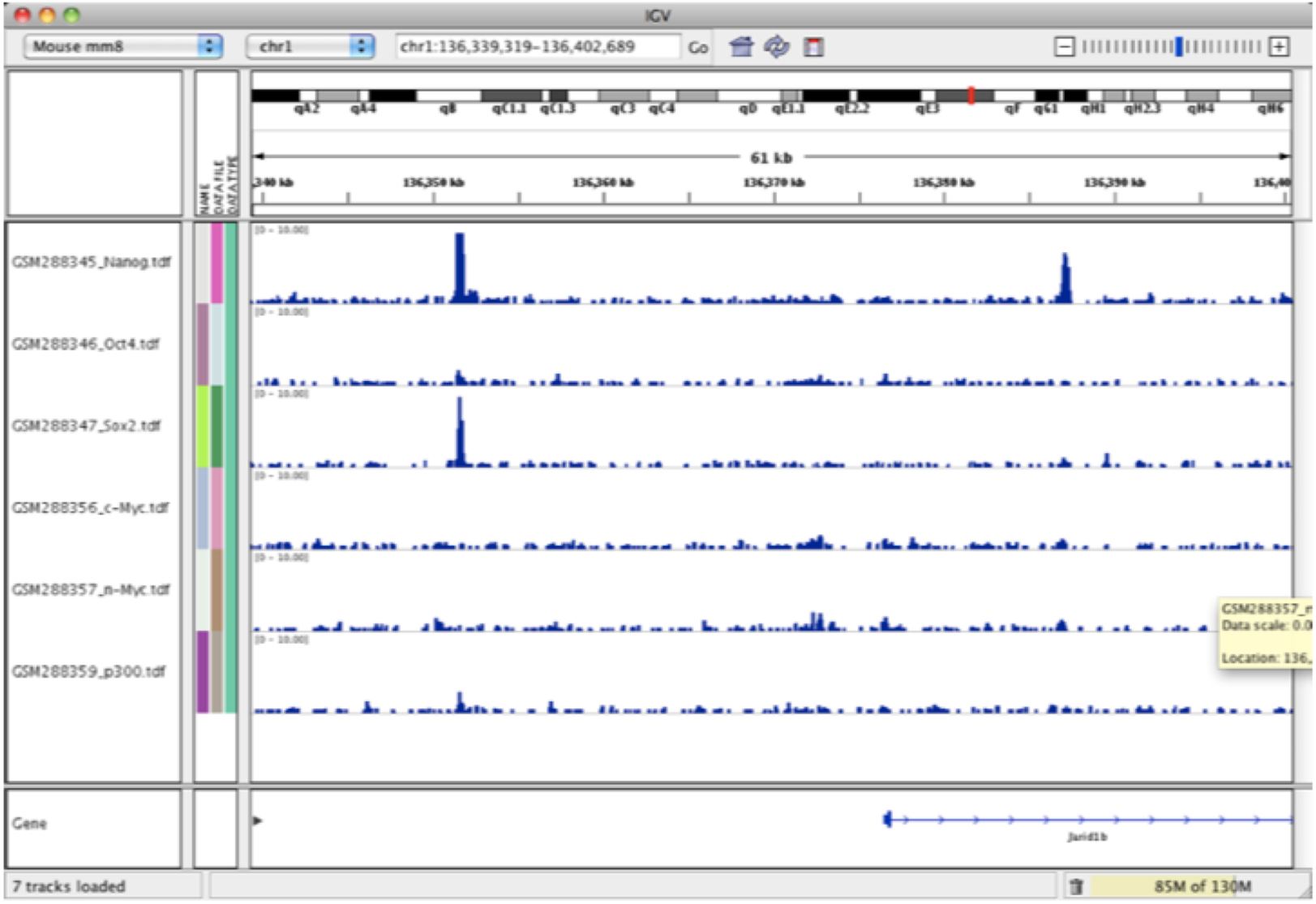
Visualization of sequencing data – Integrated Genome Viewer (IGV)

- "The **Integrative Genomics Viewer (IGV)** is a high-performance visualization tool for interactive exploration of large, integrated datasets. It supports a wide variety of data types including sequence alignments, microarrays, and genomic annotations."
- Written in Java and runs on all OS.
- Very versatile and fast.
- Ability to connect to data server and display some public data (from ENCODE, broad, etc.)

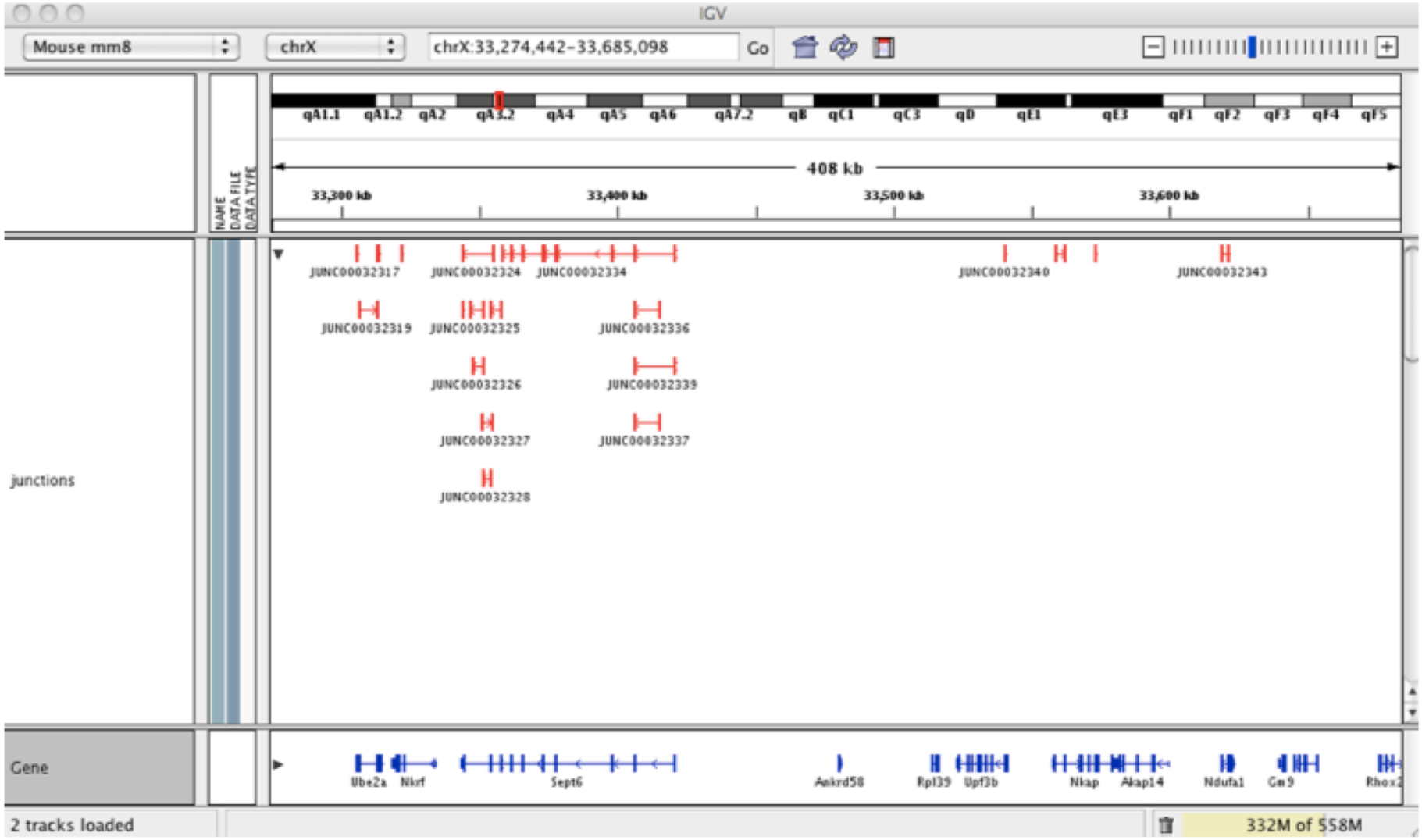
Aligned reads on IGV



ChIP-seq data on IGV



RNA-seq junction reads on IGV



Review

- We've covered
  - Basics of second generation sequencing: technology, data, application
  - Alignment using bowtie
  - Manipulation of alignment results using samtools
  - Import alignment into R using Rsamtools.
  - Visualization using IGV.