J Mol Model (2011) 17:2601–2607
DOI 10.1007/s00894-011-0960-4

ORIGINAL PAPER

SWIFT MODELLER: A JAVA based GUI for molecular modeling

Abhinav Mathur · Shankaracharya · Ambarish S. Vidyarthi

Received: 26 November 2010 / Accepted: 4 January 2011 / Published online: 22 January 2011

Abstract MODELLER is command line argument based software which requires tedious formatting of inputs and writing of Python scripts which most people are not comfortable with. Also the visualization of output becomes cumbersome due to verbose files. This makes the whole software protocol very complex and requires extensive study of MODELLER manuals and tutorials. Here we describe *SWIFT MODELLER*, a GUI that automates formatting, scripting and data extraction processes and present it in an interactive way making MODELLER much easier to use than before. The screens in *SWIFT MODELLER* are designed keeping homology modeling in mind and their flow is a depiction of its steps. It eliminates the formatting of inputs, scripting processes and analysis of verbose output files through automation and makes pasting of the target sequence as the only prerequisite. Jmol (3D structure visualization tool) has been integrated into the GUI which opens and demonstrates the protein data bank files created by the MODELLER software. All files required and created by the software are saved in a folder named after the work instance's date and time of execution. *SWIFT MODELLER* lowers the skill level required for the software through automation of many of the steps in the original software protocol, thus saving an enormous amount of time per instance and making MODELLER very easy to work with.

Keywords Loop modeling · MODELLER · Molecular modeling · *SWIFT MODELLER*

A. Mathur · Shankaracharya · A. S. Vidyarthi (✉)
Department of Biotechnology, Birla Institute of Technology, Mesra, 835215, Jharkhand, India
e-mail: asvidyarthi@bitmesra.ac.in
URL: http://www.bitmesra.ac.in/swift-modeller/swift.htm

Introduction

MODELLER is a computer program for comparative protein structure modeling developed by Andrej Sali [1, 2] at the University of California, San Francisco. The input to the program is an alignment of a sequence to be modeled with the template structure(s), the atomic coordinates of the template(s), and a simple script file. MODELLER then calculates a model containing all non-hydrogen atoms. Apart from model building, MODELLER can perform auxiliary tasks such as fold-assignment [3] alignment of two protein sequences or their profiles [4, 5], multiple-alignment of protein sequences and/or structures [6], clustering of sequences and/or structures, and *ab initio* modeling of loops in protein structures [2].

MODELLER works on Command Line arguments, and lacks a Graphic User Interface (GUI). It presents a complex software protocol requiring an extensive study of manuals and tutorials. Knowledge of Python scripting becomes a must for advanced usage. Furthermore, the output is stored in verbose files which make data analysis cumbersome. Altogether a lot of time is wasted in writing scripts, formatting the inputs and data analysis.

There have been few attempts earlier to ease out use of MODELLER by providing or developing a GUI base. These were MINT and EasyModeller [7]. The former was developed on MODELLER 8 and is since a dormant product. EasyModeller was developed later to MINT and it maintains a single screen approach where all functionalities of MODELLER have been placed on activity buttons and arranged on a single screen. Sequential flow of information in MODELLER has not been given due importance in arrangement of the buttons.

Here we present *SWIFT MODELLER*, a GUI that encompasses the limitations of MODELLER and provides

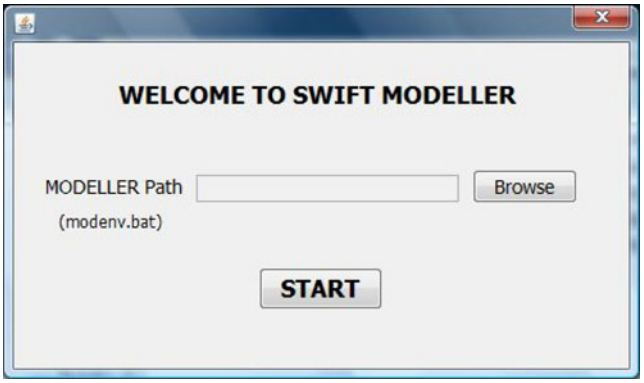


Fig. 1 Login screen of *SWIFT MODELLER*

a much easier way to use the same. It automates the tiresome steps such as formatting of inputs, writing and execution of python scripts and meaningful data extraction from output files, thus saving an enormous amount of time and human effort for performing homology modeling with MODELLER. In addition it makes pasting of the protein sequence as the only prerequisite, thereby lowering the skill level required for using MODELLER.

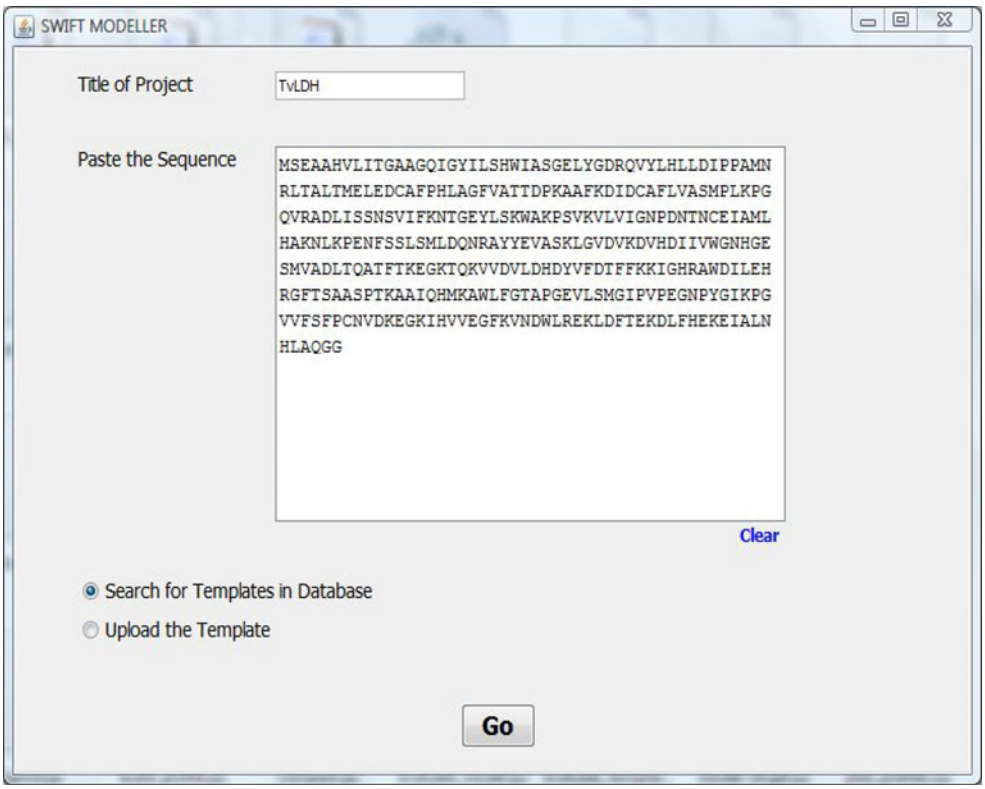
Implementation

The screens of SWIFT MODELLER are designed in such a way that they depict the MODELLER protocol and steps of homology modeling. First time users of the GUI are greeted with a login screen where they would be required to enter the installation path of MODELLER (Fig. 1). This screen will appear only once on a terminal until and unless the file containing the path is deleted.

The protein sequence input is taken along with a suitable five letter title to the project in the first screen of the GUI (Fig. 2). The PIR database containing PDB sequences clustered at 95% identity has been downloaded and embedded in this software as an offline database. The first screen gives an option to the user to search templates for the target sequence in this offline database, alternatively user can upload previously downloaded sequences (after doing BLAST search of the target sequence) also.

On selecting the database search option the software edits a python file named "build_profile.py", in the background and executes the same via MODELLER which results in creation of an output file named "build_profile.prf", which contains search results. Our software extracts data from this file and presents it in a tabular and well readable form (Fig. 3). The table contains template sequence codes, sequence identity percentage and E-value, the first column being a checkbox for selection of suitable templates. On selection of the relevant templates the system downloads these from the PDB website at the background and extracts these in the working directory. By default a single template is considered however, an option of using multiple templates is also available.

Fig. 2 Input sequence and title



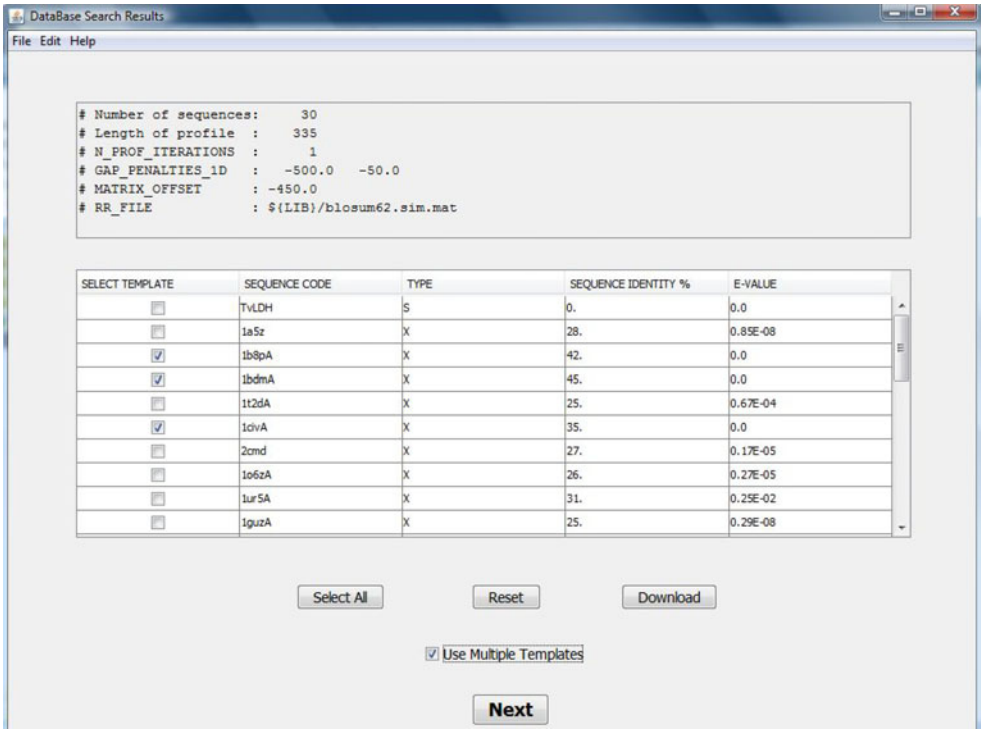


Fig. 3 Database search results

These templates become input to another python file named "compare.py" which is executed and data (dendrogram) is extracted from a log file named "compare.log" and displayed (Fig. 4). Radio buttons against each template is provided for easy selection on which the target sequence is to be modeled. The selection of the template is made on the basis of crystallographic resolution factor and phylogeny score. The target sequence is then aligned to the selected template via "align2d.py" and the alignment is then displayed (Fig. 5). The user is asked to provide the number of models to be constructed on the basis of the target-template alignment.

The models are then constructed by the "model-single.py" (or "model_mult.py" depending upon the mode

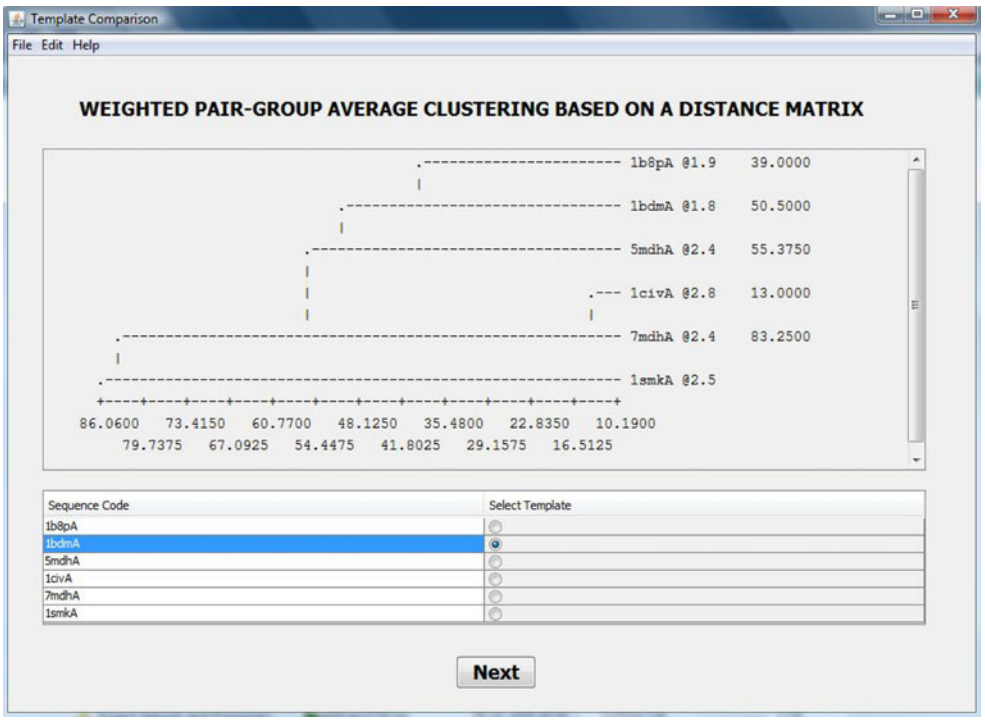


Fig. 4 Dendrogram showing crystallographic R-factor and phylogenic score

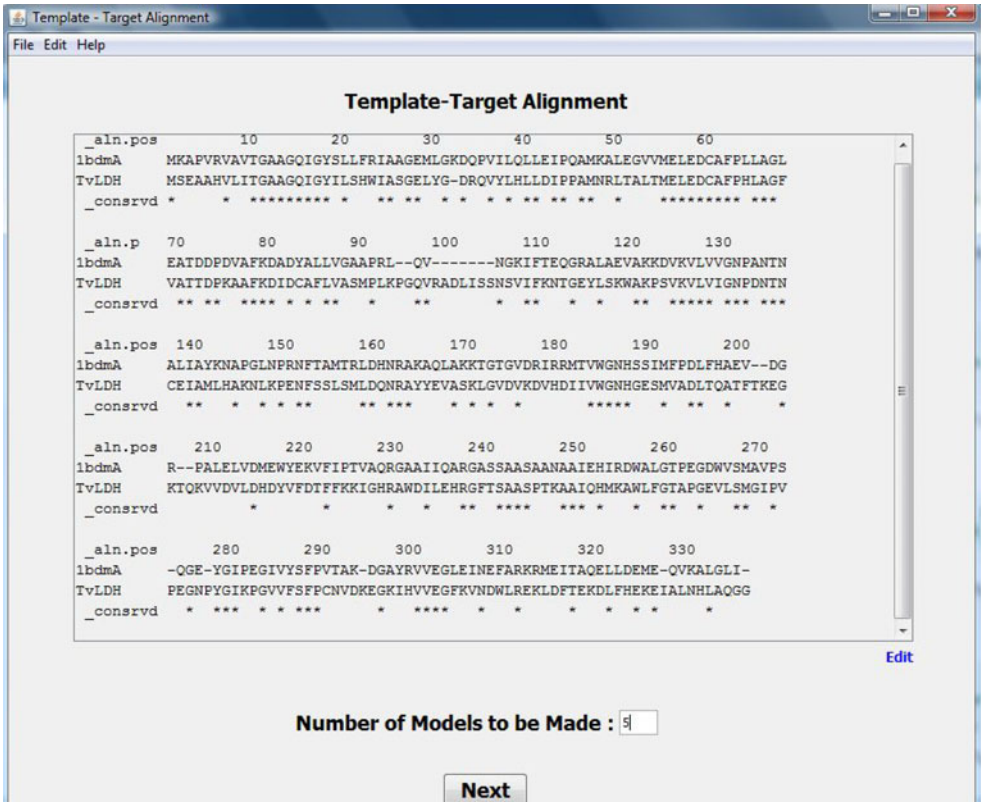


Fig. 5 Target-template alignment

selected – single template or multiple templates). The DOPE assessment score of the built models is displayed along with GA341 and molpdf scores (Fig. 6). Evaluation graphs are now plotted for each model which gives the comparison in the pseudo energy level of every amino acid of the built model and the template used. A series of python scripts are edited and executed for this purpose ("evaluate_model.py", "evaluate_template.py" and "plot_profiles.py"). This plotted result is then framed and displayed with an option for loop modeling (Fig. 7).

Loop Modelling is performed if the evaluation graph shows irregularities between the model and template's plots in some pockets of amino acid residues. The user is required to enter the amino acid residue range and the number of loop models to be made, on selection of the loop modeling option. "loop_refine.py" performs loop modeling on the selected model in the denoted amino acid residue range. The assessment scores of the constructed loop models are displayed. The evaluation graphs for the newly constructed models can be plotted through this screen

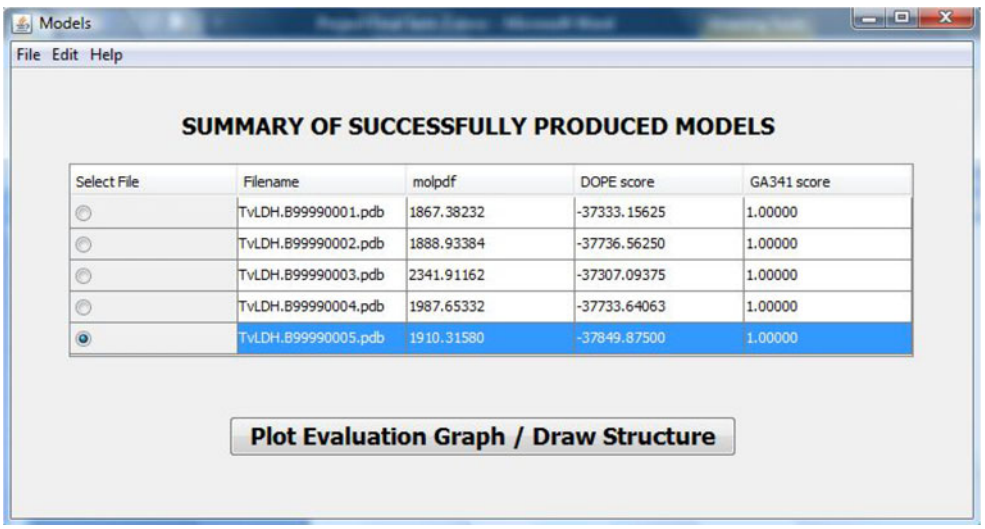


Fig. 6 List of models with DOPE and GA341 scores

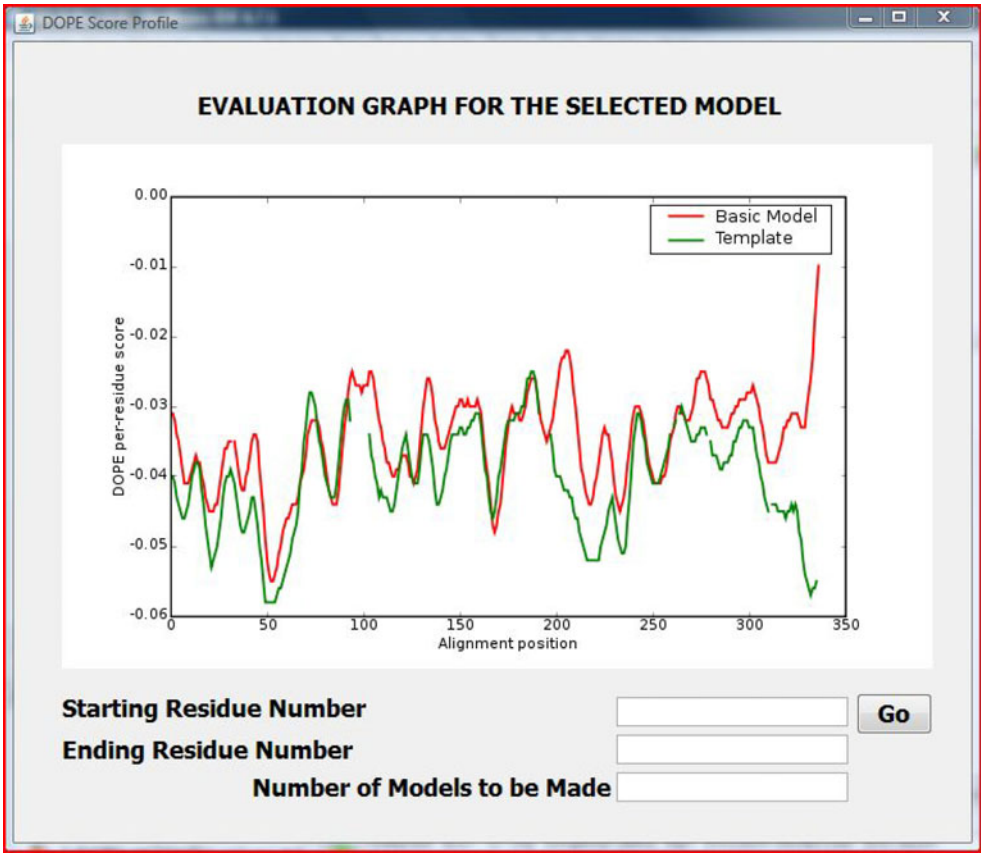


Fig. 7 DOPE score plot with loop modeling enabled

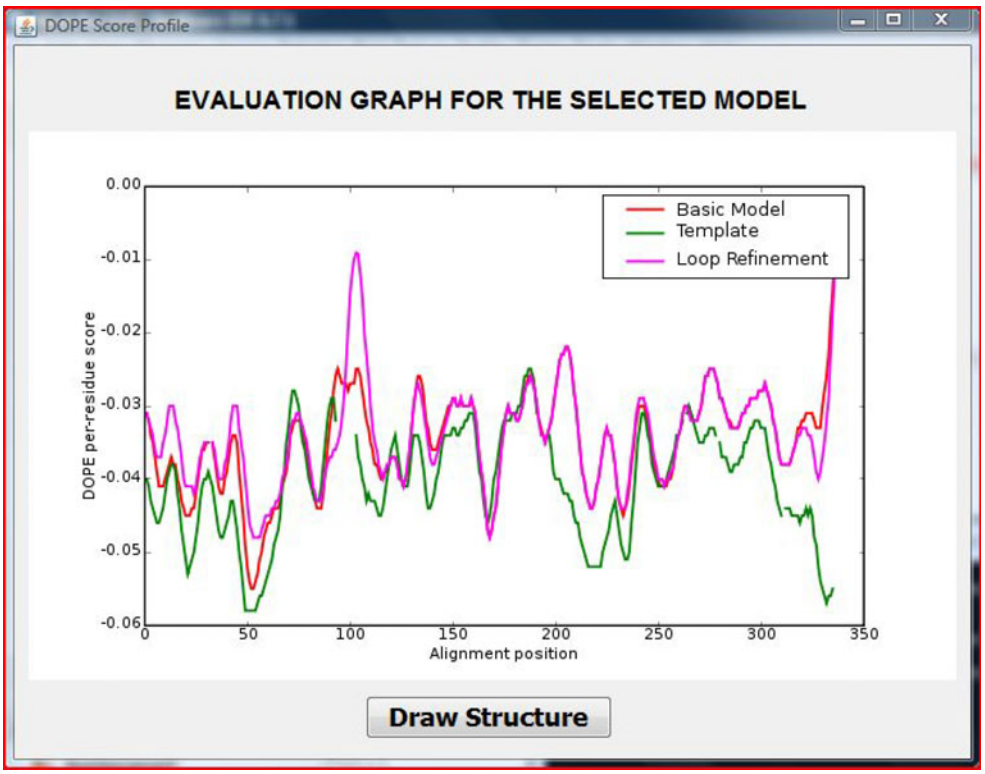


Fig. 8 DOPE score plot after loop refining

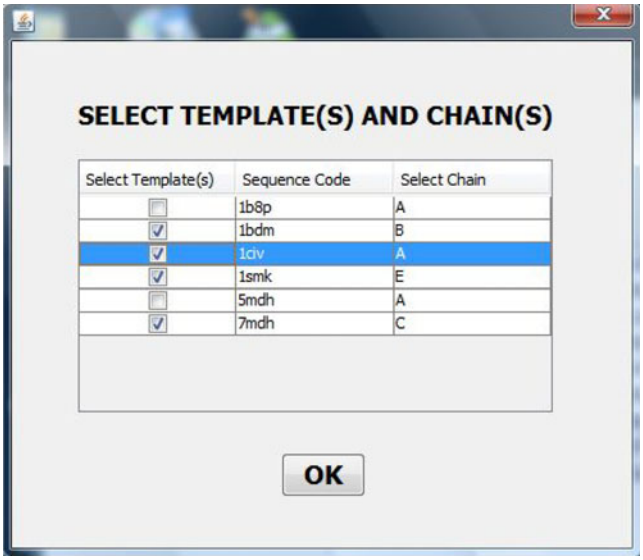


Fig. 9 Table for selection of chains extracted from the uploaded templates

which contains plots for the template, original model and the loop model in different shades for better visualization (Fig. 8). Finally Jmol: an open-source Java viewer for chemical structures in 3D. http://www.jmol.org/ [8] is run through the GUI which displays the 3D animated view of the model thus constructed.

Results

All the python scripts utilized by the MODELLER for performing homology modeling are edited and executed at the backend by *SWIFT MODELLER* along with processes such as formatting of protein sequence input into MODELLER recognizable ALI format, and data extraction from verbose output files which is presented in meaningful graphs and tables.

Furthermore, *SWIFT MODELLER* extracts the PDB chains from the templates selected by the user and displays them for easy selection (Fig. 9). Also, the constructed models are automatically opened with Jmol for immediate viewing and analysis of the selected model (Fig. 10). *SWIFT MODELLER* has adopted a sequential approach

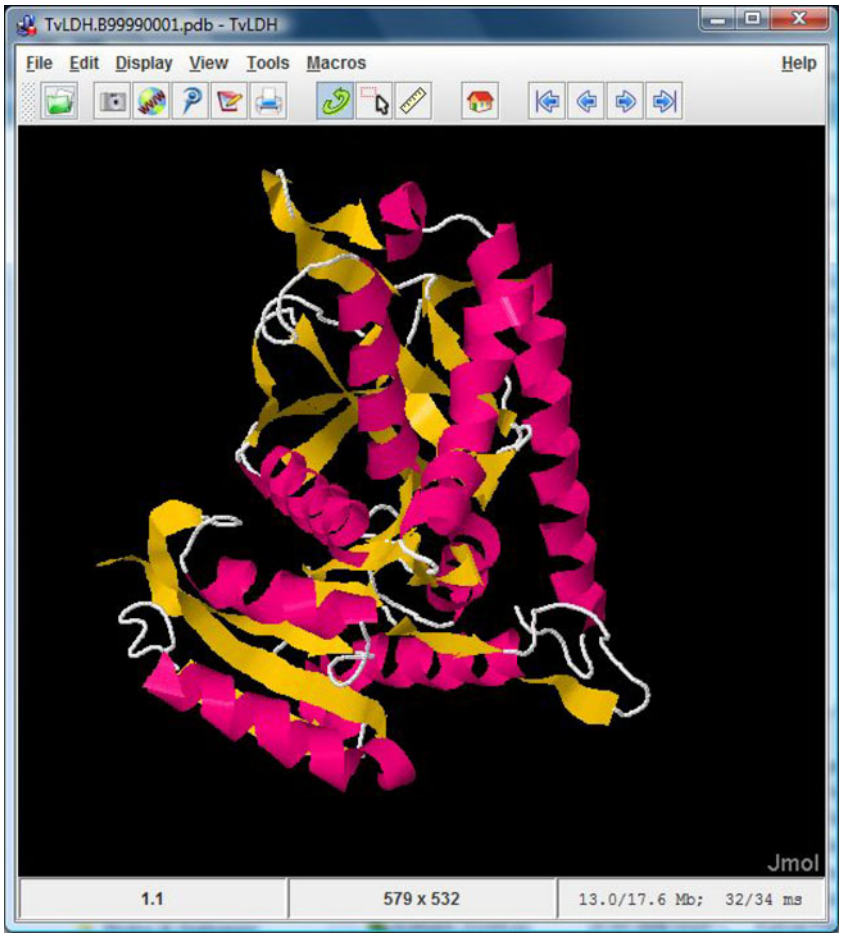


Fig. 10 Jmol showing the animated 3D structure of the constructed model

Table 1 Time saving benefits of *SWIFT MODELLER*

Activity	Time taken by the MODELLER procedure [a]	Time taken by the *SWIFT MODELLER* procedure [b]
Save target sequence in MODELLER recognized format	1 min	–
Selection and download of suitable template(s) from Protein Data Bank website	5 min	3 min [c]
Edit the required python files	30 min [d]	–
Execution of the python files using MODELLER	15 min	15 min
Open the model.pdb file with a visualizing software like Jmol	1 min	–
Total time taken	***52 min***	***18 min***

[a] Time considered varies with user's capability and system's configuration

[b] '–' indicates automated process for which time taken can be rendered negligible

[c] Time saved due to automated downloading function present in *SWIFT MODELLER*

[d] Time calculated as an approximate estimate to modify 15 python scripts

where individual operations are described and defined on separate screens. These have primarily been done to offer ease of operation and provide a definite flow path to the software. These enhancements make *SWIFT MODELLER* a silver-edged solution over other available products like EasyModeller.

The automation of the afore mentioned steps leads to a lot of saved time (see Table 1) and lowering of the skill level required for using the MODELLER software making pasting of the protein sequence as the only prerequisite.

Availability and requirements

Project name:	SWIFT MODELLER
Project homepage:	http://www.bitmesra.ac.in/swift-modeller/swift.htm
Operating system:	Windows XP, Windows VISTA and Windows 7
Other requirements:	The system must have MODELLER (any version) and Python (2.3.5) along with matplotlib (0.90.1) and numpy (1.0.4) preinstalled. Microsoft Windows Vista or 7 users must install MODELLER in a partitioned drive other than C:/ to avoid administrator rights related hassles.
License:	Free to use
Any restriction to use by non-academics:	None

Acknowledgments The authors are thankful to the Sub-Distributed Information Center (BTISnet SubDIC), Department of Biotechnology (No. BT/BI/04/065/04), New Delhi, India for financial support and Department of Biotechnology, Birla Institute of Technology, Mesra, Ranchi for providing the infrastructure facilities for the present study. The first author acknowledges the contribution of his colleague Mr Navjeet Kumar for his technical support.

References

1. Šali A, Blundell TL (1993) Comparative protein modelling by satisfaction of spatial restraints. J Mol Biol 234:779–815
2. Fiser A, Do RK, Šali A (2000) Modeling of loops in protein structures. Protein Sci 9:1753–1773
3. Eswar N, Madhusudhan MS, Marti-Renom MA, Šali A (2005) BUILD_PROFILE: A module for calculating sequence profiles in MODELLER
4. Marti-Renom MA, Madhusudhan MS, Šali A (2004) Alignment of protein sequences by their profiles. Protein Sci 13:1071–1087
5. Eswar N, Madhusudhan MS, Marti-Renom MA, Šali A (2005) PROFILE_SCAN: A module for fold-assignment using profile-profile scanning in MODELLER
6. Madhusudhan MS, Marti-Renom MA, Sanchez R, Šali A (2006) Variable gap penalty for protein sequence-structure alignment. Protein Eng Des Sel 19:129–133
7. Kuntal BK, Aparoy P, Reddanna P (2010) EasyModeller: A graphical interface to MODELLER. BMC Res Notes 3:226–330
8. Herráez A (2006) Biomolecules in the computer: Jmol to the rescue. Biochem Mol Biol Edu 34:255–261