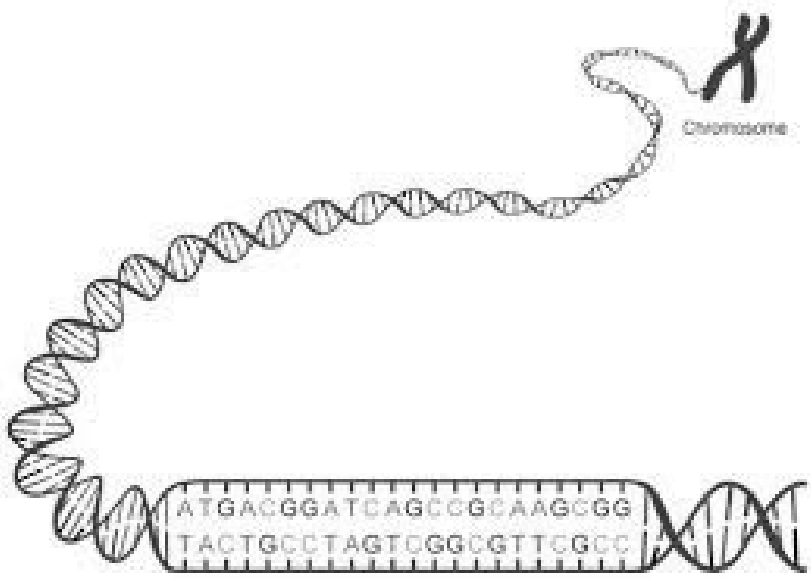
Genome assembly and analysis

Anna-Sophie Fiston-Lavier
Assistant Professor,
University of Montpellier2

HMIN305

Institut des Sciences de l'Evolution-Montpellier

DNA sequence



DNA Sequence

Sequencing or how to read a sequence

DNA Sequence



Set of strings based on
4 letters as DNA alphabet {A,C,G,T}

Sequencing or how to read a sequence

DNA Sequence

Set of strings based on
4 letters as DNA alphabet {A,C,G,T}
Source of information





Sequencing or how to read a sequence

DNA Sequence



Set of strings based on
4 letters as DNA alphabet {A,C,G,T}



Short reads



Long reads

Sequencing or how to read a sequence

DNA Sequence



Set of strings based on
4 letters as DNA alphabet {A,C,G,T}



Short reads

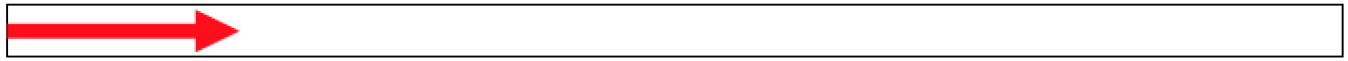


Long reads

Different read types

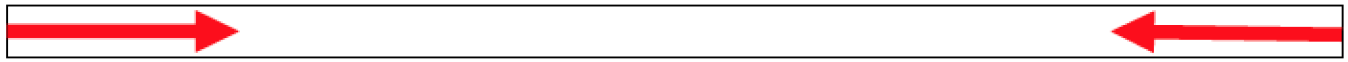
Single end read

– Only have sequence from one end of fragment

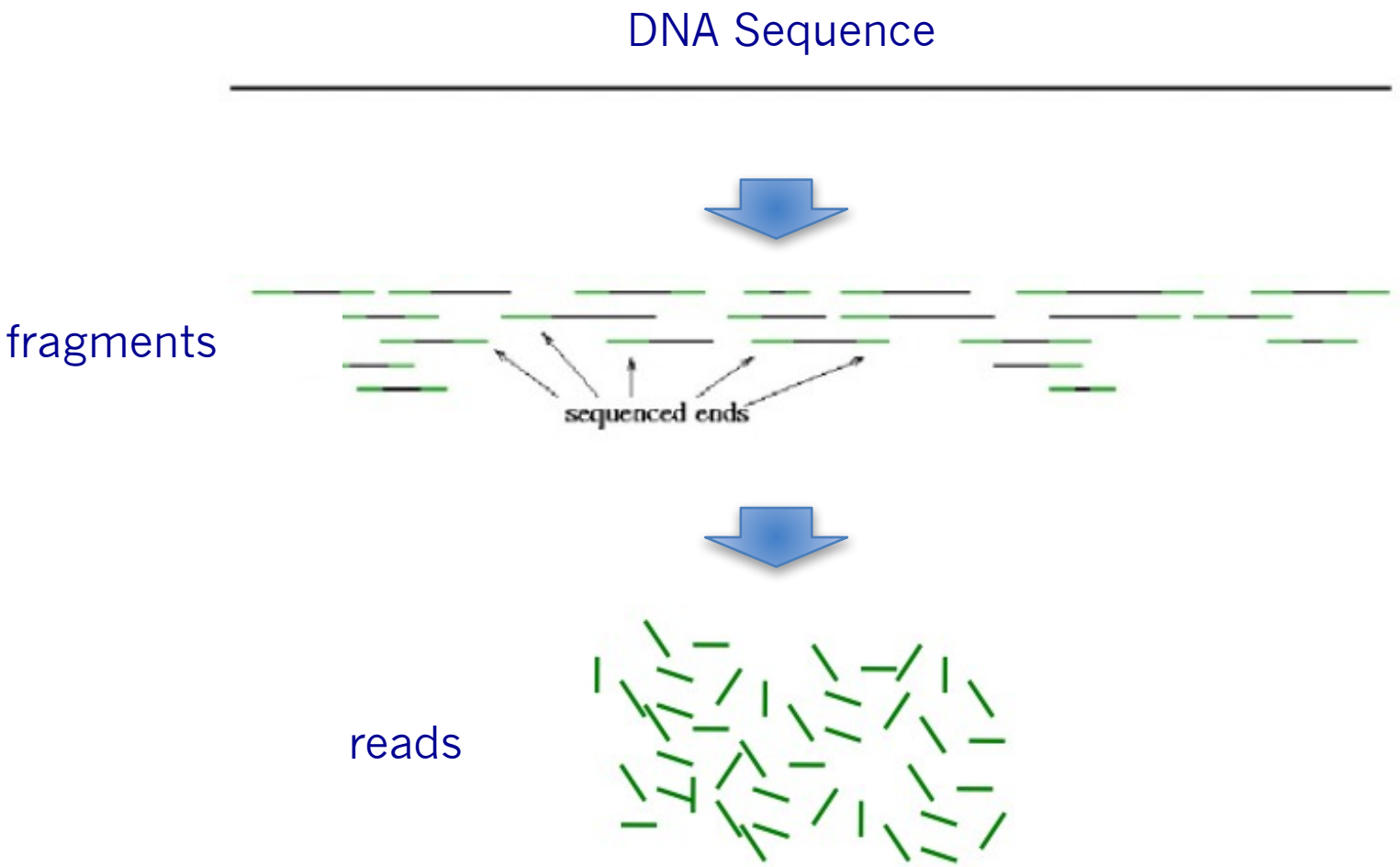


Paired / Mated read

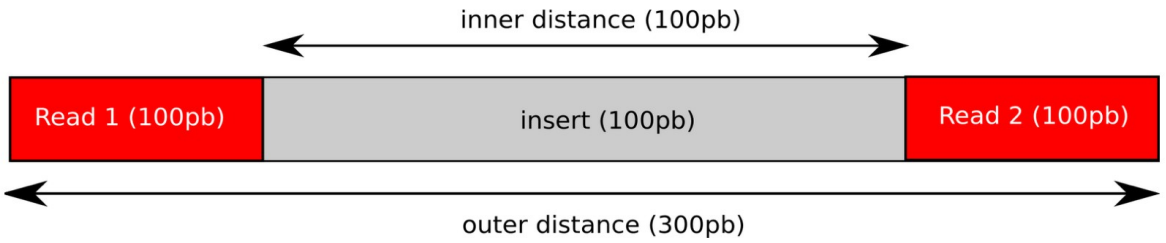
– Have sequence from both ends of fragment



Paired-end sequencing



How to define a pair of reads?



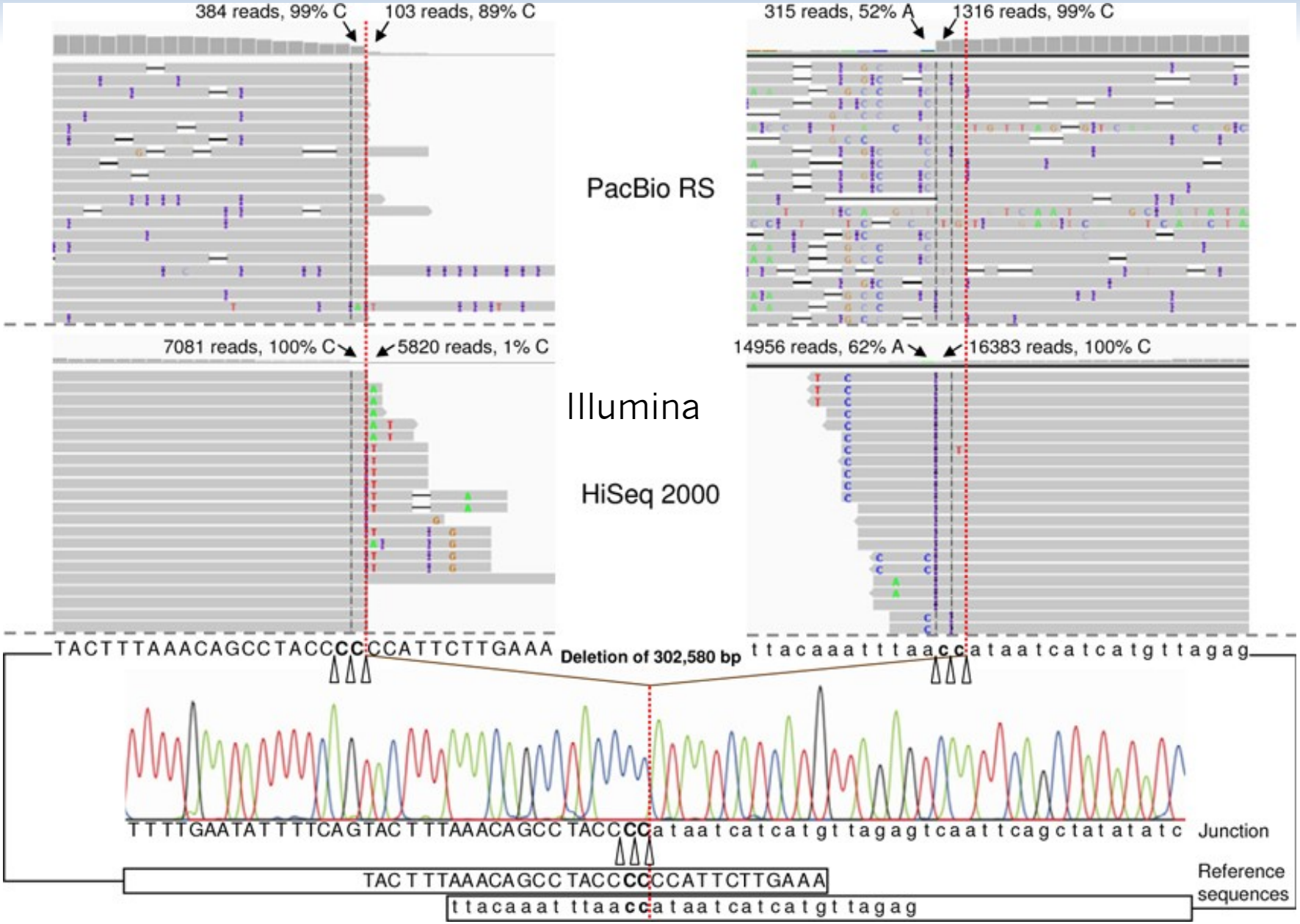
= Insert size

Different sequencing technologies

Method	bp	Attributes
Sanger	1200	low yield, low quality at both ends
Roche 454	700	Homo-polymer errors
Illumina	150	Low quality at 3' end
SOLiD	75	Difficult to work in colour-space
IonTorrent	100	High error rate, homo-polymers
PacBio	3000	Very high error rate, can iterate

Different sequencing technologies

Method	bp	Attributes
Sanger	1200	low yield, low quality at both ends
Roche 454	700	Homo-polymer errors
Illumina	150	Low quality at 3' end
SOLiD	75	Difficult to work in colour-space
IonTorrent	100	High error rate, homo-polymers
PacBio	3000	Very high error rate, can iterate


384 reads, 99% C
103 reads, 89% C
315 reads, 52% A
1316 reads, 99% C
PacBio RS
7081 reads, 100% C
5820 reads, 1% C
14956 reads, 62% A
16383 reads, 100% C
Illumina
HiSeq 2000
TACTTTAAACAGCCTACCCCCATTCTTGAAA
Deletion of 302,580 bp
ttacaaatttaaccataatcatcatgttagag
TTTTGAATATTTTCAGTACTTTAAACAGCCTACCCCataatcatcatgttagagtcaattcagctatatatc
Junction
TACTTTAAACAGCCTACCCCCATTCTTGAAA
ttacaaatttaaccataatcatcatgttagag
Reference sequences

Different sequencing technologies

Method	bp	Attributes
Sanger	1200	low yield, low quality at both ends
Roche 454	700	Homo-polymer errors
Illumina	150	Low quality at 3' end
SOLiD	75	Difficult to work in colour-space
IonTorrent	100	High error rate, homo-polymers
PacBio	3000	Very high error rate, can iterate

An assembly

What you get from the sequencing

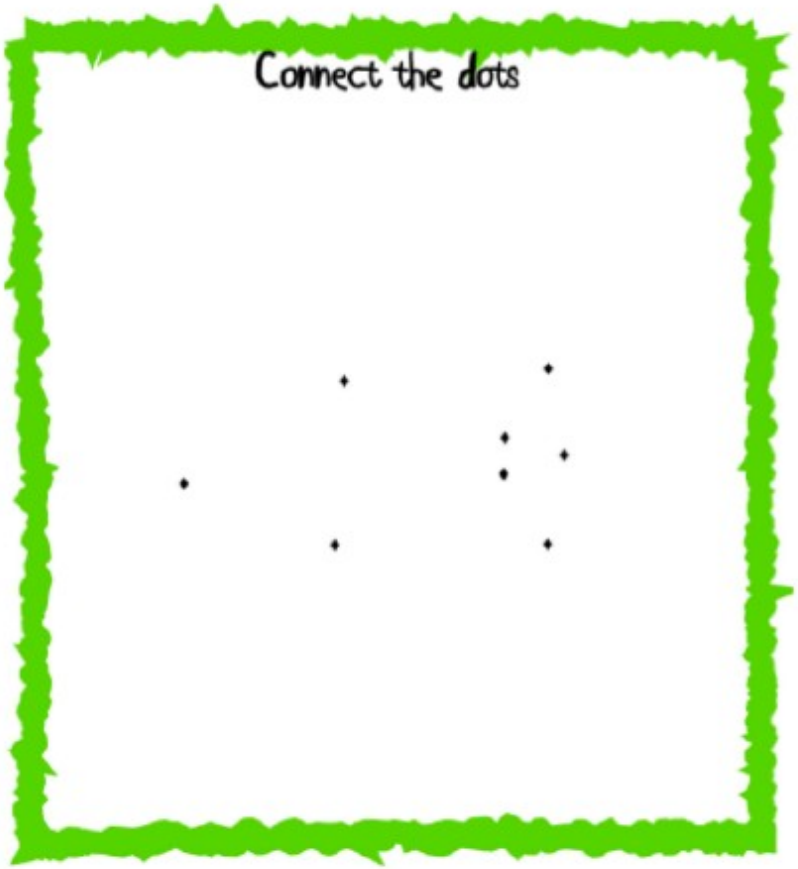
reads



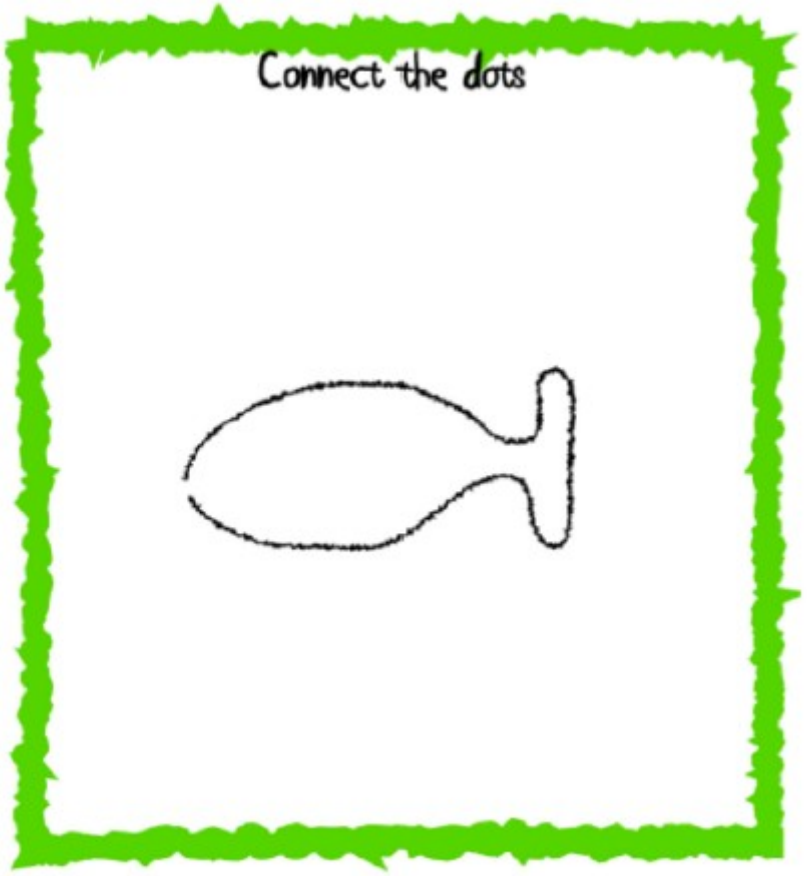
What you want



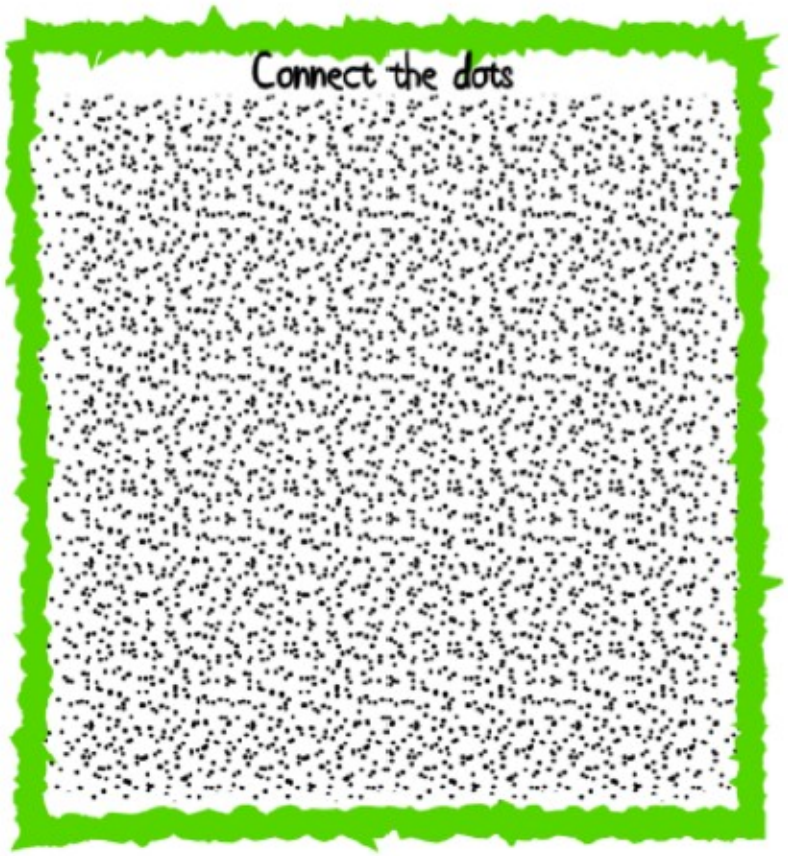
Assembly = Solving Puzzles Without a Picture



Assembly = Solving Puzzles Without a Picture



Assembly = Solving Puzzles Without a Picture



Assembly = Solving Puzzles Without a Picture



Assembly = Solving Puzzles Without a Picture



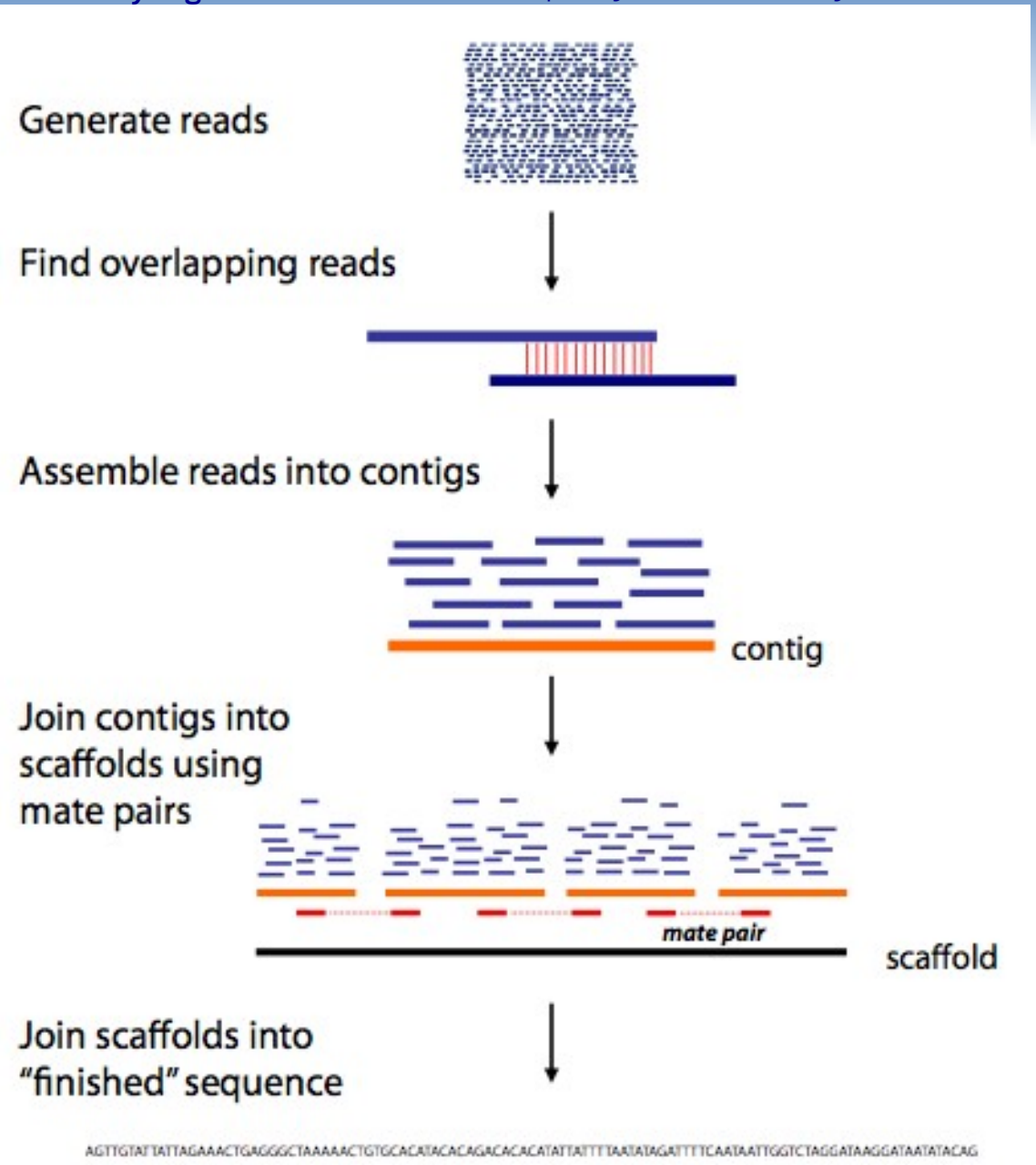
The assembly process is complicated by the fact that underlying assumption is incorrect.

Assembly process

Contig = a set of reads

Unitig = a contig formed from overlapping unambiguously unique sequences (*i.e.*, a high-confidence contig)

Scaffold = an ordered and oriented set of one or more contigs with distances assigned to the gaps between contigs



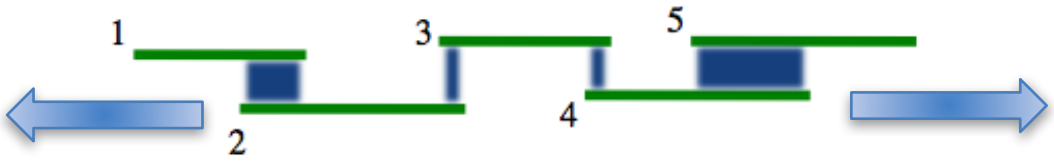
Assembly paradigms

Assemblers are based on one of several different paradigms:

1. Greedy
2. Overlap-Layout-Consensus (OLC)
3. De Bruijn Graph

1/Greedy

- While there are sequences with overlap:
  - Find sequences with largest overlap
  - Merge those sequences



The choices made by the assembler are inherently local and do not take into account the global relationship between the reads.

1/Greedy

- **Advantage**:
  - Simple
- **Disadvantage**:
  - Early mistakes create bad assemblies

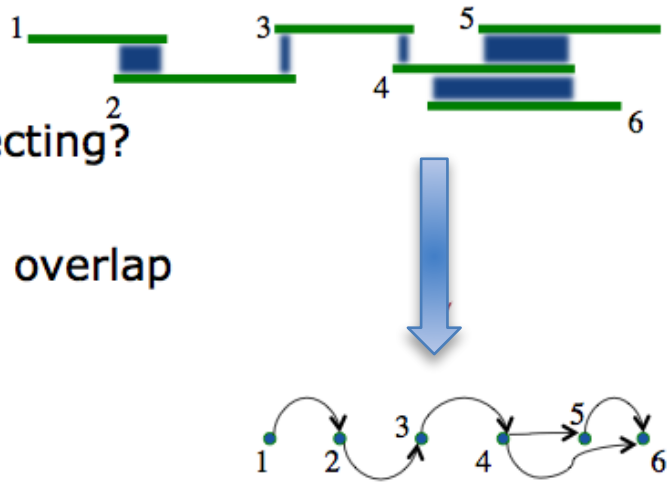
2/OLC: Overlap-Layout-Consensus

2/OLC: Overlap-Layout-Consensus

- **Clean your input**
  Remove "vector sequence", low quality, etc

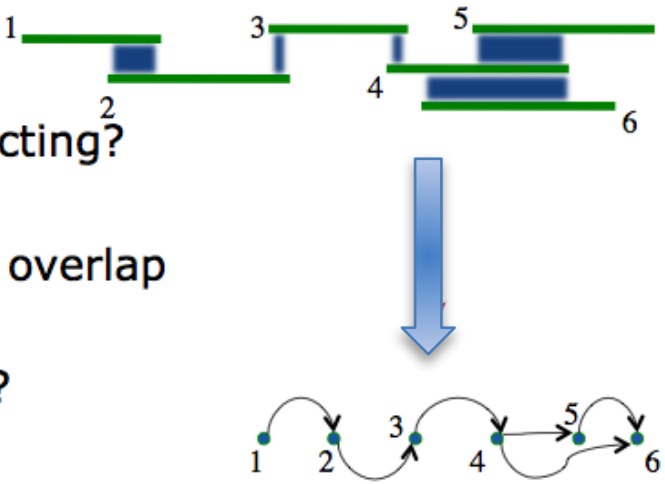
2/OLC: Overlap-Layout-Consensus

- **Clean your input**
  Remove "vector sequence", low quality, etc
- **Overlap**: What reads are intersecting?
  - Create a node for each read
  - Create directed edge for each overlap



2/OLC: Overlap-Layout-Consensus

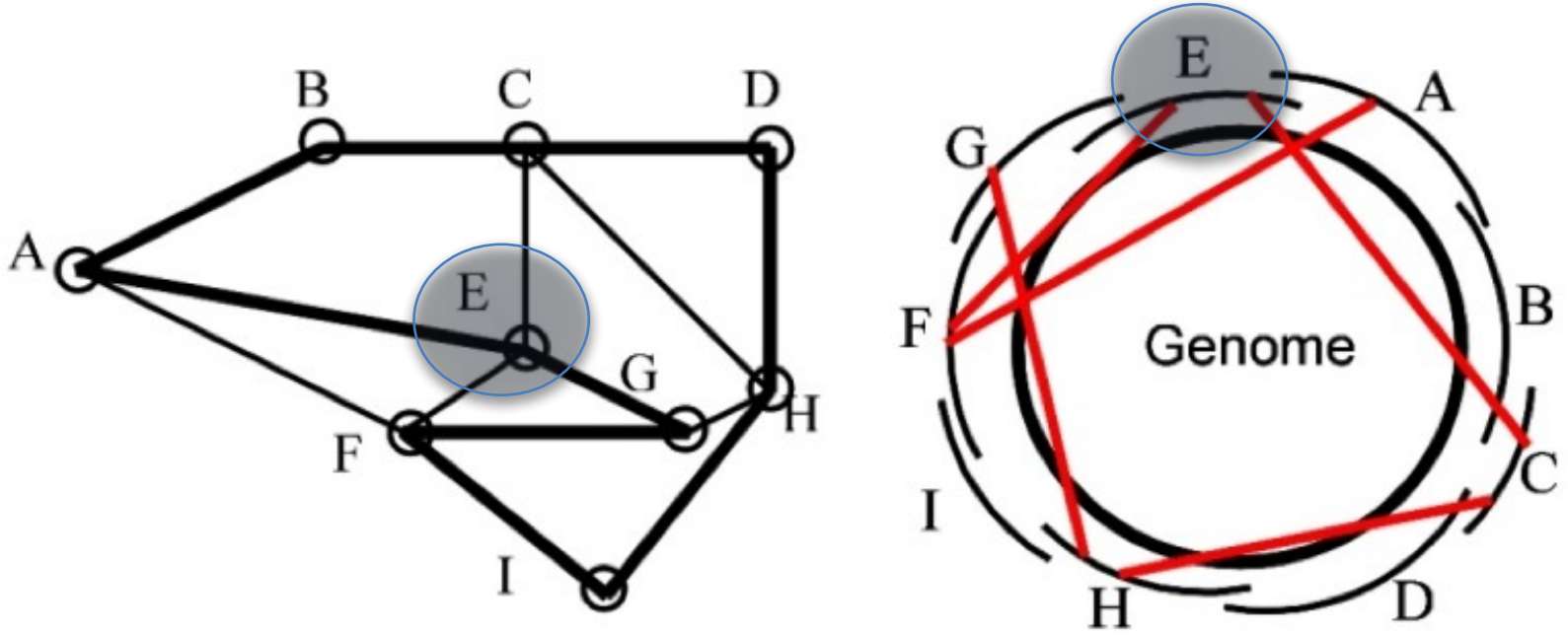
- **Clean your input**
  Remove "vector sequence", low quality, etc
- **Overlap**: What reads are intersecting?
  - Create a node for each read
  - Create directed edge for each overlap
- **Layout**: How combine the reads?
  - Simplify graph
  - Find suitable paths in the graph



Determine Hamiltonian path

« In the mathematical field of graph theory, a Hamiltonian path (or traceable path) is a path in an undirected or directed graph that visits each node exactly once. »

2/OLC: Overlap-Layout-Consensus



From http://www.cbcb.umd.edu/research/assembly_primer.shtml

2/OLC: Overlap-Layout-Consensus

Exemple:

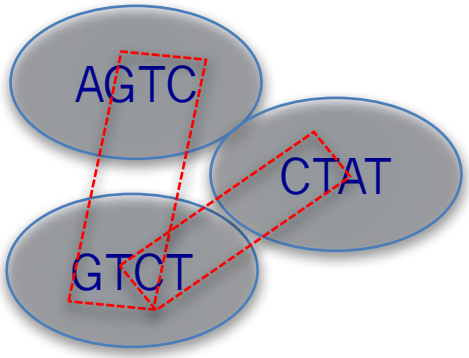
- True sequence (7bp) AGTCTAT
- 3 Reads (4bp each) AGTC (A), GTCT (B), CTAT (C)
- Alignments

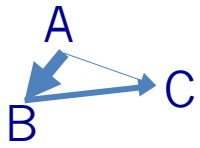
```
AGTC-      AGTC---      GTCT--
 -GTCT        ---CTAT      --CTAT
```

2/OLC: Overlap-Layout-Consensus

- Nodes are the 3 read sequences
- Edges are the overlap alignment with orientation
- Edge thickness represents score of overlap

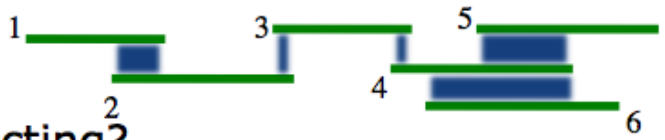
Optimal path: A -> B -> C



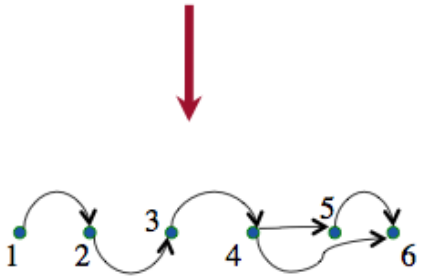


2/OLC: Overlap-Layout-Consensus

- **Clean your input**
  Remove "vector sequence", low quality, etc



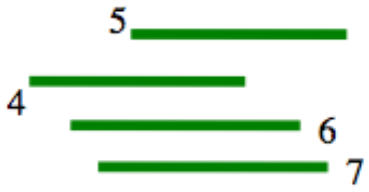
- **Overlap**: What reads are intersecting?
  - Create a node for each read
  - Create directed edge for each overlap

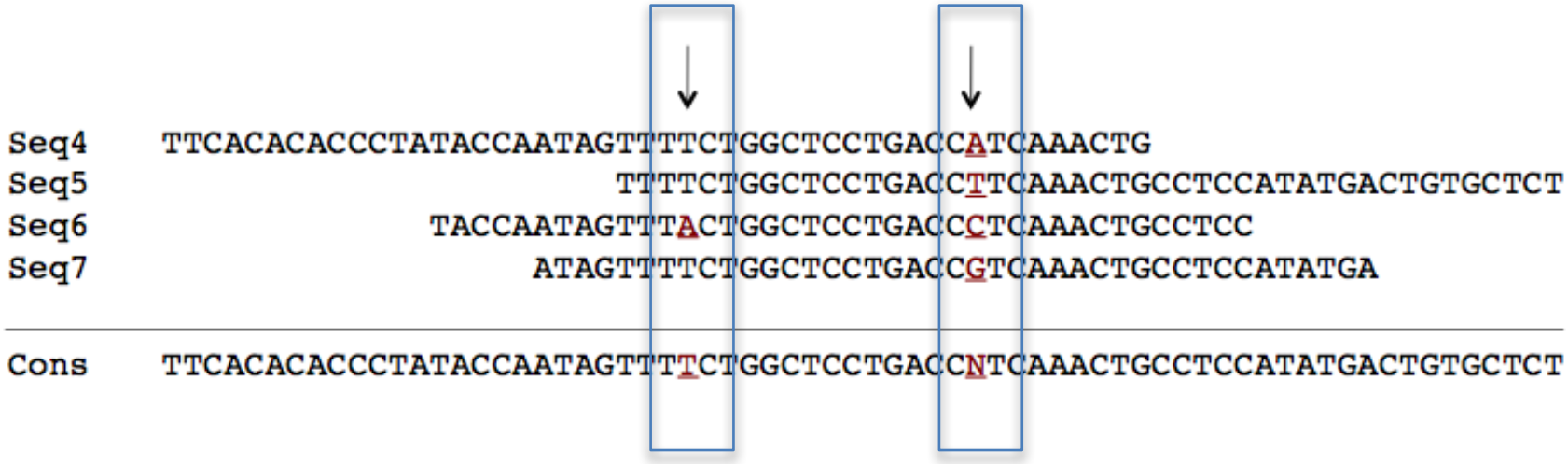


- **Layout**: How combine the reads?
  - Simplify graph
  - Find suitable paths in the graph

- **Consensus:** Derive contigs from layout

2/OLC: Overlap-Layout-Consensus



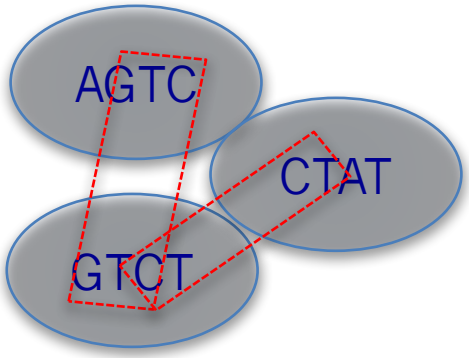


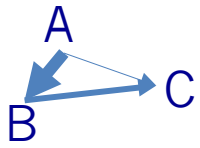
2/OLC: Overlap-Layout-Consensus

- Nodes are the 3 read sequences
- Edges are the overlap alignment with orientation
- Edge thickness represents score of overlap

Optimal path: A -> B -> C

Consensus aGTCTat





2/OLC: Overlap-Layout-Consensus

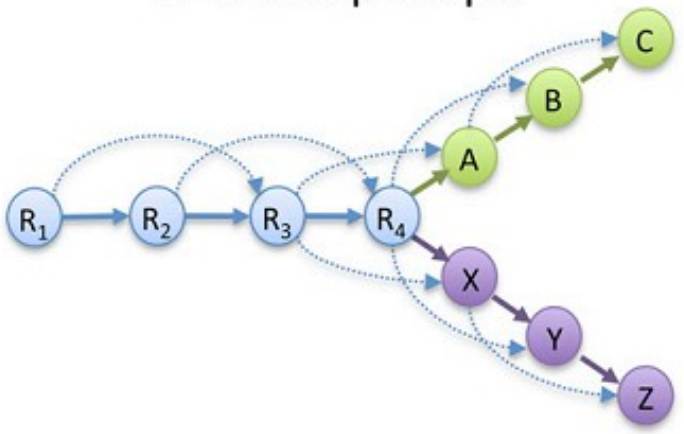
The OLC paradigm was made **popular** by the work of Gene Myers, embodied in Celera Assembler and **dominated the assembly world until the emergence of the new generation of short-read sequencing technologies.**

3/De Bruijn Graphs

A Read Layout

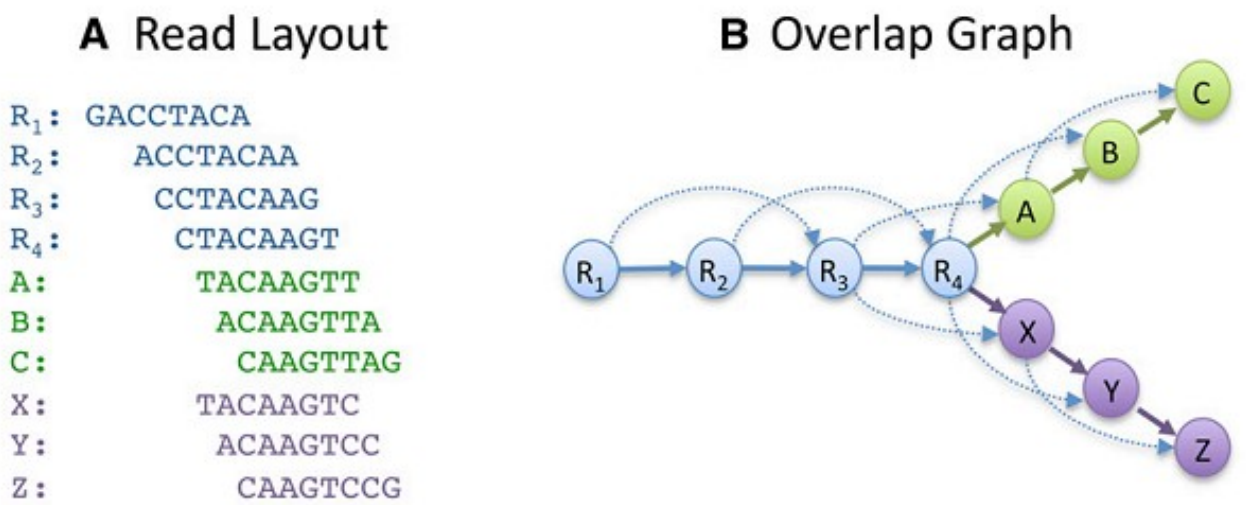
```
R1: GACCTACA
R2:   ACCTACAA
R3:    CCTACAAG
R4:     CTACAAGT
A:       TACAAGTT
B:        ACAAGTTA
C:         CAAGTTAG
X:       TACAAGTC
Y:        ACAAGTCC
Z:         CAAGTCCG
```

B Overlap Graph

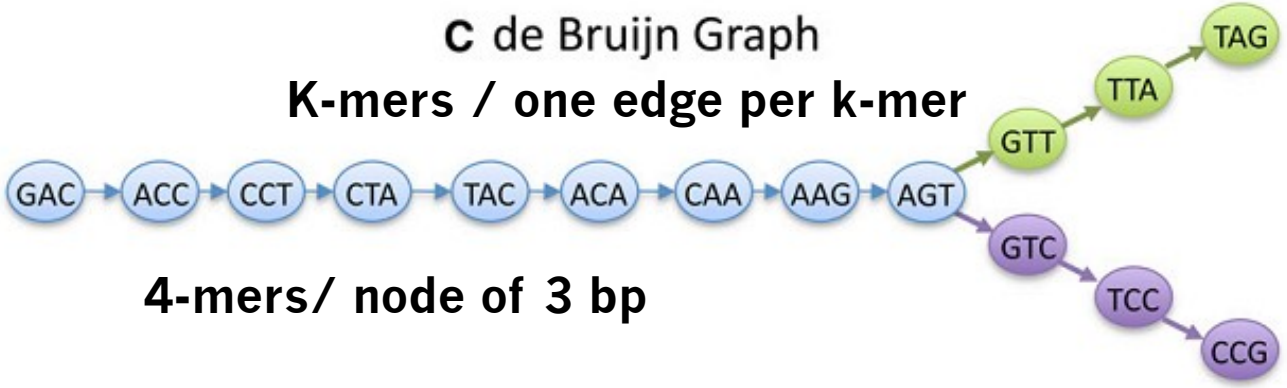


3/De Bruijn Graphs

OLC



DBG



3/De Bruijn Graphs

- Divide reads into smaller strings of size *k* (*k*-mers)
  - Break reads of *L* bp into *L-k+1* k-mers per read

3/De Bruijn Graphs

- Divide reads into smaller strings of size *k* (*k*-mers)
  - Break reads of *L* bp into *L-k+1* k-mers per read

    If L=36 and k=31, we will get 36-31+1=6 k-mers

3/De Bruijn Graphs

- Divide reads into smaller strings of size *k* (*k*-mers)
  - Break reads of *L* bp into *L-k+1* k-mers per read

    If L=36 and k=31, we will get 36-31+1=6 k-mers
  - Why create even smaller segments?

3/De Bruijn Graphs

- Divide reads into smaller strings of size *k* (*k*-mers)
  - Break reads of *L* bp into *L-k+1* k-mers per read

    If L=36 and k=31, we will get 36-31+1=6 k-mers
  - Why create even smaller segments?
    - Smaller chance of containing erroneous base
    - But the tradeoff is that repetitive sequences are more common and harder to resolve

1. Construct a de Bruijn graph (DBG)
   1. Nodes = one for each unique k-mer
   2. Edges = k-1 exact overlap between two nodes
2. Graph simplification
   1. Merge chains, remove bubbles and tips
3. Find a Eulerian path through the graph
   1. Linear time algorithm, unlike Hamiltonian

1. Construct a de Bruijn graph (DBG)

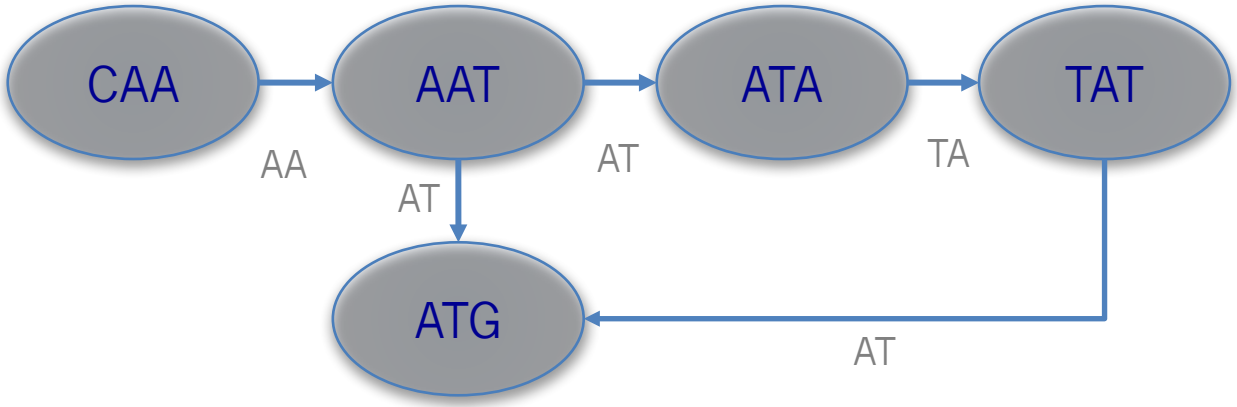
- Sequence CAATATG
- K-mers (k=3) CAA AAT ATA TAT ATG
- Graph

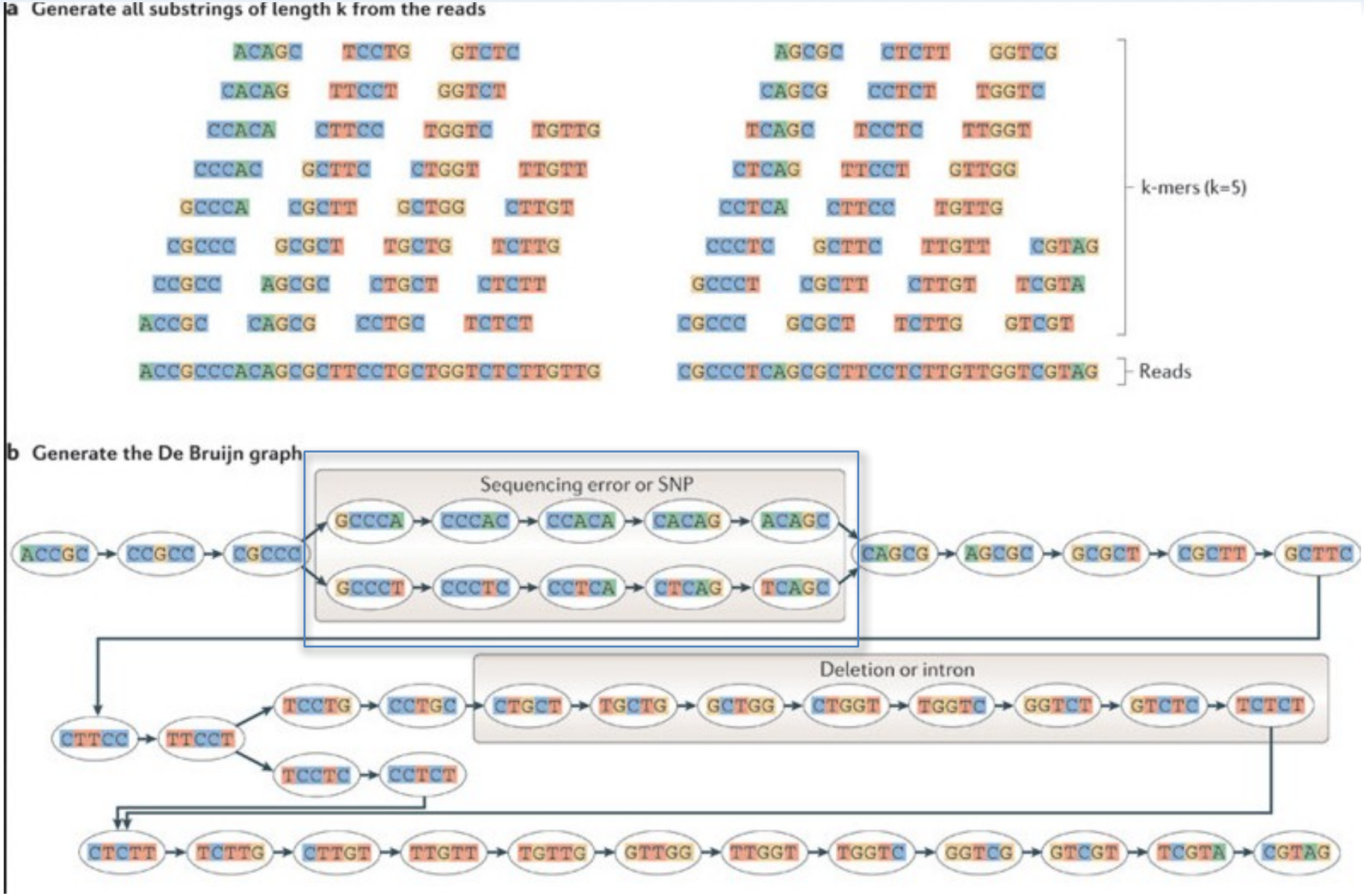
Nodes = one for each unique k-mer
Edges = k-1 exact overlap between two nodes

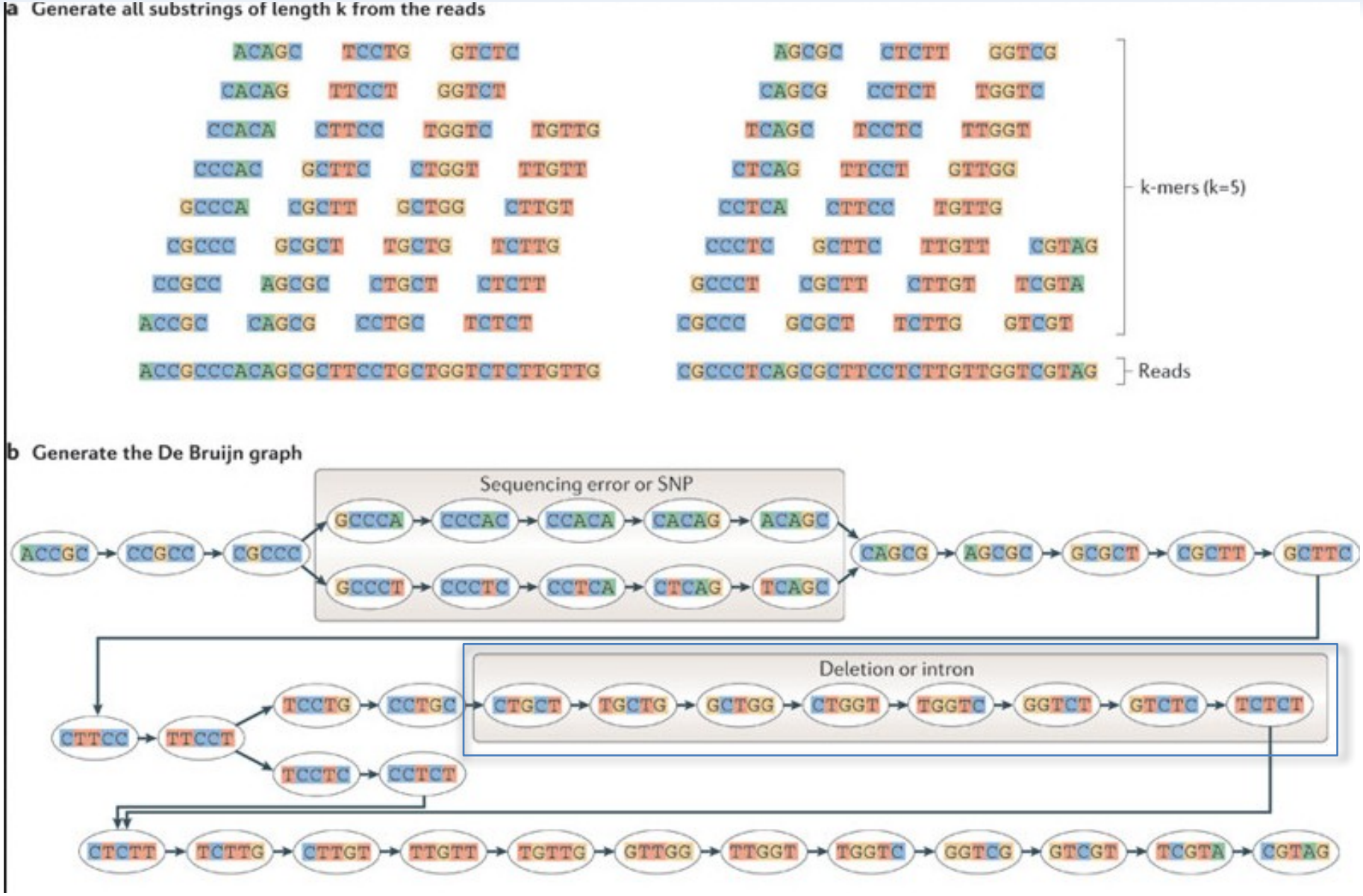
1. Construct a de Bruijn graph (DBG)

- Sequence CAATATG
- K-mers (k=3) CAA AAT ATA TAT ATG
- Graph

Nodes = one for each unique k-mer
Edges = k-1 exact overlap between two nodes



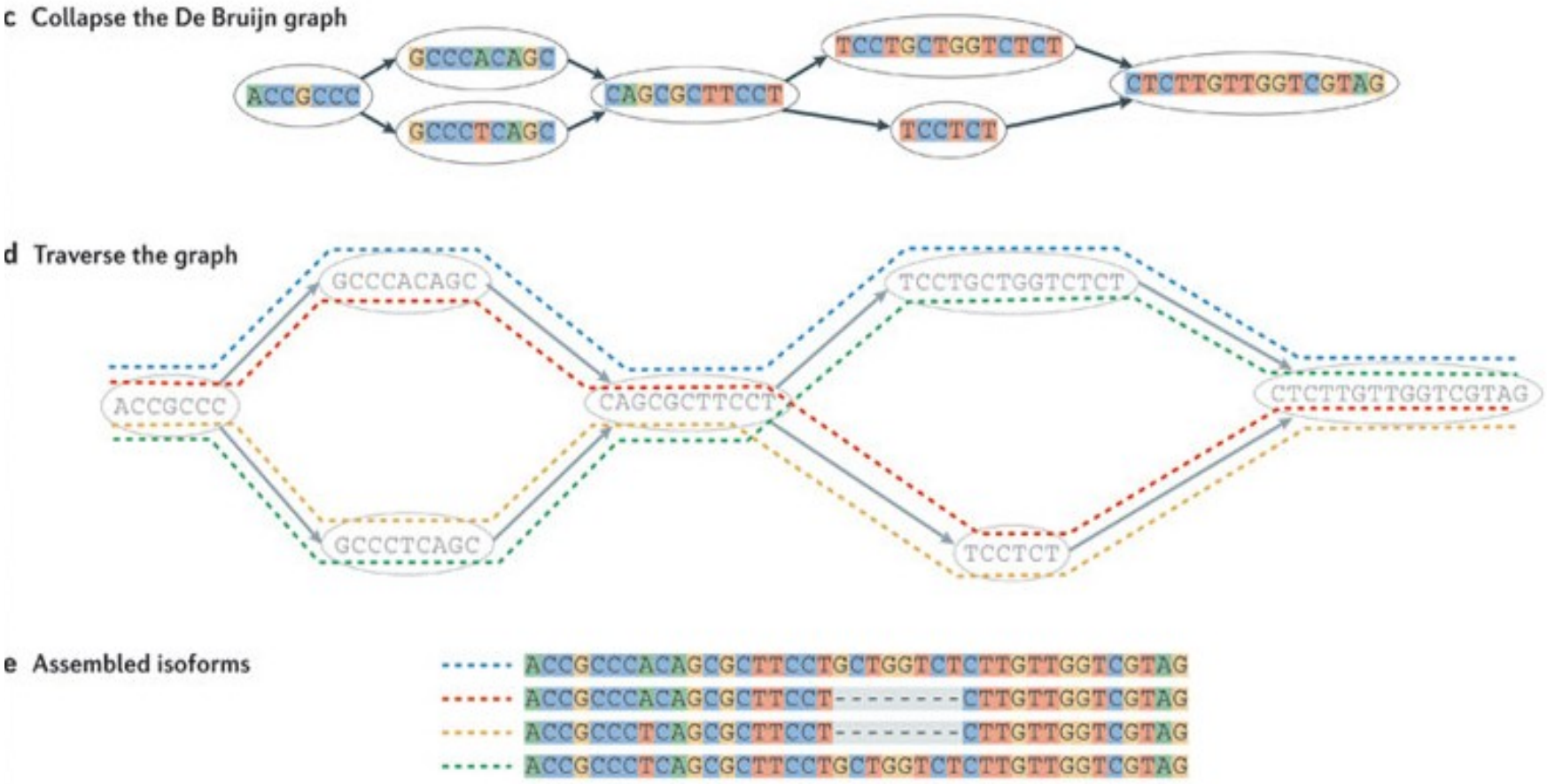

a Generate all substrings of length k from the reads
ACAGC TCCTG GTCTC
CACAG TTCCT GGTCT
CCACA CTTCC TGGTC TGTTG
CCCAC GCTTC CTGGT TTGTT
GCCCA CGCTT GCTGG CTTGT
CGCCC GCGCT TGCTG TCTTG
CCGCC AGCGC CTGCT CTCTT
ACCGC CAGCG CCTGC TCTCT
ACCGCCCACAGCGCTTCCTGCTGGTCTCTTGTTG
AGCGC CTCTT GGTCG
CAGCG CCTCT TGGTC
TCAGC TCCTC TTGGT
CTCAG TTCCT GTTGG
CCTCA CTTCC TGTTG
CCCTC GCTTC TTGTT CGTAG
GCCCT CGCTT CTTGT TCGTA
CGCCC GCGCT TCTTG GTCGT
CGCCCTCAGCGCTTCCTCTTGTTGGTCGTAG
k-mers (k=5)
Reads
b Generate the De Bruijn graph
Sequencing error or SNP
GCCCA CCCAC CCACA CACAG ACAGC
GCCCT CCCTC CCTCA CTCAG TCAGC
ACCGC CCGCC CGCCC
CAGCG AGCGC GCGCT CGCTT GCTTC
Deletion or intron
TCCTG CCTGC CTGCT TGCTG GCTGG CTGGT TGGTC GGTCT GTCTC TCTCT
CTTCC TTCCT
TCCTC CCTCT
CTCTT TCTTG CTTGT TTGTT TGTTG GTTGG TTGGT TGGTC GGTCG GTCGT TCGTA CGTAG


a Generate all substrings of length k from the reads
ACAGC TCCTG GTCTC
CACAG TTCCT GGTCT
CCACA CTTCC TGGTC TGTTG
CCCAC GCTTC CTGGT TTGTT
GCCCA CGCTT GCTGG CTTGT
CGCCC GCGCT TGCTG TCTTG
CCGCC AGCGC CTGCT CTCTT
ACCGC CAGCG CCTGC TCTCT
AGCGC CTCTT GGTCG
CAGCG CCTCT TGGTC
TCAGC TCCTC TTGGT
CTCAG TTCCT GTTGG
CCTCA CTTCC TGTTG
CCCTC GCTTC TTGTT CGTAG
GCCCT CGCTT CTTGT TCGTA
CGCCC GCGCT TCTTG GTCGT
k-mers (k=5)
ACCGCCCACAGCGCTTCCTGCTGGTCTCTTGTTG
CGCCCTCAGCGCTTCCTCTTGTTGGTCGTAG
Reads
b Generate the De Bruijn graph
Sequencing error or SNP
ACCGC CCGCC CGCCC
GCCCA CCCAC CCACA CACAG ACAGC
GCCCT CCCTC CCTCA CTCAG TCAGC
CAGCG AGCGC GCGCT CGCTT GCTTC
Deletion or intron
CTTCC TTCCT
TCCTG CCTGC
TCCTC CCTCT
CTGCT TGCTG GCTGG CTGGT TGGTC GGTCT GTCTC TCTCT
CTCTT TCTTG CTTGT TTGTT TGTTG GTTGG TTGGT TGGTC GGTCG GTCGT TCGTA CGTAG

2. graph simplification

- Remove tips or spurs
  - Dead ends in graph due to errors at read end
- Collapse bubbles
  - Errors in middle of reads
  - But could be true SNPs or diploidity
- Remove low coverage paths
  - Possible contamination

➔ Makes final Eulerian path easier and hopefully more accurate contigs



« In graph theory, an Eulerian trail (or Eulerian path) is a trail in a graph which visits every edge exactly once. »

3/De Bruijn Graphs

De Bruijn approach was popularized by the assembler Euler17 and has dominated the design of **modern assemblers** targeted at short-read sequencing data, such as **Velvet, SOAPdenovo and ALLPATHS.**

De Bruijn-graph-based approaches have been successful in assembling **highly accurate short reads** (<~100 bp, such as those generated by the Illumina Solexa technology), whereas overlap-based approaches (such as OLC or string graph) are mostly used for longer, more inaccurate data (>200 bp, such as Roche 454 and Sanger sequencing data)

Modern Sequence Assemblers

Assemblers	Technology	Availability	Notes	Refs
Genome assemblers				
ALLPATHS-LG	Illumina, Pacific Biosciences	ftp://ftp.broadinstitute.org/pub/crd/ALLPATHS/Release-LG	Requires a specific sequencing recipe (BOX 3)	40
SOAPdenovo	Illumina	http://soap.genomics.org.cn/soapdenovo.html	Also used for transcriptome and metagenome assembly	22
Velvet	Illumina, SOLiD, 454, Sanger	http://www.ebi.ac.uk/~zerbino/velvet	May have substantial memory requirements for large genomes	20
ABySS	Illumina, SOLiD, 454, Sanger	http://www.bcgsc.ca/platform/bioinfo/software/abyss	Also used for transcriptome assembly	21
Metagenome assemblers				
Genovo	454	http://cs.stanford.edu/group/genovo	Uses a probabilistic model for assembly	66
MetaVelvet	Illumina, SOLiD, 454, Sanger	http://metavelvet.dna.bio.keio.ac.jp	Based on Velvet	4
Meta-IDBA	Illumina	http://i.cs.hku.hk/~alse/hkubrg/projects/metaidba	Based on IDBA	5
Transcriptome assemblers				
Trinity	Illumina, 454	http://trinityrnaseq.sourceforge.net	Tailored to reconstruct full-length transcripts; may require substantial computational time	8
Oases	Illumina, SOLiD, 454, Sanger	http://www.ebi.ac.uk/~zerbino/oases	Based on Velvet	72
Single-cell assemblers				
SPAdes	Illumina	http://bioinf.spbau.ru/en/spades		7
IDBA-UD	Illumina	http://i.cs.hku.hk/~alse/hkubrg/projects/idba_ud	Based on IDBA	6

Note that only a few of the popular and freely available assemblers are included here for each application (a more complete list is provided in Supplementary information S1 (table)), and all of the listed assemblers (except Genovo) are based on de Bruijn graph construction. IBDA, Iterative De Bruijn graph short read Assembler.

Modern Sequence Assemblers

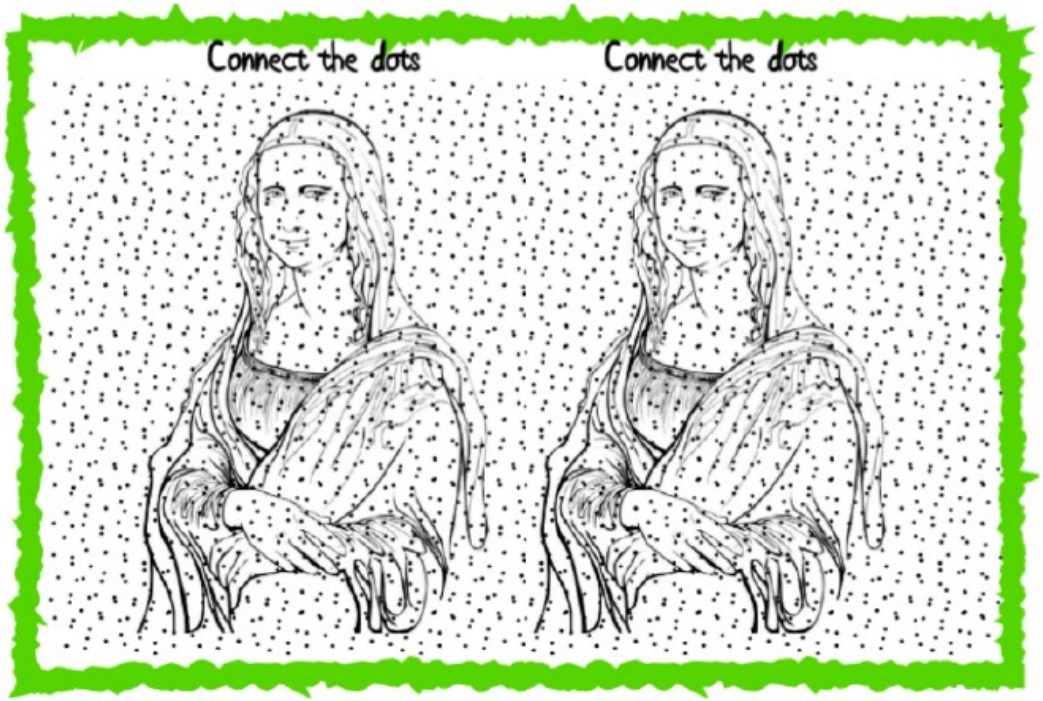
Assemblers	Technology	Availability	Notes	Refs
Genome assemblers				
ALLPATHS-LG	Illumina, Pacific Biosciences	ftp://ftp.broadinstitute.org/pub/crd/ALLPATHS/Release-LG	Requires a specific sequencing recipe (BOX 3)	40
SOAPdenovo	Illumina	http://soap.genomics.org.cn/soapdenovo.html	Also used for transcriptome and metagenome assembly	22
Velvet	Illumina, SOLiD, 454, Sanger	http://www.ebi.ac.uk/~zerbino/velvet	May have substantial memory requirements for large genomes	20
ABySS	Illumina, SOLiD, 454, Sanger	http://www.bcgsc.ca/platform/bioinfo/software/abyss	Also used for transcriptome assembly	21
Metagenome assemblers				
Genovo	454	http://cs.stanford.edu/group/genovo	Uses a probabilistic model for assembly	45
MetaVelvet	Illumina, SOLiD, 454, Sanger	http://metavelvet.dna.bio.keio.ac.jp	Based on Velvet	4
Meta-IDBA	Illumina	http://i.cs.hku.hk/~alse/hkubrg/projects/metaidba	Based on IDBA	5
Transcriptome assemblers				
Trinity	Illumina, 454	http://trinityrnaseq.sourceforge.net	Tailored to reconstruct full-length transcripts; may require substantial computational time	8
Oases	Illumina, SOLiD, 454, Sanger	http://www.ebi.ac.uk/~zerbino/oases	Based on Velvet	72
Single-cell assemblers				
SPAdes	Illumina	http://bioinf.spbau.ru/en/spades		7
IDBA-UD	Illumina	http://i.cs.hku.hk/~alse/hkubrg/projects/idba_ud	Based on IDBA	6

Note that only a few of the popular and freely available assemblers are included here for each application (a more complete list is provided in Supplementary information S1 (table)), and all of the listed assemblers (except Genovo) are based on de Bruijn graph construction. IBDA, Iterative De Bruijn graph short read Assembler.

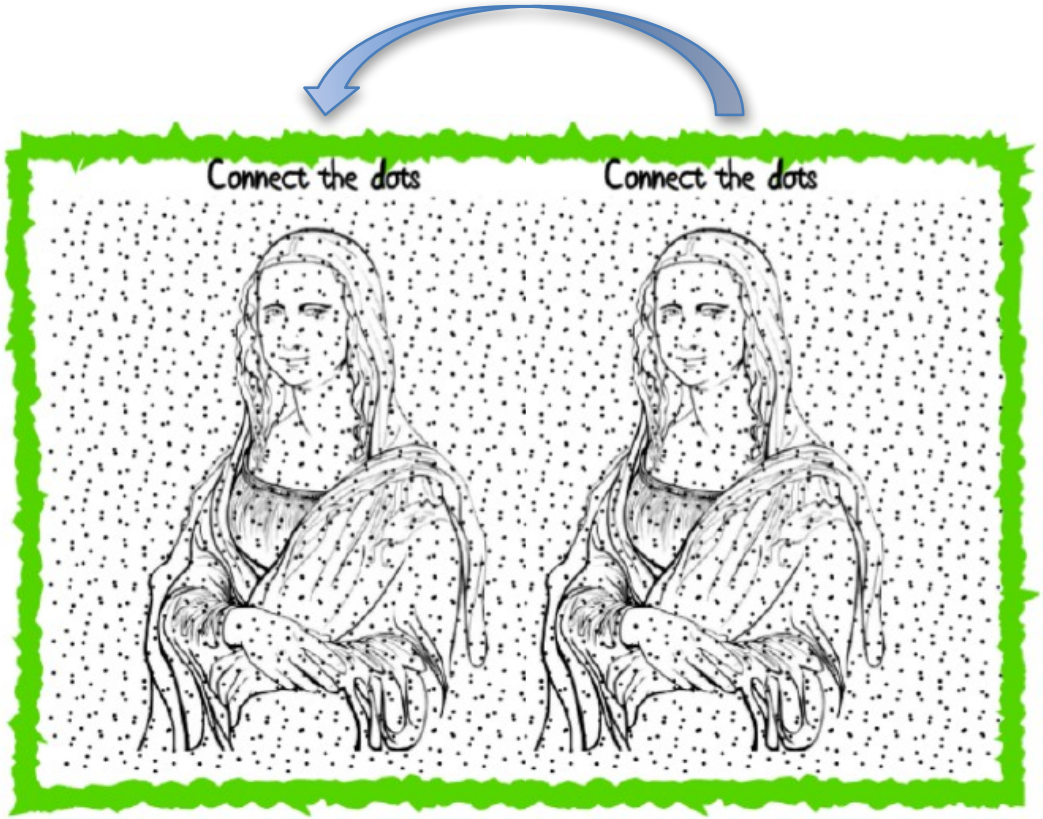
Genome Assembly Gold-Standard Evaluations

http://gage.cbcb.umd.edu

The main source of mis-assembly: Repeats



The main source of mis-assembly: Repeats

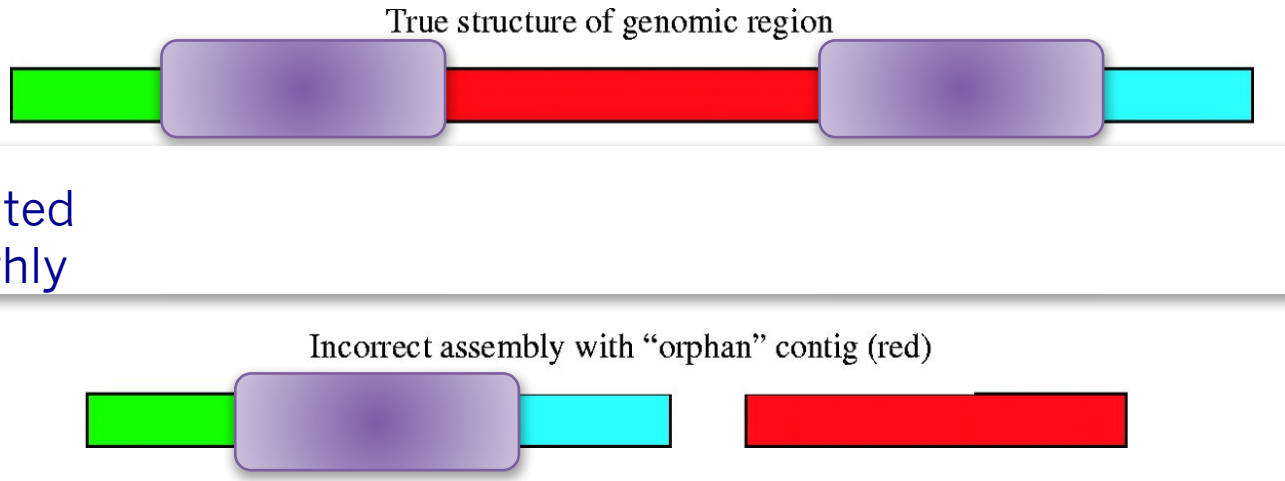


The main source of mis-assembly: Repeats



The main source of mis-assembly: Repetitive DNA

Segments of DNA repeated yield fragments with highly similar sequences that originate from different places in the genome.



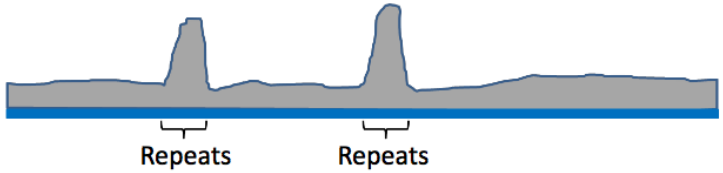
(Salzberg and Yorke, 2005; Miller et al., 2010)

Omic impact

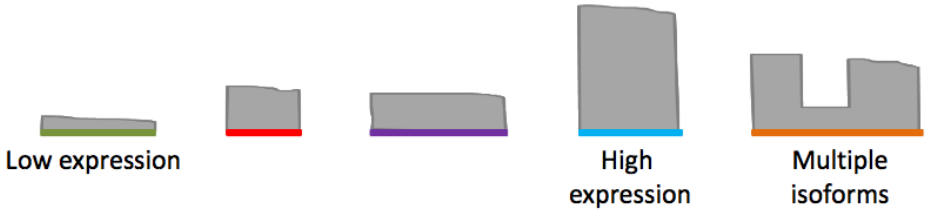
Transcriptome

➔ nearly identical sequences may originate from different transcripts

Genome assemblers expect even coverage.
Assembled regions with high coverage are assumed to be repeats.



But individual genes within a transcriptome will have very different amounts of coverage...

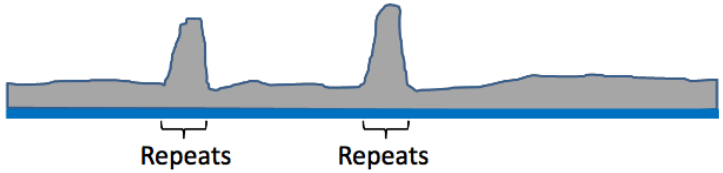


Omic impact

Transcriptome

➔ nearly identical sequences may originate from different transcripts

Genome assemblers expect even coverage.
Assembled regions with high coverage are assumed to be repeats.



But individual genes within a transcriptome will have very different amounts of coverage...

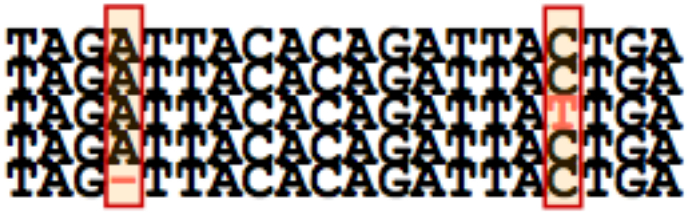


Metagenomic samples

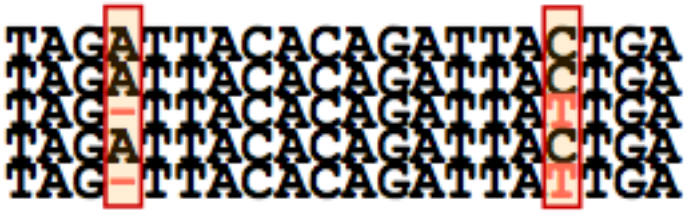
➔ nearly identical sequences may originate from genomes within the sample.

Other sources of mis-assembly

Sequencing error



Polymorphism



-> Increase the coverage

Assembly quality relies on the tech. of sequencing

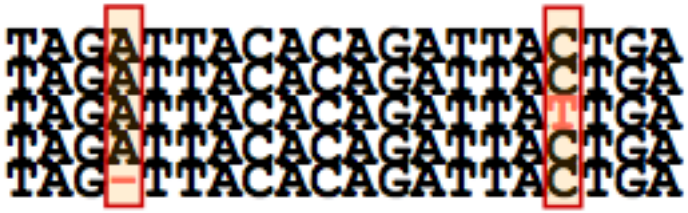
Technology	Read length (bp)	Error rate	Native paired-end read support
ABI/Solid	75	Low (~2%)	Yes
Illumina/Solexa	100–150	Low (<2%)	Yes
IonTorrent	~200	Medium (~4%)*	No
Roche/454	400–600	Medium (~4%)*	No
Sanger	Up to ~2,000 bp	Low (~2%)	Yes
Pacific Biosciences	Up to ~15,000‡	High (~18%)	Yes (in strobe read mode)

Larger are the repeats and higher is the rate of mis-assembly. We need long reads with low error rate.

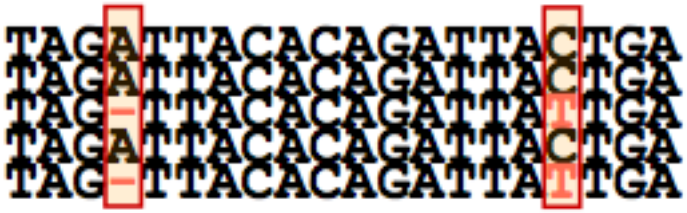
(Nagarajan and Pop 2013)

Other sources of mis-assembly

Sequencing error



Polymorphism



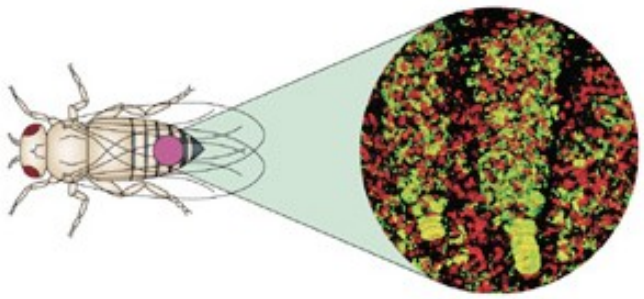
-> Increase the coverage

Contamination of endosymbiotic bacteria (*i.e.* that live inside invertebrate species) from its host in the sequencing library.



-> Test contamination before the assembly

Drosophila



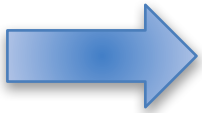
Wolbachia

(Werren et al., 2008)

Sequencing to Assembly

In:
A set of reads





Out:
A genome model

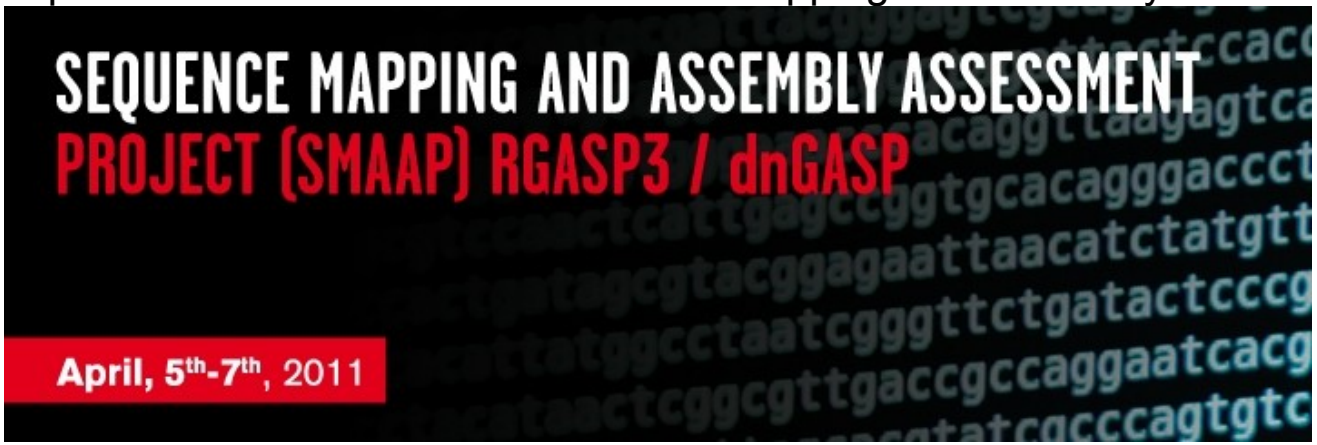


In practice:
A set of *contigs*



Sequence Mapping and Assembly Assessment Project (SMAAP)

Initiative to compare and evaluate the best tools for mapping and assembly.



http://www.biocat.cat/es/cidc/programa-de-actividades/sequence-mapping-and-assembly-assessment-project-smaap

Assemblathon: A competitive assessment of de novo short read assembly methods



Resource

Assemblathon 1: A competitive assessment of de novo short read assembly methods

Dent Earl,[1,2] Keith Bradnam,[3] John St. John,[1,2] Aaron Darling,[3] Dawei Lin,[3,4] Joseph Fass,[3,4] Hung On Ken Yu,[3] Vince Buffalo,[3,4] Daniel R. Zerbino,[2] Mark Diekhans,[1,2] Ngan Nguyen,[1,2] Pramila Nuwantha Ariyaratne,[5] Wing-Kin Sung,[5,6] Zemin Ning,[7] Matthias Haimel,[8] Jared T. Simpson,[7] Nuno A. Fonseca,[9] İnanç Birol,[10] T. Roderick Docking,[10] Isaac Y. Ho,[11] Daniel S. Rokhsar,[11,12] Rayan Chikhi,[13,14] Dominique Lavenier,[13,14,15] Guillaume Chapuis,[13,14] Delphine Naquin,[14,15] Nicolas Maillet,[14,15] Michael C. Schatz,[16] David R. Kelley,[17] Adam M. Phillippy,[17,18] Sergey Koren,[17,18] Shiaw-Pyng Yang,[19] Wei Wu,[19] Wen-Chi Chou,[20] Anuj Srivastava,[20] Timothy I. Shaw,[20] J. Graham Ruby,[21,23] Peter Skewes-Cox,[21,22,23] Miguel Betegon,[21,23] Michelle T. Dimon,[21,23] Victor Solovyev,[24] Igor Seledtsov,[25] Petr Kosarev,[25] Denis Vorobyev,[25] Ricardo Ramirez-Gonzalez,[26] Richard Leggett,[27] Dan MacLean,[27] Fangfang Xia,[28] Ruibang Luo,[29] Zhenyu Li,[29] Yinlong Xie,[29] Binghang Liu,[29] Sante Gnerre,[30] Iain MacCallum,[30] Dariusz Przybylski,[30] Filipe J. Ribeiro,[30] Shuangye Yin,[30] Ted Sharpe,[30] Giles Hall,[30] Paul J. Kersey,[8] Richard Durbin,[7] Shaun D. Jackman,[10] Jarrod A. Chapman,[11] Xiaoqiu Huang,[31] Joseph L. DeRisi,[21,23] Mario Caccamo,[26] Yingrui Li,[29] David B. Jaffe,[30] Richard E. Green,[2] David Haussler,[1,2,23] Ian Korf,[3,32] and Benedict Paten[1,2,33]

[1–32] [Author affiliations appear at the end of the paper.]

Low-cost short read sequencing technology has revolutionized genomics, though it is only just becoming practical for the high-quality de novo assembly of a novel large genome. We describe the Assemblathon 1 competition, which aimed to comprehensively assess the state of the art in de novo assembly methods when applied to current sequencing technologies. In a collaborative effort, teams were asked to assemble a simulated Illumina HiSeq data set of an unknown, simulated diploid genome. A total of 41 assemblies from 17 different groups were received. Novel haplotype aware assessments of coverage, contiguity, structure, base calling, and copy number were made. We establish that within this benchmark: (1) It is possible to assemble the genome to a high level of coverage and accuracy, and that (2) large differences exist between the assemblies, suggesting room for further improvements in current methods. The simulated benchmark, including the correct answer, the assemblies, and the code that was used to evaluate the assemblies is now public and freely available from http://www.assemblathon.org/.

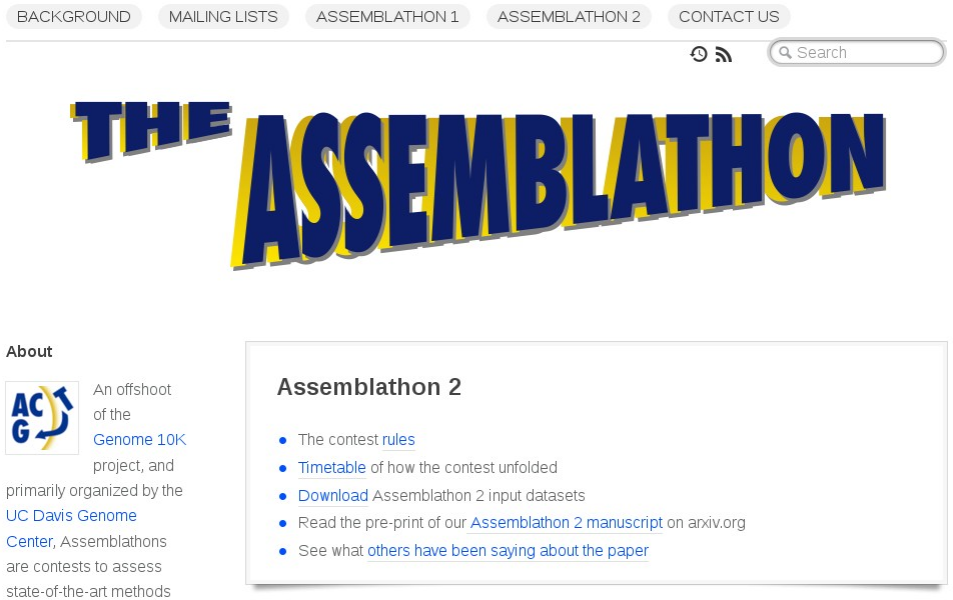
[Supplemental material is available for this article.]

Sequence assembly is the problem of merging and ordering shorter fragments, termed "reads," sampled from a set of larger sequences in order to reconstruct the larger sequences. The output of an assembly is typically a set of "contigs," which are contiguous sequence fragments, ordered and oriented into "scaffold" sequences, with gaps between contigs within scaffolds representing regions of uncertainty.

There are numerous subclasses of assembly problems that can be distinguished by, among other things, the nature of (1) the reads, (2) the types of sequences being assembled, and (3) the availability of homologous (related) and previously assembled sequences, such as a reference genome or the genome of a closely related species (Pop and Salzberg 2008; Chaisson et al. 2009; Trapnell and Salzberg 2009). In this work we focus on the evaluation of methods for de novo genome assembly using low-cost "short read" technology, where the reads are comparatively short in length but large in number, the sequences being assembled represent a novel diploid genome and the nearest homologous genome to that being assembled is significantly diverged.

[33]Corresponding author.
E-mail benedict@soe.ucsc.edu.
Article published online before print. Article, supplemental material, and publication date are at http://www.genome.org/cgi/doi/10.1101/gr.126599.111. Freely available online through the *Genome Research* Open Access option.

21:2224–2241 © 2011 by Cold Spring Harbor Laboratory Press; ISSN 1088-9051/11; www.genome.org



BACKGROUND MAILING LISTS ASSEMBLATHON 1 ASSEMBLATHON 2 CONTACT US

Search

THE ASSEMBLATHON

About

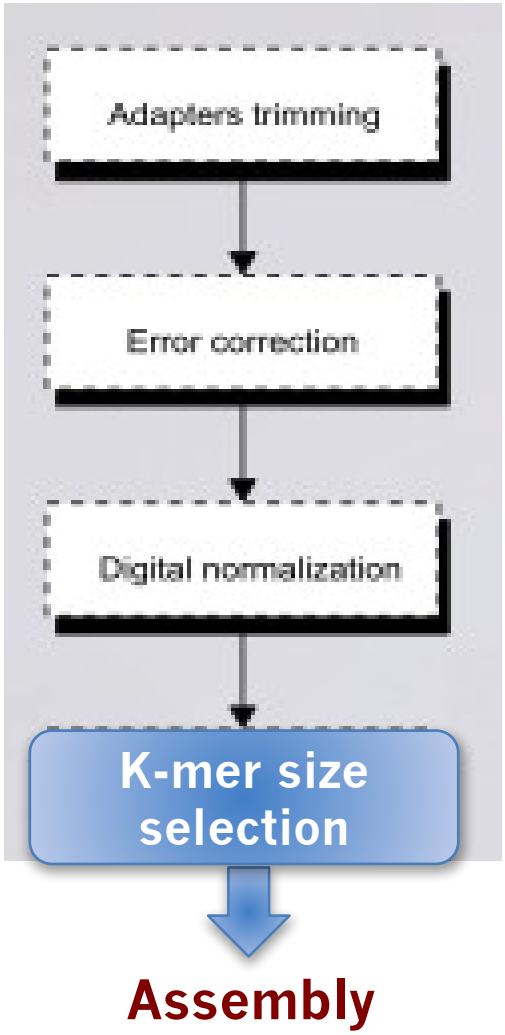
An offshoot of the Genome 10K project, and primarily organized by the UC Davis Genome Center, Assemblathons are contests to assess state-of-the-art methods in the field of genome assembly.

Assemblathon 1 occurred at the end of 2010 and the results were published in late 2011. A second Assemblathon, using real data from three vertebrate species, started in June 2011 and a manuscript has recently been submitted.

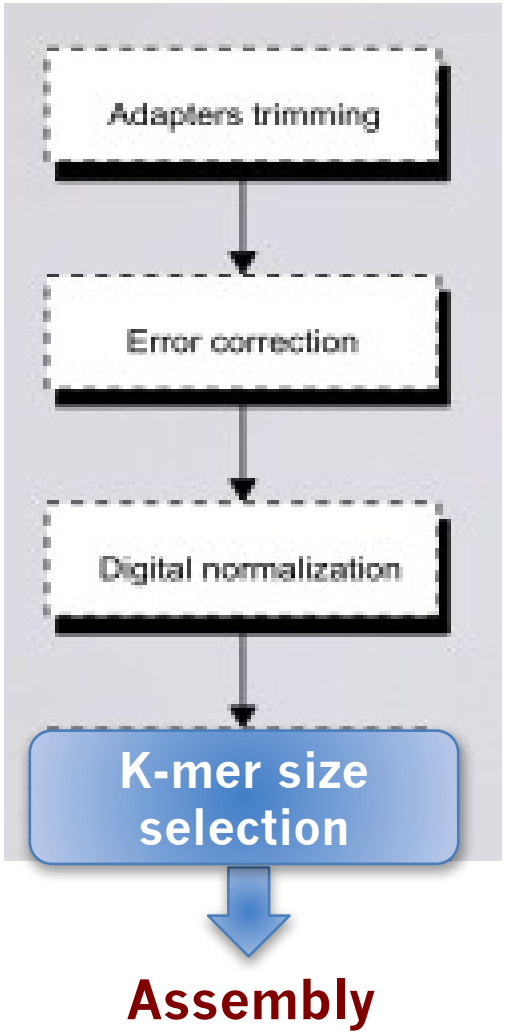
Assemblathon 2

- The contest rules
- Timetable of how the contest unfolded
- Download Assemblathon 2 input datasets
- Read the pre-print of our Assemblathon 2 manuscript on arxiv.org
- See what others have been saying about the paper

From reads to assembly



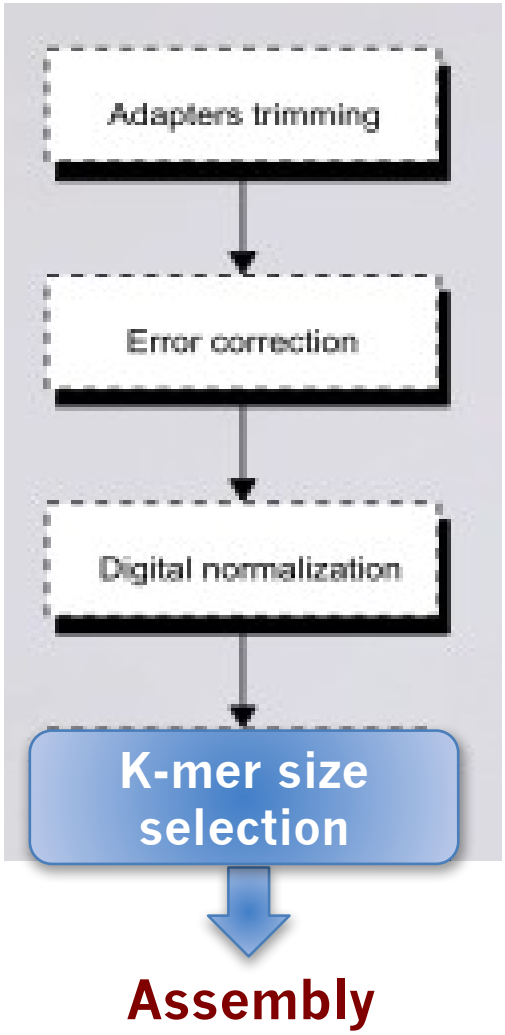
From reads to assembly



Most assemblers cut reads into k-mers

read	ACTGATGAC
k-mers (k=3)	ACT
	CTG
	TGA
	GAT
	ATG
	TGA
	GAC

From reads to assembly

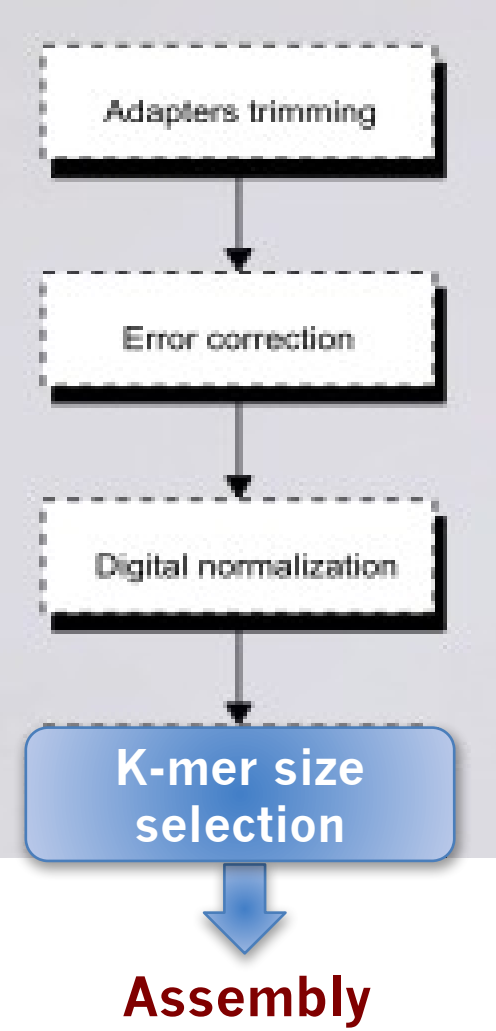


Most assemblers cut reads into k-mers

read	ACTGATGAC
k-mers (k=3)	ACT
	CTG
	TGA
	GAT
	ATG
	TGA
	GAC

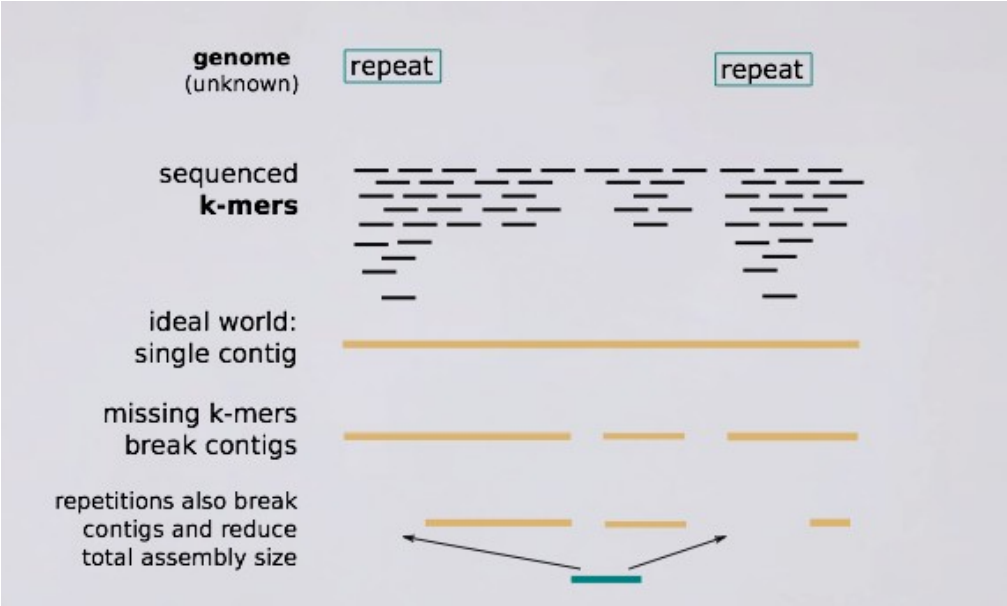
Why this is so important?

From reads to assembly



Why this is so important?

Again repeats!!!



K should be kept small to prevent the overuse of computer memory, while still large enough so that most k-mers are unique in the genome

k-mer size < mean repeat size

How to estimate the optimum k-mer size?

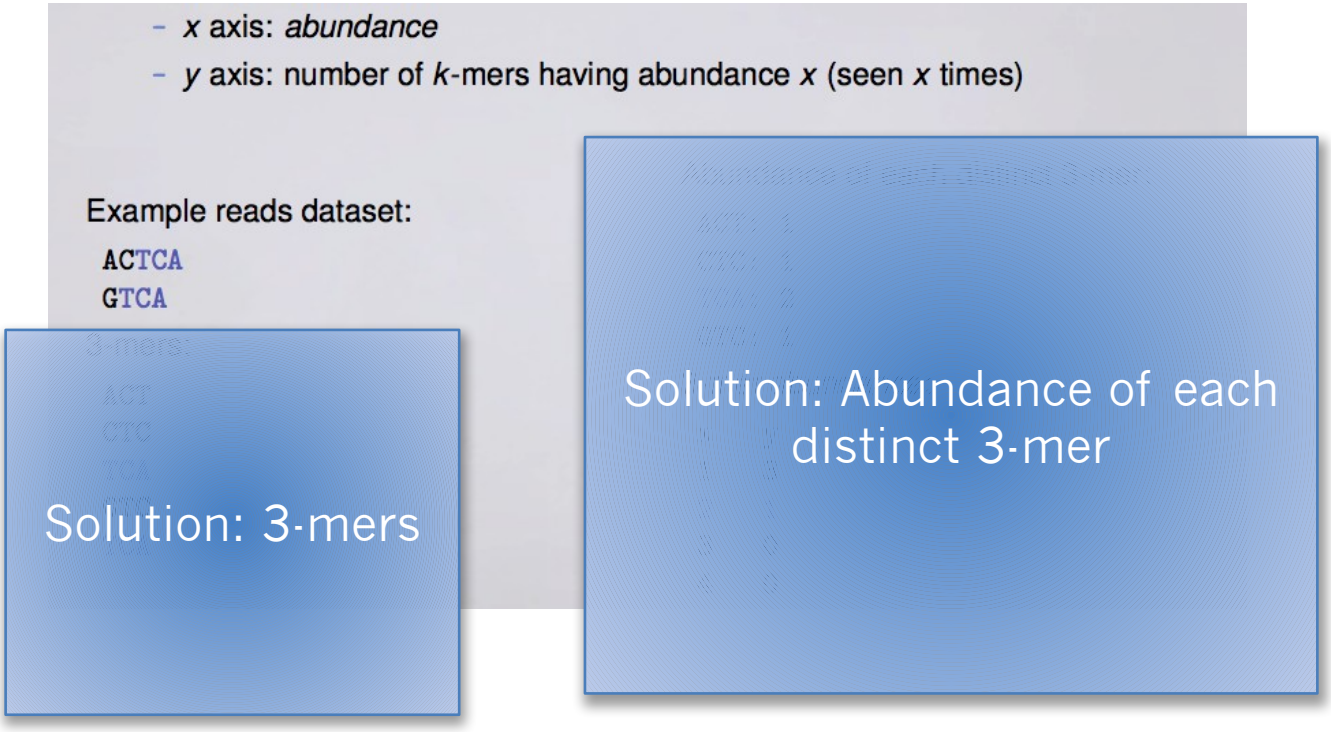
Without looking at the data

$$\mathbf{k_{opt} = | \ (N_k/G) - C \ |}$$

where N_k is the total number of k-mers in the reads,
G is the estimated genome size,
C is the desired k-mer coverage.

How to estimate the optimum k-mer size?

Looking at the data -> Compute the k-mer abundance histogram



How to estimate the optimum k-mer size?

Looking at the data -> Compute the k-mer abundance histogram

- *x* axis: *abundance*
- *y* axis: number of *k*-mers having abundance *x* (seen *x* times)

Example reads dataset:

ACTCA
GTCA

3-mers:

ACT
CTC
TCA
GTC
TCA

Solution: Abundance of each distinct 3-mer

How to estimate the optimum k-mer size?

Looking at the data -> Compute the k-mer abundance histogram

- *x* axis: *abundance*
- *y* axis: number of *k*-mers having abundance *x* (seen *x* times)

Example reads dataset:

ACTCA
GTCA

3-mers:

ACT
CTC
TCA
GTC
TCA

Abundance of each distinct 3-mer:

ACT: 1
CTC: 1
TCA: 2
GTC: 1

3-mer abundance:

x	y
1	3
2	1
3	0
4	0

How to estimate the optimum k-mer size?

Looking at the data -> Compute the k-mer abundance histogram

- Severals tools to build the k-mer histograms already exist (e.g. **k-mer counting, Jellyfish, DSK**...)

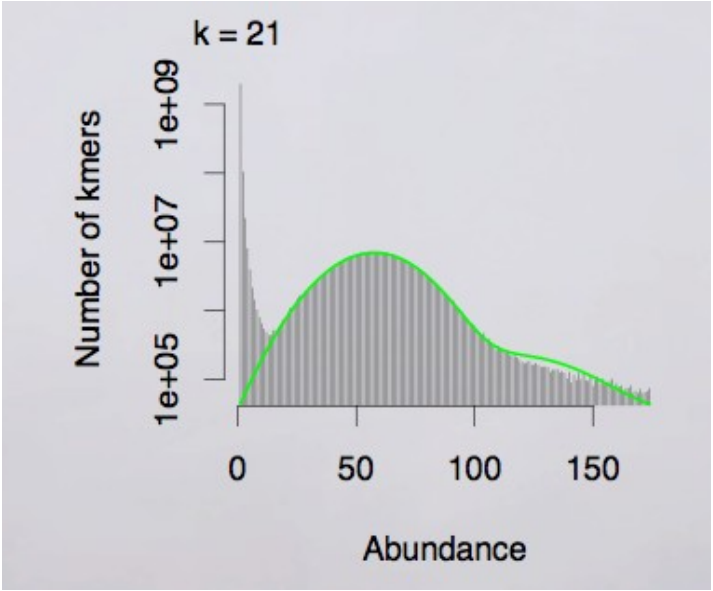
- Chikhi R., Medvedev P. designed one approach to estimate the optimim k-mer (**KmerGenie** Bioinformatics 2013)

Understand the k-mer histogram

GAGE dataset
Human chr14 ~ 88 Mb

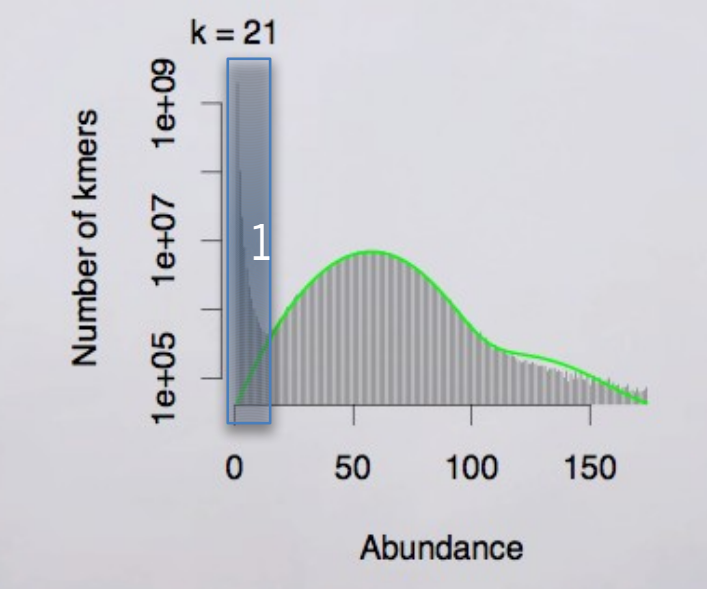
k=21

We expect to see multiple peaks, from different causes.



Understand the k-mer histogram

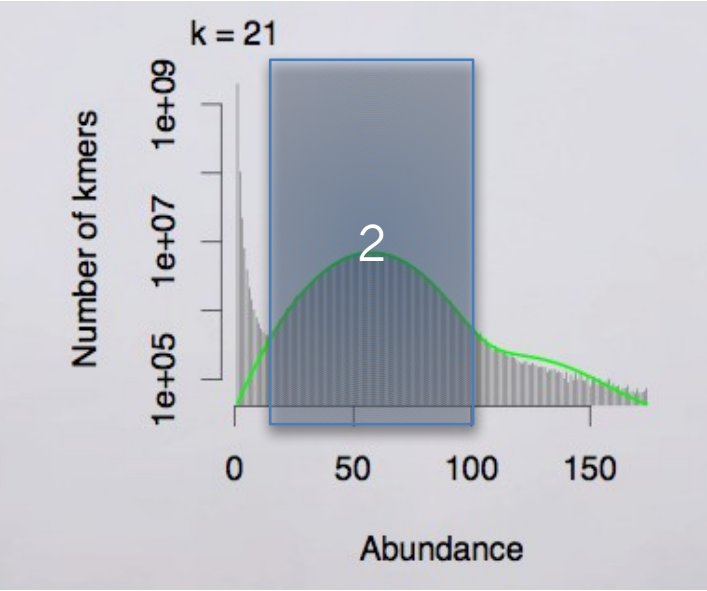
Low k-mer frequency = Erroneous k-mers



- Sequencing errors,
- PCR amplification errors,
- Polymorphisms...

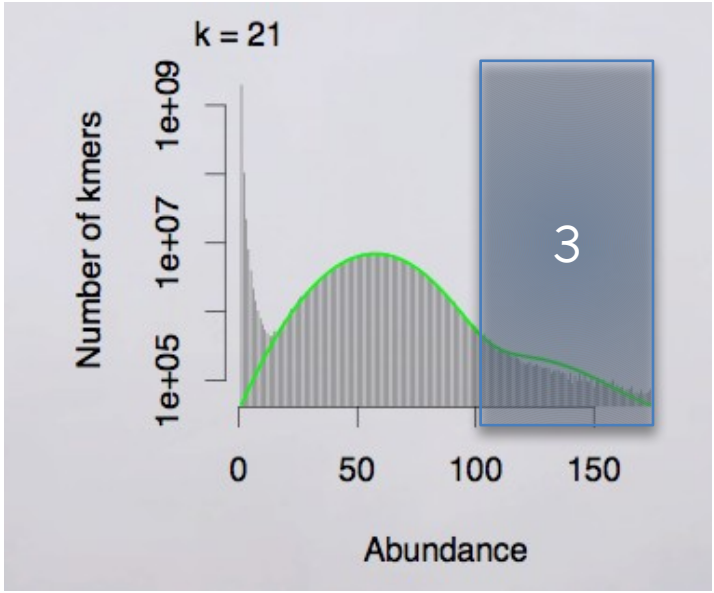
Note: For metagenomic libraries, that polymorphism may be a major contributor to the initial peak

Understand the k-mer histogram



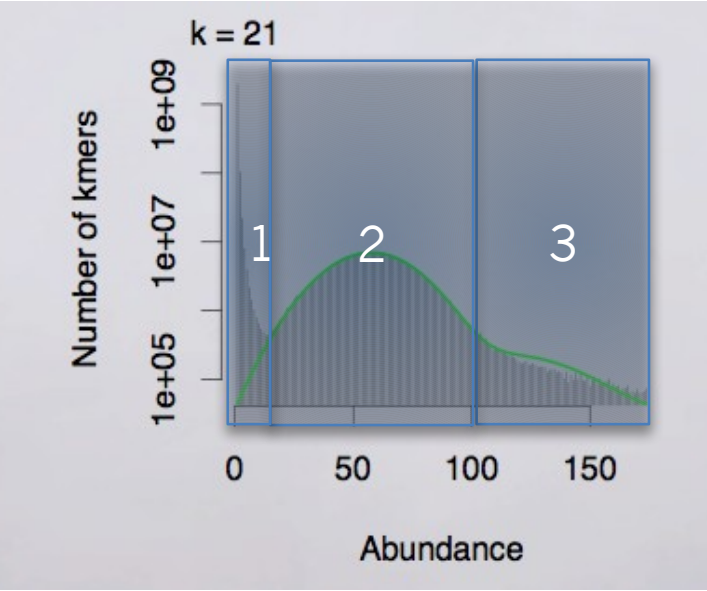
Genomic non-repeated k-mers

Understand the k-mer histogram



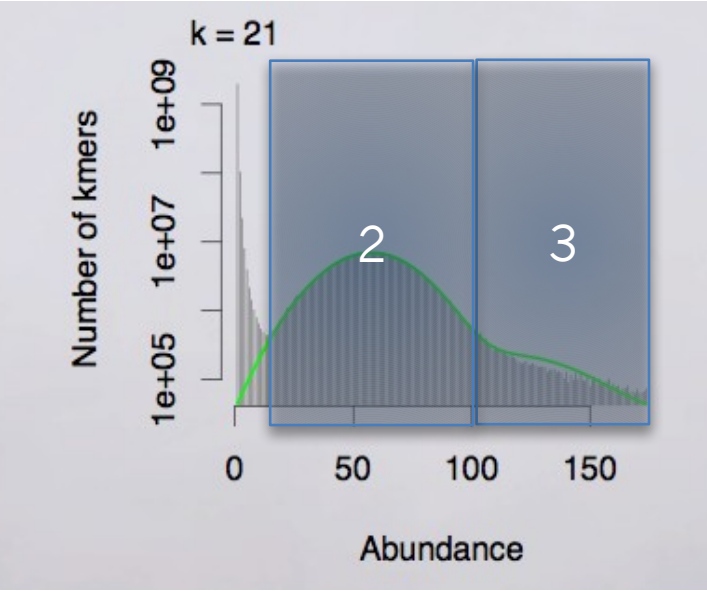
Genomic repeated k-mers or artifacts

Understand the k-mer histogram



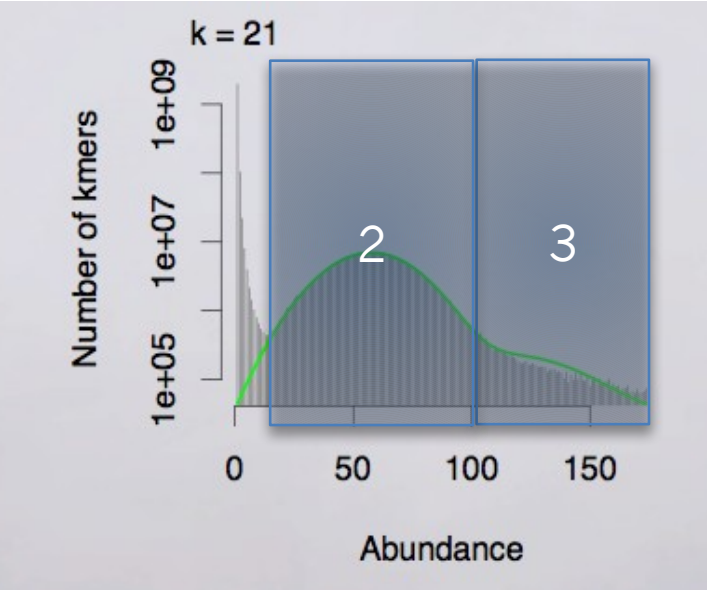
-> genomic and sequencing characteristics

Understand the k-mer histogram



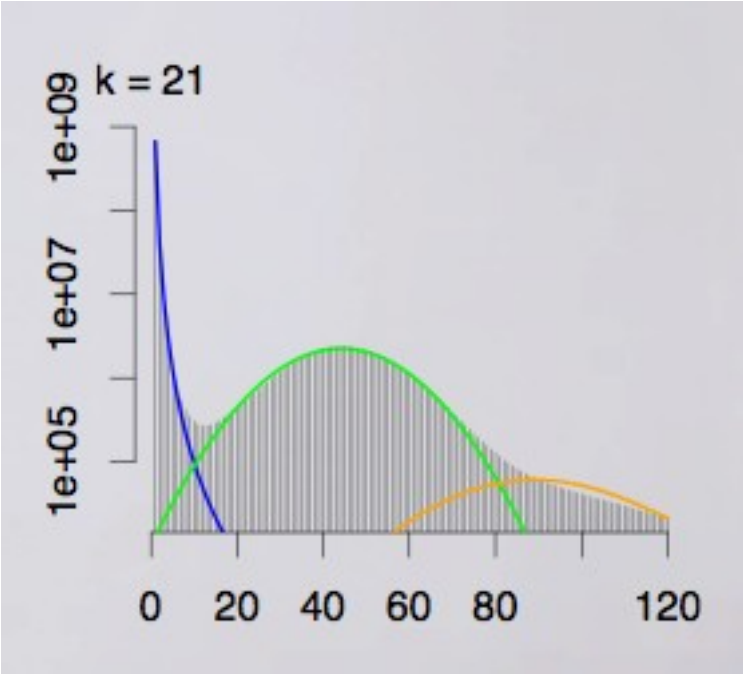
2 + 3 = total number of distinct k-mers covering the genome

Understand the k-mer histogram



2 + 3 = total number of distinct k-mers covering the genome

How to determine exactly the area



Quake's statistical model

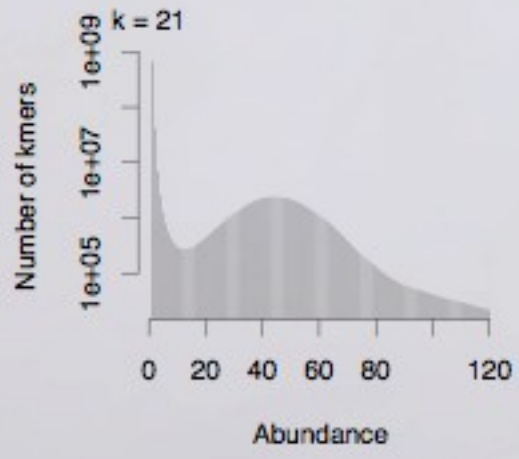
They fit a Gamma distribution to the untrusted k-mers, a Gaussian for unrepeated trusted k-mers and a Zeta for the high frequency repeated trusted k-mers.

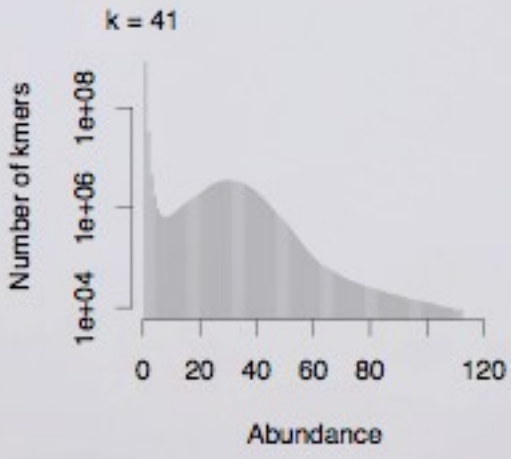
The distribution of the trusted reads is actually expected to be Poisson, but the variance is significantly larger than the mean due to sequencing biases

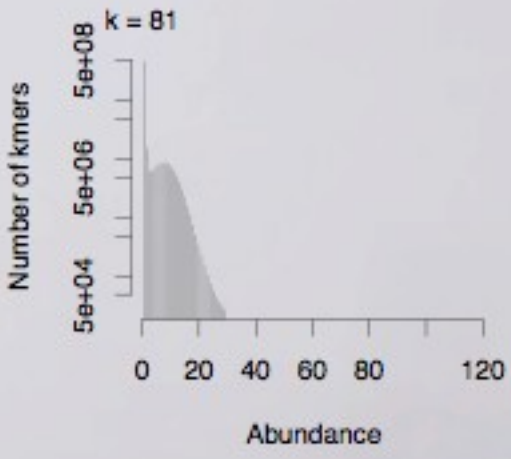
(Chor et al., 2009; Kelley et al., 2010)

Now, read the k-mer histograms

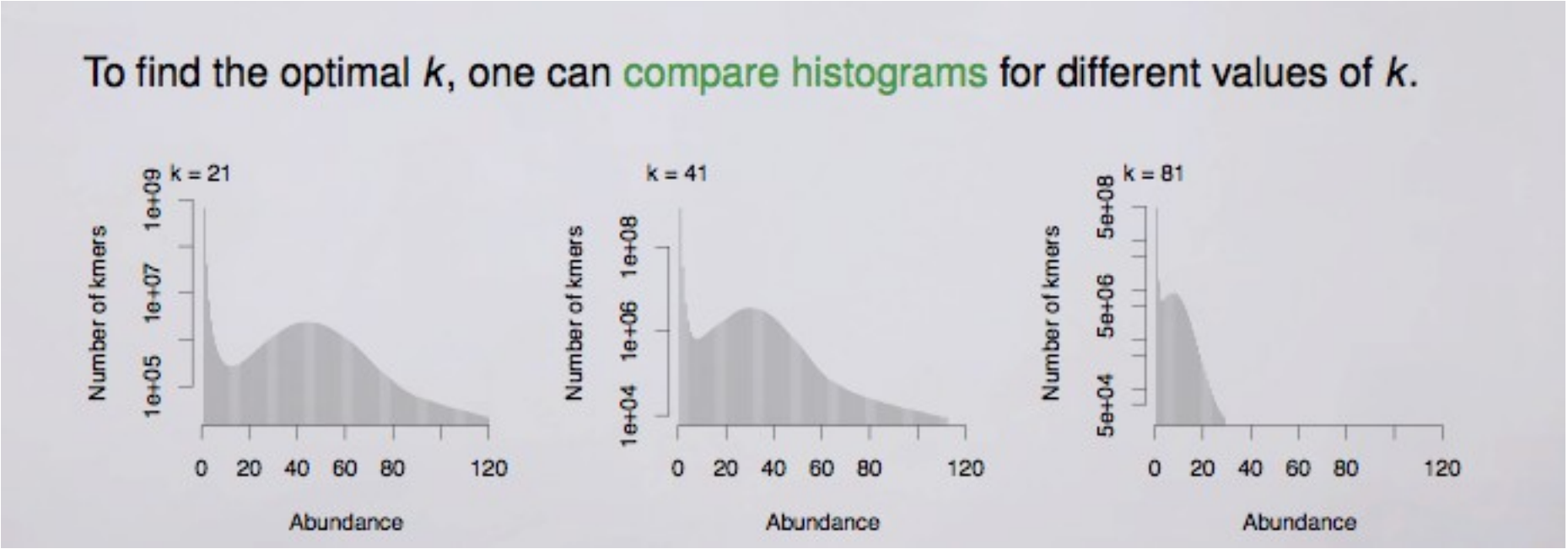
To find the optimal *k*, one can compare histograms for different values of *k*.





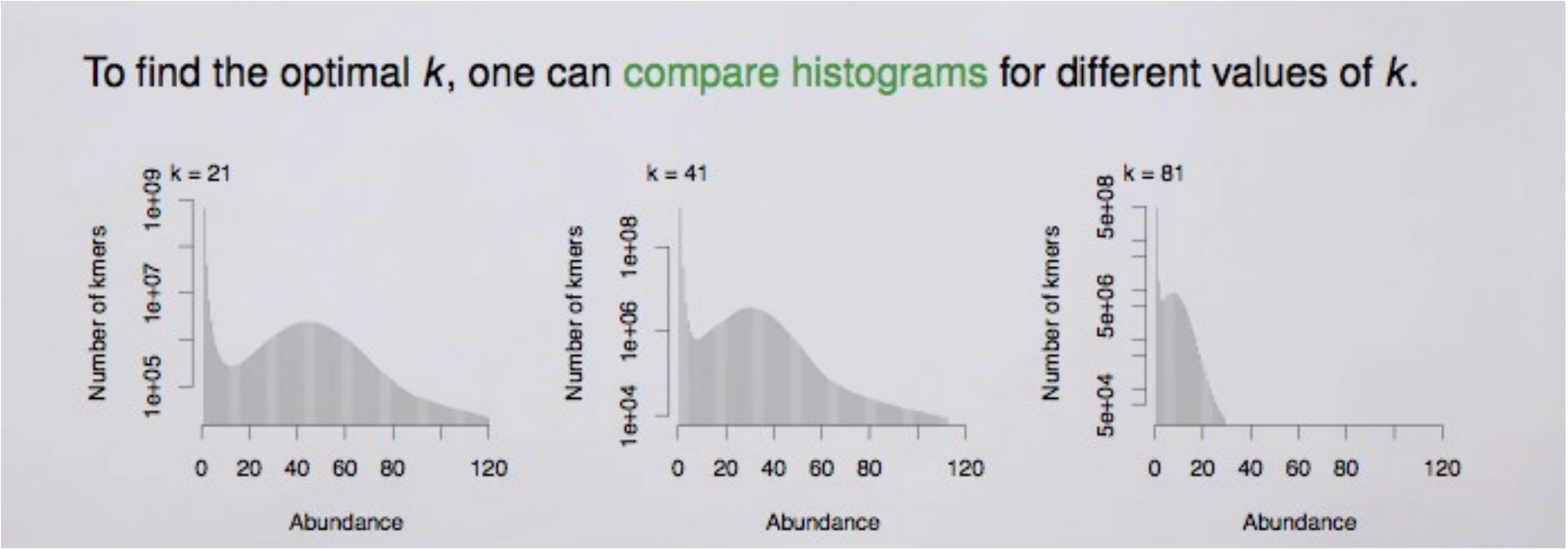


Now, read the k-mer histograms



Choose the k value which maximizes the assembly size

Now, read the k-mer histograms

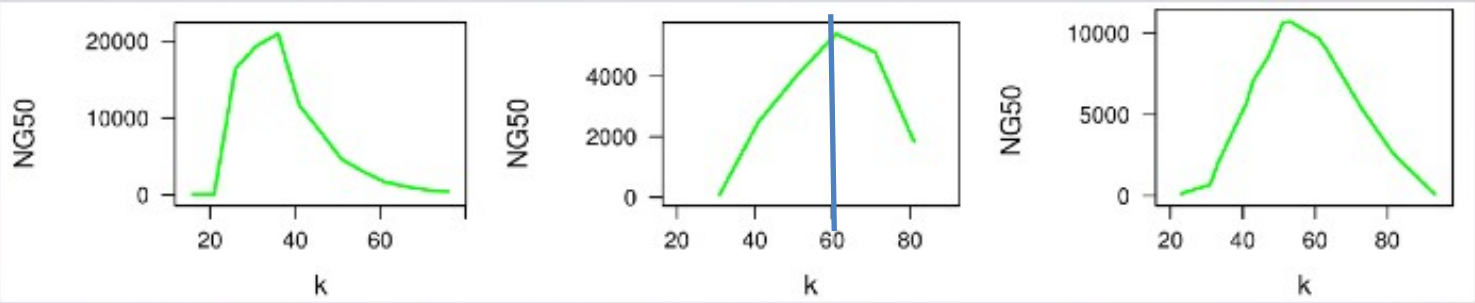


Computing a single histogram is time and memory expensive.
=> quickly estimates the histograms using sampling

The k-parameter

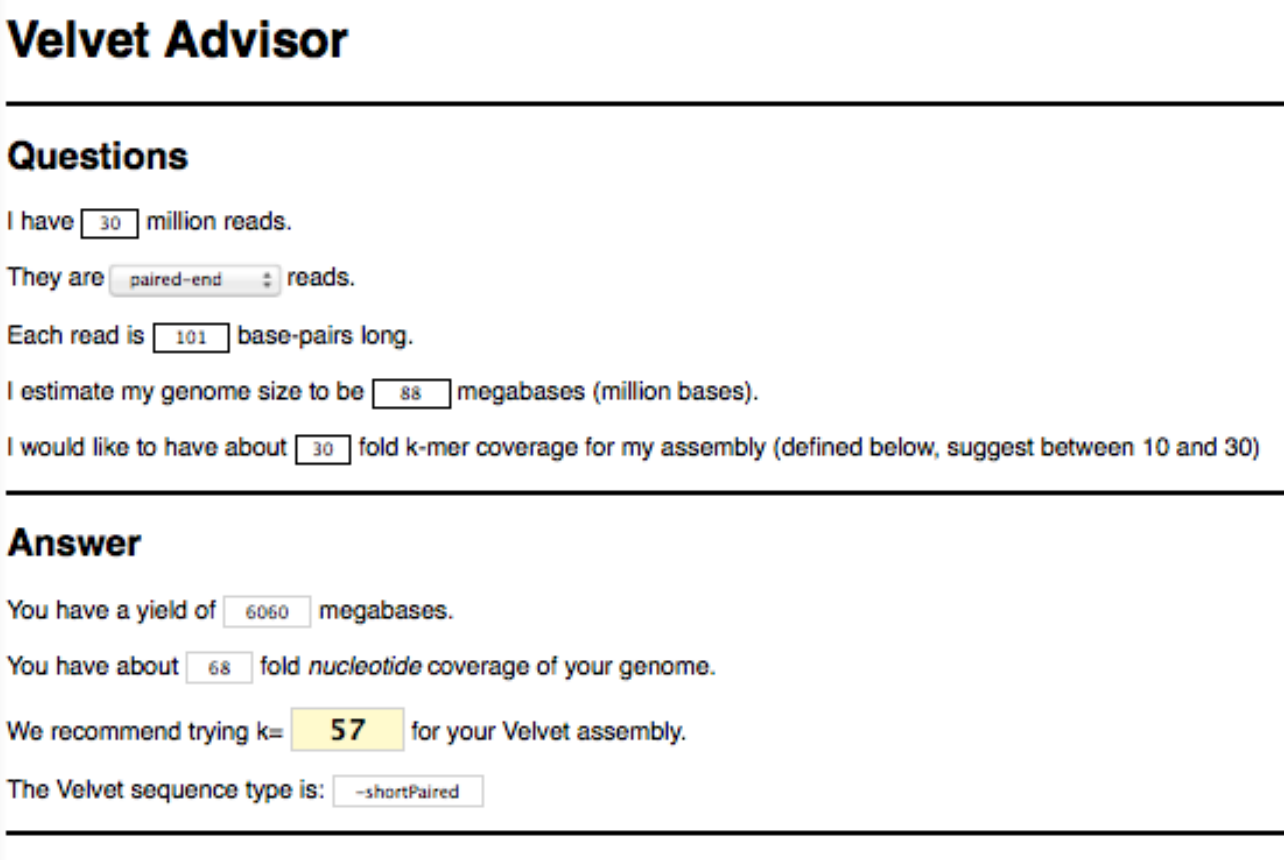
Assembly is not robust with respect to the parameter *k*. Because the ideal *k*-mer size depends on :

- sequencing coverage
- sequencing error rate
- genome complexity



k vs NG50 for 3 organisms : bacteria (S. aureus), human chr14, whole bumblebee genome (B. impatiens)

How to estimate the optimum k-mer size?



Velvet Advisor

Questions

I have 30 million reads.

They are paired-end reads.

Each read is 101 base-pairs long.

I estimate my genome size to be 88 megabases (million bases).

I would like to have about 30 fold k-mer coverage for my assembly (defined below, suggest between 10 and 30)

Answer

You have a yield of 6060 megabases.

You have about 68 fold *nucleotide* coverage of your genome.

We recommend trying k= **57** for your Velvet assembly.

The Velvet sequence type is: -shortPaired

http://dna.med.monash.edu.au/~torsten/velvet_advisor/

The main stats to estimate the quality of an assembly

- Number of contigs/scaffolds
- Total length of the assembly
- Length of the largest contig/scaffold
- Percentage of gaps in scaffolds ('N')
- N50/NG50 of contigs/scaffolds
- Number of predicted genes
- Number of core genes

The main stats to estimate the quality of an assembly

- Number of contigs/scaffolds
- Total length of the assembly
- Length of the largest contig/scaffold
- Percentage of gaps in scaffolds ('N')
- N50/NG50 of contigs/scaffolds
- Number of predicted genes
- Number of core genes

K-mer!!!!

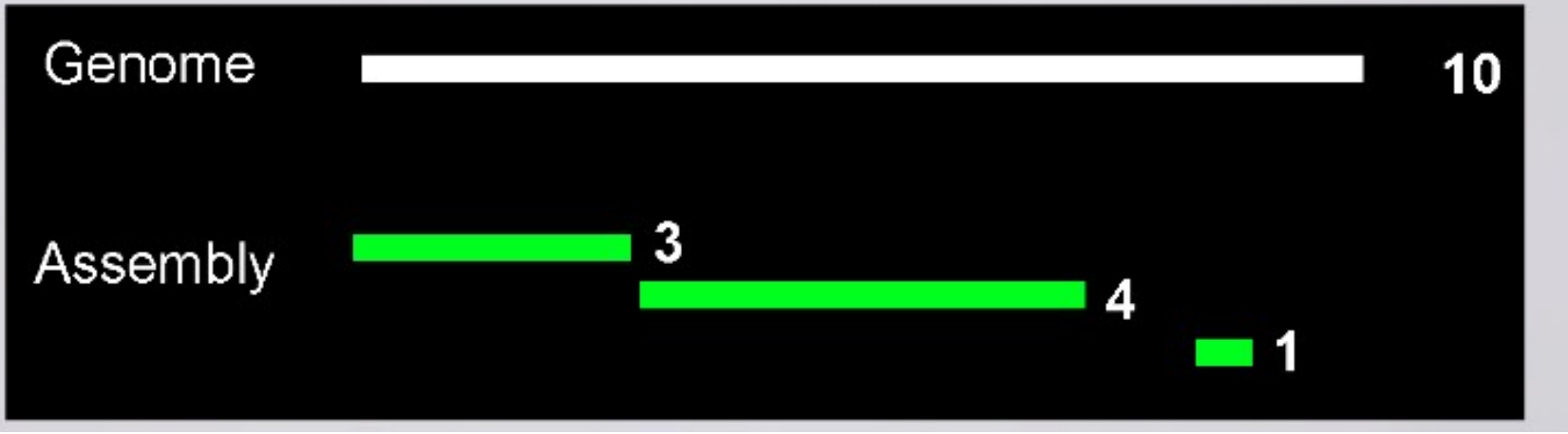
The main stats to estimate the quality of an assembly

- Number of contigs/scaffolds
- Total length of the assembly
- Length of the largest contig/scaffold
- Percentage of gaps in scaffolds ('N')
- N50/NG50 of contigs/scaffolds
- Number of predicted genes
- Number of core genes

N50/NG50

N50 = Largest contig length at which longer contigs cover 50% of the total **assembly** length

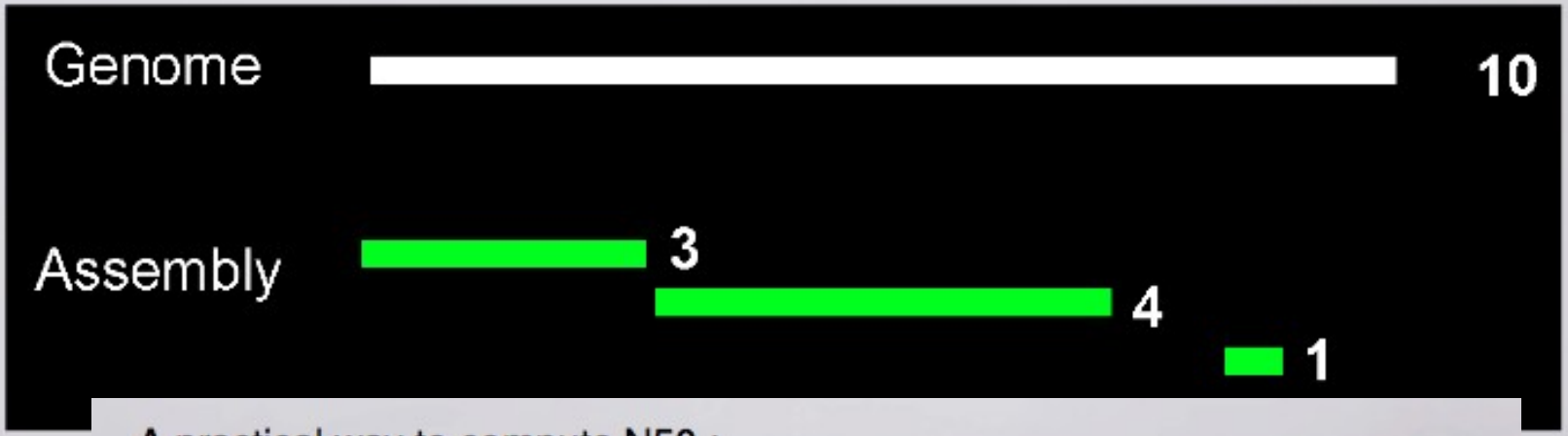
NG50 = Largest contig length at which longer contigs cover 50% of the total **genome** length



N50/NG50

N50 = Largest contig length at which longer contigs cover 50% of the total **assembly** length

NG50 = Largest contig length at which longer contigs cover 50% of the total **genome** length

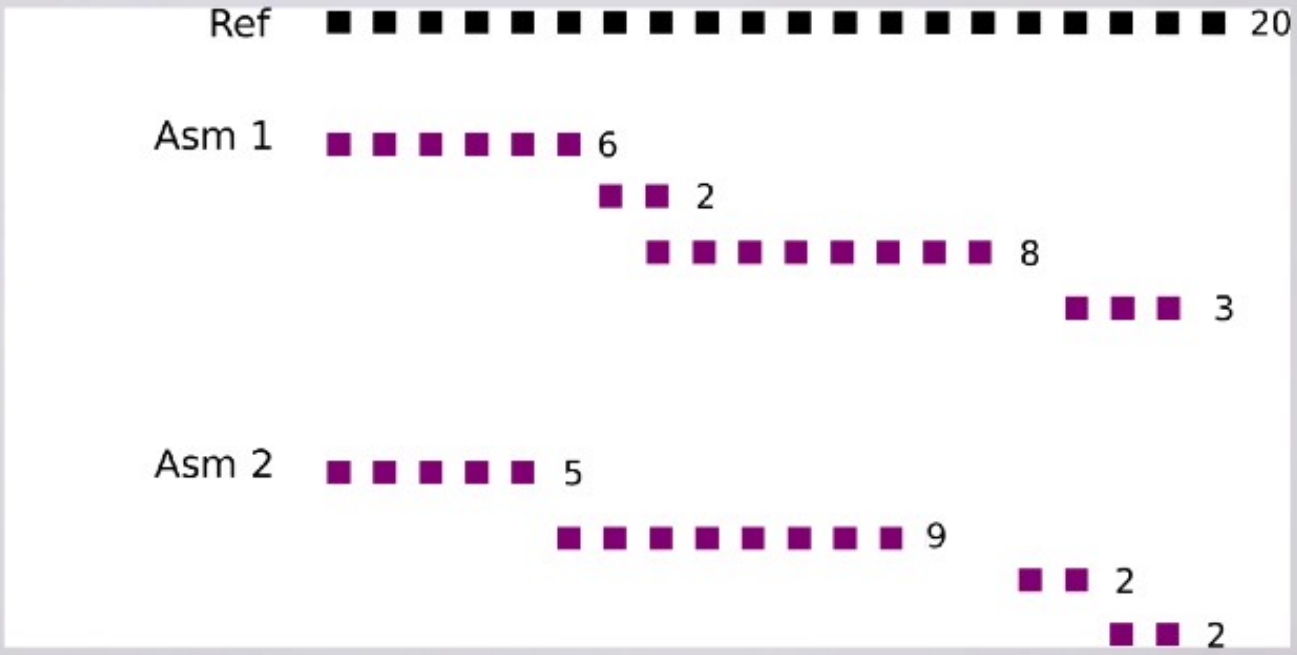


A practical way to compute N50 :

- Sort contigs by decreasing lengths
- Take the first contig (the largest) : does it cover 50% of the assembly ?
- If yes, this is the N50 value. Else, try the next one (the second largest), and so on..

Let's do it

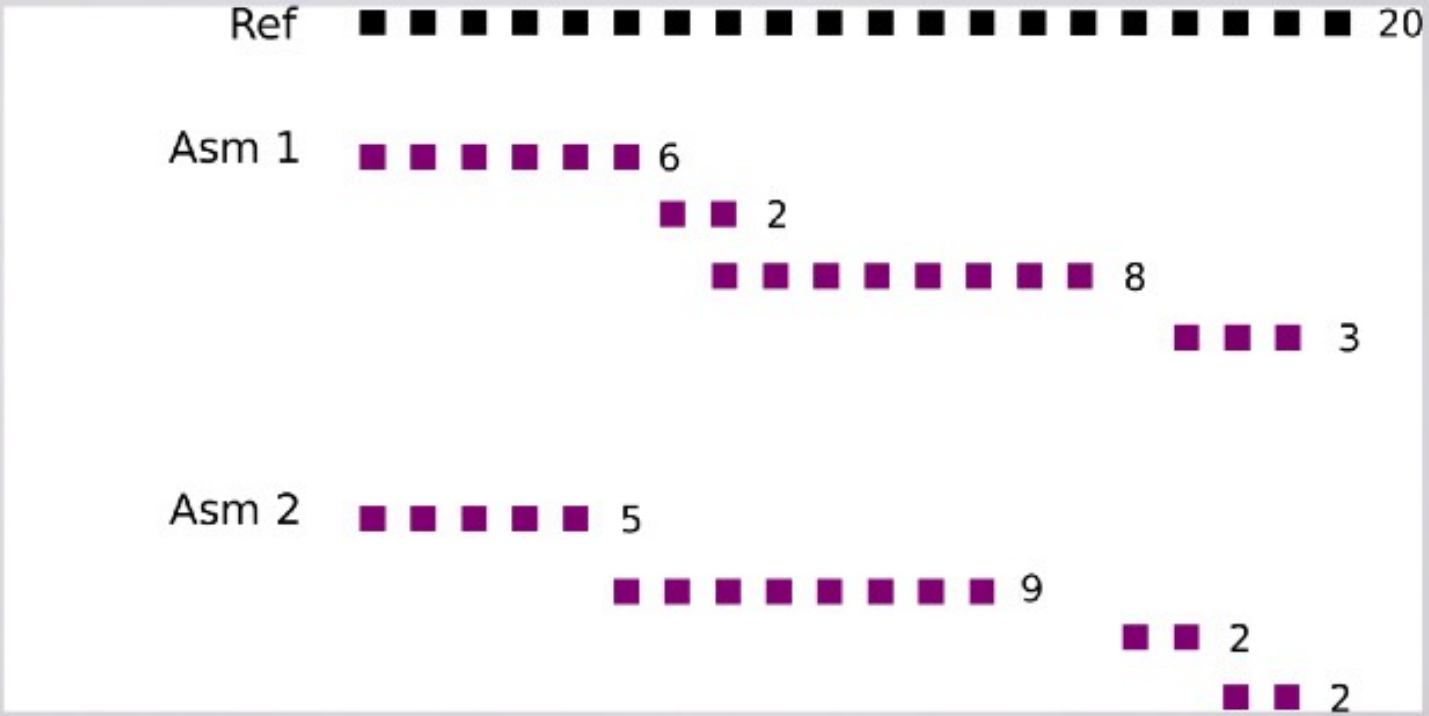
Here are two assemblies, aligned to the same reference :



- For each, compute the following metrics :
  - Total size of the assembly, N50, NG50 (bp)
  - Coverage (%)
- Which one is better than the other ?

Solution

Here are two assemblies, aligned to the same reference :



- For each, compute the following metrics :
  - Total size of the assembly (19 bp, 18 bp), N50 (6 bp, 9 bp), NG50 (6 bp, 5 bp)
  - Coverage (%) (90, 90)
- Which one is better than the other ? (I would say first one)

High N50 but bad assembly

"The standard of judging assembly quality by size of contigs is questionable. Large contigs may simply reflect overly aggressive joining of contigs, thereby creating larger contigs with mis-assemblies. As a consequence, genome scientists who are not experts at assembly can be completely misled by statistics about contig sizes, and as a result might prefer the 'larger' but incorrect assembly when given a choice."

Salzberg & Yorke, 2005

Assembly score

N_{50} is the N50 statistic of the assembly

G_S is the number of contigs contained in the assembly

M_S is the mean assembly contig length

then $G_S M_S := L_S$ is the estimated size of the assembled genome

L_E is the expected (actual) genome size

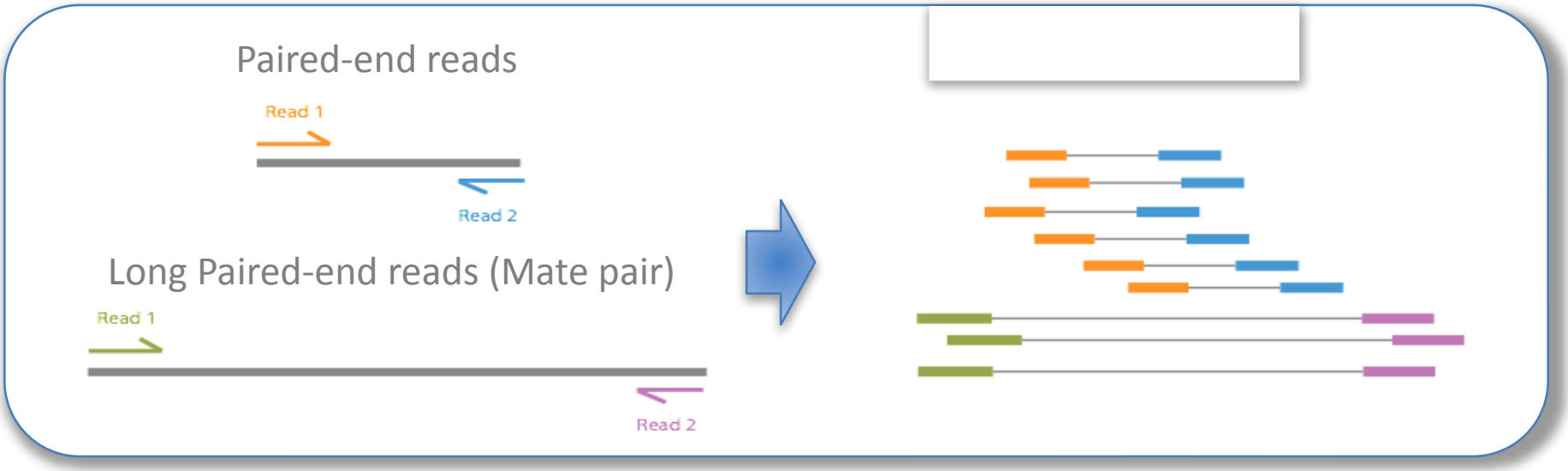
then $|L_E - L_S|$ is the error between the expected size of the (actual) genome and the estimated size of the assembled genome, and in non-pathological cases, $|L_S - (L_E + 1)| > 0$

$\mathcal{A}$ is the Assembly Score

$$\mathcal{A} = \frac{N_{50}}{G_S|L_S - (L_E + 1)|}\left(\frac{1}{10}\right)$$

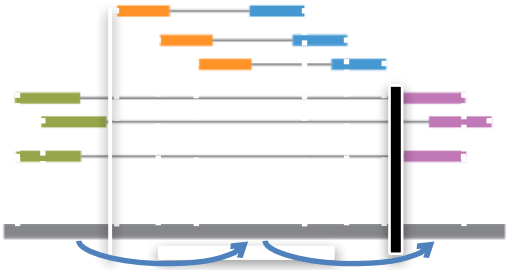
Note that the $\frac{1}{10}$ multiplier is only added for ease of interpretation of the scores. It has no biological meaning.

Mapping *vs.* Assembly



No Reference genome

De novo **Assembly**



Reference genome

Mapping



Mapping

- https://liorpachter.wordpress.com/2015/11/01/what-is-a-read-mapping/



Bits of DNA

Reviews and commentary on computational biology by Lior Pachter

Subscribe to feed Home *Seq About

Search

RECENT COMMENTS

Joseph Russell on Straining metagenomics



Debora Marks (@debor... on Something the NIH can learn fr...

Nicolas Bray on Near-optimal RNA-Seq quantific...

What is a read mapping?

November 1, 2015 in expository | Tags: lightweight alignment, pseudoalignment, quasi-mapping, read alignment, read mapping

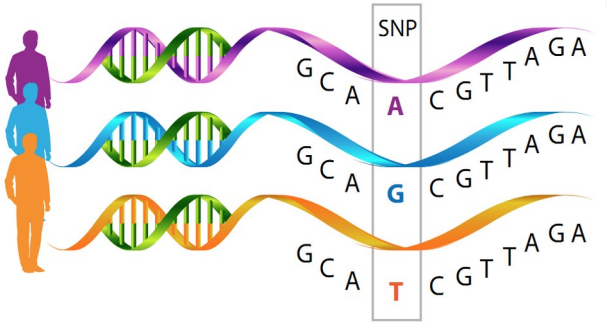
In the 1930s the pseudonym Nicolas Bourbaki was adopted by a group of French mathematicians who sought to axiomatize and formalize mathematics in a rigorous manner. Their project greatly influenced mathematics in the 20th century and contributed to its present "definition -> theorem -> proof" format (Bourbaki went a bit overboard with abstraction and generality and eventually the movement petered out).

In biology definitions seem to be hardly worth the trouble. For example, the botanist Willhelm Johannsen coined the term "gene" in 1909 but there is still no universal agreement on a "definition". How could there be? Words associated

Mapping algorithms

Common algorithms for mapping short reads to a reference genome

Software	**Technique**	**Developer**	**License**
Eland	Hashing reads	Illumnia	?
SOAP	Hashing refs	BGI	Academic
Maq	Hashing reads	Sanger (Li, Heng)	GNUPL
Bowtie	BWT	Salzberg/UMD	GNUPL
BWA	BWT	Sanger (Li, Heng)	GNUPL
SOAP2	BWT & hashing	BGI	Academic



Stuctural variants

Genetic Variation Among People

0.1% difference among people



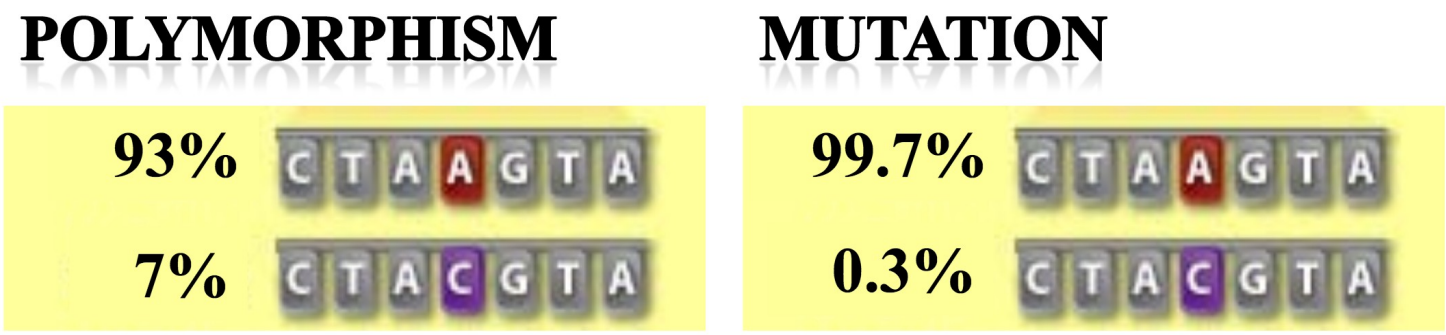



What we define as variations

- **Single Nucleotide Polymorphism (SNP) including Short indel (insertion/deletion; < 10bp)**

- **Structural variation**
  - Inversion
  - Translocation
  - Large indel (> 10 bp)
  - Copy number variation (*i.e.*, DNA repeat)
    - Transposable element (also called Mobile DNA)
    - Interspersed duplication
    - Tandem duplication

SNP *vs.* mutation



Polymorphism: Single DNA base change found in >1% of population

Mutation: Single DNA base change found in <1% of population

Genetic mutations are one type of genetic polymorphism

Read Depth *vs* coverage

```
Read 1: GAATCCGATCGATCATATCAGTCGATCG
Read 2:       GATCGATCATATCAGTCGAcCGATCGATCAT
Read 3:                 ATCAGTCGATCGATCGATCATCGACTAC
Read 4:                           CGATCGATCGATCATCGACTACGTA
Depth:  11111122222222223333333333344333333333222222222221111
```

Coverage Equation

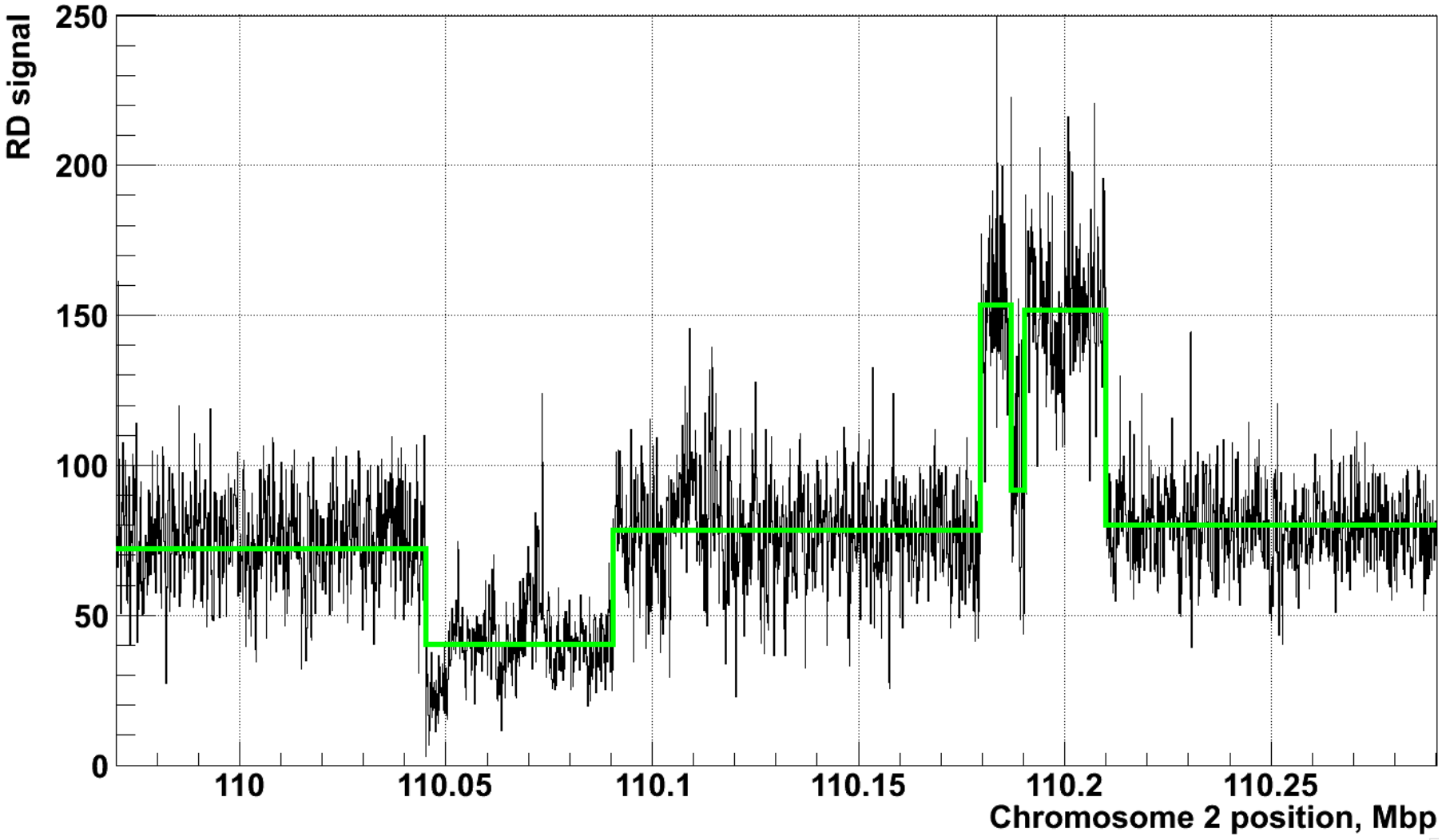
The Lander/Waterman equation is a method for computing coverage[1].
The general equation is:

$$C = LN / G$$

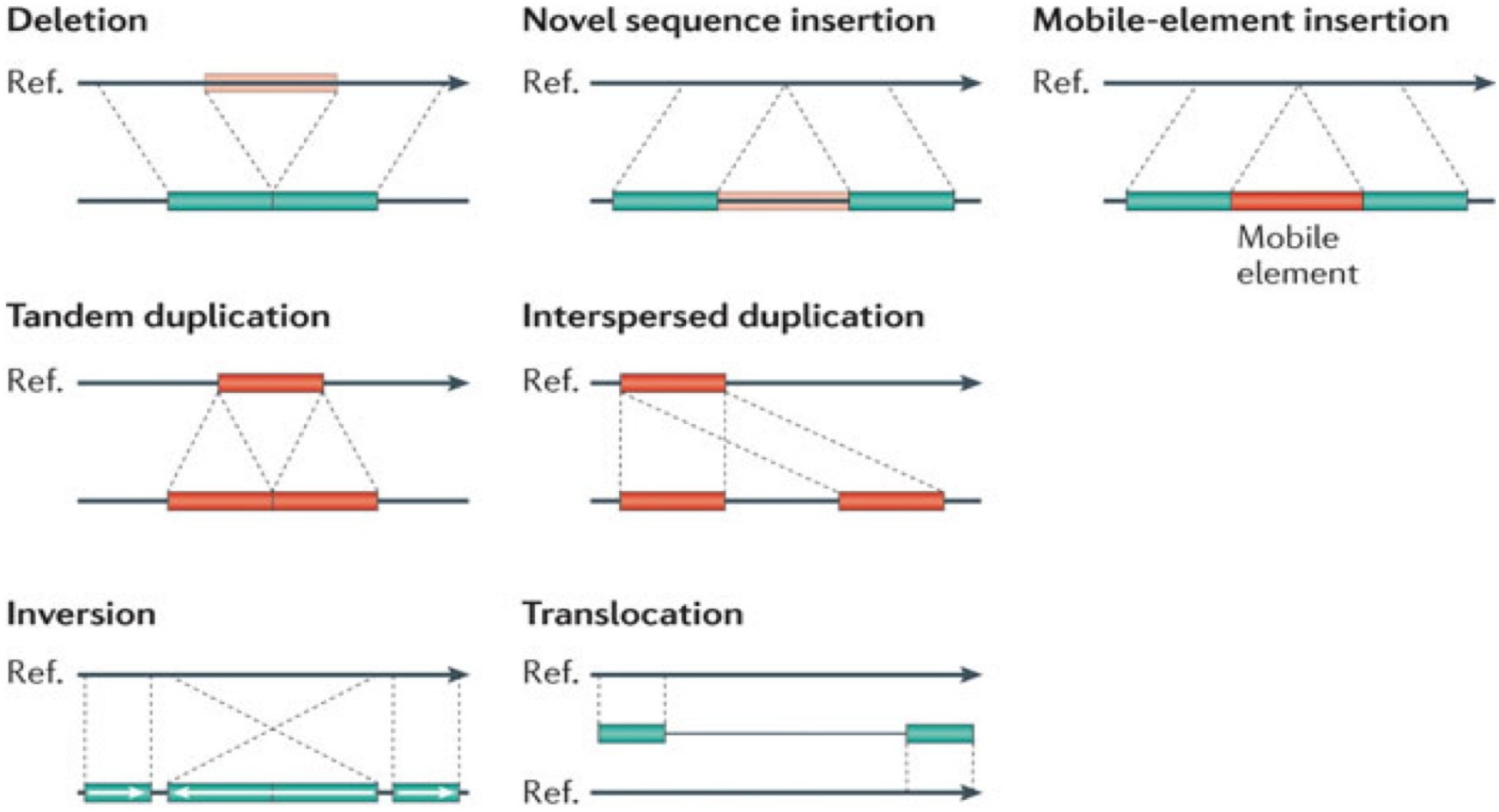
- **C stands for coverage**
- **G is the haploid genome length**
- **L is the read length**
- **N is the number of reads**

http://support.illumina.com/downloads/sequencing_coverage_calculator.html

Read Depth coverage

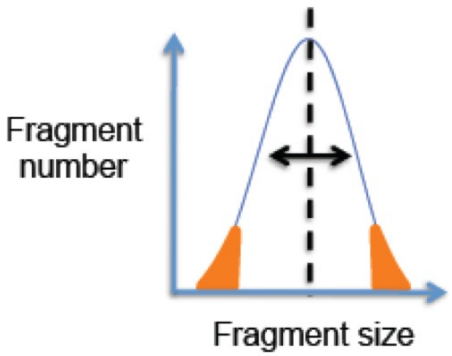


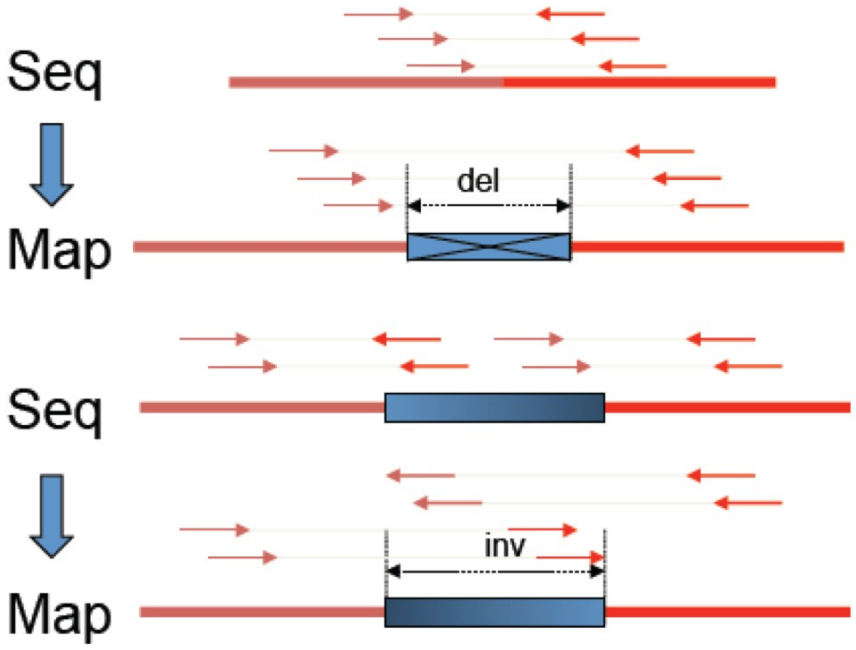
SV are always described as a structural changes with regards to the reference sequences



Nature Reviews | Genetics

SV detection using paired-end

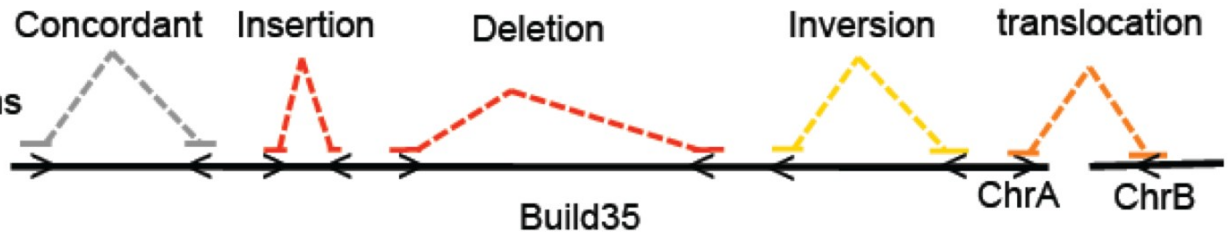




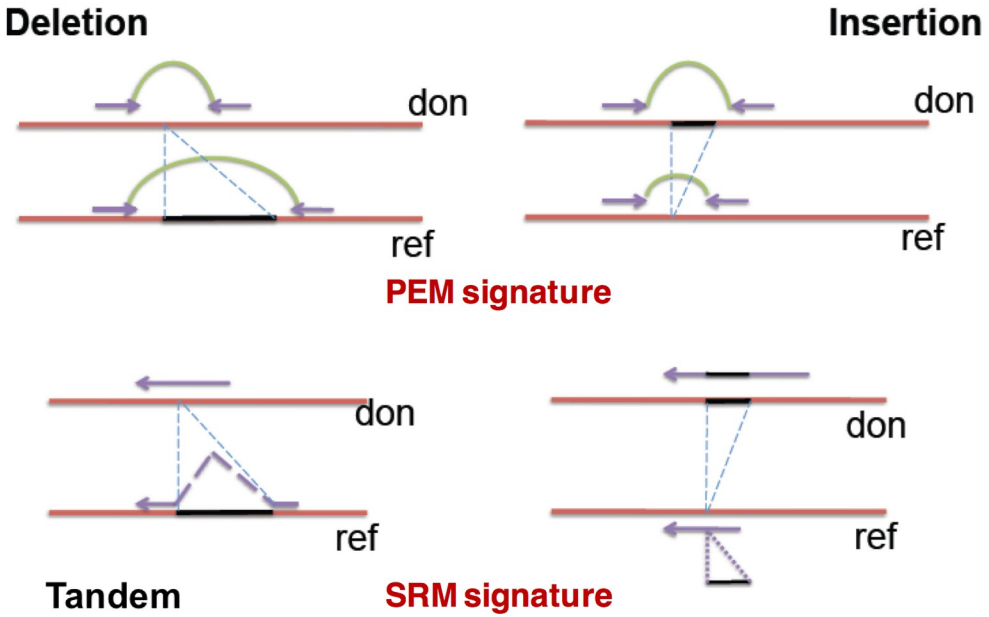
Mapping of read pairs to reference

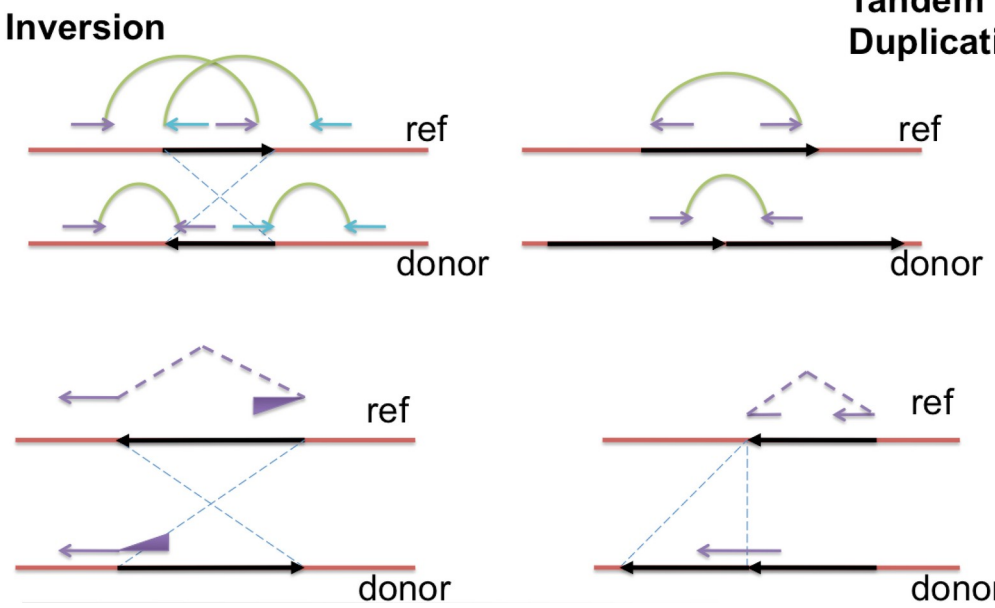
- Spanning unexpected distance
- Unexpected orientation

•Detection of:
- •Deletions
- •Insertions
- •Translocations
- •Inversions



Impact of slip-read mapping (SRM)





RD *vs* paired-end

- Difficulty in finding highly repetitive CNVs (LINE, SINE, etc.)
- Uncertain in CNV location
- **Uses mutual information of both ends, better mapping and ascertainment in homologous region**
- **Ascertains complex regions**
- **Can find large insertions**
- Can be used with paired-end, single-end and mixed data

- Can detect highly repetitive CNVs (LINE, SINE, etc.)
- Defines precise location of CNV
- **Relies on confident independent mapping of each end, problems in regions flanked by repeats**
- Small span between ends limits resolution of complex regions
- Large span between ends limits resolution of break points

Higher resolution with slip-reads