Lab 1 - Gr. 14 - Bioinformatics (732A93)

tbd

Contents

Assignment 1

Question 1: Hardy–Weinberg equilibrium

Question 1.1

We define the following probability space

$$(\Omega, \mathcal{F}, P)$$

Where,

$$\Omega = \{(A,A),(A,a),(a,A),(a,a)\}$$

We also define a probability measure P such as

$$P(X,Y)$$
$$X,Y \in \{A,a\}$$

So $P(X,Y)$ is the probability that an allele is (X,Y).

By definition, p is the proportion of $A's$ in the allele population and q is the proportion of $a's$ in the allele population, so:

$$P(A) = P(A,A) + P(A,a) + P(a,A) = p$$

The same applies to $P(a)$.

$$P(a) = P(a,a) + P(a,A) + P(A,a) = q$$

The fact that we asume random mating means that X and Y are IID, which entails the following:

$$P(A,A) = P(A) * P(A) = p^2$$
$$P(a,a) = P(a) * P(a) = q^2$$
$$P(A,a) = P(a,A) = P(A) * P(a) = pq$$

The probability of an allele of it being a heterozygote is:

$$P(A,a) + P(a,A) = 2pq$$

$$P(\Omega) = P(A,A) + P(a,a) + 2P(A,a) = p^2 + q^2 + 2pq = 1$$

Thus, we show that by random mating the proportion of $A's$ and $a's$ is the same in the offsprings and the Hardy-Weinberg equilibrium is obtained and can't deviate from it.

Question 1.2

We now look at the MN blood group, that has two possible co–dominating alleles, L^M (denoted M) and L^N (denoted N). In a population of $n = 1000$ Americans of Caucasian descent, the following genotype counts were observed:

- $n_{MM} = 357$ individuals were MM homozygotes;
- $n_{MN} = 485$ individuals were MN heterozygotes;
- $n_{NN} = 158$ individuals were NN homozygotes;

The relatives proportion, obtained by dividing the genotype counts by the total population, are 0.357, 0.485 and 0.158 respectively. According to the Hardy–Weinberg principle, we expect these quantities to follow the proportions p^2, $2pq$ and q^2, where p and q are the proportions of N and M in the alleles. In formulas:

$$\begin{aligned} p &= \frac{n_{MM}}{n} + \frac{1}{2} \cdot \frac{n_{MN}}{n} = 0.357 + \frac{1}{2} \cdot 0.485 = 0.5995 \\ q &= \frac{n_{NN}}{n} + \frac{1}{2} \cdot \frac{n_{MN}}{n} = 0.158 + \frac{1}{2} \cdot 0.485 = 0.4005 \end{aligned} \tag{1}$$

With these empirical quantities we formulate what would the population look if it was on equilibrium and compare them with the real proportions ($p_{MM} = 0.375, p_{NN} = 0.158, p_{MN} = 0.485$):

$$\begin{aligned} p^0_{MM} &= p^2 = 0.3594002 \\ p^0_{NN} &= q^2 = 0.1604002 \\ p^0_{MN} &= 2pq = 0.4801995 \end{aligned}$$

Performing a chi–square goodness of fit test (`chisq.test()` function in R) results in *p-value* = 1. This result shows that there is not enough statistical evidence to reject the hypothesis that the population is in a Hardy-Weinberg equilibrium.

```
## Warning in chisq.test(c(0.357, 0.485, 0.158), p = c(p_2, pq, q_2)): Chi-
## squared approximation may be incorrect
```

```
##
##  Chi-squared test for given probabilities
##
## data:  c(0.357, 0.485, 0.158)
## X-squared = 9.9938e-05, df = 2, p-value = 1
```

Assignment 2 : Exploring a genomic sequence

Note that, by convention, a *coding strand* is used when displaying a DNA sequence. "A coding strand, is the segment within double-stranded DNA that runs from 5' to 3', and which is complementary to the antisense strand of DNA, or template strand, which runs from 3' to 5"' (https://en.wikipedia.org/wiki/Sense_strand).

Question 2.1

We go to the following link https://www.ncbi.nlm.nih.gov/nuccore/CU329670, scroll down to "Features", click on "CDS" to see the first 5662 nucleotides of the sequence.

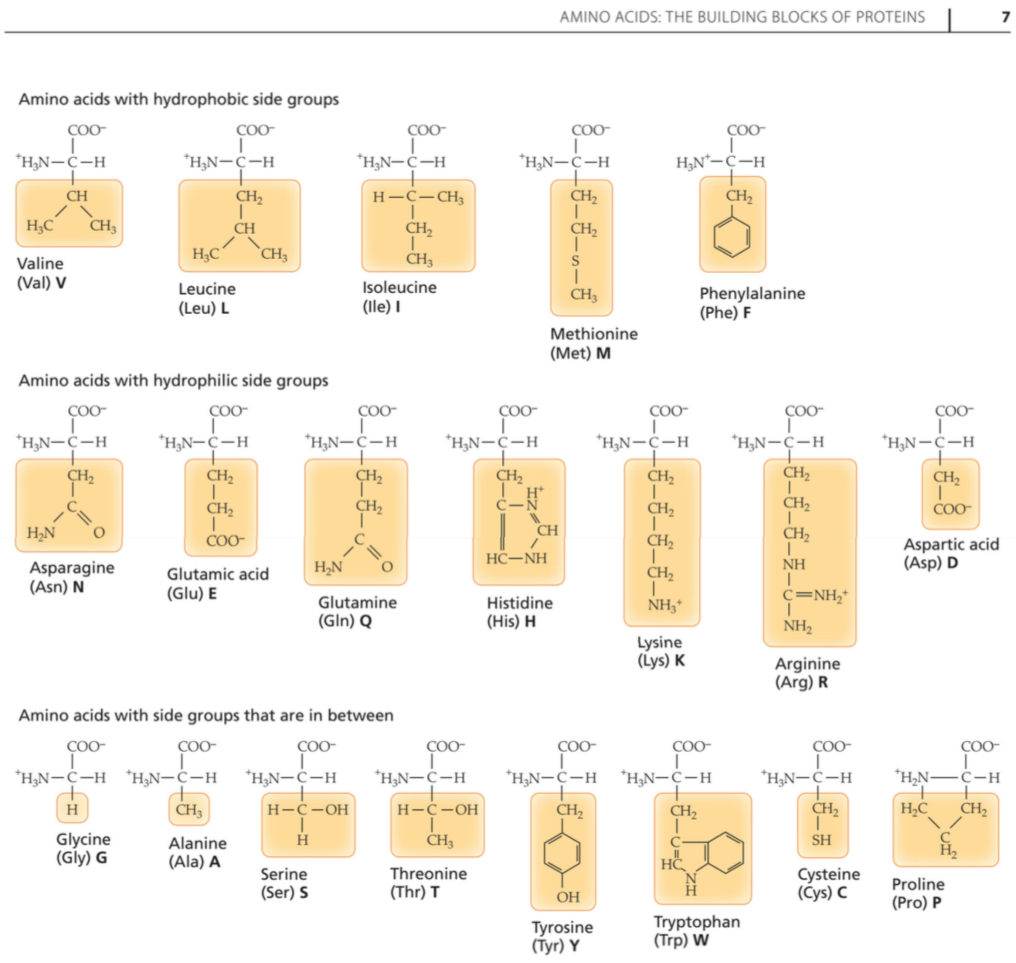
The protein product is: **RecQ type DNA helicase**

Question 2.2

Note that proteins are made up of 20 different amino acids (linked together). Amino acids are molecules composed of an alpha carbon, a carboxyl group, an amino group and a side chain. The side chain is what makes an amino acid unique (see p. 5, Concepts in Bioinformatics and Genomics). The 20 different amino acids can be found in the picture below.

In our case, the first four amino acids are:

1. M (= **Methionine**, coded by ATG, representing the starting sequence)
2. V (= **Valine**)
3. V (= **Valine**)
4. A (= **Alanine**)

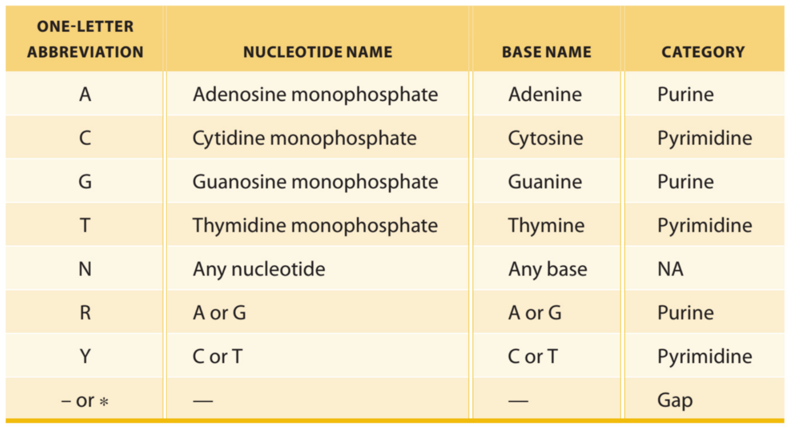


Question 2.3

Note that nucleotides code for amino acids. They are molecules composed of a sugar, a phosphate and a base. The part of nucleotides that distinguishes them is the base.

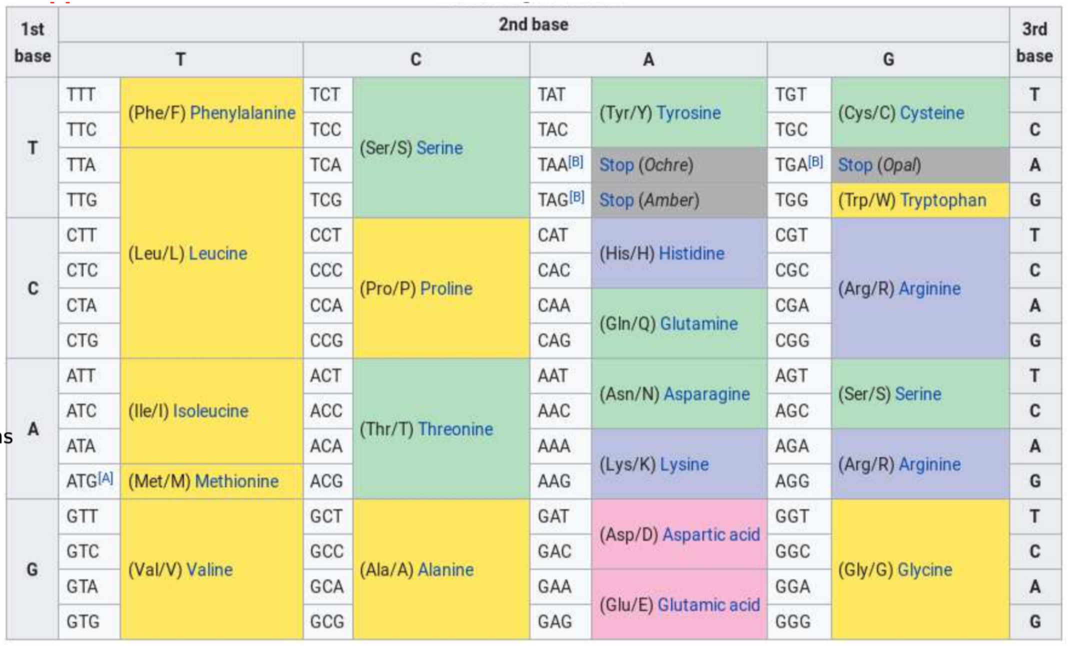
There are four possible bases: A (Adenine), C (Cytosine), G (Guanine), T (Thymine)

TABLE 1-1. SINGLE-LETTER ABBREVIATIONS USED FOR DNA NUCLEOTIDE SEQUENCES



ONE-LETTER ABBREVIATION	NUCLEOTIDE NAME	BASE NAME	CATEGORY
A	Adenosine monophosphate	Adenine	Purine
C	Cytidine monophosphate	Cytosine	Pyrimidine
G	Guanosine monophosphate	Guanine	Purine
T	Thymidine monophosphate	Thymine	Pyrimidine
N	Any nucleotide	Any base	NA
R	A or G	A or G	Purine
Y	C or T	C or T	Pyrimidine
– or ∗	—	—	Gap

Specific combinations of these nucleotides code for specific amino acids. In genetic code, three nucleotides (called codon) always code for one amino acid. E.g. ATG codes for met (Methionine) which is a starting point of a protein. Which codos code for which amino acids is shown in the overview below:



1st base	2nd base								3rd base
	T		C		A		G		
T	TTT	(Phe/F) Phenylalanine	TCT	(Ser/S) Serine	TAT	(Tyr/Y) Tyrosine	TGT	(Cys/C) Cysteine	T
	TTC		TCC		TAC		TGC		C
	TTA	(Leu/L) Leucine	TCA		TAA[B]	Stop (*Ochre*)	TGA[B]	Stop (*Opal*)	A
	TTG		TCG		TAG[B]	Stop (*Amber*)	TGG	(Trp/W) Tryptophan	G
C	CTT		CCT	(Pro/P) Proline	CAT	(His/H) Histidine	CGT	(Arg/R) Arginine	T
	CTC		CCC		CAC		CGC		C
	CTA		CCA		CAA	(Gln/Q) Glutamine	CGA		A
	CTG		CCG		CAG		CGG		G
A	ATT	(Ile/I) Isoleucine	ACT	(Thr/T) Threonine	AAT	(Asn/N) Asparagine	AGT	(Ser/S) Serine	T
	ATC		ACC		AAC		AGC		C
	ATA		ACA		AAA	(Lys/K) Lysine	AGA	(Arg/R) Arginine	A
	ATG[A]	(Met/M) Methionine	ACG		AAG		AGG		G
G	GTT	(Val/V) Valine	GCT	(Ala/A) Alanine	GAT	(Asp/D) Aspartic acid	GGT	(Gly/G) Glycine	T
	GTC		GCC		GAC		GGC		C
	GTA		GCA		GAA	(Glu/E) Glutamic acid	GGA		A
	GTG		GCG		GAG		GGG		G

Saving the nucleotide sequence from GenBank:

The complete nucleotide sequence of the coding strand from GenBank (https://www.ncbi.nlm.nih.gov/nuccore/CU329670.1?from=1&to=5662) was saved in FASTA format as **2.3_nucleotid-sequence.FASTA**. Note that only the last 12 characters AGCGACGACCAT actually correspond to the amino acids MVVA if the reverse compliment is taken (see 2.4).

Using backtrack to obtain the amino acid sequence from the protein sequence:

We used backtrack to obtain the amino acid sequence corresponding to the protein sequence `MVVA` (https://www.ebi.ac.uk/Tools/st/emboss_backtranseq/). After pasting `MVVA` and selecting “Schizosaccharomyces pombe (CAI equivalent)”, the following sequence is returned: **ATGGTCGTCGCT**

Question 2.4

The above mentioned obtained coding strand sequence ATGGTCGTCGCT does not exist in the provided nucleotide sequence (that was saved in FASTA format as 2.3_nucleotid-sequence.FASTA) since this sequence is not found when searching the file.

Option 1

However, we can modify the displayed sequence to get what we are looking for. On GenBank, under "Display options", one has to select both "Show sequence" and **"Show reverse complement"** (https://www.ncbi.nlm.nih.gov/nuccore/CU329670.1?from=1&to=5662). Afterwards, the displayed nucleotid sequence under "ORIGIN" starts with ATGGTCGTCGCT which codes for our amino acids MVVA.

Option 2

Alternatively, one can take the last 12 characters from the original nucleotid sequence (the one in the FASTA file, i.e. the sequence before selecting "Show reverse complement" on GenBank). These characters are: AGCGACGACCAT. Copy this string and paste it here: http://arep.med.harvard.edu/labgc/adnan/projects/Utilities/revcomp.html Then click **"reverse complement"**. The result is again the sequence that we are looking for: ATGGTCGTCGCT which codes for our amino acids MVVA.

Conclusion

ATGGTCGTCGCT is exactly what we got in 2.3 using backtranseq to obtain the amino acid sequence from the protein sequence `MVVA`. The only tricky thing here was that we needed to take the reverse complement of the correct characters from the nucleotid sequence (namely the last 12 characters).

Question 2.5

Number range that corresponds to these amino acids (protein sequence):

In the saved FASTA file, the *last 12 characters* AGCGACGACCAT correspond to the amino acids MVVA (after taking the reverse complement). The number range corresponding to them is 5651 to 5662.

If we had selected "Show reverse complement" on GenBank before, then the *first 12 characters* ATGGTCGTCGCT would correspond to the amino acids MVVA. The number range would then be 1 to 12.

Stop codon in the nucleotide sequence:

Note that there are three possible stop codons: TAA, TAG, TGA. Still, it is not easy to identify the stop codon manually. However, one can use: https://www.ncbi.nlm.nih.gov/orffinder and paste the complete (reverse complement) nucleotide sequence in FASTA format. With the default parameters, this is automatically translated to the correct amino acid sequence. Also, after scolling down a bit, one can see that the last stop happens at 5661. We find TC from 5661 to 5662 (and at 5662, we have the last nucleotid). From 5658 to 5660, we have TGA which is one of the above mentioned stop codons. We can conclude that the stop codon in the nucleotid sequence is hence TGA from 5658 to 5660 (looking at the reverse complement nucleotid sequence).

Chromosome on which the genomic sequence lies:

We can see on GenBank (in the header under the definition as well as in the features under source), that this genomic sequence lies on the chromosome 1 of schizosaccharomyces pombe.

Assignment 3 : Exploring a genomic sequence

Question 3.1

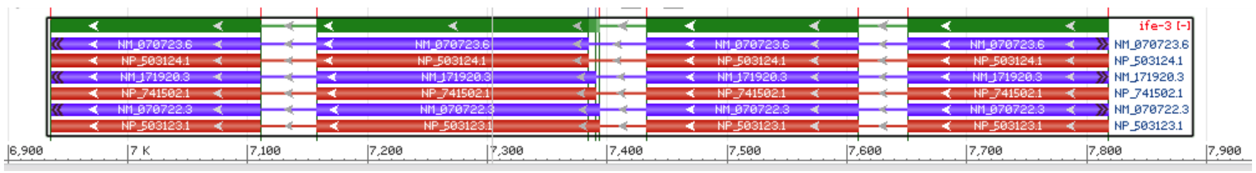
Caenorhabditis elegans is a free-living, nematode worm, which lives in soil environments all over the world. it is divided in two sexes:

i) male
ii) self-fertilizing hermaphrodite

Finally, it has all human sensations, such as taste and smell, despite the fact that is has no eyes. (https://en.wikipedia.org/wiki/Caenorhabditis_elegans)

In general C.elegans has a similar construction and characteristics as humans. It is also important that the genes, which are responsible for human's evolution, had similar ancestor with C.elegans. Therefore, instead of making experiments on humans, which most of the time is expensive and difficult, scientist can work with C.elegans. That is the reason why studying C.elegans biology is crucial for scientific field. (http://www.people.ku.edu/~erikl/Lundquist_Lab/Why_study_C._elegans.html)

Question 3.2

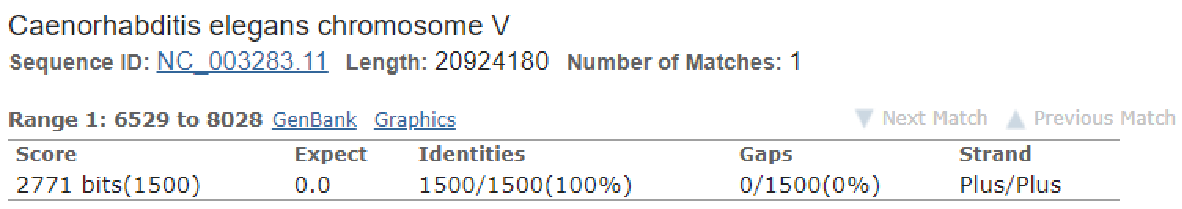


This diagram shows exons and introns. We have 4 exons in our searching query. Our searching query found between 6529 and 8028. Our exons and introns:

- introns1: 6529 - 6935
- exon1: 6936 - 7110
- introns2: 7111 - 7157
- exon2: 7158 - 7393
- introns3: 7394 - 7432
- exon3: 7433 - 7609
- introns4: 7610 - 7650
- exon4: 7651 - 7818
- introns5: 7819 - 8028

Question 3.3

We can find all the information needed from the summary :


Caenorhabditis elegans chromosome V
Sequence ID: NC_003283.11 Length: 20924180 Number of Matches: 1

Range 1: 6529 to 8028 GenBank Graphics

Score	Expect	Identities	Gaps	Strand
2771 bits(1500)	0.0	1500/1500(100%)	0/1500(0%)	Plus/Plus

The numbering of the sequences in the alignment is 1500. The identities section in the summary shows this result. In general, identities shows the number of identical bases between the query and the subject sequence.

The summary also conteins information about the orientation both of the query and the subject sequence, which can be found in "Strand" section.In this case the the database genomic sequence progress has the same direction as the query sequence because both of them are plus and they are increasing.

Question 3.4

Our query sequence is found at *"Caenorhabditis elegans chromosome V"* and its range is between 6529 and 8028. We can see chromosome name in the figure of question 3.3.

Question 3.5

We will find sequences of exons by this code:

```
## gets sequence, starting index and ending index as a parameter
## split and return the seqeunce between start and index
get_exon = function(seq, start, end){
  org_seq_start = 6529
  fst_index = start - org_seq_start + 1
  return(substr(seq, fst_index, fst_index + (end-start)))
}

## read the query sequence
fasta_file = readLines("732A51_BioinformaticsHT2018_Lab01Ex03.fasta")
```

```
## Warning in readLines("732A51_BioinformaticsHT2018_Lab01Ex03.fasta"):
## incomplete final line found on
## '732A51_BioinformaticsHT2018_Lab01Ex03.fasta'
```

```
query_sequence = paste(fasta_file[-1], collapse = "")

## Starting and end indexes of exons.
exon1_start = 6936
exon1_end = 7110

exon2_start = 7158
exon2_end = 7393

exon3_start = 7433
exon3_end = 7609

exon4_start = 7651
exon4_end = 7818
```

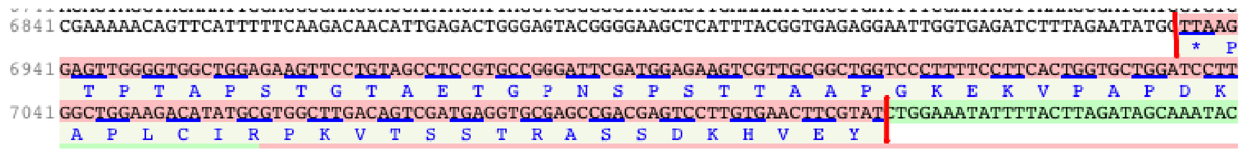
```r
## get exons with our split function
exon1 = get_exon(query_sequence, exon1_start, exon1_end)
exon2 = get_exon(query_sequence, exon2_start, exon2_end)
exon3 = get_exon(query_sequence, exon3_start, exon3_end)
exon4 = get_exon(query_sequence, exon4_start, exon4_end)

## save exons for protein analysis
writeLines(exon1, file("q3.5_exon1.txt"))
writeLines(exon2, file("q3.5_exon2.txt"))
writeLines(exon3, file("q3.5_exon3.txt"))
writeLines(exon4, file("q3.5_exon4.txt"))
```

exon 1

- **get_exon** function output:

```
## TTAAGGAGTTGGGGTGGCTGGAGAAGTTCCTGTAGCCTCCGTGCCGGGAT ...
```

- **Sequence Text View Tool** in blast nucleotide search



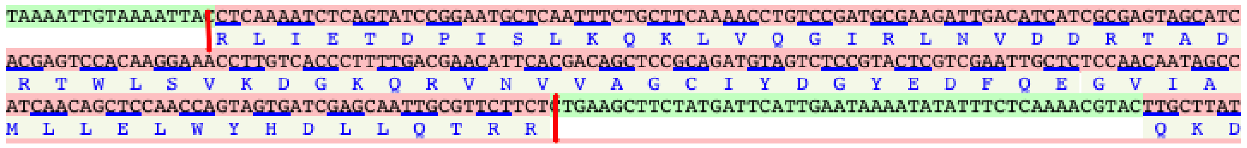
```
6841 CGAAAAACAGTTCATTTTTCAAGACAACATTGAGACTGGGAGTACGGGGAAGCTCATTTACGGTGAGAGGAATTGGTGAGATCTTTAGAATATGCTTAAG
                                                                                                       *  P
6941 GAGTTGGGGTGGCTGGAGAAGTTCCTGTAGCCTCCGTGCCGGGATTCGATGGAGAAGTCGTTGCGGCTGGTCCCTTTTCCTTCACTGGTGCTGGATCCTT
       T  P  T  A  P  S  T  G  T  A  E  T  G  P  N  S  P  S  T  T  A  A  P  G  K  E  K  V  P  A  P  D  K
7041 GGCTGGAAGACATATGCGTGGCTTGACAGTCGATGAGGTGCGAGCCGACGAGTCCTTGTGAACTTCGTATCTGGAAATATTTTACTTAGATAGCAAATAC
      A  P  L  C  I  R  P  K  V  T  S  S  T  R  A  S  S  D  K  H  V  E  Y
```

exon 2

- **get_exon** function output:

```
## CTCAAAATCTCAGTATCCGGAATGCTCAATTTCTGCTTCAAAACCTGTCC ...
```

- **Sequence Text View Tool** in blast nucleotide search



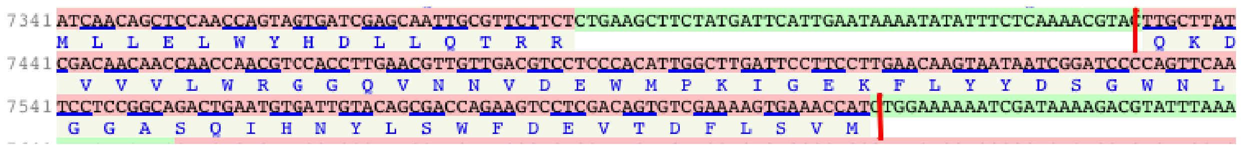
```
TAAAATTGTAAAATTACCTCAAAATCTCAGTATCCGGAATGCTCAATTTCTGCTTCAAAACCTGTCCGATGCGAAGATTGACATCATCGCGAGTAGCATC
                  R  L  I  E  T  D  P  I  S  L  K  Q  K  L  V  Q  G  I  R  L  N  V  D  D  R  T  A  D
ACGAGTCCACAAGGAAACCTTGTCACCCTTTTGACGAACATTCACGACAGCTCCGCAGATGTAGTCTCCGTACTCGTCGAATTGCTCTCCAACAATAGCC
 R  T  W  L  S  V  K  D  G  K  Q  R  V  N  V  V  A  G  C  I  Y  D  G  Y  E  D  F  Q  E  G  V  I  A
ATCAACAGCTCCAACCAGTAGTGATCGAGCAATTGCGTTCTTCTCTGAAGCTTCTATGATTCATTGAATAAAATATATTTCTCAAAACGTACTTGCTTAT
M  L  L  E  L  W  Y  H  D  L  L  Q  T  R  R                                                  Q  K  D
```

exon 3

- **get_exon** function output:

```
## TTGCTTATCGACAACAACCAACCAACGTCCACCTTGAACGTTGTTGACGT ...
```

- **Sequence Text View Tool** in blast nucleotide search



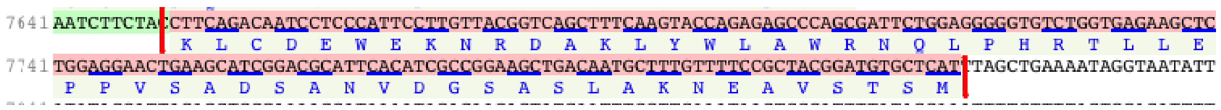
```
7341 ATCAACAGCTCCAACCAGTAGTGATCGAGCAATTGCGTTCTTCTCTGAAGCTTCTATGATTCATTGAATAAAATATATTTCTCAAAACGTACTTGCTTAT
     M  L  L  E  L  W  Y  H  D  L  L  Q  T  R  R                                                  Q  K  D
7441 CGACAACAACCAACCAACGTCCACCTTGAACGTTGTTGACGTCCTCCCACATTGGCTTGATTCCTTCCTTGAACAAGTAATAATCGGATCCCCAGTTCAA
       V  V  V  L  W  R  G  G  Q  V  N  N  V  D  E  W  M  P  K  I  G  E  K  F  L  Y  Y  D  S  G  W  N  L
7541 TCCTCCGGCAGACTGAATGTGATTGTACAGCGACCAGAAGTCCTCGACAGTGTCGAAAAGTGAAACCATCTGGAAAAAAATCGATAAAAGACGTATTTAAA
      G  G  A  S  Q  I  H  N  Y  L  S  W  F  D  E  V  T  D  F  L  S  V  M
```

exon 4

- **get_exon** function output:

```
## CTTCAGACAATCCTCCCATTCCTTGTTACGGTCAGCTTTCAAGTACCAGA ...
```

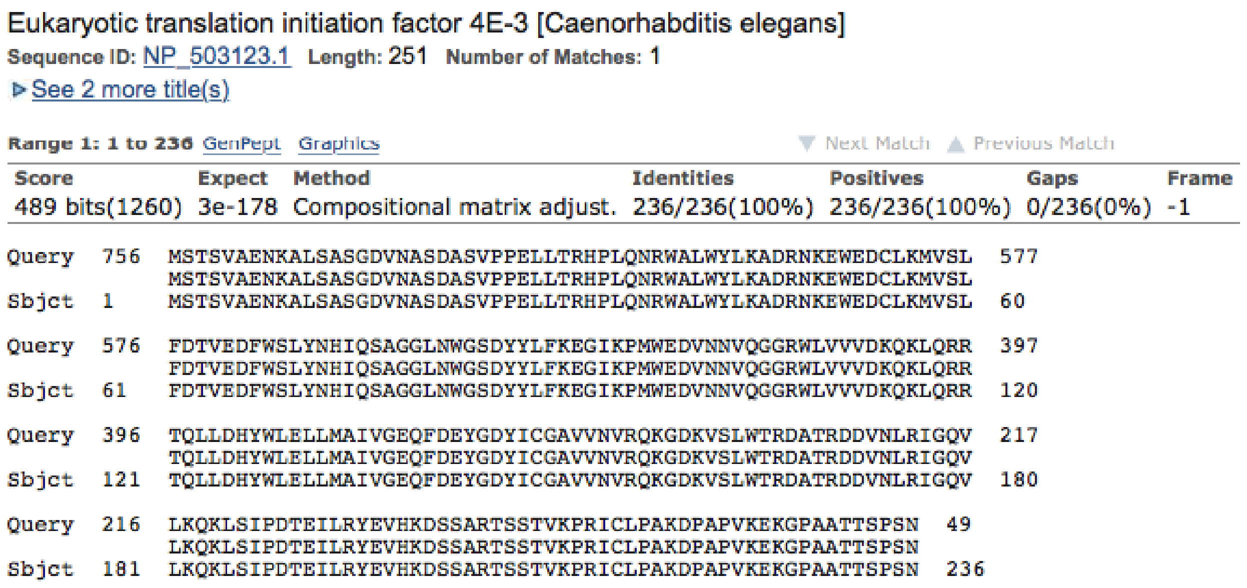
- **Sequence Text View Tool** in blast nucleotide search



```
7641 AATCTTCTACCTTCAGACAATCCTCCCATTCCTTGTTACGGTCAGCTTTCAAGTACCAGAGAGCCCAGCGATTCTGGAGGGGGTGTCTGGTGAGAAGCTC
             K  L  C  D  E  W  E  K  N  R  D  A  K  L  Y  W  L  A  W  R  N  Q  L  P  H  R  T  L  L  E
7741 TGGAGGAACTGAAGCATCGGACGCATTCACATCGCCGGAAGCTGACAATGCTTTGTTTTCCGCTACGGATGTGCTCATTTAGCTGAAAATAGGTAATATT
      P  P  V  S  A  D  S  A  N  V  D  G  S  A  S  L  A  K  N  E  A  V  S  T  S  M
```

After getting whole DNA code from all exons, we can search it on blastx to find which protein it is and its protein sequence. We can also see the protein sequence in the screenshots of *Sequence Text View Tool* in blast nucleotide search for each exons.

- **blastx**



Eukaryotic translation initiation factor 4E-3 [Caenorhabditis elegans]

Sequence ID: NP_503123.1 Length: 251 Number of Matches: 1

See 2 more title(s)

Range 1: 1 to 236 GenPept Graphics Next Match Previous Match

Score	Expect	Method	Identities	Positives	Gaps	Frame
489 bits(1260)	3e-178	Compositional matrix adjust.	236/236(100%)	236/236(100%)	0/236(0%)	-1

```
Query  756   MSTSVAENKALSASGDVNASDASVPPELLTRHPLQNRWALWYLKADRNKEWEDCLKMVSL  577
             MSTSVAENKALSASGDVNASDASVPPELLTRHPLQNRWALWYLKADRNKEWEDCLKMVSL
Sbjct  1     MSTSVAENKALSASGDVNASDASVPPELLTRHPLQNRWALWYLKADRNKEWEDCLKMVSL  60

Query  576   FDTVEDFWSLYNHIQSAGGLNWGSDYYLFKEGIKPMWEDVNNVQGGRWLVVVDKQKLQRR  397
             FDTVEDFWSLYNHIQSAGGLNWGSDYYLFKEGIKPMWEDVNNVQGGRWLVVVDKQKLQRR
Sbjct  61    FDTVEDFWSLYNHIQSAGGLNWGSDYYLFKEGIKPMWEDVNNVQGGRWLVVVDKQKLQRR  120

Query  396   TQLLDHYWLELLMAIVGEQFDEYGDYICGAVVNVRQKGDKVSLWTRDATRDDVNLRIGQV  217
             TQLLDHYWLELLMAIVGEQFDEYGDYICGAVVNVRQKGDKVSLWTRDATRDDVNLRIGQV
Sbjct  121   TQLLDHYWLELLMAIVGEQFDEYGDYICGAVVNVRQKGDKVSLWTRDATRDDVNLRIGQV  180

Query  216   LKQKLSIPDTEILRYEVHKDSSARTSSTVKPRICLPAKDPAPVKEKGPAATTSPSN  49
             LKQKLSIPDTEILRYEVHKDSSARTSSTVKPRICLPAKDPAPVKEKGPAATTSPSN
Sbjct  181   LKQKLSIPDTEILRYEVHKDSSARTSSTVKPRICLPAKDPAPVKEKGPAATTSPSN  236
```

As we can see from output of blastx, all exons are used for producing *Eukaryotic translation initiation factor 4E-3 [Caenorhabditis elegans]* protein.
After DNA code translation, blastx compared our protein sequence with a negative direction. While our query sequence order is decreasing, out subject sequence order is increasing.

Appendix

```r
knitr::opts_chunk$set(fig.width = 7, fig.height = 3, echo = FALSE)

library(dplyr)
library(tidyr)
library(magrittr)
library(kableExtra)

# ---------------------------------------------------------------------------
# Question 1.2
# ---------------------------------------------------------------------------
# The prob of one allele is the proportion of the homozigote + half the proportion of the heterozygote
p = 0.357 + 0.485 / 2
q = 0.158 + 0.485 / 2

# Expected values (Hardy-Weinberg equilibrium)
p_2 = p^2
q_2 = q^2
pq = 2*p*q

Xsq = chisq.test(c(0.357, 0.485, 0.158), p=c(p_2, pq, q_2))
Xsq
# Xsq$observed
# Xsq$expected

knitr::include_graphics("images/2.2_amino-acids.png")
knitr::include_graphics("images/2.3_nucleotides.png")
knitr::include_graphics("images/2.3_codons-aminoacids")
## gets sequence, starting index and ending index as a parameter
## split and return the seqeunce between start and index
get_exon = function(seq, start, end){
  org_seq_start = 6529
  fst_index = start - org_seq_start + 1
  return(substr(seq, fst_index, fst_index + (end-start)))
}

## read the query sequence
fasta_file = readLines("732A51_BioinformaticsHT2018_Lab01Ex03.fasta")
query_sequence = paste(fasta_file[-1], collapse = "")

## Starting and end indexes of exons.
exon1_start = 6936
exon1_end = 7110

exon2_start = 7158
exon2_end = 7393

exon3_start = 7433
exon3_end = 7609

exon4_start = 7651
exon4_end = 7818

## get exons with our split function
exon1 = get_exon(query_sequence, exon1_start, exon1_end)
exon2 = get_exon(query_sequence, exon2_start, exon2_end)
exon3 = get_exon(query_sequence, exon3_start, exon3_end)
exon4 = get_exon(query_sequence, exon4_start, exon4_end)

## save exons for protein analysis
writeLines(exon1, file("q3.5_exon1.txt"))
writeLines(exon2, file("q3.5_exon2.txt"))
writeLines(exon3, file("q3.5_exon3.txt"))
writeLines(exon4, file("q3.5_exon4.txt"))
```

```r
cat(substr(exon1,1,50), "...")
cat(substr(exon2,1,50), "...")
cat(substr(exon3,1,50), "...")
cat(substr(exon4,1,50), "...")
```