prrn (ver.4.0.4)

Multiple sequence alignment by the doubly nested randomized iterative strategy (for UNIX)

Osamu Gotoh
<<o.gotoh@aist.go.jp>>

Computational Biology Research Center (CBRC)
National Institute of Advanced Industrial Science and Technology (AIST)
AIST Tokyo Waterfront Bio-IT Research Building
2-4-7 Aomi, Koto-ku, Tokyo, 135-0064, Japan

Original version: August 1992
Ver. 3.1.0b: March 2000
Ver. 3.1.1b: December 2004
Ver. 3.1.0: December 2009
Ver.4.0.0: January 2013
Ver.4.0.1: July 17 2013
Ver.4.0.4: June 2 2014

Overview

The program **prrn** is an implementation of the randomized iterative strategy for multiple sequence alignment (MSA). The older protein version **'prrp'** has been merged into **prrn**, which accepts either nucleotide or protein sequences. **Prrn** repeatedly uses pairwise group-to-group alignment to improve the overall weighted sum-of-pairs score at each iterative step, where the pair weights are introduced to correct for uneven representations of the sequences to be aligned. The doubly-nested randomized iterative (DNR) method described in Ref. [8] tries to make alignment, phylogenetic tree and pair weights mutually consistent. This is a hill-climbing strategy, and the true optimum may not be attained. The strategy works most effectively for refining a crude alignment obtained by other more rapid methods, e.g. progressive alignment. Upon reading the input sequences, **Prrn** judges whether they are already pre-aligned. For a pre-aligned input, **Prrn** immediately starts iterative refinement. Otherwise, **Prrn** first constructs an initial alignment by a progressive method, and then enters into the iterative refinement phase.

The major feature incorporated into version 4 is a scoring scheme that favors conserved exon-intron organizations of the parental genes. **Prrn** version 4 also supports a double affine gap penalty function first introduced into the brother MSA program **Prime** [10]. In addition, many parts of the program were rewritten in C++ language, though the conversion is not yet completed.

Run Prrn

It is quite easy to run **Prrn** if it has been properly installed. The command:

```
% prrn [options] seq1 [seq2 … seqN]
```

will report the resulting multiple alignment of seq1, seq2,□..., seqN to stdout. Each input may be a pre-aligned or unaligned group of sequences. If N = 1, seq1 must contain at least two sequences. If N > 1 and `-U` option is set, the pre-alignment is "thawed" and iterative optimization is performed for individual sequences. Otherwise, iterative optimization is performed for groups of sequences, where each intra-group alignment remains "frozen". Even if more than two sequences have the same label, they are treated as distinct entities in the default setting. However, older sequence(s) (those in left groups in the argument list) is replaced by the newest (right-most) sequence of the same label when the –U option is set. This option is particularly useful to improve MSA and gene structure prediction in a concerted manner. Details of the command options, file formats, and another way to run **`Prrn`** are described below.

System

The programs are written in C++, and distributed as source code. Users must compile the programs on their own systems. Although the programs have been tested only on a few platforms, they are likely to be portable to most UNIX systems with little or no modifications. The distribution package also contains source codes for a few related programs, including **`aln`**, which is used for pairwise sequence alignment between single or pre-aligned groups of sequences. By following the instructions below, you should obtain executable forms of all of the programs in the package.

Install

After decompressing and extracting `prraln.xxx.tar.gz`, you will find seven directories under `./prralnxxx` directory (`xxx` is a version code):

`bin`: binary files will be stored
`doc`: contains this document
`perl`: perl scripts
`sample`: sample sequences in various legal formats
`seqdb`: indexed sequences are stored
`src`: source codes
`table`: score matrix of amino-acid substitutions and other tables

To compile

(1) Move to the directory `'./prralnxxx/src'`.
(2) `% ./configure`
(3) Edit the first line in `Makefile` so that "exec_prefix" defines the directory to which the executables are copied.
(4) Edit the second line in `Makefile` so that "table_dir" defines the directory where the contents of "table" are copied.
(5) If CC indicates a C compiler, edit `Makefile` so that CC indicates a C++ compiler (ex. g++).
(6) `% make all`.
(7) `% make install`. You must have the right to write into the ”`exec_prefix`” directory.
(8) `% make clearall`, which deletes all .o, .a and executables.

The "`make install`" command uses **`makmdm`** to create two files, "`mdm_cmp`" and "`mdm_mtx`", in the "`table`" directory. The present version of **`makmdm`** produces PET91 matrixes (Jones et al.

(1992) Comput. Applic. Biosci. 8, 275-282) by default. The `-y78` option forces it to create the original MDM78 matrixes of Dayhoff et al.

Sequence File

A sequence file may contain either a single sequence or a set of unaligned or pre-aligned sequences. Avoid file names starting with a numeral.

<Single sequence>

The format of sequence file is virtually the same as the FASTA format, with only a few differences. Each file may contain one or more protein or nucleotide sequence. A sequence name must be given with a prefix of '>' or '<'. The first token delimited by a white character is regarded as the sequence label. The rest of the line is ignored. The prefix '<' is allowed only for nucleotide sequences, and the sequence will be automatically reversed and complemented upon reading. A line starting with a semicolon `';'` is a comment line. A line starting with ";C" or ";M" has a special meaning described later in more detail. Only single-letter code is accepted for an amino acid. The nomenclature recommended by NC-IUB (Eur. J. Biochem. (1985) 150, 1-5) is followed to represent ambiguous nucleotides. The programs are not case-sensitive.

Sequence files in NBRF-PIR, SWISS-PROT, GenBank, and EMBL formats can also be accessed. These formats are recognized by the first word in the first non-comment line: 'ENTRY' for NBRF-PIR, 'LOCUS' for GenBank, and 'ID' for EMBL or SWISS-PROT. Like FASTA format, sequences in these formats can be concatenated to represent multiple sequences. Mixing different formats in a single file may cause unexpected troubles.

<Sequential format of multiple sequences>

The concatenation of multiple sequences in one of the above single-sequence formats may be used as a program input. Deletion characters are automatically padded at the end of a shorter sequence. If the sequences are pre-aligned, internal deletion characters are preserved. Two examples of sequential formats are shown below.

```
File 1:                    |      File 2:
                           |
>Seq1                      |      LOCUS  Seq1
;C (1..10, 101..200)       |      ORIGIN
aaatttcccggg               |             1 AAATTTCCCGGG
>Seq2                      |      //
atcgatcgatcgat             |      LOCUS  NCODE
; this is comment          |      ORIGIN
>NCODE                     |             1 ACMGR-SVTWYHKDBN
ACMGRSVTWYHKDBN            |      //
```

<Interleaved (native) format of multiple sequences>

The alignment produced by **`aln`** can be used as an input to **`aln`** or **`prrn`**, and this is the most common way to have access to sets of pre-aligned sequences. Thus, the format of an aligned sequence file is the same as the output format of **`aln`**. The first line in a file

must indicate the number of sequences, *N*, involved in the alignment. This number is obtained as the sum of numbers in square brackets, e.g., when the first line is

```
Seq1[3] - Seq2[4]
```

N is calculated to be 7. Subsequent lines up to the first blank line are ignored. The rest of the file is composed of one or more blocks of a fixed column width of less than 255 characters. Each block is composed of N 'sequence lines' and other (optional) 'non-sequence lines'. The general format of a sequence line is:

```
<Position> <Sequence|> <Name>
```

where <Position> is a numeral that indicates the sequence position of the first letter in <Sequence|> (Usually all <Position>s in the first block are 1, but it is not a prerequisite. Negative values are also accepted). A line lacking the <Position> field is regarded as a 'non-sequence line' and ignored upon reading. The *i*-th sequence line in the second block is concatenated to the *i*-th sequence line in the first block, and so on. There is no particular limit on *N* or the length, but the total number of characters to be stored is limited by `MAXAREA` defined in `src/seq.h`. Several examples of native format are provided in the sample directory.

<Special Characters>

Three characters, dash `'-'`, tilde `'~'` and caret `'^'` have special meanings in a multiple sequence file of the native format. A dash indicates a 'deletion' introduced by some alignment procedure. Be careful not to use a space or dot instead of a dash. Spaces and dots are simply ignored, so that the file may be interpreted in a totally unexpected way. A tilde means the same residue as that in the first sequence line on that column in the block. On the other hand, a caret means the same residue as that in the previous sequence line on that column. These ditto characters are convenient in representing an alignment of closely related sequences. Neither `'~'` nor `'^'` is allowed in the first sequence line in each block.

<Information on gene structures>

To represent the exon-intron structure of the parental gene, the FASTA file should be extended. A line starting with ';C' shows the exon boundaries (inclusive). More than one line may be used if necessary. The format after ";C " is essentially the same as that of Feature field of a GenBank file. Start and end positions of each exon are separated by two dots. Individual exons are delimited by a comma. The term 'complement' indicates that the corresponding gene lies in the complementary strand of the genomic sequence. An example is the second line in Seq1 shown above.

When the parental gene contains one or more frame shifts, an output from `aln` or `spaln` contains one or more lines starting with ';M' such as:

;M Deleted *n* chars at *p*
;M Insert *n* chars at *p*

The first line indicates that *n* nucleotides (*n* = 1 or 2) have been deleted from the parental genomic sequence beginning at site *p*. Likewise, the second line indicates that *n* blank characters are inserted after site *p* to maintain the open reading frame. Such kinds of information are uaed to properly juxtapose intron positions along the alignment.

For native format of multiple sequences, lines starting with ";B ", ";b ", and ";m " represent the information about organizations of corresponding genes. The first number that follows ";B ", *NP*, indicates the number of alignment positions where an intron intervenes at least one of the genes corresponding to the aligned protein or cDNA sequences. For a protein sequence, the position means that of coding nucleotide sequence so that phase as well alignment column is also significant. The second number in this line, *NI*, indicates the total number of introns. The numbers that follow ";b " indicate the number of sequences that contain the intron at the 1^{st}, 2^{nd}, ..., *NP*-th position. The numbers that follow ";m " present the list of sequences (1..*N*) that contain the intron at the 1^{st}, 2^{nd}, ..., *NP*-th position in this order. See the example ce13a.mfa in sample/pas directory.

User Operation

There are two methods to invoke a program. Although the conversational mode is currently disabled, that mode is described here for users of older and potentially future versions.

```
[1] prrn [options1] [options2]
[2] prrn [options1] [options2] seq1 [seq2 seq3 ...]
```

Options1 is specific to **prrn**, whereas options2 is common to other programs, and default values will be used if it is omitted. Method [1] is a 'conversational mode', which will be discussed in detail below. Method [2] is a 'command-line mode'; the calculation starts immediately using the sequence data in file(s) `seqn`. When two or more sequences or multiple sequences are given, they are combined and pre-aligned according to a progressive method. The "-b tree" option described below generates the initial alignment by a progressive method using the given tree as the guide tree.

Notice

At the beginning of each process, **prrn** reads the score table files. Follow the instructions above to tell the program where these files are located.

[1] Conversational mode (presently disabled)

This menu-driven mode provides an easy way to use the program. You may read a sequence file, change parameter values, or send the results to a desired device or file. However, the command-line mode is preferable after you have become familiar with **prrn**.

Menu

In response to the prompt, you can select one or more single-letter commands. Case is significant, except for a few commands. Numerals must be separated from other commands by one or more spaces, whereas other characters should be typed contiguously. The commands are processed from left to right. For example

```
Menu Prompt: 1 pgr
```

is equivalent to

```
Menu Prompt: 1
```

```
Menu Prompt: p
Menu Prompt: g
Menu Prompt: r
```

Sequence

When the programs successfully start, or when you select '1' as the menu command, input of a sequence is prompted. (If you add a plus sign, i.e. 1+, the new sequence is added (concatenated) to the existing one, otherwise the new sequence replaces the older one.) You should respond with

```
[[path][filename]] [start] [end] [attribute]
```

Any terms may be omitted, except start when you wish to specify end. When path is omitted, the default path is used (these can be specified with options2, see [2] instant mode below). Start and end specify the range of the sequence to be analyzed. The default values are 1 and the last position of the sequence, respectively. Some attribute terms are applicable only to a nucleotide sequence. Currently meaningful attributes are defined below.

^	: the sequence is complemented. (nucleotide sequence only)
-	: the sequence is reversed.
<	: the sequence is reverse-complemented.
%	: forces the sequences into profile.
A	: amino acid sequence.
P	: amino acid sequence.
D	: nucleotide sequence.
R	: nucleotide sequence.
N	: nucleotide sequence, delete ambiguous codes ('N').
T	: tron sequence. See ref. 5.
@*N*	: read coding parts only, according to the *N*-th CDS (GenBank only).
#*N*	: retain only the columns composed of >*N*% of non-deletion characters.

If there is no attribute character (default), the sequence remains as read. If filename is specified, the remaining terms are reset to the default values, otherwise non-specified terms are unchanged. Of course, filename should not be omitted in the beginning session.

Parameters

When you select the 'p' command in the menu, you can change the current parameter values, which are shown in parentheses. Carriage/Return terminates the input, leaving the remaining terms unchanged. Hence, only hit Carriage/Return if you do not want to change the current values shown in the line. The following tables show the default values and meaning of the parameters used.

Parameters for protein sequences

	Default	Meaning
pam	250	PAM level of Dayhoff's mutation data matrix MDM [0-300] step by 10.

`bias`	0	Value to be added to each element of MDM.
`scale`	1	Precision of the MDM matrix (>= 1).
`v`	2	Constant term of the gap-weighting function.
`u`	9	Proportional coefficient of the gap-weighting function for a gap shorter than or equal to k1.

Parameters for nucleotide sequences

`s[=]`	2	Similarity measure between identical nucleotides.
`s[#]`	-2	Similarity measure between different nucleotides.
`v`	4	Constant term of the gap-weighting function.
`u`	2	Proportional coefficient of the gap-weighting function.

Parameters common to nucleotide and protein sequences

`shldr`	100	Window size (>= 0), see note 1.
`series`	1	Number of iteration series. The result with the best score among the series is finally reported.
`omode`	0	Output mode, see note 2.
`algor`	0	Select the algorithm for group-to-group alignment. If 0, the program chooses conceivably the best one.
`dmode`	2	This mode controls the way multiple sequences are divided into two groups, see note 3.
`newrn`	1	Set the seed of a series of quasi-random numbers.
`group`	2	Mutual alignment between some sequences in the given alignment may be fixed, see note 4.

(note 1)
The program sets a window of size ($|M - N| + 2 *$ `shldr`) along the center of the main diagonal, where M and N are the lengths of the sequences.

(note 2)
omode = 1: Output multiple alignment
omode = 2: Output the results of outlier analysis
omode = 4: Output the normalized sum-of-score and weighted sum-of-score.
All of these messages are reported to `stdout`.

(note 3)
Mode = 1: A randomly chosen single sequence is aligned with the remaining (M-1) sequences. The average convergence takes O(M) steps.
Mode = 2: Choose only the divisions that bisect an unrooted tree representing the mutual relationship of the sequences in the given alignment. The average convergence takes O(2M) steps. (default)

(note 4)
Mode = 0: Previous grouping is used.
Mode = 1: Groups should be manually indicated.
Mode = 2: No grouping.

Go and Reprint

The 'g' command conducts the alignment process. The program prompts for where the result is to be output. At the prompt, select a number (-1, 0, 1, 2, 3, or 4, the default is 0). When you select 3 or 4, the destination file name is prompted (if you select 4, the file is opened in an append mode).

Quit

To exit from the program, type 'q' in response to the menu prompt.

[2] Instant mode

This mode is the simplest to use. The sequence file containing a set of unaligned sequences or a preliminary alignment(s) is given in the command line arguments, and the result is reported to SO or to the file specified by option -o (see below). Most of the command line options are also effective in the conversational mode, but they are described here since they are most likely to be used in this mode. Each option is specified by a dash followed by a single character. Some options take a second argument as shown below.

Options1		Default
-b[*S*]	tree	Pre-align by a progressive method using S as the guide tree.
-E*S*	SE	Destination of supplementary messages. No message if E is empty.
-G*N*	2	Grouping.
-H*N*	20	Threshold value used in finding conserved regions.
-I*N*	10	Maximum number of iterations in the outer loop.
-J*N*	1	*N*=1: UPGMA method; *N*=2: NJ method for tree.
-O*N*	1	Set output mode (omode) to *N*. 1: alignment 2: outlier information 4: normalized alignment scores.
-R*N*		Seed of the series of random numbers.
-S*N*	1	Number of iteration series.
-U		The members in seq1 are replaced by sequences of the same names in seqn (n>1). Seq1 must be a multiple alignment whose members should have unique names.
-po		Output only summary of the results of outlier analysis
-ps		The order of sequences in the output file is rearranged according to the calculated phylogenetic relationship.

Options2		Default
-A*N*	0	Use algorithm *N*= [0-5]. A value of other than 0 is not recommended.
-F*N*	1	Format of output. *N*=1-5: native; *N*=6: Phylip; *N*=7: GCG; *N*=8: GDE; N=9: Concatenated Fasta

-l*N*	60	Set lpw (# of residues per line) = *N* > 8.
-m*S*		Amino acid or nucleotide substitution matrix.
-o*S*	SO	Output resultant alignment to file.
-s*S*	./	Set the default path to sequence files to *S*.
-u*N*	2	Set gap-extension penalty u = *N*.
-v*N*	4/9	Set gap-opening penalty v = *N*.
-w*N*	100	Set shldr = *N*.
-yp*N*	250	Set pam = *N*.
-ye*N*	0	Background gap extension penalty.
-ym*N*	2	Set nucleotide match score.
-yn*N*	-2	Set nucleotide mismatch score.
-yJ*N*	10	Bonus given to a pair of matched intron positions.

(Note) SE and SO mean standard error (stderr) and standard output (stdout), respectively.

caveat

If you use the NJ method to construct a phylogenetic tree, a negative edge length may appear. No particular countermeasure has been worked out, so the program may stop unexpectedly or produce nonsensical results in such cases.

Major changes from versions 2.5

1. Some serious bugs were fixed. The frequency of core dumps, which often occurred when many or complex gaps were present, has been greatly reduced.
2. An "update mode" was added, with which certain members within an existing alignment are replaced by new ones. This mode is designed to cooperate with a eukaryotic gene structure prediction method so that predicted structures of a set of paralogous genes can be refined to yield progressively better alignment of translation products (Ref. 10).
3. After version 3.0, the single program **'prrn'** can accept either nucleotide or protein sequences. The program usually judges the type according to the composition of input characters.
4. After version 3.0, the objective score is maximized rather than minimized, so that the sign of a score value is opposite that reported by previous versions of the programs.
5. Rigorous optimal alignment between two groups of sequences is the key procedure in the present iterative strategy. An improved algorithm was devised, which simplified the codes and improved the execution rate by an average of about 20%.
6. When either of the two groups contains no internal gap (typically a single sequence), the alignment algorithm can be simplified and calculation time is considerably reduced. Distinct subroutines were prepared to incorporate these, which are found in more than half of tree-dependent partitioning.
7. Optional output formats now include GCG's MSF format (-F7).
8. Compatibility with Phylip format has been improved (-F6).

Major changes from versions 3.0

1. After version 3.1, the source codes are ANSI-C/C++ compatible but no longer

compatible with a KR-compiler.
2. `-L[N]` option is added. If set, terminal gaps are not penalized, i.e. semi-global alignment is obtained. However, this option sometimes produces unexpected results. Use this option only to refine an existing alignment.

Major changes from versions 3.1

1. Conservation of gene (exon-intron) structure can be taken into account by setting `-yJN` option ($N > 0$). This option is motivated by the fact that gene structures are generally better conserved than amino acid sequences.

Major changes from versions 4.0

1. Gene-structure-aware multiple protein sequence alignment (GSA-MPSA) is evoked without additional option if the information about exon-intron organizations of the corresponding genes is provided in the input sequence file(s).
2. Algorithm C in Ref 4, instead of Algorithm D is now used by default.
3. The -yl3 option can handle double affine gap penalty to allow for long gaps.
4. When an unaligned set of sequences are given as the input, prrn now construct an initial alignment by a progressive method.
5. The conversational mode described above is currently disabled.
6. The code is written in C++ and no longer compiled with a C compiler.

References

[1] Gotoh, O. (1982) "An improved algorithm for matching biological sequences." *J. Mol. Biol.* **162**, 705-708.
[2] Gotoh, O. (1990) "Optimal sequence alignment allowing for long gaps." *Bull. Math. Biol.*, **52**, 359-373.
[3] Berger, M.P., and Munson, P.J. (1991) "A novel randomized iterative strategy for aligning multiple protein sequences." *Comput. Applic. Biosci.* **7**, 479-484.
[4] Gotoh, O. (1993) "Optimal alignment between groups of sequences and its application to multiple sequence alignment." *Comput. Applic. Biosci.* **9**, 361-370.
[5] Gotoh, O. (1993) "Extraction of conserved or variable regions from a multiple sequence alignment." Proceedings of *Genome Informatics Workshop IV*, pp. 109-113.
[6] Gotoh, O. (1994) "Further improvement in group-to-group sequence alignment with generalized profile operations." *Comput. Applic. Biosci.* **10**, 379-387.
[7] Gotoh, O. (1995) "A weighting system and algorithm for aligning many phylogenetically related sequences." *Comput. Applic. Biosci.* **11**, 543-551.
[8] Gotoh, O. (1996) "Significant improvement in accuracy of multiple protein sequence alignments by iterative refinement as assessed by reference to structural alignments." *J. Mol. Biol.* **264**, 823-838.
[9] Gotoh, O. (1999) "Multiple sequence alignment: algorithms and applications." *Adv. Biophys.* **36**, 159-206.
[10] Yamada, S., Gotoh, O., Yamana, H. (2006) Improvement in accuracy of multiple sequence alignment using novel group-to-group sequence alignment algorithm with piecewise linear gap cost, *BMC Bioinformatics*, **7**, 524.
[11] Gotoh, O., Yamada, S., and Yada, T. (2006) Multiple Sequence Alignment, in *Handbook of Computational Molecular Biology,* (Aluru, S. ed.) Chapman & Hall/CRC, Computer and Information Science Series, Vol. 9, pp. 3.1-3.36.

[12] Gotoh, O. (2013) "Heuristic Alignment Methods, in *Multiple Sequence Alignment Methods, (Russel, D. ed.), Methods in Molecular Biology*, Vol. **1079**, pp. 29-44, Humana Press.
[13] Gotoh, O., Morita, M. Nelson, D.R. (2014) "Assessment and refinement of eukaryotic gene structure prediction with gene-structure-aware multiple protein sequence alignment", submitted

---

Osamu Gotoh <o.gotoh@aist.go.jp>
Last modified on 2014-06-02