

RAPTOR USER MANUAL

This user manual is also available online at
<http://www.bioinformaticssolutions.com/products/raptor/doc/raptordoc.php>.

INTRODUCTION

RAPTOR (Rapid Protein Threading Predictor) is developed by Dr. Jinbo Xu and Professor Ming Li at the University of Waterloo. It applies novel Linear Programming techniques to the Protein Threading problem and has achieved great success. In December 2002, RAPTOR was ranked number one in fold recognition among non-meta programs in CAFASP3 (Critical Assessment of Fully Automated Structure Prediction), the renowned international competition held by the protein research community. In December 2004, it was ranked No. 2.

See `raptor_workflow.pdf` for a general overview of the workflow.

Please use the following reference if you use RAPTOR in your research:

Jinbo Xu, Ming Li, Dongsup Kim and Ying Xu,
RAPTOR: Optimal Protein Threading by Linear Programming the inaugural issue, JBCB, 2004

INSTALLATION

Please see the `README.txt` for a complete installation guide.

USAGE

You can start a command window and type the following (assuming you have added `$RAPTOR_HOME/bin` to your `PATH` environmental variable):

```
RAPTOR_server -s [seq_file_name]
```

where `[seq_file_name]` is a plain-text FASTA file containing your input sequence. It can be in the working directory, or you can type the full path (e.g. `C:\RAPTOR\data\seq_test\1234.seq`).

The program starts running with RAPTOR's default settings.

`RAPTOR_server` manages the workflow from running PSI-BLAST to ranking template hits. This is the default interface most users will interact with.

Options:

1. If you wish to change the default settings for your needs, refer to “Configurations” below for instructions.
2. If you wish to execute individual components, refer to the “Components” section below for their individual usage.

CONFIGURATIONS

The RAPTOR.conf file records all options used by RAPTOR. Such options in essence control the way RAPTOR behaves. You can find a sample RAPTOR.conf file in RAPTOR_HOME/data/parameters, where explanations for each option are available as comments preceding that option. Modify the options if you are confident that you know what you are doing. The default options are sufficient for most users.

In certain instances you may want to maintain multiple copies of the RAPTOR.conf file. e.g. different configurations for different sets of sequence data. Then you should put RAPTOR.conf in the working directory (where you invoke RAPTOR). Otherwise, just leave the file in \$RAPTOR_HOME/data/parameters (the default location). If both are present, the former takes effect.

Example options:

OutputPath: specifies where to place output files.

CallModeller: instructs the program to prepare MODELLER input files (top script, atm file with pir alignment and pdb file for the template), and invokes the modelling program.

THREADING METHODS

RAPTOR comes with 3 threading algorithms, or methods, to approach the target sequence and produce results that usually differ from one another. It is up to the user to select methods to run in order to obtain optimal outputs. There are “session” settings in RAPTOR.conf specifying threading method-specific options. Each session represents one threading method and is enclosed in a pair of SESSION_START and SESSION_END lines. A session can be turned on/off by specifying the SessionActive option (immediately following SESSION_START) to yes/no. A “yes” to SessionActive indicates RAPTOR will run that method. The default settings for each SESSION blocks are sufficient for most users.

The three methods are:

1. NP Core: the template is partitioned into a series of cores (which essentially are structurally significant secondary structures).
2. No Core: the template is treated as a sequence rather than a series of cores.

3. IP: the threading algorithm formulates the protein threading problem as a large scale integer/linear programming problem based on the template's contact map graph.

FILE FORMATS

Input

Sequence files should be in FASTA format like the following:

```
>256ba ( len=106 )
ADLEDNMETLNDNLKVIKADNAAQVKDALTKMRAALDAQKATPPKLEDKSPDSPEMKD
FRHGF D I L V G Q I D D A L K L A N E G K V K E A Q A A A E Q L K T T R N A Y H Q K Y R
```

TIP: it is strongly recommended to cut a long sequence into several domains if possible. For example, a sequence of length 600 will not only impact speed, but also accuracy. The optimal sequence length is between 100 and 400. You can use tools like Pfam or ProDom to perform the segmentation.

Threading Output

Threading results are stored in the directory as specified in RAPTOR.conf's "OutputPath". There are several files:

1. XML Output:

Typically named in the pattern of [sequence_file].[x].xml (x is a number), this file contains scores, alignments, and PDB backbones (if PDB files are installed and OutputBackbone is turned on in RAPTOR.conf, which is the default). This file can be visualized in PROView or a web browser. See XMLTagsExplained.txt for a detailed schema.

2. Hits and Scores:

*.[x].scoreRank: a tabular ranking of templates (sorted by E value).

To specify the sorting score, set "TempRankMethod" appropriately in RAPTOR.conf.

*.[x].align.res: contains the alignment position matrices;

*.[x].score.res: has scores of each alignment pair;

The last two are mostly for development and training purposes and are not the most readable.

3. Alignments (MODELLER Input):

The alignment results are in the directory specified in RAPTOR.conf's "ModelPath". The *-model.pir file contains alignments of all threading pairs in one file for easy navigation. The *-*.pir files each contain individual alignment of a threading pair. They are generated since some modelling software do not accept multiple alignments in the same input file.

*-model.al file contains the alignment between the sequence and the template in CASP5 format. Please refer to CASP5 website for detailed information.

We also provide a complete interface to MODELLER, the industry standard tool for modelling. MODELLER inputs are generated in directory specified by option "ModellerInDir". Aside from the alignment files mentioned above, there are *.atm files with ATOM information of the templates extracted from their individual PDB files (which are assumed to be located at \$RAPTOR_HOME/data/pdb/). Lastly, a run.top MODELLER script is in place to start MODELLER.

How to create your own templates from a proprietary database

We assume your proprietary structures have been experimentally obtained and are available in PDB format. You will need the DSSP program (<http://www.cmbi.kun.nl/gv/dssp/>) to assign secondary structures. Please put the 'dsspcmbi' executable under \$RAPTOR_HOME/dssp/.

You can then execute the following command to generate your own template file (see "Templates" below for format explanation):

```
make_template -pdbfile <file> [-c <chain id>] [-n <name>] [-d <start res> <end res>] [-o <file>]
```

What does it mean?

a) -pdbfile <file>

Full path to the PDB format file that the template will be based on. e.g.
C:\RAPTOR\data\pdb\1234.pdb on Windows, or
/home/username/RAPTOR/data/pdb/1234.pdb on Linux.

b) -c <chain id>

The chain to use for the template. NOTE: Prior to 1996 it was not mandatory to indicate chain IDs in PDB files, but since then more and more PDB records have chain IDs explicitly stated. See <http://www.rcsb.org/pdb/lists/pdb-l/200001/msg00019.html>. To decide if you need this option, check if the chain identifier field is in your PDB file.

c) -n <name>

Base name for the template.

d) -d <start res> <end res>

If you are creating a template based on a domain, rather than a chain, this selects the area between and including the start residue ID and the end residue ID. Otherwise, this option can be omitted.

e) -o <file>

Name of the output template.

e.g. Create a (full) template for chain A of 1a2o:

```
make_template -pdbfile data/pdb/1a2o.pdb -c a
```

The output will consist of the template file (.fssp), profiling information in a PSM file from PSI-BLAST, the frequency file .freq.full based on the sequence, as well as the secondary structure prediction by ss_predictor (.ss) based on the frequency file just generated.

NOTE: the “TemplatePath” and “PSMPath” options in RAPTOR.conf refer to the directories where fssp and .psm files are stored, respectively. After you have generated your own templates, you need to change these two options to point to where you stored your newly generated .fssp and .psm files. e.g.

Your 1234.fssp and 1234.psm files are in /home/userName/RAPTOR/data/my_fssp/ and /home/userName/RAPTOR/data/my_psm/ respectively. Your options in RAPTOR.conf should read like this:

```
TemplatePath      $RAPTOR_HOME/data/my_fssp
PSMPath          $RAPTOR_HOME/data/my_psm
```

Where \$RAPTOR_HOME has the value /home/userName/RAPTOR (see RAPTOR.conf for more detail).

Templates

Each template file contains coordinates of backbone alpha- and beta-carbon atoms, among other information. Here is an example:

```
REM lesso with 154 residues (58, 52) and 9 cores
REM  F RS NUM  SS ACC  x-Cb  y-Cb  z-Cb  x-Ca  y-Ca  z-Ca
RES  1  A  2  L  84  3.272  3.144  -9.294  2.420  1.910  -9.163
RES  4  S  3  E  69  3.293  -1.759  -6.554  4.056  -0.439  -6.662
RES  4  E  4  E  78  8.146  -0.153  -4.910  7.456  -1.438  -5.377
RES  4  K  5  E 162  9.042  -5.858  -4.437  8.815  -4.549  -3.671
RES  4  V  6  E  3 10.608  -3.092  0.509 11.042  -4.049  -0.610
RES  4  E  7  E 130 14.309  -7.618  -0.130 13.145  -6.962  0.609
RES  4  M  8  E  0 12.084  -7.323  5.161 13.400  -7.190  4.397
RES  4  N  9  E  43 17.660  -8.126  6.095 16.260  -8.570  6.535
RES  4  L 10  E  38 17.388  -10.690 10.731 17.001  -9.323 10.157
RES  4  V 10A E  9 18.814  -5.342 11.790 19.210  -6.815 11.942
RES  2  T 10B E  76 22.120  -8.432 15.178 21.395  -7.098 15.002
RES  1  S 10C T  58 24.149  -4.845 18.202 23.867  -4.830 16.703
```

Header: a general description of the template:

1eso: template name, same as PDB code
154 residues: total number of residues in the template name
(58, : number of residues as a-helix or β -sheet, i.e., with flag 2-4
52): number of residues considered as core residues, i.e., with flag 4
9 cores: number of core secondary structures

The field labels:

REM: entry label (REM for remarks and RES for protein residues)

F: flag (1 for sequence only, 2 for an alpha-helix or beta-sheet residue, 3 for an alpha-helix or beta-sheet residue without C-alpha coordinates, and 4 for a core residue)

RS: one-letter code of amino acid type (X for residues other than the standard 20 amino acid types).

NUM: residue number in PDB (including possible extra character associated with it).

SS: secondary structure type, using the same convention as in DSSP (e.g., H for alpha-helix and E for beta-sheet).

ACC: Solvent accessible surface area calculated by DSSP.

x-Cb, y-Cb, z-Cb: C-beta coordinates.

x-Ca, y-Ca, z-Ca: C-alpha coordinates.

NOTE: templates in XML format as in the PROSPECT suite are compatible with RAPTOR as well, as any other non-FSSP information (i.e. other than the above) is considered superfluous and is ignored by RAPTOR.

Template Updates

We will periodically update the template database from newly added PDB files. New templates will be integrated into new releases, which are covered under your Annual Technical Support Contract.

Profiling Output

Results of running PSI-BLAST are stored in the directory as specified in RAPTOR.conf's "PSPath". The *.chk file is the machine-dependent binary file comprised of the sequence description, followed by the profile; the *.psp file is the position-specific score matrix calculated. RAPTOR will only use the *.psp file.

Secondary Structure Output

Depending on the predictor program, this step generates different formats, both of which are recognized by RAPTOR. PSIPRED generates its own horizontal and vertical formats (two out files *.ss2 and *.horiz are used by RAPTOR, please do not modify them manually). SS_PRED produces PHD format.

If both SS_PRED's and PSIPRED's outputs are in the same directory specified by SSPath (see RAPTOR.conf), then SS_PRED's outputs are used.

Log Files

There are various logs detailing errors and peculiarities during execution. Here is a brief list:

- * IPThread_cpu.log: each line records the names and sizes of the template/sequence pair, as well as the algorithm and the CPU time (in seconds) used, the number of branch nodes generated if the IP method is used to thread.
- * IPThread_error.log: records exceptions in threading, e.g. if there is no valid alignment between a certain threading pair. Most of the error messages will be in this file, please check this file after threading. Usually, messages "no valid alignment" and "use no core align" do not indicate errors.
- * IPThread_out_of_time.log: records threading pairs that exceeded MaxThreadTime (and hence aborted) as specified in RAPTOR.conf, if any. This file is not always generated.
- * PSP_error.log: records errors in using profiling information in threading.
- * SS_error.log: records errors in using predicted secondary structure information in threading.

COMPONENTS

(This information is provided to you so that you can see all of RAPTOR components. We recommend, however, that you do not touch them unless you call our technical support team for further explanation as they are for Advanced Users of the RAPTOR program and typically do not need to be changed.)

RAPTOR contains/incorporates the following components in its workflow:

1. Profiling information generation via PSI-BLAST.

PSI_BLAST from NCBI is included in the RAPTOR suite and is invoked by the 'GenPSP' script (as specified by the "PSP_program" in RAPTOR.conf).

Usage: GenPSP [seq_file] [output_directory]

Profiling outputs are stored in the directory as specified by "PSPPath" in RAPTOR.conf.

2. Secondary structure prediction.

The default predictor used is `ss_predictor` (from RAPTOR's precursor PROSPECT), which can be invoked by the script '`SS_PRED`' (as specified by "`SSpred_program`" in `RAPTOR.conf`).

Usage: `SS_PRED [seq_file] [output_directory]`

RAPTOR also comes with a script `PSIPRED` (in `$RAPTOR_HOME/bin/`), that invokes the `PSIPRED` program, which is another predictor (<http://bioinf.cs.ucl.ac.uk/psipred/>). If you have it installed on your system, please put it at `$RAPTOR_HOME/PSIPRED/` (you can make a link there pointing to wherever it is located).

Usage: `PSIPRED [seq_file] [output_directory]`

The predicted secondary structure outputs are stored in the directory as specified by "`SSPATH`" in `RAPTOR.conf`.

3. Threading engine.

The engine of RAPTOR is the 'thread' program. Usage:

`thread [-a algorithm] [-s ssfile] [-t template_type] [-w weight-file] [-l] [-r] templateName sequenceName`

Options:

1) `-a algorithm`: specifies threading methods (one of the following):

`NoCoreAlign`: dynamic programming used to align two sequences (the query sequence and the template sequence)

`np`: dynamic programming used to align the query sequence to the template, but the template is parsed as a series of cores connected by loops.

`dc`: the divide and conquer algorithm used to align the query sequence to the template, pairwise interactions are treated rigorously.

`IP`: the integer programming algorithm used to align the query sequence to the template, pairwise interactions are treated rigorously.

`smart`: a new algorithm combining the advantage of `dc` and `IP`, pairwise interactions are treated rigorously.

2) `-s ssfile`: specifies the predicted secondary structure file. If not provided, the program will go to `SSPath` (set in `RAPTOR.conf`) to find one.

3) `-t template_type`: specifies the type of templates – or the suffix of template files. It can be set as "`TemplateType`" in `RAPTOR.conf`.

4) `-w weight_file`: weight factor file can be set in `RAPTOR.conf` (`PWWeightFile` and `NPWeightFile`, for different threading techniques). It specifies how much each component in the energy function should

contribute in the final score. Note that weights have been trained and tested for optimal result, so the default files is adequate for most users. If you do use your own weight factors, then please use your own SVM model to rank the templates as well.

5) -l: for weight factor training purposes only. If the IP method is used, this option will only generate linear solution rather than integral solution. Note: When solving Integer Programming, we always first relax it to a linear program. The linear program is solved and its solution is called linear solution. Then a branch-and-bound method is used to convert the linear solution to integral solution (the solution of integer program). If the -l option is used, then the branch-and-bound process is not applied. So the solution is just linear.

6) -r: for training and assessment purposes only. If the SARF result of this threading pair is available, this option will calculate the alignment accuracy of this pair generated.

templateName: name of the template file (no suffix) to run threading against.

sequenceName: input sequence file name (no suffix).

e.g. to thread a sequence file called 256ba.seq against template file 7rsa.xml (in /data/templates/), type (ignore other options):

```
thread 7rsa 256ba
```

For Windows users, the program's path can be "C:\Program Files\thread". If there are white paces in the parameter value, use quotes "".

4. Batch threading.

RAPTOR has a "batch" program, 'bThread_smp', that automates *ONLY* the threading process of a large number of sequences (if you need all of preprocessing, threading, and postprocessing automated, please use the above RAPTOR_server command). It is multithreaded, and hence can utilize multiple CPUs if they are available.

Usage: bThread_smp [-a algorithm] [-c thread_program_name]

1) -a algorithm, same as "-a" option in thread

2) -c thread_program_name, this option allows you to specify the threading program path, the default value is thread.

5. SVM estimate.

Because templates vary in length and other characteristics, the "raw" energy scores are not suited for direct comparison. Therefore a standardized statistical measure, Z-score, is used instead. It is finalized by application of Support Vector Machine (SVM). In the RAPTOR suite it is done by the tool 'estimate'.

Usage: estimate -s score_file -a align_file [-t top] [-n seq_name]

Options:

- 1) -s score_file, the score file generated by threading programs, it has the same meaning as ThreadResultFile specified in RAPTOR.conf
- 2) -a align_file, the alignment file generated by threading programs, it has the same meaning as ThreadAlignFile specified in RAPTOR.conf
- 3) -t top, output "top" alignment models for this sequence. Default value is 10.
- 4) -n seq_name, specified the sequence name, if specified, estimate will generate an output file named <seq_name>.scoreRank. Otherwise, estimate will generate an out file named raptor000.svm.data.res

estimate will also read RAPTOR.conf, so the users can set the method to rank the templates in RAPTOR.conf before invoking estimate.

6. PROView: sequence-structure alignment viewer.

RAPTOR comes with a PyMOL-based molecular 3D visualization tool, PROView, which is used to view threading outputs (sequence alignment and backbone structure).

REFERENCES

For detailed information on the threading algorithm, please refer to the following two papers:

- 1) Jinbo Xu and Ming Li,
Assessment of RAPTOR's Linear Programming Approach in CAFASP3,
CASP5 special issue, Proteins: Structure, Function and Genetics
- 2) Jinbo Xu, Ming Li, Dongsup Kim and Ying Xu,
RAPTOR: Optimal Protein Threading by Linear Programming
Journal of Bioinformatics and Computational Biology,
Vol. 1, No. 1 (2003) 95-117

The first one aims for biologists and the latter contains more mathematical content.

COPYRIGHTS

Copyrights of the following external programs and tools belong to their individual copyrights holders.

1. Profiling is done via the BLASTPGP tool, part of the PSI-BLAST suite developed by NCBI (National Centre of Biotechnology Information). More information on the BLAST family of tools can be found at

<http://www.ncbi.nlm.nih.gov/BLAST/>

The executable can be obtained from

<ftp://ftp.ncbi.nih.gov/blast/executables/>

2. PyMOL is developed and maintained by DeLano Scientific LLC.
(<http://pymol.sourceforge.net/>)

Thank you for choosing BSI products. Please do not hesitate in contacting us at support@BioinformaticsSolutions.com for more information, or to give us comments and suggestions.

The BSI team.