

# Finding Homologs to Nucleic Acid or Protein Sequences Using the Framesearch Program

UNIT 3.2

Framesearch (Edelman et al., 1995) is an extension of the classic Smith-Waterman pairwise sequence comparison algorithm. As illustrated in the upper portion of Figure 3.2.1, when the classic Smith-Waterman search algorithm is used to compare a nucleotide query sequence against a database of peptide sequences, it compares the six possible translations of that nucleotide query sequence as peptide sequences against the peptide sequence database. Single-nucleotide *indels* (INsertion or DEletion errors) are not taken into account because the alignments are between whole codons translated into amino acids. On the other hand, as illustrated in the lower portion of Figure 3.2.1, the Framesearch algorithm includes the possibility of a frameshift error in its alignment algorithm, and therefore can find alignments that span different reading frames. Unfortunately, the great power of Framesearch in finding sequence similarity despite frameshift errors comes at a high cost in computing time. Since Framesearch is an extension of the Smith-Waterman algorithm, it is even more CPU-intensive than a Smith-Waterman search.

Basic Protocol 1 in this unit describes the use of Framesearch in the GCG Wisconsin Package environment to search a *protein* sequence database for sequences that are similar to a query *nucleotide* sequence. Basic Protocol 2 describes the use of Framesearch to search a *nucleotide* sequence database for sequences that are similar to a query *protein* sequence. Three Alternate Protocols describe ways to improve the speed of Framesearch and thus make it practical for routine use (Alternate Protocols 1, 2, and 3). The Support Protocol describes how to convert FASTA files to GCG format so they are suitable for Framesearch. Framesearch is especially appropriate for low-quality single-read nucleotide sequence data, such as ESTs (expressed sequence tags) or early drafts of genomic sequences; it does *not* offer any significant advantage over less CPU-intensive algorithms for relatively high-quality nucleotide sequences without many single-nucleotide insertion or deletion errors.

## FRAMESEARCH USING A NUCLEIC ACID QUERY SEQUENCE

**BASIC  
PROTOCOL 1**

This protocol describes the use of Framesearch in the GCG Wisconsin Package environment to search a *protein* sequence database for sequences that are similar to a query *nucleotide* sequence. Any user familiar with the GCG Package will find using Framesearch in that environment straightforward. Framesearch has recently been added to the algorithms supported by the SeqWeb version of the GCG Package (Accelrys, 2001), so users who prefer a Web-based interface may find it simpler to run Framesearch in the SeqWeb environment (if locally available) rather than at the command line as described in this protocol.

### *Necessary Resources*

#### *Hardware*

Framesearch can be run on any Unix or VMS system that has the Wisconsin Package installed; because it is so CPU-intensive, Framesearch should be run on the fastest computer available to the user

#### *Software*

GCG Wisconsin Package (v. 8.1 or higher)

**Finding  
Similarities and  
Inferring  
Homologies**

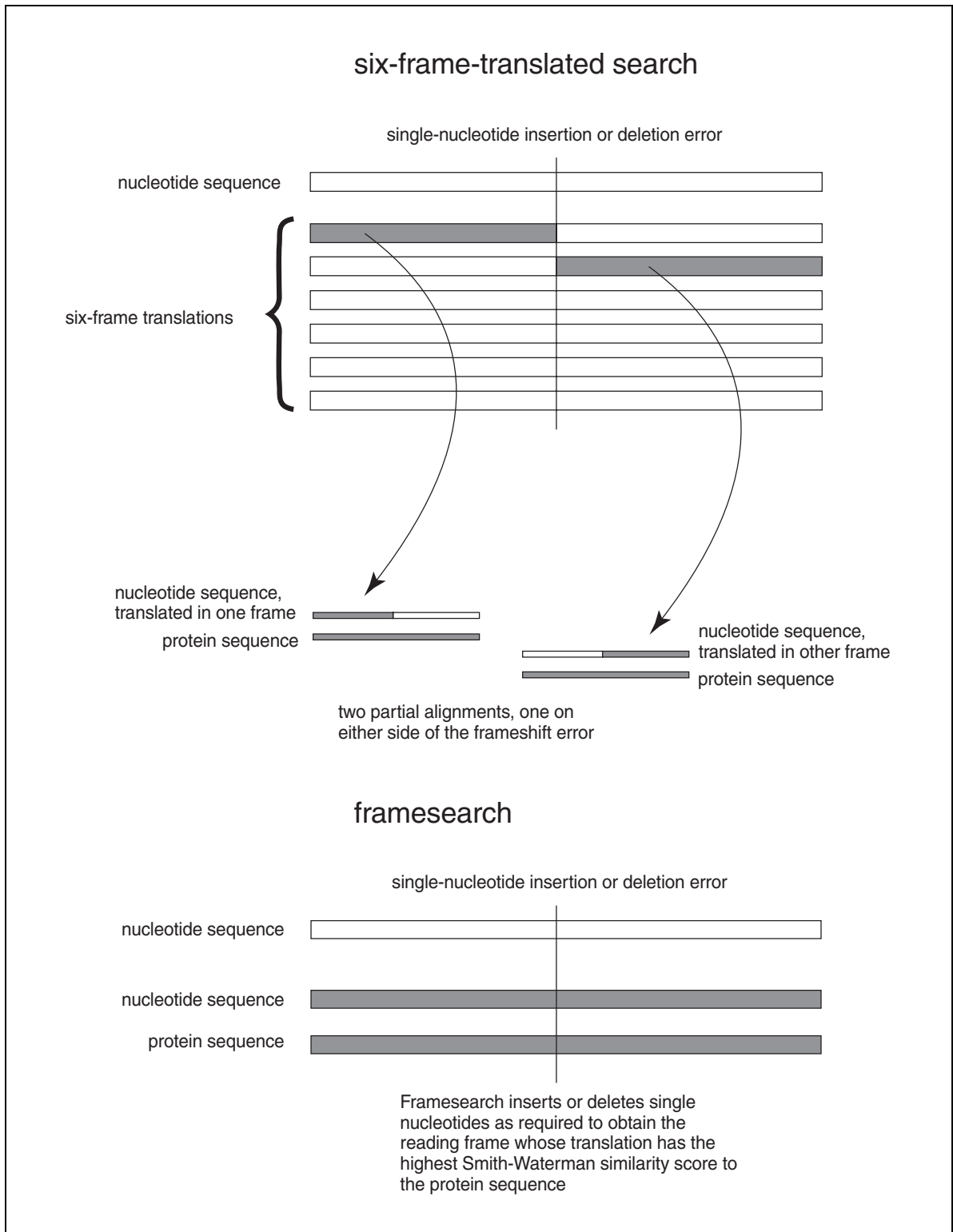
---

Contributed by Matthew Healy

*Current Protocols in Bioinformatics* (2003) 3.2.1-3.2.23

Copyright © 2003 by John Wiley & Sons, Inc.

**3.2.1**



**Figure 3.2.1** Six-frame-translated search versus Framesearch.

## Files

DNA sequence file of interest (this will be the query sequence; maximum length, 350 kb)

Protein database of sequences to which the DNA sequence will be compared

*For example, BA000007.faa contains the amino acid translations of all putative genes found in this bacterial genome by the lab where it was sequenced, as a single FASTA format text file (APPENDIX 1B).*

*Both the query sequence and the database files must be converted to the GCG format (Support Protocol).*

*The files used in this example should be downloaded from NCBI or from the Current Protocols Web site ([http://www3.interscience.wiley.com/c\\_p/cpbi\\_sampledatafiles.htm](http://www3.interscience.wiley.com/c_p/cpbi_sampledatafiles.htm)) and converted to GCG format, as described in the Support Protocol.*

1. Download and convert the sequence files as described in the Support Protocol below. To launch Framesearch, initialize the GCG Wisconsin Package environment and then type:

```
framesearch -check -batch
```

and hit the return key.

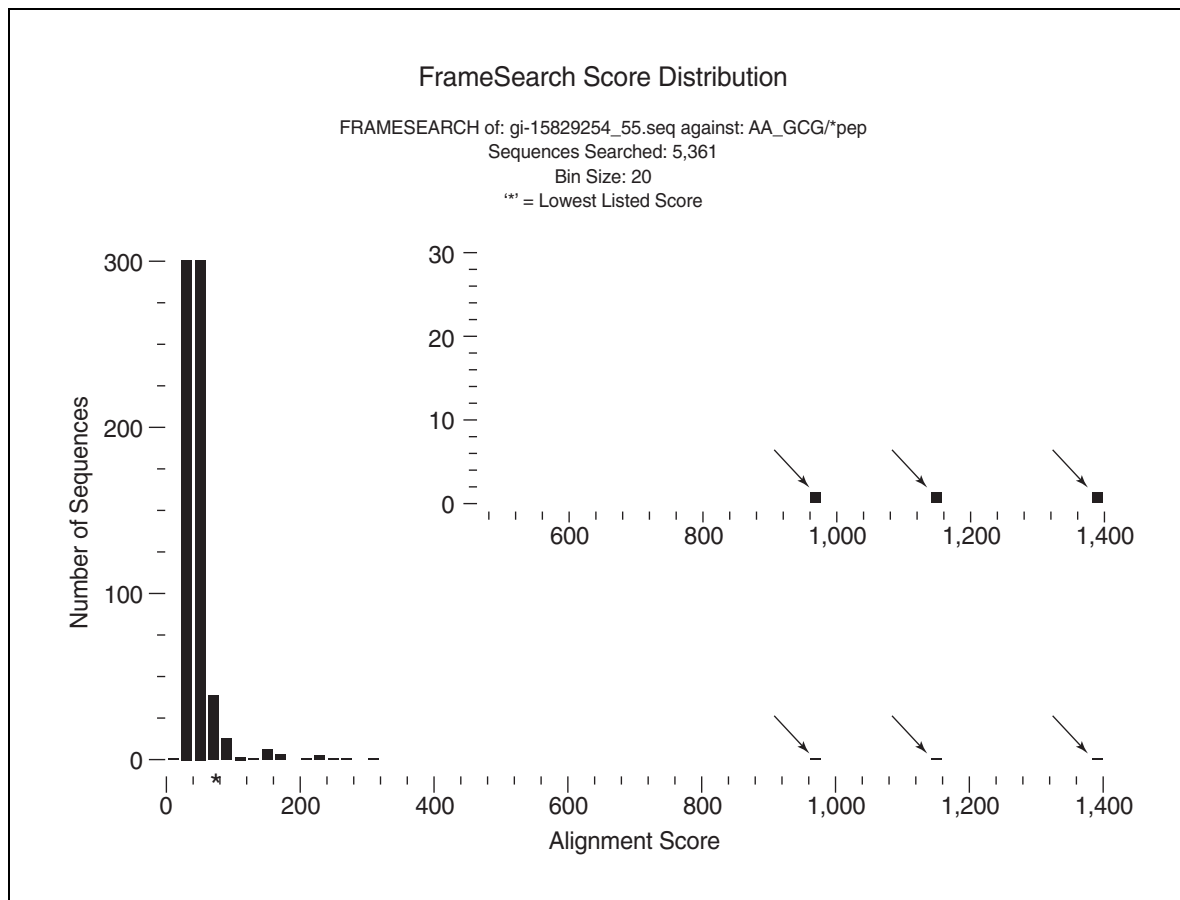
*The -batch parameter will make the actual search run in the background, which is desirable with such a CPU-intensive program, and the -check parameter will make Framesearch interactively prompt the user for the most commonly needed parameters before launching the actual search process in background mode. The documentation that comes with GCG explains all the parameters for Framesearch in exhaustive detail, but the -check parameter will prompt the user for the most common ones. As with all GCG programs, at many prompts a default value appears surrounded by parentheses and stars; one can accept that default by simply hitting the return key.*

*For example, when the Framesearch program asks Begin (\* 1 \*)?, by hitting the return key one accepts the default value of 1. Similarly, when the Framesearch program later asks what is the frameshift penalty (\* 0 \*)?, by hitting the return key one accepts the default value of 0.*

2. Enter the query sequence(s) at the prompt; for this example, type Nuc\_GCG/gi-15829254\_01.seq (the name given to the first part of the nucleotide sequence that was downloaded and converted to GCG format in the Support Protocol). Next, specify the beginning and ending nucleotide positions, defining the portion of the query sequence to use in the search.

*For this example, simply hit the return key after each of these two prompts, since the default values of Begin and End are the beginning and end of the entire query sequence. In Alternate Protocol 1, below, these parameters will be used to run Framesearch using only a portion of the query sequence.*

3. Specify what sequence(s) to search. For this example, type AA\_GCG/\*pep, which will make it search for regions of similarity to the query nucleotide sequence in all the peptide sequences that were downloaded in the Support Protocol.
4. Enter the gap creation, gap extension, and frameshift penalty values, and the name for the output file; for this example simply accept default values of all these parameters. (See Critical Parameters for discussion of these parameters).
5. Determine if a graphical histogram of Framesearch quality scores (Fig. 3.2.2) is desired. This plot can be an extremely valuable aid in the interpretation of Framesearch output. The Framesearch program will offer the user a choice of printing the plot on an HP7550 (an old device few sites today are likely to have), generating a FIGURE file, or not generating a plot at all.



**Figure 3.2.2** Distribution of scores generated by using Framesearch to compare nucleotides 52500 through 55000 of *gi-15829254\_55.seq* with all peptide sequences from the example bacterial genome. Since the selected region comprises all of one gene and parts of two flanking genes, there are three very strong hits, highlighted by arrows above. There are also many lower-quality hits with scores below 400. Most likely, hits with scores above 200 represent genes related to the three genes contained in this region, while hits with scores between 100 and 200 may represent borderline matches, but scores below 100 probably do not represent biologically significant matches.

*The suggested option is to generate a FIGURE file. However, to view this FIGURE file, the GCG graphical environment must also be configured. Unfortunately, the details of configuring GCG for graphical output are extremely installation-specific, so one must consult local support staff to learn them.*

6. Once all parameters are specified, Framesearch will search in background mode. Even on a relatively powerful computer, this may take as long as a day or more. When the search is done, the output file will appear in the directory where Framesearch was launched.

*For the reader's convenience, a sample copy of this output file can be found on the Current Protocols Web site at the address [http://www3.interscience.wiley.com/c\\_p/cpbi\\_sample\\_datafiles.htm](http://www3.interscience.wiley.com/c_p/cpbi_sample_datafiles.htm).*

*Figures 3.2.3 and 3.2.4 show the most important portions of the Framesearch output generated by this example. For a detailed discussion of these figures, see Guidelines for Understanding Results.*

```

**SEQUENCE_LIST 1.0
FRAMESEARCH of: /healy/FS/AA_GCG/gi-15829254_01.seq chunk: 918
from 1 to: 110000

gi 15829254_01.seq Continuation (1 of 99) of gi-15829254 from base 1
WFOOZSEK7
Sequence split into 95 fragments LOCUS gi-15829254
Fragment Name          Begin      End
gi-15829254_01.seq    1          110000
gi-15829254_02.seq    100001     210000 . . .

TC: AA_GCG/*pep Sequences: 5,361 Total-length: 1,609,188 December
5, 2001 04:14

Scoring matrix: GenBankData:bjquinn2.gap
Translation table: GenBankData:translate.txt

Gap creation penalty:      8
Gap extension penalty:    2
Frame shift penalty:      0

The best scores are:
..

/healy/FS/AA_GCG/gi-13359492.pe... 5450
/healy/FS/AA_GCG/gi-13359488.pe... 4973
/healy/FS/AA_GCG/gi-13359519.pe... 4971
/healy/FS/AA_GCG/gi-13359478.pe... 4342

[Middle portion of list cut for length]

/healy/FS/AA_GCG/gi-13359533.pe... 1870
/healy/FS/AA_GCG/gi-13359547.pe... 1864
/healy/FS/AA_GCG/gi-13359550.pe... 1619

**End of list

```

**Figure 3.2.3** The list of hits from a Framesearch run in which a nucleic acid sequence was used to search a number of peptide sequences. The name of the query sequence, the wildcard expression specifying the target sequences, and the name of the peptide sequence with the best match have been **boldfaced** in the sample output.

**FRAMESEARCH USING A PROTEIN QUERY SEQUENCE**

This protocol describes the use of Framesearch in the GCG Wisconsin Package environment to search a *nucleotide* sequence database for sequences that are similar to a query *protein* sequence. Any user familiar with the GCG Package will find using Framesearch in that environment straightforward. Framesearch has recently been added to the algorithms supported by the SeqWeb version of the GCG Package (Accelrys, 2001), so users who prefer a Web-based interface may find it simpler to run Framesearch in the SeqWeb environment (if locally available) rather than at the command line as described in this protocol.

**Necessary Resources**

*Hardware*

Framesearch can be run on any Unix or VMS system that has the Wisconsin Package installed; because it is so CPU-intensive, Framesearch should be run on the fastest computer available to the user

*Software*

GCG Wisconsin Package (v. 8.1 or higher)

**BASIC  
PROTOCOL 2**

**Finding  
Similarities and  
Inferring  
Homologies**

**3.2.5**

```

gi-15829254_01.seq
gi-13359492.pep
          Quality: 94.0          Length: 3319
          Ratio: 5.079          Gaps: 0
Percent Similarity: 100.000   Percent Identity: 100.000

15201 ATGCCAAAAGCGTACAGATATATAAAGTACCCCTGATTCCTGATCGGGAAC 15240
      ||| ||| ||| ||| ||| ||| ||| ||| |||
      1 MetProLysArgGlnAspIleIysSerIleLeuIleLeuGlyAlaGlyPr 17

15241 GATTGGCTATCGGTCAGGCGGTGTGAGTTTGACTACTCTGGGCGCCAAAGTAT 15290
      ||| ||| ||| ||| ||| ||| ||| ||| ||| |||
      18 GileValIleGlyGlnAlaCysGlnPheAspIleSerAlaGlnAlaG 34

15291 GTAAAGCCCTCGCGGAAAGGCGTACGTCGGTATCTGAGTCAACTGCTTAC 15340
      ||| ||| ||| ||| ||| ||| ||| ||| |||
      35 YsLysAlaLeuAspGluGluGlyTyrArgValIleLeuValArgSerAsn 50

```

[Middle portion of alignment omitted]

```

15391 TGCAGATAAAGTGCATATATGACGCGACGCTCAAGCCCTCCGCGTACC 15440
      ||| ||| ||| ||| ||| ||| ||| ||| |||
      1059 euGlnCysIysValHisCysAspIleThrLeuAspGlyGlyPheAlaThr 1069

15341 GCGATGGCGCTGATGCGGATGCGACTGAGGCGGAAATTCGCTCCAGCA 15390
      ||| ||| ||| ||| ||| ||| ||| ||| |||
      1051 AlaSerAlaLeuAspAlaAspAlaPheHisIleIysValIleSerValGlnG 1067

15391 AATGCAAGCAGATCAAA 15409
      ||| ||| ||| |||
      1058 uMetHisAlaGlnIleIys 1073

```

**Figure 3.2.4** Alignment of a nucleotide query sequence against a peptide database sequence, generated by Framesearch. Note that the middle portion has been omitted here. The names of the query and database sequences, just above this alignment, have been **boldfaced** for emphasis.

### Files

- Protein sequence file of interest (this will be the query sequence)
- Nucleic acid database of sequences to which the protein sequence will be compared

*For example, BA000007.fna contains the nucleotide sequence of all putative genes found in this bacterial genome by the laboratory where it was sequenced, as a single FASTA format text file (APPENDIX 1B).*

*Both the query sequence and the database files must be converted to the GCG format (Support Protocol).*

*The files used in this example should be downloaded from NCBI or from the Current Protocols Web site ([http://www3.interscience.wiley.com/c\\_p/cpbi\\_sampledatafiles.htm](http://www3.interscience.wiley.com/c_p/cpbi_sampledatafiles.htm)) and converted to GCG format, as described in the Support Protocol.*

1. Download and convert the sequence files as described in the Support Protocol below. To launch Framesearch, initialize the GCG Wisconsin Package environment, and then type:

```
framesearch -check -batch
```

and hit the return key.

The `-batch` parameter will make the actual search run in the background, which is desirable with such a CPU-intensive program, and the `-check` parameter will make Framesearch interactively prompt the user for the most commonly needed parameters before launching the actual search process in background mode. The documentation that comes with GCG explains all the parameters for Framesearch in exhaustive detail, but the `-check` parameter will prompt the user for the most common ones. As with all GCG programs, at many prompts a default value appears surrounded by parentheses and stars; one can accept that default by simply hitting the return key.

For example, when the Framesearch program asks `Begin (* 1 *)?`, by hitting the return key one accepts the default value of 1. Similarly, when the Framesearch program later asks what is the frameshift penalty `(* 0 *)?`, by hitting the return key one accepts the default value of 0.

2. Enter the name of the query sequence(s); for this example, type `AA_GCG/gi-13361126.pep` (the name that FROMFASTA gave to one of the protein sequences that was downloaded and converted to GCG format in the Support Protocol). Next, specify the beginning and ending residue positions, defining the portion of the query sequence to use in the search.

*For this example, simply hit the return key after each of these two prompts, since the default values of `Begin` and `End` are the beginning and end of the entire query sequence. In Alternate Protocol 2, below, these parameters will be used to run Framesearch using only a portion of the query sequence.*

3. Specify what sequence(s) to search against. For this example type `Nuc_GCG/*seq`, which will search for regions of similarity to the query protein sequence in all the nucleotide sequences that were downloaded in the Support Protocol.
4. Specify the gap creation, gap extension, and frameshift penalty values, and a name for the output file; for this example simply accept default values of all these parameters. (See Critical Parameters for discussion of these parameters).
5. Determine if a graphical histogram of Framesearch quality scores is desired. This plot can be an extremely valuable aid in the interpretation of Framesearch output. The Framesearch program will offer the user a choice of printing the plot on an HP7550 (an old device few sites today are likely to have), generating a `FIGURE` file, or not generating a plot at all.

*The suggested option is to generate a `FIGURE` file. However, to view this `FIGURE` file, the GCG graphical environment must also be configured. Unfortunately, the details of configuring GCG for graphical output are extremely installation-specific, so one must consult local support staff to learn them.*

6. Once all parameters are specified, Framesearch will search in background mode. Even on a relatively powerful computer, this may take as long as a day or more to run. When the search is complete, the output file will appear in the directory where Framesearch was launched.

*For the reader's convenience, a sample copy of this output file can be found on the Current Protocols Web site at the address [http://www3.interscience.wiley.com/c\\_p/cpbi\\_sample\\_datafiles.htm](http://www3.interscience.wiley.com/c_p/cpbi_sample_datafiles.htm).*

*Figures 3.2.5 and 3.2.6 show the most important portions of the Framesearch output generated by this example. For a detailed discussion of these figures, see *Guidelines for Understanding Results*.*

```

!!SEQUENCE_LIST 1.0

  FRAMESSEARCH of: /healy/FS/AA_GCG/gi-13361126.pap
  check: 4908   from: 1   to: 494

  gi|13361126|dbj|B835085.1   hypothetical protein [Escherichia coli
  O157:H7]

  TO: Nuc_GCG/*.*. Sequences: 55   Total-length:
  6,038,450   December 3, 2001 19:13

  Scoring matrix: GenRunData-blosum62.cmp

  Translation table: GenRunData:translate.txt

  Gap creation penalty:      8
  Gap extension penalty:    2
  Frameshift penalty:       0

  The best scores are:
  ..

  /healy/FS/Nuc_GCG/gi-15829254_1... 2597
  /healy/FS/Nuc_GCG/gi-15829254_4... 66
  /healy/FS/Nuc_GCG/gi-15829254_2... 64
  /healy/FS/Nuc_GCG/gi-15829254_5... 62
  [Middle portion of list cut for length]

  /healy/FS/Nuc_GCG/gi-15829254_4... 53
  /healy/FS/Nuc_GCG/gi-15829254_3... 53
  /healy/FS/Nuc_GCG/gi-15829254_0... 53
  /healy/FS/Nuc_GCG/gi-15829254_1... 53

  \\End of list

```

**Figure 3.2.5** The list of hits from a Framesearch run in which an amino acid sequence was used to search a number of nucleotide sequences. The name of the query sequence, the wildcard expression specifying the target sequences, and the name of the nucleotide sequence with the best match have been **boldfaced** in the sample output.

**ALTERNATE  
PROTOCOL 1**

**PREFILTERING WITH A SEARCH ALGORITHM TO IMPROVE THE  
SPEED OF FRAMESEARCH WITH A NUCLEIC ACID QUERY SEQUENCE**

The major advantage of Framesearch is its facility at aligning protein sequences to nucleotide sequences that contain numerous single-nucleotide insertion or deletion errors. When a nucleotide sequence is compared to a protein sequence, most similarity search algorithms translate all six possible reading frames into protein, then perform a protein sequence comparison (UNIT 3.4). If a region of nucleotide sequence that is similar to a protein sequence is interrupted by a frameshift error, then what should be one match will become two shorter matches in different reading frames.

```

gi-15829254_17.seq /rev
gi-13361126.pap

      Quality:  2597          Length:   1483
      Ratio:    5.257         Gaps:     0
Percent Similarity: 100.000  Percent Identity: 100.000

60990 ATGGCGAAAATAATTACTTCATTTCAAAGTTGTTTAAAGTTACTTCTACC 60941
      ||| | | | | | | | | | | | | | | | | | | | | | | | | |
1 MetArgLysAlaIleIleThrEsnPheLysValValLeuThrLeuLeuLeuPa 17

60940 AGTAAAGCGTATCTGGCCAGCAGATACAGTGGCAATTCCTGTATGGCCAGTC 60891
      ||| | | | | | | | | | | | | | | | | | | | | | | | | |
18 ovalThrValSerAlaClnGlnIleGlnTrpGlnSerCysMetAlaSerG 34

60890 AATTCAGCACTGGTTTGGTGAGGAGAAACCGCTCTCTGACTTACTATGT 60841
      ||| | | | | | | | | | | | | | | | | | | | | | | | | |
35 InPheAsnIleTrpPheGlyGluGluLysProSerProAspLeuLeuCys 50

60840 GGTATTGTGTGTGTCCATTAAATATACAGACACAGCGGAGATGCTTC 60791
      |||| | | | | | | | | | | | | | | | | | | | | | | | |
52 GlyTyrLeuSerValProLeuLysTyrThrAspThrGlyGlyAspAlaSe 67

[Middle portion of alignment omitted]

59640 AATGGTGGATGGCCATACATTAGCTCTCCCGGAGTTAATTTATGTGTAGA 59591
      ||| | | | | | | | | | | | | | | | | | | | | | | | | |
451 AsnGlyAspGlyHisThrLeuAlaLeuThrGlyValAsnLeuCysValAs 467

59590 TAAAGCAGTTGTACATCAGCTGATCACTCCAGAAAACAGAAATATAT 59541
      |||| | | | | | | | | | | | | | | | | | | | | | | | |
463 pLysAlaValValHisHisLeuIleThrProGlnLysIleGluAsnIleT 484

59540 ACTGCCCCAGGAATTCGAAGCAGAAATACAA 59509
      |||| | | | | | | | | | | | | | | | | | | | | | | | |
485 yrCysProGlyAsnSerGluAlaGluIleGln 494

```

**Figure 3.2.6** Alignment of an amino acid query sequence against a nucleotide database sequence, generated by Framesearch. Note that the middle portion has been omitted here. The names of the query and database sequences, just above this alignment, have been **boldfaced** for emphasis. **Also note** that following the name of the nucleotide sequence in this example is the string “/rev”, which means this alignment is to the **reverse complement** of this nucleotide sequence.

However, unless the stretches of nucleotide sequence between single-nucleotide insertions or deletions (indels) are very short, a translated search with BLAST (UNIT 3.4) is likely to find at least part of the alignment, if not the whole stretch of similar sequence. BLAST (UNIT 3.4) will do a typical similarity search several orders of magnitude faster than Framesearch. Therefore, one can take advantage of the speed offered by BLAST (UNIT 3.4) to find possible hits, then use Framesearch to improve the alignment near each hit.

**Necessary Resources**

*Hardware*

Framesearch can be run on any Unix or VMS system that has the Wisconsin Package installed; because it is so CPU-intensive, Framesearch should be run on the fastest computer available to the user

*Software*

- GCG Wisconsin Package (v. 8.1 or higher)
- BLAST program (UNIT 3.4)

In the GCG environment assumed for these examples, both BLAST and Framesearch are included.

### Files

DNA sequence file of interest (this will be the query sequence; maximum length, 350 kb)

Protein database of sequences to which the DNA sequence will be compared

*For example, BA000007.faa contains the amino acid translations of all putative genes found in this bacterial genome by the lab where it was sequenced, as a single FASTA format text file (APPENDIX 1B).*

*Both the query sequence and the database files must be converted to the GCG format (Support Protocol).*

*The files used in this example should be downloaded from NCBI or from the Current Protocols Web site ([http://www3.interscience.wiley.com/c\\_p/cpbi\\_sampledatafiles.htm](http://www3.interscience.wiley.com/c_p/cpbi_sampledatafiles.htm)) and converted to GCG format, as described in the Support Protocol.*

1. Download the files, set up the GCG environment, and run FROMFASTA on the data files as described in the Support Protocol below.
2. Use your nucleotide sequence as the query in BLAST searches of the appropriate protein database (see *UNITS 3.3 & 3.4* for more about BLAST).

*If a BLAST search using the default parameters does not find any matches, it may be necessary to increase the cutoff E-score to a value greater than 1, possibly even as high as 10 or more, in order to obtain at least one match. Note which regions of the nucleotide sequence are similar to which protein sequences.*

3. For each BLAST hit found in step 2, run a Framesearch job as described above in Basic Protocol 1, but modified so it only uses a *portion* of the nucleotide query sequence to search a *single* peptide database sequence. In the Framesearch job corresponding to each BLAST hit, use the Begin and End parameters to specify a region of the nucleotide sequence that brackets the region of nucleotide sequence where BLAST found a match. Instead of giving a wildcard expression that makes it search thousands of protein sequences, specify only the name of the protein sequence from this BLAST hit.

*For example, from BLAST searches one can determine that nucleotides 52500 through 55000 of Nuc\_GCG/gi-15829254\_55.seq correspond to all of the amino acid sequence AA\_GCG/gi-133795.pep, plus portions of two flanking genes. Therefore, one can use Framesearch to compare only this region of Nuc\_GCG/gi-15829254\_55.seq with AA\_GCG/gi-133795.pep by responding to Framesearch prompts as follows:*

*when asked Framesearch with what query sequence(s)?, type Nuc\_GCG/gi-15829254\_55.seq and hit the return key. When asked Begin (\* 1 \*)?, type 52500 and when asked End (\* 98450 \*)?, type 55000. When asked Search for query in what sequence(s)? type AA\_GCG/gi-133795.pep. Accept default values for all remaining prompts by hitting the return key after each prompt.*

*Because each of these Framesearch runs is comparing a portion of the nucleotide query sequence to a single protein sequence, instead of comparing the entire nucleotide query sequence to all the protein sequences, this method will take dramatically less time than Basic Protocol 1.*

4. The output files from these Framesearch runs will be very similar to the output files generated by Basic Protocol 1 discussed above. The interpretation of Framesearch output is discussed later in this unit (see Guidelines for Understanding Results).

## PREFILTERING WITH A SEARCH ALGORITHM TO IMPROVE THE SPEED OF FRAMESEARCH WITH A PROTEIN QUERY SEQUENCE

The major advantage of Framesearch is its facility at aligning protein sequences to nucleotide sequences that contain numerous single-nucleotide insertion or deletion errors. When a nucleotide sequence is compared to a protein sequence, most similarity search algorithms translate all six possible reading frames into protein, then perform a protein sequence comparison (e.g., *UNIT 3.4*). If a region of nucleotide sequence that is similar to a protein sequence is interrupted by a frameshift error, then what should be one match will become two shorter matches in different reading frames.

However, unless the stretches of nucleotide sequence between single-nucleotide insertions or deletions (indels) are very short, a translated search with BLAST (*UNIT 3.4*) is likely to find at least part of the alignment, if not the whole stretch of similar sequence. BLAST (*UNIT 3.4*) will do a typical similarity search several orders of magnitude faster than can Framesearch. Therefore, one can take advantage of the speed offered by BLAST (*UNIT 3.4*) to find possible hits, then use Framesearch to improve the alignment near each hit.

### *Necessary Resources*

#### *Hardware*

Framesearch can be run on any Unix or VMS system that has the Wisconsin Package installed; because it is so CPU-intensive, Framesearch should be run on the fastest computer available to the user

#### *Software*

GCG Wisconsin Package (v. 8.1 or higher)  
BLAST program (*UNIT 3.4*)

*In the GCG environment assumed for these examples, both BLAST and Framesearch are included.*

#### *Files*

Protein sequence file of interest (this will be the query sequence)  
Nucleic acid database of sequences to which the protein sequence will be compared

*For example, BA000007.fn contains the nucleotide sequence of all putative genes found in this bacterial genome by the laboratory where it was sequenced, as a single FASTA format text file (APPENDIX 1B).*

*Both the query sequence and the database files must be converted to the GCG format (Support Protocol).*

*The files used in this example should be downloaded from NCBI or from the Current Protocols Web site ([http://www3.interscience.wiley.com/c\\_p/cpbi\\_sampledatafiles.htm](http://www3.interscience.wiley.com/c_p/cpbi_sampledatafiles.htm)) and converted to GCG format, as described in the Support Protocol.*

1. Download the files, set up the GCG environment, and run FROMFASTA on the data files as described in the Support Protocol below.
2. Use your protein sequence as the query in BLAST searches of the appropriate nucleotide database (see *UNITS 3.3 & 3.4* for more about BLAST).

*If a BLAST search using the default parameters does not find any matches, it may be necessary to increase the cutoff E-score to a value greater than 1, possibly even as high as 10 or more, in order to obtain at least one match. Note which regions of each nucleotide hit sequence are similar to the protein query sequence.*

3. For each BLAST hit found in step 2, run a Framesearch job as described above in Basic Protocol 1, but modified as follows. First, use the *nucleotide* sequence from each BLAST alignment as the query sequence, and specify the original protein sequence as the *sequence to be searched for matches to that query*. This reversal is because GCG Framesearch asks the user which portion of the query sequence to use for searching, so by using each database hit found by BLAST as a Framesearch query sequence, one can restrict the search to a particular region of that nucleotide sequence. When prompted for the Begin and End parameters of the nucleotide sequence, use values that surround the region where the alignment found by BLAST is located. Instead of giving a wildcard expression that makes it search thousands of protein sequences, specify only the name of the protein query sequence for the database to search.

*For example, from BLAST searches one can determine that nucleotides 53299 through 54102 of Nuc\_GCG/gi-15829254\_55.seq correspond to the amino acid sequence AA\_GCG/gi-133795.pep. Therefore, one can use Framesearch to compare only a region surrounding this portion of Nuc\_GCG/gi-15829254\_55.seq with AA\_GCG/gi-133795.pep by responding to Framesearch prompts as follows:*

*when asked Framesearch with what query sequence(s)?, type Nuc\_GCG/gi-15829254\_55.seq and hit the return key. When asked Begin (\* 1 \*)?, type 52500 and when asked End (\* 98450 \*)?, type 55000. When asked Search for query in what sequence(s)?, type AA\_GCG/gi-133795.pep. Accept default values for all remaining prompts by hitting the return key after each prompt. Note that although the initial query sequence used for the BLAST searches was a protein sequence, for Framesearch the nucleotide sequence is used as the query sequence, so that one has an opportunity to specify the nucleotide region of interest.*

*Because each of these Framesearch runs is comparing a portion of a single nucleotide database sequence to the protein sequence, instead of comparing the protein query sequence to the entire length of all nucleotide sequences in the database, this method will take dramatically less time than Basic Protocol 2.*

4. The output files from these Framesearch runs will be very similar to the output files generated by Basic Protocol 1 discussed above. The interpretation of Framesearch output is discussed later in this unit (see Guidelines for Understanding Results).

### ALTERNATE PROTOCOL 3

### IMPROVING SPEED OF FRAMESEARCH BY USING SPECIALIZED HARDWARE

While Alternate Protocols 1 and 2 make practical the routine use of Framesearch on an ordinary computer, it is not a perfect solution to the Framesearch performance problem. Restricting Framesearch to regions surrounding BLAST hits will permit alignments that are interrupted by frameshift errors to span reading frames, thus converting a short BLAST hit into a longer Framesearch hit. However, there may be hits that BLAST (UNIT 3.3 & 3.4) will not find at all, because no single reading frame has enough sequence identity for BLAST to find.

Several companies sell specialized computing hardware that can run Framesearch several orders of magnitude faster than can a typical general-purpose CPU, thus making it practical to run a full Framesearch on large amounts of sequence data.

Special-purpose genomics computing systems from several companies, including CompuGen (<http://www.cgen.com>), Paracel (<http://www.paracel.com>), and TimeLogic (<http://www.timelogic.com>), use massively parallel arrays of special processors to perform such CPU-intensive algorithms as Smith-Waterman, Framesearch, and hidden Markov models at speeds that bring these rigorous dynamic programming algorithms into the realm of practical daily use. For example, while the Framesearch run that generated

Alignment of a nucleotide query vs a protein database, 100% identity, using translated Smith-Waterman algorithm:

```
Q      1 MKKLCDFPKVWVLTLLLPVTVSAQQIQXQSCMASQFNHWFGREKPSPDLLCGYLSVPLKYYT
T      1 MKKLCDFPKVWVLTLLLPVTVSAQQIQXQSCMASQFNHWFGREKPSPDLLCGYLSVPLKYYT
```

```
Q      61 TGGDASYEKKSQVKLALTKLPA
T      61 TGGDASYEKKSQVKLALTKLPA
Q      61 TGGDASYEKKSQVKLALTKLPA
```

Same query sequence, with six nucleotides (chosen at random) deleted, best alignment found now by S-W algorithm:

```
Q      2 KNCDFPKVWVLTLLLPVTVSAQ  RYSGNFWYFNKSTFDGLWQRTVS+LTCMLFVCSIKVTD
T      2 K  CDFPKVWVLTLLLPVTVSAQ  ++      K  G  +  L  +L  V  IRYTD
Q      61 TGGDASYEKKSQV
T      61 TGGDASYEKKSQV
Q      61 TGGDASYEKKSQV
T      61 TGGDASYEKKSQV
```

Same query sequence, with six nucleotides (chosen at random) inserted, best alignment found now by S-W algorithm:

```
Q      3 K+LIEWVWVLDLDPVTVSAQQRTVAITVQQSIQPIAW*GNR---LLTYVWVLEDFH+
T      3 K+KRVWVLDLDPVTVSAQQ  +  +  Q  W  G  +  L  +  Y  +  L
Q      59 NTQFTGGDASYEKKSQVKLALTKL
T      59 NTQFTGGDASYEKKSQVKLALTKL
Q      59 NTQFTGGDASYEKKSQVKLALTKL
T      59 NTQFTGGDASYEKKSQVKLALTKL
```

**Figure 3.2.7** Illustration of how insertion and deletion errors affect alignments generated by the six-frame-translated Smith-Waterman algorithm. Note that this example was generated on a DeCypher genomics accelerator, manufactured by TimeLogic. SSEARCH in the GCG environment would give very similar results, in a slightly different format. The nucleotides selected are the reverse complement of those nucleotides from the *E. coli* O157:H7 genome, NCBI REFSEQ number NC\_002695, which correspond to amino acids 1 to 84 of the protein with NCBI gi number 13361126.

Figures 3.2.3 and 3.2.4 took more than 32 hours of CPU time on a large Unix server, the Framesearch runs that generated Figures 3.2.7 through 3.2.9 took only a few minutes each on a TimeLogic machine. Similar machines from Compugen and Paracel also run Framesearch at speeds that make it practical for interactive use.

The basic steps involved in using Framesearch on specialized genomic computing systems are very similar to those described in the Basic and Support Protocols of this unit: download the sequences, copy the database onto the special-purpose computer, select the algorithm and search parameters, and run the search. However, since the details are different for each brand of special-purpose computer, it is not possible to describe them here. Readers who have access to such hardware should refer to the documentation that comes with it.

Fortunately, all three vendors listed above include Web interfaces with their products. Therefore, these devices are quite simple to use for single searches. Paste the query sequence into a Web page, select the various options, and click a submit button.





BA000007.fna

*This file contains the nucleotide sequence of this bacterial genome, again as a single FASTA text file (APPENDIX 1B).*

3. In the directory where the two downloaded files have been placed, create two subdirectories called “AA\_GCG” and “Nuc\_GCG.” While this step is not essential, it will be much more convenient to put these files into their own subdirectories before running FROMFASTA, since the FROMFASTA steps below will create thousands of individual GCG-format sequence files.

4. Initialize the GCG environment. The exact commands may vary. For example:

```
source /gcg/gcgstartup
gcg
```

5. Copy or move the FASTA file BA000007.faa to the “AA\_GCG” subdirectory and put a copy of the FASTA file BA000007.fna into the “Nuc\_GCG” subdirectory (APPENDIX 1C).

6. Type the following commands to generate the GCG-format files needed for later steps:

```
cd AA_GCG
FROMFASTA
```

and the FROMFASTA program will ask FROMFASTA of what FASTA sequence file(s)? Type the name of the amino acid file to be converted:

```
BA000007.faa
```

and hit the return key.

*Then wait while FROMFASTA converts this FASTA file into 5361 individual GCG-format peptide files with names ending in “.pep” (FROMFASTA will correctly determine that these sequences are peptides).*

7. To convert the nucleotide sequence file, type the following commands:

```
cd ../Nuc_GCG_/
FROMFASTA
```

the FROMFASTA program will ask FROMFASTA of what FASTA sequence file(s)? Type the name of the nucleotide file to be converted:

```
BA000007.fna
```

and hit the return key.

*Wait while FROMFASTA converts this FASTA file into GCG sequence format; FROMFASTA will correctly recognize that the single sequence in this FASTA file is a nucleotide sequence. Because this nucleotide sequence is fairly long, being an entire genome, FROMFASTA will convert it into 55 overlapping chunks of 110000 bp, with an overlap of 10000 bp. (The filenames will be gi-15829254\_1.seq, gi-15829254\_2.seq, gi-15829254\_3.seq, and so on through gi-15829254\_54.seq and gi-15829254\_55.seq.).*

8. Change the working directory back to the starting directory by typing:

```
cd ..
```

*Now that the files have been converted into GCG format, the Protocols above can be performed on these sequences.*

## GUIDELINES FOR UNDERSTANDING RESULTS

The output from Framesearch begins with a list of hits similar to that shown in Figure 3.2.3, sorted by the quality score. For each of the top hits there is then an alignment section as shown in Figure 3.2.4, in which the alignment is preceded by some general statistics that summarize this match. The TimeLogic implementation of Framesearch adds “P-Scores,” statistical estimates of the probability that a given match could have occurred by chance.

By default, the GCG version of Framesearch will return the top 40 matches, ranked by the quality score. This can be changed using the parameters `-LISTsize` which controls the number of hits appearing in the list and `-ALIGN` which controls the number of complete alignments displayed. Other Framesearch implementations have different ways of specifying how many hits to show; for example on a TimeLogic DeCypher system one can specify a cutoff quality score or statistical probability score as well as the maximum numbers of scores and alignments to present.

There are a number of things one should consider in judging whether the degree of sequence similarity presented by a given pairwise alignment does or does not support the hypothesis that these two sequences are in fact homologous. These include the distribution of quality scores in the list, the P-Score value if one was computed, the distribution of scores in the histogram if one was printed, and the percentages of similarity and identity computed by the program. In the author’s experience, when a Framesearch alignment is generated on a TimeLogic DeCypher system the best overall indicator of biological significance is the P-Score. Hits with P-Score values of less than about  $1.0E-05$  are highly likely to represent valid matches, while hits with P-Score values of more than about 0.05 are questionable. When the P-Score falls in between these limits, assessing the validity of a hit is a judgment call; there is no hard-and-fast rule.

Since the GCG version of Framesearch does not compute a statistical probability estimate, determining the validity of a Framesearch hit generated by the GCG implementation is somewhat more difficult. One must therefore look through the hits to judge whether there is a point after which the hits are of dramatically lower quality than those with higher scores, rather as a teacher might look over a list of test scores to judge which are close enough to the top to merit the grade of “A.” For instance, in the example given for Basic Protocol 2 above, as shown in Figure 3.2.5, an *E. coli* protein sequence is compared with all nucleotide sequences from the same genome. The top hit has a quality score of 2597, and the alignment shown in Figure 3.2.6 shows that this hit is 100% identical. However, the second hit in this search has a quality score of just 66. Therefore, it would seem likely that only the first hit in this example is valid.

On the other hand, in the example given for Basic Protocol 1 above, where a relatively large stretch of *E. coli* nucleotide sequence is compared with all proteins identified in that genome, the highest-scoring hit (as shown in Figure 3.2.3) has a score of 5450, but a number of other hits have scores of over 1800. In this case, probably most or even all of the 40 hits given by Framesearch are biologically valid. In biological terms, Framesearch has found the genes contained in the genomic region used, plus, possibly, genes that are homologous to those genes.

Keep in mind however that comparing finished genomic sequence data against a database of protein sequences identified in that very genome is a somewhat artificial example. Frequently, in practice, one does not see such a dramatic difference between the top hit and the second hit as was seen in Figure 3.2.5. More commonly, one sees a small number of hits with high-quality scores and percentage identity values, followed by a more hits

with somewhat lower-quality scores, followed by a large number of hits with much lower-quality scores. Figure 3.2.2 shows such an example.

Figure 3.2.2 depicts the score distribution generated by using Framesearch to compare nucleotides 52500 through 55000 of `Nuc_GCG/gi-15829254_55.seq` against all the example peptide sequences. That is, in step 3 of Alternate Protocols 1 and 2 above, instead of specifying a single sequence as described there in response to the prompt `Search for query in what sequence(s)?` one can generate this score distribution by typing `AA_GCG/* .pep` thus doing the search against 5,301 amino acid sequences instead of against only one amino acid sequence. Since the selected region comprises all of one gene and parts of two flanking genes, there are three very strong hits with scores of 978, 1154, and 1391, highlighted by arrows in Figure 3.2.2. There are also many lower-quality hits with scores below 400. From this histogram, it seems likely that hits with scores above 200 might represent genes related to the three genes contained in this region, while hits with scores below 100 probably do not represent biologically significant matches. The validity of hits with scores between 100 and 200 would need to be assessed on a case-by-case basis.

Numbers computed by programs do not replace human judgment. With experience, one develops an intuitive feeling for the overall “shape” of pairwise alignments. It is also important to look at the annotations of the database entries that match the query sequence. Often one gains more insight from reading the descriptions of multiple hits than one can gain from reading the descriptions of the top one or two hits. For instance, if the top few hits are described in words that relate to very similar biological concepts, but the next few hits have very different descriptions, then one should be somewhat more skeptical of the latter alignments. On the other hand, if there are several strong matches whose descriptions do sound plausible and consistent, but among them is one match with a very different description, then one might suspect that description could be incorrect. One may also have an expectation based on the source of the query sequence. Unfortunately, given the variable quality of annotations found in most large genetic sequence databases, interpreting the hits from similarity search algorithms is more of an art than an exact science.

## COMMENTARY

### Background Information

#### *Advantages of Framesearch*

A nucleotide sequence can be compared at the nucleotide level with nucleotide databases, using the BLAST, FASTA, or SSEARCH algorithms (UNITS 3.3 & 3.4). However, because protein sequences are usually of greater biological significance, a translated search against a protein database is often much more informative. If the nucleotide sequence is of high quality, then good results can be obtained by translating the entire nucleotide sequence in all six reading frames and using a standard protein versus protein search algorithm (e.g., UNIT 3.4). But if the nucleotide sequence is of low quality, particularly if it contains many single-nucleotide insertion or deletion errors, then a six-frame-translated search may not work very well. Comparing low-quality nucleotide sequences to protein sequences is the forte of Framesearch.

When six-frame-translated searches are used with nucleotide sequences that contain many indels, every single-nucleotide indel creates a frameshift. If a region of nucleotide sequence that is similar to a protein sequence contains one or more frameshift errors, then what should be a single alignment becomes multiple shorter alignments with any six-frame-translated search algorithm. Figure 3.2.7 shows how indels can prevent the six-frame-translated Smith-Waterman algorithm from finding more than a small part of the region of sequence similarity because none of the six translations has very long regions of good alignment.

Framesearch does not work by generating the six possible translations of each nucleotide sequence. Instead, it translates the nucleotide sequences on the fly, taking into account all possible frameshifts as well as the amino acid insertions, deletions, and mismatches consid-

ered by the Smith-Waterman algorithm. It assumes the protein sequence is accurate, and dynamically inserts or deletes single nucleotides as needed to obtain the best possible alignment between the nucleotide sequence and the protein sequence. This often means that Framesearch can obtain much longer and better alignments between low-quality nucleotide sequences and protein databases than is possible with a six-frame-translated algorithm. Figures 3.2.8 and 3.2.9 show how, given the same input data as used with the Smith-Waterman algorithm to generate Figure 3.2.7, Framesearch can still align nearly the entire length of the protein sequence with the corresponding nucleotide sequence, even though five frameshift errors have been introduced into the nucleotide sequence.

The author of this unit has found Framesearch to be especially valuable when searching for possible homologs of known genes in two types of nucleotide sequence databases: human EST sequences and unfinished microbial genomic sequences. Unfortunately, this power comes at a considerable cost in CPU time, because Framesearch is very slow when run on a general purpose computer.

#### **Motivation behind Framesearch**

The Framesearch algorithm was first presented in a poster (Edelman et al., 1995) describing the joint work of researchers at GCG in Wisconsin and Compugen in Israel. Its development was motivated by the observations that: (1) EST sequences had roughly 10 times as many indels as did the rest of GenBank at that time; (2) about half of all sequence errors in GenBank at that time were indels; and (3) nucleotide versus nucleotide sequence comparisons tended to be much less biologically meaningful than protein versus protein sequence comparisons (GCG, 1995). Furthermore, by 1995 it was clear that in the very near future large amounts of genomic sequence data would soon become available, and “searching a protein database with a translated nucleotide query sequence offers important insights in nucleotide sequencing projects, even in the early stages, when sequence data may be particularly error prone.” However, the authors noted, six-frame translated searches “have difficulty dealing with sequencing errors, particularly insertion and deletion errors in nucleotide sequences. Thus, significant database similarities may go undetected” (GCG, 1995).

#### **The algorithm**

The theory of Framesearch is very similar to that of the classic Smith-Waterman algorithm. The translated Smith-Waterman search generates the six possible translations of the entire nucleotide query sequence, then attempts to align each translation with each peptide sequence in the database. Thus, if there is a frameshift error in the middle of an alignment, a six-frame-translated search algorithm generally cannot find the entire alignment because it considers only one reading frame at a time. With both a Smith-Waterman search and Framesearch, amino acid substitutions are scored using a scoring matrix (UNIT 3.5), and gaps of one or more amino acids in the query or database sequence are scored using affine gap penalty parameters. Framesearch scores amino acid substitutions and whole-codon gaps much as the Smith-Waterman algorithm does, and uses a similar dynamic programming matrix (UNIT 3.1) to search the entire space of possible alignments. However, instead of translating the entire nucleotide query sequence in each of six reading frames, Framesearch inserts or deletes individual nucleotides in order to generate the best possible alignment even when that alignment crosses a reading frame shift. This represents a dramatic expansion of the search space that must be considered, which is why Framesearch is even more CPU-intensive than a Smith-Waterman search.

Like the Smith-Waterman algorithm and heuristic algorithms based on Smith-Waterman such as BLAST (UNITS 3.3 & 3.4) and FASTA, Framesearch assumes that the probability of a given amino acid being found at a given position is independent of its position along the sequence. By contrast, profile-based search algorithms, such as hidden Markov models, profile searches, and PSI-BLAST, take into account position-specific probability distributions.

BLAST (UNITS 3.3 & 3.4) and FASTA are known as “heuristic” algorithms because they use methods that work well in the vast majority of cases to find regions of possible alignment, which are then improved and possibly extended by exact dynamic programming searches (UNIT 3.1) that are restricted to those regions. This makes BLAST (UNITS 3.3 & 3.4) and FASTA much less CPU-intensive than the Smith-Waterman and Framesearch algorithms, at some cost in sensitivity.

All search algorithms that are based on Smith-Waterman, including BLAST (UNITS 3.3

& 3.4), FASTA, and Framesearch, share the limitation that they compute similarity scores based only on the probability that, for instance, a typical cysteine residue would mutate to a phenylalanine residue in the course of evolution. That is, the only source of biological knowledge considered by such algorithms is encoded in the scoring matrix, so any Cys-Phe substitution will be given the same weight in scoring of potential alignments.

However, when groups of related proteins are compared, normally one finds that some residues are much more highly conserved. This generally indicates that these residues are for some reason important to the biological function of these proteins. Scoring matrices of the type used by Smith-Waterman and related algorithms consider only the overall frequency with which a given amino acid is replaced by another, not the fact that, for instance, one cysteine might be in a binding pocket while another cysteine is in a position of less functional importance.

In a profile-based method, a multiple alignment of related sequences is used to build a profile, a statistical summary of the multiple alignment that is then compared against a database in a search for additional sequences that might also be related to the input sequences. Since this profile is based on multiple sequences, it can take into account the fact that certain positions are more highly conserved among the input sequences summarized by that profile, and thus are possibly of greater functional importance than other positions where there is more variation among the input sequences. Some gene-finding programs combine HMM-based methods with Framesearch-like dynamic programming to search nucleotide-sequence databases for potential exons; as one might imagine, such programs are capable of consuming remarkable amounts of CPU time.

#### *Other options for similar analysis*

In this unit, we present as an alternative to a full Framesearch the option of first doing a BLAST search and then using Framesearch to improve and extend the alignments in a region surrounding each BLAST hit (Alternate Protocols 1 and 2). Yet another way to search for protein sequences that are similar to a nucleotide sequence that contains many single-nucleotide indels is to use FASTX or FASTY (Zhang et al., 1997). Note that FASTX and FASTY take nucleotide query sequences and search amino acid databases. To compare an

amino acid query sequence against a nucleotide database, use the TFASTX and TFASTY programs. These recent additions to the FASTA family of heuristic search programs are to classic FASTA what Framesearch is to classic Smith-Waterman. While FASTX and FASTY are a good deal slower than six-frame-translated BLAST (UNIT 3.4) or FASTA, they are considerably faster than Framesearch.

Both the FASTX program and the FASTY program allow for frameshifts in the nucleotide query sequence. However, the FASTX program only considers frameshifts between codons while the FASTY program also considers frameshifts within codons. Thus, FASTY is significantly more CPU-intensive than is FASTX, but produces better alignments with poor quality nucleotide sequences. In some cases, FASTX and FASTY may find hits that BLAST does not detect. Thus, if a BLAST search (UNIT 3.4) does not find any good hits, and dedicated hardware for accelerated Framesearch is not available, one might try FASTX or FASTY to find possible hits and then attempt to improve any alignments found with Framesearch as in Alternate Protocol 1 described above.

Recently (Halperin et al., 1999), Compugen has invented a new algorithm called Frameplus, a further extension of Framesearch, which they claim offers improved sensitivity when the nucleotide sequence data may contain both single-nucleotide indels and longer deletions. Code for a software implementation of this new algorithm is available on the Compugen Web site, and the company's bioinformatics products now include a hardware-accelerated version of Frameplus. The author of this unit has no experience with Frameplus, but users who have access to Frameplus may wish to try it on their data.

TimeLogic bioinformatics computers offer two proprietary algorithms that may be of particular interest to users who work with EST or unfinished genomic sequence data, known as "Semi-global Smith-Waterman" and "Symmetric Frame Independent" (TimeLogic, 2001). Both of these algorithms are designed for comparing a low-quality nucleotide query sequence against a database of low-quality nucleotide sequences.

Semi-global Smith-Waterman is a modification of classic Smith-Waterman, representing a compromise between global (Needleman-Wunsch) and local (Smith-Waterman) pairwise sequence alignment. The semi-global algorithm does a local alignment, but it will prefer alignments that fall near one end of either of

the two nucleotide sequences being aligned. This can help to assemble fragments of low-quality sequence into contigs.

Symmetric Frame Independent search is an extension of Framesearch that compares two nucleotide sequences, dynamically translated into protein, with allowance for frameshifts in either sequence. The reader should be aware that SFI takes almost as long on *accelerated* hardware as Framesearch does on *conventional* hardware.

### **Critical Parameters and Troubleshooting**

This section should be read in conjunction with the documentation of the specific Framesearch implementation being used. While the GCG version of Framesearch offers more optional algorithmic parameters than most others, any version of Framesearch should support the critical parameters reviewed here.

As with any bioinformatics algorithm, the choice of Framesearch options is something of an art. While most implementations supply reasonable defaults thought suitable for general use, these should be considered a starting point to be adjusted as needed. A major benefit from using special-purpose genomic computing hardware, if available, is that when Framesearch takes minutes instead of hours it is practical to experiment, trying different combinations of parameters to see their effect on results.

#### ***Scoring matrix selection***

Probably the single most important parameter for getting good results with the Framesearch algorithm is the choice of scoring matrix (*UNIT 3.5*). A scoring matrix is based on specific assumptions about how amino acids change in the evolutionary process. Typically, such matrices will come in a series, such as the PAM series, the BLOSUM series, or the less-widely-used GONNET series and the other specialized matrices. Each member of such a series of matrices is given a number related to the degree of evolutionary distance between sequences for which that matrix is designed. A search for close relatives of the query sequence should use a matrix designed for finding close relatives, while a search for more-distant relatives of the query sequence should use a matrix designed for sequences with a greater amount of divergence. Note that with some matrices, a larger number indicates greater assumed sequence similarity, while with other matrices a larger number indicates greater assumed evolutionary

distance, so one must read the documentation for the matrix series being used with close attention.

*UNIT 3.5* provides a detailed discussion of how to select an appropriate scoring matrix. In many cases, it is a very good idea to try a search several times, specifying a different scoring matrix each time, and then compare the results.

#### ***Selecting the coding strand***

If the nucleotide query sequence is known to represent the coding strand, the search will take half as long if Framesearch is told to use only the forward strand of the query sequence, and not the reverse complement of the query sequence as well. If the coding direction is not known a priori, translated BLAST searches may provide a good indication.

For the GCG version of Framesearch, the `-ONEstrand` parameter causes Framesearch to assume the given nucleotide sequence represents the coding strand. In the GCG implementation of Framesearch, one cannot specify a similar restriction when using a protein query sequence to search a nucleotide database. Therefore, when using a protein query sequence, if one wishes to search only one strand of the nucleotide database, one must resort to a procedure similar to that of Alternate Protocol 2, above. By contrast, the Timelogic implementation of Framesearch found on their DeCypher systems has a much more flexible provision for specifying which strand to search.

#### ***Setting the frameshift penalty***

The default frameshift penalty for the GCG version of Framesearch is zero. According to the GCG manual, selecting a nonzero frameshift penalty makes the search take significantly longer, but rarely yields significant benefits in exchange for that cost in CPU time. When special-purpose computing hardware is available, it may be appropriate to try a range of values for the frameshift penalty.

#### ***Setting the gap penalty***

As with the classic Smith-Waterman algorithm of which it is an extension, Framesearch includes options for gap creation and extension penalties. To discourage gaps, use values for both gap parameters larger than the program defaults. To find regions of high similarity that are separated by long gaps, use a smaller value for the gap extension penalty. In addition to the classic affine gap penalty parameters, the GCG version of Framesearch offers an optional

maximum gap penalty. If this optional parameter is set, then very long gaps will never be penalized by more than this value. The GCG manual says this option can be used to find matches that are interrupted by introns, but cautions that setting a maximal gap penalty increases the time required to run Framesearch.

#### ***Other parameters***

Most Framesearch implementations provide numerous other options as described in their manuals. The GCG version of Framesearch includes many parameters that control internal details of how the algorithm generates pairwise alignments; for general use these should probably be left at their default values. While the TimeLogic version of Framesearch offers somewhat less scope for fine-tuning the internal behavior of its alignment algorithm than does the GCG version, the TimeLogic version offers far more scope for changing the format of its output than does the GCG version. The Compugen extension of Framesearch, Framplus, has additional gap penalty parameters because it accounts for two different classes of gaps in the nucleotide sequence, single-nucleotide indels and longer deletions.

Before experimenting with the more exotic parameters of Framesearch, it is best to gain experience with the effects of changing the basic options such as matrix choice and gap penalties.

### **Suggestions for Further Analysis**

#### ***Nucleic acid query sequence***

After using Framesearch to find one or more protein sequences with some degree of similarity to the nucleotide query sequence, the logical next step is to perform a multiple sequence alignment of the original query sequence with the top protein hits found by Framesearch, because a multiple alignment is often far more informative than a single pairwise alignment. When a TimeLogic system is used to perform a Framesearch query, one can choose to have the alignments include the translation of the nucleotide sequence as determined by Framesearch, taking into account all the indels identified by comparison with protein sequence. If the user has access to such a system, then this translation plus several of the top protein hits can be run through ClustalW (UNIT 2.3) or GCG PileUp UNIT 3.6 to create a multiple sequence alignment.

#### ***Amino acid query sequence***

After using Framesearch to find one or more nucleotide sequences with some degree of similarity to a protein query sequence, the logical next step is to perform a multiple sequence alignment of the protein query sequence with translations of the top nucleotide hits found by Framesearch, because a multiple alignment is often far more informative than a single pairwise alignment. When a TimeLogic system is used to perform a Framesearch query, one can readily obtain translations of the nucleotide hits as determined by Framesearch, taking into account all the indels identified by comparison with protein sequence. If the user has access to such a system, then these translations, plus the protein query sequence, can be run through ClustalW (UNIT 2.3) or GCG PileUp UNIT 3.6 to create a multiple sequence alignment.

#### ***Either type of query sequence***

Unfortunately, the GCG version of Framesearch does not provide a translation of the nucleotide sequence in a Framesearch alignment. Therefore the GCG Framesearch user will need to use Seqedit to extract the relevant portion of the nucleotide sequence from each alignment, then feed that to GCG Pepdata to generate its six-frame translations, and run these translations plus the protein sequence through a multiple alignment program to get an idea what the best translations may be.

Comparing this multiple alignment with the pairwise alignments generated by Framesearch can often give the user an excellent idea where the nucleotide sequence is likely to contain indels, where the nucleotide sequence is of relatively high quality, and where there may be problems in the nucleotide sequence data. In sum, neither Framesearch nor any other analysis can magically convert low-quality nucleotide sequence data into high-quality data. But, when used with care, and especially when used in conjunction with multiple alignments, the Framesearch algorithm can sometimes extract value from poor sequence data where other methods fail to deliver.

### **Literature Cited**

- Accelrys. 2001. Announcement of new features in SeqWeb version 2 [http://www.accelrys.com/products/seqweb/whats\\_new2p0.html](http://www.accelrys.com/products/seqweb/whats_new2p0.html).
- Edelman, I., Faigler, S., Mintz, E., Natan, A., and Devereux, J. 1995. Framesearch: A rigorous alignment program for searching protein databases with nucleic acid queries. Poster, Genome

Sequence and analysis Conference, Hilton Head, South Carolina, 1995.

NOTE: *The text of this poster can be found at <http://sulu.gcg.com/company/posters/framesearch.html>.*

GCG. 1995. GCG Transcript 3:2. Genetics Computing Group, Madison, Wisconsin.

NOTE: *The GCG Transcript, subtitled "Bio-Computing News for Users of the Wisconsin Package," was published by the company for a number of years. The text of this issue, which features a discussion of the newly-added Framesearch program, can be found at [http://sulu.gcg.com/pub/newsletter/vol3\\_no2\\_nov95.html](http://sulu.gcg.com/pub/newsletter/vol3_no2_nov95.html).*

Halperin, E., Faigler, S., and Gill-More, R. 1999. FramePlus: Aligning DNA to protein sequences. *Bioinformatics* 15(11):867-873.

TimeLogic. 2001. Manuals supplied with a DeCypher bioinformatics accelerator. TimeLogic Corporation, Incline Village, Nevada.

Zhang, Z., Pearson, W.R., and Miller, W. 1997. Aligning a DNA sequence with a protein sequence. *Journal of Computational Biology* 4(3):339-349.

## Key References

Edelman et al., 1995. See above.

*The key reference for the Framesearch algorithm is the poster by Edelman. The key reference for a particular implementation of Framesearch is the documentation supplied with that implementation.*

## Internet Resources

<http://www.accelerys.com/>

*Web site of Accelerys, the corporate parent of GCG.*

<http://www.cgen.com/>

*Web site of the Compugen company.*

<http://www.paracel.com/>

*Web site of the Paracel company.*

<http://www.timelogic.com>

*Web site of the TimeLogic company.*

---

Contributed by Matthew Healy  
Bristol-Myers Squibb Pharmaceutical  
Research Institute  
Wallingford, Connecticut