Low level Processing and Visualization of Genome Sequencing Data

GSND 5340Q, BMDA

W. Evan Johnson, Ph.D.
Professor, Division of Infectious Disease
Director, Center for Data Science
Rutgers University – New Jersey Medical School

2025-04-23

Sequencing data alignment

Next-Generation Sequencing (NGS) Data present new challenges:

- Map 'reads' to genome
- Call SNPs and variants from the reads
- Other Applications: RNA-seq, miRNAs, alternative splicing, ChIP-seq, BS-seq, RNA editing

Sequencing data alignment

Obstacles:

- Massive data size
- Repeat regions, rare variants

Goal: Develop an approach to put the puzzle together!

Mapping Algorithms

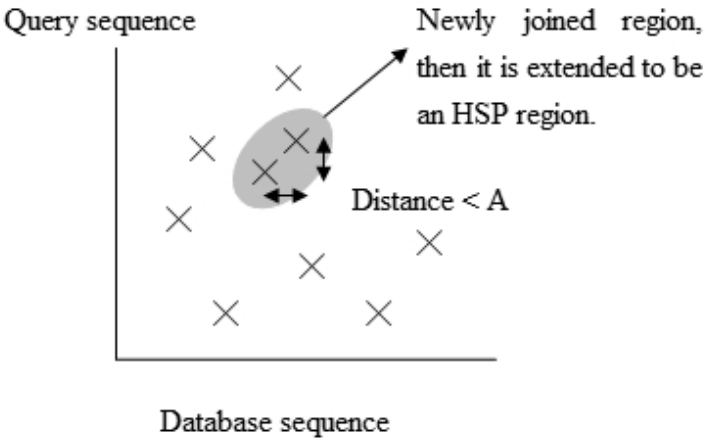
Alignment algorithms (not comprehensive – some of my favorites!)

- BLAST
- BLAT
- SOAP2
- Bowtie2
- BWA
- STAR
- Subread (Rsubread)
- GNUMAP
- NovoAlign
- Many others!!

Basic Local Alignment Search Tool (BLAST)

BLAST is a "heuristic" method that identifies and combined *High-scoring Segment Pairs (HSPs)* between two sequences. BLAST searches for high scoring sequence alignments between the query sequence and the existing sequences in the database using an approach that approximates the Smith-Waterman algorithm (described later).

Basic Local Alignment Search Tool (BLAST)

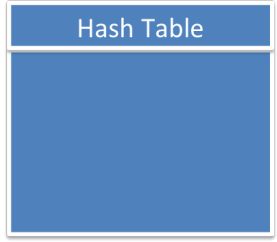


Basic Local Alignment Search Tool (BLAST)

BLAST can be found at: https://blast.ncbi.nlm.nih.gov/Blast.cgiBLAST. Try "BLASTING" the following sequences:

```
>NC_045512.2 Severe acute respiratory syndrome coronavirus 2 isolate Wuhan-Hu-1
ATTAAAGGTTTATACCTTCCCAGGTAACAAACCAACCAACTTTCGATCTCTTGTAGATCTGTTCTCTAAACGAACTTTAA

>NC_001802.1 Human immunodeficiency virus 1
GGTCTCTCTGGTTAGACCAGATCTGAGCCTGGGAGCTCTCTGGCTAACTAGGGAACCCACTGCTTAAGCCTCAATAAAGC

>NR_102810.2 Mycobacterium tuberculosis H37Rv 16S ribosomal RNA
TGTTTGGAGAGTTTGATCCTGGCTCAGGACGAACGCTGGCGGCGTGCTTAACACATGCAAGTCGAACGGA
AAGGTCTCTTCGGAGATACTCGAGTGGCGAACGGGTGAGTAACACGTGGGTGATCTGCCCTGCACTTCGG

>NR_041248.1 Bacillus anthracis strain ATCC 14578 16S ribosomal RNA
AGAGCTTGCTCTTATGAAGTTAGCGGCGGACGGGTGAGTAACACGTGGGTAACCTGCCCATAAGACTGGG
```

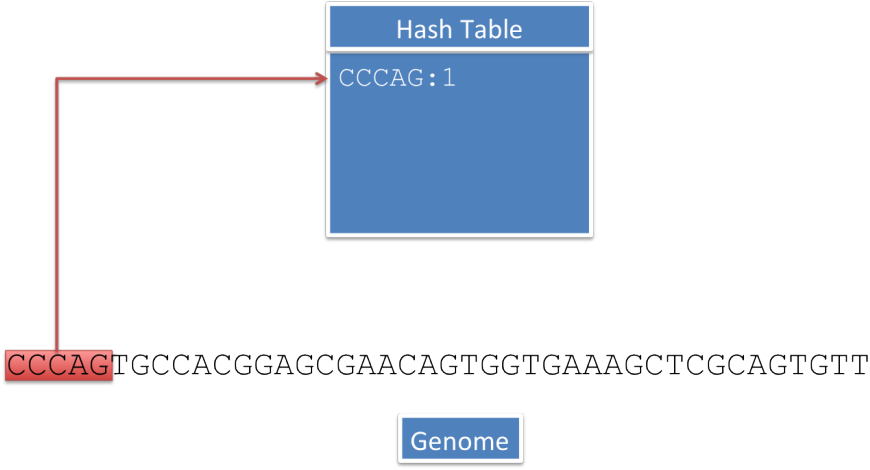
Creating a Hash Index Table



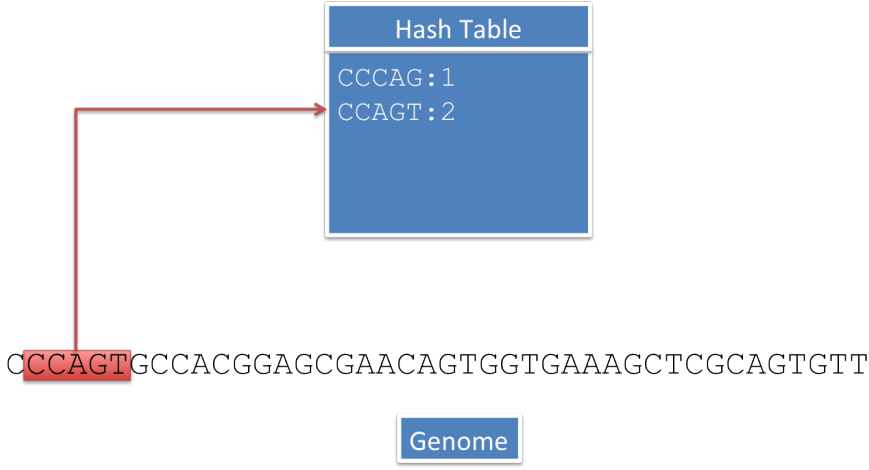
CCCAGTGCCACGGAGCGAACAGTGGTGAAAGCTCGCAGTGTT

Genome

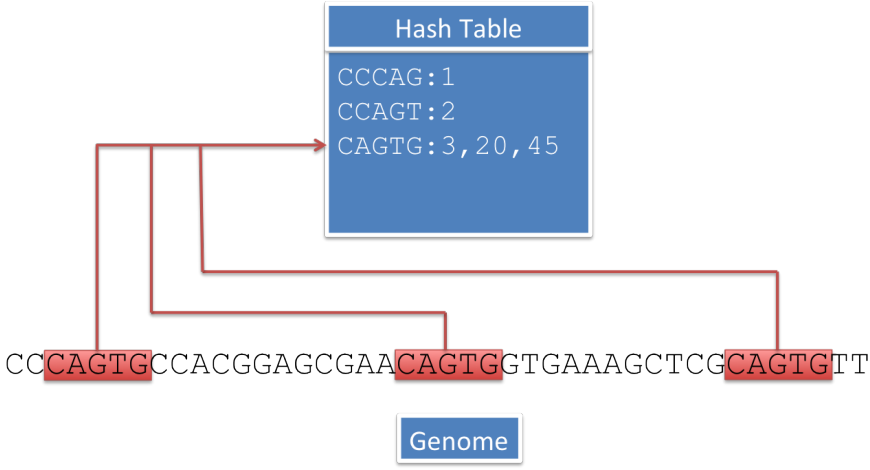
Creating a Hash Index Table



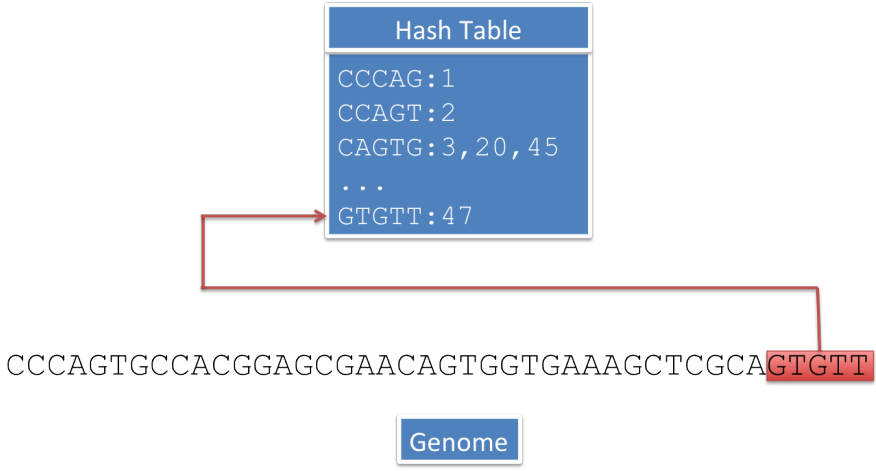
Creating a Hash Index Table



Creating a Hash Index Table



Creating a Hash Index Table



Read Lookup

Read

CAGTGCC

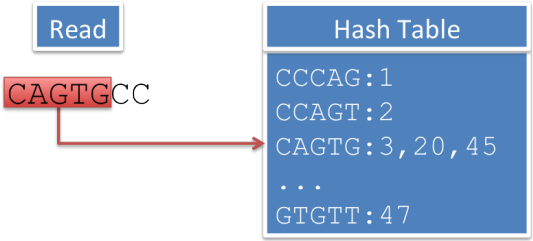
Hash Table

```
CCCAG:1
CCAGT:2
CAGTG:3,20,45
...
GTGTT:47
```

CCCAGTGCCACGGAGCGAACAGTGCTGAAAGCTCGCAGTGTT

Genome

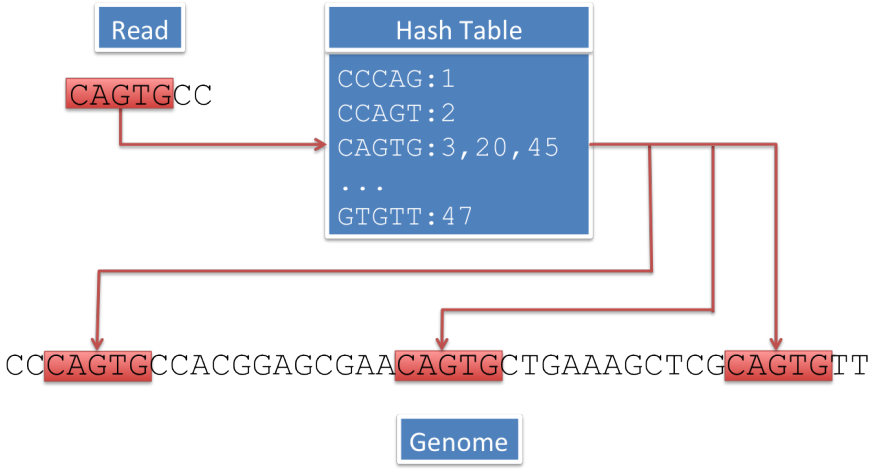
Read Lookup



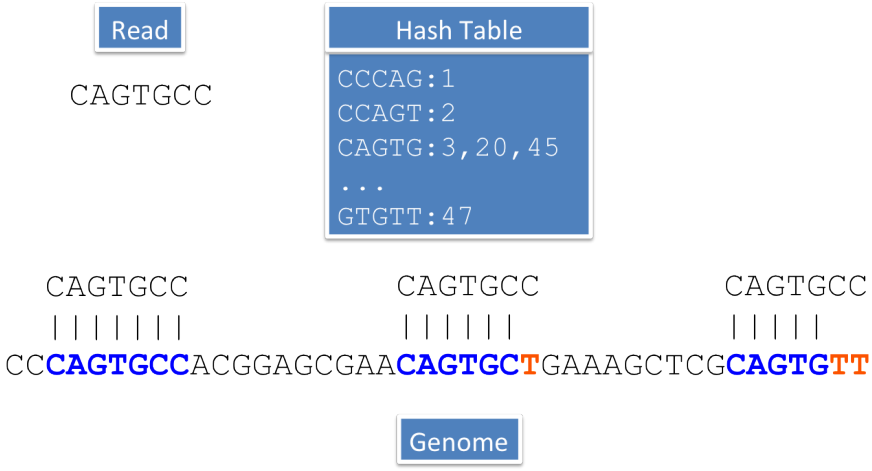
CCCAGTGCCACGGAGCGAACAGTGCTGAAAGCTCGCAGTGTT

Genome

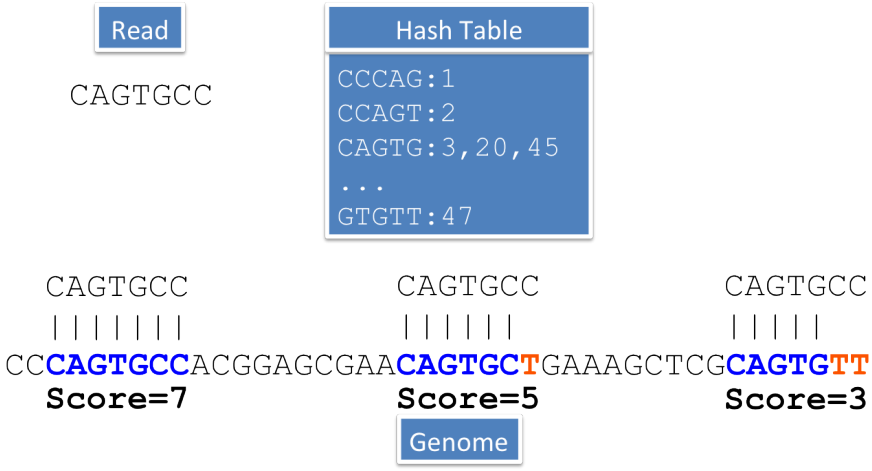
Read Lookup


Read
CAGTGCC
Hash Table
CCCAG:1
CCAGT:2
CAGTG:3,20,45
...
GTGTT:47
CCCAGTGCCACGGAGCGAACAGTGCTGAAAGCTCGCAGTGTT
Genome

Read Lookup


Read
CAGTGCC
Hash Table
CCCAG:1
CCAGT:2
CAGTG:3,20,45
...
GTGTT:47
CAGTGCC
CAGTGCC
CAGTGCC
CCCAGTGCCACGGAGCGAACAGTGCTGAAAGCTCGCAGTGTT
Genome

Read Lookup


Read
CAGTGCC
Hash Table
CCCAG:1
CCAGT:2
CAGTG:3,20,45
...
GTGTT:47
CAGTGCC
CAGTGCC
CAGTGCC
CCCAGTGCCACGGAGCGAACAGTGCTGAAAGCTCGCAGTGTT
Score=7
Score=5
Score=3
Genome

Needleman-Wunsch

T A C A T C

T T C A - C

		t	a	c	a	t	c
t							
t							
c							
a							
c							

- Allows for mismatches and gaps
- Scoring function: +1 match; -1 for mismatch; -2 for gap
- Produces maximum likelihood alignment

Needleman-Wunsch

T A C A T C

T T C A - C

		t	a	c	a	t	c
	0	-2	-4	-6	-8	-10	-12
t	-2						
t	-4						
c	-6						
a	-8						
c	-10						

- Allows for mismatches and gaps
- Scoring function: +1 match; -1 for mismatch; -2 for gap
- Produces maximum likelihood alignment

Needleman-Wunsch

```
T A C A T C
|
T T C A - C
```

		t	a	c	a	t	c
	0	-2	-4	-6	-8	-10	-12
t	-2	1					
t	-4						
c	-6						
a	-8						
c	-10						

- Allows for mismatches and gaps
- Scoring function: +1 match; -1 for mismatch; -2 for gap
- Produces maximum likelihood alignment

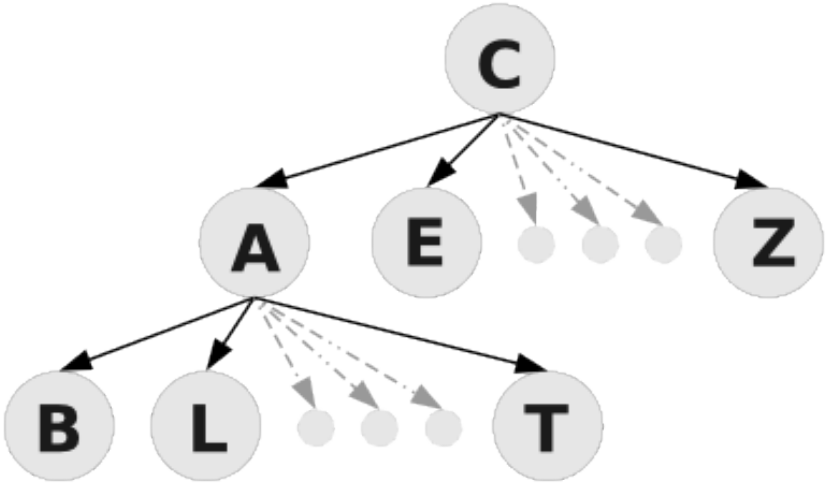
Needleman-Wunsch

```
T A C A T C
|   | |   |
T T C A - C
```

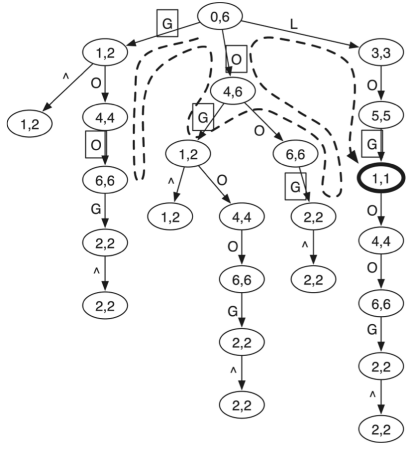
		t	a	c	a	t	c
	0	-2	-4	-6	-8	-10	-12
t	-2	**1**	-1	-3	-5	-7	-9
t	-4	-1	**0**	-2	-4	-4	-6
c	-6	-3	-2	**1**	-1	-3	-3
a	-8	-5	-2	-1	**2**	**0**	-2
c	-10	-7	-4	-1	0	1	**1**

- Allows for mismatches and gaps
- Scoring function: +1 match; -1 for mismatch; -2 for gap
- Produces maximum likelihood alignment

Prefix Trees



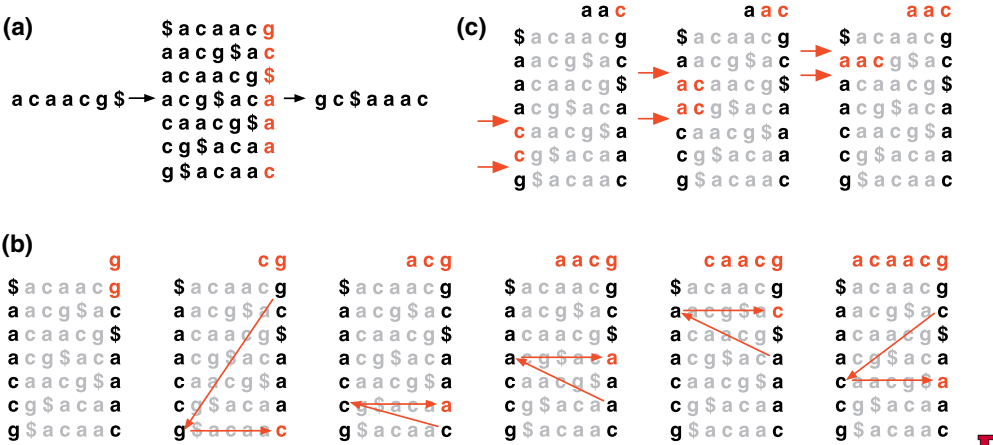
Prefix Trees



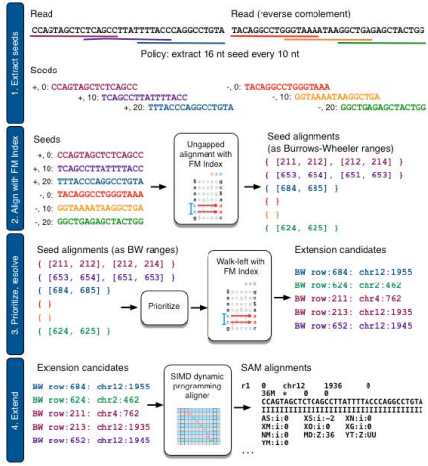
Li H , and Durbin R Bioinformatics 2009;25:1754-1760

Burrows-Wheeler Transformation (BWA, Bowtie)

Burrows-Wheeler Transformation (BWA, Bowtie)



Hybrid Methods (BWA-SW, Bowtie2)



Read alignments are stored in SAM/BAM/CRAM

- BAM and CRAM are compressed SAM files
- One line per alignment
- Use `samtools` to view, sort, merge, concatenate, index, get statistics, etc, on alignment files.

SAM/BAM/CRAM Format – Header

The *header* lines start with @:

- @HD = header definition
- @SQ = a sequence in the reference file you used, followed by how long it was and it's comment (from the reference file)
- @RG = read groups you assigned while mapping the reads
- @PG = programs used to obtain this bam, in order

SAM/BAM/CRAM Format – Body

Each alignment has 11 mandatory fields:

Col	Field	Type	Regexp/Range	Brief description
1	QNAME	String	`[!-?A-~]{1,255}`	Query template NAME
2	FLAG	Int	$[0,2^{16}-1]$	bitwise FLAG
3	RNAME	String	`\*|[!-()+-<>-~][!-~]*`	Reference sequence NAME
4	POS	Int	$[0,2^{29}-1]$	1-based leftmost mapping POSition
5	MAPQ	Int	$[0,2^{8}-1]$	MAPping Quality
6	CIGAR	String	`\*|([0-9]+[MIDNSHPX=])+`	CIGAR string
7	RNEXT	String	`\*|=|[!-()+-<>-~][!-~]*`	Ref. name of the mate/next segment
8	PNEXT	Int	$[0,2^{29}-1]$	Position of the mate/next segment
9	TLEN	Int	$[-2^{29}+1,2^{29}-1]$	observed Template LENgth
10	SEQ	String	`\*|[A-Za-z=.]+`	segment SEQuence
11	QUAL	String	`[!-~]+`	ASCII of Phred-scaled base QUALity+33

SAM/BAM/CRAM Format – Flag

The flag is the summation of the following binary attributes:

Binary (Decimal)	Hex	Description
00000000001 (1)	0x1	Is the read paired?
00000000010 (2)	0x2	Are both reads in a pair mapped "properly" (i.e., in the correct orientation with respect to one another)?
00000000100 (4)	0x4	Is the read itself unmapped?
00000001000 (8)	0x8	Is the mate read unmapped?
00000010000 (16)	0x10	Has the read been mapped to the reverse strand?
00000100000 (32)	0x20	Has the mate read been mapped to the reverse strand?
00001000000 (64)	0x40	Is the read the first read in a pair?
00010000000 (128)	0x80	Is the read the second read in a pair?
00100000000 (256)	0x100	Is the alignment not primary? (A read with split matches may have multiple primary alignment records.)
01000000000 (512)	0x200	Does the read fail platform/vendor quality checks?
10000000000 (1024)	0x400	Is the read a PCR or optical duplicate?

Example flags:

Single end read mapped to + strand: 0 (no flags apply)

Single end read mapped to - strand: 16

Unmapped, paired end first mate read with unmapped mate: 69 = 1 + 4 + 8 + 64

SAM/BAM/CRAM Format – Example

```
@HD     VN:1.0  SO:coordinate
@SQ     SN:chr20        LN:64444167
@PG     ID:TopHat       VN:2.0.14       CL:/srv/dna_tools/tophat/tophat -N 3 --read-edit-dist 5 --read-rea
lign-edit-dist 2 -i 50 -I 5000 --max-coverage-intron 5000 -M -o out /data/user446/mapping_tophat/index/chr
20 /data/user446/mapping_tophat/L6_18_GTGAAA_L007_R1_001.fastq
HWI-ST1145:74:C101DACXX:7:1102:4284:73714        16      chr20   190930  3       100M    *       0       0
       CCGTGTTTAAAGGTGGATGCGGTCACCTTCCCAGCTAGGCTTAGGGATTCTTAGTTGGCCTAGGAAATCCAGCTAGTCCTGTCTCTCAGTCCCCCCTCT
C    BBDCCDDCCDDDDCDDDDDDCDCCCDBC?DDDDDDDDDDDDDDDCCDCDDDDDDDDDDCCCCEDDDC?DDDDDDDDDDDDDDDDDDDDDBDHFFFFDC@@
    AS:i:-15        XM:i:3  XO:i:0  XG:i:0  MD:Z:55C20C13A9 NM:i:3  NH:i:2  CC:Z:=  CP:i:55352714   HI:i:0
HWI-ST1145:74:C101DACXX:7:1114:2759:41961        16      chr20   193953  50      100M    *       0       0
       TGCTGGATCATCTGGTTAGTGGCTTCTGACTCAGAGGACCTTCGTCCCCTGGGGCAGTGGACCTTCCAGTGATTCCCCTGACATAAGGGGCATGGACGA
G    DCDDDDEDDDDDDDCDDDDDDDCCCDDDCDDDDDEEC>DFFFEJJJJJIGJJJJIHGBHHGJIJJJJJJGJJJIJJJJJIHJJJJJJHHHHHFFFFFCCC
    AS:i:-16        XM:i:3  XO:i:0  XG:i:0  MD:Z:60G16T18T3 NM:i:3  NH:i:1
HWI-ST1145:74:C101DACXX:7:1204:14760:4030        16      chr20   270877  50      100M    *       0       0
       GGCTTTATTGGTAAAAAAGGAATAGCAGATTTAATCAGAAATTCCCACCTGGCCCAGCAGCACCAACCAGAAAGAAGGGAAGAAGACAGGAAAAAACCA
C    DDDDDDDDDCCDDDDDDDDDDEEEEEEEFFFEFFEGHHHHFGDJJIHJJIJIJJJIIIIGGFJJIHIIIIJJJJJJIGHHFAHGFHJHFGGHFFFDD@BB
    AS:i:-11        XM:i:2  XO:i:0  XG:i:0  MD:Z:0A85G13    NM:i:2  NH:i:1
HWI-ST1145:74:C101DACXX:7:1210:11167:8699        0       chr20   271218  50      50M4700N50M     *       0
       0       GTGGCTCTTCCACAGGAATGTTGAGGATGACATCCATGTCTGGGGTGCACTTGGGTCTCCGAAGCAGAACATCCTCAAATATGACCTCTCG
```

SAM/BAM/CRAM Format

- Use `samtools` view to see the content of a SAM/BAM/CRAM file.
- Always sort and index them
- Use `samtools view -f` (include) and `-F` (exclude) to filter by flags
- Use `samtools view -q` to filter by quality.
- More on `samtools` later!!

BWA Example (OnDemand Amarel Desktop)

```
# load bwa
module load bwa

# Index the genome:
bwa index genomefile.fa

# Align the reads:
bwa mem genomefile.fa myfastqfile.fq > bwaalign.sam

# Note: for use with gatk (will learn later) bwa mem
# should include a `read group' header as well, e.g.:
-R "@RG\tID:group1\tSM:sam1\tPL:illumina\tLB:lib1\tPU:unit1"
```

Bowtie2 Example (OnDemand Amarel Desktop)

```
# Load Bowtie2
module load bowtie2

# Index the genome:
bowtie2-build genomefile.fa output/genomefile

# Align the reads:
bowtie2 -x output/genomefile -U myfastqfile.fq \
    -S bowtiealign.sam
```

RSubread Example (OnDemand RStudio)

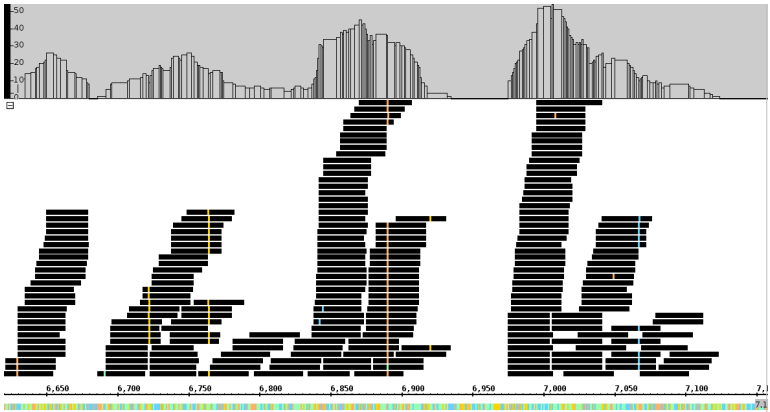
```r
# Install the Bioconductor installer
install.packages("BiocManager")

# Install Rsubread
BiocManager::install("Rsubread")

# Index the genome:
buildindex("genomefile", "genomefile.fa")

# Align the reads:
align("genomefile", "myfastqfile.fq",
      output_file="myfastq.bam", nthreads = 4)
```

Visualization of Aligned Sequencing Data

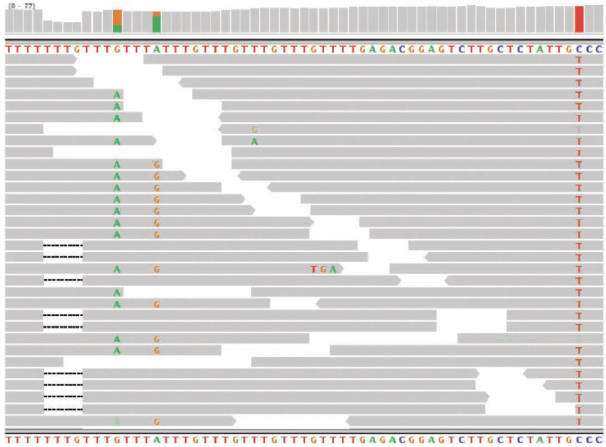
Options for data visualization

- UCSC Genome Browser
- IGB (http://bioviz.org/igb/)
- IGV (http://www.broadinstitute.org/igv/)
- SAMtools

IGB Example


50
40
30
20
10
6,650
6,700
6,750
6,800
6,850
6,900
6,950
7,000
7,050
7,100

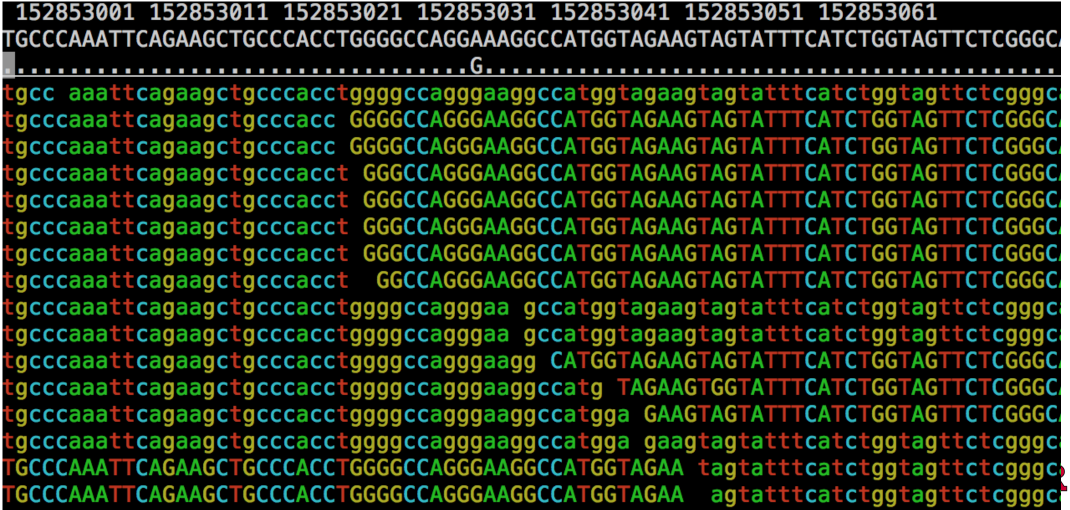
IGV Example



DePristo et al. *Nature Genetics 2011; 43: 491-8.*

WEJ LAB

SAMtools



Center for Data Science

SAMtools Example (Amarel Desktop)

```
# Load SAMtools
module load samtools

# Convert SAM file to BAM format:
samtools view -bS myalignments.sam > myalignments.bam

# Sort the BAM file:
samtools sort myalignments.bam -o myalignments.sorted.bam

# Index the BAM file:
samtools index myalignments.sorted.bam

# View BAM file in SAMtools:
samtools tview myalignments.sorted.bam genomefile.fa
```

SAMtools Example

SAMtools commands:

- The one command to remember: '?'
- g: go to a specific location (i.e. chrX:152852988 or chrX_149913753:2939235)
- m,b: mapping quality, base quality
- n: color by nucleotide
- '.': dot/base view
- r: read name
- q: quit SAMtools

SAM/BAM/CRAM Format – Conversions

```
# Convert SAM to BAM
samtools view -bS in.sam > out.bam

# Convert BAM to SAM
samtools view -ho out.sam in.bam

# Convert BAM to fastq
samtools bam2fq in.bam > out.fastq
```

RSamtools

`Rsamtools` is a very useful (although somewhat limited) version of Samtools available in R:

```
# Install Rsamtools
BiocManager::install("Rsamtools")

# Convert SAM to BAM
asBam(in.sam, out.bam)
```

You can also index and sort .bam files, as well as extract alignments from a .bam file (very useful!).

Other Data Formats

Standard format for keeping tables

field1	field2	field3	...
...	...	...	...

Fields (columns) separated by a character on each line:

- Comma (or Character) Separated Vector (CSV)
- Tab Separated Vector (TSV)
- Some interpreters take any space (space or tab) as a separator (such as `awk`, `cut`).
- Some have column name as first row (header), some don't

Genomic regions

- A region is defined by three required fields:
  - sequence name (e.g. chromosome)
  - start coordinate
  - end coordinate
- Define regions of interest: introns, exons, genes, etc.
- Additional information saved as fields after the first three.
- Three standard tab-separated formats: BED, GFF, GTF No headers

BED Format

Mandatory fields:

1. chrom - Name of the chromosome/scaffold/reference sequence
2. chromStart - 0-based starting position of the feature on chrom
3. chromEnd - Ending position of the feature in the chromosome or scaffold.

The chromEnd base is not included in the display of the feature.

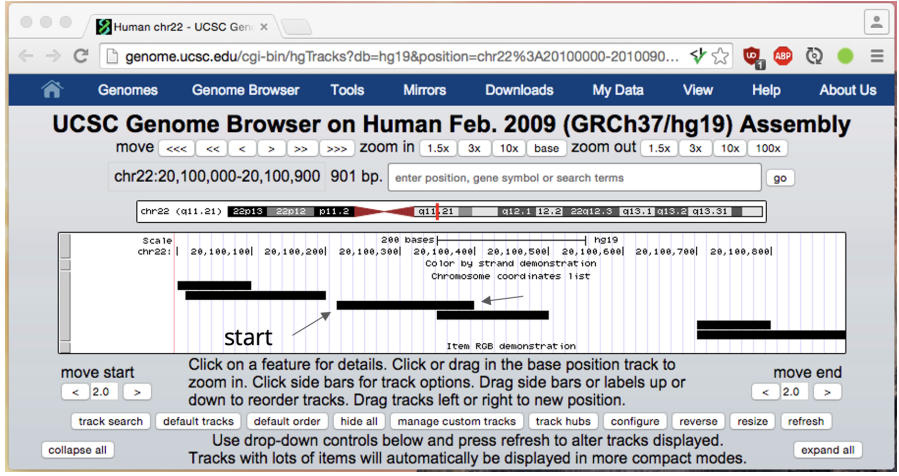
For example, the first 100 bases of a chromosome are defined as:

- chromStart=0
- chromEnd=100
- span the bases numbered 0-99

BED Format

chr1	11873	14409	uc001aaa.3	0	+	11873	11873
chr1	11873	14409	uc010nxr.1	0	+	11873	11873
chr1	11873	14409	uc010nxq.1	0	+	12189	13639
chr1	14361	16765	uc009vis.3	0	-	14361	14361
chr1	14361	19759	uc009vit.3	0	-	14361	14361
chr1	14361	19759	uc009viu.3	0	-	14361	14361
chr1	14361	19759	uc001aae.4	0	-	14361	14361
chr1	14361	29370	uc001aah.4	0	-	14361	14361
chr1	14361	29370	uc009vir.3	0	-	14361	14361
chr1	14361	29370	uc009viq.3	0	-	14361	14361
chr1	14361	29370	uc001aac.4	0	-	14361	14361
chr1	14406	29370	uc009viv.2	0	-	14406	14406
chr1	14406	29370	uc009viw.2	0	-	14406	14406
chr1	15602	29370	uc009vix.2	0	-	15602	15602
chr1	15795	18061	uc009vjd.2	0	-	15795	15795
chr1	16606	29370	uc009viy.2	0	-	16606	16606
chr1	16606	29370	uc009viz.2	0	-	16606	16606
chr1	16857	17751	uc009vjc.1	0	-	16857	16857
chr1	16857	19759	uc001aai.1	0	-	16857	16857

BED Format


Human chr22 - UCSC Gen
genome.ucsc.edu/cgi-bin/hgTracks?db=hg19&position=chr22%3A20100000-2010090...
Genomes
Genome Browser
Tools
Mirrors
Downloads
My Data
View
Help
About Us
UCSC Genome Browser on Human Feb. 2009 (GRCh37/hg19) Assembly
move
zoom in
zoom out
chr22:20,100,000-20,100,900
901 bp.
enter position, gene symbol or search terms
go
chr22 (q11.21)
Scale
chr22:
200 bases
hg19
20,100,100| 20,100,200| 20,100,300| 20,100,400| 20,100,500| 20,100,600| 20,100,700| 20,100,800|
Color by strand demonstration
Chromosome coordinates list
start
Item RGB demonstration
move start
move end
Click on a feature for details. Click or drag in the base position track to zoom in. Click side bars for track options. Drag side bars or labels up or down to reorder tracks. Drag tracks left or right to new position.
track search
default tracks
default order
hide all
manage custom tracks
track hubs
configure
reverse
resize
refresh
Use drop-down controls below and press refresh to alter tracks displayed.
Tracks with lots of items will automatically be displayed in more compact modes.
collapse all
expand all

BED Format

Optional fields:

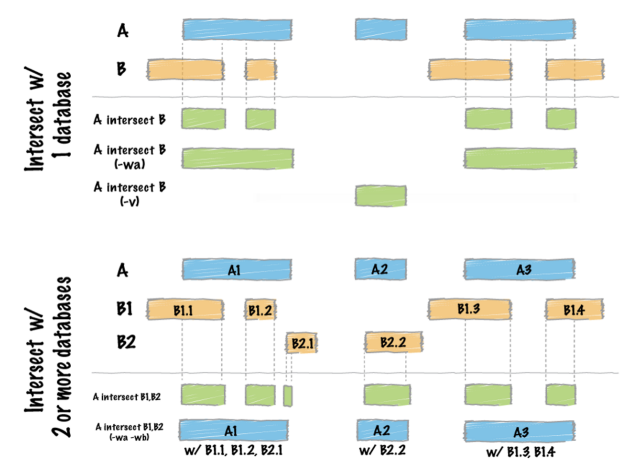
4. Name
5. Score
6. Strand
7. 7-12. Display options (thick starts and end, color, blocks...) to control the view on the genome browser

bedtools

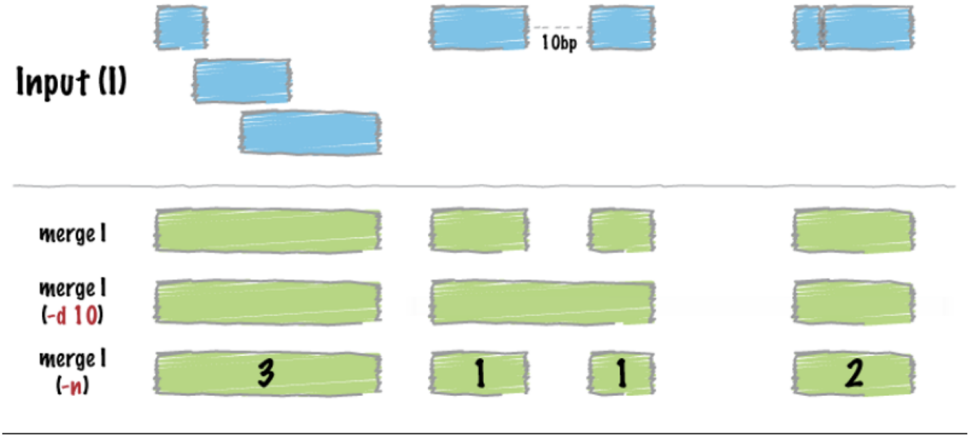
http://bedtools.readthedocs.io/

- sort (sort bed files)
- Intersect (get intersections of bed files)
- merge
- coverage
- overlap
- substract
- ...

bedtools intersect



bedtools merge



GFF - General features

9 mandatory fields, tab separated:

1. seqname - The name of the sequence. Must be a chromosome or scaffold.
2. source - The program that generated this feature.
3. feature - The name of this type of feature (e.g. gene, exon, etc).
4. start - The starting position of the feature in the sequence (1-based)
5. end - The ending position of the feature (inclusive).
6. score - A score between 0 and 1000.
7. strand - Valid entries include "+", "-", or "." (for don't know/don't care).
8. frame - If the feature is a coding exon, frame should be a number between 0-2 that represents the reading frame of the first base. If the feature is not a coding exon, the value should be ".".
9. group - All lines with the same group are linked together into a single item

Gene Information

GTF (Gene Transfer Format, GTF2.2)

- Extension to GFF2, backwards compatible
- First eight GTF fields are the same as GFF
- *feature field* is the same as GFF, has controlled vocabulary:
  - *gene, transcript, exon, CDS, 5UTR, 3UTR, inter, inter_CNS, and intron_CNS, etc*
- group field expanded into a list of attributes (i.e. key/value pairs)

The attribute list must begin with the one mandatory attribute: - gene_id value - A globally unique identifier for the genomic source of the sequence

GTF format

```
##description: evidence-based annotation of the human genome (GRCh38), version 27 (Ensembl 90)
##provider: GENCODE
##contact: gencode-help@sanger.ac.uk
##format: gtf
##date: 2017-08-01
chr1    HAVANA  gene    923928  944581  .       +       .       gene_id "ENSG00000187634.11"; gene_type
"protein_coding"; gene_name "SAMD11"; level 2; havana_gene "OTTHUMG00000040719.10";
```

- seqname: chr1
- source: HAVANA
- feature: gene
- start: 923928
- end: 944581
- score: . (no score)
- strand: +
- frame: . (not coding feature)
- attributes:
  - gene_id: ENSG00000187634.11
  - gene_type: protein_coding
  - gene_name: SAMD11
  - level: 2
  - havana_gene: OTTHUMG00000040719.10

GFF/GTF encodes relationships

- Features are hierarchical, e.g.:
  - A gene has 1 or more transcripts
  - A transcript has 1 or more exons
  - An exon is a coding sequence (CDS)
- Relationships encoded in attributes

```
##description: evidence-based annotation of the human genome (GRCh38), version 27 (Ensembl 90)
##provider: GENCODE
##contact: gencode-help@sanger.ac.uk
##format: gtf
##date: 2017-08-01
chr1    HAVANA  gene        923928  944581  .       +       .       gene_id "ENSG00000187634.11"; gene_type
"protein_coding"; gene_name "SAMD11"; level 2; havana_gene "OTTHUMG00000040719.10";

chr1    HAVANA  transcript  923928  939291  .       +       .       gene_id "ENSG00000187634.11";
transcript_id "ENST00000420190.6"; gene_type "protein_coding"; gene_name "SAMD11"; transcript_type
"protein_coding"; transcript_name "SAMD11-203";

chr1    HAVANA  exon        923928  924948  .       +       .       gene_id "ENSG00000187634.11";
transcript_id "ENST00000420190.6"; gene_type "protein_coding"; gene_name "SAMD11"; transcript_type
"protein_coding"; transcript_name "SAMD11-203"; exon_number 1; exon_id "ENSE00001637883.2";

chr1    HAVANA  CDS         924432  924948  .       +       0       gene_id "ENSG00000187634.11";
transcript_id "ENST00000420190.6"; gene_type "protein_coding"; gene_name "SAMD11"; transcript_type
"protein_coding"; transcript_name "SAMD11-203"; exon_number 1; exon_id "ENSE00001637883.2";
```

SNP and Variant Calling

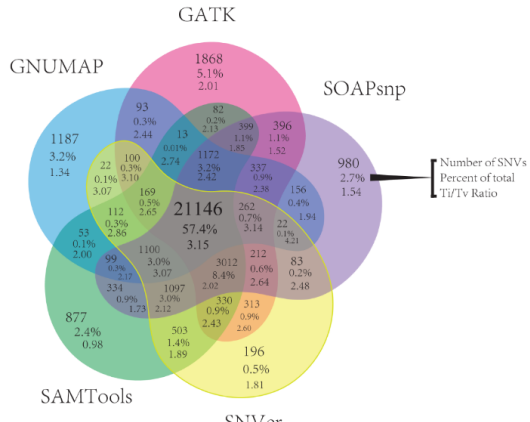
Methods for SNP Calling

Methods for SNP calling:

- Mapper/callers: MAQ, SOAPsnp, GNUMAP, Crossbow (Bowtie)
- Callers: SAMtools (mpileup), GATK (HaplotypeCaller, Mutect2), FreeBayes, others

Inconsistencies Among Aligners

Low concordance of variant-calling pipelines (O'Rawe, *Genome Med*, 2013)



SAMtools Example

Multiple sample SNP calling:

```
samtools mpileup -f genomefile.fa \
  myalignments.sorted.bam > myalignments.vcf
```

VCF files

8 fixed columns: #CHROM, POS, ID, REF, ALT, QUAL, FILTER, INFO, FORMAT
Additional columns, one for each sample, with sample ID

(a) VCF example

Header

```
##fileformat=VCFv4.1
##fileDate=20110413
##source=VCFtools
##reference=file:///refs/human_NCBI36.fasta
##contig=<ID=1,length=249250621,md5=1b22b98cdeb4a9304cb5d48026a85128,species="Homo Sapiens">
##contig=<ID=X,length=155270560,md5=7e0e2e580297b7764e31dbc80c2540dd,species="Homo Sapiens">
##INFO=<ID=AA,Number=1,Type=String,Description="Ancestral Allele">
##INFO=<ID=H2,Number=0,Type=Flag,Description="HapMap2 membership">
##FORMAT=<ID=GT,Number=1,Type=String,Description="Genotype">
##FORMAT=<ID=GQ,Number=1,Type=Integer,Description="Genotype Quality">
##FORMAT=<ID=DP,Number=1,Type=Integer,Description="Read Depth">
##ALT=<ID=DEL,Description="Deletion">
##INFO=<ID=SVTYPE,Number=1,Type=String,Description="Type of structural variant">
##INFO=<ID=END,Number=1,Type=Integer,Description="End position of the variant">
```

Body

#CHROM	POS	ID	REF	ALT	QUAL	FILTER	INFO	FORMAT	SAMPLE1	SAMPLE2
1	1	.	ACG	A,AT	40	PASS	.	GT:DP	1/1:13	2/2:29
1	2	.	C	T,CT	.	PASS	H2;AA=T	GT	0|1	2/2
1	5	rs12	A	G	67	PASS	.	GT:DP	1|0:16	2/2:20
X	100	.	T	<DEL>	.	PASS	SVTYPE=DEL;END=299	GT:GQ:DP	1:12:.	0/0:20:36

Danecek et al. Bioinformatics 2011

Complete Variant Calling Pipeline (Outdated!)}

Our analysis pipeline consisted of the following:

- Align the FASTQ files to genome
- Convert SAM file to BAM, add read group info
- Filter the reads based on quality (BAMTools)
- Samtools to sort and index, and use Picard to mark duplicates
- GATK calibration, realignment, variant calling (HaplotypeCaller, Mutect2)
- Filter the called variants (GATK filtersnps and filterindels).
- Annotation of SNPs (snpEff, condel)
- Filter by frequency (thousand genomes, TCGA, etc.)
- Downstream analysis (rare variants, pedigree, pathway level, etc)

Downstream Annotation and Analysis (Outdated!)

Downstream Annotation Tools (old list):

- snpEff (http://snpeff.sourceforge.net/)
- Condel (http://bg.upf.edu/condel/home)
- SIFT http://sift.jcvi.org/
- Polyphen 2 http://genetics.bwh.harvard.edu/pph2/
- http://mutationassessor.org/
- Ensembl variant effect predictor (http://www.ensembl.org/info/docs/variation/vep/index.html)
- Thousand Genomes variant frequency (e.g. 1% threshold) and Exome Sequence Project variant frequency (e.g. 1%).

Session info

```
sessionInfo()
```

```
## R version 4.4.2 (2024-10-31)
## Platform: aarch64-apple-darwin20
## Running under: macOS Sequoia 15.3.2
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/lib/libRlapack.dylib;  LAPACK version 3.12.0
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## time zone: America/New_York
## tzcode source: internal
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## loaded via a namespace (and not attached):
##  [1] digest_0.6.37     fastmap_1.2.0     xfun_0.51         Matrix_1.7-2
##  [5] lattice_0.22-6    reticulate_1.40.0 knitr_1.50        htmltools_0.5.8.1
##  [9] png_0.1-8         rmarkdown_2.29    cli_3.6.4         grid_4.4.2
## [13] compiler_4.4.2    rstudioapi_0.17.1 tools_4.4.2       evaluate_1.0.3
## [17] Rcpp_1.0.14       yaml_2.3.10       jsonlite_2.0.0    rlang_1.1.5
```