A Basic Analysis Toolkit for Biological Sequences

Università degli Studi di Palermo, Department of Mathematics, Computer Science Section

BATS
User and Installation
GUIDE

Contents

Chapter 1

BATS User Guide

This chapter illustrates the various functions of the `BATS` GUI, together with examples and data formats.

1.1 File

It is used to set-up various GUI parameters for data visualization (see Fig.1.1). `Columns` specifies the number of characters that must be displayed on a single line, when displaying alignments. `Cut Gap-Only Rows`, when set to yes, causes the lines consisting of gaps only not to be displayed, i.e., the alignment is compressed by eliminating rows of gaps. When set to no, the alignment is displayed normally. The default is no. `Show Results Window`, when set to yes, the results of a computation are displayed in HTML format (but not stored). That is the default setting.

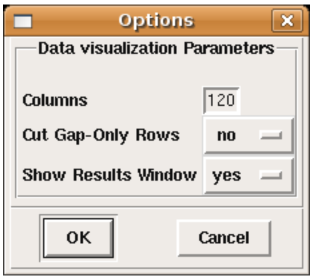


Figure 1.1: The File-Option Window.

1.2 K-Mismatch

Once that the K-Mismatch function has been selected in the top bar of the GUI, the user is prompted to provide various inputs and parameters that, with reference to Fig. 1.2, we present next.

- In the INPUT section, the user can give in input a text and pattern files using Browse buttons (a) and (b), respectively. Since those files can be placed in any directory of the file system, their path names must be given as input. For multiple searches of the same pattern in different text files, one can place the text files in one directory and, with the use of either the shift or control key, one can select a subset of the text files in the directory.
- The maximum number k of mismatches is provided via the OPTIONS section.

- Independently of whether or not the result of the search is displayed at the end of the computation (see section 1.1), one can indicate, via the GUI OUTPUT section, where the result must be stored and in which format, once the computation is finished. Two formats are available: ASCII (button (c)) and HTML (button (d)). For each input text file, the corresponding result file is named according to the following standard: <*text_file_name*>_<*pattern_file_name*>, if the file is in FastA format the file name is substituted from the first word of the line description. We point out that, if the output of a K-Mismatch computation is to be processed by Filter, then the results must be stored in ASCII format.

- Finally, the Compute button (e) is used to start the computation.

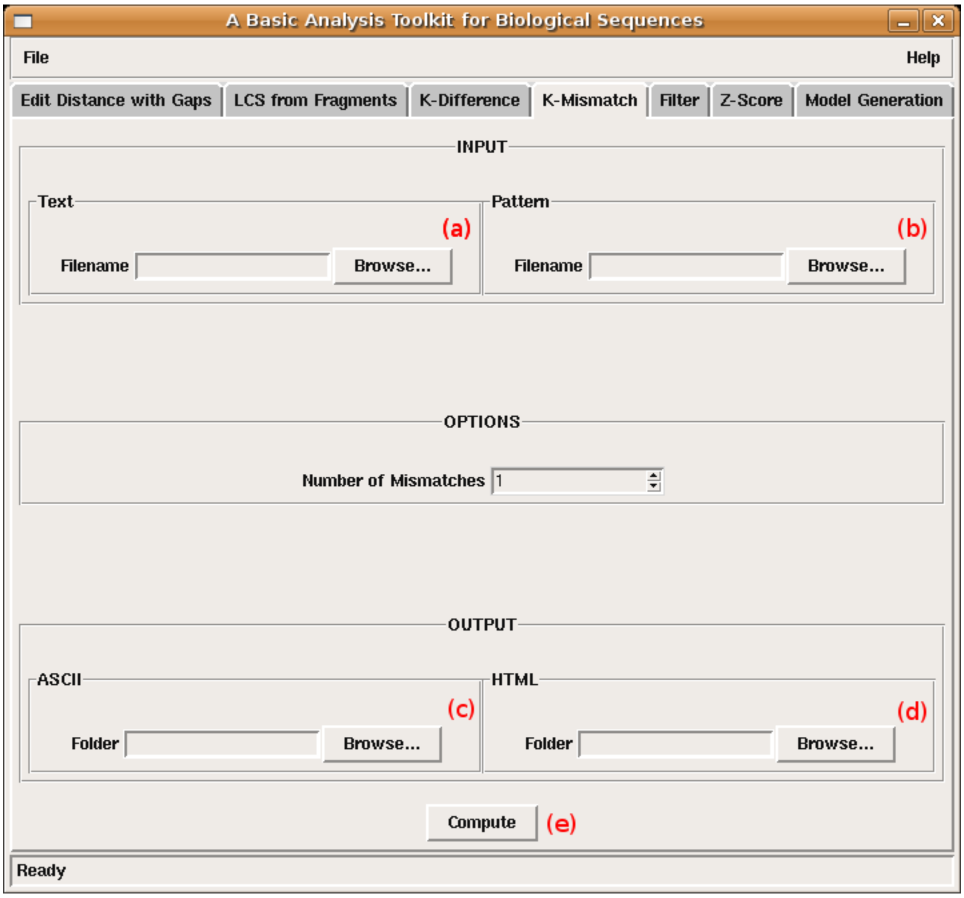


Figure 1.2: The K-Mismatch Tab.

1.2.1 Example

Assume the following: (A) the pattern TCTAGAA and the text are stored in file BATS/example/K-Mismatch/pattern.fasta and BATS/example/K-Mismatch/RAP1.fasta, respectively; (B) one wants to find all occurrences of the pattern in the text, up to $k = 3$ mismatches; (C) the result must be stored in the directory BATS/example/K-Mismatch, for instance in ASCII format. Then, one needs to take the following actions:

- Set the text field (button (a) of the INPUT section) equal to BATS/example/K-Mismatch/RAP1.fasta. Analogously, the pattern field (button (b)) is set to BATS/example/K-Mismatch/pattern.fasta.

- The number of mismatches in the OPTIONS section is set to 3.

- The file where the output is to be stored is specified by setting button (c) of the OUTPUT section to BATS/example/K-Mismatch.
- The computation is then started with button (e).
- The result is then displayed (see Fig. 1.3) and stored.

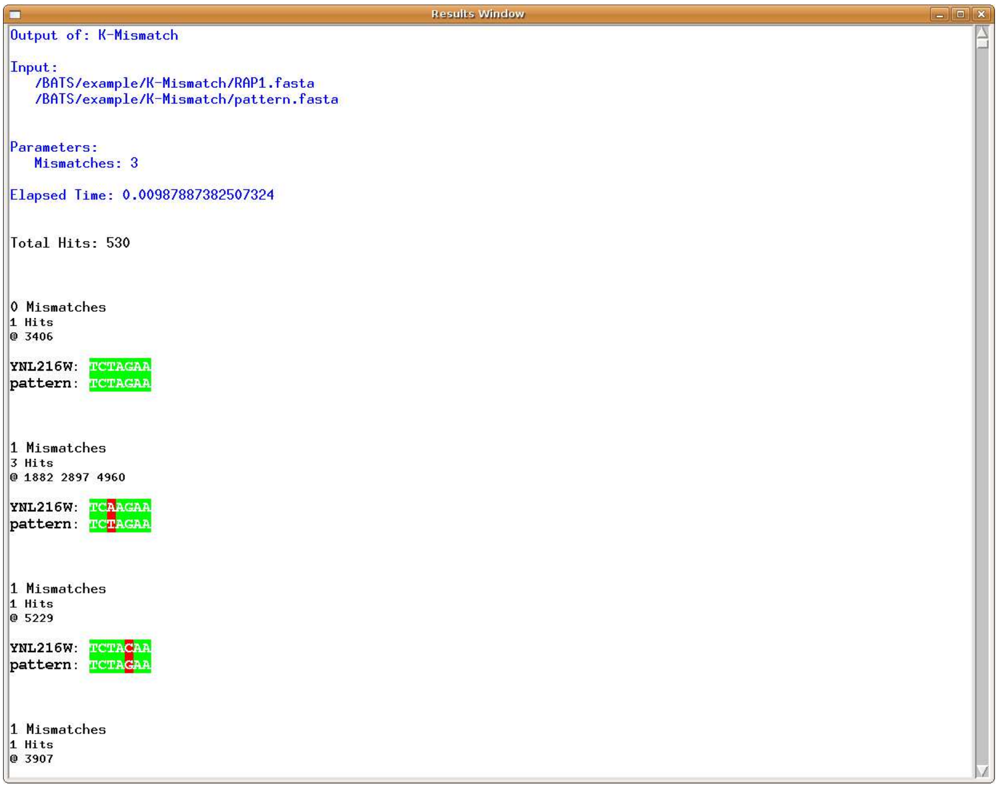


Figure 1.3: An example output of K-Mismatch.

1.2.2 Data Formats

The BATS GUI supports both FastA and Plain text formats, as a default, for the inputs to the K-Mismatch function. Moreover, if BATS is used in conjunction with BioPerl, the GUI supports any other data format known to BioPerl (i.e. FastA, GenBank, EMBL, etc...) for input.

The ASCII output format is illustrated by means of the following example (the description of each field is in italic and not part of the output format):

Output of: K-Mismatch
Input:
/your_path/text.fasta
/your_path/pattern.fasta
Parameters:
Mismatches: 1

The lines beginning with # are comments on input and parameters.

! Elapsed Time: 0.145873069763184

The line beginning with ! contains the elapsed time in seconds for the computation.

& Total Hits: 51

The line beginning with & contains the total number of hits found in the text file. A list of those occurrences then follows, sorted by number of mismatches. We limit ourselves to show one occurrence only.

$ 0 Mismatches
% 1 Hits
@ 13181
text:AAGTTTCC
pattern:AAGTTTCC

The line beginning with $ gives the number of differences/mismatches.
The line beginning with % gives the number of occurrences of a particular instance of the pattern (with the specified number of mismatches/differences) found in the text.
The line beginning with @ contains the sorted list of all of those occurrences. Finally, the aligned sequences are also given.

The HTML output format is well illustrated in Fig. 1.3. However, some parameters in the BATS.css, the default style-sheet, can be set in order to visualize the alignments. Their default setting is reported next (the description of the function of the tag is in italic and not part of BATS.css):

```
.m {
        color: white;
        font-weight: bold;
        background-color: green;
        }
```

m is the tag for matching characters.

```
.s {
        color: white;
        font-weight: bold;
        background-color: red;
        }
```

s is the tag for aligned mismatching characters.

```
.g {
        color: white;
        font-weight: bold;
        background-color: black;
        }
```

g is the tag for an insertion/deletion, i.e., a gap.

1.3 K-Difference

The parameter setting and input/output features, including data formats, of the K-Difference function are analogous to the ones discussed for the K-Mismatch function. Its tab is displayed in Fig. 1.4. Moreover, Fig. 1.5 displays the output produced by K-Difference on the same example of section 1.2.1

1.4 LCS from Fragments

With reference to Fig. 1.6, the input/output options of the FLCS function are as in the K-Mismatch function, given in section 1.2. However, the OPTIONS section is different. Indeed, one needs to indicate how to provide fragments to the algorithm, by choosing one of the following:

- The minimum length of a fragment. In that case, all matching substrings between the two input strings, of at least the specified length, are computed and used for the construction of the LCS.
- A file containing a given list of fragments, i.e., a user-selected subset of matching substrings of the two input strings. The LCS is computed using only the given set.

1.4.1 Example

Assume the following: (A) the two input files are in BATS/example/FLCS/P12028.fasta and BATS/example/FLCS/P24331.fas respectively; (B) one wants to compute the LCS of those two strings. Then, one must take the following action:

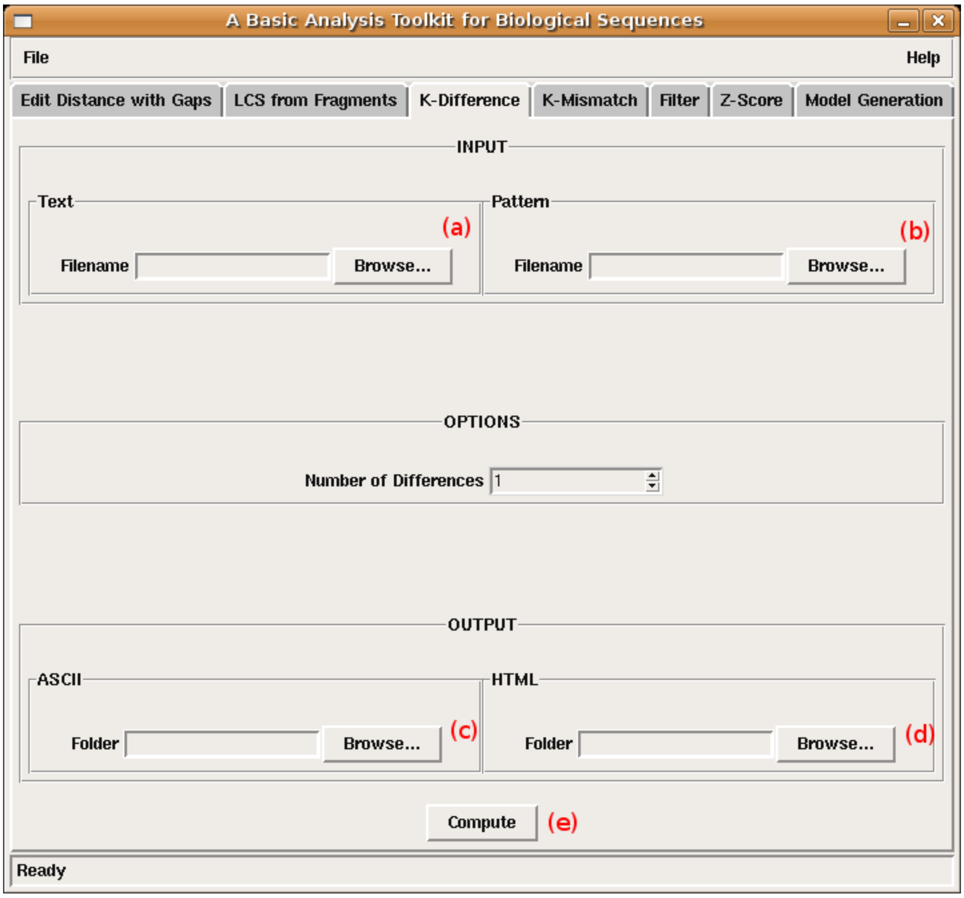


Figure 1.4: The K-Difference Tab.

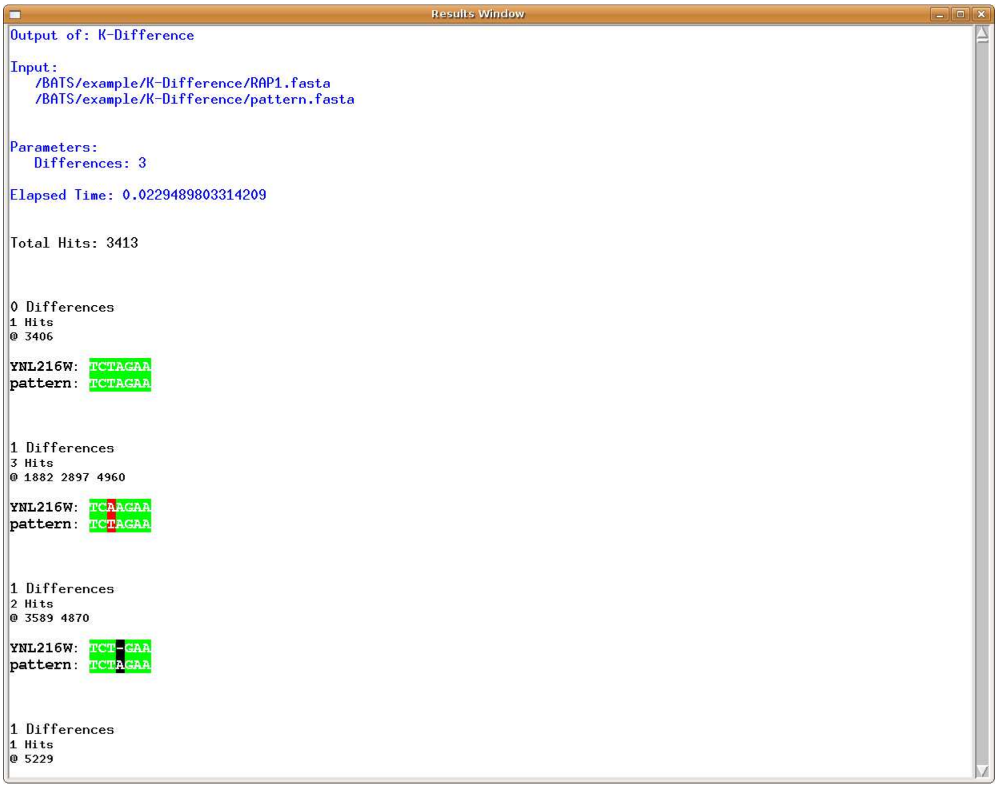


Figure 1.5: An example output of K-Difference.

- Buttons (a) and (b) are used to input the file names.
- The minimum length of a fragment is set to 1 in the corresponding field of the OPTIONS section.
- The computation is then started with button (e).
- The result is then displayed (see Fig. 1.7), but not saved in a file.

1.4.2 Data Formats

The formats for the files of the INPUT section of the GUI are identical to the ones for the K-Mismatch function. In case one specifies a fragments file in the OPTIONS section of the GUI, each row of the file represents a fragment and must satisfy the following format:

i<space>j<space>l<return>

where i is the substring starting point in the first input sequence, j is the substring starting point in the second input sequence and l is the length of the matching substrings. As for the output format, we need to describe only the ASCII format, which is best presented by means of an example (the description of each field is in italic and not part of the output format):

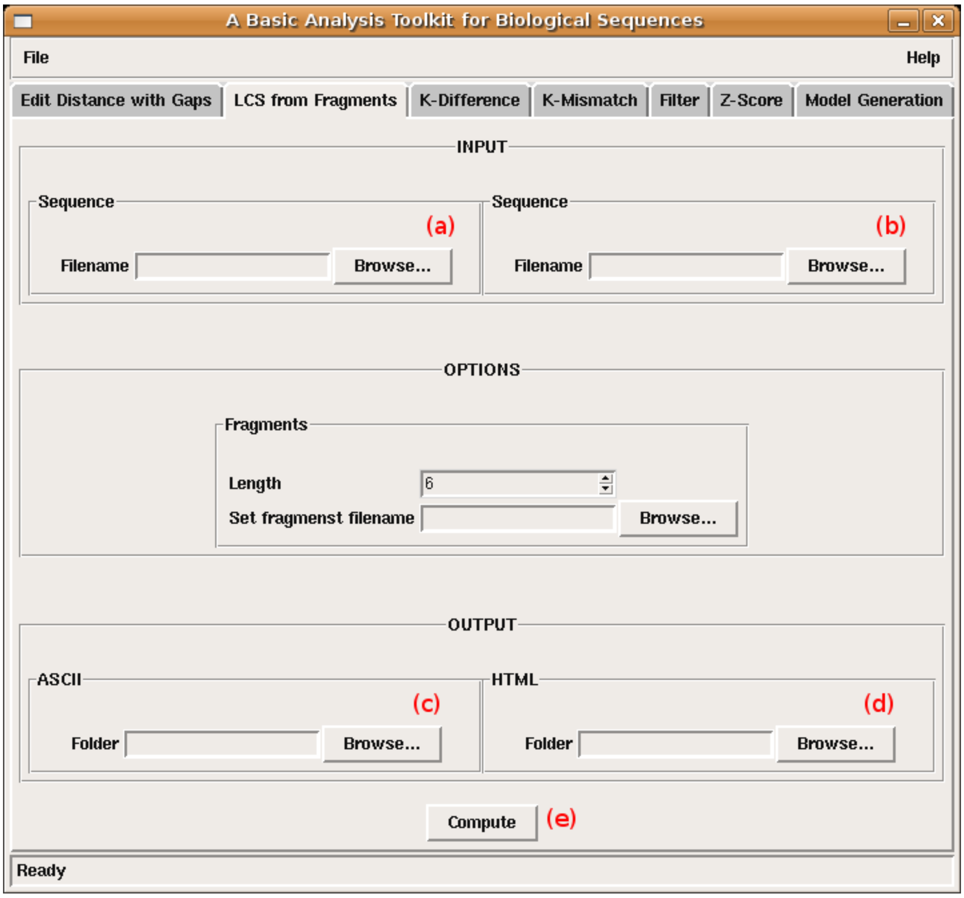


Figure 1.6: The FLCS Tab.

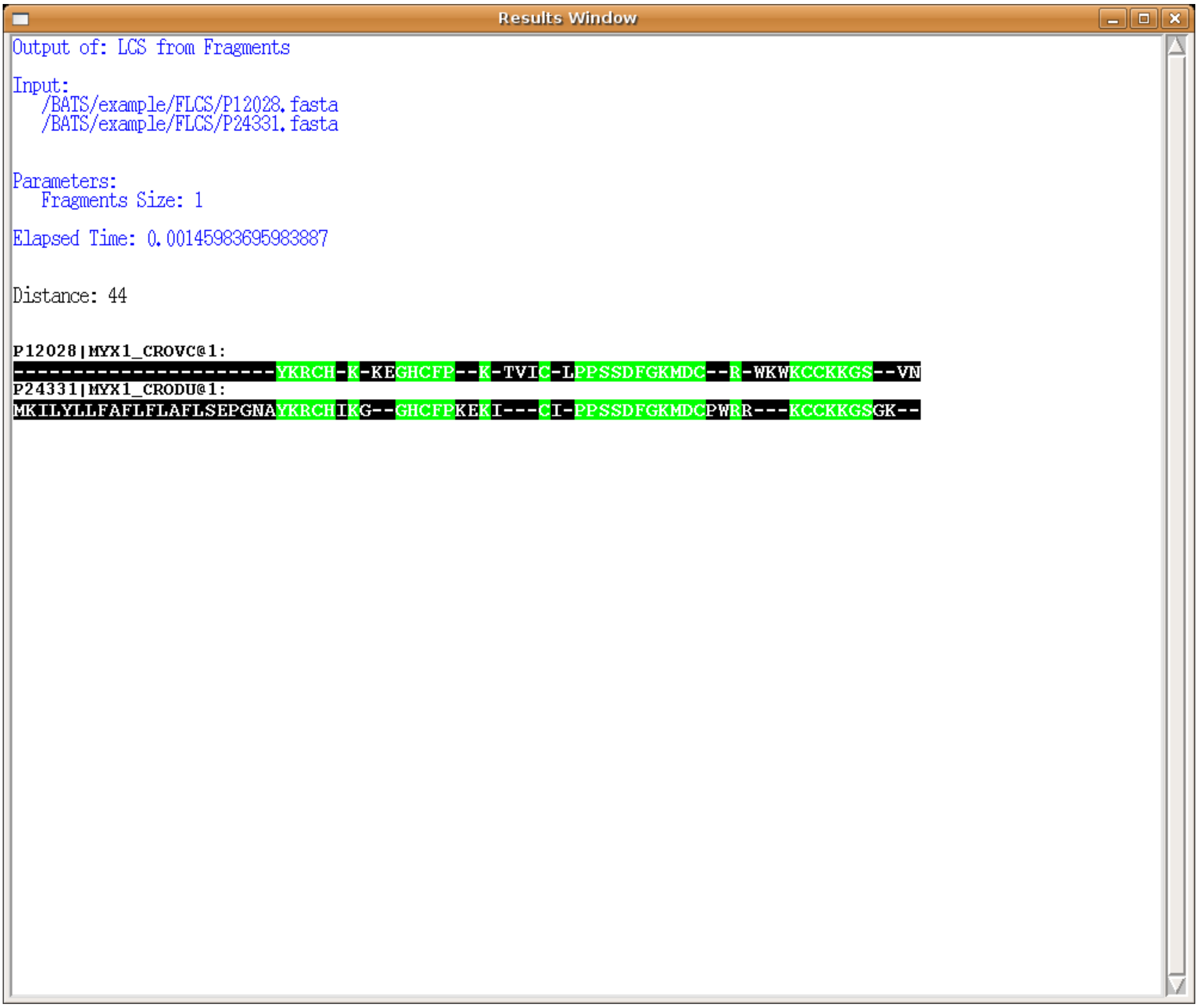


Figure 1.7: An example output of FLCS.

Output of: LCS from Fragments
Input:
/your_path/sequence1.fasta
/your_path/sequence2.fasta
Parameters:
Fragments Size: 1
The lines beginning with # are comments on input and parameters.

! Elapsed Time: 0.00145983695983887

The line beginning with ! contains the elapsed time in seconds for the computation.

$ Distance: 44

The line beginning with $ contains the value of the Edit Distance based on the FLCS. The aligned sequences are reported on two different lines, as follows:

sequence1@1:$YKRCH-K-KEGHCF----$
sequence2@1:$YKRCHIKG--GHCFMKIL$

1.5 Edit Distance with Gaps

With reference to Fig. 1.8, the input/output options of the Edit Distance with Gaps function are as in the K-Mismatch function, given in section 1.2. However, the OPTIONS section is different. The user can specify a cost function for gaps in each list of sequences (i.e. the cost function box on the right corresponds to the list on the right and the cost function box on the left corresponds to the list on the left). Indeed, one needs to indicate how to provide the cost function to the algorithm, by choosing one of the following:

- in case the default cost functions are used, one needs to specify its parameters W_a (Gap Opening Cost), W_g (Gap Extension Cost) and $base$ (log base, for convex cost functions), where the default functions are:

  The convex function:
  $$W_a + W_g log_{base}(k - l)$$
  The affine function:
  $$W_a + W_g(k - l)$$

- write in the edit box a custom convex cost function of two vaiables, denoted $\$l$ and $\$k$. The function can be written according to the rules of the eval module in Perl, e.g.,
  $$0.2 + 4 * log(log(\$k - \$l) + 3)$$

1.5.1 Example

Assume the following: (A) the two input files are in BATS/example/Edit_Distance_with_Gap/Q9VVC9.fasta and BATS/example/Edit_Distance_with_Gap/Q29EX5.fasta, respectively; (B) one wants to compute the Edit Distance with Gaps of those two strings with a default convex function with $W_a = 2$, $W_g = 1$ and $base = 2$. Moreover, the default symbol substitution matrix must be used, i.e., zero cost for a match and cost one for any mismatch. Then, one must take the following action:

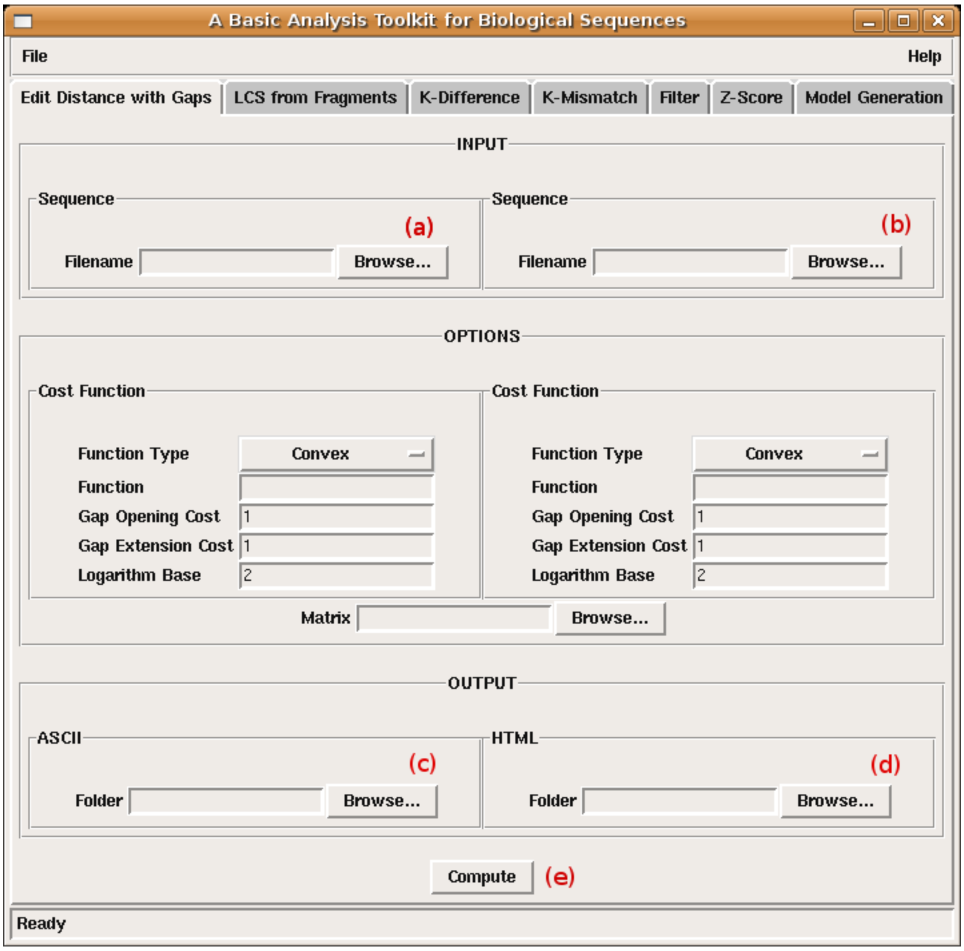


Figure 1.8: Edit Distance with Gaps Tab.

- Buttons (a) and (b) are used to input the file names.
- Set to 2 the W_a , 1 to W_g and 2 to $base$ in the corresponding field of the OPTIONS section.
- The matrix field must be left empty.
- The computation is then started with button (e).
- The result is then displayed (see Fig. 1.9), but not saved in a file.

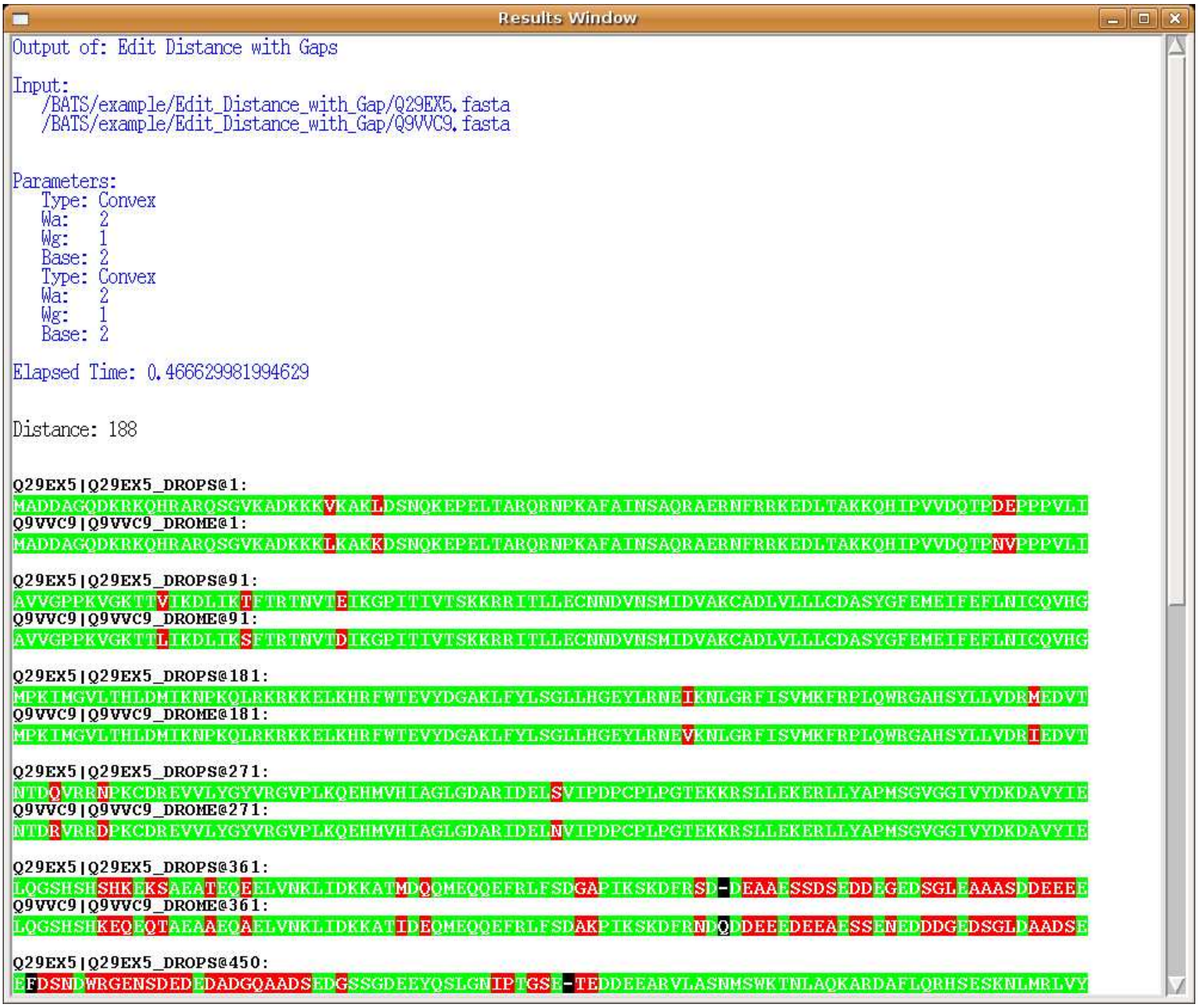


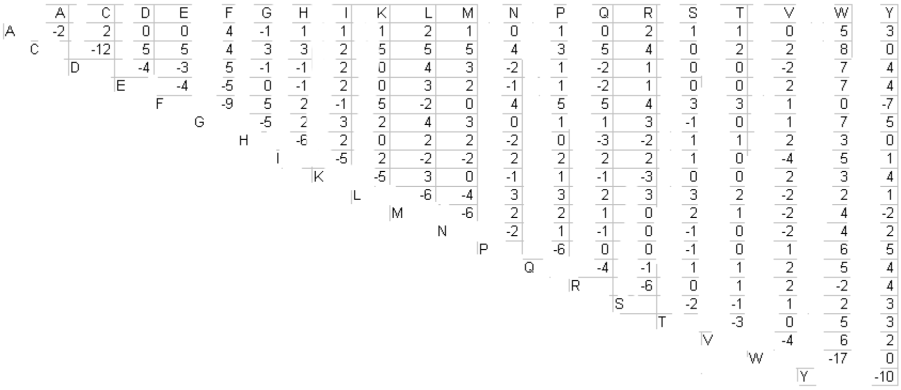
Figure 1.9: Edit Distance with Gaps Tab.

1.5.2 Data Formats

The input sequences specified in button (a) of the GUI can be in any of the input formats described for the K-Mismatch function. Assuming that we have an alphabet $\Sigma = \{\delta_1, ..., \delta_n\}$, the file format for the file in the OPTIONS section of the GUI (the character substitution matrix file) is as follows:

δ_1<space>δ_2<space>... <space>δ_{n-1}<space> δ_n<return>
$a_{1,1}$<space>$a_{1,2}$<space>... <space>$a_{1,n-1}$<space> $a_{1,n}$<return>
<space><space>$a_{2,2}$<space>... <space>$a_{2,n-1}$<space> $a_{2,n}$<return>
...
<space><space>...<space> <space> $a_{n,n}$

Notice that the alphabet is given first and from that we can recover the correspondence between the numeric values of the remaining part and the alphabet characters, as illustrated in the example of Fig. 1.10 (the characters on the diagonal are shown for illustration purposes only).



	A	C	D	E	F	G	H	I	K	L	M	N	P	Q	R	S	T	V	W	Y
A	-2	2	0	0	4	-1	1	1	1	2	1	0	1	0	2	1	1	0	5	3
C		-12	5	5	4	3	3	2	5	5	5	4	3	5	4	0	2	2	8	0
D			-4	-3	5	-1	-1	2	0	4	3	-2	1	-2	1	0	0	-2	7	4
E				-4	-5	0	-1	2	0	3	2	-1	1	-2	1	0	0	2	7	4
F					-9	5	2	-1	5	-2	0	4	5	5	4	3	3	1	0	-7
G						-5	2	3	2	4	3	0	1	1	3	-1	0	1	7	5
H							-6	2	0	2	2	-2	0	-3	-2	1	1	2	3	0
I								-5	2	-2	-2	2	2	2	2	1	0	-4	5	1
K									-5	3	0	-1	1	-1	-3	0	0	2	3	4
L										-6	-4	3	3	2	3	3	2	-2	2	1
M											-6	2	2	1	0	2	1	-2	4	-2
N												-2	1	-1	0	-1	0	-2	4	2
P													-6	0	0	-1	0	1	6	5
Q														-4	-1	1	1	2	5	4
R															-6	0	1	2	-2	4
S																-2	-1	1	2	3
T																	-3	0	5	3
V																		-4	6	2
W																			-17	0
Y																				-10

Figure 1.10: example matrix input file

The ASCII format file for the output is identical to the one used for the LCS from Fragments (see section 1.4.2).

1.6 Filter

With reference to Fig. 1.11, the Filter function takes in input a directory of files, by means of the Browse button (a) of the INPUT section. Each string in one of those result files corresponds to an occurrence of a pattern in a text. The number and location of its occurrences are also specified. In order to get a number of strings occurring in a user-specified number of texts, one must use the OPTIONS fields, as follows:

- Hits: for each string in a file, the minimum number of its occurrences in the corresponding text, in order to be considered for the next step.
- Files: minimum percentage of input files that must have a particular string in common, among the ones that have successfully passed the previous selection.

The procedure to store the output is identical to the one of the other functions already discussed. The computation is started via the Compute button (d).

1.6.1 Example

Assume the following: (A) we have two files, both in BATS/example/Filter/, where one file is the output of example 1.2.1, while the other is obtained as output of a computation with the same parameter settings, but with a different text file; (B) one is interested in knowing which occurrences in the two files appear in both text sequences at least 1 time; (C) the results need to be stored in ASCII in the directory BATS/example/Output_Filter/. Then, one needs to take the following actions:

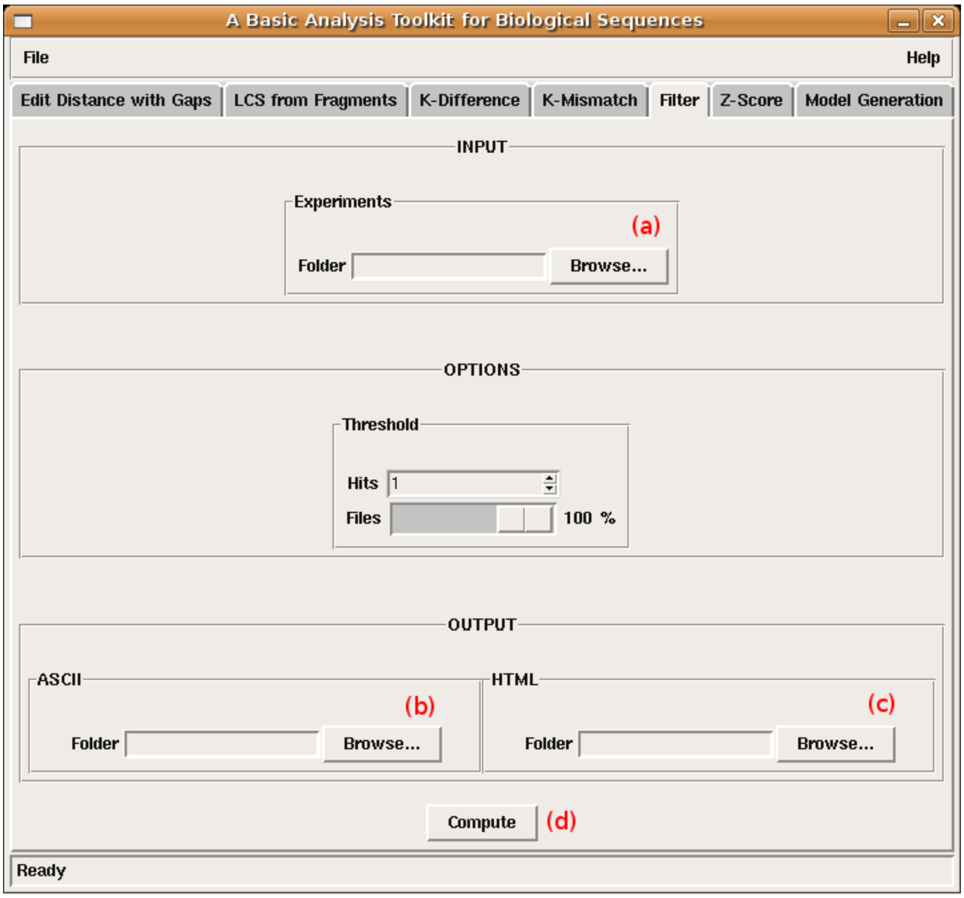


Figure 1.11: The Filter Tab.

- Give in input the directory BATS/example/Filter/, via button (a).
- Set the Hits Threshold to 1 and the Files Threshold to 100% in the OPTIONS section.
- Set the output path to BATS/example/Output_Filter/ via button (b).
- Start the computation through button (d).
- The result is then displayed (see Fig. 1.12) and stored.

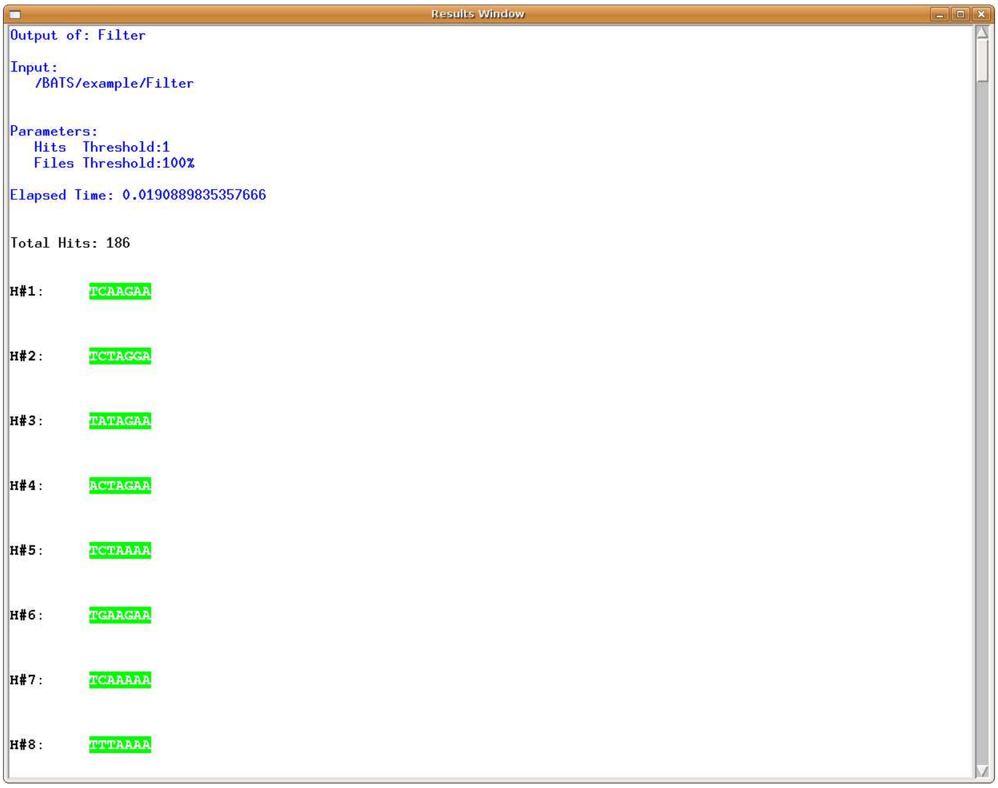


Figure 1.12: An example output of Filter.

1.6.2 Data Formats

The format of each input file must be the ASCII format adopted for the output of K-Mismatch and K-Difference. As for the output format, we need to describe only the ASCII format, which is best presented by means of an example (the description of each field is in italic and not part of the output format):

Output of: Filter
Input: # /BATS/example/
Parameters: # Hits Threshold:1 # Files Threshold:100%

The lines beginning with # are comments on input and parameters.

! Elapsed Time: 0.330241918563843

The line beginning with ! contains the elapsed time in seconds for the computation.

& Total Hits: 257

The line beginning with & contains the total number of hits found. A sorted list of all strings passing the filter test follows, each reported on a line starting with H#.

H#1: TCTAGAA

1.7 Z-Score

With reference to Fig. 1.13, the Z-Score function takes in input a directory containing text files (button (a) in the GUI INPUT section) and a file containing a set of strings (button (b) in the GUI INPUT section), referred to as set of patterns. The rules to properly set those fields are as in the K-Mismatch function, given in section 1.2. In order to compute the statistical significance of the occurrence of strings in the set of patterns in the text strings, one needs a background probabilistic model. That is specified by giving the path name of the relevant file in the OPTIONS section of the GUI (see also section 1.8). Finally, the computation is started using button (e) and eventually the numeric value of the Z-Score is returned. We point out that the Z-Score function works only for DNA strings, i.e., strings on the well known four letter alphabet $\{A, C, G, T\}$.

1.7.1 Example

Assume the following: (A) the output of example 1.6.1 be stored in BATS/example/Z-Score/filter.txt; (B) one wants to know how significant are the occurrences of strings in Filter_Filter.txt in two (text) strings stored in BATS/example/Z-Score/sequence; (C) all the mentioned files are part of Human DNA. Then, one needs to take the following actions:

- Give in input the directory containing the text sequences by setting button (a) to BATS/example/Zscore/sequence.
- Give in input the pattern file by setting button (b) to BATS/example/Zscore/Filter_Filter.txt.
- Give the proper organism file via the OPTION section. In this case, BATS/example/Zscore/Human_table.3
- Start the computation via button (e).
- The result is then displayed (see Fig. 1.14), but not saved in a file.

1.7.2 Data Formats

The input sequences specified in button (a) of the GUI can be in any of the input formats described for the K-Mismatch function. The file specifying the set of strings (button (b) of the GUI) must have the following format:

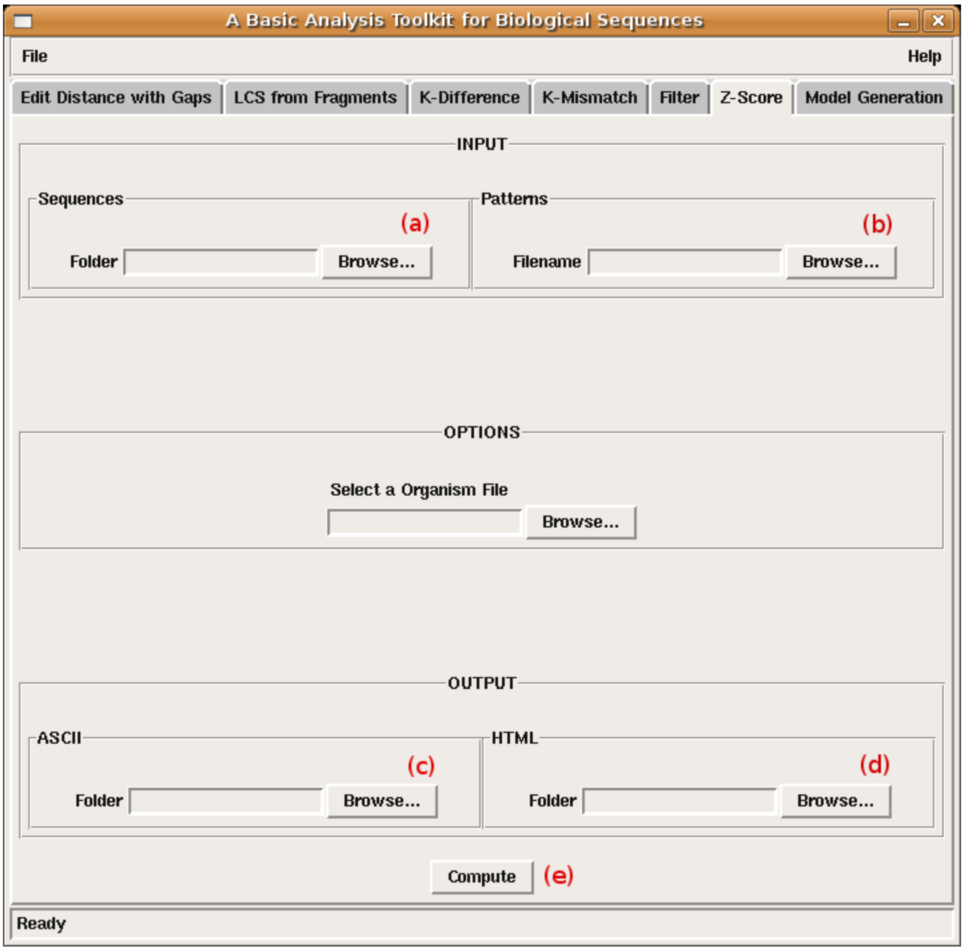


Figure 1.13: The Z-Score Tab.

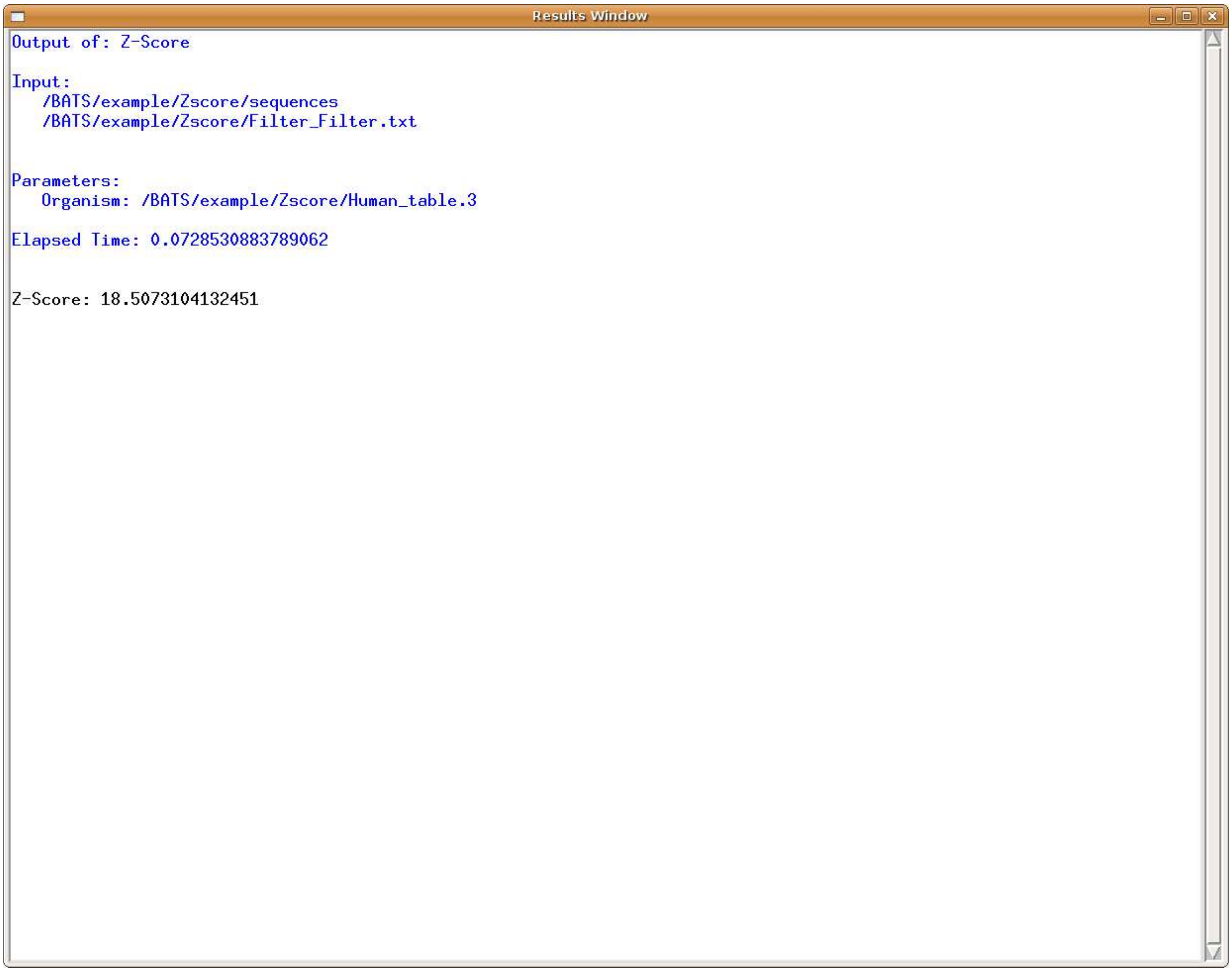


Figure 1.14: An example output of Z-Score.

$pattern_1$<return>
$pattern_2$<return>
...
$pattern_n$<return>

where each row gives a string in the set. Moreover, one can give in input a ASCII file produced as output by either K-Mismatch or Filter, with no format modifications.

The file format for the file in the OPTIONS section of the GUI (the organism file) is presented in section 1.8.1.

1.8 Model Generation

With reference to Fig.1.15, a set of files contained in a directory is given by specifying the proper path name in Browse button (a). Those files are used to compute empirical probability distributions for the order three Markov Chain representing the model, i.e., the background stochastic process that generates strings characterizing a given organism. The OPTIONS section of the GUI is used to specify the organism name. The directory where the new model must be placed is specified via the button (b) of the GUI OUTPUT section. Then, once the computation is done, a file <name of organism>_table.3 is created in that directory. Finally, the computation is started via button (c).

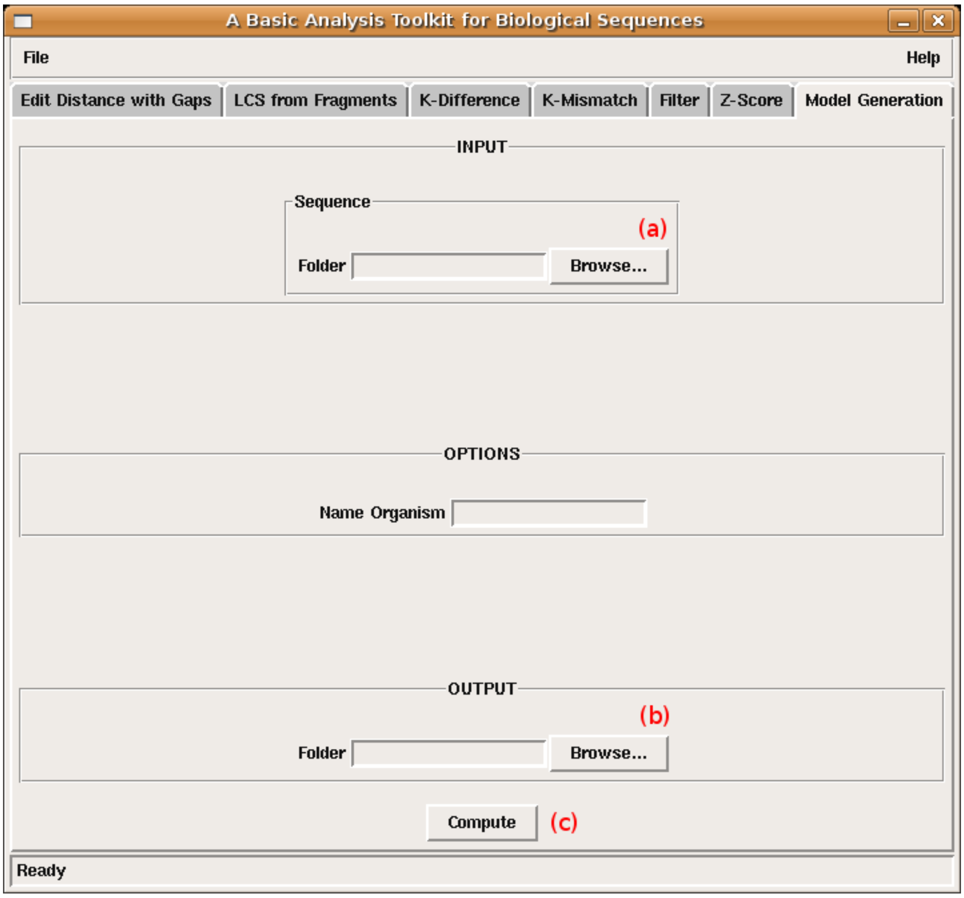


Figure 1.15: The Model Generation Tab.

1.8.1 Data Formats

The files format for the INPUT section of the GUI is as the ones discussed for the K-Mismatch function. As for the output file format, it has the following order and form:

- A 64 by 64 probability transition matrix, stored in row major order:

$a_{1,1}$<*space*>$a_{1,2}$<*space*>*...*<*space*>$a_{1,64}$<*return*>
$a_{2,1}$<*space*>$a_{2,2}$<*space*>*...*<*space*>$a_{2,64}$<*return*>
...
$a_{64,1}$<*space*>$a_{64,2}$<*space*>*...*<*space*>$a_{64,64}$<*return*>

- A vector of 64 elements, giving the initial state probability distribution:

b_1<*space*>b_2<*space*>*...*<*space*>b_{64}<*return*>

- Two additional 64 by 64 probability transition matrices, stored each in row major order:

$c_{1,1}$<*space*>$c_{1,2}$<*space*>*...*<*space*>$c_{1,64}$<*return*>
$c_{2,1}$<*space*>$c_{2,2}$<*space*>*...*<*space*>$c_{2,64}$<*return*>
...
$c_{64,1}$<*space*>$c_{64,2}$<*space*>*...*<*space*>$c_{64,64}$<*return*>
$d_{1,1}$<*space*>$d_{1,2}$<*space*>*...*<*space*>$d_{1,64}$<*return*>
$d_{2,1}$<*space*>$d_{2,2}$<*space*>*...*<*space*>$d_{2,64}$<*return*>
...
$d_{64,1}$<*space*>$d_{64,2}$<*space*>*...*<*space*>$d_{64,64}$<*return*>

Chapter 2

GUI Installation

In order to use the BATS GUI, one needs a Perl interpreter and some Perl modules. ActivePerl (from ActiveState) is suggested, since it makes the installation of BATS easy.

2.1 Installation of Precompiled Modules, via ActivePerl

To install the precompiled packages, i.e., PPDs, ActivePerl (from ActiveState) is required.

1. Download from the BATS home page (in particular http://math.unipa.it/~raffaele/BATS/BATS_precompiled.html) and Unpack BATS Perl modules (PPDs) with any zip pack/unpack program for Windows or *tar zxvf* BATS.*tar.gz* for other Operating Systems. Then, the unpacked folder must be accessed in command line mode.

2. Install the desired BATS Perl modules, with command

   ppm install BATS-*<module>.ppd*

   from the command line prompt–one needs to have administrator privileges. The available modules are:

   - Edit Distance with Gaps
   - LCS from Fragments
   - K-Difference
   - K-Mismatch
   - Z-Score
   - Model Generation

3. Download the BATS GUI from the same page as above and install it as follows:

   - for Windows: execute *BATS_setup.exe*
   - for Linux/Unix or Mac Os X: unpack with tar zxvf BATS*_setup.tar.gz*, and run the GUI, from the command line, as follow:
     - cd BATS-GUI
     - perl BATS.pl

4. ***Optional***: installation of BioPerl Core package grants parsing of most common sequence formats; otherwise one can only use FastA and Plain text formats.

2.2 Installation with Compilation of Modules

This part regards users that would like to compile the modules or that do not have ActivePerl available.

To install the BATS GUI, one needs to have C, Perl and Perl/Tk compilers.

One has two options, depending on whether the eclipse editor is available or not. In the former case, instructions are given in the next chapter and the reader may skip the remaining part of this chapter. The instructions below will work only for Linux/Unix and Mac Os X. They can be modified also to work for Windows, but the installation becomes far from trivial.

1. Download sources from BATS home page (in particular `http://www.math.unipa.it/~raffaele/BATS/BATS_compiled.html`)
2. Unpack sources with any zip pack/unpack program for Windows or *tar zxvf* BATS*.tar.gz* for other Operating Systems
3. (a) If one wants to compile all of the modules functions then cd BATS and one procedes step 4.
   (b) Compile each desired BATS Perl module, by cd BATS/<*module*>
      - Edit Distance with Gaps -> *EditDistance*
      - LCS from Fragments -> *FLCS*
      - K-Difference -> *K_Difference*
      - K-Mismatch -> *K_Mismatch*
      - Z-Score -> *ZScore*
      - Model Generation -> *Module_Generation*
4. *make module*
5. *make module_install*–one needs to have administrator privileges to properly execute the command.
6. Download the BATS GUI from the same page as above and install it: unpack with tar zxvf BATS*_setup.tar.gz*
7. ***Optional***: installation of BioPerl Core package grants parsing of most common sequence formats; otherwise one can only use FastA and Plain text format.
8. Run the GUI, from the command line, as follow:
   - cd BATS-GUI
   - perl BATS.pl

Note: When executing *make module* and *make module_install*, the current directory absolute path name must contain no spaces in it. For instance, the path

C:\Documents and Settings>

would cause a compilation problem, while

C:\Documents_and_Settings>

is acceptable.

Chapter 3

BATS GUI with eclipse

To use the BATS GUI with eclipse, one needs to have C, Perl and Perl/Tk compilers (we suggest ActivePerl from ActiveState) as well as C/C++ and Perl eclipse plug-ins. Moreover, if one has a Windows O.S., *nmake* is required.

When eclipse is present, one can compile and use the entire BATS system, via the GUI, in a few very simple steps, as follows:

1. Download BATS and BATS-GUI eclipse projects from BATS home page (in particular http://www.math.unipa.it/~raffaele/BATS/BATS_GUI_withEclipe.html);
2. Unpack them with any zip pack/unpack program or *tar zxvf* BATS.*tar.gz* on other O.S.;
3. Import BATS-GUI and BATS project in some workspace (setting the copy project flag as in Fig 3.1);
4. Build BATS project ;
5. Run BATS.pl in BATS-GUI project.

Note One: When importing the BATS and BATS-GUI eclipse project, the workspace directory absolute path name must contain no spaces in it. For instance, the path

C:\Documents and Settings>

would cause a compilation problem, while

C:\Documents_and_Settings>

is acceptable.

Note Two: If one uses Windows and has Microsoft Visual C++ installed, one needs to take the following actions in all module directories, except for Filter:

1. rename Makefile in Makefile.other
2. rename Makefile.Win32 in Makefile

Moreover, set *nmake* in the Build Command of the C/C++ Make Project via BATS project properties.

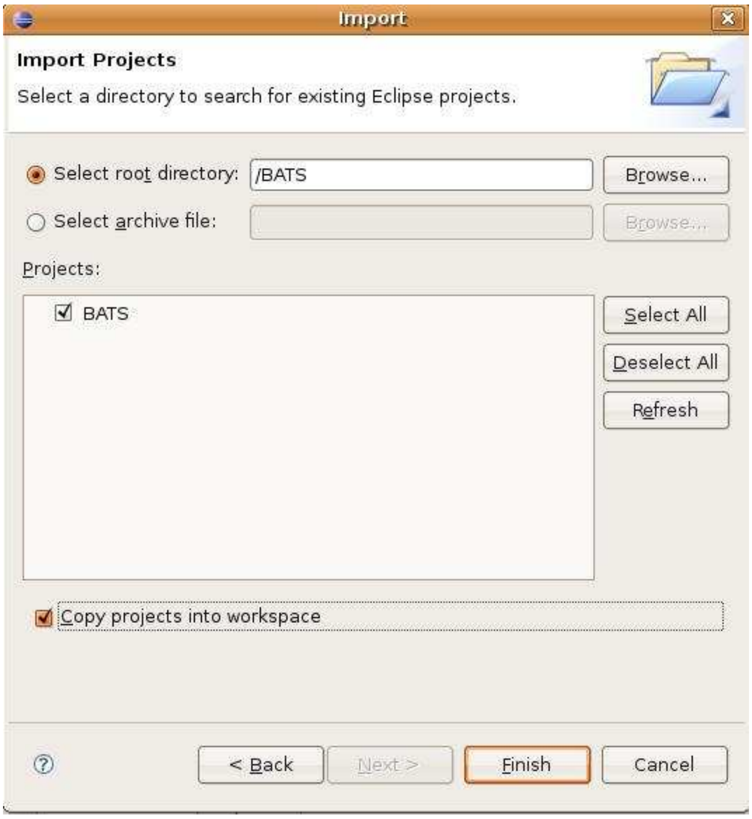


Figure 3.1: Import `BATS` eclipse project

Chapter 4

BATS without GUI

The BATS library can be included in/or linked to other C/C++ programs, in the usual standard way. It complies with ANSI C. Moreover, the Perl Modules rely on the BATS C library and offer an interface to them. If one uses Windows, *nmake* is required.

4.1 The C/C++ library

1. Download sources from BATS home page (in particular `http://www.math.unipa.it/~raffaele/BATS/BATS_Clibrary.html`);

2. Unpack sources with any zip pack/unpack program for Windows or *tar zxvf* BATS*.tar.gz* on other O.S.

3. (a) If one wants to compile all of the library functions then cd BATS and one procedes step 4.

   (b) Compile each desired BATS C library, by cd BATS/<*library*>

   - Edit Distance with Gaps -> *Edit_Distance_Gaps*
   - LCS from Fragments -> *FLCS*
   - K-Difference -> *K_Difference*
   - K-Mismatch -> *K_Mismatch*
   - Z-Score -> *ZScore*
   - Model Generation -> *Model_Generation*

4. *make library*

5. *make library_install*–***Strongly Recommended***–one needs to have administrator privileges to properly execute the command. Command not supported by Windows.

If one has not created the library within the system (see step 6 above) or one has a Windows operating system, then it is required to link to a BATS C library, as follows:

-l<library>

where library is one of:

- Edit Distance with Gaps -> *edit_distance_with_gap*
- LCS from Fragments -> *flcs*
- K-Difference ->*k_difference*
- K-Mismatch -> *k_mismatch*

- Z-Score -> *zscore*
- Model Generation -> *model_generation*

One needs also to make sure that `BATS` libraries are into ld search path.

Note: If one uses Windows and has Microsoft Visual C++ installed, one needs to take the following actions in all module directories, except for Filter:

1. rename Makefile in Makefile.other
2. rename Makefile.Win32 in Makefile

4.2 The Perl library

This part follows exactly the same procedures of the previous two chapters, where only the instructions part regarding the GUI must be omitted. Relevant software is at `http://www.math.unipa.it/~raffaele/BATS/BATS_PerlModule.html`.

Appendix A

License and Copyright for BATS

Appendix B

GNU Licence

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software—to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE

Terms and Conditions For Copying, Distribution and Modification

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The “Program”, below, refers to any such program or work, and a “work based on the Program” means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term “modification”.) Each licensee is addressed as “you”.

   Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program’s source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

   You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

   (a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

   (b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

   (c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

   These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

   Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

   In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

   (a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
   (b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
   (c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

   The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

   If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

   Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

No Warranty

11. Because the program is licensed free of charge, there is no warranty for the program, to the extent permitted by applicable law. Except when otherwise stated in writing the copyright holders and/or other parties provide the program "as is" without warranty of any kind, either expressed or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The entire risk as to the quality and performance of the program is with you. Should the program prove defective, you assume the cost of all necessary servicing, repair or correction.

12. In no event unless required by applicable law or agreed to in writing will any copyright holder, or any other party who may modify and/or redistribute the program as permitted above, be liable to you for damages, including any general, special, incidental or consequential damages arising out of the use or inability to use the program (including but not limited to loss of data or data being rendered inaccurate or losses sustained by you or third parties or a failure of the program to operate with any other programs), even if such holder or other party has been advised of the possibility of such damages.

END OF TERMS AND CONDITIONS