OSPREY 3: Open-Source Protein Redesign for You, Refactored, with Powerful New Features

Donald Lab

February 28, 2017

1 Abstract

2 Introduction

For over a decade, the OSPREY software package [2, 4] has offered the protein design community a unique combination of continuous flexibility modeling, ensemble modeling, and algorithms with provable guarantees. Having begun as a software release for the K^* algorithm, which approximates binding constants using ensemble modeling, it now boasts a wide array of algorithms found in no other software. OSPREY has been used in many designs that were empirically successful, *in vitro* and *in vivo* as well as in non-human primates. However, as we added more and more algorithms into OSPREY, the code became somewhat complicated and messy. Thus, we have now refactored it, to facilitate the adding of new features both by ourselves and by any others. We have also introduced a convenient Python interface and GPU support, allowing designs to be completed much more quickly and easily than in previous version of OSPREY. We believe OSPREY 3 will be a very useful tool for both developers and users of provably accurate protein design algorithms.

3 New features

3.1 LUTE: Putting advanced modeling into a form suitable for efficient, discrete design calculations

OSPREY 3 comes with LUTE [6], a new algorithm that addresses two issues with previous versions of OSPREY.

First, previous versions modeled continuous flexibility by enumerating conformations in order of a *lower bound* on minimized conformational energy [1,3]. This approach is often inefficient in that many conformations—possibly even a number exponential in the number of mutable residues—can have lower bounds below the GMEC energy, and thus will all have

to be enumerated. Only a small gain in efficiency is obtained by minimizing the energies of the partial conformations corresponding to nodes of the A* tree [5], again because of the gap between lower bounds and actual minimized energies. LUTE addresses this problem by directly optimizing the minimized energies of full conformations, which are estimated using an expansion in low-order tuples of residue conformations. Thus, the burden of modeling continuous flexibility is shifted from the combinatorial optimization (A*) step, which has unfavorable asymptotic scaling, to a precomputation step that only scales quadratically with the number of residues. This precomputation step consists of sampling a "training set" of conformations, computing their minimized energies, and then inferring the coefficients of the expansion. These coefficients can then be used as residue interaction energies in combinatorial search, whether single- or multistate. The combinatorial search will have the form of a discrete search and thus achieve high efficiency, but will accurately match the results of a continuously flexible search.

Second, all previous combinatorial protein design algorithms have relied on an explicit decomposition of the energy as a sum of local (e.g., pairwise) terms. This made design with energy functions that do not have this form difficult. For example, previous use of the Poisson-Boltzmann [9] energy function, the gold standard of implicit solvent modeling, in design has relied either on *post-hoc* reranking of a limited number of favorable designs from a calculation based on pairwise energies, which would cause all other designs favored by the Poisson-Boltzmann energetics to be missed, or on a decomposition that is incompatible with continuous flexibility [10]. However, LUTE need only calculate the energies of entire conformations in order to infer its coefficients—explicit pairwise energies are not part of this calculation. Thus LUTE can straightforwardly support general energy functions, and as shown in [6] it can obtain good fits at least in the case of Poisson-Boltzmann energies.

OSPREY users can now turn on LUTE for continuously flexible calculations simply by setting the configuration "useTupExp" to true. OSPREY 3 also supports design with Poisson-Boltzmann solvation energy calculations, which use the DelPhi [7, 8] software for the single-point Poisson-Boltzmann calculations (we ask the user to download DelPhi separately for licensing reasons). But as an algorithm, LUTE's abilities go well beyond these features—it is a general tool for taking advanced modeling of a single voxel in a system's conformation space and putting into a suitable form for efficient, discrete combinatorial optimization calculations yielding the best design sequence. As mentioned in [6], we are currently working on adding other capabilities like continuous entropy modeling this way. Moreover, any other researchers who would like to model some phenomenon in protein design, but find it difficult to fit into the usual discrete pairwise framework used in design calculations, are encouraged to try LUTE and OSPREY 3 as a framework for their modeling. Such improved modeling is essential to increasing the reliability of and range of feasible uses for computational protein design.

3.2 CATS: Local backbone flexibility in all biophysically feasible dimensions

OSPREY pioneered protein design calculations that model local continuous flexibility of sidechains in the vicinity of rotamers in all biophysically feasible dimensions (i.e., the sidechain dihedrals). This continuous flexibility was often critical in finding optimal sequences [1], and especially in eliminating artificial steric problems for ideal rotameric conformations that are chosen without consideration of protein context. In OSPREY 3, we now extend this ability to the backbone: allowing local continuous backbone flexibility in the vicinity of the native backbone in all biophysically feasible dimensions.

This flexibility is enabled by the CATS algorithm [**?**]. CATS uses a new parameterization of backbone conformational space, along with the voxel framework that OSPREY has always included. CATS is equivalent to searching over all changes in backbone dihedrals (ϕ and ψ) subject to keeping the protein conformation constant outside of a specified flexible region. This constraint is necessary to keep larger backbone dihedral changes from propagating down the backbone and unfolding the protein. CATS includes an efficient Taylor series-based algorithm for computing atomic coordinates from its new degrees of freedom, enabling efficient energy minimization. CATS is intended to be run along with OSPREY's other algorithms, yielding efficient calculations with continuous flexibility in both the sidechains and the backbone. In Ref **?**, we have shown that backbone flexibility as modeled by CATS is sometimes critical for resolving artificial steric problems and often affects energetics significantly, just as has previously been shown for continuous sidechain flexibility [1].

References

[1] Pablo Gainza, Kyle Roberts, and Bruce R. Donald. Protein design using continuous rotamers. *PLoS Computational Biology*, 8(1):e1002335, 2012.

[2] Pablo Gainza, Kyle E. Roberts, Ivelin Georgiev, Ryan H. Lilien, Daniel A. Keedy, Cheng-Yu Chen, Faisal Reza, Amy C. Anderson, David C. Richardson, Jane S. Richardson, and Bruce R. Donald. OSPREY: Protein design with ensembles, flexibility, and provable algorithms. *Methods in Enzymology*, 523:87–107, 2013.

[3] Ivelin Georgiev, Ryan H. Lilien, and Bruce R. Donald. The minimized dead-end elimination criterion and its application to protein redesign in a hybrid scoring and search algorithm for computing partition functions over molecular ensembles. *Journal of Computational Chemistry*, 29(10):1527–1542, 2008.

[4] Ivelin Georgiev, Kyle E. Roberts, Pablo Gainza, Mark A. Hallen, and Bruce R. Donald. OSPREY (Open Source Protein Redesign for You) user manual. Available online: `www.cs.duke.edu/donaldlab/software.php`. Updated, 2015. 94 pages., 2009.

[5] Mark A. Hallen, Pablo Gainza, and Bruce R. Donald. A compact representation of continuous energy surfaces for more efficient protein design. *Journal of Chemical Theory and Computation*, 11(5):2292–2306, 2015.

[6] Mark A. Hallen, Jonathan D. Jou, and Bruce R. Donald. LUTE (Local Unpruned Tuple Expansion): Accurate continuously flexible protein design with general energy functions and rigid-rotamer-like efficiency. In *International Conference on Research in Computational Molecular Biology*, pages 122–136. Springer, 2016.

[7] Anthony Nicholls and Barry Honig. A rapid finite difference algorithm, utilizing successive over-relaxation to solve the Poisson-Boltzmann equation. *Journal of Computational Chemistry*, 12(4):435–445, 1991.

[8] Walter Rochia, Sundaram Sridharan, Anthony Nicholls, Emil Alexov, Alessandro Chiabrera, and Barry Honig. Rapid grid-based construction of the molecular surface and the use of induced surface charge to calculate reaction field energies: Applications to the molecular systems and geometric objects. *Journal of Computational Chemistry*, 23(1):128–137, 2002.

[9] Doree Sitkoff, Kim A. Sharp, and Barry Honig. Accurate calculation of hydration free energies using macroscopic solvent models. *Journal of Physical Chemistry*, 98:1978–1988, 1994.

[10] Christina L. Vizcarra, Naigong Zhang, Shannon A. Marshall, Ned S. Wingreen, Chen Zeng, and Stephen L. Mayo. An improved pairwise decomposable finite-difference Poisson-Boltzmann method for computational protein design. *Journal of Computational Chemistry*, 29(7):1153–1162, 2008.