Interval Data Types

István Albert

Bioinformatics Consulting Center

Penn State

First: get a good text editor!

Desired features:

1. syntax highlighting
2. line numbering
3. ability to view **white** space

Recommendation **Komodo Edit –** cross platform editor (download the Editor not IDE)

Countless other options: Sublime Text, TextMate,

Biological file formats

Each file format represents

1. **Information** – type of “knowledge” that is stored in the file

2. **Optimization** – the types of operations that are easy and efficient to perform

The above implies that some information may not be present or it cannot be easily extracted from a certain file format.

Successive Transformation

- Data analysis is not about the tools!
- It is about performing a succession of transformations on the data
- You will need to understand what each format contains

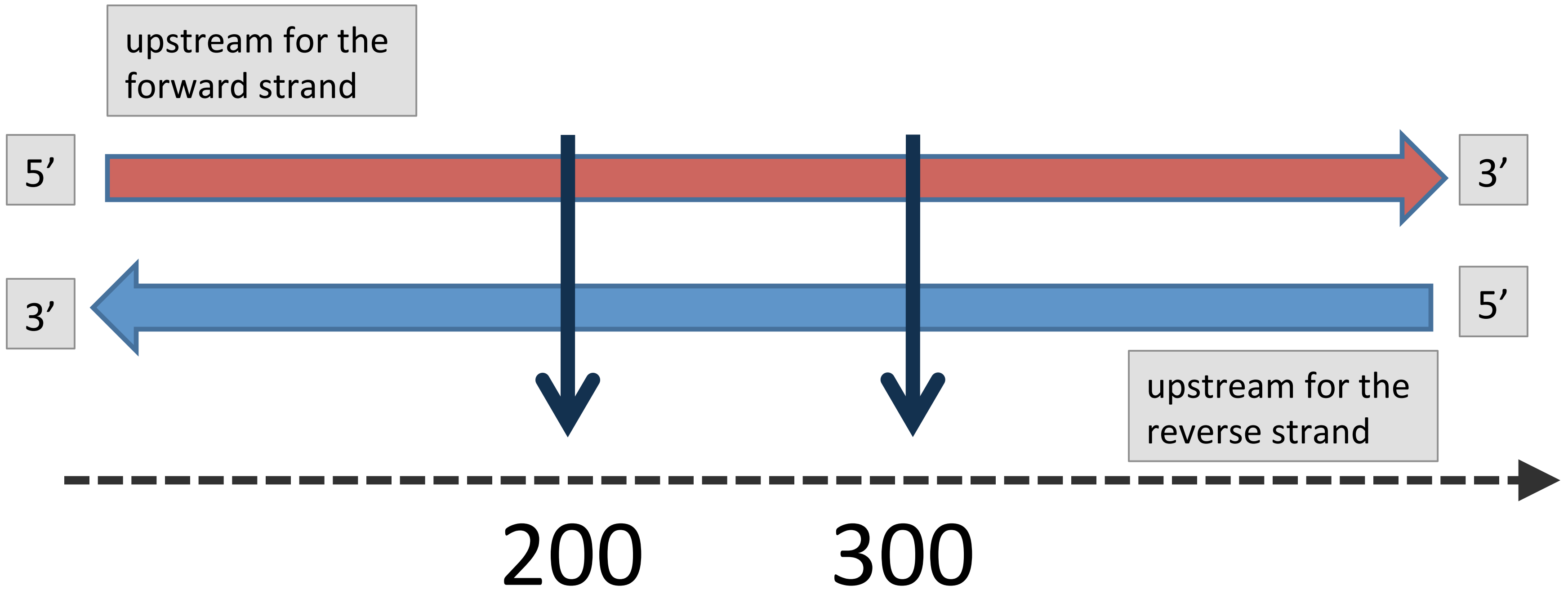
Genome representation concepts

- At the simplest level of abstraction the genome is represented by a one dimensional "space" (lines)
- Genome is two stranded → a line corresponds to each strand
- Each strand has a polarity → each line has a direction
- Strands may be paired → lines are 'paired'
- The smallest unit is one base → one integer on the number line
- Annotations (features) are segments (coordinates) on each line

Genomic coordinates – brief overview

DNA two stranded and directional

But there is only one coordinate system


upstream for the forward strand
5’
3’
3’
5’
upstream for the reverse strand
200
300

Standard formats use **start < end** even for the reverse strand

The **upstream region** – before the 5’ end relative to the direction of transcription

Coordinate systems

- 0 based → 0, 1, 2, … 9
- 1 based → 1, 2, 3, … 10

Typically

- 0 based are non-inclusive **10:20** → **[10, 20)**
- 1 based include both ends **10:20** → **[10, 20]**

Comparing coordinate systems

```
        1 based indexing

Third element     :  3

First ten         :  1,  10
Second ten        :  11, 20
Third ten         :  21, 30

One base long     : start, start

Size of interval  : end - start + 1

Empty interval    :  ?

          Five elements
        starting at 1000

         1000, 1000 + 4
```

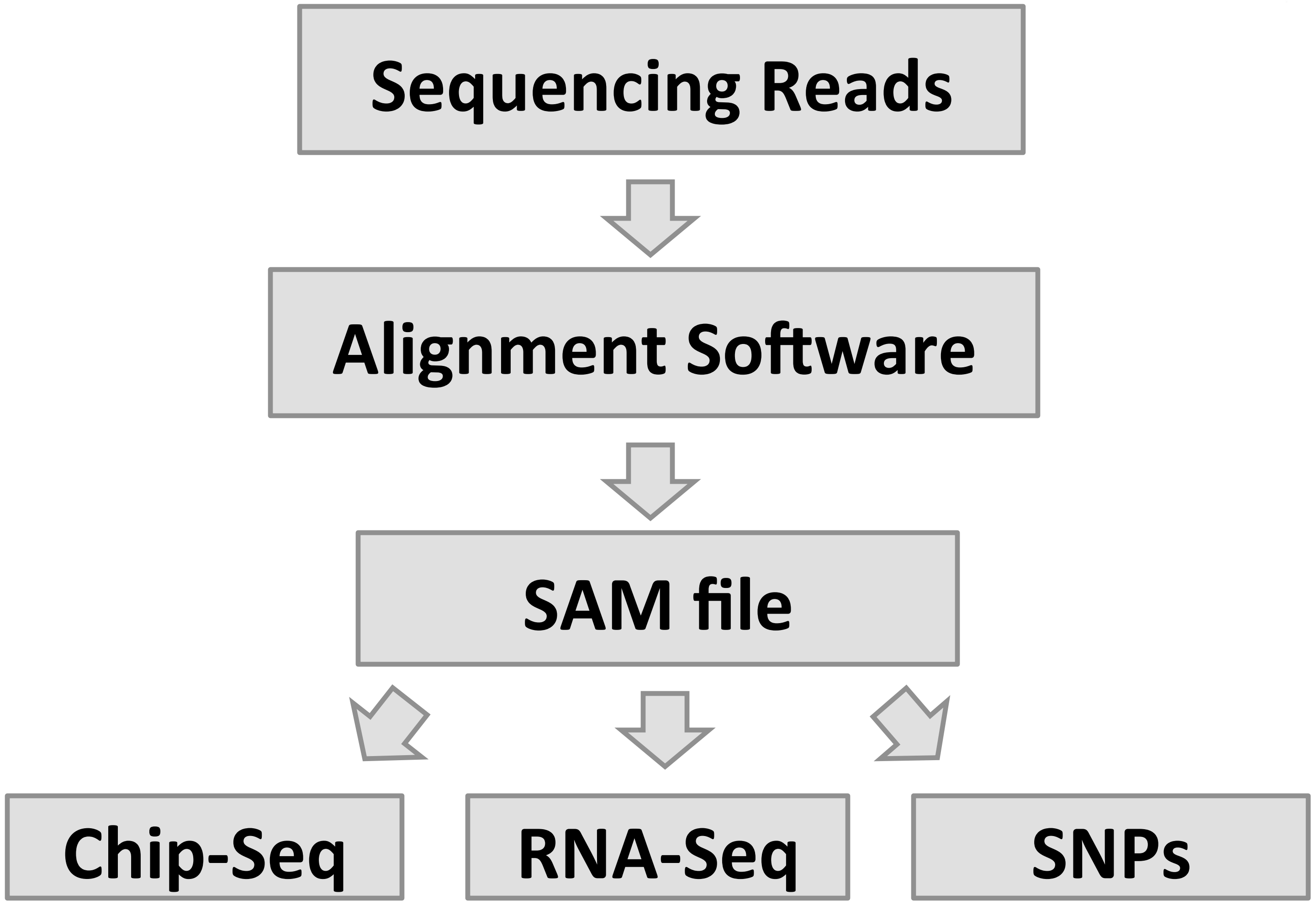
```
        0 based indexing

Third element     :   2

First ten         :   0, 10
Second ten        :   10, 20
Third ten         :   20, 30

One base long     : start, start +1

Size of interval  : end – start

Empty interval    : start, start

          Five elements
        starting at 1000

         1000,  1000 + 5
```

Vote for what you think is better

Fundamental interval formats

- **SAM/BAM** – Sequence Alignment Map
- **BED/GFF →** Gene Annotation representation
- **VCF/BCF →** for variant calls

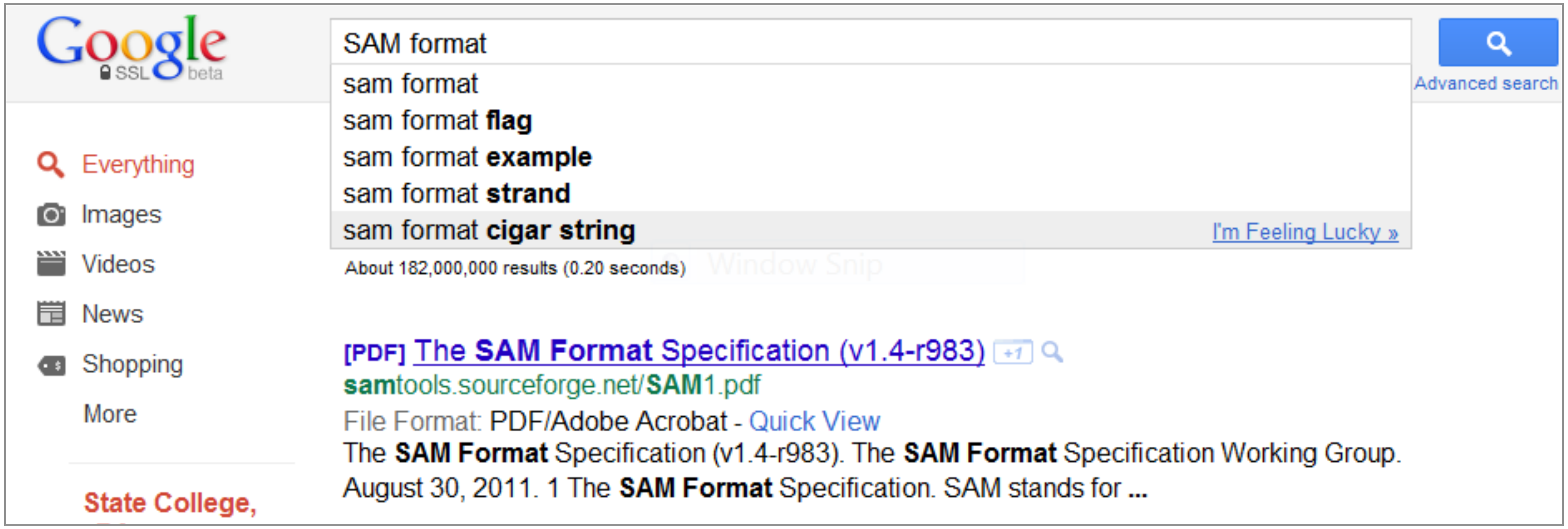
The vast majority of data analyses split off after generating a SAM file!



SAM format: tabular text format

Published as
The Sequence Alignment/Map format and SAMtools by **Heng Li** et al
Bioinformatics 25, Volume 25, Issue 16, 2009

"A TAB-delimited text format consisting of a header section, which is optional, and an alignment section."



SAM format

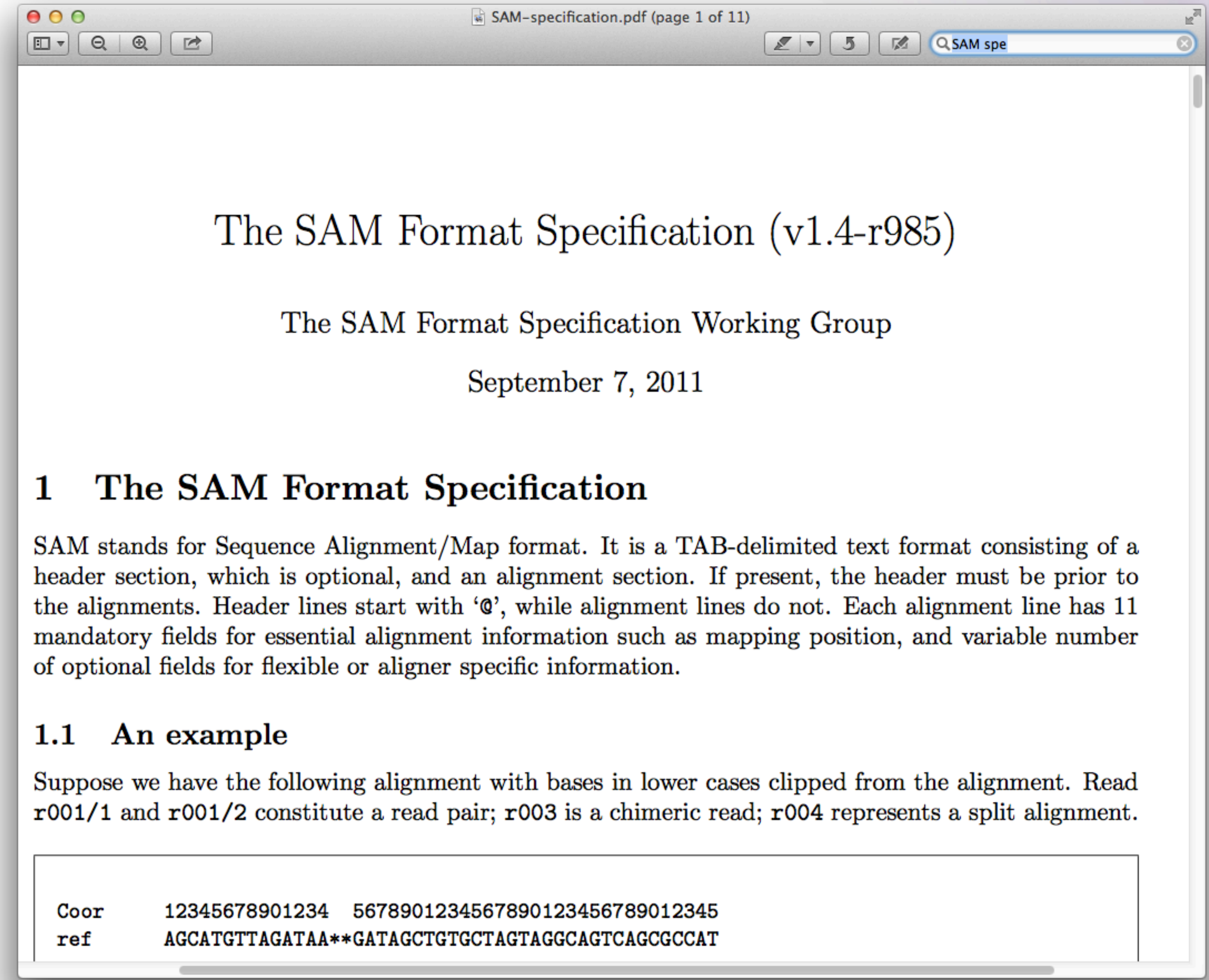
sam format
sam format **flag**
sam format **example**
sam format **strand**
sam format **cigar string**

About 182,000,000 results (0.20 seconds)

[PDF] The **SAM Format** Specification (v1.4-r983)
samtools.sourceforge.net/**SAM**1.pdf
File Format: PDF/Adobe Acrobat - Quick View
The **SAM Format** Specification (v1.4-r983). The **SAM Format** Specification Working Group. August 30, 2011. 1 The **SAM Format** Specification. SAM stands for **...**

Resource: SAM specification

11 required column + optional fields



The SAM Format Specification (v1.4-r985)

The SAM Format Specification Working Group

September 7, 2011

1 The SAM Format Specification

SAM stands for Sequence Alignment/Map format. It is a TAB-delimited text format consisting of a header section, which is optional, and an alignment section. If present, the header must be prior to the alignments. Header lines start with '@', while alignment lines do not. Each alignment line has 11 mandatory fields for essential alignment information such as mapping position, and variable number of optional fields for flexible or aligner specific information.

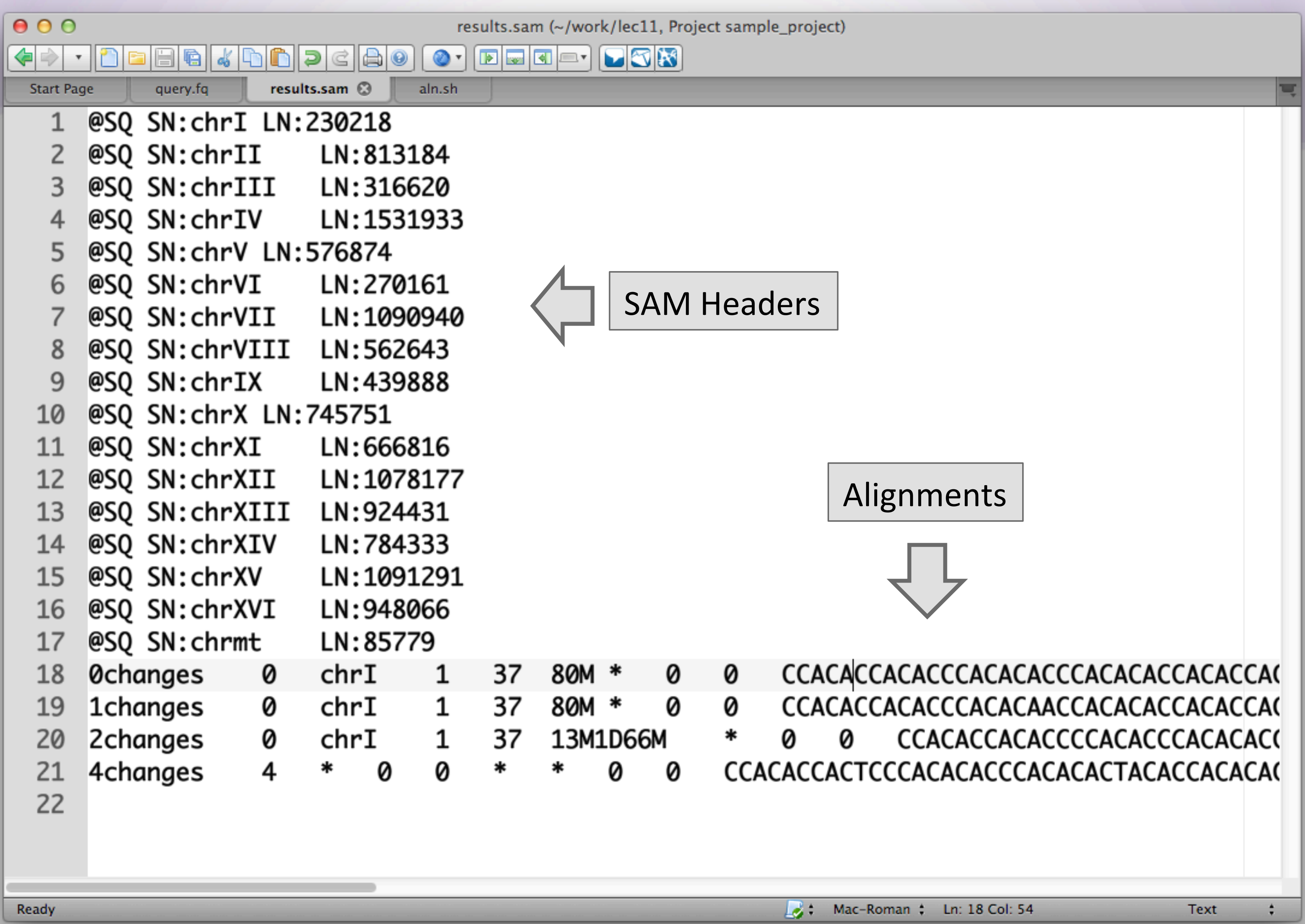
1.1 An example

Suppose we have the following alignment with bases in lower cases clipped from the alignment. Read r001/1 and r001/2 constitute a read pair; r003 is a chimeric read; r004 represents a split alignment.

```
Coor     12345678901234  5678901234567890123456789012345
ref      AGCATGTTAGATAA**GATAGCTGTGCTAGTAGGCAGTCAGCGCCAT
```

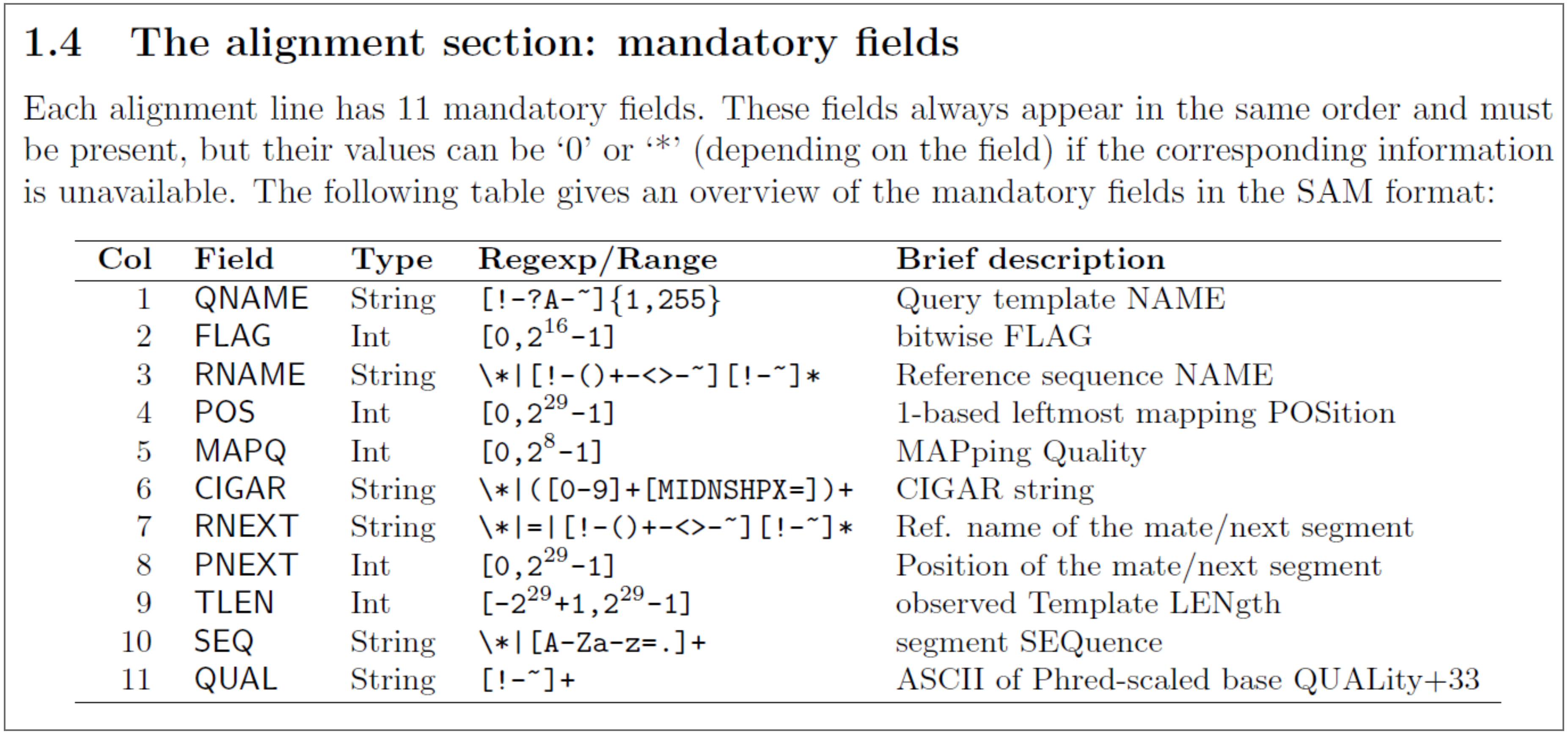
Run the tutorial example

The structure of the SAM file



```
@SQ SN:chrI LN:230218
@SQ SN:chrII     LN:813184
@SQ SN:chrIII    LN:316620
@SQ SN:chrIV     LN:1531933
@SQ SN:chrV LN:576874
@SQ SN:chrVI     LN:270161
@SQ SN:chrVII    LN:1090940
@SQ SN:chrVIII   LN:562643
@SQ SN:chrIX     LN:439888
@SQ SN:chrX LN:745751
@SQ SN:chrXI     LN:666816
@SQ SN:chrXII    LN:1078177
@SQ SN:chrXIII   LN:924431
@SQ SN:chrXIV    LN:784333
@SQ SN:chrXV     LN:1091291
@SQ SN:chrXVI    LN:948066
@SQ SN:chrmt     LN:85779
0changes    0    chrI    1    37   80M *    0    0    CCACACCACACCCACACACCCACACACCACACCA
1changes    0    chrI    1    37   80M *    0    0    CCACACCACACCCACACAACCACACACCACACCA
2changes    0    chrI    1    37   13M1D66M     *    0    0    CCACACCACACCCCACACCCACACAC
4changes    4    *    0    0    *    *    0    0    CCACACCACTCCCACACACCCACACACTACACCACACA
```

A SAM alignment consist of 11 tab delimited columns + one optional column that in turn can contain lots of other elements



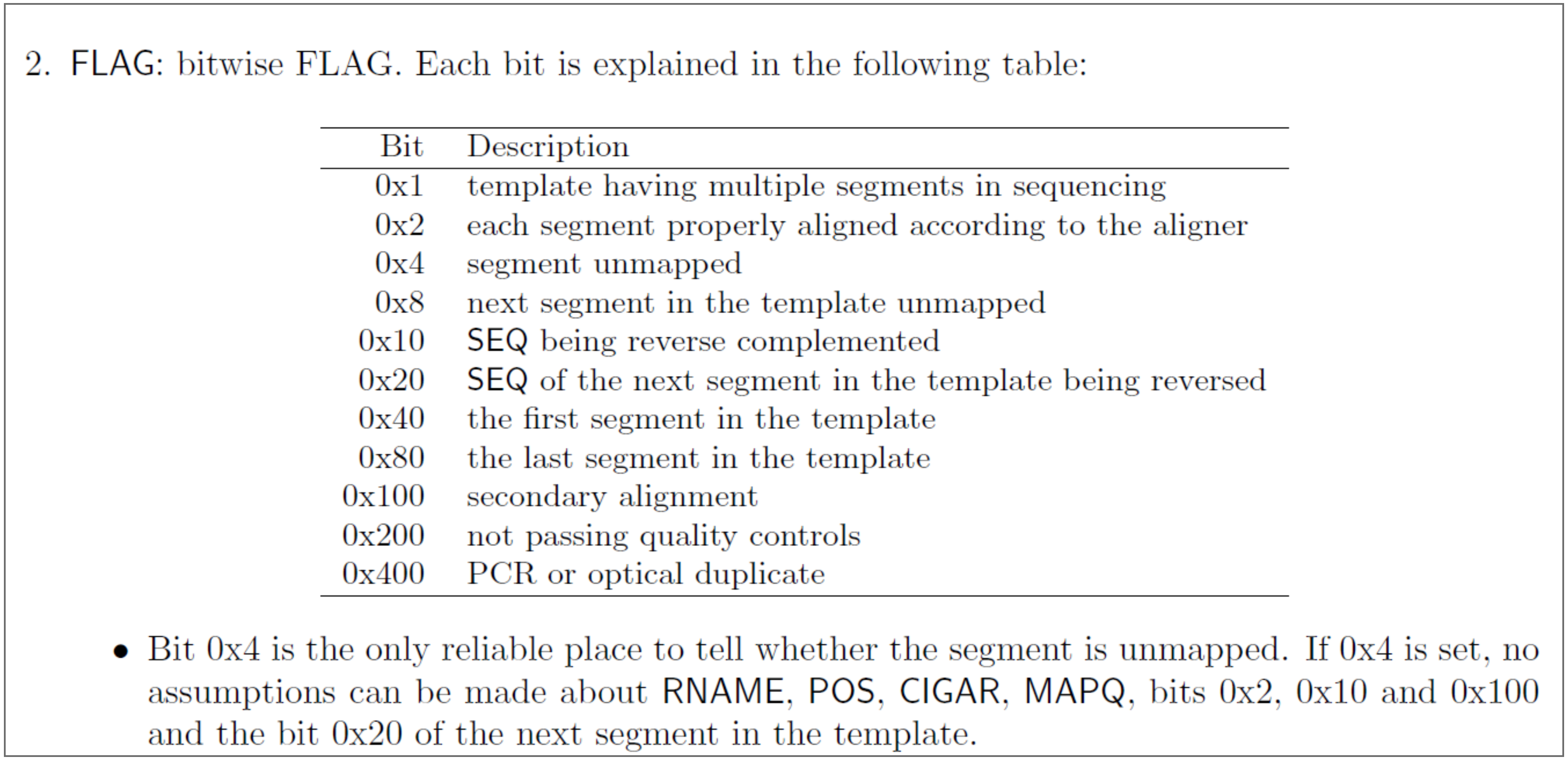
1.4 The alignment section: mandatory fields

Each alignment line has 11 mandatory fields. These fields always appear in the same order and must be present, but their values can be '0' or '*' (depending on the field) if the corresponding information is unavailable. The following table gives an overview of the mandatory fields in the SAM format:

Col	Field	Type	Regexp/Range	Brief description
1	QNAME	String	[!-?A-~]{1,255}	Query template NAME
2	FLAG	Int	[0,2^{16}-1]	bitwise FLAG
3	RNAME	String	*|[!-()+-<>-~][!-~]*	Reference sequence NAME
4	POS	Int	[0,2^{29}-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	*|=|[!-()+-<>-~][!-~]*	Ref. name of the mate/next segment
8	PNEXT	Int	[0,2^{29}-1]	Position of the mate/next segment
9	TLEN	Int	[-2^{29}+1,2^{29}-1]	observed Template LENgth
10	SEQ	String	*|[A-Za-z=.]+	segment SEQuence
11	QUAL	String	[!-~]+	ASCII of Phred-scaled base QUALity+33

Column 1 and 2: QNAME and FLAG (Query name and bitwise flags)

QNAME: the name of the query sequence



2. FLAG: bitwise FLAG. Each bit is explained in the following table:

Bit	Description
0x1	template having multiple segments in sequencing
0x2	each segment properly aligned according to the aligner
0x4	segment unmapped
0x8	next segment in the template unmapped
0x10	SEQ being reverse complemented
0x20	SEQ of the next segment in the template being reversed
0x40	the first segment in the template
0x80	the last segment in the template
0x100	secondary alignment
0x200	not passing quality controls
0x400	PCR or optical duplicate

- Bit 0x4 is the only reliable place to tell whether the segment is unmapped. If 0x4 is set, no assumptions can be made about RNAME, POS, CIGAR, MAPQ, bits 0x2, 0x10 and 0x100 and the bit 0x20 of the next segment in the template.

Column 2: FLAG
the bitwise representation

1	= 00000001	→	paired end read
2	= 00000010	→	mapped as proper pair
4	= 00000100	→	unmappable read
8	= 00001000	→	read mate unmapped
16	= 00010000	→	read mapped on reverse strand

The flag **11** → **1** + **2** + **8** = **0001011** (conditions 1, 2 and 8 satisfied)

It is used to save space – but it does make things a bit more difficult.

Usually very few flags are needed in practice – 0, 4, 16 are the most generic ones

If you need to construct a more complex flag search for explain SAM flags:

http://picard.sourceforge.net/explain-flags.html

Columns 3, 4: RNAME and POS Reference and Position

Column 4 POS: **1-based leftmost mapping POSition of the first matching base**.

Very important to remember later when we need to find the 5’ end (the actual start)

Column 5: MAPQ - Mapping Quality

- Phred score, identical to the quality measure in the fastq file. quality **Q**, probability **P**:

P = 10 ^ (-Q / 10.0)

If **Q=30**, **P=1/1000** → on average, one of out 1000 alignments will be wrong

As good as this sounds it is not easy to compute such a quality.

Details of the mapping quality computation – not easy to find good answers

- Tool specific – there is no standard of what it should be
- The repeat structure of the reference. Reads falling in repetitive regions usually get very low mapping quality.
- The base quality of the read. Low quality means the observed read sequence is possibly wrong, and wrong sequence may lead to a wrong alignment.
- The sensitivity of the alignment algorithm. The true hit is more likely to be missed by an algorithm with low sensitivity, which also causes mapping errors.
- Paired end or not. Reads mapped in pairs are more likely to be correct.

(*from the MAQ manual*)

BWA specific high scores

A read alignment with a mapping quality 30 or above usually implies

- The overall base quality of the read is good.
- The best alignment has few mismatches.
- The read has few or just one `good' hit on the reference, which means the current alignment is still the best even if one or two bases are actually mutations or sequencing errors.

BWA specific low scores

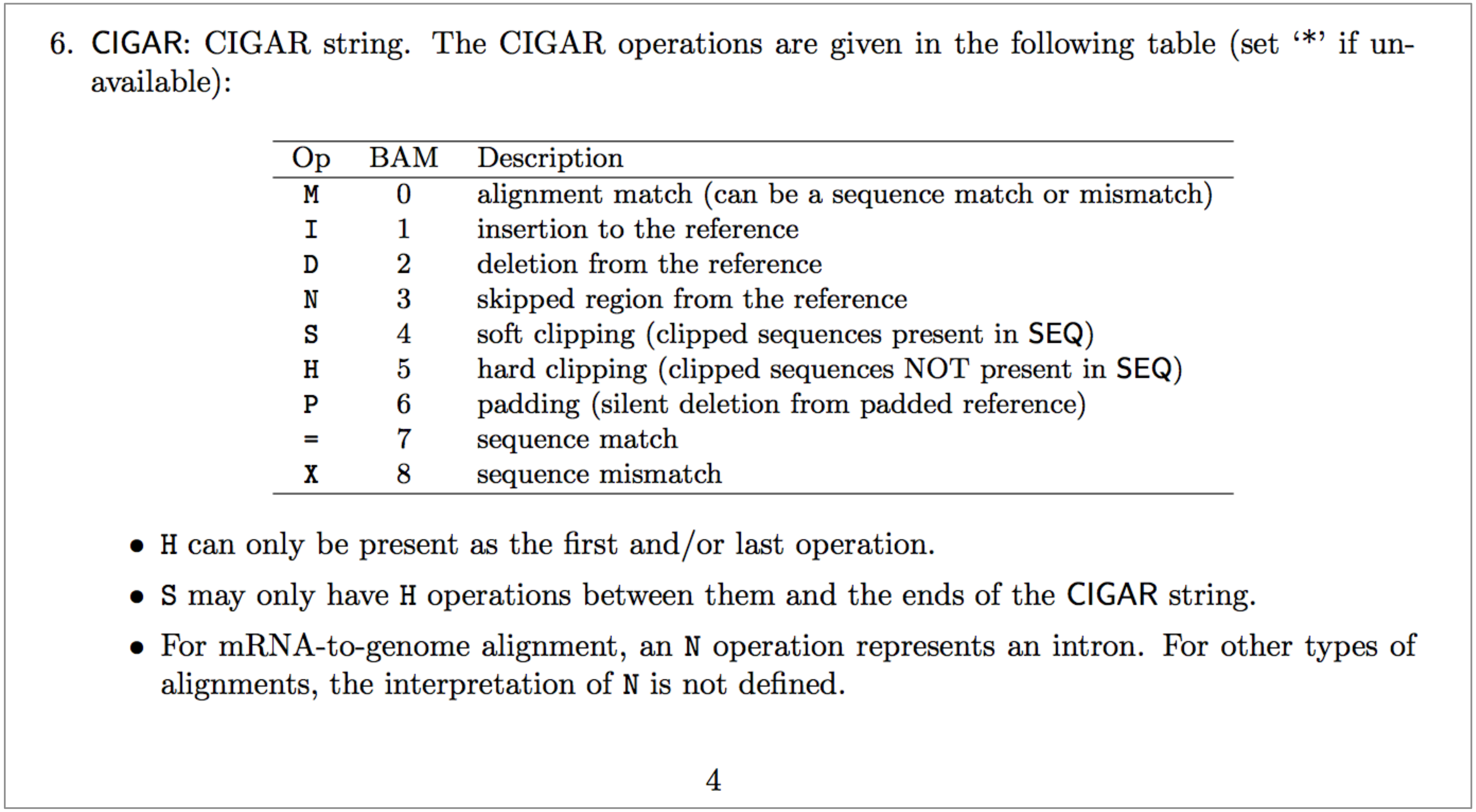
Surprisingly difficult to track down the exact behavior

- Q=0 → if a read can be aligned equally well to multiple positions, BWA will randomly pick one position and give it a mapping quality zero.

- Q=25 → the edit distance equals mismatches and is greater than zero

Column 6: CIGAR

- CIGAR = Compact Idiosyncratic Gapped Alignment Report



6. CIGAR: CIGAR string. The CIGAR operations are given in the following table (set '*' if unavailable):

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

- `H` can only be present as the first and/or last operation.
- `S` may only have `H` operations between them and the ends of the CIGAR string.
- For mRNA-to-genome alignment, an `N` operation represents an intron. For other types of alignments, the interpretation of `N` is not defined.

4

Columns 7, 8, 9: RNEXT, PNEXT, TLEN (used in paired end read sequencing)

- **RNEXT**: the name of the pair
- **PNEXT**: the position of the pair
- **TLEN**: the distance between the leftmost positions of the pairs

These can show the position of reads that are distant – allow us to infer genomic variations

Column 10, 11

- SEQ: the query sequence
- QUAL: the phred encoded quality sequence

SEQ may contain the original sequence or the segment it was aligned to. Not all tools do the same thing.

Column 12 and beyond

Optional information about the alignment process that the tool was able to establish.

TAG_NAME : TYPE : VALUE

For example: NM : i : 1

NM (number of mismatches) are an integer (i) and there was 1 mismatch in this alignment

Session 2

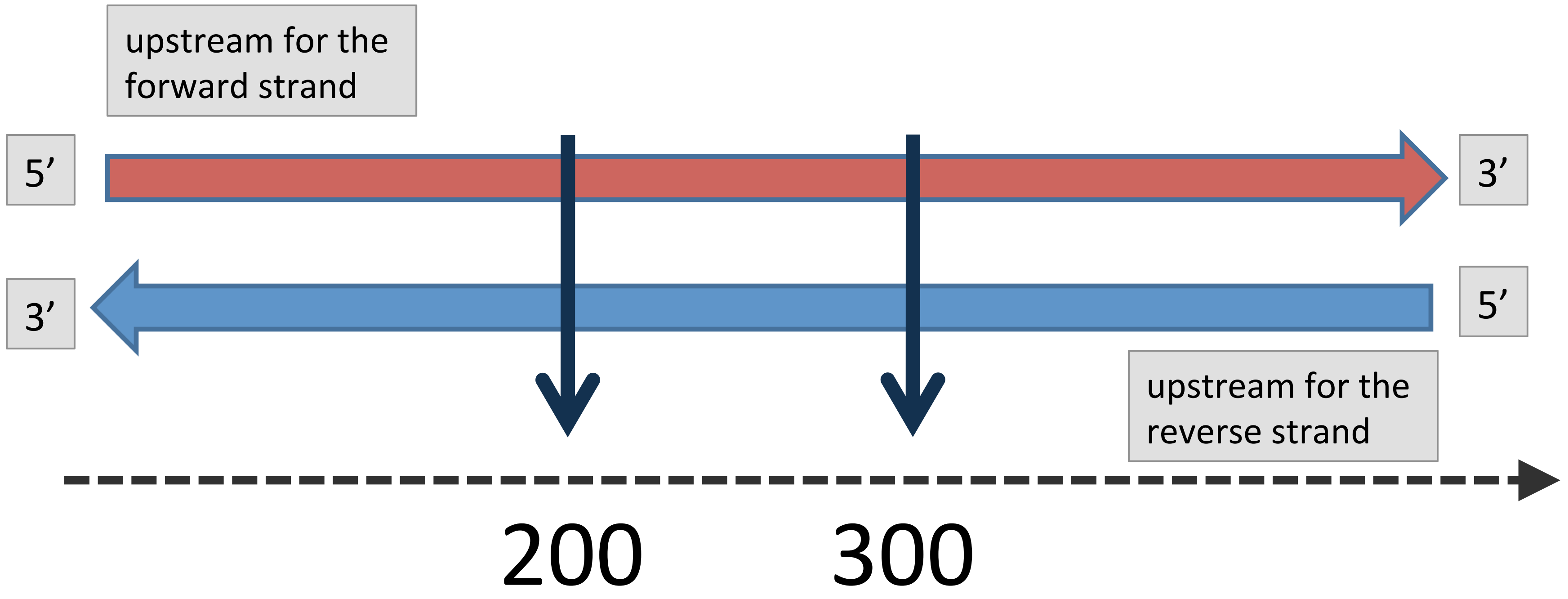
Genome representation concepts

- At the simplest level of abstraction the genome is represented by a one dimensional “space” (lines)
- Genome is two stranded → a line corresponds to each strand
- Each strand has a polarity → each line has a direction
- Strands are paired → lines are ‘paired’
- The smallest unit is one base → one integer on the number line
- Annotations (features) are segments (coordinates) on each line

Genomic coordinates – brief overview

DNA two stranded and directional

But there is only one coordinate system



upstream for the forward strand
5’
3’
3’
5’
upstream for the reverse strand
200
300

Standard formats use **start < end** even for the reverse strand

The **upstream region** – before the 5’ end relative to the direction of transcription

Coordinate systems

- 0 based → 0, 1, 2, … 9
- 1 based → 1, 2, 3, … 10

Typically

- 0 based are non-inclusive **10:20** → **[10, 20)**
- 1 based include both ends **10:20** → **[10, 20]**

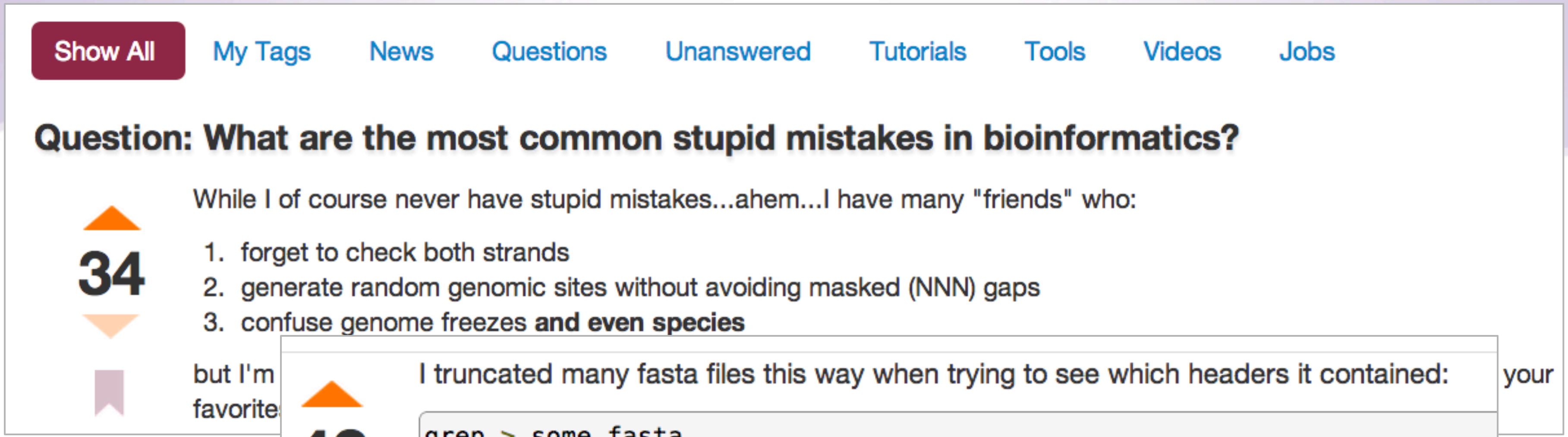
Comparing coordinate systems

```
        1 based indexing

Third element     :  3

First ten         :  1,  10
Second ten        :  11, 20
Third ten         :  21, 30

One base long     : start, start

Size of interval  : end - start + 1

Empty interval    :  ?

          Five elements
        starting at 1000

         1000, 1000 + 4
```

```
        0 based indexing

Third element     :   2

First ten         :   0, 10
Second ten        :   10, 20
Third ten         :   20, 30

One base long     : start, start +1

Size of interval  : end – start

Empty interval    : start, start

          Five elements
        starting at 1000

         1000,  1000 + 5
```

Vote for what you think is better

Pick one coordinate system and stick with it!



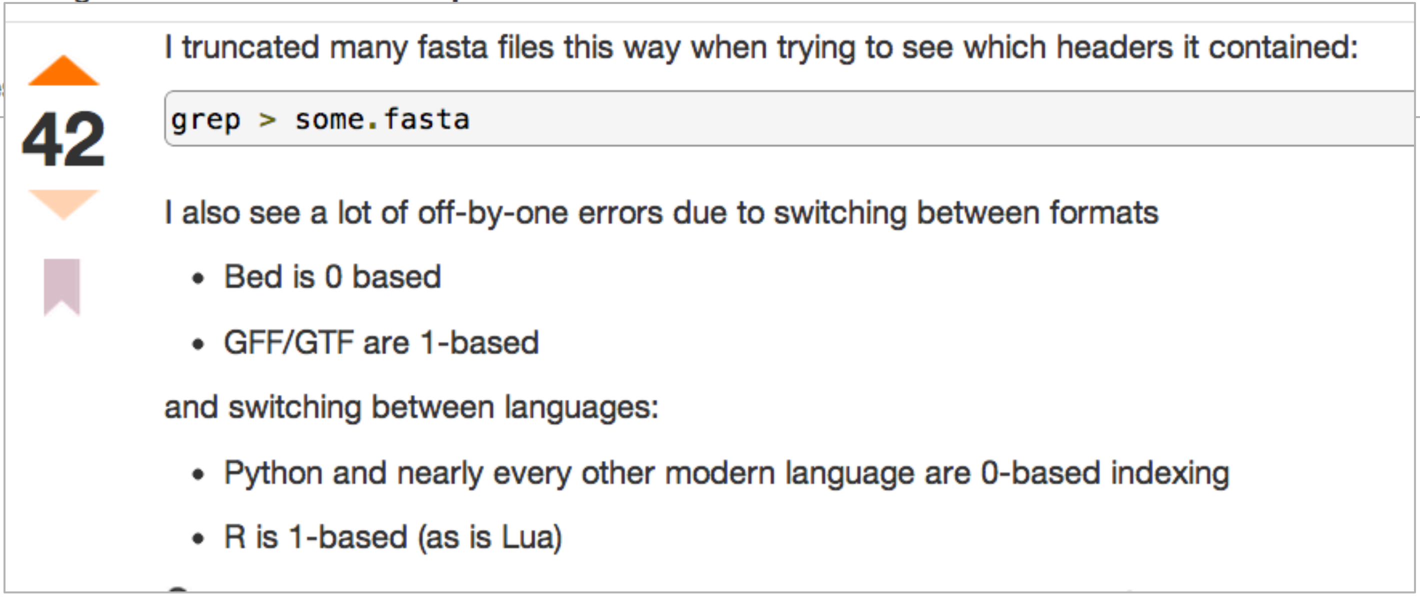
Question: What are the most common stupid mistakes in bioinformatics?

34

While I of course never have stupid mistakes...ahem...I have many "friends" who:

1. forget to check both strands
2. generate random genomic sites without avoiding masked (NNN) gaps
3. confuse genome freezes **and even species**

but I'm … your favorite



42

I truncated many fasta files this way when trying to see which headers it contained:

```
grep > some.fasta
```

I also see a lot of off-by-one errors due to switching between formats

- Bed is 0 based
- GFF/GTF are 1-based

and switching between languages:

- Python and nearly every other modern language are 0-based indexing
- R is 1-based (as is Lua)

1. Pick the standard your group is using and convert every new data to this standard!
2. If you have a choice of what to use pick the one based system! (GFF).

What is a genomic feature?

- Feature: a genomic region (interval) associated with a certain annotation (description).

Typical attributes to describe a feature

1. chromosome
2. start
3. end
4. strand
5. name

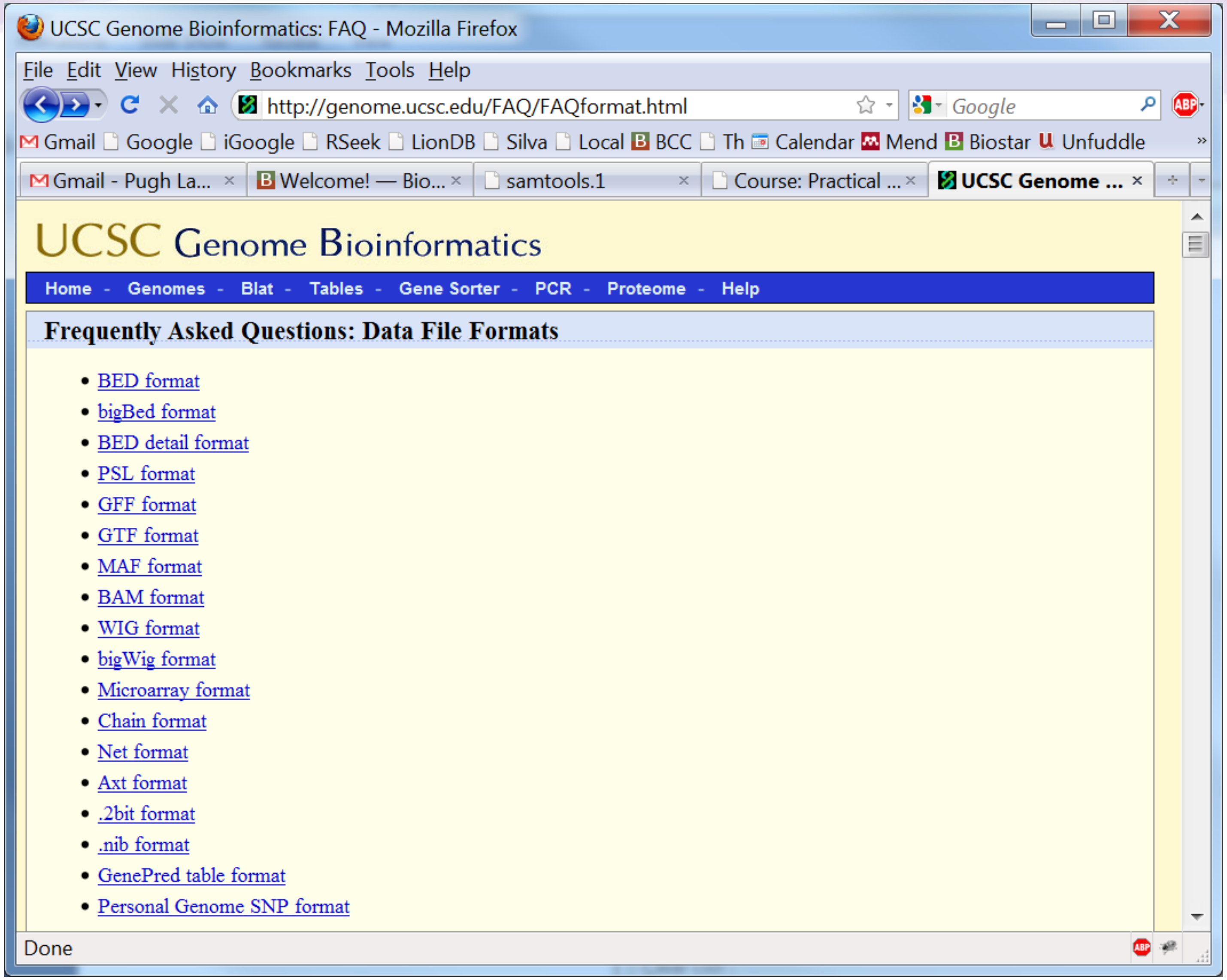
Values on intervals

- A single value characterizes an entire interval → score (value) for the interval

- Continuous values → different value for each base of the interval → analogous to a series of 1bp long intervals

Different data representation formats

http://genome.ucsc.edu/FAQ/FAQformat.html


UCSC Genome Bioinformatics: FAQ - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://genome.ucsc.edu/FAQ/FAQformat.html
Google
Gmail Google iGoogle RSeek LionDB Silva Local BCC Th Calendar Mend Biostar Unfuddle
Gmail - Pugh La...
Welcome! — Bio...
samtools.1
Course: Practical ...
UCSC Genome ...
UCSC Genome Bioinformatics
Home - Genomes - Blat - Tables - Gene Sorter - PCR - Proteome - Help
Done

Frequently Asked Questions: Data File Formats

- BED format
- bigBed format
- BED detail format
- PSL format
- GFF format
- GTF format
- MAF format
- BAM format
- WIG format
- bigWig format
- Microarray format
- Chain format
- Net format
- Axt format
- .2bit format
- .nib format
- GenePred table format
- Personal Genome SNP format

Two commonly used formats

- **BED** – UCSC genome browser → 0 based non inclusive → also used to display tracks in the genome browser (US "standard")
  (variants: **bigBed**, **bedgraph**)

- **GFF** – Sanger institute in Great Britain → 1 based inclusive indexing system ("European standard"),
  (variants: **GTF**, **GFF 2.0**)

BED format

Search for BED format

Tab separated 3 required and 9 optional columns. Lower numbered filed must be filled.

1.	**chrom**	(name of the chromosome, sequence id)
2.	**chromStart**	(starting position on the chromosome)
3.	**chromEnd**	(end position of the chromosome, **note** this base is not included!)
4.	**name**	(feature name)
5.	**score**	(between 0 and 1000)
6.	**strand**	(+ or -)
7.	**thickStart**	(the starting position at which the feature is drawn thickly)
8.	**thickEnd**	(the ending position at which the feature is drawn thickly)
9.	**itemRGB**	(RGB color → 255, 0, 0 display color of the data contained)
10.	**blockCount**	(the number of blocks (exons) in the BED line.)
11.	**blockSizes**	(a comma-separated list of the block sizes)
12.	**blockStarts**	(a comma-separated list of the block starts)

GFF format

Search for GFF3 → http://www.sequenceontology.org/gff3.shtml

Tab separated with 9 columns. Missing attributes may be replaced with a dot → .

1.	**Seqid**	(usually chromosome)
2.	**Source**	(where is the data coming from)
3.	**Type**	(usually a term from the sequence ontology)
4.	**Start**	(interval start relative to the seqid)
5.	**End**	(interval end relative to the seqid)
6.	**Score**	(the score of the feature, a floating point number)
7.	**Strand**	(+ or –)
8.	**Phase**	(used to indicate reading frame for coding sequences)
9.	**Attributes**	(semicolon separated attributes → Name=ABC;ID=1)

Wiggle format

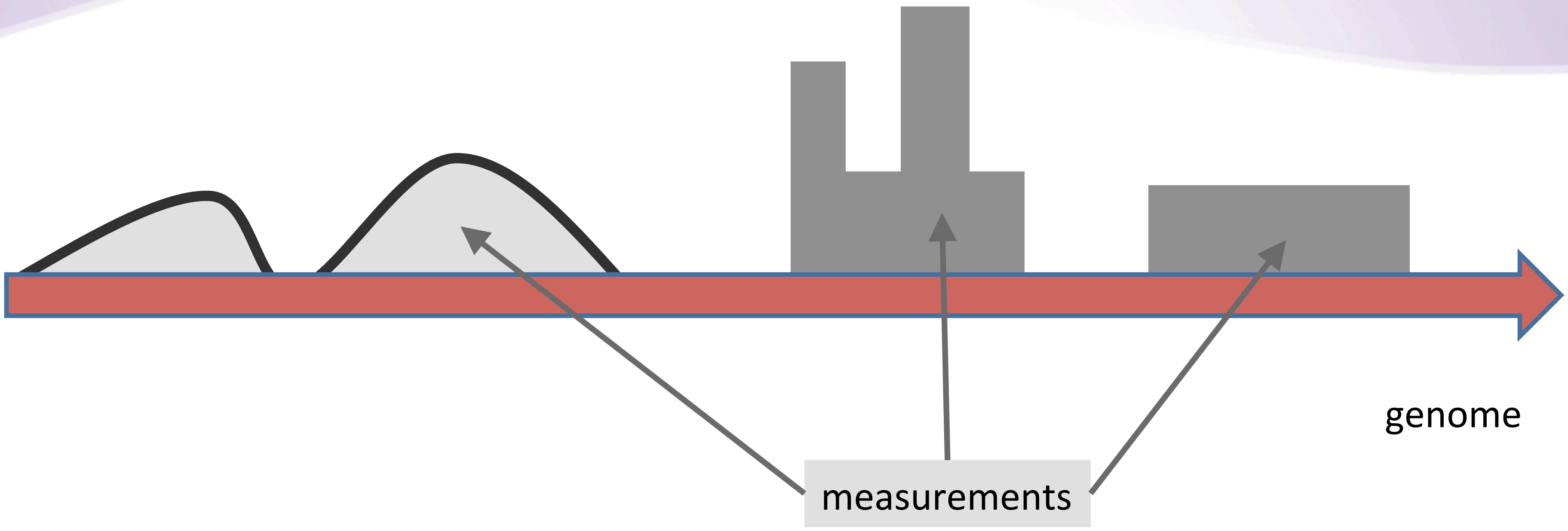
- two versions → fixed step, variable step each trying to optimize the amount of data storage

```
fixedStep chrom=chr1 start=100 step=1
10
15
11
22
… … …
```

```
variableStep chrom=chr1
100 10
101 15
102 11
103 22
variableStep chrom=chr2
2000 23
2005 40
… … …
```

it has other parameters → can be deceiving (looks simpler than it is)

Picking a representation depends on the data



1) variable step wiggle
2) fixed step wiggle
3) multiple intervals as GFF

We may have data in different coordinate systems!

Being "**one off**" is one of the most common errors in bioinformatics.

Conversion from GFF to BED

(start, end) → (start – 1, end)

Conversion from BED to GFF

(start, end) → (start + 1, end)

Not that there will be differences when selecting positions that depend on the END coordinate!

Handling coordinates relative to intervals

What are the coordinate of the base preceding and following the interval

GFF [start, end] → preceding base is at **start - 1**

BED [start, end) → preceding base is at **start - 1**

GFF [start, end] → next base is at **end + 1**

BED [start, end) → next base is at **end**

Optimized representation formats

- **BED**, **GFF** → intervals with a single score, binary version **BED** → **bigBed** (binary, indexed **BED** file)

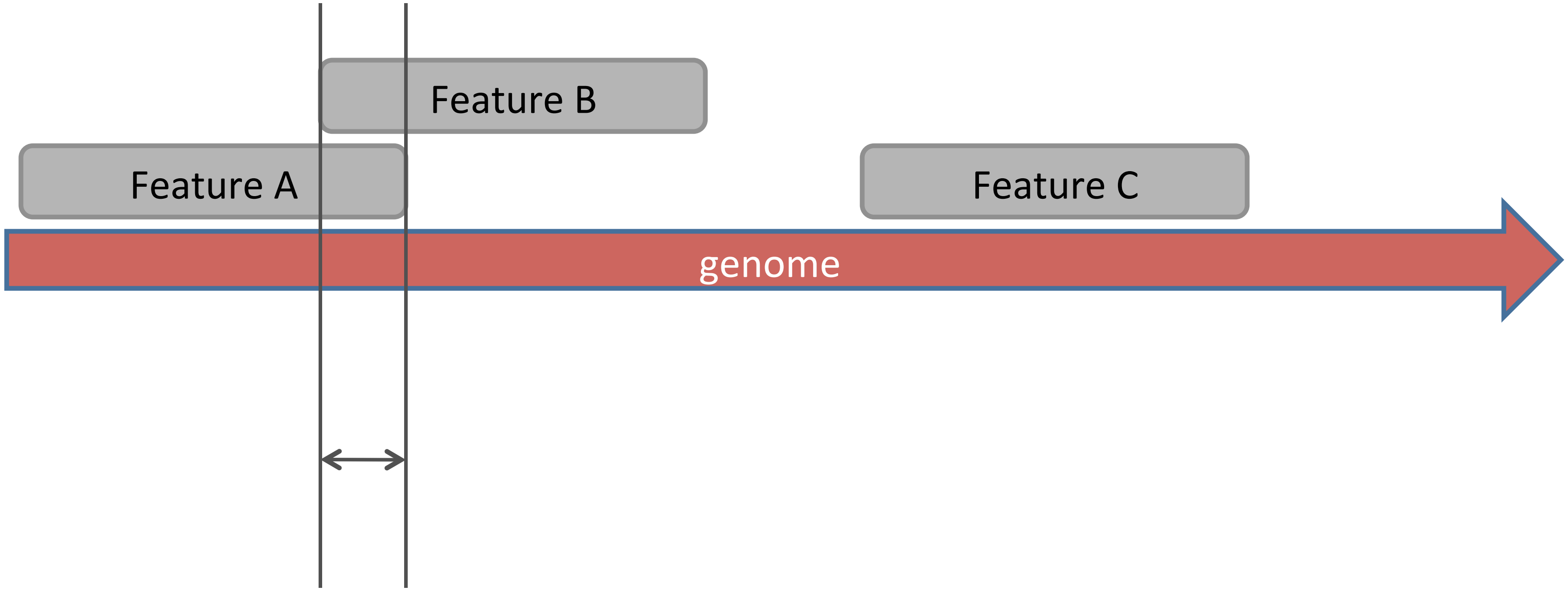
- **WIG** (wiggle) → value for each base over the genome → binary version of wiggle **bigWig** (binary indexed WIG file)

- **binary → smaller file**
- **indexed → searchable (querying enabled)**

bigBed/bigWig converters available from UCSC

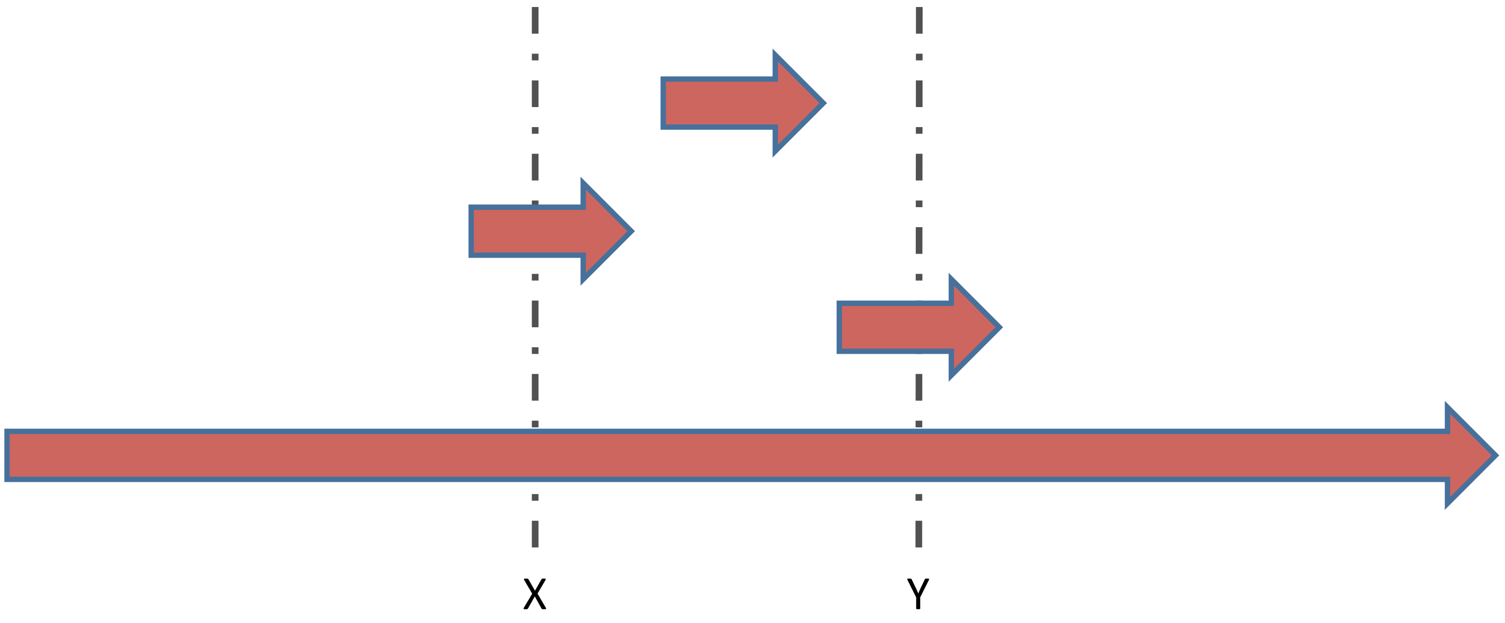
Overlap/intersect

- Two features are said to overlap or intersect if they share at least one base in common.



Interval overlap concepts

- Unexpectedly complex tasks as it needs to account for various types of positioning: full containment of either interval or partial overlaps



useful formulas (X,Y is the query interval):

- **midpoint = (start + end) / 2**
- **overlap condition: (start < Y) and (end > X)**

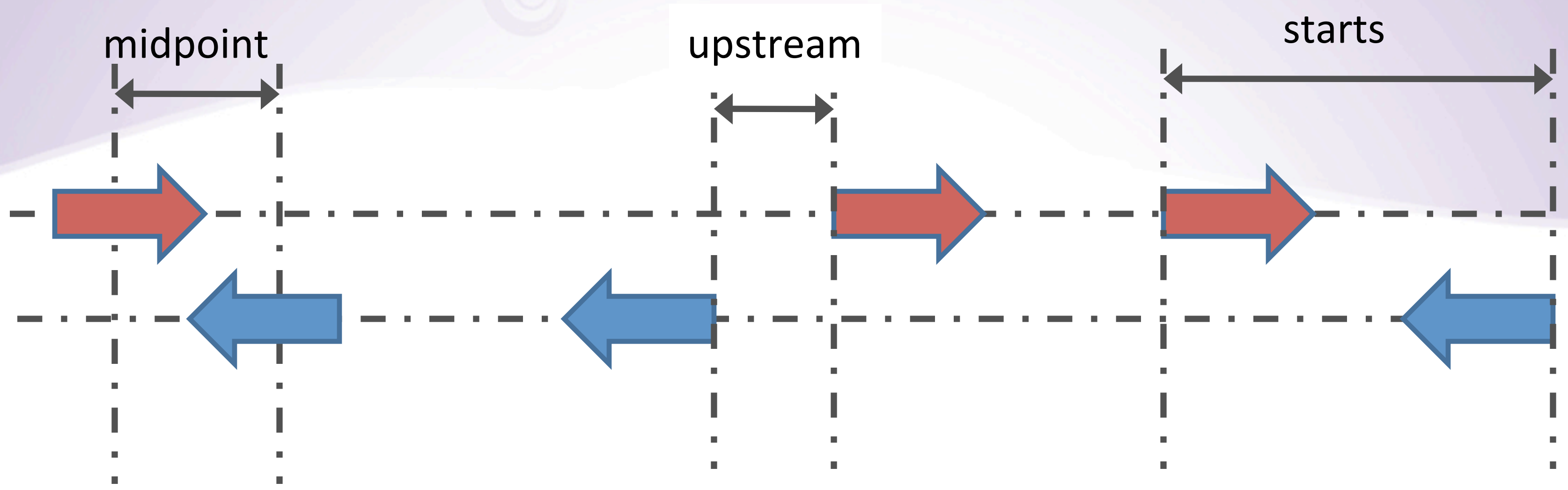
Examples of interval related tasks

Finding intervals relative to one another

- for each feature find the intervals from another dataset that are close/overlapping with it
- for each interval on one strand find the closest on the other strand

An interval is not a point! - the question needs to be specified more precisely

Other important details



- What are the anchor points (the locations that represent the intervals)
- Which direction does the comparison proceed – upstream, downstream?
- Often (always?) we need to create another transformed interval data that conforms to what we actually need

Interval representation

- binning → redundantly storing data at different zoom levels - originally implemented in UCSC genome browser (also used in BAM and BedTools)

- A different option → interval tree, usually supported by programming languages

- For intervals that are very similar in size or have other constraints: a sort + binary search may also work well

BedTools

- High performance software package that operates on multiple interval oriented data formats: BED, GFF, SAM, BAM and VCF

- Download and install bedtools

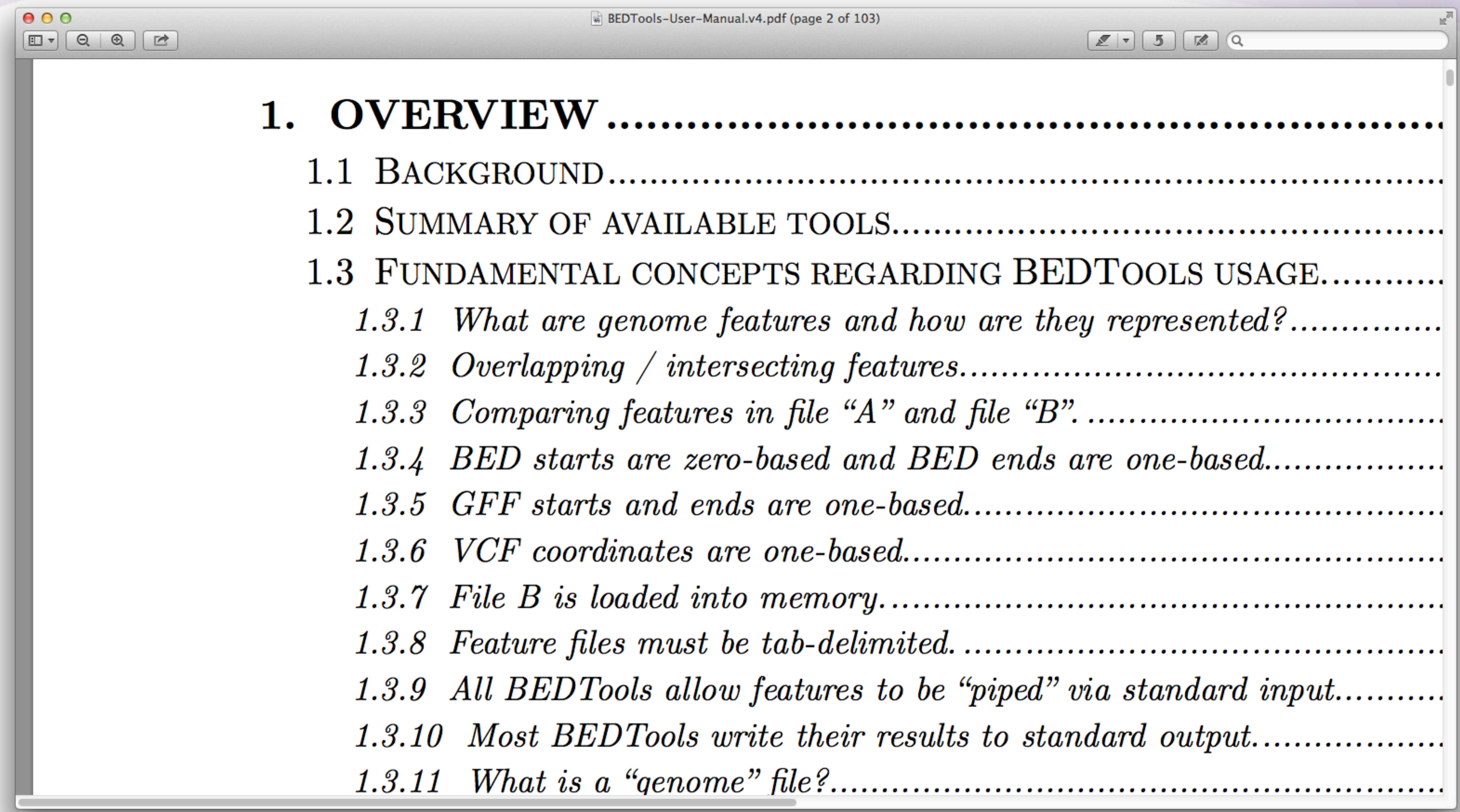
  http://code.google.com/p/bedtools/

Quinlan AR and Hall IM,
BEDTools: a flexible suite of utilities for comparing genomic features.
Bioinformatics. 26, 6, (2010)

BedTools concepts

- There are many (25 and growing) tools with different names
- Most tools write to the standard output
- The – (minus) character specifies the standard input
- Can be chained with *pipes* like all UNIX commands
- Most tools write their help when invoked, others need –h flag
- Flag options can substantially change the output format

BedTools has an excellent user manual



1. OVERVIEW ....

1.1 BACKGROUND ....

1.2 SUMMARY OF AVAILABLE TOOLS ....

1.3 FUNDAMENTAL CONCEPTS REGARDING BEDTOOLS USAGE ....

1.3.1 What are genome features and how are they represented? ....

1.3.2 Overlapping / intersecting features ....

1.3.3 Comparing features in file "A" and file "B". ....

1.3.4 BED starts are zero-based and BED ends are one-based. ....

1.3.5 GFF starts and ends are one-based. ....

1.3.6 VCF coordinates are one-based. ....

1.3.7 File B is loaded into memory. ....

1.3.8 Feature files must be tab-delimited. ....

1.3.9 All BEDTools allow features to be "piped" via standard input. ....

1.3.10 Most BEDTools write their results to standard output. ....

1.3.11 What is a "genome" file? ....

Basic concepts

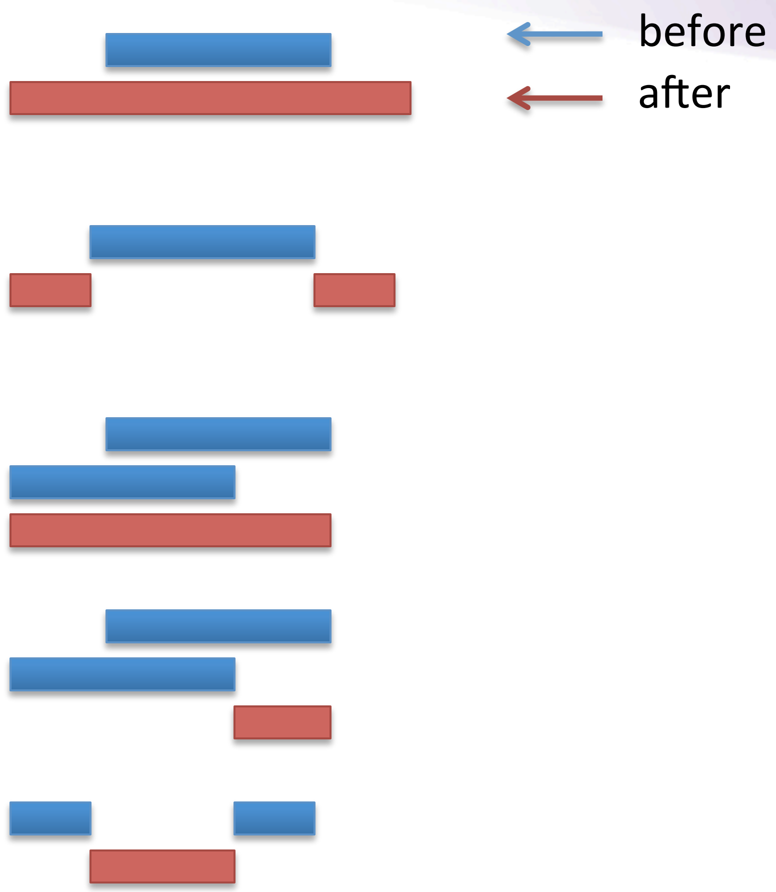
- For any operation that requires **two files** the tools asks for file **A** and file **B**
- Each element in file **A** is matched against each element in file **B**
- File **B** is loaded into memory – try to make that the **smaller** file

Two identical ways to invoke

- The old style mode contains a different tool for each task (the manual covers these tools):
  - **intersectBed**
  - **windowBed**
  - **closestBed**
- A new style mode that contains only one tool that takes commands like **samtools**:
  - **bedtools** intersect
  - **bedtools** window
  - **bedtools** closest

BedTools operators

- slop (extend)
- flank
- merge
- subtract
- complement



Essential feature: Strand Awareness

- Some tools take a –l (left), -r (right) parameter that will have a different effect if the “stranded” mode is turned on

1. **default mode**: left, right are interpreted on the default coordinate system (screen)

2. **stranded mode**: left, right are interpreted in the transcriptional direction 5’to 3’