

Genome 540 Discussion

Feb. 5th 2026



HW4 Questions???

Overview

1. Write a program to find the highest weight path on a WDAG
2. Convert (by hand!) the example WDAG provided, run your program on that.
3. Write another program to read in a genome and scoring scheme, generate a linked list (simple WDAG, edges = nucleotides)
4. Run your program from Step 1 on the linked list

Formatting WDAGS

V (vname1)

V (vname2)

V (vname3)

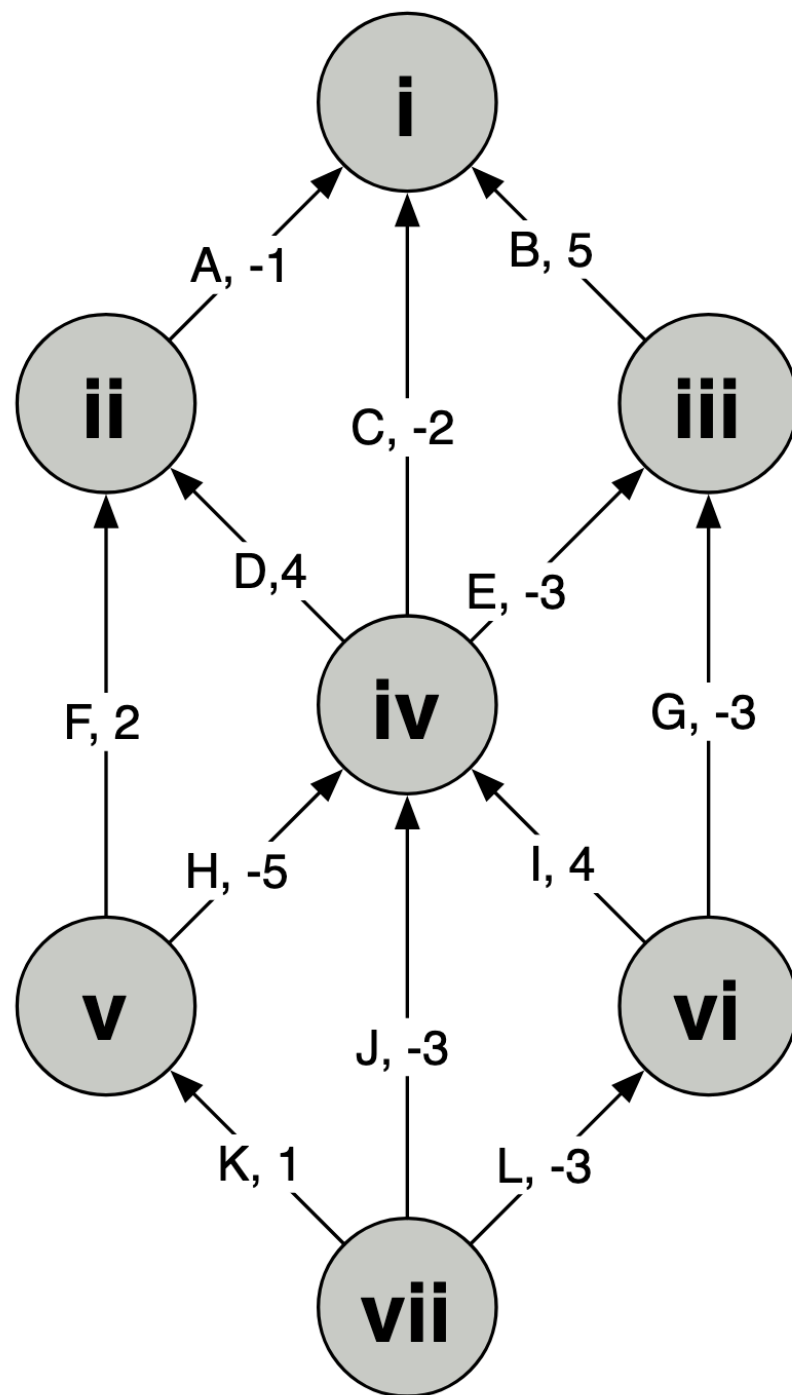
....

E (ename1) (vname1) (vname2) (edgeweight)

E (ename2) (vname2) (vname5) (edgeweight)

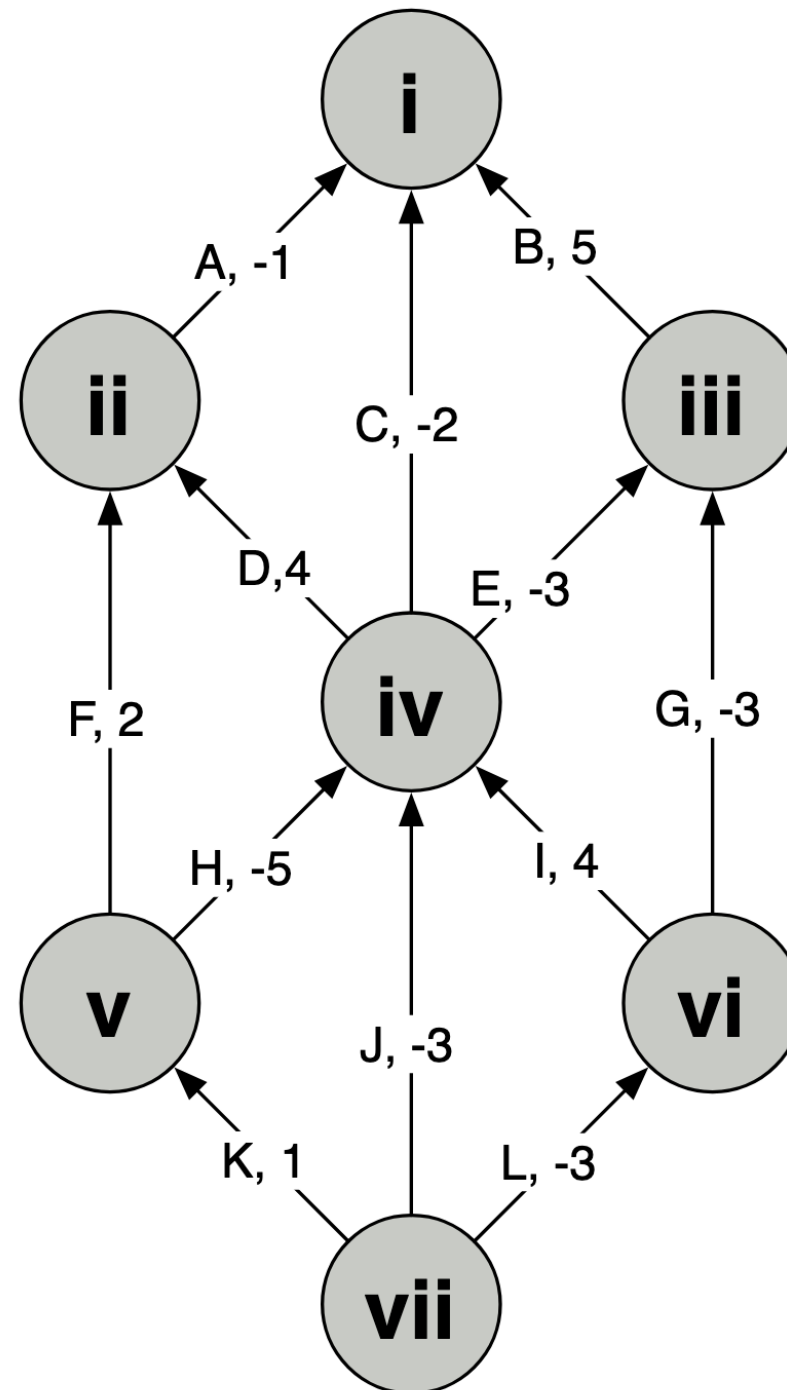
E (ename3) (vname3) (vname5) (edgeweight)

...



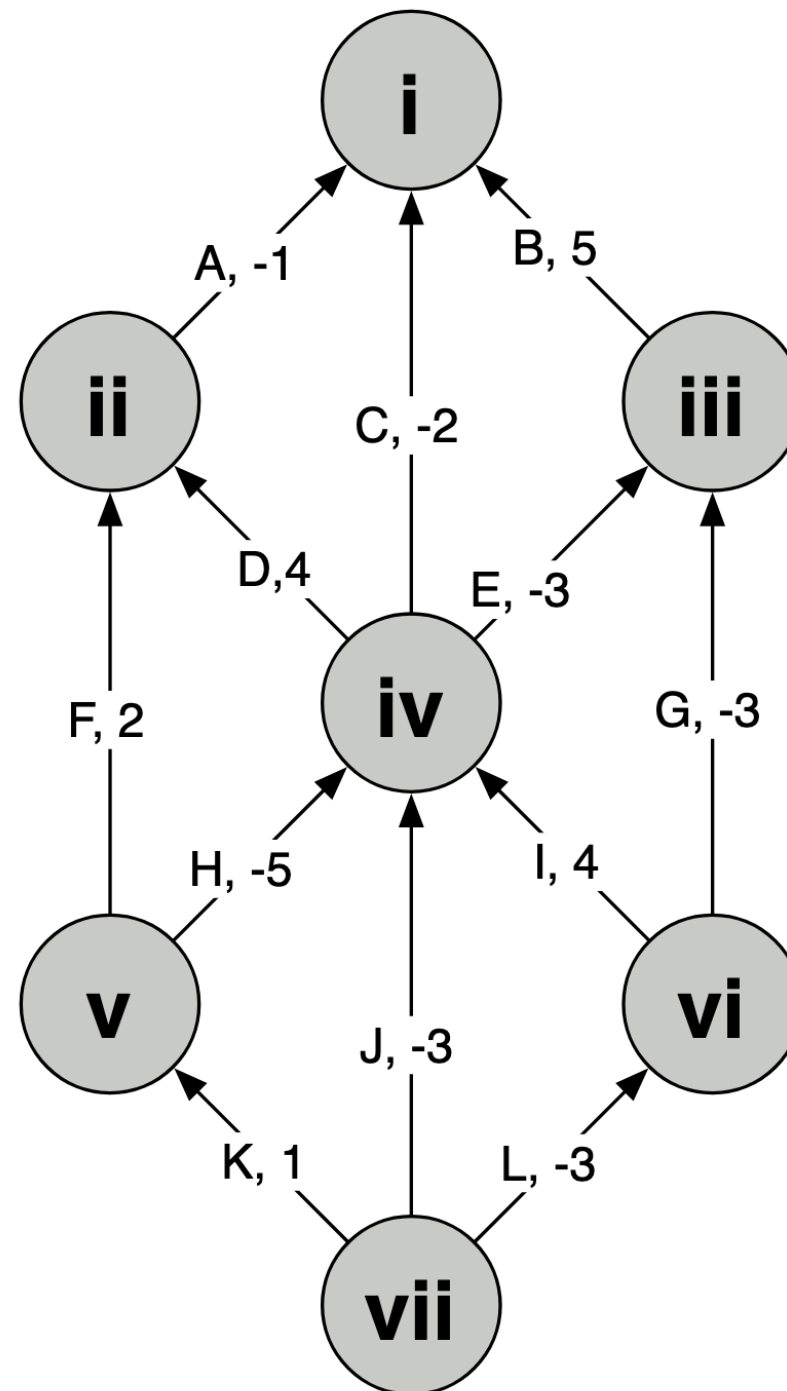
What is the highest weight path if the ends are constrained to (vii) and (i)?

What is the highest weight path if the ends are unconstrained?

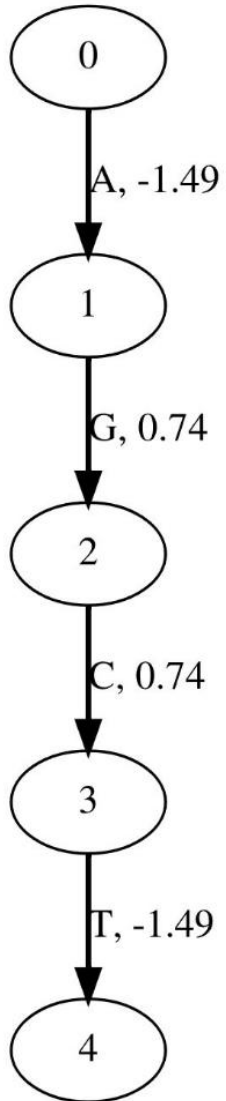


How should you structure your data in your program?

What would be useful?
An easy way to go from child to parent? Parent to child? From node to incoming edges? Outgoing edges?



Linked List from DNA



Sequence: AGCT

Scoring File:

A = -1.49

T = -1.49

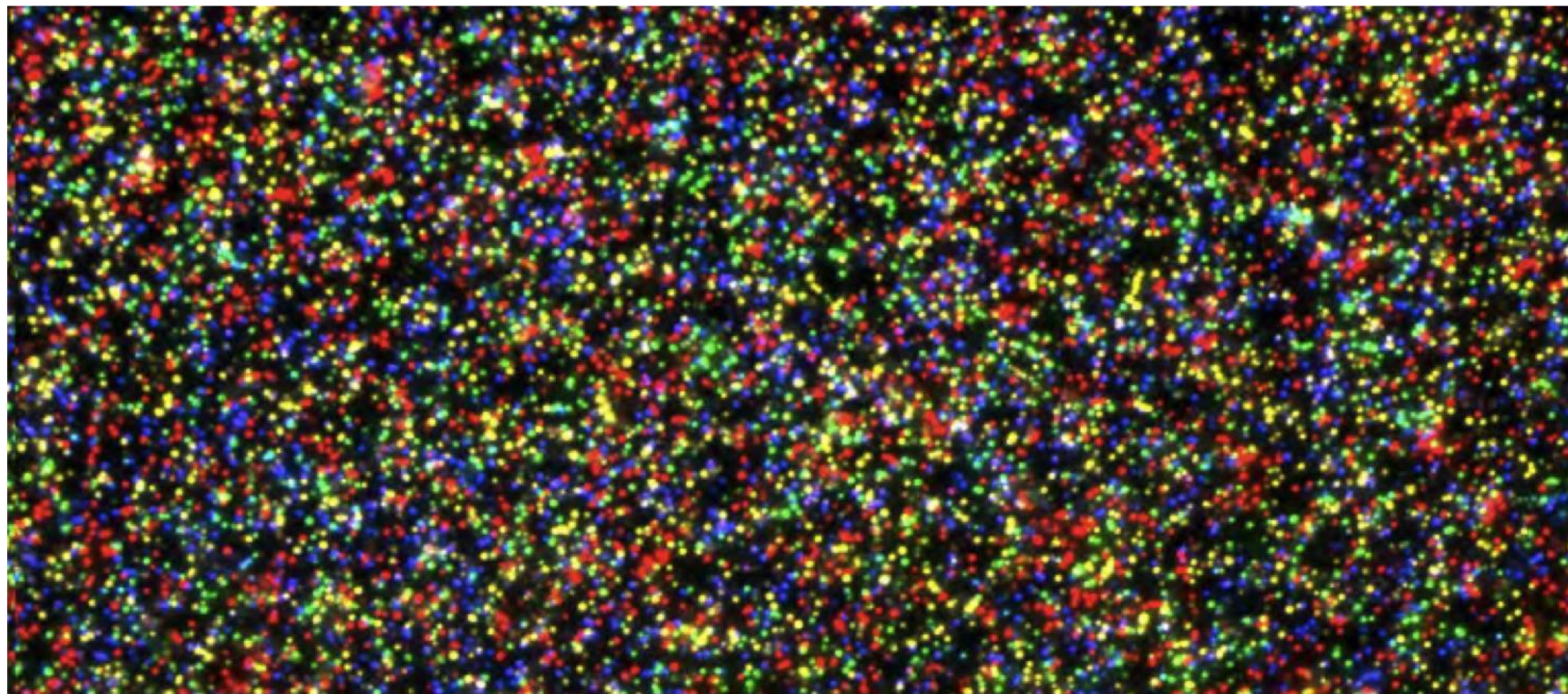
C = 0.74

G = 0.74

What will the
corresponding WDAG
file look like?

Raw Data (Illumina)

Raw Data (Illumina)



File Formats: Bcl

```
* BCL Files are base call and quality score binary files containing a (base,quality) pair for successive clusters.
* The file is structured as followed:
* Bytes 1-4 : unsigned int numClusters
* Bytes 5-numClusters + 5 : 1 byte base/quality score
* <p/>
* The base/quality scores are organized as follows (with one exception, SEE BELOW):
* The right 2 most bits (these are the LEAST significant bits) indicate the base, where
* A=00(0x00), C=01(0x01), G=10(0x02), and T=11(0x03)
* <p/>
* The remaining bytes compose the quality score which is an unsigned int.
* <p/>
* EXCEPTION: If a byte is entirely 0 (e.g. bytesRead == 0) then it is a no call, the base
* becomes '.' and the Quality becomes 2, the default illumina masking value
* <p/>
* (E.g. if we get a value in binary of 10001011 it gets transformed as follows:
* <p/>
* Value read: 10001011(0x8B)
* <p/>
* Quality      Base
* <p/>
* 100010      11
* 00100010    0x03
* 0x22        T
* 34          T
* <p/>
* So the output base/quality will be a (T/34)
*/
```

Intermediate file
format. Basically
always just go straight
to fastqs.

File Formats: Fasta

```
>MCHU - Calmodulin - Human, rabbit, bovine, rat, and chicken
MADQLTEEQIAEFKEAFSLFDKDGDTITTKELGTVMRSLGQNPTEAELQDMINEVDADGNGTID
FPEFLTMMARKMKDSTDSEEEIREAFRVFDKDGNGYISAAELRHVMTNLGEKLTDEEVDEMIREA
DIDGDGQVNYEEFVQMMTAK*
```

```
;LCB0 - Prolactin precursor - Bovine
; a sample sequence in FASTA format
MDSKGSSQKGSRLLLLLVSNLLLCQGVVSTPVCNPGPGNCQVSLRDLFDRAMVSHYIHDLS
EMFNEFDKRYAQKGKGFITMALNSCHTSSLPTPEDKEAQQTHHEVLSLILGLLRWNPPLYHL
VTEVRGMKGAPDAILSRATIEEEENKRLLEGMEMIFGQVIPGAKETEPYPVWSGLPSLQTKDED
ARYSAFYNLLHCLRRDSSKIDTYLKLNCRIIYNINC*
```

```
>MCHU - Calmodulin - Human, rabbit, bovine, rat, and chicken
MADQLTEEQIAEFKEAFSLFDKDGDTITTKELGTVMRSLGQNPTEAELQDMINEVDADGNGTID
FPEFLTMMARKMKDSTDSEEEIREAFRVFDKDGNGYISAAELRHVMTNLGEKLTDEEVDEMIREA
DIDGDGQVNYEEFVQMMTAK*
```

```
>gi|5524211|gb|AAD44166.1| cytochrome b [Elephas maximus maximus]
LCLYTHIGRNIYGSYLYSETWNTGIMLLITMATAFMGYVLPWGQMSFWGATVITNLFSaipYIGTNLV
EWIWGGFSVDKATLNRFFAFHFILPFTMVALAGVHLTFLHETGSNNPLGLTSDSDKIPFHPYYTIKDFLG
LLILILLLLLLALLSPDMLGDPDNHMPADPLNTPLHIKPEWYFLFAYAILRSVPNKLGGVLALFLSIVIL
GLMPFLHTSKHRSMMLRPLSQALFWTLTMDLLTLTWIGSQPVEYPYTIIGQMASILYFSIILAFPLIAGX
IENY
```

File Formats: Fastq

```
@SEQ_ID
GATTTGGGGTTCAAAGCAGTATCGATCAAATAGTAAATCCATTTGTTCAACTCACAGTTT
+
!' '*((( (**+))%%%++) (%%%) .1***-+*' ' ) ) **55CCF>>>>>CCCCCCC65
```

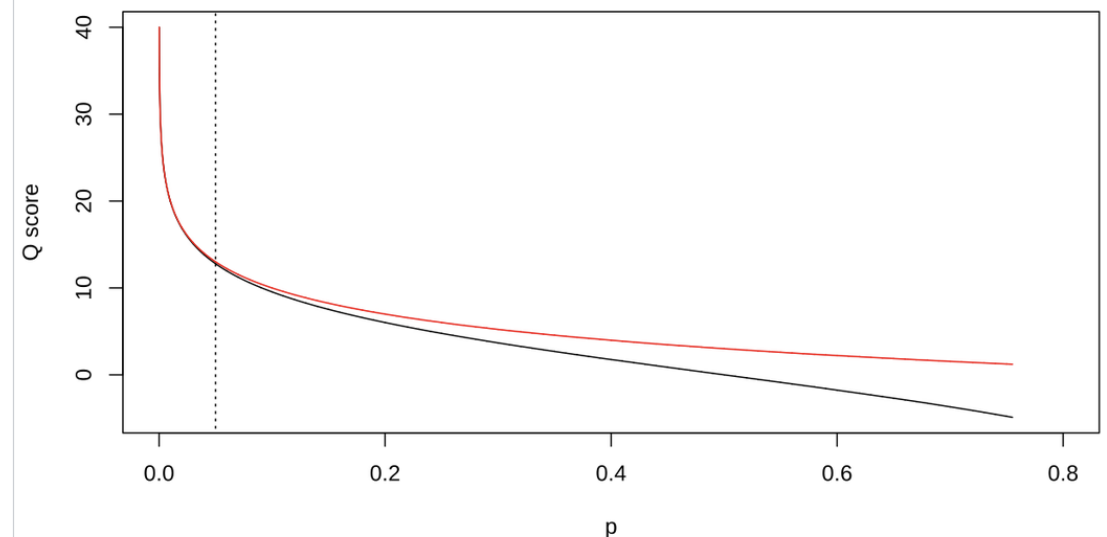
```
!"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|}~
```

0x21 (!) -> lowest quality

0x7e(~) -> highest quality

$$Q = -\log_{10}(p)$$

$p = \text{prob}(\text{wrong base call})$



File Formats: Fastq

```
@SEQ_ID
GATTTGGGGTTCAAAGCAGTATCGATCAAATAGTAAATCCATTTGTTCAACTCACAGTTT
+
!' '*((( (***) )%%%++) (%%%) .1***-+*' ' ) )**55CCF>>>>>CCCCCCCC65
```

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

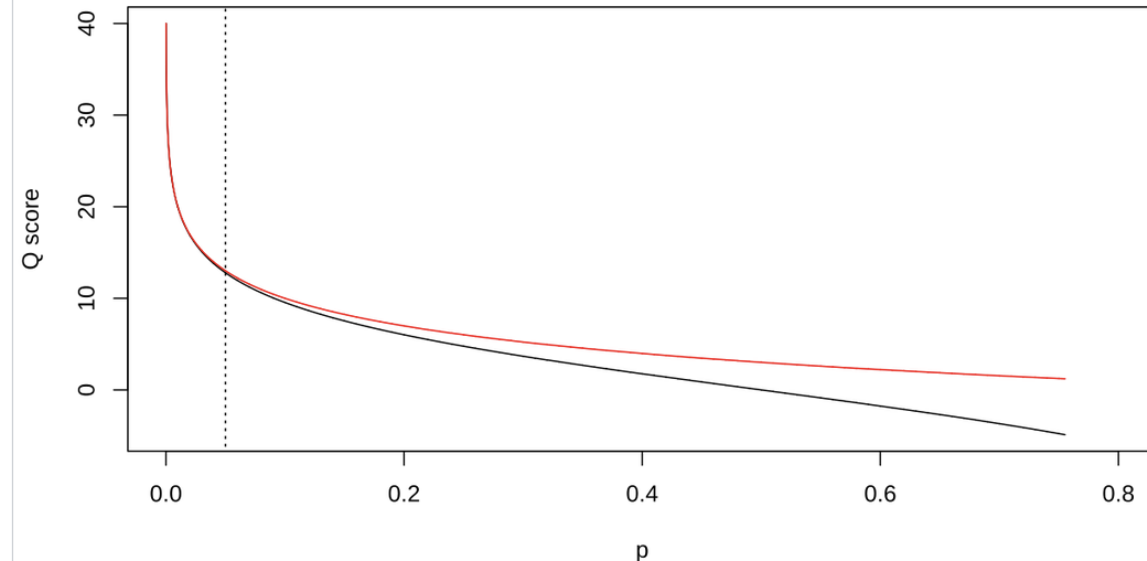
File Formats: Fastq

```
@SEQ_ID
GATTTGGGGTTCAAAGCAGTATCGATCAAATAGTAAATCCATTGTGTTCAACTCACAGTTT
+
!' '*((( (**+))%%%++) (%%%) .1***-+*' '))**55CCF>>>>>CCCCCCC65
```

$$Q = 10^* -\log_{10}(p)$$

$p = \text{prob}(\text{wrong base call})$

0 -> 0x21 (!) -> lowest quality
93 -> 0x7e(~) -> highest quality
(max usually around 60)



File Formats: Fastq

```
$ fastq-dump -X 2 SRR001666 --split-3
Read 2 spots for SRR001666
Written 2 spots for SRR001666
$ head SRR001666_1.fastq SRR001666_2.fastq
==> SRR001666_1.fastq <==
@SRR001666.1 071112_SLXA-EAS1_s_7:5:1:817:345 length=36
GGGTGATGGCCGCTGCCGATGGCGTCAAATCCCACC
+SRR001666.1 071112_SLXA-EAS1_s_7:5:1:817:345 length=36
IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII9IG9IC
@SRR001666.2 071112_SLXA-EAS1_s_7:5:1:801:338 length=36
GTTTCAGGGATACGACGTTTGTATTTTAAGAATCTGA
+SRR001666.2 071112_SLXA-EAS1_s_7:5:1:801:338 length=36
IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII6IBI

==> SRR001666_2.fastq <==
@SRR001666.1 071112_SLXA-EAS1_s_7:5:1:817:345 length=36
AAGTTACCCTTAACAACCTTAAGGGTTTTCAAATAGA
+SRR001666.1 071112_SLXA-EAS1_s_7:5:1:817:345 length=36
IIIIIIIIIIIIIIIIIIIIIIIDIIIIIII>IIIIII/
@SRR001666.2 071112_SLXA-EAS1_s_7:5:1:801:338 length=36
AGCAGAAGTCGATGATAATACGCGTCGTTTTATCAT
+SRR001666.2 071112_SLXA-EAS1_s_7:5:1:801:338 length=36
IIIIIIIIIIIIIIIIIIIIIIIGII>IIIII-I)8I
```

```
@EAS139:136:FC706VJ:2:2104:15343:197393 1:Y:18:ATCACG
```

EAS139	the unique instrument name
136	the run id
FC706VJ	the flowcell id
2	flowcell lane
2104	tile number within the flowcell lane
15343	'x'-coordinate of the cluster within the tile
197393	'y'-coordinate of the cluster within the tile
1	the member of a pair, 1 or 2 (<i>paired-end or mate-pair reads only</i>)
Y	Y if the read is filtered (did not pass), N otherwise
18	0 when none of the control bits are on, otherwise it is an even number
ATCACG	index sequence

File Formats: SAM

[illegible]

Col	Field	Type	Regex/Range	Brief description
1	QNAME	String	[!-?A-~]{1,254}	Query template NAME
2	FLAG	Int	[0,2 ¹⁶ -1]	bitwise FLAG
3	RNAME	String	* [!-()+-<>-~] [!-~]*	Reference sequence NAME
4	POS	Int	[0,2 ³¹ -1]	1-based leftmost mapping POSition
5	MAPQ	Int	[0,2 ⁸ -1]	MAPping Quality
6	CIGAR	String	* ([0-9]+[MIDNSHPX=])+	CIGAR string
7	RNEXT	String	* = [!-()+-<>-~] [!-~]*	Ref. name of the mate/next read
8	PNEXT	Int	[0,2 ³¹ -1]	Position of the mate/next read
9	TLEN	Int	[-2 ³¹ +1,2 ³¹ -1]	observed Template LENgth
10	SEQ	String	* [A-Za-z=.] +	segment SEQUENCE
11	QUAL	String	[!-~]+	ASCII of Phred-scaled base QUALity+33

File Formats: SAM

For example:

```
RefPos:      1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19
Reference:   C  C  A  T  A  C  T  G  A  A  C  T  G  A  C  T  A  A  C
Read:  ACTAGAATGGCT
```

Aligning these two:

```
RefPos:      1  2  3  4  5  6  7      8  9 10 11 12 13 14 15 16 17 18 19
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
```

With the alignment above, you get:

```
POS: 5
CIGAR: 3M1I3M1D5M
```

Op	Description
M	Match (alignment column containing two letters). This could contain two different letters (mismatch) or two identical letters. USEARCH generates CIGAR strings containing Ms rather than X's and ='s (see below).
D	Deletion (gap in the target sequence).
I	Insertion (gap in the query sequence).
S	Segment of the query sequence that does not appear in the alignment. This is used with soft clipping, where the full-length query sequence is given (field 10 in the SAM record). In this case, S operations specify segments at the start and/or end of the query that do not appear in a local alignment.
H	Segment of the query sequence that does not appear in the alignment. This is used with hard clipping, where only the aligned segment of the query sequences is given (field 10 in the SAM record). In this case, H operations specify segments at the start and/or end of the query that do not appear in the SAM record.
=	Alignment column containing two identical letters. USEARCH can read CIGAR strings using this operation, but does not generate them.
X	Alignment column containing a mismatch, i.e. two different letters. USEARCH can read CIGAR strings using this operation, but does not generate them.

File Formats: BED

```
chr1 213941196 213942363
chr1 213942363 213943530
chr1 213943530 213944697
chr2 158364697 158365864
chr2 158365864 158367031
chr3 127477031 127478198
chr3 127478198 127479365
chr3 127479365 127480532
chr3 127480532 127481699
```

1. Name
 - Label to be displayed under the feature, if turned on in “Configure this page”.
2. Score
 - A score between 0 and 1000. See [\[track lines\]\(http://asia.ensembl.org/info/website/upload/bed.html#track-lines\)](http://asia.ensembl.org/info/website/upload/bed.html#track-lines), below, for ways to configure the display style of scored data.
3. Strand
 - defined as + \((forward\)) or – \((reverse\))
4. thickStart
 - coordinate at which to start drawing the feature as a solid rectangle
5. thickEnd
 - coordinate at which to stop drawing the feature as a solid rectangle
6. itemRgb
 - an RGB colour value \((e.g. 0,0,255\))\). Only used if there is a track line with the value of itemRgb set to “on” \((case-insensitive\))\).
7. blockCount
 - 1. the number of sub-elements \((e.g. exons\)) within the feature
8. blockSizes
 - the size of these sub-elements
9. blockStarts
 - the start coordinate of each sub-element

File Formats: BED

```
chr7 127471196 127472363 Pos1 0 + 127471196 127472363 255,0,0
chr7 127472363 127473530 Pos2 0 + 127472363 127473530 255,0,0
chr7 127473530 127474697 Pos3 0 + 127473530 127474697 255,0,0
chr7 127474697 127475864 Pos4 0 + 127474697 127475864 255,0,0
chr7 127475864 127477031 Neg1 0 - 127475864 127477031 0,0,255
chr7 127477031 127478198 Neg2 0 - 127477031 127478198 0,0,255
chr7 127478198 127479365 Neg3 0 - 127478198 127479365 0,0,255
chr7 127479365 127480532 Pos5 0 + 127479365 127480532 255,0,0
chr7 127480532 127481699 Neg4 0 - 127480532 127481699 0,0,255
```

1. Name

- Label to be displayed under the feature, if turned on in “Configure this page”.

2. Score

- A score between 0 and 1000. See [track lines](<http://asia.ensembl.org/info/website/upload/bed.html#track-lines>), below, for ways to configure the display style of scored data.

3. Strand

- defined as + \ (forward) or - \ (reverse).

4. thickStart

- coordinate at which to start drawing the feature as a solid rectangle

5. thickEnd

- coordinate at which to stop drawing the feature as a solid rectangle

6. itemRgb

- an RGB colour value \ (e.g. 0,0,255\). Only used if there is a track line with the value of itemRgb set to “on” \ (case-insensitive).

7. blockCount

- 1. the number of sub-elements \ (e.g. exons) within the feature

8. blockSizes

- the size of these sub-elements

9. blockStarts

- the start coordinate of each sub-element