Introduction to Comparing Large Sequence Sets

The growing flood of genomic sequences from microbes to mice requires tools capable of efficiently handling very large sequences. Comparisons of whole genomes can yield important insights into the evolution of genome structure, such as the role of inversions in bacterial evolution (Eisen et al., 2000) and the identification of large-scale duplications in the human genome (Venter et al., 2001). This chapter describes two tools for aligning whole genome sequences: MUMmer (*UNIT 10.3*) and PipMaker (*UNIT 10.2*). These tools differ in both the underlying algorithms used, and in the interface they present to the user.

MUMmer (*UNIT 10.3*) is a suite of Unix command-line tools for aligning genomes. At the core of these tools is the concept of a maximal unique match (MUM) between two sequences. The algorithms will find all subsequences longer than a specified length k that are identical in two sequences. By setting the value of k to be some value above that expected by chance alone, the program will ignore the many short subsequences that represent random matches. The programs are very fast (whole bacterial genomes can be aligned in less than a minute on a standard desktop computer). The MUMmer package offers a range of tools for aligning nucleotide and protein sequences, and the user has considerable scope for changing parameters that affect the analysis. The program does not directly generate graphical output, but the output files can be used together with other software to generate dot plots to facilitate interpretation of the results.

In contrast to MUMmer, PipMaker (*UNIT 10.2*) has a World Wide Web interface and provides a rich graphical output. At the heart of PipMaker is an alignment algorithm that identifies regions with high percent identity. These regions are displayed in percent identity plots, or "pips." In addition to the standard dot plot, PipMaker can produce more detailed plots of pips that enable the user to clearly identify regions of sequence similarity. A great strength of the system is the ability to add annotations, such as the locality of introns, exons, conserved regions, and other features of interest. This enables the user to align a new genomic sequence against a previously known and well-annotated genome, and hence guide the transfer of the annotation from one genome to the other. Being a Web service provided from a single site, PipMaker imposes limits on the computational time taken by an analysis, so as not to overburden the server. However, if that time limit becomes a problem, the core tools can be downloaded and run on the user's own computer.

The tools described in this chapter are applicable to a number of different tasks in comparative genomics, beyond simple alignment. Both MUMmer and PipMaker can work with contigs, so that unfinished or "working draft" genomes can be compared with complete genomes. In addition, different assemblies of the sequences resulting from whole-genome shotgun sequencing can be compared to assist reconstructing the final genome sequence. Whole genome alignments can also be used to locate regions of synteny, in which gene order and orientation (and presumably function) are conserved between two organisms.

Contributed by Roderic D.M. Page

Current Protocols in Bioinformatics (2003) 10.1.1-10.1.2

LITERATURE CITED

Eisen, J.A., Heidelberg, J.F., White, O., and Salzberg, S.L. 2000. Evidence for symmetric chromosomal inversions around the replication origin in bacteria. *Genome Biol.* 1:1101-1109.

Venter, J.G., Adams, M.D., Myers, E.W., Li, P.W., Mural, R.J., Sutton, G.G., Smith, H.O., Yandell, M., Evans, C.A., Holt, R.A., et al. 2001. The sequence of the human genome. *Science* 291:1304-1351.

Contributed by Roderic D.M. Page
University of Glasgow
Glasgow, Scotland

PipMaker: A World Wide Web Server for Genomic Sequence Alignments

PipMaker (*http://bio.cse.psu.edu/*) is a World Wide Web site used to compare two long genomic sequences and identify conserved segments between them (Schwartz et al., 2000). This unit describes the use of the PipMaker server and gives an explanation of the resulting output files.

PipMaker has several strong features for use in comparisons of genomic DNA sequences.

1. Very long genomic DNA sequences can be aligned. The underlying alignment software, Blastz, was developed with effectively no limit on the file sizes to be compared. The PipMaker server has a limit based on the time required for alignment, but, typically, sequences of up to 2 Mb will be aligned before the time limit is imposed. Longer sequences can be aligned upon request.
2. The alignments and output are generated quickly. The program Blastz will align two sequences of 1 Mb each in less than 1 min, so in almost all cases users receive output very quickly.
3. The alignments are highly sensitive and specific.
4. The output is easy to interpret, incorporates extensive annotation (e.g., for genes and repeats), and is adjustable to users' preferences.
5. Although PipMaker is frequently used to compare two finished sequences, the server supports analysis of unfinished or "working draft" sequences by permitting the second sequence to be in unoriented and unordered contigs (contiguous segments). The server also provides an ordered and oriented version of the sequence based on the alignment order of contigs in the comparison.
6. Pipmaker is appropriate for comparing genomic sequences from any two related species. However, the type of information that can be inferred depends on the level of conservation and the time and divergence rate since the separation of the species (see Background Information).

The Basic Protocol describes the use of the PipMaker server to align two genomic sequences. Support Protocols 1 to 4 describe the preparation of the various files and optional underlays that the user will supply to the PipMaker program in the Basic Protocol. Support Protocol 5 describes installation of the stand-alone Blastz alignment program.

STRATEGIC PLANNING

In order to use PipMaker effectively, one needs two genomic sequences whose evolutionary distance (i.e., degree of similarity) is appropriate for the features that one is trying to distinguish. These files must be in FASTA format (*APPENDIX 1B*), and the first one must be continuous, though the second can consist of multiple, unordered contigs.

A variety of annotation files can also be supplied. For example, it is strongly recommended that a file be submitted specifying the locations of interspersed repeats in the first sequence. This will not only eliminate many spurious matches from the results, but will also make it possible to align much longer sequences before the PipMaker server times out. Other types of files, describing features such as gene and exon locations, color underlays, and hyperlinks to other resources, are optional. All of these files and their

Contributed by Laura Elnitski, Cathy Riemer, Scott Schwartz, Ross Hardison, and Webb Miller
Current Protocols in Bioinformatics (2003) 10.2.1-10.2.23

formats are discussed in detail later in this unit. Note that PipMaker will compute and display CpG islands in the first sequence automatically.

In addition to preparing one's input files, one needs to decide what type(s) of output to request, as well as any other options that one wishes to specify, such as whether to search for matches in both strands or just the forward strand, whether to show all matches or use the chaining or single coverage options, whether to request higher sensitivity at the expense of a shorter time limit, and whether one wants PipMaker to attempt to order and orient the contigs in the second sequence. Again, all of these options will be described in more detail later.

As its name implies, PipMaker's default output is a pip (percent identity plot). However other forms of output are also available, including a dot plot, a concise listing of aligning regions, a traditional nucleotide-level text alignment, an analysis showing where one's exons align in the second sequence (complete with putative start/stop codons and splice junctions), and a raw Blastz output file for use by other programs. The dot plot, in particular, is useful for observing genomic rearrangements between species, and can also help to identify assembly problems. In the dot plot, alignments of individual contigs are separated by dotted lines, and can optionally be arranged in the order and orientation matching the first sequence. The graphical output files (pip and dot plot) are available in either PostScript or PDF format, and with PDF one has the additional option of embedding the contig names as pseudo-hyperlinks, so they will be displayed by Acrobat Reader when the mouse is pointed at the corresponding contig.

Note that all output will be sent by E-mail to the address specified by the user; it is the user's responsibility to ensure that this E-mail account has a sufficiently high message-size threshold and that the mailbox is not too full to hold it. Output messages of 5 Mb or more are not uncommon, especially if the traditional text alignments are requested, which are very verbose and generally not recommended for large regions.

While the online PipMaker server is recommended in most cases, some users may occasionally find that they need more flexibility than the server offers. For example, if the submitted sequences are very long, the server might time out, even if all of the repeats have been masked. There might also be a specialized need to adjust the alignment scoring parameters. In cases like these, one can download and run the Blastz alignment program (which is the foundation of PipMaker) directly onto one's own computer. However, Blastz only produces raw alignment output, so other tools (discussed later) are needed to display and analyze it. At the time of this writing, the software for the rest of PipMaker (drawing pips, etc.) is not ready for public distribution.

BASIC PROTOCOL

SUBMITTING SEQUENCES TO PipMaker

A flowchart for the use of PipMaker is shown in Figure 10.2.1. Genomic DNA sequences of interest are submitted to the PipMaker Web server for alignment. Along with the sequences, a user should submit a file containing the types and locations of repeats within the first sequence. The local-alignment program Blastz computes the alignment using a scoring scheme for matches, mismatches and gaps. Output files from the server are high-resolution displays of the resulting alignments, such as percent identity plots (pips; Fig. 10.2.2) and dot plots (Fig. 10.2.3), or various text files, such as nucleotide level alignments. Positions along the horizontal axis of the pip can be annotated with features such as exons and repetitive elements, and colors can be used to clarify and enhance the display. In addition to computing sequence conservation between genomic sequences, PipMaker comparisons are helpful for identifying putative exons that lack annotation, exons that are not conserved between species, and conserved noncoding regions that may be functional.

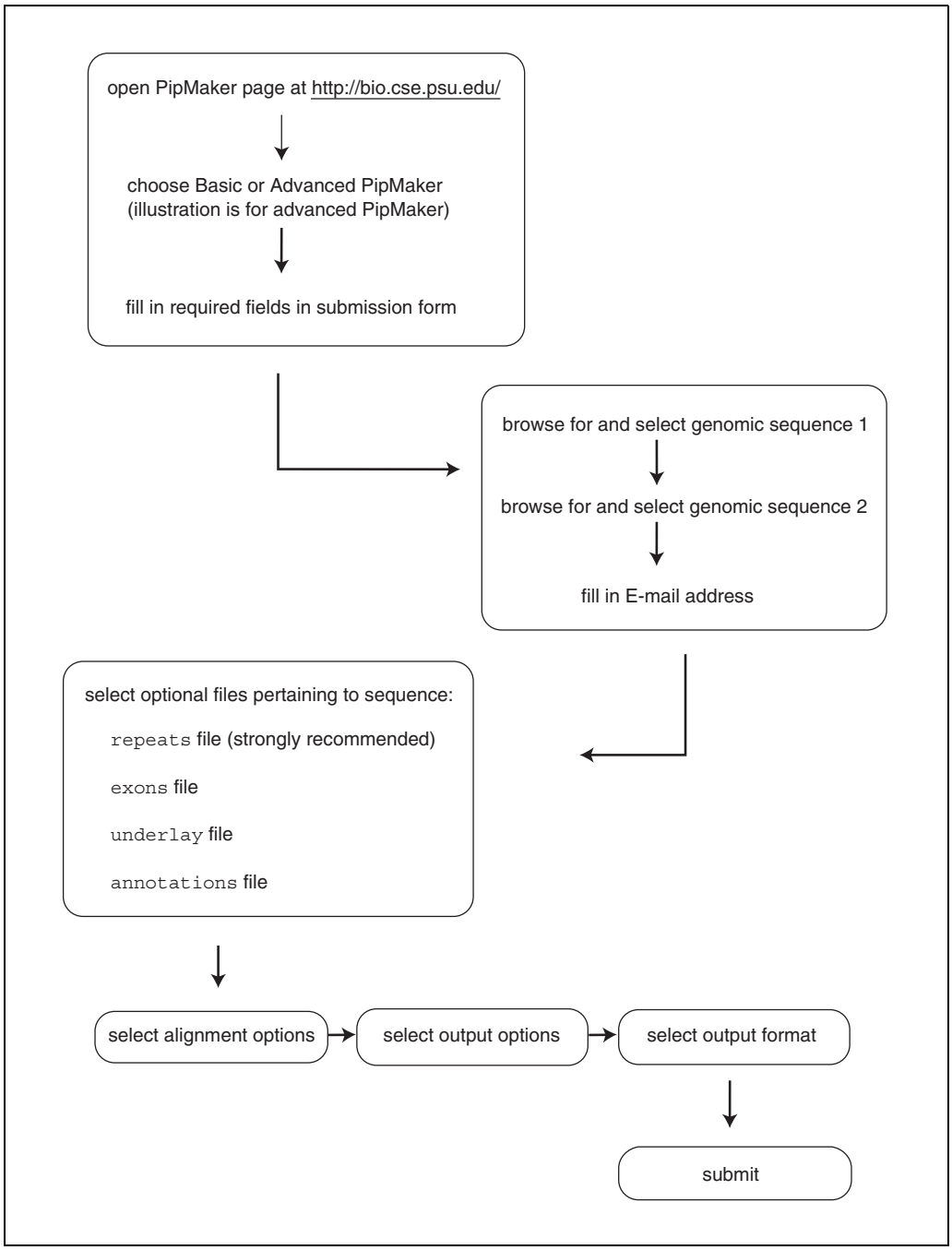


Figure 10.2.1 Flowchart outlining the steps for using PipMaker.

Necessary Resources

Hardware

PipMaker can be accessed and used by any computer with a World Wide Web browser and E-mail access.

Software

PipMaker is accessible via a Web interface at *http://bio.cse.psu.edu/*. All output files will be returned to the user via E-mail. The E-mail account and software must be capable of handling large messages. Viewing the output from PipMaker requires a PDF viewer to display the pip or dot plot, such as Aladdin

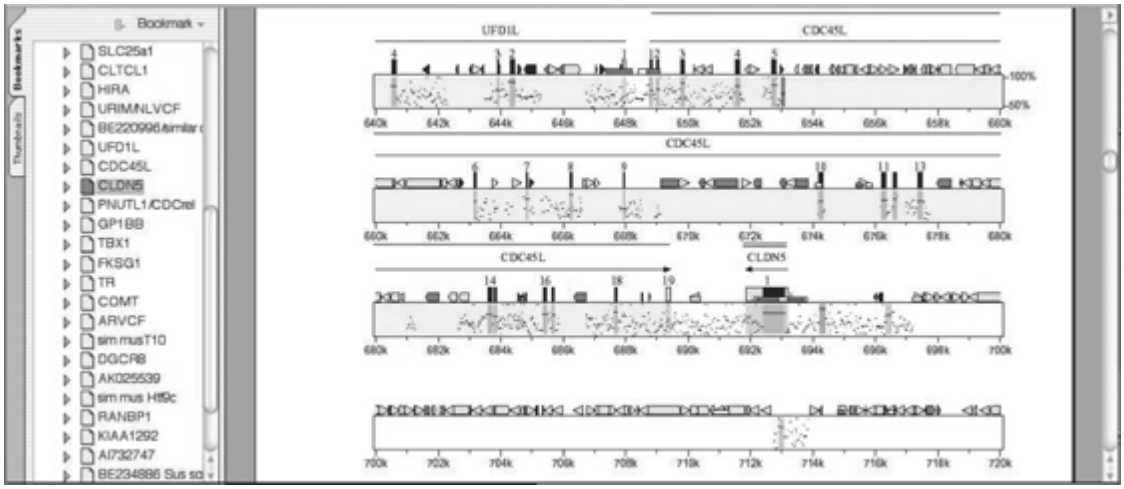


Figure 10.2.2 A pip showing human chromosome 22 in the region associated with velocardiofacial syndrome aligned with the orthologous region in mouse. A 60-kb segment surrounding the *CDC45L* and *CLDN5* genes illustrates the alignments and annotations on a pip. Each aligning segment is displayed as a series of horizontal lines whose positions correspond to the first sequence used in the alignment. The aligning segments are drawn according to their percent identity, which is shown on the vertical axis from 50% to 100%. A number of optional annotation files allow a user to augment the information content of the display. For instance, the names of genes and their direction of transcription is part of the `exons` file format. An `underlay` file specifies what color to draw genomic features such as exons (e.g., light blue), introns (e.g., light yellow), UTRs (e.g., light orange), and conserved noncoding regions (e.g., shades of red). An `annotations` file provides colored, horizontal lines above the alignment that are hyperlinks in the PDF file and provide direct links to relevant Internet sites—e.g., appropriate PubMed citation(s) for the gene (red), the LocusLink entry for the gene (blue), or a protein sequence from GenBank (green). The bookmarks along the left side provide links to compiled information about the various genes and other annotations describing the comparative analysis of the sequences. The bookmarks represent a much larger region than that shown in the image. *This black and white facsimile of the figure is intended only as a placeholder; for full-color version of figure, go to http://www.currentprotocols.com/colorfigures.*

GhostScript or Adobe Acrobat Reader. These are available for free download at *http://www.cs.wisc.edu/~ghost/* and *http://www.adobe.com/*, respectively. At the present time, Acrobat Reader has better support for hyperlinks in PDF files, which are an option in PipMaker. PipMaker can optionally generate a PostScript version of the output files. This feature is useful for importing the plot into a graphics program in preparation for publication.

Files

The following file types are used:

Sequences: The PipMaker server accepts two DNA sequences in FASTA format (*APPENDIX 1B*) only. These sequence files must be in plain text format, consisting of A, C, G, T, N, and X, typically uppercase. Line length should be within ~70 characters. The first sequence should be in one contiguous piece, while the second sequence can be in unordered, unoriented contigs.

`Repeats` *file* (see Support Protocol 1)

`Exon` *file* (optional; see Support Protocol 2)

`Underlay` *file* (optional; see Support Protocol 3)

`Annotation` *file* (optional; see Support Protocol 4)

1. In a Web browser, open the server page *http://bio.cse.psu.edu/*, click on the (Multi)PipMaker link, and then scroll down the page that appears and click on the link to the Advanced PipMaker page.

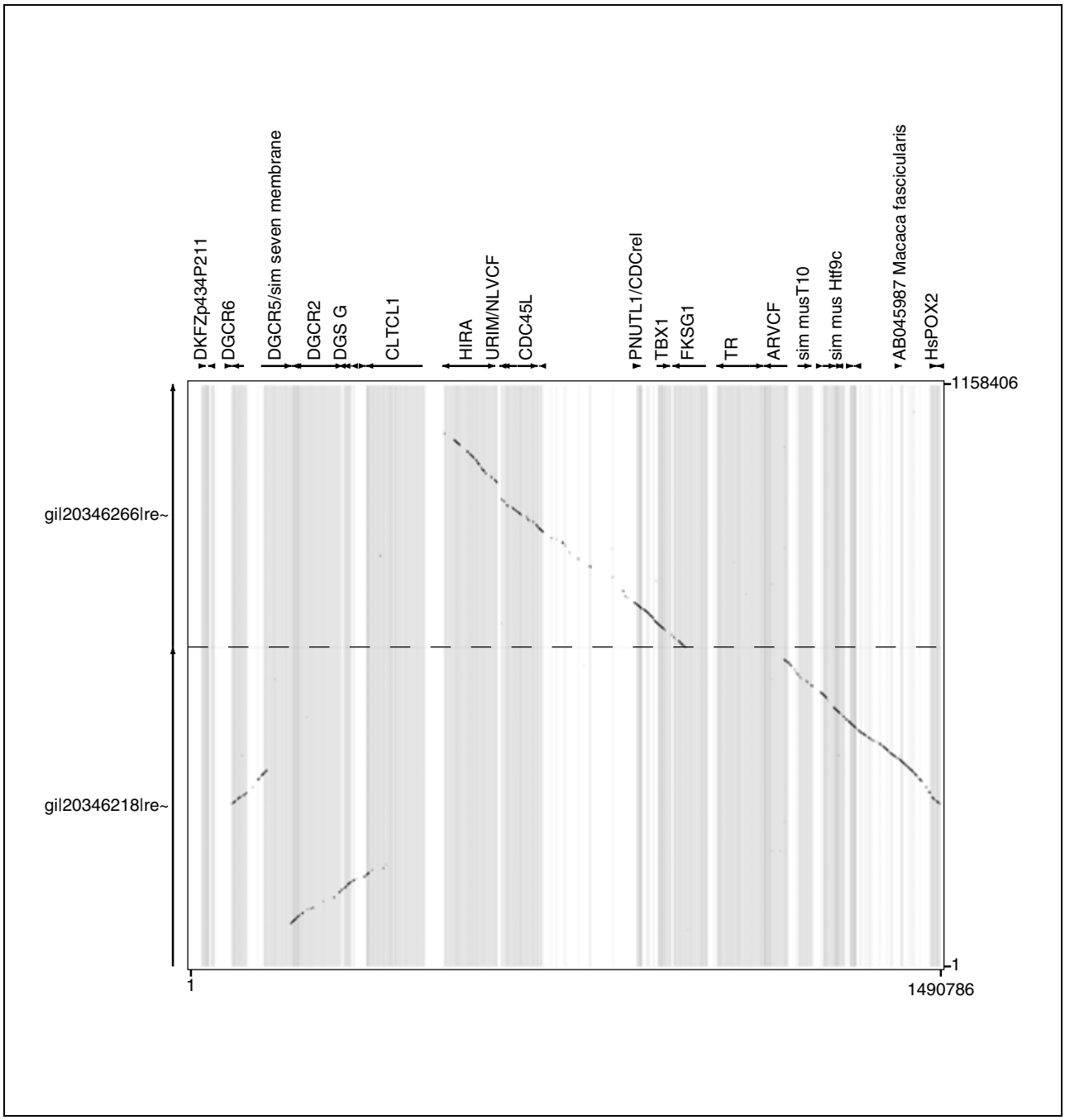


Figure 10.2.3 A dot plot of the 1.5-Mb region from human chromosome 22 associated with velocardiofacial syndrome, aligned to the orthologous sequences from mouse. Annotations used in the alignment are displayed along the horizontal axis as gene names with the direction of transcription. The mouse sequence is represented by two contigs that are labeled along the vertical axis of the plot (gi: 20346266 and 20346218). The Order and Orient option attempts to arrange the mouse sequences in the same relative order as the human and indicates the presence of rearrangements in the mouse sequence relative to the human sequence. The dot plot uses the same `underlay` file as the pip to color the image. Note that the gene names used in this example are not all recognized by the HUGO nomenclature committee, and serve as illustrations only.

Each line on the data entry page indicates the type of data that is required, e.g., the box entitled "First sequence (FASTA format)...or filename (file must be plain text only)" reminds the user of the correct file format. Links to the Basic Pipmaker page are active, but note that most of the following options are only available on the Advanced PipMaker page. The authors have had reports of bugs in some versions of Microsoft Internet Explorer. If trouble is encountered, try using Netscape or some other Web browser.

2. In the PipMaker submission page which then appears, enter the first genomic sequence for the alignment.

   The DNA sequence can come from many sources, including the Human Genome Browser (UNIT 1.4), GenBank (UNIT 1.3), and Ensembl (see Internet Resources).

3. Enter the second sequence.

4. Supply a correct E-mail address to obtain results of the alignment.

   If a user fails to receive output from PipMaker, the most likely cause is an error in the E-mail address (see Critical Parameters and Troubleshooting) .

5. In the text box labeled First Sequence Mask, enter the file of repeats for the first sequence generated using the RepeatMasker Web server (*http://ftp.genome.washington.edu/cgi-bin/RepeatMasker*) or enter an alternative repeats file (see Support Protocol 1).

6. Choose the alignment style.

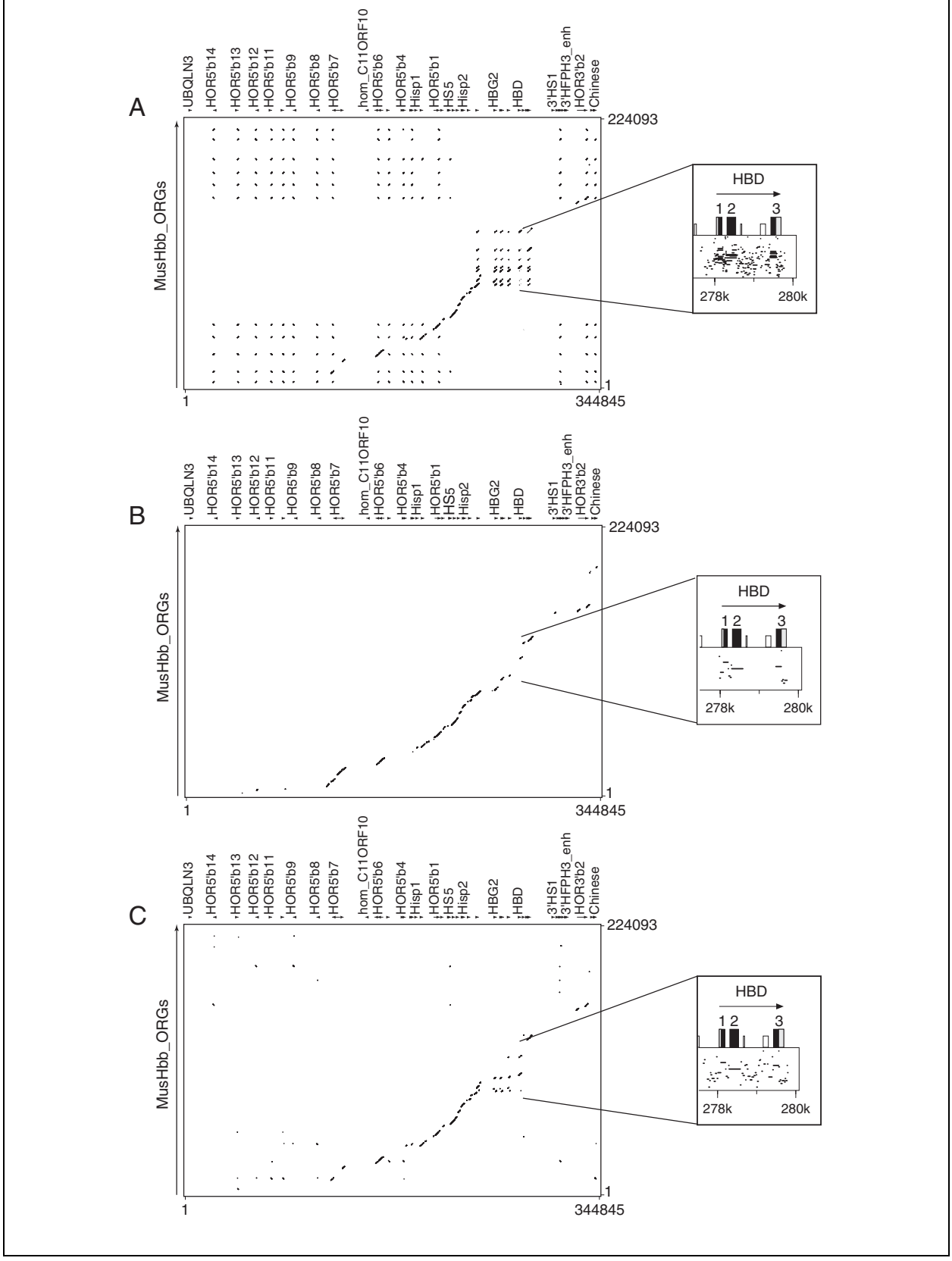
   a. Select the radio button for either Search One Strand or Search Both Strands at the bottom of the PipMaker submission page.

      When searching one strand, the alignment program can report only regions of similarity between the first sequence and the second sequence in its given orientation. The default of searching both strands aligns the first sequence with both directions of the second sequence.

   b. Also at the bottom of the PipMaker submission page, select the radio button for either Show All Matches, Chaining, or Single Coverage.

      The default setting of Show All Matches allows a region in the first sequence to align with several regions of the second sequence (Fig. 10.2.4A). In general, these instances occur because of duplications of a genomic region or incomplete masking of interspersed repeats or low-complexity regions. They appear as multiple lines in the pip occupying the same position in the sequence. In the example alignment between the human and mouse globin

Figure 10.2.4 *(see next page)* Illustration of the PipMaker options (**A**) Show All Matches, (**B**) Chaining, and (**C**) Single Coverage. The human β-globin gene locus contains a family of duplicated genes which, when aligned to the orthologous region in mouse, shows matches to multiple family members in the mouse globin locus. The option to Show All Matches reveals extensive sequence similarity between globin gene clusters in human and mouse sequences (panel A, from left to right: dot plot of the extended genomic region and the pip of the human δ-globin gene, *HBD)*. In addition, a cluster of related olfactory receptor genes surrounds the globin locus in both species and creates the checkerboard pattern. The Chaining option reduces the amount of aligning sequence by showing alignments of sequences that appear in the same relative order between the two species (panel B, leftmost box). For this reason, most of the ORGs disappear. One aligning segment is identified for each human globin gene and multiple hits are removed from the pip (panel B, rightmost box). The Single Coverage option identifies the highest-scoring alignments and allows any position in the first sequence to align only once to the second sequence. Therefore, no alignments are in the same vertical space, although they may appear to be very close if the display is of insufficient resolution. In the globin locus, several genes are most similar to the same sequence in the mouse locus (see panel C, leftmost and middle boxes) and appear on the same horizontal line. The pip (panel C, rightmost box) shows more alignments than with the Chaining option because the best match is not restricted to being in the same relative order along the two sequences.


A
B
C
UBQLN3
HOR5'b14
HOR5'b13
HOR5'b12
HOR5'b11
HOR5'b9
HOR5'b8
HOR5'b7
hom_C11ORF10
HOR5'b6
HOR5'b4
Hisp1
HOR5'b1
HS5
Hisp2
HBG2
HBD
3'HS1
3'HFPH3_enh
HOR3'b2
Chinese
MusHbb_ORGs
224093
1
344845
HBD
1 2
3
278k
280k

loci illustrated in Figure 10.2.4A, duplicate matches occur between the related globin family members in each locus (Bulger et al., 1999) as well as in the olfactory receptor genes, ORGs, that surround each locus (Bulger et al., 2000). PipMaker provides two options for eliminating such duplicate matches, as follows.

The Chaining option identifies and plots only those alignments that appear in the same relative order in the first and second sequences. This option may overlook duplicated regions and should only be used if the genomic structure of the two sequences is conserved. The use of Chaining invokes the alignment program with lower thresholds (i.e., higher sensitivity). An alignment that illustrates this option is shown in Figure 10.2.4B. Specifying the Chaining option removes multiple matches in the globin locus and most of the matches in the ORGs.

An alternative method for avoiding duplicate matches is the Single Coverage option. This alignment method returns only the highest-scoring set of alignments, restricting each region of the first sequence to be aligned only once. Single coverage does not require that matching regions be in identical order in the two sequences. Figure 10.2.4C illustrates the case where Single Coverage aligns the globin genes to the highest scoring sequence from the mouse globin locus. Alignments on the same horizontal row indicate matches to the identical region in mouse. Single coverage prevents the same human sequence from aligning to multiple regions in the mouse, which would produce alignments on the same vertical column.

Chaining is preferable to the Single Coverage option in cases where (1) the second sequence is contiguous, (2) the comparison is with just a single strand of the second sequence, and (3) the order of conserved regions is identical in the two sequences, since under conditions 1 to 3 the results from the Chaining option will be more biologically meaningful. However, if condition 3 does not hold, the Chaining option may give inferior results. A benefit of the Single Coverage option is that it guarantees single coverage even if the second sequence is compared in both orientations, or if it is fragmented (see below). In such cases, the Chaining option is applied separately each time the first sequence is compared with an orientation or fragment of the second sequence, so multiple coverage can result.

c. If appropriate, check the box for High Sensitivity and Low Time Limit.

The default settings of PipMaker are tuned to perform well when comparing two mammalian sequences. An option for High Sensitivity is provided for comparisons of sequence pairs separated by a greater evolutionary distance, such as human-Fugu. Comparison of two more similar sequences (e.g., human-mouse) at this setting may cause the alignment program to run for too long a period of time without finding a significant number of additional informative alignments. Excessively long runs will be terminated by the server, so use of High Sensitivity for human-mouse comparisons should be avoided in most cases. The time allowed before termination still permits a successful human-Fugu alignment over the length of two BACs using the High Sensitivity setting.

7. In the fields at the very bottom of the PipMaker submission page, select the desired form of output.

The user selects output files from the following list: PIP, a (one-page) dot-plot view of the alignments, a condensed (Concise Textual) form of the alignment, the Traditional Textual form of the alignment, an Analysis of Exons, Raw Blastz Output, and Order and Orient Contigs (with output files predicting the order and orientation of the contigs in the second sequence, assuming the second sequence file comprises unordered contigs). For descriptions of all of these forms of output, see Guidelines for Understanding Results.

8. Enter an optional title.

If no title is specified, the title is taken from the first line of the `exons` *file.*

9. Click the Submit button.

A page will appear in the browser that indicates a successful submission or the detection of an error.

See Guidelines for Understanding Results for a discussion on interpreting the output.

GENERATING A `REPEATS` FILE FOR USE WITH PipMaker

SUPPORT PROTOCOL 1

`Repeats` files document the position and type of repeats in the first sequence used in an alignment. PipMaker will mask the positions of repeats with lowercase letters, allowing the alignment program to skip these positions during the first step of the alignment. In the subsequent step after the primary seeds for the alignment have been established, these "soft-masked" regions are considered in the alignment process. One method for generating a `repeats` file is the program RepeatMasker (A.F.A. Smit and P. Green, unpub. observ.; *http://ftp.genome.washington.edu/cgi-bin/RepeatMasker/*). PipMaker accepts the documentation format of the RepeatMasker output that describes repeat families and their positions (but do not try to submit the masked sequence itself as a `repeats` file). An example line is illustrated in Figure 10.2.5.

A simpler format for a `repeats` file is also allowed, as shown in Figure 10.2.6. In this case, the first line of the file contains a specific tag to distinguish it from RepeatMasker output, and each subsequent line specifies the start, end, direction, and type of a particular feature.

The allowed repeat families are: *Alu*, B1, B2, SINE, LINE1, LINE2, MIR, LTR, DNA, RNA, Simple, and Other. Simple repeats do not require a direction.

```
413 5.6 0.0 0.0 HUMAN 1 54 (92195) C Alu SINE/Alu (238) 62 9 SINE/Alu (238) 62  9
```

Figure 10.2.5 Example of documentation output from Repeatmasker.

```
%:repeats
1081 1364 Right Alu
1365 1405 Simple
```

Figure 10.2.6 Alternate format for `repeats` file.

GENERATING AN `EXONS` FILE FOR USE WITH PipMaker

SUPPORT PROTOCOL 2

The `exons` file format is an optional text file providing the positions of transcriptional units in the first sequence for the PipMaker drawing program. This information can be generated from a number of resources, including GenBank entries (*http://www.ncbi.nlm.nih.gov/*) and Genscan (Burge and Karlin, 1997; *http://genes.mit.edu/GENSCAN.html*) predictions. The first line for each gene indicates the gene name, directionality (< or >), and start and end positions. It is also permissible to begin the first line with |, in which case the symbol indicating the gene's position in the first sequence is drawn without an arrowhead. Subsequent lines specifying the start and end positions of each exon are used by PipMaker to draw the locations of exons within each gene. An optional second line beginning with a + character can indicate the first and last nucleotides of the translated region (including the initiation codon, Met, and the stop codon). Blank lines are ignored. Exons must be specified in order of increasing address even if the gene is on the reverse strand (<). The first line of the file may contain an optional title for the pip. An `exons`

```
> 648772 689427 CDC45L
+ 648796 687735
648772 648846
648985 649044
649780 649872
651517 651654
652689 652832
663154 663209
664808 664856
666213 666274
667928 667978
674189 674308
676213 676344
676593 676691
677357 677518
683576 683714

< 691854 693201 CLDN5
+ 692426 693081
691855 693201
```

Figure 10.2.7 An `exons` file containing two genes from human chromosome 22 that are transcribed in opposite orientations.

file containing two genes from human chromosome 22 that are transcribed in opposite orientations is shown as an example in Figure 10.2.7.

The exon numbers supplied by PipMaker can be overridden by specifying a number (or a name) at the end of each exon line. These numbers will be drawn above the gene along with the other annotations (e.g., alternatively spliced exons can be numbered 1a and 1b, or if a pip only displays the 3′ exons of a gene and does not start at exon 1).

SUPPORT PROTOCOL 3

GENERATING COLOR UNDERLAYS FOR USE WITH PipMaker

Regions of interest in the first sequence, such as the locations of exons or regulatory elements, can be shaded in the background to help distinguish them from other genomic features. To use this feature of Advanced PipMaker, upload an `underlay` file that describes features in the first sequence of the alignment. The `underlay` file contains a list of coordinate pairs, each with a color label or a textual description that is associated with a color label. For instance the region of the *CLDN5* gene has a color underlay specified as shown in Figure 10.2.8 where LightYellow indicates an intron, LightOrange the UTRs, and LightBlue the coding region. All colors must start with a capital letter and must be selected from a restricted list. Current color choices are: `Black`, `White`, `Blue`, `Cyan`, `Gray`, `Green`, `Orange`, `Pink`, `Purple`, `Red`, and `Yellow` (and `Light` and `Dark` versions of all but `White` or `Black`). The list is maintained at the PipMaker Web

```
691854 693201 LightYellow
691854 692425 LightOrange
692426 693081 LightBlue
693082 693201 LightOrange
694200 694350 Red
```

Figure 10.2.8 An example of an `underlay` file which refers to Figure 10.2.7.

```
LightYellow   Intron
LightOrange   UTR
LightBlue     Coding_region
Red           Strongly_conserved

691854 693201 Intron
691854 692425 UTR
692426 693081 Coding_region
693082 693201 UTR
694200 694350 Strongly_conserved
```

Figure 10.2.9 Alternate format for an `underlay` file which refers to Figure 10.2.8.

```
694200 694350 Strongly_conserved +
```

Figure 10.2.10 An example of a line in the body of an `underlay` file used to paint just the upper or lower half of a region by using a + or – sign.

site (see Basic Protocol). The appearance of the colors will vary from one printer or monitor to the next and should be chosen carefully.

Alternatively, genomic features such as introns and exons or features of interest such as strongly conserved regions can be labeled directly in the underlay file. This is illustrated in a second format for underlay files (see Fig. 10.2.9) in which the color and associated feature are listed at the top of the file, while the coordinates and type of feature are listed within the body of the file. The labels, which must consist of a single word, appear in the underlay legend.

The header lines list the color describing each feature. The lines in the body indicate where the colors are placed. It is possible to paint just the upper or lower half of a region by using a + or – sign, as shown in Figure 10.2.10.

This option makes it feasible to differentiate between the two strands or to plot potentially overlapping features like gene predictions and database matches in separate spaces. In normal use, colors are overlaid one on top of the other; therefore, the last color assigned to a region is displayed. Thus, in the first *CLDN5* underlay example shown above, the intron coordinates are specified as the entire length of the gene, 691854-693201, but only those regions with no overlapping colors for UTRs or exons are colored light yellow to indicate the presence of an intron.

GENERATING `ANNOTATION` FILES FOR USE WITH PipMaker

SUPPORT PROTOCOL 4

Advanced PipMaker accepts a file containing annotations that associate World Wide Web hyperlinks with specified positions in the first sequence. The resulting PDF file is decorated with colored bars, which are clickable when viewed using Acrobat Reader. The file defines various types of hyperlinks and associates a color with each of them, then specifies the type, position, description, and URL for each annotated feature. For instance, in the example shown in Figure 10.2.11, each feature subsequently identified as a `LocusLink` entry will be colored `blue`. The `Name` field must be a single word, perhaps containing underline characters, as in `LocusLink_entry`. Colors start with capital letters and have the same list of choices as the `underlay` file (mentioned above). Multiple type definition entries can be listed in the header of the annotation file with

```
%define type
%name LocusLink
%color Blue
```

Figure 10.2.11 A sample type definition entry for the header of an `annotation` file for Advanced PipMaker.

```
%define annotation
%type LocusLink
%range 691855 693202
%label CLDN5
%summary this link takes you to a locus link entry for human CLDN5
 with additional links available at that site.
%url http://www.ncbi.nlm.nih.gov/LocusLink/LocRpt.cgi?l=7122
```

Figure 10.2.12 A sample annotation entry for the body of an `annotation` file for Advanced PipMaker.

variations in the name and color fields. The subsequent lines in the body of the file associate a type of annotation with positions in the first sequence and provide a URL (see Fig. 10.2.12).

Note that the `Summary` and `URL` fields (but not `Label`) can be broken into several lines for convenience; the line breaks are removed when the file is read and they are not replaced with spaces. Thus, a continuation line for a summary typically begins with a space to separate it from the last word of the previous line, while a URL continuation does not. If the summary is omitted, it is assumed to be the same as the label. Since hyperlink bars are drawn on separate lines in the pip, individual entries in the `annotation` file can have overlapping coordinates. Each `%define` line is immediately preceded by a blank line.

Clicking on an annotation bar with the pip will bring up a page that displays a description of the feature (as given in the `%summary field`), including a "hand" icon that can be clicked to visit the specified URL. Within the PDF document, Acrobat Reader's navigational controls should be used; e.g., the Web browser's Back button will not return to the pip.

SUPPORT PROTOCOL 5

INSTALLING STAND-ALONE Blastz

The Blastz program was designed to handle relatively long sequences, and it can also be used if the second sequence is split into contigs. The Pipmaker Web server imposes restrictions on the size of the data that the program is willing to handle, primarily to limit the load on the authors' server. Stand-alone Blastz can be used on much larger inputs. For instance, when run in the most sensitive mode (the default), aligning a 1-Mb human contig against a 30-Mb mouse genomic sequence typically requires less than 400 megabytes of memory. Less sensitive modes require an order of magnitude less space. Note that the stand-alone Blastz program computes an alignment but not a pip. The output from this program is a raw Blastz file that can be viewed with Laj (see Guidelines for Understanding Results and see Suggestions for Further Analysis). Stand-alone Blastz is not recommended for novice users.

Necessary Resources

Hardware

The authors test and use Blastz on Solaris/Sparc and Linux/x86 platforms, but it should be portable to virtually any ANSI/POSIX system, including Windows and Macintosh.

Software

The current development snapshot of Blastz is available on the authors' Web site (*http://bio.cse.psu.edu/*), in a `tar.gz` file. To unpack it, `tar` and `gzip` (or compatible programs) will be needed. An ANSI-compatible C compiler and the `make` utility will be needed to compile and install it.

Files

The stand-alone of the Blastz program uses the same sequence and repeats files as the PipMaker Web server (see Basic Protocol).

Note: For an introduction to Unix, see *APPENDIX 1C*.

Installing Blastz

In Unix, type:

```
zcat blastz.tar.gz | tar -xf -
```

then

```
cd blastz-source
```

Edit the `Makefile` to suit the system being used. If a Linux system is used, it may be necessary to uncomment `DUSE_OBSTACK`; this should improve performance in any case. Next, run `make` to compile the program. The executable is named `blastz`, and it does not depend on any other files. One can copy it to one's `bin` directory or run it in place.

Running Blastz

Syntax and arguments can be found by typing `blastz`. A `README` documentation file is available with the download file.

GUIDELINES FOR UNDERSTANDING RESULTS

PipMaker

PipMaker returns the alignments generated by Blastz in any or all of five different formats: a pip, a dot plot, a concise listing of the coordinates of the aligning segments, a conventional textual alignment, and raw Blastz output (for Laj or SGP-1). In addition to alignments, other output data are available, including an analysis of exons, predictions of contig order and orientation, and enhanced PDF with embedded contig names.

The Pip and Dot Plot

The first page of Advanced PipMaker's pip shows a compact overview of the alignment image (Fig. 10.2.13). If an `underlay` file was given, the first row of the overview shows it. The bottom (and perhaps only) row of the overview shows aligned regions in green and strongly aligned regions (at least 100 bp without a gap and with at least 70% nucleotide identity) in red. In both versions of PipMaker, a legend describes the icons representing interspersed repeats. In Advanced PipMaker, there are additional legends describing the meanings of colors in the `underlay` and `annotation` files.

PipMaker plots the position (in the first sequence) and percent identity of each gap-free segment of the alignments using a pip display (Fig. 10.2.2) and the position of each alignment relative to both sequences using a dot plot (Fig. 10.2.3). The top horizontal axis

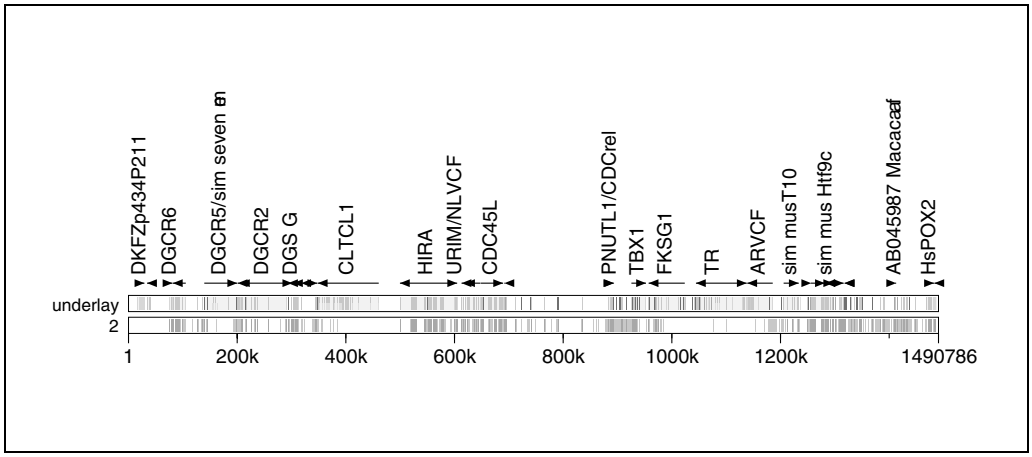


Figure 10.2.13 Compact overview of the alignment. The two-panel image shows the locations of aligned regions (upper panel) and the position of colors specified by the `underlay` file (lower panel). Green bars represent all regions within an alignment and red bars are those regions that align at a high level of similarity (at least 100 bp without a gap and with at least 70% nucleotide identity). The colors are specified from the `underlay` file and the gene names and directionality come from the `exons` file.

of the pip is automatically decorated with icons (Fig. 10.2.14) representing repeats (from the `repeats` file), `exons` (from the exons file), and CpG islands (which it computes automatically from the sequence). The coordinates (lower horizontal axis) are the nucleotide positions in the first sequence. The dot plot displays gene names relative to their positions along the horizontal axis and contig names along the vertical axis of the plot, and shows the lengths of both sequences.

The portions of the pip corresponding to specific features can be color-coded. For instance, in the sample figure, coding exons are colored light blue, light orange corresponds to UTRs, and light yellow indicates intronic sequences. Conserved sequences that are not annotated as coding regions are of interest for their potential as regulatory elements, and are highlighted using a red underlay. In a fully annotated pip, one can identify clusters of repeats, presumably formed by recursive integration into closely linked sites, as well as more dispersed repetitive elements. Furthermore, the pip also shows the locations of CpG islands, which often flank the 5′ ends of human genes. A legend for the annotations supplied as hyperlinks above the alignments appears as the last page of the PDF file.

If the second sequence has multiple contigs, the pip shows all alignments as a continuous plot relative to the first sequence. Several options allow a user to determine where the breaks in continuity occur. For example, in the dot plot, horizontal lines separate contigs. Alternatively, the "concise" output file (see below) arranges the information about aligning segments in sections according to contig with individual header lines. Finally, contig names embedded in the PDF output serve as useful guides to the underlying identity of an aligning segment.

Concise Textual Form of the Alignments

The "concise" output file is useful for precisely identifying sequence positions corresponding to conserved regions indicated in the pip. There is a one-to-one correspondence between lines in this file and the small horizontal line segments in the pip (each of which corresponds to a region between successive gaps in a local alignment). For example, some alignments in the *CLDN5* gene on chromosome 22 are shown in Figure 10.2.15.

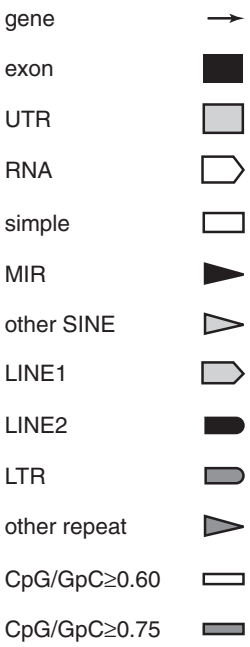


Figure 10.2.14 Icons used in a pip that represent features in a genomic sequence, such as exons, repeats, and CpG islands.

```
Positions in      Positions in
 sequence 1        sequence 2

692296-692305 <--> 28038-28047 70% (10 nt)
692320-692395 <--> 28048-28123 62% (76 nt)
692397-693139 <--> 28124-28866 85% (743 nt)
693140-693197 <--> 28878-28935 52% (58 nt)
693208-693216 <--> 28936-28944 56% (9 nt)
```

Figure 10.2.15 An example of the concise output file.

The first line asserts that positions 692296-692305 in the first sequence (10 base pairs) aligns to positions 28038-28047 of the second sequence, without gaps and at 70% identity.

The Traditional Textual Form of the Alignments

The textual alignment is a nucleotide level view of the alignment, with the first sequence on the top line and the second sequence on the lower line (see Fig. 10.2.16). In between, vertical bars represent matching bases and colons represent mismatches that are transitions. Dashes in a sequence indicate gaps.

Raw Blastz Output

The alignment file produced by PipMaker's Blastz program can be viewed in Laj, which is an interactive tool for viewing and manipulating pairwise alignment output (Wilson et al., 2001). The Laj program, which is written in Java and available at *http://bio.cse.psu.edu/*, must be downloaded and run on the user's computer. Also, alignments in this format can be submitted to the SGP-1 gene prediction program, which uses homology for gene prediction (Wiehe et al, 2001; *http://soft.ice.mpg.de/sgp-1*). Laj

```
692320     .    :    .    :    .    :    .    :    .
GCCCAGGCCTGGGCAGCGAGAGGGCCCTGCTCCCCGCTCAAGGCTCCCAG
||||||:| ||::||::|:|||| |||   |||:||:|||| ::|:::::||
GCCCAGACATGAACAATGGGAGGCCCC--CTCTCCACTCACAACCTTTAG
28048
```

Figure 10.2.16 The traditional textual form of alignment output.

```
> CDC45L:   648772-689427   254390-25347
     CDS:   648796-687735   254413-23908      ATG-TAG ATG-GAG
       1:   648772-648846   254390-254463       GT-AG   GT-AG
       2:   648985-649044   254614-254673       GT-AG   GT-AG
       3:   649780-649872   255497-255589       GT-AG   GT-AG
       4:   651517-651654   257109-257246       GT-AG   GT-AG
       5:   652689-652832   258555-258698       GT-AG   GT-AG
       .
       .
       .
      18:   687671-687736   288843-288909       GT-AG   GT-AG
      19:   689308-689427   292320-292547
```

Figure 10.2.17 Analysis of Exons output when the second sequence has only one contig.

uses most of the same input files as PipMaker, but accepts only the coordinate-based format of the `repeats` file.

Analysis of Exons

The optional Analysis of Exons feature is provided to give the user putative exon annotations in the second sequence. PipMaker attempts to use the alignments to map each position in the `exons` file for the first sequence onto the corresponding position in the second sequence. For each coding-region specification (i.e., line beginning with +) the first and last codons are displayed, and for each intron the first and last two nucleotides are shown. These tri- and dinucleotides are shown for the first sequence and, if possible, for the second sequence. Question marks, when present, indicate that the corresponding region did not align with the second sequence. The *CDC45L* gene located adjacent to the *CLDN5* gene serves as an example in Figure 10.2.17. Note that the stop codon in human *CDC45L*, TAG, differs from the mouse stop codon, TGA. However, the mouse stop codon (T) is aligned with a gap in the human sequence, resulting in the remaining GA and following G aligning with the human TAG. Analysis of Exons only reports the aligning nucleotides when predicting coding regions in a second sequence, and users should check output carefully. This output shows that the *CDC45L* coding region in both sequences begins with ATG and ends with TAG or GAG, respectively. Similarly, all introns conform to the normal GT-AG splicing consensus. If the coding region is specified, then the putative coding region (CDS) is printed for both sequences, if possible. The coding sequence is based on coordinates in the exon file for sequence 1 and predicted using the alignment information for sequence 2 (see Fig. 10.2.18).

```
>CDC45L, putative CDS for sequence 1 (1701 bp)
ATGTTCGTGTCCGATTTCCGCAAAGAGTTCTACGAGGTGGTCCAGAGCCA
GAGGGTCCTTCTCTTCGTGGCCTCGGACGTGGATGCTCTGTGTGCGTGC…

>CDC45L, putative CDS for sequence 2 (1702 bp)
ATGTTCGTGACCGATTTCCGCAAGGAGTTCTACGAGACGGTCCACAACCA
GAGGGTCCTTCTCTTTGTGGCCTCGGACGTGGATGCCTTGTGTGCTTGC…
```

Figure 10.2.18 Putative coding sequence from optional Analysis of Exons output.

```
Index of fragments of the second sequence:
    1: >Contig101
    2: >Contig102
```

Figure 10.2.19 A sample index for Analysis of Exons output when the second sequence file contains multiple contigs.

```
>  CDC45L:   648772-689427   1:254390-2:25347
     CDS:   648796-687735   1:254413-2:23908     ATG-TAG ATG-GAG
       1:   648772-648846   1:254390-1:254463      GT-AG    GT-AG
       2:   648985-649044   1:254614-1:254673      GT-AG    GT-AG
       3:   649780-649872   2:255497-2:255589      GT-AG    GT-AG
       .
       .
       .
```

Figure 10.2.20 A sample listing of the exon positions for Analysis of Exons output (analogous to Fig. 10.2.17) when the second sequence file contains multiple contigs.

When the second sequence file contains multiple contigs, Analysis of Exons output is somewhat more complicated. It begins with an enumeration of the contigs (using their FASTA header lines), and positions in the second sequence are identified as `i:j` which denotes position `j` in contig `i`. An example of an index is shown in Figure 10.2.19 and an example of a listing of exon positions is shown in Figure 10.2.20. Here, the left-most CDS position aligns with position 254413 in contig 1, which has FASTA header line `>Contig101`.

Order and Orient Contigs

PipMaker can predict the order and orientation of contigs in the second sequence. This option returns three files: a `text` file, a `dot plot` file, and a FASTA `sequence` file of the ordered and oriented contigs.

The `text` file describes the predicted arrangement by using one line for each ordered and oriented contig (see Fig. 10.2.21). Each line begins with the range of coordinates, relative to the first sequence, that encompasses all the aligning regions from the corresponding contig. The next column of data is the normalized score of the contig's highest-scoring local alignment. Thus, a small value, e.g., less than 5, indicates that the contig's predicted position is weakly supported. The third item of data in each row is the contig's FASTA header line (`>` followed by text), where an appended character `-` indicates

```
154650-163329     3.9    >AF031572
74963-1169968    50.6    >AC003063
352200-360259     2.8    >AW044803-
```

Figure 10.2.21 `Text` file describing the predicted arrangement of ordered and oriented contigs.

```
>SequenceX an unordered, working draft file
ACGT...TGCAN100NAAGT...ACGT
```

Figure 10.2.22 An example of a second sequence consisting of two segments separated by 100 `N`s, which is then is treated as shown in Figure 10.2.23.

```
> SequenceX an unordered, working draft file
ACGT...TGCA
> SequenceX.C2
AAGT...ACGT
```

Figure 10.2.23 See Figure 10.2.22.

the reverse complement. Any contigs for which a prediction could not be made are listed at the bottom of the file in the order that they appeared in the submitted sequence file.

The `dot plot` file shows the alignment predicted by the ordering and orientation operations. If a dot plot using the original order and orientation is desired, it must be explicitly requested when the Order and Orient option is used.

The default for the Order and Orient option assumes that the second sequence consists of contigs, separated either by FASTA header lines, or by runs of 100 or more of the letter `N`. PipMaker replaces each run of `N`'s by a synthesized FASTA header line of the form `>word.Ci`, where `word` is the first word on the closest preceding FASTA header line present in the original submission file, and `i` is a number (2, 3, 4, etc.). Thus, a second sequence consisting of two segments separated by 100 `N`'s, as shown in Figure 10.2.22, is treated as shown in Figure 10.2.23.

Some users prefer to submit a genomic sequence in which the nucleotides encompassed by repeats are masked with the letter `N`. However, for use with the Order and Orient utility, it is best that interspersed repeats in the second sequence not be replaced by long runs of `N`'s. Alternatively, the user can disable the `Break at NN..` option, and thereby get order-and-orientation predictions when the letter `N` is used to mask regions of the second sequence.

PDF with Embedded Contig Names

When this option is chosen, local alignments in the pip can be pointed at with the mouse to reveal the identity of the second sequence contig represented in the alignment, or to report a gap between two alignments. The first word on the FASTA header line for the second sequence, or a contig thereof, is reported. The pip must be viewed as a PDF file in Adobe Acrobat Reader to use this option. For example, each of the messages shown in

```
Positions in
 Sequence 1   Message
    1-1400    1400 bp unmatched
 1401-1586    1401-1586 matches 3146-3324 of Contig1
 1587-2191    605 bp gap; 889 bp in Contig1
 2192-3367    2192-3367 matches 4214-5531 of Contig1
 3368-3888    521 bp gap; between Contig1 and Contig2-
 3889-3934    3889-3934 matches 6-51 of Contig2-
 3935-4089    155 bp gap; 1013 bp in Contig2-
 4090-4375    4090-4375 matches 1065-1340 of Contig2-
```

Figure 10.2.24 Example messages from a PDF file with embedded contig names.

Figure 10.2.24 might be returned as the identity of the second sequence in an aligning segment.

The notation `Contig2-` refers to the reverse complement of the sequence with FASTA header `>Contig2...`. In this example, sequence 1 has two local alignments with Contig1, followed by two local alignments to the reverse complement of Contig2. Note that the last gap is 155 bp (positions 3935-4089) in sequence1 and 1013 bp in the second sequence (Contig2), perhaps due to an insertion in the latter.

The messages embedded in the pip are not intended to be clickable hyperlinks. The authors are storing the messages as URLs because it is the most convenient way to display a short string in the status area, but this is not the normal use of URLs in a PDF file. Acrobat Reader may need to be configured to display these messages: follow the menu File > Preferences > Weblink and set the Link Information selector to Always Show.

COMMENTARY

Background Information

As complete genomic sequences become readily available, the tools for analyzing them must handle increasingly large amounts of data and provide efficient methods of analysis. The PipMaker server aligns very long genomic DNA sequences, up to about 2 Mb, quickly and with high sensitivity. Few studies on the specificity of genomic sequence alignments have been done, and although objective assessment of specificity is desirable, additional resources may be needed, such as the development of reference data sets. In the authors' experience, the alignments generated by PipMaker have been useful and it is rare that informative similar regions are missed.

While PipMaker is appropriate for comparing genomic sequences from any two related species, the type of information that can be inferred depends on the level of conservation and the time and divergence rate since the separation of the species. For instance, human and chimp sequences align at a high percent identity throughout, so only small insertions and deletions stand out as differences. In contrast, comparisons of human and mouse DNA show that much of the DNA does not align. Within the sequences that do align, coding exons stand out as ungapped alignments of roughly 100 bp or more. In the noncoding, aligning DNA, some gene regulatory elements are detectable as highly conserved not only in human and mouse, but also in other pairs of species that diverged as long as 100 to 300 million years ago, including *Caenorhabditis elegans* and *C. briggsae*, or *Escherichia coli* and *Salmonella* spp. (Oeltjen et al., 1997; Kent and Zahler, 2000; McClelland et al., 2000).

Once orthologous sequences are aligned, their interpretation leads to insights into the genome landscape, including identification of protein-coding segments in both species (Jang et al., 1999; Liang et al., 1999), identification of regulatory signals (Gumucio et al., 1996; Hardison et al., 1997; Oeltjen et al., 1997; Loots et al., 2000), deduction of mechanisms of genome evolution, and location of differences in genome organization between species. Pips

```
     A     C     G     T
A   91  -114   -31  -123
C -114   100  -125   -31
G  -31  -125   100  -114
T -123   -31  -114    91
```

Figure 10.2.25 The default scoring matrix.

provide advantages over conventional dot plots by conveying information about both the percent identity and location of the gaps within an aligned region. Pips also provide a visual display of annotated features alongside information about the level of sequence conservation of those features. This information can be used to examine correlations between the positions of features within a genome (Chiaromonte et al., 2002), to investigate the organization of genes within related clusters (Hardison and Miller, 1993), or to analyze genomic regions that are devoid of any genes, but still show sequence conservation between species (Kurihara et al., 2002).

The dot plot display, while missing information about quality of the alignments, does show where the alignments occur in the two species. Thus, the dot plot is the better output for visualizing duplications, inversions, and insertion/deletion events between orthologous sequences (Hardison and Miller, 1993; Endrizzi et al., 2000). The dot plot is particularly useful when using PipMaker to find relationships among unordered and unoriented contigs. Application of the Order and Orient function in PipMaker provides a fusion of the contigs in the second sequence guided by the position and orientation of homologous sequences in the first sequence.

As large, complex genomes are sequenced, it becomes more efficient to precompute alignments of the sequences and provide the alignments and graphs to users without their having to find the sequences and annotate them. The December 2001 human genome assembly (Lander et al., 2001) has been aligned with a working draft sequence of mouse (Waterston et al., 2002). The Blastz alignments and graphs of them can be obtained as pips from the authors' link to whole genome human-mouse alignments (via the PipDispenser at *http://bio.cse.psu.edu/*) or as tracks on the Genome Browser (*http://genome-test.cse.ucsc.edu/*; Kent et al., 2002; *UNIT 1.5*). Other whole genome alignments are available for enterobacteria at the EnteriX server at *http://bio.cse.psu.edu/* (Florea et al., 2000). As more genomes are sequenced, integration of precomputed whole-genome alignments with other annotations will become an even more significant part of genomic sequence analysis.

Critical Parameters and Troubleshooting

The scoring matrices

The default scoring matrix is optimized for human-mouse comparisons and its development is described in Chiaromonte et al. (2002). In brief, substitutions are scored by the matrix shown in Figure 10.2.25.

Affine gap penalties (*UNIT 3.1*) are applied, with a gap open score of –400 and a gap extension score of –30.

User-controlled masking of interspersed repeats and low-complexity regions

PipMaker attempts to eliminate spurious alignments caused by interspersed repeat elements and low-complexity regions, yet indicate meaningful alignments that may involve such elements. For instance, an interspersed repeat or trinucleotide repeat sequence may be part of a protein-coding region. The strategy is not perfect, however, and the user may want to take control of the way these elements are handled during construction of the alignment.

The alignment program used by PipMaker interprets lowercase letters as indicating regions of the sequence that are to be masked at early stages of the alignment process, but not at later stages. The alignment program follows the general design of the "gapped BLAST" family of programs, which start by finding short, exact matches, then extend those matches to alignments that include gaps (Altschul et al., 1997). PipMaker ignores regions of the first sequence that contain lowercase letters when searching for exact matches, but utilizes those regions when expanding exact matches to form longer alignments. Regions containing only `N`

or X characters are not aligned in either phase. Users of the Advanced PipMaker page can control masking by submitting sequences that follow this convention.

Common errors

E-mail errors: Mail servers are frequently configured to reject large messages or to impose mailbox quotas on individual users. PipMaker ouput ranges from small file sizes, e.g., 294 kb, to an upper limit of 60 Mb, depending on the size of the input files. A user's mail system may reject the message, depending on the size of preset limits to incoming mail. Another common E-mail error occurs when users type the wrong return address. The output message may bounce back to the server administrator, but the intended recipient never receives the message.

Format errors: PipMaker uses a variety of formats for its various input files. Occasionally a user will confuse the proper syntax format for a file. Another common error occurs when data from a Web page is cut and pasted into the submission form. This data is formatted as HTML and is uninterpretable by PipMaker. Microsoft Word documents are not readable by PipMaker either, and should be saved in a plain text format.

Timeout errors: The PipMaker server incorporates a number of "sanity checks" to limit the run time of submitted jobs. A short time limit is imposed when RepeatMasker information is omitted, due to the extended time necessary to align these types of files.

Repeated errors: Jobs that fail due to syntax errors or timeouts are reported in the message page or E-mail message. Continued resubmission without correction or elimination of the errors results in continued error messages. Furthermore, an automated approach to submitting jobs via a robot can generate a great many failures in short order. This problem poses a difficult solution and may involve redesigning PipMaker to accommodate it.

Bookmark errors: Users frequently bookmark the cgi submission page instead of the top-level page (*http://bio.cse.psu.edu/*). Links to the PipMaker server, updated versions of PipMaker, and complementary programs appear on the top-level page and may otherwise be overlooked.

Program errors: User-identified bugs in the program are repaired when practical. If not, a work-around may be suggested. Comments should be directed to *pipmaster@bio.cse.psu.edu*.

Browser errors: Some versions of MS Internet Explorer (MSIE) have bugs that interact with PipMaker. An error such as "page not found" sometimes occurs instead of the appropriate error message. Work-arounds may be provided when a problem is identified. A second option is to try another Web browser.

Privacy policy

Data submitted to PipMaker are completely confidential. However, as an essential part of the effort to improve the server's utility, the authors regularly inspect submissions to see how the server is being used. It is intended that all copies of user data be deleted within 24 hr of submission.

Suggestions for Further Analysis

Complementary tools

Users interested in complementary programs for annotating sequences should go to the PipTools link at *http://bio.cse.psu.edu/* (Elnitski et al., 2002). PipTools is a collection of small programs designed to help users of the PipMaker server to prepare their input files more conveniently. Preparing and manipulating these files manually can be time consuming, so the authors provide this set of tools to facilitate the process. There are also a few tools for searching PipMaker's output that work with either the "raw Blastz output" or the "concise textual form of the alignments."

Laj is a tool for viewing and manipulating the output from pairwise alignment programs such as Blastz. It can display interactive dot plot, pip, and text representations of the alignments, a diagram showing the locations of exons and repeats, and annotation bars similar to PipMaker's. It is available for download from *http://bio.cse.psu.edu/*.

Whole genome alignments

Precomputed human and mouse whole genome alignments are available at *http://bio.cse.psu.edu/*. Links at this site provide nucleotide level alignments of conserved regions. The alignments are currently shown as percent identity plots annotated with the names of RefSeq genes, but this may change as annotations are updated.

Literature Cited

Altschul, S.F., Madden, T.L., Schaffer, A.A., Zhang, J., Zhang, Z., Miller, W., and Lipman, D.J. 1997. Gapped BLAST and PSI-BLAST: A new generation of protein database search programs. *Nucleic Acids Res*. 25:3389-3402.

Bulger, M., van Doorninck, J.H., Saitoh, N., Telling, A., Farrell, C., Bender, M.A., Felsenfeld, G., Axel, R., Groudine, M., and von Doorninck, J.H. 1999. Conservation of sequence and structure flanking the mouse and human beta-globin loci: The beta-globin genes are embedded within an array of odorant receptor genes. *Proc. Natl. Acad. Sci. U.S.A.* 96:5129-5134.

Bulger, M., Bender, M.A., van Doorninck, J.H., Wertman, B., Farrell, C.M., Felsenfeld, G., Groudine, M., and Hardison, R. 2000. Comparative structural and functional analysis of the olfactory receptor genes flanking the human and mouse beta-globin gene clusters. *Proc. Natl. Acad. Sci. U.S.A.* 97:14560-14565.

Burge, C. and Karlin, S. 1997. Prediction of complete gene structures in human genomic DNA. *J. Mol. Biol*. 268:78-94.

Chiaromonte, F., Yap, V.B., and Miller, W. 2002. Scoring pairwise genomic sequence alignments. *Pac. Symp. Biocomput.* 2002:115-126.

Elnitski, L., Riemer, C., Petrykowska, H., Florea, L., Schwartz, S., Hardison, R., and Miller, W. 2002. PipTools: A computational toolkit to prepare and evaluate annotated pairwise comparisons of genomic sequences. *Genomics*. In press.

Endrizzi, M.G., Hadinoto, V., Growney, J.D., Miller, W., and Dietrich, W.F. 2000. Genomic sequence analysis of the mouse Naip gene array. *Genome Res*. 10:1095-1102.

Florea, F., Riemer, C., Schwartz, S., Zhang, Z., Stojanovic, N., Miller, W., and McClelland, M. 2000. Web-based visualization tools for bacterial genome alignments. *Nucleic Acids Res*. 28:3486-3496.

Gumucio, D., Shelton, D., Zhu, W., Millinoff, D., Gray, T., Bock, J., Slightom, J., and Goodman, M. 1996. Evolutionary strategies for the elucidation of *cis* and *trans* factors that regulate the developmental switching programs of the -like globin genes. *Mol. Phylogenet. Evol*. 5:18-32.

Hardison, R. and Miller, W. 1993. Use of long sequence alignments to study the evolution and regulation of mammalian globin gene clusters. *Mol. Biol. Evol*. 10:73-102.

Hardison, R., Slightom, J.L., Gumucio, D.L., Goodman, M., Stojanovic, N., and Miller, W. 1997. Locus control regions of mammalian globin gene clusters: Combining phylogenetic analyses and experimental results to gain functional insights. *Gene* 205:73-94.

Jang, W., Hua, A., Spilson, S.V., Miller, W., Roe, B.A., and Meisler, M.H. 1999. Comparative sequence of human and mouse BAC clones from the mnd2 region of chromosome 2p13. *Genome Res*. 9:53-61.

Kent, W.J. and Zahler, A.M. 2000. Conservation, regulation, synteny, and introns in a large-scale *C. briggsae–C. elegans* genomic alignment. *Genome Res*. 10:1115-1125.

Kent, W.J., Sugnet, C.W., Terrence, S.F., Roskin, K.M., Pringle, T.H., Zahler, A.M., and Haussler, D. 2002. The Human Genome Browser at UCSC. *Genome Res.* 12:996-1006.

Kurihara, L.J., Semenova, E., Miller, W., Ingram, R.S., Guan, X.J., and Tilghman, S.M. 2002. Candidate genes required for embryonic development: A comparative analysis of distal mouse chromosome 14 and human chromosome 13q22. *Genomics* 79:154-161.

Lander, E.S., Linton, L.M., Birren, B., Nusbaum, C., Zody, M.C., Baldwin, J., Devon, K., Dewar, K., Doyle, M., FitzHugh, W., Funke, R., et al. 2001. Initial sequencing and analysis of the human genome. *Nature* 409:860-921.

Liang, Y., Wang, A., Belyantseva, I., Anderson, D., Probst, F.J., Barber, T.D., Miller, W., Touchman, J., Jin, L., and Sullivan, S. 1999. Structure and expression of the human and mouse novel unconventional myosin XV genes responsible for hereditary deafness, *DFNB3* and *shaker-2*. *Genomics* 61:243-258.

Loots, G.G., Locksley, R.M., Blankespoor, C.M., Wang, Z.E., Miller, W., Rubin, E.M., and Frazer, K.A. 2000. Identification of a coordinate regulator of interleukins 4, 13, and 5 by cross-species sequence comparisons. *Science* 288:136-140.

McClelland, M., Florea, L., Sanderson, K., Clifton, S.W., Parkhill, J., Churcher, C., Dougan, G., Wilson, R.K., and Miller, W. 2000. Comparison of the *Escherichia coli* K-12 genome with sampled genomes of a *Klebsiella pneumoniae* and three *Salmonella enterica* serovars, Typhimurium, Typhi and Paratyphi. *Nucleic Acids Res*. 28:4974-4986.

Oeltjen, J.C., Malley, T.M., Muzny, D.M., Miller, W., Gibbs, R.A., and Belmont, J.W. 1997. Large-scale comparative sequence analysis of the human and murine Bruton's tyrosine kinase loci reveals conserved regulatory domains. *Genome Res*. 7:315-329.

Schwartz, S., Zhang, Z., Frazer, K.A., Smit, A., Riemer, C., Bouck, J., Gibbs, R., Hardison, R., and Miller, W. 2000. PipMaker: A web server for aligning two genomic DNA sequences. *Genome Res.* 10:577-586.

Waterston, R., Lindblad-Toh, K., Birney, E., Rogers, J., Abril, J.F., Agarwal, P., Agarwala, R., Ainscough, R., Alexandersson, M., An P., Antonarakis, S.E., Attwood, J., Baertsch, R.Bailey, J., Barlow, K., Beck, S., Berry, E., Birren, B., Bloom, T., Bork, P., Botcherby, M., Bray, N., Brent, M.R., Brown, D.B., Bult, C., Burton, J., Butler, J., Campbell, R.D., Carninci, P., Cawley, S., Chinwalla, A., Church, D., Clamp, M., Clee, C., Collins, F.S., Cook, L., Copley, R.R., Coulson, A., Couronne, O., Cuff, J., Curwen, V., Cutts, T., Daly, M., David, R., Davies, J., Delehaunty, K., Deri, J., Dermitzakis, E.T., Dewey, C., Dickens, N.J., Diekhans, M., Dodge, S., Dubchak, I., Dunn, D.M., Eddy, S.R., Elnitski, L., Emes, R.D., Eswara, P., Eyras, E., Felsenfeld, A., Fewell, G., Flicek, P., Foley, K., Frankel, W.N., Fulton, L., Fulton, R., Furey, T.S., Gage, D., Gibbs, R.A., Glusman, G., Gnerre, S., Goldman, N., Goodstadt, L., Graffham, D., Graves, T.,

Green, E.D., Gregory, S., Guigo, R., Guyer, M., Hardison, R.C., Haussler, D., Hayashizaki, Y., Hillier, L., Hinrichs, A., Hlavina, W., Holzer, T., Hsu, F., Hua, A., Hubbard, T., Hunt, A., Jackson, I., Jaffe, D.B., Johnson, L.S., Jones, M., Jones, T.A., Joy, A., Kamal, M., Karlsson, E.K., Karolchik, D., Kasprzyk, A., Kawai, A., Keibler, E., Kells, C., Kent, W.J., Kirby, A., Kolbe, D., Korf, I., Kucherlapati, R.S., Kulbokas, R.J. III, Kulp, D., Landers, T., Leger, J.P., Leonard, S., Letunic, I., Levine, R., Li, J., Li, M., Lloyd, C., Lucas, S., Ma, B., Maglott, D.R., Maier, J., Mardis, E.R., Matthews, L., Mauceli, E., Mayer, J.H., McCarthy, M., McCombie, R., McLaren, S., McLay, K., McPherson, J., Meldrim, J., Meredith, B., Mesirov, J.P., Miller, W., Miner, T., Mongin, E., Montgomery, K.T., Morgan, M., Mott, R., Mullikin, J.C., Muzny, D.M., Nash, W., Nelson, J., Nhan, M., Nicol, R., Ning, Z., Nusbaum, C., O'Connor, M.J., Okazaki, Y., Oliver, K., Overton-Larty, E., Pachter, L., Parra, G., Pepin, K., Peterson, J., Pezvner, P., Plumb, R., Pohl, C., Poliakov, A., Ponce, T., Ponting, C., Potter, S., Quail, M., Reymond, A., Roe, B.A., Roskin, K.M., Rubin, E., Rust, A.G., Santos, R., Sapojnikov, V., Schultz, B., Schultz, J., Schwartz, M.S., Schwartz, S., Scott, C., Seaman, S., Searle, S., Sharpe, T., Sheridan, A., Shownkeen, R., Sims, S., Singer, J.B., Slater, G., Smit, A., Smith, D.R., Spencer, B., Stabenau, A., Stange-Thomann, N., Sugnet, C., Suyama, N., Tesler, G., Thompson, J., Torrents, D., Trevaskis, E., Tromp, J., Ucla, C., Ureta-Vidal, A., Vinson, J.P., von Niederhausern, A.C., Wade, C.M., Wall, M., Weber, R.J., Weiss, R.B., Wendl1, M., West, T., Wetterstrand, C., Wheeler, R., Wierzbowski, J., Willey, T., Williams, S., Wilson, R., Winter, E, Worley, K.C., Wyman, D., Yang, S., Shiaw-Pyng Ya, Zdobnov, E., Zody, M.C., and Lander, E.S. 2002. Initial sequencing and comparative analysis of the mouse genome. In press.

Wiehe, T., Gebauer-Jung, S., Mitchell-Olds, T., and Guigo, R. 2001. SGP-1: Prediction and validation of homologous genes based on sequence alignments. *Genome Res*. 11:1574-1583.

Wilson, M.D., Riemer, C., Martindale, D.W., Schnupf, P., Boright, A.P., Cheung, T.L., Hardy, D.M., Schwartz, S., Scherer, S.W., Tsui, L.C., Miller, W., and Koop, B.F. 2001. Comparative analysis of the gene-dense ACHE/TFR2 region on human chromosome 7q22 with the orthologous region on mouse chromosome 5. *Nucleic Acids Res*. 29:1352-1365.

Key References

Schwartz et al. 2000. See above.

Detailed documentation for the PipMaker server.

Internet Resources

http://bio.cse.psu.edu/

Links to alignment programs and complementary programs, including the PipMaker server homepage, the list of PipMaker underlay and annotation colors, PipMaker examples, whole genome human/mouse homology, and Laj download and instruction site.

http://www.cs.wisc.edu/~ghost/

Aladdin homepage (for the GhostScript program).

http://www.adobe.com/

Adobe homepage (for the Acrobat Acrobat Reader program).

http://ftp.genome.washington.edu/cgi-bin/RepeatMasker/

RepeatMasker Web site.

http://www.ncbi.nlm.nih.gov/

Contains link to GenBank Web site.

http://genome.cse.ucsc.edu/

Human Genome Browser.

http://www.ensembl.org/

Ensembl Genome Browser.

http://soft.ice.mpg.de/sgp-1

SGP-1 (Syntenic Gene Prediction Program) server.

Contributed by Laura Elnitski, Cathy Riemer, Scott Schwartz, Ross Hardison, and Webb Miller
The Pennsylvania State University
University Park, Pennsylvania

Using MUMmer to Identify Similar Regions in Large Sequence Sets

The MUMmer sequence alignment package (Delcher et al., 1999, 2002) is a suite of programs to detect similar regions between biological sequences. The programs are specifically designed to rapidly detect regions of very high similarity in very long sequence sets, such as complete bacterial genomes or entire eukaryotic chromosomes.

The MUMmer package is set of programs, all of of which are designed to be as independent as possible so that the user can employ them as parts of other scripts. Most of the programs have simple text input and output formats that can easily be edited or modified by other scripts. Several Unix shell scripts are provided to run the system in the modes described below.

The methodology employed by all of the MUMmer scripts is to first identify a set of identical regions between the sequences being compared, and then cluster/combine the exact matches into larger regions of similarity. Alignments are provided for regions where the match is inexact. The fundamental type of exact match is a maximal unique match (MUM). This is an exact match between two sequences that occurs exactly once in each sequence (uniqueness) and is not contained within any larger match (maximality). The MUMmer programs use suffix trees to find exact matches (Gusfield, 1997).

In what follows, the terms *query sequence* and *reference sequence* refer to the two sequences (or sets of sequences) being compared. Depending on the protocol, there may be restrictions on the format of one of these, and in general the algorithms are not symmetric with respect to these sequences, so that different results may be obtained if the roles of the query and reference sequences are reversed.

The Basic Protocol describes the use of the MUMmer2 script to identify regions of similarity between a collection of DNA sequences and a single reference sequence. Three alternate protocols are then described. The NUCmer protocol (Alternate Protocol 1) relaxes the restriction that requires the reference to be a single sequence, thus allowing a set of multiple DNA sequences to be compared to another such set. The PROmer protocol (Alternate Protocol 2) compares DNA sequences by performing a 6-frame protein translation and then searching for matches in the resulting protein space. PROmer generally is able to detect more divergent matches that are too dissimilar to be detected directly at the DNA level by NUCmer. Alternate Protocol 3 uses the original MUMmer1 script to compute the longest single consistent alignment between two single DNA sequences. Finally, the Support Protocol describes how to obtain the MUMmer package from the TIGR Web site. Each of these protocols runs in the Unix environment. Users unfamilar with Unix should consult *APPENDIX 1C*.

MUMmer2: COMPARING A SET OF SEQUENCES TO A SINGLE REFERENCE SEQUENCE

BASIC PROTOCOL

This protocol takes a collection of query sequences and searches for unambiguous alignments between them and a single reference sequence. For example, a user could use this protocol to find alignments between shotgun sequence fragments and a reference genome, or to compare the genomes of closely related species, or to quickly compare alternate assemblies of the same genome.

It is important to note that not all matches are found—if there are repeat regions in the reference sequence that cannot be distinguished, then matches to neither will be reported.

Contributed by Arthur L. Delcher, Steven L. Salzberg, and Adam M. Phillippy
Current Protocols in Bioinformatics (2003) 10.3.1-10.3.18

Thus, this protocol should not be used when a comprehensive list of all possible matches is desired. Rather, its purpose is to identify high-fidelity matches as quickly as possible.

The protocol presents two ways of running MUMmer. The preferred method requires the user to invoke each of the programs from the Unix command line in turn. As an alternative, a Unix shell script is provided with the system.

This script uses only default parameter settings; however, the user is strongly encouraged to understand and modify the options of the different commands to suit his or her situation. Users unfamiliar with Unix should consult *APPENDIX 1C*.

Necessary Resources

Hardware

Unix or Linux workstation. The largest program used in this protocol requires main memory of approximately 20 bytes per base of reference sequence plus 1 byte per base of query sequence. Thus, to compare 2 million bases of query sequence to 3 million bases of reference sequence, the computer should have at least (20 × 3 Mb) + (1 × 2 Mb) = 62 Mb of main memory.

Software

MUMmer 2.12 package (see Support Protocol for download and installation)

Files

A multi-FASTA query file and a single-FASTA reference file (see *APPENDIX 1B* for information on FASTA). The files used in this example are complete genomic sequences from two strains of *Helicobacter pylori*—known as 26695 and J99. These sequences can be downloaded from TIGR's Comprehensive Microbial Resource at *http://www.tigr.org/CMR*, from the NCBI at *http://www.ncbi.nlm.nih.gov/PMGifs/Genomes/micr.html*, or from the *Current Protocols in Bioinformatics* Web site at *http://www3.interscience.wiley.com/c_p/cpbi_sampledatafiles.htm*.

Running MUMmer Interactively from the Command Line

Find MUMs between the query and reference sequences

1. Determine the minimum-length MUM to be reported. In general, large values should be used for highly similar sequences and for long sequences, whereas lower values are better for less similar and shorter sequences. A reasonable starting value for bacterial genomes is 20, which is the value that will be used in this example.

   Because this program runs very quickly, the user can run the program with different values of this parameter and determine which is best based on the resulting output. Generally, there is little point in using a value less than the length of match one would expect for random sequence, which for DNA sequences is ~$\log_4 R$, where R is the length of the reference sequence. For example, for a 4-million base genome, $\log_4 R = 11$ and the minimum MUM length should be significantly longer than 11 bp.

2. Determine the orientations of the sequences to be compared. In the comparison, the user has three choices: use only the forward orientation of the query sequences (default); use only the reverse orientation of the query sequences (`-r` option); or use both forward and reverse orientations (`-b` option). This example will use both forward and reverse orientations.

3. Run the program `mummer2` from the Unix command line. For this example, the user would type:

```
mummer2 -b -c -l20 hp26695.seq hpj99.seq >j99vs26695.out
```

```
> gi|12057207|gb|AE001439.1|AE001439
     183        17        22
     238        72       108
     347       181        92
     458       292        50
     509       343        35
     545       379        94
     640       474        25
     666       500        38
     705       539        44
     750       584        38
     807       641        23
     834       668        35
     870       704        24
     921       755        26
     966       800        29
    1014       848        20
       .         .         .
       .         .         .
       .         .         .
```

Figure 10.3.1 Beginning of output from program `mummer2` for *H. pylori* strains 26695 and J99. The first line is the FASTA header line from the query sequence in the file `hpj99.seq`. The following lines list the locations of all MUMs between this sequence and the reference sequence, which was in the file `hp26695.seq`. The first column is the beginning position of the MUM in the reference sequence 26695; the second column is the beginning position of the MUM in the query sequence J99; and the third column is the length of the MUM. The MUMs are listed in order by position in the query sequence.

`hp26695.seq` and `hpj99.seq` are names of the files containing the genomes of strains 26695 and J99 in FASTA format.

The MUMs found by the `mummer2` *program are identical sequences contained in both the query and reference strings, such that these sequences are not contained in any larger matches and are unique in the reference string. Note that these matches may not be unique in the query string, so that it is possible that two or more locations in the query string may match the same position in the reference string. If the user desires MUMs that are unique in both sequences, the program* `mummer1` *can be used instead of* `mummer2`*. However, this program uses substantially more memory than* `mummer2`*.*

The `-b` *option means that both the forward and reverse-complement orientations will be used (independently) to find MUMs. The* `-c` *option means that the positions of MUMs on the reverse-complement strand will be reported relative to the original string, instead of relative to the reverse-complement string, which is the default without the* `-c` *option. For example, a MUM of length 20 reported at position 20 on the reverse strand using the* `-c` *option is at positions* `1..20` *of the original reference string, whereas without the* `-c` *option, a MUM of length 20 reported at position 20 on the reverse strand is at positions* `20..39` *from the end of the original reference string. The* `-l20` *option means that only MUMs of length at least 20 will be output. Note that this option is a lower-case letter "ell" (*`l`*). The end of the command,* `>j99vs26695.out`*, directs the output of the command to a file named* `j99vs26695.out`*. The beginning of this file is shown in Figure 10.3.1, and the beginning of the portion of the file showing MUMs on the reverse-complement strand is shown in Figure 10.3.2. For each sequence in the query file (and its reverse-complement if the* `-b` *option is used), the locations of all MUMs between it and the reference sequence are given. The format is one MUM per line, where the first column is the starting position of the MUM in the reference sequence, the second column is the starting position of the MUM in the query sequence, and the third column is the length of the MUM.*

```
> gi|12057207|gb|AE001439.1|AE001439 Reverse
 1314091   1637275        22
 1348297   1626178        20
 1612843   1623008        23
 1283825   1615114        22
 1615880   1611836        31
 1615915   1611801        38
 1616062   1611654        37
 1616125   1611591        23
 1616149   1611567        25
 1616175   1611541        30
 1616236   1611480        74
      .         .         .
      .         .         .
      .         .         .
```

Figure 10.3.2 Beginning of the second part of the output from program `mummer2` for *H. pylori* strains 26695 and J99 showing the matches for the reverse-complement strand (indicated by the word `Reverse` at the end of the header line) of the query sequence J99. The format is the same as in the previous figure. If the query file contained more than one sequence, the matches for each sequence would follow in the same format, with each set of matches preceded by the FASTA header line of the sequence in the query file.

Cluster MUMs into consistent sets of matches

4. Determine parameters for grouping MUMs into clusters. Because MUMs are exact matches, to find similar regions that are not exact, MUMs that are likely to belong in the same region of similarity are grouped together. There are several parameters that determine how clusters are formed, all of which are specified as options to the program `mgaps`:

   a. Maximum separation between consecutive MUMs in a cluster, as measured by the number of bases between the MUMs in the query sequence. This is set by the `-s` option. For highly similar sequences use a small value; for dissimilar sequences use a large value.

   b. Maximum diagonal difference between consecutive MUMs in a cluster. The diagonal of a MUM is the difference between its position in the query and its position in the reference. The diagonal difference between consecutive MUMs is a measure of the size of insertions or deletions between them. There are two ways to specify the diagonal difference. The first is to set a fixed maximum diagonal difference with the `-d` option. The second is to set the maximum diagonal difference as a fraction of the separation, which is done with the `-f` option. If both options are used, then clusters meeting either criterion will be output.

   c. Minimum cluster length. This is set by the `-l` option. By default, the length of a cluster is defined as the sum of the lengths of the MUMs within it. If the `-e` option is specified, then the length of a cluster will be the distance from the start of the first MUM in the cluster to the end of the last MUM in the cluster as measured in the reference sequence.

5. Run the program `mgaps` from the Unix command line. For this example, the user would type:

```
mgaps -l100 -f0.12 -s600 <j99vs26695.out >j99vs26695.gaps
```

The `-l` option sets the minimum cluster length to 100. The `-f` option sets the maximum diagonal difference between consecutive MUMs to 12% of the separation. The `-s` option

```
#
   131579   124441     74    none      -      -
   131654   124516     30    none      1      1
   131685   124547    100    none      1      1
   131785   124710     17      -6      0     63
   131843   124747     29    none     41     20
   131893   124797    111    none     21     21
   132296   125201     20    none    292    293
   132316   125222     16      -5      0      1
   132377   125283    182    none     45     45
   132560   125466     41    none      1      1
      .        .      .       .       .      .
      .        .      .       .       .      .
      .        .      .       .       .      .
```

Figure 10.3.3 Part of the output from program `mgaps` showing the beginning of one of the clusters of MUMs. As before, the FASTA header lines from the query file are reproduced and clusters for the same query are separated by lines containing a single # character. The first three columns are the MUMs. The fourth column indicates any overlap between MUMs. The final two columns are the number of characters between the start of the MUM and the end of the preceding MUM in the reference and query sequence, respectively.

sets the maximum separation between consecutive MUMs to 600. Input is taken from the file `j99vs26695.out` *(the output of the preceding step) and output is directed to the file* `j99vs26695.gaps`*. The output file has the FASTA header lines from the input file separating the clusters for each query string and its reverse complement. Different clusters for the same query string are separated by a line containing a single pound (#) character. The beginning of one of the clusters of MUMs in file* `j99vs26695.gaps` *is shown in Figure 10.3.3. The first three columns are the same format as the input file: the start position of the MUM in the reference string, the start in the query string, and the length of the MUM. The fourth column is an indication of whether the MUM overlaps the preceding MUM in the cluster. If not, the word* `none` *appears in this column; otherwise, the number of characters of overlap is indicated as a negative number (see the next section for an example). The final two columns show the number of characters between the MUM and the preceding MUM in the cluster—the fifth column is in the reference sequence, the sixth is the query sequence.*

Compute alignments in regions between MUMs in clusters

6. Determine the error-rate parameter for computing alignments. This is set by the `-e` option of program `combineMUMs`.

   When computing alignments in gaps between consecutive MUMs in a cluster, the program computes the alignment from the ends of the MUMs bounding the gap. The alignment stops when the error rate exceeds the value specified, and the cluster is split into separate clusters at that point.

7. Run the program `combineMUMs` from the Unix command line. For this example, the user would type:

```
combineMUMs -x -e0.10 -W j99vs26695.witherrs hp26695.seq
hpj99.seq j99vs26695.gaps >j99vs26695.align
```

   The `-e` *option sets the alignment error rate to 10%: when the number of errors in the alignment exceeds 10% of the length of the alignment, the alignment is aborted. The reference sequence is* `hp26695.seq`*, the query sequence is* `hpj99.seq` *as before, and* `j99vs26695.gaps` *is the output from the prior step. The* `-w` *option specifies the name (*`j99vs26695.witherrs` *in this example) of an additional output file described below. The alignment output is directed to the file* `j99vs26695.align`*. The* `-x` *option*

```
#
  131579   124441     74    none      -      -
  131654   124516     30    none      1      1
     Errors = 1
A: gtgtattaatggtcttttgtt
B: gtgtattaattgtcttttgtt
             ^

  131685   124547    100    none      1      1
     Errors = 1
A: taactctatcgcactcttatt
B: taactctatctcactcttatt
             ^

  131785   124710     17      -6      0     63
     Errors = 63
A: tgtttgtttt----------------------------------------------------
B: tgtttgtttttttgtctttcttgttctaactctatctcactcttatttgtcttctgtcttt
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

A: -------------ctgtctttct
B: cttgttctgttttctgtctttct
   ^^^^^^^^^^^^^

  131843   124747     29    none     41     20
     Errors = 24
A: tcttgttctagttctatcccactcttgttcgctctatccctagcgttttcagcttccttt
B: tcttgttctaactttatcccactctt---------at---t----t-----gcttccttt
           ^^ ^            ^^^^^^^^^  ^^^ ^^^^ ^^^^^

A: t
B: t

  131893   124797    111    none     21     21
     Errors = 3
A: ctagttctatctgtttatcagtatcaccagcgctacaagcg
B: ctagttctatttgtttatcaatatcaccagtgctacaagcg
           ^          ^          ^

  132296   125201     20    none    292    293
     Errors = 36
A: ctttatcgtctaattctttcgctttcttgcacgcaatatgggtcaatactactaatgccg
B: ctttatcgtccaattcttttgccctcttgcacgcaatatgggtcaatactactaatgcca
           ^         ^  ^^                                    ^
A: cactgcttttcttggggtgtttttttacaagtcctctaactctctttgcaatttttgtaa
B: cactgcttttcttgggatgtttttttcacaagttctctaactttctttgcaatttttgtaa
                ^              ^          ^          ^
A: aaaacccaccaactgatctgattatatttgtaatggcgctcacttgtttaaaatgagtct
B: aaatcccaccaattgatcggagcgtatttgtaatggcactcgcttgtttaaaatgagtct
     ^         ^     ^  ^^^             ^   ^
A: ctttatttgcctttgtattagccatctctgtattcttgccaaccttatttgtttttcctg
B: ctttacttgcctctgtgttagctgtctctgtgtttttgccaactttatttgtttttactg
      ^      ^  ^      ^^        ^  ^         ^            ^
A: tttttactgattccattaccaacctttytttaamaaacaatcttactttwcaaaatyata
B: tttttactgattccattaccaacctttttttcaaaaaacaatcttacccatcaaaactata
                           ^  ^  ^            ^^^^     ^^
A: tc-atggaaacta
B: tcaatggaaacta
    ^

  132316   125222     16      -5      0      1
     Errors = 1
A: atacgaaaaa-tccactcaag
B: atacgaaaaaatccactcaag
           ^

  132377   125283    182    none     45     45
     Errors = 12
A: caagttagattgatttatttattatcaatgttcagttaagccatgttcttttagttcttt
B: caagttagatcaacccctttattatcaacgtgcagttaagccatggtctttcaactcttt
           ^^ ^^^^           ^  ^              ^     ^ ^^
A: ttcta
B: ttcta

  132560   125466     41    none      1      1
              .
              .
              .
Region:     131579 .. 139347        124441 .. 132253         301 / 7813
3.85%
```

Figure 10.3.4 Part of the output from program `combineMUMs` showing the beginning of the same cluster as in the previous figure. Note the alignment above for MUM `131785 124710 17`, which overlaps the preceding MUM by 6 characters (indicated by the `-6` in the column that has `none` when there is no overlap). The overlap can be seen in that the last 6 characters before the gap match the last 6 characters in the gap. The phenomenon of overlapping MUMs generally indicates a variation in the number of occurrences of a tandem repeat in the two sequences. In this instance, the 63-character insertion occurred within a repeated run of `gtttt`, where the B sequence had one more occurrence of `gtttt` than did the A sequence.

suppresses the output of two coverage files, which simply show the percentage of N's in each cluster and the regions between them in the two input files.

The alignment output for the same cluster shown previously is shown in Figure 10.3.4. This output duplicates the information in the `.gaps` *file, but interlaces it with alignment information. Specifically, after each MUM (except the first) in a cluster, the alignment of the gap separating that MUM from the preceding MUM is shown. Each alignment begins with a line stating the number of errors in the alignment and errors are indicated by carets (*`^`*) in the alignment itself. The alignment is then shown; for clarity, it begins with the last 10 characters from the preceding MUM and includes the first 10 characters from the current MUM as a point of reference. The* `A` *sequence in the alignment is the query sequence and the* `B` *sequence is the reference sequence. Additionally, at the end of a cluster, the extent of the cluster and an overall alignment score for it is given. Note that if an alignment fails to span a gap in a cluster, then the cluster will be split. Thus, the clusters in the* `.gaps` *file will not necessarily match those in the* `.align` *file.*

An additional file, in this example named `j99vs26695.witherrs`*, is also produced by the* `combineMUMs` *program. This file is identical to the input* `.gaps` *file, with the exception that a seventh column is added, which indicates the number of errors in the alignment between the MUM and its predecessor. If an alignment fails to span a gap (because of the error-rate threshold or because the gap is too large), no value is reported in the seventh column and there is simply a dash there. The name of this file is specified by the* `-W` *option. If this option is not used, the default name of this file is* `wither-rors.gaps`.

Running MUMmer via the Shell Script

As an alternative to performing each of the above steps separately, a Unix shell script named `run-mummer2.csh` is provided with the system. It is invoked with three parameters: the reference file name, the query file name, and a tag for outputs. In the example above, the user would simply type:

```
run-mummer2.csh hp26695.seq hpj99.seq j99vs26695
```

This script uses only default parameter settings, however, and the user is strongly encouraged to understand and modify the options of the different commands to suit the particular situation.

NUCmer: COMPARING A SET OF SEQUENCES TO ANOTHER SET OF SEQUENCES

ALTERNATE PROTOCOL 1

The NUCmer package is a wrapper around the basic MUMmer programs that extends MUMmer by permitting the reference sequence to be a set of strings, in the form of a multi-FASTA file (*APPENDIX 1B*). NUCmer accomplishes this by first concatenating all the reference strings together into a single string, then running MUMmer on that single string, then splitting the results back to appropriate coordinates on the separate strings.

Necessary Resources

Hardware

Unix or Linux workstation. The largest program in the suite requires main memory of approximately 20 bytes per base of reference sequence plus 1 byte per base of query sequence. Thus, to compare 2 million bases of query sequence to 3 million bases of reference sequence, the computer should have at least (20 × 3 Mb) + (1 × 2 Mb) = 62 Mb of main memory.

Software

NUCmer is included in the MUMmer 2.12 package (see Support Protocol for download and installation)

Files

A multi-FASTA query file and a multi-FASTA reference file (see *APPENDIX 1B* for information on FASTA). The files used in this example are sequences extracted from alignment regions of the *H. pylori* genomes used in the Basic Protocol. File `26695parts.seq` has five 2-kb sequences extracted in order from file `hp26695.seq`, and file `j99parts.seq` has five corresponding 2-kb sequences from file `hpj99.seq` but in permuted order, with 2 sequences reversed. The positions of the sequences in the files from which they were extracted are indicated in the FASTA header lines in the files. These files can be obtained from the *Current Protocols in Bioinformatics* Web site at *http://www3.interscience.wiley.com/c_p/cpbi_sampledatafiles.htm*. The first field after the `>` on the FASTA header line of each sequence will be used to identify each sequence; therefore these field should be unique both within and between the query and reference files.

Edit and run the NUCmer script

1. Determine the desired alignment parameters. The most important parameters correspond to those used in the Basic Protocol and are set by options on the NUCmer script command line. Specifically they are:

   `-l` the minimum MUM size to use (default value, 20)
   `-g` the largest separation in the `mgaps` program (default value, 90)
   `-c` the minimum cluster length in `mgaps` (default value, 65)
   `-d` the diagonal difference fraction in `mgaps` (default value, 0.12).

   There are several additional parameters that can be set by command-line options.The `-a` *option can be used to specify the program that finds the initial matches. Possible values for this parameter are* `mummer2` *and* `max-match`*. The first is the default and finds matches as described previously.* `max-match` *finds all maximal exact matches between the query and reference sequences with no regard for uniqueness. The* `-b` *option can be used to specify the length that alignments are extended into poor-scoring regions before being abandoned. Its default value is 200. The* `-f` *option specifies that only the forward strand of the query sequence should be used. The* `-r` *option specifies that only the reverse strand of the query sequence should be used. The default behavior is to use both the forward and reverse strands. The* `-p` *option can be used to specify the file name prefix to use for outputs. The* `-o` *option directs NUCmer to automatically generate the* `.coords` *file described below and should be used in most cases.*

   This example uses the default option values in the script as downloaded, plus the `-o` *and* `-p` *options.*

2. Run the NUCmer script. For this example, the user would type:

```
nucmer -o -p nuc 26695parts.seq j99parts.seq
```

   The `-o` *option will generate the* `.coords` *output file and the* `-p` *option specifies the prefix tag to use for the output files. The last two parameters are the reference and query sequence files. These files may be masked to indicate regions within them that should not be matched. For example, the user might employ a utility such as* `dust` *to identify and mask low-complexity sequence. The format of the masking is to replace the bases to be ignored with X's.*

   The result of the script will be three files: `nuc.cluster`*,* `nuc.coords`*, and* `nuc.delta`*. The file* `nuc.cluster` *contains clusters of MUMs found between any of the query sequences and any of the reference sequences. Each cluster has a FASTA-style header line indicating which reference and which query sequence are involved and the orientations of those sequences. The file* `nuc.coords` *contains summary information about all matches found by computing alignments in cluster gaps and off the ends of clusters. The contents of this file are shown in Figure 10.3.5. The file* `nuc.delta` *contains information about alignments in the matches and is not intended for direct viewing, but is input to the* `show-coords` *and* `show-aligns` *programs.*

```
26695parts.seq j99parts.seq
NUCMER

    [S1]     [E1]  |     [S2]     [E2]  |  [LEN 1]  [LEN 2]  |  [% IDY]  | [TAGS]
================================================================================
       1     2000  |        1     2000  |     2000     2000  |    95.95  | 26695.seq1    j99.seq3
       1     2000  |     2000        1  |     2000     2000  |    93.67  | 26695.seq2    j99.seq2
     178      881  |      187      894  |      704      708  |    92.26  | 26695.seq3    j99.seq5
    1218     1905  |     1313     2000  |      688      688  |    96.08  | 26695.seq3    j99.seq5
       1     1997  |     2000        1  |     1997     2000  |    91.42  | 26695.seq4    j99.seq4
       1     1965  |        1     1981  |     1965     1981  |    91.59  | 26695.seq5    j99.seq1
```

Figure 10.3.5 Contents of file `nuc.coords`. This file summarizes every match found between a query sequence and a reference sequence. The first two columns indicate the position of the match in the reference sequence. The next two columns are the position of the match in the query sequence. Note that if the start position [`S2`] is greater than the end position [`E2`] then the match is to the reverse-complement strand of the query sequence. Columns 5 and 6 are the lengths of the matches in the respective sequences. Column 7 is the percentage of bases that match between the two sequences. Columns 8 and 9 are the tags of the sequences—these are the first fields on the FASTA header lines of the sequence in the input files. Optionally, in column 10 there may be a note about the match (not shown). A note of `[DUPLICATE]` means this match is identical to the one preceding it. `[CONTAINS]` indicates this match contains the preceding match within it. `[SHADOWED]` indicates this match is contained within the preceding match. `[OVERLAPS]` indicates this match shares some positions with the preceding match. These conditions occur because overlapping MUMs may fail to combine with one another.

```
26695parts.seq j99parts.seq

============================================================
-- Alignments between 26695.seq1 and j99.seq3

-- BEGIN alignment [ +1 1 - 2000 | +1 1 - 2000 ]


1          agtgattagtgattatagcatcatttttaaatttaggtataaaacacc
1          agtgattagtgattacagcatcatttttaaatttaggcataaaacgcc
                        ^                         ^      ^


50         ctcaattcaagggtttttgagtgagctttttgctcaaagaatccaagat
50         cttaaatcaagggtttttgagcgagctttttgctcaaagaatccaagat
             ^  ^               ^


99         agcgtttaaaaatttaggggtgttaggctcagcgtagagtttgccaagc
99         agcgtttaaaaatttaggggtgttaggctcagcgtagagtttgccaagc


148        tctatgcattcattgatgatgatagggttttgcgtgggcgtgaagccaa
148        tctatgcattcattgatgatgatagggttttgtgtgggcgtgaagccaa
                                        ^


197        tttcatacgctcctaagcgtaaaatcgccttttccatgctccctaatcg
197        tttcatacgctcctaagcgtaaaatcgccttttccatgctccctaatcg


246        cttgaaatcccagtcttttaaatgcggctcgatgagggcgtcaatttca
246        cttgaaatcccagtcttttaaatgcggttcgatgagagcgtcaatctca
                                     ^        ^        ^
     ⋮
```

Figure 10.3.6 Beginning of an alignment created by the `show-aligns` program for NUCmer output.

View alignments

3. Run the `show-aligns` program to view alignments between a query sequence and a reference sequence. For example, to view the alignment for the first match listed in the `nuc.coords` file, the user would enter:

```
show-aligns nuc.delta 26695.seq1 j99.seq3
```

where `nuc.delta` is the file produced in the previous step and `26695.seq1` and `j99.seq3` are the sequence tags of the reference and query sequences in the match. The beginning of the resulting alignment is shown in Figure 10.3.6.

The sequence tags are listed on the right of each entry in the `nuc.coords` *file. Since this output is 175 lines long, the user may wish to direct it to an output file by appending* `> show-aligns.out` *to the end of the above command line.*

ALTERNATE PROTOCOL 2

PROmer: COMPARING SEQUENCES USING PROTEIN TRANSLATIONS

The PROmer package is another wrapper around the basic MUMmer programs, which allows users to detect similarity between species whose DNA sequences have diverged too much, erasing most or all significant nucleotide matches. It accomplishes this by first translating the sequences into their protein equivalents in all six reading frames, then running a NUCmer-like script on these sequences to detect matches, and finally converting the results back to appropriate coordinates in the original sequences. Proteins produced by orthologous genes are usually conserved long after the nucleotide sequences of the respective genes have diverged substantially. By searching for matches in amino acid sequences, PROmer is often able to detect matches that NUCmer would not.

Necessary Resources

Hardware

Unix or Linux workstation. The largest program in the suite requires main memory of approximately 20 bytes per base of reference sequence plus 1 byte per base of query sequence. Thus, to compare 2 million bases of query sequence to 3 million bases of reference sequence, the computer should have at least (20 × 3 Mb) + (1 × 2 Mb) = 62 Mb of main memory.

Software

PROmer is included in the MUMmer 2.12 package (see Support Protocol for download and installation)

Files

A multi-FASTA query file and a multi-FASTA reference file (see *APPENDIX 1B* for information on FASTA). The files used in this example are sequences extracted from alignment regions of the *H. pylori* genomes used in the Basic Protocol. File `26695parts.seq` has five 2-kb sequences extracted in order from file `hp26695.seq`, and file `j99parts.seq` has five corresponding 2-kb sequences from file `hpj99.seq` but in permuted order, with 2 sequences reversed. The positions of the sequences in the files from which they were extracted are indicated in the FASTA header lines in the files. These files can be obtained from the *Current Protocols in Bioinformatics* Web site at *http://www3.interscience.wiley.com/c_p/cpbi_sampledatafiles.htm*. The first field after the `>` on the FASTA header line of each sequence will be used to identify each sequence; therefore these field should be unique both within and between the query and reference files.

Edit and run the PROmer script

1. Determine the desired alignment parameters. The fundamental parameters are the same as in the preceding NUCmer protocol (Alternate Protocol 1). Note that all numeric values are now in terms of amino acid residues instead of nucleic acid bases.

   `-l` the minimum MUM size to use (default value, 6)
   `-g` the largest separation in the `mgaps` program (default value, 30)
   `-c` the minimum cluster length in `mgaps` (default value, 20)
   `-d` the diagonal difference fraction in `mgaps` (default value, 0.11)
   `-b` the maximum bad alignment extension (default value, 60)
   `-a` the initial match algorithm (default value, `mummer2`)

   There are two additional parameters that can be set by command-line options. The `-m` *option can be used to specify the number of amino acid residues between stop codons that will be ignored in comparisons. If this value is 6, for example, any string of* `6` *or fewer amino acids between stop codons will be excluded from the search for matches. The default value is 8. The* `-x` *option can be used to specify which BLOSUM matrix (Henikoff et al., 2000; UNIT 3.5) will be used to compute protein alignment. Possible values are* `1`, `2`, *and* `3` *corresponding to BLOSUM 45, 62, and 80 matrices, respectively. The default value is* `2` *for the BLOSUM 62 matrix.*

   This example again uses the default parameter values in the script as downloaded, plus the `-o` *and* `-p` *options.*

2. Run the PROmer script. For this example, the user would type:

```
promer -o -p pro 26695parts.seq j99parts.seq
```

   The `-o` *option will generate the* `.coords` *output file and the* `-p` *option specifies the prefix tag to use for the output files. The last two parameters are the reference and query sequence files. These files may be masked to indicate regions within them that should not be matched. For example, the user might use a utility such as dust to identify and mask low complexity sequence. The format of the masking is to replace the bases to be ignored with X's.*

   The result of the script will be three files: `pro.cluster`, `pro.coords` *and* `pro.delta`. *File* `pro.cluster` *contains clusters of MUMs found between any of the query sequences and any of the reference sequences. Each cluster has a FASTA-style header line indicating which reference and which query sequence are involved. In addition, the header line indicates the respective orientation (positive or negative) of the two sequences and the frames of the matches. File* `pro.coords` *contains summary information about all matches found by computing alignments in cluster gaps and off the ends of clusters. The contents of this file are shown in Figure 10.3.7. File* `pro.delta` *contains information about alignments in the matches and is not intended for direct viewing, but is input to the* `show-coords` *and* `show-aligns` *programs.*

View alignments

3. Run the `show-aligns` program to view alignments between a query sequence and a reference sequence. For example, to view the alignment for the first match listed in the `pro.coords` file, the user would enter:

```
show-aligns pro.delta 26695.seq1 j99.seq3
```

   where `pro.delta` is the file produced in step 2 and `26695.seq1` and `j99.seq3` are the sequence tags of the reference and query sequences in the match. The beginning of the resulting alignment is shown in Figure 10.3.8.

   The sequence tags are listed in columns 12 and 13 in the `pro.coords` *file. This output is often long and the user may wish to direct it to an output file by appending* `>show-aligns.out` *to the end of the above command line.*

```
95parts.seq j99parts.seq
MER

 [S1]     [E1]  |     [S2]     [E2]  |  [LEN 1]  [LEN 2]  |  [% IDY]  [% SIM]  [% STP]  | [FRM]  [TAGS]
==========================================================================================================
    1     1998  |        1     1998  |     1998     1998  |    90.99    93.09     6.16  |  1  1  26695.seq1    j99.seq3
    2     1999  |        2     1999  |     1998     1998  |    93.99    95.35     4.88  |  2  2  26695.seq1    j99.seq3
    3     2000  |        3     2000  |     1998     1998  |    90.09    92.49     6.83  |  3  3  26695.seq1    j99.seq3
 1998        1  |     1998        1  |     1998     1998  |    94.44    96.25     3.75  | -3 -3  26695.seq1    j99.seq3
 1999        2  |     1999        2  |     1998     1998  |    90.69    93.54     7.06  | -2 -2  26695.seq1    j99.seq3
 2000        3  |     2000        3  |     1998     1998  |    90.69    92.94     7.13  | -1 -1  26695.seq1    j99.seq3
    1     1998  |     2000        3  |     1998     1998  |    92.50    93.85     6.67  |  1 -1  26695.seq2    j99.seq2
    2     1999  |     1999        2  |     1998     1998  |    85.16    87.56     9.00  |  2 -2  26695.seq2    j99.seq2
    3     2000  |     1998        1  |     1998     1998  |    86.21    90.85     4.12  |  3 -3  26695.seq2    j99.seq2
 1998        1  |        3     2000  |     1998     1998  |    85.76    89.81     9.00  | -3  3  26695.seq2    j99.seq2
 1999        2  |        2     1999  |     1998     1998  |    85.76    88.46     6.90  | -2  2  26695.seq2    j99.seq2
 2000        3  |        1     1998  |     1998     1998  |    94.15    96.25     0.15  | -1  1  26695.seq2    j99.seq2

   .
   .
   .
```

Figure 10.3.7 Beginning of file `pro.coords`. This file summarizes every match found between a query sequence and a reference sequence. The first 7 and last 3 columns are as in Figure 10.3.5. Column 8 [`%SIM`] is the percent similarity according to the specified BLOSUM matrix. Column 9 [`%STP`] is the percentage of codons in the match that are stop codons. Columns 10 and 11 [`FRM`] indicate the reading frame of the match in the reference and query sequence, respectively, positive for forward strand, negative for reverse strand.

```
26695parts.seq j99parts.seq

============================================================
-- Alignments between 26695.seq1 and j99.seq3

-- BEGIN alignment [ +2 2 - 1999 | +2 2 - 1999 ]


2           VISDYSIIF*I*V*NTLNSRVFE*AFCSKNPR*RLKI*GC*AQRRVCQA
            VISDYSIIF*I* *N L SRVFE AFCSKNPR*RLKI*GC*AQRRVCQA
2           VISDYSIIF*I*A*NALKSRVFERAFCSKNPR*RLKI*GC*AQRRVCQA

149         LCIH****GFAWA*SQFHTLLSVKSPFPCSLIA*NPSLLNAAR*GRQFH
            LCIH****GF WA*SQFHTLLSVKSPFPCSLIA*NPSLLNA R* RQ H
149         LCIH****GFVWA*SQFHTLLSVKSPFPCSLIA*NPSLLNAVR*ERQSH

296         *FFLTRH*KGLKRKRVGCF*SFFLLTC*KRFF*FLHYRSQTHTTIQPQP
            *FFLTRH*KGLKRKR GCF*SFFLLTC KRFF*FLHYRSQTHTTIQPQP
296         *FFLTRH*KGLKRKRAGCF*SFFLLTCWKRFF*FLHYRSQTHTTIQPQP

443         PWLEFVSPF*PLRVWHKLNNSMRVLIASKPLLPALLPALSIACSILSVV
            PWLEFVSPF* LRVWHKLNNSMRVLIASKPLLPALLPALSIACSILSVV
443         PWLEFVSPF*SLRVWHKLNNSMRVLIASKPLLPALLPALSIACSILSVV

      ⋮
```

Figure 10.3.8 Beginning of an alignment created by the `show-aligns` program for PROmer output.

MUMmer1: ALIGNING TWO SINGLE SEQUENCES

The MUMmer1 script is provided for backward compatibility with release 1.0 of the MUMmer package. This script is designed to find the single longest chain of matches between two single sequences.

Necessary Resources

Hardware

Same as the MUMmer2 protocol except that more memory is required: ~25 bytes per base of both the query and reference sequences. Thus, to compare two 2-megabase genomes will require ~100 Mb of main memory.

Software

The MUMmer1 script is included in the MUMmer 2.12 package (see Support Protocol for download and installation)

Files

A multi-FASTA query file and a single-FASTA reference file (see *APPENDIX 1B* for information on FASTA). The files used in this example are complete genomic sequences from two strains of *Helicobacter pylori*—known as 26695 and J99. These sequences can be downloaded from TIGR's Comprehensive Microbial Resource at *http://www.tigr.org/CMR*, from the NCBI at *http://www.ncbi.nlm.nih.gov/PMGifs/Genomes/micr.html*, or from the *Current Protocols in Bioinformatics* Web site at *http://www3.interscience.wiley.com/c_p/cpbi_sampledatafiles.htm*. These are the same files as in the example used for the Basic Protocol.

Run the MUMmer1 script

1. For this example, the user would type:

```
run-mummer1.csh hp26695.seq hpj99.seq mum1
```

The first two parameters are the reference and query sequence files. The third parameter is a tag name used to identify the output files. The user optionally may specify a fourth parameter `-r` *that will cause the reference sequence to be reverse complemented before any comparisons are made.*

View outputs and alignments

2. The script will produce four output files analogous to those created by the MUMmer2 script: `mum1.out`, `mum1.gaps`, `mum1.errorsgaps`, and `mum1.align`.

File `mum1.out` *contains the MUMs found by the* `mummer1` *program, one MUM per line. The values on each line are: reference start position, query start position, and length. The* `mummer1` *program differs from the* `mummer2` *program in that the matches it finds are unique in both the query and reference sequence. The minimum-length match output is determined by the* `MIN_UNIQUE_MATCH_LEN` *constant at the top of the file* `mummer1`*. The default value is 20. If the user changes this parameter, the program will need to be recompiled by typing* `make mummer1`.

File `mum1.gaps` *contains the MUMs from file* `mum1.out` *partitioned into two groups. The first group, with a header line saying* `Consistent matches`*, is the longest possible set of MUMs that occur in the same order in both sequences, known as a longest ascending subsequence. Length here means the number of bases covered in the reference sequence, not the number of separate MUMs. The second group of MUMs, indicated by a header line saying* `Other matches`*, is all MUMs not in the first group, sorted by position in the query sequence. The columns in this file have the same meaning as the columns in the* `.gaps` *file of the MUMmer2 script (see Fig. 10.3.3).*

Files `mum1.align` *and* `mum1.errorsgaps` *are analogous to the corresponding MUMmer2 files. The first shows alignments of the regions between consecutive MUMs from the* `mum1.gaps` *file (see Fig. 10.3.4); however, no alignment is computed if the region is too long. File* `mum1.errorsgaps` *recapitulates the information in* `mum1.gaps`*, but adds an additional column indicating the number of errors in each gap between consecutive MUMs.*

SUPPORT PROTOCOL

OBTAINING AND INSTALLING THE MUMmer PACKAGE

The MUMmer 2.12 package is available from the TIGR Web site at *http://www.tigr.org/software/mummer*. The package is free of charge to nonprofit researchers using it for noncommercial purposes. After filling out the online license agreement, such users will receive instructions via E-mail on how to download the system. For-profit institutions will find instructions for obtaining a license at the same Web site.

The system is contained in a compressed Unix tar file named `MUMmer2.12.tar.gz`.

Type:

```
gunzip MUMmer2.12.tar.gz
```

at a Unix command-line prompt to uncompress the file, and then type:

```
tar xf MUMmer2.12.tar
```

to create the MUMmer2.12 directory. To install the system type:

```
cd MUMmer2.12
```

to change to the base directory, and then type:

```
make install
```

to compile and build all the programs. The executable files and scripts will be placed in the base directory which can be added to the path.

Additional information is in the files `INSTALL` and `README` and in the `docs` subdirectory.

GUIDELINES FOR UNDERSTANDING RESULTS

It is important for the user of MUMmer to understand that MUMmer alignments are not exhaustive. By definition MUMs must be unique so that matches involving exact repeats in the reference sequence will not be found unless the `max-match` program is used. In most genomic comparisons, however, there are substantial regions of unique sequence that will be matched, after which repeats will be found by aligning the regions between MUMs. Even for repeats, if there is any sequence variation that makes the repeat unique, then MUMmer will find the match.

Repeats can also cause artifactual MUMs. Suppose, for example, that a query sequence contains a region of 30 consecutive A's and that a reference sequence contains multiple regions of more than 30 consecutive A's. It is likely that a few nucleotides immediately following the query poly(A) region will uniquely match the bases after one of the reference poly(A) regions and form a MUM. Thus, matches to exact repeats can become MUMs because of a few randomly matched bases around the repeat.

COMMENTARY

Background Information

Genome sequence alignment research has developed highly efficient algorithms for alignment of sequences, including BLAST (*UNITS 3.3 & 3.4*; Altschul et al., 1990) and FASTA (Pearson, 2000) systems. In 1999, as complete genome sequences increasingly became available, the authors of this unit introduced a method for efficient alignment of large-scale DNA sequences, on the order of millions of nucleotides (Delcher et al., 1999). This alignment system, which the authors called MUMmer, was capable of aligning complete bacterial genomes in <1 min on a standard desktop computer. In 2002 the authors released a new version of the system, MUMmer2, with greater functionality and more efficient implementations of core algorithms (Delcher et al., 2002).

The algorithmic design strategy of MUMmer is a three-step process. First a series of anchor matches are found. The anchor matches are then grouped together into clusters or runs that appear together in the genome. Finally, alignments are computed between matches in the runs and off the ends of the first and last match.

The core match-finding algorithms of MUMmer all take two input sequences, either DNA or proteins, and find *all* subsequences longer than a specified minimum length *k* that are identical between the two inputs. These matches are guaranteed to be maximal, in that they cannot be extended on either end without incurring a mismatch—or, stated differently, no match is completely contained in a larger match. Depending on the specific algorithm used, the matches have varying uniqueness properties. In the original MUMmer algorithm, matches are unique in both input sequences, i.e., the matching sequence occurs exactly once in each of the two input sequences. These matches are found by putting both input sequences into a data structure known as a suffix tree, and then identifying matches by traversing that tree. See Gusfield (1997) for a comprehensive explanation of suffix trees and Kurtz (1999) for details on memory-efficient implementations of them. In this case, the remarkable fact about suffix trees is that this entire matching algorithm can be accomplished in linear time and space, i.e., in time proportional to the lengths of the two sequences, the suffix tree can be built and all MUMs found.

For very long sequences, such as eukaryotic genomes hundreds of megabases long, it is impractical to store both sequences to be compared in computer memory as a single suffix tree. To facilitate comparisons between such sequences, the authors implemented a variation of the basic match algorithm that requires only one of the sequences (the reference sequence) to be stored in the suffix tree. The other sequence (the query) can then be read and matched against the suffix tree in a streaming fashion, outputting the matches as they are found (Chang and Lawler, 1994). This allows arbitrarily long and/or multiple query sequences to be matched against a single reference sequence. The only drawback to this method, however, is that matches are guaranteed to be unique only within the reference sequence. Uniqueness in the query cannot be achieved because matches are output as they are found, before the entire query sequence has finished being read, and the very same match could occur again later in the query. This semi-uniqueness can be desirable when comparing a set of queries representing fragments or a partial assembly to a reference genome. Because the fragments may overlap, it is desirable to allow matches that occur in multiple query elements.

Software design philosophy

MUMmer is specifically designed to be a suite of programs, each with a single function and simple input/output format, that can be used in various combinations. The different match-finding algorithms, described above, can be used interchangeably. The programs also lend themselves readily to being incorporated into other scripts. For example, a user can freely edit the output of the match-finding program to add or remove specific matches before further processing by the clustering and alignment programs.

In addition, it is important to note that MUMmer is distributed as source code that the user can freely modify. The system is written in standard C++. Users with programming expertise should readily be able to modify the programs to introduce variations in the algorithms or modify input/output formats.

Examples of using MUMmer

The MUMmer system has been used to make a number of significant discoveries about large-scale genome structure. Alignments of related bacterial species led to the discovery that chromosome-scale inversions are a com-

mon evolutionary phenomenon in these species, and that the inversions are nearly always symmetric about the origin of replication (Eisen et al., 2000). These inversions show up as X-shaped alignments in the dot plot of all the DNA sequences conserved between two species. The X alignments have been observed by running MUMmer to compare numerous pairs of related bacterial species.

MUMmer was used to construct alignments of the five chromosomes of the model plant *Arabidopsis thaliana*, which range in size from 17 to 29 million base pairs (Mbp), against one another. These alignments revealed the striking discovery that the entire genome appears to have duplicated recently, and over 60% of the genome as it exists today is part of large-scale duplications (Arabidopsis Genome Initiative, 2000). An earlier MUMmer-based analysis on the first two complete chromosomes had found a 5-Mbp duplication (Lin et al., 1999).

The protein-level matches of MUMmer were used to align human chromosomes to each other, searching for duplications similar to those that the authors had found in *Arabidopsis*. The authors' initial searches using DNA sequence alignments revealed no large-scale duplications in the human genome. For each human chromosome, all proteins were concatenated in order (regardless of strand) to create 24 mini-proteomes. MUMmer was then used to align each chromosome to the entire genome at the protein level. This was very successful, revealing hundreds of small-scale and many large-scale duplications, all of them apparently quite ancient. For example, one of the most striking findings was that over 70% of human chromosome 14 appears to be an ancient duplication of part of chromosome 2 (Venter et al., 2001).

Protein-level matches also were used to find alignments between two *Plasmodium* species, *falciparum* and *yoelii,* the parasites that cause malaria in humans and rodents, respectively. These matches were used to aid the assembly of *P. yoelii*, providing a map to guide finishing efforts (Delcher et al., 2002). The `max-match` program of MUMmer was used in the analysis of *E. coli* strain O157 (Perna et al., 2001) and MUMmer was used to find regions of conserved synteny between the Celera assembly of the mouse genome and the human genome (Mural et al., 2002).

Critical Parameters

The `-l` option of program `mummer2` is probably the most important parameter. For extremely closely related sequences, the exact value of this parameter is not very critical; for example, if the inputs are two successive assemblies of the same genome using different shotgun sequence data, then the identity will be so high that MUM lengths anywhere from 20 to 100 or more will give very similar results. As the MUM length is reduced, the system will find increasing numbers of "random" matches; these are sequences that match between the two inputs but that are not shared due to common ancestry. Thus a larger value of this parameter will generate less noise. Balancing this is the fact that for more distantly related input sequences, a large value will result in too few MUMs being identified. Because the system runs in less than 30 seconds when comparing typical bacterial genomes, it is advisable to run it several times and then view the results with a simple plotting tool such as GNUPLOT (see Fig. 10.3.9; *http://www.gnuplot.info*).

The clustering parameters used in the `mgaps` program are also important. These values determine how MUMs will be combined into groups where alignments will be computed. For highly similar sequences, large overall clusters can be specified with small gap sizes and diagonal differences. If one wishes every MUM to appear in a cluster, then the minimum cluster size can be set to 1. Since a cluster will be broken if an alignment cannot be found between its adjacent MUMs, the error-rate parameter for alignments must be set low enough to allow MUMs to be connected.

Troubleshooting

The `makefile` for the system automatically attempts to find the software needed to build and run MUMmer. The necessary programs are `/bin/sh`, `perl`, `sed`, `g++`, and `csh`. If any of these components cannot be found, an error will result. One reason a component may not be found is that it may be installed in a directory that is not on your path. Users unfamiliar with Unix should see *APPENDIX 1C* or consult their system administrator about locating the required component and setting the path. If the necessary component is not on one's system, it will need to be installed; consult the system administrator on how to do this. All of the above programs are included with most distributions of Linux, and are also freely available on the internet.

One should also be sure your system has enough memory for the sizes of the input files. If MUMmer works for small input files, but not larger files, this may be the problem. If the

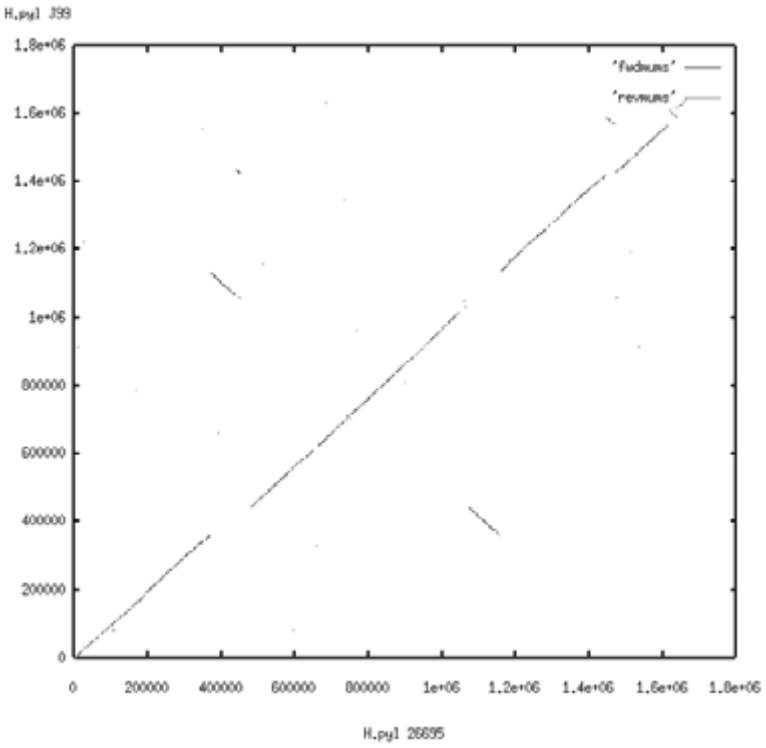


Figure 10.3.9 Dot plot of MUMs between *H. pylori* strains 26695 and J99. MUMs between both forward strands are shown in red and MUMs between the forward and reverse strands are shown in blue. Each line segment connects the start position of the MUM to the end position of the MUM. *This black and white facsimile of the figure is intended only as a placeholder; for full-color version of figure, go to http://www.currentprotocols.com/colorfigures.*

computer system is shared by multiple users, another user may be preventing you from accessing sufficient memory.

Suggestions for Further Analysis

To visualize MUMmer alignments, a simple dot plot can easily be constructed by plotting a line segment for each MUM on a two-dimensional axis. Alignments between the positive strands of each sequence can be plotted in one color with positive slope and positive-negative alignments in a different color with negative slope. An example of such a plot, made using a simple awk script and the free plotting program GNUPLOT, for the MUMs in file `j99vs26695.out` is shown in Figure 10.3.9. (The MUMmer header lines—all lines beginning with >—need to be changed to comment lines or deleted for compatibility with GNUPLOT. The script `mumdotplot.awk` does this in addition to producing the plot.) Aligned regions between two genomes usually show up as a series of colinear dots in these plots, with a slope of +1 or −1 depending on whether there has been an inversion in one of the genomes.

MUMmer is best suited for computing alignments between very long sequences with regions of high similarity. To find matches between short sequences and a large database, or to find matches between evolutionarily distant sequences, a tool such as BLAST (*UNITS 3.3 & 3.4*) is more appropriate.

PIPmaker is another system available for comparing lengthy DNA sequences (*UNIT 10.2*; Schwartz et al., 2000). It uses a hashing approach based on the BLAST algorithm, with improvements to handle large input sizes using only linear space (Chao et al., 1995). PIPmaker finds both approximate and exact alignments and also generates a very useful graphical display, showing which portions of the alignment match at different percent identities.

Acknowledgements

This work was supported in part by grants IIS-9902923 to SLS and IIS-9820497 to ALD from the National Science Foundation, and by

grant R01-LM06845 to SLS from the National Institutes of Health.

Literature Cited

Altschul, S.F., Gish, W., Miller, W., Myers, E.W., and Lipman, D.J. 1990. Basic local alignment search tool. *J. Mol. Biol*. 215:403-410.

Arabidopsis Genome Initiative. 2000. Analysis of the genome sequence of the flowering plant *Arabidopsis thaliana*. *Nature* 408:796-815.

Chang, W.I. and Lawler, E.L. 1994. Sublinear expected time approximate string matching and biological applications. *Algorithmica* 12:327-344.

Chao, K.M., Zhang, J., Ostell, J., and Miller, W. 1995. A local alignment tool for very long DNA sequences. *Comput. Appl. Biosci*. 11:147-153.

Delcher, A.L., Kasif, S., Fleischmann, R.D., Peterson, J., White, O., and Salzberg, S.L. 1999. Alignment of whole genomes. *Nucleic Acids Res*. 27:2369-2376.

Delcher, A.L., Phillipy, A., Carlton, J., and Salzberg, S.L. 2002. Fast algorithms for large-scale genome alignment and comparison. *Nucleic Acids Res*. 30:2478-2483.

Eisen, J.A., Heidelberg, J.F., White, O., and Salzberg, S.L. 2000. Evidence for symmetric chromosomal inversions around the replication origin in bacteria. *Genome Biol*. 1:research11.01-09.

Gusfield, D. 1997. Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology. Cambridge University Press, New York.

Henikoff, J.G., Pietrokovski, S., McCallum, C.M., and Henikoff, S. 2000. Blocks-based methods for detecting protein homology. *Electrophoresis* 21:1700-1706.

Kurtz, S. 1999. Reducing the space requirement of suffix trees. *Software Practice and Experience* 29:1149-1171.

Lin, X., Kaul, S., Rounsley, S., Shea, T.P., Benito, M.I., Town, C.D., Fujii, C.Y., Mason, T., Bowman, C.L., Barnstead, M. et al. 1999. Sequence and analysis of chromosome 2 of the plant *Arabidopsis thaliana*. *Nature* 402:761-768.

Mural, R.J., Adams, M.D., Myers, E.W., Smith, H.O., Miklos, G.L.G., Wides, R., Halpern, A., Li, P.W., Sutton, G., Nadeau, J., et al. 2002. A comparison of whole-genome shotgun-derived mouse chromosome 16 and the human genome. *Science* 296:1661-1671.

Pearson, W.R. 2000. Flexible sequence similarity searching with the FASTA3 program package. *Methods Mol. Biol*. 132:185-219.

Perna, N.T., Plunkett, G., 3rd, Burland, V., Mau, B., Glasner, J.D., Rose, D.J., Mayhew, G.F., Evans, P.S., Gregor, J., Kirkpatrick, H.A. et al. 2001. Genome sequence of enterohaemorrhagic *Escherichia coli* O157:H7. *Nature* 409:529-533.

Schwartz, S., Zhang, Z., Frazer, K.A., Smit, A., Riemer, C., Bouck, J., Gibbs, R., Hardison, R., and Miller, W. 2000. PipMaker—a web server for aligning two genomic DNA sequences. *Genome Res*. 10:577-586.

Venter, J.C., Adams, M.D., Myers, E.W., Li, P.W., Mural, R.J., Sutton, G.G., Smith, H.O., Yandell, M., Evans, C.A., Holt, R.A. et al. 2001. The sequence of the human genome. *Science* 291:1304-1351.

Key References

Delcher et al., 1999. See above.

This describes the original MUMmer1 algorithm.

Delcher et al., 2002. See above.

This describes the enhancements in version 2 of MUMmer, including improved efficiency, more flexible clustering and alignment options, and the ability to handle files with multiple sequences.

Gusfield, 1997. See above.

This is a comprehensive treatment of suffix trees and sequence alignment algorithms for those interested in computer science details.

Internet Resources

http://www.tigr.org/software/mummer

The MUMmer homepage.

Contributed by Arthur L. Delcher
The Institute for Genomic Research
Rockville, Maryland
and Computer Science Department
Loyola College in Maryland
Baltimore, Maryland

Steven L. Salzberg and Adam M. Phillippy
The Institute for Genomic Research
Rockville, Maryland

MultiPipMaker: Comparative Alignment Server for Multiple DNA Sequences

MultiPipMaker (*http://www.bx.psu.edu*) is a World Wide Web site for aligning multiple, related sequences and visualizing sequences conserved between them (Schwartz et al., 2003a). This unit describes the use of the MultiPipMaker server and gives an explanation of the resulting output files and supporting tools. MultiPipMaker has several strong features for use in comparison of genomic DNA sequences.

1. Many genomic sequences can be aligned simultaneously. The server accepts a maximum of 20 sequences.
2. Long genomic sequences on the order of 1 Mb each can be aligned.
3. The server generates a true, nucleotide-level multiple alignment. Additionally, pairwise alignments are shown as percent identity plots in a visual display.
4. Alignments and output are generated quickly. Alignment time is a function of length, the number of sequences, and server usage. For example, a 16-sequence multiple alignment containing a reference sequence of 1.8 Mb aligned to 15 other sequences ranging from 200 kb to 1.5 Mb (the *CFTR* locus) took 25 min.
5. The PDF output format displays genomic annotations, user-specified features, and clickable links providing additional information. The user supplies all annotations.
6. The server supports the alignments of unordered, unoriented sequences; all but the first sequence can be draft quality.
7. MultiPipMaker is appropriate for aligning sequences from any related species, or any related loci in the same species.

The Basic Protocol describes the use of the MultiPipMaker server to align multiple sequences. The Support Protocol describes the installation of the `refine` and `single_cov`, which are necessary for users of the stand-alone application. The program `single_cov` prunes overlapping regions in the initial pairwise alignments to create a crude multiple alignment. The program `refine` iteratively improves the crude multiple alignment to produce an optimal multiple alignment. The preparation of user-supplied MultiPipMaker annotation files and optional underlays, as well as installation of the `blastz` program, are described in *UNIT 10.2*, Support Protocols 1 to 5.

STRATEGIC PLANNING

The MultiPipMaker World Wide Web server is analogous to the Advanced PipMaker server (described in *UNIT 10.2*), except that up to 20 sequences can be submitted for generating a multiple sequence alignment. It uses the same file formats and annotation files described in the PipMaker protocol (*UNIT 10.2*). Multiple alignments generated by the server come in two formats: (1) a series of pairwise alignments that are stacked to show similarity among regions aligning to the same position in the reference sequence; and (2) a refined multisequence alignment comprising sequences from the submission, at nucleotide-level resolution. The pairwise alignment panels present percent identity plots (pips) that display the length and percent identity of each gap-free alignment at its location in the reference sequence. The server can align both paralogous sequences (i.e., genes that diverged after a duplication event) and orthologous sequences (i.e., genes that diverged after a speciation event), and care must be taken not to assume the type of relationship

Contributed by Laura Elnitski, Cathy Riemer, Richard Burhans, Ross Hardison, and Webb Miller
Current Protocols in Bioinformatics (2005) 10.4.1-10.4.12

based on alignments alone. All sequences must have some similarity to the reference sequence for a successful alignment. Even so, comparisons at the same phylogenetic distance will vary in the amount of conservation, depending on the locus.

When provided with user-supplied annotation files, MultiPipMaker produces alignments annotated with genomic features such as genes, the direction of transcription, repetitive elements, and CpG islands. Furthermore, colored underlays are applied to the pairwise alignment panels from `underlay` files. Features that can be colored include standard items like genes, introns, coding exons, and UTRs, or user-defined items like known regulatory regions and conserved synteny breakpoints, among others. The user defines the choice of color and the placement of the color stripe, i.e., in the upper or lower half of the panel, or both.

There are basically three simple steps in submitting an alignment request to the server: (1) prepare and submit sequence and annotation files; (2) select desired output format; and (3) retrieve results from an e-mail or WWW interface. E-mail programs often have a limit on incoming file sizes; therefore most users benefit from using the WWW option to download their results. Output files must be saved prior to the end of a 24-hr time period, after which they are deleted from the server. MultiPipMaker has a 30 min CPU time limit for normal use. Masking of repeats in the reference sequence is strongly encouraged; there is a time limit of 5 min on any unmasked submission. Experienced users wishing to align more than 20 sequences, or avoid these other limits, may want to download the alignment software and run the command-line programs themselves.

BASIC PROTOCOL

MultiPipMaker: MULTIPLE ALIGNMENT SERVER

A flowchart describing the submission of files to the MultiPipMaker server is presented in Figure 10.4.1.

Upon entering the MultiPipMaker WWW interface, users are asked about the number of sequences to submit. In response, a page appears with the exact number of required slots. The user then fills in a text box with his or her e-mail address and submits the appropriate files using the Browse buttons.

The user is asked to provide a label for each sequence; these names are limited to a single word that contains only letters, digits, and the underscore character. In addition, each sequence file should begin with a unique FASTA header line for use in reporting the full names of the sequences.

The only required files are the FASTA sequences. The `mask` file (e.g., RepeatMasker output; *UNIT 4.10*) is recommended for comparisons between the same species, or any species in which interspersed repeats align (e.g., mammals). The `exons`, `underlay`, and `annotations` files are exactly as described for PipMaker, in Support Protocols 2 to 4 of *UNIT 10.4*.

When these choices are submitted by clicking the Submit button, an acknowledgement page appears listing the selected options and information about the uploaded files (Fig. 10.4.2). This text is also returned later as a separate file with the output.

The alignment process begins with a set of pairwise alignments generated by the `blastz` program (Schwartz et al, 2000, 2003b). Next, the `refine` program recursively reduces the subalignments and gaps into an optimal multiple alignment relative to the reference sequence. This process generates the raw nucleotide level alignments. Lastly, the PDF visual display is built, including annotations and pairwise percent identity plots.

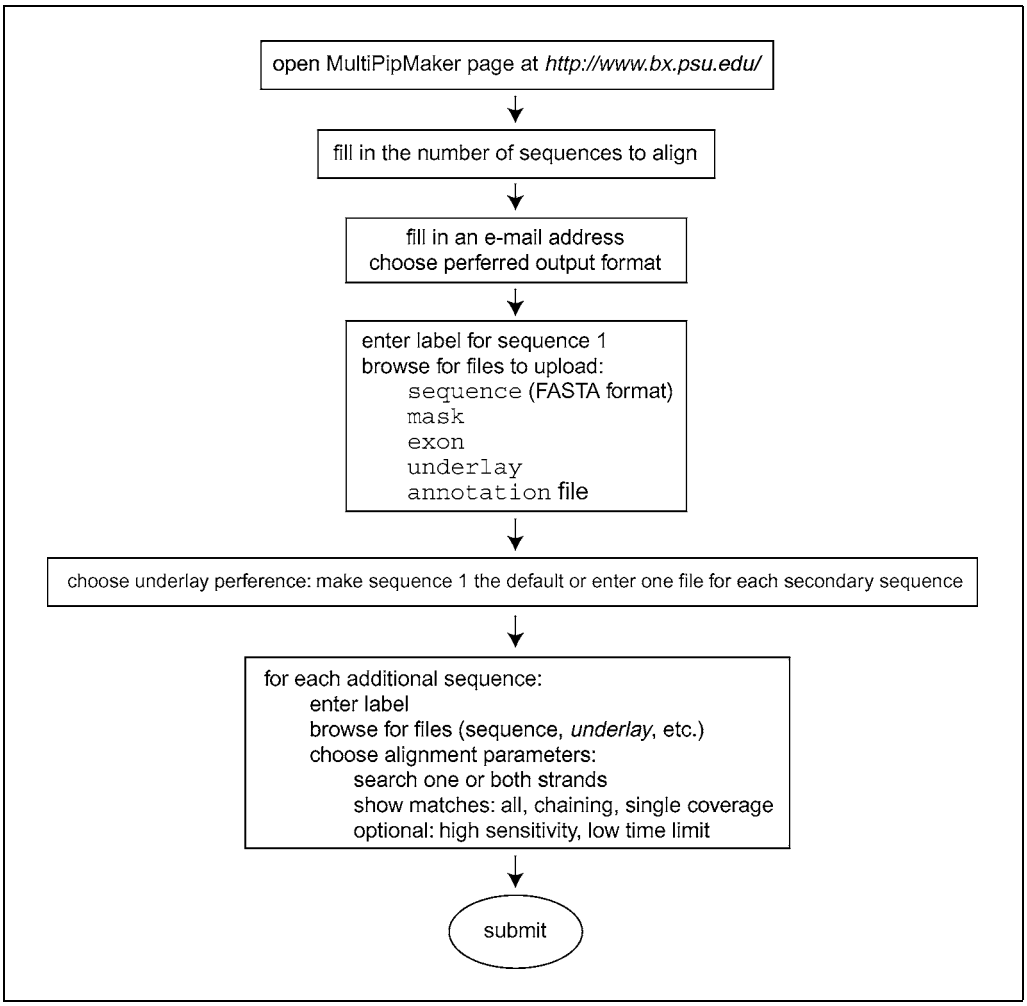


Figure 10.4.1 Flow chart outlining the steps for using MultiPipMaker.

Necessary Resources

Hardware

Any computer with Internet and e-mail access

Software

A Web browser is necessary to access the server's submission form.

An e-mail account is necessary to receive notice that a MultiPipMaker job has finished. The message will point to a server page containing the results if the default output format is selected, or will contain the output files as attachments, but the server page is the optimal way of receiving MultiPipMaker files because most e-mail servers place limitations on incoming mail sizes. This option gives users the ability to download their output for a 24-hr time period.

Output files can be viewed with a PDF viewer such as Adobe Reader (available for free from *http://www.adobe.com*). A verbose text output format for the nucleotide level alignment is viewable in any text-editing program.

Files

Formatted files for MultiPipMaker are the same as for Advanced PipMaker (*UNIT 10.2*).

Sequences: Sequences submitted to the MultiPipMaker server for alignment should be in FASTA format (*APPENDIX 1B*) and in plain text (specific formatting

```
email: user@location.edu

exons: 227 lines
underlay: 238annotation: 0 lines
seq01mask: 2885 lines
make a pip: yes
nucleotude level view: no
output pdf: yes
output text: no
retrieve with www: yes
number of sequences: 16

seq01name: human_1
seq01data: length 1877426: >human_T1 (UCSC April 2002 chr7:115977709-117855134)

seq02name: baboon
seq02data: length 1508613: >baboon_T1 (1 - 1508613)
seq02strand: 2
seq02chain: yes
seq02sensitivity: normal
seq02underlay: 0 lines

seq03name: chimp
seq03data: length 1417858: >chimp_T1 (1 - 1417858)
seq03strand: 2
seq03chain: yes
seq03sensitivity: normal
seq03underlay: 0 lines

seq04name: macaque
seq04data: length 1498014: >macaque_T1 (1 - 1498014)
seq04strand: 2
seq04chain: yes
seq04sensitivity: normal
seq04underlay: 0 lines

seq05name: cat
seq05data: length 1457638: >gi|14572136 (1 - 1457638)
seq05strand: 2
seq05chain: yes
seq05sensitivity: normal
seq05underlay: 0 lines

seq06name: dog
seq06data: length 1396669: >dog_T1 (1 - 1396669)
seq06strand: 2
seq06chain: yes
seq06sensitivity: normal
seq06underlay: 0 lines
```

Figure 10.4.2 Textual summary of a MultiPipMaker submission. This returned acknowledgement page indicating that the job has been accepted is also included with the output. The information details the length of each sequence and displays a portion of the FASTA header line. It also gives information such as the options selected and the order of the sequences in the alignment.

introduced into Microsoft Word `.doc` files is not readable) The letters `A`, `G`, `C`,`T`, `N`, and `X` are all acceptable characters; dashes (`-`) are not. The first sequence should be in one contiguous piece. All others can be unassembled, as long as each contig has a unique FASTA header line.

`Repeats` file (recommended; see *UNIT 10.2*, Support Protocol 1 for a detailed description)

`Exons` file (optional; see *UNIT 10.2*, Support Protocol 2)

`Underlays` file (optional; see *UNIT 10.2*, Support Protocol 3)

`Annotations` file (optional; see *UNIT 10.2*, Support Protocol 4)

1. Point the Web browser to *http://www.bx.psu.edu/*. On the page that appears, select the link PipMaker/MultiPipMaker, then select the MultiPipMaker link on the following page.

2. Click the text box on the page that appears and enter the number of sequences to align. Click the Enter key.

   Up to 20 sequences are allowed. The number of entries cannot be changed without returning to this page; therefore, if the wrong number of sequences is entered here it will be necessary to return to this page to make the correction and then re-enter information on the subsequent submission page. Blank fields for sequences are not allowed.

3. At the top of the page that now appears, enter a valid e-mail address to be notified when the job has finished.

   Without a correct e-mail address there will be no further notification of results, so check carefully.

4. A few lines below this on the same page, select the radio button for the desired data retrieval option via e-mail or over the Web.

   For data retrival via the Web, the URLs are sent via e-mail when the job has finished. The authors recommend the use of the WWW option because of limitations on the size of e-mail messages enforced by many e-mail servers.

5. Select output format(s) using the check box(es) on the subsequent lines.

   Every submission produces a multi-pip alignment, presented as a PDF file. Choices for additional output files include the nucleotide-level view of the refined alignment, which can be received in PDF format, or a machine-readable text format (in which no line breaks are placed in any sequence of the alignment). This can be very large in size; therefore the default option is to compress the file using the gzip program (GNU zip; the program needed for unzipping these files is available at http://www.gzip.org).

6. In the text fields under "Your data," enter information about the reference sequence and annotation files.

   Each sequence in the alignment must have a label. These names are restricted to a single word containing only letters, numbers, and the underscore character. Filenames can be typed in directly or selected via the Browse buttons.

   The reference sequence's `underlay` *file can be used to paint all panels in the comparison when the "use as default in pip" check box is chosen, or a separate underlay file can be entered for each pairwise comparison. Note that, in the latter case, underlay files should not be submitted for both the reference sequence and the second sequence at the same time, unless they are specific for the (nonoverlapping) upper and lower panels of the pip.*

7. In the subsequent text fields, enter information about the other sequences, with alignment options.

   Alignment parameters can be chosen for each secondary sequence. "Search one strand" aligns the sequences only in their given orientation, whereas "Search both strands" aligns the first sequence to both directions of the second sequence.

   Options are available for: "Showing all matches" (multiple hits to the same region are displayed in the pip), "Chaining" (restricts aligning elements to the same relative order and orientation in both sequences), and "Single coverage" (prevents the reference sequence from aligning to more than one position in the subsequent sequences, but does not require the same linear order of elements in the two sequences to find a match between them). For a more detailed description of these options see the Basic Protocol for PipMaker in UNIT 10.2.

Check the "High-sensitivity and low time limit box" only if appropriate. The authors suggest using this option only for very distant species comparisons such as human-Fugu. Use of this option on human-mouse comparisons will increase the total time without adding significant information.

8. Click the Submit button at the bottom of the page.

   An acknowledgement page will appear that details all of the sequences in the submission, including their labels, lengths, and the options that were chosen by the user.

SUPPORT PROTOCOL

INSTALLING AND USING STAND-ALONE `Refine` AND `Single_cov`

The preparation of the various files (`repeats`, `exons`, and `annotation`) and the generation of optional color underlays for use with MultiPipMaker are described in Support Protocols 1 to 4 of *UNIT 10.2*. The installation of the `blastz` program, which is used by the MultiPipmaker server, is described in Support Protocol 5 of *UNIT 10.2*. The protocol below describes the installation of the two additional programs `refine` and `single_cov`, which are used by the MultiPipmaker server, and are necessary only for the self-installation protocol. In brief, the sequence of events in generating the refined multiple alignment is as follows. (1) `blastz` produces a pairwise alignment, which comprises a set of local alignments. These local alignments may overlap with each other. (2) `single_cov` prunes a pairwise alignment produced by `blastz` such that each nucleotide in the reference sequence is aligned to at most one nucleotide in the secondary sequence. (3) `refine` creates a crude multiple alignment from a set of `blastz` pairwise alignments that have been pruned using `single_cov`. `refine` then iteratively improves this crude multiple alignment to produce an optimal multiple alignment. Self-installation and running of the multiple alignment programs are recommended only for experienced programmers. Users desiring graphical output formats such as PDF files for the visual inspection of annotated nucleotide-level alignments or percent identity plots should use the server.

Necessary Resources

Hardware

The authors test and use `refine` and `single_cov` on Solaris/SPARC and Linux/x86 platforms. The programs also run on OS X/Power PC and should be portable to virtually any ANSI/POSIX system, including Windows and Macintosh.

Software

The current development snapshots of `refine` and `single_cov` are available on the authors' Web site (*http://bio.cse.psu.edu*) in a `tar.gz` file. To unpack it, `tar` and `gzip` (or compatible programs) will be needed (see *APPENDIX 1C*). An ANSI-compatible C compiler and the `make` utility will be needed to compile and install it.

Files

The stand-alone versions of the `refine` and `single_cov` programs use the same `sequence` and `mask` files as the MultiPipMaker Web server

Install `refine` *and* `single_cov`

1. Unpack the tarball. For example, in Unix, type:

   ```
   zcat refine.tar.gz | tar --xf --
   ```

   then

   ```
   cd refine-source
   ```

2. Edit the `Makefile` to suit the system being used.

3. Run `make` to compile the program.

   This will create executables named `refine` *and* `single_cov`*, which do not depend on any other files. These can be copied to a* `bin` *directory or run in place.*

4. Syntax and arguments can be obtained by typing `refine` and `single_cov`, respectively. A `README` documentation file is included with the download.

Use* `blastz`*,* `single_cov`*, and* `refine` *to create a Multiple Alignment

5. In this example, a reference sequence, `ref`, and two secondary sequences, `seq1` and `seq2`, will be used. First, use `blastz` to create a set of initial pairwise alignments:

```
blastz ref seq1 > seq1.blastz
blastz ref seq2 > seq2.blastz
```

6. Use `single_cov` to produce a single-coverage set for each pairwise alignment (requiring strict adherence to the naming convention `*.align`):

```
single_cov seq1.blastz > seq1.align
single_cov seq2.blastz > seq2.align
```

7. Finally, `refine` creates the multiple alignment:

```
refine ref seq1 seq2 > alignment.refine
```

GUIDELINES FOR UNDERSTANDING RESULTS

The default output from MultiPipMaker is a MultiPip, which is a set of percent identity plots showing the positions and percent identities of gap-free alignments between each secondary sequence and the primary (reference) sequence. Because the first sequence serves as the reference sequence, most annotations, including the locations, names, and transcriptional orientations of genes, and the icons that represent repetitive elements, are specified relative to the coordinates in the first sequence. The `underlay` file for the reference sequence can be used for all of the panels, thus allowing ease of visual comparison. Alternatively, a separate `underlay` file can be submitted for each secondary sequence, in which case the underlay coordinates are specified relative to the secondary sequence. This will allow visualization of the differences between species.

To illustrate the features of a MultiPip, an alignment of the four human *HOX* clusters is shown in Figure 10.4.3. This illustrates a situation in which paralogous genomic loci can be aligned that are sequenced and annotated as part of the finished human genome, but for which precomputed alignments are not available. The *HOX* gene clusters arose through ancient gene duplication events and retain similar features and gene numbers despite having distinct identities. These homeoboxes are clustered in four complex *HOX* loci on chromosomes 2, 7, 12, and 17. A nearly one-to-one correspondence of individual *HOX* genes in different chromosomal loci suggests this locus evolved first through duplications of a founder gene, followed by duplication of the entire locus (Boncinelli et al., 1998).

The loci in the MultiPip are displayed as pairwise alignments of *HOXA* to *HOXB*, *HOXC*, or *HOXD* (Fig. 10.4.3 shows an excerpt from the larger genomic region). The panels are stacked to allow visualization of sequence conservation corresponding to the same regions of the reference sequence. Annotation bars are displayed as colored, clickable links that appear at the top of the alignment image. These links connect directly to an index page containing further textual information (such as references, written explanations, etc.) and hyperlinks to Internet sites. The annotation tracks are analogous to those used in the PipMaker output files (*UNIT 10.2*). The information and links found in the

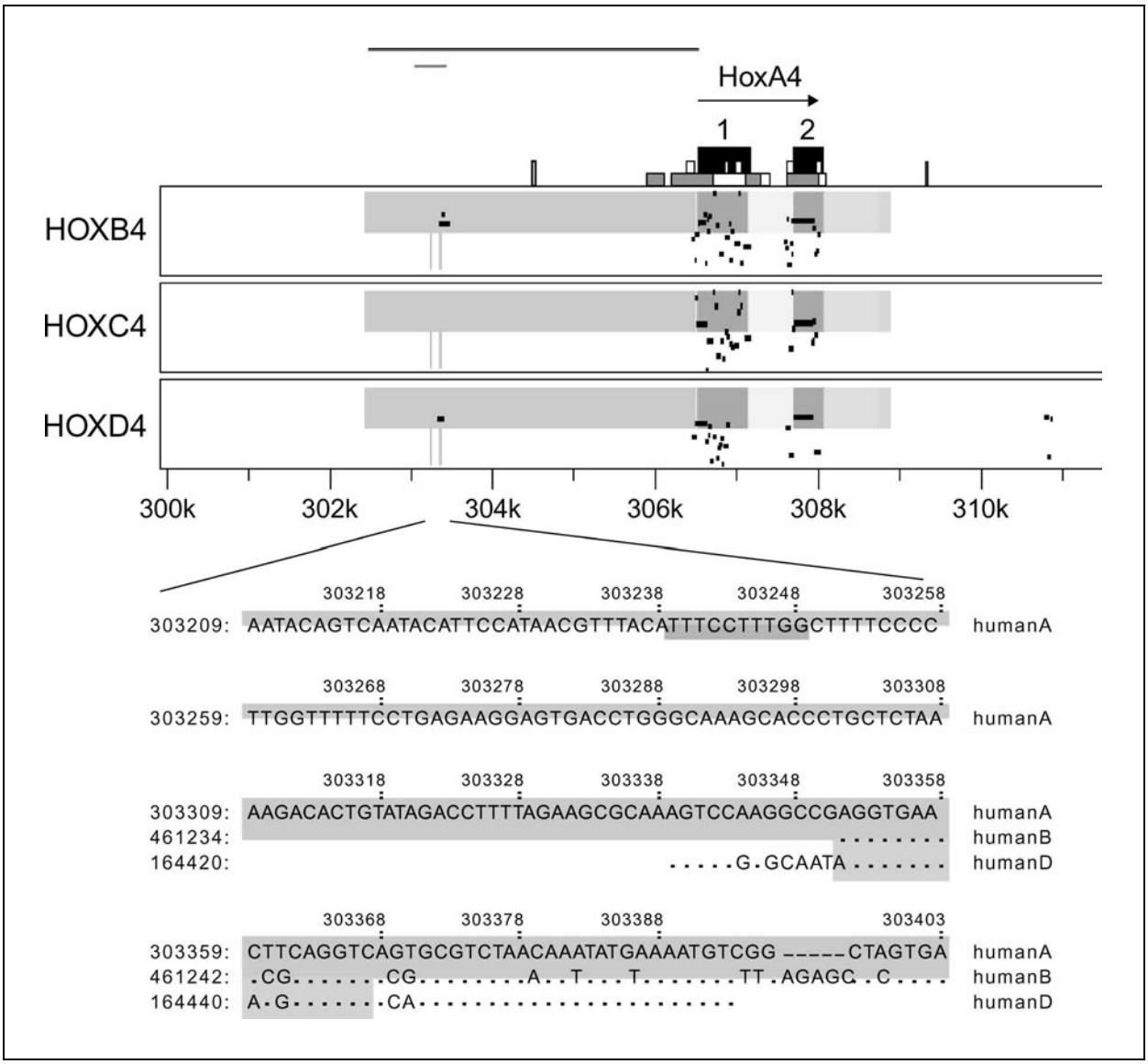


Figure 10.4.3 Alignment of the paralogous *HOX* clusters in human. The *HOXA4* locus is aligned to *HOXB*, *HOXC*, and *HOXD*. Each panel in the MultiPip represents a pairwise alignment to the *HOXA* reference sequence. The horizontal axis shows icons representing simple repeats (vertical bars), CpG islands (low rectangles), exons (numbered black boxes), the gene name, and direction of transcription. Underlays represent annotations from the *HOXA* cluster, including characterized promoter regions (pink), UTRs represented in the EST libraries (green), coding exons (red), introns (yellow), and annotated UTRs (light orange, top half). Vertical bars in the bottom half represent experimentally characterized protein binding sites for AP2 (purple) and the RARE sequence (orange). The lower panel is an excerpt from a nucleotide-level alignment in the region of the conserved RARE and nonconserved AP2 transcription factor binding site. Conserved nucleotides are shown as dots. This black and white facsimile of the figure is intended only as a placeholder; for full-color version of figure go to *http://www.interscience.wiley.com/c_p/colorfigures.htm*.

annotation link are for optional use and are generated by the user. In this example, the links refer to genes and regulatory regions described in Locus Link (*http://www.ncbi.nlm.nih.gov/LocusLink/index.html*), PubMed (*http://www.ncbi.nlm.nih.gov/entrez/query.fcgi*), and GenBank (*http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=Nucleotide*). Gene annotations (described in the `exons` file) include a gene name, direction of transcription, and numbered exons. An option allows the user to number exons of a truncated gene, or annotate alternatively spliced exons (e.g., 1A, 1B, etc.). Additional annotations include CpG islands, which are generated automatically using the FASTA sequence file. The software identifies CpG islands based on characteristics described in

MultiPipMaker

Gardiner-Garden and Frommer (1987). Color underlays are shown within each panel as discrete background regions (specified with a start and stop coordinate, and optional upper or lower placement in the panel). Colors can be used to represent any genomic features, including coding and noncoding exons, introns, antisense transcripts, regulatory regions, transcription factor binding sites, and unsequenced segments. For more information about colors and icons in this figure, see Figure 10.2.14.

This alignment was done using single coverage, i.e., restricting the reference sequence to align in only one location in each of the secondary sequences. *HOXA4* shows similarity within the coding exons of all group 4 paralogs, and especially in the homeobox domain (located in exon 2). This alignment is gap-free for much of exon 2, indicated by the unbroken horizontal line. The first exon has undergone much greater divergence, as indicated by the dispersed horizontal lines representing small, aligned regions separated by gaps. An aligning region upstream of *HOXA*, *HOXB*, and *HOXD4* corresponds to a known functional element in the *HOXA* promoter containing the Retinoic Acid Response Element (RARE) that mediates up-regulation in response to retinoic acid treatment in cultured cells (Doerksen et al., 1996). It is functional in the paralogous human *HOXD* gene (Morrison et al., 1996). This element is highlighted by an orange vertical stripe at position 303 kb of the *HOXA* locus. The nucleotide-level alignment below the MultiPip shows the region as a conserved motif that has undergone minimal changes in two other *HOX* clusters. The AP2 binding site, identified by the purple underlay (Doerksen et al., 1996), does not align with *HOXB*, *HOXC*, or *HOXD*.

Additional multiple alignment examples are available online at *http://www.bx.psu.edu* following the links to MultiPipMaker.

COMMENTARY

Background Information

Multiple sequence alignment is a highly regarded method of analyzing sequences to find functional regions. Precomputed, whole-genome alignments are available in several genome browsers including Ensembl (*http://www.ensembl.org*) and UCSC (*http://www.genome.ucsc.org*). Furthermore, additional tracks that show results from algorithms specifically designed to analyze alignment patterns or to compute the significance of conserved nucleotides provide quantitative scores for the genomic sequences (Elnitski et al., 2003; Siepel and Haussler, 2003; Kolbe et al., 2004). Thus, a tremendous amount of information is currently available through the World Wide Web.

No alignments have been precomputed for the vast majority of interesting comparisons, including paralogous sequences examined in the same species or across species. For instance, the evolutionary history of duplicated regions can be traced by examining their patterns of divergence both within a species and between species (Tufarelli et al., 2004). Researchers therefore need reliable tools to align multiple sequences and interpret the results themselves. Furthermore, researchers generating their own sequences for loci in multiple species will want to produce alignments of those sequences. The MultiPipMaker server and supporting tools provide the capability for individual labs to align their own sequences and evaluate the results.

A great many analyses use sequence similarity as a method for finding genes that code for proteins. However, the rules for finding functional, noncoding regions are not so well understood. These elements, present in intra- and intergenic regions, are often identified because they have a higher level of similarity than surrounding regions (Margulies et al., 2003; Bejerano et al., 2004). In the *HOX* gene cluster, human and mouse genomic sequences are very similar, including the noncoding regions. Thus human *HOX* sequences are compared to species that are more distant on the evolutionary scale, such as teleost fish, which diverged ~500 million years ago (Pough et al., 1999; Santini et al., 2003), to reveal discrete, putative functional elements that are apparent as conserved regions.

Whole-genome human-mouse-rat sequence alignments provided a data set whose evolutionary distance (50 to 100 million years; Waterston et al., 2002) was long enough to

look for meaningful patterns of conservation in known regulatory regions (Elnitski et al., 2003). A Markov model was designed to examine characteristics in alignments of these regulatory regions for the purpose of discriminating functional regions of the genome from neutrally evolving regions. The scores from the Markov model were used to create a genome-wide annotation track for predicted regulatory elements, known as "Regulatory Potential Scores." Adjusting these scores for local variations in the neutral rate of evolution added small but robust improvements to the predicted regions (Kolbe et al., 2004). Alternate ways to increase the accuracy of a prediction include increasing the number of species in a comparison and/or using large phylogenic distances (e.g., human-Fugu).

As many as 36 species have been sequenced in the region orthologous to the 1.5-Mb human *CFTR* locus (Thomas et al., 2003). This flagship region represents the future trend of having a great depth of sequence over a genomic locus. For example, the ENCODE project (*http://www.genome.gov/10005107*) is targeting 1% of the human genome for sequencing and in-depth investigation. Orthologous sequence data will be generated in many other species as the project continues. With the ultimate goal of finding all functional elements in the human genome, the ENCODE participants will certainly make use of multiple alignment tools such as the MultiPipMaker server, knowing that sequence conservation can be a guide to finding functional, noncoding elements in genomic DNA.

Critical Parameters and Troubleshooting

Users should not interpret the presence of an orthologous relationship from any aligned region. These regions may be related by duplication events, both within and between species. Users should also be cautious when inferring relationships between two secondary sequences that align to the same position in the reference sequence.

As mentioned earlier, the number of sample entries requested on the first page of the MultiPipMaker server can only be changed from that page. Labels describing an entry must be limited to a single word. Users should also avoid mistakes that result in an error message, e.g., annotation files should be in the correct format and contain coordinates that fall within the length of the sequences. Note that all secondary sequences must align with the reference sequence somewhere; if one or more sequences that do not align at all are supplied, this is assumed to be a mistake and results in an error message.

Suggestions for Further Analysis

Complementary tools

Complementary programs that are useful for analyzing nucleotide-level alignments include `subalign`, `multi_pat`, `infocon` (Stojanovic et al., 1999), and `tffind` (Schwartz et al., 2003a), which can be found at *http://pipmaker.bx.psu.edu/pipmaker/tools.html*. All four programs read the raw text files generated by the MultiPipMaker server (i.e., verbose output).

The tool `subalign` extracts the corresponding regions from each sequence in an alignment that falls within a user-specified range of coordinates. The program `multi_pat` finds conserved regions that match a user-specified nucleotide pattern; for instance it can identify a conserved consensus sequence that is not represented by a position-weight matrix. It has an option to search for the reverse complement of a sequence as well. The program `infocon` searches the alignment file for all conserved regions that match a user specified length and threshold score (known as information content), and returns the sequences ranked by score and size.

The program `tffind` identifies matches to position-weight matrices (PWMs) for transcription factor binding sites in any number of sequences in an alignment. The program sequentially searches through each sequence in the alignment, recording hits to the PWM that score above a specified threshold. It then determines which recognition sites correspond to more than one species and fall within aligning coordinates. Output from the program includes the name of the matrix, the orientation of the hit on each sequence (i.e., top strand or reverse complement), and the score of the hit relative to the most commonly recorded recognition sequence. `tffind` also allows the user to search for clusters of putative transcription factor binding sites within a multiple sequence alignment. Users have several options to choose from, including a factor name or consensus sequence, the distance between clustered sites, the minimum number of aligned sequences that must match the motif, minimum threshold score, and range coordinates for limiting the search. The user can also exclude categories of regions from the search, e.g., coding regions or repetitive elements. Documentation for `tffind` is available at the PipMaker Web site (*http://www.bx.psu.edu*).

Literature Cited

Bejerano, G., Pheasant, M., Makunin, I., Stephen, S., Kent, W.J., Mattick, J.S., and Haussler, D. 2004. Ultraconserved elements in the human genome. *Science* 304:1321–1325.

Boncinelli, E., Somma, R., Acampora, D., Pannese, M., D'Esposito, M., Faiella, A., and Simeone, A. 1988. Organization of human homeobox genes. *Hum. Reprod.* 3:880–886.

Doerksen, L.F., Bhattacharya, A., Kannan, P., Pratt, D., and Tainsky, M.A. 1996. Functional interaction between a RARE and an AP-2 binding site in the regulation of the human *HOX A4* gene promoter. *Nucleic Acids Res.* 24:2849–2856.

Elnitski, L., Hardison, R.C., Li, J., Yang, S., Kolbe, D., Eswara, P., O'Connor, M.J., Schwartz, S., Miller, W., and Chiaromonte, F. 2003. Distinguishing regulatory DNA from neutral sites. *Genome Res.* 13:64–72.

Gardiner-Garden, M. and Frommer, M. 1987. CpG islands in vertebrate genomes. *J. Mol. Biol.* 196:261-282.

Kolbe, D., Taylor, J., Elnitski, L., Eswara, P., Li, J., Miller, W., Hardison, R., and Chiaromonte, F. 2004. Regulatory potential scores from genome-wide three-way alignments of human, mouse, and rat. *Genome Res.* 14:700-707.

Margulies, E.H., Blanchette, M., Haussler, D., Green, E.D., and NISC Comparative Sequencing Program. 2003. Identification and characterization of multi-species conserved sequences. *Genome Res.* 13:2507-2518.

Morrison, A., Moroni, M.C., Ariza-McNaughton, L., Krumlauf, R., and Mavilio, F. 1996. In vitro and transgenic analysis of a human *HOXD4* retinoid-responsive enhancer. *Development* 122:1895–1907.

Pough, F.H., Janis, C.M., and Heiser, J.B. 1999. Vertebrate Life. Prentice Hall, Englewood Cliffs, N.J.

Santini, S., Boore, J.L., and Meyer, A. 2003. Evolutionary conservation of regulatory elements in vertebrate *Hox* gene clusters. *Genome Res.* 13:1111–1122.

Schwartz, S., Zhang, Z., Frazer, K.A., Smit, A., Riemer, C., Bouck, J., Gibbs, R., Hardison, R.C., and Miller, W. 2000. PipMaker—a Web server for aligning two genomic DNA sequences. *Genome Res.* 10:577-586.

Schwartz, S., Elnitski, L., Li, M., Weirauch, M., Riemer, C., Smit, A., Green, E.D., Hardison, R.C., Miller, W., and NISC Comparative Sequencing Program. 2003a. MultiPipMaker and supporting tools: Alignments and analysis of multiple genomic DNA sequences. *Nucleic Acids Res.* 31:3518-3524.

Schwartz, S., Kent, W.J., Smit, A., Zhang, Z., Baertsch, R., Hardison, R.C., Haussler, D, and Miller, W. 2003b. Human mouse alignments with BLASTZ. *Genome Res.* 13:103-107.

Siepel, A. and Haussler. D. 2003. Combining phylogenetic and hidden Markov models in biosequence analysis. *In* Proceedings of the Seventh Annual International Conference on Computational Molecular Biology (RECOMB 2003) pp. 277-286. International Society for Computational Biology, La Jolla, Calif.

Stojanovic, N., Florea, L., Riemer, C., Gumucio, D., Slightom, J., Goodman, M., Miller, W., and Hardison, R. 1999. Comparison of five methods for finding conserved sequences in multiple alignments of gene regulatory regions. *Nucleic Acids Res.* 27:3899-3910.

Thomas, J.W., Touchman, J.W., Blakesley, R.W., Bouffard, G.G., Beckstrom-Sternberg, S.M., Margulies, E.H., Blanchette, M., Siepel, A.C., Thomas, P.J., McDowell, J.C., Maskeri, B., Hansen, N.F., Schwartz, M.S., Weber, R.J., Kent, W.J., Karolchik, D., Bruen, T.C., Bevan, R., Cutler, D.J., Schwartz, S., Elnitski, L., Idol, J.R., Prasad, A.B., Lee-Lin, S.Q., Maduro, V.V., Summers, T.J., Portnoy, M.E., Dietrich, N.L., Akhter, N., Ayele, K., Benjamin, B., Cariaga, K., Brinkley, C.P., Brooks, S.Y., Granite, S., Guan, X., Gupta, J., Haghighi, P., Ho, S.L., Huang, M.C., Karlins, E., Laric, P.L., Legaspi, R., Lim, M.J., Maduro, Q.L., Masiello, C.A., Mastrian, S.D., McCloskey, J.C., Pearson, R., Stantripop, S., Tiongson, E.E., Tran, J.T., Tsurgeon, C., Vogt, J.L., Walker, M.A., Wetherby, K.D., Wiggins, L.S., Young, A.C., Zhang, L.H., Osoegawa, K., Zhu, B., Zhao, B., Shu, C.L., De Jong, P.J., Lawrence, C.E., Smit, A.F., Chakravarti, A., Haussler, D., Green, P., Miller, W., and Green, E.D. 2003. Comparative analyses of multi-species sequences from targeted genomic regions. *Nature* 424:788–793.

Tufarelli, C., Hardison, R., Miller, W., Hughes, J., Clark, K., Ventress, N., Frischauf, A.M., and Higgs, D.R. 2004. Comparative analysis of the alpha-like globin clusters in mouse, rat, and human chromosomes indicates a mechanism underlying breaks in conserved synteny. *Genome Res.* 14:623–630.

Waterston, R.H., Lindblad-Toh, K., Birney, E., Rogers, J., Abril, J.F., Agarwal, P., Agarwala, R., Ainscough, R., Alexandersson, M., An, P., Antonarakis, S.E., Attwood, J., Baertsch, R., Bailey, J., Barlow, K., Beck, S., Berry, E., Birren, B., Bloom, T., Bork, P., Botcherby, M., Bray, N., Brent, M.R., Brown, D.G., Brown, S.D., Bult, C., Burton, J., Butler, J., Campbell, R.D., Carninci, P., Cawley, S., Chiaromonte, F., Chinwalla, A.T., Church, D.M., Clamp, M., Clee, C., Collins, F.S., Cook, L.L., Copley, R.R., Coulson, A., Couronne, O., Cuff, J., Curwen, V., Cutts, T., Daly, M., David, R., Davies, J., Delehaunty, K.D., Deri, J., Dermitzakis, E.T., Dewey, C., Dickens, N.J., Diekhans, M., Dodge, S., Dubchak, I., Dunn, D.M., Eddy, S.R., Elnitski, L., Emes, R.D., Eswara, P., Eyras, E., Felsenfeld, A., Fewell, G.A., Flicek, P., Foley, K., Frankel, W.N., Fulton, L.A., Fulton, R.S., Furey, T.S., Gage, D., Gibbs, R.A., Glusman, G., Gnerre, S., Goldman, N., Goodstadt, L., Grafham, D., Graves, T.A., Green, E.D., Gregory, S., Guigo, R., Guyer, M., Hardison, R.C., Haussler, D., Hayashizaki, Y., Hillier, L.W., Hinrichs, A., Hlavina, W., Holzer, T., Hsu, F., Hua, A., Hubbard, T., Hunt, A., Jackson, I., Jaffe,

D.B., Johnson, L.S., Jones, M., Jones, T.A., Joy, A., Kamal, M., Karlsson, E.K., Karolchik, D., Kasprzyk, A., Kawai, J., Keibler, E., Kells, C., Kent, W.J., Kirby, A., Kolbe, D.L., Korf, I., Kucherlapati, R.S., Kulbokas, E.J., Kulp, D., Landers, T., Leger, J.P., Leonard, S., Letunic, I., Levine, R., Li, J., Li, M., Lloyd, C., Lucas, S., Ma, B., Maglott, D.R., Mardis, E.R., Matthews, L., Mauceli, E., Mayer, J.H., McCarthy, M., McCombie, W.R., McLaren, S., McLay, K., McPherson, J.D., Meldrim, J., Meredith, B., Mesirov, J.P., Miller, W., Miner, T.L., Mongin, E., Montgomery, K.T., Morgan, M., Mott, R., Mullikin, J.C., Muzny, D.M., Nash, W.E., Nelson, J.O., Nhan, M.N., Nicol, R., Ning, Z., Nusbaum, C., O'Connor, M.J., Okazaki, Y., Oliver, K., Overton-Larty, E., Pachter, L., Parra, G., Pepin, K.H., Peterson, J., Pevzner, P., Plumb, R., Pohl, C.S., Poliakov, A., Ponce, T.C., Ponting, C.P., Potter, S., Quail, M., Reymond, A., Roe, B.A., Roskin, K.M., Rubin, E.M., Rust, A.G., Santos, R., Sapojnikov, V., Schultz, B., Schultz, J., Schwartz, M.S., Schwartz, S., Scott, C., Seaman, S., Searle, S., Sharpe, T., Sheridan, A., Shownkeen, R., Sims, S., Singer, J.B., Slater, G., Smit, A., Smith, D.R., Spencer, B., Stabenau, A., Stange-Thomann, N., Sugnet, C., Suyama, M., Tesler, G., Thompson, J., Torrents, D., Trevaskis, E., Tromp, J., Ucla, C., Ureta-Vidal, A., Vinson, J.P., Von Niederhausern, A.C., Wade, C.M., Wall, M., Weber, R.J., Weiss, R.B., Wendl, M.C., West, A.P., Wetterstrand, K., Wheeler, R., Whelan, S., Wierzbowski, J., Willey, D., Williams, S., Wilson, R.K., Winter, E., Worley, K.C., Wyman, D., Yang, S., Yang, S.P., Zdobnov, E.M., Zody, M.C., Lander, E.S., and the Mouse Genome Sequencing Consortium. 2002. Initial sequencing and comparative analysis of the mouse genome. *Nature* 420:520-562.

Key References

Schwartz et al., 2003. See above.

Describes the predecessor to the MultiPipMaker server, which is the PipMaker server. PipMaker computes and displays pairwise alignments whereas MultiPipMaker computes and displays multiple sequence alignments.

Internet Resources

http://www.bx.psu.edu

Links to alignment programs including PipMaker and MultiPipMaker, software, publications, example pages, and instructions for use of the tools and the server.

http://www.adobe.com

Home page for downloading Adobe Reader.

http://www.gzip.org

Home page for the GNUzip software package.

http://www.ncbi.nlm.nih.gov/LocusLink/index.html

LocusLink Web site containing annotations of genes and links to related resources.

http://www.ncbi.nlm.nih.gov/entrez/query.fcgi

PubMed Web site: repository of published scientific literature.

http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=Nucleotide

GenBank: ENTREZ server containing sequenced DNA from numerous sources.

http://www.genome.gov/10005107

ENCODE Project homepage. Links to information, sequences, and general information about the project.

Contributed by Laura Elnitski, Cathy Riemer, Richard Burhans, Ross Hardison, and Webb Miller
The Pennsylvania State University
University Park, Pennsylvania

Using Galaxy to Perform Large-Scale Interactive Data Analyses

James Taylor,[1] Ian Schenck,[2] Dan Blankenberg,[2] and Anton Nekrutenko[2]

[1]New York University, New York, New York
[2]Penn State University, University Park, Pennsylvania

ABSTRACT

While most experimental biologists know where to download genomic data, few have a concrete plan on how to analyze it. This situation can be corrected by: (1) providing unified portals serving genomic data and (2) building Web applications to allow flexible retrieval and on-the-fly analyses of the data. Powerful resources, such as the UCSC Genome Browser already address the first issue. The second issue, however, remains open. For example, how to find human protein-coding exons with the highest density of single nucleotide polymorphisms (SNPs) and extract orthologous sequences from all sequenced mammals? Indeed, one can access all relevant data from the UCSC Genome Browser. But once the data is downloaded how would one deal with millions of SNPs and gigabytes of alignments? Galaxy (*http://g2.bx.psu.edu*) is designed specifically for that purpose. It amplifies the strengths of existing resources (such as UCSC Genome Browser) by allowing the user to access and, most importantly, analyze data within a single interface in an unprecedented number of ways. *Curr. Protoc. Bioinform.* 19:10.5.1-10.5.25.

Keywords: comparative genomics • genomic alignments • Web application • genomic sequences

INTRODUCTION

Most experimental biologists cannot fully take advantage of genomic data due to a formidable wall of countless and unnecessary computational issues. The goal of Galaxy (Blankenberg et al., 2007) is to solve these issues. Consider the following example: A researcher wants to identify protein-coding exons containing the highest density of SNPs. Most biologists know the three primary sources of genome-wide data for vertebrates: Entrez at NCBI (*UNIT 1.3*; Maglott et al., 2005), the Genome Browser at the UCSC (*UNIT 1.4*; Karolchik et al., 2003), and Ensembl (*UNIT 1.15*; Birney et al., 2004) at the EBI/Wellcome Trust Sanger Institute (UK). Although the three sources offer extensive information about genes, including genomic structure, gene expression profiles, and SNPs, the end user must still perform this task elsewhere—the listed resources do not provide functionality necessary to perform this analysis. Typically, this project ends up in the hands of a graduate student who might initially try to achieve this using popular desktop applications. Unfortunately, Excel (like any other desktop application) cannot handle that much data (the latest release of human genome contains over 12 million SNPs). As a result, this relatively simple task becomes a complex endeavor that may easily take weeks or months. In the authors' view this does not have to be complicated. Galaxy bridges the gap between data and analyses by allowing experimental biologists without programming experience to easily perform large scale studies from within their Web browsers.

In this unit the authors describe the functionality of Galaxy using a series of examples which correspond to the following protocols: Basic Protocol 1 covers the most fundamental features of Galaxy. Basic Protocol 2 elaborates on different types of data accepted by

Current Protocols in Bioinformatics 10.5.1-10.5.25, September 2007
Published online September 2007 in Wiley Interscience (www.interscience.wiley.com).
DOI: 10.1002/0471250953.bi1005s19

Galaxy. It also shows the user how to upload data and set data attributes. Basic Protocol 3 shows that manipulation of genomic intervals is one of Galaxy's greatest strengths. This protocol also explains all interval operations in detail. Basic Protocol 4 explains how Galaxy allows users to manipulate pairwise and multiple alignments. Basic Protocol 5 shows how Galaxy attempts to unify eukaryotic and prokaryotic genome resources in one interface. This protocol also explains how to access all sequenced bacterial genomes through Galaxy.

In addition each protocol has a corresponding screencast—a QuickTime movie clip—hosted at the author's Web site at *http://g2.trac.bx.psu.edu/wiki/ScreenCasts*. (QuickTime player is required for viewing the screencasts and can be downloaded free of charge from *http://www.apple.com*.) The screencasts correspond to the protocols as follows: Screencast 1 shows Basic Protocol 1;. Screencast 2 shows Basic Protocol 2; Screencast 3A to 3G show Basic Protocol 3; Screencast 4 shows Basic Protocol 4; Screencast 5 shows Basic Protocol 5. Additional screencasts are discussed in the Commentary.

BASIC PROTOCOL 1

FINDING HUMAN CODING EXONS WITH HIGHEST SNP DENSITY

Suppose one wants to find the top hundred protein-coding exons in the human genome with the highest density of single nucleotide polymorphisms (SNPs). Answering this question is not trivial. To do so one needs to compare all human exons to all human SNPs. To put this into perspective the current version of the human genome (hg18) includes 357,517 known exons and 12,351,941 SNPs. Galaxy is specifically designed to make such large-scale analyses fast and user-friendly. Galaxy's interface is accessible from *http://g2.bx.psu.edu* where you will see the available versions of Galaxy along with news about Galaxy and additional tutorials. In the following protocol the authors will use exons and SNPs annotated on chromosome 22 to make the corresponding screencast as short as possible. The protocol is illustrated in Screencast 1 at the Galaxy Web site (*http://g2.trac.bx.psu.edu/wiki/ScreenCasts*).

Necessary Resources

Hardware

UNIX, Macintosh, or Windows workstation connected to the Internet

Software

Internet browser that supports JavaScript (e.g., most current browsers such as Mozilla Firefox, Safari, Opera, or Microsoft Internet Explorer)

Files

None

1. Open Galaxy's homepage by pointing your Web browser to *http://g2.bx.psu.edu*.

   This opens Galaxy's "cover" page serving as a switchboard for different versions of the system and providing supporting information. An important feature of this page is the link to Galaxy screencasts.

2. Click on the USE link, which will bring up the main Galaxy server at *http://main.g2.bx.psu.edu*.

   The Galaxy interface is shown in Figure 10.5.1.

3. Click on the Account: create link.

   The center frame of the Galaxy interface will change into a form asking you to provide user account details. Although Galaxy can be used without creating an account, the authors highly recommend registering. First, having an account allows you to access your data from any machine connected to the Internet (not only the one currently used).

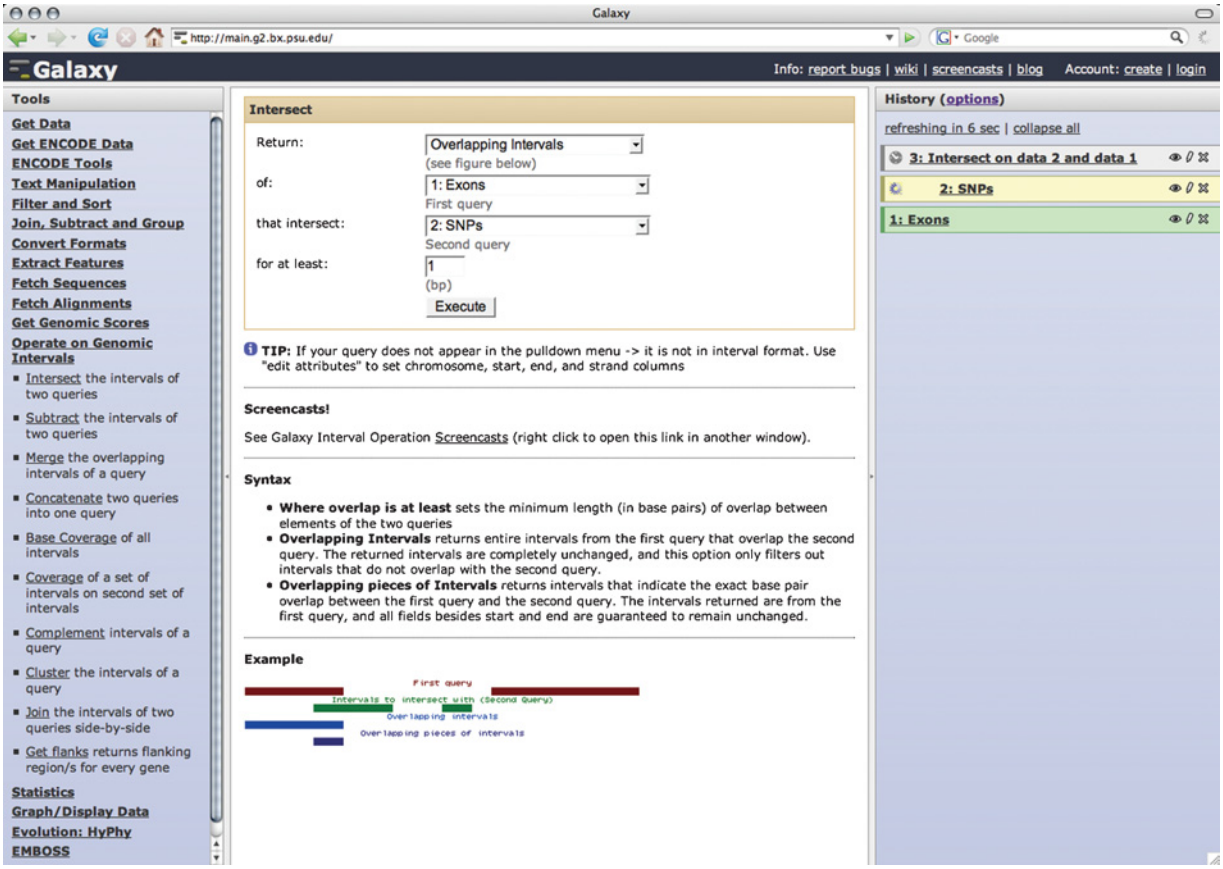


Figure 10.5.1 Galaxy interface contains four areas: the upper bar, tool frame (left column), detail frame (middle column), and history frame (right column). The upper bar contains user account controls as well as help and contact links. The left frame lists the analysis tools and data sources available to the user. The middle frame displays interfaces for tools selected by the user. The right frame (the history frame) shows data and the results of analyses performed by the user. Pictured here are three history items in different stages of completion: Green background = ready; Yellow background with rotating hourglass = computation (in this case upload) in progress; Gray with clock icon = queued (in this case it waits to be executed until history item 2 is finished uploading. This is because history item 3 will contain results of intersection between item 1 and 2). Every action by the user generates a new history item, which can then be used in subsequent analyses, downloaded, or visualized. The Galaxy history page can display results from multiple genome builds, and a single user can have multiple histories.

Second, having an account safeguards data stored in the history—accountless histories are cleared on a systematic basis. When registering for the account, note the mailing list subscription checkbox. By checking it a new user will be subscribed to the mailing list galaxy-user@bx.psu.edu. This list is used to occasionally send important announcements about new functionality and periodic maintenance. Thus, it is important to use an active e-mail address as your account name.

4. Login using your e-mail and password.
5. Click the Get Data link at the top of left pane.
6. Click the UCSC Main link.

   The UCSC Table Interface will appear in the middle pane of the Galaxy screen.

7. Import coordinates of protein-coding exons of human known genes from the UCSC Table Browser to Galaxy. Make sure the following parameters are set (Fig. 10.5.2A):

   clade: Vertebrate
   genome: Human

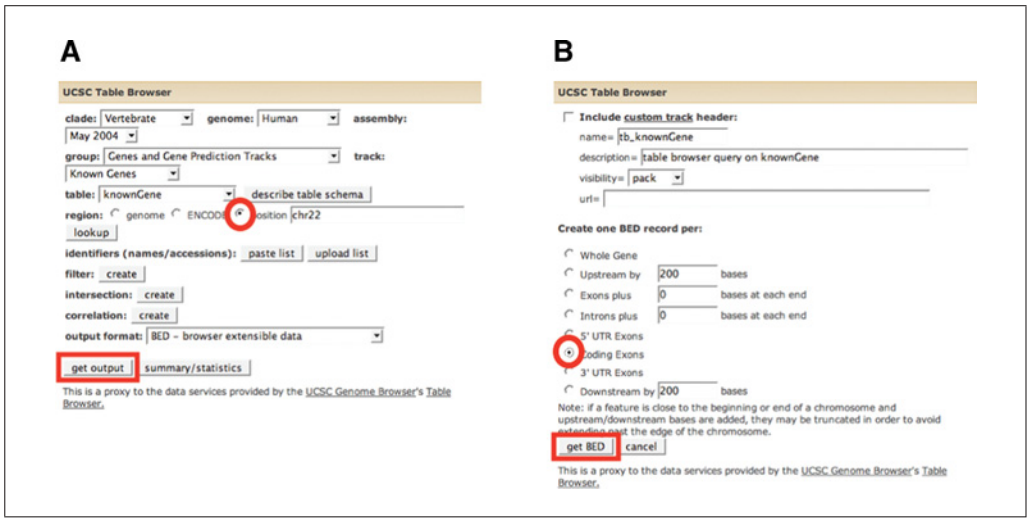


Figure 10.5.2 Uploading coordinates of protein-coding exons of known human genes from the UCSC Table browser involves two steps (A and B) described in the text.

assembly: May 2004
group: Genes and Gene Predictions Tracks
track: Known Genes
radio button Region is set to "position."

a. Type `chr22` in the position text box.
b. Click the "get output" button.

This brings up the next screen of the Table Browser interface (Fig. 10.5.2B)

c. Select Coding Exons radio button.
d. Click the "get BED" button.

This will create the first item called `UCSC: knownGene (chr22)` *in your history (the right pane of the Galaxy screen).*

8. Understand Galaxy history interface.

Figure 10.5.3 shows an example of a Galaxy's history item in expanded form. To expand an item click on the underlined text. Icons in the upper right corner (e.g., eye, pencil, and ×) as well as save, main, and test links allow one to perform tasks described in the legend for Figure 10.5.3.

9. Click the Get Data link.
10. Click the UCSC main link.
11. Import coordinates of SNPs from the UCSC Table Browser to Galaxy. Make sure the following parameters are set (Fig. 10.5.4A):

clade: Vertebrate
genome: Human
assembly: May 2004
group: Variation and Repeats
track: snp125
radio button region is set to "position."

a. Type `chr22` in position text box.
b. Click the "get output" button.

This brings up the next screen of the Table Browser interface (Fig. 10.5.4B).

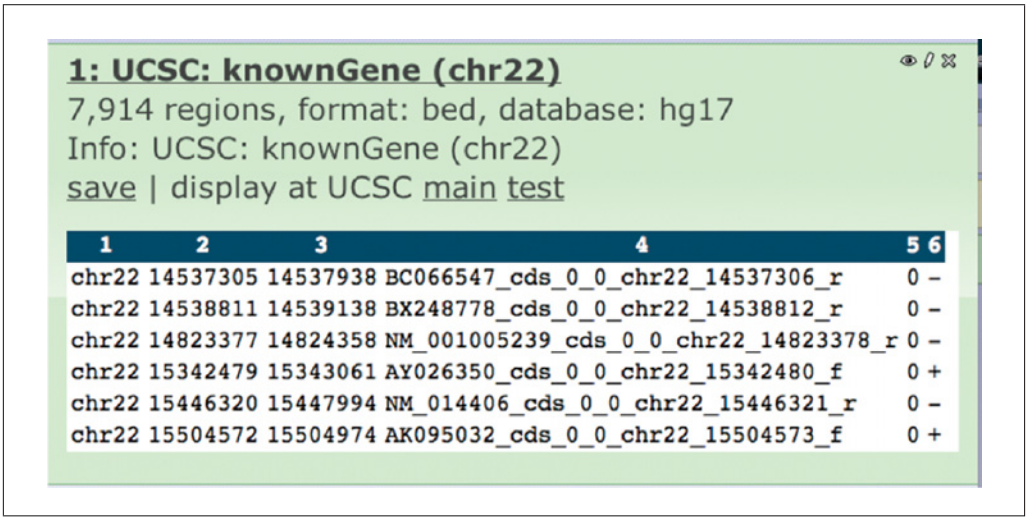


Figure 10.5.3 Close up of Galaxy history item. Clicking on links and icons trigger the following events: eye = shows entire dataset in the browser window; pencil = open metadata editor. This brings up interface in the middle pane of the Galaxy screen that allows one to edit the attributes of the current history item. For example, one may wish to give the history item a more descriptive name or change column assignments (see Basic Protocol 2); × = delete item from the history **(This action cannot be undone!)**; "save" = copy dataset on the desktop of your computer; "main" or "test" = show the dataset as a custom track in main or test version of the UCSC Genome Browser, respectively.

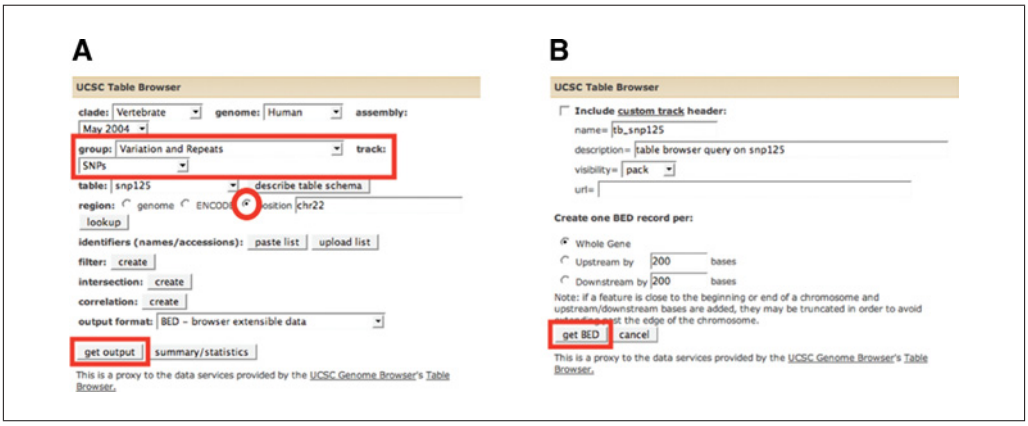


Figure 10.5.4 Uploading coordinates of single nucleotide polymorphisms (SNPs) from the UCSC Table browser involves two steps (A and B) described in the text.

c. Click the "get BED" button.

This will create the second item called `UCSC: snp125 (chr22)` *in your history (the right pane of the Galaxy screen).*

12. Click Operate of Genomic Intervals

13. Click Join to perform join operation (see Basic Protocol 3 and Fig. 10.5.14 for a detailed description of join function and other interval operations).

a. Select exons [UCSC: knownGene (chr22)] as the first dataset.

b. Select SNPs [UCSC: snp125 (chr22)] as the second dataset.

c. Make sure Join type is set to "Return only records that are joined (INNER JOIN)."

For explanation of various join options, see Figure 10.5.14.

d. Click Execute.

Because you are joining two large datasets this may take a few minutes to compute.

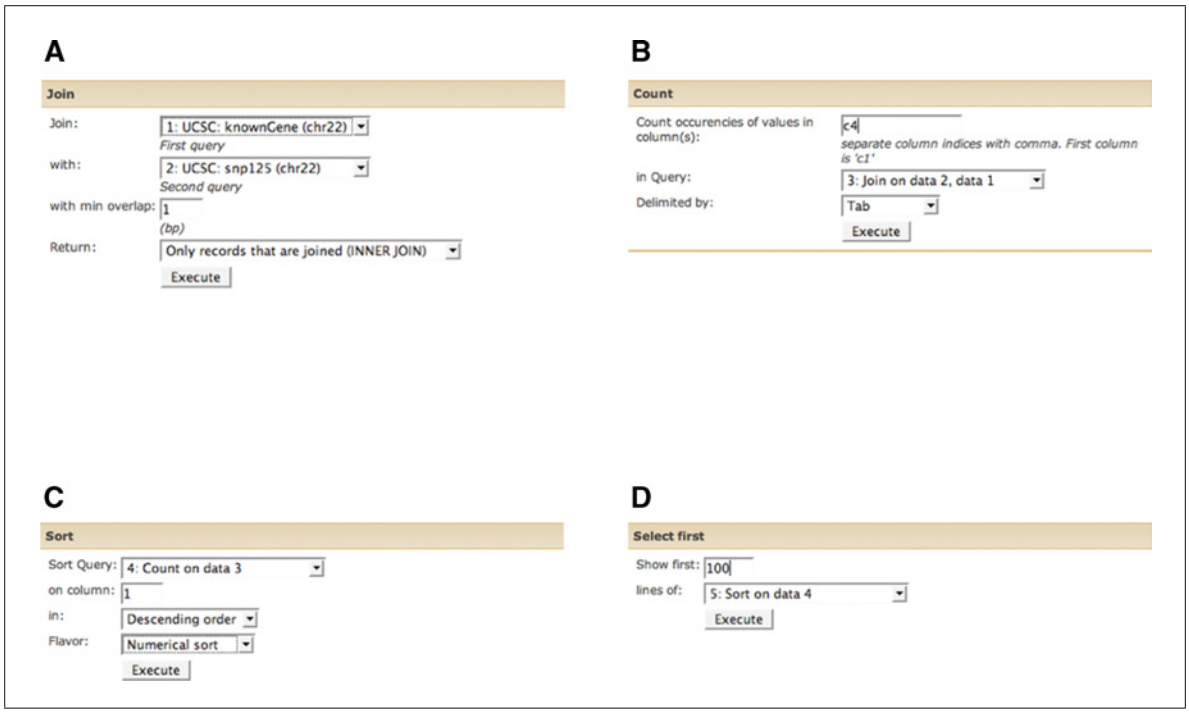


Figure 10.5.5 Parameter settings for Join (A), Count (B), Sort (C), and Select first lines (D), tools used in Basic Protocol 1 (see text for details).

The join tool allows the user to rapidly find intersections between two sets of genomic intervals. In our case we will be joining protein-coding exons and SNPs (Fig. 10.5.5A).

The result of this operation is shown in Figure 10.5.6. The first six columns represent protein-coding exons while the last six represent SNPs. The six columns are: (1) chromosome, (2) start position, (3) end position, (4) description, (5) score (always 0 in this example), and (6) strand (+ or −). Figure 10.5.6 highlights (gray shading) a single exon (located on chromosome 22 between positions 14,537,305 and 14,356,918), which contains (overlaps with) 18 SNPs. One can see that coordinates of SNPs (columns eight and nine) are always within start and end position of the exon (columns two and three).

14. Click Statistics.

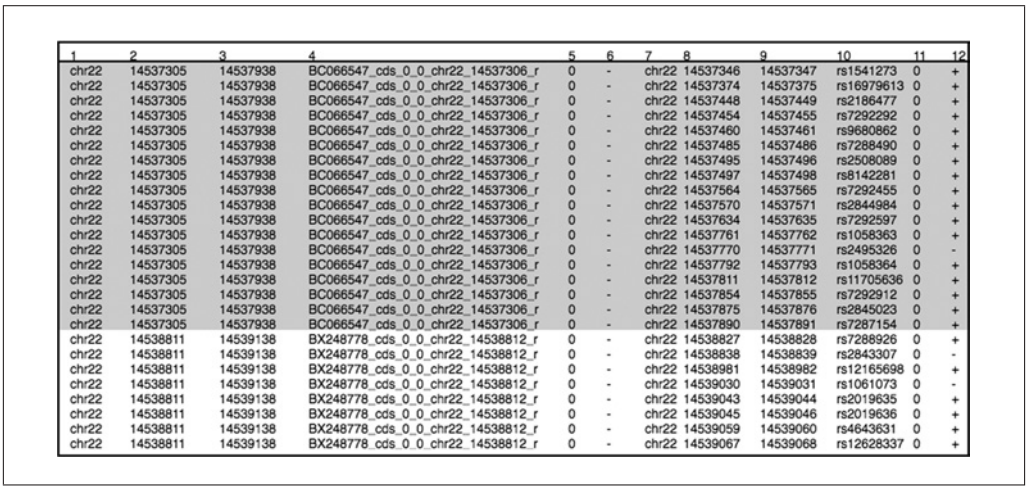
15. Count number of SNPs per exon by clicking Count (Fig. 10.5.5B).

    a. Type `c4` in the text box by Count occurrences of values in column(s).
    b. Select results of the join operations (from step 13) as In Query.
    c. Click the Execute button.

    As our goal is to produce the list of exons containing the highest number of SNPs, data produced in step 13 can be used to achieve this goal. Figure 10.5.6 shows that the number of times each exon is listed equals to the number of SNPs that exon overlaps with. Thus, by counting the number of occurrences of every exon in this dataset one can compute how many SNPs each exon overlaps with.

16. Sort results by the number of SNPs per exon (Fig. 10.5.5C).

    a. Click Filter and Sort.
    b. Click Sort.
    c. Select results of the count operation (step 15) as Sort query.
    d. Type `1` in On column text box.
    e. Set “in” to Descending order.



1	2	3	4	5	6	7	8	9	10	11	12
chr22	14537305	14537938	BC066547_cds_0_0_chr22_14537306_r	0	-	chr22	14537346	14537347	rs1541273	0	+
chr22	14537305	14537938	BC066547_cds_0_0_chr22_14537306_r	0	-	chr22	14537374	14537375	rs16979613	0	+
chr22	14537305	14537938	BC066547_cds_0_0_chr22_14537306_r	0	-	chr22	14537448	14537449	rs2186477	0	+
chr22	14537305	14537938	BC066547_cds_0_0_chr22_14537306_r	0	-	chr22	14537454	14537455	rs7292292	0	+
chr22	14537305	14537938	BC066547_cds_0_0_chr22_14537306_r	0	-	chr22	14537460	14537461	rs9680862	0	+
chr22	14537305	14537938	BC066547_cds_0_0_chr22_14537306_r	0	-	chr22	14537485	14537486	rs7288490	0	+
chr22	14537305	14537938	BC066547_cds_0_0_chr22_14537306_r	0	-	chr22	14537495	14537496	rs2508089	0	+
chr22	14537305	14537938	BC066547_cds_0_0_chr22_14537306_r	0	-	chr22	14537497	14537498	rs8142281	0	+
chr22	14537305	14537938	BC066547_cds_0_0_chr22_14537306_r	0	-	chr22	14537564	14537565	rs7292455	0	+
chr22	14537305	14537938	BC066547_cds_0_0_chr22_14537306_r	0	-	chr22	14537570	14537571	rs2844984	0	+
chr22	14537305	14537938	BC066547_cds_0_0_chr22_14537306_r	0	-	chr22	14537634	14537635	rs7292597	0	+
chr22	14537305	14537938	BC066547_cds_0_0_chr22_14537306_r	0	-	chr22	14537761	14537762	rs1058363	0	+
chr22	14537305	14537938	BC066547_cds_0_0_chr22_14537306_r	0	-	chr22	14537770	14537771	rs2495326	0	-
chr22	14537305	14537938	BC066547_cds_0_0_chr22_14537306_r	0	-	chr22	14537792	14537793	rs1058364	0	+
chr22	14537305	14537938	BC066547_cds_0_0_chr22_14537306_r	0	-	chr22	14537811	14537812	rs11705636	0	+
chr22	14537305	14537938	BC066547_cds_0_0_chr22_14537306_r	0	-	chr22	14537854	14537855	rs7292912	0	+
chr22	14537305	14537938	BC066547_cds_0_0_chr22_14537306_r	0	-	chr22	14537875	14537876	rs2845023	0	+
chr22	14537305	14537938	BC066547_cds_0_0_chr22_14537306_r	0	-	chr22	14537890	14537891	rs7287154	0	+
chr22	14538811	14539138	BX248778_cds_0_0_chr22_14538812_r	0	-	chr22	14538827	14538828	rs7288926	0	+
chr22	14538811	14539138	BX248778_cds_0_0_chr22_14538812_r	0	-	chr22	14538838	14538839	rs2843307	0	-
chr22	14538811	14539138	BX248778_cds_0_0_chr22_14538812_r	0	-	chr22	14538981	14538982	rs12165698	0	+
chr22	14538811	14539138	BX248778_cds_0_0_chr22_14538812_r	0	-	chr22	14539030	14539031	rs1061073	0	-
chr22	14538811	14539138	BX248778_cds_0_0_chr22_14538812_r	0	-	chr22	14539043	14539044	rs2019635	0	+
chr22	14538811	14539138	BX248778_cds_0_0_chr22_14538812_r	0	-	chr22	14539045	14539046	rs2019636	0	+
chr22	14538811	14539138	BX248778_cds_0_0_chr22_14538812_r	0	-	chr22	14539059	14539060	rs4643631	0	+
chr22	14538811	14539138	BX248778_cds_0_0_chr22_14538812_r	0	-	chr22	14539067	14539068	rs12628337	0	+

Figure 10.5.6 Result of joining two interval datasets. See Figure 10.5.14 and Basic Protocol 1 (step 13) for description.

f. Set Flavor to Numerical sort.

g. Click the Execute button.

A new history item produced by this operation will contain two columns. The first column shows the number of SNPs in each exon, and the second shows the exon's id.

17. Select top 100 exons from this list (Fig. 10.5.5D).

a. Click Text Manipulations.

b. Click Select first lines from a Query.

c. Set Select first to `100`.

d. Set "lines of" to the history item containing results from the step 16.

e. Click the Execute button.

After execution is finished your new history item will contain a list of the exons with the highest SNP density. CPB Screencast 1 continues past this point to show how to extract orthologous sequences from other sequenced mammals.

LOADING DATA AND UNDERSTANDING DATATYPES

BASIC PROTOCOL 2

In Galaxy, information is stored in "datasets" which are analogous to files. Datasets can be added to your history by uploading files from your computer, using one of the external data sources integrated with Galaxy (such as the UCSC Table Browser), or by running Galaxy tools. In addition to their contents, datasets also contain information that describes the dataset ("metadata"). Metadata includes the datatype, which specifies the format of the data in the file. While Galaxy will allow a dataset to be created from any file, specific tools and analysis require data to be in a particular format. Metadata also includes a name for the dataset, additional user-defined information for the dataset, the genome and build the data is associated with (if any), and possibly additional datatype-specific values.

Necessary Resources

Hardware

UNIX, Macintosh, or Windows workstation connected to the Internet

Software

Internet browser that supports JavaScript (e.g., most current browsers such as Mozilla Firefox, Safari, Opera, or Microsoft Internet Explorer)

Files

`1.2.txt` and `1.4.bed` are used in this example and can be downloaded at *http://www.bx.psu.edu/trac/local/CPB_data/*

1. Return to the main Galaxy interface by going to the URL *http://main.g2.bx.psu.edu*.
2. Create a dataset from a local file:
   a. Click Get Data in the tools frame, the submenu will expand.
   b. Click Upload File, a form will load in the center frame.
   c. Click Choose File and select `1.2.txt` (alternatively, the file can be downloaded directly by typing *http://www.bx.psu.edu/trac/local/CPB_data/1.2.txt* within the URL/Text window; Fig. 10.5.7A).
   d. Click Execute.

      The file will be uploaded and a new dataset will appear in the History frame. Click on the name of the dataset to show its attributes and the first few lines of the file. Because the file type was not specified Galaxy has detected that the file is of format "tabular."
3. Rename and link the dataset to a specific genomic build (Fig. 10.5.7B):
   a. Click the "pencil" icon next to the dataset name in the History frame. This brings up the Edit Attributes form in the center frame.
   b. In the Name text box type: `Uploaded file 1.2.txt.`
   c. From the drop down menu labeled Database select Human May 2004 (hg17).
   d. Click Submit.

      The History frame will refresh and the database attribute of your dataset will now say "hg17."

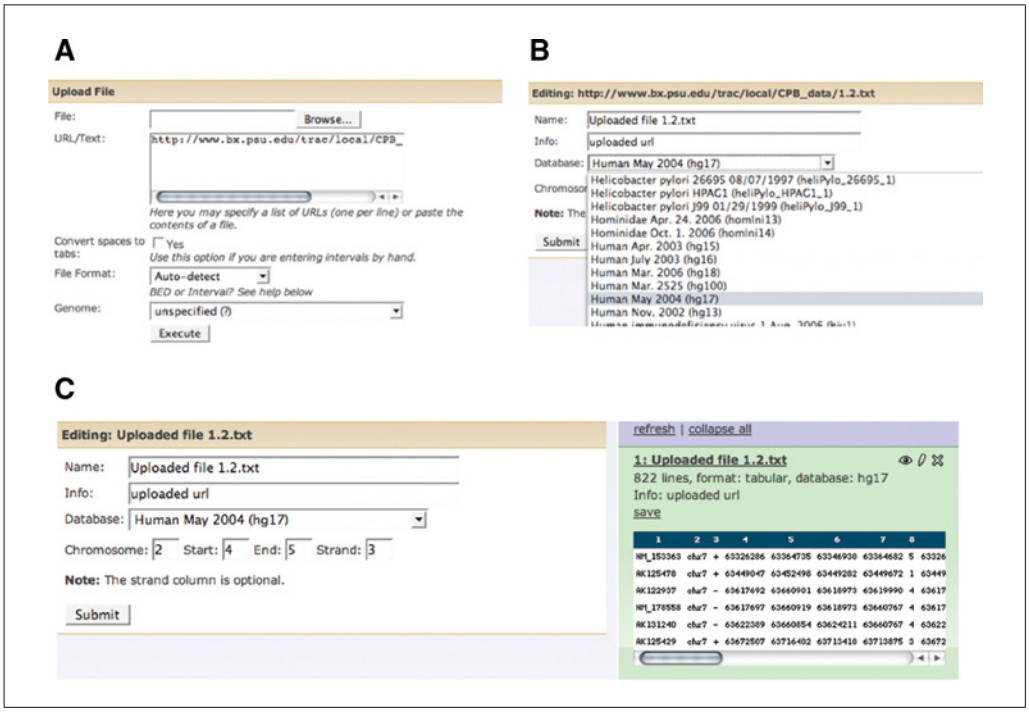


Figure 10.5.7 Uploading data (A), setting genome built (B), and editing metadata in Galaxy (C; see Basic Protocol 2 for details).

4. Specify that the dataset contains "genomic intervals" (Fig. 10.5.7C). Genomic intervals are a special subtype of tabular data in which each row of the table corresponds to a particular region of a genome. The chromosome, start, end, and strand specifying the interval can be in any column of the table, but Galaxy needs to be told which columns to use.

   a. Click the "pencil" icon again to bring up the Edit Attributes form.

   b. Specify which columns of the file contain the information specifying genomic intervals: in the Chromosome column enter `2`, in the Start column enter `4`, in the End column enter `5`, and in the Strand column enter `3`.

   c. Click Submit.

      The History frame will refresh and your dataset will now indicate "format: interval." The dataset can now be used with tools that expect interval data (see Basic Protocol 3).

5. Upload a "bed" file and allow type and metadata to be determined based on format:

   a. Click Get Data in the tools frame, the submenu will expand.

   b. Click Upload File, a form will load in the center frame.

   c. Click Choose File and select `1.4.bed`.

   d. From the Genome select list choose Human May 2004 (hg17).

   e. Click Execute.

      For "bed" files (UCSC browser extensible data format; http://genome.ucsc.edu/FAQ/FAQformat) Galaxy automatically determines correct column assignments.

      Your history will now contain a new dataset named `1.4.bed` *If you expand it you will see that its format is "bed." If you go to the Edit Attributes form by clicking the "pencil" icon you will see that the appropriate columns have been set automatically.*

6. Load data from a URL:

   a. Click Get Data in the tools frame, the submenu will expand.

   b. Click Upload File, a form will load in the center frame.

   c. In the text box labeled URL/Text enter: *http://www.ncbi.nlm.nih.gov/entrez/viewer.fcgi?dopt=fasta&sendto=t&list_uids=28380636*.

   d. Click Execute.

      Your history will contain a new dataset named for the above URL. Galaxy has automatically detected that this dataset is FASTA format sequence data.

7. Change the name and info of a dataset:

   a. Click on the "pencil" icon for the dataset created in step 5.

   b. Change the field labeled Name to contain HBB sequence.

   c. Change the field labeled Info to contain Human Beta Globin Region Sequence.

   d. Click Submit.

      Your history will refresh and the dataset name and info will be changed.

8. Display datasets:

   a. Click on the "eye" icon for the dataset created in step 5.

      The dataset will be displayed in the browser window. Click the back button in your browser to return to the Galaxy interface.

USING INTERVAL OPERATIONS

The protocol describing finding human exons with highest SNP density (Basic Protocol 1) used the Join operation to find all protein-coding exons that contain SNPs. This is just one of many interval operations offered in Galaxy, which are based on the bx-python package (J. Taylor, unpub. observ.) maintained at the Penn Sate University Center for Comparative Genomics. These include intersect, subtract, complement, merge, concatenate, cluster, coverage, base coverage, and join. Some operations are analogous to relational database queries, such as join and coverage. Other operations are analogous to set operations. Figures 10.5.8 and 10.5.9 show examples of input and output produced

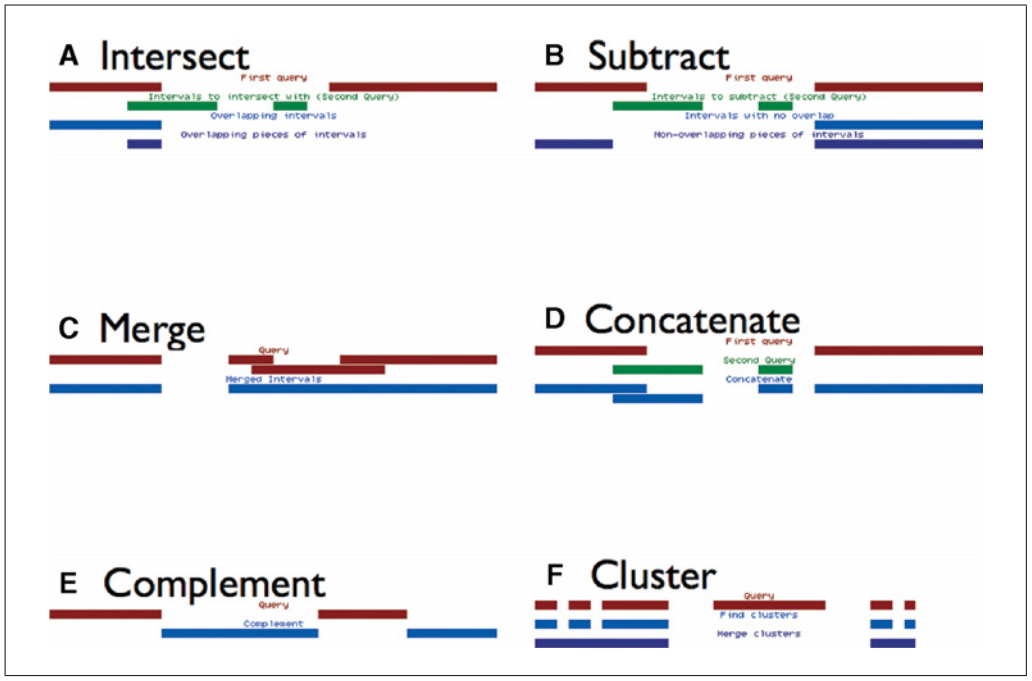


Figure 10.5.8 Graphical examples of Intersect (A), Subtract (B), Merge (C), Concatenate (D), Complement (E), and Cluster (F) interval operations supported by Galaxy.

if First query are exons:

```
chr11 5203271 5204877 NM_000518_1 0 -
chr11 5210634 5212434 NM_000518_2 0 -
chr11 5226077 5227663 NM_000518_3 0 -
```

and Second query are repeats:

```
chr11 5203895 5203991 L1MA6      500 +
chr11 5204163 5204239 A-rich     219 +
chr11 5211034 5211167 (CATATA)n  245 +
chr11 5211642 5211673 AT_rich     24 +
chr11 5226551 5226606 (CA)n      303 +
chr11 5228782 5228825 (TTTTTG)n  208 +
chr11 5229045 5229121 L1PA11     440 +
chr11 5229133 5229319 MER41A    1106 +
chr11 5229374 5229485 L2         244 -
chr11 5229751 5230083 MLT1A      913 -
chr11 5231469 5231526 (CA)n      330 +
```

the Result is the coverage density of repeats in the genes:

```
chr11 5203271 5204877 NM_000518_1 0 - 172   0.107098
chr11 5210634 5212434 NM_000518_2 0 - 164   0.091111
chr11 5226077 5227663 NM_000518_3 0 -  55   0.034678
```

For example, the following line of output:

```
chr11 5203271 5204877 NM_000518_1 0 - 172   0.107098
```

implies that 172 nucleotides accounting for 10.7% of the this interval (chr11:5203271-5204877) overlap with repetitive elements.

Figure 10.5.9 An example highlighting the functionality of coverage tool.

by individual interval operations. In the following protocol the authors use two datasets as examples. The first dataset is referred to as Exons and represents protein-coding exons annotated on chromosome 22. The second dataset is called Repeats and lists coordinates of interspersed repeats (also called *transposable elements* or simply *repeats* in the text) along chromosome 22. The next section (Prepare data) explains how to import these datasets into Galaxy interface.

Necessary Resources

Hardware

UNIX, Macintosh, or Windows workstation connected to the Internet

Software

Internet browser that supports JavaScript (e.g., most current browsers such as Mozilla Firefox, Safari, Opera, or Microsoft Internet Explorer)

Files

None

Prepare data

1a. *To retrieve exons for chromosome 22:* Click the Get Data link in the tools frame on the left-hand side. Within that menu, click the UCSC Main link. This will load the UCSC table browser in the center of the Web browser. Make sure the following parameters are set:

clade: Vertebrate
genome: Human
assembly: May 2004
group: Genes and Gene Predictions Tracks
track: Known Genes
radio button "region" is set to "position."

a. Type `chr22` in position text box.

b. Click the "get output" button.

This brings up the next screen of the Table Browser interface.

c. Select Coding Exons radio button.

d. Click the "get BED" button.

The history item should appear after a moment (10 to 20 sec) with the name: `UCSC: knownGene (chr22)`

Use the pencil icon (see Fig. 10.5.3) to change the name of the dataset to Exons.

1b. *To retrieve repeats for chromosome 22:* Click the Get Data link in the tools frame on the left-hand side. Make sure the following parameters are set:

clade: Vertebrate
genome: Human
assembly: May 2004
group: Variation and Repeats
track: RepeatMasker
radio button region is set to "position."

a. Type `chr22` in "position" text box.

b. Click the “get output” button.

This brings up the next screen of the Table Browser interface

c. Click the “get BED” button.

The history item should appear after a moment (10 to 20 sec) with the name: `UCSC: repeatMasker (chr22)`

Use the pencil icon (see Fig. 10.5.3) to change the name of the dataset to Repeats.

2. Intersect.

Intersect (Fig. 10.5.8A) allows for the intersection of two queries to be found. The intersect tool can output either the entire intervals from the first dataset that overlap the second dataset (e.g., all exons containing repeats), or it can return just the intervals representing the overlap between the two datasets (e.g., all regions that are both exonic and repetitive).

When finding entire intervals (by setting Return to Overlapping Intervals), the order of the datasets is important. The operation will output all of the intervals in the first query that overlap any interval in the second query. It can also be thought of as a filter: intervals that do not overlap any interval in the second query will be filtered out.

When finding pieces of intervals, or the regions representing the overlap between the two datasets (by setting Return to Overlapping Pieces of Intervals), the output will be the intervals of the first dataset with the nonoverlapping subregions removed.

Example: To find the whole exons that overlap one or more transposable elements (for parameter settings see Fig. 10.5.10A)

a. Click Operate on Genomic Intervals in the tool frame on the left.

b. Click Intersect under Operate on Genomic Intervals.

The Intersect tool should load in the center frame.

c. Set drop-down list labeled Return: to Overlapping intervals.

d. Set the drop-down list labeled “of:” to the Exons dataset.

This is the dataset containing coding exons.

e. Set the drop-down list labeled “that intersect:” to the Repeats dataset.

This is the dataset containing transposable elements.

f. The default minimum overlap is 1, which indicates that any overlapping regions will be output.

g. Click the Execute button.

This will launch the intersect operation. A new history item should appear in the history frame on the right labeled: Intersect on data X, data X where X is the history number of the datasets.

The new dataset generated will contain every coding exon that overlaps at least 1 base pair of a transposable element. The entire intervals from the coding exons dataset are output whenever there is an overlap with any transposable element interval.

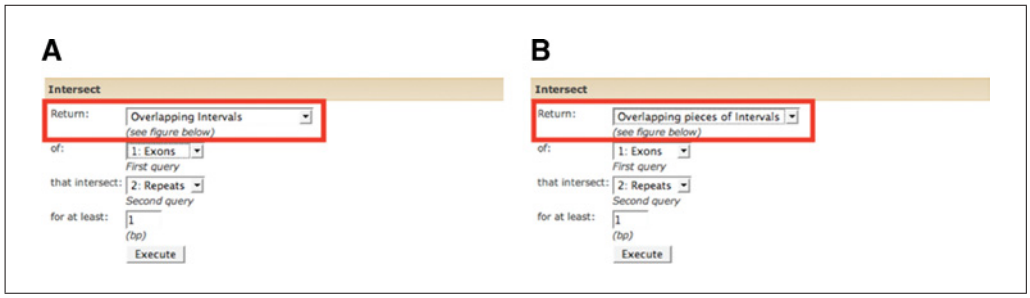


Figure 10.5.10 Settings for the Overlapping Intervals (A) and Overlapping pieces of intervals (B) version of Intersect tool.

Example: To find the regions within exons that also overlap repeats elements (for parameter settings see Fig. 10.5.10B)

a. Click Operate on Genomic Intervals in the tool frame on the left.

b. Click Intersect under Operate on Genomic Intervals.

The Intersect tool should load in the center frame.

c. Set drop-down list labeled Return: to Overlapping pieces of intervals.

d. Set the drop-down list labeled "of:" to the Exons dataset.

This is the dataset containing coding exons.

e. Set the drop-down list labeled "that intersect:" to the Repeats dataset.

This is the dataset containing transposable elements.

f. The default minimum overlap is 1, which indicates that any overlapping regions will be output.

g. Click the Execute button.

This will launch the intersect operation. A new history item should appear in the history frame on the right labeled: Intersect on data X, data X where X is the history number of the datasets.

The dataset output will be the subregions of the exons that overlap with the intervals of the repeats.

The field Where overlap is at least can be set to any integer >0. This can be used to filter out small overlaps, or to specifically find intervals with a certain amount of overlap.

3. Subtract.

Subtract (Fig. 10.5.8B) does the opposite of intersect. It removes the intervals or parts of intervals in the first dataset that are found in the second dataset (Fig. 10.5.8B). Like Intersect, Subtract can treat intervals as a whole, removing or keeping entire intervals, or it can break them apart, removing overlapping subregions.

As with arithmetic subtraction, the order of the datasets is important. The second dataset is subtracted from the first dataset. The output is a variation of the first dataset and all of its columns. When subtracting whole intervals (by setting Return to Intervals with no overlap), the output will be the intervals of the first dataset that do not overlap any part of intervals of the second dataset.

When subtracting overlapping subregions (by setting Return to Non-overlapping pieces of intervals), the output will be the intervals of the first dataset with the overlapping subregions removed.

Example: To find the exons that do not contain any repeats elements (for parameter settings see Fig. 10.5.11A)

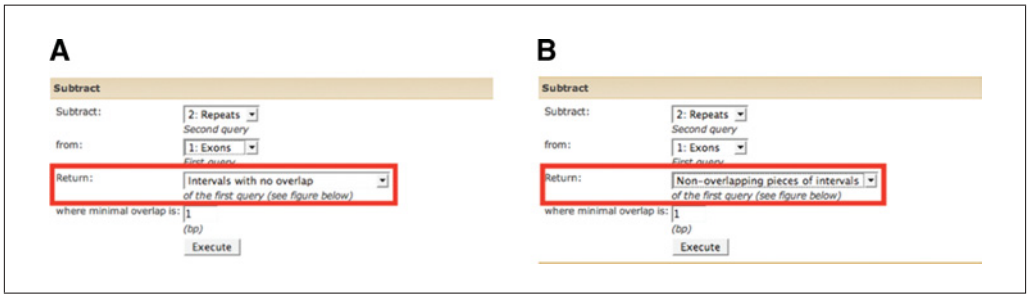


Figure 10.5.11 Settings for Intervals with no overlap (A) and Non-overlapping pieces of intervals (B) versions of Subtract tool.

a. Click Operate on Genomic Intervals in the tool frame on the left.

b. Click Subtract under Operate on Genomic Intervals.

The Subtract tool should load in the center frame.

c. Set the drop-down list labeled Subtract to the Repeats dataset.

This is the dataset containing repetitive elements to be subtracted from exons.

d. Set the drop-down list labeled "from" to the Exons dataset.

This is the dataset containing exons.

e. Make sure the drop-down list labeled Return: is set to Intervals with no overlap.

f. The default minimum overlap is 1, which indicates that any overlapping regions will be removed from the output.

g. Click the Execute button.

This will launch the subtract operation. A new history item should appear in the history frame on the right labeled: Subtract on data X, data X where X is the history number of the datasets.

The dataset output will be exons that contain no transposable elements; each exon that overlaps a transposable element is removed from the output.

Example: To remove the subregions of exons that overlap with transposable elements (for parameter settings see Fig. 10.5.11B)

a. Click Operate on Genomic Intervals in the tool frame on the left.

b. Click Subtract under Operate on Genomic Intervals.

The Subtract tool should load in the center frame.

c. Set the drop-down list labeled Subtract to the Repeats dataset.

This is the dataset containing repetitive elements to be subtracted from exons.

d. Set the drop-down list labeled "from" to the Exons dataset.

This is the dataset containing exons.

e. Make sure the drop-down list labeled Return: is set to "Non-overlapping pieces of intervals."

f. The default minimum overlap is 1, which indicates that any overlapping regions will be removed from the output.

g. Click the Execute button.

This will launch the subtract operation. A new history item should appear in the history frame on the right labeled: Subtract on data X, data X where X is the history number of the datasets.

The dataset output will be the exons minus the subregions that overlap transposable elements. This is different from the previous example: only the overlapping subregions of the exons are removed. Regions or intervals not overlapping are preserved.

4. Merge and Concatenate.

Concatenate and Merge (Fig. 10.5.8C, D) together are analogous to addition or union (Fig. 10.8.16C, D). They can be used together to combine datasets and merge (or flatten) the intervals.

Concatenate simply combines two interval datasets. The option Both queries are exactly the same filetype indicates that columns in both datasets are the same. If this option is unchecked, then the second dataset is adjusted to match the column assignments of the first. However, since the columns chromosome, start, end, and strand are the only columns used by the operations, all other columns will be replaced in the second dataset with a

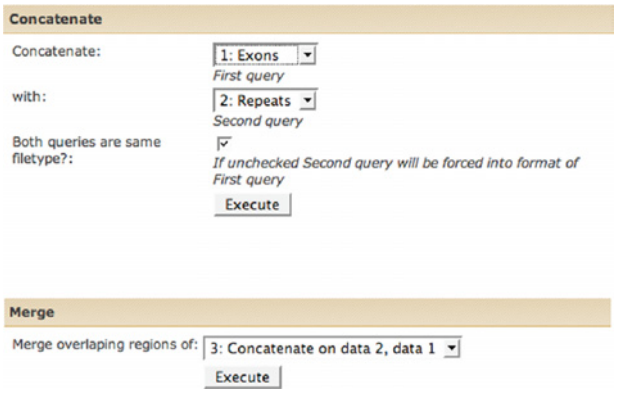


Figure 10.5.12 Parameter settings for Concatenate and Merge tools (Basic Protocol 3, step 4).

period. Typically this option is left checked, as BED files are the typical interval format used within Galaxy.

Merge reads a dataset, and combines all overlapping intervals into single intervals. When merging intervals, all columns besides chromosome, start, and end are lost. When two intervals are combined into one, it is ambiguous what the other columns represent or which field should be carried over to the resulting interval. For this reason, all columns except for chromosome, start, and end are omitted from the output.

Example: Concatenate and Merge coding exons and transposable elements (for parameter settings see Fig. 10.5.12)

a. Click Operate on Genomic Intervals in the tool frame on the left.

b. Click Concatenate under Operate on Genomic Intervals.

The Concatenate tool should load in the center frame.

c. Set the drop-down list labeled Concatenate (using this file format) to the Exons dataset.

d. Set the drop-down list labeled "with" to the Repeats dataset.

e. Leave the checkbox labeled Both queries are exactly the same filetype checked. Both datasets are the BED format.

f. Click the Execute button.

This will launch the concatenate operation. A new history item should appear in the history frame on the right labeled: Concatenate on data X, data X where X is the history number of the datasets.

g. After the operation has completed, the history item will change to a light-green color. You may click on the title of the history item to view the first few lines, or click the eye icon to view the dataset. This dataset is both datasets combined into one dataset.

h. Click Merge under Operate on Genomic Intervals.

The Merge tool should load in the center frame.

i. The previous dataset Concatenate on data X, data X should be selected in the drop-down list labeled Merge overlapping regions of. If it is not, select the concatenated dataset.

j. Click the Execute button.

This will launch the merge operation. A new history item should appear in the history frame on the right labeled: Merge on data X where X is the history number of the dataset.

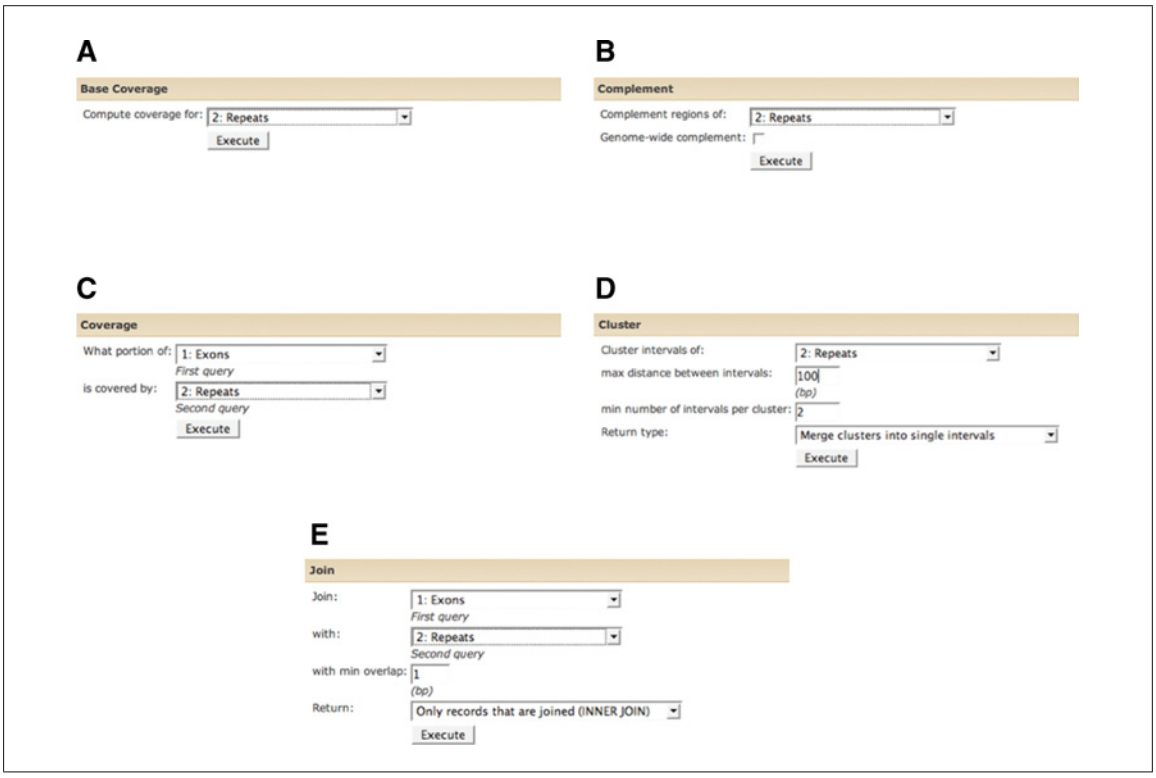


Figure 10.5.13 Parameter settings for Base Coverage (A), Complement (B), Coverage (C), Cluster (D), and Join (E).

In this example, the two datasets are first concatenated. This outputs a BED file containing all of the intervals of both datasets (Fig. 10.5.8D). The next step, merge, merges all of the overlapping regions into single intervals (Fig. 10.5.8C). The resulting dataset is a representation of all of the regions on chromosome 22 that are either a coding exon, a transposable element, or both.

Concatenate combines datasets, and has the ability to combine interval datasets of different types. Merge combines overlapping intervals into single intervals. Together, the two operations can be used to combine intervals from different datasets into simple regions.

5. Base Coverage.

   The base coverage tool calculates the number of bases covered by all of the intervals in a dataset. It does not count overlapping bases more than once; if there are two intervals referring to the same region, those bases are only counted once.

   Example: Calculate the number of bases covered by transposable elements (for parameter settings see Fig. 10.5.13A)

   a. Click Operate on Genomic Intervals in the tool frame on the left.

   b. Click Base Coverage under Operate on Genomic Intervals.

      The base coverage tool should load in the center frame.

   c. Set the drop-down list labeled Count total bases covered by to the Repeats dataset.

   d. Click the Execute button.

      This will launch the base coverage operation. A new history item should appear in the history frame on the right labeled: Base Coverage on data X where X is the history number of the dataset.

      Click on the title of history item. The item will expand and display a single number in a white area. That is the total number of bases covered by transposable elements.

6. Complement:

   Complement inverts a dataset (Fig. 10.5.8E). Complement reads in all of the regions of a dataset, and outputs the regions not covered by any intervals in that dataset. The option Genome-wide complement allows for the entire genome to be complemented, regardless of whether a chromosome, contig, scaffold, etc. is represented in the query dataset. In a genome-wide complement of a dataset, any chromosome that does not have any intervals in the query dataset will be output in the result as the entire chromosome. In a normal complement, only the chromosomes, contigs, scaffolds, etc. that are referenced in the query dataset will be represented in the output.

   When complementing a chromosome, the length of the chromosome is needed. Galaxy uses the chromInfo tables available through the UCSC Table Browser for this information. For complements on builds not available through UCSC, a default chromosome length of 512 megabases is assumed.

   Example: Chromosome complement of repeats on chromosome 22 (for parameter settings see Fig. 10.5.13B)

   a. Click Operate on Genomic Intervals in the tool frame on the left.
   b. Click Complement under Operate on Genomic Intervals.

      The Complement tool should load in the center frame.

   c. Set the drop-down list labeled Complement regions of to the Repeats dataset.
   d. Leave the checkbox labeled Genome-wide complement unchecked. Only chromosome 22 will be complemented.
   e. Click the Execute button.

      This will launch the coverage operation. A new history item should appear in the history frame on the right labeled: Complement on data X where X is the history number of the dataset.

      The resulting dataset will contain intervals representing regions that are NOT transposable elements. Also, a normal complement is done in contrast to a genome-wide complement because when obtaining the simple repeats, chromosome 22 was specifically specified, and the other chromosomes were explicitly omitted.

7. Coverage.

   Coverage (Fig. 10.5.9) is a combination of intersect and base coverage. Coverage finds the number of bases each interval in the first dataset covers of the second dataset. In addition, it finds the fraction of the interval's total length that covers intervals in the second query. The resulting dataset is all of the intervals from the first input dataset, with two columns added to the end: bases covered and fraction covered. The additional two columns can be manipulated with other tools such as Filter under the Filter and Sort section of the toolbox or with Compute under the Text Manipulations section of the toolbox.

   Example: Find the coverage of each coding exon on simple repeats (for parameter settings see Fig. 10.5.13C)

   a. Click Operate on Genomic Intervals in the tool frame on the left.
   b. Click Coverage under Operate on Genomic Intervals.

      The Coverage tool should load in the center frame.

   c. Set the drop-down list labeled "What portion of" to the Exons dataset.

      This is the dataset containing coding exons.

   d. Set the drop-down list labeled "is covered by" to the Repeats dataset.
   e. Click the Execute button.

This will launch the coverage operation. A new history item should appear in the history frame on the right labeled: Coverage on data X, data X where X is the history number of the datasets.

The resulting dataset will be all of the coding exons, with two additional columns. The first additional column is the number of bases that the interval covers in the transposable elements dataset. The second additional column is the fraction of that interval that covers bases represented by the transposable elements dataset.

8. Cluster:

Cluster is one of the most versatile and powerful interval operations (Fig. 10.5.8F). Cluster finds clusters of intervals, and has a wide range of behavior depending on the options specified. The Maximum distance parameter specifies the maximum distance allowed between regions for those regions to be considered a cluster. Maximum distance can be a positive number, zero, or a negative number. When maximum distance is a positive number, regions that are at most that distance from each other are considered to be a cluster. When maximum distance is zero, cluster considers intervals that are touching to be a cluster. This is similar to the behavior of the merge tool, but is more flexible and specific. When maximum distance is a negative number, intervals that have that amount of overlap are considered to be a cluster.

A cluster will be ignored unless it has at least as many intervals within it as specified by the parameter Minimum intervals per cluster. If this is set to 1 or lower, then all intervals, even single intervals that do not cluster with any surrounding intervals, are included in the output.

Cluster has three options for output as found in the drop-down list Return type: Merge clusters into single intervals, Find cluster intervals; preserve comments and order, and Find cluster intervals; output grouped by clusters:

***Merge clusters into single intervals** finds all of the clusters according to the criteria set by maximum distance and minimum intervals per cluster, and outputs the start and end of each cluster as an interval. The result is that clustered intervals become one large, continuous interval spanning all of the intervals within that cluster. Setting maximum distance to 0 and minimum intervals per cluster to 1 with this option produces exactly the same output as the Merge tool.*

***Find cluster intervals; preserve comments and order** finds all of the clusters according to the criteria set by maximum distance and minimum intervals per cluster, and outputs those intervals in the original order they were encountered in the input dataset. This option can be thought of as a filter that removes the intervals that are not found within a cluster.*

***Find cluster intervals; output grouped by clusters** finds all of the clusters according to the criteria set by maximum and minimum intervals per cluster. It is the same as the previous option, except that the intervals are grouped together in the output by cluster.*

Example: merge clusters of at least 2 transposable elements within 100 base pairs into single regions elements (for parameter settings see Fig. 10.5.13D)

a. Click Operate on Genomic Intervals in the tool frame on the left.

b. Click Cluster under Operate on Genomic Intervals.

The Cluster tool should load in the center frame.

c. Set the drop-down list labeled Cluster intervals of to the Repeats dataset.

d. Set Maximum distance to `100`.

e. Set Minimum intervals per cluster to `2`.

f. Make sure the drop-down list labeled “Return type” is set to “Merge clusters into single intervals.”

g. Click the Execute button.

This will launch the cluster operation. A new history item should appear in the history frame on the right labeled: Cluster on data X where X is the history number of the dataset. The operation is launched on the server, and every 10 sec the browser will check to see if it has finished running.

h. After the operation has completed, the history item will change to a light-green color. You may click on the title of the history item to view the first few lines, or click the eye icon to view the dataset.

The dataset returned will represent clusters of at least transposable elements within 100 bp of each other. Screencast 3F continues past this point and shows how to highlight intervals assigned to each cluster.

9. Join:

The join operation is similar to joins done by database management systems such as MySQL (Fig. 10.5.14). Join looks at two datasets of intervals, and joins them based on interval overlap. Any interval in the second dataset that overlaps an interval in the first dataset will be appended to the line from the first dataset and output.

Like intersect, join allows a minimum overlap to be specified. Intervals must exceed the minimum overlap to be joined. There are several types of join that can be done. These are specified by the drop-down list labeled Join Type (Fig. 10.5.14):

Return only records that are joined *will only return intervals in the first query that overlap and are joined to an interval in the second query. For users of SQL databases, this is similar to an INNER JOIN.*

Return all records of first query (fill null with '.') *returns all intervals from the first query. Any interval in the first query that does not join an interval in the second query will have the extra fields padded with a period.*

Return all records of second query (fill null with '.') *returns all intervals from the second query. Any interval in the second query that is not joined to an interval in the first query will have fields filled in with a period. Because the intervals are filled in with a period, this may output an invalid interval dataset. Further operations on the resulting dataset may not be possible, since chromosome, start, and end will be replaced with a period, which is not a valid value.*

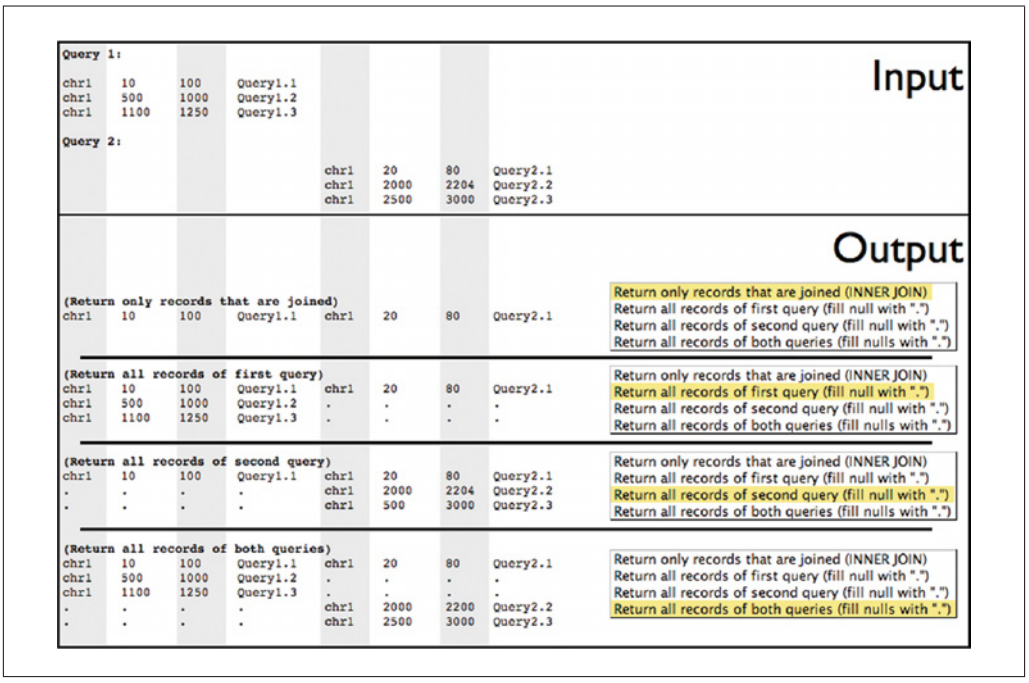


Figure 10.5.14 Different options of the join tool and resultant output.

Return all records of both queries (fill nulls with a '.') *returns all of the intervals from both queries. Intervals that do not join have fields filled in with a period. As with the previous option, this could result in an interval dataset with a period in the chromosome, start, and end fields. This would result in a dataset that cannot have any further operations performed on it.*

The output of each option is explained in Figure 10.5.14. Notice that in the last two options/examples intervals with invalid chromosome, start, and end are output, rendering those datasets no longer usable without filtering out the invalid intervals.

Example: Join coding exons with transposable elements (for parameter settings see Fig. 10.5.13E)

a. Click Operate on Genomic Intervals in the tool frame on the left.

b. Click Join under Operate on Genomic Intervals.

The Join tool should load in the center frame.

c. Set the drop-down list labeled Join to the Exons dataset.

d. Set the drop-down list labeled "with" to the Repeats dataset.

e. Leave the Minimum Overlap field set to `1`.

f. Make sure the drop-down list labeled Return is set to "Only records that are joined (INNER JOIN)."

g. Click the Execute button.

This will launch the join operation. A new history item should appear in the history frame on the right labeled: Join on data X, data X where X is the history numbers of the datasets. The operation is launched on the server, and every 10 sec the browser will check to see if it has finished running.

h. After the operation has completed, the history item will change to a light-green color. You may click on the title of the history item to view the first few lines, or click the eye icon to view the dataset.

The dataset returned will contain only the coding exons that overlap a transposable element, and will have the corresponding simple repeat added as extra columns to the end of each line. Further analysis could use the coverage tool on this resulting dataset to calculate the amount of coverage each exon has on each repeat.

BASIC PROTOCOL 4

EXTRACTING ALIGNMENTS IN GALAXY

In addition to allowing alignment data to be uploaded or loaded from external data sources like the UCSC genome browser, Galaxy includes tools to extract pairwise and multiple sequence alignments corresponding to sets of genomic intervals.

1. Obtain interval data for which alignments will be extracted (coding exons on chromosome 22):

   a. Click Get Data in the Tools frame.

   b. Click UCSC Main Table Browser.

   c. Make sure the following parameters are set:

      clade: Vertebrate
      genome: Human
      assembly: May 2004
      group: Genes and Gene prediction Tracks
      track: Known Genes
      radio button region is set to "position."

   d. Type `chr22` in the position box.

e. Click the “get output” button.

This brings up the next screen of the Table Browser interface

f. Select Coding Exons radio button.

g. Click the “get BED” button.

This will create a new history item named `UCSC: knownGene (chr22).`

2. Extract multiple sequence alignments (Fig. 10.5.15):

a. In the Tools frame click Extract MAF blocks.

b. In the select list labeled Interval file choose `UCSC: knownGene (chr22).`

c. Click Next step.

d. From the select list labeled Choose MAF source select “8-way multiZ (hg17).”

This job may also take a few minutes to run. After it finishes, your history will contain a new dataset with the portions of human multiple alignments that overlap with human coding exons.

3. Convert multiple alignments to FASTA format (Fig. 10.5.16):

Many programs that can work with alignment data do not understand the MAF format. Galaxy allows these alignments to be converted to FASTA format.

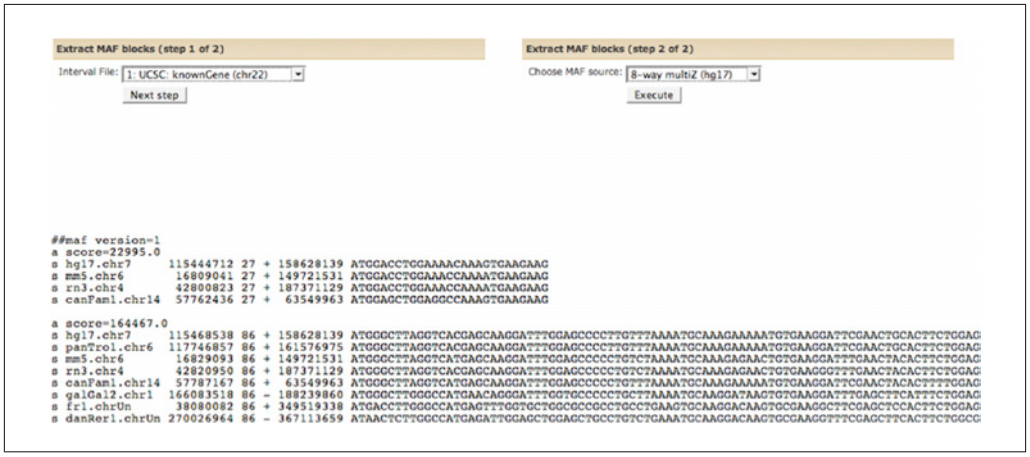


Figure 10.5.15 Parameter settings and output (MAF format) of the tool for multiple alignment extraction (Basic Protocol 4, step 3).

Figure 10.5.16 Parameter settings and output (FASTA format) of tool for MAF-to-FASTA conversion (Basic Protocol 4, step 4).

a. In the Tools frame click Convert Formats.
b. In the menu that appears click MAF to FASTA.
c. In the select list labeled MAF dataset to convert choose Extract MAF blocks on data X.
d. Click Next Step.
e. Click the checkboxes labeled "danRer1," "fr1," and "hg17."
f. Click Execute.

A new dataset containing three FASTA records will be added to your history.

BASIC PROTOCOL 5

BACTERIAL DATA IN GALAXY

The National Center for Biotechnology Information (NCBI) contains over 400 completely sequenced microbial genomes, but this information is only available primarily through an FTP server and is not in a form that is usable by most experimentalists. The Archaea Browser (Schneider et al., 2006) does a good job of providing genomic visualization and datasets for several microbes, but, like the UCSC Genome Browser, does not make it easy to perform large-scale analyses (see Introduction). Galaxy provides its own repository for bacterial data. Thus, you can use Galaxy tools, such as interval operations, on bacterial data much like you would do that on human genome. The following protocol explains how to obtain bacterial data from Galaxy interface.

1. Click the Get Data link at the top left pane.

   This will expand Galaxy's available data sources.

2. Click the Get Microbial Data link.

   The first page of the microbial data interface will appear in the center pane of the Galaxy screen. Here, users are able to select which kingdom their desired organism resides (Archaea or Bacteria). If uncertain as to which kingdom an organism of interest is assigned to or to see which organisms are currently available, there is a link in the onscreen help at each step, which will open an alphabetical table of available organisms. Once the appropriate kingdom is selected, press the "Execute" button.

3. Select organism's phylogenetic group (Fig. 10.5.17A).

   Organisms have been arranged by phylogenetic grouping as assigned by NCBI. Select the group to which an organism of interest belongs. If unsure of the appropriate group, use "Click here for a list of available species and their location."

4. Once the desired group is selected, click Execute.

5. Select the desired organism (Fig. 10.5.17B).

   Here one can select an organism of interest. Located next to the name of each organism is an "about" link. Clicking this link will bring up the NCBI Genome Project homepage for that particular organism. Organisms listed in bold are available at the UCSC Browsers; for these organisms, results can be viewed within the UCSC genome browser, and one may seamlessly mix data from the UCSC table browser and Galaxy's repository.

6. Select the desired organism, and click Execute.

7. Select the desired data (Fig. 10.5.17C).

   Datasets are organized under headers by type of data. Currently available types of data are coding sequences (CDS), tRNA, rRNA, GeneMark Annotations, GeneMarkHMM Annotations, Glimmer3 Annotations, and DNA sequences. Within each type of data, there is a separate entry for each chromosome (i.e., true chromosomes and plasmids). Next to each chromosome entry is an "about" link which will bring the user to the NCBI ENTREZ page for that particular chromosome. Here, unlike the previous steps, the user can select

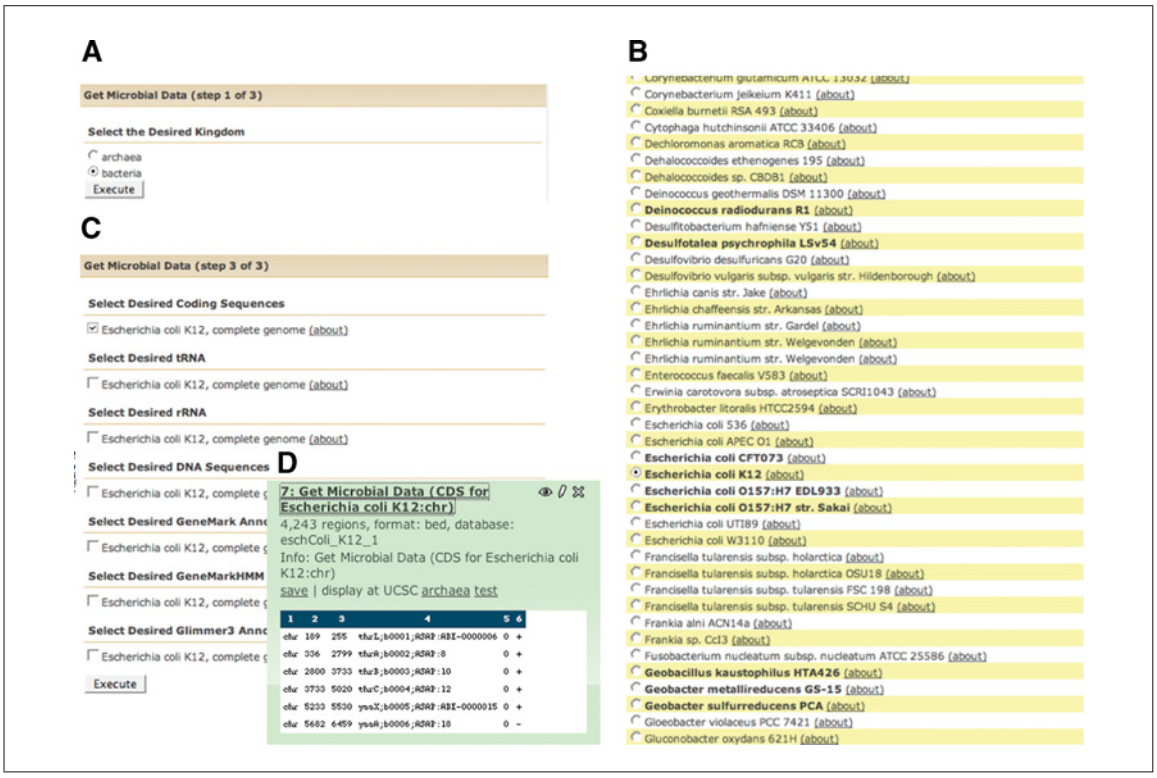


Figure 10.5.17 To download coordinates of protein-coding regions with *E. Coli* genome one needs to select Kingdom (A), Species (B), and, finally, coding sequences (C). Coding sequences results shown (D).

any number of datasets at a time. After selecting one or more datasets, click Execute. Each dataset will appear in the user's history as a separate item.

Once the bacterial data appears in the history you can apply the same Galaxy tools that were used in earlier protocols. For example, you will be able to perform interval operations or extract sequences corresponding to protein-coding genes (see Screencast 5). In addition, for some species one can visualize data in the UCSC browser (e.g., click on the "archaea" link shown in Fig. 10.5.17D).

GUIDELINES FOR UNDERSTANDING RESULTS

Galaxy was designed to be an interactive system and in most of the cases results will be self-descriptive depending on which tools were applied to the original data. As always caution should be used when interpreting genomic data—the information produced by Galaxy is only as good as the underlying data uploaded from databases. As Galaxy obtains most of the data from the UCSC Table Browser, refer to *UNIT 1.4* for an excellent overview of common pitfalls in the data.

COMMENTARY

Background Information

Modern Web-based genomic resources offer many facilities for retrieving and visualization of data. However, few of these resources offer sophisticated tools for further analysis of these data. As a result, almost every experimental biologist has to analyze data on his/her own, struggling with numerous difficulties arising from format incompatibility or incomprehensible user interfaces. Although our computational colleagues are happy to help, few are willing to devote time and resources to this effort, for developing of good user interfaces is a significant challenge. Galaxy is the system designed to help both sides. For experimental biologists it connects to most widely used data sources (the UCSC Table Browser and BioMart), provides an intuitive user interface, and features a unique history system (workspace) that stores and displays data and actions for every user. For computational biologists Galaxy provides a

framework that can integrate command-line tools with almost no effort. For each tool Galaxy generates interface and provides all housekeeping (e.g., input and output management, job control, error catching, and testing facilities). This test was compiled with experimental biologists in mind. Thus, it does not contain any information on technical aspects of the Galaxy system. This information can be found at *http://g2.bx.psu.edu*.

Critical Parameters and Troubleshooting

Galaxy allows performing of an infinite number of analyses on genomic data. In designing their system the authors tried to put as few constraints on the user as possible. In that sense Galaxy is similar to a car with the manual gearbox—it gives you more control if you know what you are doing (e.g., you do not shift from fifth to reverse). Fortunately user's feedback provides convincing evidence that a short test drive is sufficient to understand how Galaxy works. This text is equivalent to such a test drive. Below the authors list most common problems encountered by Galaxy users. They can be condensed into two categories: (1) data format issues and (2) genome build incompatibilities.

Data format issues

At the time of writing, Galaxy "understood" several datatypes including genomic coordinates (e.g., BED, GFF, and abd interval), sequences (e.g., FASTA), and alignments (e.g., FASTA, and MAF). Most of the tools require data to be in one of these formats. For example, operations of genomic intervals described in Basic Protocol 3 can be only performed on data in *Interval* format. In most cases changing your data to interval format is as simple as correctly setting metadata as shown in Step 3 of Basic Protocol 2.

Genome build incompatibilities

To the best of the authors' knowledge Galaxy is the only interactive genome analysis portal that allows mixing different genomes within single analysis space (History). In the authors' opinion such "mixing" is essential for a true comparative genomics resource. The ease of mixing also means that in some cases users will accidentally attempt comparing data from different genomes. Thus, when using tools that operate on more than one history item (i.e., most genomic interval operations) make sure that all data come from the same genome build.

If things go wrong

Galaxy is a rapidly developing system. The authors take great care in fine-tuning existing functionality and fixing bugs. But as with any software Galaxy contains bugs. If you encounter a bug or just have a question e-mail the authors at *galaxy-bugs@bx.psu.edu*.

Additional Screencasts

In addition to protocols described in this unit *http://g2.bx.psu.edu* features the following screencasts:

Screencast 6: Using Galaxy to create UCSC Genome Browser Custom Track.

Screencast 7: Tools for editing genomic data.

Screencast 8: Filtering, sorting, and comparing datasets with Galaxy.

Acknowledgements

A vision for Galaxy was originally articulated by Ross Hardison, who is also the major source of support and critical feedback. The authors would like to thank Jim Kent and David Haussler for their continuing support and making UCSC Genome Browser uplink and connection possible. Istvan Albert pioneered initial aspects of Galaxy design. The following individuals contributed to Galaxy project at different stages: Richard Burhans, Laura Elnitski, Belinda Giardiane, Bob Harris, Jianbin He, Webb Miller, Cathy Riemer, and Yi Zhang. Galaxy hardware is maintained by Nate Coraor. Robert Castelo, France Denoeud, Roderic Guigo, Erika Kvikstad, Julien Lagarde, and Kateryna Makova provided critical comments during software testing. This work is supported by funds provided by the Eberly College of Science, Huck Institutes of the Life Sciences at Penn State University, and NSF BD&I grant 0543285 to AN.

Literature Cited

Birney, E., Andrews, D., Bevan, P., Caccamo, M., Cameron, G., Chen, Y., Clarke, L., Coates, G., Cox, T., Cuff, J., Curwen, V., Cutts, T., Down, T., Durbin, R., Eyras, E., Fernandez-Suarez, X.M., Gane, P., Gibbins, B., Gilbert, J., Hammond, M., Hotz, H., Iyer, V., Kahari, A., Jekosch, K., Kasprzyk, A., Keefe, D., Keenan, S., Lehvaslaiho, H., McVicker, G., Melsopp, C., Meidl, P., Mongin, E., Pettett, R., Potter, S., Proctor, G., Rae, M., Searle, S., Slater, G., Smedley, D., Smith, J., Spooner, W., Stabenau, A., Stalker, J., Storey, R., Ureta-Vidal, A., Woodwark, C., Clamp, M., and Hubbard, T. 2004. Ensembl 2004. *Nucl. Acids Res.* 32:D468–D470.

Blankenberg, D., Taylor, J., Schenck, I., He, J., Zhang, Y., Ghent, M., Veeraraghavan, N., Albert, I., Miller, W., Makova, K.D.,

Hardison, R.C., and Nekrutenko, A. 2007. A frame-work collaborative analysis of ENCODE data: Making large-scale analyses biologist-friendly. *Genome Res.* 17:960-964.

Karolchik, D., Baertsch, R., Diekhans, M., Furey, T.S., Hinrichs, A., Lu, Y.T., Roskin, K.M., Schwartz, M., Sugnet, C.W., Thomas, D.J., Weber, R.J., Haussler, D., Kent, W.J.; University of California Santa Cruz. 2003. The UCSC Genome Browser Database. *Nucl. Acids Res.* 31:51–54.

Maglott, D., Ostell, J., Pruitt, K.D., and Tatusova, T. 2005. Entrez gene: Gene-centered information at NCBI. *Nucl. Acids Res.* 33:D54–D58.

Schneider, K.L., Pollard, K.S., Baertsch, R., Pohl, A., and Lowe, T.M. 2006. The UCSC Archaeal Genome Browser. *Nucl. Acids Res.* 34:D407–D410.

Obtaining Comparative Genomic Data with the VISTA Family of Computational Tools

Igor Ratnere[1] and Inna Dubchak[1]

[1]Lawrence Berkeley National Laboratory, Berkeley, California

ABSTRACT

Comparison of DNA sequences from different species is a fundamental method for identifying functional elements, such as exons or enhancers, as they tend to exhibit significant sequence similarity due to purifying selection. Availability of whole-genome sequences for a constantly growing number of organisms makes identification of such elements within these genomes possible. There are two distinct phases in comparisons of genomic sequences: in the first, the sequences are aligned, and in the second, the resulting alignments are analyzed to find conservation signals that may be indicative of functional regions. Due to the considerable length of alignments, good visual representation techniques are a necessity for effective isolation of regions of interest. The VISTA family of tools provides biomedical investigators with a unified framework for the alignment of long genomic sequences and whole-genome assemblies, interactive visual analysis of alignments along with functional annotation, and many other comparative genomics capabilities. *Curr. Protoc. Bioinform.* 26:10.6.1-10.6.17. © 2009 by John Wiley & Sons, Inc.

Keywords: comparative genomics • DNA alignment • VISTA • genome browser

INTRODUCTION

Unlike the comprehensive sources of genomic information and analytical tools, such as NCBI (*UNIT 1.3*; Wheeler et al., 2007), UCSC Browser (*UNIT 1.4*; Karolchik et al., 2008), and ENSEMBL (*UNIT 1.15*; Hubbard et al., 2002), the VISTA family of tools is mostly focused on the comparative analysis of genomic sequences. This family, accessible through the VISTA home page at *http://genome.lbl.gov/vista*, consists of databases, tools, and online servers for different types of such analysis. Researchers can use the VISTA Browser to examine alignments of whole-genome assemblies among a variety of species, or use the servers to submit their own sequences for analysis. The VISTA servers section includes several independent tools: mVISTA enables you to align and compare your own genomic sequences. rVISTA finds conserved transcription factor binding sites—combining a comparative sequence analysis with a search for transcription factor binding sites using the TRANSFAC database. GenomeVISTA aligns your DNA sequence to one of the genomes available at the VISTA Web site. Phylo-VISTA is an advanced visualization tool used to analyze multiple DNA sequence alignments, considering the phylogenetic relationship of your sequences.

The Precomputed Whole Genome Alignments section consists of three parts: (1) The VISTA Browser allows for the navigation through whole-genome alignments, as well as through your own alignments. (2) The Whole Genome rVISTA provides results of a precomputed whole-genome Regulatory VISTA analysis (rVISTA). It searches for over-represented transcription factors in the upstream regions of genes using the TRANSFAC database. (3) The Microbial Genomes collection includes precomputed alignments of a large set of complete genomes.

Current Protocols in Bioinformatics 10.6.1-10.6.17, June 2009
Published online June 2009 in Wiley Interscience (www.interscience.wiley.com).
DOI: 10.1002/0471250953.bi1006s26

In this unit, the authors describe the functionality of selected VISTA modules in the following protocols. Basic Protocol 1 covers main features of the VISTA Browser. Basic Protocol 2 describes a more detailed analysis of alignment, including SNP viewing. Basic Protocol 3 demonstrates how to access all major alignment data, including conserved regions. Basic Protocol 4 presents the GenomeVISTA tool for mapping user-submitted sequences on any reference genome. Basic Protocol 5 guides a user through the method of predicting transcription factor binding sites in the context of sequence conservation. Basic Protocol 6 introduces a collection of experimentally verified human enhancers.

BASIC PROTOCOL 1

ANALYZING COMPARATIVE GENOMIC DATA WITH THE VISTA BROWSER

VISTA Browser allows the user to interactively visualize alignments of genomic sequences and whole-genome assemblies, and quickly identify conserved regions. A number of whole-genome pair-wise and multiple alignments of vertebrates, plants, insects, and representatives of many other phyla, computed by the VISTA group, are available for the exploration with the VISTA Browser (accessible from the VISTA home page and the gateway page *http://pipeline.lbl.gov/*). More than 2000 precomputed full scaffold alignments, of close to 400 genomes of bacteria and archaea, are available as the VISTA component of IMG (Integrated Microbial Genomes; Markowitz et al., 2008). Access to these results is provided through the Integrated Microbial Genomes link on the VISTA home page.

In this protocol, we will analyze the result of the multiple alignments of human, rhesus monkey, dog, horse, mouse, rat and chicken genomes. It contains the instructions on changing the base genome and a genome position and adjusting the curve parameters, as well as description of graphical user interface (GUI), navigation, and more.

Necessary Resources

Hardware

Computer with Internet access

Software

Web browser
JAVA

1. Open the VISTA gateway page, at *http://pipeline.lbl.gov/* or through the VISTA Browser link on the home page (Fig. 10.6.1).

2. Using the input form, select Human, Mar. 2006 as the base genome, type in the Position box `zak` for a gene name, and then press the Go button.

   Alternatively, you can select VISTA tracks on the UCSC Browser check box to view the VISTA conservation curves of the same interval aligned with the rest of the tracks of the UCSC genome browser. In addition to the VISTA tracks, you will see the VISTA control panel for changing parameters of the curves displayed. This option does not require Java to be installed on your computer, and allows for visualizing the level of sequence conservation in the context of many UCSC annotation tracks.

3. This query will return a small window displaying two RefSeq isoforms of the gene *ZAK*. Choose the longer isoform 1 and this will open the VISTA Browser window (Fig. 10.6.2) with the results of the multiple alignments of Human, Rhesus, Dog, Horse, Mouse, Rat, and Chicken genomes.

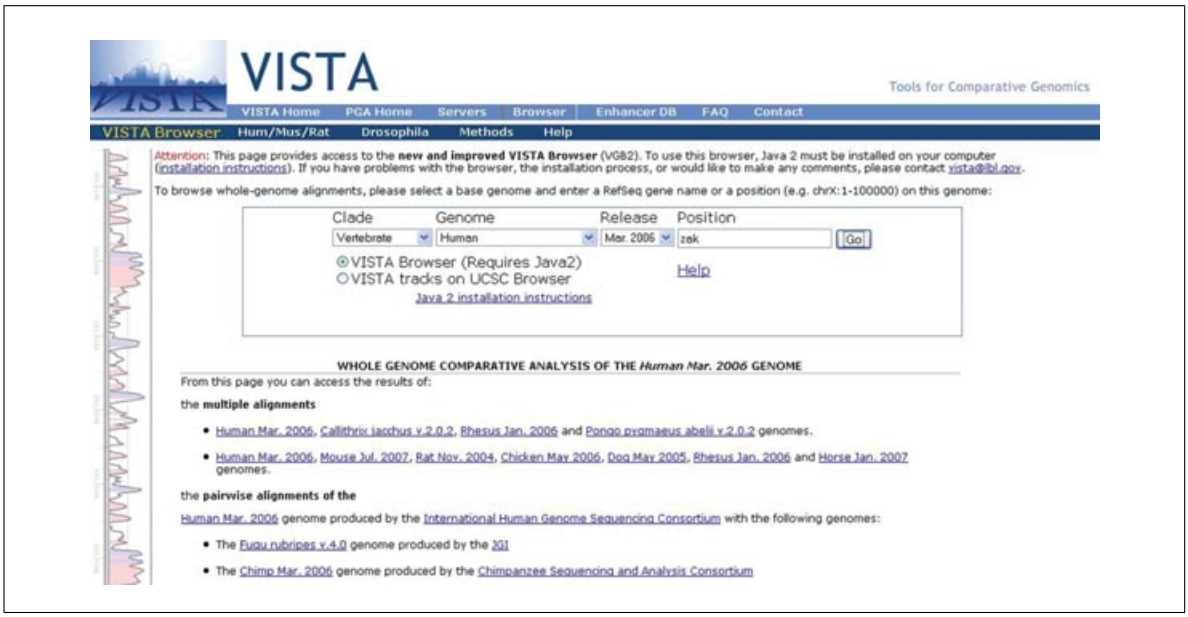


Figure 10.6.1 VISTA Browser gateway page consists of three main parts. The top panel contains a toolbar with the links to different VISTA tools and servers. The middle panel contains a menu for the selection of a base (reference) genome and a position on it, which are coordinates on the genome, a gene name, a SNP index, or a contig name. The bottom panel contains the links to the sources of the genome assemblies.

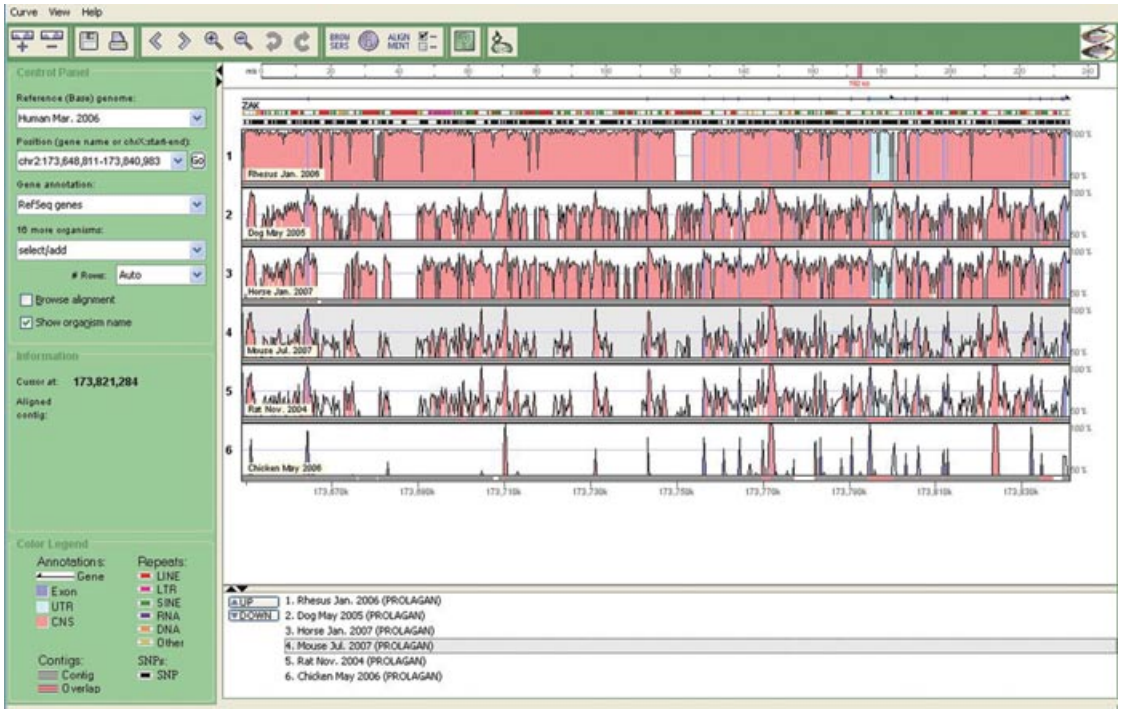


Figure 10.6.2 VISTA Genome Browser shows the results of the multiple alignment. To change the order in which the curves are displayed, select the curve you want to move up or down and click the Up/Down buttons next to the curve name at the bottom of the screen.

IMPORTANT NOTE: *VISTA Browser requires Java be installed on your computer. A Java check window may popup during this step; please see a help page for the instructions on installing Java.*

4. Locate Control Panel on the left (Fig. 10.6.2) and check off the "Show organism name" box to display the organism name on a graph.

   Use the control panel to change the reference genome, position, or gene annotation, to add more alignments, to change number of rows a curve breaks into, or to show base-pair level alignment panel.

Understanding the display (see Fig. 10.6.2)

The annotation tracks are shown above the conservation curves. The first track from the top is gene annotation, where dark and light blue boxes represent exons and UTRs, respectively. Gene name appears underneath the track, the arrow points in the direction of the gene. The middle track shows the repeats, whereas SNPs are indicated as black marks on the track directly above the graphs. The VISTA curve is calculated as a windowed-average identity score for the alignment. Each "peaks and valleys" graph represents percent conservation between aligned sequences at a given coordinate on the base genome; top and bottom percentage bounds are shown to the right of a graph. Regions are classified as "conserved" by analyzing scores for each base pair in the genomic interval, that is Min Cons Width (default value 100 bp) and Cons Identity (default value 70%). A region is considered conserved if the conservation over this region is greater than or equal to the Conservation Identity and has the minimum length of Minimum Conserved Width. Regions of high conservation are colored according to the annotation as exons (dark blue), UTRs (light blue), or noncoding (red). The thresholds that determine what is colored, as well as the minimum and maximum percentage bounds, can be easily adjusted (see action Changing curve parameters, below).

You can display different annotations by selecting them from the Gene annotation list on the control panel. SNPs are also shown on the base-pair level alignment panel (see below, Fig. 10.6.6).

Due to resolution constraints when visualizing large alignments, it is often necessary to condense information about a hundred or more base pairs into one display pixel, this is done by only graphing the maximal score of all the base pairs covered by that pixel.

Changing curve parameters

5. To adjust the parameters for a particular curve, first select this curve with a mouse click, and then click the Parameters button. A window with adjustable parameters will appear (Fig. 10.6.3).

Changing the base genome

6. It is critically important for the analysis of conservation which organism is being used as a base or reference. Thus, VISTA Browser gives you the capability to change a base genome. As an example, let us change the Reference (base) genome in Figure 10.6.2 to Mouse. To do this:

   a. Right-click on a Mouse curve, this will invoke a pull-down context menu (Fig. 10.6.4), select Change base genome from the menu.

   b. In this case, more than one region of the Mouse genome was aligned to Human; you will be presented with a choice of matches. Select the appropriate location (e.g., the top one) and click OK.

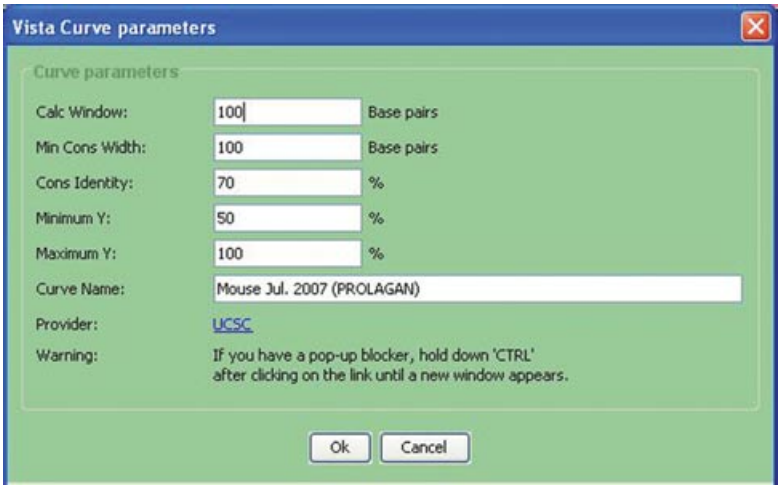


Figure 10.6.3 Changing curve parameters. **Calc Window:** the size of the sliding window used to calculate conservation scores at each base pair for the VISTA curve. **Min Cons Width:** minimum width of a conserved region. **Cons Identity:** minimum percent identity over the window (Min Cons Width) for a region to be considered conserved. **Minimum Y, Maximum Y:** lower and upper boundaries of the graph, dropping the minimum Y value in areas of low conservation will allow you to see the smaller peaks. **Curve Name:** the label that is associated with the curve. **Provider:** the source of the reference genome assembly.

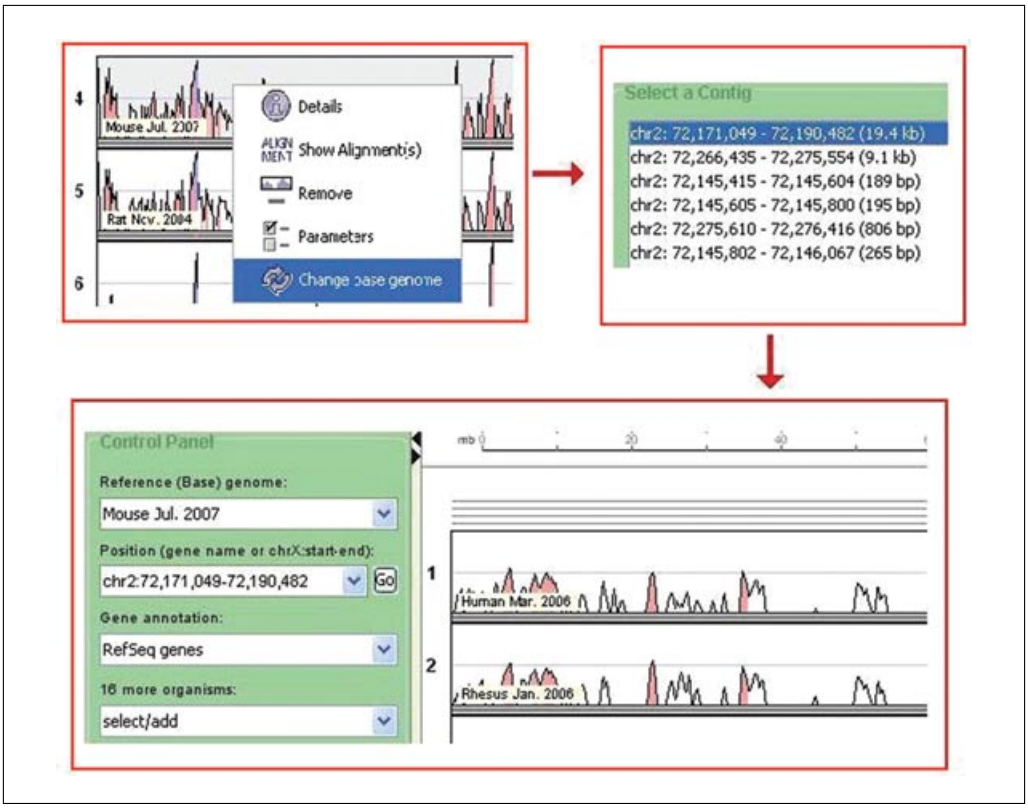


Figure 10.6.4 Changing the base genome. Pull-down context menu is invoked on the Human/Mouse graph. Click the Change base genome button to use a selected organism as a base or reference. In this case, more than one region of the Mouse genome was aligned to the human interval displayed in the browser. Using the mouse, click on the selected region to bring up the VISTA browser with the VISTA curves generated for all mouse alignments covering this region.

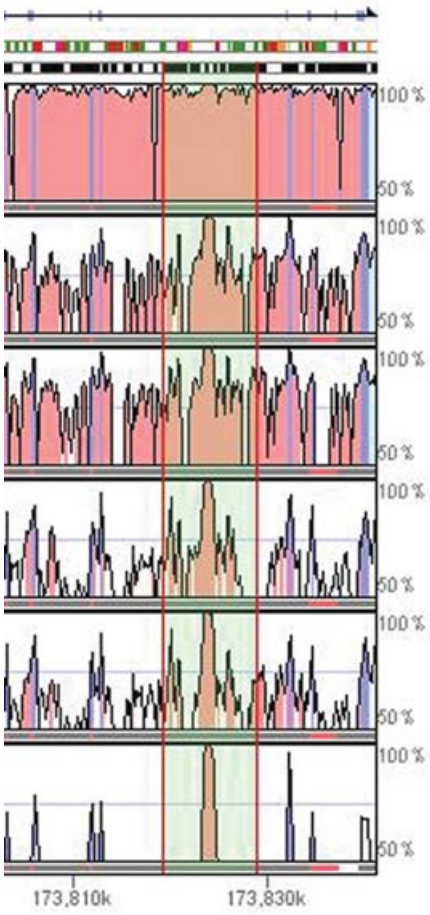


Figure 10.6.5 Zooming in the high conservation noncoding interval of the alignment. Highly conserved noncoding regions are identified by pink color. To zoom in, highlight the area you want to see in detail by holding down the left mouse button, while moving the mouse over the region of interest, the same way you would highlight a sentence in Word. The browser will zoom in on the selected area once you let go of the mouse button. For the color version of this figure go to *http://www.currentprotocols.com*.

c. The Reference (base) genome will be changed to Mouse, along with the position that you selected.

If there is a one-to-one correspondence between two regions, the browser will skip the menu and go straight to the correct location on the other organism.

Navigation (scrolling, zooming, browsing history)

In addition to navigating the browser by entering exact coordinates in the position of the control panel or searching by gene or SNP name, the browser provides standard scrolling and zooming functions.

7. Zoom in on one of the conserved areas by selecting it with the mouse (Fig. 10.6.5). You will see the smaller area of the alignment with several SNPs and repeats, continue zooming in by highlighting the area that includes flanking SNPs around the high conservation site, and you will see the result displayed in Figure 10.6.6, top panel.

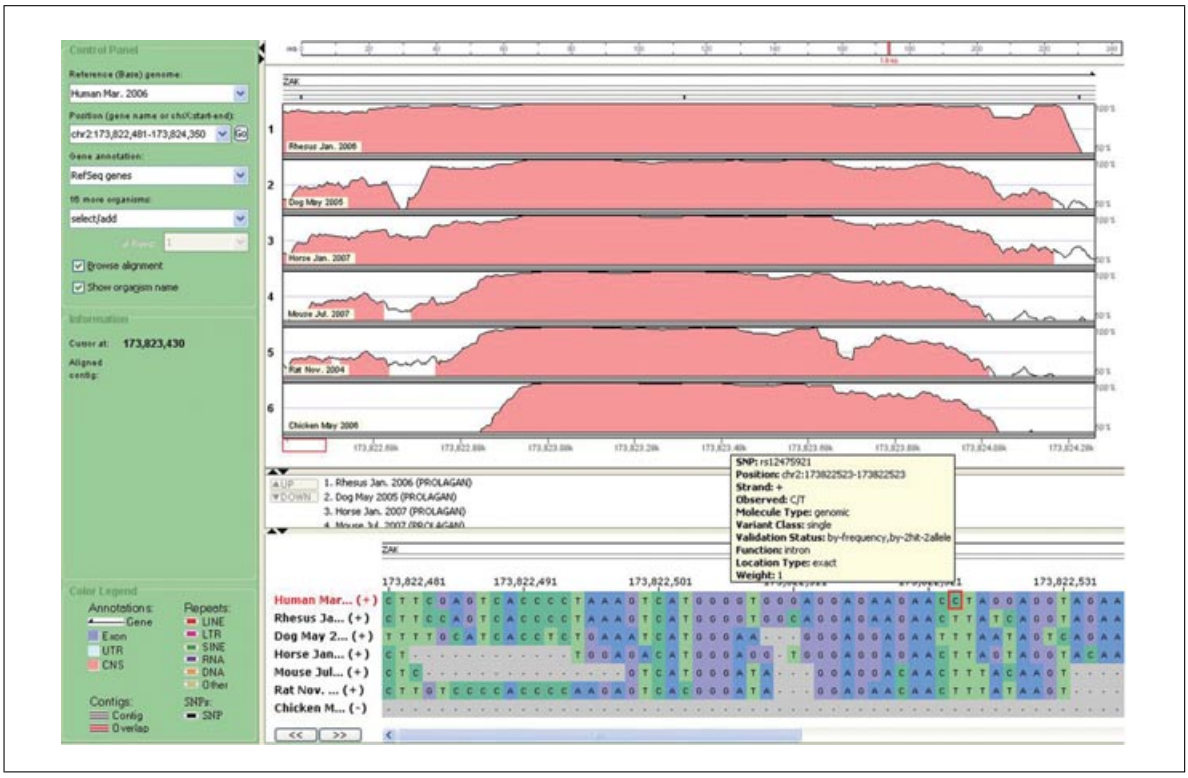


Figure 10.6.6 Base-pair level alignment panel (the bottom panel). Base organism's name is displayed in red. Strand directions are indicated by a (+) or (-) sign. Coordinates on the base genome are drawn above the sequences. A single nucleotide polymorphism (SNP) is indicated by a red border around a base pair. Positioning a mouse over a SNP displays its summary information. For the color version of this figure go to *http://www.currentprotocols.com*.

BROWSING THE ALIGNMENT AND RETRIEVING SNP INFORMATION USING BASE-PAIR LEVEL ALIGNMENT PANEL

BASIC PROTOCOL 2

In the following protocol, we will show a quick and easy way to browse the alignment, get the information about single nucleotide polymorphisms (SNPs), and map the region from the Human, Mar. 2006 genome assembly onto a different assembly, the Human, May 2004, by using SNP information. The more general method of mapping is provided through the Genome VISTA tool (Basic Protocol 4).

Necessary Resources

Hardware

Computer with Internet access

Software

Web browser

1. Using the results from Basic Protocol 1, check the Browse alignment box (Fig. 10.6.2).

   NOTE: *To get this panel, you need to have precisely either one pair-wise or multiple alignment displayed in the browser. So, if you have several stacked pair-wise alignments, you would need to remove all of them except the one you want to study.*

   If the region of the base genome is ~2 Kb or shorter, you will see a red rectangle (slider) under the graph, and the nucleotide level panel will be displayed simultaneously (Fig. 10.6.6).

The red slider appears or disappears while zooming in/out. Dragging the slider changes the region displayed on the nucleotide panel. This mode can be turned on/off by clicking the Browse alignment check box on the control panel (Fig. 10.6.2). There may be cases when curves share the same base organism, but have no common alignment (e.g., for the two curves Human-Mouse and Human-Dog, there is no alignment between Mouse and Dog). For such cases, the Browse alignment mode is disabled.

2. Drag the slider on the left so it is positioned under the SNP (Fig. 10.6.6).

Understanding the base-pair level alignment panel

The panel contains two tracks, genome annotations, and repeats; they are the same as on the VISTA graph panel. The top sequence is the base genome sequence. It is always displayed on the positive strand. Strand directions are indicated by a (+) or (-) sign preceded by the organism's name. The name of the base organism is displayed in red. Coordinates on the base genome are drawn above the sequences. Placing a mouse pointer over a nucleotide letter will reveal its coordinate on a genome. The navigation buttons allow you to move to the previous or next region along the base genome.

If there are several sub-alignments in the same region of the base genome, the alignment with the highest score will be set by default. The scores will be displayed in the alignments list, and changing a score selection will allow you to view alternative alignments.

An SNP is indicated by a red border around a base pair. Moving the mouse over an SNP displays its summary information from the UCSC SNP report (*UNIT 1.4*; Karolchik et al., 2008). Right clicking the mouse on an SNP invokes a pop-up menu with the links to UCSC and PolyPhen reports. PolyPhen (for **poly**morphism **phen**otyping) is a tool that predicts the possible impact of an amino acid substitution on the structure and function of a human protein using straightforward physical and comparative considerations (Ramensky et al., 2002). The PolyPhen link will be disabled when there is no PolyPhen prediction for an SNP available.

3. Using SNPs as genetic markers, map the base genome region onto the Human, May 2004 genome assembly by performing the following:

   a. Select Human, May 2004 as reference genome.
   b. Enter the name of the left flanking SNP, `rs12475921` in the Position box, and click Go.
   c. Locate the area on the base genome flanked by the same SNPs by zooming out and scrolling forward.

BASIC PROTOCOL 3

OBTAINING DETAILED COMPARATIVE DATA INCLUDING GENOMIC COORDINATES AND PARAMETERS OF CONSERVED REGIONS WITH THE TEXT BROWSER

Highly conserved noncoding regions are identified by pink color (Fig. 10.6.6). The calculation of the VISTA curve and thresholds of conservation that determine what is colored, are described in the Understanding the display section of Basic Protocol 1.

We will use the Text Browser to retrieve their sequences with the options that facilitate the design of PCR primers for further investigation of these sequences.

Necessary Resources

Hardware

Computer with Internet access

Software

Web browser

Figure 10.6.7 Text Browser shows detailed information about the aligned regions. The annotation of the base genome can be retrieved by clicking on the Download RefSeq genes link. If there are several annotations available, you can change the annotation to download by using the Change Annotation list.

1. Using the results from Basic Protocol 1 (Fig. 10.6.6, top), identify highly conserved noncoding regions (they are colored in pink).
2. Right-click the mouse on the Mouse, Jul. 2007 graph, this will bring up a pull-down context menu (see Fig. 10.6.4). Now, click on the Details button in the menu. A Text Browser window will open with detailed information regarding the segment of the multiple alignment you were looking at. Another way to open the Text Browser window is to select a graph and click the **"i"** button on the top panel of the browser.

   To view the underlying alignments for any curve, click on the curve to select it, and then click the Show Alignment(s) button. If you were looking at a region that contained only one alignment, it will be displayed immediately. If there were several alignments in the region of interest, a window with details about each of the alignments will open, and you will need to select which one you want to look at. Each alignment that forms a given graph will be shown here, including the overlapped ones. You can get the alignments by clicking the appropriate links in the right-hand column.
3. In the Text Browser window (Fig. 10.6.7), you can see detailed information about the aligned regions, including their genomic coordinates. For example, the coordinates of the mouse region that aligned to human can be found in the Location on Mouse column.
4. To retrieve the coordinates of the regions conserved between human and mouse, select the "Get CNS: the Human-Mouse" link or the "CNS: Human-Mouse" link found in the "Alignment" column. This will open up a page with the coordinates of conserved noncoding sequences. Clicking on the links on that page will give you the sequences of the conserved regions, with retrieval options that facilitate the design of PCR primers for further experimental studies.

FINDING CANDIDATE ORTHOLOGOUS REGIONS ON A BASE GENOME USING THE GenomeVISTA SERVER

BASIC PROTOCOL 4

GenomeVISTA is an automatic server that allows the user to find candidate orthologous regions (derived from the region in the last common ancestral species, and thus usually having similar functions) on a base genome for a draft or finished DNA sequence from any species, and provides detailed comparative analysis. You can align your sequence to a wide selection of genomes provided by the group. GenomeVISTA uses a computational strategy, where query sequence contigs are anchored on the base genome by local alignment matches and then globally aligned to candidate regions by the AVID program (Bray et al., 2003; Couronne et al., 2003). A sequence up to 300 kilobases long can be submitted by pasting it into a window in plain FASTA format, by uploading

FASTA file from your computer, or by providing the GenBank accession number to the server. After submitting the sequence, you will receive a link to the results. The resulting alignments and detailed analysis of conservation can be viewed via the Vista Browser and its extensions. In the VISTA Browser, you will be able to compare the alignment of your sequence against the genome of your choice to the available precomputed alignments of other species against the same interval of this genome.

Necessary Resources

Hardware

Computer with Internet access

Software

Web browser

1. Click on the Servers tab in the navigation bar at the top of the VISTA gateway page (Fig. 10.6.1) to go to the VISTA server page. On that page, click on the GenomeVISTA link, this will open the submission form (Fig. 10.6.8).
2. Using the results from Basic Protocol 3 (Fig. 10.6.7), click on the Sequence (softmasked) link under the Location on Mouse column.
3. Copy the sequence and paste it into the form (Fig. 10.6.8).
4. Select Human, May 2004 as a Base Genome. Enter your e-mail address and a name for your project, and click the Submit Query button.
5. After a few minutes, you will receive an e-mail with a link to your results, click on the VISTA Browser link. This will open a new browser window with your sequence aligned to the Human, May 2004 genome (Fig. 10.6.9).

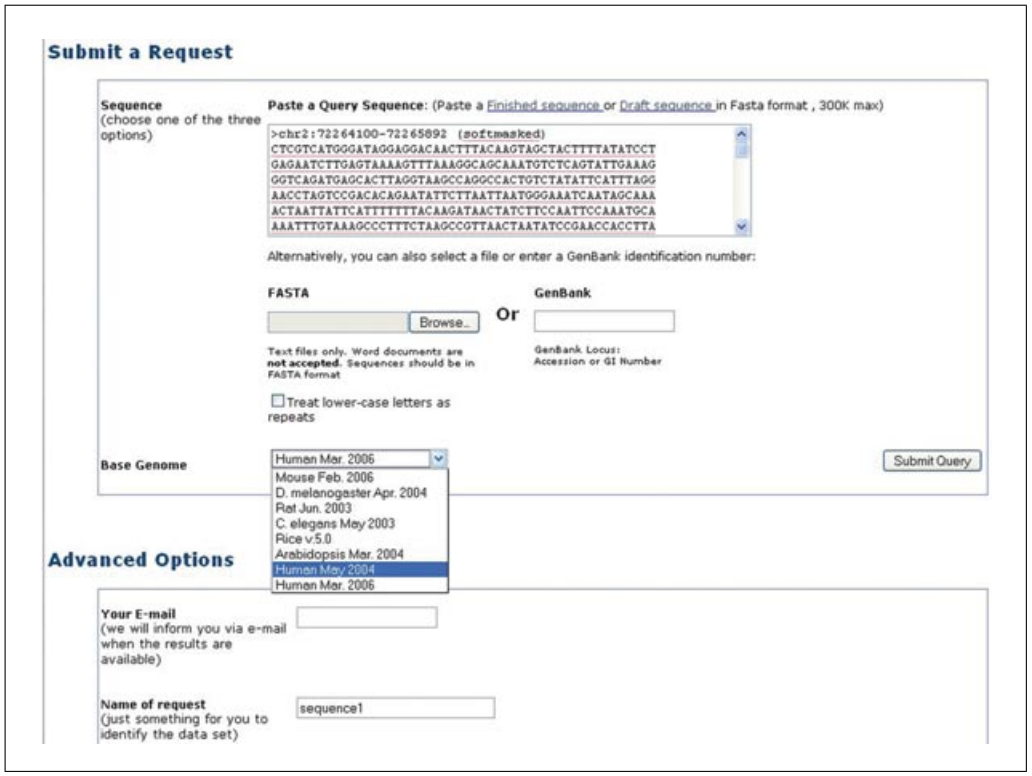


Figure 10.6.8 Genome VISTA submission form. The list of reference genomes available for comparison is shown in the drop-down list.

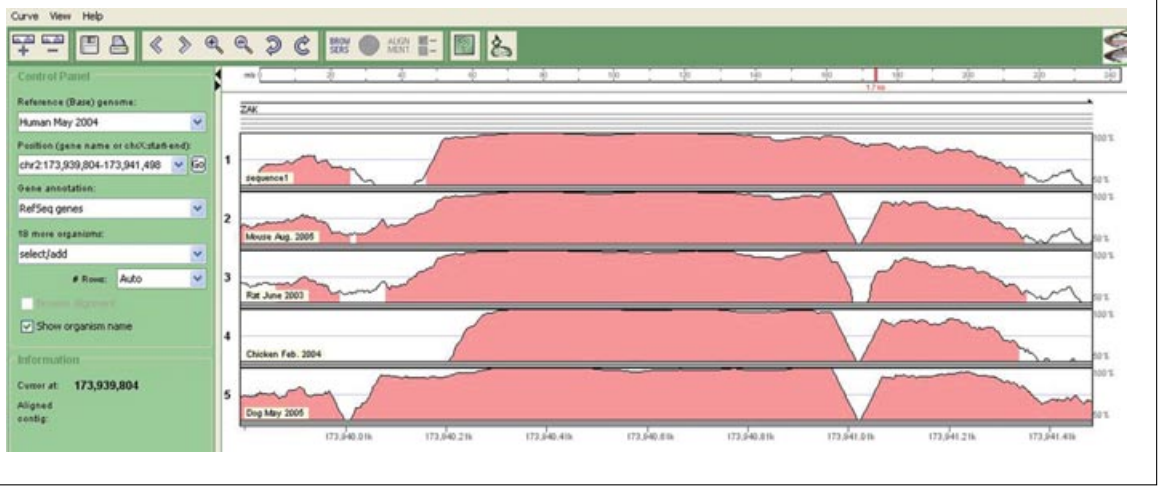


Figure 10.6.9 VISTA Browser displays the alignments with Human May 2004 genome assembly as a base genome. Sequence1—sequence from Mouse, July 2007 genome assembly. The rest of the graphs belong to the multiple alignment.

FINDING PUTATIVE TRANSCRIPTION FACTOR BINDING SITES (TFBS) USING rVISTA SERVER

BASIC PROTOCOL 5

Finding potential regulatory elements in noncoding regions of the human genome is a challenging problem. Analyzing novel sequences for the presence of known transcription factor binding sites, or their weight matrices, produces a huge number of false-positive predictions that are randomly and uniformly distributed. The rVISTA (regulatory VISTA) server combines a search of the TRANSFAC Professional database with a comparative sequence analysis, thus reducing the number of predicted transcription factor binding sites by several orders of magnitude (Loots et al., 2002). The user does not need a license for the TRANSFAC Professional database to see results of the rVISTA analysis. This license is required if the user needs more details on the TRANSFAC Professional Position Weight Matrices of TFBS. In this protocol, we will identify putative binding sites of IRF1 transcription factor in a highly conserved noncoding region of *ZAK* gene.

Necessary Resources

Hardware

Computer with Internet access

Software

Web browser

1. Using the results from Basic Protocol 3 (see Fig. 10.6.7), click on the rVISTA: Human-Mouse link in the Alignment column. This will prompt you with a submission form; enter your e-mail address and submit the form.
2. Keep prefilled values on the next screen (Fig. 10.6.10) and submit the form.
3. You will be prompted with a list of possible transcription factor binding sites (matrices) for which to check; find and check the box labeled IRF1 and submit the form.

   Note that the more transcription factor binding sites you select for the analysis, the slower the program works.

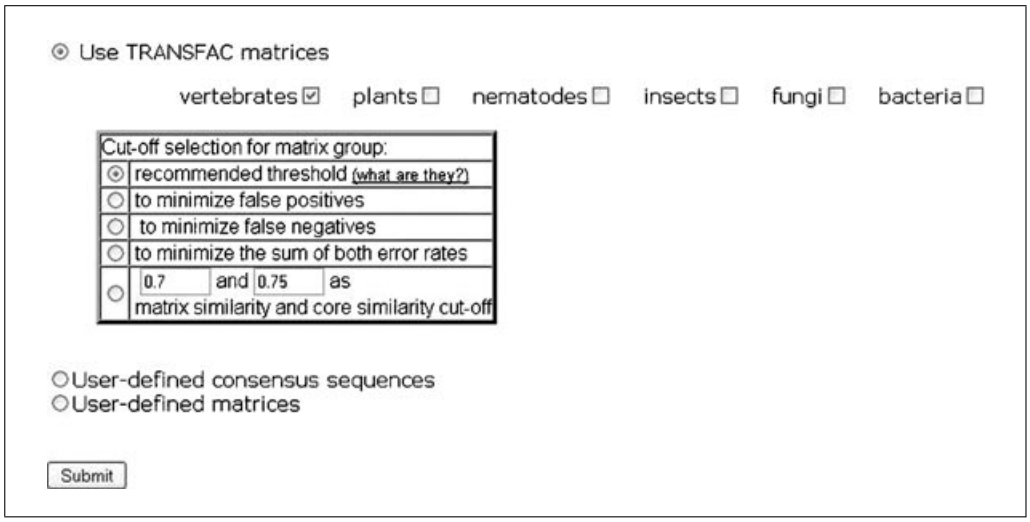


Figure 10.6.10 The rVISTA submission options. rVISTA makes predictions by the Match program based on the TRANSFAC Professional database, user-defined consensus sequences, or user-defined matrices. TRANSFAC searches are performed using a default matrix-similarity value of 0.70 and a core similarity value of 0.75. Please consult the extensive help page supplied with the tool on the explanation of the TRANSFAC cut-off selections and on the format of user-defined consensus sequences or matrices.

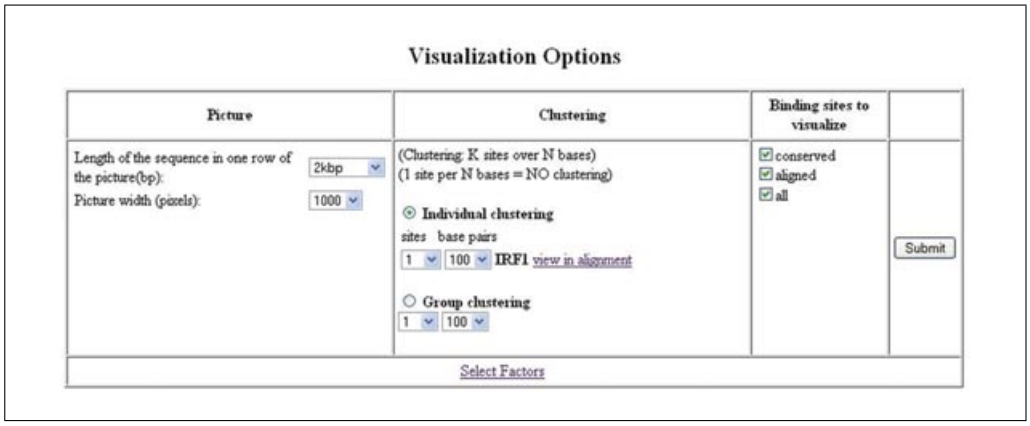


Figure 10.6.11 The rVISTA visualization options. Clustering allows the users to identify transcription factor binding sites that are present in groups or clusters. Conserved binding sites are defined as predicted binding sites located in the sequence fragments conserved between two species at the level of over 80% over a 24 bp window. Aligned binding sites are those where core positions of the potential binding sites on the sequences corresponded to each other in the alignment. "All" binding sites shows all sites, regardless of the alignment and conservation.

4. Upon job completion, you will receive an e-mail with a link to a form with the rVISTA visualization options (Fig. 10.6.11).
5. Select the following options:
   a. Check the "conserved," "aligned," and "all" boxes in the Binding sites to visualize column.
   b. Adjust picture parameters to display 2 kbp in one row, set picture width to 1000 pixels.
   c. Submit the form to look at the predicted transcription binding sites (shown as tick marks above a regular Vista curve; Fig. 10.6.12).

See http://genome.lbl.gov/vista/rvista/instructions.shtml for details on using rVISTA.

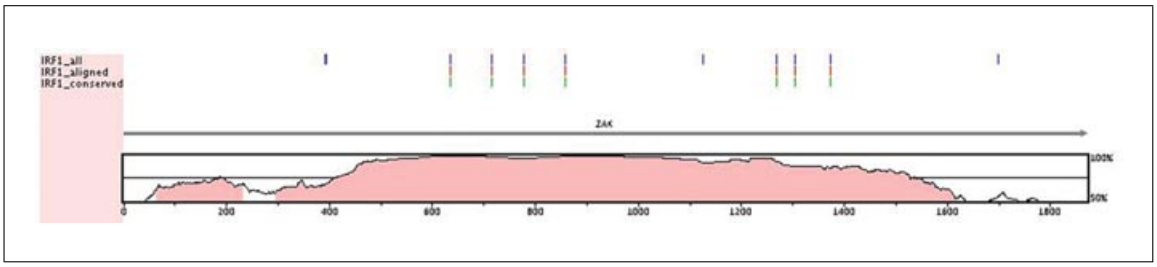


Figure 10.6.12 rVISTA results. Transcription factors binding sites are shown as tick marks above a regular VISTA curve. Green tick marks represent conserved binding sites, red represents aligned sites, and blue tick marks represent all found sites. For the color version of this figure go to *http://www.currentprotocols.com*.

Here, we have shown how to use rVISTA with Text Browser; you can also access it from the mVISTA server while analyzing your sequences, or use it directly. All VISTA servers are accessible from the main VISTA page, at http://genome.lbl.gov/vista, or through the Servers link in the VISTA gateway page.

Genome-wide search and analysis of regulatory elements, not covered in these protocols, can be carried out by using the Whole Genome rVISTA application, accessible from the VISTA home page.

FINDING EXPERIMENTALLY VERIFIED ENHANCERS USING THE VISTA ENHANCER BROWSER

BASIC PROTOCOL 6

The VISTA Enhancer Browser is a central resource for experimentally validated human noncoding fragments with gene enhancer activity, as assessed in transgenic mice. In the following protocol, we will find positive enhancers that have been experimentally verified to be expressed in the neural tube. One of these elements is located in the intron of the *ZAK* gene, and is used as an example in our protocols.

Necessary Resources

Hardware

Computer with Internet access

Software

Web browser

1. Click on the Enhancer DB tab in the navigation bar at the top of the VISTA gateway page (Fig. 10.6.1). On the Enhancer page (Fig. 10.6.13), click on the Advanced Search link on the bottom of the page.
2. Fill in the advanced search form as follows (Fig. 10.6.14):
   a. Check the box next to “neural tube” as the Expression pattern.
   b. Check the “ultra” box for highly conserved elements together with “chicken.”
   c. Select the radio button to restrict search to the elements, positive for enhancer activity by selecting the Positives option.

   "Ultra" elements are those conserved in human/mouse/rat at 100% identity over 200 base pairs or more.
3. Click the Search button and examine the resulting table (Fig. 10.6.15). Please note that a site on the bottom of the figure with the *ZAK* gene on it was used earlier as an example in our protocols for the location on Human, May 2004 genome assembly (see Fig. 10.6.9).

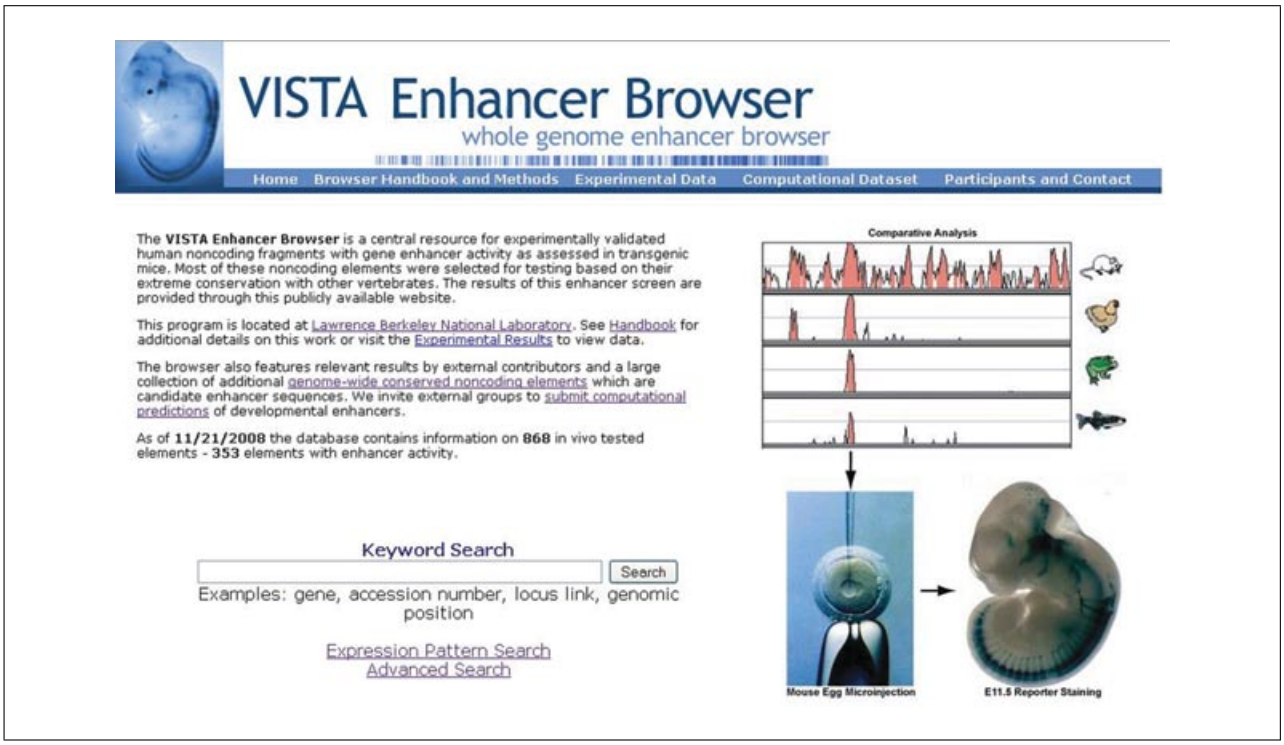


Figure 10.6.13 VISTA Enhancer Browser home page. Searchable keywords include Gene Symbols, GenBank Accession Numbers, and Entrez Gene Numbers. Additional search options are available in the Advanced Query form.

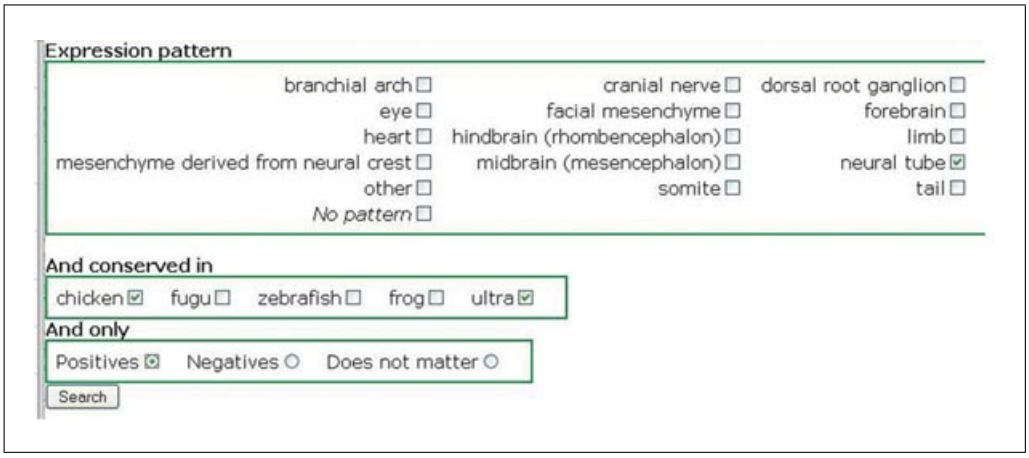


Figure 10.6.14 VISTA Enhancer Browser advanced query form. Restricting searches to ultra-conserved elements (i.e., conserved in human/mouse/rat at 100% identity over 200 base pair or more), also conserved in human-chicken, expressed in the neural tube, positive for enhancer activity.

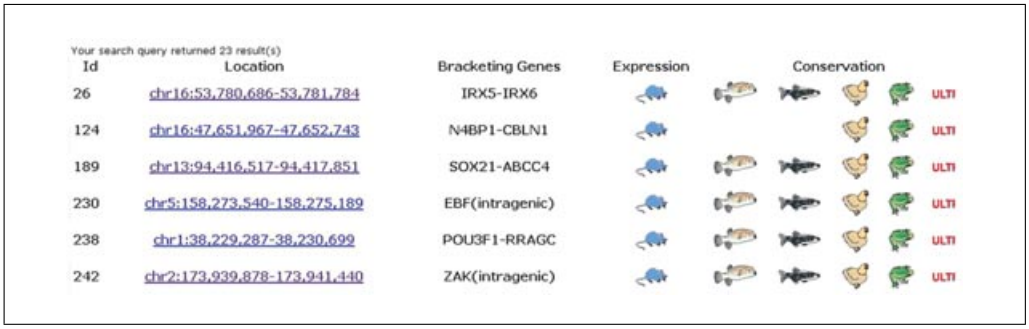


Figure 10.6.15 VISTA Enhancer Browser search results. Search was restricted to ultra-conserved elements, also conserved in human-chicken, expressed in the neural tube, positive for enhancer activity.

4. Click the Location link to further examine the expression pattern and view the experimental data. Consult *http://enhancer.lbl.gov/aboutproject.html* for more information on using the Enhancer Browser.

UNDERSTANDING RESULTS

We have used the region of the *ZAK* gene as an example throughout this document in different informational contexts. Table 10.6.1 provides an informational context of each tool presented in the document.

Table 10.6.1 VISTA Tools Informational Contexts

Tool	Informational context	Links to
VISTA Genome Browser, graph panel	Gene annotation, repeats, SNPs, conservation curves	Text Browser, additional external browsers
VISTA Genome Browser, base-pair level panel	Gene annotation, repeats, SNPs, SNP summary, sequences	UCSC and PolyPhen SNP reports
Text Browser	Conservation curves, genome location, alignment length	Gene annotation, alignment in MFA, blast-like or PDF format, rankVISTA, CNS, rVISTA, sequence, VISTA Browser, UCSC VISTA Track
Genome VISTA	Orthologous location on base genome	Text Browser, VISTA Browser, UCSC VISTA Track
rVISTA	Putative transcription factor binding sites plotted above a VISTA conservation curve	Alignment with marked TFBSs
Enhancer Browser	Enhancer location, image of expression pattern in embryo, conservation, type of enhancer activity, flanking genes	Sequences, additional results of comparative analysis

COMMENTARY

Background Information

VISTA tools are based on a solid algorithmic platform. Several powerful algorithms and software for the alignment of long genomic sequences and whole-genome assemblies have been developed in the group. Among them are AVID (Bray et al., 2003), LAGAN (Brudno et al., 2003a), Shuffle-LAGAN (Brudno et al., 2003b), and several global-local whole-genome alignment methods (Couronne et al., 2003; Brudno et al., 2004; Dubchak et al., 2009). VISTA plot, introduced in the early publication (Mayor et al., 2000), had quickly become very popular among biologists due to its comprehensible display, which allows for easy understanding of comparative results. This signature plot unifies all VISTA tools, allowing for easy comparison of results obtained by different tools.

All VISTA tools are supplied with extensive help pages and background information. A full list of the VISTA group publications, describing different tools, is available in the Publication section of the main VISTA page. The same section contains a long list of selected studies in different areas of genomics and genetics that present results obtained with the VISTA tools.

Critical Parameters and Troubleshooting

VISTA genome browser

Retrieving a large region of a genome may slow down the performance of the browser; the

upper limit size of a region on a base genome is 5 megabases long.

Though a VISTA conservation curve is built upon the underlying alignment data, its visual representation can be altered by several control parameters. Among those parameters are the size of the sliding window used to calculate conservation scores, the minimum width of a conserved region, and the minimum percent identity over the window for a region to be considered conserved. For details, please see the Changing curve parameters section of Basic Protocol 1.

Base pair level alignment panel

In order to display this panel, the region on a base genome should be 2 kilobases long or smaller, and there should be precisely one pair-wise or multiple alignment displayed in the browser. In case of several alignments in a region, the panel displays the one that has the highest score; you can view alternative alignments by changing a score selection.

Genome VISTA server

The upper limit on the size of a user-submitted sequence is 300 kilobases.

rVISTA server

Note that the large number of selected TRANSFAC factors submitted for an analysis slows down the calculation.

Acknowledgments

The VISTA project is an ongoing collaborative effort of a large group of scientists and engineers. It has been developed and maintained in the Genomics Division of Lawrence Berkeley National Laboratory. You can find the names of all contributors at the VISTA Web site (*http://genome.lbl.gov/vista/aboutus.shtml*).

This work was performed under the auspices of the U.S. Department of Energy's Office of Science, Biological, and Environmental Research Program and by the University of California, Lawrence Livermore National Laboratory under contract no. DE-AC52-07NA27344, Lawrence Berkeley National Laboratory under contract no. DE-AC02-05CH11231, and Los Alamos National Laboratory under contract no. DE-AC02-06NA25396. This work was partially supported by grant no. HL88728, Berkeley-PGA, under the Programs for Genomic Application, funded by National Heart, Lung, and Blood Institute, USA

Literature Cited

Bray, N., Dubchak, I., and Pachter, L. 2003. AVID: A global alignment program. *Genome Res* 13:97-102.

Brudno, M., Do, C.B., Cooper, G.M., Kim, M.F., Davydov, E., Green, E.D., Sidow, A., and Batzoglou, S. 2003a. LAGAN and Multi-LAGAN: Efficient tools for large-scale multiple alignment of genomic DNA. *Genome Res.* 13:721-731.

Brudno, M., Malde, S., Poliakov, A., Do, C.B., Couronne, O., Dubchak, I., and Batzoglou, S. 2003b. Global alignment: Finding rearrangements during alignment. *Bioinformatics* 19:i54-i62.

Brudno, M., Poliakov, A., Salamov, A., Cooper, G.M., Sidow, A., Rubin, E.M., Solovyev, V., Batzoglou, S., and Dubchak, I. 2004. Automated whole-genome multiple alignment of rat, mouse, and human. *Genome Res.* 14:685-692.

Couronne, O., Poliakov, A., Bray, N., Ishkhanov, T., Ryaboy, D., Rubin, E., Pachter, L., and Dubchak, I. 2003. Strategies and tools for whole-genome alignments. *Genome Res.* 13:73-80.

Dubchak, I., Poliakov, A., Kislyuk, A., and Brudno, M. 2009. Multiple whole-genome alignments without a reference organism. *Genome Res.* 19:682-689.

Hubbard, T., Barker, D., Birney, E., Cameron, G., Chen, Y., Clark, L., Cox, T., Cuff, J., Curwen, V., Down, T., Durbin, R., Eyras, E., Gilbert, J., Hammond, M., Huminiecki, L., Kasprzyk, A., Lehvaslaiho, H., Lijnzaad, P., Melsopp, C., Mongin, E., Pettett, R., Pocock, M., Potter, S., Rust, A., Schmidt, E., Searle, S., Slater, G., Smith, J., Spooner, W., Stabenau, A., Stalker, J., Stupka, E., Ureta-Vidal, A., Vastrik, I., and Clamp, M. 2002. The Ensembl genome database project. *Nucleic Acids Res.* 30:38-41.

Karolchik, D., Kuhn, R.M., Baertsch, R., Barber, G.P., Clawson, H., Diekhans, M., Giardine, B., Harte, R.A., Hinrichs, A.S., Hsu, F., Kober, K.M., Miller, W., Pedersen, J.S., Pohl, A., Raney, B.J., Rhead, B., Rosenbloom, K.R., Smith, K.E., Stanke, M., Thakkapallayil, A., Trumbower, H., Wang, T., Zweig, A.S., Haussler, D., and Kent, W.J. 2008. The UCSC Genome Browser Database: 2008 update. *Nucleic Acids Res.* 36:D773-D779.

Loots, G.G., Ovcharenko, I., Pachter, L., Dubchak, I., and Rubin, E.M. 2002. rVista for comparative sequence-based discovery of functional transcription factor binding sites. *Genome Res.* 12:832-839.

Markowitz, V.M., Szeto, E., Palaniappan, K., Grechkin, Y., Chu, K., Chen, I.M., Dubchak, I., Anderson, I., Lykidis, A., Mavromatis, K., Ivanova, N.N., and Kyrpides, N.C. 2008. The integrated microbial genomes (IMG) system in 2007: Data content and analysis tool extensions. *Nucleic Acids Res.* 36:D528-D533.

Mayor, C., Brudno, M., Schwartz, J.R., Poliakov, A., Rubin, E.M., Frazer, K.A., Pachter, L.S., and

Dubchak, I. 2000. VISTA: Visualizing global DNA sequence alignments of arbitrary length. *Bioinformatics* 16:1046-1047.

Ramensky, V., Bork, P., and Sunyaev, S. 2002. Human non-synonymous SNPs: Server and survey. *Nucleic Acids Res.* 30:3894-3900.

Wheeler, D.L., Barrett, T., Benson, D.A., Bryant, S.H., Canese, K., Chetvernin, V., Church, D.M., DiCuccio, M., Edgar, R., Federhen, S., Geer, L.Y., Helmberg, W., Kapustin, Y., Kenton, D.L., Khovayko, O., Lipman, D.J., Madden, T.L., Maglott, D.R., Ostell, J., Pruitt, K.D., Schuler, G.D., Schriml, L.M., Sequeira, E., Sherry, S.T., Sirotkin, K., Souvorov, A., Starchenko, G., Suzek, T.O., Tatusov, R., Tatusova, T.A., Wagner, L., and Yaschenko, E. 2007. Database resources of the National Center for Biotechnology Information. *Nucleic Acids Res.* 35:D5-D12.