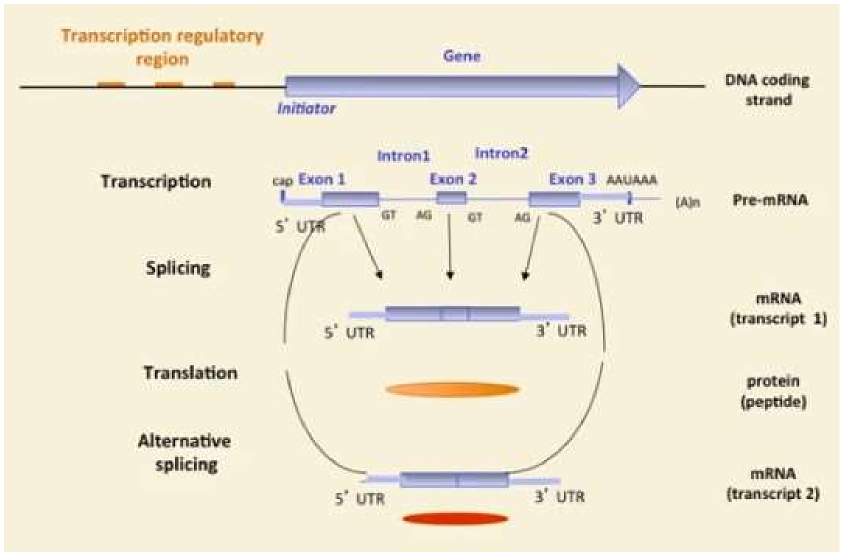
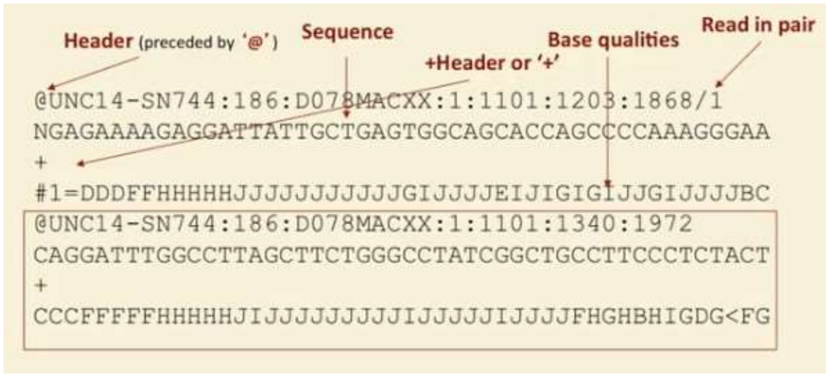
Eukaryotic Gene Expression



Fastq Format for Next Gen Sequencing Data


Header (preceded by '@')
Sequence
+Header or '+'
Base qualities
Read in pair

```
@UNC14-SN744:186:D078MACXX:1:1101:1203:1868/1
NGAGAAAAGAGGATTATTGCTGAGTGGCAGCACCAGCCCCAAAGGGAA
+
#1=DDDFFHHHHHJJJJJJJJJJGIJJJJEIJIGIGIJJGIJJJJBC
@UNC14-SN744:186:D078MACXX:1:1101:1340:1972
CAGGATTTGGCCTTAGCTTCTGGGCCTATCGGCTGCCTTCCCTCTACT
+
CCCFFFFFHHHHHJIJJJJJJJJJIJJJJJIJJJJFHGHBHIGDG<FG
```

Base Quality Scores

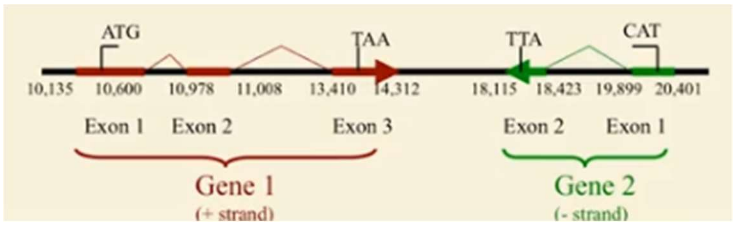
- Let p_b = probability that the call at base *b* is correct
- Quality value: $Q_{sanger} = -10 \log_{10} p_b$ (integer)
- Sanger (Phred quality scores): 0…93 (ASCII characters 33…126)

  !"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`

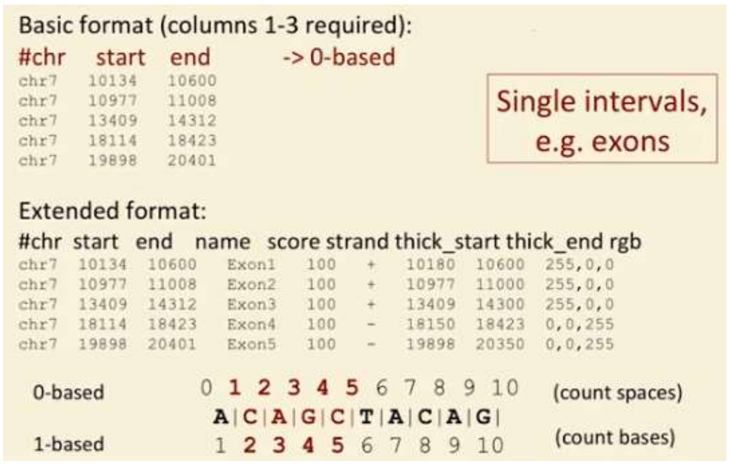
  abcdefghijklmnopqrstuvwxyz{|}~
- In practice, the maximal quality value is ~40, while quality values below 20 are typically considered low.

Genomic Features and Annotation

- **Genome annotation** = determine the precise location and structure (intervals, or lists of intervals, and associated biological information) of genomic features along the genome
- **Genomic features**: genes, promoters, protein binding sites, translation start/stop site, DNaseI sites, etc.
- **Example – gene annotations**:
    - Exon/intron structure (exon and intron start-end coordinates)
    - Strand (+ or -)
    - Start and end sites for translation (ORF)



BED Format



Basic format (columns 1-3 required):

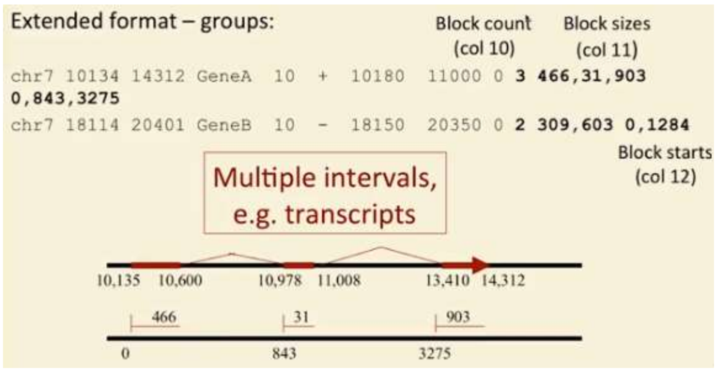
```
#chr   start   end        -> 0-based
chr7   10134   10600
chr7   10977   11008
chr7   13409   14312
chr7   18114   18423
chr7   19898   20401
```

Single intervals, e.g. exons

Extended format:

```
#chr  start  end    name   score strand thick_start thick_end rgb
chr7  10134  10600  Exon1  100   +      10180       10600     255,0,0
chr7  10977  11008  Exon2  100   +      10977       11000     255,0,0
chr7  13409  14312  Exon3  100   +      13409       14300     255,0,0
chr7  18114  18423  Exon4  100   -      18150       18423     0,0,255
chr7  19898  20401  Exon5  100   -      19898       20350     0,0,255
```

```
0-based    0 1 2 3 4 5 6 7 8 9 10   (count spaces)
            A|C|A|G|C|T|A|C|A|G|
1-based     1 2 3 4 5 6 7 8 9 10    (count bases)
```



Extended format – groups:

Block count (col 10) — Block sizes (col 11) — Block starts (col 12)

```
chr7 10134 14312 GeneA  10  +  10180  11000 0 3 466,31,903
0,843,3275
chr7 18114 20401 GeneB  10  -  18150  20350 0 2 309,603 0,1284
```

Multiple intervals, e.g. transcripts

GTF (Genomic Transfer Format)

- Each interval feature takes one line
- Columns 1-9 separated by tab '\t'; fields within column 9 separated by space ' '
- Column 9 can have additional attributes
- Coordinates are 1-based
- Lines are grouped by gene_id

```
#chr program feature start end strand frame gene_id; txpt_id

chr7 GF exon 10135 10600 100 + . gene_id "genA"; transcript_id "genA.1";
chr7 GF exon 10978 11008 100 + . gene_id "genA"; transcript_id "genA.1";
chr7 GF exon 13410 14312 100 + . gene_id "genA"; transcript_id "genA.1";
chr7 GF exon 18115 18423 100 - . gene_id "genB"; transcript_id "genB.1";
chr7 GF exon 19899 20401 100 - . gene_id "genB"; transcript_id "genB.1";
```

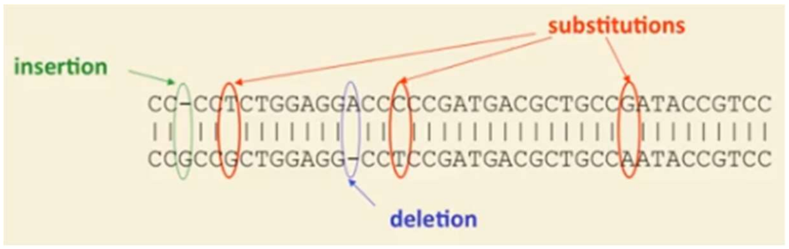
GFF3 (Genomic Features Format Version 3)

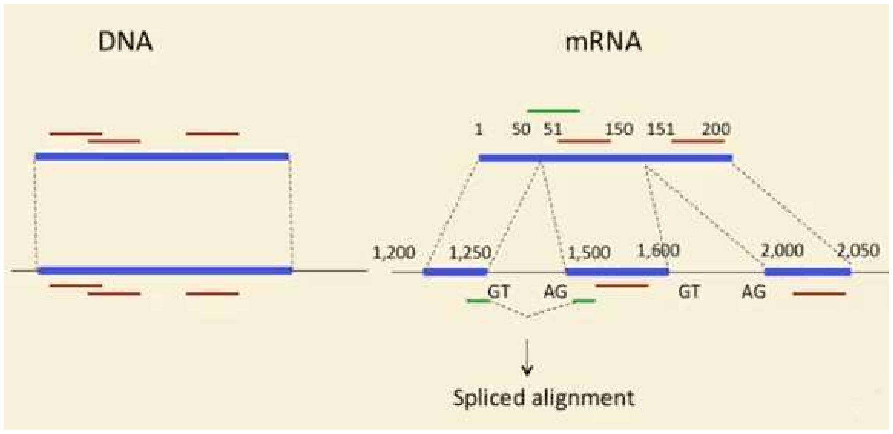
- Again, one line for every exon in the gene
- Also a line for each mRNA. Last column identifies which mRNA the exon is in.

```
#chr source feature start end strand frame ID;Name;Parent

##gff-version 3
chr7 GF mRNA 10135 14312 100 + . ID=mrna001;Name=genA
chr7 GF exon 10135 10600 100 + . ID=exon00001;Parent=mrna001
chr7 GF exon 10978 11008 100 + . ID=exon00002;Parent=mrna001
chr7 GF exon 13410 14312 100 + . ID=exon00003;Parent=mrna001
Chr7 GF mRNA 18115 18423 100 - . ID=mrna002;Name=genB
chr7 GF exon 18115 18423 100 - . ID=exon00004;Parent=mrna002
chr7 GF exon 19899 20401 100 - . ID=exon00005;Parent=mrna002
```

Alignment

- Sequence a fragment of the gene (RNA) or genomic region (DNA), then map (align) it to the genome
- Alignment = a mapping between the letter of the two sequences, with spacers (indels)
- The alignment will take into account differences such as polymorphisms and sequencing errors, and introns (for genes)


substitutions
insertion
CC-CCTCTGGAGGACCCCCGATGACGCTGCCGATACCGTCC
CCGCCGCTGGAGG-CCTCCGATGACGCTGCCAATACCGTCC
deletion

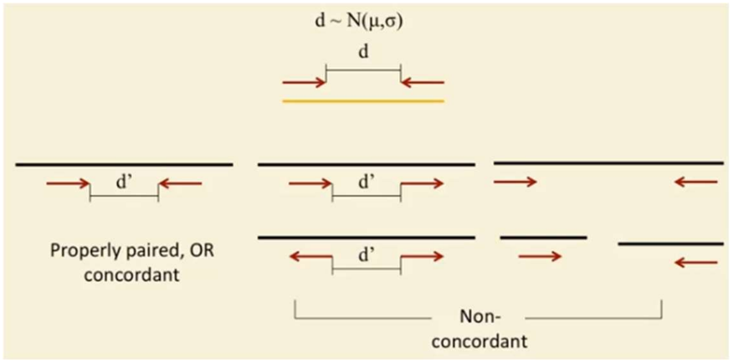


Red: continuous alignment

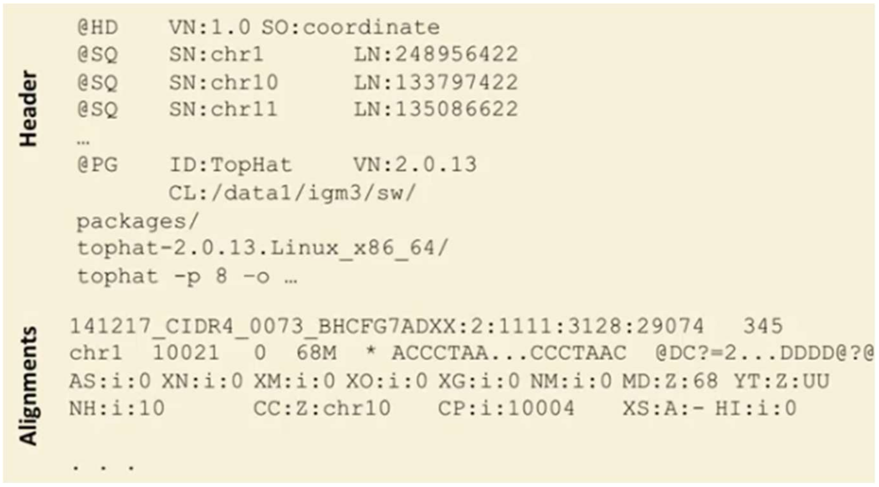
Green: spans the boundary between two exons, so they will be divided along the genome

NGS Alignments

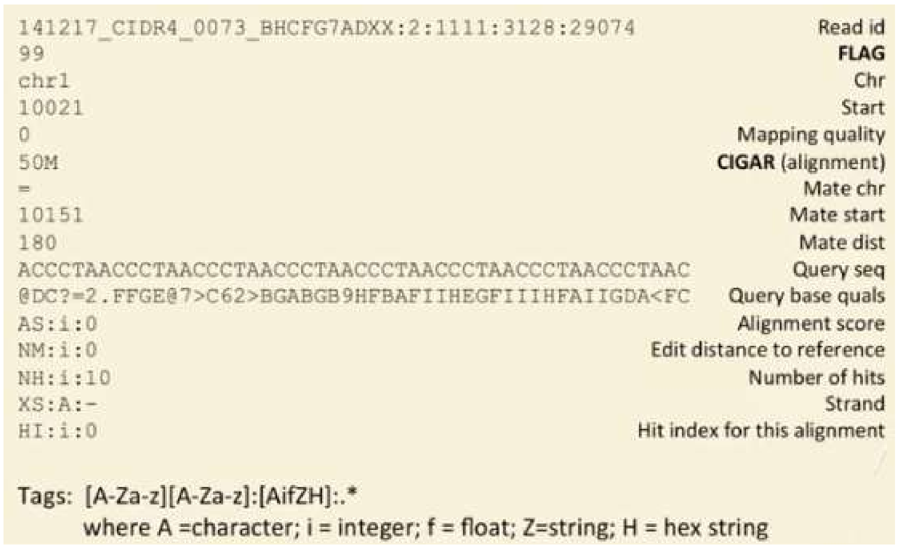
- **Properly paired** (**concordant**): reads are in opposite directions and facing each other, and distance between them is within the specified boundaries (not too far apart or on different chromosomes)



SAM/BAM Format


Header
Alignments

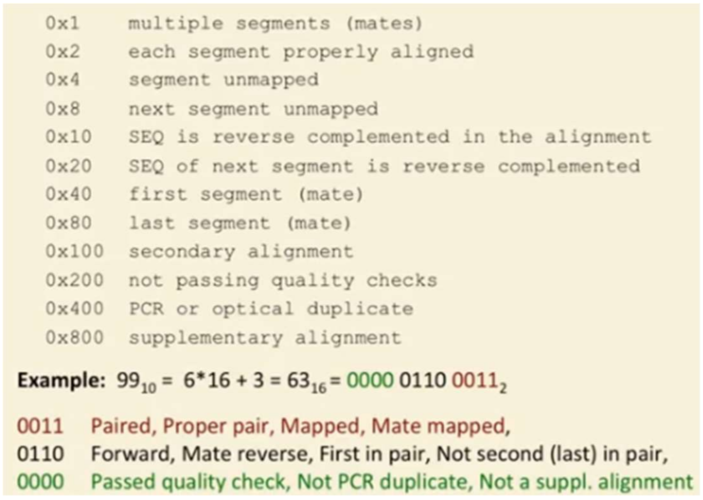
```
@HD    VN:1.0 SO:coordinate
@SQ    SN:chr1       LN:248956422
@SQ    SN:chr10      LN:133797422
@SQ    SN:chr11      LN:135086622
…
@PG    ID:TopHat     VN:2.0.13
       CL:/data1/igm3/sw/
packages/
tophat-2.0.13.Linux_x86_64/
tophat -p 8 -o …

141217_CIDR4_0073_BHCFG7ADXX:2:1111:3128:29074   345
chr1  10021  0  68M  * ACCCTAA...CCCTAAC  @DC?=2...DDDD@?@
AS:i:0 XN:i:0 XM:i:0 XO:i:0 XG:i:0 NM:i:0 MD:Z:68 YT:Z:UU
NH:i:10      CC:Z:chr10    CP:i:10004    XS:A:- HI:i:0

. . .
```



141217_CIDR4_0073_BHCFG7ADXX:2:1111:3128:29074	Read id
99	**FLAG**
chr1	Chr
10021	Start
0	Mapping quality
50M	**CIGAR** (alignment)
=	Mate chr
10151	Mate start
180	Mate dist
ACCCTAACCCTAACCCTAACCCTAACCCTAACCCTAACCCTAACCCTAAC	Query seq
@DC?=2.FFGE@7>C62>BGABGB9HFBAFIIHEGFIIIHFAIIGDA<FC	Query base quals
AS:i:0	Alignment score
NM:i:0	Edit distance to reference
NH:i:10	Number of hits
XS:A:-	Strand
HI:i:0	Hit index for this alignment

Tags: [A-Za-z][A-Za-z]:[AifZH]:.*
where A =character; i = integer; f = float; Z=string; H = hex string

FLAG

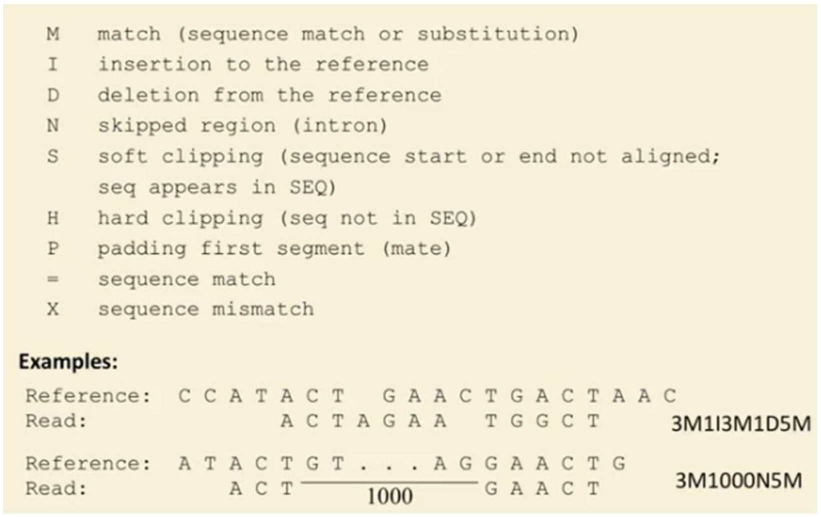


```
0x1    multiple segments (mates)
0x2    each segment properly aligned
0x4    segment unmapped
0x8    next segment unmapped
0x10   SEQ is reverse complemented in the alignment
0x20   SEQ of next segment is reverse complemented
0x40   first segment (mate)
0x80   last segment (mate)
0x100  secondary alignment
0x200  not passing quality checks
0x400  PCR or optical duplicate
0x800  supplementary alignment
```

Example: $99_{10} = 6*16 + 3 = 63_{16} = 0000\ 0110\ 0011_2$

0011	Paired, Proper pair, Mapped, Mate mapped,
0110	Forward, Mate reverse, First in pair, Not second (last) in pair,
0000	Passed quality check, Not PCR duplicate, Not a suppl. alignment

CIGAR



```
M    match (sequence match or substitution)
I    insertion to the reference
D    deletion from the reference
N    skipped region (intron)
S    soft clipping (sequence start or end not aligned;
     seq appears in SEQ)
H    hard clipping (seq not in SEQ)
P    padding first segment (mate)
=    sequence match
X    sequence mismatch
```

Examples:

```
Reference:  C C A T A C T   G A A C T G A C T A A C
Read:               A C T A G A A   T G G C T          3M1I3M1D5M

Reference:  A T A C T G T . . . A G G A A C T G
Read:           A C T ----1000---- G A A C T           3M1000N5M
```

read spanning two exons

SAMtools

- **samtools1**: load the SAMtools package
- **flagstat**: provides a quick summary of alignment statistics
    - Input: BAM/SAM file
    - Outputs counts for: total reads, mapped/unmapped reads, properly paired reads, duplicate reads, singleton reads
    - Useful for basic quality control and alignment assessment
- **sort**: sorts SAM/BAM files by coordinate or read name
    - Required before indexing or variant calling
    - Input: SAM/BAM file → Output: sorted BAM file
    - Usage: samtools1 sort input.bam -o sorted.bam
    - Improves performance for downstream tools
- **index**: creates an index (.bai) file for a sorted BAM file
    - Enables random access to BAM file regions (e.g., specific chromosomes or loci)
    - Required for tools like samtools view with region specification
    - Usage: samtools1 index sorted.bam
- **merge**: combines multiple BAM files into a single BAM file
    - Useful for merging data from multiple sequencing runs or lanes
    - All input files must be coordinate-sorted
    - Usage: samtools1 merge merged.bam input1.bam input2.bam ...
- **view**: converts between SAM and BAM formats.
    - Filters alignments by region, flag, mapping quality, etc.
    - -h shows alignments and header (full SAM file); -H shows precisely the header
    - <u>Convert SAM → BAM</u>: samtools1 view -bT input.sam > output.sam.bam
        - -b means binary output; T means there is a reference file
    - <u>Extract alignments in region</u>: samtools1 view input.bam chr1:1000-2000
    - Versatile tool for inspecting or manipulating alignment data
- **zcat**: inspect contents of file without unzipping
    - Example: zcat NA12814_1.fastq.gz | wc -l
- **nohup**: ignore the SIGHUP (hangup) signal, so the command/script will continue to run in the background even after the user logs out or the terminal session is closed

BEDtools

- **intersect**: finds overlapping intervals between two genomic datasets (e.g., BED, GTF, VCF files)
    - Identify regions where two datasets (like ChIP-seq peaks and gene annotations) intersect
    - bedtools intersect -a fileA.bed -b fileB.bed
    - **-wa**: Reports only the entries from file A (the first file) that overlap with any entry in file B.
    - **-wb**: Reports the entry from file A and the entire overlapping entry from file B for each overlap.
    - **-wo**: Like -wa -wb, but adds a column showing the number of overlapping base pairs between A and B entries.
    - **-wao**: Reports all entries from file A, with overlaps from file B and the number of overlapping base pairs; if no overlap, outputs B fields as . and overlap as 0.
- **bamtobed**: converts aligned reads from BAM format to BED format
    - Useful for downstream processing and analysis of alignments using BEDtools
    - bedtools bamtobed -i input.bam
- **bedtobam**: converts BED-formatted intervals back into BAM format
    - Needed when converting processed BED files (e.g., filtered reads) back to BAM for visualization or further alignment-based tools
    - bedtools bedtobam -i input.bed -g genome.txt
- **getfasta**: Extracts DNA sequences from a reference FASTA file based on BED intervals
    - Retrieve sequences underlying genomic features such as peaks or exons
    - bedtools getfasta -fi genome.fa -bed regions.bed -fo output.fa
- bedtools >& bedtools.log to see all