



# Genome Data Acquisition and Processing

Borchert, Dr. Schapranow  
Data Management for Digital Health  
Winter 2023

# Agenda

## Pillars of the Lecture

### Medical Use Cases



Biology Recap



Oncology



Nephrology



Infectious  
Diseases

### Technology Foundation



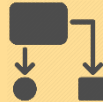
Data  
Sources



Data  
Formats



Processing and  
Analysis



Software  
Architectures

### Machine Learning

Data



Refine



Evaluate



Prediction +  
Probability

**Genome Data  
Acquisition and  
Processing**

Data Management for  
Digital Health, Winter  
2023  
2

# Agenda

## Pillars of the Lecture

### Medical Use Cases



Biology Recap



Oncology



Nephrology



Infectious  
Diseases

### Technology Foundation



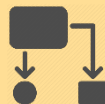
Data  
Sources



Data  
Formats



Processing and  
Analysis



Software  
Architectures

### Machine Learning

Data



Refine



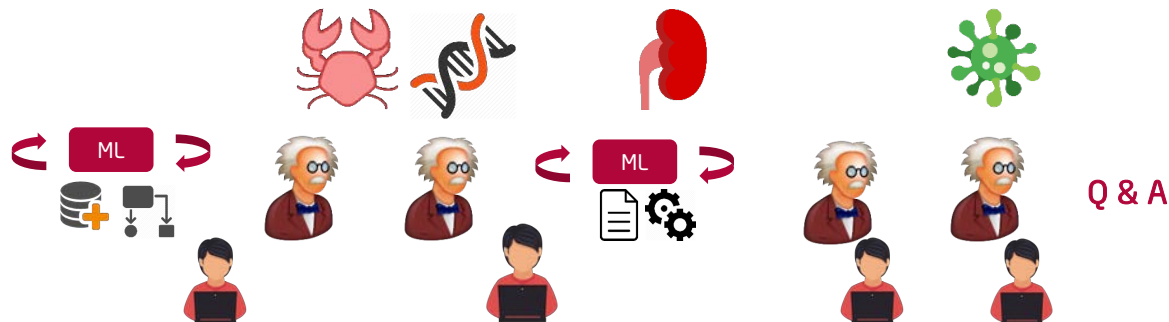
Evaluate

Prediction +  
Probability

### Genome Data Acquisition and Processing

Data Management for  
Digital Health, Winter  
2023

# Lecture Schedule



Final Exam  
Feb 13, 2024  
11:00am,  
Lecture Hall HS1

Nov

Dec

Jan

Feb

- Lecture Kickoff
- Actors in Healthcare
- Digital Health Data

- Machine Learning (ML) Foundations
- Use Case Oncology
- Biology Recap

- Natural Language Processing
- Use Case Nephrology & Intensive Care
- Supervised ML & Deep Learning

- Use Case Infectious Diseases
- Unsupervised ML

## Genome Data Acquisition and Processing

Data Management for Digital Health, Winter 2023

# Numbers You Should Know

## << QUIZ >>

- What do you think are the approx. costs for sequencing a full human genome?
  - A. 100,000 EUR
  - B. 10,000 EUR
  - C. 1,000 EUR
  - D. 100 EUR



### Genome Data Acquisition and Processing

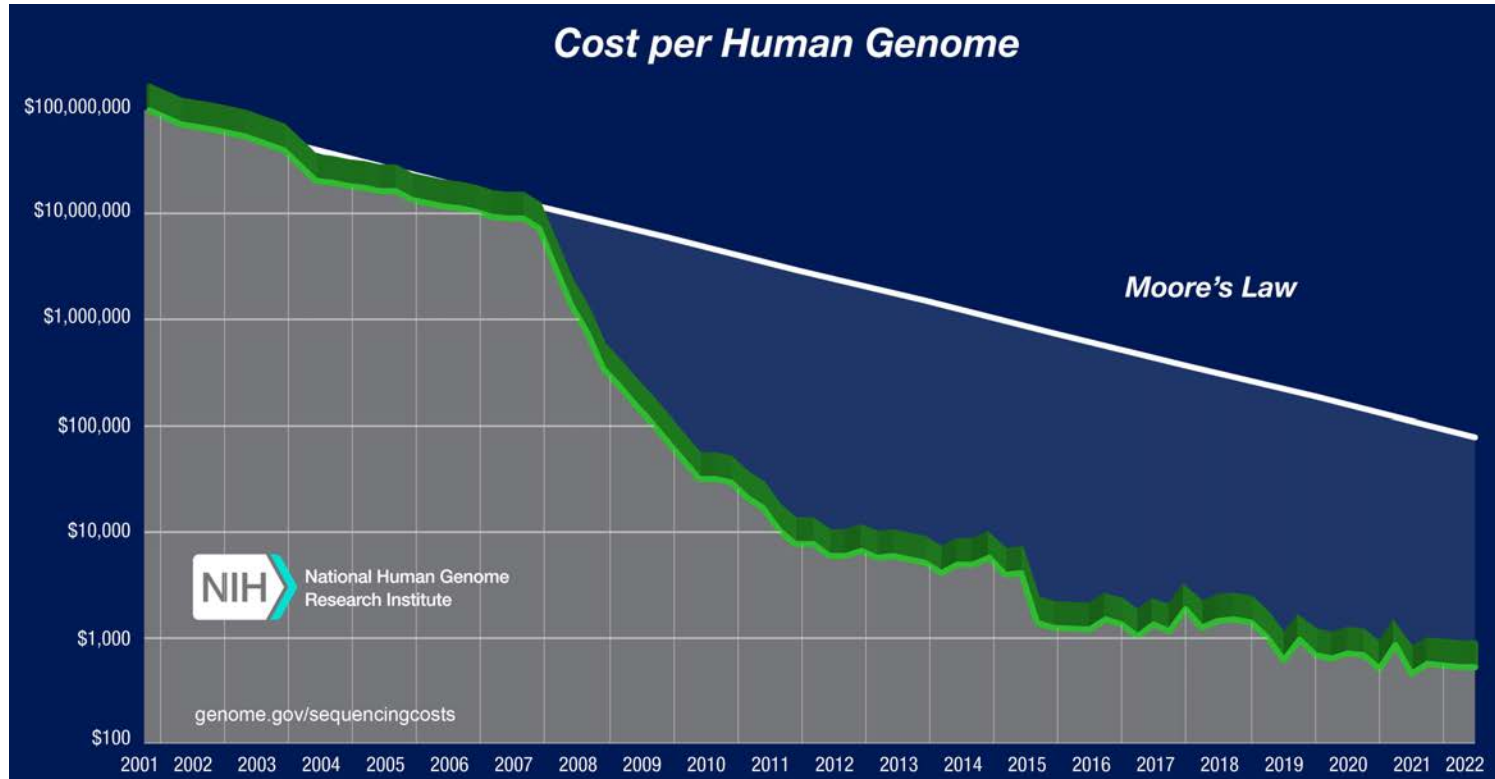
Data Management for  
Digital Health, Winter  
2023

# Numbers You Should Know

## << QUIZ >>

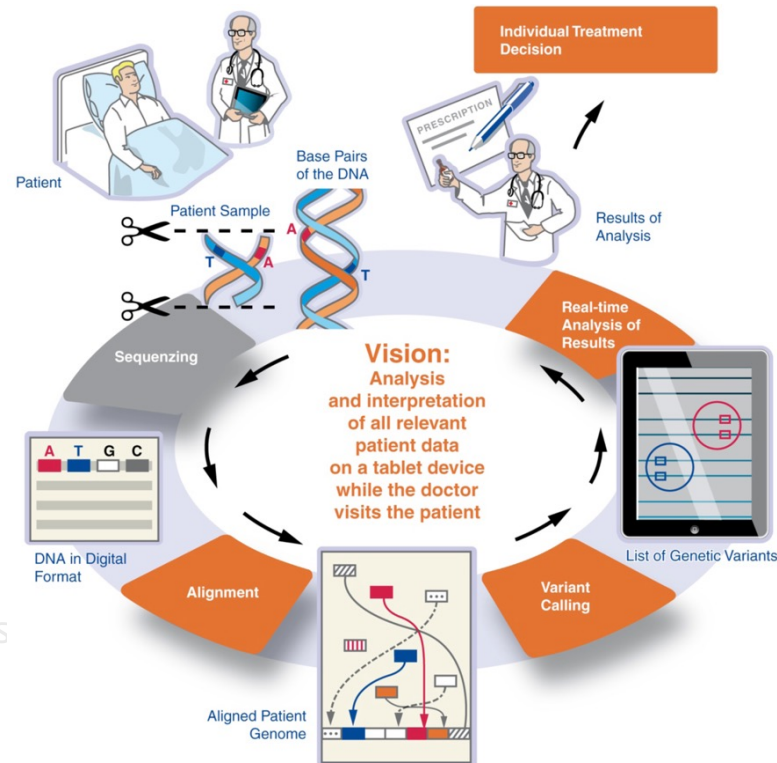
■ What do you think

- A. 100,000 EUR
- B. 10,000 EUR
- C. 1,000 EUR
- D. 100 EUR



# From Raw Genome Data to Analysis

- **DNA Sequencing:** Transformation of analogues DNA into digital format
- **Alignment:** Reconstruction of complete genome with snippets
- **Variant Calling:** Identification of genetic variants
- **Data Annotation:** Linking genetic variants with research findings



## Genome Data Acquisition and Processing

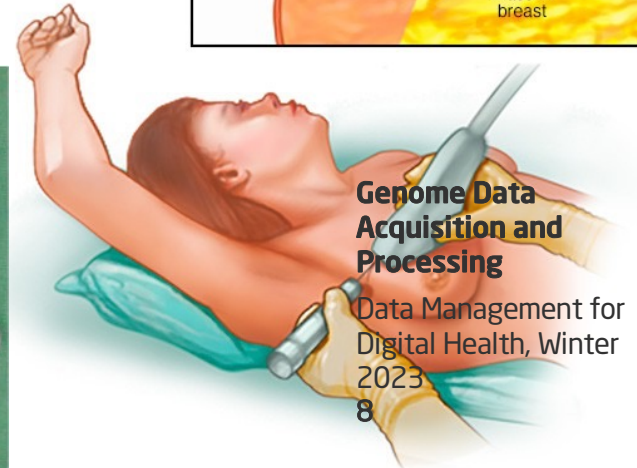
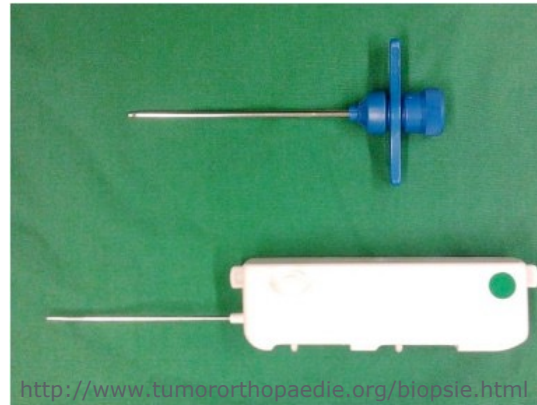
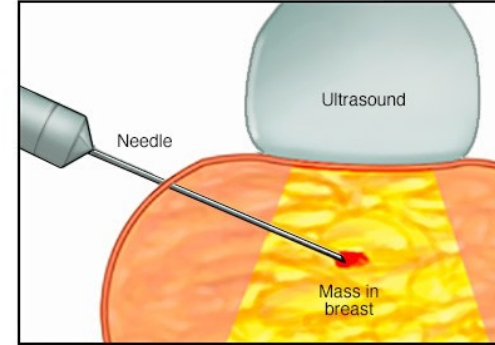
Data Management for Digital Health, Winter 2023



# Biopsy



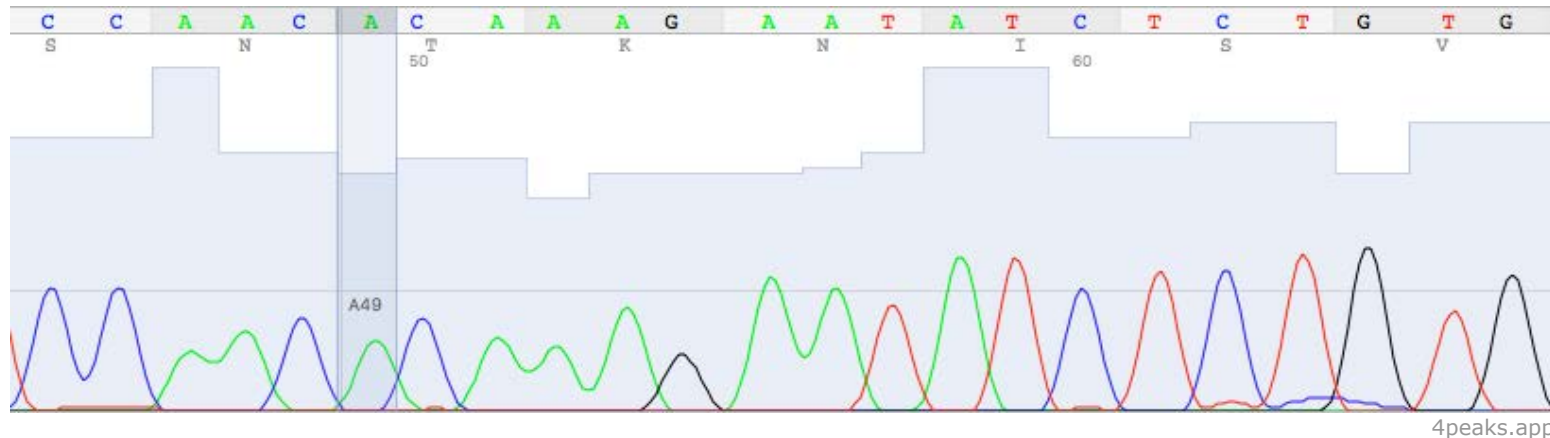
- **Biopsy** := Extraction of cells/tissue from the body
- Purpose: Obtain sample for analysis, e.g. abnormal vs. normal tissue
- Sample is typically processed by department of pathology and a report is created (duration: minutes to days)
- New trend: liquid biopsy to acquire cells from blood stream
- Foundation for treatment decision
- Sample can be used for further tests, e.g. genome sequencing





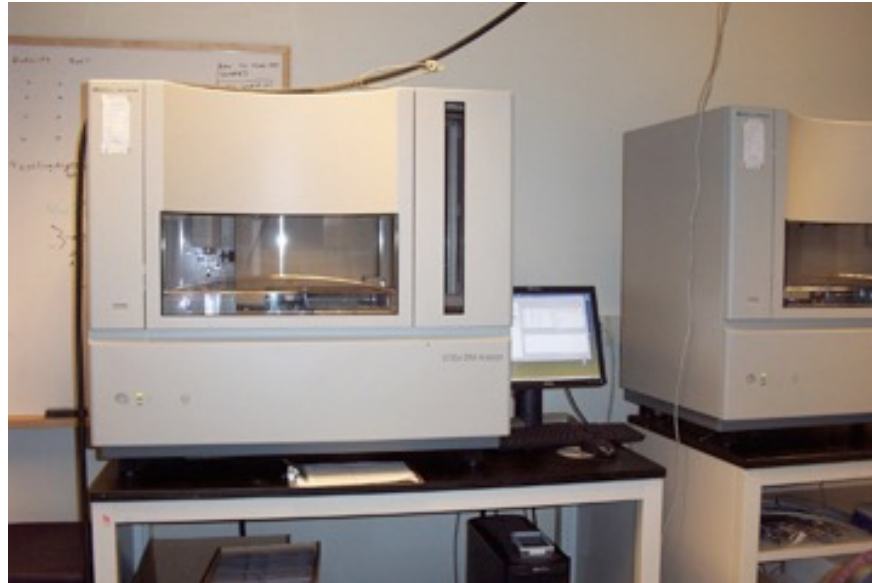


- **Sequencing** := Transformation of analogous DNA into digital format (A/D converter)
- Input: Chunks of DNA
- Output: DNA reads in digital form, e.g. in FASTQ format



# ABI Sequencing (1<sup>st</sup> gen)

- 2002: Sanger sequencing provides very high accuracy
- Accuracy: > 99.999%
- Throughput: 100 kbp / run (3hrs)
- Read length: 0.6-1 kbp
- Issues: time-intensive

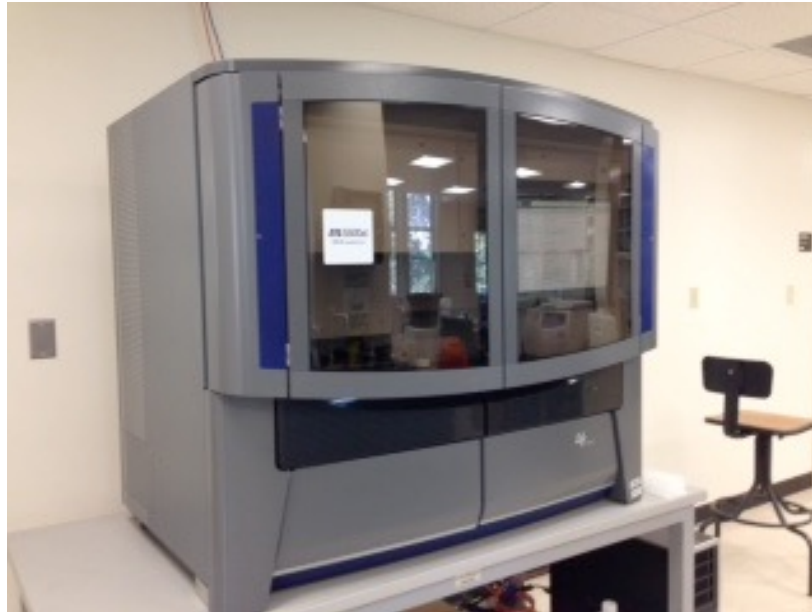


## Genome Data Acquisition and Processing

Data Management for  
Digital Health, Winter  
2023  
10

# ABI Sequencing (2<sup>nd</sup> gen)

- 2006: Sequencing by Oligonucleotide Ligation and Detection (SOLiD)
- Accuracy: > 99.99%
- Throughput: 60 Gbp / run (5-10 days)
- Read length: 35-100 bp
- Issues: time-intensive



## Genome Data Acquisition and Processing

Data Management for  
Digital Health, Winter  
2023

11

# Roche-454 Sequencing

- 2005-2013: Roche-454 Life Sciences launched first NGS device using pyrosequencing / sequencing by synthesis approach
- Accuracy: >99.9%
- Throughput: 400-600 Mbp / run
- Read length: 200-400 bp (2009) later up to 700 bp
- Issues: Homopolymer repeat regions



## Genome Data Acquisition and Processing

Data Management for  
Digital Health, Winter  
2023  
12

- 2006: Solexa introduced **Genome Analyzer**
- 2007: Illumina acquired Solexa
- Accuracy: >99.9%
- Throughput:
  - 2006: 1 Gbp / run (2006),
  - 2016: up to 1 Tbp / run (6 days)
- Read length: 200-600 bp
- Issues: cheap but less accurate



## Genome Data Acquisition and Processing

Data Management for  
Digital Health, Winter  
2023  
13

# Illumina Sequencing

- Illumina HiSeq series is discontinued from 2023/2024 on
- New flagship NovaSeq series
  - Enables up to 4x throughput
  - Runtime < 1d per genome
- Read length: 125-250 bp
- Issues: relatively expensive device but through high parallelization competitive per genome costs



	NextSeq <sup>†</sup>	HiSeq 4000 <sup>*</sup>	NovaSeq 6000 <sup>††</sup>
<b>Output Range</b>	20–120 Gb	125–1500 Gb	134–6000 Gb
<b>Run Time</b>	11–29 hr	< 1–3.5 days	13–44 hr
<b>Reads per Run</b>	130–400 million	2.5–5 billion	Up to 20 billion
<b>Maximum Read Length</b>	2 × 150 bp	2 × 150 bp	2 × 150 bp
<b>Samples per Run<sup>‡</sup></b>	1	6–12	4–48



- 2013: PacBio introduces long-read sequencer supporting innovative sequence assembling
- Accuracy: >99% (at high coverage)
- Throughput: approx. 160 Gbp / 30hrs (continuous long read)
- Read length: approx. 35 kbp (→ DeNovo Alignment)
- Issues: still comparable slow and lacks precision



## Genome Data Acquisition and Processing

Data Management for  
Digital Health, Winter  
2023

15

- Very cheap and mobile long-read alternative (< 1,000 USD)
- Accuracy: up to 99%
- Throughput (theoretical max): 50 Gbp / run (72 hrs)
- Read length: 500bp to 4Mbp+
- Parallel processing, e.g. PromethION 48 (48 flow cells x 250 Gbps)
  
- Issues:
  - Early product build
  - After prep flow cell needs to be fed with new material for throughput (72 hrs)
  - High tech investments over the past years



M. Schapranow

## Genome Data Acquisition and Processing

Data Management for  
Digital Health, Winter  
2023  
16

# What to take Home?

- Throughput increased over the past decade
- Accuracy is comparable high due to n-times coverage of individual genome locations

Year	Method	Read Length	Accuracy	Throughput (per day)
2002	Sanger ABI 3730xl	Up to 1 kbp	>99.999 %	400 kbp
2008	Roche 454 GS FLX+	700 bp	>99.9 %	700 Mbp
2012	Illumina HiSeq 2500	2x125 kbp (paired)	>99.9 %	160 Gbp (paired)
2019	Pacific Biosciences Sequel Sequencer II	35 kbp	>90% / >99 % (multi-pass)	128 Gbp
2021	Oxford Nanopore PromethION P48	Up to 4.2 Mbp	Up to 99% (multi-pass)	4.0 Tbp

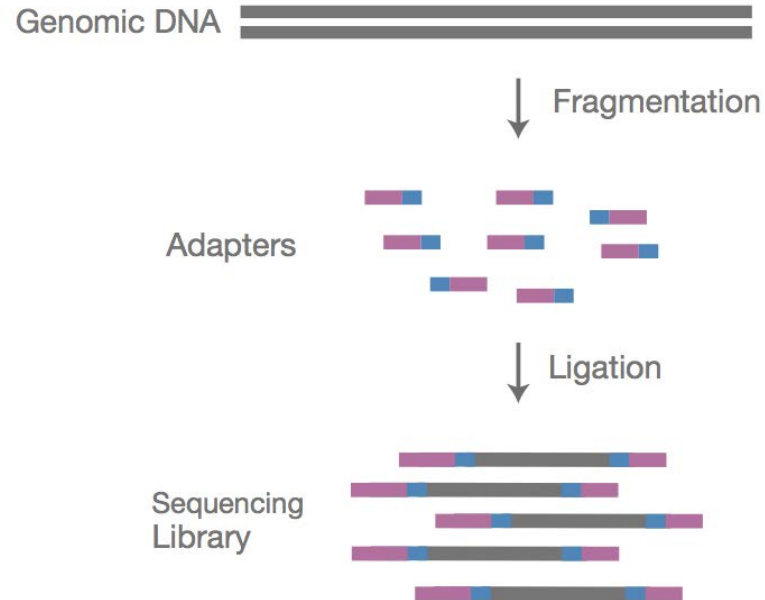
## Genome Data Acquisition and Processing

Data Management for Digital Health, Winter 2023

# Illumina/Solexa Sequencing Process

## 1. Preparation

- Double-stranded DNA is split into chunks of 200-800 bp
- Adapters are ligated to chunks to bind DNA fragments



### Genome Data Acquisition and Processing

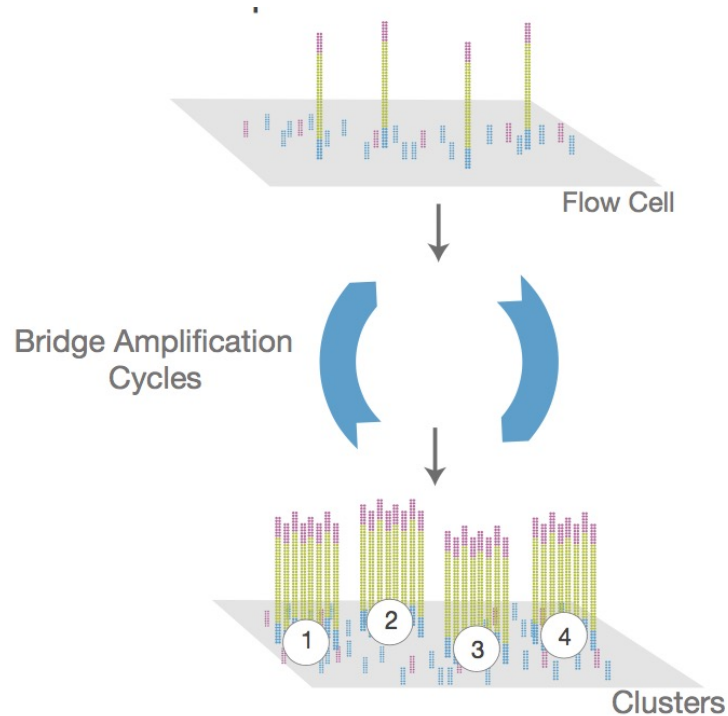
Data Management for  
Digital Health, Winter  
2023

18

# Illumina/Solexa Sequencing Process

## 2. Amplification

- Polymerase Chain Reaction (PCR) is used for amplification of DNA chunks



Illumina: An Introduction to Next-Generation Sequencing Technology (2016)

**Genome Data  
Acquisition and  
Processing**

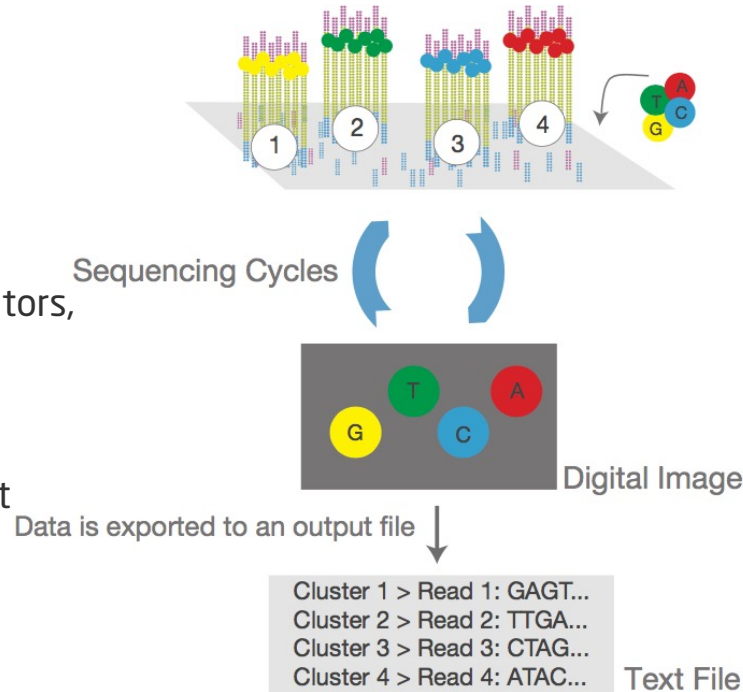
Data Management for  
Digital Health, Winter  
2023

19

# Illumina/Solexa Sequencing Process

## 3. Sequencing

- pos = 0
- While (pos < read length) do
  - pos++
  - Wash-off terminators
  - Add primers with fluorescently terminators, i.e. A, C, G, T + stop codon
  - Record laser light reflection image
  - Process image and write textual output
- Done

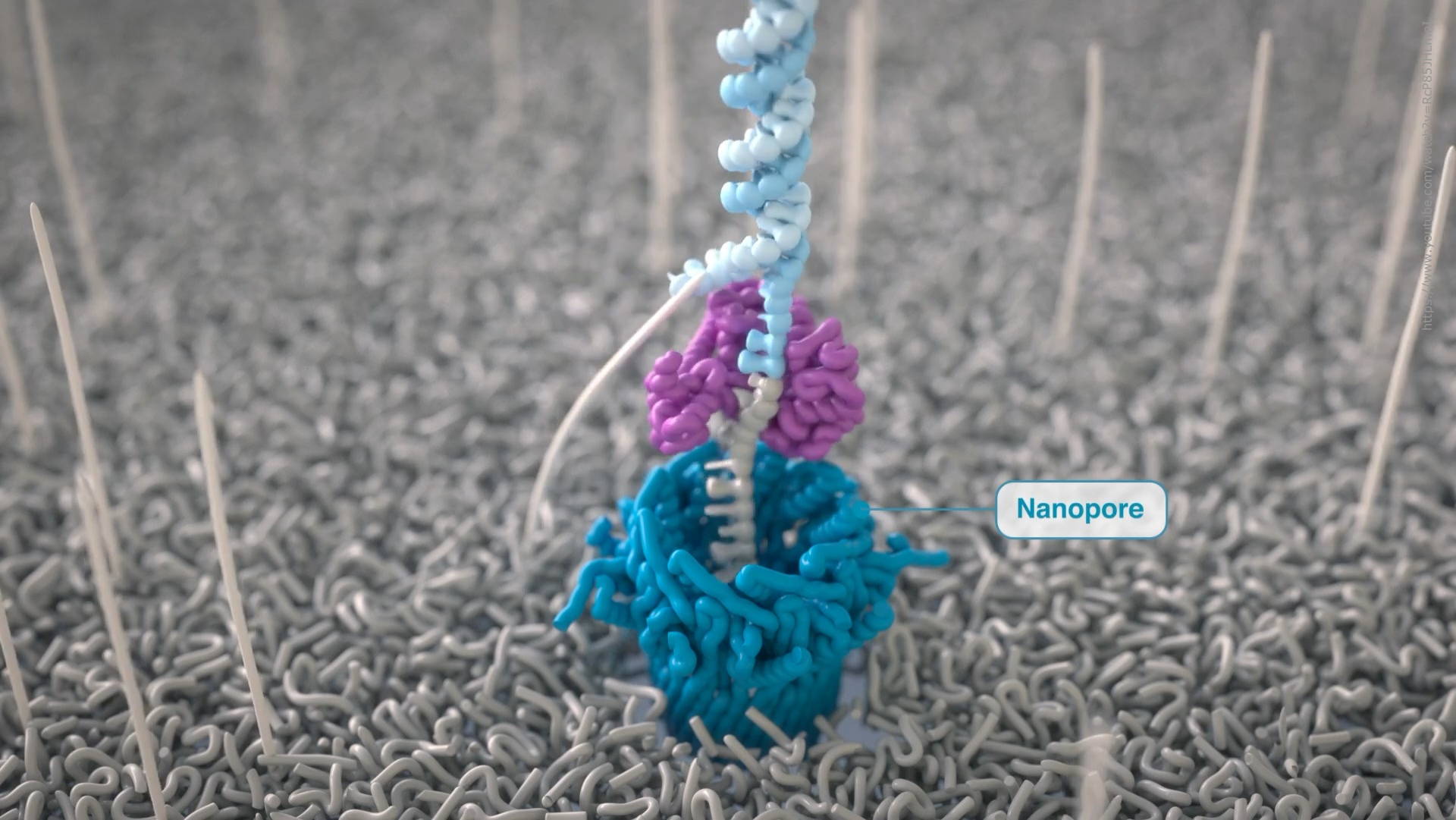


### Genome Data Acquisition and Processing

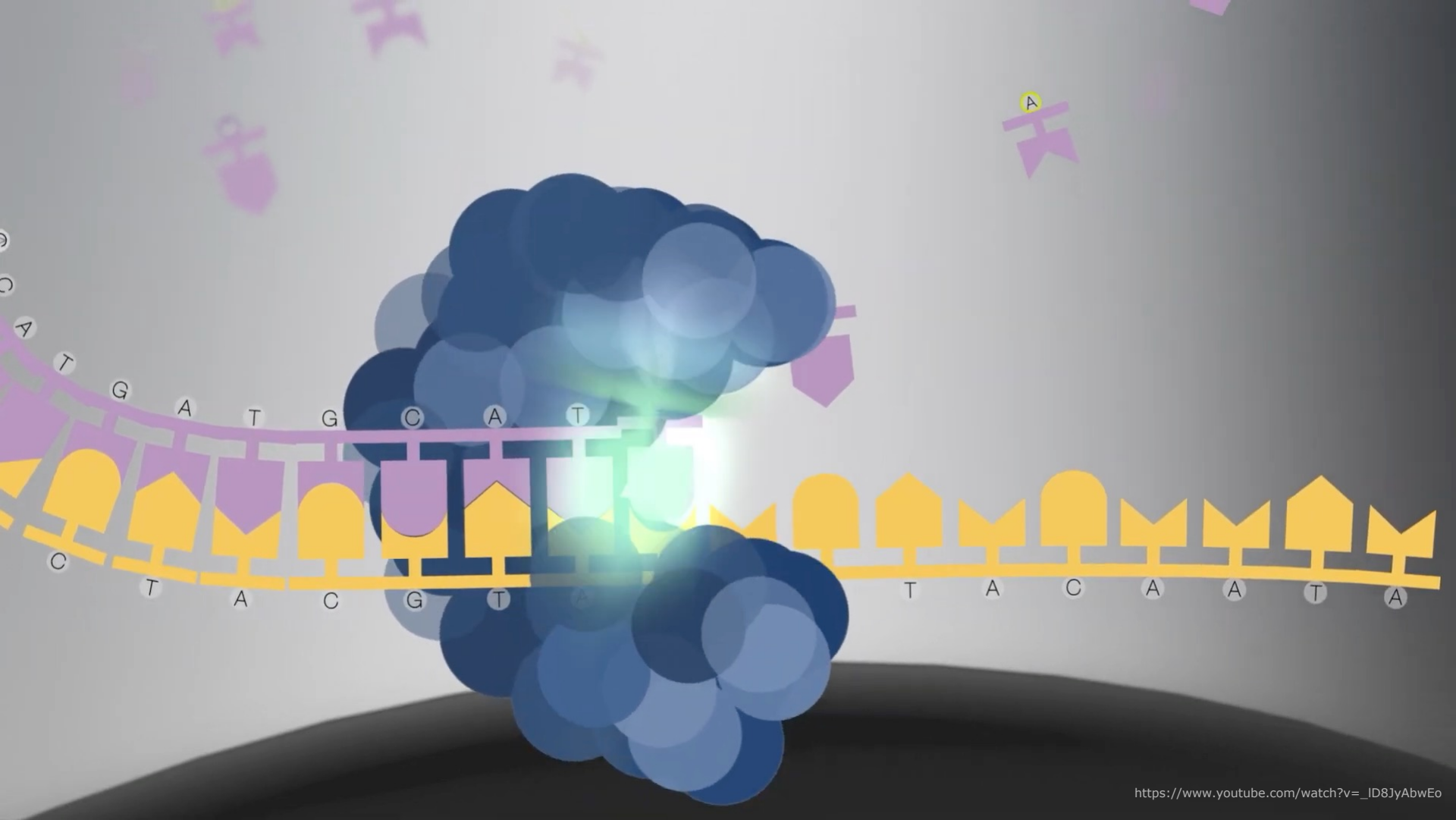
Data Management for  
Digital Health, Winter  
2023  
20



- Double-stranded DNA is split into chunks of 200-800 bp length
- Adaptors attached to DNA chunks
- Separation of double-strand into two strands using sodium hydroxide
- DNA chunks are washed across flowcell, i.e. DNA not binding to primers is removed
- Polymerase Chain Reaction (PCR) is used for amplification of DNA chunks
- Nucleotide bases and DNA polymerase are added to build bridges b/w primers
- Double strand is split-up using heat → dense clusters of identical DNA sequences
- Primers with fluorescently terminators are added, e.g. A, C, G, T + stop codon
- Primers attach to DNA chunks and DNA polymerase attaches to terminator
- Laser passes flowcell, i.e. each terminator type emits unique light
- Terminators are removed and new terminators are added to next DNA position



Nanopore



# What to take Home?

## Output of Sequencing

- Sample preparation results in **chunks of DNA**
- DNA sequencing is highly automated and results in **FASTQ file**
- FASTQ format used for further processing
- One read is a quart-tuple of:
  1. Sequence identifier / description
  2. Raw sequence
  3. Strand / direction
  4. Quality values per sequenced base

```

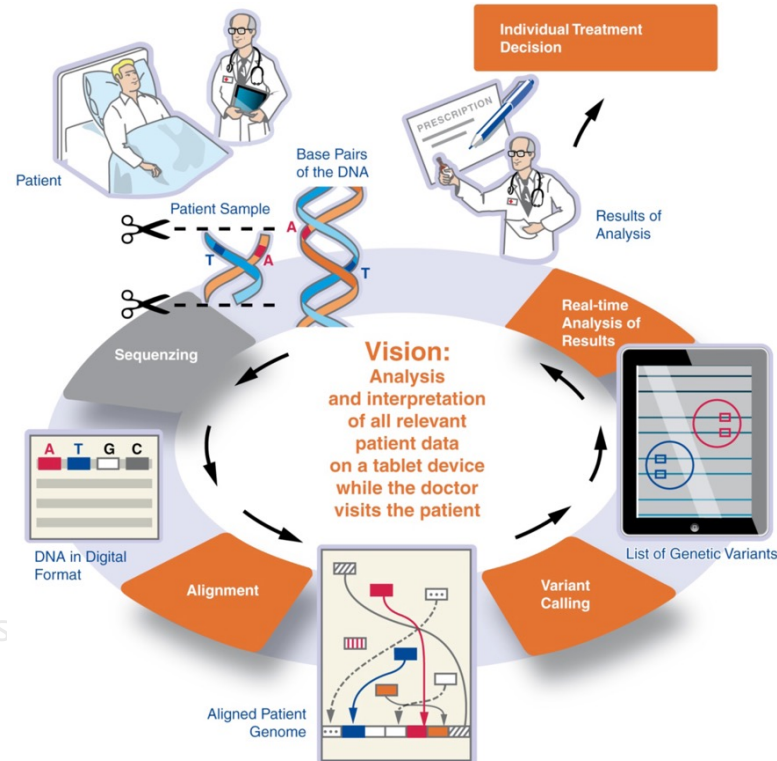
@HJ40ITD02IGHKD rank=0016764 x=3351.0 y=603.5
CGTATCTACACAGGGTCAGGGTCTGGATATTGGGAGAATATGGA
+
IIIIIIIIII=422:CA22///CFGGIIHHHBB>:/11::;2/4
@HJ40ITD02HBT0Z rank=0016788 x=2887.0 y=3969.
CGTATCTACACAGGGTCAGGGTTCGGAGATCAGGTAACGAA
+
A@ADFDDBA?=8,,//,/—/111141428:7667...4200
@HJ40ITD02GKSZP rank=0016806 x=2580.0 y=819.0
CGTATCTACACAGGGTCAGGGTCTGGATATAGGGCAGCACGGAC
+
FFFFFFFFFFFFD666ADD666??DFFFFHHHIIHHHHIIFFFFFFFE
@HJ40ITD02F4FE5 rank=0016858 x=2393.0 y=2687.
CGTATCTACACAGGGTCAGGGTATGGATATCAGGTAACAGTCA
+
IIIIIIIIII@@@IIHHHIIIIIGEEE@A<:5211121DDAD
@HJ40ITD02HNVGV rank=0017026 x=3025.5 y=893.0
CGTATCTACACAGGGTCAGGGTCTGGATATTGGGAGAATATGA
+
IIIIIIIIIIIIHHHIIHHHIIIIIIIIIGG333390::C?@@@
@HJ40ITD02GIZMW rank=0017128 x=2559.5 y=2134.
CGTATCTACACAGGGTCAGGGTCTGGATATTGACCTAACTGCTG
+
IIIIIIIIIIIIHHHIIHHHIIIIIIIIIIIIIIIIIIIIIIIIIIII
  
```

### Genome Data Acquisition and Processing

Data Management for  
Digital Health, Winter  
2023

# From Raw Genome Data to Analysis

- **DNA Sequencing:** Transformation of analogues DNA into digital format
- **Alignment:** Reconstruction of complete genome with snippets
- **Variant Calling:** Identification of genetic variants
- **Data Annotation:** Linking genetic variants with research findings



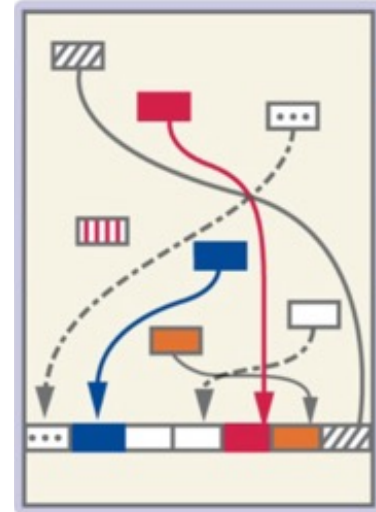
## Genome Data Acquisition and Processing

Data Management for Digital Health, Winter 2023  
25



# Alignment Overview

- **Alignment** := Mapping of DNA reads to a reference
- Input:
  - **DNA reads** := Sequence of nucleotides with a length of 100 bp up to some 1 kbp
  - **Reference genome** := Blueprint for alignment of DNA reads
- Output: Mapped DNA reads
  
- Bear in mind:
  - Less fraction in DNA reads, i.e. longer reads, allows more precise alignment
  - Reference from same origin improves mapping quality



## Acquisition and Processing

Data Management for  
Digital Health, Winter  
2023  
26



# Selected Alignment Algorithms

## Needleman-Wunsch Algorithm

---

- **Global alignment strategy:**
  - Initialize first row/column
  - Fill matrix; alignment score is defined by the value in the most lower right cell of the matrix
  - Perform backtracing to derive alignment
- **Needleman**, S. B. and **Wunsch**, C. D. (1970). "A general method applicable to the search for similarities in the amino acid sequence of two proteins" in "Molecular Biology", 48(3): 443-53

# Needleman-Wunsch Algorithm

- What is a best global alignment for the sequence CTG and the reference ACTGC?

	-0	A <sub>1</sub>	C <sub>2</sub>	T <sub>3</sub>	G <sub>4</sub>	C <sub>5</sub>
-0						
C <sub>1</sub>						
T <sub>2</sub>						
G <sub>3</sub>						

# Needleman-Wunsch Algorithm

## Assumptions

- $D(i,j)$  defines value of matrix at coordinates  $(i,j)$

	$-0$	$A_1$	$C_2$	$T_3$	$G_4$	$C_5$
$-0$						
$C_1$						
$T_2$						
$G_3$						

# Needleman-Wunsch Algorithm

## Assumptions

- Weight function  $w(a_i, b_j) :=$   $\begin{cases} +1 & \text{if } a_i == b_j, // \text{ match} \\ -1 & \text{if } a_i != b_j, // \text{ mismatch} \\ -2 & \text{if } (a_i == -) \text{ or } (b_j == -) // \text{ a.k.a. gap function} \end{cases}$

		$a_i$					
		-	A <sub>1</sub>	C <sub>2</sub>	T <sub>3</sub>	G <sub>4</sub>	C <sub>5</sub>
$b_j$	-						
	C <sub>1</sub>						
	T <sub>2</sub>						
	G <sub>3</sub>						

# Needleman-Wunsch Algorithm

## 1. Matrix Initialization

- $D(0,0) = 0$

	-0	A <sub>1</sub>	C <sub>2</sub>	T <sub>3</sub>	G <sub>4</sub>	C <sub>5</sub>
-0	0					
C <sub>1</sub>						
T <sub>2</sub>						
G <sub>3</sub>						

# Needleman-Wunsch Algorithm

## 1. Matrix Initialization

- $D(0,0) = 0$
- $D(i,0) = D(i-1,0) + w(a_i,-)$ ,  $1 \leq i \leq m$  // apply gap function to 1<sup>st</sup> horizontal row

	$-0$	$A_1$	$C_2$	$T_3$	$G_4$	$C_5$
$-0$	0	← -2	← -4	← -6	← -8	← -10
$C_1$						
$T_2$						
$G_3$						

# Needleman-Wunsch Algorithm

## 1. Matrix Initialization

- $D(0,0) = 0$
- $D(i,0) = D(i-1,0) + w(a_i,-)$ ,  $1 \leq i \leq m$
- $D(0,j) = D(0,j-1) + w(-,b_j)$ ,  $1 \leq j \leq n$  // apply gap function to 1<sup>st</sup> vertical row

	-0	A <sub>1</sub>	C <sub>2</sub>	T <sub>3</sub>	G <sub>4</sub>	C <sub>5</sub>
-0	0	← -2	← -4	← -6	← -8	← -10
C <sub>1</sub>	↑ -2					
T <sub>2</sub>	↑ -4					
G <sub>3</sub>	↑ -6					

# Needleman-Wunsch Algorithm

## 2. Fill Matrix

- $D(1,1) :=$  maximum of
  - ①  $D(0,0) + w(A_1, C_1) = 0 + (-1) = -1$  // Match or mismatch

	-0	A <sub>1</sub>	C <sub>2</sub>	T <sub>3</sub>	G <sub>4</sub>	C <sub>5</sub>
-0	0	① ← -2	← -4	← -6	← -8	← -10
C <sub>1</sub>	↑ -2	D(1,1)				
T <sub>2</sub>	↑ -4					
G <sub>3</sub>	↑ -6					



# Needleman-Wunsch Algorithm

## 2. Fill Matrix

- $D(1,1) :=$  maximum of
  - ①  $D(0,0) + w(A_1, C_1) = 0 + (-1) = -1$  // Match or mismatch
  - ②  $D(0,1) + w(-_0, C_1) = -2 + (-2) = -4$  // Deletion

	-0	A <sub>1</sub>	C <sub>2</sub>	T <sub>3</sub>	G <sub>4</sub>	C <sub>5</sub>
-0	0	← -2	← -4	← -6	← -8	← -10
C <sub>1</sub>	↑ -2	D(1,1)				
T <sub>2</sub>	↑ -4					
G <sub>3</sub>	↑ -6					

# Needleman-Wunsch Algorithm

## 2. Fill Matrix

- $D(1,1) :=$  maximum of
  - ①  $D(0,0) + w(A_1, C_1) = 0 + (-1) = -1$  // Match or mismatch
  - ②  $D(0,1) + w(-_0, C_1) = -2 + (-2) = -4$  // Deletion
  - ③  $D(1,0) + w(A_1, -_0) = -2 + (-2) = -4$  // Insertion

	-0	A <sub>1</sub>	C <sub>2</sub>	T <sub>3</sub>	G <sub>4</sub>	C <sub>5</sub>
-0	0	① ← ③ -2	← -4	← -6	← -8	← -10
C <sub>1</sub>	↑ -2	D(1,1)				
T <sub>2</sub>	↑ -4					
G <sub>3</sub>	↑ -6					

# Needleman-Wunsch Algorithm

## 2. Fill Matrix

- $D(1,1) :=$  maximum of
  - ①  $D(0,0) + w(A_1, C_1) = 0 + (-1) = -1$  // Match or mismatch
  - ②  $D(0,1) + w(-, C_1) = -2 + (-2) = -4$  // Deletion
  - ③  $D(1,0) + w(A_1, -) = -2 + (-2) = -4$  // Insertion

	-0	A <sub>1</sub>	C <sub>2</sub>	T <sub>3</sub>	G <sub>4</sub>	C <sub>5</sub>
-0	0	① ↘ -2	← -4	← -6	← -8	← -10
C <sub>1</sub>	↑ -2	③ ↓ -1				
T <sub>2</sub>	↑ -4	② →				
G <sub>3</sub>	↑ -6					

# Needleman-Wunsch Algorithm

## 2. Fill Matrix

- For all  $D(i,j)$ 
  - ①  $D(i-1,j-1) + w(a_i,b_j)$  // Match or mismatch
  - ②  $D(i-1,j) + w(a_{i-1},b_j)$  // Deletion
  - ③  $D(i,j-1) + w(a_i,b_{j-1})$  // Insertion
- Bear in mind: Filling the matrix can be performed in parallel

	-0	A <sub>1</sub>	C <sub>2</sub>	T <sub>3</sub>	G <sub>4</sub>	C <sub>5</sub>
-0	0	← -2	← -4	← -6	← -8	← -10
C <sub>1</sub>	↑ -2	↖ -1	↖ -1	← -3	← -5	←↖ -7
T <sub>2</sub>	↑ -4					
G <sub>3</sub>	↑ -6					

# Needleman-Wunsch Algorithm

## 2. Fill Matrix

- Repeat for all  $D(i,j)$

- Bear in mind: Filling the matrix can be performed in parallel

	$-0$	$A_1$	$C_2$	$T_3$	$G_4$	$C_5$
$-0$	0	$\leftarrow -2$	$\leftarrow -4$	$\leftarrow -6$	$\leftarrow -8$	$\leftarrow -10$
$C_1$	$\uparrow -2$	$\nwarrow -1$	$\nwarrow -1$	$\leftarrow -3$	$\leftarrow -5$	$\leftarrow \nwarrow -7$
$T_2$	$\uparrow -4$	$\nwarrow \uparrow -3$	$\nwarrow -2$	$\nwarrow 0$	$\leftarrow -2$	$\leftarrow -4$
$G_3$	$\uparrow -6$	$\nwarrow \uparrow -5$	$\nwarrow \uparrow -4$	$\uparrow -2$	$\nwarrow 1$	$\leftarrow -1$

# Needleman-Wunsch Algorithm




## 3. Perform back tracing to determine best global alignment

- Trace path back from  $D(m,n)$  to origin  $D(0,0)$  based on your decision during alignment.
























	$-0$	$A_1$	$C_2$	$T_3$	$G_4$	$C_5$
$-0$	0	$\leftarrow -2$	$\leftarrow -4$	$\leftarrow -6$	$\leftarrow -8$	$\leftarrow -10$
$C_1$	$\uparrow -2$	$\nearrow -1$	$\nearrow -1$	$\leftarrow -3$	$\leftarrow -5$	$\leftarrow \nearrow -7$
$T_2$	$\uparrow -4$	$\nearrow \uparrow -3$	$\nearrow -2$	$\nearrow 0$	$\leftarrow -2$	$\leftarrow -4$
$G_3$	$\uparrow -6$	$\nearrow \uparrow -5$	$\nearrow \uparrow -4$	$\uparrow -2$	$\nearrow 1$	$\leftarrow -1$

# Needleman-Wunsch Algorithm Questions?

- Reference: ACTGC
- Alignment: -CTG-
- Score of the alignment is: -1

-  Match or mismatch
-  Gap / Insertion
-  Gap / Deletion

- Bear in mind: Back tracing can be performed in parallel

	-0	A <sub>1</sub>	C <sub>2</sub>	T <sub>3</sub>	G <sub>4</sub>	C <sub>5</sub>
-0	0	 -2	 -4	 -6	 -8	 -10
C <sub>1</sub>	 -2	 -1	 -1	 -3	 -5	 -7
T <sub>2</sub>	 -4	 -3	 -2	 0	 -2	 -4
G <sub>3</sub>	 -6	 -5	 -4	 -2	 1	 -1

## Genome Data Acquisition and Processing

Data Management for  
Digital Health, Winter  
2023  
41



# Selected Alignment Algorithms

## Smith-Waterman Algorithm

---

- Determine the best local alignment
- Adaption of Needleman-Wunsch algorithm
  - Initialize all cells within first row and column with zero
  - Alignment score is defined by highest value somewhere in the matrix
  - Backtracing from cell with alignment score to first cell containing zero
- **Smith**, T. F. and **Waterman**, M. S. (1981). "Identification of Common Molecular Subsequences" in "Molecular Biology", 147: 195-7.

# Smith-Waterman Algorithm

- What is a best local alignment for the sequence CTG and the reference ACTGC?

	-0	A <sub>1</sub>	C <sub>2</sub>	T <sub>3</sub>	G <sub>4</sub>	C <sub>5</sub>
-0						
C <sub>1</sub>						
T <sub>2</sub>						
G <sub>3</sub>						

# Smith-Waterman Algorithm

## 1. Matrix Initialization

- $M(0,0) = M(i,0) = M(0,j) = 0 \mid 0 \leq i \leq m, 0 \leq j \leq n$

	-0	A <sub>1</sub>	C <sub>2</sub>	T <sub>3</sub>	G <sub>4</sub>	C <sub>5</sub>
-0	0	← 0	← 0	← 0	← 0	← 0
C <sub>1</sub>	↑ 0					
T <sub>2</sub>	↑ 0					
G <sub>3</sub>	↑ 0					

# Smith-Waterman Algorithm

## 2. Fill Matrix

- Weight function  $w(a,b) := \begin{cases} +1 & \text{if } a == b // \text{ Match} \\ -1 & \text{if } a != b // \text{ Mismatch} \end{cases}$
- $\text{gap}() := -2$ , i.e. gap cost function
- $D(i,j)$  defines value of matrix at coordinates  $(i,j)$

	-	A <sub>1</sub>	C <sub>2</sub>	T <sub>3</sub>	G <sub>4</sub>	C <sub>5</sub>
-	0	← 0	← 0	← 0	← 0	← 0
C <sub>1</sub>	↑ 0					
T <sub>2</sub>	↑ 0					
G <sub>3</sub>	↑ 0					

# Smith-Waterman Algorithm

## 2. Fill Matrix

- $D(1,1) :=$  maximum of
  - ①  $D(0,0) + w(A_1, C_1) = 0 + (-1) = -1$  // Match or mismatch

	-0	A <sub>1</sub>	C <sub>2</sub>	T <sub>3</sub>	G <sub>4</sub>	C <sub>5</sub>
-0	0	① ← 0	← 0	← 0	← 0	← 0
C <sub>1</sub>	↑ 0	D(1,1)				
T <sub>2</sub>	↑ 0					
G <sub>3</sub>	↑ 0					

# Smith-Waterman Algorithm

## 2. Fill Matrix

- $D(1,1) :=$  maximum of
  - ①  $D(0,0) + w(A_1, C_1) = 0 + (-1) = -1$  // Match or mismatch
  - ②  $D(0,1) + \text{gap}() = 0 + (-2) = -2$  // Deletion

	-0	A <sub>1</sub>	C <sub>2</sub>	T <sub>3</sub>	G <sub>4</sub>	C <sub>5</sub>
-0	0	← 0	← 0	← 0	← 0	← 0
C <sub>1</sub>	↑ 0	D(1,1)				
T <sub>2</sub>	↑ 0					
G <sub>3</sub>	↑ 0					

# Smith-Waterman Algorithm

## 2. Fill Matrix

- $D(1,1) :=$  maximum of
  - ①  $D(0,0) + w(A_1, C_1) = 0 + (-1) = -1$  // Match or mismatch
  - ②  $D(0,1) + \text{gap}() = 0 + (-2) = -2$  // Deletion
  - ③  $D(1,0) + \text{gap}() = 0 + (-2) = -2$  // Insertion

	-0	A <sub>1</sub>	C <sub>2</sub>	T <sub>3</sub>	G <sub>4</sub>	C <sub>5</sub>
-0	0	③ ← 0	← 0	← 0	← 0	← 0
C <sub>1</sub>	↑ 0	① D(1,1)				
T <sub>2</sub>	↑ 0	②				
G <sub>3</sub>	↑ 0					



# Smith-Waterman Algorithm

## 2. Fill Matrix

- $D(1,1) :=$  maximum of
  - ①  $D(0,0) + w(A_1, C_1) = 0 + (-1) = -1$  // Match or mismatch
  - ②  $D(0,1) + \text{gap}() = 0 + (-2) = -2$  // Deletion
  - ③  $D(1,0) + \text{gap}() = 0 + (-2) = -2$  // Insertion
  - ④ 0 // Default

	-0	A <sub>1</sub>	C <sub>2</sub>	T <sub>3</sub>	G <sub>4</sub>	C <sub>5</sub>
-0	0	0	← 0	← 0	← 0	← 0
C <sub>1</sub>	↑ 0	D(1,1)				
T <sub>2</sub>	↑ 0					
G <sub>3</sub>	↑ 0					

Diagram illustrating the Smith-Waterman algorithm matrix filling process. The matrix is a 4x7 grid. The first row and column are labeled with -0, A<sub>1</sub>, C<sub>2</sub>, T<sub>3</sub>, G<sub>4</sub>, and C<sub>5</sub>. The first cell (0,0) contains 0. The first row (0,1) to (0,6) contains 0, ← 0, ← 0, ← 0, ← 0, ← 0. The first column (1,0) to (3,0) contains ↑ 0, ↑ 0, ↑ 0. The cell (1,1) contains D(1,1). Red arrows indicate the path from (0,0) to (1,1) via (0,1) and (1,0). A red arrow also points from (1,1) to (1,2). Circled numbers 1, 2, 3, and 4 are placed near the arrows corresponding to the list items.

# Smith-Waterman Algorithm

## 2. Fill Matrix

- $D(1,1) :=$  maximum of
  - ①  $D(0,0) + w(A_1, C_1) = 0 + (-1) = -1$  // Match or mismatch
  - ②  $D(0,1) + \text{gap}() = 0 + (-2) = -2$  // Deletion
  - ③  $D(1,0) + \text{gap}() = 0 + (-2) = -2$  // Insertion
  - ④ 0 // Default

	-0	A <sub>1</sub>	C <sub>2</sub>	T <sub>3</sub>	G <sub>4</sub>	C <sub>5</sub>
-0	0	0	← 0	← 0	← 0	← 0
C <sub>1</sub>	↑ 0	0	← 0			
T <sub>2</sub>	↑ 0					
G <sub>3</sub>	↑ 0					

Diagram illustrating the Smith-Waterman algorithm matrix filling process. The matrix is a 4x7 grid with columns labeled -0, A<sub>1</sub>, C<sub>2</sub>, T<sub>3</sub>, G<sub>4</sub>, and C<sub>5</sub>, and rows labeled -0, C<sub>1</sub>, T<sub>2</sub>, and G<sub>3</sub>. The value 0 is shown in the top-left cell (row -0, column -0). Red arrows indicate the calculation of D(1,1) from its neighbors: a horizontal arrow from (0,0) to (1,1) labeled ①, a vertical arrow from (0,1) to (1,1) labeled ②, and a diagonal arrow from (0,0) to (1,1) labeled ③. A yellow arrow labeled ④ points to the cell (1,1), indicating the default value 0 is chosen as the maximum.

# Smith-Waterman Algorithm

## 2. Fill Matrix

- Repeat for all  $D(i,j)$  until matrix is filled

- Bear in mind: Filling the matrix can be performed in parallel

	$-0$	$A_1$	$C_2$	$T_3$	$G_4$	$C_5$
$-0$	0	$\leftarrow 0$	$\leftarrow 0$	$\leftarrow 0$	$\leftarrow 0$	$\leftarrow 0$
$C_1$	$\uparrow 0$	$\leftarrow \nwarrow \uparrow 0$	$\nwarrow 1$	$\leftarrow \nwarrow \uparrow 0$	$\leftarrow \nwarrow \uparrow 0$	$\nwarrow 1$
$T_2$	$\uparrow 0$	$\leftarrow \nwarrow \uparrow 0$	$\leftarrow \nwarrow \uparrow 0$	$\nwarrow 2$	$\leftarrow \nwarrow \uparrow 0$	$\leftarrow \nwarrow \uparrow 0$
$G_3$	$\uparrow 0$	$\leftarrow \nwarrow \uparrow 0$	$\leftarrow \nwarrow \uparrow 0$	$\leftarrow \nwarrow \uparrow 0$	$\nwarrow 3$	$\leftarrow 1$

# Smith-Waterman Algorithm




## 3. Perform back tracing to determine local alignments

- Trace path back from  $\max(D(i,j))$  to first  $D(i,j) = 0$  on your path

	-0	A <sub>1</sub>	C <sub>2</sub>	T <sub>3</sub>	G <sub>4</sub>	C <sub>5</sub>
-0	0	← 0	← 0	← 0	← 0	← 0
C <sub>1</sub>	↑ 0	←↖ ↑ 0	↖ 1	←↖ ↑ 0	←↖ ↑ 0	↖ 1
T <sub>2</sub>	↑ 0	←↖ ↑ 0	←↖ ↑ 0	↖ 2	←↖ ↑ 0	←↖ ↑ 0
G <sub>3</sub>	↑ 0	←↖ ↑ 0	←↖ ↑ 0	←↖ ↑ 0	↖ 3	← 1

# Smith-Waterman Algorithm Questions?

- Reference: ACTGC
- Local alignment: CTG
- Score of the alignment is: 3
  
- Bear in mind: Back tracing can be performed in parallel for multiple local optima

-  Match or mismatch
-  Gap / Insertion
-  Gap / Deletion

	-0	A <sub>1</sub>	C <sub>2</sub>	T <sub>3</sub>	G <sub>4</sub>	C <sub>5</sub>
-0	0	← 0	← 0	← 0	← 0	← 0
C <sub>1</sub>	↑ 0	←↖ ↑ 0	↖ 1	←↖ ↑ 0	←↖ ↑ 0	↖ 1
T <sub>2</sub>	↑ 0	←↖ ↑ 0	←↖ ↑ 0	↖ 2	←↖ ↑ 0	←↖ ↑ 0
G <sub>3</sub>	↑ 0	←↖ ↑ 0	←↖ ↑ 0	←↖ ↑ 0	↖ 3	← 1

## Genome Data Acquisition and Processing

Data Management for  
Digital Health, Winter  
2023  
53

# Selected Alignment Algorithms

## Burrows-Wheeler Aligner

---

- Alignment of short read against long reference sequence
- Uses Burrows-Wheeler Transform (BWT) to optimize search
- **BWT** := Rearrangement of character string, which aims to group similar characters by rotation and lexicographic ordering aka block-sorting compression
- → BWT output might be more applicable for compression schemes
  
- Li, H. and Durbin, R. (2009). "Fast and accurate short read alignment with Burrows-Wheeler transform" in "Bioinformatics", 25(14): 1754-60

# Burrows-Wheeler Transform Example

- BWT of character string: \*ENGINEERING#
  
- Assumptions:
  - \* = START
  - # = END
  - $\text{num}(\text{START}) < \text{num}(\text{END})$ , i.e. numeric representation of START is smaller than END

# Burrows-Wheeler Transform

## BWT("ENGINEERING")=?

1. Note down all rotations, i.e. move the word character by character

1	*ENGINEERING#
2	ENGINEERING#*
3	NGINEERING#*E
4	GINEERING#*EN
5	INEERING#*ENG
6	NEERING#*ENGI
7	EERING#*ENGINE
8	ERING#*ENGINEE
9	RING#*ENGINEER
10	ING#*ENGINEERI
11	NG#*ENGINEERIN
12	G#*ENGINEERING
13	#*ENGINEERING

### Genome Data Acquisition and Processing

Data Management for  
Digital Health, Winter  
2023  
56



# Burrows-Wheeler Transform

## BWT("ENGINEERING")=?

1. Note down all rotations, i.e. move the word character by character
2. Order rotations lexicographically based on first character

7	EERING#*ENGIN
2	ENGINEERING#*
8	ERING#*ENGINE
4	GINEERING#*EN
12	G#*ENGINEERIN
5	INEERING#*ENG
10	ING#*ENGINEER
6	NEERING#*ENGI
3	NGINEERING#*E
11	NG#*ENGINEERI
9	RING#*ENGINEE
1	*ENGINEERING#
13	#*ENGINEERING

### Genome Data Acquisition and Processing

Data Management for  
Digital Health, Winter  
2023  
57

# Burrows-Wheeler Transform

## BWT("ENGINEERING")=?

1. Note down all rotations, i.e. move the word character by character
2. Order rotations lexicographically based on first character
3. Assemble result, i.e. read all rotations by last character

7	EERING#*ENGIN
2	ENGINEERING#*
8	ERING#*ENGINE
4	GINEERING#*EN
12	G#*ENGINEERIN
5	INEERING#*ENG
10	ING#*ENGINEER
6	NEERING#*ENGI
3	NGINEERING#*E
11	NG#*ENGINEERI
9	RING#*ENGINEE
1	*ENGINEERING#
13	#*ENGINEERING

### Genome Data Acquisition and Processing

Data Management for  
Digital Health, Winter  
2023  
58

# Burrows-Wheeler Transform

## BWT("ENGINEERING")=?

- BWT("\*ENGINEERING#") = N\*ENNGRIEIE#G
- → N\*EN<sup>2</sup>GR(IE)<sup>2</sup>#G // run-length encoding (RLE)
- Bear in mind: BWT does not always improve compressibility!

7	EERING#*ENGIN
2	ENGINEERING#*
8	ERING#*ENGINE
4	GINEERING#*EN
12	G#*ENGINEERIN
5	INEERING#*ENG
10	ING#*ENGINEER
6	NEERING#*ENGI
3	NGINEERING#*E
11	NG#*ENGINEERI
9	RING#*ENGINEE
1	*ENGINEERING#
13	#*ENGINEERING

# Burrows-Wheeler Transform Inverse

$BWT^{-1}("NR^*A^3G\#M")=?$

1. Add given characters to *BWT* column, add indices for multiple occurrences of the same letter

BWT			
N <sub>1</sub>			
R <sub>1</sub>			
*			
A <sub>1</sub>			
A <sub>2</sub>			
A <sub>3</sub>			
G <sub>1</sub>			
#			
M <sub>1</sub>			

# Burrows-Wheeler Transform Inverse

## $BWT^{-1}("NR^*A^3G\#M")=?$

1. Add given characters to *BWT* column, add indices for multiple occurrences of the same letter
2. Order all characters lexicographically in *Sorted* column

BWT		Sorted	
N <sub>1</sub>	→	A <sub>1</sub>	
R <sub>1</sub>	→	A <sub>2</sub>	
*	→	A <sub>3</sub>	
A <sub>1</sub>	→	G <sub>1</sub>	
A <sub>2</sub>	→	M <sub>1</sub>	
A <sub>3</sub>	→	N <sub>1</sub>	
G <sub>1</sub>	→	R <sub>1</sub>	
#	→	*	
M <sub>1</sub>	→	#	

# Burrows-Wheeler Transform Inverse

$BWT^{-1}("NR^*A^3G\#M")=?$

1. Add given characters to *BWT* column, add indices for multiple occurrences of the same letter
2. Order all characters lexicographically in *Sorted* column
3. Lookup terminal '#' in *BWT* column (because we know it is the end of our string), add corresponding character from *Sorted* column to the output.

BWT		Sorted	
N <sub>1</sub>	→	A <sub>1</sub>	
R <sub>1</sub>	→	A <sub>2</sub>	
*	→	A <sub>3</sub>	
A <sub>1</sub>	→	G <sub>1</sub>	
A <sub>2</sub>	→	M <sub>1</sub>	
A <sub>3</sub>	→	N <sub>1</sub>	
G <sub>1</sub>	→	R <sub>1</sub>	
#	→	*	1
M <sub>1</sub>	→	#	

# Burrows-Wheeler Transform Inverse

## $BWT^{-1}("NR^*A^3G\#M")=?$

1. Add given characters to *BWT* column, add indices for multiple occurrences of the same letter
2. Order all characters lexicographically in *Sorted* column
3. Lookup terminal '#' in *BWT* column (because we know it is the end of our string), add corresponding character from *Sorted* column to the output.
4. Continue to lookup character from last *Sorted* column in *BWT*, add corresponding character from *Sorted* to output.

BWT		Sorted	Seq
N <sub>1</sub>	→	A <sub>1</sub>	4
R <sub>1</sub>	→	A <sub>2</sub>	7
*	→	A <sub>3</sub>	2
A <sub>1</sub>	→	G <sub>1</sub>	5
A <sub>2</sub>	→	M <sub>1</sub>	8
A <sub>3</sub>	→	N <sub>1</sub>	3
G <sub>1</sub>	→	R <sub>1</sub>	6
#	→	*	1
M <sub>1</sub>	→	#	9

# Burrows-Wheeler Transform Inverse

$BWT^{-1}("NR^*A^3G\#M")=?$

1. Add given characters to *BWT* column, add indices for multiple occurrences of the same letter
2. Order all characters lexicographically in *Sorted* column
3. Lookup terminal '#' in *BWT* column (because we know it is the end of our string), add corresponding character from *Sorted* column to the output.
4. Continue to lookup character from last *Sorted* column in *BWT*, add corresponding character from *Sorted* to output.
5. Assemble output by sorting according to *Seq* column.

1	2	3	4	5	6	7	8	9
*	A <sub>3</sub>	N <sub>1</sub>	A <sub>1</sub>	G <sub>1</sub>	R <sub>1</sub>	A <sub>2</sub>	M <sub>1</sub>	#

BWT		Sorted	Seq
N <sub>1</sub>	→	A <sub>1</sub>	4
R <sub>1</sub>	→	A <sub>2</sub>	7
*	→	A <sub>3</sub>	2
A <sub>1</sub>	→	G <sub>1</sub>	5
A <sub>2</sub>	→	M <sub>1</sub>	8
A <sub>3</sub>	→	N <sub>1</sub>	3
G <sub>1</sub>	→	R <sub>1</sub>	6
#	→	*	1
M <sub>1</sub>	→	#	9

## Genome Data Acquisition and Processing

Data Management for  
Digital Health, Winter  
2023  
64



# Selected Alignment Tools

---

- BWA: Smith-Waterman + BWT to keep memory footprint low
- Bowtie: Similar to Smith-Water/Needleman-Wunsch + BWT
- HANA Aligner (based on IMDB): BWA + FM index/BWT to speed-up match detection
- Isaac (commercialized by Illumina): Smith-Waterman
- Torrent Mapping Alignment Program (TMAP) (commercialized by IonTorrent): Smith-Waterman + FM index/BWT

# What to take home?

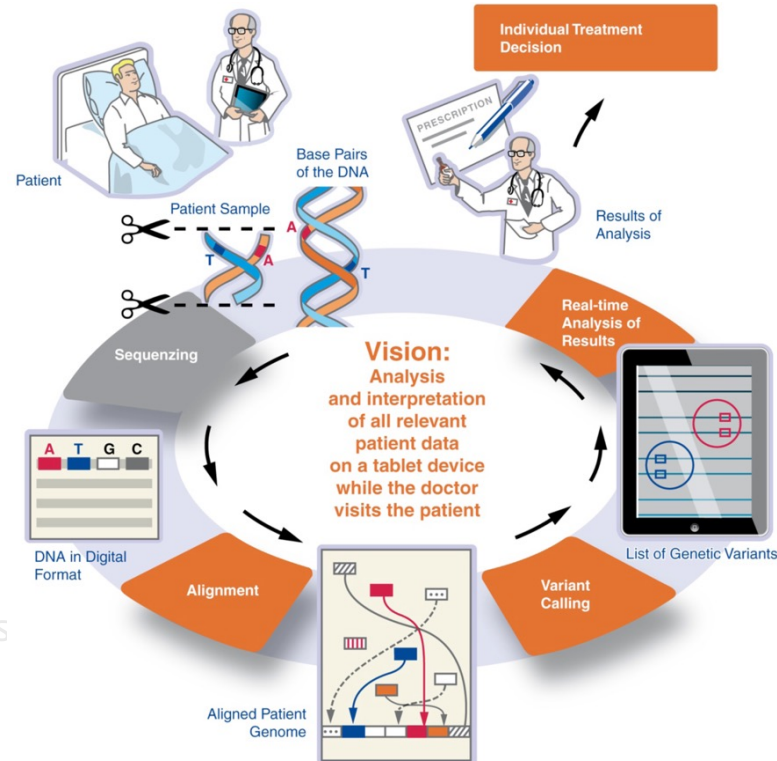
## Alignment strategies

	Needleman-Wunsch	Smith-Waterman
Purpose	Global alignment	Local alignment
Matrix values $D(i,j)$	$\in \mathbb{Z}$	$\in \mathbb{N}_0$
Initialization values	$\in -\mathbb{N}_0$	0
Data structure	Matrix	Matrix
Alignment score	Bottom right cell	Highest matrix value

- BWT aims to group similar characters → might support compression

# From Raw Genome Data to Analysis

- **DNA Sequencing:** Transformation of analogues DNA into digital format
- **Alignment:** Reconstruction of complete genome with snippets
- **Variant Calling:** Identification of genetic variants
- **Data Annotation:** Linking genetic variants with research findings

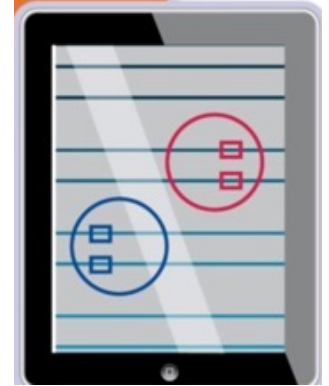


## Genome Data Acquisition and Processing

Data Management for  
Digital Health, Winter  
2023  
67

# Variant Calling Overview

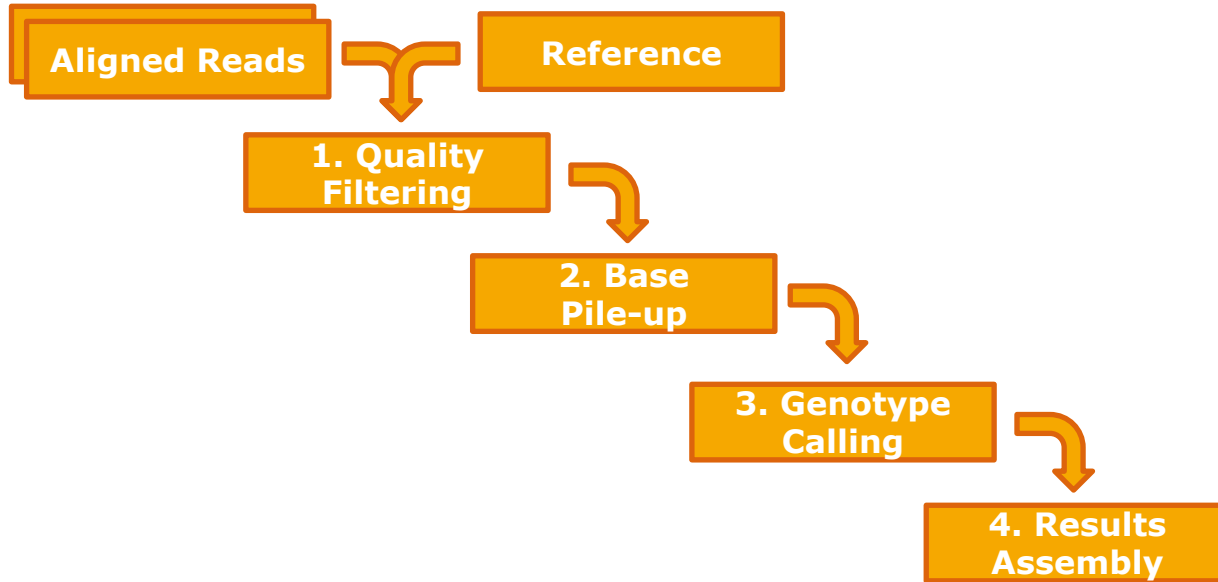
- **Variant Calling** := Detection of variations within a genome
- Input:
  - Mapped DNA reads, i.e. output of alignment process
  - Reference genome
- Output: List of variants
  
- Bear in mind:
  - **Read depth at  $pos_i$**  := Number of nucleotides storing information about  $pos_i$



## Genome Data Acquisition and Processing

Data Management for  
Digital Health, Winter  
2023  
68

# Variant Calling Process



# 1. Quality Filtering

- Extract locations from mapped reads where mapping issues were detected

- Reference

A	C	G	C	R	A	G	A	T	A
-	-	G	C	A	T	G	A	T	A
2D		8M							

- Read

- CIGAR

Op	BAM	Description
M	0	alignment match (can be a sequence match or mismatch)
I	1	insertion to the reference
D	2	deletion from the reference
N	3	skipped region from the reference
S	4	soft clipping (clipped sequences present in SEQ)
H	5	hard clipping (clipped sequences NOT present in SEQ)
P	6	padding (silent deletion from padded reference)
=	7	sequence match
X	8	sequence mismatch

## 2. Base Pile-up

- Reference (FASTA)
- Aligned read 1

A	C	G	C	R	A	G	A	T	A
		G	C	A	T	G	A	T	A

## 2. Base Pile-up

- Reference (FASTA)

- Aligned read 1

...

- Aligned read 8

- Alleles

A	C	G	C	R	A	G	A	T	A
		G	C	A	T	G	A	T	A
A	C	G	C	G	T	G	A	T	A
A	T	G	C	G	T	G	A		
A	C	G	C	G	A	G			
	C	G	C	A	T	G	A	T	A
A	C	G	C	G	T				
			C	A	T	G	A	T	A
	C	G	C	A	T	G	A	T	A

SNP or Read Error

a	4	5	7	8	4	1	7	6	5	5
b	0	1	0	0	4	7	0	0	0	0

↑  
aa

↑  
ab

↑  
bb

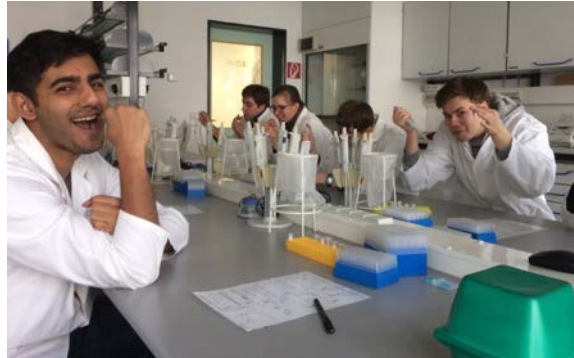
**Genome Data  
Acquisition and  
Processing**

Data Management for  
Digital Health, Winter  
2023  
72



# Reasons for Mismatches?

- Error(s) in the wet lab process
- Error(s) during alignment phase, i.e. incorrect mapping of individual DNA chunks
- Error(s) during base calling, i.e. algorithm only indicates probability
- Incorrect reference



- Bear in mind: Better references and algorithms may reduce the error!

# 3. Genotype Calling

- Purpose: Eliminate impact of noise and poor reading quality
- How: Compute probability for a genotype G given read context data D per sample
- Uses Bayes' theorem

- Recap Bayes' theorem:
$$P(A | B) = \frac{P(B | A) \cdot P(A)}{P(B)}$$
- Given are two events A and B:
  - P(A|B) defines the conditional probability for event A after event B
  - P(B|A) defines the conditional probability for event B after event A
- Relates conditional probability P(A|B) to P(B|A) for events A and B and P(B) > 0.

# 3. Genotype Calling

- Which genotype  $G$  has the highest posterior probability given the read data  $D$ ?
- Therefore, calculate posterior probability  $P(G|D)$ .

- Bayes' Theorem:

- $D$ : All observation about current position  $i$   $\{D_i, \dots, D_n\}$
- $P(G)$ : Genotype probability  $\{AA, AC, AG, AT, CC, CG, CT, GG, GT, TT\}$
- $P(G_i)$ : Prior probability
- $P(D|G_i)$ : Genotype likelihood

$$P(G|D) = \frac{P(D|G)P(G)}{P(D)}$$
$$= \frac{P(D|G) P(G)}{\sum_{i=1}^n P(D|G_i) P(G_i)}$$

**Genome Data  
Acquisition and  
Processing**

Data Management for  
Digital Health, Winter  
2023  
75

- Heng Li (2011): "A statistical framework for SNP calling, mutation discovery, association mapping and population genetical parameter estimation from seq. data"

# 3. Genotype Calling

## Computation of Prior Probability $P(G_i)$

- Affected by:
  - Number of (known) SNPs across the complete genome
  - Distribution of SNPs
  - Allele frequency

An example of prior probability for a dbSNP G/T site used in Li et al (2009)

	A	C	G	T
A	$4.55 \cdot 10^{-7}$	$9.11 \cdot 10^{-8}$	$9.1 \cdot 10^{-5}$	$9.1 \cdot 10^{-5}$
C		$4.55 \cdot 10^{-7}$	$9.1 \cdot 10^{-5}$	$9.1 \cdot 10^{-5}$
G			.454	.0909
T				.454

$$P(G|D) = \frac{P(D|G)P(G)}{P(D)}$$
$$= \frac{P(D|G) P(G)}{\sum_{i=1}^n P(D|G_i) P(G_i)}$$

**Genome Data  
Acquisition and  
Processing**

Data Management for  
Digital Health, Winter  
2023

76

### 3. Genotype Calling

## Computation of Genotype Likelihood $P(D|G_i)$

- Genotype likelihood depends on the surrounding data at position  $i$
- Includes present values and base quality scores from sequencing
- Where to find base quality score?
- Recap FASTQ file format
- Line 4 describes base quality score

```
@HJ40ITD02GKSZP rank=0016806 x=2580.0 y=819.0
CGTATCTACACAGGGTCAGGGTTCTGGATATAGGGCAGCACGGAC
+
FFFFFFFFFFFFD666ADD666??DFFFFFFHHHHHHHHIHHFFFFFFE
@HJ40ITD02F4FE5 rank=0016858 x=2393.0 y=2687.0
CGTATCTACACAGGGTCAGGGTTATGGATATCAGGTAACAGTCA
+
IIIIIIIIIIII@@@IIHHHHIIIIIGEEE@A<:5211121DDAD
@HJ40ITD02HNVGV rank=0017026 x=3025.5 y=893.0
CGTATCTACACAGGGTCAGGGTTCTGGATATTGGGGAGAATATGA
+
IIIIIIIIIIIIHHHIIHHHHIIIIIIIIIGG333390:C?@@@
@HJ40ITD02GIZMW rank=0017128 x=2559.5 y=2134.0
CGTATCTACACAGGGTCAGGGTTCTGGATATTGACCTAACTGCTG
+
IIIIIIIIIIIIHHHIIHHHIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
```

$$P(G|D) = \frac{P(D|G)P(G)}{P(D)}$$

$$= \frac{P(D|G)P(G)}{\sum_{i=1}^n P(D|G_i)P(G_i)}$$

**Genome Data Acquisition and Processing**

### 3. Genotype calling

## Genotype consensus

---

- Select the genotype with the highest probability, i.e.
  - $g_i = \operatorname{argmax} P(g_i|D)$  for  $g_i$  in ( $\langle a,a \rangle$ ,  $\langle a,b \rangle$ ,  $\langle b,b \rangle$ )
  - Assembly results for all positions  $i$ .
- 
- 
- 
- 
- 
- 
- 
- 
- 
- 
- Bear in mind: Variant calling can be performed for multiple loci in parallel.

## 4. Results Assembly

- Results are stored in Variant Calling Format (VCF)
- VCF is extensible, i.e. can store an arbitrary number of attribute/value pairs
- Result consists of:
  - Header defining attributes

##FORMAT=<ID=GT,Number=1,Type=String,Description="Genotype">

- One entry per variant (fixed number of attributes)

CHROM	POS	ID	REF	ALT	QUAL	FILTER	INFO	FORMAT	SAMPLE
chr7	14075333 6	rs11348802 2	T	A	61	PASS	NS=1	GT	0/1

- Li H, Ruan J, Durbin R (2008) Mapping short DNA sequencing reads and calling variants using mapping quality scores *Genome Research* 18:1851-1858
- Maq was the first widely used variant caller
- Latest examples:
  - Broad's Genome Analysis Tool Kit
    - Unified Genotyper
    - Haplotype Caller
    - ...

## Run Maq Now

Follow these steps to try Maq. All you need is a reference sequence file in the FASTA format.

1. Prepare a reference sequence (ref.fasta). Better a bacterial genome.
2. Download maq, maq-data and maqview at the [download page](#).
3. Copy maq, maq.pl and maq\_eval.pl to the \$PATH or to the same directory.
4. Simulate diploid reference and read sequences, map reads, call variants and evaluate the results in one go:

```
maq.pl demo ref.fasta calib-30.dat
```

where *calib-30.dat* is contained in maq-data.

5. View the alignment:

```
cd maqdemo/easyrun;  
maqindex -i -c consensus.cns all.map;  
maqview -c consensus.cns all.map
```

**Even for advanced maq users, running `maq.pl demo' is recommended. You may find something helpful.**

<http://maq.sourceforge.net/>

## Processing

Data Management for  
Digital Health, Winter  
2023  
80



# What To Take Home?

- Sequencing is comparable to a A/D converter resulting in sequences of nucleobases
- Alignment aims to map chunks / reads to their best fitting location compared to a given reference
- Variant calling derives the genotype for a given location based on observational and probabilistic data models



## Genome Data Acquisition and Processing

Data Management for  
Digital Health, Winter  
2023

81