The PyRosetta Toolkit: A Graphical User Interface for the Rosetta Software Suite

Jared Adolf-Bryfogle[1,2], Roland L. Dunbrack Jr.[1]*

1 Institute for Cancer Research, Fox Chase Cancer Center, Philadelphia, Pennsylvania, United States of America, **2** Drexel University College of Medicine, Program in Molecular and Cell Biology and Genetics, Philadelphia, Pennsylvania, United States of America

Abstract

The Rosetta Molecular Modeling suite is a command-line-only collection of applications that enable high-resolution modeling and design of proteins and other molecules. Although extremely useful, Rosetta can be difficult to learn for scientists with little computational or programming experience. To that end, we have created a Graphical User Interface (GUI) for Rosetta, called the *PyRosetta Toolkit*, for creating and running protocols in Rosetta for common molecular modeling and protein design tasks and for analyzing the results of Rosetta calculations. The program is highly extensible so that developers can add new protocols and analysis tools to the PyRosetta Toolkit GUI.

Citation: Adolf-Bryfogle J, Dunbrack RL Jr (2013) The PyRosetta Toolkit: A Graphical User Interface for the Rosetta Software Suite. PLoS ONE 8(7): e66856. doi:10.1371/journal.pone.0066856

Editor: Heinrich Sticht, Universität Erlangen-Nürnberg, Germany

Received January 31, 2013; **Accepted** May 11, 2013; **Published** July 9, 2013

Funding: Funding from National Institutes of Health R01 GM84453. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing Interests: The authors have declared that no competing interests exist.

* E-mail: Roland.Dunbrack@fccc.edu

Introduction

The Rosetta Molecular Modeling suite is a collection of command-line-only applications encompassing approximately 1.7 million lines of C++ code. Within the suite there are numerous applications for modeling and design ranging from minimization of the scoring function to enzyme redesign. Rosetta is an extremely useful piece of software for a variety of molecular modeling tasks [1,2,3,4,5,6].

Due to the wealth of useful C++ classes within Rosetta and the ease of Python programming for beginners and advanced users alike, Sergey Lyskov and the PyRosetta team created Python bindings for Rosetta [7]. PyRosetta allows direct access to nearly all Rosetta functions and classes using Python scripts, programs, and the interactive IPython/Python terminal.

To allow beginning users, including molecular biologists with little or no computational experience, to use Rosetta with ease, we have created a Graphical User Interface (GUI), which we call the *PyRosetta Toolkit*, using PyRosetta as the underlying Rosetta code. The PyRosetta Toolkit was created with code simplicity in mind, allowing users to add their own functions, menus, and windows. Although stand-alone Foldit [8,9] can be used to run some basic Rosetta algorithms such as repacking and minimization on a single model, many modeling tasks are either difficult or non-existent in FoldIt. As such, we believe the PyRosetta Toolkit represents the first major GUI to the full functionality of the Rosetta software suite.

A complete overview and description of the PyRosetta Toolkit GUI code base and a tutorial on how to add to the GUI can be found online through the PyRosetta website (www.pyrosetta.org/documentation) and in the GUI's documentation directory. Here we provide an introduction to the GUI and an overall description.

Results

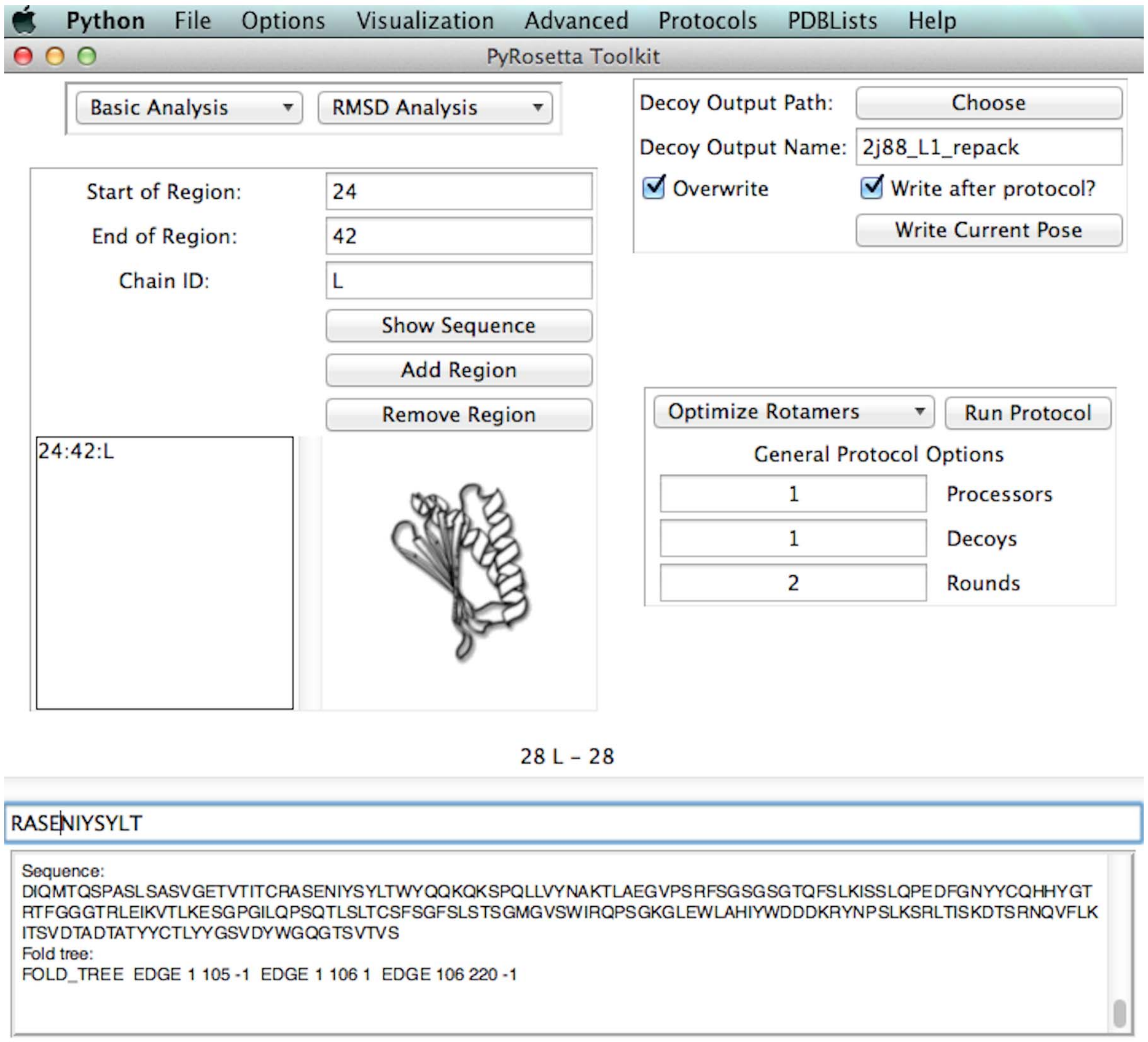
The PyRosetta Toolkit is composed of two main areas – the *main window* and the *main menu*. The main window, shown in Figure 1, allows users to specify protein regions and output options, perform quick analyses, or run standard protocols such as relaxing or repacking structures and regions. The main menu houses many additional functions and more advanced Rosetta methods and protocols. In addition, PyMOL [10] visualization of poses, coordinate changes, and energetics through the PyMOL-Mover [11] are integrated into the GUI.

The main window (Figure 1) acts as a place for the user to prepare or quickly analyze a protein of interest as well as to setup decoy and output options for the session. Main functions in this window encompass region selection where a user can: choose multiple loops, chains, and termini; control decoy output, energy, and RMSD analysis; and perform common protocols such as energy minimization or rotamer repacking through both Rosetta and SCWRL4 [12]. In addition, it has a textbox where most Rosetta and Python output has been redirected to make it easier to observe results of an analysis, protocol, or file load. The sequence of the selected region is also given, and clicking next to a residue will show the PDB number and chain as well as the internal Rosetta residue numbers, which are used in a number of Rosetta applications outside of the GUI.

The menus in the PyRosetta Toolkit are as follows.

File Menu:

The File Menu allows the user to load a structure from a PDB file or directly from the Protein Data Bank, prepare a PDB for use in Rosetta, save and load GUI sessions, and import or export a variety of Rosetta filetypes (Table 1).



Decoy Output Path:
Decoy Output Name: 2j88_L1_repack
[✓ Overwrite] [✓ Write after protocol?]

Start of Region: 24
End of Region: 42
Chain ID: L

24:42:L

1 Processors
1 Decoys
2 Rounds

28 L – 28

RASENIYSYLT

```
Sequence:
DIQMTQSPASLSASVGETVTITCRASENIYSYLTWYQQKQKSPQLLVYNAKTLAEGVPSRFSGSGSGTQFSLKISSLQPEDFGNYYCQHHYGT
RTFGGGTRLEIKVTLKESGPGILQPSQTLSLTCSFSGFSLSTSGMGVSWIRQPSGKGLEWLAHIYWDDDKRYNPSLKSRLTISKDTSRNQVFLK
ITSVDTADTATYYCTLYYGSVDYWGQGTSVTVS
Fold tree:
FOLD_TREE  EDGE 1 105 -1  EDGE 1 106 1  EDGE 106 220 -1
```

Figure 1. PyRosetta Toolkit GUI. *Top-Left*: Energy and RMSD analysis functions. *Bottom-Left*: Region selection. *Top-Right*: Output options. *Bottom-Right*: Decoy and Protocol options as well as some common modeling protocols.
doi:10.1371/journal.pone.0066856.g001

In addition, the RosettaFlagFileBuilder GUI can be launched using this menu. This GUI is independent of PyRosetta and functions in helping a user create a Rosetta command line flag file, run Rosetta applications, and launch them on a cluster using qsub, a commonly used job submission and queuing system for cluster computers (Figure 2). It parses Rosetta Doxygen documentation for recommended options/values and the many other options available for each C++ application. A user can explore the documentation for the many applications within Rosetta interactively, as well as get information and default values for each option of each application. Functions are available for constructing or loading a flag file (usable by Rosetta command-line applications), running them locally, or submitting them to the qsub queue.

Options Menu:

The Options Menu allows the user to set the number of processors to use, setup the main score function, and interact with the Rosetta options system.

The score function window (Figure 3) allows a user to choose any weight set or patch in Rosetta, as well as modify the weight of any score term or add any score term that is not being used to the current score function. This score function is then used by any function in the GUI that requires one and can be modified at any time. A user can also save the new score function or set the GUI to use a particular set of weights by default.

General Rosetta options, such as -dun10 (the 2010 Dunbrack rotamer library [13]) or -ex2 (extra chi2 rotamers), can be set through a window where a user can select from a few common options, enter custom ones, save and load a set of options, or set defaults for the GUI.

Visualization Menu:

The Visualization Menu allows the user to integrate modeling tasks directly with PyMOL using Rosetta's PyMOLMover. A user can set Rosetta to continually send models to the PyMOL program upon structural change or send the current structure (or pose in Rosetta parlance). The small Advanced PyMOL Visualization window allows users to color and label per-residue energies based on the score function or individual score term and send other useful information including DSSP secondary structure, identified hydrogen bonds, and the polar identity of each residue. This window becomes an integral part of interactive Rosetta

Table 1. Function Overview.

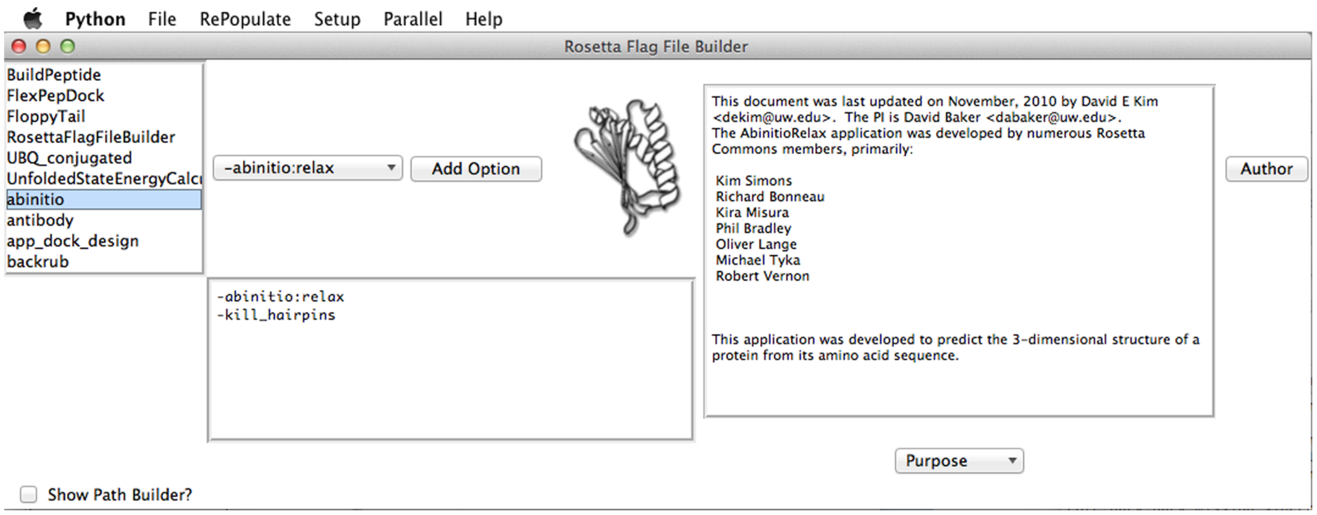
Input filetypes	
PDB File	Rosetta Loop File
RCSB PDB ID	ParameterPath list
PDBList	GUI Session
Output filetypes	
PDB File	
FASTA for protein and/or regions	Rosetta Flag File
Rosetta Loop File	SCWRL Seq File
Rosetta Resfile	Parameter PathList
Rosetta Blueprint file	GUI Session
Protocols	
Fixed Backbone Design	Rotamer Repacking (With SCWRL Integration)
Low Resolution Docking	Low Resolution Loop Modeling (CCD + KIC)
High Resolution Docking	High Resolution Loop Modeling (CCD + KIC)
FastRelax	Grafting
ClassicRelax	FloppyTail
Energy Minimization	
Server Links	
ROSIE [32]	Robetta – Fragments [35]
Rosetta Backrub server [36]	Robetta – Interface Alanine Scan [37]
Rosetta Scaffold Select [38]	Robetta – DNA Interface Scan
Analysis	
RMSD Tools	Packing Analyzer
Energy Analysis	Design Result Breakdown
Interface Analyzer	Rotamer/Per-Residue analysis
Loop Analyzer	Arbitrary data insertion into B-Factor column
VIP Analyzer	
PDBList Functions	
Design Result Breakdowns	Top % or # by energy
Rescore PDBList	Energy vs RMSD
FASTA output of proteins and regions	
PDB Prep functions	
Water removal	Insertion code expansion
HETATM removal	Residue and atom renaming
Renumber from 1	Detection of unknown residues/Loading of params for off-by-default residues
Other	
Constraints	Scorefunction Creation
Non-Canonical AA integration	Resfile Creation
Variant + NCAA Mutagenesis	

doi:10.1371/journal.pone.0066856.t001

modeling through the PyRosetta Toolkit. Note that PyMOL needs to be run separately alongside the GUI to take advantage of these visualization tools.

Advanced Menu:

The Advanced Menu houses a variety of sub-windows and useful functions for analyzing Rosetta results. Four Rosetta-specific analyzers are implemented, including the Void Identification and Packing Analyzer (VIP) [14], Packstat [15], InterfaceAnalyzer [6], and LoopAnalyzer [6]. A user can also enable constraints for the pose and score function in the menu. A sub-window for setting up and exporting a Rosetta resfile, a file for specifying mutatable residues for a design run, is shown in Figure 4. Commonly used data for accessible surface area, surface probability [16], and relative mutability [17] are given for each current residue and potential mutant. In addition, a user can select all conservative mutations for a given position or a range of positions as well as all residues of a specific type (hydrophobic, hydrophilic, charged, etc.). A Rosetta resfile for the pose can then be exported, and the fixed-backbone protocol can be run from within the GUI, making setting up and running the Rosetta design protocol easier.



```
-abinitio:relax
-kill_hairpins
```

This document was last updated on November, 2010 by David E Kim <dekim@uw.edu>. The PI is David Baker <dabaker@uw.edu>.
The AbinitioRelax application was developed by numerous Rosetta Commons members, primarily:

Kim Simons
Richard Bonneau
Kira Misura
Phil Bradley
Oliver Lange
Michael Tyka
Robert Vernon

This application was developed to predict the 3-dimensional structure of a protein from its amino acid sequence.

Figure 2. Rosetta Flag File Builder. *Top-Left*: All Rosetta applications found in a user-specified directory. *Top-Middle*: All options parsed from Rosetta Doxygen documentation housed in rosetta_source. Clicking an option gives a description, if any is found, in the textbox on the right. *Bottom-Middle*: Text window that functions in building a config file. Adding an option will add it to this textbox, while the PathBuilder allows users to search for various files and add them to the textbox. *Right*: Information textbox which gives information on individual options as well as each major component of Rosetta Doxygen documentation (Purpose, Unparsed Options, Code and Demo, References, Algorithm, Limitations, Modes, Input Files, Tips, Expected Outputs, and Post Processing).
doi:10.1371/journal.pone.0066856.g002

Two other integral sub-windows are housed in the Advanced Menu. These include the Ligand/Post-Translational Modification (PTM)/Non-canonical Amino Acid (NCAA) Manager and a window for per-residue control and analysis.

The Ligand/PTM/NCAA Manager (Figure 5) deals with the various parameter and patch files housed in the Rosetta chemical database. Due to memory restrictions, not all of these are enabled by default in Rosetta. Besides functioning to enable these non-canonical amino acids, post-translational modifications, and ligands that the parameter files describe, a user can explore these files in a way that enables identification of the three letter residue code Rosetta reads, determination of whether it is on by default, and of what variant type it is if any. In addition, functions that change the score function to model the physical properties of the non-standard residues or residue modifications, such as the mm_std score function developed by Renfrew et al.[18], are provided. Users may also mutate any residue to these non-canonical amino acids from this window, as long as they are enabled and the appropriate rotamer library has been included.

Finally, the Per-Residue Control and Analysis Window (Figure 6) allows a user to manipulate, design, and analyze individual residues of a pose. Per-residue information includes rotamer energy, approximate rotamer probability calculated by $p = e^{-E}$, residue energy, and energy of the residue by individual score term. Users can change individual dihedral angles, mutate to any canonical amino acid, add variant types such as phosphorylations or acetylations, and repack rotamers and relax individual residues and residues in the vicinity of the chain.

Protocols Menu:

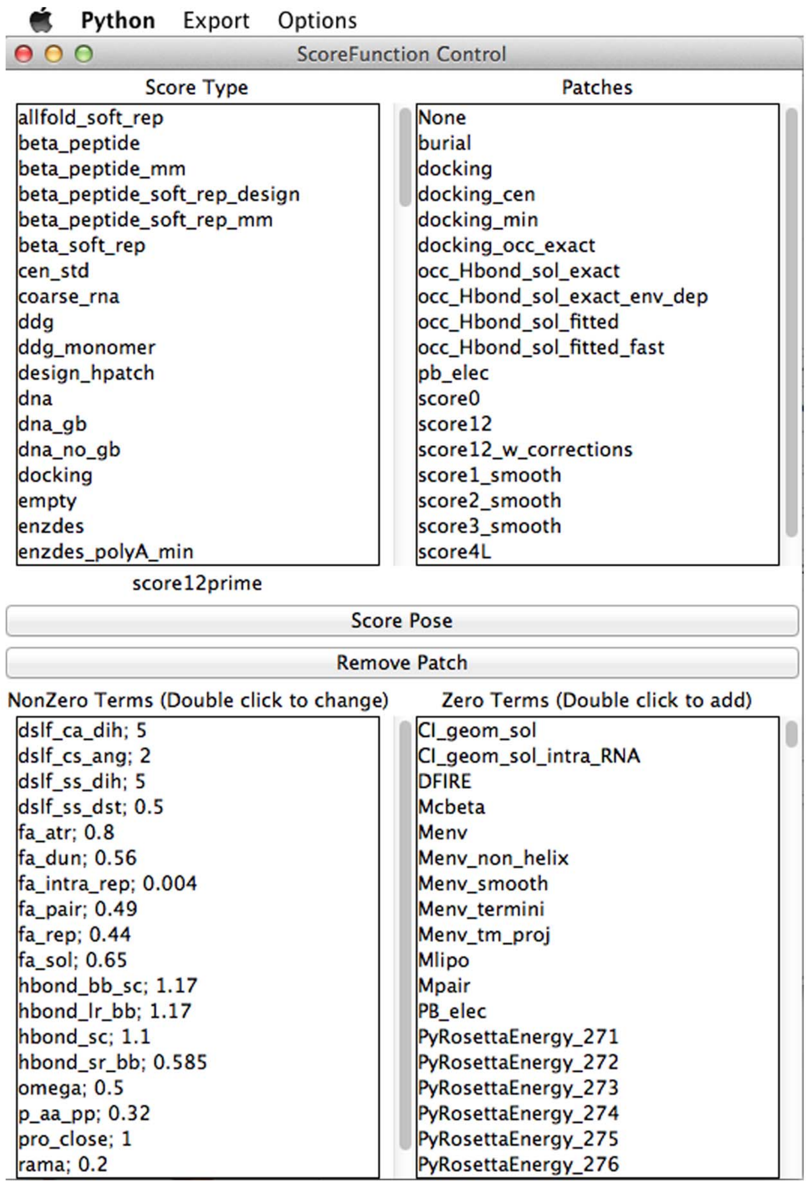
Protocols in the GUI are handled by Python's multiprocessing module, allowing the user to specify the number of processors for production runs. Many protocols have their own setup windows associated with them. Currently available protocols include fixed-backbone design [2,19,20,21], FastRelax [22], ClassicRelax [23], rotamer repacking [21,24], Floppytail [25], grafting, high- and low-resolution docking [26,27,28,29], and high- and low-resolution loop modeling with both CCD [9,28,30] and KIC [31]; with more to be added in the future. In addition, the protocols menu has links to Rosetta online servers such as ROSIE [32] and Robetta [33].

PDBLists Menu:

A PDBList is a very simple text file listing a path to a PDB file on each line. This PDBList is generally used for analysis of the large number of decoy files created in a typical Rosetta simulation. Methods for loading, creating, and using this list to analyze and group results are housed in the PDBLists Menu.

Through a function in the menu, a user can create a PDBList from all the coordinate files that match user-defined filename identifiers in a given directory or recursively. This PDBList can then be used not only for the GUI but also for command-line Rosetta runs using the in: file:l flag designation. A PDBList becomes a major point of analysis after a Rosetta protocol run when a user is left with thousands upon thousands of decoy structures, even those in multiple directories with identical PDB filenames.

Using the PDBList, a user can output a FASTA-format file containing the sequence for each PDB specified or each region specified, which can then be used by many online servers for further analysis. If a user has results from a design simulation, a design breakdown can be performed that analyzes percentages of each residue type of each region specified and outputs data into a text file, an SQLITE3 database format, and auto-generated graphs in PDF format from the R program [34]. This can be extremely useful for rational design. For energy analysis, a PDBList can be rescored or a Rosetta scorefile can be read. A user can then get the top score or best-scoring poses by number or percent. These energies are then output to a file with full paths to each decoy or structure for further analysis, and if the user wishes, the top scoring poses may be copied to a directory. In addition, an energy vs. RMSD calculation for the full protein or each region can be performed using the PDBList compared to the loaded pose.



Score Type

allfold_soft_rep
beta_peptide
beta_peptide_mm
beta_peptide_soft_rep_design
beta_peptide_soft_rep_mm
beta_soft_rep
cen_std
coarse_rna
ddg
ddg_monomer
design_hpatch
dna
dna_gb
dna_no_gb
docking
empty
enzdes
enzdes_polyA_min

score12prime

Patches

None
burial
docking
docking_cen
docking_min
docking_occ_exact
occ_Hbond_sol_exact
occ_Hbond_sol_exact_env_dep
occ_Hbond_sol_fitted
occ_Hbond_sol_fitted_fast
pb_elec
score0
score12
score12_w_corrections
score1_smooth
score2_smooth
score3_smooth
score4L

NonZero Terms (Double click to change)

dslf_ca_dih; 5
dslf_cs_ang; 2
dslf_ss_dih; 5
dslf_ss_dst; 0.5
fa_atr; 0.8
fa_dun; 0.56
fa_intra_rep; 0.004
fa_pair; 0.49
fa_rep; 0.44
fa_sol; 0.65
hbond_bb_sc; 1.17
hbond_lr_bb; 1.17
hbond_sc; 1.1
hbond_sr_bb; 0.585
omega; 0.5
p_aa_pp; 0.32
pro_close; 1
rama; 0.2

Zero Terms (Double click to add)

CI_geom_sol
CI_geom_sol_intra_RNA
DFIRE
Mcbeta
Menv
Menv_non_helix
Menv_smooth
Menv_termini
Menv_tm_proj
Mlipo
Mpair
PB_elec
PyRosettaEnergy_271
PyRosettaEnergy_272
PyRosettaEnergy_273
PyRosettaEnergy_274
PyRosettaEnergy_275
PyRosettaEnergy_276

Figure 3. Score Function Control and Creation. *Top:* All score function weights and patch files found in the Rosetta chemical database. These weights and patches can be set as the current GUI score function or saved as the default. *Bottom-Left:* Terms in the current score function and their associated weights. Weights can be changed by double-click. *Bottom-Right:* All Rosetta energy terms can be enabled at a certain weight by double-click.
doi:10.1371/journal.pone.0066856.g003

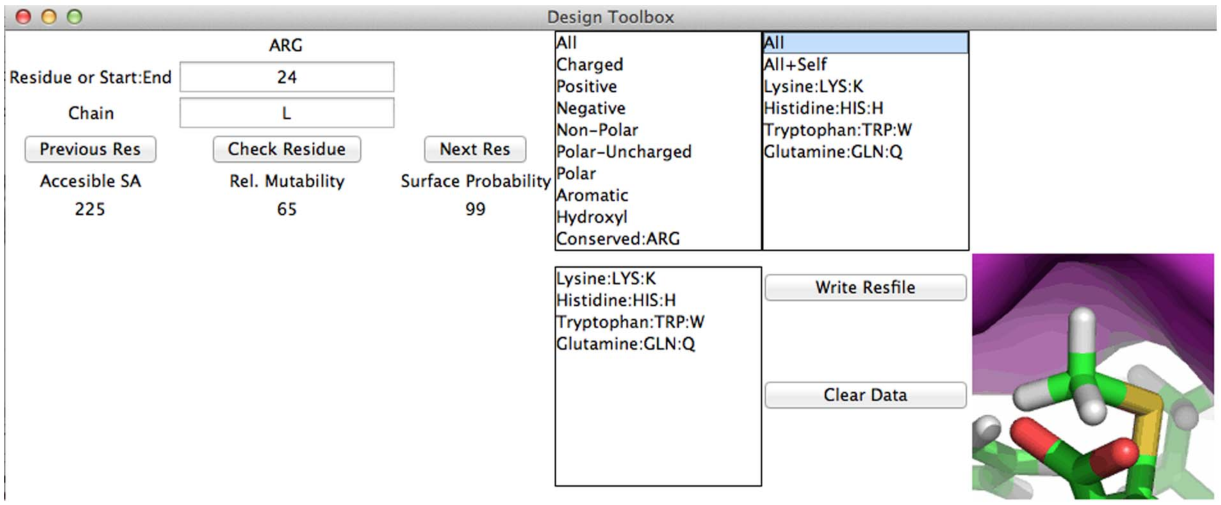
Finally, a help menu is available with links to the RosettaCommons bug tracker, the RosettaCommons user forum, general Rosetta manuals, as well as help for specific GUI tasks and setup instructions for both SCWRL and PyMOL integration.

Discussion

The PyRosetta Toolkit was designed for ease of use and modification by incorporating the simplicity of PyRosetta and Python's Tkinter Application Programming Interface (API). We hope that the community will adopt and expand the GUI as a way to interface with their own applications, classes, and scripts. Long-term future projects may involve an interface of the PyRosetta Toolkit with the Chimera Suite, and/or the creation of a native C++ GUI. The lab will continue to develop the GUI for use with the growing number of applications and tools in Rosetta, while adding functions and tools to aid in molecular modeling and design as a whole.

Methods

The GUI was written in python, using ActiveState's Komodo 6 Integrated Development Environment (IDE). The Netbeans (Oracle) and Eclipse (Eclipse Foundation) IDE's were used to explore, edit, and debug Rosetta C++ code where necessary. The open-source Clang LLVM compiler was used to compile PyRosetta binaries from the Rosetta C++ source code during development. Subversion (Apache Software Foundation) and Git (open-source) were used for version control. The GUI was tested


Design Toolbox

ARG

Residue or Start:End 24

Chain L

Accesible SA	Rel. Mutability	Surface Probability
225	65	99

All
Charged
Positive
Negative
Non-Polar
Polar-Uncharged
Polar
Aromatic
Hydroxyl
Conserved:ARG

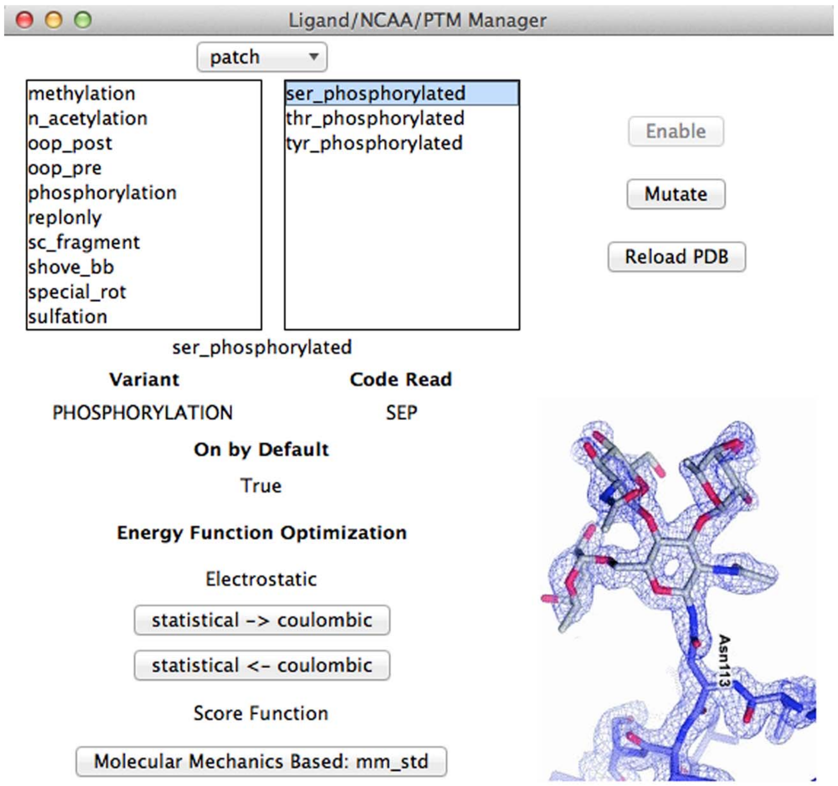
All
All+Self
Lysine:LYS:K
Histidine:HIS:H
Tryptophan:TRP:W
Glutamine:GLN:Q

Lysine:LYS:K
Histidine:HIS:H
Tryptophan:TRP:W
Glutamine:GLN:Q

Figure 4. Design – Setup Resfile. *Left*: A resfile for design is constructed for individual residues or stretches of residues. Biochemical data is given for each current residue. *Right*: Selections of residue types. Once selected, individual residues in the category are added to the box on the right. Added residues are found in the lower box, where the current designable residues can be edited or cleared.
doi:10.1371/journal.pone.0066856.g004

on Ubuntu linux release versions 10.04 LTS and 12.04 LTS (Canonical, Ltd) as well as Apple Mac OS 10.6, 10.7, and 10.8.

The PyRosetta Toolkit and RosettaFlagFileBuilder require Python 2.6 or higher and are included in both the precompiled PyRosetta binaries available at www.pyrosetta.org (/GUIs directory) and Rosetta version 3.5. Complete documentation for the GUIs can be found at http://www.pyrosetta.org/documentation and within the documentation directory each GUI. A sub project in the Rosetta Mantis Bugtracker (bugs.rosettacommons.org) is available for toolkit-specific bugs and feature requests.


Ligand/NCAA/PTM Manager

patch

methylation
n_acetylation
oop_post
oop_pre
phosphorylation
replonly
sc_fragment
shove_bb
special_rot
sulfation

ser_phosphorylated
thr_phosphorylated
tyr_phosphorylated

ser_phosphorylated

Variant	Code Read
PHOSPHORYLATION	SEP

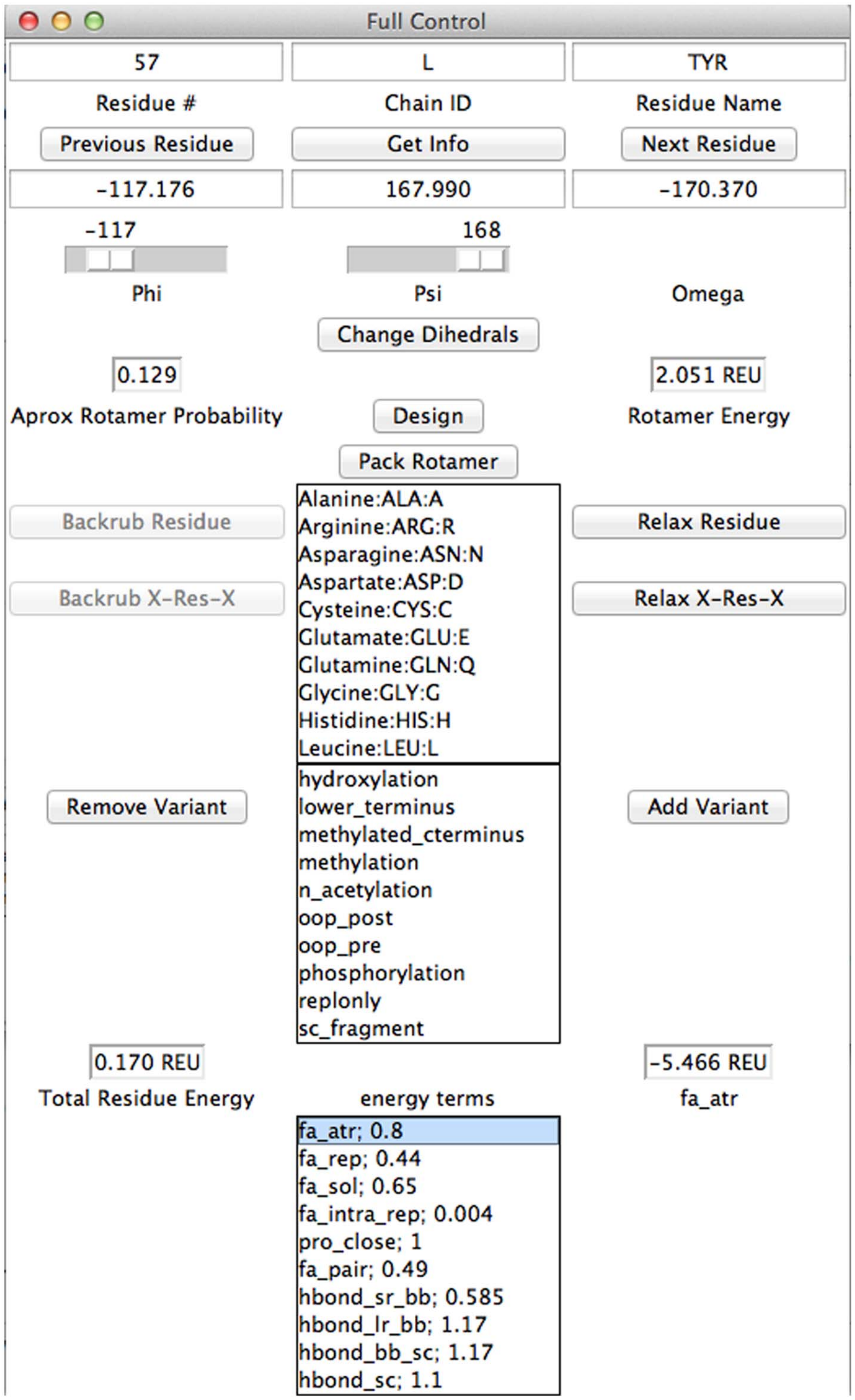
On by Default
True

Energy Function Optimization

Electrostatic

Score Function

Figure 5. Ligand/NCAA/PTM Manager. *Top*: Selection is grouped first by patch/ligand/polymer, then by specific biochemical property. *Middle*: Rosetta related information is given about the particular selection. *Bottom:* Functions for optimizing the current energy function for use with ligands, non-canonical amino acids (NCAA), and post-translational modifications (PTM).
doi:10.1371/journal.pone.0066856.g005


Full Control

57	L	TYR
Residue #	Chain ID	Residue Name
-117.176	167.990	-170.370
-117	168	
Phi	Psi	Omega
0.129		2.051 REU
Aprox Rotamer Probability		Rotamer Energy
0.170 REU		-5.466 REU
Total Residue Energy		fa_atr

```
Alanine:ALA:A
Arginine:ARG:R
Asparagine:ASN:N
Aspartate:ASP:D
Cysteine:CYS:C
Glutamate:GLU:E
Glutamine:GLN:Q
Glycine:GLY:G
Histidine:HIS:H
Leucine:LEU:L
```

```
hydroxylation
lower_terminus
methylated_cterminus
methylation
n_acetylation
oop_post
oop_pre
phosphorylation
replonly
sc_fragment
```

energy terms

```
fa_atr; 0.8
fa_rep; 0.44
fa_sol; 0.65
fa_intra_rep; 0.004
pro_close; 1
fa_pair; 0.49
hbond_sr_bb; 0.585
hbond_lr_bb; 1.17
hbond_bb_sc; 1.17
hbond_sc; 1.1
```

Figure 6. Per-Residue Control and Analysis. A collection of functions for analyzing, modeling, and designing individual residues. Per-residue energies, probabilities, and dihedral angles are given. Variants may be added or removed from residues, any residue may be designed or mutated, and individual rotamers may be optimized.
doi:10.1371/journal.pone.0066856.g006

Acknowledgments

We acknowledge helpful comments from Sergey Lyskov, Jason Labonte, Steven Lewis, Brian Weitzner, Steven Combs, Mark Andrake, Benjamin North, and the RosettaCommons community.

Author Contributions

Conceived and designed the experiments: JAB RLD. Performed the experiments: JAB. Analyzed the data: JAB. Wrote the paper: JAB RLD.

References

1. Das R, Baker D (2008) Macromolecular modeling with rosetta. Ann. Rev. Biochem. 77: 363–382.
2. Hu X, Wang H, Ke H, Kuhlman B (2007) High-resolution design of a protein loop. Proc. Natl. Acad. Sci. USA 104: 17668–17673.
3. Huang PS, Ban YE, Richter F, Andre I, Vernon R, et al. (2011) RosettaRemodel: A Generalized Framework for Flexible Backbone Protein Design. PLoS ONE 6: e24109.
4. Kaufmann KW, Lemmon GH, Deluca SL, Sheehan JH, Meiler J (2010) Practically useful: what the Rosetta protein modeling suite can do for you. Biochemistry 49: 2987–2998.
5. Leaver-Fay A, Tyka M, Lewis SM, Lange OF, Thompson J, et al. (2011) Chapter nineteen – Rosetta3: An Object-Oriented Software Suite for the Simulation and Design of Macromolecules. In: Michael LJ, Ludwig B, editors. Methods Enz.: 545–574.
6. Lewis SM, Kuhlman BA (2011) Anchored design of protein-protein interfaces. PLoS ONE 6: e20872.
7. Chaudhury S, Lyskov S, Gray JJ (2010) PyRosetta: a script-based interface for implementing molecular modeling algorithms using Rosetta. Bioinformatics 26: 689–691.
8. Eiben CB, Siegel JB, Bale JB, Cooper S, Khatib F, et al. (2012) Increased Diels-Alderase activity through backbone remodeling guided by Foldit players. Nature Biotech. 30: 190–192.
9. Parslow GR (2013) Commentary: Crowdsourcing, foldit, and scientific discovery games. Biochem. Mol. Biol. Ed. 41: 116–117.
10. Delano WL (2002) The PyMOL Molecular Graphics System.
11. Baugh EH, Lyskov S, Weitzner BD, Gray JJ (2011) Real-time PyMOL visualization for Rosetta and PyRosetta. PLoS ONE 6: e21931.
12. Krivov GG, Shapovalov MV, Dunbrack RL Jr (2009) Improved prediction of protein side-chain conformations with SCWRL4. Proteins 77: 778–795.
13. Shapovalov MV, Dunbrack RL Jr (2011) A smoothed backbone-dependent rotamer library for proteins derived from adaptive kernel density estimates and regressions. Structure 19: 844–858.
14. Carrozzini B, Cascarano GL, De Caro L, Giacovazzo C, Marchesini S, et al. (2004) Phasing diffuse scattering. Application of the SIR2002 algorithm to the non-crystallographic phase problem. Acta Cryst. A 60: 331–338.
15. Elser V (2003) Solution of the crystallographic phase problem by iterated projections. Acta Cryst. A 59: 201–209.
16. Payne PR, Borlawsky TB, Kwok A, Greaves AW (2008) Supporting the design of translational clinical studies through the generation and verification of conceptual knowledge-anchored hypotheses. AMIA Annual Symposium Proc.: 566–570.
17. Winum JY, Cecchi A, Seridi A, Scozzafava A, Montero JL, et al. (2006) Carbonic anhydrase inhibitors. N-cyanomethylsulfonamides–a new zinc binding group in the design of inhibitors targeting cytosolic and membrane-anchored isoforms. J. Enz. Inh. Med. Chem. 21: 477–481.
18. Dunbrack RL Jr, Cohen FE (1997) Bayesian statistical analysis of protein side-chain rotamer preferences. Protein Sci. 6: 1661–1681.
19. Kuhlman B, Dantas G, Ireton GC, Varani G, Stoddard BL, et al. (2003) Design of a novel globular protein fold with atomic-level accuracy. Science 302: 1364–1368.
20. Dantas G, Kuhlman B, Callender D, Wong M, Baker D (2003) A large scale test of computational protein design: folding and stability of nine completely redesigned globular proteins. J. Mol. Biol. 332: 449–460.
21. Leaver-Fay A, Kuhlman B, Snoeyink J (2005) An adaptive dynamic programming algorithm for the side chain placement problem. Pac. Symp. Biocomputing: 16–27.
22. Tyka MD, Keedy DA, Andre I, Dimaio F, Song Y, et al. (2011) Alternate states of proteins revealed by detailed energy landscape mapping. J. Mol. Biol. 405: 607–618.
23. Bradley P, Misura KM, Baker D (2005) Toward high-resolution de novo structure prediction for small proteins. Science 309: 1868–1871.
24. Kuhlman B, Baker D (2000) Native protein sequences are close to optimal for their structures. Proc. Natl. Acad. Sci. USA 97: 10383–10388.
25. Kleiger G, Saha A, Lewis S, Kuhlman B, Deshaies RJ (2009) Rapid E2-E3 assembly and disassembly enable processive ubiquitylation of cullin-RING ubiquitin ligase substrates. Cell 139: 957–968.
26. Dunbrack RL Jr, Karplus M (1994) Conformational analysis of the backbone-dependent rotamer preferences of protein sidechains. Nature Struct. Biol. 1: 334–340.
27. Wang C, Schueler-Furman O, Baker D (2005) Improved side-chain modeling for protein-protein docking. Protein Sci. 14: 1328–1339.
28. Wang C, Bradley P, Baker D (2007) Protein-protein docking with backbone flexibility. J. Mol. Biol. 373: 503–519.
29. Chaudhury S, Berrondo M, Weitzner BD, Muthu P, Bergman H, et al. (2011) Benchmarking and analysis of protein docking performance in Rosetta v3.2. PLoS ONE 6: e22477.
30. Qian B, Raman S, Das R, Bradley P, McCoy AJ, et al. (2007) High-resolution structure prediction and the crystallographic phase problem. Nature 450: 259–264.
31. Mandell DJ, Coutsias EA, Kortemme T (2009) Sub-angstrom accuracy in protein loop reconstruction by robotics-inspired conformational sampling. Nature Methods 6: 551–552.
32. Lyskov S, Chou F-C, Connor SÓ, Der BS, Drew K, et al. (2013) Serverification of Molecular Modeling Applications: the Rosetta Online Server that Includes Everyone (ROSIE). PLoS ONE 8: e63906.
33. Kim DE, Chivian D, Baker D (2004) Protein structure prediction and analysis using the Robetta server. Nucleic Acids Res. 32: W526–531.
34. R Development Team (2011) R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing.
35. Simons KT, Kooperberg C, Huang E, Baker D (1997) Assembly of protein tertiary structures from fragments with similar local sequences using simulated annealing and Bayesian scoring functions. J. Mol. Biol. 268: 209–225.
36. Lauck F, Smith CA, Friedland GF, Humphris EL, Kortemme T (2010) RosettaBackrub–a web server for flexible backbone protein structure modeling and design. Nucleic Acids Res. 38: W569–575.
37. Kortemme T, Kim DE, Baker D (2004) Computational alanine scanning of protein-protein interfaces. Science's STKE 2004: pl2.
38. Choi EJ, Jacak R, Kuhlman B (2013) A structural bioinformatics approach for identifying proteins predisposed to bind linear epitopes on pre-selected target proteins. Prot. Eng. Des. Selection 26: 283–289.