



2013 - BMMB 597D: Analyzing Next Generation Sequencing Data

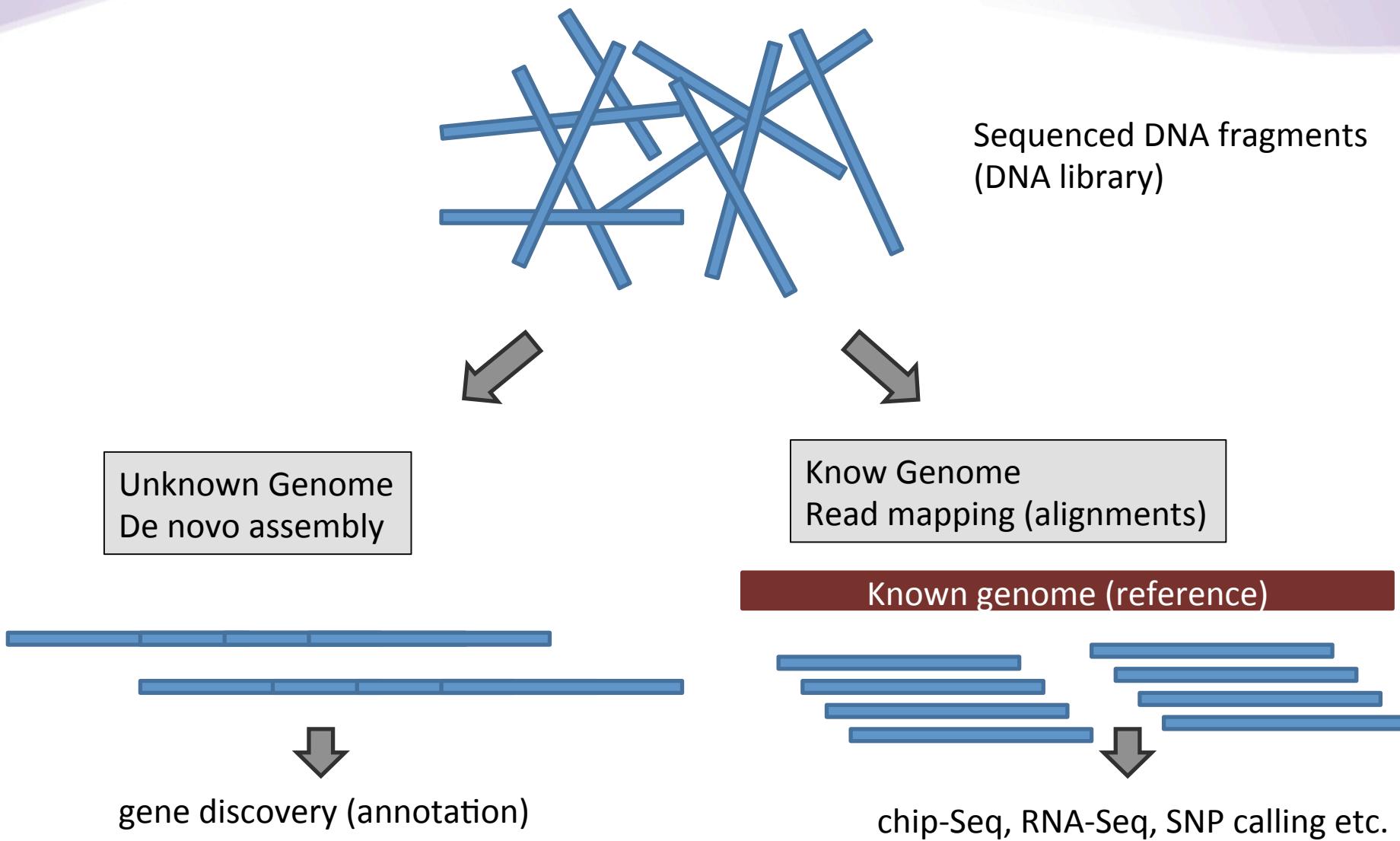
# Week 7, Lecture 14

István Albert

Biochemistry and Molecular Biology  
and Bioinformatics Consulting Center

Penn State

# Processing sequencing reads



# Sequence alignments

- Arranging two or more sequences such as to maximize the length of the common regions between the two
- It is a very well developed field – the roots of the bioinformatics started with various alignment software
- We will only cover pair-wise alignments searching a database with a query
- High throughput sequencing poses special constraints: a **very large number** of very short reads - traditional methods were not feasible → **semi-global alignment**

# Alignment concepts

indels

perfect match

GCAAG

one mismatch

GCAAG

insertion vs ref.

GCAAG

deletion vs ref

GCAA-G

AGCAAGTATGTAAGGGCGCAGAAAAGCAAAG

**NOTE: mismatches or indels can be longer than 1 base!**

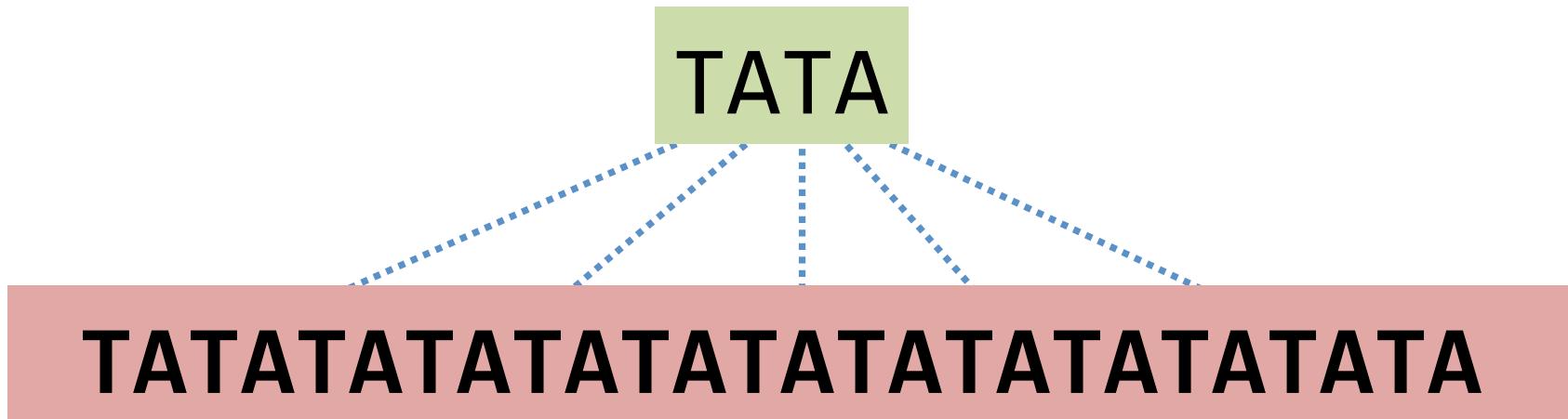
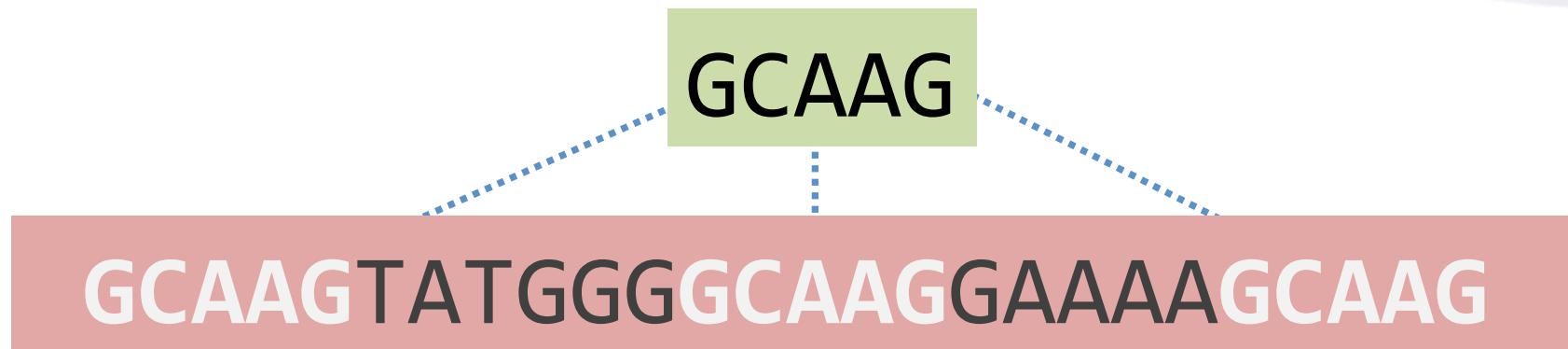
**It gets complicated very quickly**

Alignment scoring depends on mismatch scoring (different across bases!),  
gap open, gap extension penalties

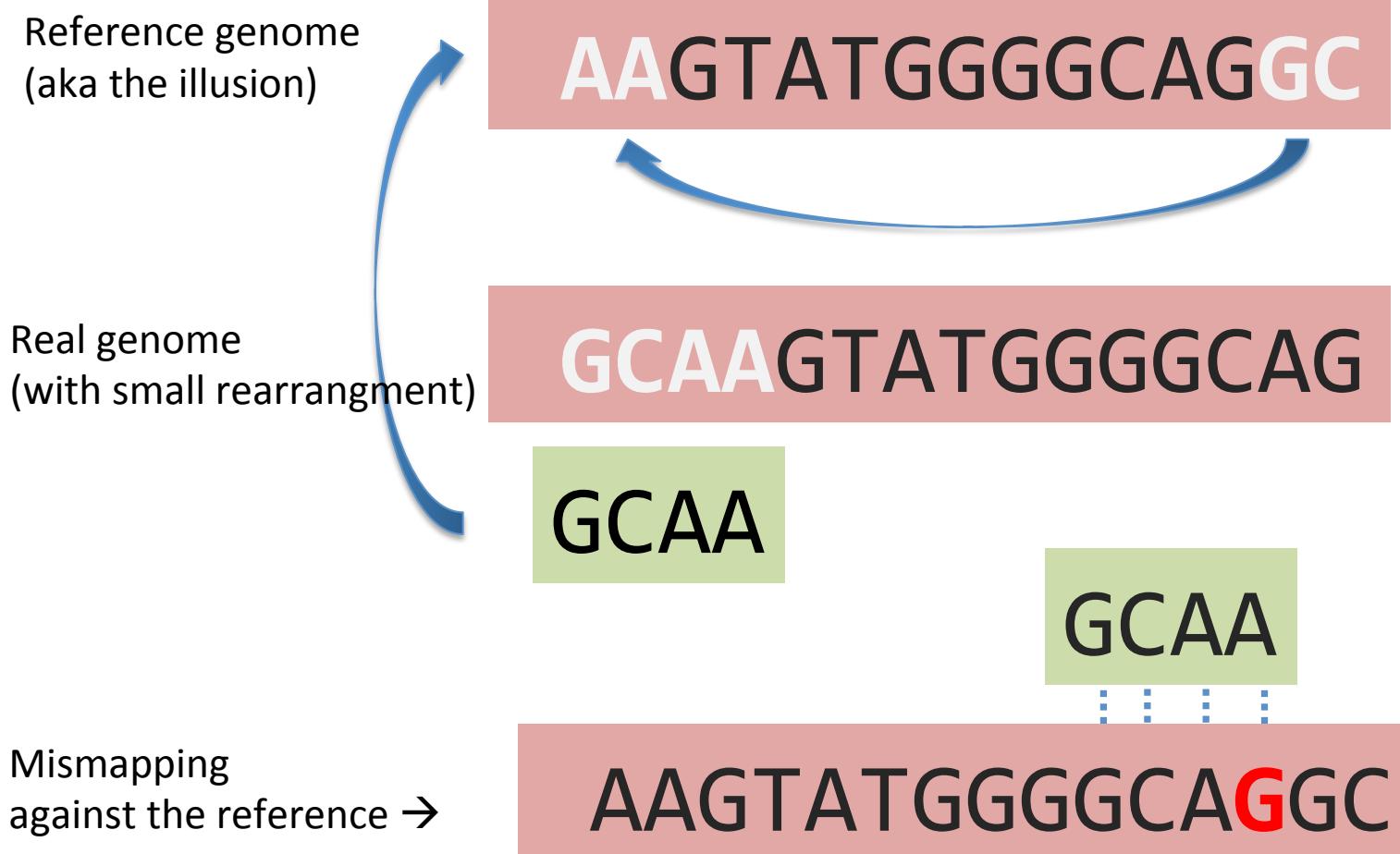
# Use the terminology correctly

- A mismatch is not a SNP (single nucleotide polymorphisms)
- But SNPs will show up as mismatches
- indels are not DIPs (insertion/deletion polymorphisms)
- But DIPs will show up as indels

# Challenges: repetitive and low complexity regions



# Challenge: genomic reorganization can easily lead to mis-mapping



# Short Read Mappers (aligners)

- Use **heuristics** to quickly identify locations (hits) where the reads match
- Heuristics → not all hits will be found!
- Tradeoff: **resource usage vs speed vs accuracy vs usability**
- Each domain of application may have more appropriate tools

## A few popular aligners

- BWA, bowtie, SOAP2, Shrimp, BFAST and many others

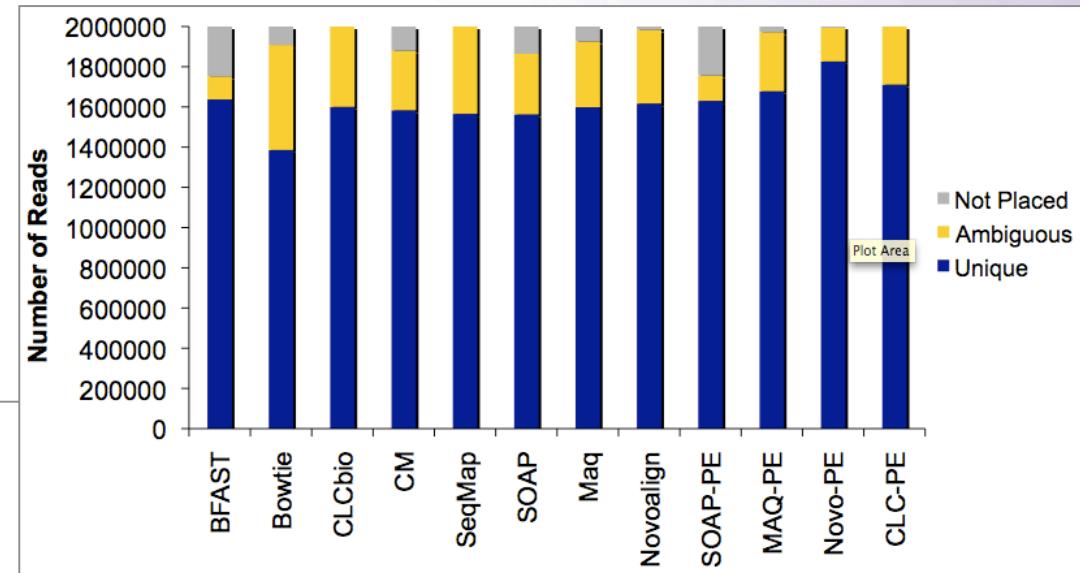
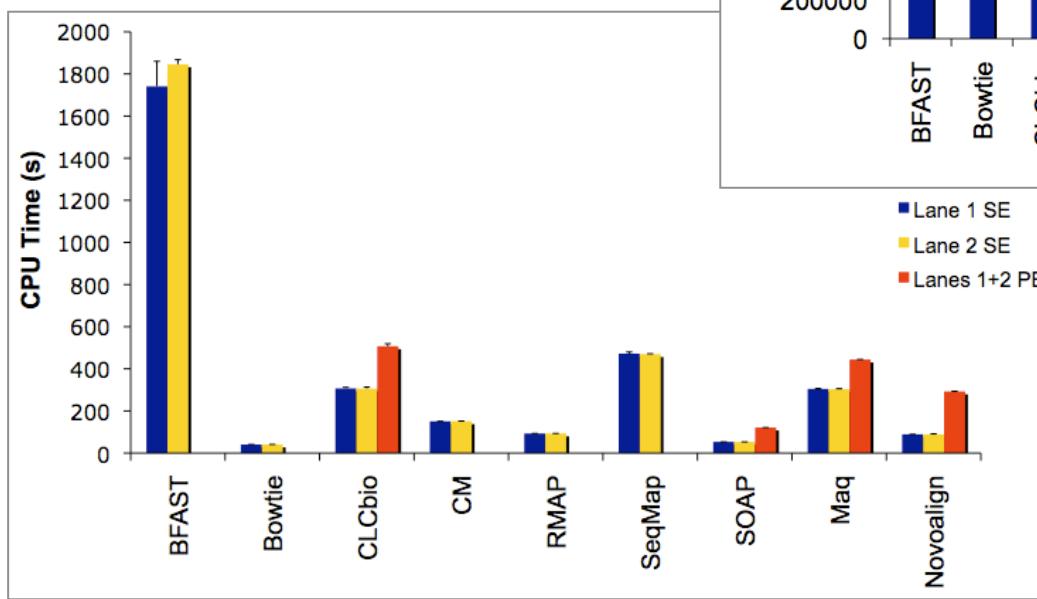
There is no **single best tool**, the issues to consider:

1. **documentation** → can we figure out how it works
2. **input features** → what type of input can it handle
3. **reporting features** → will it produce the type of output that we can use
4. **performance** → is it feasible to run on my resources

# Aligner features

- Some cannot handle **indels** (insertion/deletions) → these tools are typically extremely fast will run on a laptop for even large genomes (**bowtie 1**) → application Chip-Seq
- May or may not use the quality score during the alignment (usually not, that would alter the alignment score)
- Differences in the they report alignments:
  - unique alignments only (note that this is an ill defined concept!)
  - best alignments above a cutoff
  - all possible alignments (can slow down the process greatly)

# CPU time to align 1 million reads



from: <http://www.massgenomics.org/short-read-aligners>

# Tool comparisons

The vast majority of tools comparisons lead to “red-herring” type fallacies

1. Tool A is faster than Tool B and we tacitly assume that the results are the same
2. Tool A is more ‘accurate’ than Tool B and we assume the resource utilization to be the same

# Robust results should be tool independent!

- We need tools that are correct and finish running within our resources
- Results should be hold across methods
- There are exception to this – but those need to be explicitly documented – and you need to know why

# BWA (Burrows-Wheeler Aligner)

## by Heng Li

### Burrows - Wheeler Aligner

[Home](#)

#### Introduction

Burrows-Wheeler Aligner (BWA) is an efficient program that aligns relatively short nucleotide sequences against a long reference sequence such as the human genome. It implements two algorithms, bwa-short and BWA-SW. The former works for query sequences shorter than 200bp and the latter for longer sequences up to around 100kbp. Both algorithms do gapped alignment. They are usually more accurate and faster on queries with low error rates. Please see the [BWA manual page](#) for more information.

#### BWA:

[SF project page](#)  
[SF download page](#)  
[Mailing list](#)  
[BWA manual page](#)  
[Repository](#)

#### Links:

<http://bio-bwa.sourceforge.net/>

Download, unpack, compile with make and link to bin

Read also the **bwa-mem** controversy – the **bwa-mem** paper rejection

# Download – unpack and make

```
ialbert@porthos ~/src
$ tar jxf bwa-0.7.5a.tar.bz2

ialbert@porthos ~/src
$ cd bwa-0.7.5a

ialbert@porthos ~/src/bwa-0.7.5a
$ make
```

Uses **bz2** compression hence the **j** flag (the **z** command would expand a gzip file)

# Link it the tool into your bin folder

```
ialbert@porthos ~/src/bwa-0.7.5a
$ ln -s ~/src/bwa-0.7.5a/bwa ~/bin/bwa

ialbert@porthos ~/src/bwa-0.7.5a
$ ~/bin/bwa
```

# Aligning reads with BWA (old-style)

For BWA is three step process (this is different for each aligner):

- 1. Index the genome** – this only needs to be done once for each genome → **bwa index ...**
- 2. Create the alignment** → **bwa aln ...**
- 3. Format the alignment** → **bwa samse ...**

# Aligning reads with BWA (new-style)

For BWA is three step process (this is different for each aligner):

- 1. Index the genome** – this only needs to be done once for each genome → **bwa index ...**
- 2. Create the alignment** → **bwa mem ...**

# There is built-in help

```
ialbert@porthos ~/work/lec10
$ ~/bin/bwa index

Usage: bwa index [-a bwtsw|is] [-c] <in.fasta>

Options: -a STR      BWT construction algorithm: bwtsw or is [auto]
          -p STR      prefix of the index [same as fasta name]
          -6          index files named as <in.fasta>.64.* instead of <in.fa
sta>.*

Warning: `'-a bwtsw' does not work for short genomes, while `'-a is' and
`'-a div' do not work not for long genomes. Please choose `'-a'
according to the length of the genome.

ialbert@porthos ~/work/lec10
$
```

# Index the genome



```
ialbert@porthos ~
$ ~/bin/bwa index ~/refs/yeast/sc.fa
```

For long genomes (over 2GB) you need to index with the **-a bwtsw** option.

Periodically re-read the manual to remind yourself of what the tool can do.

# Create a FASTQ query sequence

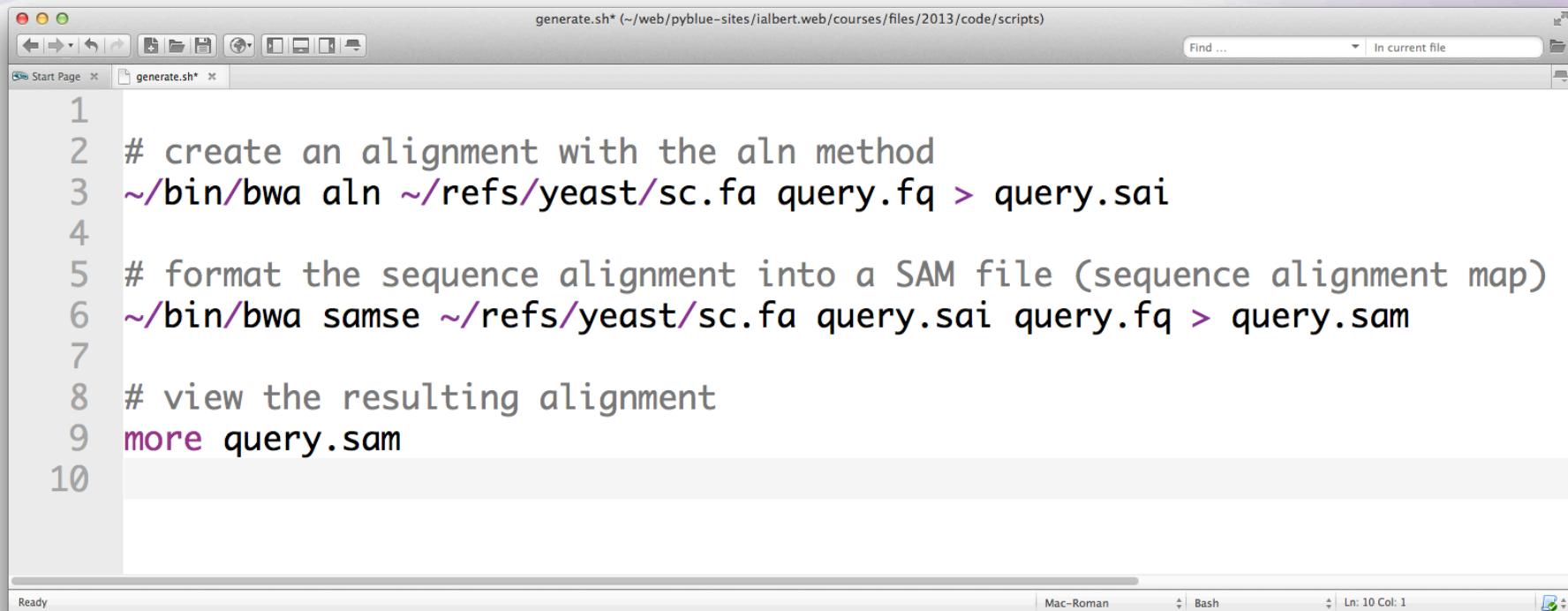
- Select some bases from the beginning of chromosome 1

This works because quality strings could be letters up to I anyhow

# BWA concepts

- the **index or prefix (\*.fasta)**: the full path to the fasta reference sequence that has been indexed
- the **query (\*.fq)**: the file containing the sequences that we want to look up in the reference
- the **alignment (\*.sai)**: a binary intermediate file that contains the alignment information for each read
- the **sequence alignment map (\*.sam)**: the result file that contains the alignment cross referenced with the query
- **bwa** can operate with different alignment methods! These may generate very different outputs!

# Run and format with BWA aln



A screenshot of a Mac OS X terminal window titled "generate.sh\* (~/web/pyblue-sites/ialbert.web/courses/files/2013/code/scripts)". The window contains the following shell script:

```
1 # create an alignment with the aln method
2 ~/bin/bwa aln ~/refs/yeast/sc.fa query.fq > query.sai
3
4 # format the sequence alignment into a SAM file (sequence alignment map)
5 ~/bin/bwa samse ~/refs/yeast/sc.fa query.sai query.fq > query.sam
6
7 # view the resulting alignment
8 more query.sam
9
10
```

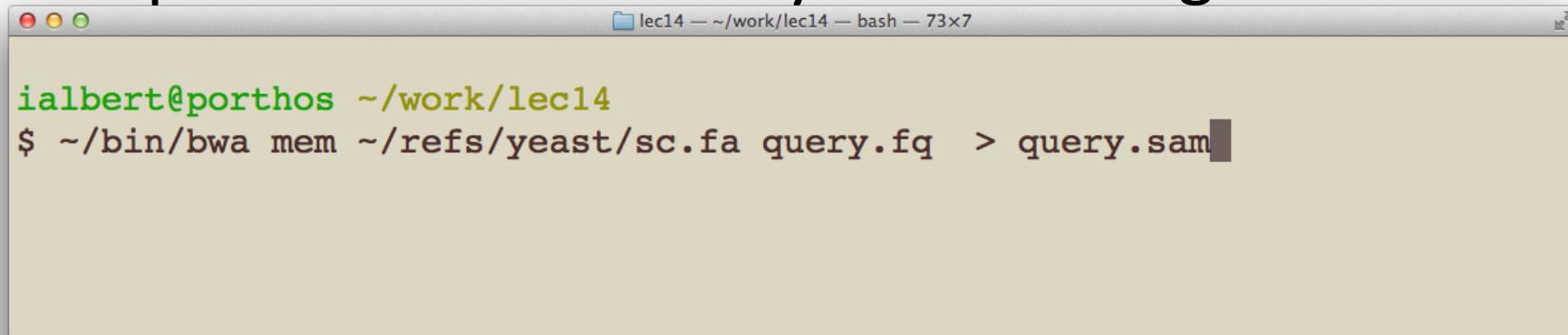
The terminal interface includes standard Mac OS X window controls (red, yellow, green buttons), a menu bar, and a status bar at the bottom indicating "Ready", "Mac-Roman", "Bash", and "Ln: 10 Col: 1".

A multistep process that is best used via a shell script.

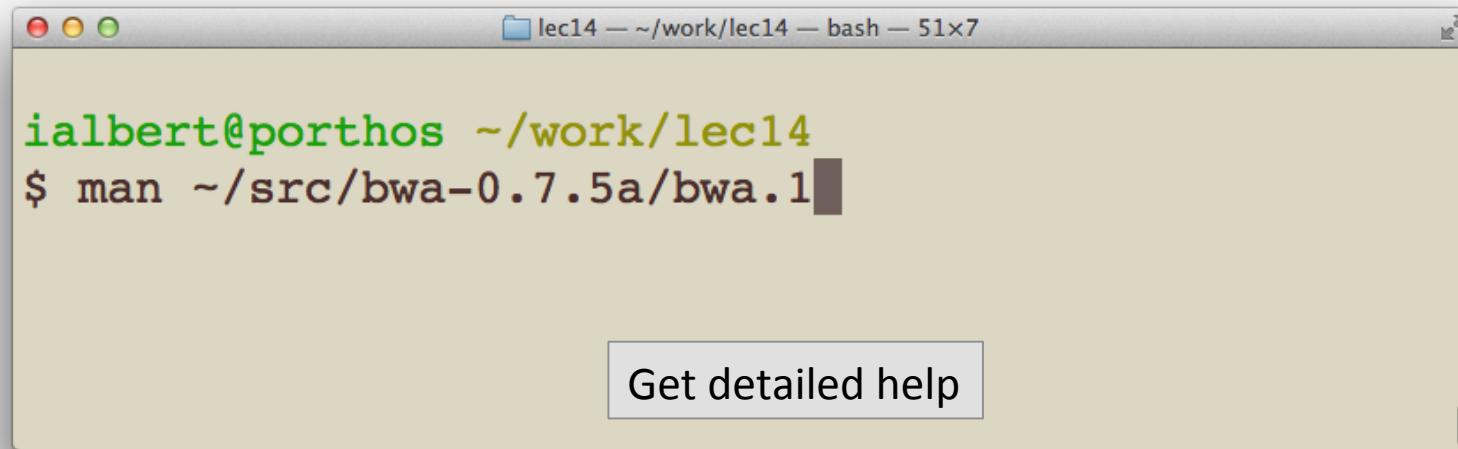
The file **query.sam** is the final output of the alignment

# Use the bwa mem method

- This is a different alignment strategy that operates via a radically different algorithm



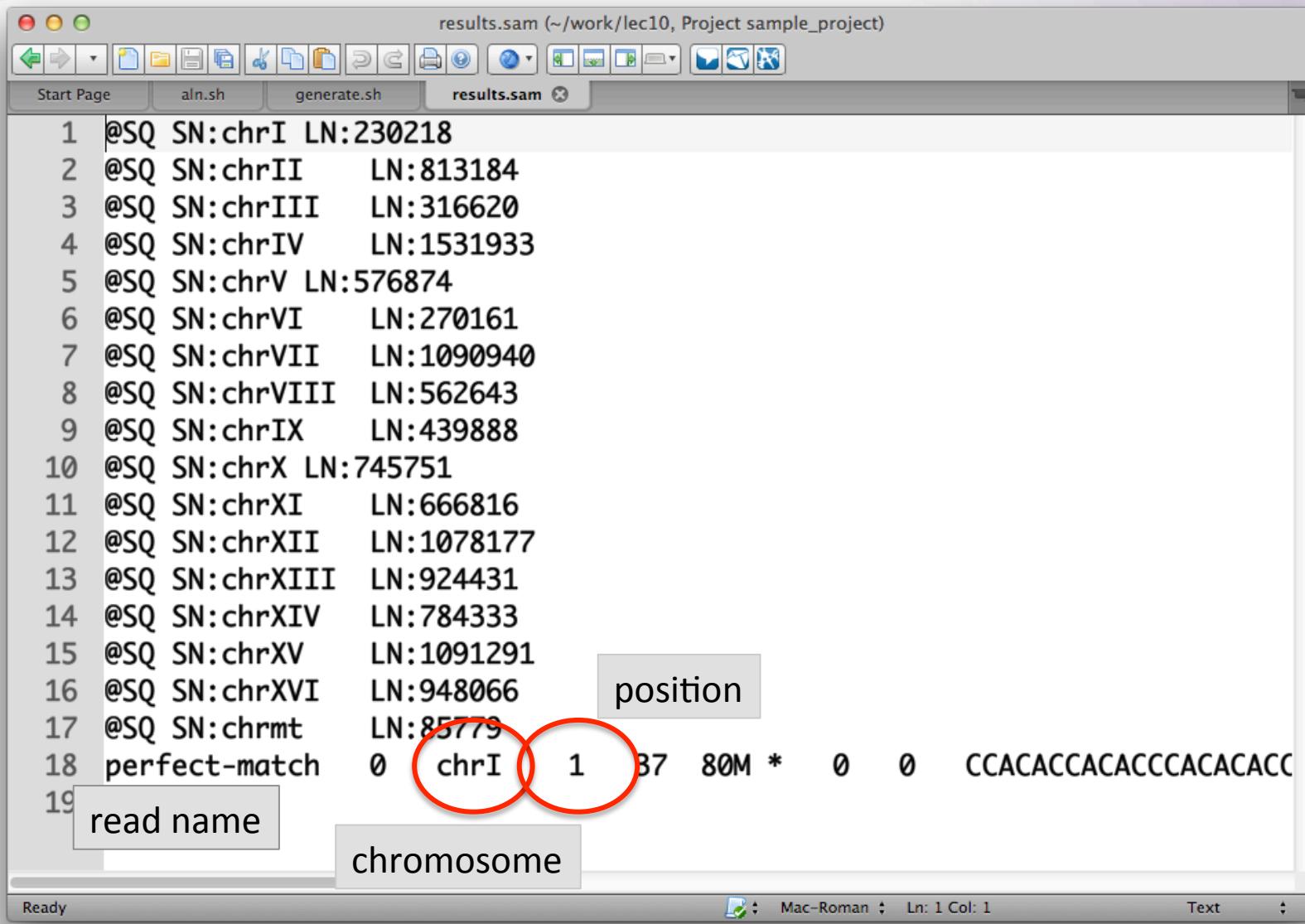
A screenshot of a Mac OS X terminal window titled "lec14 — ~/work/lec14 — bash — 73x7". The window shows a command being typed: `ialbert@porthos ~/work/lec14` followed by `$ ~/bin/bwa mem ~/refs/yeast/sc.fa query.fq > query.sam`. The cursor is at the end of the command line.



A screenshot of a Mac OS X terminal window titled "lec14 — ~/work/lec14 — bash — 51x7". The window shows a command being typed: `ialbert@porthos ~/work/lec14` followed by `$ man ~/src/bwa-0.7.5a/bwa.1`. The cursor is at the end of the command line.

[Get detailed help](#)

# The result of the alignments



```
results.sam (~/work/lec10, Project sample_project)
Start Page aln.sh generate.sh results.sam
1 @SQ SN:chrI LN:230218
2 @SQ SN:chrII LN:813184
3 @SQ SN:chrIII LN:316620
4 @SQ SN:chrIV LN:1531933
5 @SQ SN:chrV LN:576874
6 @SQ SN:chrVI LN:270161
7 @SQ SN:chrVII LN:1090940
8 @SQ SN:chrVIII LN:562643
9 @SQ SN:chrIX LN:439888
10 @SQ SN:chrX LN:745751
11 @SQ SN:chrXI LN:666816
12 @SQ SN:chrXII LN:1078177
13 @SQ SN:chrXIII LN:924431
14 @SQ SN:chrXIV LN:784333
15 @SQ SN:chrXV LN:1091291
16 @SQ SN:chrXVI LN:948066
17 @SQ SN:chrmt LN:85779
18 perfect-match 0 chrI 1 37 80M * 0 0 CCACACCACACCCACACACC
19 read name chromosome position
```

read name

chromosome

position

Ready Mac-Roman Ln: 1 Col: 1 Text

# Homework 14

Create a FASTQ file using the first 60 or so bases from chromosome 1 of the yeast genome.

Align your query file with both BWA and BLAST and look at the output of both processes.

Both are sequence aligners, what is the main difference that you observe?

(remember to turn your fastq file into fasta when blasting)