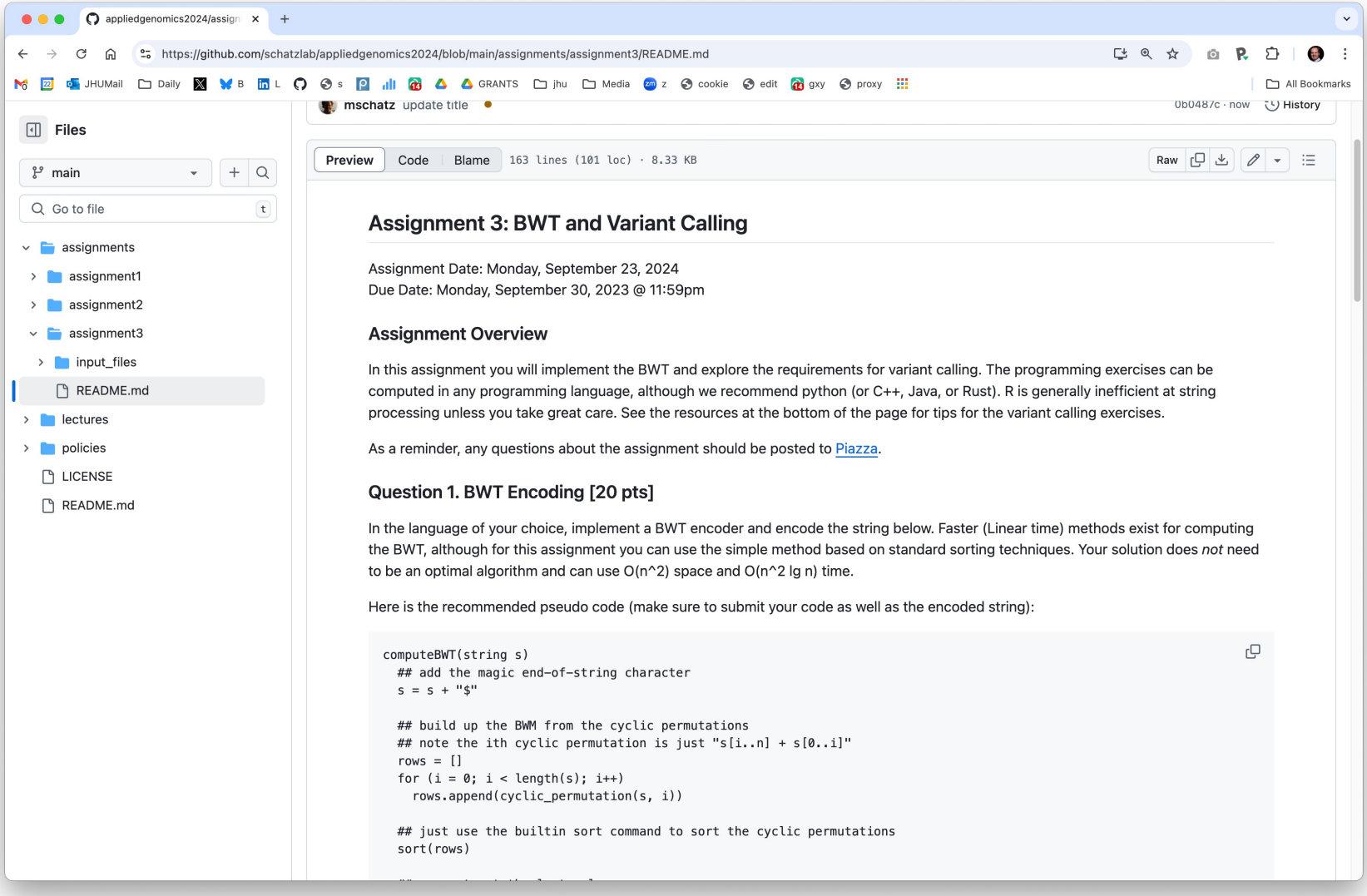
Variant Calling

Michael Schatz

Sept 23, 2024

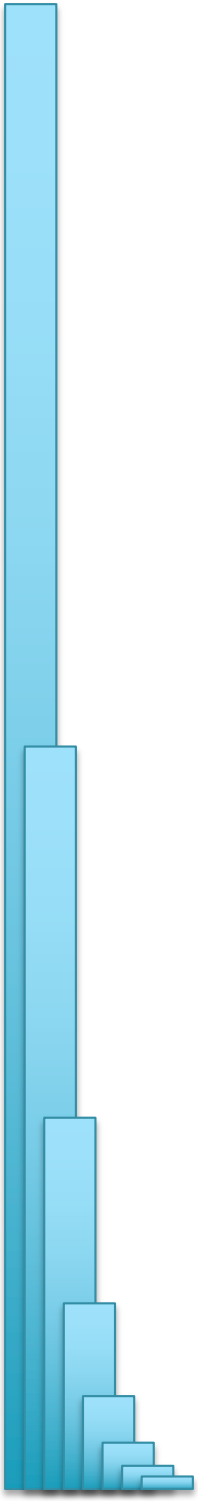
Lecture 8: Applied Comparative Genomics

Assignment 3:Variant Finding
Due Monday Sept 30 by 11:59pm



https://github.com/schatzlab/appliedgenomics2024/tree/main/assignments/assignment3

Check Piazza for questions!



Read Mapping

Exact Matching Review & Overview

Where is GATTACA in the human genome?

Brute Force (3 GB)

```
BANANA
BAN
  ANA
    NAN
      ANA
```

O(m * n)

Slow & Easy

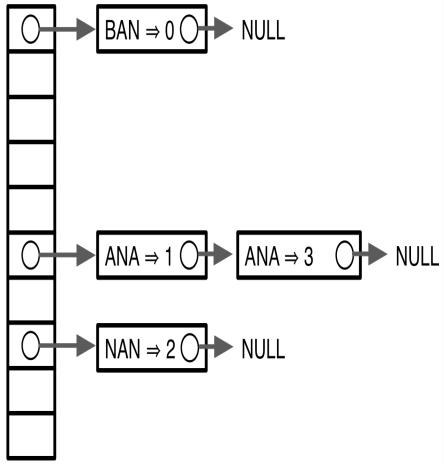
Suffix Array (>15 GB)

6	$
5	A$
3	ANA$
1	ANANA$
0	BANANA$
4	NA$
2	NANA$

O(m + lg n)

Full-text index

Hash Table (>15 GB)



O(1)

Fixed-length lookup

BWT (3 GB)

```
BANANA$
   =>
$BANAN A
A$BANA N
ANA$BA N
ANANA$ B
BANANA $
NA$BAN A
NANA$B A
   =>
ANNB$AA
```

O(m)

Full-text and concise

*** These are general techniques applicable to any text search problem ***

Bowtie2 Algorithm Overview

1. Split read into segments

Read: CCAGTAGCTCTCAGCCTTATTTTACCCAGGCCTGTA

Read (reverse complement): TACAGGCCTGGGTAAAATAAGGCTGAGAGCTACTGG

Policy: extract 16 nt seed every 10 nt

Seeds

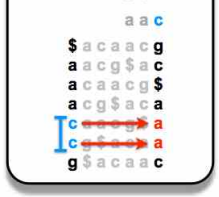
+, 0: CCAGTAGCTCTCAGCC
+, 10: TCAGCCTTATTTTACC
+, 20: TTTACCCAGGCCTGTA
-, 0: TACAGGCCTGGGTAAA
-, 10: GGTAAAATAAGGCTGA
-, 20: GGCTGAGAGCTACTGG

2. Lookup each segment and prioritize

Seeds

+, 0: CCAGTAGCTCTCAGCC
+, 10: TCAGCCTTATTTTACC
+, 20: TTTACCCAGGCCTGTA
-, 0: TACAGGCCTGGGTAAA
-, 10: GGTAAAATAAGGCTGA
-, 20: GGCTGAGAGCTACTGG

→ Ungapped alignment with FM Index



→ Seed alignments (as B ranges)

{ [211, 212], [212, 214] }
{ [653, 654], [651, 653] }
{ [684, 685] }
{ }
{ }
{ [624, 625] }

3. Evaluate end-to-end match

Extension candidates

SA:684, chr12:1955
SA:624, chr2:462
SA:211: chr4:762
SA:213: chr12:1935
SA:652: chr12:1945

→ SIMD dynamic programming aligner

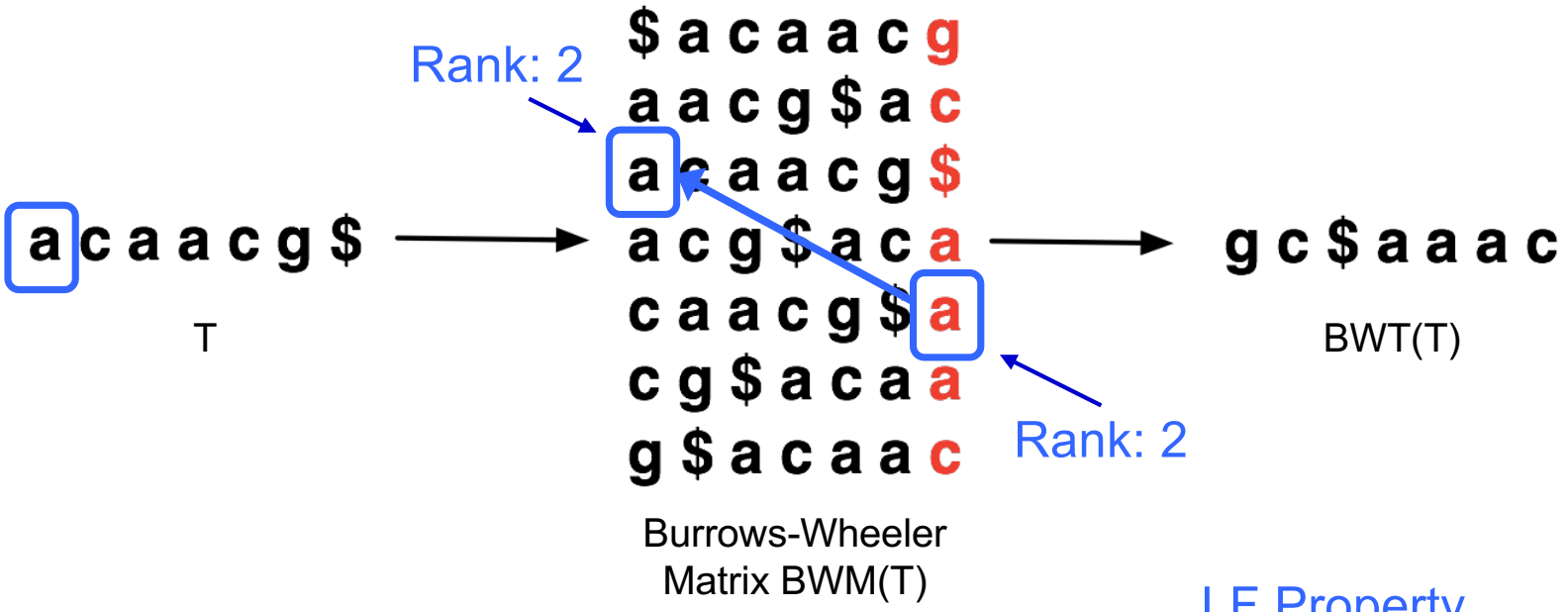
→ SAM alignments

```
r1   0    chr12    1936     0
     36M  *    0    0
     CCAGTAGCTCTCAGCCTTATTTTACCCAGGCCTGTA
     IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
     AS:i:0    XS:i:-2   XN:i:0
     XM:i:0    XO:i:0    XG:i:0
     NM:i:0    MD:Z:36   YT:Z:UU
     YM:i:0
...
```

(Langmead & Salzberg, 2012)

Burrows-Wheeler Transform

- Reversible permutation of the characters in a text



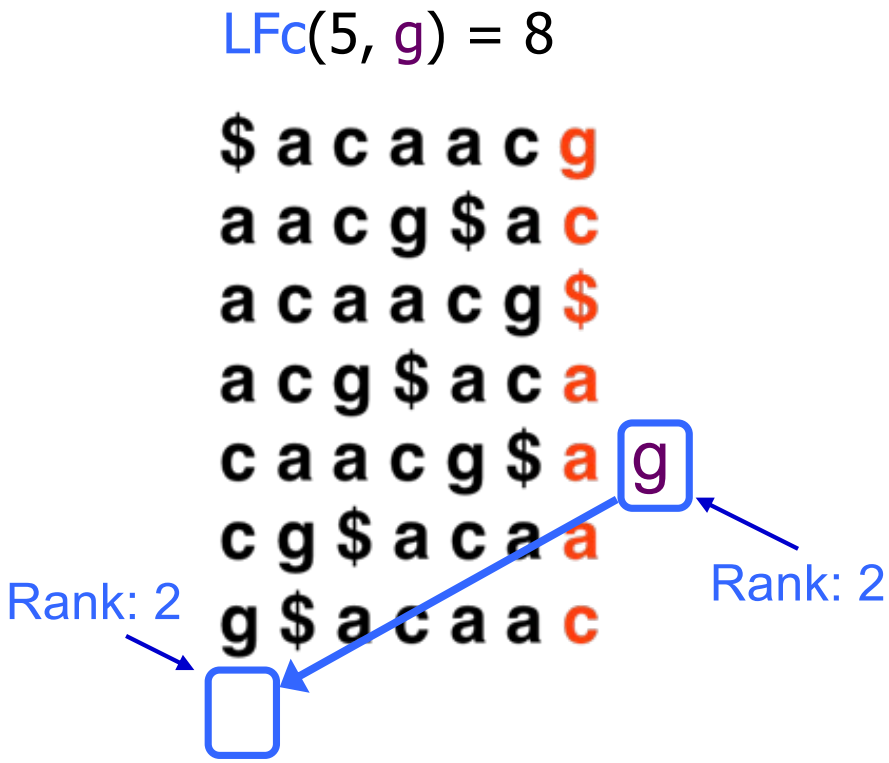
LF Property implicitly encodes Suffix Array

- BWT(T) is the index for T

A block sorting lossless data compression algorithm.
Burrows M, Wheeler DJ (1994) *Digital Equipment Corporation.* Technical Report 124

BWT Exact Matching

- **LFc**(r, c) does the same thing as **LF**(r) but it ignores r's actual final character and "pretends" it's c:

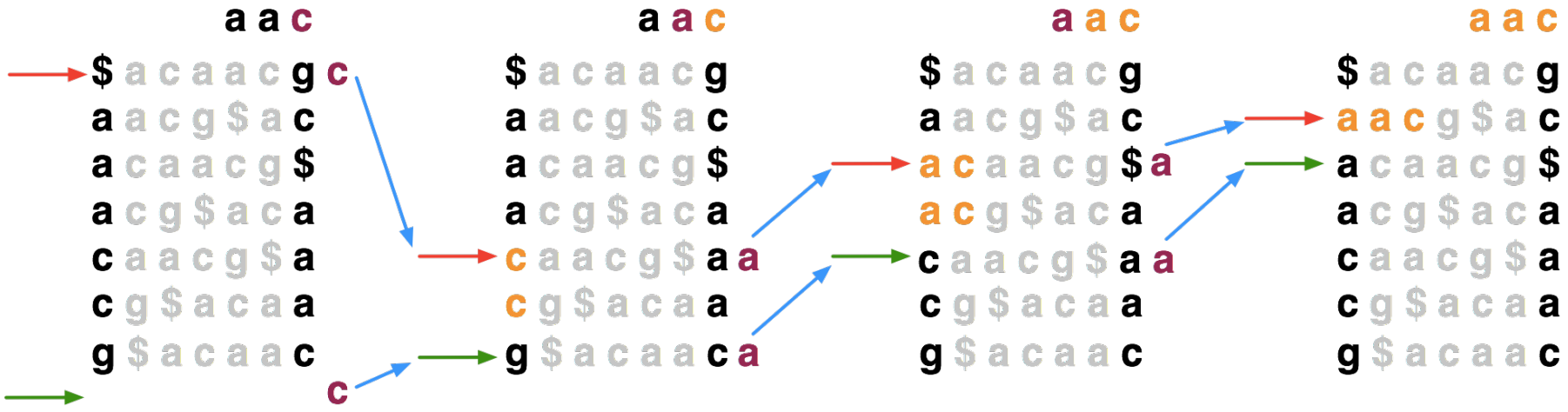


BWT Exact Matching

- Start with a range, (**top**, **bot**) encompassing all rows and repeatedly apply **LFc**:

  top = LFc(top, qc); bot = LFc(bot, qc)

  qc = the next character to the left in the query



Ferragina P, Manzini G: Opportunistic data structures with applications. *FOCS. IEEE Computer Society; 2000.*

[Search for TTA this BWT string: ACTGA$TTA]

FM Index

Where is this row in the original text?

- Could repeatedly apply LF transformation, but could require many steps: O(N)
- Instead, periodically record the position: sample the suffix array every 100th row

Which 'a' is this?

- Could count from the top of the matrix, but this will be very slow: O(n)
- Instead, periodically (every 100th row) keep track of the number of occurrences of the characters that have seen so far: $:1, a:2, c:1, g:1, t:0

FM-index: BWT + SA sample + count table
3Gb + (30M ints) + (30M * 5 ints)
3Gb + 120Mb + 600Mb = 3.72Gb

Opportunistic Data Structures with Applications
Ferragina and Manzini (2000). Proceedings of the 41st Annual Symposium on Foundations of Computer Science. p.390.

Algorithm Overview

1. Split read into segments

Read

CCAGTAGCTCTCAGCCTTATTTTACCCAGGCCTGTA

Read (reverse complement)

TACAGGCCTGGGTAAAATAAGGCTGAGAGCTACTGG

Policy: extract 16 nt seed every 10 nt

Seeds

+, 0: CCAGTAGCTCTCAGCC

+, 10: TCAGCCTTATTTTACC

+, 20: TTTACCCAGGCCTGTA

-, 0: TACAGGCCTGGGTAAA

-, 10: GGTAAAATAAGGCTGA

-, 20: GGCTGAGAGCTACTGG

2. Lookup each segment and prioritize

Seeds

+, 0: CCAGTAGCTCTCAGCC

+, 10: TCAGCCTTATTTTACC

+, 20: TTTACCCAGGCCTGTA

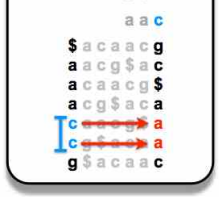
-, 0: TACAGGCCTGGGTAAA

-, 10: GGTAAAATAAGGCTGA

-, 20: GGCTGAGAGCTACTGG

→

Ungapped alignment with FM Index



→

Seed alignments (as B ranges)

{ [211, 212], [212, 214] }

{ [653, 654], [651, 653] }

{ [684, 685] }

{ }

{ }

{ [624, 625] }

3. Evaluate end-to-end match

Extension candidates

SA:684, chr12:1955

SA:624, chr2:462

SA:211: chr4:762

SA:213: chr12:1935

SA:652: chr12:1945

→

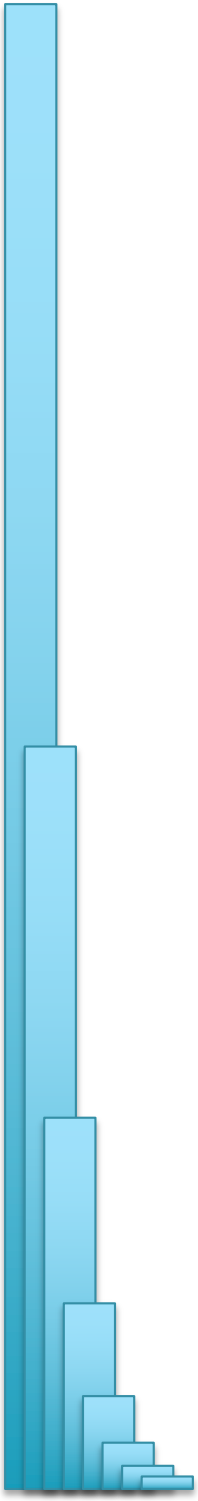
SIMD dynamic programming aligner

→

SAM alignments

```
r1   0    chr12    1936     0
     36M  *    0    0
     CCAGTAGCTCTCAGCCTTATTTTACCCAGGCCTGTA
     IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
     AS:i:0    XS:i:-2   XN:i:0
     XM:i:0    XO:i:0    XG:i:0
     NM:i:0    MD:Z:36   YT:Z:UU
     YM:i:0
...
```

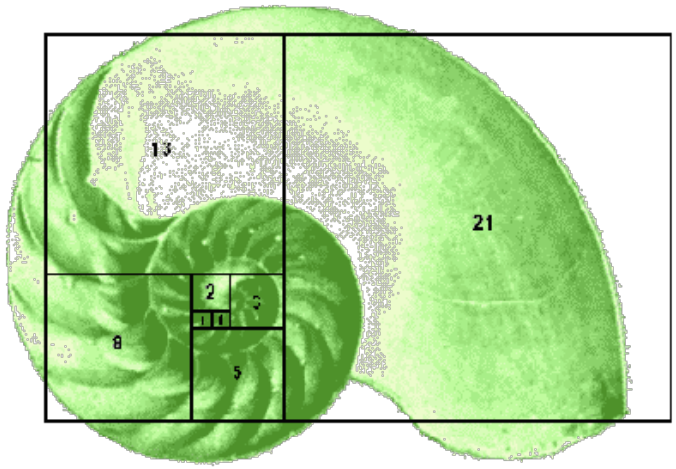
(Langmead & Salzberg, 2012)

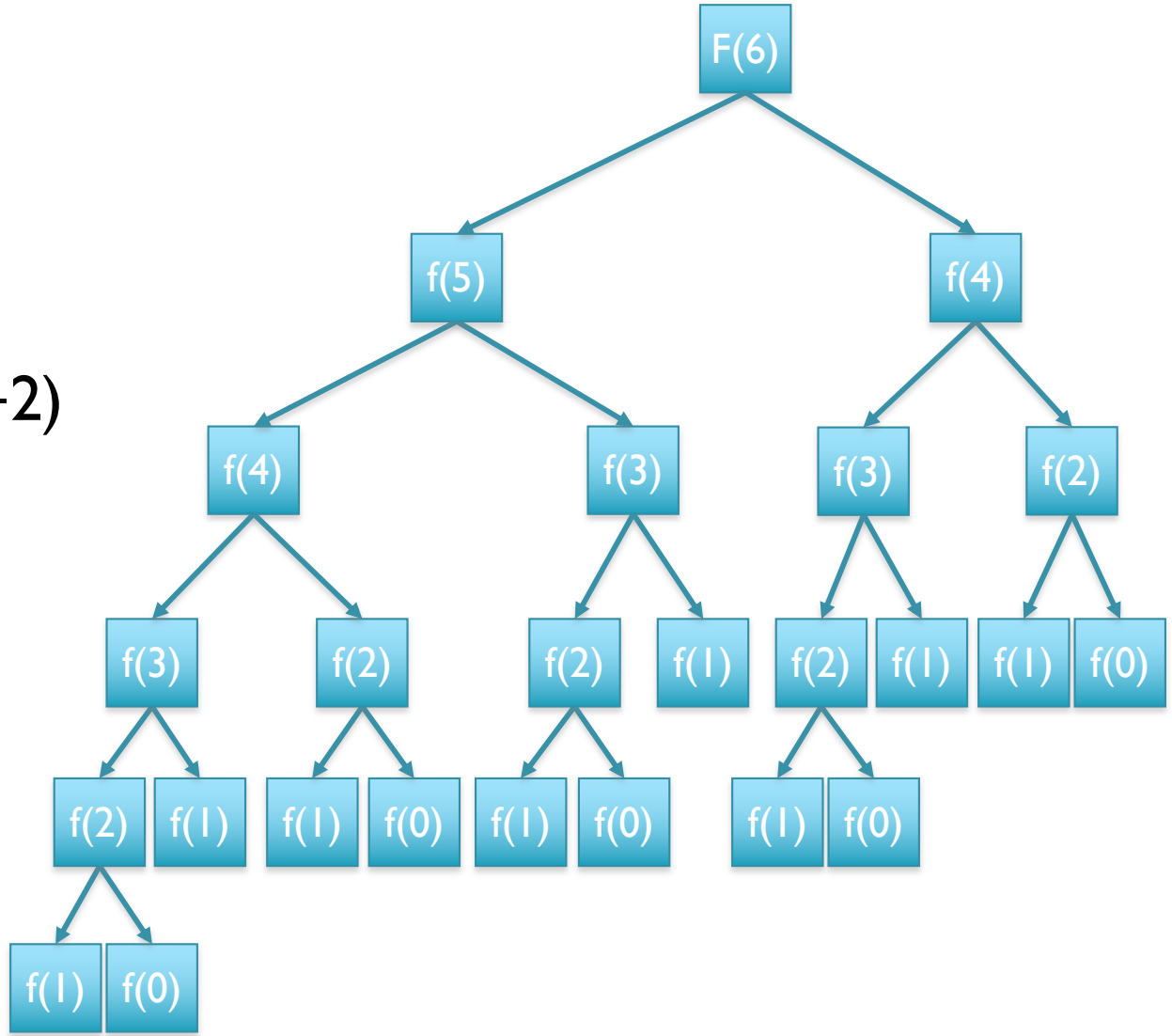


Dynamic Programming

Fibonacci Sequence

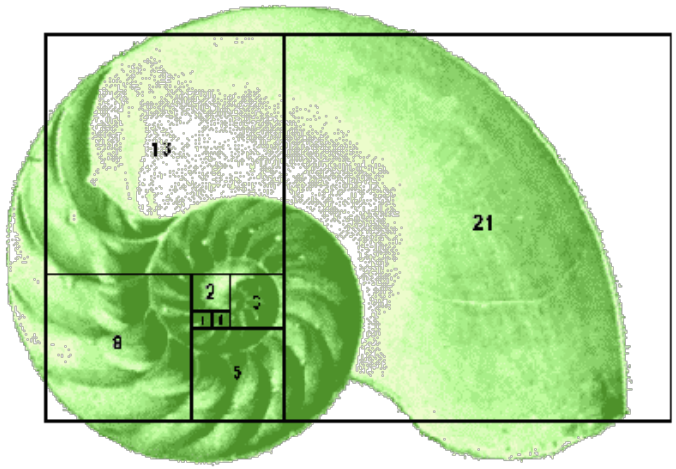
```
def fib(n):
  if n == 0 or n == 1:
     return n
  else:
  return fib(n−1) + fib(n−2)
```

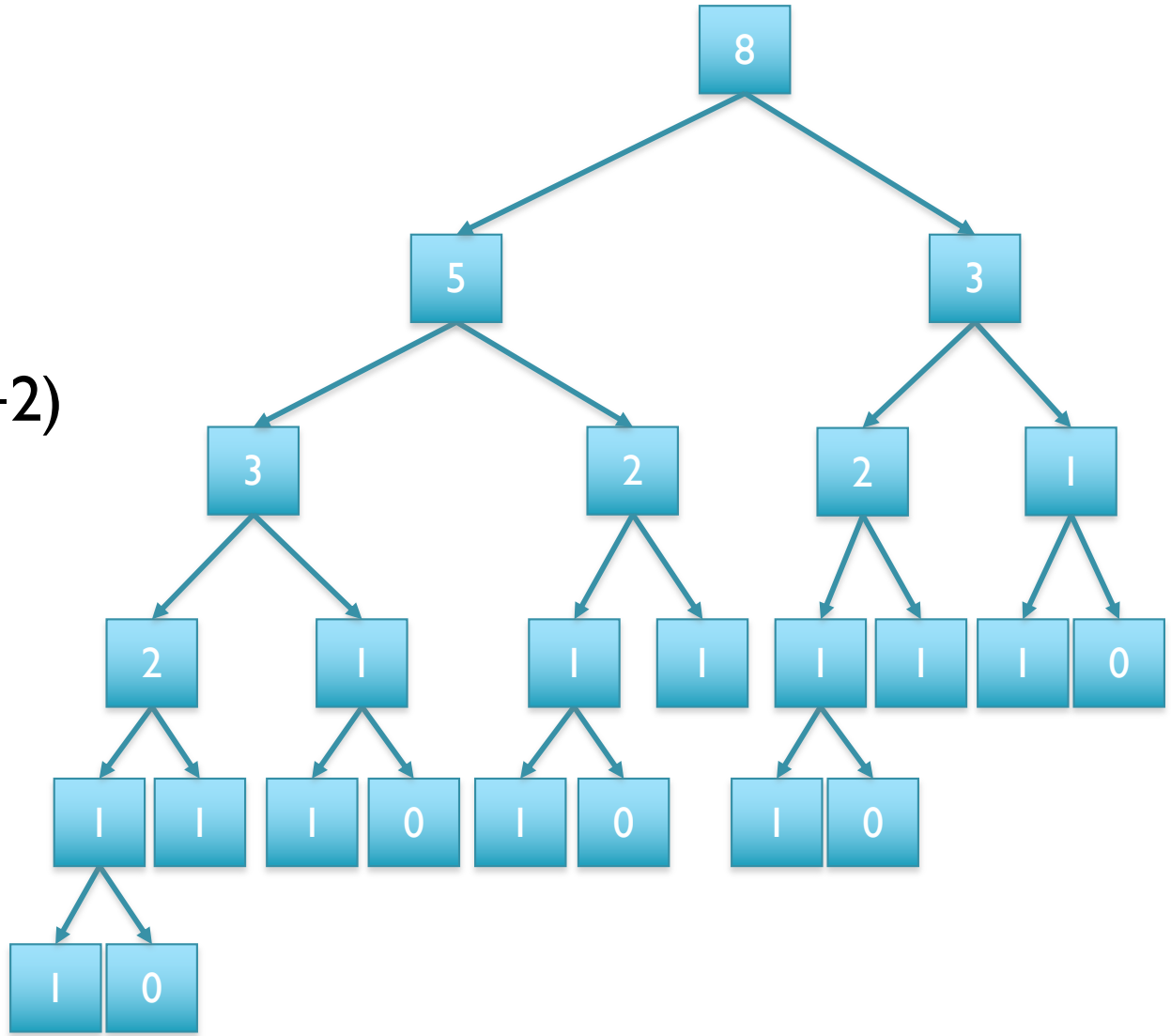




Fibonacci Sequence

```
def fib(n):
  if n == 0 or n == 1:
     return n
  else:
  return fib(n−1) + fib(n−2)
```





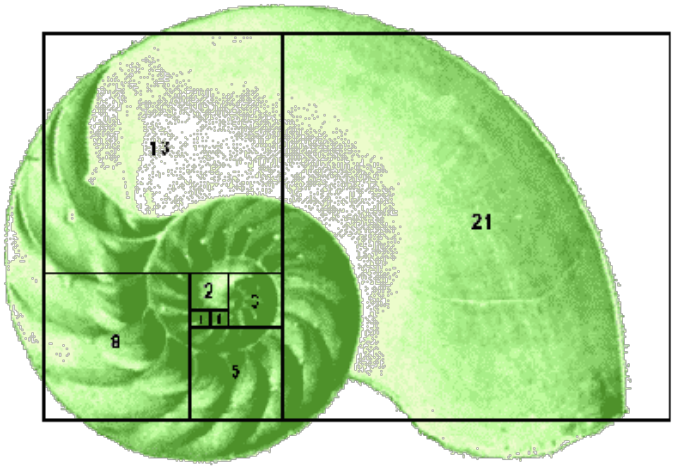
[What is the running time?]

Bottom-up Fibonacci Sequence

```
def fib(n):
  table = [0] * (n+1)
  table[0] = 0
  table[1] = 1
  for i in range(2,n+1):
    table[i] = table[i−2] + table[i−1]
return table[n]
```

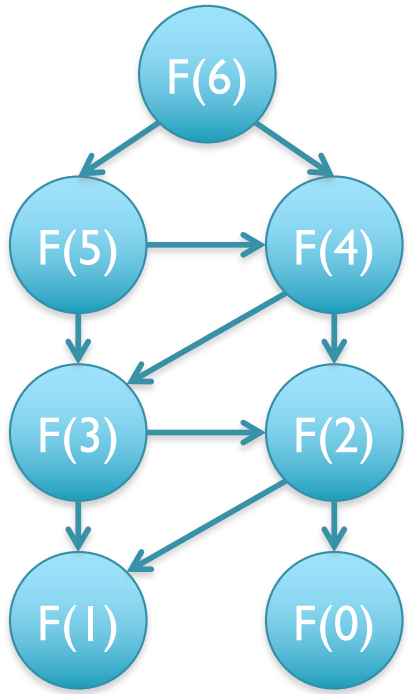
0	1	2	3	4	5	6
0	1	1	2	3	5	8

[What is the running time?]



Dynamic Programming

- General approach for solving (some) complex problems
  - When applicable, the method takes far less time than naive methods.
    - Polynomial time (O(n) or $O(n^2)$) instead of exponential time ($O(2^n)$ or $O(3^n)$)
- Requirements:
  - **Overlapping subproblems**
  - **Optimal substructure**
- Applications:
  - Fibonacci
  - Longest Increasing Subsequence (Bonus Slides!)
  - Sequence alignment, Dynamic Time Warp, Viterbi



- Not applicable:
  - Traveling salesman problem, Clique finding, Subgraph isomorphism, …
  - The cheapest flight from airport A to airport B involves a single connection through airport C, but the cheapest flight from airport A to airport C involves a connection through some other airport D.

Similarity metrics

- Hamming distance
  - Count the number of substitutions to transform one string into another

```
MIKESCHATZ
||X||XXXX|
MICESHATZZ
     5
```

- Edit distance
  - The minimum number of substitutions, insertions, or deletions to transform one string into another

```
MIKESCHAT-Z
||X||X|||X|
MICES-HATZZ
     3
```

Reverse Engineering Edit Distance

D(AGCACACA, ACACACTA) = ?

Imagine we already have the optimal alignment of the strings, the last column can only be 1 of 3 options:

…**M**	…**I**	…**D**
…A	…-	…A
…A	…A	…-

The optimal alignment of last two columns is then 1 of 9 possibilities

…**MM**	…**IM**	…**DM**	…**MI**	…**II**	…**DI**	…**MD**	…**ID**	…**DD**
…CA	…-A	…CA	…A-	…--	…A-	…CA	…-A	…CA
…TA	…TA	…-A	…TA	…TA	…-A	…A-	…A-	…--

The optimal alignment of the last three columns is then 1 of 27 possibilities…

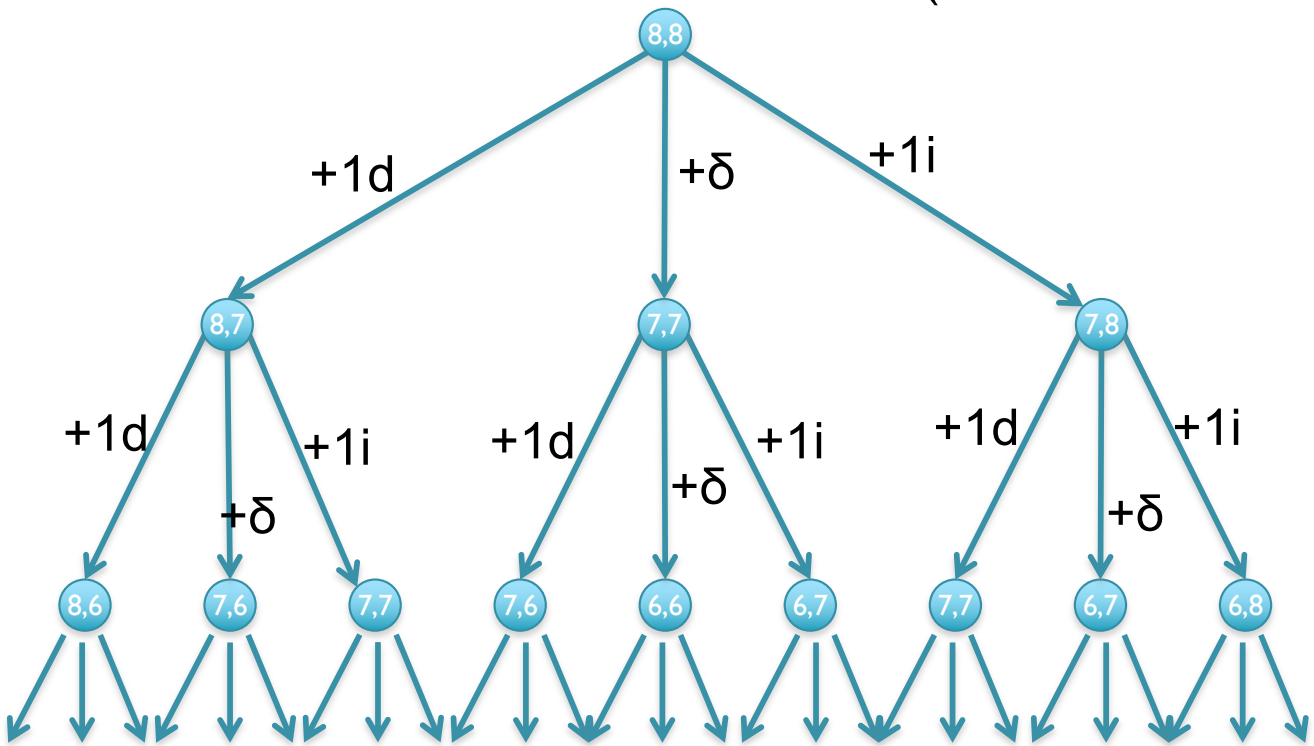
…**M**…	…**I**…	…**D**…
…X…	…-…	…X…
…Y…	…Y…	…-…

Eventually spell out every possible sequence of {I,M,D}

Recursive solution

- Computation of D is a recursive process.
  - At each step, we only allow matches, substitutions, and indels
  - D(i,j) in terms of D(i', j') for i' ≤ i and j' ≤ j.

D(AGCACACA, ACACACTA) = min{D(AGCACACA, ACACACT) + I,
D(AGCACAC, ACACACTA) + I,
D(AGCACAC, ACACACT) +δ(A, A)}



[What is the running time?]

Dynamic Programming

- We could code this as a recursive function call...

  ...with an exponential number of function evaluations

- There are only (n+1) x (m+1) pairs i and j
  - We are evaluating D(i,j) multiple times

- Compute D(i,j) bottom up.
  - Start with smallest (i,j) = (1,1).
  - Store the intermediate results in a table.
    - Compute D(i,j) *after* D(i-1,j), D(i,j-1), and D(i-1,j-1)

Recurrence Relation for D

Find the edit distance (minimum number of operations to convert one string into another) in O(mn) time

•Base conditions:

- D(i,0) = i, for all i = 0,...,n
- D(0,j) = j, for all j = 0,...,m

•For i > 0, j > 0:

$$
\begin{aligned}
D(i,j) = \min \{ & \\
& D(i-1,j) + 1, && \text{// align 0 chars from S, 1 from T} \\
& D(i,j-1) + 1, && \text{// align 1 chars from S, 0 from T} \\
& D(i-1,j-1) + \delta(S(i),T(j)) && \text{// align 1+1 chars} \\
\} &
\end{aligned}
$$

[Why do we want the min?]

Dynamic Programming Matrix

		A	C	A	C	A	C	T	A
	0	1	2	3	4	5	6	7	8
A	1								
G	2								
C	3								
A	4								
C	5								
A	6								
C	7								
A	8								

[What does the initialization mean?]

Dynamic Programming Matrix

		A	C	A	C	A	C	T	A
	0	1	2	3	4	5	6	7	8
A	1	0							
G	2								
C	3								
A	4								
C	5								
A	6								
C	7								
A	8								

$$D[A,A] = \min\{D[A,]+1,\ D[,A]+1,\ D[,]+\delta(A,A)\}$$

Dynamic Programming Matrix

		A	C	A	C	A	C	T	A
	0	1	2	3	4	5	6	7	8
A	1	0	1						
G	2								
C	3								
A	4								
C	5								
A	6								
C	7								
A	8								

$$D[A,AC] = \min\{D[A,A]+1,\ D[,AC]+1,\ D[,A]+\delta(A,C)\}$$

Dynamic Programming Matrix

		A	C	A	C	A	C	T	A
	0	1	2	3	4	5	6	7	8
A	1	0	1	2					
G	2								
C	3								
A	4								
C	5								
A	6								
C	7								
A	8								

D[A,ACA] = min{D[A,AC]+1, D[,ACA]+1, D[,AC]+δ(A,A)}

Dynamic Programming Matrix

		A	C	A	C	A	C	T	A
	0	1	2	3	4	5	6	7	8
A	1	0	1	2	3	4	5	6	7
G	2								
C	3								
A	4								
C	5								
A	6								
C	7								
A	8								

D[A,ACACACTA] = 7

```
-------A
*******|
ACACACTA
```

[What about the other A?]

Dynamic Programming Matrix

		A	C	A	C	A	C	T	A
	0	1	2	3	4	5	6	7	8
A	1	0	1	2	3	4	5	6	7
G	2	1	1	2	3	4	5	6	7
C	3								
A	4								
C	5								
A	6								
C	7								
A	8								

D[AG,ACACACTA] = 7

----AG--

****|***

ACACACTA

Dynamic Programming Matrix

		A	C	A	C	A	C	T	A
	0	1	2	3	4	5	6	7	8
A	1	0	1	2	3	4	5	6	7
G	2	1	1	2	3	4	5	6	7
C	3	2	1	2	2	3	4	5	6
A	4	3	2	1	2	2	3	4	5
C	5	4	3	2	1	2	2	3	4
A	6	5	4	3	2	1	2	3	3
C	7	6	5	4	3	2	1	2	3
A	8	7	6	5	4	3	2	2	2

D[AGCACACA,ACACACTA] = 2

```
AGCACAC-A
|*|||||*|
A-CACACTA
```

[Can we do it any better?]

Alignment Ambiguity

Which of these is the correct alignment?

```
GATTTACA        GATTTACA        GATTTACA
GATT-ACA        GAT-TACA        GA-TTACA
```

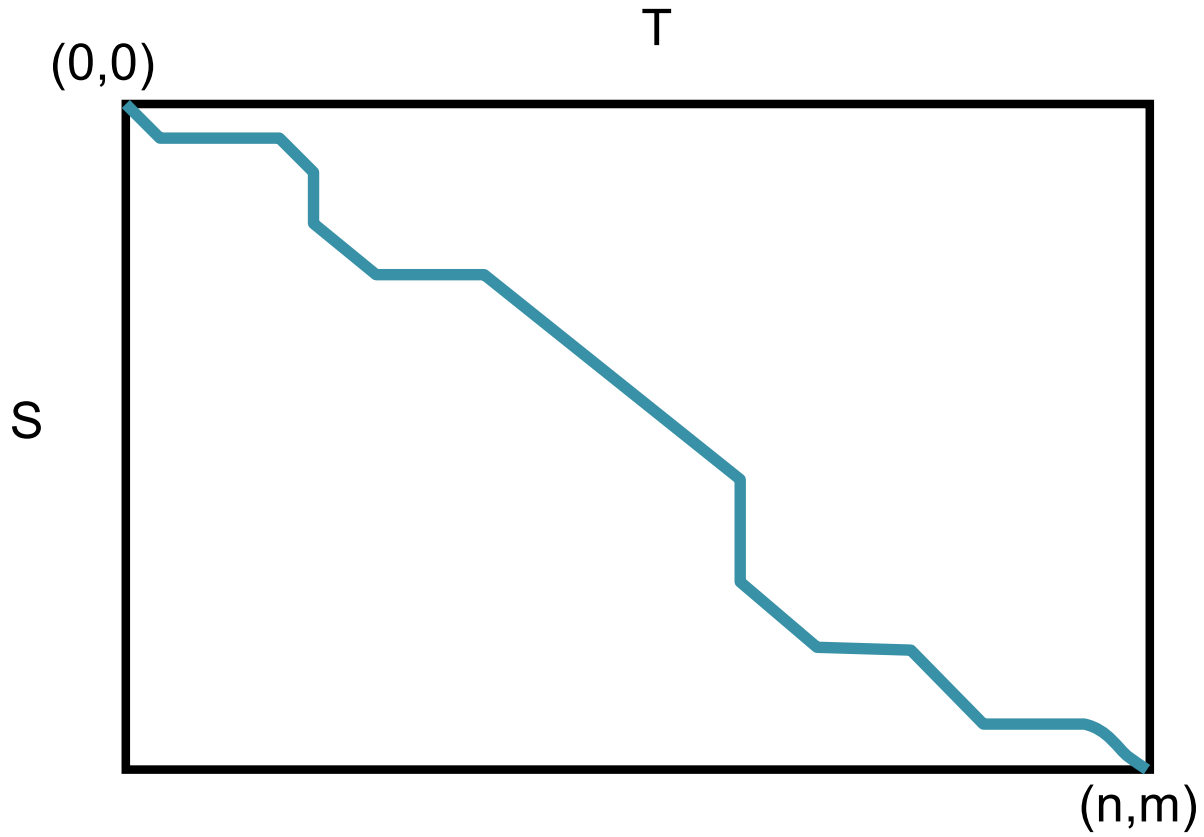
Which of these is the correct alignment?

```
GATATATACA      GATATATACA      GATATATACA
GATAT--ACA      GAT--ATACA      GAT-TA-ACA
```

They are all equally good!

- Generally prefer to merge gaps together when possible (affine gaps)
- Use variant “normalization” to shift gaps into a canonical position
  - Commonly shift gaps as far to the left as possible

Global Alignment Schematic



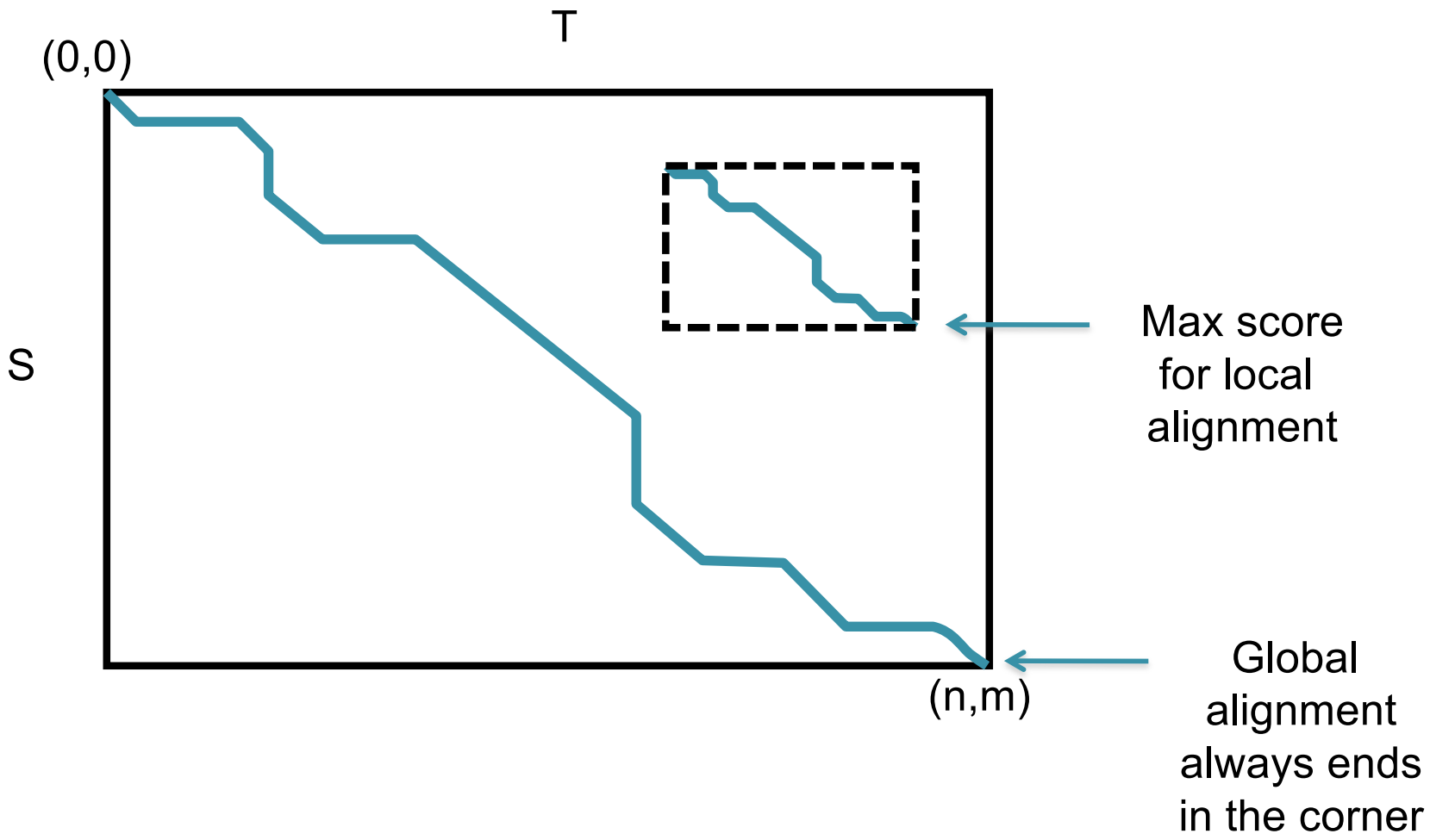
- A high quality alignment will stay close to the diagonal
  - If we are only interested in high quality alignments, we can skip filling in cells that can't possibly lead to a high quality alignment
  - Find the global alignment with at most edit distance d: O(2dn)

Local vs. Global Alignment

- The Global Alignment Problem tries to find the best end-to-end alignment between the two strings
  - Only applicable for very closely related sequences

- The Local Alignment Problem tries to find pairs of **substrings** with highest similarity.
  - Especially important if one string is substantially longer than the other
  - Especially important if there is only a distant evolutionary relationship

Global vs Local Alignment Schematic



Local vs. Global Alignment (cont' d)

- Global Alignment

```
--T--CC-C-AGT--TATGT-CAGGGGACACG-A-GCATGCAGA-GAC
  |  || |  ||  | | | |||    || | | |  | ||||   |
AATTGCCGCC-GTCGT-T-TTCAG----CA-GTTATG-T-CAGAT--C
```

- Local Alignment—better alignment to find conserved segment

```
                tccCAGTTATGTCAGgggacacgagcatgcagagac
                   ||||||||||||
aattgccgccgtcgttttcagCAGTTATGTCAGatc
```

SAM / BAM Files

1.4 The alignment section: mandatory fields

In the SAM format, each alignment line typically represents the linear alignment of a segment. Each line consists of 11 or more TAB-separated fields. The first eleven fields are always present and in the order shown below; if the information represented by any of these fields is unavailable, that field's value will be a placeholder, either '0' or '*' as determined by the field's type. The following table gives an overview of these mandatory fields in the SAM format:

Col	Field	Type	Regexp/Range	Brief description
1	QNAME	String	`[!-?A-~]{1,254}`	Query template NAME
2	FLAG	Int	$[0, 2^{16} - 1]$	bitwise FLAG
3	RNAME	String	`\*|[:rname:^*=][:rname:]*`	Reference sequence NAME[11]
4	POS	Int	$[0, 2^{31} - 1]$	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	`\*|=|[:rname:^*=][:rname:]*`	Reference name of the mate/next read
8	PNEXT	Int	$[0, 2^{31} - 1]$	Position of the mate/next read
9	TLEN	Int	$[-2^{31} + 1, 2^{31} - 1]$	observed Template LENgth
10	SEQ	String	`\*|[A-Za-z=.]+`	segment SEQuence
11	QUAL	String	`[!-~]+`	ASCII of Phred-scaled base QUALity+33

[11] Reference sequence names may contain any printable ASCII characters with the exception of certain punctuation characters, and may not start with '*' or '='. See Section 1.2.1 for details and an explanation of the `[:rname:]` notation.

https://samtools.github.io/hts-specs/SAMv1.pdf

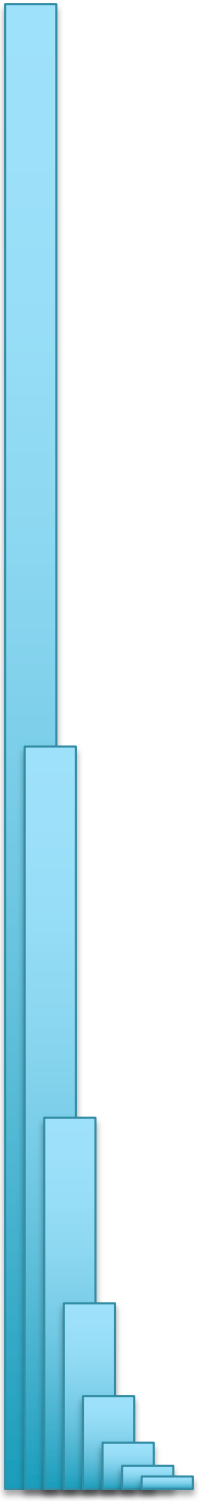
SAM / BAM Files

```
Coor      12345678901234  5678901234567890123456789012345
ref       AGCATGTTAGATAA**GATAGCTGTGCTAGTAGGCAGTCAGCGCCAT

+r001/1         TTAGATAAAGGATA*CTG
+r002         aaaAGATAA*GGATA
+r003       gcctaAGCTAA
+r004                     ATAGCT..............TCAGC
-r003                           ttagctTAGGC
-r001/2                                        CAGCGGCAT
```

```
@HD VN:1.6 SO:coordinate
@SQ SN:ref LN:45
r001   99 ref  7 30 8M2I4M1D3M = 37  39 TTAGATAAAGGATACTG *
r002    0 ref  9 30 3S6M1P1I4M *  0   0 AAAAGATAAGGATA    *
r003    0 ref  9 30 5S6M       *  0   0 GCCTAAGCTAA       * SA:Z:ref,29,-,6H5M,17,0;
r004    0 ref 16 30 6M14N5M    *  0   0 ATAGCTTCAGC       *
r003 2064 ref 29 17 6H5M       *  0   0 TAGGC             * SA:Z:ref,9,+,5S6M,30,1;
r001  147 ref 37 30 9M         =  7 -39 CAGCGGCAT         * NM:i:1
```

https://samtools.github.io/hts-specs/SAMv1.pdf

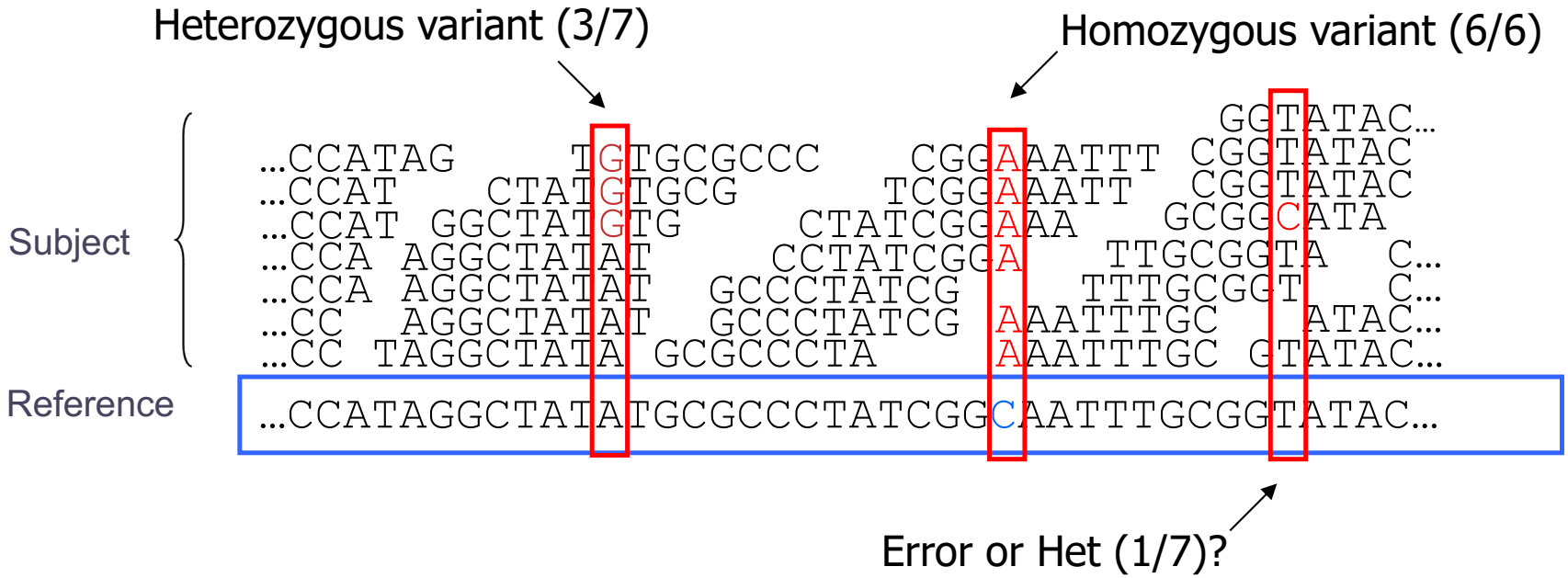


Part 2: Variant Calling

Variant Calling Overview

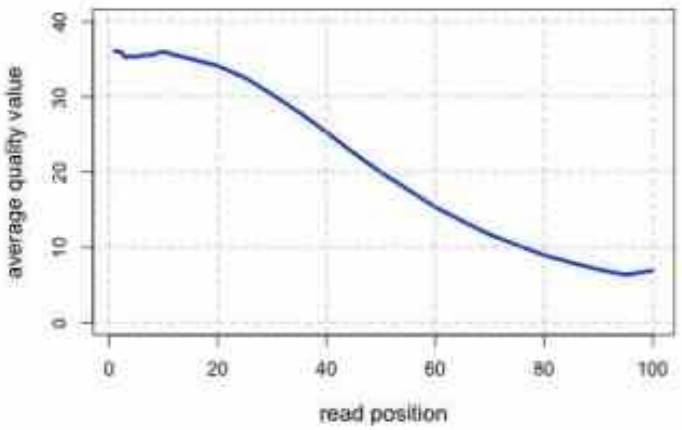
→ FASTQ → Align (BWA) → BAM → Detect SNP/INDELs (GATK or FreeBayes) → VCF

Genotyping Theory



- If there were no sequencing errors, identifying SNPs would be very easy: any time a read disagrees with the reference, it must be a variant!

- Sequencing instruments make mistakes
  - Quality of read decreases over the read length

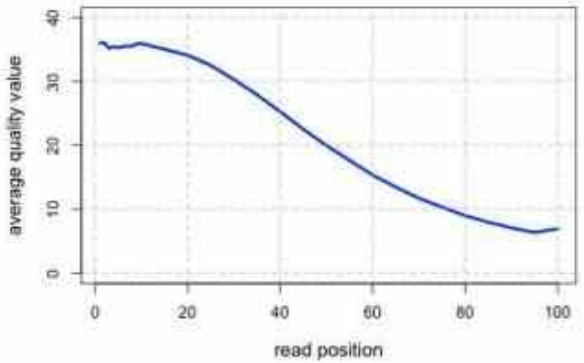


- A single read differing from the reference is probably just an error, but it becomes more likely to be real as we see it multiple times

Illumina Quality

QV	p_{error}
40	1/10000
30	1/1000
20	1/100
10	1/10

$$Q_{\text{sanger}} = -10 \log_{10} p$$




```
  SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS.....................................................
  ..........................XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX.....................
  ..............................IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII.........................
  .................................JJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJ.......................
  LLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLL.....................................................
  !"#$%&'()*+,-./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)
```

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

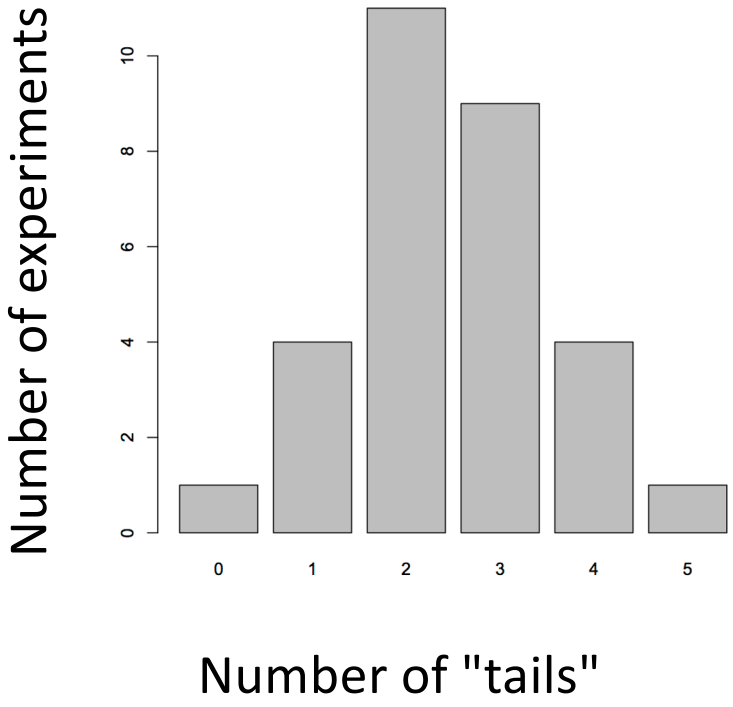
The Binomial Distribution: Adventures in Coin Flipping

P(heads) = 0.5

P(tails) = 0.5

Aaron Quinlan

What is the distribution of tails (alternate alleles) do we expect to see after 5 tosses (sequence reads)?



R code:

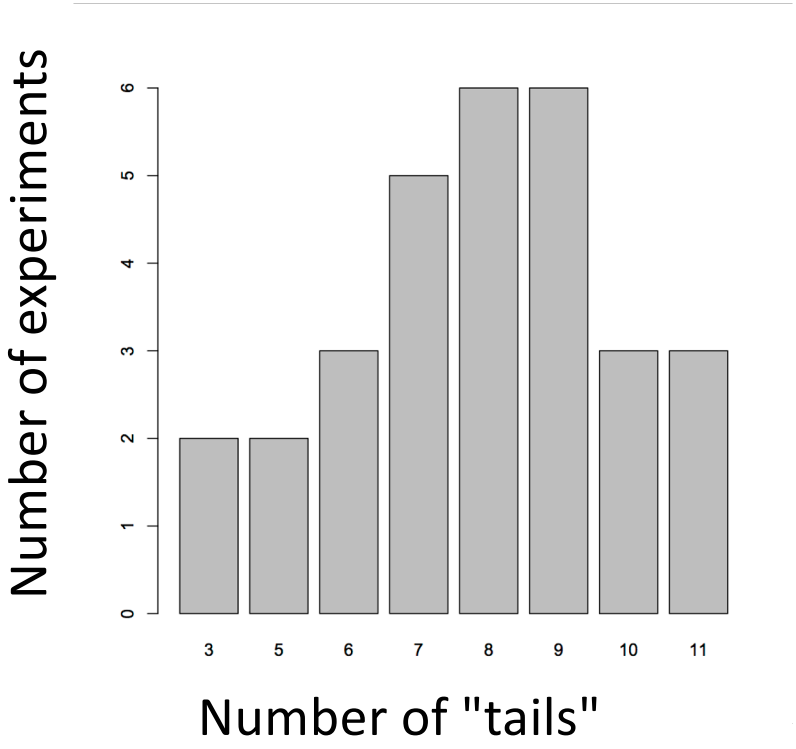
```
barplot(table(rbinom(30, 5, 0.5)))
```

30 experiments (students tossing coins)

5 tosses each

Probability of Tails

What is the distribution of tails (alternate alleles) do we expect to see after 15 tosses (sequence reads)?



R code:

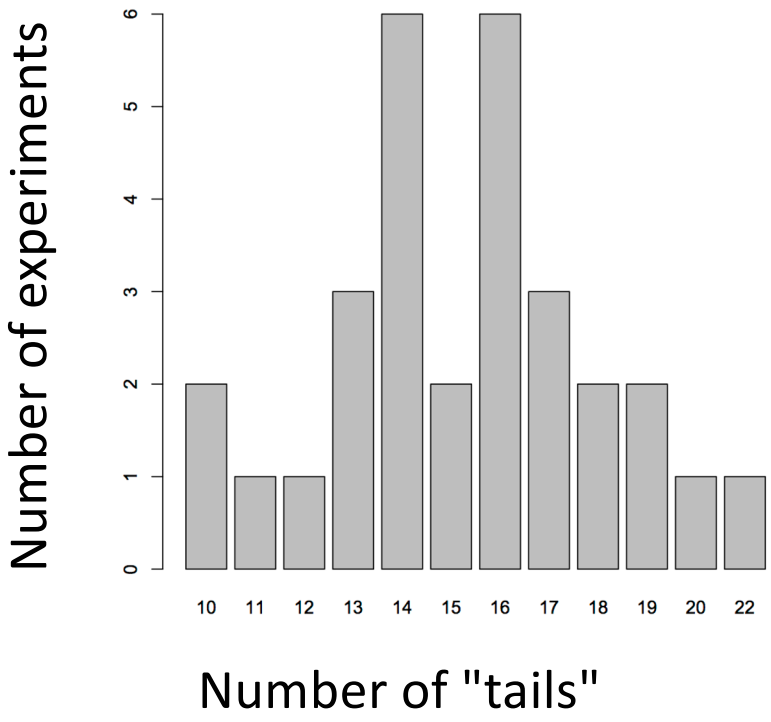
```
barplot(table(rbinom(30, 15, 0.5)))
```

30 experiments (students tossing coins)

15 tosses each

Probability of Tails

What is the distribution of tails (alternate alleles) do we expect to see after 30 tosses (sequence reads)?



R code:

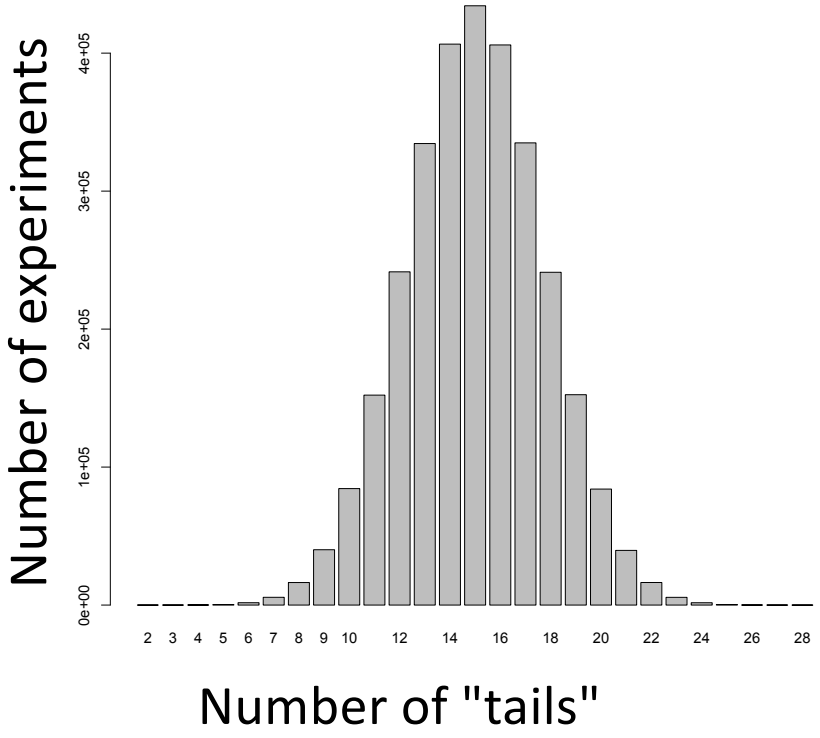
```
barplot(table(rbinom(30, 30, 0.5)))
```

30 experiments (students tossing coins)

30 tosses each

Probability of Tails

What is the distribution of tails (alternate alleles) do we expect to see after 30 tosses (sequence reads)?



R code:

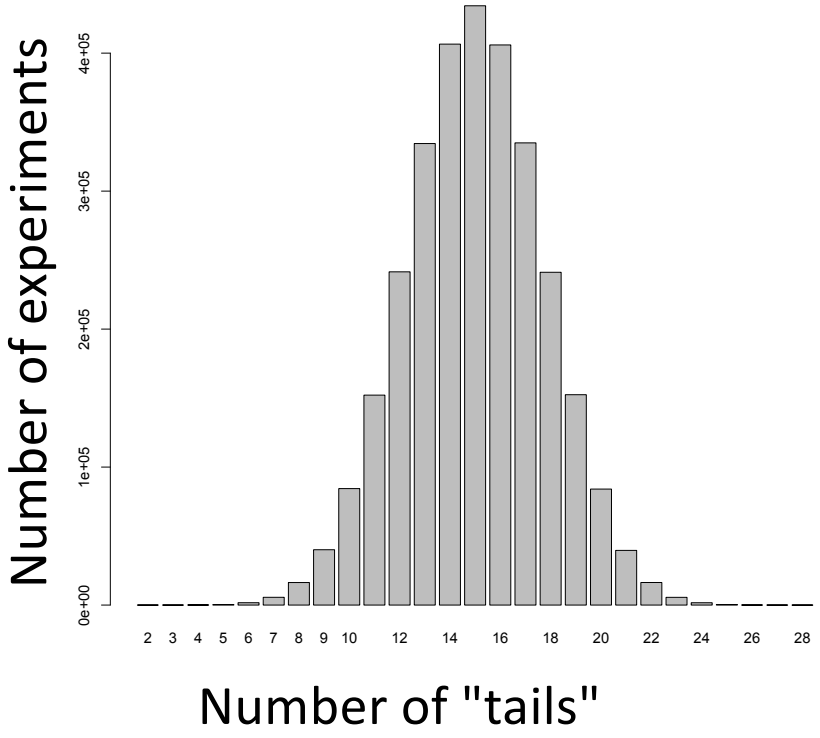
```
barplot(table(rbinom(3e6, 30, 0.5)))
```

3M experiments (students tossing coins)

30 tosses each

Probability of Tails

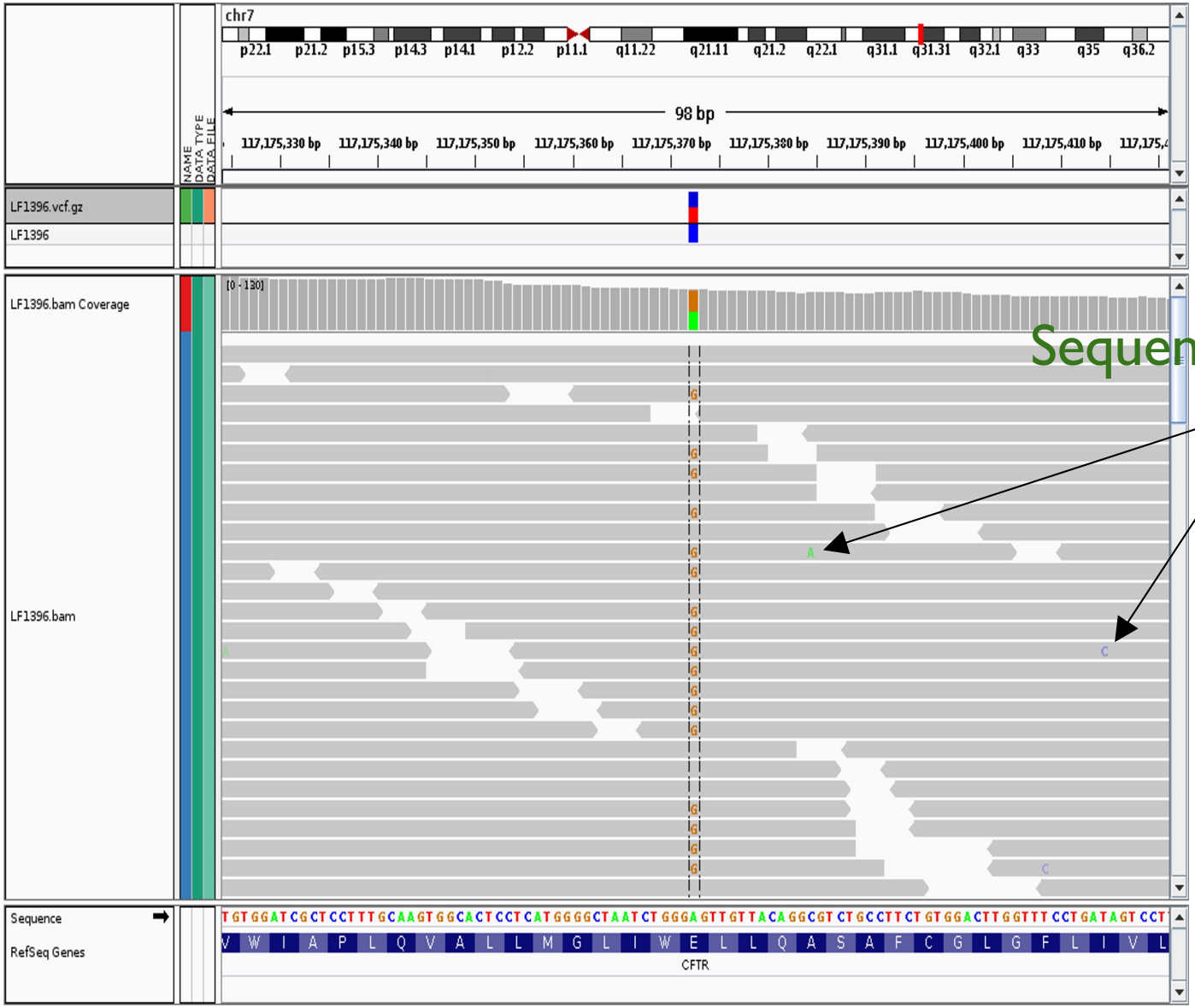
So, with 30 tosses (reads), we are much more likely to see an even mix of alternate and reference alleles at a heterozygous locus in a genome



This is why at least a "30X" (30 fold sequence coverage) genome is recommended: it confers sufficient power to distinguish heterozygous alleles and from mere sequencing errors

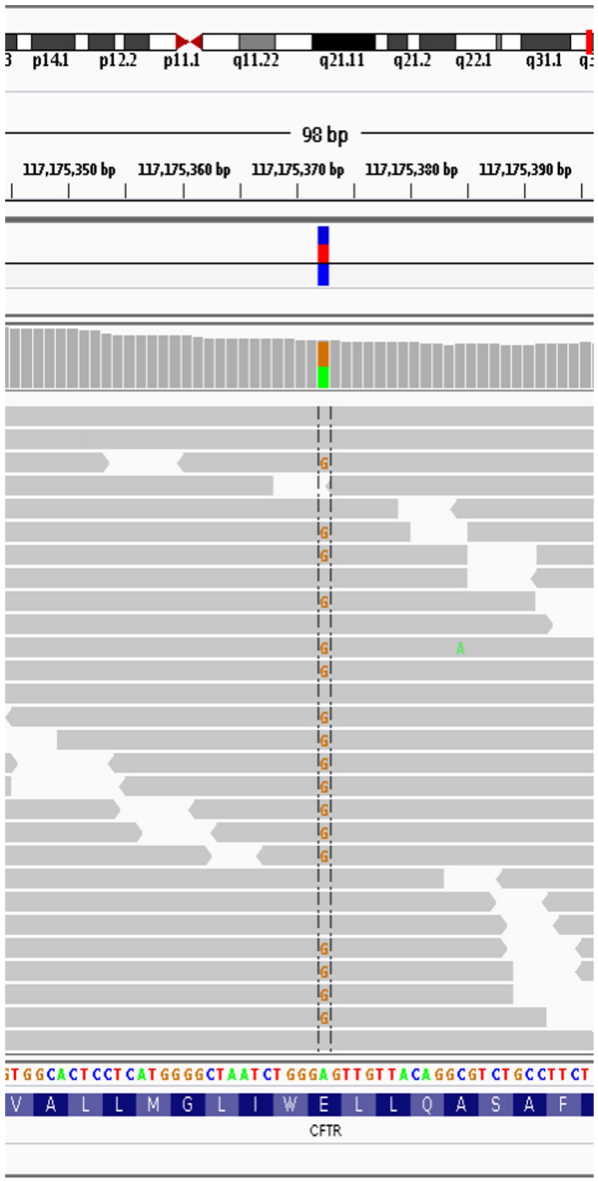
P(3/30 het) <?> P(3/30 err)

Sequencing errors fall out as noise (most of the time)



IGV: https://igv.org/

What information is needed to decide if a variant exists?



- Depth of coverage at the locus
- Bases observed at the locus
- The base qualities of each allele
- The strand composition
- Mapping qualities
- Proper pairs?
- Expected polymorphism rate

https://genomebiology.biomedcentral.com/articles/10.1186/gb-2010-11-10-r99

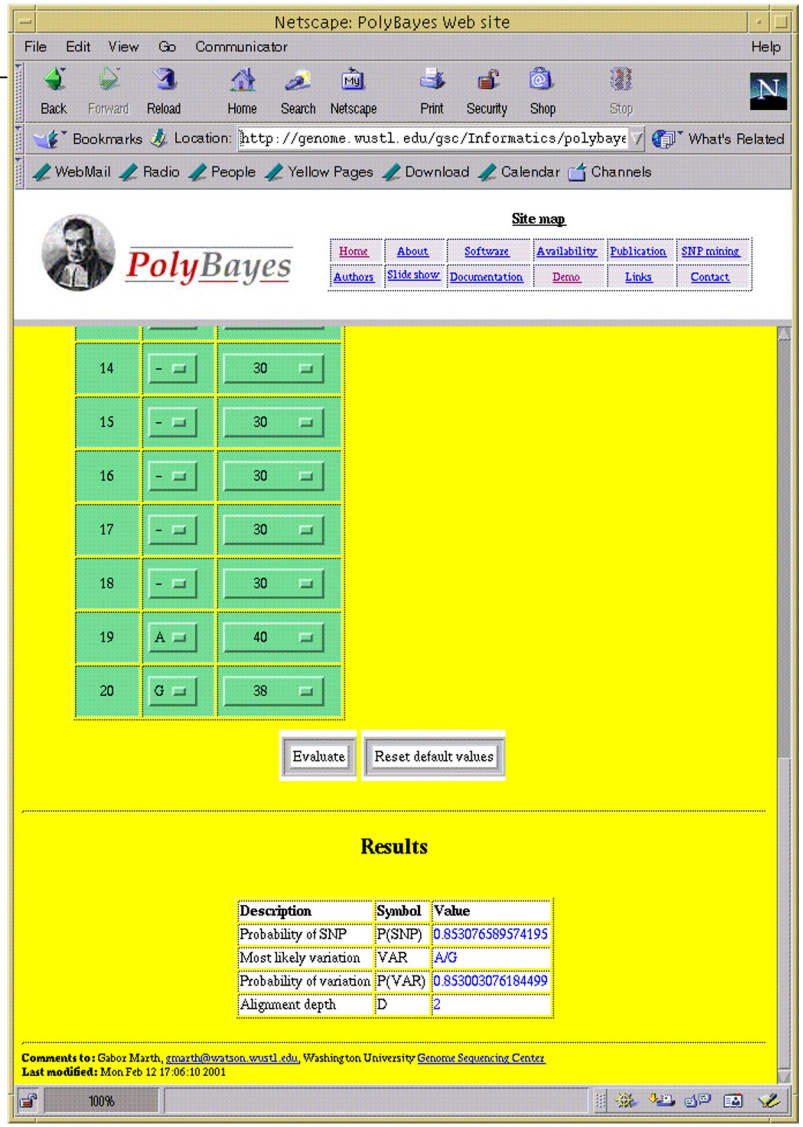
PolyBayes: The first statistically rigorous variant detection tool.

letter

A general approach to single-nucleotide polymorphism discovery

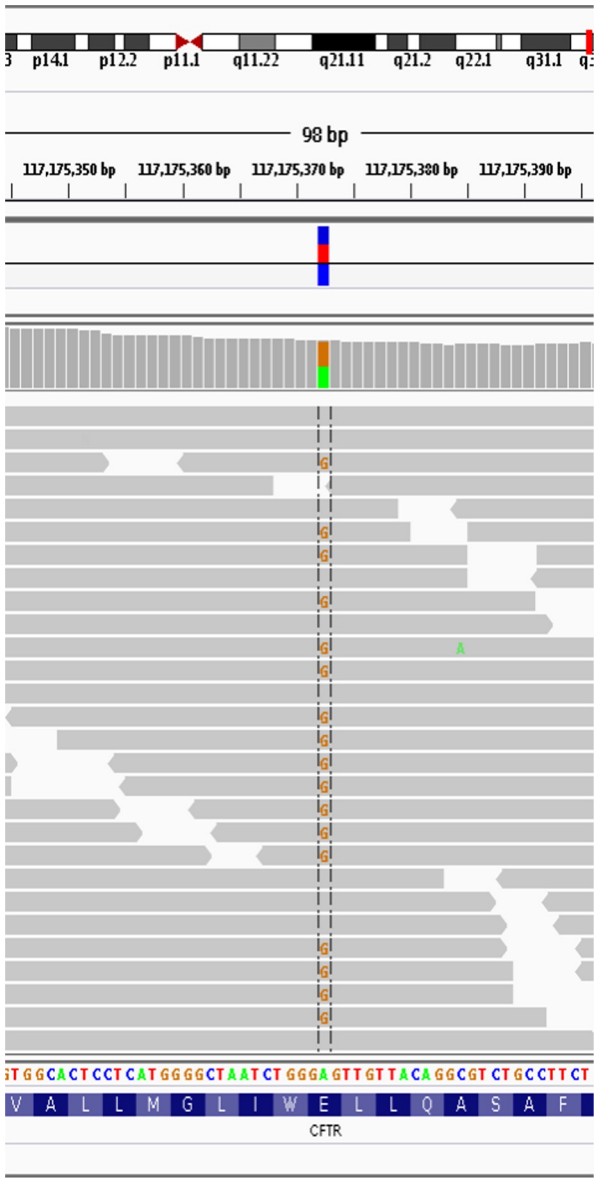
Gabor T. Marth[1], Ian Korf[1], Mark D. Yandell[1], Raymond T. Yeh[1], Zhijie Gu[2], Hamideh Zakeri[2], Nathan O. Stitziel[1], LaDeana Hillier[1], Pui-Yan Kwok[2] & Warren R. Gish[1]

Its main innovation was the use of Bayes's theorem



http://www.nature.com/ng/journal/v23/n4/full/ng1299_452.html

Bayesian SNP calling



$$P(SNP|Data) = \frac{P(Data|SNP) * P(SNP)}{P(Data)}$$

http://www.nature.com/ng/journal/v23/n4/full/ng1299_452.html

PolyBayes: The first statistically rigorous variant detection tool.

letter

A general approach to single-nucleotide polymorphism discovery

Gabor T. Marth[1], Ian Korf[1], Mark D. Yandell[1], Raymond T. Yeh[1], Zhijie Gu[2], Hamideh Zakeri[2], Nathan O. Stitziel[1], LaDeana Hillier[1], Pui-Yan Kwok[2] & Warren R. Gish[1]

Bayesian posterior probability

Base call + Base quality

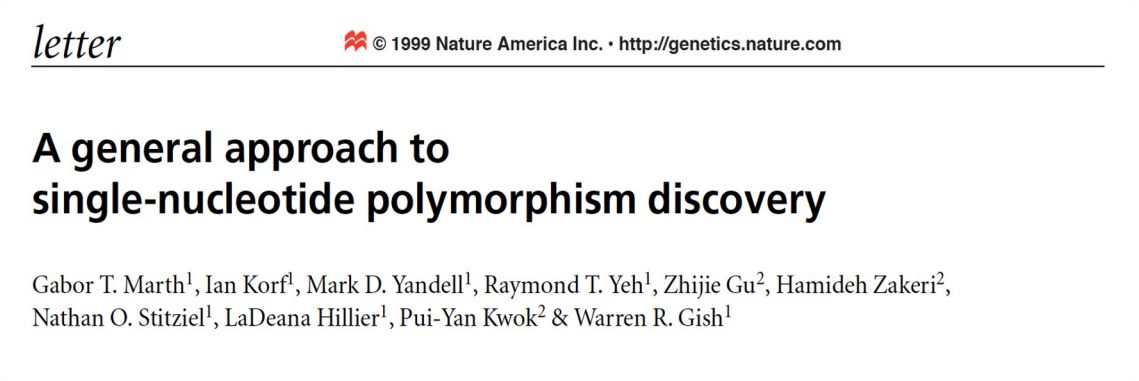
Expected (prior) polymorphism rate

$$P(SNP) = \sum_{all\ variable\ S} \frac{\frac{P(S_1 \mid R_1)}{P_{Prior}(S_1)} \cdot \ldots \cdot \frac{P(S_N \mid R_N)}{P_{Prior}(S_N)} \cdot P_{Prior}(S_1,\ldots,S_N)}{\sum_{S_{i_1} \in [A,C,G,T]} \ldots \sum_{S_{i_N} \in [A,C,G,T]} \frac{P(S_{i_1} \mid R_1)}{P_{Prior}(S_{i_1})} \cdot \ldots \cdot \frac{P(S_{i_N} \mid R_1)}{P_{Prior}(S_{i_N})} \cdot P_{Prior}(S_{i_1},\ldots,S_{i_N})}$$

Probability of observed base composition (should model sequencing error rate)

http://www.nature.com/ng/journal/v23/n4/full/ng1299_452.html

PolyBayes: The first statistically rigorous variant detection tool.



letter © 1999 Nature America Inc. • http://genetics.nature.com

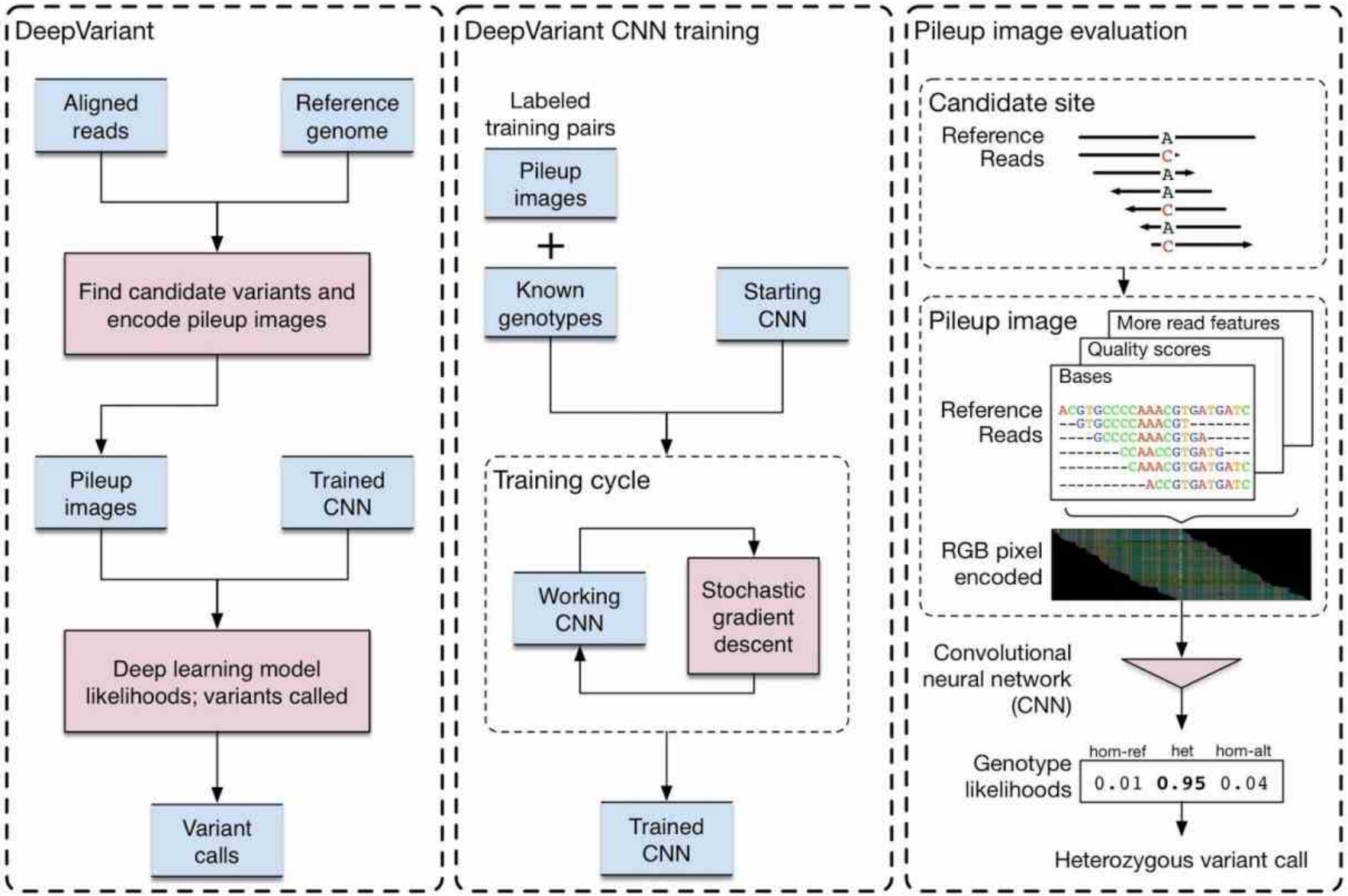
A general approach to single-nucleotide polymorphism discovery

Gabor T. Marth[1], Ian Korf[1], Mark D. Yandell[1], Raymond T. Yeh[1], Zhijie Gu[2], Hamideh Zakeri[2], Nathan O. Stitziel[1], LaDeana Hillier[1], Pui-Yan Kwok[2] & Warren R. Gish[1]

This Bayesian statistical framework has been adopted by most other modern SNP/INDEL callers such as FreeBayes, GATK, and samtools

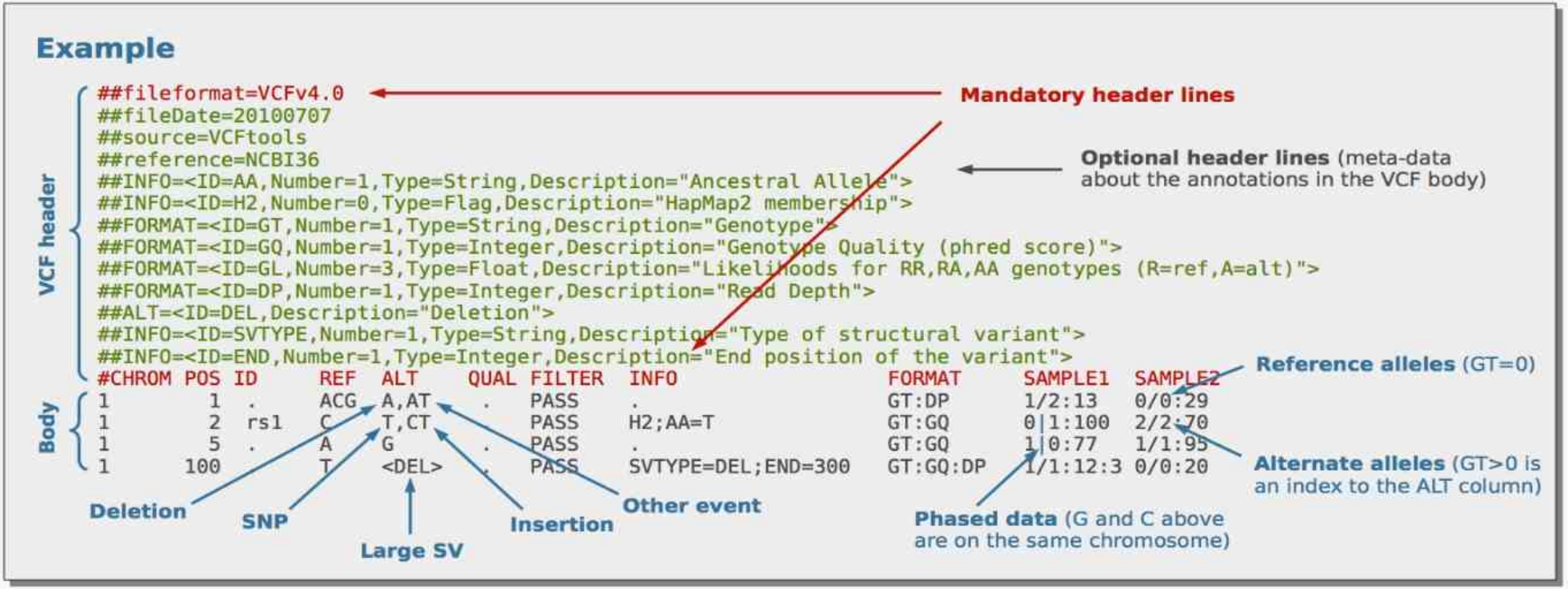
http://www.nature.com/ng/journal/v23/n4/full/ng1299_452.html

DeepVariant

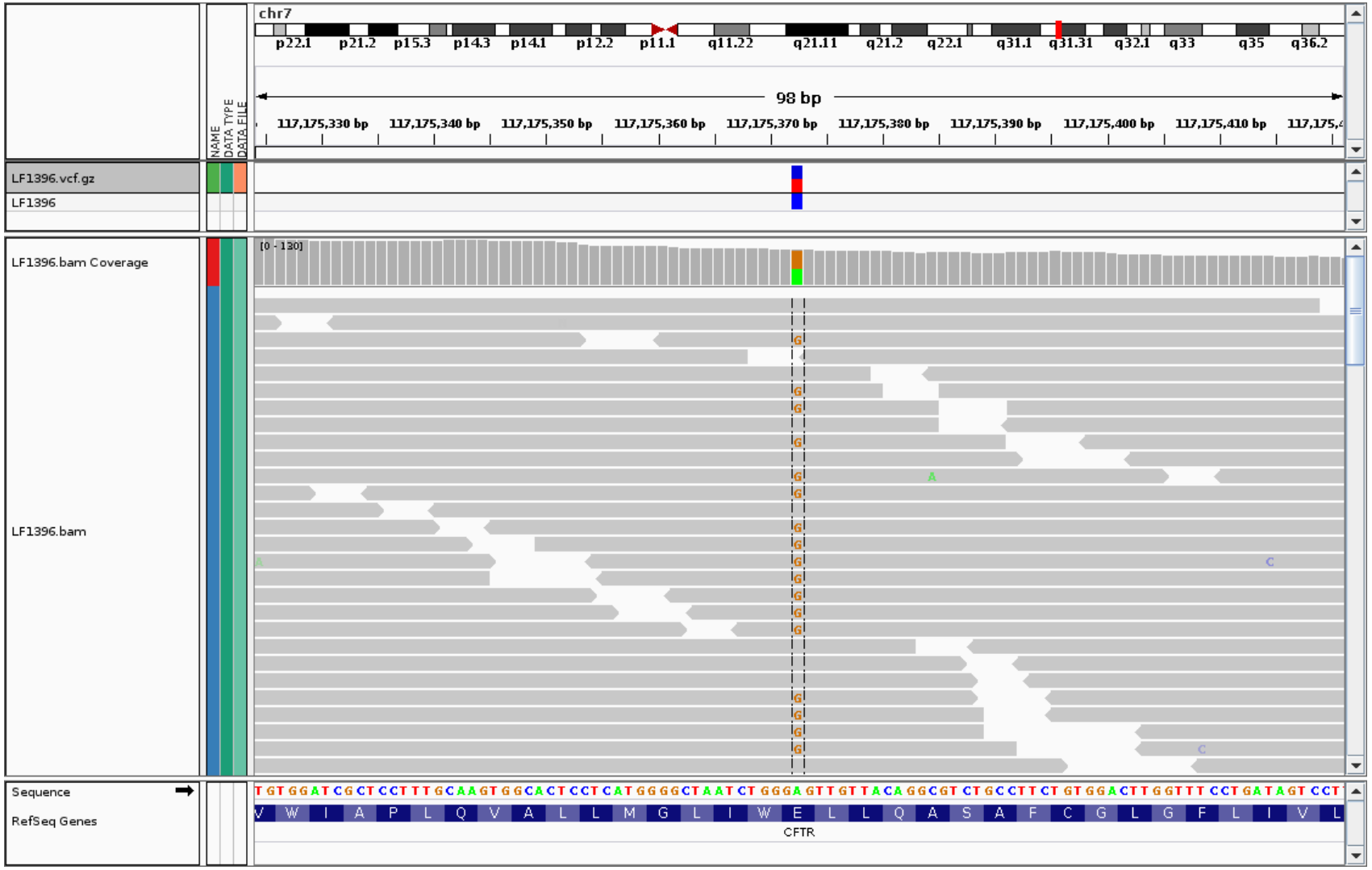


Creating a universal SNP and small indel variant caller with deep neural networks
Poplin et al. (2018) Nature Biotechnology. https://www.nature.com/articles/nbt.4235

VCF Format


Example
VCF header
Body
##fileformat=VCFv4.0
##fileDate=20100707
##source=VCFtools
##reference=NCBI36
##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 (phred score)">
##FORMAT=<ID=GL,Number=3,Type=Float,Description="Likelihoods for RR,RA,AA genotypes (R=ref,A=alt)">
##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">
#CHROM POS ID REF ALT QUAL FILTER INFO FORMAT SAMPLE1 SAMPLE2
1 1 . ACG A,AT . PASS . GT:DP 1/2:13 0/0:29
1 2 rs1 C T,CT PASS H2;AA=T GT:GQ 0|1:100 2/2:70
1 5 . A G . PASS . GT:GQ 1|0:77 1/1:95
1 100 T <DEL> . PASS SVTYPE=DEL;END=300 GT:GQ:DP 1/1:12:3 0/0:20
Mandatory header lines
Optional header lines (meta-data about the annotations in the VCF body)
Reference alleles (GT=0)
Alternate alleles (GT>0 is an index to the ALT column)
Deletion
SNP
Large SV
Insertion
Other event
Phased data (G and C above are on the same chromosome)

VCF Format




```
#CHROM POS       ID  REF ALT QUAL   FILTER INFO    FORMAT LF1396
chr7   117175373 .   A   G   90     PASS   AF=0.5  GT     0/1
```