An Introduction to RNA-sequencing

QC, Preprocessing, and Alignment

W. Evan Johnson, Ph.D.
Professor, Division of Infectious Disease
Director, Center for Data Science
Co-Direcor, Center for Biomedical Informatics and Health AI
Rutgers University – New Jersey Medical School

2025-07-28

Installing R Packages:

Install the following tools: `Rsubread`, `Rsamtools`, and `SummarizedExperiment`. We will also need help from the `tidyverse`.

```
if (!requireNamespace("BiocManager", quietly = TRUE))
  install.packages("BiocManager")
BiocManager::install(c("Rsubread","Rsamtools",
              "tidyverse","SummarizedExperiment"))
```

Load Packages for Today

We will be using the following packages for today's lecture:

```
library(tidyverse) ## tools for data wrangling
library(Rsubread) ## alignment and feature counts
library(Rsamtools) ## managing .sam and .bam files
library(SummarizedExperiment) ## managing counts data
```

Objective

- Disclaimer: non-comprehensive introduction to RNA-sequencing
- Introduce preprocessing steps
- Visualization
- Analytical methods
- Common software tools

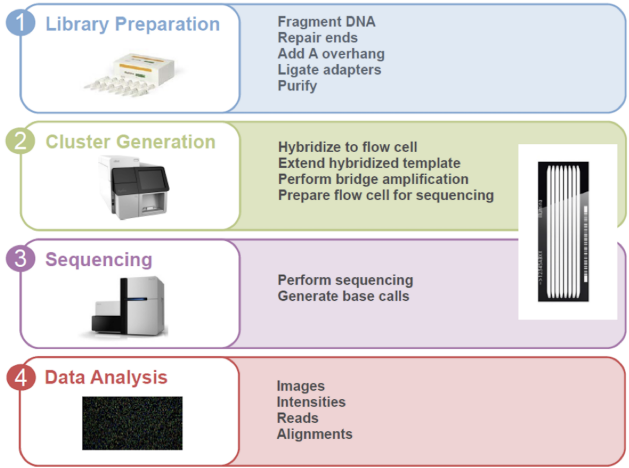
Steps to an RNA-seq Analysis (Literacy)

1. Preprocessing and QC:
   - Fasta and Fastq files
   - FastQC: good vs. bad examples
   - Visualization
2. Alignment
   - Obtaining genome sequence and annotation
   - Software: Bowtie, TopHat, STAR, Subread/Rsubread
3. Expression Quantification
   - Count reads hitting genes, etc
   - Approaches/software: HT-Seq, STAR, Cufflinks, RPKM FPKM or CPM, RSEM, edgeR, findOverlaps (GenomicRanges). featureCounts (Rsubread)

Steps to an RNA-seq Analysis (Literacy)

4. More visualization
   - Heatmaps, boxplots, PCA, t-SNE, UMAP
5. Differential Expression
   - Batch correction
   - Overdispersion
   - General Workflow
   - Available tools: edgeR, DESeq, Limma/voom
   - Even more visualization!!

Illumina Sequencing Workflow



Sequencing Data Formats

Genome sequcencing data is often stored in one of two formats, FASTA and FASTQ text files. For example a FASTA files looks like the following:

```
>chrX
ttgaactcctgacctcaggtgatccgcccgccttgacctcccaaagcgct
cctgcctcagcctcccgagtagctgggactacaggtgcctgccaccatgc
....
caggctaatttttgtatttttagtagagacggggtttcaccatgttagc
caggatggtctcaatctcctgacctcatgatccgcctgcctcggcctccc
>chrY
tgtacactttaaatgggtgaattttatggaatgtgaattataCGTGTTGG
CTTGTAAAAAAAAATGATGGAGATGGAGACGTGACTCTAGCGTGAAGGGG
...
```

FASTQ Files

We can also store confidence or quality scores using a FASTQ format:

```
@HWI-EAS240_0001:2:1:1142:17571#0/1
CTCTCTTTTCTCCCCANGTCTCCTCATGACCATATCCNTGTTGTCCATTGTGTANGGNNNCTTGCGCTGCAACATC
+HWI-EAS240_0001:2:1:1142:17571#0/1
bababb`abbbbabbYB^[^^`_`_bbbbbbbbbb`_B`_``^`^bbbba^`^^BOOBBBNOOOObY^^bbbbbbb
@HWI-EAS240_0001:2:1:1142:6453#0/1
CAGGACGTGCACTATGNCATCCACGATGCAAGTCTTANCATTATTCAGGATACANATANNGGCAAACGTGTAATTA
+HWI-EAS240_0001:2:1:1142:6453#0/1
_U][aa_aa^^```^[B][][]]]S```Y``a`aaZ]B]]^^^[^```[OVY]XBNNNBB]]]]]L^^^^```[V`
@HWI-EAS240_0001:2:1:1142:19443#0/1
TCAGAAAACAGAAAGGTCATTTTCTTACTTCTTTGCANGATGCCACCCTCCAGANCAGNNAATTGCTACTTTGCAC
+HWI-EAS240_0001:2:1:1142:19443#0/1
bbbabb^^bbb^bbabb]bb]^]`]]]bbbbbbbbYYBYXOOOOObb`bbVY^HBOOOBBOJOLO_bbbb]b_b`a
```

.FASTQ Paired End

fastq_SRR1997469_1.fastq

```
@SRR1997469.1 1 length=36
CAGTCTTCTTAGAAATATCCACTTCGGAATAAAAGA
+SRR1997469.1 1 length=36
BBBBBFFFFF<FFFFFFFFFFFFFBFFFFFFFFFF
@SRR1997469.2 2 length=36
ACAGTTGAACGATCCTTTACAGANAGNAGNCTNGTA
+SRR1997469.2 2 length=36
<BBBBFFFBBF<BFF/<FFFFFF##############
...
```

fastq_SRR1997469_2.fastq

```
@SRR1997469.1 1 length=36
AGATAAGATGGTAATCTTTGATGGAGAACATTAAGA
+SRR1997469.1 1 length=36
BBBBBFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
@SRR1997469.2 2 length=36
ACTGGGAANCCTTCTGTCTAGCCTTATATGAAAAAA
+SRR1997469.2 2 length=36
BBB/BFFB#BB/FFFF/FFFFFFBBFFFFFFFFFBF
...
```

FASTQ Encoding

In order to translate FASTQ quality scores:

```
 SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS.....................................................
 ..........................XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX.....................
 ............................IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII.....................
 ..............................JJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJ.....................
 LLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLL.....................................................
 !"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|}~
  |                         |    |        |                              |                     |
33                        59   64       73                            104                   126

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)
```

https://en.wikipedia.org/wiki/Phred_quality_score

Sequence Quality Score (PHRED)

Phred quality scores are logarithmically linked to error probabilities

Phred Quality Score	Probability of incorrect base call	Base call accuracy
10	1 in 10	90%
20	1 in 100	99%
30	1 in 1000	99.9%
40	1 in 10,000	99.99%
50	1 in 100,000	99.999%
60	1 in 1,000,000	99.9999%

Sequence Quality Score (PHRED)

Quality scores (Phred)

- Sanger Phred: Range=(0,40), $P = 1 - 10^{-(ASCII-33)/10}$
- Solexa: Range= (-5,40), $P = \frac{10^{(ASCII-64)/10)}}{1+10^{(ASCII-64)/10}}$
- Illumina 1.3: (0,40), $P = \frac{10^{(ASCII-64)/10)}}{1+10^{(ASCII-64)/10}}$
- Illumina 1.5: Range=(2,40), $P = 1 - 10^{-(ASCII-64)/10}$
- Illumina 1.8: Same as Sanger except Range=(0,41)

FASTQ Probability

> Sanger

	ASCII	Quality
!	33	0.0000
"	34	0.2057
#	35	0.3690
$	36	0.4988
%	37	0.6019
&	38	0.6838
'	39	0.7488
(	40	0.8005
)	41	0.8415
*	42	0.8741
+	43	0.9000
,	44	0.9206
-	45	0.9369

> Solexa

	ASCII	Quality
;	59	0.2403
<	60	0.2847
=	61	0.3339
>	62	0.3869
?	63	0.4427
@	64	0.5000
A	65	0.5573
B	66	0.6131
C	67	0.6661
D	68	0.7153
E	69	0.7597
F	70	0.7992
G	71	0.8337
H	72	0.8632
I	73	0.8882

> Illumina1.3

	ASCII	Quality
@	64	0.5000
A	65	0.5573
B	66	0.6131
C	67	0.6661
D	68	0.7153
E	69	0.7597
F	70	0.7992
G	71	0.8337
H	72	0.8632
I	73	0.8882
J	74	0.9091
K	75	0.9264
L	76	0.9406
M	77	0.9523
N	78	0.9617

> Illumina1.5

	ASCII	Quality
B	66	0.3690
C	67	0.4988
D	68	0.6019
E	69	0.6838
F	70	0.7488
G	71	0.8005
H	72	0.8415
I	73	0.8741
J	74	0.9000
K	75	0.9206
L	76	0.9369
M	77	0.9499
N	78	0.9602
O	79	0.9684
P	80	0.9749

Preprocessing and QC using FASTQC

FastQC provides a simple way to do QC checks on raw sequence data:

- Import of data from BAM, SAM or FastQ files
- Quick overview and summary graphs and tables to quickly assess your data
- Export of results to an HTML based permanent report
- Offline operation to allow automated generation of reports without running the interactive application

Preprocessing and QC using FASTQC

Interactive GUI from Amarel Desktop:

```
module spider fastqc
#module avail fastqc
module load FastQC
fastqc
```

FastQC Example

Running FastQC from command-line (single file):

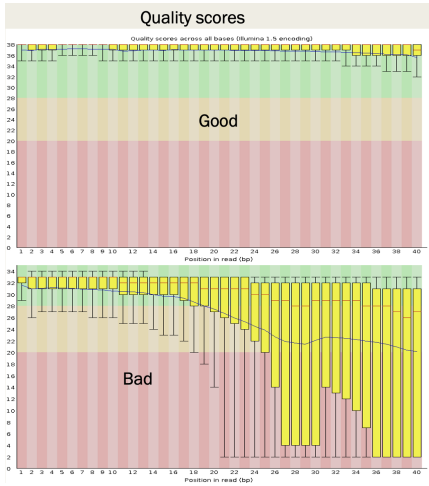
```
mkdir example_data/fastqc
fastqc example_data/reads/R01_10_short500K.fq.gz \
       --outdir=example_data/fastqc
```

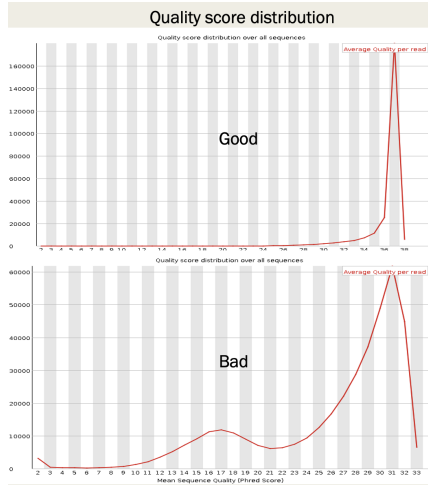
FastQC Example

Running FastQC from command-line (multiple files):

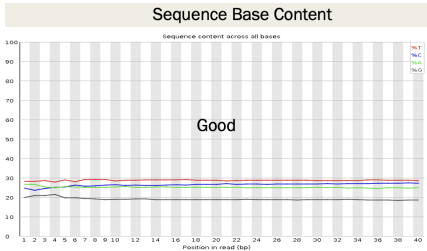
```
fastqc *.fq.gz --outdir=example_data/fastqc
```

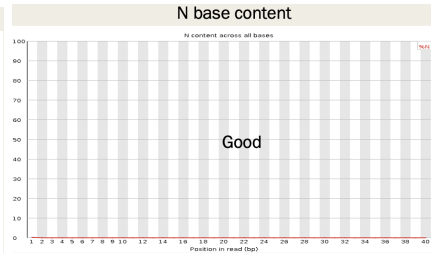
FastQC Score Distribution

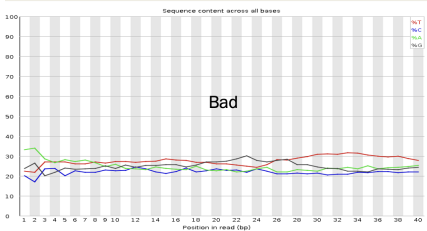


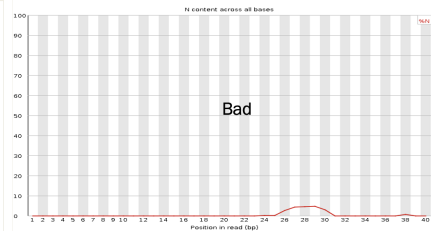


FastQC Base and N Distribution









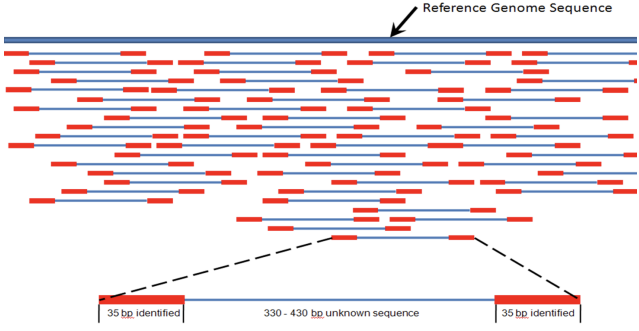
MultiQC Example

You can use `multiqc` to combine FastQC results. On Amarel Desktop do the following:

```
module load miniconda
conda init
source ~/.bashrc
conda install bioconda::multiqc
#pip install multiqc
cd example_data/fastqc
multiqc .
```

Alignment to the Reference Genome

Goal: Find the genomic location for the sequencing read. Software: `Bowtie2`, `TopHat`, `STAR`, `Subread/Rsubread`, many others!

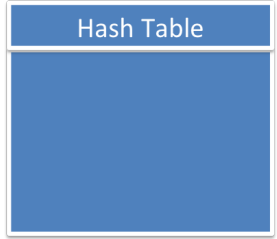


Indexing your genome

Abraham Lincoln: "Give me six hours to chop down a tree and I will spend the first four sharpening the axe."

(4 hours indexing the genome, 2 hours aligning the reads)

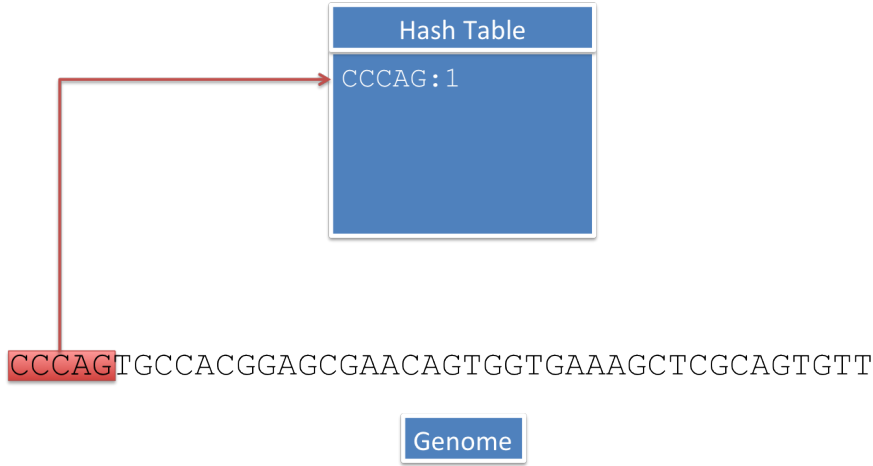
Creating a Hash Index Table



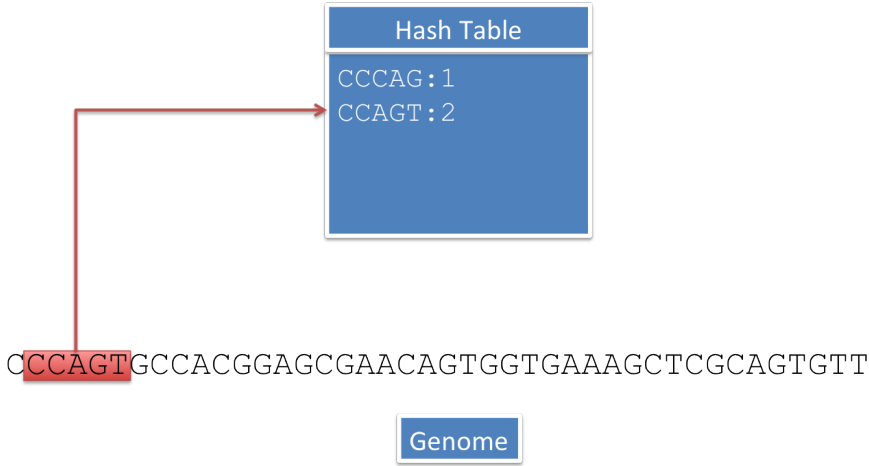
CCCAGTGCCACGGAGCGAACAGTGGTGAAAGCTCGCAGTGTT

Genome

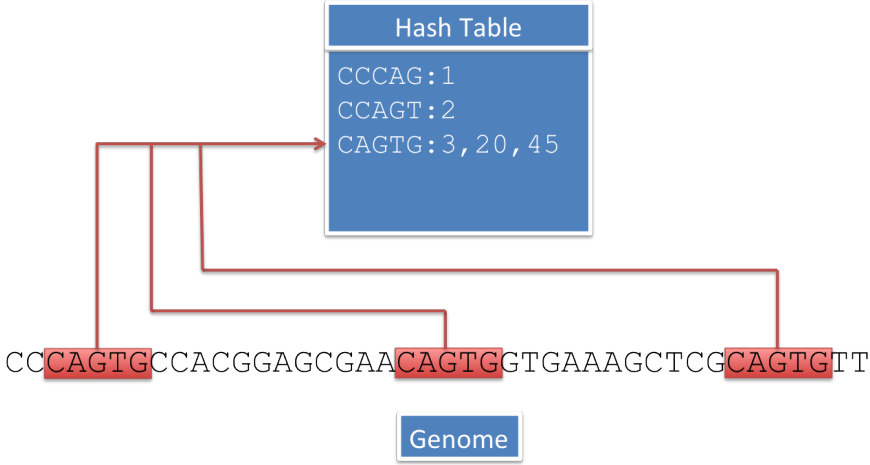
Creating a Hash Index Table



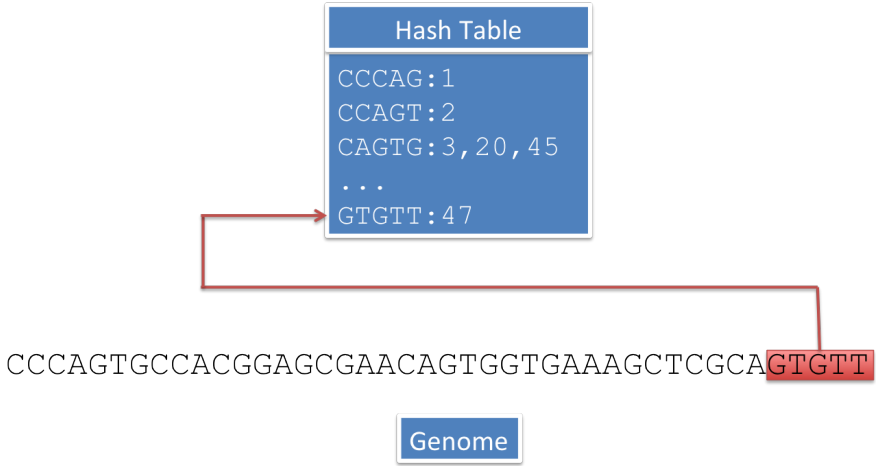
Creating a Hash Index Table



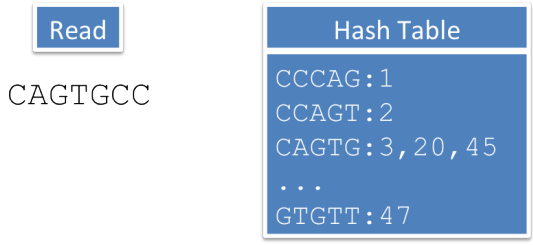
Creating a Hash Index Table



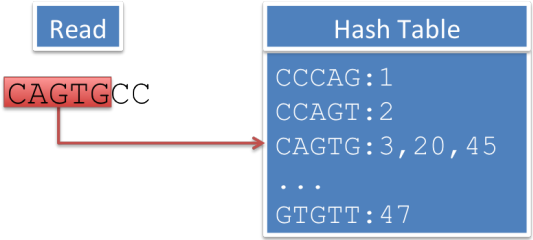
Creating a Hash Index Table



Read Lookup


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

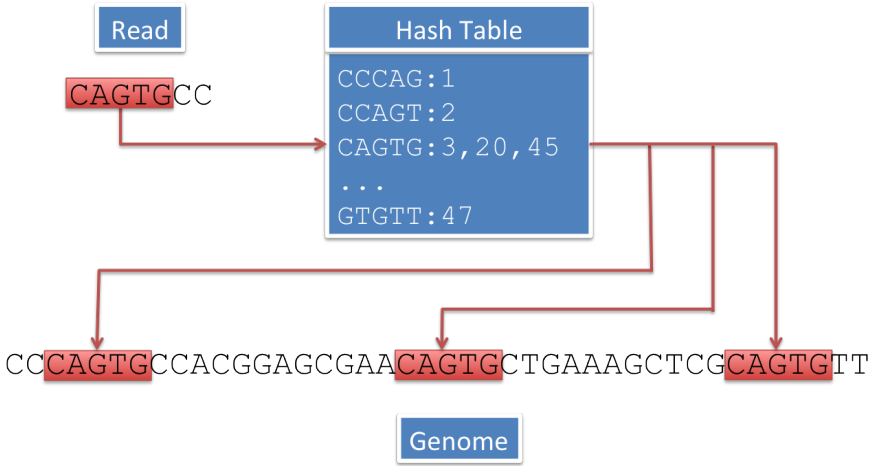
Read Lookup



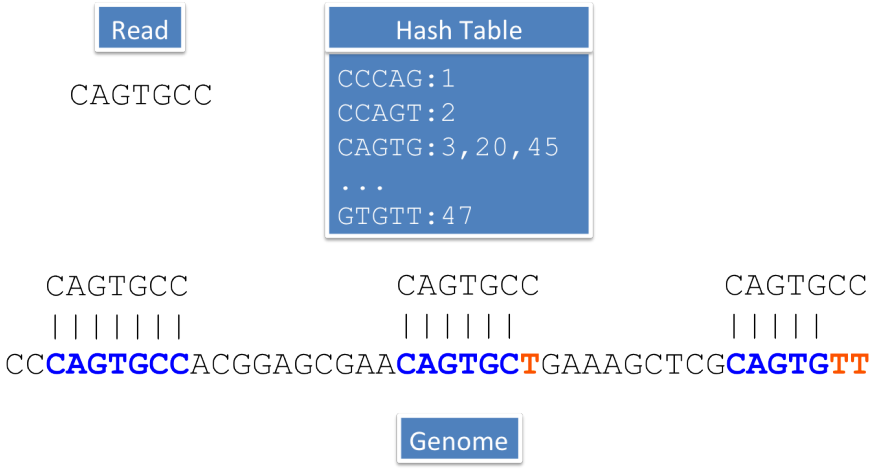
CCCAGTGCCACGGAGCGAACAGTGCTGAAAGCTCGCAGTGTT

Genome

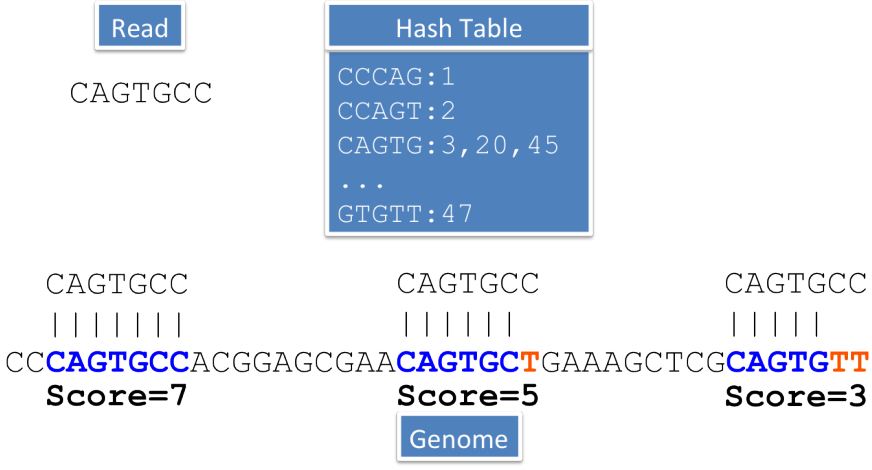
Read Lookup


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

Session info

```
sessionInfo()
```

```
## R version 4.5.1 (2025-06-13)
## Platform: aarch64-apple-darwin20
## Running under: macOS Sequoia 15.5
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.5-arm64/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.5-arm64/Resources/lib/libRlapack.dylib;  LAPACK version 3.12.1
##
## 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] stats4    stats     graphics  grDevices utils     datasets  methods
## [8] base
##
## other attached packages:
##  [1] SummarizedExperiment_1.38.1 Biobase_2.68.0
##  [3] MatrixGenerics_1.20.0       matrixStats_1.5.0
##  [5] Rsamtools_2.24.0            Biostrings_2.76.0
##  [7] XVector_0.48.0              GenomicRanges_1.60.0
##  [9] GenomeInfoDb_1.44.1         IRanges_2.42.0
## [11] S4Vectors_0.46.0            BiocGenerics_0.54.0
```