

# The Staden Package Mini-Manual

---

Last update on 25 November 2011

James Bonfield, Kathryn Beal, Mark Jordan,  
Yaping Cheng and Rodger Staden

---

Copyright © 1999-2002, Medical Research Council, Laboratory of Molecular Biology. Made available under the standard BSD licence.

---

Copyright © 2002-2006, Genome Research Limited (GRL). Made available under the standard BSD licence.

---

Portions of this code are derived from a modified Primer3 library. This bears the following copyright notice:

Copyright © 1996,1997,1998 Whitehead Institute for Biomedical Research. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. Redistributions of source code must also reproduce this information in the source code itself.
2. If the program is modified, redistributions must include a notice (in the same places as above) indicating that the redistributed program is not identical to the version distributed by Whitehead Institute.
3. All advertising materials mentioning features or use of this software must display the following acknowledgment: This product includes software developed by the Whitehead Institute for Biomedical Research.
4. The name of the Whitehead Institute may not be used to endorse or promote products derived from this software without specific prior written permission.

We also request that use of this software be cited in publications as

Steve Rozen, Helen J. Skaletsky (1996,1997,1998) Primer3. Code available at [http://www-genome.wi.mit.edu/genome\\_software/other/primer3.html](http://www-genome.wi.mit.edu/genome_software/other/primer3.html)

THIS SOFTWARE IS PROVIDED BY THE WHITEHEAD INSTITUTE “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE WHITEHEAD INSTITUTE BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

---

Permission is given to duplicate this manual in both paper and electronic forms.

## Short Contents

1	Preface .....	1
2	Sequence Assembly and Finishing Using Gap4 .....	3
3	Searching for point mutations using pregap4 and gap4 .....	21
4	Preparing Readings for Assembly Using Pregap4 .....	37
5	Viewing Traces Using Trev .....	49
6	Analysing Sequences Using Spin .....	53



# Table of Contents

<b>1</b>	<b>Preface</b>	<b>1</b>
<b>2</b>	<b>Sequence Assembly and Finishing Using Gap4</b>	<b>3</b>
2.1	Introduction	3
2.1.1	Summary of the Files used and the Preprocessing Steps	4
2.1.2	Summary of Gap4's Functions	6
2.1.3	Introduction to the gap4 User Interface	7
2.1.3.1	Introduction to the Contig Selector	8
2.1.3.2	Introduction to the Contig Comparator	9
2.1.3.3	Introduction to the Template Display	11
2.1.3.4	Introduction to the Consistency Display	13
2.1.3.5	Introduction to the Restriction Enzyme Map	15
2.1.3.6	Introduction to the Stop Codon Map	16
2.1.3.7	Introduction to the Contig Editor	17
2.1.3.8	Introduction to the Contig Joining Editor	20
<b>3</b>	<b>Searching for point mutations using pregap4 and gap4</b>	<b>21</b>
3.1	Introduction to mutation detection	21
3.1.1	Mutation Detection Programs	26
3.1.2	Mutation Detection Reference Data	26
3.1.3	Reference Sequences	26
3.1.4	Reference Traces	27
3.1.5	Using The Template Display With Mutation Data	28
3.1.6	Configuring The Gap4 Editor For Mutation Data	29
3.1.7	Using The Gap4 Editor With Mutation Data	30
3.1.8	Processing Batches Of Mutation Data Trace Files	31
3.1.9	Processing Batches Of Mutation Data Trace Files Using Pregap4	33
3.1.10	Configuration Of Pregap4 For Mutation Data	34
3.1.11	Discussion Of Mutation Data Processing Methods	35
<b>4</b>	<b>Preparing Readings for Assembly Using Pregap4</b>	<b>37</b>
4.1	Introduction	37
4.1.1	Summary of the Files used and the Processing Steps	37
4.1.2	Introduction to the Pregap4 User Interface	42
4.1.2.1	Introduction to the Files to Process Window	44
4.1.2.2	Introduction to the Configure Modules Window	46
4.1.2.3	Introduction to the Textual Output Window	47
4.1.2.4	Introduction to Running Pregap4	48

<b>5</b>	<b>Viewing Traces Using Trev .....</b>	<b>49</b>
<b>6</b>	<b>Analysing Sequences Using Spin .....</b>	<b>53</b>
6.1	Introduction .....	53
6.1.1	Summary of the Spin Single Sequence Functions .....	53
6.1.2	Summary of the Spin Comparison Functions .....	53
6.1.3	Introduction to the Spin User Interface .....	54
6.1.3.1	Introduction to the Spin Plot .....	55
6.1.3.2	Introduction to the Spin Sequence Display .....	60
6.1.3.3	Introduction to the Spin Sequence Comparison Plot..	62
6.1.3.4	Introduction to the Spin Sequence Comparison Display .....	66

# 1 Preface

This manual describes the sequence handling and analysis software developed at the Medical Research Council Laboratory of Molecular Biology, Cambridge, UK, which has come to be known as the Staden Package.

The vast bulk of work on the package was done at LMB within Rodger Staden's group, which over time has consisted of Tim Gleeson, Simon Dear, James Bonfield, Kathryn Beal, Mark Jordan and Yaping Cheng. Besides the group members a number of people have made important contributions; most notably including David Judge and John Taylor for feedback / tutorials and developing the Windows release respectively.

Since mid-2003 the group in LMB no longer exists. The package became "open source" and moved onto SourceForge in early 2004. The only active maintainer (James Bonfield) now works at the Wellcome Trust Sanger Institute. The new package homepage may be found at

<http://staden.sourceforge.net/> and the SourceForge project page is at <https://sourceforge.net/projects/staden/>.

The focus of the development since 1990 has been to produce improved methods for processing the data for large scale sequencing projects, and this is reflected in the scope of the package: the most advanced components (trev, prefinish, pregap4 and gap4) are those used in that area. Nevertheless the package also contains a program (spin) for the analysis and comparison of finished sequences. The latter also provides a graphical user interface to EMBOSS.

Since the LMB group disbanded it has become necessary to reduce the scope of further development, so active work is primarily being directed to the Gap4 program.

Gap4 performs sequence assembly, contig ordering based on read pair data, contig joining based on sequence comparisons, assembly checking, repeat searching, experiment suggestion, read pair analysis and contig editing. It has graphical views of contigs, templates, readings and traces which all scroll in register. Contig editor searches and experiment suggestion routines use confidence values to calculate the confidence of the consensus sequence and hence identify only places requiring visual trace inspection or extra data. The result is extremely rapid finishing and a consensus of known accuracy.

Pregap4 provides a graphical user interface to set up the processing required to prepare trace data for assembly or analysis. It also automates these processes. The possible processes which can be set up and automated include trace format conversion, quality analysis, vector clipping, contaminant screening, repeat searching and mutation detection.

Trev is a rapid and flexible viewer and editor for ABI, ALF, SCF and ZTR trace files.

Prefinish analyses partially completed sequence assemblies and suggests the most efficient set of experiments to help finish the project.

Tracediff and hetscan automatically locate mutations by comparing trace data against reference traces. They annotate the mutations found ready for viewing in gap4.

Spin analyses nucleotide sequences to find genes, restriction sites, motifs, etc. It can perform translations, find open reading frames, count codons, etc. Many results are presented graphically and a sliding sequence window is linked to the graphics cursor. Spin also compares pairs of sequences in many ways. It has very rapid dot matrix analysis, global and local alignment algorithms, plus a sliding sequence window linked to the graphical plots. It can compare nucleic acid against nucleic acid, protein against protein, and protein against nucleic acid.

The manual describes, in turn, each of the main programs in the package: gap4, and then pregap4 and its associated programs such as trev, and then spin. This is followed by a description of the graphical user interface, the ZTR, SCF and Experiment file formats used by our software, UNIX manpages for several of the smaller programs, and finally a list of papers published about the software. The description for each of the programs includes an introductory section which is intended to be sufficient to enable people to start using them, although in order to get the most from the programs, and to find the most efficient ways of using them we recommend that the whole manual is read once. The mini-manual is made up from the introductory sections for each of the main programs.



## 2 Sequence Assembly and Finishing Using Gap4

### 2.1 Introduction

Gap4 is a Genome Assembly Program. The program contains all the tools that would be expected from an assembly program plus many unique features and a very easily used interface. The original version was described in *Bonfield, J.K., Smith, K.F. and Staden, R. A new DNA sequence assembly program. Nucleic Acids Res. 24, 4992-4999 (1995)*

Gap4 is very big and powerful. Everybody employs a subset of options and has their favourite way of accessing and using them. Although there is a lot of it, users are encouraged to go through the whole of the documentation once, just to discover what is possible, and the way that best suits their own work. At the very least, the whole of this introductory chapter should be read, as in the long run, it will save time.

This chapter serves as a cross reference point, to give an overview of the program and to introduce some of the important ideas which it uses. The main topics that are introduced are listed in the current section. We introduced the use of base call accuracy values for speeding up sequencing projects. The ability to annotate segments of readings and the consensus can be very convenient. Generally the 3' ends of readings from sequencing instruments are of too low a quality to be used to create reliable consensus, but they can be useful, for example, for finding joins between contigs.

One of the most powerful features of gap4 is its graphical user interface which enables the data to be viewed and manipulated at several levels of resolution. The displays which provide these different views are introduced, with several screenshots.

It is important to understand the different files used by our sequence assembly software, and how the data is processed before it reaches gap4.

Note that gap4 is a very flexible program, and is designed so that it can easily be configured to suit different purposes and ways of working. For example it is easy to create a beginners version of gap4 which has only a subset of functions. What is described in this manual is the full version, and so is likely to contain some perhaps more esoteric options that few people will need to use. This introductory section also contains a complete list of the options in the gap4 main menus.

In addition to sequence assembly, gap4 can be used for managing mutation study data and for helping to discover and check for mutations.

Two further useful facilities of gap4 are "Lists" and "Notes". For many operations it is convenient to be able to process sets of data together - for example to calculate a consensus sequence for a subset of the contigs. To facilitate this gap4 uses lists A 'Note' is an arbitrary piece of text which can be attached to any reading, any contig, or to the database in general.

### 2.1.1 Summary of the Files used and the Preprocessing Steps

Gap4 stores the data for an assembly project in a gap4 database. Before being entered into the gap4 database the data must be passed through several preassembly steps, usually via pregap4. These steps are outlined below.

The programs can handle data produced by a variety of sequencing instruments. They can also handle data entered using digitisers or that has been typed in by hand. Usually the trace files in proprietary format, such as those of ABI, are converted to SCF files or ZTR files. As originally put forward in *Bonfield, J.K. and Staden, R. The application of numerical estimates of base calling accuracy to DNA sequencing projects. Nucleic Acids Research 23, 1406-1410 (1995)*. gap4 makes important use of basecall confidence values, which are normally stored in the reading's SCF file.

One of the first steps in the preprocessing is to copy the base calls from the trace files to text files known as Experiment files. All the subsequent processes operate on the Experiment files. Other preassembly steps include quality and vector clipping. Each step is performed by a specific program controlled by the program pregap4.

Experiment file format is similar to that of EMBL sequence entries in that each record starts with a two letter identifier, but we have invented new records specific to sequencing experiments. One of pregap4's tasks is to augment the Experiment files to include data about the vectors, primers and templates used in the production of each reading, and if necessary it can extract this information from external databases. Some of the information is needed by pregap4 and some by gap4. (Note that in order to get the most from gap4 it is essential to make sure that it is supplied, via the Experiment files, with all the information it needs.)

The trace files are not altered, but are kept as archival data so that it is always possible to check the original base calls and traces. Any changes to the data prior to assembly (and we recommend that none are made until readings can be viewed aligned with others) are made to the copy of the sequence in the Experiment file.

The reading data, in Experiment file format, is entered into the project database, usually via one of the assembly engines. Because Experiment file format was based on EMBL file format, EMBL files can also be entered and their feature tables will be converted to tags. There is no limit to the length of readings which can be entered.

All the changes to the data made by gap4 are made to the copies of the data in the project database. Once the data has been copied into the gap4 database the Experiment files are no longer required.

Gap4 uses the trace files to display the traces, and to compare the edited bases with the original base calls. However gap4 databases do not store trace files: they record only the names of the trace files (which are copied from the readings' Experiment files). This means that if the trace files for a project are not in the same directory/folder as the gap4 database, gap4 needs to be told where they are, otherwise it cannot use them. Ideally, all the trace files for a project should be stored in one directory. To tell gap4 where they are the "Trace file location" command in the Options menu should be used.

Gap4 databases have a number of size constraints, some of which can be altered by users and others which are fixed.

While gap4 is running it often needs to calculate a consensus. The maximum size of this sequence is controlled by a variable "maxseq". Most routines are able to automatically increase the value of maxseq while they are running, but some of the older functions, including some of the original assembly engines, are not. This means that it is important for users to set maxseq to a sufficiently high value before running these elderly routines. By default maxseq is currently set to 100000, but users can set it on the command line or from within the Options menu.

Gap4 databases contain one record for each reading and one for each contig. The sum of these two sets of records is the "database\_size", and the maximum value that database\_size is permitted to reach is "maxdb". When databases are initialised maxdb is set, by default, to 8000. Users can alter this value on the command line or from within the Options menu of gap4.

Gap4 databases also limit the number and names of readings so that various output routines know how many character positions are required: the maximum number imposed in this way is 99,999,999, and the maximum reading name length is 40.

Currently we have sites with single gap4 databases containing over 200,000 readings with consensus sequences in excess of 7,000,000 bases.

A gap4 database can be used by several users simultaneously, but only one is allowed to change the contents of the database, and the others are given "readonly" access. As part of its mechanism to prevent more than one person editing a database at once gap4 uses a "BUSY" file to signify that the database is opened for writing. Before opening a database for writing, gap4 checks to see if the BUSY file for that database exists. If it does, the database is opened only for reading, if not it creates the file, so that any additional attempts to open the database for writing will be blocked. When the user with write access closes the database, the BUSY file is deleted, hence re-enabling its ability to be opened for changes. It is worth remembering that a side effect of this mechanism, is that in the event of a program or system crash the BUSY file will be left on the disk, even though the database is not being used. In this case users must remove the BUSY file before using the database.

The final result from a sequencing project is a consensus sequence and gap4 can write these in Experiment file format, fasta format or staden format. Of course the whole database and all the trace files are also useful for future reference as they allow any queries about the accuracy of the sequence to be answered.

### 2.1.2 Summary of Gap4's Functions

The tasks which gap4 can perform can be roughly divided into assembly, finishing, and editing. But gap4 contains many other functions which can help to complete a sequencing project with the minimum amount of effort, and some of these are listed below.

Readings are entered into the gap4 database using the assembly algorithms. In general these algorithms will build the largest contigs they can by finding overlaps between the readings, however some, perhaps more doubtful, joins between contigs may be missed, and these can be discovered, checked and made using Find Internal Joins, Find repeats and Join Contigs. Find Internal Joins compares the ends of contigs to see if there are possible overlaps and then presents the overlap in the Contig Joining Editor, from where the user can view the traces, make edits and join the contigs. Find Repeats can be used in a similar way, but unlike Find Internal Joins it does not require the matches it finds to continue to the ends of contigs.

Read-pair data can be used to automatically put contigs into the correct order, and information about contigs which share templates can be plotted out. The relationships of readings and templates, within and between contigs can also be shown by the Template Display which has a wide selection of display modes and uses.

Problems with the assembly can be revealed by use of Check Assembly, Find repeats, and Restriction Enzyme mapping. Check Assembly compares every reading with the segment of the consensus it overlaps to see how well it aligns. Those that align poorly are plotted out in the Contig Comparator. Find Repeats also presents its results in the Contig Comparator, so if used in conjunction with Check Assembly, it can show cases where readings have been assembled into the wrong copy of a repeated element. At the end of a project the Restriction Enzyme map function can be used to compare the consensus sequence with a restriction digest of the target sequence. Problems can also be found by use of the various Coverage Plots available in the Consistency Display. These plots will show regions of low or high reading coverage, places with data for only one strand, or where there is no read-pair coverage. Errors can be corrected by Disassemble Readings and Break Contig which can remove readings from contigs or databases or can break contigs.

The general level of completeness of the consensus sequence can be seen diagrammatically using the Quality Plot, and the confidence values for each base in the consensus sequence can be plotted.

The most powerful component of gap4 is its Contig Editor. which has many display modes and search facilities to enable very rapid discovery and fixing of base call errors.

If working on a protein coding sequence, the consensus can be analysed using the Stop Codon Map, and its translation viewed using the Contig Editor.

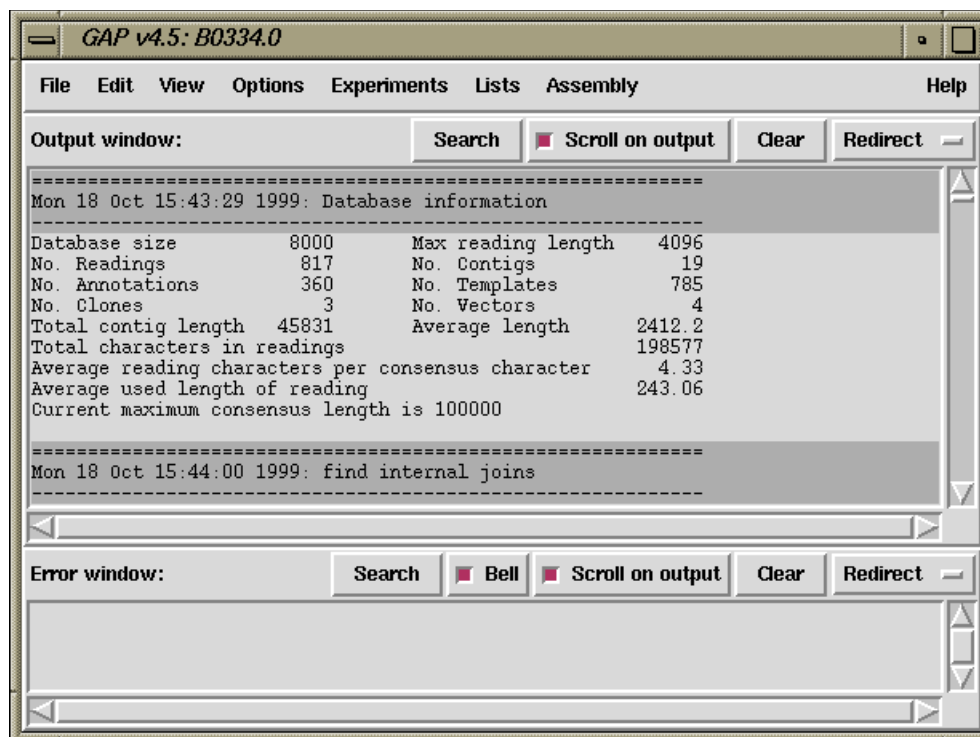
The final result from a sequencing project is a consensus sequence.

### 2.1.3 Introduction to the gap4 User Interface

Gap4 has a main window from which all the main options are selected from menus. When a database is open it also has a Contig Selector which will transform into a Contig Comparator whenever needed. In addition many of the gap4 functions, such as the Contig Editor or the Template Display will create their own windows when they are activated. All the graphical displays and the Contig Editor can be scrolled in register. The base of the graphical display windows usually contains an Information Line for showing short textual data about results or items touched by the mouse cursor. Gap4 is best operated using a three button mouse, but alternative keybindings are available. Full details of the user interface are described elsewhere, and here we give an introduction based around a series of screenshots.

The main window (shown below) contains an Output window for textual results, an Error window for error messages, and a series of menus arranged along the top. The contents of the two text windows can be searched, edited and saved. Each set of results is preceded by a header containing the time and date when it was generated.

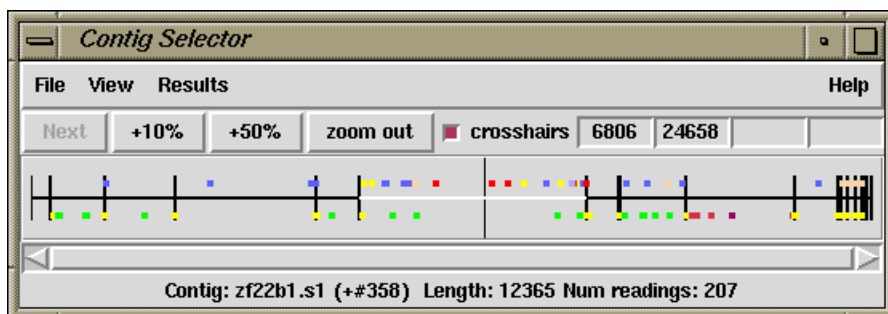
Some of the text will be underlined and shaded differently. These are hyperlinks which perform an operation when clicked (with the left mouse button) on, typically invoking a graphical display such as the contig editor. Clicking on these with the right mouse button will bring up a menu of additional operations. At present only a few commands (Show Relationships and the Search functions) produce hypertext, but if there is sufficient interest this may be expanded on.



### 2.1.3.1 Introduction to the Contig Selector

The gap4 Contig Selector is used to display, select and reorder contigs. In the Contig Selector all contigs are shown as colinear horizontal lines separated by short vertical lines. The length of the horizontal lines is proportional to the length of the contigs and their left to right order represents the current ordering of the contigs. Users can change the contig order by dragging the lines representing the contigs. This is done by clicking and holding the middle mouse button, or Alt left mouse button, on a line and then moving the mouse cursor. The Contig Selector can also be used to select contigs for processing. For example, clicking with the right mouse button on the line representing a contig will invoke a menu containing the commands which can be performed on that contig. There are several alternative ways of specifying which contig an operation should be performed on. Contigs are identified by the name or number of any reading they contain. When a dialogue is requesting a contig name, using the left mouse button to click on the contig in the Contig Selector will transfer its name to the dialogue box. Other methods are available.

As the mouse is moved over a contig, it is highlighted and the contig name (left most reading name) and length are displayed in the Information Line. The number in brackets is the contig number (actually the number of its leftmost reading). Tags or annotations can also be displayed in the Contig Selector window.



The figure shows a typical display from the Contig Selector. At the top are the File, View and Results menus. Below that are buttons for zooming and for displaying the crosshair. The four boxes to the right are used to display the X and Y coordinates of the crosshair. The rightmost two display the Y coordinates when the contig selector is transformed into the Contig Comparator. The two leftmost boxes display the X coordinates: the leftmost is the position in the contig and the other is the position in the overall consensus. The crosshair is the vertical line spanning the panel below. Tags are shown as coloured rectangles above and below the lines.

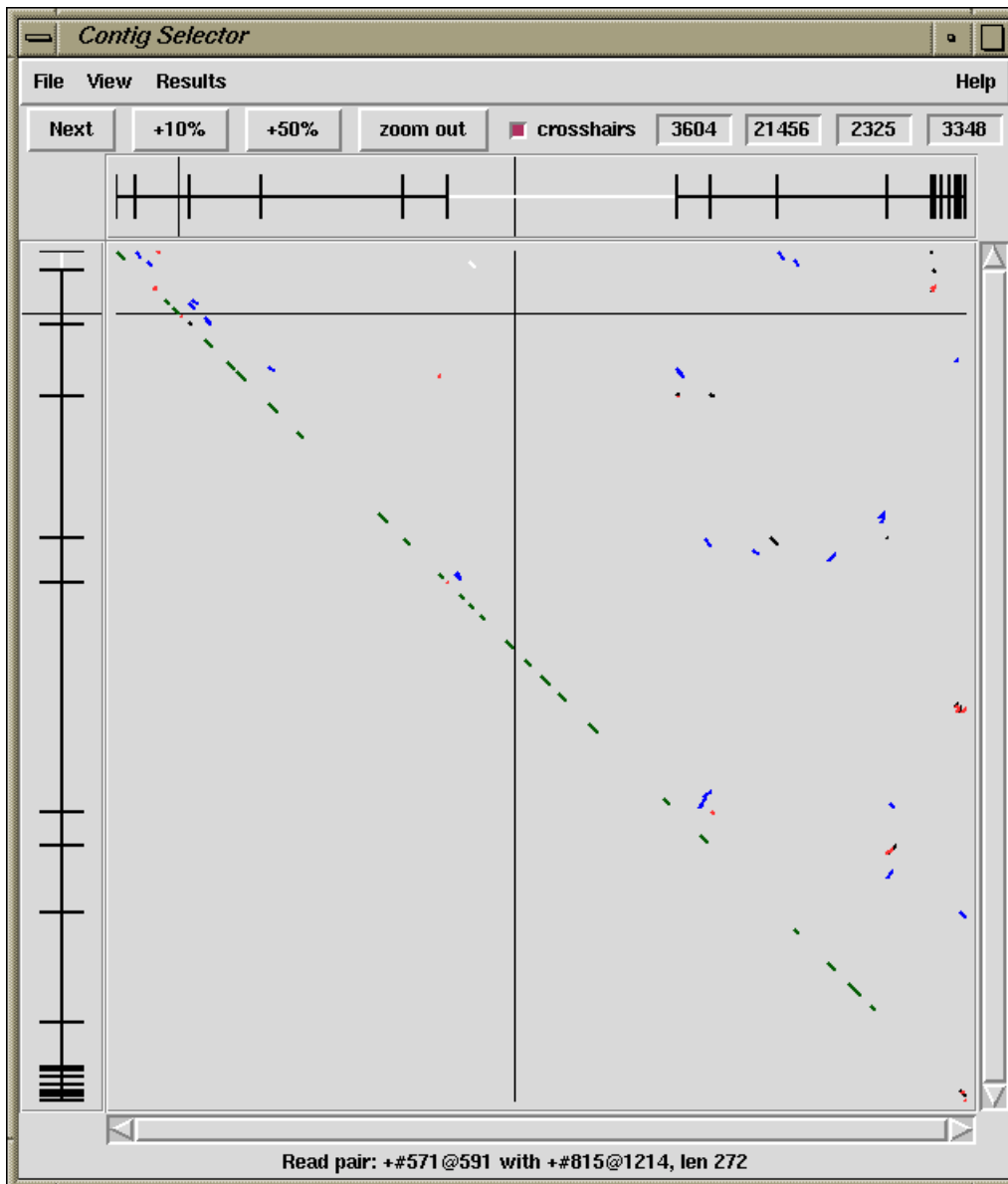
### 2.1.3.2 Introduction to the Contig Comparator

Gap4 commands such as Find Internal Joins, Find Repeats, Check Assembly, and Find Read Pairs automatically transform the Contig Selector to produce the Contig Comparator. To produce this transformation a copy of the Contig Selector is added at right angles to the original window to create a two dimensional rectangular surface on which to display the results of comparing or checking contigs.

Each of the functions plots its results as diagonal lines of different colours. In general, if the plotted points are close to the main diagonal they represent results from pairs of contigs that are in the correct relative order. Lines parallel to the main diagonal represent contigs that are in the correct relative orientation to one another. Those perpendicular to the main diagonal show results for which one contig would need to be reversed before the pair could be joined. The manual contig dragging procedure can be used to change the relative positions of contigs. As the contigs are dragged the plotted results will automatically be moved to their corresponding new positions. This means that, in general, if users drag the contigs to move their plotted results close to the main diagonal they will simultaneously be putting their contigs into the correct relative positions.

This plot can simultaneously show the results of independent types of search, making it easy for users to see if different analyses produce corroborating evidence for the ordering of contigs. Indications that a reading may have been assembled in an incorrect position can also be seen - if for example a result from Check Assembly lies on the same horizontal or vertical projection as a result from Find Repeats, users can see the alternative position to place the doubtful reading.

The plotted results can be used to invoke a subset of commands by the use of pop-up menus. For example if the user clicks the right mouse button over a result from Find Internal Joins a menu containing Invoke Join Editor and Invoke Contig Editors will pop up. If the user selects Invoke Join Editor the Join Editor will be started with the two contigs aligned at the match position contained in the result. If required one of the contigs will be complemented to allow their alignment.



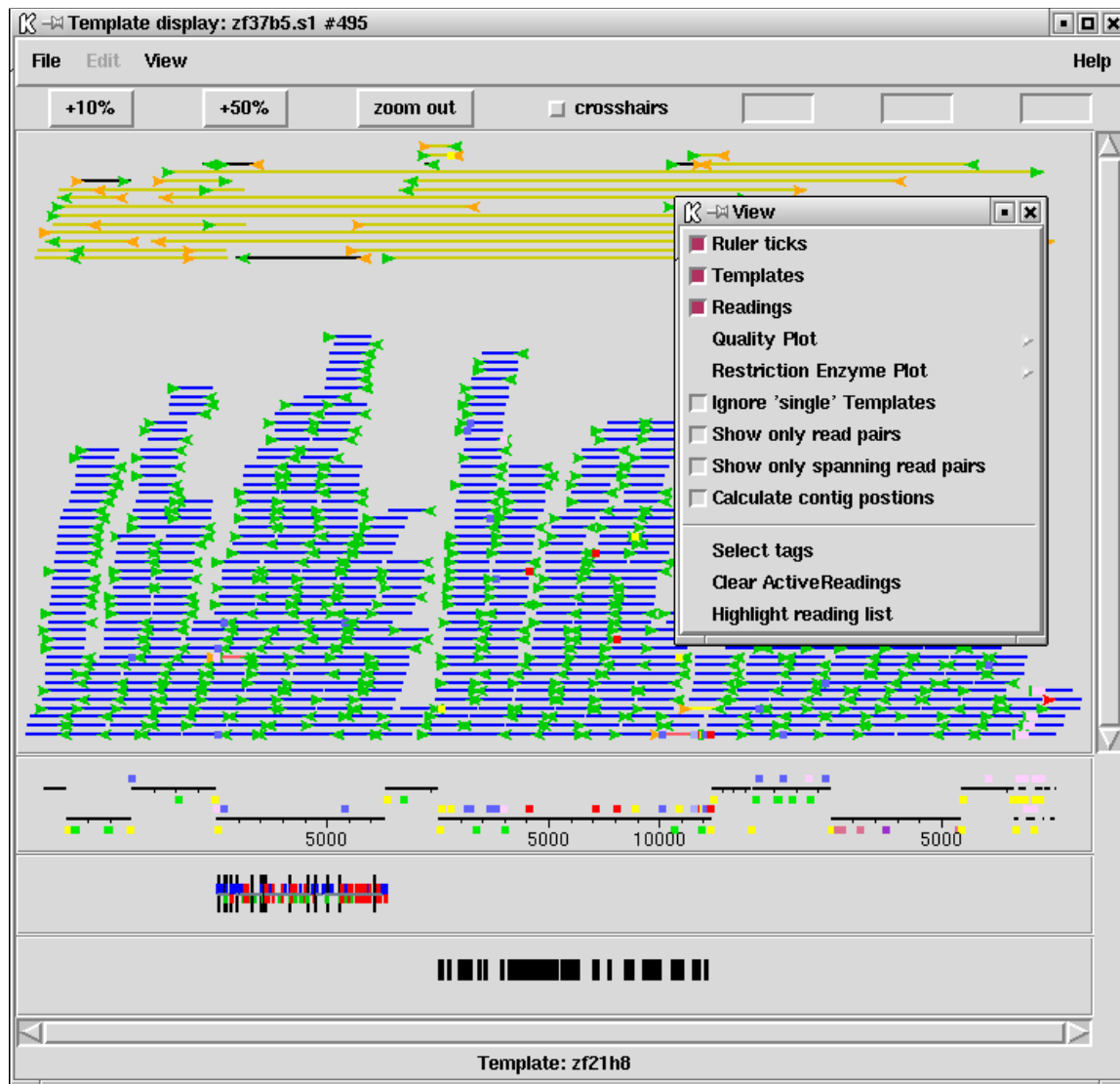
A typical display from the Contig Comparator is shown above. It includes results for Find Internal Joins in black, Find Repeats in red, Check Assembly in green, and Find Read Pairs in blue. Notice that there are several internal joins, read pairs and repeats close to the main diagonal near the top left of the display. This indicates that the contigs represented in that area are likely to be in the correct positions relative to one another. In the middle of the bottom right quadrant there is a blue diagonal line perpendicular to the main diagonal. This indicates a pair of contigs that are in the wrong relative orientation. The crosshairs show the positions for a pair of contigs. The vertical line continues into the Contig Selector part of the display, and the position represented by the horizontal line is also duplicated there.



### 2.1.3.3 Introduction to the Template Display

The Template Display can show schematic plots of readings, templates, tags, restriction enzyme sites and the consensus quality. Colour coding distinguishes reading, primer and template types. The Template Display can also be used to reorder contigs and to invoke the Contig Editor.

An example showing all these information types can be seen in the Figure below.



The large top section contains lines and arrows representing readings and templates. Beneath this are rulers; one for each contig, and below those is the quality plot. The template and reading section of the display is in two parts. The top part contains the templates which have been sequenced from both ends but which are in some way inconsistent - for example given the current relative positions of their readings, they may have a length that is larger or greater than that expected, or the two readings may, as it were, face

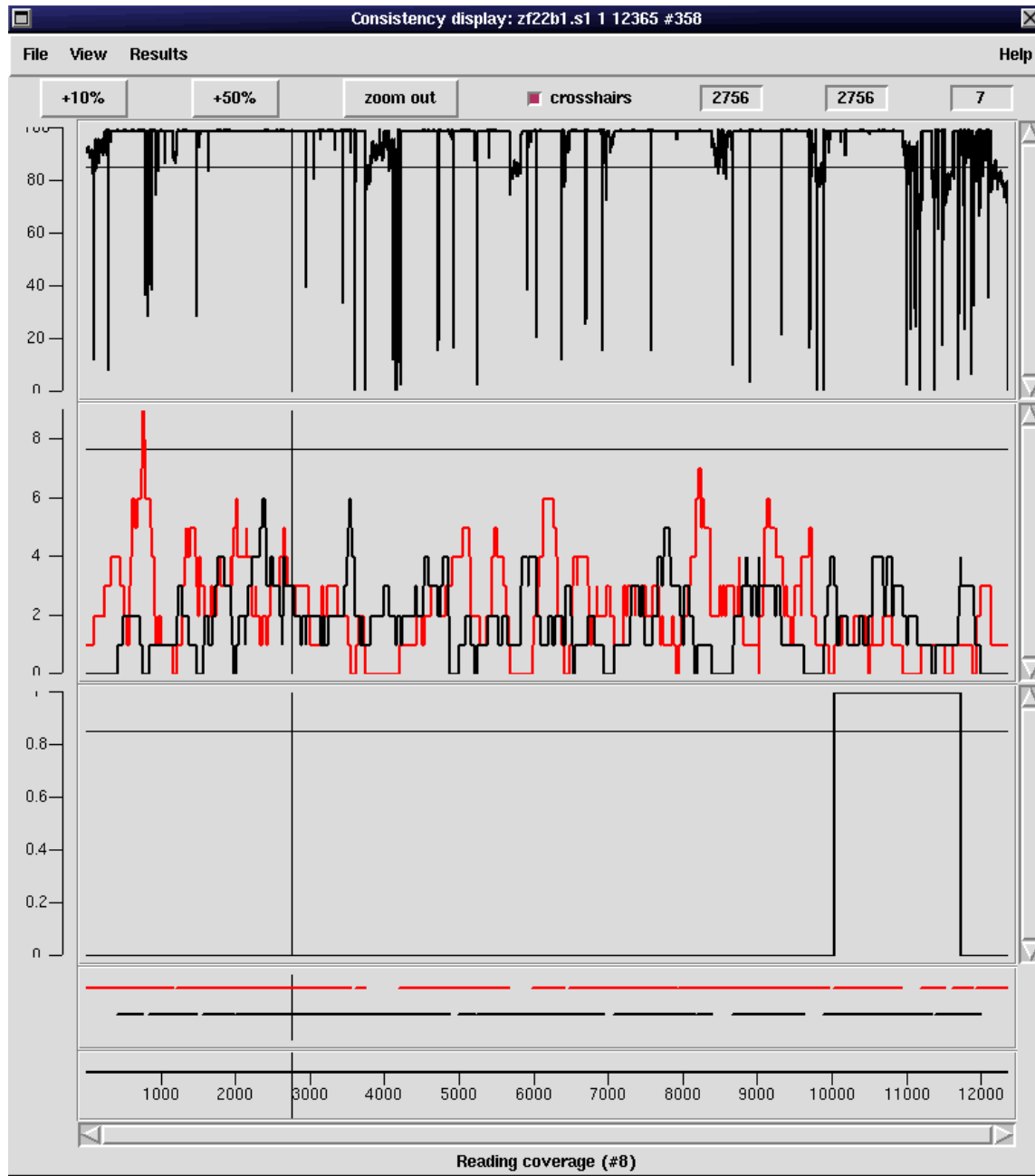
away from one another. Colour coding is used to distinguish between different types of inconsistency, and whether or not the inconsistency involves readings within or between contigs. For example, most of the problems shown in the screendump above are coloured dark yellow, indicating an inconsistency between a pair of contigs. The rest of the data, (mostly dark blue indicating templates sequenced from only one end), is plotted below the data for the inconsistent templates. Forward readings are light blue and reverse readings are orange. Templates in bright yellow have been sequenced from both ends, are consistent and span a pair of contigs (and so indicating the relative orientation and separation of the contigs).

At the bottom is the restriction enzyme plot. The coloured blocks immediately above and below the ruler are tags. Those above the ruler can also be seen on their corresponding readings in the large top section. The display can be zoomed. The position of a crosshair is shown in the two left most boxes in the top right hand corner. The leftmost shows the distance in bases between the crosshair and the start of the contig underneath the crosshair. The middle box shows the distance between the crosshair and the start of the first contig. The right box shows the distance between two selected cut sites in the restriction enzyme plots.

### 2.1.3.4 Introduction to the Consistency Display

The Consistency Display provides plots designed to highlight potential problems in contigs. It is invoked from the main gap4 View menu by selecting any of its plots. Once a plot has been displayed, any of the other types of consistency plot can be displayed within the same frame from the View menu of the Consistency Display.

An example showing the Confidence Values Graph and the corresponding Reading Coverage Histogram, Read-Pair Coverage Histogram and Strand Coverage is shown below.



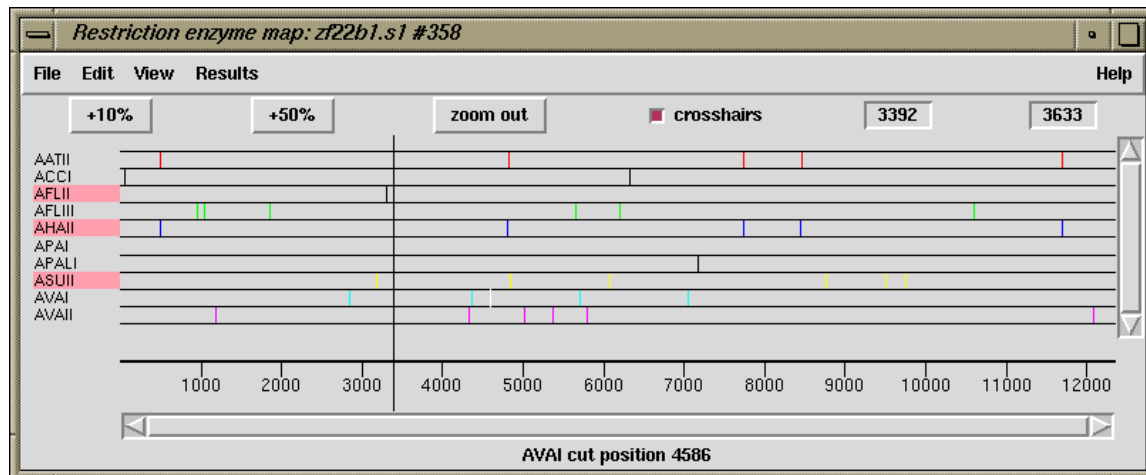
If more than one contig is displayed, the contigs are drawn immediately after one another but are staggered in the y direction.

The ruler ticks can be turned on or off from the View menu of the consistency display. The plots can be enlarged or reduced using the standard zooming mechanism.

The crosshair toggle button controls whether the crosshair is visible. This is shown as a black vertical and horizontal line. The position of the crosshair is shown in the 3 boxes to the right of the crosshair toggle. The first box indicates the cursor position in the current contig. The second box indicates the overall position of the cursor in the consensus. The last box shows the y position of the crosshair..

### 2.1.3.5 Introduction to the Restriction Enzyme Map

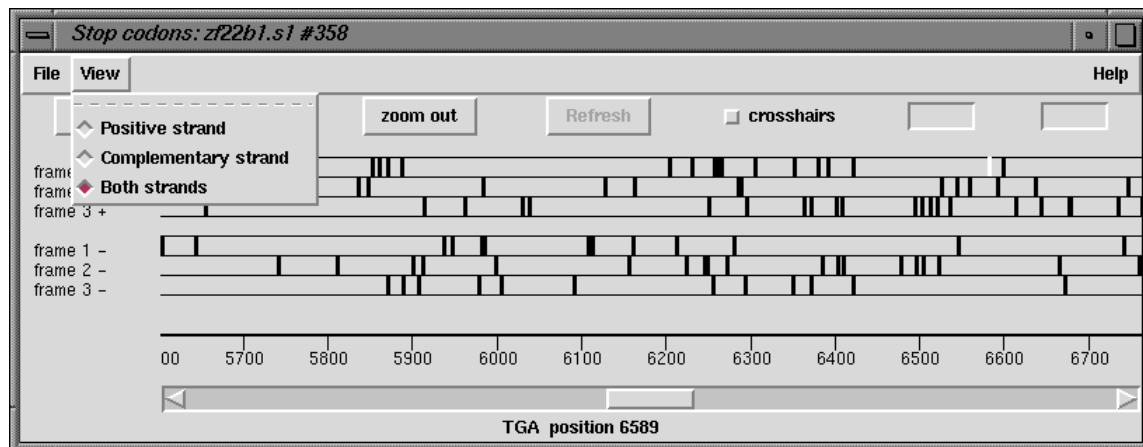
The restriction enzyme map function finds and displays restriction sites within a specified region of a contig. Users can select the enzyme types to search for and can save the sites found as tags within the database.



This figure shows a typical view of the Restriction Enzyme Map in which the results for each enzyme type have been configured by the user to be drawn in different colours. On the left of the display the enzyme names are shown adjacent to their rows of plotted results. If no result is found for any particular enzyme eg here APAI, the row will still be shown so that zero cutters can be identified. Three of the enzymes types have been selected and are shown highlighted. The results can be scrolled vertically (and horizontally if the plot is zoomed in). A ruler is shown along the base and the current cursor position (the vertical black line) is shown in the left hand box near the top right of the display. If the user clicks, in turn, on two restriction sites their separation in base pairs will appear in the top right hand box. Information about the last site touched is shown in the Information line at the bottom of the display. At the top the edit menu is shown and can be used to create tags for highlighted enzyme types.

### 2.1.3.6 Introduction to the Stop Codon Map

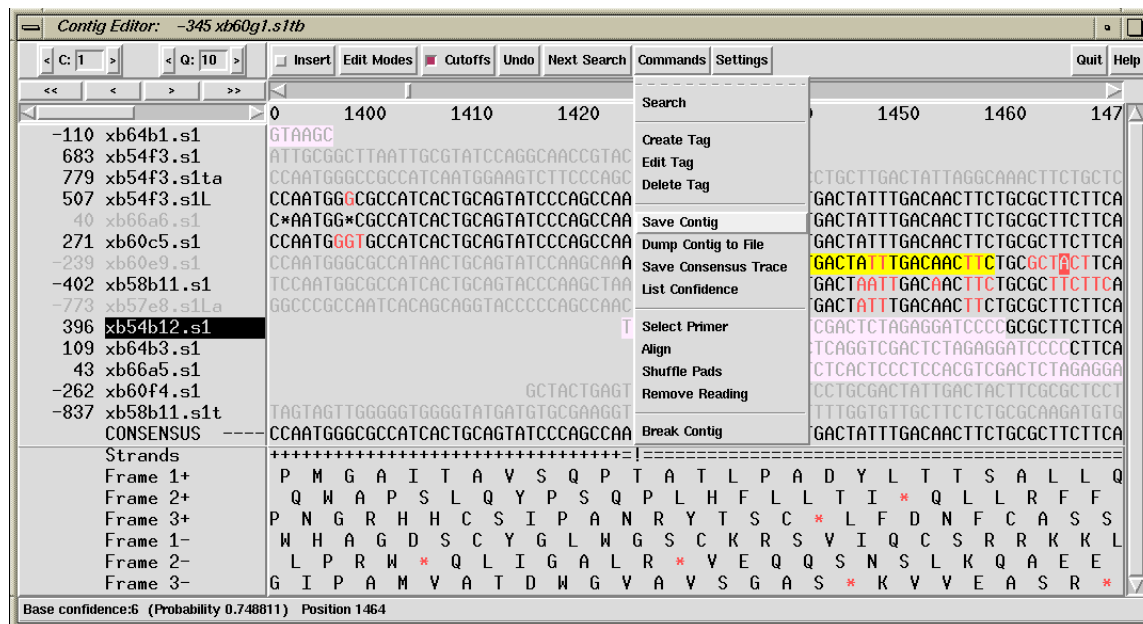
The Stop Codon Map plots the positions of all the stop codons on one or both strands of a contig consensus sequence. If the Contig Editor is being used on the same contig, the Refresh button will be enabled, and if used, will fetch the current consensus from the editor, repeat the search and replot the stop codons.



The figure shows a typical zoomed in view of the Stop Codon Map display. The positions for the stop codons in each reading frame (here all six frames are shown) are displayed in horizontal strips. Along the top are buttons for zooming, the crosshair toggle, a refresh button and two boxes for showing the crosshair position. The left box shows the current position and the right-hand box the separation of the last two stop codons selected by the user. Below the display of stop codons is a ruler and a horizontal scrollbar. The information line is showing the data for the last stop codon the user has touched with the cursor. Also shown on the left is the View menu which is used to select the reading frames to display.

### 2.1.3.7 Introduction to the Contig Editor

The gap4 Contig Editor is designed to allow rapid checking and editing of characters in assembled readings. Very large savings in time can be achieved by its sophisticated problem finding procedures which automatically direct the user only to the bases that require attention. The following is a selection of screenshots to give an overview of its use.



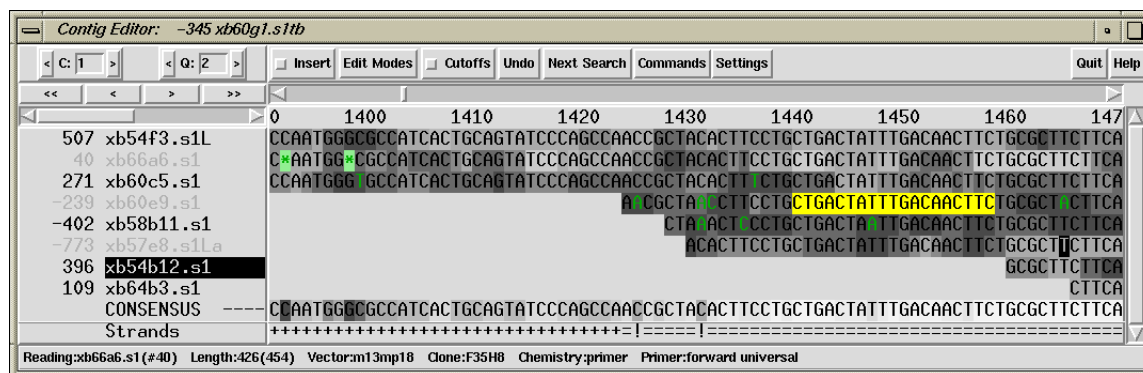
The figure above shows a screendump from the Contig Editor which contains segments of aligned readings, their consensus and a six phase translation. The Commands menu is also shown. The main components are: the controls at the top; reading names on the left; sequences to their right; and status lines at the bottom. Some of the reading names are written in light grey which indicates that their traces/chromatograms are being displayed (in another window, see below).

One reading name is written with inverse colours, which indicates that it has been selected by the user. To the left of each reading name is the reading number, which is negative for readings which have been reversed and complemented. The first of the status lines, labelled "Strands", is showing a summary of strand coverage. The left half of the segment of sequence being displayed is covered only by readings from one strand of the DNA, but the right half contains data from both strands.

Along the top of the editor window is a row of command buttons and menus. The rightmost pair of buttons provide help and exit. To their left are two menus, one of which is currently in use. To the left of this is a button which initially displays a search dialogue, and then pressing it again, will perform the selected search. Further left is the undo button: each time the user clicks on this box the program reverses the previous edit command. The next button, labelled "Cutoffs" is used to toggle between showing or hiding the reading data that is of poor quality or is vector sequence. In this figure it has been activated, showing the poor quality data in light grey. Within this, sequencing vector is displayed in

lilac. The next button to the left is the Edit Modes menu which allows users to select which editing commands are enabled. The next command toggles between insert and replace and so governs the effect of typing in the edit window.

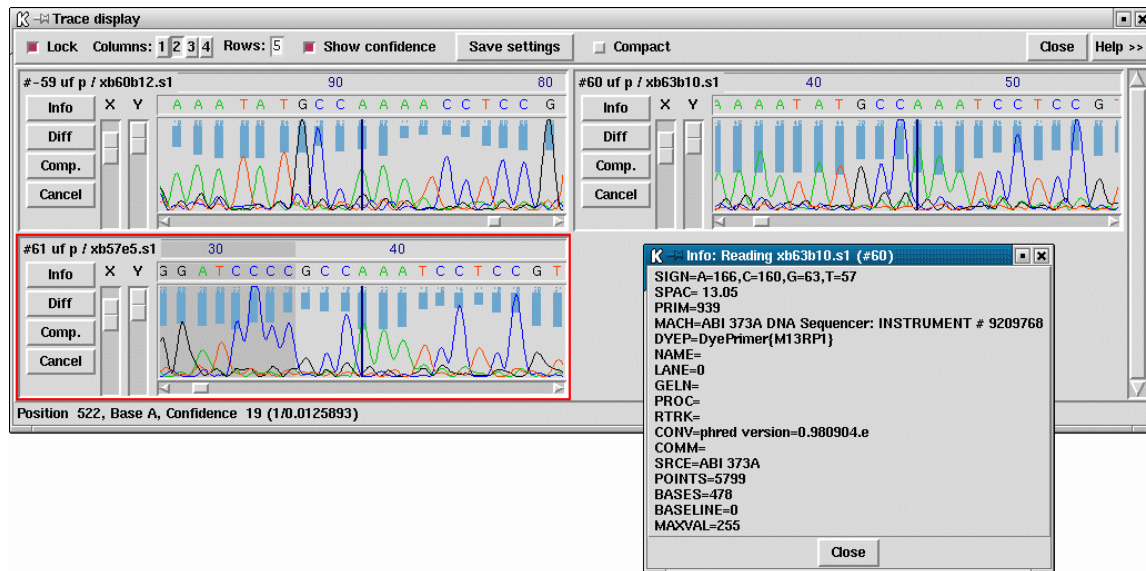
One of the readings contains a yellow tag, and elsewhere some bases are coloured red, which indicates they are of poor quality. The Information Line at the bottom of the window can show information about readings, annotations and base calls. In this case it is showing information about the reliability of the base beneath the editing cursor.



A better way of displaying the accuracy of bases is to shade their surroundings so that the lighter the background the better the data. In the figure above, this grey scale encoding of the base accuracy or confidence has been activated for bases in the readings and the consensus. This screenshot also shows the Contig Editor displaying disagreements and edits. Disagreements between the consensus and individual base calls are shown in dark green. Notice that these disagreements are in poor quality base calls. Edits (here they are all pads) are shown with a light green background. When they are present, replacements/insertions are shown in pink, deletions in red and confidence value changes in purple. The consensus confidence takes into account several factors, including individual base confidences, sequencing chemistry, and strand coverage. It can be seen that the consensus for the section covered by data from only one strand has been calculated to be of lower confidence than the rest. The Status Line includes two positions marked with exclamation marks (!) which means that the sequence is covered by data from both strands, but that the consensus for each of the two strands is different. The Information Line at the bottom of the window is showing



information about the reading under the cursor: its name, number, clipped length, full length, sequencing vector and BAC clone name.



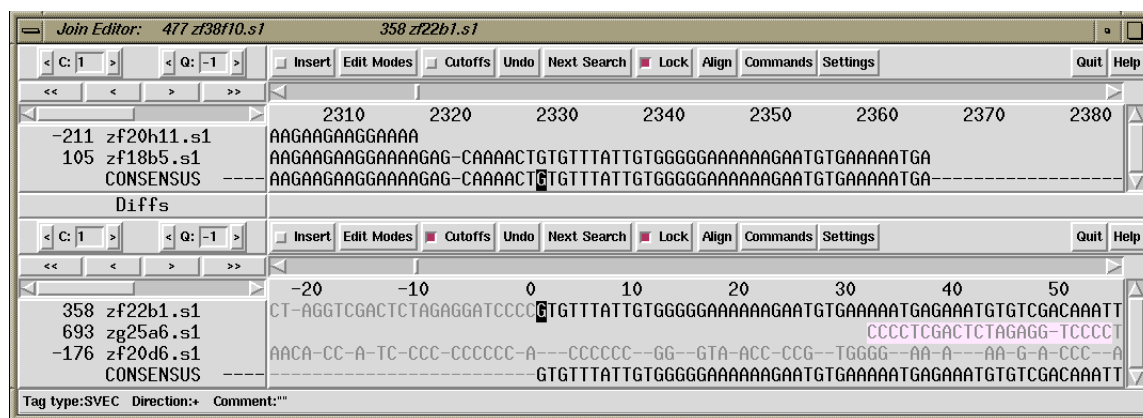
The Contig Editor can rapidly display the traces for any reading or set of readings. The number of rows and columns of traces displayed can be set by the user. The traces scroll in register with one another, and with the cursor in the Contig Editor. Conversely, the Contig Editor cursor can be scrolled by the trace cursor. A typical view is shown above.

This figure is an example of the Trace Display showing three traces from readings in the previous two Contig Editor screendumps. These are the best two traces from each strand plus a trace from a reading which contains a disagreement with the consensus. The program can be configured to automatically bring up this combination of traces for each problem located by the "Next search" option. The histogram or vertical bars plotted top down show the confidence value for each base call. The reading number, together with the direction of the reading (+ or -) and the chemistry by which it was determined, is given at the top left of each sub window. There are three buttons ('Info', 'Diff', and 'Quit') arranged vertically with X and Y scale bars to their right. The Info button produces a window like the one shown in the bottom right hand corner. The Diff button is mostly used for mutation detection, and causes a pair of traces to be subtracted from one another and the result plotted, hence revealing their differences..

### 2.1.3.8 Introduction to the Contig Joining Editor

Contigs are joined interactively using the Join Editor. This is simply a pair of contig editor displays stacked one above the other with a "differences" line in between. The Contig Join Editor is usually invoked by clicking on a Find Internal Joins, or Find Repeats result in the Contig Comparator. In which case the two contigs will appear with the match found by these searches displayed.

The few differences between the Join Editor and the Contig Editor can be seen in the figure below. Otherwise all the commands and operations are the same as those for the Contig Editor.



In this figure the Cutoff or Hidden data is being displayed for the right hand contig. One difference between the Contig Editor and the Join Editor is the Lock button. When set (as it is in the illustration) the two contigs scroll in register, otherwise they can be scrolled independently.

The Align button aligns the overlapping consensus sequences.

### 3 Searching for point mutations using pregap4 and gap4

The original version of these methods was described in *James K Bonfield, Cristina Rada and Rodger Staden, "Automated detection of point mutations using fluorescent sequence trace subtraction", Nucleic Acids Res. 26, 3404-3409, 1998.* The more recent work has been done by Mark Jordan and James Bonfield with advice from Graham Taylor, Andrew Wallace, Will Wang and others.

#### 3.1 Introduction to mutation detection

Our methods for detecting mutations are based on the alignment and comparison of the fluorescent traces produced by Sanger DNA sequencing. To use clinical terminology, samples from patients are compared to standard reference traces. Patient and reference traces should be produced using the same primers and sequencing chemistry, ideally from both strands of the DNA. The data shown in the examples below is from exon 11 of the BRCA1 gene.

The basic idea is illustrated in the following two figures which are screen dumps from our program gap4. The first shows a sample containing a point mutation and the second contains a heterozygous base position. The displays are bisected vertically: at the top left is the sample trace from one strand of the DNA, below that the reference trace for that strand, and underneath the difference between these traces which is obtained by subtracting one from the other. On the right is corresponding data from the other DNA strand (shown complemented).

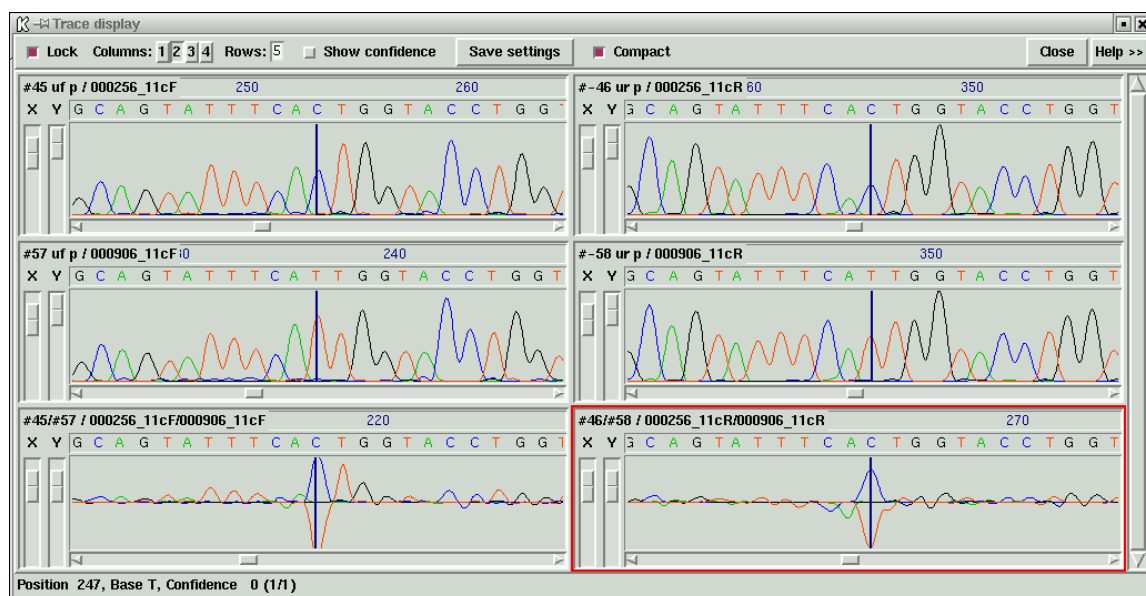


Figure 1. Top and bottom strand differences for a point mutation.

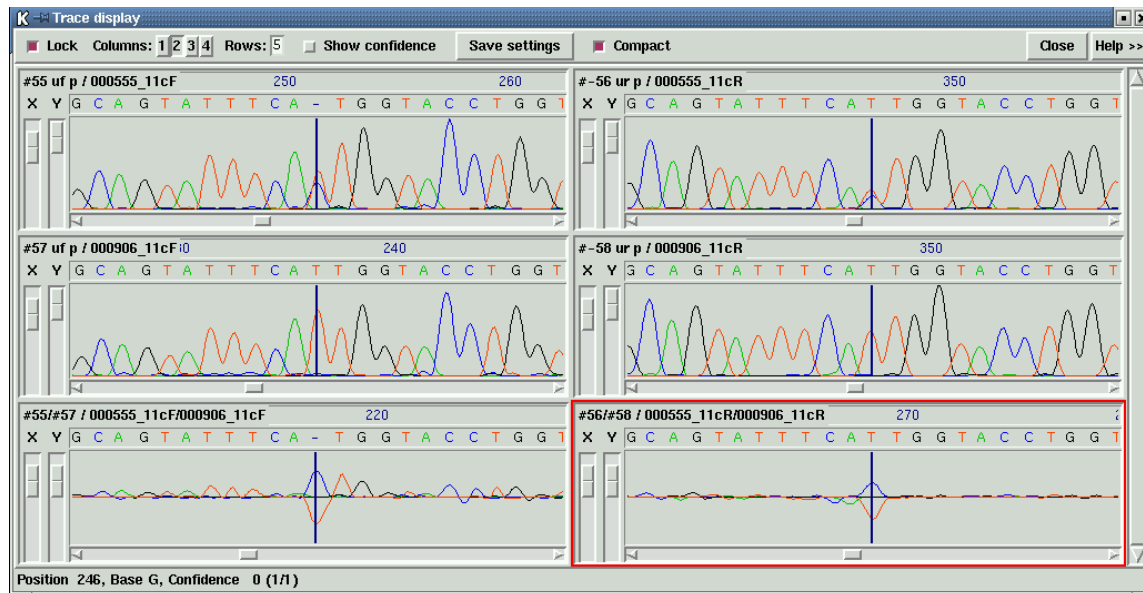


Figure 2. Top and bottom strand differences for a heterozygous base.

As can be seen, although no vertical scaling is performed the difference trace is quite flat or is consistently either above or below the mid-line, except at the sites of mutations. Near these are strong peaks, but notice that only for the mutated base are there peaks both above and below the mid-line. The context effects caused by the mutation produce peaks only in one direction.

It is perhaps necessary to point out that analysis of the traces is essential because base callers make mistakes: they can assign the wrong base types and also assign single bases where the DNA is heterozygous. An example of the latter can be observed in Figure 2: on one strand the base caller has assigned a "-" symbol at position 251, at least indicating uncertainty, but on the other strand it has assigned "T". The DNA is clearly heterozygous at this position. This means that simply looking for differences between patient sequences and reference sequences will cause point mutations and heterozygous bases to be missed (of course base calling errors will also create false differences).

These trace displays alone are very useful for visual inspection of data and are all some users want. However we also have programs which automatically analyse the trace differences and tag the bases which have significant peaks as possible sites of mutation.

Trace viewing is initiated from within the gap4 editor. Each record in the editor shows an individual reading with its number and name at the left. Negative numbers denote readings which have been complemented. Several sequences have special status. At the top is a sequence labelled with a letter S at the left edge. This is the reference sequence, here the EMBL entry HSLBRCA1 which covers the entirety of the BRCA1 gene. The numbering at the top of the display corresponds to positions in this reference sequence. The program has also coloured (green) all exons on the reference sequence. The bottom

DNA sequence in the editor is labelled "CONSENSUS". For mutation detection work this sequence is forced to be identical to the reference. Below the CONSENSUS sequence is the amino acid sequence for the reference. This is calculated on the fly using the feature table of the reference sequence and so translates only exons and in their correct reading frames. Two other sequences (near the top) are labelled R and F. These are the readings providing the reverse and forward reference traces for this segment of the data.

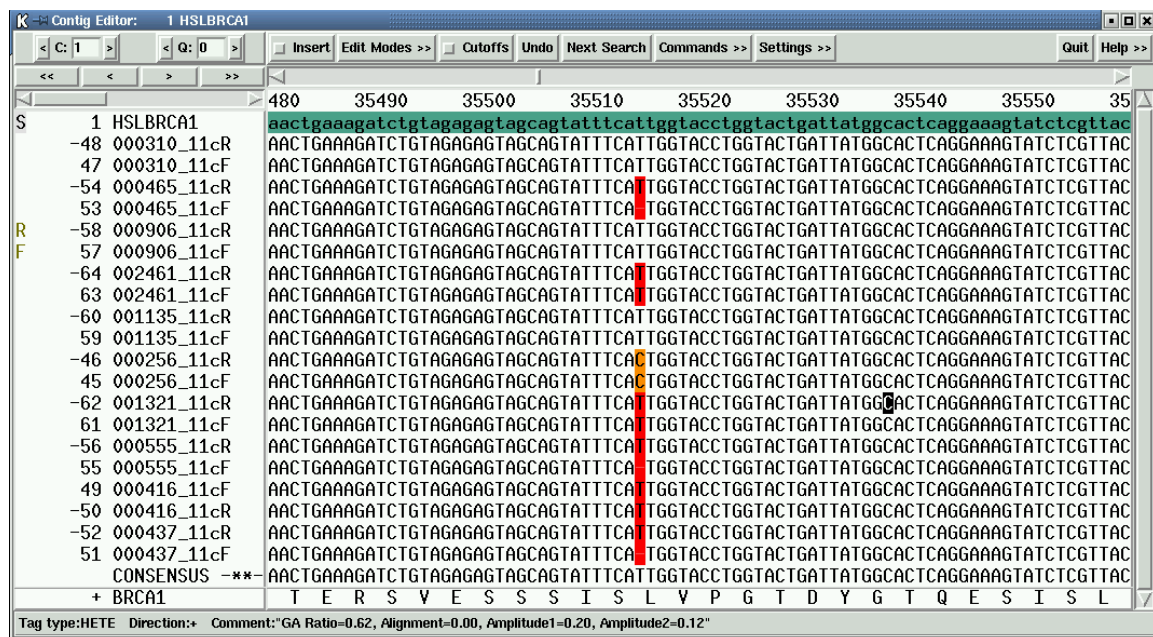


Figure 3. A set of aligned sequence readings displayed in the gap4 editor.

At the very bottom of the editor is an information line which is used to display data about items touched by the mouse cursor. Here it is showing data about one of the positions tagged as possibly being heterozygous. It includes the observed base types (G and A) and the scores achieved by the automated analysis.

The editor can be set to show only differences between readings and the reference; all matching bases appear as dots. For example, Figure 4. shows the same data as Figure 3,

but with the editor set to show differences, and the information line showing details about a possible mutation.

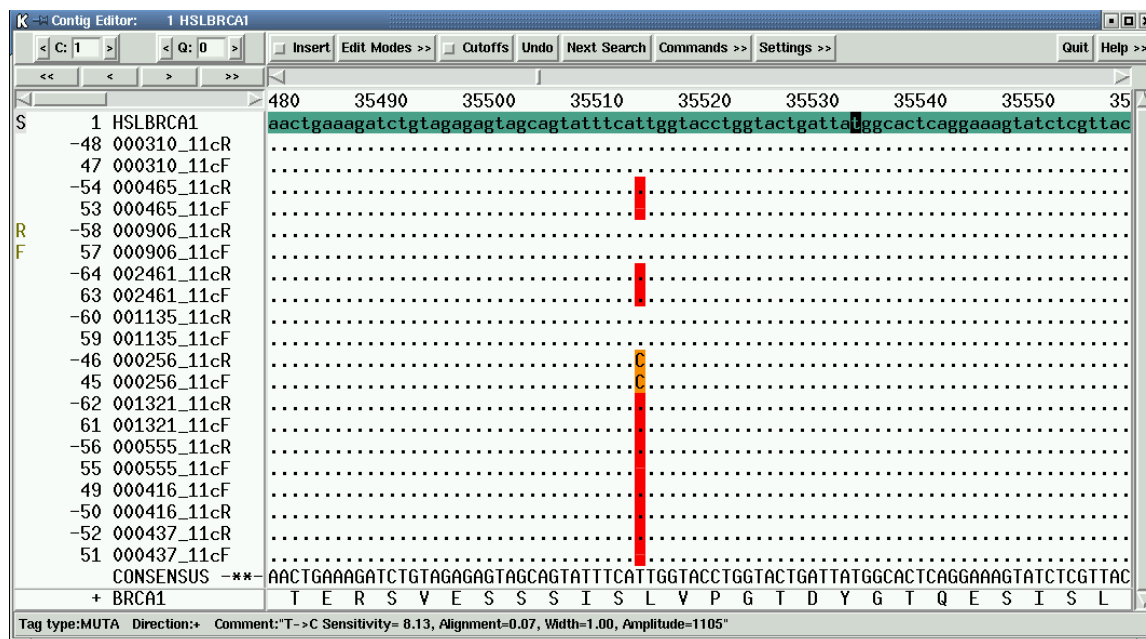


Figure 4. An alternative view of aligned sequence readings in the gap4 editor.

One column contains several bases tagged in red, signifying possible heterozygotes, and some in orange denoting possible point mutations. During visual inspection the program can be made to move the cursor from one tag to the next and to display the aligned traces as shown above in Figures 1 and 2.

It is also possible to have positive controls for displaying the trace differences; i.e. reference traces which contain the mutation. In this case the traces appear as shown in figure 5. Here the forward and reverse positive controls are shown to the right of the normal plots.

In Figure 5 the positive control difference plots are quite flat hence, in this case, providing confirmation of the presence of the heterozygous base.

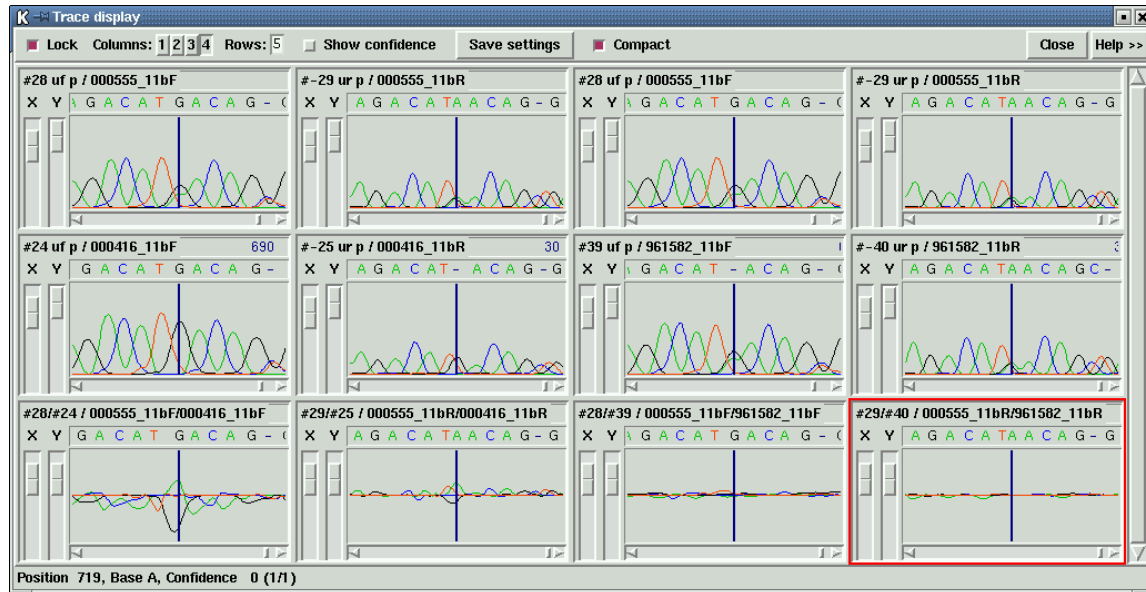


Figure 5. Top and bottom strand differences and positive control for a heterozygous base.

As mentioned above the package contains programs which can automatically compare the traces and their reference sequences. The output from these programs are the tags shown in the editor. Users can check the traces at these positions using the displays shown in Figures 1, 2 and 5; if necessary removing or adding tags. Alternatively users can rely entirely on visual inspection and create all tags themselves.

Once all the mutations are correctly tagged the program can produce a report which includes the reading names, mutation positions relative to the reference sequence, the actual change, its effect, and the evidence. An example is shown below in Figure 6.

```
001321_11aF 33885T>Y (silent F) (strand - only)
001321_11aF 34407G>K (expressed E>[ED]) (strand - only)
001321_11cF 35512T>Y (silent L) (double stranded)
001321_11cF 35813C>Y (expressed P>[PL]) (double stranded)
001321_11dF 36314A>R (expressed E>[EG]) (double stranded)
001321_11eF 36749A>R (expressed K>[KR]) (double stranded)
001321_11eF 37313T>K (noncoding) (strand - only)
000256_11eF 36749A>G (expressed K>R) (double stranded)
```

Figure 6. How gap4 reports mutations.



Here the first record is for reading 001321\_11aF, position 33885, T changed to T and C (i.e. is heterozygous) to produce no amino acid change, with evidence coming only from the complementary strand. The last record is for reading 000256\_11eF, position 36749, A changed to G, producing an amino acid change K to R, with evidence from both strands of the sequence. The penultimate record denotes a heterozygote in a noncoding region.

### 3.1.1 Mutation Detection Programs

The software handles batches of trace data from sequencing instruments. It performs all processing except base calling (although it can employ third party programs such as phred for this step). This includes file format conversions, quality clipping, scanning for mutations and heterozygotes, multiple sequence alignment, easy visual inspection of traces, production of reports, and the accumulation and storage of readings and traces. The software also handles the initialisation/configuration of standard reference files and databases for any project. The two main programs are pregap4 and gap4. Pregap4 prepares data for gap4 by automatically using a variety of smaller programs, including those used to search for mutations: mutscan (see [\[Mutation Scanner\]](#), page [\[Mutation Scanner\]](#)). Gap4 is used to store the aligned readings, to view the sequences and traces, and to produce a report listing the observed mutations.

Any number of sequences can be processed in a single run, and for each individual patient sample the operation is generally performed in two steps. First, via pregap4, the traces are aligned and compared to the reference traces and any possible mutations or heterozygous bases marked. Secondly, the data is transferred into a gap4 database from where users can visually check the differences between the reference and patient traces.

The program mutscan can automatically compare patient and reference traces to find point mutations and heterozygous bases. Users can set parameters which control the sensitivity of the algorithms (and hence which determine the ratio of false negative and positive results). Mutscan adds tags of type “mutation” or “heterozygous” to the patient files. The tags contain the numerical scores achieved at the site of the reported base changes, and they can be viewed via the gap4 editor. Mutscan is normally run via pregap4.

The description of the programs given below is presented in reverse order of use i.e. gap4 then pregap4, but first we give further details about the use of reference data.

### 3.1.2 Mutation Detection Reference Data

The mutation detection methods require reference traces and optionally reference sequences. Reference traces are used for automatic mutation detection and for visual inspection of trace differences. Reference sequences are used in gap4 to provide a base numbering standard, and if required to provide feature table entries to control translation and mutation reporting.

### 3.1.3 Reference Sequences

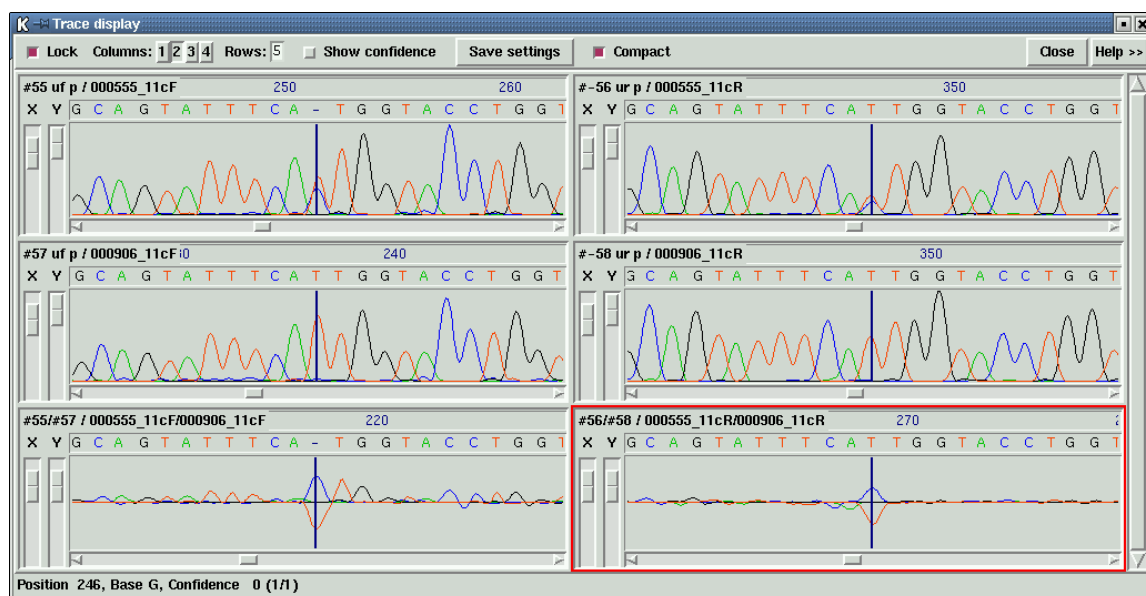
Reference sequences are used in gap4. Here they can be used to define a numbering system independent of gaps introduced to produce alignments. The numbering can start at any point in the reference sequence. If the reference sequence is entered with a feature table the features are converted to tags and can be used to control translation of the sequence in the contig editor. For mutation detection work the reference sequence and feature table enable mutations to be reported using positions defined by the reference sequence, and also



allows the effect of the mutations to be noted. Gap4 is able to store entries from the EMBL sequence library complete with their feature tables. These feature tables are converted to gap4 database annotations (tags), which means that they can be selectively displayed in the template display and editor, and used to translate only the exons (in the correct reading frame). Obviously it may be useful to augment the feature tables with the sites of known polymorphisms or deleterious mutations so that they can be displayed in gap4 as landmarks. When it comes to producing a report of the observed mutations the feature table is used to work out if a mutation is expressed and if so what the amino acid change is. Additional tags can be created to specify the positions of the primers or restriction sites used to obtain data covering segments of the sequence. For any project the reference sequence need only be set up once. Either project databases can be started with the reference sequence already configured or the reference can be assembled along with the reading data. The reference sequence can be designated (or reassigned) as follows. In pregap4 it can be named in the module "Reference Traces". In the gap4 editor it can be set by right clicking on its name. Once set it should appear labelled "S" at the left edge of the editor.

### 3.1.4 Reference Traces

References traces are used by the automatic mutation detection program mutscan (see [\[Mutation Scanner\]](#), page [\[undefined\]](#)), and by the trace difference display in the gap4 editor. Ideally forward and reverse reference traces should be available and should be obtained using the same primers and sequencing chemistry as the patient data. From the "settings" menu of the editor the trace display can be set to "Auto-Diff traces". Once this is activated, whenever the user double clicks on a base in the editor sequence display, not only is the reading's trace displayed, but also its designated reference trace plus the difference between them. If its complementary reading is available, its trace and reference trace and their differences are also displayed. These trace displays and the editing cursor scroll in synch.



Top and bottom strand differences for a heterozygous base.

The preferred way of assigning reference traces to readings is by use of "naming conventions"; that is to have a simple set of rules which control the names given to the trace files. It can be seen in the figures showing the editor that forward and reverse readings from the same patient have names with a common root but which end either F or R. This both ties the two together (so the software knows which is the corresponding complementary trace when the user double clicks on a reading) and also enables the association of readings and their reference traces. Once a convention has been adopted the rules can be defined for pregap4 by loading them via the "Load Naming Scheme" option in its File menu. For any batch of readings the reference traces are defined within pregap4's "Reference Traces" module. Note that this mode of operation, by allowing the specification of only one forward and one reverse trace, limits each batch of traces processed to those which correspond to a given pair of reference traces. The size of the batch is unlimited.

The alternative way of specifying the reference traces is to right click on their names in the editor. This also allows positive trace controls to be specified (which is not possible in pregap4).

### 3.1.5 Using The Template Display With Mutation Data



Figure 7. The template display showing the whole of the BRCA1 gene (exons in green).

The view obtained from the Template display and shown in Figure 7 is not of practical use but serves here to illustrate the overall arrangement of the data for our chosen example the BRCA1 gene. This figure shows the entirety of the EMBL entry HSLBRCA1 with its exons marked in green. Only exon 11 has patient trace data stacked above it.

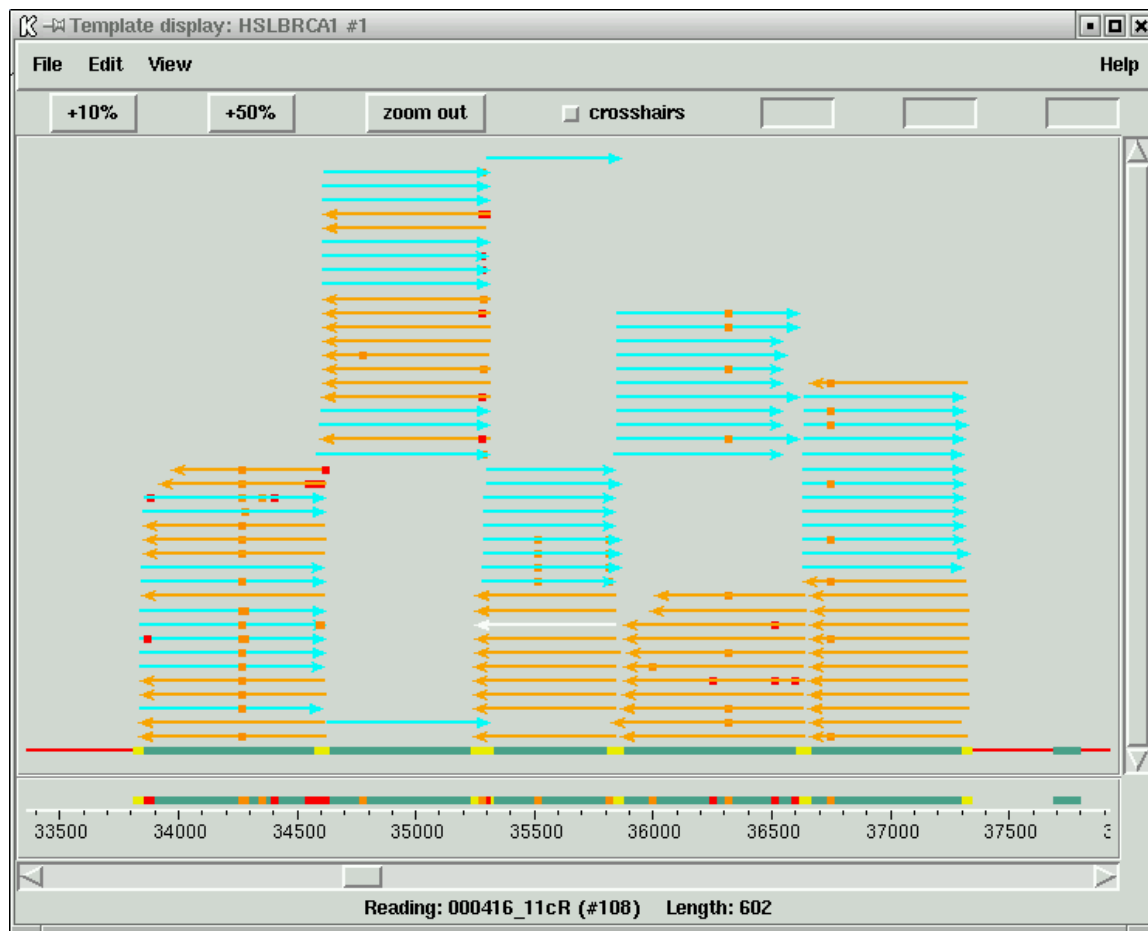


Figure 8. A zoomed-in version of the data shown in Figure 7.

Here we can see all the readings covering exon 11. Forward readings are light blue, reverse readings orange, primers are marked in yellow, mutations in red and orange. A common mutation appears in the leftmost set of readings and illustrates the value of using the template display for visualising the overall pattern of the tagged mutations.

### 3.1.6 Configuring The Gap4 Editor For Mutation Data

The current version of the gap4 editor contains very many options that are not needed for mutation data. Given sufficient demand a version tailored for mutation studies could be produced. For now it might make it easier to understand the program if its origin as a genome assembly program is borne in mind. Here we outline the options and settings

relevant to mutation studies. The assignment of reference sequence and traces is described above. From the editor they can be set by right clicking on the reading names.

Gap4 enables segments of sequences to be annotated (or tagged). Each tag has a type (eg primer) and each type has an associated colour. Each instance of a tag can include editable text. This text can be viewed and edited by right clicking on the tag and selecting "Edit tag", after which a text box will appear. Gap4 can display annotations/tags as background colour and the user can specify which tag types are shown. For mutation studies the following tag types may usefully be activated, and all others turned off. Using the "Set Active Tags" option in the "Settings" menu first click on "Clear all". Then click on "primer". To add further types you must hold down the "Ctrl" key on the keyboard while clicking. Now scroll down and click on "Mutation", "Heterozygous" and "FEATURE CDS". Add any others required, then click "OK".

The following configurations are performed via the "Settings" menu.

Gap4 has three consensus generation algorithms. When using a reference sequence it is convenient if the consensus shown in the editor is forced to be the same as the reference. This will be the case if either the "Weighted base frequencies" or the "Confidence values" consensus algorithms are being used. This selection is made using the "Consensus algorithm" option.

Translations are shown in what gap4 refers to as the "Status" line. To enable automatic translation of the exons defined in the reference sequence, in the "Status Line" option set "Translate using feature tables".

To enable automatic display of trace differences, in the "Trace Display" option set "Auto-Diff Traces".

To show only the base differences between the consensus/reference, set "Highlight Disagreements". These can be shown by dots or colour.

To show base confidence values set "Show reading quality" and also make sure that the value in the box labelled "Q" at the top left of the editor is set to 0 or greater.

To force forward and reverse reading pairs to be shown in adjacent records in the editor set "Group readings by templates" (NB this assumes that an appropriate naming scheme has been used).

If a reference sequence is assigned, the numbering at the top of the sequence will reflect the base positions in that sequence. Any pads in the reference sequence are ignored. If no reference sequence is assigned, the numbering will ignore pads if the "Show unpadded positions" option is activated.

At the bottom of the "Settings" menu is an option to "Save settings". Use of this will mean that the current configuration will be set automatically next time the editor is used (and hence the steps just described only need to be performed once).

### 3.1.7 Using The Gap4 Editor With Mutation Data

The current version of the editor has a fixed width and a maximum height. If too many sequences are present at any position a vertical scrollbar on the right edge can be used to move them up and down. The CONSENSUS line will always be visible, but at present,

the reference sequence is scrolled along with all the other sequences and so may disappear. Horizontal scrolling is achieved in the usual ways, plus by use of the >, >> and <, << buttons. The reading names can be moved left and right using the scrollbar above them.

Configure the editor as described above.

The traces for readings (and their reverse) can be examined over their full length one at a time by simply double clicking on them then scrolling along. Any mutations observed can be labelled by right clicking on the base in the editor display and invoking the "Create tag" option. This brings up a dialogue box. At the top is a button marked "Type:comment"; clicking on this will bring up another dialogue with a list of all the tag types; choose the appropriate one ("Heterozygous" or "Mutation"). There are obviously many advantages to examining the traces like this using gap4. However, if the automated mutation detection methods are trusted, or used in way that makes them trustworthy for the type of study being undertaken, then there are quicker ways of examining the data.

The "Next Search" button at the top of the editor gives access to many types of search, one of which is "tag type". If this is selected a button appears labelled "Tag type COMM(Comment)". Clicking on this will bring up a dialogue showing all the available tag types. If the user selects, say "Mutation", each time the "Next Search" button is used the program will position the editing cursor on the next mutation tag. Double clicking will automatically bring up the appropriate traces as shown in figures 1, 2 and 5. The user can view the traces and if necessary alter the tag (eg delete it if it is a false positive).

Once all the data has been checked and all mutations and heterozygous bases have been tagged a report can be generated using the "Report Mutations" option in the editor "Commands" menu. Note that it is also possible to simply report all differences between base calls and the reference, but the usual procedure is for the program to report all bases tagged as "Mutation" or "Heterozygous". Example output is shown above in Figure 6. The report appears in the gap4 "Output window" which can be saved to disk by right clicking on the text and selecting "Output to disk".

### 3.1.8 Processing Batches Of Mutation Data Trace Files

It is not clear which is the best way of organising the data for the simplest and most efficient processing using the current programs, but for now we make the following suggestions.

We assume that the region of the DNA being studied has a standard set of forward and reverse primer pairs covering all segments of interest and that a standard reference sequence in EMBL format is available.

We recommend that batches of data from single primer pair combinations are processed separately, using separate temporary gap4 databases. For example, exon 11 of BRCA1 can be covered by five pairs of forward and reverse primers and we suggest that batches of traces obtained from each of these primer pairs should be processed using five gap4 databases.

Each processing run should create a new database and should enter, not only the new sets of patient data for that particular primer pair, but also the corresponding reference sequence and reference traces.

Obviously when several primer pairs are needed to cover a given region of the DNA (eg for BRCA1) the same reference sequence would be used for all the primer pairs.

An alternative to the above is to create a template database for each primer pair which contains the data for the corresponding forward and reverse reference traces plus the fully annotated reference sequence. These template databases are copied to create a temporary database for each new batch of data for the given primer pair.

Whichever of these two strategies is adopted each batch of new data is processed, analysed and assembled into these temporary databases, inspected visually, and a mutation report generated.

The use of separate temporary databases simplifies the assignment of reference traces and the use of the report generation function.



Figure 9. An overview of a database containing data for only one primer pair of BRCA1

For long term storage and to facilitate larger studies, the content of each of these temporary databases is then transferred to archive databases, after which the temporary databases are no longer needed. The archive databases could be restricted to individual primer pairs or could accommodate data covering the whole of the reference sequence.

### 3.1.9 Processing Batches Of Mutation Data Trace Files Using Pregap4

All the data processing other than visual inspection of traces and report generation is handled by the program pregap4. Pregap4 achieves this by running a set of individual programs selected by the user.

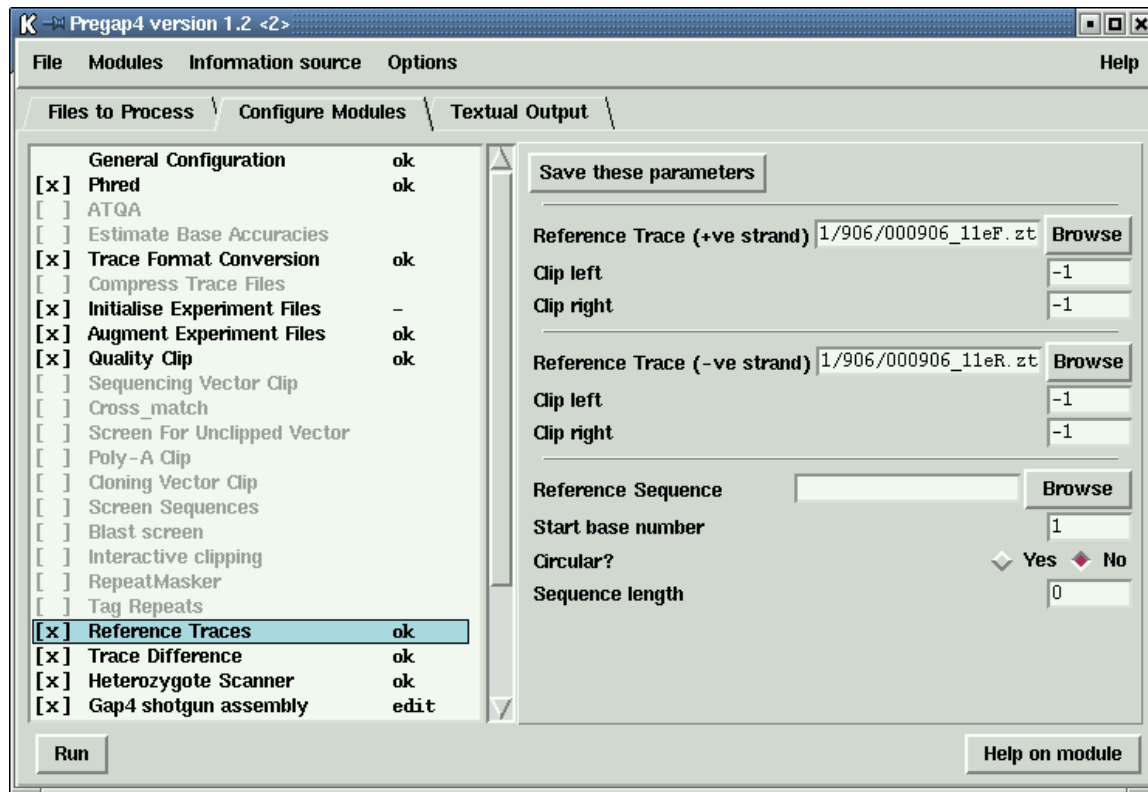


Figure 10. The pregap4 Configure Modules window showing a typical list of mutation data option selections.

The "Configure Modules" window shown in Figure 10. is used to select which programs to apply to a batch of data, and to configure their usage. On the left is a list of programs and options, with "x" showing the ones that have been selected. If the user clicks on an option name its name is given a blue background and its configurable parameters are shown in the right hand panel to enable the user to alter them. Here "Reference Traces" has been selected which enables the user to set the reference traces and sequence.

The other selected options (marked with "x") are typical of the ones used for mutation detection studies. Below we describe the use of each plus a few alternatives. All of the options are descibed in more detail elsewhere in our documentation, our intention here is to give an overview of their use during mutation studies.

Note that the window labelled "Files to Process" is used to tell the program which files to process as a batch.

### 3.1.10 Configuration Of Pregap4 For Mutation Data

#### *General Configuration*

This option allows the user to select whether the trace names used for the samples should be the same as their file names or should be the names stored inside the files.

#### *Phred*

Phred is a base caller which also assigns confidence values to each base. Generally the data passed to pregap4 has already been base called. However not all base callers assign confidence values and so it can be useful to apply phred or ATQA (which does not base call but does assign confidence values). Alternatively "Estimate Base Accuracies" can be applied which is a simple program for providing numerical values which reflect the signal to noise ratio for each base, and which can be used instead of confidence values. (Note that if quality clipping is used, its score thresholds depend on whether confidence values of eba values are used).

#### *Trace Format Conversion*

This option can be used to convert bulky files such as those of ABI to a compact format such as SCF or ZTR without loss of the data required for trace display.

#### *Initialise Experiment Files*

The input to gap4 and several of the other programs used here is a data format known as Experiment file format. This step, which has no configurable parameters is essential for mutation data processing.

#### *Augment Experiment Files*

The section on Reference Traces outlined the use of "Naming Schemes" for associating pairs of forward and reverse readings, and for assigning reference traces. The naming scheme must be loaded from pregap4's File menu. "Augment Experiment Files" must be activated in order for the naming scheme to be applied. No parameters need be set.

#### *Quality Clip*

The reliability of the base calls varies with position along the sequence. Near to both ends the data is less reliable. The "Quality Clip" option trims the ends of the sequences by analysing their confidence values or accuracy estimates (if present) or the density of unknown bases in the sequence. By observing these "clip points" other processing programs will work more reliably.

#### *Reference Traces*

As explained above it is necessary to specify a reference trace (preferably one for each strand of the data if processing data from both strands). The Reference sequence can also be set here. Note that even if our suggestion to preload the reference traces into the gap4 database is followed, it is still necessary to specify them here for use by the mutation detection modules.

#### *Trace Difference*

This is the program which compares the patient and reference traces to search for possible mutations. It adds data to the experiment files to mark each predicted mutation, and this data will appear as tags in the gap4 database. It



can also create a new trace file containing the difference of the reference and the sample. The numerical parameters control the sensitivity of the algorithms, and hence the ratio between the numbers of false positive and negative results.

#### *Heterozygote Scanner*

This is the program which compares the patient and reference traces to search for possible heterozygous bases. It adds data to the experiment files to mark each predicted heterozygous base, and this data will appear as tags in the gap4 database. The numerical parameters control the sensitivity of the algorithms, and hence the ratio between the numbers of false positive and negative results.

#### *Gap4 shotgun assembly*

In order to be able report the positions of mutations relative to the reference sequence, and to be able to compare sets of samples from patients, it is necessary to perform multiple sequence alignment on the data. This is termed "assembly" and is usually performed by gap4, although other programs can be operated via pregap4. If following the suggestion to preload the reference sequence to a temporary database for each batch, supply the name of this database here. Otherwise a new database should be named and created from this option. (If this strategy is adopted make sure that the reference sequence and the references traces are assembled!) The parameters that control the assembly process and are described elsewhere.

Note that pregap4 has the facility to save its configuration and parameter settings. This means that the current configuration will be set automatically next time the program is used (and hence the steps just described only need to be performed once). In addition pregap4 can be run non-interactively by typing a single line on the command line. Taking these two capabilities together, means that only one line need be typed in order to process all subsequent batches of data (assuming the file names are reused, which is easy to arrange.)

### **3.1.11 Discussion Of Mutation Data Processing Methods**

At present pregap4 and gap4 clearly show their primary usage in the field of genome assembly, but versions tailored to mutation studies can be created once the requirements are agreed. Ideally all processing should be controlled by a single program which once configured for any project should require users to provide only the project name - all other file names and parameters could be preset, and all processing, including archiving and backup, performed automatically, leaving the data ready for visual inspection.

The automatic mutation and heterozygote detection programs work well on all the test data we have but now they require evaluation by external groups. Such analysis would enable us to improve the algorithms and to tune their parameters. At present we know that sometimes a base will be declared both as a mutation and as a heterozygous position when visual inspection shows that it is one or the other.

There is still much that can be done overall to improve the methods, but the text above summarises their status in July 2002. Although currently valuable for real scientific and clinical work they should perhaps be viewed as prototypes.



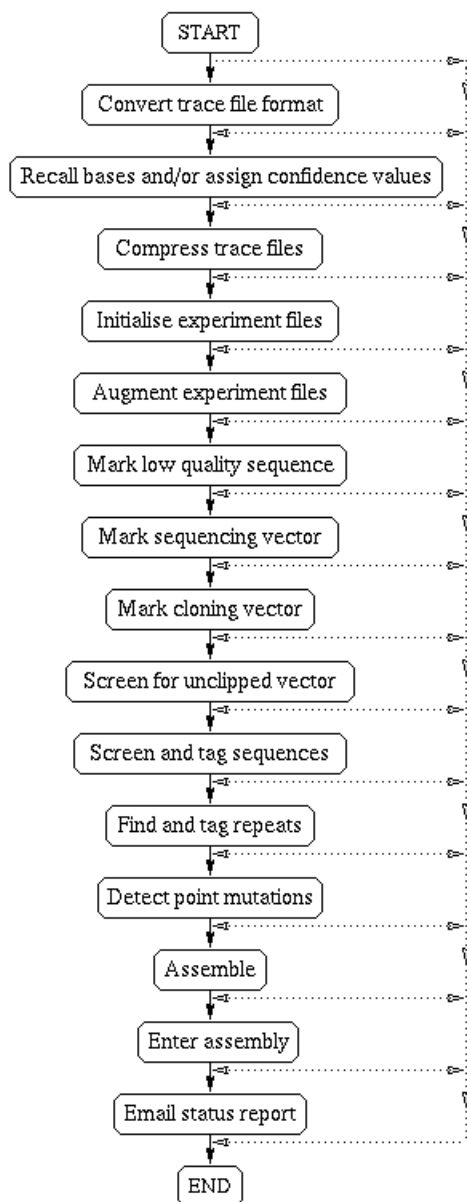
## 4 Preparing Readings for Assembly Using Pregap4

### 4.1 Introduction

Before entry into a gap4 database the raw data from sequencing instruments needs to be passed through several processes, such as screening for vectors, quality evaluation, and conversion of data formats. Pregap4 is used to pass a batch of readings through these steps in an automatic way. It provides an interface for setting up and configuring the processing and for controlling the passage of the readings through each stage. The separate tasks are termed "modules" and each module is typically managed by a dedicated program. Pregap4 wraps all of these modules into a single easy to use environment, whilst maintaining the flexibility to select and extend the processing modules. It is an, as yet, unpublished replacement of the program pregap *Bonfield, J.K. and Staden, R. Experiment files and their application during large-scale sequencing projects. DNA Sequence 6, 109-117 (1996).*

#### 4.1.1 Summary of the Files used and the Processing Steps

Gap4 stores the data for an assembly project in a gap4 database. Before being entered into the gap4 database the data must be passed through several steps via pregap4. The range of tasks that can be performed using pregap4 are shown schematically in the following figure.



The package can handle data produced by a variety of sequencing instruments, and also data entered using digitisers or that has been typed in by hand. One of the first steps is to convert trace files, such as those of ABI, which are in proprietary format, to SCF files.

Next, as originally put forward in *Bonfield, J.K. and Staden, R. The application of numerical estimates of base calling accuracy to DNA sequencing projects. Nucleic Acids Research 23, 1406-1410 (1995)*, if they are not already included in the files, base call confidence values are calculated, and are normally stored in the reading's SCF file.

Next the base calls are copied from the trace files to text files known as Experiment files.

Note it is also possible to enter sequence readings in the form of FASTA files for use at this stage of the processing, in which case they will be automatically converted to Experiment file format.

All the subsequent processes operate on the Experiment files.

Experiment file format is similar to that of EMBL sequence entries in that each record starts with a two letter identifier, but we have invented new records specific to sequencing experiments. Gap4 can make use of information about readings which may not be contained within the raw data files, such as sequencing chemistry and whether it is a forward or reverse reading. Gap4 will work without this information, but at a reduced level. For instance knowing which forward and reverse readings belong together allows gap4 to check the validity of assembly and for automatic ordering of contigs.

One of pregap4's next tasks is to augment the Experiment files to include data about the chemistry, vectors, primers and templates used in the production of each reading, and if necessary it can extract this information from external databases, or via local reading name conventions. Once the Experiment file for a reading contains all the necessary information the remaining processing programs can be used in turn to analyse the data.

First the reading is marked at both ends to define the range of reasonable quality base calls.

Then the reading is searched for the presence of sequencing vector at the 5' end 3' ends.

Next the sequence is checked for the presence of "cloning" vector, i.e. non-sequencing vectors, such as those of BACs.

The final check of this type is to screen the reading for any vector that may have been missed in the previous searches.

The next check is to screen the reading for any set of sequences which it may be contaminated by, such as *E. coli*.

Note that vector sequence files are normally stored in the package vectors directory/folder. If a file of vector file names is used the vector sequences can also be stored in its directory/folder. Files of file names and vector-primer files can also contain environment variables to define the location of vector files.

Vector\_primer files, vector sequence files and files of file names must be stored in plain text files.

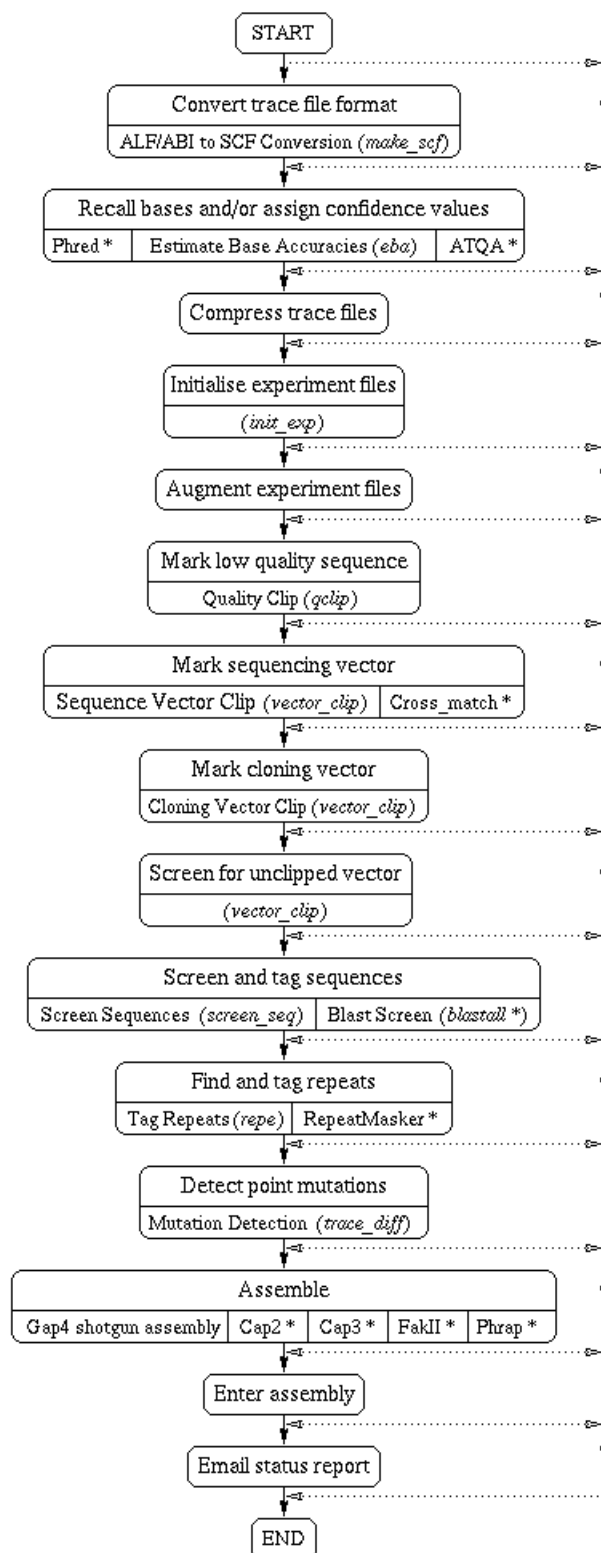
Pregap4 is usually used non-interactively once the modules have been configured, but some groups prefer (or have the time) to check the data by eye using the program `trev` at this stage.

Another option is to search the readings for families of known repeats. This will tag any regions which are found to match known repeats.

Some groups are using the package for mutation studies and the final `pregap4` option, prior to assembly is to use the mutation scanner program to search the readings for mutations (see [\[Mutation Scanner\]](#), page [\[Mutation Scanner\]](#)).

Pregap4 can also be used to assemble the readings into a `gap4` database, or to assemble the readings using an external assembly engine such as `FAKII`, and then to enter that assembly into a `gap4` database.

The following figure shows an overview of the range of tasks that can be performed by `pregap4`, plus the names of the programs which can be used. The program names marked with an asterisk (\*) are not included in the Staden Package and must be obtained from elsewhere.



It is unlikely that any particular user will want to employ all of these options and one of pregap4's modes of use is to enable users to configure the program for their work. Not only can they select which tasks should be performed, and which of the alternative programs ("modules") should be used for them, but also the order in which they are applied. Although it is very rarely a problem, this high level of flexibility comes at a price in the current version of pregap4: pregap4 does not include code to check on the logicity of the configuration set by a user and will attempt to execute the modules in the order given. There are some users, who having read this section, will configure pregap4 to perform assembly before creating the Experiment files from the trace files. Pregap4 will attempt to do this and no data will be assembled as the files given to the assembly engine will be in the wrong format. This is just something to be aware of.

Pregap4 uses configuration files to remember the setup for each user or project. These files define which modules are activated and what their parameter settings are. These files, which can obviously save considerable amounts of time, are created automatically and can be saved from the Configure Modules Window once the configuration is complete.

The trace files are not altered, but are kept as archival data so that it is always possible to check the original base calls and traces. The trace files are used by gap4 to display traces and to compare the final consensus sequence with the original data, therefore they must be kept online for the lifetime of the project. To save disk space it is best to use SCF files and, if they were derived from a proprietary format such as that of ABI, to remove the originals.

Any changes to the data prior to assembly (and we recommend that none are made until readings can be viewed aligned with others) are made to the copy of the sequence in the Experiment file. For example the results of all the searching procedures outlined above are added as new records to each reading's Experiment file. The reading data, in Experiment file format, is entered into the project database, usually via one of the assembly engines. All the changes to the data made by gap4 are made to the copies of the data in the project database. Once the data has been copied into the gap4 database the Experiment files are no longer required.

During processing pregap4 uses temporary files. The number and nature of these files depends on the modules used. At the very least pregap4 will produce files containing the names of the input files and the result of their processing. Those that were processed successfully will be stored in a file with a name ending ".passed" and those that failed in one ending ".failed". The ".passed" file can be used as a file of input file names for assembly into gap4 (assuming that a pregap4 assembly module has not already been used).

While it is running, pregap4 will create files with a file name prefix defined by the user, and store them in an output directory of the user's choice.

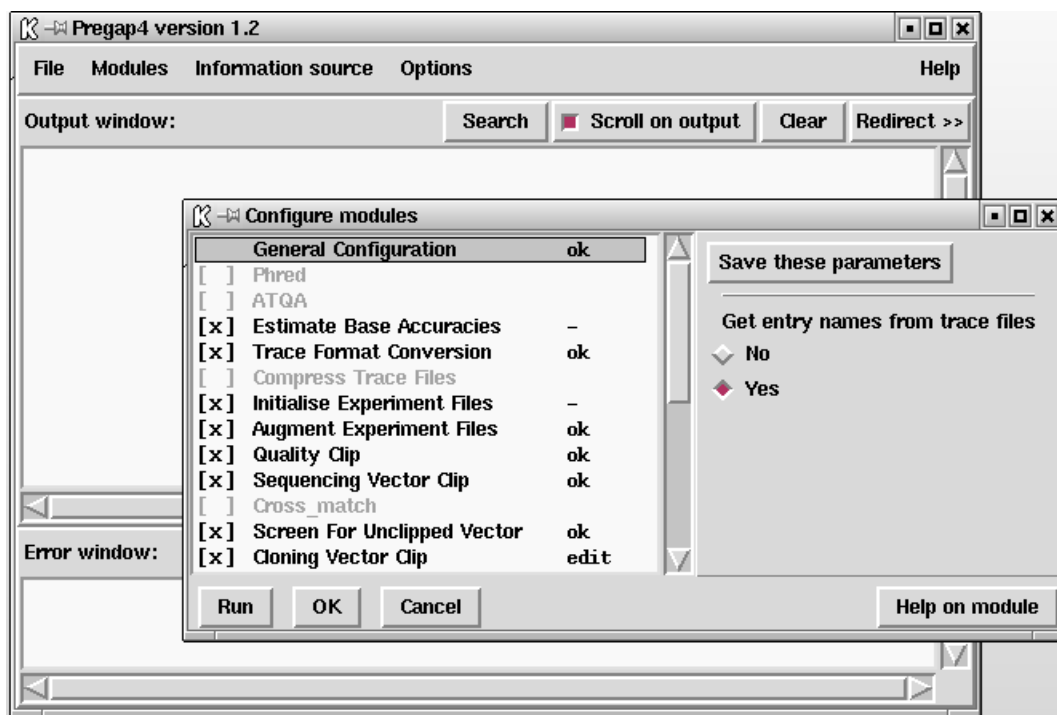
When processing has finished pregap4 will produce a report containing information from each module and the final list of passed and failed sequences.

### 4.1.2 Introduction to the Pregap4 User Interface

Pregap4 provides interfaces to define the batch of data files to be processed, which modules are to applied to them; to configure the modules, and to start the processing. It also provides mechanisms for adding and removing modules, but this facility will be used far less often than the others.

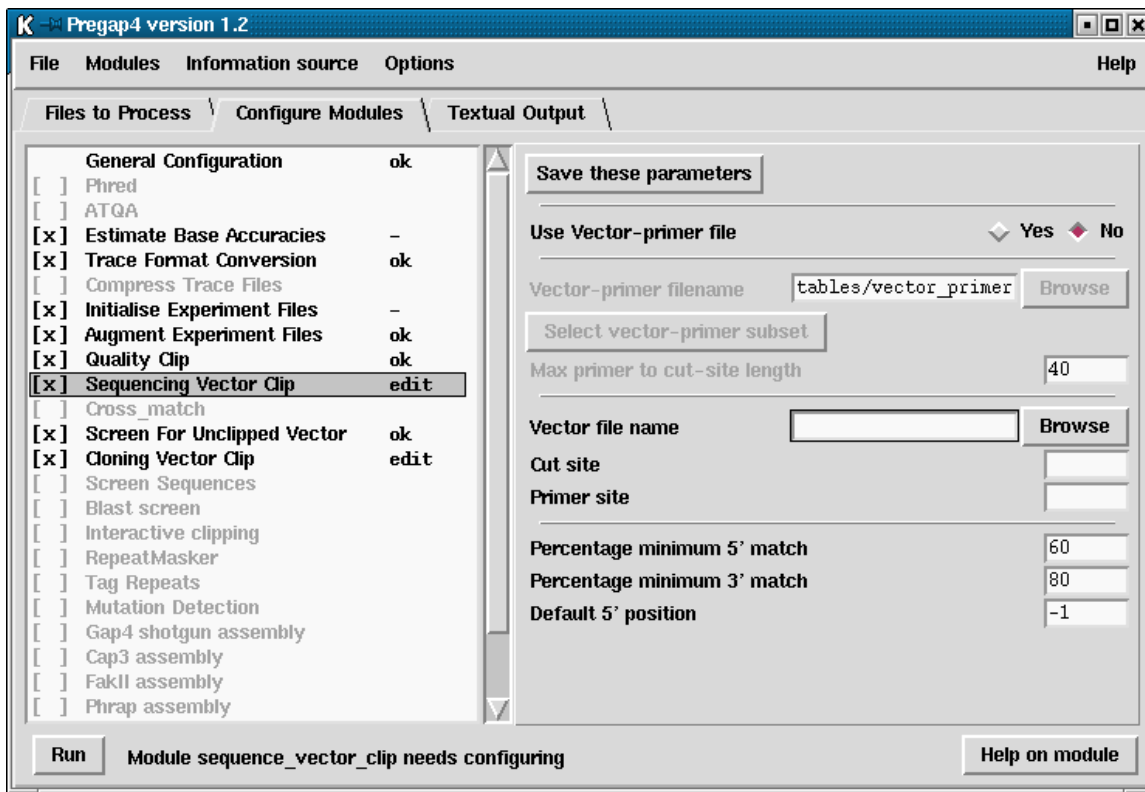


Pregap4 supports two styles of windowing. The default method is a compact mode, with the alternative being "separate" mode - similar to gap4 and spin.



This is the "separate" window style. Here the main window is always visible, with commands in the main window bringing up new windows. In the picture above the configure window can be seen on top of the main window.

The second style is "compact" mode.

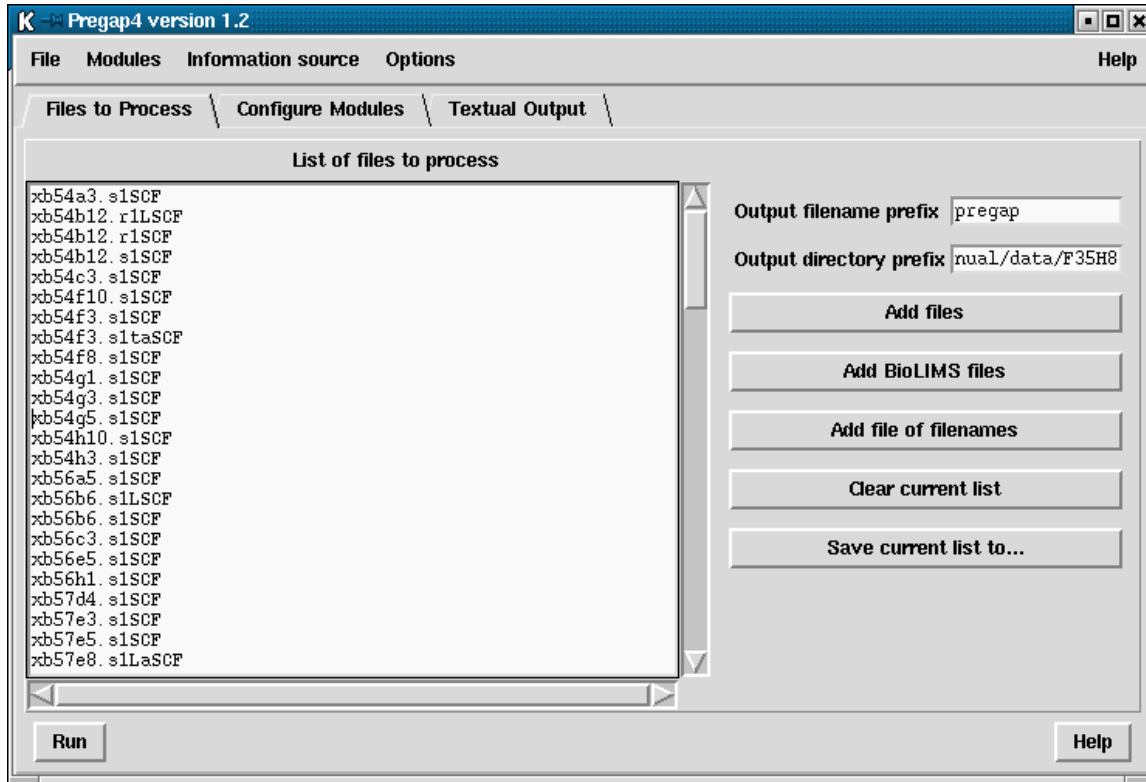


In the compact picture above the most common top level windows are "pages" in a tabbed notebook. The benefit is greatly reduced screen space and quicker controls, but the text output window is no longer permanently visible. The Window Style can be changed using the options menu.

#### 4.1.2.1 Introduction to the Files to Process Window

Pregap4 operates on batches of files. These files can be binary trace files (in ABI, ALF or SCF format), Experiment Files, or plain text, and do not need to all be in the same format. The Files to Process Window is used to define which files are to be processed. The "Files

to Process" dialogue (see below) can be brought up from the File menu, or by pressing the appropriate tab when in compact\_win mode.



On the left hand side of the figure is the current list of files to process. This list can be edited simply by clicking with the mouse and typing.

On the right side of the panel is the pregap4 output filename prefix, the output directory name, and several buttons. The filename prefix is used when pregap4 needs to create files. For example after processing there may be *prefix.passed*, *prefix.failed* files. All files will be created within the output directory.

The buttons allow selection of the files to process. The "Add files" button will bring up a file browser, which will allow one or more files to be selected. Pressing Ok on the file browser will then add the selected files to the "List of files to process" panel on the left side of the pregap4 window.

The "Add file of filenames" button may be used to select a list of files whose filenames have been written to a 'file of filenames'.

The "Clear current list" button will remove all filenames from the list.

Both the "Add files" and "Add file of filenames" button append their selections to the list of files to process, so to replace the current list the "Clear current list" button must first be used.

The "Save current list to..." button may be used to produce a new file of filenames, containing the combined list of files to process.

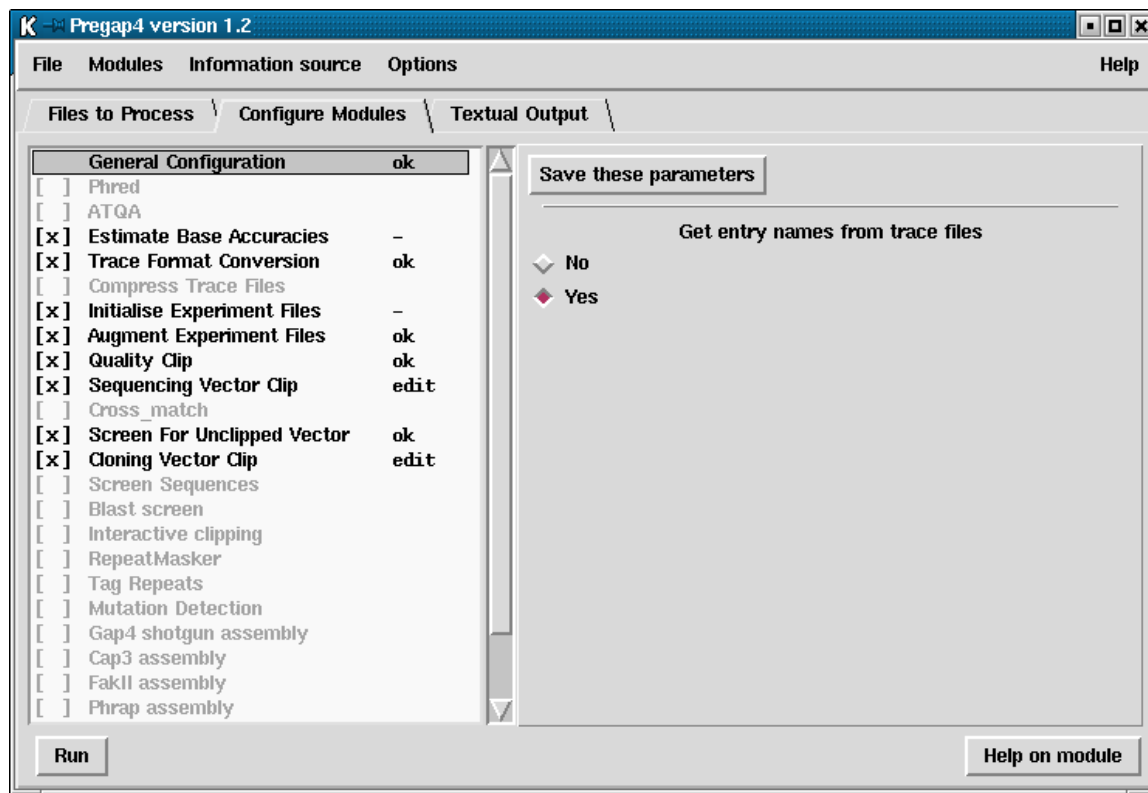
It is also possible to specify the files to process on the command line. Three examples:

```
pregap4 -fofn files
pregap4 xb54a3.s1SCF xb54b12.r1LSCF xb54b12.r1SCF
pregap4 *SCF
```

#### 4.1.2.2 Introduction to the Configure Modules Window

The "Configure Modules" dialogue is available from the Modules menu or, when using the compact window style, by pressing the Configure Modules tab.

As can be seen in the figure below, the left side of the display contains a list of the currently loaded modules. One module in this list will be highlighted. The right side of the display shows the configuration panel for this highlighted module and is module specific.



The module list shown on the left consists of a series of module names and their status, and is termed the "enable status". The tick or cross at the left of the name indicates whether this module is enabled. The text to the right of the module name indicates whether the module has been given all the parameters needed for it to run. This will be one of "ok" (all configuration options have been filled in), "-" (no configuration options exist for this module), "edit" (further configuration is required) or blank (this module is disabled).

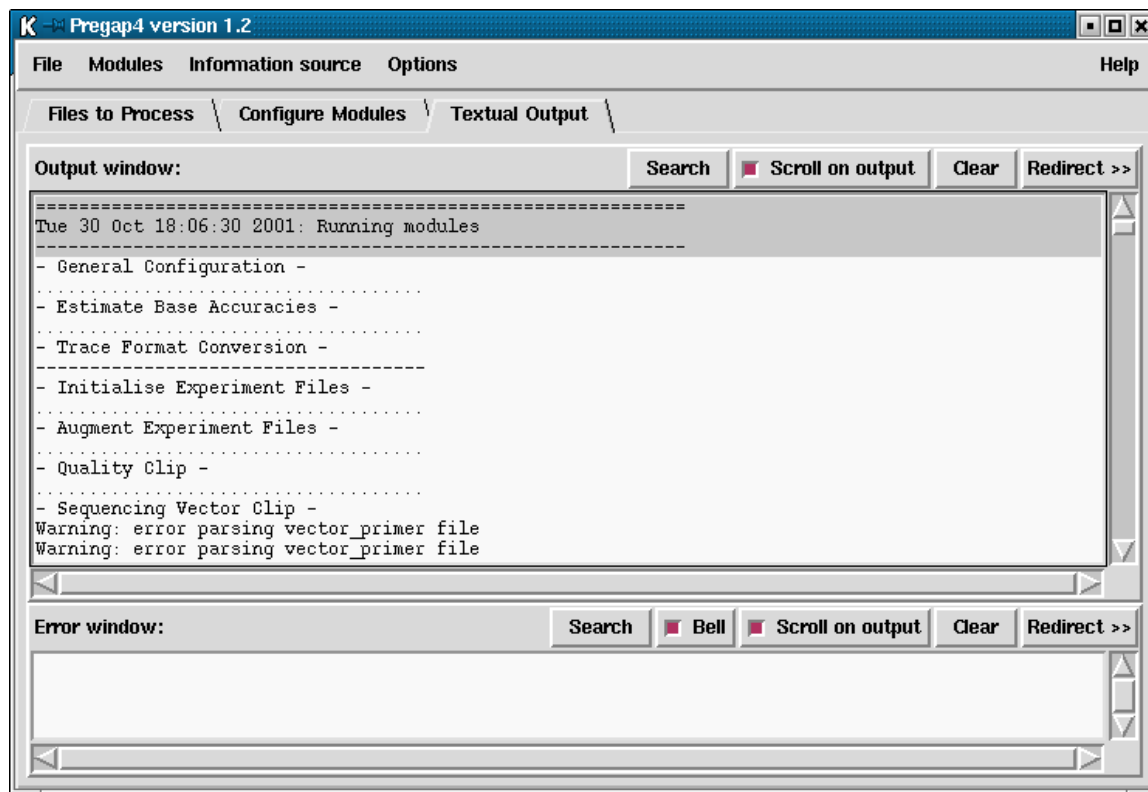
The "enable status" can be toggled by left clicking on the tick/cross to the left of the module name. The enable status can be written to the current Pregap4 configuration file using the "Save Module List" or "Save All Parameters" commands in the Modules menu. Left clicking anywhere on a module name in the module list will switch the pane on the right side of the window to display any available parameters for this module. Not all modules will have parameters to configure.

For modules that do have parameters, the top line of the configuration panel will contain two buttons labelled "Select params to save" and "Save these parameters". The "Select params to save" button will add check boxes next to each parameter. Clicking on these check boxes allows selection of individual parameters to save for this module. Once these have been selected pressing the "Save" will save only those selected to the pregap4 configuration file. Pressing the "Save these parameters" button will save all parameters for this module to the configuration file.

The bottom strip of the window is an "Information Line".

#### 4.1.2.3 Introduction to the Textual Output Window

Pregap4 has a main text output window identical to that of gap4 and spin. It is used for showing textual results in the top section and error messages in the lower part. Full details of the user interface are given elsewhere, but an example of the Text Output Window is given below.



#### 4.1.2.4 Introduction to Running Pregap4

When pregap4 is started the user first needs to select the files to process. This is done using the "Files to Process" command (from the File menu). Alternatively the files can be specified on the command line at the time of starting up Pregap4. The "Configure Modules" tab allows for the currently available modules to be enabled or disabled, and the module parameters edited accordingly.

Once all modules have been configured (so that none have `edit` listed next to their name) pregap4 is ready to begin processing. This is started by pressing "Run" or by selecting "Run" from the File menu.

When pregap4 has a setup that would be useful in the future "Save All Parameters (in all modules)" from the Modules menu can be used, and pregap4 will store all the module parameters to a configuration file ready for subsequent runs.

To run pregap4 in a non interactive mode use "**pregap4 -nowin**". This will not bring up a graphical interface and will attempt to "Run" automatically. Hence it is necessary to also specify the files to process on the command line and also to have previously configured pregap4.

When processing has finished pregap4 will produce a report containing information from each module and the final list of passed and failed sequences.

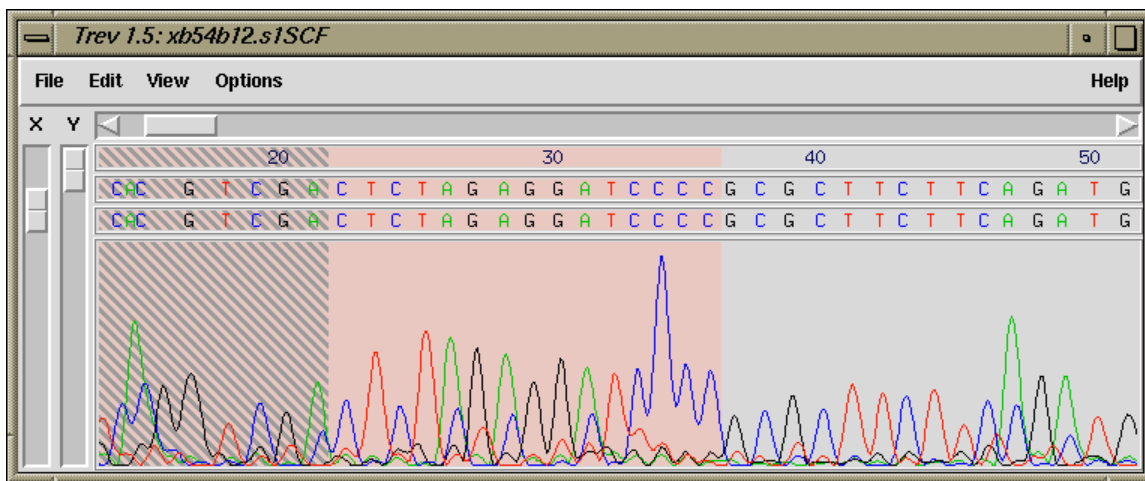
If for any reason pregap4 fails a particular step in the processing, users are strongly recommended to correct whatever has caused the module to fail, clean up any files it has created, and then repeat the whole process. That is, until users have a good understanding of what happens at each stage of processing, it is better to repeat all the steps with the original list of files, than to try to guess which step to continue from.

## 5 Viewing Traces Using Trev

For some types of sequencing project it is convenient to view and edit the chromatogram data prior to assembly into a gap4 database, and this is the function of the program *trev*.

*Trev* displays the original trace data, its base calls and confidence values, and it allows the sequence of the trace to be edited and the left and right cutoffs to be defined. Several file formats can be read in addition to our own Experiment Files, and 'SCF' files. Any edits made are normally saved to Experiment files, not to the chromatogram files which we regard as archival data.

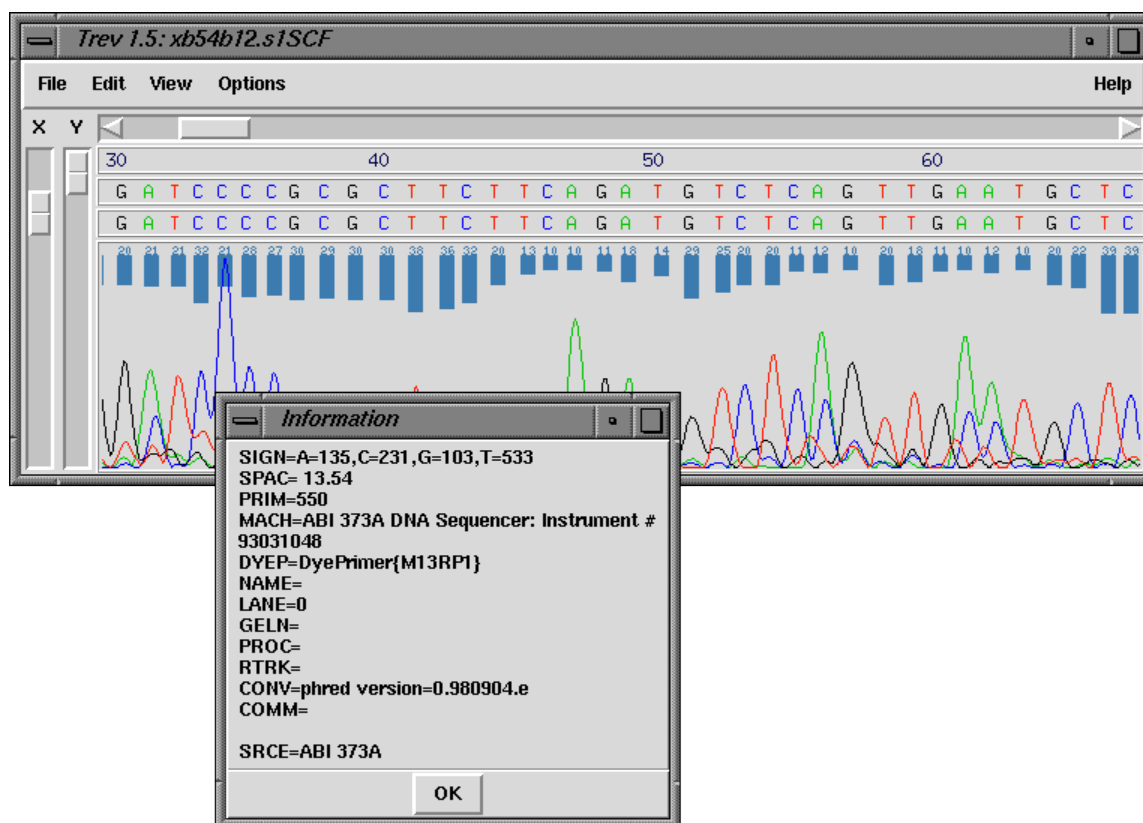
A typical display from *trev* is shown below. It includes the trace data, the original sequence, the edited sequence, the menu bar, and the name of the sequence being edited. The left cutoff region is shown shaded.



The trace can be scrolled using the scrollbar directly beneath the menubar. The trace can be magnified in the vertical and horizontal directions using the scale bars to the left of the trace.

The base numbers, original sequence, edited sequence, confidence values and the trace can each be switched on or off, and the font for the original and edited sequence is selectable.

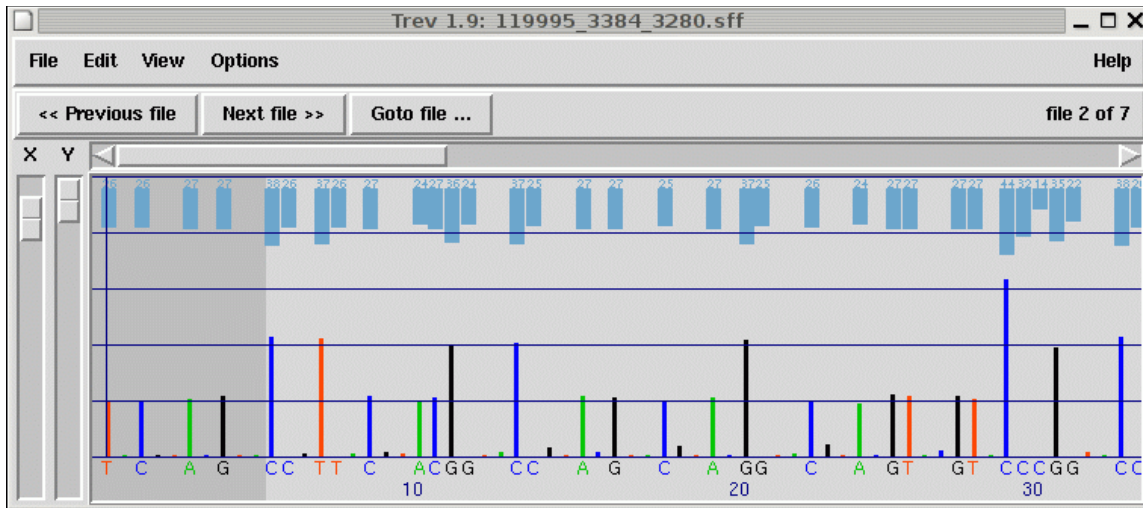
The figure below shows the bases, edited bases, a histogram of the confidence values, the traces, and the Information Window which can be switched on from the View Menu.



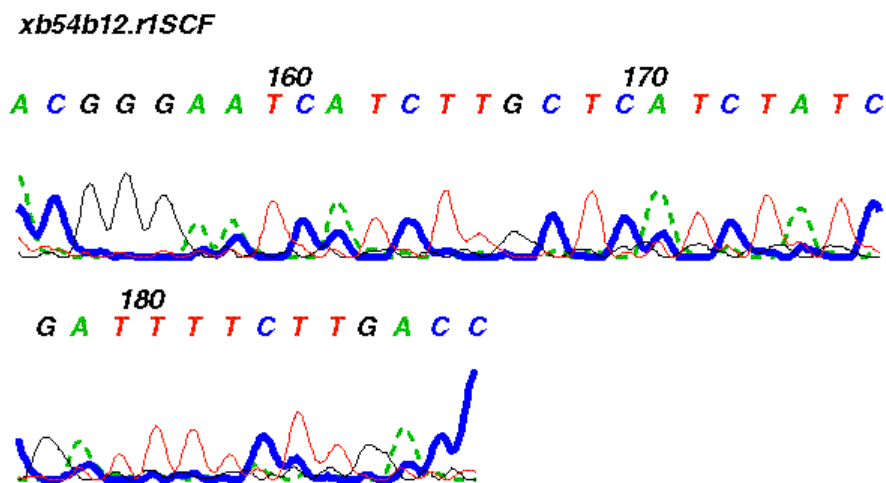
Trev uses "io-lib" for handling the various sequencing instrument file formats. This means it has support for ABI, MegaBace (when saved in ABI format), SCF (used by LiCor and some other manufacturers), ZTR and SFF (454).



The above pictures all come from instruments using the Sanger sequencing method, however more recently support has been added for pyrosequencing methods (as used by 454 Life Sciences amongst others). An example of this is below.



Trev can be used to produce postscript files of the traces so that they can be printed. The colours, line widths, etc are configurable. An example is shown in the figure below.



Note that we strongly recommend that readings are not edited prior to assembly as it is far better to edit them when their alignment with other readings can be seen.



## 6 Analysing Sequences Using Spin

### 6.1 Introduction

Spin is an interactive and graphical program for analysing and comparing sequences. It contains functions to search for restriction sites, consensus sequences/motifs and protein coding regions, can analyse the composition of the sequence and translate DNA to protein. It also contains functions for locating segments of similarity within and between sequences, and for finding global and local alignments between pairs of sequences. To help assess the statistical significance of comparisons the program can calculate tables of expected and observed score frequencies for each score level. Most analytical functions which operate on single sequences add their graphical results to a "SPIN Sequence Plot" that is associated with the sequence being analysed. (An exception is the restriction enzyme search which produces its own separate window.) Most functions which compare pairs of sequences add their results to a "SPIN Sequence Comparison Plot". The SPIN Sequence Plot and the SPIN Sequence Comparison Plot each have associated sequence display windows: the Sequence Display and the Sequence Comparison Display. These allow the text of the sequences to be viewed and use cursors to show the corresponding positions in the graphical displays. The graphical plots can be zoomed, and cursors or crosshairs can be used to locate the positions of the individual results. Plots can be superimposed.

#### 6.1.1 Summary of the Spin Single Sequence Functions

Spin's main single sequence analytical functions are accessed via the Statistics, Translation and Search menus. The "Statistics" menu contains options to count and plot the base composition and also to count the dinucleotide frequencies.

The "Translation" menu contains options to set the genetic code, translate to protein, find open reading frames and write the results in either feature table format or as fasta format protein sequence files, and to calculate codon tables.

The "Search" menu contains a variety of different searching techniques. "Protein genes" has four methods for finding protein genes, (accessed as a subcomponent of the Codon Usage Method), and. There is also a method to search for tRNA genes. It is also possible to perform subsequence or string searches and restriction enzyme searches. There are searches for start and stop codons, splice junction searches, and general motif searches using weight matrices.

#### 6.1.2 Summary of the Spin Comparison Functions

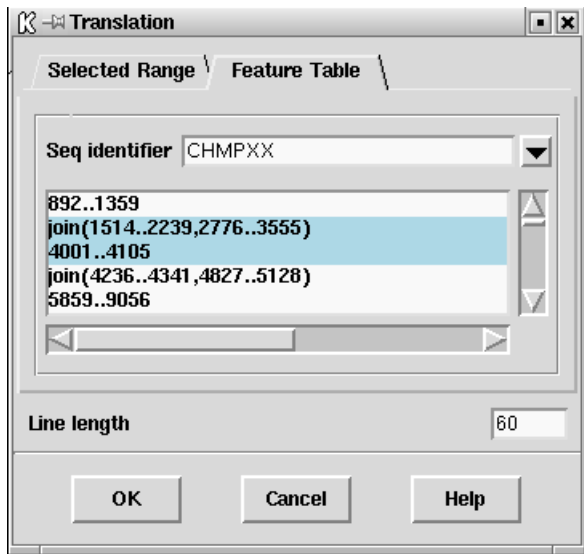
This section outlines the functions obtained from the Comparison menu. All produce graphical and textual output. Using a score matrix, the "Find similar spans" function compares every segment of one sequence with all those of the other and reports those that reach a user defined score. The segments are of a fixed length (span) set by the user. To look for short matching segments of any length, and allowing gaps, a local dynamic programming routine can be used. The fastest routine for locating segments of similarity (and generally only suitable for DNA sequences) finds all identical subsequences (or words). For a quick global comparison of sequences using a combination of the Matching Words and Matching Spans algorithms the "Find best diagonals" algorithm can be used. Global alignments can be produced and plotted using a dynamic programming algorithm.

### 6.1.3 Introduction to the Spin User Interface

Spin has several main displays. The first is a top level window from which all the main options are selected and which receives textual results. Most analytical functions which operate on single sequences add their graphical results to a "SPIN Sequence Plot" that is associated with the sequence being analysed. (An exception is the restriction enzyme search which produces its own separate window.) Most functions which compare pairs of sequences add their results to a "SPIN Sequence Comparison Plot". The SPIN Sequence Plot and the SPIN Sequence Comparison Plot each have associated sequence display windows: the Sequence Display and the Sequence Comparison Display. These allow the text of the sequences to be viewed and use cursors to show the corresponding positions in the graphical displays.

Spin is best operated using a three button mouse, but alternative keybindings are available. Full details of the user interface are described elsewhere, and here we give an introduction based around a series of screenshots.

The main window (shown below) contains an Output Window for textual results, an Error window for error messages, and a series of menus arranged along the top. The contents of the two text windows can be searched, edited and saved. Each set of results is preceded by a header containing the time and date when it was generated.

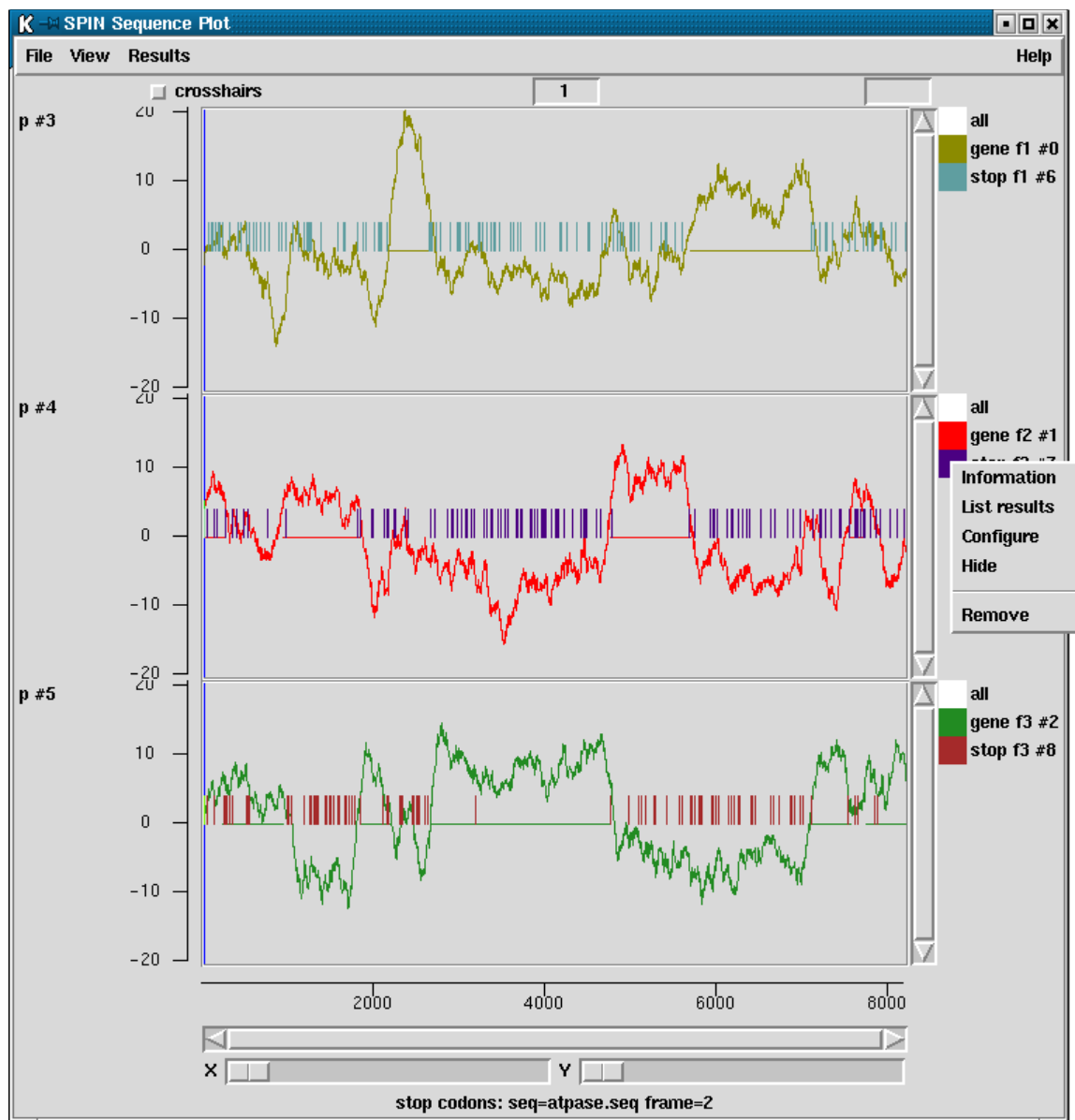


As can be seen the main menu bar contains File, View, Options, Sequences, Statistics, Translation, Comparison, Search and Emboss menus. In general most functions add their graphical results to a "SPIN Sequence Plot", but those obtained from the Comparison menu add their results to a "SPIN Sequence Comparison Plot".

### 6.1.3.1 Introduction to the Spin Plot

Most of the spin functions display their results in a two-dimensional plot called a "spin plot". Sets of matches from a single invocation of a function are termed "a result". Each result is plotted using a single colour which can be configured via the results manager.

The figure shown below shows a spin plot window containing the results of a gene search method based on codon usage, superimposed on a search for stop codons. Each plot window contains a cross hair. Its x position is shown in sequence base numbers in the left hand box above the plot, and the y coordinate, expressed using the score values of the gene search, is shown in the right hand box.



At the right hand side of each panel is a set of square boxes with the same colours as the lines drawn in the adjacent plot. These icon-like objects represent individual results and allow the user to operate on them. For example at the right of the middle panel is a pop-up menu containing the items: "Information", "List results", "Configure", "Hide" and "Remove"..

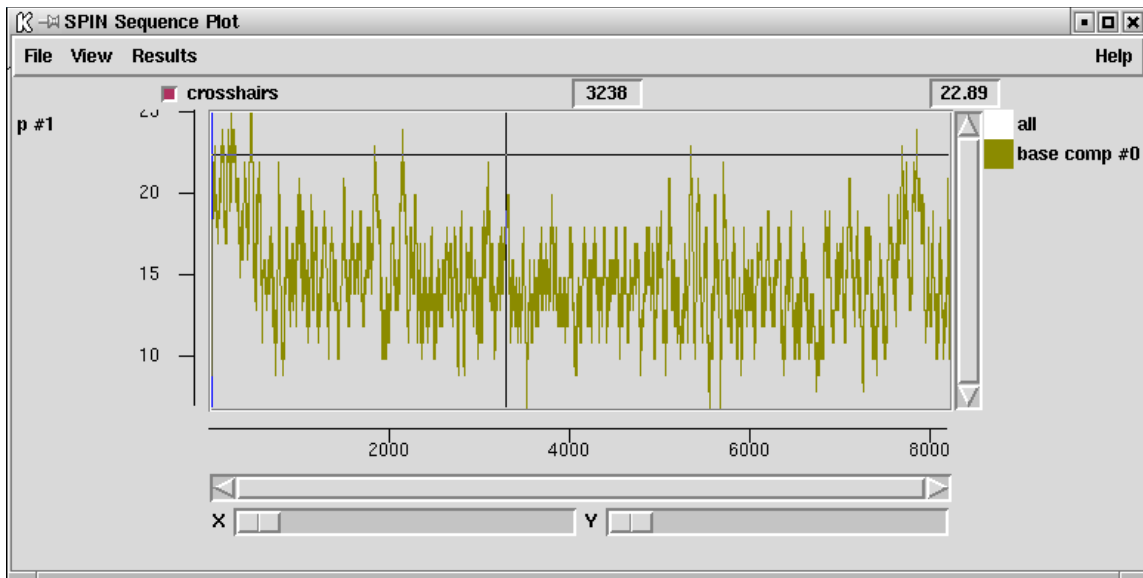
These icons can also be used to drag and drop the results to which they correspond. This is activated by pressing the middle mouse button, or Alt left mouse button, over the box and then moving the cursor over the spin plot to the new location or anywhere outside the spin plot

Each spin plot window also contains a cursor that denotes the position of the cursor in the Sequence display window. The user can move a cursor by clicking and dragging with the middle mouse button, or Alt left mouse button. This will move the cursor in the sequence display and all other cursors displayed that relate to the sequence.

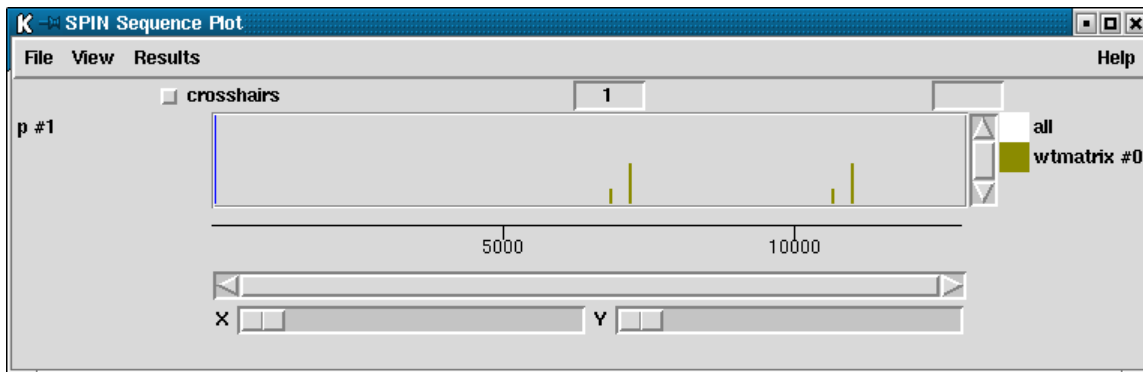
The graphical results can be zoomed and scrolled in both x and y directions. Zooming is achieved using the X and Y scale bars at the top left hand corner of the plot. The individual plots can be scrolled in y using the scroll bars attached to their right hand edge. The sequence can be scrolled using the scroll bar at the base of the plot.



The figure above shows the results of a search for restriction enzymes.

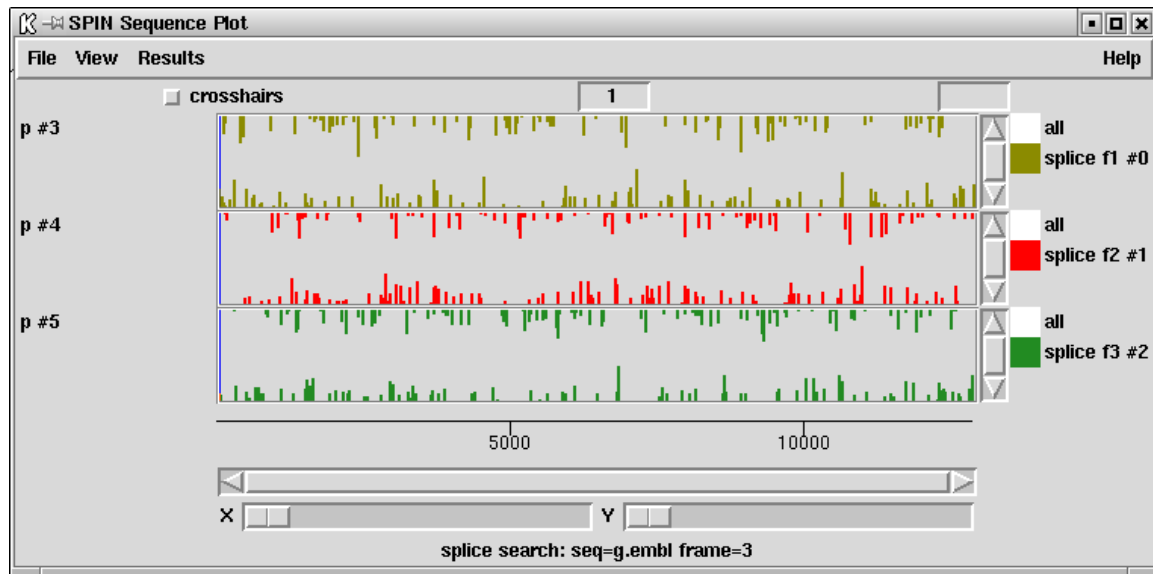


The figure above is a plot of the base composition of a sequence.

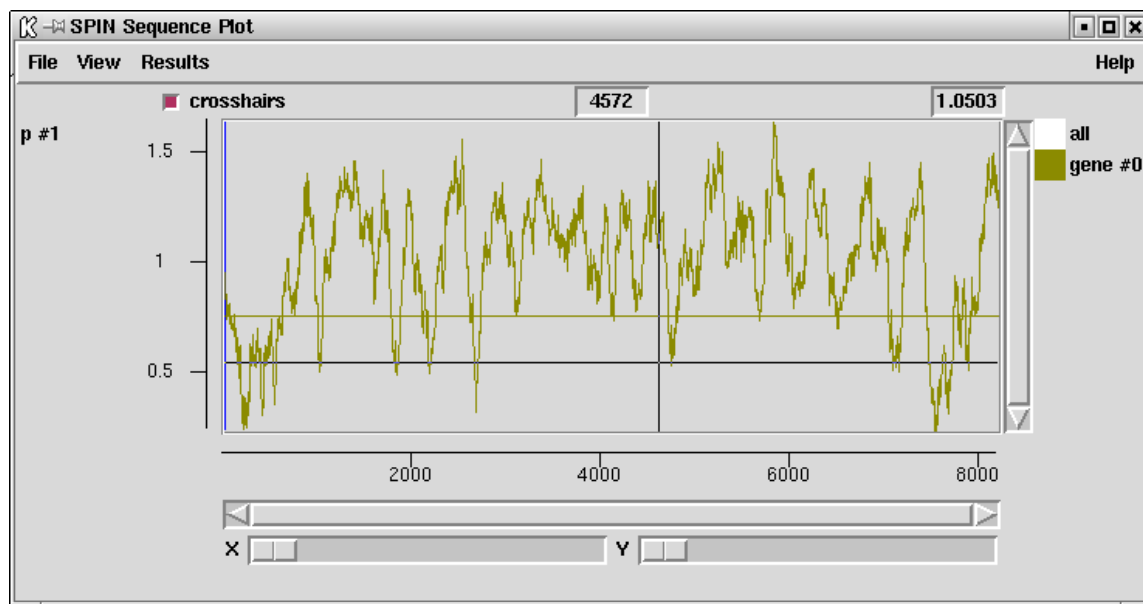




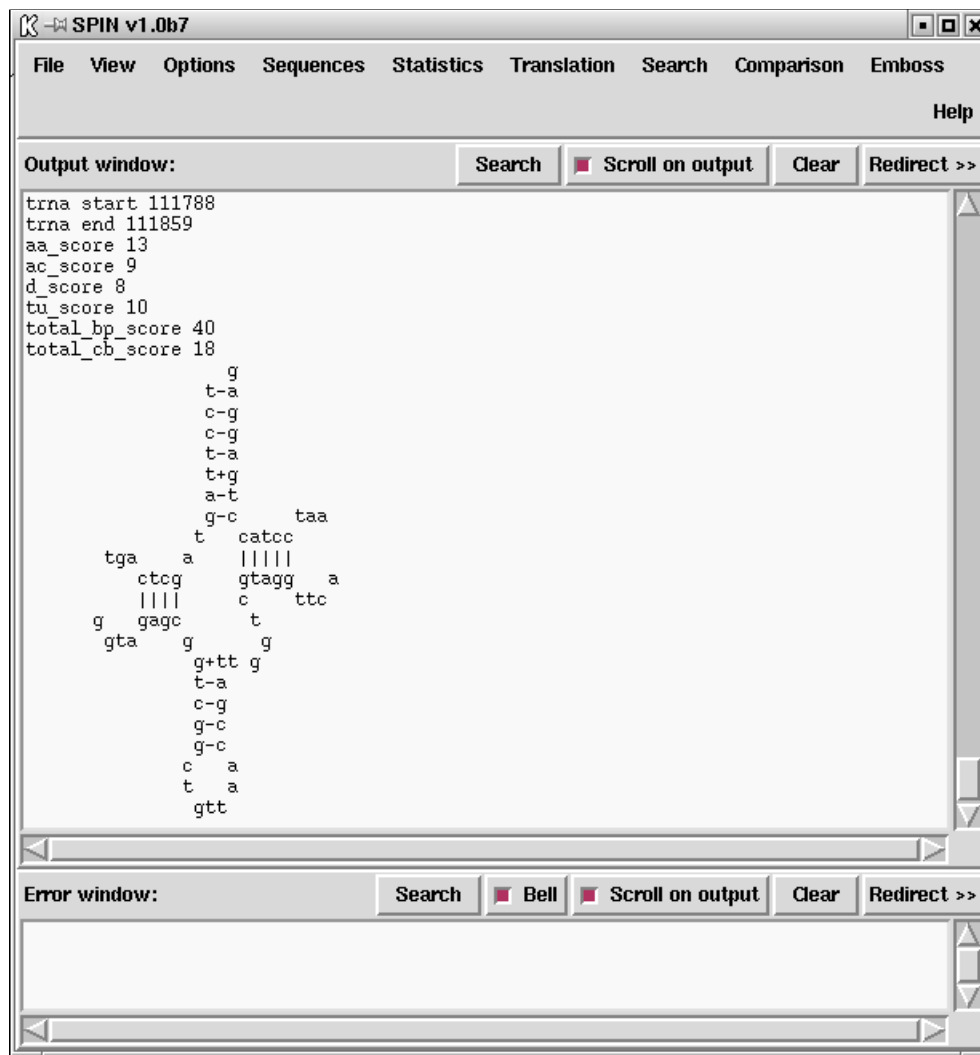
The figure above shows the way in which the results of weight matrix searches for motifs are plotted.



The figure about shows the way in which the results of searches for splice junctions are plotted. The donor and acceptor predictions are separated and a different colour is used for each reading frame.



The figure above shows a method for finding protein coding regions which does not distinguish reading frame or strand.

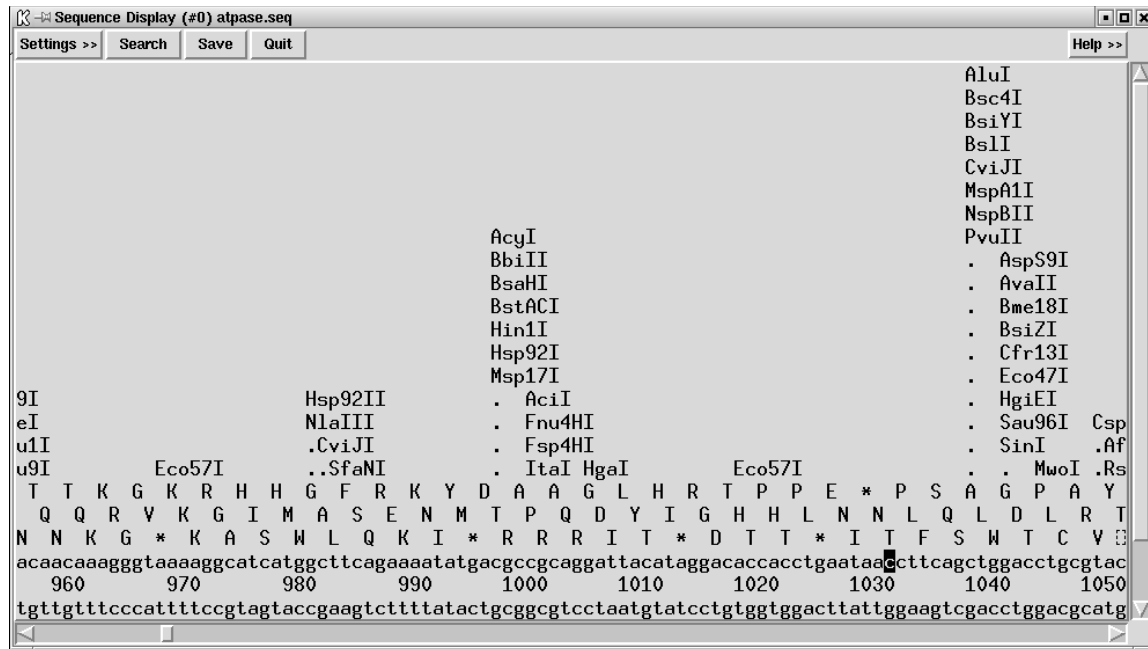


The figure above shows how results from the tRNA gene search function are displayed in the Output window.

### 6.1.3.2 Introduction to the Spin Sequence Display

Spin also has a sequence display window in which the user can view the sequence in textual form. This window allows the user to scroll along the sequence. Users can view one or both strands, can switch on displays of the encoded amino acids in up to six reading frames, can switch on a display of the restriction enzyme sites, and can perform other simple subsequence or string searches to locate features in the sequence. In the figure shown below the user has

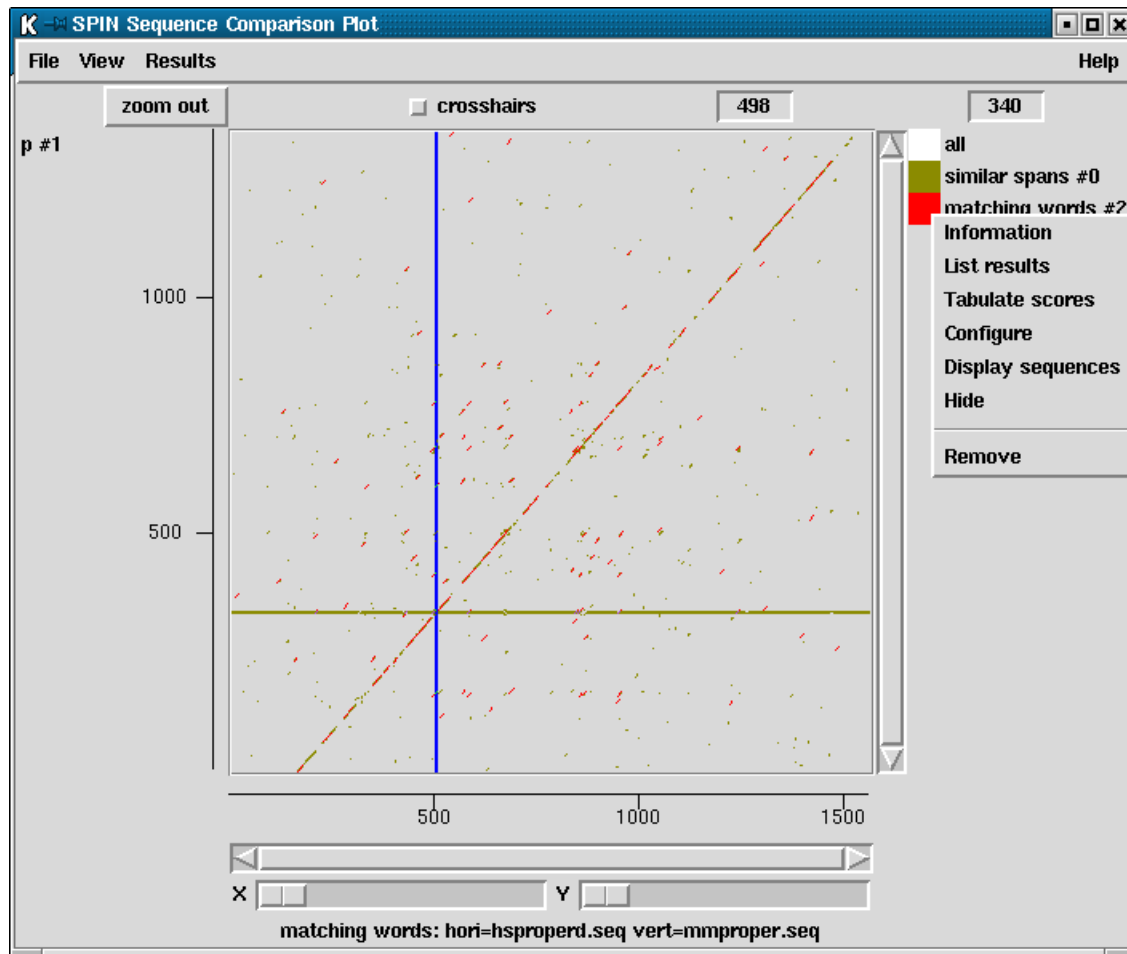
switched on a three phase translation on the top strand, double stranded sequence, and a restriction enzyme search.



The sequence cursor can be under the control of the graphics cursor i.e. the cursor in the sequence viewer can be moved by the user dragging the cursor in the graphics window. Similarly the cursor in the graphics plots can be moved by the sequence viewer cursor.

### 6.1.3.3 Introduction to the Spin Sequence Comparison Plot

All of the spin comparison functions display their results as points or lines in a two-dimensional plot called a "Spin Sequence Comparison Plot". Sets of matches from a single invocation of a comparison command are termed "a result". Each result is plotted using a single colour which can be configured via the results manager.



The diagram above shows the results of a "Find similar spans" search (olive), and a "Find matching words" (red).

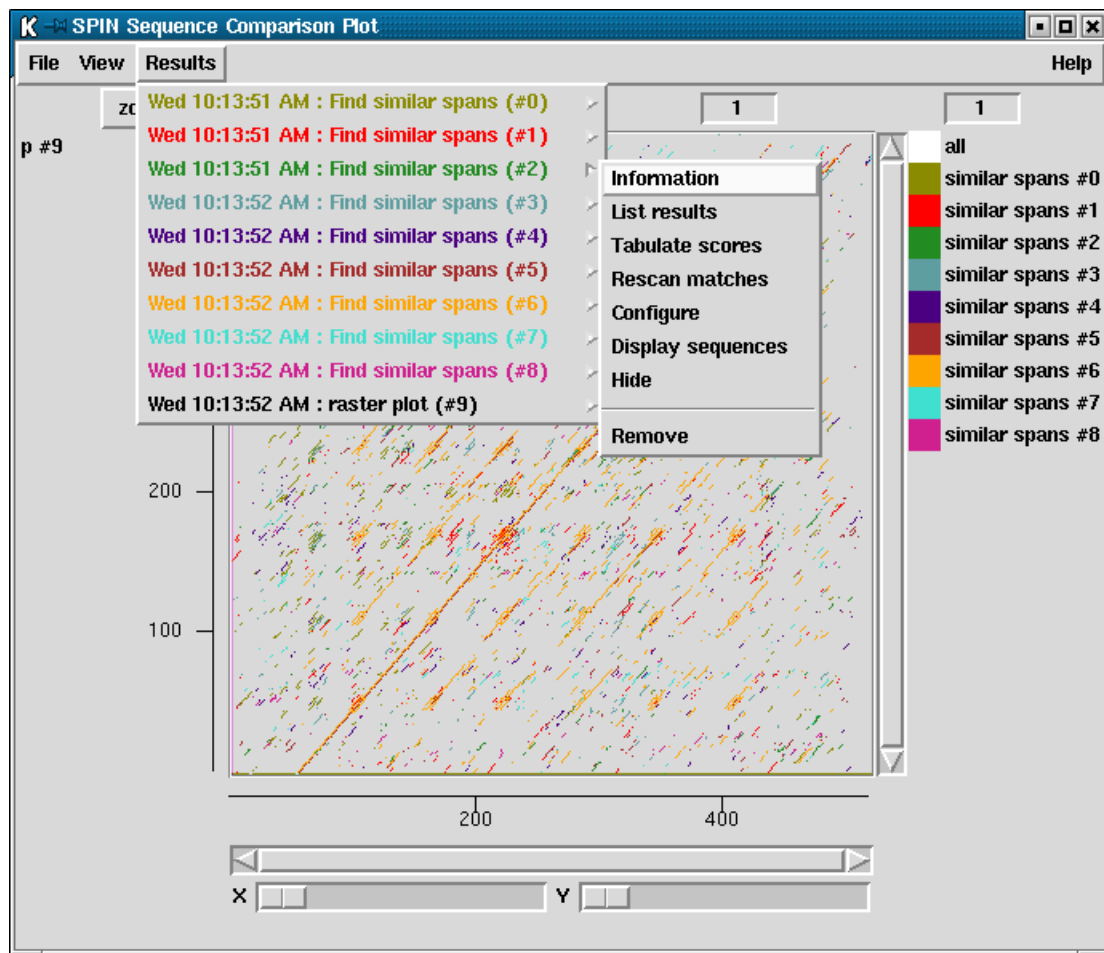
At the right hand side is a set of square boxes with the same colours as the dots drawn in the adjacent plot. These icon-like objects represent individual results and allow the user to operate on them. For example clicking with the right mouse button brings up the pop-up menu beneath the "matching words" result contains the results menu for this result. These icons can also be used to drag and drop the results to which they correspond. This is activated by pressing the middle mouse button, or Alt left mouse button, over the box and then moving the cursor over the Spin Sequence Comparison Plot to the new location or anywhere outside the Spin Sequence Comparison Plot.

Crosshairs can be turned on or off using the check button labelled "crosshairs". The x and y positions of the crosshairs are indicated in the two boxes to the right of the check box.

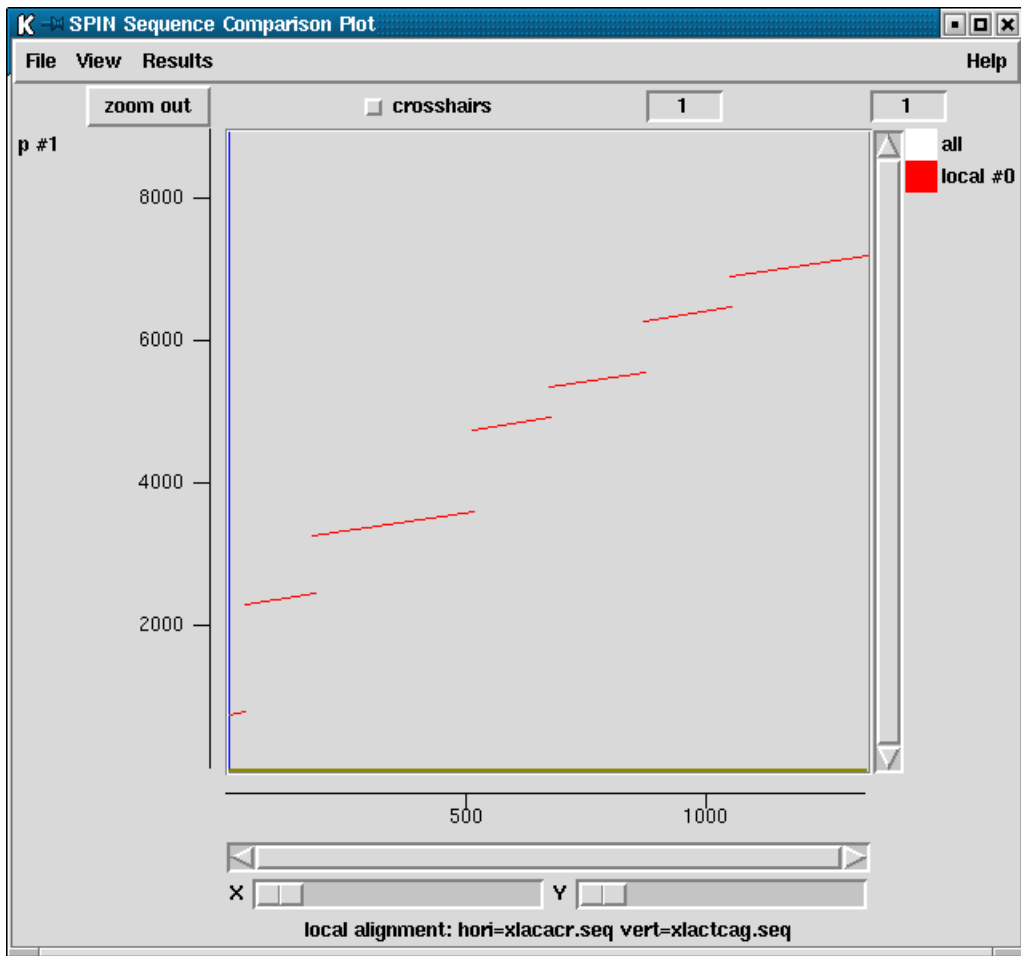
Each sequence displayed in a Spin Sequence Comparison Plot will have a cursor of a particular colour. In the picture above, the sequence on the horizontal axis has a vertical blue cursor whereas the sequence on the vertical axis has a horizontal olive green cursor. In general, the same sequence displayed in several Spin Sequence Comparison Plots will have a cursor of the same colour. The user can move a cursor by clicking and dragging with the middle mouse button, or with Alt left mouse button. This will move all other cursors displayed that relate to the sequence, whether they are in different Spin Sequence Comparison Plots or within the sequence display.

Plots can be enlarged either by resizing the window or zooming. Zooming is achieved by holding down the control key and right mouse button and dragging out a rectangle. This process can be repeated. The Back button will restore the plot to the previous magnification.

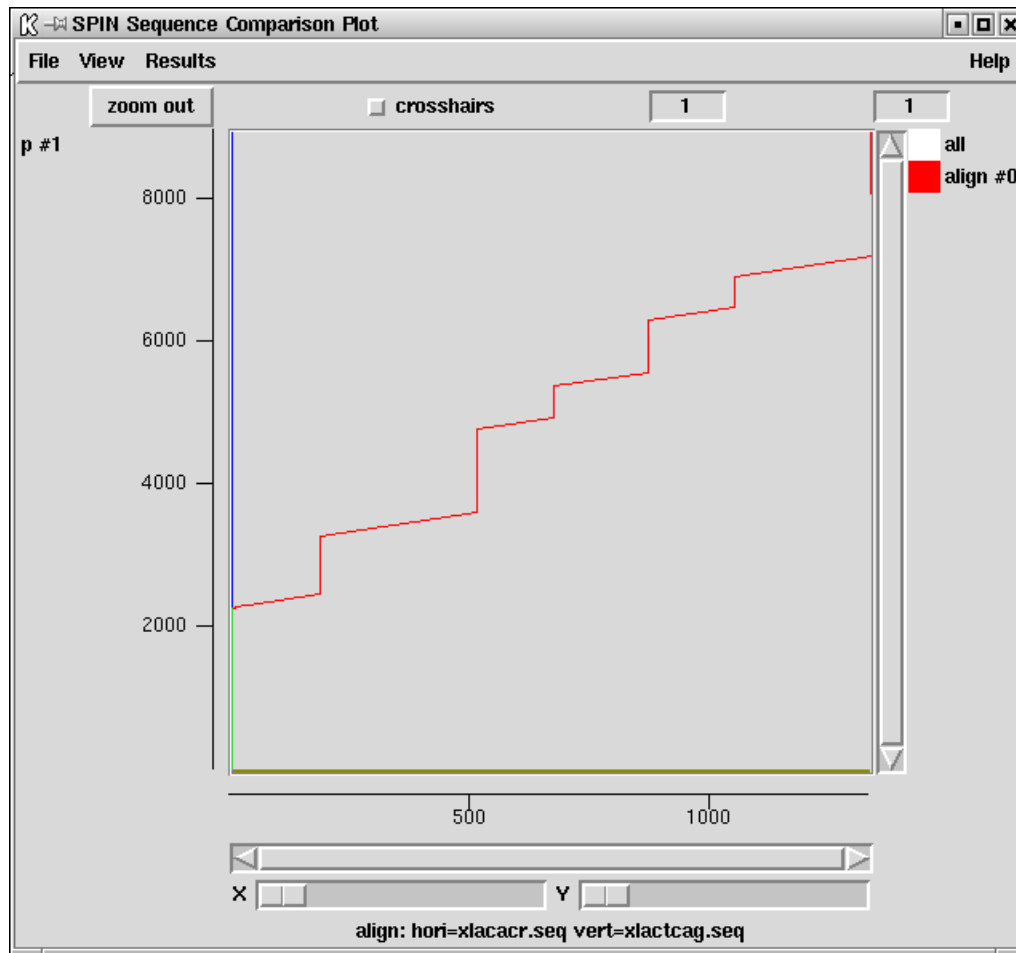
To illustrate further uses of the program we include some more screen dumps below.



The picture above shows the results after performing a "Find similar spans" comparison between the three reading frames of two DNA sequences, producing nine superimposed sets of results.



Local alignment searches join similar segments with lines. The above screen dump shows such an analysis in which genomic DNA containing 7 exons and is compared to its corresponding cDNA.



The above screendump shows a global alignment of the same pair of sequences.

#### 6.1.3.4 Introduction to the Spin Sequence Comparison Display

A sequence comparison display is associated with a single set of results and can be invoked by bringing up a pop up menu for the required result, either from the Results manager, the



Results menu in the Spin Sequence Comparison Plot, or the coloured square icon on the right of the plot.



The horizontal sequence is drawn above the vertical sequence and the central panel indicates characters which are identical. The buttons (< >) and (<< >>) scroll the sequences. Pressing the Lock button forces the sequences to scroll together. Movement of the sequences is also controlled by the scrollbars or by moving the corresponding cursor in the Spin Sequence Comparison Plot. The black cursors in the sequence display correspond to the position of the cursor in the Spin Sequence Comparison Plot. The sequences can be made to 'jump' to the nearest match in those results by pressing the "Nearest match" or "Nearest dot" buttons.

