



Earlham  
Institute

# Earlham Institute summer school on bioinformatics

25-29 July 2016

## Basic Bioinformatics Sessions

### Practical 2: Pairwise Alignment

Monday 18 July 2016

## Sensitive Pairwise Alignment

The purpose of this exercise is to look at some aspects of **Pairwise Sequence Alignment** using the most accurate methods available.

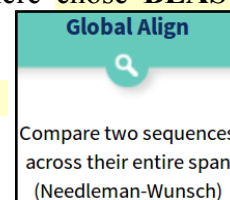
As hopefully has been discussed, sequences can be aligned using a **global** strategy, in which the two sequences being aligned are assumed to be homologous from end to end, or using a **local** approach, in which the sequences are assumed to just have homologous region(s).

### Global Pairwise Sequence Comparison

First the **global** approach. In a previous exercise, you already have used the **blast** facility at the **NCBI** to perform crude pairwise alignment. **blast** also offers a sensitive option, so maybe that would be a good place to start.

So, once more to the **NCBI** home page (<http://www.ncbi.nlm.nih.gov/>). From there chose **BLAST** from the

**Popular Resources** list. Scroll down to the **Specialized searches** section and chose the



option.

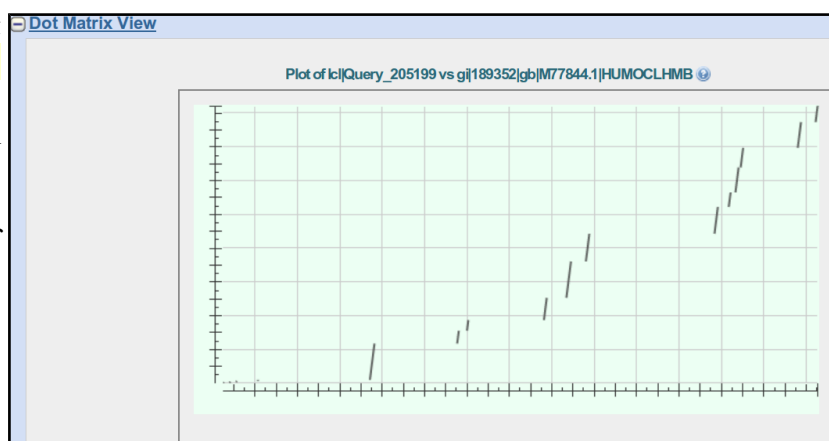
A choice of settings for **Nucleotide** or **Protein** alignment is offered. As we are going to investigate the alignment of DNA sequences, the default choice is fine. For the first sequence, browse for the file **pax6\_genomic.fasta**, which you created when looking at Ensembl. It contains the region of **Chromosome 11** containing the entire **PAX6** gene (with a few extra base pairs either end).

To specify the second sequence, you could load the file **pax6\_mrna.fasta**, but just typing the corresponding **Accession** code in the appropriate box seems far more sophisticated, so that is what I chose to do.

Open the **Algorithmic Parameters** section, and see that they are as one might expect. The defaults are fine here as the alignment to be computed is trivial (given the way **blast** will go about the task), so anything not outrageous should work.

Ask to **Show results in a new window** and then click on the **Align** button.

After some significant **Rollin' and Tumblin' blast** will proclaim its lyrical conclusions. First examine the **Dot Matrix View**. This sort of representation has rather gone out of fashion in recent years. A shame, I say, this picture represents such a succinct summary of what should be expected of the textual alignment(s) that are the “real” detailed output of this sort of program.



How would you interpret this picture?

What do the diagonal(ish) lines represent?

What are the gaps in between the lines?

Which axis represents the genomic sequence and which the mrna?

Move down to the textual alignment. There are some weird little bits and pieces at the front of the alignment which defy logic. I decide not to dwell on these too much, beyond noting that the mRNA has some odd bases at the front.

Also, I have faith that the alignment you look at yields the highest alignment score, but equally. I doubt most **people** would have chosen to throw these odd bases about with quite such abandon! **People** are best!

You can just see evidence of the little patches of whimsy in the **Dot Matrix View**.

Query	481	AAGTTAGCGCTGCTGAGCACCCCTTTCTTATCATTGACATTAAACTCTGGGCGAG	540
Sbjct	1	TATC-----	4
Query	541	GTCTCTGCGTAGAACGCGGCTGTGAGATCGCCACTTCCCTGCCGAGCGCGGTGAGAA	600
Sbjct		-----	
Query	601	GTGTGGGAACGCGCTGCCAGGCTCACCTGCCTCCCGCCCTCCGCTCCAGGTAACCG	660
Sbjct		-----	
Query	661	CCCGGCGCTCCGCCCCGCGCGGCTCGGGGCGCGGGGCTCTCCGCTGCCAGCGACTG	720
Sbjct		-----	
Query	721	CTGTCCCAAAATCAAAGCCCGCCCAAGTGGCGCCGGGCTTGATTTTGTCTTTAAAG	780
Sbjct		-----	
Query	781	GAGGCATACAAAGATGGAAGCGAGTTACTGAGGGAGGGATAGGAAGGGGGGTGAGGAGG	840
Sbjct	5	-----GATA-----	8

Moving down there are a series of far more convincing near perfect alignments.

You must know what these aligned regions represent by now?

But, just in case:

What do you suppose these regions represent? \_\_\_\_\_

How many are there and do they correspond nicely to the lines of the Dot Matrix View?"

How many exons would you say this mRNA has?

If one was to forgive the strange "bits" at the start, would you say blast seems to have done a reasonable job here?

I think I would.

Query	24301	TTTTGTGTAGTTCTGGCACAATATGGAATACTACTCTTTTCTGAGATTGAGAGAAC	24360
Sbjct	1045	-----AGTTTGAGAGAAC-----	1057
Query	24361	CCATTATCCAGATGTGTTTGGCCGAGAAAGACTAGCAGCCAAAATAGATCTACCTGAAGC	24420
Sbjct	1058	CCATTATCCAGATGTGTTTGGCCGAGAAAGACTAGCAGCCAAAATAGATCTACCTGAAGC	1117
Query	24421	AAGAATACAGGTACCGAGAGACTGTGCACTTTGCACCTTTGTGATTACATCACTTTGTCT	24480
Sbjct	1118	AAGAATACAGGTA-----	1130
Query	24481	TTCTAGAGACAGAGGTGCTTGTACAGAGTACTATTTATTTATAGGACTAATAATAAA	24540
Sbjct		-----	
Query	24541	AAGGTTCACTGCTAAATGCTCTGCTGCCATGGGCGTGGGAGGGCAGCAGTGGAGGTG	24600
Sbjct		-----	
Query	24601	CCAAGGTGGGCTGGGCTCGACGTAGACACAGTGTAACTGTCCACCTGATTTCCAGG	24660
Sbjct		-----	
Query	24661	TATGTTTTCTAATCGAAGGGCCAAATGGAGAAGAGAAGAAAACTGAGGAATCAGAGAA	24720
Sbjct	1131	--TGTTTTCTAATCGAAGGGCCAAATGGAGAAGAGAAGAAAACTGAGGAATCAGAGAA	1188
Query	24721	GACAGGCGAGCAACACACCTAGTCTATTTCTATCAGCAGTAGTTTCAGCACCAGTGTCT	24780
Sbjct	1189	GACAGGCGAGCAACACACCTAGTCTATTTCTATCAGCAGTAGTTTCAGCACCAGTGTCT	1248
Query	24781	ACCAACCAATTCACAACCCACACCGGTAATTTGAAATACTAATACTACGAATCAA	24840
Sbjct	1249	ACCAACCAATTCACAACCCACACCGG-----	1278
Query	24841	TGCTTTTAAACCTGTTTGTCTCGGGCTCTGACTCTACTCTGACTACTGTCTTTCTCTT	24900
Sbjct		-----	
Query	24901	GCCCTCAGTTTCTCTCTTACATCTGGCTCCATGTTGGGCGGAACAGACACAGCCCTCAC	24960
Sbjct	1279	-----TTTCTCTTACATCTGGCTCCATGTTGGGCGTAACAGACACAGCCCTCAC	1330
Query	24961	AAACACCTACAGCGCTCTGCCCTATGCCAGCTTCCACATGGCAAAATACCTGCTAT	25020
Sbjct	1331	AAACACCTACAGCGCTCTGCCCTATGCCAGCTTCCACATGGCAAAATACCTGCTAT	1390
Query	25021	GCAAGTAAGTGGGCTGTTGCTGCATAACCCAGGCCCCAGAGAAGTGAAGAGTGG	25080
Sbjct	1391	GCAA-----	1394
Query	25081	CTCAGGGCTCGCGACCTCATTGGCTGTGTCTGCACCTTGAGAGCTTTTGCACACTACAG	25140
Sbjct		-----	
Query	25141	TGATTGGCTTGACCAAGTCAAGTCGGAGACAGTCAATCCCATCACTTTTAAGTGATTGACT	25200
Sbjct		-----	

Query	28381	GTATTCATGTCTGTTTCTCAAAGGGGAATATCATATGGCTATTTCTTTTCACTTCTCT	28440
Sbjct		-----	
Query	28441	AGGACTCATTTCCTGTTGTTGTCAGTTCCAGTTCAAGTTCCCGGAAGTGAACCTGATAT	28500
Sbjct	1547	---ACTCATTTCCTGTTGTTGTCAGTTCCAGTTCAAGTTCCCGGAAGTGAACCTGATAT	1603
Query	28501	GTCTCAATACTGGCCAAGATTACAGTAAAAAAAAAAAAAAAAAAAAAAAAAGGAAGAAAT	28560
Sbjct	1604	GTCTCAATACTGGCCAAGATTACAGTAAAAAAAAAAAAAAAAAAAAAAAA	1643
Query	28561	ATTGTGTTAATTCAGTCAGTGACTATGGGGACACACAGTTGAGCTTTCAGGAAGAAAG	28620
Query	28621	AAAAATGGCTGTAGAGCGCTTCAGTTCTACAATTGTGCTGTATTGTACCACTGGGG	28680

The final alignment section even has a poly A tail!

Wonderful, but it is not safe to assume that just selecting any service that claims to do a sensitive global pairwise alignment will just work for any pair of sequences. In fact, pretty though it appears, the alignment **blast** has generated is not as entirely logical as it might first seem. For example, consider:

How might the gap around 24,500 in the genomic sequence been positioned more intelligently? \_\_\_\_\_

Next, try aligning the same two sequences with another program (implementing the same algorithm) at the **EBI**.

Go to the **Pairwise Sequence Alignment EBI** page (<http://www.ebi.ac.uk/Tools/psa/>). From there, select the **Nucleotide** option for the **Global Alignment** program **Needle**. **Needle** implements the best global pairwise algorithm faithfully.

**Pairwise Sequence Alignment (NUCLEOTIDE)**  
EMBOSS Needle reads two input sequences and writes their optimal global sequence alignment to file.

This is the form for nucleotide sequences. Please go to the [protein](#) form if you wish to align protein sequences.

**STEP 1 - Enter your nucleotide sequences**

Enter or paste your first nucleotide sequence in any supported format:

Or, upload a file:  pax6\_genomic.fasta

**AND**

Enter or paste your second nucleotide sequence in any supported format:

Or, upload a file:  pax6\_mrna.fasta

**STEP 2 - Set your pairwise alignment options**

MATRIX:  GAP OPEN:  GAP EXTEND:  OUTPUT FORMAT:

END GAP PENALTY:  END GAP OPEN:  END GAP EXTEND:

**STEP 3 - Submit your job**

☐ Be notified by email (Tick this box if you want to be notified by email when the results are available)

## Global Alignment

Global alignment tools create an end-to-end alignment of the sequences to be aligned. There are separate forms for protein or nucleotide sequences.

Needle (EMBOSS)

EMBOSS Needle creates an optimal global alignment of two sequences using the Needleman-Wunsch algorithm.

[Protein](#) [Nucleotide](#)

Load up the first sequence from **pax6\_genomic.fasta**.

Load up the second sequence from **pax6\_mrna.fasta**.

Click on the **More options** button to see what parameters you can set. They should be as you might expect. The defaults are fine for the first run.

Click on the **Submit** button to get **Needle** into action.

Well! Nothing like as convincing as the alignment **blast** produced!

Alignment does not even begin until over **22,000** base pairs along the genomic sequence. Even then it is not convincing, as in **wrong**, if we accept the results already obtained for **blast** as a fair approximation of the truth.

There are some well aligned regions after genomic position **23,600**.

Then a resumption of chaos after **25,000** or so.

How many convincingly aligned regions did you see? \_\_\_\_\_

How many did you expect? \_\_\_\_\_

Clearly, this alignment is not correct. Can you explain why? \_\_\_\_\_

```

11      22101 TCTGTAAACACAATGTGGCCGCTGCACGCTCAAGAGAATC-----CT 22144
HUMOCLHMB      1 -----TATCGATAAGTT-----12

11      22145 TTTGTTGTCCGCGCTCATTGTAGCTCAAAT-TCTGCCACGAAAGTTT 22193
HUMOCLHMB     13 TTTT-----TTATTGT-----CAATCTCTG-----34

11      22194 GCCAACGCTCTGCCCCAGGAGTTAATAGTTTCCCTTACTCGCGGGGCA 22243
HUMOCLHMB     35 -----TCTCCT-TCCAGGAATCTGAGGATTGCTCTTACACAC-----71

11      22244 TTGTGCAGCGCTGAAAGAGCGCCCTCGCTTAATCAAGTGTGGTGTC- 22292
HUMOCLHMB     72 -----CAACCCAGCAA-CATCC-----GTGGAGAA-----95

11      22293 ---TCTCAATAG-ATCTCCAAGGCCCATATGTTGGCCAGTGCCGATGAA 22338
HUMOCLHMB     96 AACTCTCACCAGCACTCC-----114

11      22339 TCCGCTGTTTAAATGGGGGAGAAAGTTGGGTTTTAAACAT----- 22381
HUMOCLHMB    115 -----TTTAAA-----ACACGT-----CATTCAACCAATTGGTGC 147

11      22382 TTCAA-----AGTTCTGTGAAAGATCCC-----ACT----- 22407
HUMOCLHMB    148 TTCAAGCAACAACAGCAGCACAACCAACCAACCAACCAACCACTTGA 197

11      23546 TACCTTGGGAATGTTTGGTGA---GGCTGTGGGATATAATGCTCTTG 23592
HUMOCLHMB     804 -----ATGT-----TGAACGGGAGAGCCGG-----AAGC---TGG 830

11      23593 AGTTTAAAGACTACACAGGCCCTT-TTGGAGGCTCCAAGTTAATCC--A 23639
HUMOCLHMB     831 GG-----CACCG--CCCTGGTTGG-----TATCCGGG-----856

11      23640 AATTTCCTTAC---CATCTATTCTTTTGTTCAGATGGCTGCAAGC 23685
HUMOCLHMB     857 GACTTCGGTGCCAGGGCAACCTA-----CGCAAGATGGCTGCAAGC 897

11      23686 AACAGGAAGGAGGGGAGAGAATACCAACTCCATCAGTTTCAACGGAGAA 23735
HUMOCLHMB     898 AACAGGAAGGAGGGGAGAGAATACCAACTCCATCAGTTTCAACGGAGAA 947

11      23736 GATTAGATGAGGCTCAATGCGACTTCAGCTGAAGCGGAAGCTGCAAG 23785
HUMOCLHMB     948 GATTAGATGAGGCTCAATGCGACTTCAGCTGAAGCGGAAGCTGCAAG 997

11      23786 AAATAGAACTCTTTACC CAAGAGCAAAATGAGGCCCTGGAGAAAGGTG 23835
HUMOCLHMB     998 AAATAGAACTCTTTACC CAAGAGCAAAATGAGGCCCTGGAGAA-----1042

11      23836 ATAGAGTTTTTCAAAGTAGAGAGCAGTAAATCAAAGTAAATGCCATC 23885
HUMOCLHMB    1043 -----1042

11      24636 TAACCTGTCCACCTGATTCCAGGTATGGTTTCTAATCGAAGGGCCAA 24685
HUMOCLHMB    1125 -----CAGGTATGGTTTCTAATCGAAGGGCCAA 1153

11      24686 ATGGAGAAGAGAAGAAAACTGAGGAATCAGAGAAGACAGGCCAGCAACA 24735
HUMOCLHMB    1154 ATGGAGAAGAGAAGAAAACTGAGGAATCAGAGAAGACAGGCCAGCAACA 1203

11      24736 CACCTAGTCATATTCTCTATCAGCAGTAGTTTCAGCACCAGTGTCTACCAA 24785
HUMOCLHMB    1204 CACCTAGTCATATTCTCTATCAGCAGTAGTTTCAGCACCAGTGTCTACCAA 1253

11      24786 CCAATTCACACACCCACACACCGGTAATTTGAATACTAATACTACGA 24835
HUMOCLHMB    1254 CCAATTCACACACCCACACACCG-----1277

11      24836 ATCAATGTCTTTAAACCTGTTTGTCTCCGGGCTCTGACTCTCACTCTGACT 24885
HUMOCLHMB    1278 -----1277

11      24886 ACTGTCTTTTCTTGGCCCTCAGTTTCTGCTTACATCTGGCTCATGT 24935
HUMOCLHMB    1278 -----GTTTCTCTTACATCTGGCTCATGT 1305

11      24936 TGGGCCGAACAGACAGAGCCCTCACAACACCTACAGCGCTCTGCCGCT 24985
HUMOCLHMB    1306 TGGGCCGAACAGACAGAGCCCTCACAACACCTACAGCGCTCTGCCGCT 1355

11      24986 ATGCCAGCTTCAACATGGCAAAATACCTGCCTATGCAAGTAAGTGGCG 25035
HUMOCLHMB    1356 ATGCCAGCTTCAACATGGCAAAATACCTGCCTATGCAAGTAAGTGGCG 1394

11      25036 TGGTGGTGGCTGCATAACCCAGGCCAG--AGAAGTGAGGAGTGGCTC 25083
HUMOCLHMB    1395 -----CC-----CCAGTCCCAGCCAGA-----1413

11      25084 AGGGCTCGGAGCTCAT-----TGGCTGTGCTG--CACCTTGAGAGC 25126
HUMOCLHMB    1414 ---CCT---CCTCATACTCTGATG---CTGCCACC---AGC 1445

```

I assume you have all read the lucid answers to the question above? If so, I am confident you will agree that there are **3** ways to get an answer, similar to that generated by **blast**, from the tools offered at the **EBI**. They are:

- Make gap penalties so cheap that **Needle** will have no excuse not to avoid gaps anywhere they are needed. This works if you use a gap opening penalty of **1.0** (the lowest allowed by both the program and the web interface) and a gap extension penalty of **0.0**, allowed by the program *but not by the EBI web interface!!* The lowest value the web interface allows is **0.0005**, which really should be sufficiently small, but provably is not. The most important question being “*Why would a web interface restrict a program's capabilities other than to prevent excessive resource use?*”. I have no answer for that one, I will just petulantly include my successful low gap alignment (made without a web interface) in your **Backup\_Results** directory and retire with self righteous hauteur!

Actually, using gap penalties to suit huge gaps that are really introns, will only work when the exons are so similar (as here) that any gap penalties will work for their alignment. Generally, you need to pick gap penalties to optimise exon alignment. So this is a very horrible way to “fix” the situation anyway.

- Tell **Needle** to penalise the gaps it puts at either end of the alignment in the same way it penalises gaps it puts in the middle. By default, end gaps are free!! Which is not very logical here.
- Use **Stretcher**, which uses essentially the same algorithm as **Needle**, except, it also applies a bit of common sense (**heuristics**, if you like). **Stretcher** takes a look at the sequences before it starts to do any serious computation. It identifies any “*good regions*” (all **12** exon matches in this case) and then says “*OK, I am definitely having those, how best can I deal with the rest?*”. In essence, **Stretcher** does a quick **Dot Matrix View** before it starts and so only goes to work when it has a pretty good idea what the answer should look like. It works in this case, but not always. **Stretcher** is faster than **Needle** but does not necessarily generate the highest scoring alignment. **Stretcher** works in a fashion far closer to the way a human would work, which has to be good! Well, usually anyway.

So, try the **Needle** with penalised **End Gaps** approach by returning to the **Needle** launch page from your results. You should find the two sequences are still selected, so you should only have to click on **More Options** again and change the **END GAP PENALTY** field from **false** to **true**.

STEP 2 - Set your pairwise alignment options

MATRIX DNAAfull	GAP OPEN 10	GAP EXTEND 0.5	OUTPUT FORMAT pair
END GAP PENALTY true	END GAP OPEN 10	END GAP EXTEND 0.5	

Click on the **Submit** button and **Needle** will be on the road again.

How many matching regions are there this time? \_\_\_\_\_

Is the count **now** roughly as you would expect? \_\_\_\_\_

Finally, check that **Stretcher** works as expected.

Go again to the **Pairwise Sequence Alignment EBI** page (<http://www.ebi.ac.uk/Tools/psa/>).

From there, select the **Nucleotide** option for the **Global Alignment** program **Stretcher**.

Load up the sequences exactly as for **Needle**.

Take a look at the parameters and see there is nothing unexpected hiding there.

Set **Stretcher** sequence rope stretching.

How do you feel about the results this time?

How do you think **blast** achieve the correct results without any fuss? \_\_\_\_\_

Tools > Pairwise Sequence Alignment > EMBOSSt Stretcher

Pairwise Sequence Alignment (NUCLEOTIDE)

EMBOSS Stretcher calculates an optimal global alignment of two sequences using a modification of the classic dynamic programming algorithm which uses linear space.

This is the form for nucleotide sequences. Please go to the [protein](#) form if you wish to align protein sequences.

STEP 1 - Enter your nucleotide sequences

Enter or paste your first nucleotide sequence in any supported format:

Or, upload a file:  pax6\_genomic.fasta

AND

Enter or paste your second nucleotide sequence in any supported format:

Or, upload a file:  pax6\_mrna.fasta

STEP 2 - Set your pairwise alignment options

MATRIX DNAAfull	GAP OPEN 16	GAP EXTEND 4	OUTPUT FORMAT pair
--------------------	----------------	-----------------	-----------------------

STEP 3 - Submit your job

☐ Be notified by email (Tick this box if you want to be notified by email when the results are available)



## Pairwise Sequence Comparison using Specialised Software

None of the alignments generated thus far have been entirely correct.

By persuading the general global alignment software to treat huge gaps (i.e. the introns) in some sort of special manner, a reasonable answer was obtained. However, the general software could not know that something more than just **Substitutions** and **Indels** were at issue here. Consequently, it stood no chance of dealing with the intron/exon boundaries sensibly.

The solution is not to fiddle around with the parameters of the general tools. Aligning **mRNAs** with **Genomic** sequence is simply not "*General Alignment*". It is an example of a problem that is sufficiently particular to require specialised software for an optimal solution.

There is a program in the **EMBOSS** package (the same collection of programs as **Needle** and **Stretcher**), called **est2genome**, which is specifically designed for the alignment of cDNA/mRNA and genomic sequences. **est2genome** (and similar programs) may assume much more about the sequences to be aligned than can a general purpose alignment program. Gaps representing introns can be placed far more accurately if they are **known** to represent introns. Programs such as **est2genome** seek the highly conserved bases that occur at intron/exon boundaries, **C/T** rich intronic regions, **polyA** regions and **Stop/Start** codons to assist its detection of exons and gene structures.

**est2genome** is a fine program, but there are two other options offered at the **NCBI** in America that do the same job, I think, somewhat more nicely. Of these, I choose the program called **splign** for this exercise on the grounds it is the more sophisticated service<sup>1</sup>. To investigate, go to the home of **splign** at:

<http://www.ncbi.nlm.nih.gov/sutils/splign>

Click on the **Online** button. In the **Genomic** section, **Browse** to upload **pax6\_genomic.fasta**.

In the **cDNA** section, paste the sequence **pax6\_mrna.fasta**. Where **cDNA** and **Genomic** sequences share exons that are nearly identical, **splign** uses the comparison algorithm **megablast** (default). Where exons are less similar (e.g. when the **cDNA** and **Genomic** sequences are from different organisms) the more sensitive option **discontinuous megablast**, is a better choice<sup>2</sup>. Note the option to compare your **cDNA** with a **Whole genome** (including Human). Today, the default options are fine. Click the **Align** button.

Please specify input sequences by GI/Accession or in FASTA format.  
Examples (click to select):

- NM\_214647 / NW\_732498 (one model)
- AF238306 / NT\_033777 (one model, frameshifts)
- NM\_020978 / NG\_004750 (multiple models)

**cDNA:**

>gi|189352|gb|M77844.1|HUMOCLHMB Homo sapiens ocularhombin (PAX6) mRNA, complete cds, alternatively spliced  
TATCGATAAGTTTTTTTTTATTGTCAATCTCTGTCTCCTCCCA  
GGAATCTGAGGATTGCTCTTACACA  
CCAACCCAGCAACATCCGTGGAGAAACTCTCACCAGCACTCTCT

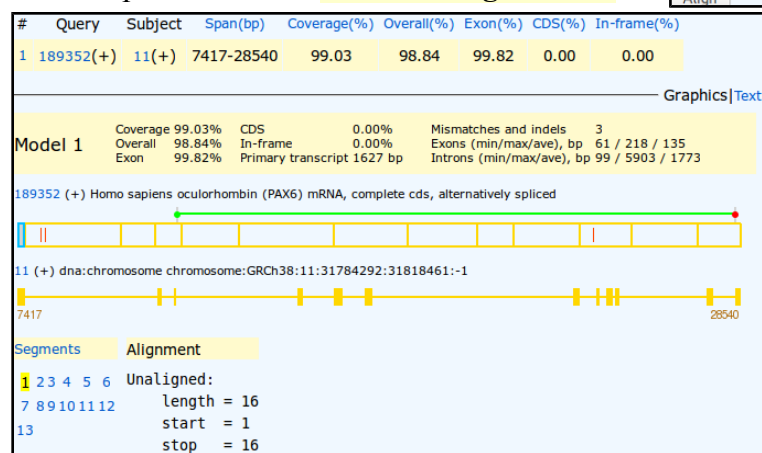
**Genomic:** From: 1 To: max

Browse... pax6\_genomic.fasta

**Whole genome:** Not selected

☐ Lower quality query sequence (e.g. EST)  
☐ Reverse and complement the query  
☐ More partial alignments  
☐ Use discontinuous megablast (e.g. for cross-species)

Align



Your results will appear showing the cDNA split into **12** sections (the predicted exons) corresponding to **12** regions of the genomic sequence indicated by yellow rectangles. A **13<sup>th</sup>** region of **16** base pairs is displayed and declared to be **unaligned**. These are the **16** mystery base pairs at the start of this particular mRNA that **Needle** and **Stretcher** had trouble treating sensibly also. I wonder what they are?

Any theories?

<sup>1</sup> The other is called **spidey**. The **spidey** home page is:

<http://www.ncbi.nlm.nih.gov/IEB/Research/Ostell/Spidey/>

It is pretty straight forward to use, just follow your nose. **est2genome**, **splign** and **spidey** should all give the same answer for this simple alignment and very similar answers for all such problems.

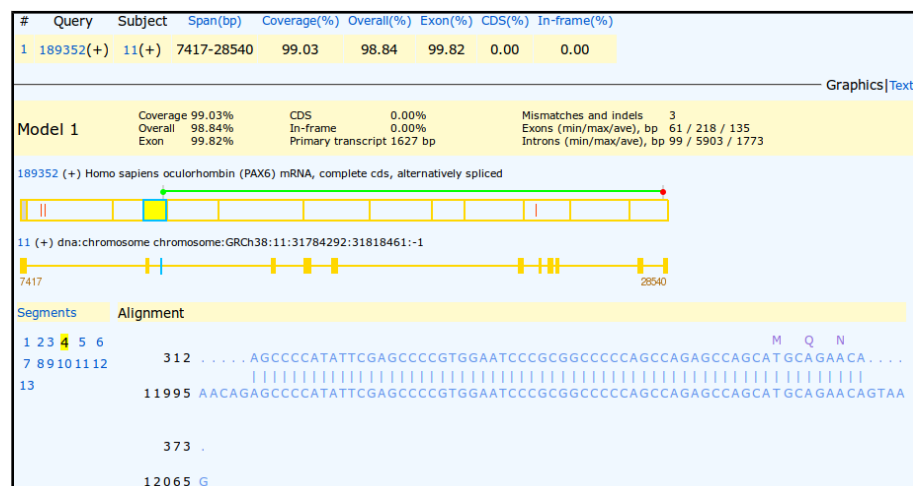
<sup>2</sup> Why this is so will be considered later when we look at the database searching program **blast**.

Click on the first exon section of the cDNA display.

Here there shows two **substitutions**. These were also apparent in the successful **blast**, **Needle** and **Stretcher** alignments. You might have spotted them?

Though these are in a non-coding region, they could easily still be very significant. However, for the purposes of this exercise, let us assume they are not.

The **Start** (green) and **Stop** (red) codons delimiting the **CoDing Sequence (CDS)** are illustrated by the bar above the cDNA display.



Click on the exon including the green **Start** codon (the 3<sup>rd</sup>).

The first coding exon is now displayed with translation of the mRNA where appropriate.

The statistics at the top of the display include the claim that there are **3** discrepancies (**Mismatches** and **Indels**) between the cDNA and **Genomic** sequences.

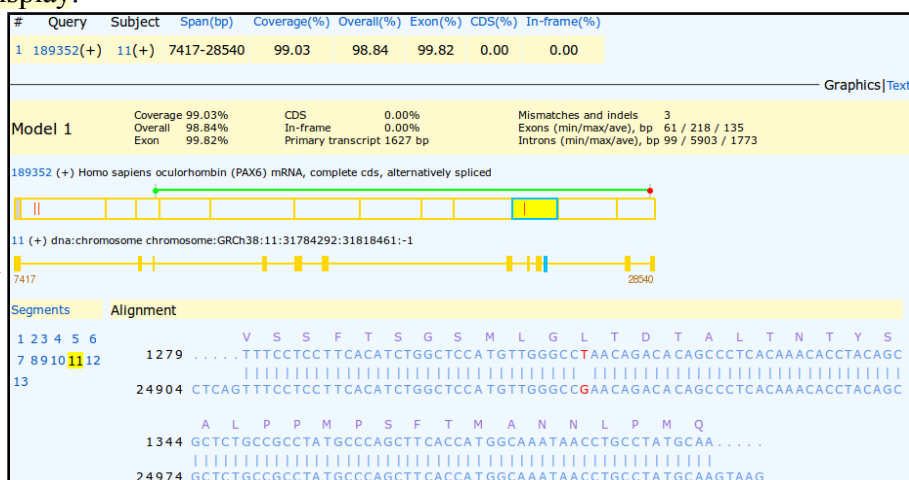
Two of these are the **substitutions** we have already seen in the first exon of the cDNA. The third is indicated by the red bar in the 10<sup>th</sup> exon of the cDNA display.

Click on the 10<sup>th</sup> exon section of the cDNA display.

The third difference, a substitution, should be clear to see. Given it changes the coded protein, this substitution is likely to be the most significant.

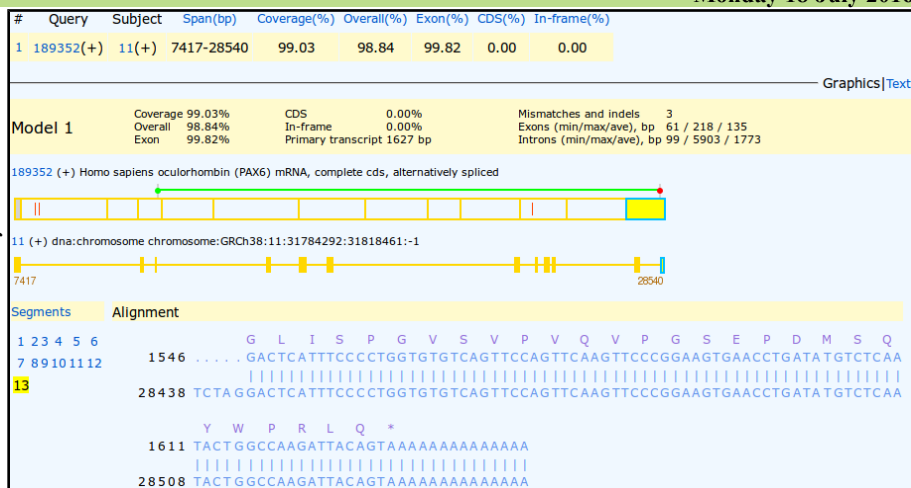
Irritatingly, in the extreme! **spline** only translates to mRNA. So one has to work to discover the alternative suggested by the Genomic sequence.

Probably vital if we were really doing this seriously, just casual interest as this is but an exercise. I do not intrude on real life much and **it**, largely, leaves **me** untouched in grateful response.



What is the amino acid corresponding to the mutated position in the **Genomic** sequence?\_\_\_\_\_

Click on the last exon section in the cDNA display. You should now see the final exon of the cDNA with the **Stop** codon and polyA region.



#	Query	Subject	Span(bp)	Coverage(%)	Overall(%)	Exon(%)	CDS(%)	In-frame(%)
1	189352(+)	11(+)	7417-28540	99.03	98.84	99.82	0.00	0.00

#	Query	Subject	Idty	Len	Q.Start	Q.Fin	S.Start	S.Fin	Type	Details
+1	189352	11	-	16	1	16	-	-	<L-Gap>	-
+1	189352	11	0.991	218	17	234	7417	7634	CA<exon>GT	M39RM8RM169
+1	189352	11	1	77	235	311	11537	11613	AG<exon>GC	M77
+1	189352	11	1	61	312	372	12000	12060	AG<exon>GT	M61
+1	189352	11	1	131	373	503	15628	15758	AG<exon>GT	M131
+1	189352	11	1	216	504	719	16686	16901	AG<exon>GT	M216
+1	189352	11	1	166	720	885	17606	17771	AG<exon>GT	M166
+1	189352	11	1	159	886	1044	23674	23832	AG<exon>GT	M159
+1	189352	11	1	83	1045	1127	24348	24430	AG<exon>GT	M83
+1	189352	11	1	151	1128	1278	24660	24810	AG<exon>GT	M151
+1	189352	11	0.991	116	1279	1394	24909	25024	AG<exon>GT	M33RM82
+1	189352	11	1	151	1395	1545	27602	27752	AG<exon>GT	M151
+1	189352	11	1	98	1546	1643	28443	28540	AG<exon>	M98

Finally, click on the **Text** link to view the textual summary of the **splign** results.

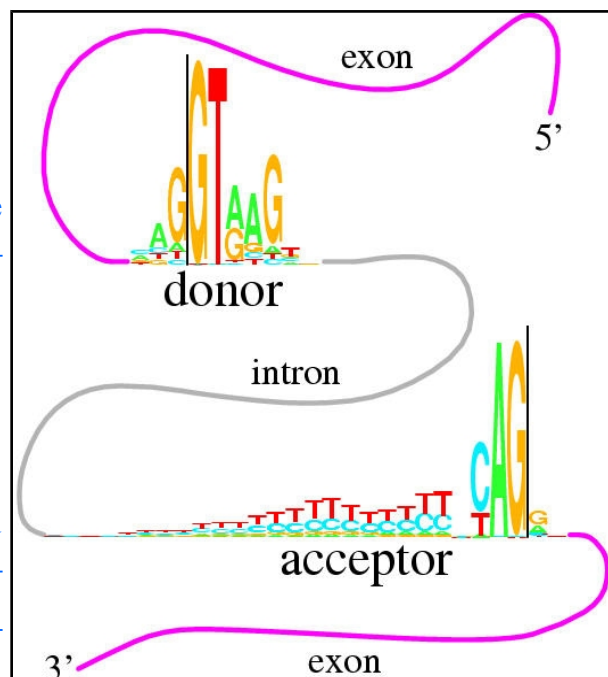
How do you interpret the **Details** column for exons 1 and 10?

Where is the 3<sup>rd</sup> substitution in the mRNA?

Where is the 3<sup>rd</sup> substitution in the Genomic Sequence?

Compare the predicted **splign** intron/exon boundaries with the conservation suggested by the logo<sup>3</sup>.

What deviation(s) from the model suggested by the logo can you see?



3 The original label for this very nice graphic is:

This figure shows two "sequence logos" which represent sequence conservation at the 5' (donor) and 3' (acceptor) ends of human introns. The region between the black vertical bars is removed during mRNA splicing. The logos graphically demonstrate that most of the pattern for locating the intron ends resides on the intron. This allows more codon choices in the protein-coding exons. The logos also show a common pattern "CAG|GT", which suggests that the mechanisms that recognize the two ends of the intron had a common ancestor. See R. M. Stephens and T. D. Schneider, "Features of spliceosome evolution and function inferred from an analysis of the information



## Sensitive Local Pairwise Sequence Comparison

Finally, a swift look at sensitive local pairwise sequence alignment. You have already used **blast** to do a local pairwise alignment in the last Practical, when you aligned the two human genomic sequencing contigs that covered the **PAX6** location in **Chromosome 11**. **blast** did not use a sensitive approach however, nothing subtle was required for that particular alignment.

For a more accurate alignment, return to the **Pairwise Sequence Alignment** **EBI** page (<http://www.ebi.ac.uk/Tools/psa/>).

From there, select the **Nucleotide** option for the **Local Alignment** program **Matcher**.

**Water** or **LALIGN** would also be fine options, but I declare **Matcher** to be choice of the day.

### Local Alignment

Local alignment tools find one, or more, alignments describing the most similar region(s) within the sequences to be aligned. There are separate forms for protein or nucleotide sequences.

**Water** (EMBOSS)

EMBOSS Water uses the Smith-Waterman algorithm (modified for speed enhancements) to calculate the local alignment of two sequences.

[Protein](#) [Nucleotide](#)

**Matcher** (EMBOSS)

EMBOSS Matcher identifies local similarities between two sequences using a rigorous algorithm based on the LALIGN application.

[Protein](#) [Nucleotide](#)

**LALIGN**

LALIGN finds internal duplications by calculating non-intersecting local alignments of protein or DNA sequences.

[Protein](#) [Nucleotide](#)

Tools > Pairwise Sequence Alignment > EMBOSS Matcher

### Pairwise Sequence Alignment (NUCLEOTIDE)

EMBOSS Matcher identifies local similarities in two input sequences using a rigorous algorithm based on Bill Pearson's lalign application, version 2.0u4 (Feb. 1996).

This is the form for nucleotide sequences. Please go to the [protein](#) form if you wish to align protein sequences.

**STEP 1 - Enter your nucleotide sequences**

Enter or paste your first nucleotide sequence in any supported format:

Or, upload a file: [Browse...](#) pax6\_genomic.fasta

AND

Enter or paste your second nucleotide sequence in any supported format:

Or, upload a file: [Browse...](#) pax6\_mrna.fasta

**STEP 2 - Set your pairwise alignment options**

MATRIX	GAP OPEN	GAP EXTEND	ALTERNATIVES MATCHES	OUTPUT FORMAT
DNAfull	16	4	1	pair

**STEP 3 - Submit your job**

☐ Be notified by email (Tick this box if you want to be notified by email when the results are available)

[Submit](#)

Load up the **Genomic** and **mRNA** sequences as you did for **Needle**.

Click on the **More options** button to see what parameters you can set. They should be as you might expect. The defaults are fine for the first run.

Click on the **Submit** button to get **Matcher** into **Matchbox** mode.

After due consideration of all the possibilities, **Matcher** will enrich your screen with its conclusions.

But, only one alignment? A good one, covering the highest scoring region of all those considered, but it cannot be the whole story, which must tell the tale of **12** exons! Here is but one.

In common with most local alignment programs, by default **Matcher** will only show you the single best local alignment between two sequences.

A good reason to have a **Dot Matrix View** to inform one of roughly what to expect, which is not one miserable alignment in this case.

11	16670	CAC TTC C C C T A T - - - G C A G G T G T C C A A C G G A T G T G A G T A A A A T T C T G G	16716
HUMOCLHMB	485	C A T T T C C C G A A T T C T G C A G G T G T C C A A C G G A T G T G A G T A A A A T T C T G G	534
11	16717	G C A G G T A T T A C G A G A C T G G C T C C A T C A G A C C C A G G G C A A T C G G T G G T A G T	16766
HUMOCLHMB	535	G C A G G T A T T A C G A G A C T G G C T C C A T C A G A C C C A G G G C A A T C G G T G G T A G T	584
11	16767	A A A C C G A G A G T A G C G A C T C C A G A A G T T G T A A G C A A A A T A G C C C A G T A T A A	16816
HUMOCLHMB	585	A A A C C G A G A G T A G C G A C T C C A G A A G T T G T A A G C A A A A T A G C C C A G T A T A A	634
11	16817	G C G G G A G T G C C C G T C C A T C T T T G C T T G G G A A A T C C G A G A C A G A T T A C T G T	16866
HUMOCLHMB	635	G C G G G A G T G C C C G T C C A T C T T T G C T T G G G A A A T C C G A G A C A G A T T A C T G T	684
11	16867	C C G A G G G G G T C T G T A C C A A C G A T A A C A T A C C A A G C G T A A G T T C A T T G A G A	16916
HUMOCLHMB	685	C C G A G G G G G T C T G T A C C A A C G A T A A C A T A C C A A G C G T G T C A T C A A T A A A C	734
11	16917	A C A - - T C T G C C C T C C C T G C C	16934
HUMOCLHMB	735	A G A G T T C T T C G C A A C C T G G C	754

Of course, it is also miserable biologically! **Matcher** fails to align the exons accurately for all the same reasons that the **Needle** failed to represent the *biological* reality.

at human splice sites", J. Mol. Biol., 228, 1124-1136, (1992).

So, what can one do but try again! By returning to the **Matcher** launch page from your results. You should find the two sequences are still selected, so you should only have to click on **More Options** again and set the **ALTERNATIVE MATCHES** field **20**.

STEP 2 - Set your pairwise alignment options

MATRIX DNafull	GAP OPEN 16	GAP EXTEND 4	ALTERNATIVES MATCHES 20	OUTPUT FORMAT pair
-------------------	----------------	-----------------	-------------------------------	-----------------------

Actually, as you know there are only **12** exons. And that some might well be close enough to be included in the same alignment, you do not need to go as high as **20**. However, the web interface restricts choice (**WHY!?**) such that this is the most sensible cautious choice.

Click on the **Submit** button and **Matcher** will trust and obey.

At the top of your output will be some nice believable local alignments, some involving more than one exon.

**Matcher** tries to make each alignment as long as it can, stopping only when, to stretch the alignment any further would involve the alignment score decreasing due to the necessity for gap penalties.

11	24655	TCCAGGTATGGTTTTCTAATCGAAGGGCCAAATGGAGAAGAGAAGAAAA	24704
HUMOCLHMB	1123	TACAGGTATGGTTTTCTAATCGAAGGGCCAAATGGAGAAGAGAAGAAAA	1172
11	24705	CTGAGGAATCAGAGAAGACAGGCCAGCAACACACCTAGTCATATTCCTAT	24754
HUMOCLHMB	1173	CTGAGGAATCAGAGAAGACAGGCCAGCAACACACCTAGTCATATTCCTAT	1222
11	24755	CAGCAGTAGTTTCAGCACCAAGTGTCTACCAACCAATTCCACAACCCACCA	24804
HUMOCLHMB	1223	CAGCAGTAGTTTCAGCACCAAGTGTCTACCAACCAATTCCACAACCCACCA	1272
11	24805	CACCGGGTAATTTGAAATACTAATACTACGAATCAATGTCTTTAAACCTG	24854
HUMOCLHMB	1273	CACCGG-----	1278
11	24855	TTTGCTCCGGGCTCTGACTCTCACTCTGACTACTGTCTTTCTCTTGCCC	24904
HUMOCLHMB	1279	-----	1278
11	24905	TCAGTTTCCTCCTTCACATCTGGCTCCATGTTGGGCCGAACAGACACAGC	24954
HUMOCLHMB	1279	----TTTCTCCTTCACATCTGGCTCCATGTTGGGCCGAACAGACACAGC	1324
11	24955	CCTCACAACACCTACAGCGCTCTGCGGCTATGCCAGCTTCACCATGG	25004
HUMOCLHMB	1325	CCTCACAACACCTACAGCGCTCTGCGGCTATGCCAGCTTCACCATGG	1374
11	25005	CAAATAACCTGCCTATGCAA	25024
HUMOCLHMB	1375	CAAATAACCTGCCTATGCAA	1394

```

=====
#
# Aligned_sequences: 2
# 1: 11
# 2: HUMOCLHMB
# Matrix: EDNAFULL
# Gap_penalty: 16
# Extend_penalty: 4
#
# Length: 58
# Identity:      39/58 (67.2%)
# Similarity:    39/58 (67.2%)
# Gaps:          6/58 (10.3%)
# Score: 83
#
#
=====
11          2353 GCTGGACGCCACCGGCCGAG--GCCGGG---CTGAGGAGCGGGGTC 2397
HUMOCLHMB   425 GCCGGACTCCACCGGCCAGAGATTGTAGAGCTAGCTCAC-AGCGGGGCC 473

11          2398 TGGCCGGG      2405
HUMOCLHMB   474 CGGCCGTG      481

=====
#
# Aligned_sequences: 2
# 1: 11
# 2: HUMOCLHMB
# Matrix: EDNAFULL
# Gap_penalty: 16
# Extend_penalty: 4
#
# Length: 46
# Identity:      31/46 (67.4%)
# Similarity:    31/46 (67.4%)
# Gaps:          1/46 ( 2.2%)
# Score: 83
#
#
=====
11          11417 ACAGTTTGACTGAGCCCTAGATGCATGTGTTTTT-CCTGAGAGTGA 11461
HUMOCLHMB   1043 AGAGTTTGAGAGAACCCATTATCCAGATGTGTTTGCCCGAGAAAGA 1088

```

Go to far down the list of alignments and you will realise what a literal interpretation **Matcher** has of its duties.

You asked for **20** alignments?

So here are the best 20 alignments and it is entirely up to you to decide where “silly” begins.

Not too difficult in this case I suggest.

Why do you suppose your aligned exons are not presented in the correct positional order?

THE END

DPJ – 2016.07.17

## Model Answers to Questions in the Instructions Text.

### Notes:

For the most part, these “**Model Answers**” just provide the reactions/solutions I hoped you would work out for yourselves. However, sometime I have tried to offer a bit more back ground and material for thought? Occasionally, I have rambled off into some rather self indulgent investigations that even I would not want to try and justify as pertinent to the objective of these exercises. I like to keep these meanders, as they help and entertain me, but I wish to warn you to only take regard of them if you are feeling particularly strong and have time to burn. Certainly not a good idea to indulge here during a time constrained course event!

Where things have got extreme, I am going to make two versions of the answer. One starting:

### Summary:

Which has the answer with only a reasonably digestible volume of deep thought. Read this one.

The other will start:

### Full Answer:

Beware of entering here! I do not hold back. Nothing complicated, but it will be long and full of pedantry.

This makes the Model answers section very big. **BUT**, it is not intended for printing or for reading serially, so I submit, being long and wordy does not matter. Feel free to disagree.

## From your investigations of Global Alignment:

What do you suppose these regions represent?

Exons

How might the gap around 24,500 in the genomic sequence been positioned more intelligently?

**blast** has positioned a gap in this region merely to maximize the overall alignment score. There is more than one way of achieving this simple goal. However, if it were to be recognized that the gap to be positioned was to represent an intron, then one of the arithmetically equivalent options becomes far more attractive than the others. This “best” option is not the one chosen by **blast**, which is forgivable as **blast** had no reason to expect an intron and was not written to understand the properties of introns anyway.

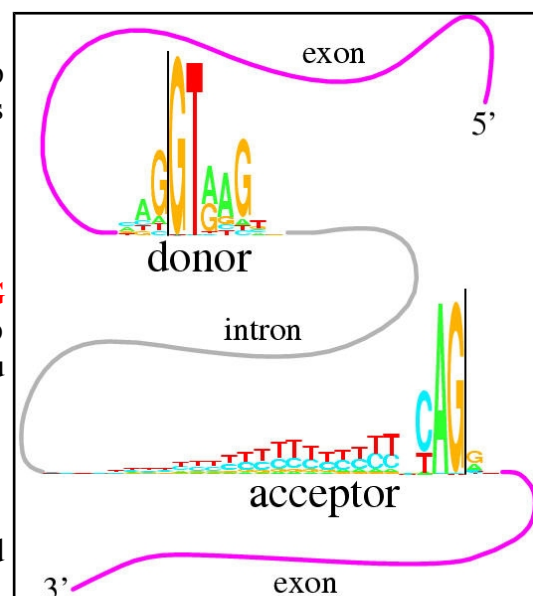
The alignment chosen for this region by **blast** was:

Genomic	24421	AAGAATACAG	GT	ACCGAGAGACTGTGCAGTTTCACACTTTGTGATTCATACCATTGTCT	24480
mRNA	1118	AAGAATACAG	GT	-----	1130
			• • •		
Genomic	24601	CCAAGGTGGGGCTGGGCTCGACGTAGACACAGTGCTAACCTGTCCCACCTGATTTCAGG		24660	
mRNA		-----			
Genomic	24661	TATGGTTTTCTAATCGAAGGGCCAAATGGAGAAGAGAAGAAAACTGAGGAATCAGAGAA		24720	
mRNA	1131	--TGGTTTTCTAATCGAAGGGCCAAATGGAGAAGAGAAGAAAACTGAGGAATCAGAGAA		1188	

Shifting the gap 3 places to the left neither changes the size of the gap nor the perfection of the alignment either side of the gap and so does not affect the alignment score.

However, it does mean the gap begins with an **GT** and ends with a **AG** which is what one might expect if it were known that the gap represented an intron. I include the beautiful **Intron/Exon** logo. As you might gather, I rather like this one.

So, if **blast** was a little better informed, the improved alignment would have been:



Genomic	24421	AAGAATACAG	GT	ACCGAGAGACTGTGCAGTTTCACACTTTGTGATTCATACCATTGTCT	24480
mRNA	1118	AAGAATACAG	---	-----	1130
			• • •		
Genomic	24601	CCAAGGTGGGGCTGGGCTCGACGTAGACACAGTGCTAACCTGTCCCACCTGATTTC	AGG	24660	
mRNA		-----	G		
Genomic	24661	TATGGTTTTCTAATCGAAGGGCCAAATGGAGAAGAGAAGAAAACTGAGGAATCAGAGAA		24720	
mRNA	1131	TATGGTTTTCTAATCGAAGGGCCAAATGGAGAAGAGAAGAAAACTGAGGAATCAGAGAA		1188	

This is the alignment that one might expect from any program customized to align **mRNA** with **Genomic** sequence, as you will see in the fullness of time.

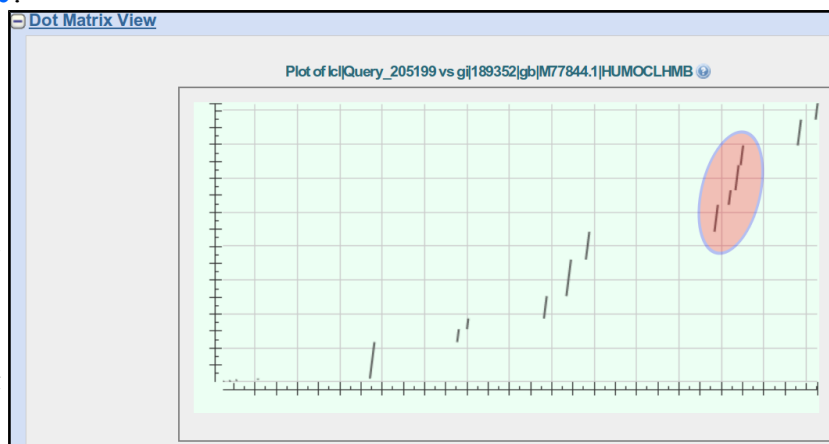
How many convincingly aligned regions did you see?

4

How many did you expect?

12, as that was how many **blast** found, not including the silly ones at the beginning.

The 4 that were found correspond to the 4 diagonal lines grouped together in the **Dot Matrix View** made by **blast**.



Clearly, this alignment is not correct. Can you explain why?

This alignment algorithm only wishes to maximise an alignment score. It sees ALL the high scoring exon regions, however, as the gaps between many of the exons (introns that is) are so long that the penalties for representing them correctly are greater than the gain achieved by the inclusion of the extra exons in the alignment. Arithmetically, it is better to align all the exons either side of the 4 exons that were aligned sensibly, in the biologically improbably fashion shown. Arithmetically the best alignment, biologically ridiculous!

This behaviour is exaggerated because this program regards the enormous gaps in has suggested at the start and end of the alignments as “free”. Some global alignment programs (including this one if you ask politely, as you will see) offer the option of penalising the ends gaps in the same way as for internal gaps. Normally, not penalising end gaps is sensible as it allows for the sequences to have slightly different lengths. In this case, penalising end gaps will result in a far better alignment.

Had you used **stretcher** (also offered by the **EBI**) you would have got a much improved answer in this case (but not necessarily in generally). This is because **stretcher** works in a way far closer to the way an informed human might think. **stretcher** does not mindlessly insist of the highest alignment score. Instead, it looks for all the high scoring regions (i.e. all the exons) and then computes the best way to link them together. The result is a far more convincing alignment, but not the arithmetically best scoring answer.

How many matching regions are there this time?

Were you to trawl through your textual output carefully (or simply take my immaculate word for it), you would find 12 perfectly (or nearly so) aligned regions, implying 12 exons.

To be pedantic, the nicely aligned regions do not match the exons exactly (as has been discussed), but well enough to claim definite evidence for the number of exons. 12 is good enough for me.

Is the count now roughly as you would expect?

Yes, exactly the same as **blast** predicted in the first place. More exons than 17 might have been a surprise as that is how many the gene record for **PAX6** at the **NCBI** suggested. Any given transcript maybe have less than 17 exons or exactly 17 exons, but not more than 17 exons if the heroes of the **NCBI** are not mistaken.

How do you think **blast** achieve the correct results without any fuss?

The only way **blast** could have got the right answer, as it did, would be to use one of the strategies listed previously. **blast** did not use the horrible idea of making gaps super cheap! Not only is that a disgustingly dirty trick, but **blast** actually declares that it is using quite sensible gap penalties.

Leaving **penalising end gaps** and/or using the same sort of heuristics employed by **stretcher**. I would strongly suspect **blast** uses a **stretcher** approach. After all, **blast** has clearly already identified all the “promising regions” in order to construct its **Dot Matrix View**. Also the **stretcher** strategy is similar to that of all **blast** searches (discussed in the next Practical). Finally, **blast** is often used to align very long DNA sequences to detect very strongly similar large regions. This is exactly what the faster (if less pure) **stretcher** approach is all about.



## From your investigations comparing mRNA/cDNA with genomic DNA:

What is the amino acid corresponding to the mutated position in the **Genomic** sequence?

```

T S G S M L G L T D T A L T N
ACATCTGGCTCCA TGTGGGCGCTAACAGACACAGCCCTCACAAC
|||||
ACATCTGGCTCCA TGTGGGCGCGAACAGACACAGCCCTCACAAC

```

The top sequence is the mRNA. **spline** is kind enough to explicitly inform us that the “mutated” codon, **CTA**, will be expressed as **Leucine**.

So, why not translate the **Genomic** sequence also **spline**?! Easy enough to look up. But I resent having to do so!

From this rather beautiful representation of the **Genetic Code**, I conclude:

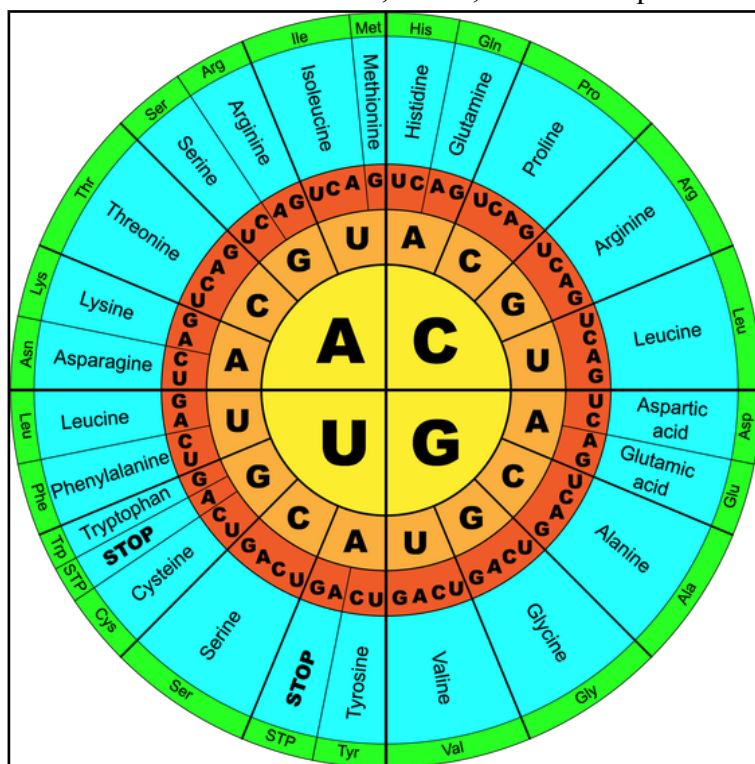
**mRNA**    **CTA** → **Leucine (L)**

**Genomic** **CGA** → **Arginine (R)**

I checked, and this does not appear to be a substitution that is associated with any “interesting” phenotype.

There is no real reason why it should. We did not pause to find out anything about the mRNA downloaded from the **NCBI**. The annotation is particularly unrevealing by itself (it is in **Backup\_Files** if you really want to check).

Let us simply assume it is a benign **Accepted Point Mutation (PAM)**. Yes indeed, that feels comfortable. Not so very tricky this Science stuff after all what!



How do you interpret the **Details** column for exons 1 and 10?

### Summary:

The **Details** column shows the alignments of each exon in a compressed format described in the **spline** documentation as illustrated.

11. Alignment transcript	Alignment transcript represents full details of the alignment in a form of a string composed of characters 'M', 'R', 'I' and 'D' where each character corresponds to an elementary command (Match, Replace, Insert or Delete) needed to transform the query segment into the subject segment. The string is encoded with RLE.
--------------------------	---

The majority of the exon alignments are trivial.

#	Query	Subject	Span(bp)	Coverage(%)	Overall(%)	Exon(%)	CDS(%)	In-frame(%)		
1	189352(+)	11(+)	7417-28540	99.03	98.84	99.82	0.00	0.00		
										<a href="#">Graphics</a>   <a href="#">Text</a>
#	Query	Subject	Idty	Len	Q.Start	Q.Fin	S.Start	S.Fin	Type	Details
+1	189352	11	-	16	1	16	-	-	<L-Gap>	-
+1	189352	11	0.991	218	17	234	7417	7634	CA<exon>GT	M39RM8RM169
+1	189352	11	1	77	235	311	11537	11613	AG<exon>GC	M77
+1	189352	11	1	61	312	372	12000	12060	AG<exon>GT	M61
+1	189352	11	1	131	373	503	15628	15758	AG<exon>GT	M131
+1	189352	11	1	216	504	719	16686	16901	AG<exon>GT	M216
+1	189352	11	1	166	720	885	17606	17771	AG<exon>GT	M166
+1	189352	11	1	159	886	1044	23674	23832	AG<exon>GT	M159
+1	189352	11	1	83	1045	1127	24348	24430	AG<exon>GT	M83
+1	189352	11	1	151	1128	1278	24660	24810	AG<exon>GT	M151
+1	189352	11	0.991	116	1279	1394	24909	25024	AG<exon>GT	M33RM82
+1	189352	11	1	151	1395	1545	27602	27752	AG<exon>GT	M151
+1	189352	11	1	98	1546	1643	28443	28540	AG<exon>	M98

For example:

For **Exon 2**, spline informs us **M77**, meaning “There are **77** bases aligned and they all **Match** perfectly”.

For **Exon 4**, spline informs us **M131**, meaning “There are **131** bases aligned and they all **Match** perfectly”.

The only **2** interesting entries are those where there are some disagreements. That is, the entries for **Exons 1** and **5**, which, following the documentation, I translate thus:

#### Exon 1 – M39RM8RM169

An alignment of **218** bases, the first **39** of which **Match** perfectly (**M39**), there then follows an **Replacement (R)**, a further **8 Matched** bases(**M8**), a second **Replacement (R)** all finished off with **169 Matched** bases (**M169**).

#### Exon 10 – M33RM82

An alignment of **116** bases, the first **33** of which **Match** perfectly (**M33**), there then follows a **Replacement (R)** and a further **82 Matched** bases(**M82**).

It's a pity there are no **Insertions (I)** and **Deletions (D)**, but this was the best **mRNA** I could find.

**Full Answer:**

A point of pedantry to commence. From a different example, which included **InDels**, I got the display illustrated.

The exon was reported as: **M53IM5IM43**

This implies that the choice of **I**nsertion (**I**) or **D**eletion (**D**) is made to describe the type of variation required to transform the **cDNA (Query)** sequence into the **genomic (Subject)**. Hence the two **InDels** displayed here are considered to be **Insertions**.

```

1 CAGAGGTCAGGCTTCGCTAATGGGCCAGTGAGGAGCGGTGGAGGCGAGGCCGG - CGCCG - CACACACACA
|||||
7245 CAGAGGTCAGGCTTCGCTAATGGGCCAGTGAGGAGCGGTGGAGGCGAGGCCGGGCGCCGGCACACACACA

```

Not that it is a vital issue, but I would have thought the other way around was more logical? That is, to consider the **genomic** sequence as the **reference** against which a particular **mRNA** might vary. In other words, what we see here would surely be more relevantly recorded as “This **mRNA/cDNA** has two **Deletions** relative to the **genomic** sequence which, presumably, attempts to represent the norm in the general population”? Just the reflection of an irretrievable pedant, but I am right, nevertheless!!!

In the documentation (see illustration in the **Summary** answer) it enigmatically states “The string is encoded with **RLE**.”. Just in case, **RLE** stands for **Run-length encoding** which is succinctly defined by **Wikipedia**. In a nutshell, it is a very simple form of data compression that recognizes that:

**XX**

can be compressed to:

**60X**

which has to be very effective for any data that has runs of identical characters of significant length. This is certainly the case here where one would expect long stretches of **Ms** in most alignments. Of course, life would get tricky if the data included numeric characters, but that is not an issue here<sup>4</sup>.

I think it worth mentioning, that this way of representing an alignment is a simplification of **CIGAR** format<sup>5</sup>. This format is used for **SAM** (Sequence Alignment Map) and **BAM** (Binary Alignment Map, exactly the same as **SAM**, except compressed) files. You will be engulfed in **SAM/BAM** files if you ever do any Next Generation Sequencing (**NGS**).

So, straight from the **SAM/BAM Format Specification** I copy the table of **CIGAR** enlightenment.

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.
- Sum of lengths of the M/I/S/=/X operations shall equal the length of SEQ.

Note, in particular, the extended range of **Operators** and the different meaning associated with the operator '**M**'. The operators '=' and '**X**' are such that any '**M**' is either an '=' or and '**X**' but never both. Which leaves one pondering when one might use '**M**' in preference to either an '=' or an '**X**'?

<sup>4</sup> The **Wikipedia** article shows how this complication might be overcome.

<sup>5</sup> There may or may not be some justification for calling the format **CIGAR**, but if there is, I have no idea what it might be.

Where is the **3<sup>rd</sup>** substitution in the Genomic Sequence?

From the alignment of **Exon 10**, the exon including the **3rd Replacement**, with a bit of squinting, it can be confirmed that the 3<sup>rd</sup> **Replacement** is at:

Base pair position **1,312** of the mRNA

Base pair position **25,042** of the **genomic** sequence

It might also have been relevant to ask which amino acid position corresponded to the **Replacement**. To discover this one would need to look at the alignment of **Exon 3**, where the coding begins.

More squinting, and I conclude the **A** of the **ATG** representing the initial **Methionine** of the protein coding region is at position **363**. That is, the **5' UTR** ends at position **362**. So the **Replacement** is at:

Base position  $1312 - 362 = 950$  of the protein coding region of the **mRNA**.

As  $950 / 3$  is 316 remainder 2, the Replacement is at codon position 2 of the 317<sup>th</sup> amino acid of the protein.

Cannot help thinking that **spline** might have helped a bit more here?

I also reflect that I cannot fully recall why I wanted to know where the mutation was, especially given we have decided to reject any chance that it might be a mutation of consequence. Oh well, some things a man must do, just because they are there to be done!!

Time to move on ... without checking my arithmetic. Bound to be right, I used to be a mathematics teacher you know! Several lifetimes ago.





From your investigations of **Local Alignment**:

Why do you suppose your aligned exons are not presented in the correct positional order?

To **Matcher**, the logical order in which to present the alignments is that governed by quality rather than position. So, the highest scoring alignment, rather than the first exon alignment, will be at the top of the list. I think this is generally logical. Once again, the program **splign**, knowing it was looking for an ordered set of exons, was more logical.

**DPJ – 2016.07.06**