TGAC

The Genome Analysis Centre™

BBSRC

Greater Norwich Development Partnership

TGAC/EMBL-EBI Summer School in Bioinformatics for Biological Researchers

25-29 July 2016

Basic Bioinformatics Sessions

Practical 2: Pairwise Alignment

Monday 4 July 2016

Sensitive Pairwise Alignment

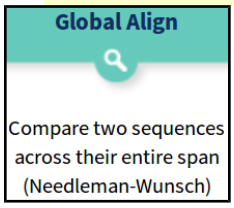
The purpose of this exercise is to look at some aspects of **Pairwise Sequence Alignment** using the most accurate methods available.

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

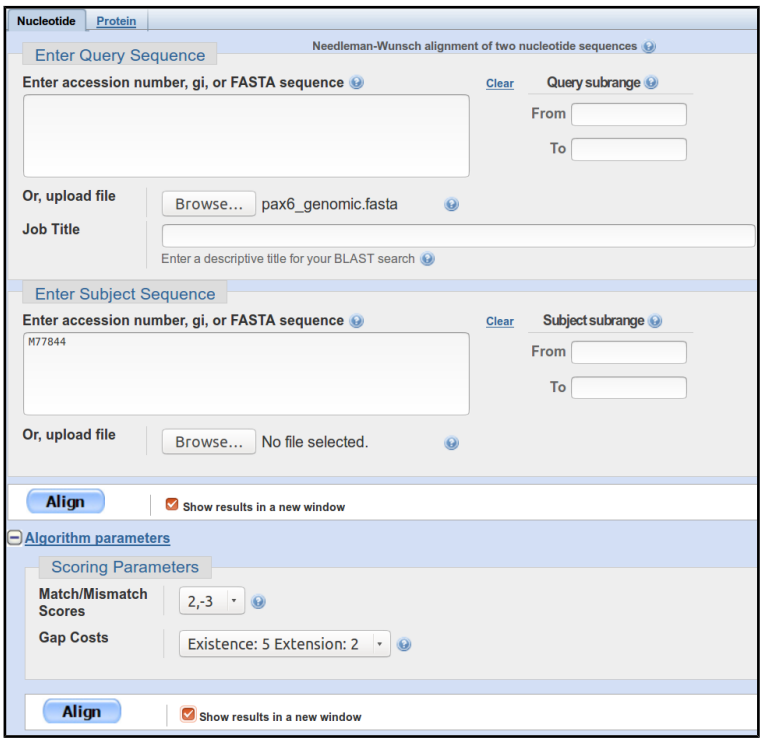
First the **global** approach. In a previous exercise, you already have used the **blast** facility at the **NCBI** to perform crude pairwise alignment. **blast** also offers a sensitive option, so maybe that would be a good place to start.

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).

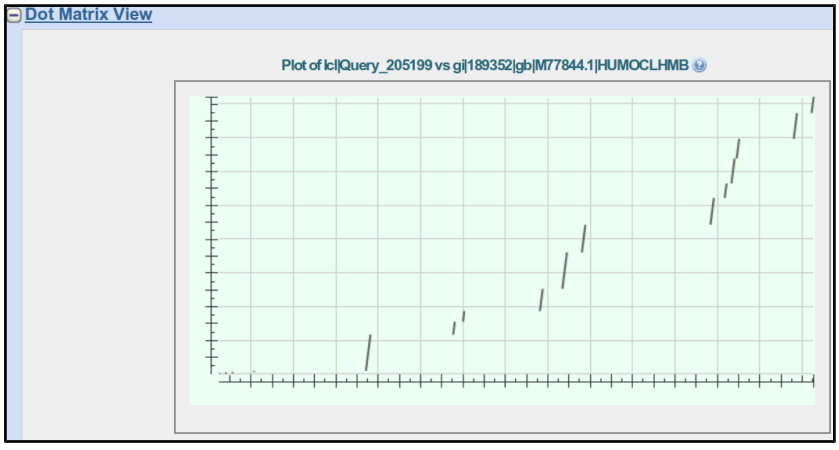


To specify the second sequence, you could load the file **pax6_mrna.fasta**, but just typing the corresponding **Accession** code in the appropriate box seems far more sophisticated, so that is what I chose to do.

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 aligment(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?

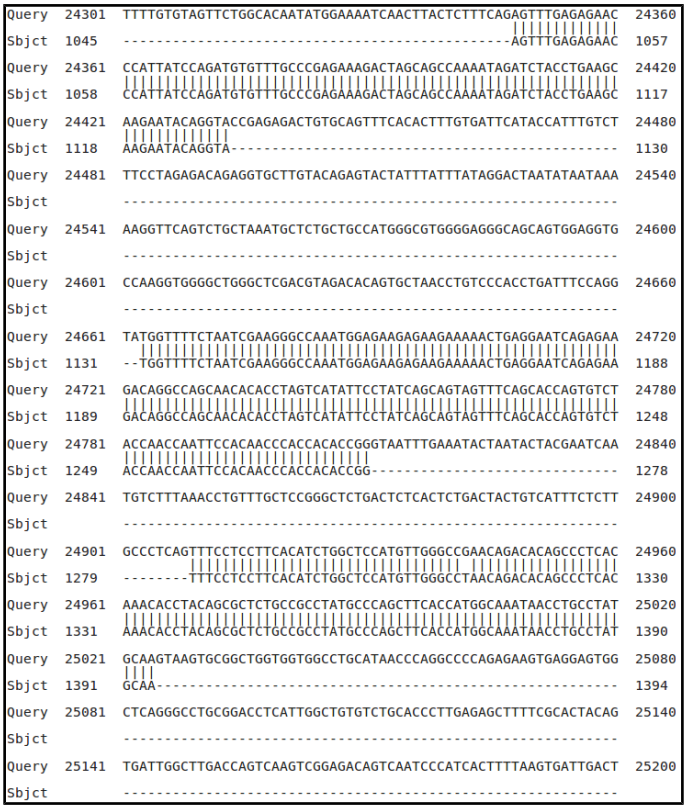
What are the gaps in between the lines?

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 to much, beyond noting that the mrna has some odd bases at the front.

Also, I have faith that the alignment you look at yeilds 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**.



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.

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. I 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? _____

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: 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	OUTPUT FORMAT
DNAfull	10	0.5	pair

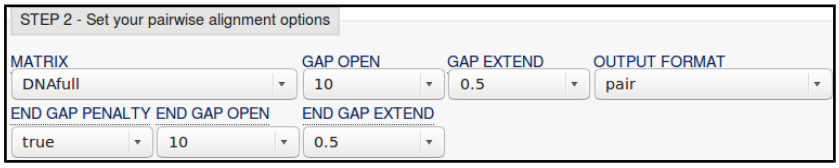
END GAP PENALTY	END GAP OPEN	END GAP EXTEND
false	10	0.5

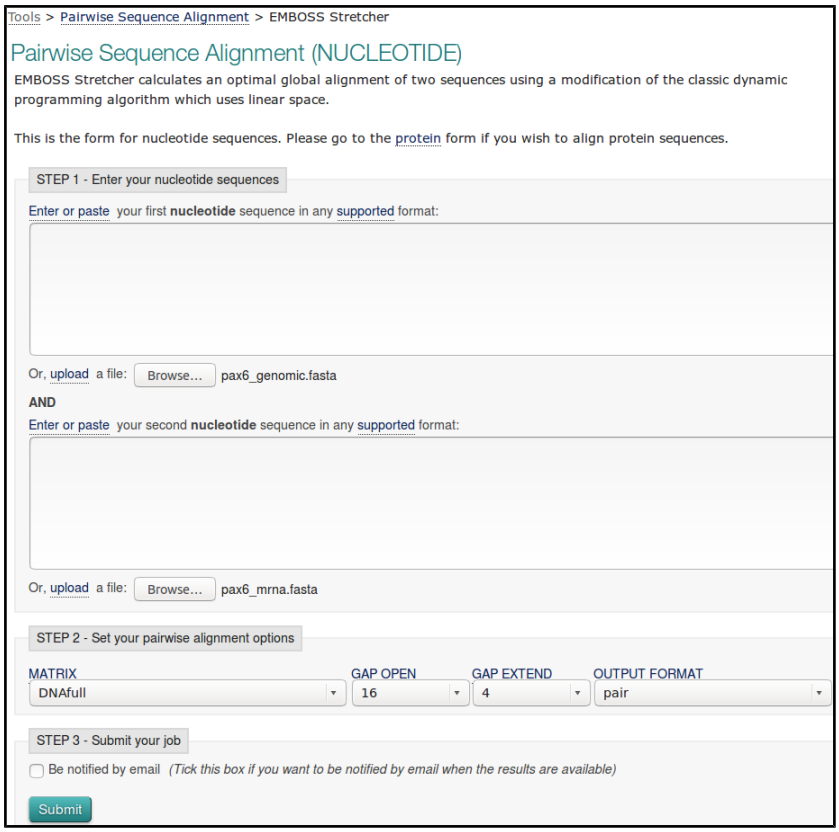
STEP 3 - Submit your job

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

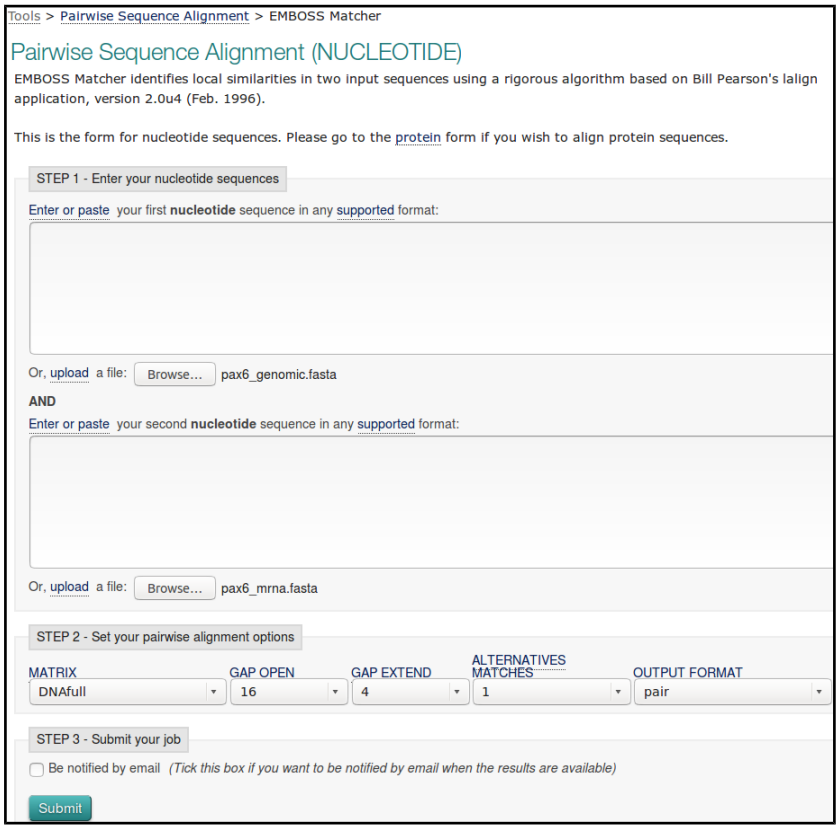
Submit

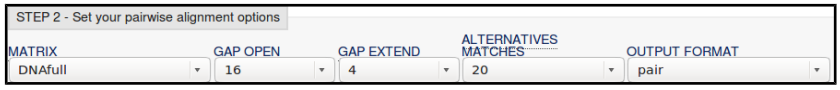
```
11              22101 TCTTGTAACACAATGTGGCCCGCTGCACGCCTCAAGAGAATC------CT  22144
                                                          .|||      .|
HUMOCLHMB           1 ------------------------------------TATCGATAAGTT     12

11              22145 TTTGTTGTCCGCGCTCATTGTAGCCTCAAAAT-TCTGCCCACGAAAGTTT  22193
                      |||.||        |.|||||       .||| ||||
HUMOCLHMB          13 TTTTTT--------TTATTGT-------CAATCTCTG-------------     34

11              22194 GCCAACGCTCCTGCCCCAGGAGTTTAATAGTTTCCCTTACTCGCGGGGCA  22243
                      .|||| .|||||||.|.|.|---||.|.|||||.|.|
HUMOCLHMB          35 ------TCTCCT-TCCCAGGAATCTGAGGATTGCTCTTACACAC------     71

11              22244 TTGTGCAGCGCTGAAAAGCAGCCCCTCGCTATTCAAGTGTTGGTGGTCA-  22292
                      ||.|.|.|.||  ||.||                ||||..|
HUMOCLHMB          72 -----CAACCCAGCAA--CATCC-------------------GTGGAGAA     95

11              22293 ---TCTCAATAG-ATCTCCAAGGGCCCATATGGTGGCCAGTGCCGATGAA  22338
                         |||||..|| |.||||
HUMOCLHMB          96 AACTCTCACCAGCAACTCC-----------------------------     114

11              22339 TCCGCCTGTTTAAATGGGGGAGAAAGTTGGGGTTTTAAAACAT-------  22381
                            ||||||      |.|..||   ..|||.|||.|||
HUMOCLHMB         115 --------TTTAAA------ACACCGT---CATTTCAAACCATTGTGGTC    147

11              22382 TTCAA--------AGTTCCTGAAAAGATCCC-----------ACT-----  22407
                      |||||        ||...|..||||.|.|||           |||
HUMOCLHMB         148 TTCAAGCAACAACAGCAGCACAAAAAAACCCCAACCAAACAAAACTCTTGA    197

11              23546 TACCTTGGGAATGTTTTGGTGA---GGCTGTCGGGATATAATGCTCTTGG  23592
                            ||||     |||   |||.|.|.||     |.||   |||
HUMOCLHMB         804 ----------ATGT-----TGAACGGGCAGACCGG-----AAGC---TGG    830

11              23593 AGTTTAAGACTACACCAGGCCCCT-TTTGGAGGCTCCAAGTTAATCC--A  23639
                      .|          ||||.|  |||| .|||           .||||  .
HUMOCLHMB         831 GG----------CACCCG--CCCTGGTTGG-----------TATCCGGG    856

11              23640 AATTTCTCTTAC----CATCCTATTCTTTTTGTTCCAGATGGCTGCCAGC  23685
                      .|.|||..|..|    ||.||||          ..|.|||||||||||||
HUMOCLHMB         857 GACTTCGGTGCCAGGGCAACCTA---------CGCAAGATGGCTGCCAGC    897

11              23686 AACAGGAAGGAGGGGGGAGAGAATACCAACTCCATCAGTTCCAACGGAGAA  23735
                      ||||||||||||||||||||||||||||||||||||||||||||||||||||
HUMOCLHMB         898 AACAGGAAGGAGGGGGGAGAGAATACCAACTCCATCAGTTCCAACGGAGAA    947

11              23736 GATTCAGATGAGGCTCAAATGCGACTTCAGCTGAAGCGGAAGCTGCAAAG  23785
                      ||||||||||||||||||||||||||||||||||||||||||||||||||||
HUMOCLHMB         948 GATTCAGATGAGGCTCAAATGCGACTTCAGCTGAAGCGGAAGCTGCAAAG    997

11              23786 AAATAGAACATCCTTTACCCAAGAGCAAATTGAGGCCCTGGAGAAAGGTG  23835
                      ||||||||||||||||||||||||||||||||||||||||||||||||||||
HUMOCLHMB         998 AAATAGAACATCCTTTACCCAAGAGCAAATTGAGGCCCTGGAGAA-----   1042

11              23836 ATAGAGTTTTTCAAAGTAGAGAAGCAGTAAATCAAAGTAAATGCCACATC  23885

HUMOCLHMB        1043 ------------------------------------------------     1042

11              24636 TAACCTGTCCCACCTGATTTCCAGGTATGGTTTTCTAATCGAAGGGCCAA  24685
                                      |||||||||||||||||||||||||||||||
HUMOCLHMB        1125 ---------------------CAGGTATGGTTTTCTAATCGAAGGGCCAA   1153

11              24686 ATGGAGAAGAGAAGAAAAACTGAGGAATCAGAGAAGACAGGCCAGCAACA  24735
                      ||||||||||||||||||||||||||||||||||||||||||||||||||||
HUMOCLHMB        1154 ATGGAGAAGAGAAGAAAAACTGAGGAATCAGAGAAGACAGGCCAGCAACA   1203

11              24736 CACCTAGTCATATTCCTATCAGCAGTAGTTTCAGCACCAGTGTCTACCAA  24785
                      ||||||||||||||||||||||||||||||||||||||||||||||||||||
HUMOCLHMB        1204 CACCTAGTCATATTCCTATCAGCAGTAGTTTCAGCACCAGTGTCTACCAA   1253

11              24786 CCAATTCCACAACCCACCACACCGGGTAATTTGAAATACTAATACTACGA  24835
                      |||||||||||||||||||||||||
HUMOCLHMB        1254 CCAATTCCACAACCCACCACACCG-------------------------   1277

11              24836 ATCAATGTCTTTAAACCTGTTTGCTCCGGGCTCTGACTCTCACTCTGACT  24885

HUMOCLHMB        1278 ------------------------------------------------   1277

11              24886 ACTGTCATTTCTCTTGCCCTCAGTTTCCTCCTTCACATCTGGCTCCATGT  24935
                                      |||||||||||||||||||||||||||||||
HUMOCLHMB        1278 ----------------------GTTTCCTCCTTCACATCTGGCTCCATGT   1305

11              24936 TGGGCCGAACAGACACAGCCCTCACAAACACCTACAGCGCTCTGCCGCCT  24985
                      ||||||-|||||||||||||||||||||||||||||||||||||||||||||
HUMOCLHMB        1306 TGGGCCTAACAGACACAGCCCTCACAAACACCTACAGCGCTCTGCCGCCT   1355

11              24986 ATGCCCAGCTTCACCATGGCAAATAACCTGCCTATGCAAGTAAGTGCGGC  25035
                      ||||||||||||||||||||||||||||||||||||||||||
HUMOCLHMB        1356 ATGCCCAGCTTCACCATGGCAAATAACCTGCCTATGCAA-----------   1394

11              25036 TGGTGGTGGCCTGCATAACCCAGGCCCCAG--AGAAGTGAGGAGTGGCTC  25083
                            ||       |||||.||||||  |||
HUMOCLHMB        1395 ---------CC-------CCCAGTCCCCAGCCAGA---------------   1413

11              25084 AGGGCCTGCGGACCTCAT-----TGGCTGTGTCTG--CACCCTTGAGAGC  25126
                            |||     ||||||     ||..||   |||  ||||      |||
HUMOCLHMB        1414 ----CCT-----CCTCATACTCCTGCATG---CTGCCCACC------AGC   1445
```


STEP 2 - Set your pairwise alignment options
MATRIX
DNAfull
GAP OPEN
10
GAP EXTEND
0.5
OUTPUT FORMAT
pair
END GAP PENALTY
true
END GAP OPEN
10
END GAP EXTEND
0.5


Tools > Pairwise Sequence Alignment > EMBOSS 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: 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
DNAfull
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)
Submit

```
11              16670 CACTTCCCCTAT---GCAGGTGTCCAACGGATGTGTGAGTAAAATTCTGG  16716
                      ||.|||||..||   ||||||||||||||||||||||||||||||||||
HUMOCLHMB         485 CATTTCCCGAATTCTGCAGGTGTCCAACGGATGTGTGAGTAAAATTCTGG    534

11              16717 GCAGGTATTACGAGACTGGCTCCATCAGACCCAGGGCAATCGGTGGTAGT  16766
                      ||||||||||||||||||||||||||||||||||||||||||||||||||||
HUMOCLHMB         535 GCAGGTATTACGAGACTGGCTCCATCAGACCCAGGGCAATCGGTGGTAGT    584

11              16767 AAACCGAGAGTAGCGACTCCAGAAGTTGTAAGCAAAATAGCCCAGTATAA  16816
                      ||||||||||||||||||||||||||||||||||||||||||||||||||||
HUMOCLHMB         585 AAACCGAGAGTAGCGACTCCAGAAGTTGTAAGCAAAATAGCCCAGTATAA    634

11              16817 GCGGGAGTGCCCGTCCATCTTTGCTTGGGAAATCCGAGACAGATTACTGT  16866
                      ||||||||||||||||||||||||||||||||||||||||||||||||||||
HUMOCLHMB         635 GCGGGAGTGCCCGTCCATCTTTGCTTGGGAAATCCGAGACAGATTACTGT    684

11              16867 CCGAGGGGGTCTGTACCAACGATAACATACCAAGCGTAAGTTCATTGAGA  16916
                      ||||||||||||||||||||||||||||||||||||....|||.|.|..
HUMOCLHMB         685 CCGAGGGGGTCTGTACCAACGATAACATACCAAGCGTGTCATCAATAAAC    734

11              16917 ACA--TCTGCCCTCCCTGCC  16934
                      |.|  |||-|.|..||||.|
HUMOCLHMB         735 AGAGTTCTTCGCAACCTGGC    754
```






```
#=======================================
#
# 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 GCTGGACGCCACCCGGCGCCAGA--GCCGGGC---CTGAGGAGCGGGGTC   2397
                     ||.||||.|||||||||...|||  |..|.||   ||.|. |||||||.|
HUMOCLHMB        425 GCCGGACTCCACCCGGCAGAAGATTGTAGAGCTAGCTCAC-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
```

Sequence Analysis

The overall target remains to discover all we can about **Aniridia** and its genetic causes. Here we will see how information can be derived from analysis of sequence data directly. Many of the analysis tools and methods we will use and discuss will be those that were used to generate the "ready made" answers you have already investigated. The conclusions of this section should not therefore contradict those you have already reached.

To start the analysis section, you should have data files including:

Two PCR primers associated with the **PAX6** gene in a way we have not yet fully determined, in a file called:

pax6_primers.fasta

The sequence of the Paired box domain of the most prolific human **PAX6** isoform in a file called:

pax_domain.fasta

The genomic sequence of the whole of the **PAX6** gene, plus an extra 500 bases in each direction, in a file called

pax6_genomic.fasta

The canonical human **PAX6** isoform in a file called:

pax6_human.fasta

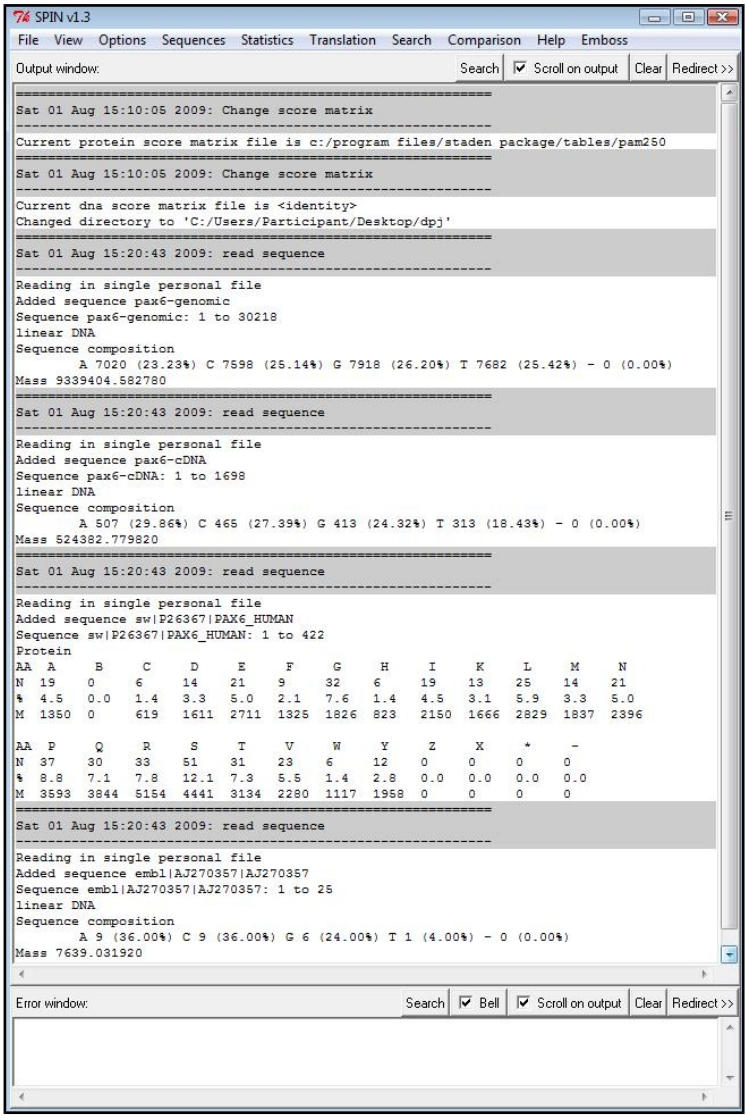
The file **pax6_genomic.fasta** contains the sequence of **PAX6** from a person(s) who, it can be assumed, does not suffer from **Aniridia**. Given a sequence from a person who **does** suffer from **Aniridia**, an obvious first step of investigation might be to compare the sequence from the affected source with the wild type. It would be reasonable to speculate that differences might explain the disease. Here we have to cheat a bit. I have provided an mRNA sequence we will suppose you sequenced from an **Aniridia** patient. It is stored in a file called:

pax6_cdna.fasta

which you will find in a directory called **Working Directory** on your **Desktop**.

Internet resources will be employed for many analyses. Software that runs under windows on your workstation will be used very occasionally. All windows software has been installed, without **Administrator** privileges, using an Installer that will be made available to you.

The **EMBOSS** package[1], using a program called **spin**[2] to provide a **G**raphical **U**ser **I**nterface (**GUI**) will be used for most of the workstation analyses. This is not ideal, particularly as support for **spin** is defunct, but the options[3] are few.



Start **Spin** in the usual windows way (it is part of the **The Staden Package**, which is in the **Start menu** under **Sequence Analysis Tools Installer**). Proceed by setting **spin** to manage its files somewhere sensible. Strangely, its default inclination is to create user output files in its installation directory? This is silly, so select **Change directory** from the **File** pull down menu and select a more sensible default directory. I suggest you might make a directory in your desktop named as you please for this purpose[4].

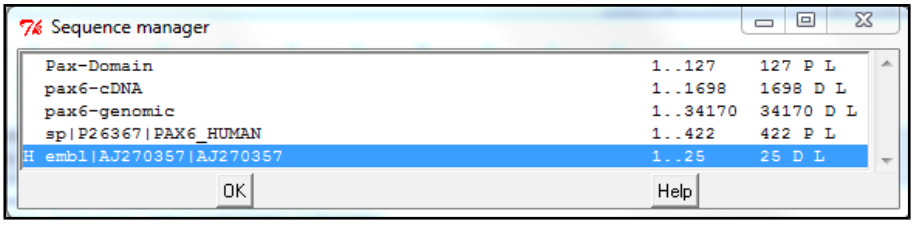
1 Documentation pages for **EMBOSS** can be accessed at the **EMBOSS** home page at: **http://emboss.sourceforge.net**

2 A part of the **Staden Package**.

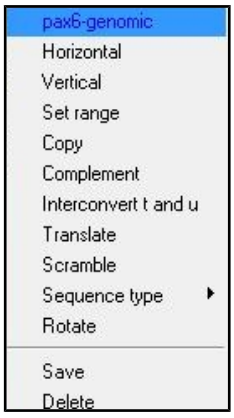
3 That is **FREE** options!

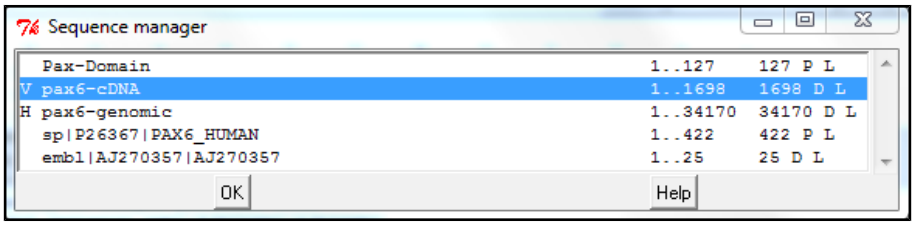
4 It would be very sensible to move **pax6_cdna.fasta** and all the files you have created thus far (**pax6_genomic.fasta**, **pax6_primers.fasta**, **pax_domain.fasta** and **pax6_human.fasta** in particular) into this directory.

Begin by loading all the sequence files you have gathered thus far. From the **File** drop down menu select **Load Sequence** and then **Simple**. This will present a new dialogue box for you to specify a sequence to be loaded. Leave the choice of **Database** as the default **Personal File**, as the all the sequences needed initially are stored in local files. To the right a text box awaits the file path for a sequence file. Use the **Browse** button to find and enter the directory you set up as your default[5]. Select all the sequence files listed above[6] and click **Open** and then **OK** in the **Load Sequence** window. From this point onwards the package will refer to these sequences by their **fasta** titles (e.g. **pax6-cDNA** and **pax6-genomic)**[7].



Take some time to become acquainted with the **spin Sequence manager**. This provides access to all loaded sequences. Its main purpose is to enable sequences to be selected for analysis. From the **File** drop down menu, select **Sequence Manager**. A window will appear listing the sequences showing their start and end, length and type (**D** for DNA or **P** for Protein). The last sequence to be loaded will be labelled **H** (for **H**orizontal)[8]. The sequence set to **H**orizontal is the one that **spin** will analyse by default. Ensure that **pax6-genomic** is set to **H**orizontal by right clicking its name and selecting **Horizontal**. Right clicking an entry brings forth a menu offering several more possibilities than we will use in these exercises. For example, it is possible to **Set range**, that is specify regions of entries to be analysed separately. Also it is possible to **Translate** or **Compliment** DNA sequences to create new entities for analysis.





In similar fashion, make **pax6-cDNA** the **V** (for **V**ertical) sequence[9].

Leave the **Sequence manager** by clicking **OK**. You are now be ready to proceed.

5 If you did things correctly, you should be in the right place without having to move anywhere.

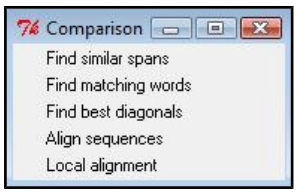
6 You can select multiple files in a number of ways including holding the **Ctrl** key down whilst clicking on each of the files you require.

7 **spin** will describe all the sequences it has loaded. Note that only the first of the primer sequences was loaded, sadly **spin** does not recognise that **pax6_primers.fasta** is a multiple sequence file. This will not matter to us as the exercises stand at present.

8 This refers, somewhat bizarrely, to the placement of the sequence on Cartesian axes.

9 This only really makes complete sense when you are computing dotplots, which you are just about to do. **spin** is limited in that it allows a maximum of 2 sequences to be selected. When only one sequence is required, the **H**orizontal selection is used.

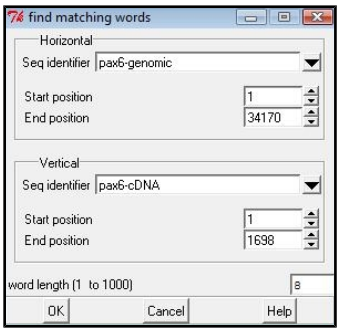
Pairwise Sequence Comparison



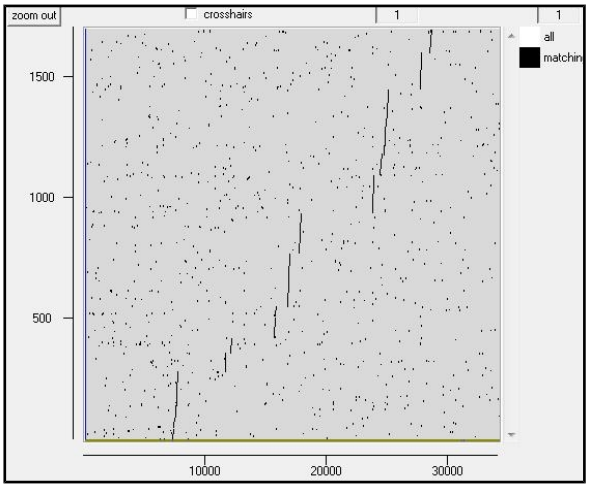
You will be using options from **spin**'s **Comparison** menu for the next few steps. To set up that menu for easy access, click on **spin**'s **Comparison** button and then on the dotted line of the menu that will appear. Position the menu window that "tears off" conveniently.

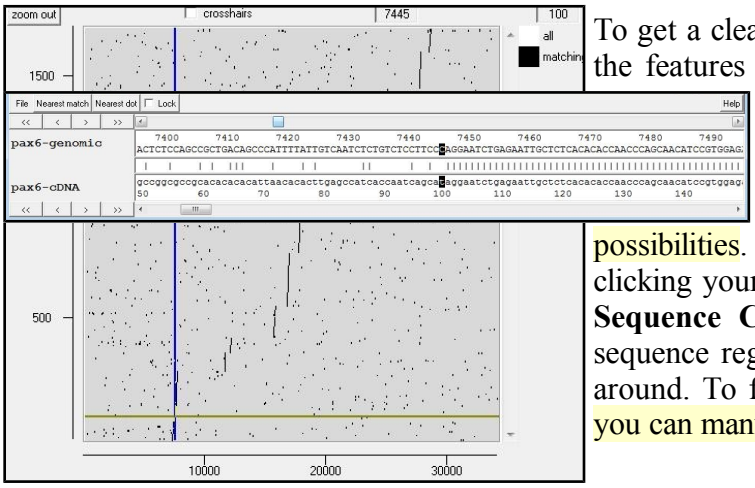
Graphical pairwise sequence comparison - DotPlots

An intuitive graphical representation of the comparison between two sequences is a **dotplot**. One sequence is represented on each of two Cartesian axes and significant matching regions are indicated as a plot. Further detail will be provided by presentation. These plots are used to indicate which sections of sequence might align with each other convincingly and thus warrant further investigation.

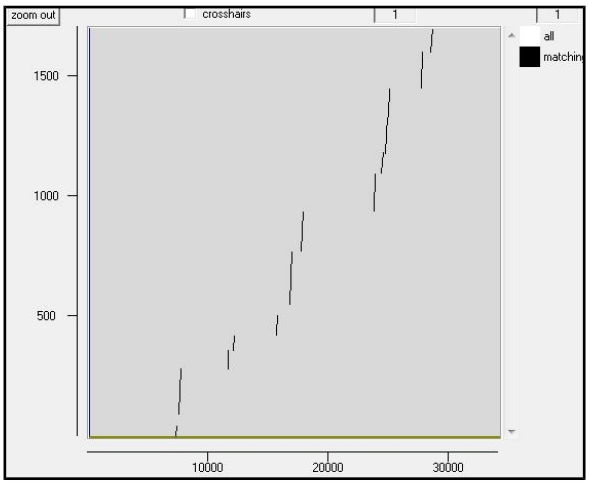


From your **Comparison** menu, select **Find matching words**, which offers a simple and fast way to draw a dot plot. Check that you have the sequences you expect in the **Horizontal** and **Vertical** positions[10]. Accept the default **word length** of **8**[11] and click **OK**. A **dotplot** will burst forth with the genomic sequence along the bottom axis, and the cDNA sequence on the vertical axis, as illustrated. As far as the resolution of the picture allows one to judge, there appears to be **11** matching regions.





To get a clearer view of the features of any of the graphics you produce with **spin**, you can always make the picture window bigger, use the **X** and **Y** controls at the bottom to reshape the view, hold the **Ctrl** key down and select a region with your right mouse button to **zoom in** to a feature[12]. Try any or all of these possibilities. You can also view the sequence of any feature by double clicking your left hand mouse button near that feature. Try this and a **Sequence Comparison Display** will appear. You can control the sequence region in view by dragging the cross hairs that will appear around. To fix the display exactly on a given feature, get as near as you can manually and then click on the **Nearest match** button.



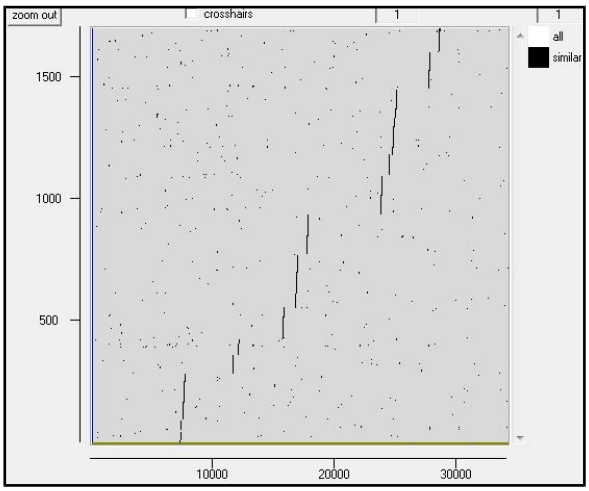
Using a **word length** as small as **8** shows the meaningfully matching regions quite well in this case, but there is quite a bit of unnecessary "noise"[13]. The features we wish to be shown are long, so it is possible to use a bigger word size without losing matches of importance.

Re-run **Find matching words**, exactly as before but this time, select a **word length** of **50**. Your new dotplot will be drawn on top of you first and so will thus be difficult to see. A plot can be repositioned by moving around its configuration button (the corresponding square in the top right hand corner) with the middle mouse button. You could move the new dotplot so it is above or below the first or into a separate window (the best choice) by dragging its configuration button into an unoccupied place on your desktop. Your second plot is essentially as the first, without the noisy background. Now it looks more like twelve matching regions. We compare genomic sequence with cDNA, so it is reasonable to assume the matching regions of this plot represent the exons found in a transcript of **PAX6**.

10 If you need to, use the pull down menus provided to select from any of the sequences in your **Sequence manager**.
11 Thus asking the program to indicate the presence of exactly matching regions of **8** base pairs between the chosen sequences.
12 There is a **zoom out** button at the top of the display to undo any **zooming in** you try.
13 I.e. dots that do not represent anything interesting.

Is the number of matching regions consistent with the number of exons you might expect?_____

If not, can you explain the discrepancy?_____

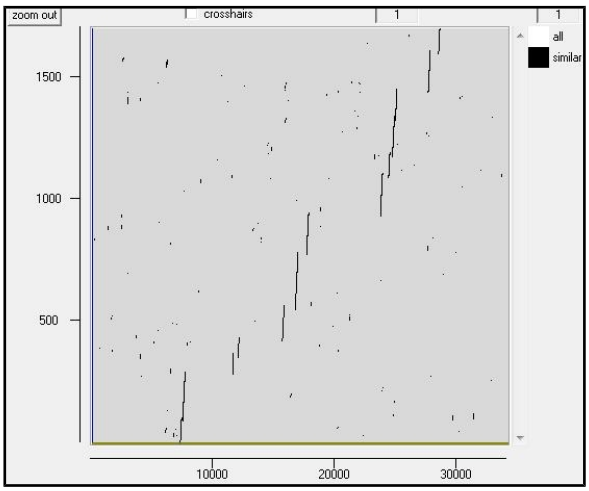


The second plot is clearer. Noise has been eliminated without loss of "real" features. The choice is not always so clear. It can be necessary to experiment using different parameters.

The algorithm you have used thus far is fast but crude. Adequate for these sequences between which all real matches are long and almost exact. However, in most circumstances a more meticulous slower[14] approach might be required in which "similar" as well as exactly matching words are recognised. From your **Comparison** menu, select **Find similar spans**, accepting the default **window length** of **11** and **minimum score** of **10**[15], click **OK**[16]. Your plot should be as illustrated.

Again, from your **Comparison** menu, select **Find similar spans**. This time, set the **window length** to **50**. **spin** automatically resets its **minimum score** field to a new default of **28**. Click **OK**. Separate out your plots so that they can be seen clearly. No surprises? A less noisy background is the most obvious effect. Also you should be able to see that the "real" features appear to have grown a little.

Why do you suppose that might be?_____



The dotplot can be useful as a rapid overview of the similarity between two sequences. Alone, it provides insufficient detail. For this, we need to generate textual alignments.

spin produces helpful graphics when textually aligning sequences. These graphics are most informative when superimposed over a corresponding dotplot. Here, the most useful dotplot is the simplest. The one you generated using the "**Find matching words**" algorithm with a word length of **50**. If you have not already done so, throw all the others away. If necessary, recreate the required dotplot.

Pairwise textual sequence alignment

Now make some alignments that will show the detail of the dotplot comparisons. The algorithms used are more rigorous than those used for searching databases, often producing more revealing alignments. Thus it can be a good idea to use these tools on similar sequences identified by database similarity searches

Global sequence alignment

A global alignment is one that aligns two sequences over their entire lengths. In **spin**'s **Comparison** menu, the option **Align sequences**[17]. Click on **spin**'s **Align sequences** option. Click **OK** accepting the defaults[18]:

score for match	**(4)**	value added to the alignment score for each correctly matched base
score for mis-match	**(-2)**	value added to the alignment score for each incorrectly matched base
penalty for starting gap	**(8)**	value subtracted from the alignment score for each gap
penalty for each residue in gap	**(1)**	value subtracted from the alignment score for each element of a gap beyond the first

spin represents the computed alignment graphically, over the dotplot. It should suggest that it has successfully aligned the group of exons around **23,500** to **25,000** in the genomic sequence and **940** to **1,450** in the mRNA[19], but then failed with the rest.

14 Not that the speed difference will be apparent with sequences of the size we are comparing here.

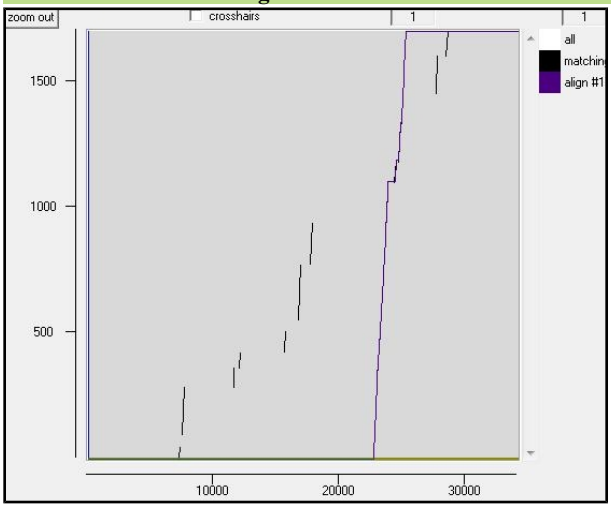
15 Thus matched windows of length **11** in which **10** of the aligned bases are identical are regarded as significant. By default, **spin** awards **1** point to matching bases and **0** points to mismatched bases. The equivalent program in the **EMBOSS** package, **dotmatcher**, awards matching bases **+5** and penalises mismatched bases **-4**. The full **EMBOSS** default DNA comparison scoring matrix can be found in an appendix.

16 Depending how you left you displays, you might need to separate out your dotplots in order to see them clearly.

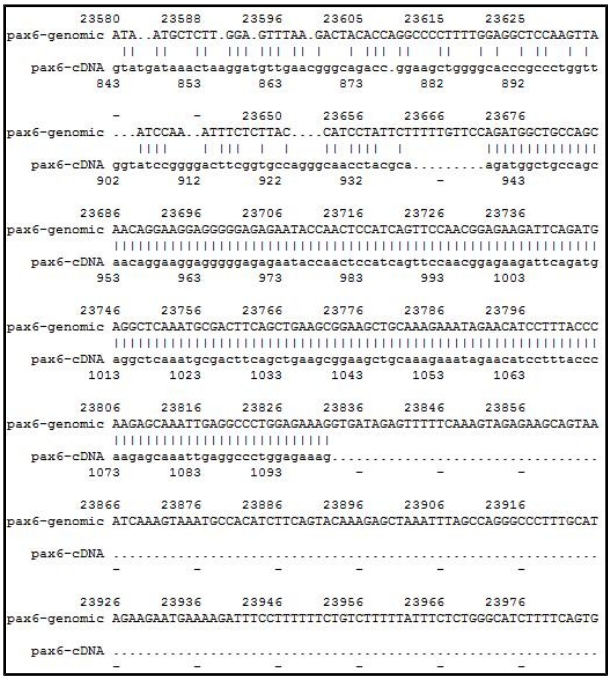
17 The **EMBOSS** package offers the program **needle**, which is a rigorous implementation of the Needleman-Wunsch algorithm for global alignment. **EMBOSS** also offers a less rigorous (i.e. faster and sloppier) program called **stretcher**.

18 A complete description of these parameters will be included in a presentation. Soon, if it has not already occurred.

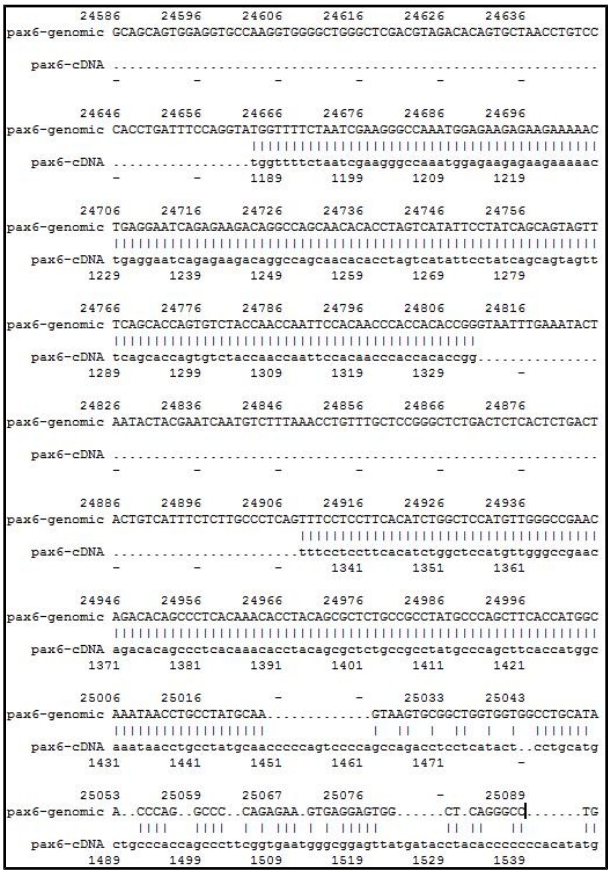
19 Click in the **crosshair** box and use the crosshair to get a good idea of where features are in your plots. I think the cross-hair is very irritating to leave active, so I suggest you click it off again once its purpose is fulfilled.



Examine how your alignment graphic matches the dotplot by right clicking its configuration button and using the **Hide**/**Reveal** option.



Now inspect the textual alignment in **spin**'s **Output window**. As the graphic suggests, the mRNA has been prefixed with over **22,000** minus signs to make it of equal length to the genomic sequence. There follows an entirely unconvincing (given the evidence of the dotplots) alignment of the first few exons of the mRNA with the region around **23,000** in the genomic sequence.



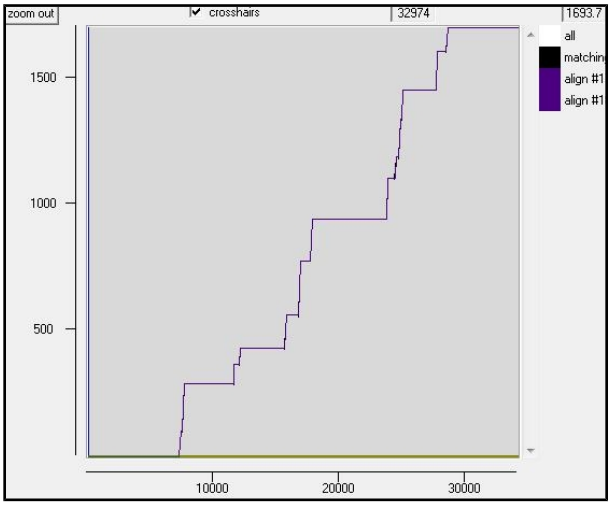
The **4** exons around **23,500-25,000** in the genomic sequence and **940-1,500** in the mRNA are aligned convincingly. The final mRNA exons are aligned, with an excess of poetry, with the bases after **25,000** of the genomic sequence.

The mRNA is then desperately padded to reach the end of the alignment. The bizarre padding at each end of the mRNA is required to fulfil the requirement of a global alignment to be “end to end”. The entirety of both sequences must be included in the alignment.

How many convincingly aligned regions did you see?_____

Roughly how many did you expect?_____

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



Try the alignment again, this time with much "cheaper" gaps. First **Hide** the plot associated with your first global alignment attempt. This should leave the dotplot in clear view. Now click on **spin**'s **Align sequences** option once again. This time, set the **penalty for each residue in gap** to **0**. This amounts to asking that all gaps be penalised exactly **8** points no matter how long they be. Click **OK**.

Now the alignment graphic shows the alignment dutifully passing though all the matching regions identified by the dotplot. Take a look at the textual output associated with this second alignment.

How many matching regions are there this time?_____

Is the count **now** roughly as you would expect?_____

These global alignments show how misleading it can be to run programs without carefully considering their assumptions and parameter values. The second time around, you achieved the "correct" alignment, but only by making the cost of gaps so cheap that huge introns could be mistaken for normal insertion/deletion events. This only worked because the mRNA/genomic sequences came from the same organism and the corresponding exons were effectively identical[20]. If the comparison was between homologous sequences from different organisms[21], it would usually be necessary to use gap penalties that reflected the way real insertion/deletion events occurred in the exons. Declaring almost free gaps to cope with introns would rarely succeed[22].

The best solution is to accept that the alignment between a cDNA/mRNA and genomic sequence is a special case. A general alignment program will not make the right assumptions for such alignments and is thus the wrong tool for the job. There is a program in the **EMBOSS** package, 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[23]. To investigate, go to the home of **splign** at:

`http://www.ncbi.nlm.nih.gov/sutils/splign`

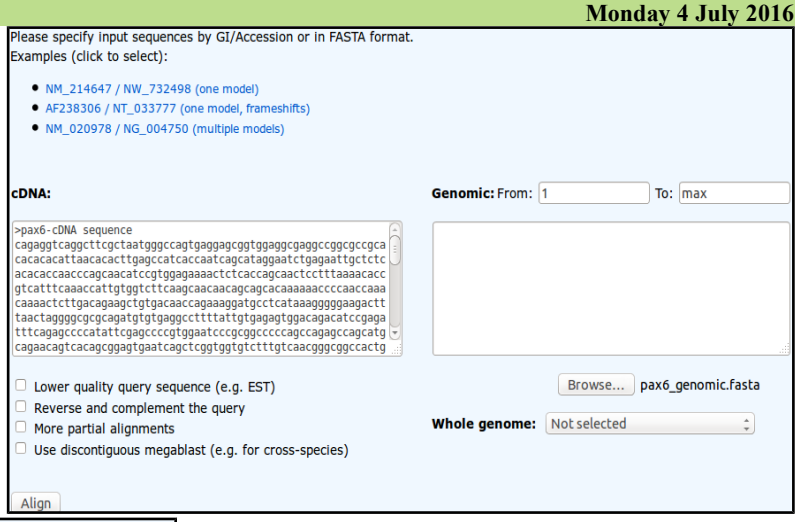
20 and so would align correctly given almost any gap penalties, or other parameter settings that might have been chosen.

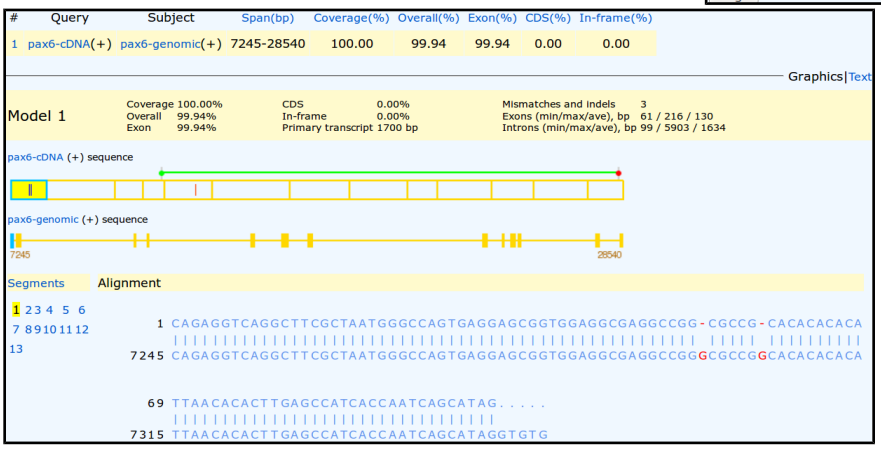
21 Comparing mRNA from one organism with genomic sequence from another is one way of investigating gene structure in newly sequenced genomic sequence.

22 One very simple solution to this problem offered by the **GCG** (now defunct) package version of **needle** (a program called **gap**) was to offer a gap penalty ceiling. The program would compute a gap penalty in the normal way until a given "cut off" was reached. **gap** would then assume it was trying to stretch over an intron rather than allow for an insertion/deletion, so it would allow the gap to increase at will without further raising the penalty. This worked much better than my intuition suggested it should!

23 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.

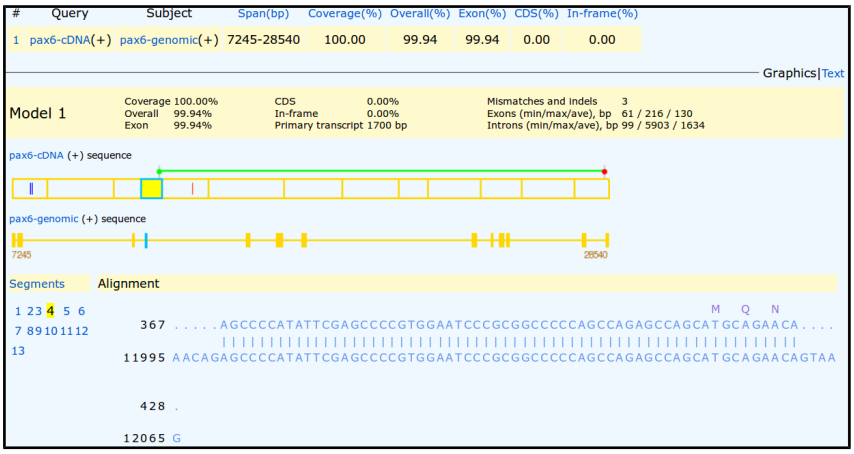
Click on the **Online** button. In the **Genomic** section, **Browse** to upload **pax6_genomic.fasta**. In the **cDNA** section, paste the sequence **pax6_cdna.fasta**. Where **cDNA** and **Genomic** sequences share exons that are nearly identical, **splign** uses the comparison algorithm **megablast** (default choice). Where exons are less similar (e.g. when the **cDNA** and **Genomic** sequences are from different organisms) the more sensitive option **discontinuous megablast**, might be a better choice[24]. Note the option to compare your **cDNA** with a **Whole genome** (including Human). Today, the default options are fine. Click the **Align** button.

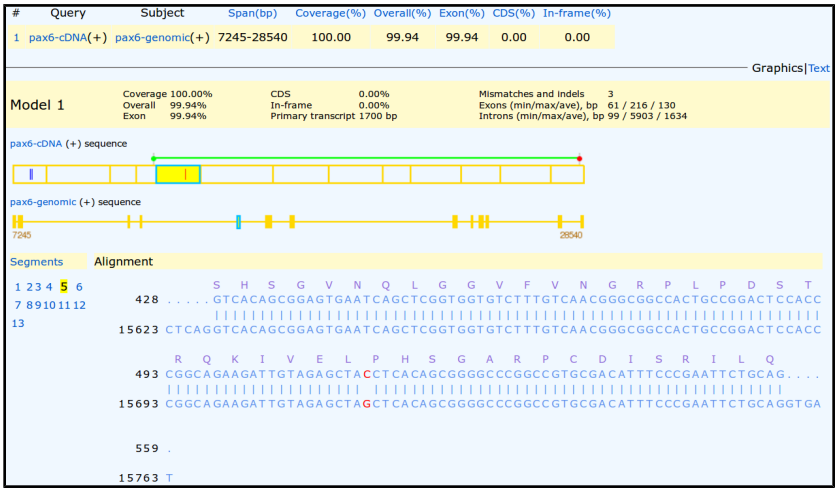




Your results will appear showing the cDNA split into **13** sections (the predicted exons) corresponding to **13** regions of the genomic sequence indicated by yellow rectangles. The first exon alignment is displayed showing two **cDNA** deletions. Though these are in a non-coding region, they could 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 of the cDNA are illustrated by the bar above the cDNA display.

Click on the exon including the green **Start** codon (the fourth). The first coding exon is now displayed. The statistics at the top of the display include the claim that there are three discrepancies (**Mismatches and indels**[25]) between the **cDNA** and **Genomic** sequences. Two of these are the deletions we have already seen in the first exon of the cDNA. The third is indicated by the red bar in the fifth exon of the **cDNA** display.





Click on the fifth exon section of the cDNA display.

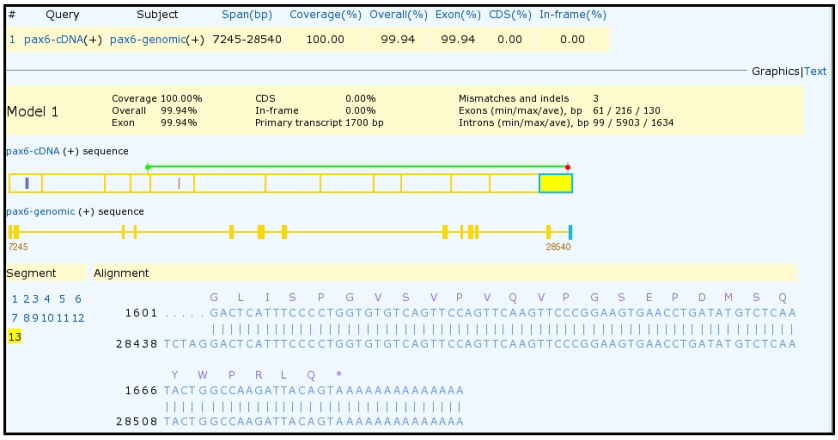
The third difference, a substitution, should be clear to see. Given it changes the coded protein, this substitution is likely to be the most significant.

What is the amino acid corresponding to this position in the mRNA of the aniridia patient[26]?_____

24 Why this is so will be considered later when we look at the database searching program **blast**.

25 An **indel** is either an **in**sertion or a **del**etion, depending upon which of the aligned sequences you are considering.

26 How this substitution affects the protein is not clear. Translating both the **Genomic** sequence and the **cDNA** might have helped? The **Standard Genetic Code** is in an Appendix, or look back to the **Uniprot** feature table for **PAX6_HUMAN**. It is the **Natural Variant** at position **33**.



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.

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

#	Query	Subject	Span(bp)	Coverage(%)	Overall(%)	Exon(%)	CDS(%)	In-frame(%)
1	pax6-cDNA(+)	pax6-genomic(+)	7245-28540	100.00	99.94	99.94	0.00	0.00

Graphics|Text

#	Query	Subject	Idty	Len	Q.Start	Q.Fin	S.Start	S.Fin	Type	Details
+1	pax6-cDNA	pax6-genomic	0.981	103	1	101	7245	7347	<exon>GT	M53IM5IM43
+1	pax6-cDNA	pax6-genomic	1	188	102	289	7447	7634	AG<exon>GT	M188
+1	pax6-cDNA	pax6-genomic	1	77	290	366	11537	11613	AG<exon>GC	M77
+1	pax6-cDNA	pax6-genomic	1	61	367	427	12000	12060	AG<exon>GT	M61
+1	pax6-cDNA	pax6-genomic	0.992	131	428	558	15628	15758	AG<exon>GT	M86RM44
+1	pax6-cDNA	pax6-genomic	1	216	559	774	16686	16901	AG<exon>GT	M216
+1	pax6-cDNA	pax6-genomic	1	166	775	940	17606	17771	AG<exon>GT	M166
+1	pax6-cDNA	pax6-genomic	1	159	941	1099	23674	23832	AG<exon>GT	M159
+1	pax6-cDNA	pax6-genomic	1	83	1100	1182	24348	24430	AG<exon>GT	M83
+1	pax6-cDNA	pax6-genomic	1	151	1183	1333	24660	24810	AG<exon>GT	M151
+1	pax6-cDNA	pax6-genomic	1	116	1334	1449	24909	25024	AG<exon>GT	M116
+1	pax6-cDNA	pax6-genomic	1	151	1450	1600	27602	27752	AG<exon>GT	M151
+1	pax6-cDNA	pax6-genomic	1	98	1601	1698	28443	28540	AG<exon>AA	M98

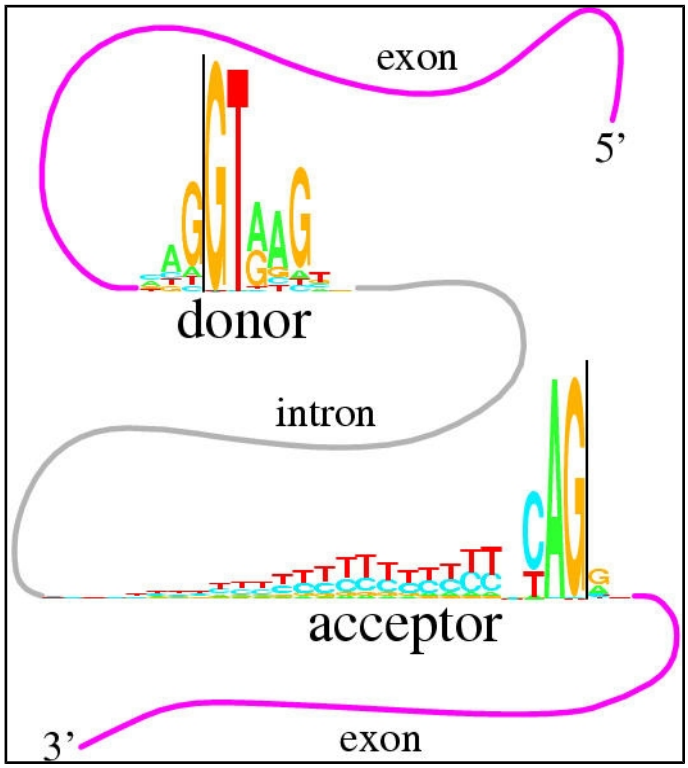
How do you interpret the **Details** column for exons 1 and 5?_____

Where is the substitution in the aniridia patient mRNA?_____

Where is the substitution in the Genomic Sequence?_____

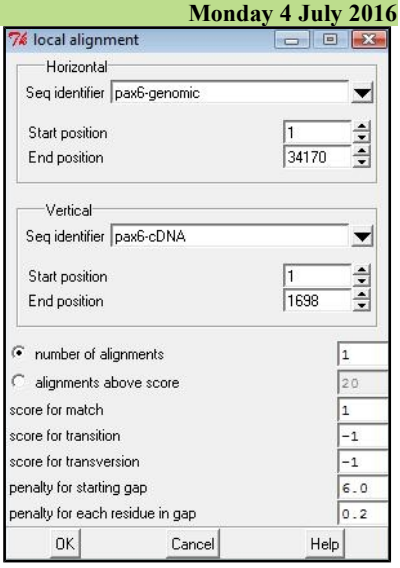
Compare the predicted **splign** intron/exon boundaries with the conservation suggested by the logo[27]?_____

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



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

Local sequence alignment

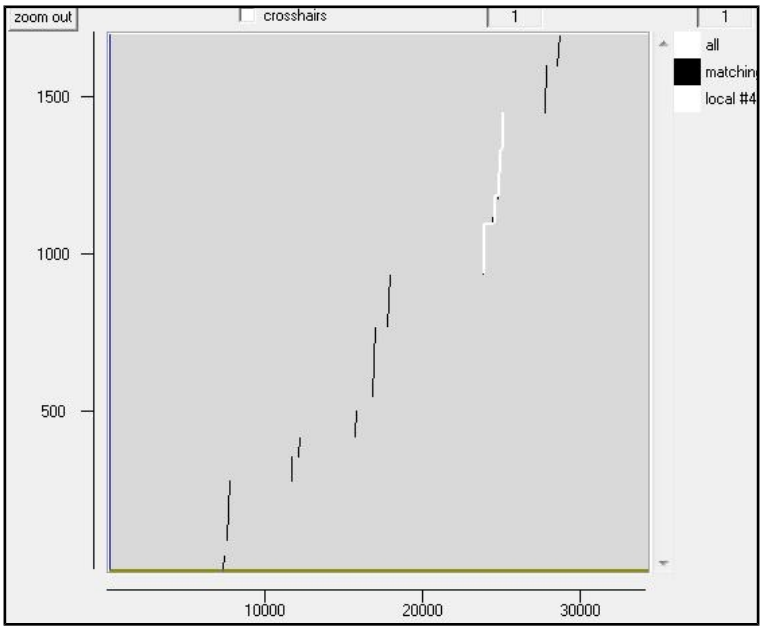


Global sequence alignment algorithms align sequences over their entire lengths. **Local** alignment methods searches for regions of similarity and need not include the entire length of the sequences. It is important to select the type of alignment that makes best sense for your sequences. Here, we expect an ordered sequence of matching regions (the exons) to occur in both sequences but spaced very differently. With an effort (or preferably appropriate software), we can get reasonably sensible alignments using a global or a local approach. However, if our sequences represented multi-domain proteins that shared some domains but not others, or the same domains but in differing orders, or proteins where there have been regions of duplication, a single sensible global alignment would not exist. A local approach would be essential.

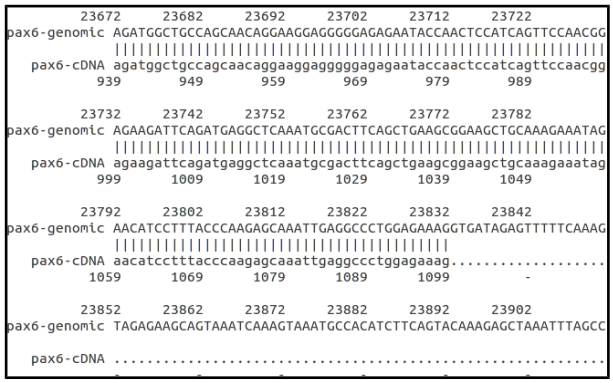
Move back to **spin**, tidy away all previous graphics plots except one clear **dotplot** between **pax6-genomic** and **pax6-cDNA**. From **spin**'s **Comparison** menu, select **Local alignment**. Note that:

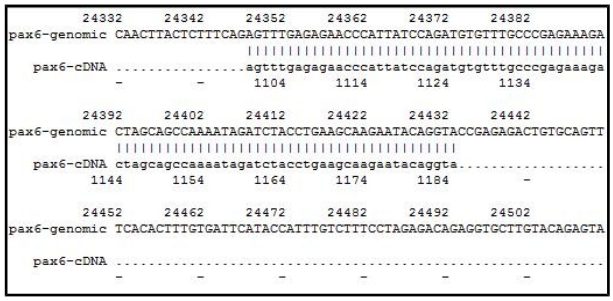
- in common with most local alignment programs, the default **number of alignments** to be reported is **1**[28]
- **transversions** and **transitions** may be penalized differently
- gaps are slightly cheaper than for **gobal** alignments

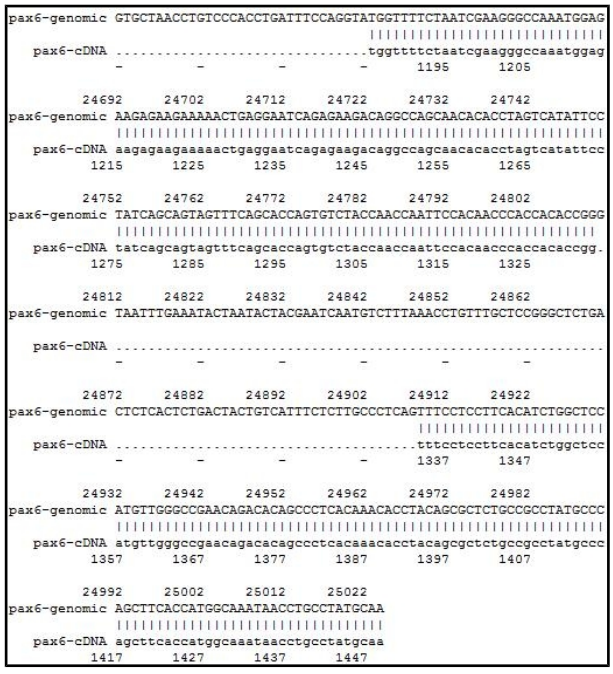
Accept all the defaults and click **OK**. The graphic suggests that the single "best" region that **spin** has elected to locally align, is the region around **25,000** in the genomic sequence where there are **4** exons that are close together. It was this region around which the global alignment algorithm chose to build its first attempt.



The choice of this region is primarily due to the choice of gap penalties. If bigger gap penalties were selected (increase **penalty for each residue in gap** to **1**, say) **spin** would cease to regard the **4** exons around **25,000** as one entity. Their individual alignment scores would look less attractive. Eventually, the region around **16,700** (the longest single exon) would easily outscore the rest. If you have time, try it.

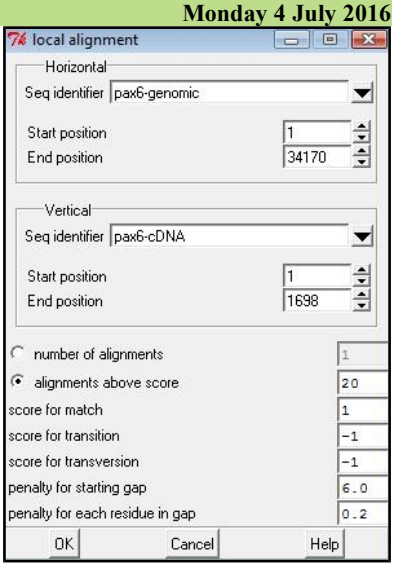






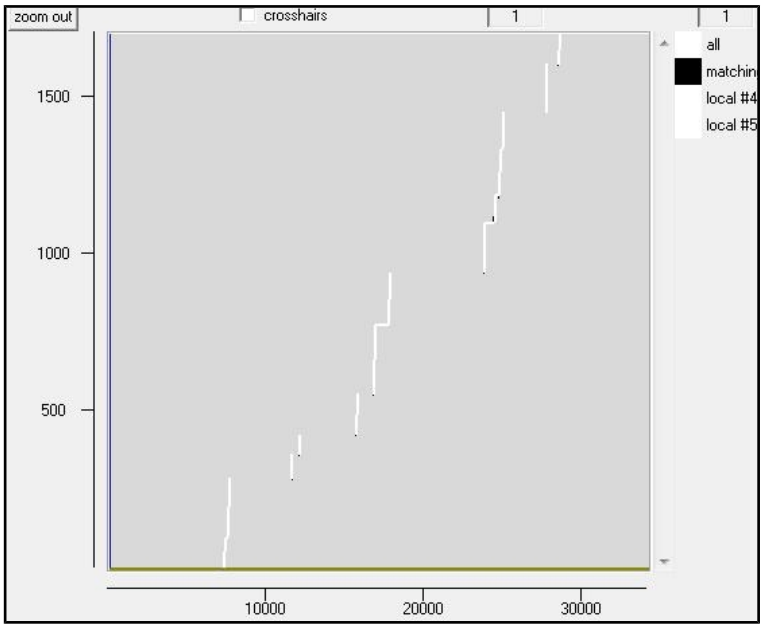
28 I.e. report the best matching region and ignore any others, however deserving they might be? This has always seemed an odd default to me.

From **spin**'s **Comparison** menu, select again **Local alignment**. Clearly, there is more than **1** meaningful local alignment. To see more, **spin** offers two options.



You could guess the **number of alignments** value. However, this is not trivial. You know the number of exons (and so the number of logical alignments expected) with some certainty by now, but this does not necessarily correspond to the expected number of local alignments. **spin** will combine close exons into single alignments if gap penalties are low enough. **spin** assumes that your choice of **number of alignments** is exactly correct. Too low and you will miss real alignments. Too high and **spin** will show alignments of ever increasing fantasy.

It is normally better, therefore, to specify the alignments you wish **spin** to display by providing a quality cut off rather than a volume cut off. To do this, turn on the **alignments above score** option. Use the default value of **20**.



It is not obvious how one might choose a sensible value. In such circumstances, the "lazy" option of just accepting the default is fine. Maybe someone sensible chose it? If not, one can always iterate to a value that works. I.e. if you would like more alignments, lower the value and try again. Too many? raise it. Repeat until the program agrees with what you wished to be correct in the first place. Is not science wonderful? With hope in your heart, click **OK**.

Looking at the graphic, I would say we got lucky this time[29]. There is a local alignment that covers every exon identified by the **dotplot** and no extra results requiring explanation.

Scan your results to find the exons. You should see that all have been correctly aligned.

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

_____

29 If you wish to be meticulously honest. You should **Hide** the graphic of your first **Local alignment** (right click its configuration box, choose **Hide**). Not that it will change anything as you will have found exactly the same alignment the second time round – along with the others.

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 moer 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 pertenent 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

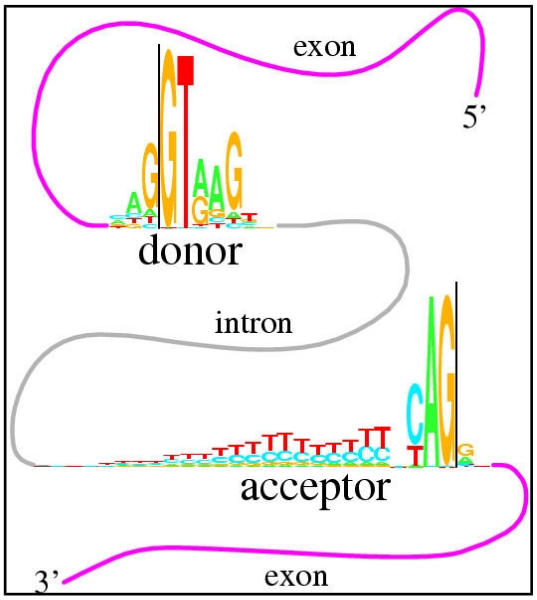
How might the gap around **24,500** in the genomic sequence been positioned more intelligently?

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 forgiveable as **blast** had nor 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  AAGAATACAGGTACCGAGAGACTGTGCAGTTTCACACTTTGTGATTCATACCATTTGTCT  24480
                 ||||||||||||
mRNA       1118  AAGAATACAGGTA----------------------------------------------  1130

                                    • • •

Genomic   24601  CCAAGGTGGGGCTGGGCTCGACGTAGACACAGTGCTAACCTGTCCCACCTGATTTCCAGG  24660

mRNA             ------------------------------------------------------------

Genomic   24661  TATGGTTTTCTAATCGAAGGGCCAAATGGAGAAGAGAAGAAAAACTGAGGAATCAGAGAA  24720
                   ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
mRNA       1131  --TGGTTTTCTAATCGAAGGGCCAAATGGAGAAGAGAAGAAAAACTGAGGAATCAGAGAA  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  AAGAATACAGGTACCGAGAGACTGTGCAGTTTCACACTTTGTGATTCATACCATTTGTCT  24480
                 |||||||||
mRNA       1118  AAGAATACAG----------------------------------------------  1130

                                    • • •

Genomic   24601  CCAAGGTGGGGCTGGGCTCGACGTAGACACAGTGCTAACCTGTCCCACCTGATTTCCAGG  24660
                                                                          |
mRNA             ------------------------------------------------------------G

Genomic   24661  TATGGTTTTCTAATCGAAGGGCCAAATGGAGAAGAGAAGAAAAACTGAGGAATCAGAGAA  24720
                 ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
mRNA       1131  TATGGTTTTCTAATCGAAGGGCCAAATGGAGAAGAGAAGAAAAACTGAGGAATCAGAGAA  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

Roughly how many did you expect?

12 or so … one per exon anyway.

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 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 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 should have resulted in a better alignment.

Had you used stretcher (the faster, less vigorous algorithm offered by the **EMBOSS** package) you would have got a much improved answer in this case (but not 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?

Where you to trawl though your textual output carefully (or simply take my immaculate word for it), you would find **13** perfectly (or nearly so) aligned regions, implying **13** exons.

To be pedantic, the nicely aligned regions do not match the exons exactly (as will become apparent later), so it is not possible to claim definite evidence for any particular number of exons. However, **13** has to be a pretty confident estimate.

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

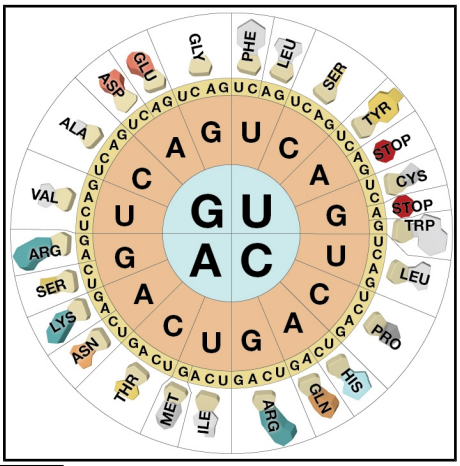
Yes, roughly as suggested by **Ensembl**.

From your investigations comparing mRNA/cDNA with genomic DNA:

What is the amino acid corresponding to this position in the mRNA of the aniridia patient?

This is easy to answer. The top sequence is the mRNA from the **aniridia** patient. **spline** is kind enough to explicitly inform us that the mutated codon, **CCT**, will be expressed a **P**roline.

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



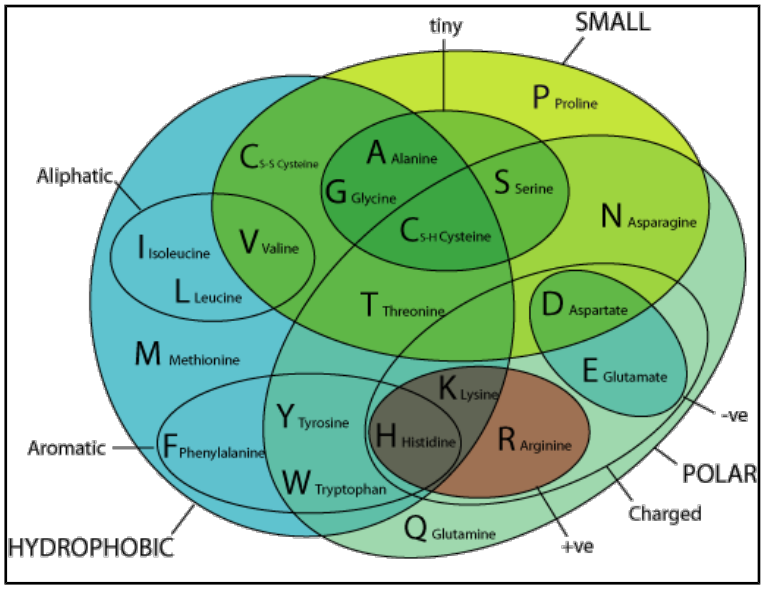
From this rather beautiful representation of the **Genetic Code**, I conclude:

Patient **CCT** → **Proline** (**P**)
Wild Type **GCT** → **Alanine** (**A**)

Feature key	Position(s)	Length	Description	Graphical view	Feature identifier
Natural variant	17 – 17	1	N → S in AN. 1 Publication		VAR_003808
Natural variant	18 – 18	1	G → W in AN. 1 Publication		VAR_003809
Natural variant	19 – 19	1	R → P in AN. 1 Publication		VAR_047860
Natural variant	22 – 26	5	Missing in AN. 2 Publications		VAR_008693
Natural variant	26 – 26	1	R → G in PETAN. 2 Publications		VAR_003810
Natural variant	29 – 29	1	I → S in AN. 1 Publication		VAR_008694
Natural variant	29 – 29	1	I → V in AN. 1 Publication		VAR_003811
Natural variant	33 – 33	1	A → P in AN. 1 Publication		VAR_008695
Natural variant	37 – 39	3	Missing in AN. 1 Publication		VAR_008696

Or, looking back at the relevant **Uniprot Feature Table**, I come to an identical conclusion. This is the Natural Variant at position **33**, as you could easily confirm with a bit of counting along the alignment.

Either way, it could be a quite serious mutation. **Alanine** and **Proline** having quite distinct properties.



Also, according to the font of all easily packaged knowledge, Wikipedia:

> "**Proline** and **glycine** are sometimes known as "**helix breakers**" because they disrupt the regularity of the α helical backbone conformation; however, both have unusual conformational abilities and are commonly found in **turns**."

The **helices** of this protein are of particular importance to its vital **DNA Binding** role.

How do you interpret the **Details** column for exons 1 and 5?

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	pax6-cDNA(+)	pax6-genomic(+)	7245-28540	100.00	99.94	99.94	0.00	0.00

Graphics|Text

#	Query	Subject	Idty	Len	Q.Start	Q.Fin	S.Start	S.Fin	Type	Details
+1	pax6-cDNA	pax6-genomic	0.981	103	1	101	7245	7347	<exon>GT	M53IM5IM43
+1	pax6-cDNA	pax6-genomic	1	188	102	289	7447	7634	AG<exon>GT	M188
+1	pax6-cDNA	pax6-genomic	1	77	290	366	11537	11613	AG<exon>GC	M77
+1	pax6-cDNA	pax6-genomic	1	61	367	427	12000	12060	AG<exon>GT	M61
+1	pax6-cDNA	pax6-genomic	0.992	131	428	558	15628	15758	AG<exon>GT	M86RM44
+1	pax6-cDNA	pax6-genomic	1	216	559	774	16686	16901	AG<exon>GT	M216
+1	pax6-cDNA	pax6-genomic	1	166	775	940	17606	17771	AG<exon>GT	M166
+1	pax6-cDNA	pax6-genomic	1	159	941	1099	23674	23832	AG<exon>GT	M159
+1	pax6-cDNA	pax6-genomic	1	83	1100	1182	24348	24430	AG<exon>GT	M83
+1	pax6-cDNA	pax6-genomic	1	151	1183	1333	24660	24810	AG<exon>GT	M151
+1	pax6-cDNA	pax6-genomic	1	116	1334	1449	24909	25024	AG<exon>GT	M116
+1	pax6-cDNA	pax6-genomic	1	151	1450	1600	27602	27752	AG<exon>GT	M151
+1	pax6-cDNA	pax6-genomic	1	98	1601	1698	28443	28540	AG<exon>AA	M98

For example:

For **Exon 2,** **spline** informs us **M188**, meaning "There are **188** bases aligned and they all **M**atch perfectly".

For **Exon 3**, **spline** informs us **M77**, meaning "There are **77** bases aligned and they all **M**atch perfectly".

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

Exon 1 – M53IM5IM43

An alignment of **103** bases, the first **53** of which **M**atch perfectly (**M53**), there then follows an **I**nsertion (**I**), a further **5 M**atched bases(**M5**), a second **I**nsertion (**I**) all finished off with **43 M**atched bases (**M43**).

Exon 5 – M86RM4R

An alignment of **131** bases, the first **86** of which **M**atch perfectly (**M86**), there them follows a **R**eplacement (**R**) and a further **44 M**atched bases(**M44**).

Full Answer:

From the individual **Exon 1** display, it can be inferred that the declaration of an **I**nsertion or a **D**eletion is made to describe the type of variation required to transform the **cDNA** (**Query**) sequence into the **genomic** (**Subject**). Hence the two **indels** (**I**nsertions or **D**eletions) are considered to be **I**nsertions.

```
   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 **D**eletions 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 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:

```
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

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 **M**s in most alignments. Of course, life would get tricky if the data included numeric characters, but that is not an issue here[30].

I think it worth mentioning, that this way of representing an alignment is a simplification of **CIGAR** format[31]. This format is used for **SAM** (**S**equence **A**lignment **M**ap) and **BAM** (**B**inary **A**lignment **M**ap, exactly the same as **SAM**, except compressed) files. You will be engulfed in **SAM/BAM** files if you ever do any **N**ext **G**eneration **S**equencing (**NGS**).

So, straight form 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 **Op**erators 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**'?

Where is the substitution in the aniridia patient mRNA?
Where is the substitution in the Genomic Sequence?

spline makes one work quite hard to answer this one! Unless I am missing something.

From the alignment of **Exon 5**, the exon including the **R**eplacement, with a bit of squinting, it can be confirmed that the **R**eplacement is at:

```
        R  Q  K  I  V  E  L  P  H  S  G  A  R  P  C  D  I  S  R  I  L  Q
  493 CGGCAGAAGATTGTAGAGCTACCTCACAGCGGGGCCCGGCCGTGCGACATTTCCCGAATTCTGCAG....
      ||||||||||||||||||||| ||||||||||||||||||||||||||||||||||||||||||||||
15693 CGGCAGAAGATTGTAGAGCTAGCTCACAGCGGGGCCCGGCCGTGCGACATTTCCCGAATTCTGCAGGTGA
```

Base pair position **514** of the aniridia patient's **mRNA**

Base pair position **15714** of the **genomic** sequence

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

```
                                                          M  Q  N
  367 .....AGCCCCATATTCGAGCCCCGTGGAATCCCGCGGCCCCCAGCCAGAGCCAGCATGCAGAACA....
            ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
11995 AACAGAGCCCCATATTCGAGCCCCGTGGAATCCCGCGGCCCCCAGCCAGAGCCAGCATGCAGAACAGTAA
```

More squinting, and I conclude the **A** of the **ATG** representing the initial **Methionine** of the protein coding region is at position **418**. That is, the **3' UTR** ends at position **417**. So the **R**eplacement is at:

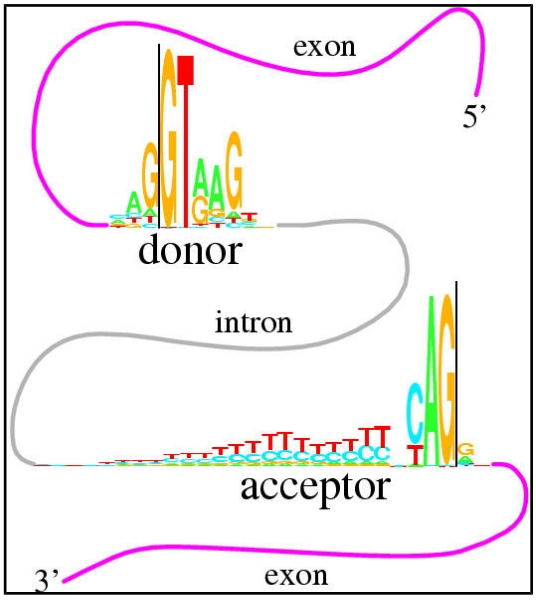
Base position **514** – **417** = **97** of the protein coding region of the **mRNA**.

As **97** / **3** is **32** remainder **1**, the **R**eplacement is at codon position **1** of the **33rd** amino acid of the protein.

Which you knew already, of course! Cannot help thinking that **spline** might have helped a bit more here?

30 The **Wikipedia** article shows how this complication might be overcome.
31 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.

Compare the predicted **splign** intron/exon boundaries with the conservation suggested by the logo?
What deviation(s) from the model suggested by the logo can you see?



You may have gathered, I rather like this logo, although I rather think it is leading me to make the same point a trifle to often?

The logo is in almost **100%** agreement with the predictions of **spline**.

As you will have noted previously, when looking at the **Ensembl** predictions of exons locations of a similar transcript of the **PAX6** human gene, there is a single exception.

Type
<exon>GT
AG<exon>GT
AG<exon>GC
AG<exon>GT
AG<exon>GT
AG<exon>GT
AG<exon>GT
AG<exon>GT
AG<exon>GT
AG<exon>GT
AG<exon>GT
AG<exon>GT
AG<exon>

The easiest way to show this in the **spline** output is to look at the **spline** text output again.

The **Type** column records the type of all the **<exon>** alignments it predicts. It also records **2 flanking intron base pairs**.

It is clear that the only time the spline prediction deviates from the model suggested by the logo is at the end of the **3rd** exon. Here there is **GC** rather then **GT**. Well, nothing is perfect!

From your investigations of **Local Alignment**:

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

To **spin**, 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 obliging.