CHAPTER SIX

Scientific Benchmarks for Guiding Macromolecular Energy Function Improvement

Andrew Leaver-Fay[*,1,2], Matthew J. O'Meara[†,1,2], Mike Tyka[‡], Ron Jacak[§], Yifan Song[‡], Elizabeth H. Kellogg[‡], James Thompson[‡], Ian W. Davis[¶], Roland A. Pache[‖], Sergey Lyskov[#], Jeffrey J. Gray[#], Tanja Kortemme[‖], Jane S. Richardson[**], James J. Havranek[††], Jack Snoeyink[†], David Baker[‡], Brian Kuhlman[*]

[*]Department of Biochemistry, University of North Carolina, Chapel Hill, North Carolina, USA
[†]Department of Computer Science, University of North Carolina, Chapel Hill, North Carolina, USA
[‡]Department of Biochemistry, University of Washington, Seattle, Washington, USA
[§]Department of Immunology and Microbial Science, The Scripps Research Institute, La Jolla, California, USA
[¶]GrassRoots Biotechnology, Durham, North Carolina, USA
[‖]Department of Bioengineering and Therapeutic Science, University of California San Francisco, San Francisco, California, USA
[#]Department of Chemical & Biomolecular Engineering, Johns Hopkins, Baltimore, Maryland, USA
[**]Department of Biochemistry, Duke University, Durham, North Carolina, USA
[††]Department of Genetics, Washington University, St. Louis, Missouri, USA
[1]These authors contributed equally to this work.
[2]Corresponding authors: e-mail address: leaverfa@email.unc.edu; momeara@cs.unc.edu

Contents

Methods in Enzymology, Volume 523
ISSN 0076-6879
http://dx.doi.org/10.1016/B978-0-12-394292-0.00006-0

Abstract

Accurate energy functions are critical to macromolecular modeling and design. We describe new tools for identifying inaccuracies in energy functions and guiding their improvement, and illustrate the application of these tools to the improvement of the Rosetta energy function. The feature analysis tool identifies discrepancies between structures deposited in the PDB and low-energy structures generated by Rosetta; these likely arise from inaccuracies in the energy function. The optE tool optimizes the weights on the different components of the energy function by maximizing the recapitulation of a wide range of experimental observations. We use the tools to examine three proposed modifications to the Rosetta energy function: improving the unfolded state energy model (reference energies), using bicubic spline interpolation to generate knowledge-based torisonal potentials, and incorporating the recently developed Dunbrack 2010 rotamer library (Shapovalov & Dunbrack, 2011).

1. INTRODUCTION

Scientific benchmarks are essential for the development and parameterization of molecular modeling energy functions. Widely used molecular mechanics energy functions such as Amber and OPLS were originally parameterized with experimental and quantum chemistry data from small molecules and benchmarked against experimental observables such as intermolecular energies in the gas phase, solution phase densities, and heats of vaporization (Jorgensen, Maxwell, & Tirado-Rives, 1996; Weiner et al., 1984). More recently, thermodynamic measurements and high-resolution structures of macromolecules have provided a valuable testing ground for energy function development. Commonly used scientific tests include discriminating the ground state conformation of a macromolecule from higher energy conformations (Novotný, Bruccoleri, & Karplus, 1984; Park & Levitt, 1996; Simons et al., 1999), and predicting amino acid sidechain conformations (Bower, Cohen, & Dunbrack, 1997; Jacobson, Kaminski, Friesner, & Rapp, 2002) and free energy changes associated with protein

mutations (Gilis & Rooman, 1997; Guerois, Nielsen, & Serrano, 2002; Potapov, Cohen, & Schreiber, 2009).

Many studies have focused on optimizing an energy function for a particular problem in macromolecular modeling, for instance, the FoldX energy function was empirically parameterized for predicting changes to the free energy of a protein when it is mutated (Guerois et al., 2002). Often, these types of energy functions are well suited only to the task they have been trained for. Kellogg, Leaver-Fay, and Baker (2011) showed that an energy function explicitly trained to predict energies of mutation did not produce native-like sequences when redesigning proteins. For many projects, it is advantageous to have a single energy function that can be used for diverse modeling tasks. For example, protocols in the molecular modeling program Rosetta for ligand docking (Meiler & Baker, 2003), protein design (Kuhlman et al., 2003), and loop modeling (Wang, Bradley, & Baker, 2007) share a common energy function, which allowed Murphy, Bolduc, Gallaher, Stoddard, and Baker (2009) to combine them to shift an enzyme's substrate specificity.

Sharing a single energy function between modeling applications presents both opportunities and challenges. Researchers applying the energy function to new tasks sometimes uncover deficiencies in the energy function. The opportunities are that correcting the deficiencies in the new tasks will result in improvements in the older tasks—after all, nature uses only one energy function. Sometimes, however, modifications to the energy function that improve its performance at one task degrade its performance at others. The challenges are then to discriminate beneficial from deleterious modifications and reconcile task-specific objectives.

To address these challenges, we have developed three tools based on benchmarking Rosetta against macromolecular data. The first tool (Section 3), a suite we call "feature analysis," can be used to contrast ensembles of structural details from structures in the PDB and from structures generated by Rosetta. The second tool (Section 4), a program we call "optE," relies on fast, small-scale benchmarks to train the weights in the energy function. These two tools can help identify and fix flaws in the energy function, facilitating the process of integrating a proposed modification. We follow (Section 5) with a curated set of large-scale benchmarks meant to provide sufficient coverage of Rosetta's applications. The use of these benchmarks will provide evidence that a proposed energy function modification should be widely adopted. To conclude (Section 6), we demonstrate our tools and benchmarks by evaluating three incremental modifications to the Rosetta energy function.

Alongside this chapter, we have created an online appendix, which documents usage of the tools, input files, instructions for running the benchmarks, and current testing results: http://rosettatests.graylab.jhu.edu/guided_energy_function_improvement.

2. ENERGY FUNCTION MODEL

The Rosetta energy function is a linear combination of terms that model interactions between atoms, solvation effects, and torsion energies. More specifically, *Score12* (Rohl, Strauss, Misura, & Baker, 2004), the default fullatom energy function in Rosetta, consists of a Lennard–Jones term, an implicit solvation term (Lazaridis & Karplus, 1999), an orientation-dependent H-bond term (Kortemme, Morozov, & Baker, 2003), sidechain and backbone torsion potentials derived from the PDB, a short-ranged knowledge-based electrostatic term, and reference energies for each of the 20 amino acids that model the unfolded state. Formally, given a molecular conformation, C, the total energy of the system is given by

$$E(C|w,\Theta) = \sum_{j}^{|T|} w_j T_j\left(C|\Theta_j\right) \qquad [6.1]$$

where each energy term T_j has parameters Θ_j and weight w_j. The feature analysis tool is meant to aid the refinement of the parameters Θ. The optE tool is meant to fit the weights, w.

3. FEATURE ANALYSIS

We aim to facilitate the analysis of distributions of measurable properties of molecular conformations, which we call "feature analysis." By formalizing the analysis process, we are able to create a suite of tools and benchmarks that unify the collection, visualization, and comparison of feature distributions. After motivating our work, we describe the components (Section 3.1) and illustrate how they can be integrated into a workflow (Section 3.2) by investigating the distribution of the lengths of H-bonds with hydroxyl donors.

Feature distributions, broadly construed, have long held a prominent role in structural biochemistry. The Boltzmann equation—relating probability with energy—has been used to justify creating knowledge-based potentials from

distribution of features from crystal structures (Miyazawa & Jernigan, 1985; Sippl, 1990); for example, rotamer libraries are often based on feature distributions (Dunbrack & Karplus, 1993; Ponder & Richards, 1987). The Boltzmann equation also motivates comparing feature distributions from predicted structures against those observed in nature: for an energy function to generate geometries that are rarely observed means the energy function is wrongly assigning low energies to high-energy geometries. Many structure validation tools, such as MolProbity (Chen et al., 2010), identify outlier features as indications of errors in a structure.

The Rosetta community has also relied on feature analysis: for example, the derivation of low-resolution (Simons et al., 1999) and high-resolution (Kortemme et al., 2003) knowledge-based potentials, and the analysis of core-packing quality (Sheffler & Baker, 2009) and surface hydrophobic patches (Jacak, Leaver-Fay, & Kuhlman, 2012). However, each feature analysis foray has been *ad hoc*, limiting reuse and reproducibility. Our primary goal was to create a unified framework for feature analysis.

Feature analysis also provides a means to tune the parameters of an energy function. Recently, Song, Tyka, Leaver-Fay, Thompson, and Baker (2011) observed peaks in the backbone φ, ψ distributions of Rosetta-generated structures, absent in those of crystal structures, which they attributed to double counting in knowledge-based potentials. In one case, a commonly observed loop motif forms an H-bond between the asparagine sidechain of residue i and the backbone of residue $i+2$, constraining ψ_i to 120°. The proliferation of this motif caused an artifact in Rosetta's knowledge-based Ramachandran energy term, leading it to favor asparagines with a ψ of 120° irrespective of H-bond formation. To correct this, they considered the Ramachandran term as a parametric model and tuned the parameters until the predicted asparagine ψ distribution matched the distribution from crystal structures. Hamelryck et al. (2010) have also observed that this tuning process is a useful way to improve an energy function. Our second major goal for the feature analysis tool is to facilitate this process of parameter tuning.

3.1. Feature analysis components

The feature analysis framework consists of two components: Feature reporters take a *batch* of structures and populate a relational database with feature data. Next, feature analysis scripts select, estimate, visualize, and compare feature distributions from the database.

3.1.1 Feature databases

To facilitate the analysis of a diversity of feature distributions, we have created a relational database architecture for feature data. Typically, when analyzing feature distributions, we decompose a basic feature distribution (e.g., H-bond length) into many conditional feature distributions (e.g., H-bond length for carboxylate-guanidine residues in beta-sheets). By putting basic features into a relational database along with other supporting data, we can perform the expensive task of extracting features from some input batch of structures once, while retaining the ability to examine arbitrary conditional feature distributions in the future.

The database schema is a hierarchy with high-level tables holding the batch of structures and low-level tables holding the basic features. For example, the HBond feature reporter manages H-bond properties with foreign-key references to the higher-level `residues` and `structures` tables (Fig. 6.1).

Each feature database holds features for a single batch of conformations. Once a batch and relevant feature reporters (Table 6.1) have been selected, the features are extracted to the database using the `ReportToDB` mover in the `RosettaScripts` XML-based protocol language (Fleishman et al., 2011). Feature extraction is robust in that it supports multiple database backends (`SQLite`, `PostgreSQL`, and `MySQL`), and incremental extraction and merging. See the online appendix for feature analysis tutorials and details about implementing new `FeatureReporters`.

3.1.2 Distribution analysis

The second component of the feature analysis suite provides tools to query feature databases, to transform features in order to correctly estimate feature distributions, and to plot those distributions. Community-created feature analysis scripts are released with Rosetta and may be found in the `rosetta/rosetta_tests/features` directory.

To run a features analysis, the user provides a configuration file specifying a set of feature databases (each extracted from a batch of structures), the analysis scripts to run, and the plot output formats. Each feature analysis script typically consists of three parts: an SQL query to retrieve features from the input databases, kernel density estimation (KDE) on the extracted features (or transformed features), and the creation of a plot using the `ggplot2` grammar-of-graphics package in *R*.

Feature analysis scripts begin by querying the input sample sources using one or more SQL statements, ending with a `SELECT` statement. These SQL queries can join multiple tables to compile arbitrarily complicated conditional feature distributions. The resulting table has rows that represent

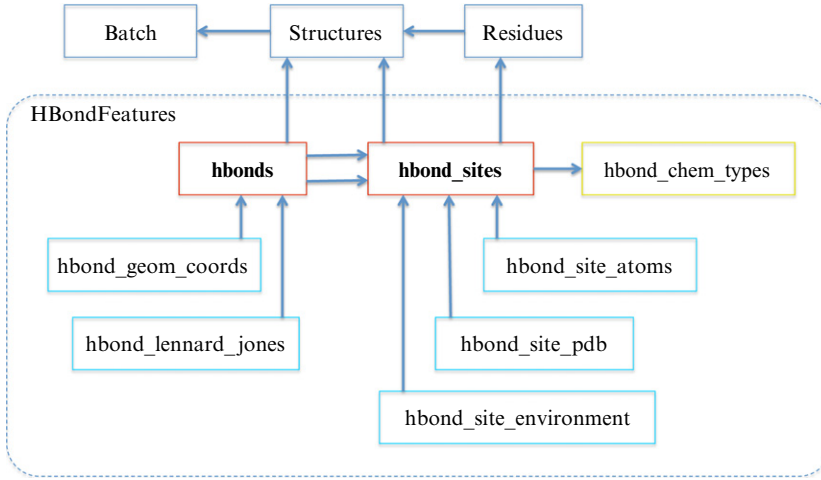

Figure 6.1 HBondFeatures database schema. The `HBondFeatures` class populates these tables with H-bond data. For each H-bond site (acceptor atom or polar hydrogen), atomic coordinates, experimental data, and solvent environment are reported. For each H-bond that forms between two H-bond sites, the geometric coordinates (i.e., distances and angles) and the sum of the Lennard–Jones energies for H-bonding atoms are reported.

feature instances, where some of the columns *identify* the feature (including which batch it came from along with other covariates), and the remaining columns *measure* the feature in the feature space.

Once the features have been retrieved, density distributions can be computed. To do this, a feature analysis script can use the split-apply-combine strategy (Wickham, 2011): feature instances are grouped by their identifying columns, and for each group, a KDE is computed over the measure columns. When computing density estimations over feature spaces, care must be taken to apply appropriate transformations to normalize the space and to handle boundary conditions. Our framework provides support for common transformations and kernel bandwith selection strategies, which control smoothness of the estimated distribution. Once a collection of feature distributions have been estimated, they can be visualized through the declaritive grammar-of-graphics method (Wickham, 2010; Wilkinson, 1999).

The scripting framework also provides support for other types of feature analysis tasks. For example, the results from prediction benchmarks (e.g., from RotamerRecovery) can be regressed against various feature types. Feature instances can also be aligned and exported to a PyMOL session for interactive inspection.

Table 6.1 Each `FeatureReporter` is responsible for extracting a particular structural feature from a structure and reporting it to a relational database

Meta	One body	Two body	Multibody
Protocol	Residue	Pair	Structure
Batch	ResidueConformation	AtomAtomPair	PoseConformation
JobData	ProteinResidueConformation	AtomInResidue–AtomInResiduePair	RadiusOfGyration
PoseComments	ProteinBackboneTorsionAngle		SecondaryStructure
	ResidueBurial	ProteinBackbone–AtomAtomPair	HydrophbicPatch
Experimental Data	ResidueSecondaryStructure		Cavity
PdbData	GeometricSolvation	HBond	GraphMotif
PdbHeaderData	BetaTurn	Orbital	SequenceMotif
DDG	RotamerBoltzmannWeight	SaltBridge	Rigidity
NMR	ResidueStrideSecondaryStructure	LoopAnchor	VoronoiPacking
DensityMap	HelixCapping	DFIREPair	InterfaceAnalysis
MultiSequenceAlignment	BondGeometry	ChargeCharge	
HomologyAlignment	ResidueLazaridisKarplusSolvation		**Energy Function**
	ResidueGeneralizedBornSolvation	**Multistructure**	ScoreFunction
Chemical	ResiduePoissonBoltzmannSolvation	ProteinRMSD	ScoreType
AtomType	Pka	ResidueRecovery	StructureScores
ResidueType	ResidueCentroids	ResiduePairRecovery	ResidueScores
		ResidueClusterRecovery	HBondParameters
		Cluster	⟨EnergyTerm⟩ Parameters

The `FeatureReporters` that are currently implemented are in black while some `FeatureReporters` that would be interesting to implement in the future are in gray.

3.2. Feature analysis workflow

Feature analysis has three common stages: sample generation, feature extraction, and distribution comparison. Feature analysis can be used to optimize the energy function parameters by iteratively modifying the energy function and comparing feature distributions from structures generated with the new energy function against those from crystal structures (Fig. 6.2).

For a demonstration, we consider the hydrogen-acceptor distance of H-bonds with nonaromatic ring hydroxyl donors (i.e., serine or theonine). In X-ray crystal structures of proteins, the most common distance for this type of H-bond is 1.65 Å, which is ~0.2 Å shorter than that for H-bonds

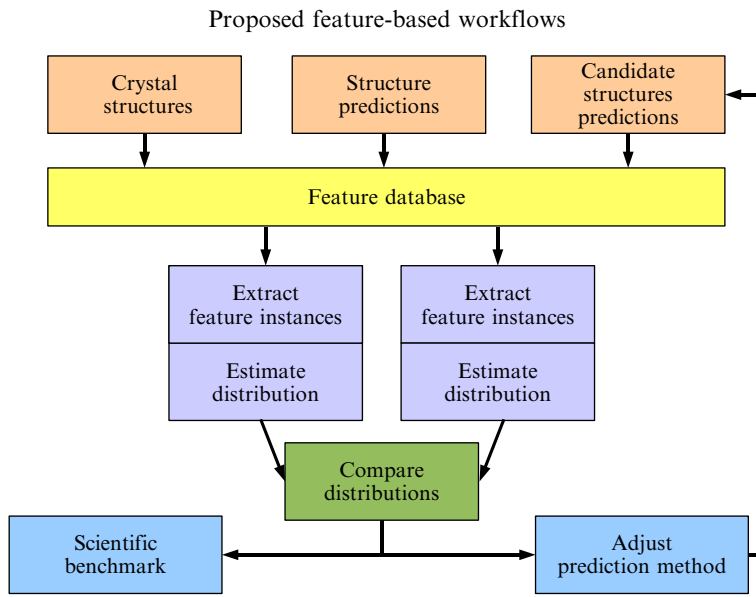

Figure 6.2 Example usage workflow for the feature analysis tool. Layer 1: Each *batch* consists of a set of molecular conformations, for example, experimentally determined or predicted conformations. Layer 2: Features from each batch are extracted into a relational database. Layer 3: Conditional feature distributions are estimated from feature instances queried from the database. Layer 4: The distributions are compared graphically. Layer 5: The comparison results are used as scientific benchmarks or to inform modifications to the structure prediction protocol and energy function, where the cycle can begin again.

involving amide donors. Rosetta does not correctly recapitulate the distance distribution for hydroxyl donors, because previously to avoid the challenge of inferring the hydroxyl hydrogen locations, the hydroxyl donor parameters were taken from the sidechain amide and carboxylamide donor parameters (Kortemme et al., 2003).

To start, we compared structures generated from Rosetta's existing energy function against native structures to verify that Rosetta does not generate the correct distribution of hydroxyl H-bond distances. We used as our reference source a subset of the `top8000` chains dataset (Keedy et al., 2012, Richardson, Keedy, & Richardson, 2013) with a maximum sequence identity of 70%, which gave 6563 chains. Hydrogen atom coordinates were optimized using Reduce (Word, Lovell, Richardson, & Richardson, 1999). We further restricted our investigation to residues with B-factors less than 30. Then, for each candidate energy function, we relaxed each protein chain with the `FastRelax` protocol in Rosetta (Khatib et al., 2011).

The analysis script has three parts: First, for each batch, the lengths of all H-bonds with serine or theonine donors to protein-backbone acceptors are extracted with an SQL query. Second, for the instances associated with each batch, a one-dimensional KDE is constructed normalizing for equal volume per unit length. Also, for each nonreference sample source, the Boltzmann distribution for the length term in the H-bond energy term is computed. Third, the density distributions are plotted. This script is available in the online appendix.

With this distribution analysis script in place, we evaluated two incremental modifications to the standard *Score12* H-bond term. The first, `NewHB`, adjusts parameters for the length term for these H-bond interactions so that the optimal length is consistent with the peak in the native distribution (panel A in Fig. 6.3). This shifts the distribution of predicted H-bonds toward shorter interactions. However, the predicted distribution does not move far enough to recapitulate the observed distribution (panels B to C in Fig. 6.3). In *Score12*, the Lennard–Jones energy term between the donor and acceptor oxygen atoms is optimal at 3 Å. However, the peak in the O—O distance distribution is at 2.6 Å (see the online appendix). Thus H-bonds with the most favorable distance according to the `NewHB` parametrization experience strong repulsion from the Lennard–Jones term. To reduce correlation between the H-bond and the Lennard–Jones energy terms, we decreased the optimal distance for the Lennard–Jones term for these specific interactions to 2.6 Å. With this second modification, Rosetta recapitulates the native distribution (panel D in Fig. 6.3).

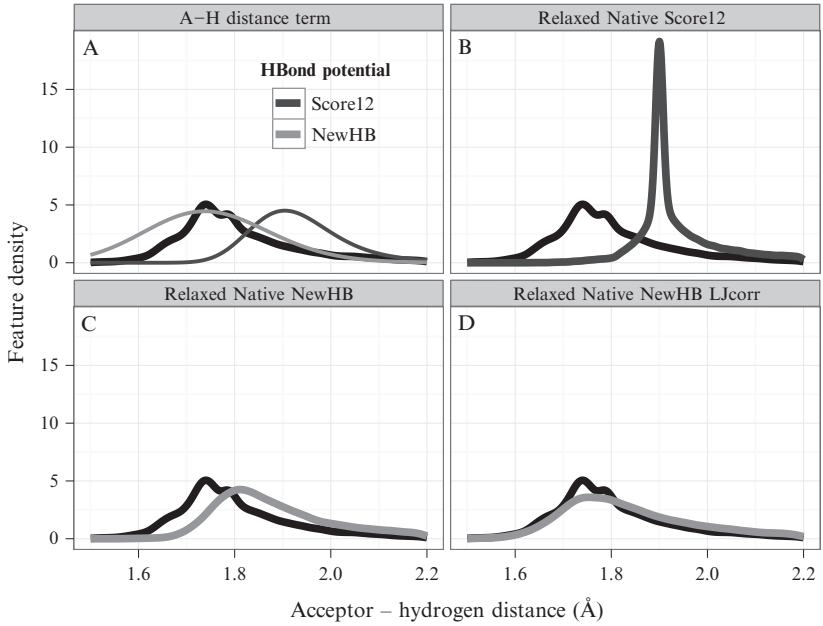

Figure 6.3 H-bond length distributions for hydroxyl donors (SER/THR) to backbone oyxgens. The thick curves are kernel density estimations from observed data normalized for equal volume per unit distance. The black curve in the background of each panel represents the Native sample source. (A) Boltzmann distribution for the length term in the Rosetta H-bond model with the *Score12* and NewHB parameterizations. (B) Relaxed Natives with the *Score12* energy function. The excessive peakiness is due to a discontinuity in the *Score12* parametrization of the H-bond model. (C) Relaxed Natives with the NewHB energy function. (D) Relaxed Natives with the NewHB energy function and the Lennard–Jones minima between the acceptor and hydroxyl heavy atoms adjusted from 3.0 to 2.6 Å, and between the acceptor and the hydrogen atoms adjusted from 1.95 to 1.75 Å.

4. MAXIMUM LIKELIHOOD PARAMETER ESTIMATION WITH optE

Recall that the Rosetta energy function is a weighted linear combination of energy terms that capture different aspects of molecular structure, as defined in Eq. (6.1). The weights, w, balance the contribution of each term to give the overall energy. Because the weights often need adjusting after modifying an energy term, we have developed a tool called "optE" to facilitate fitting them against scientific benchmarks. The benchmarks are small, tractable tests of Rosetta's ability to recapitulate experimental observations

given a particular assignment of weights. Although the weight sets that optE generates have not proven to be good replacements for the existing weights in Rosetta, we have found optE useful at two tasks: identifying problems in the Rosetta energy function and fitting the 20 amino-acid-reference-energy weights.

In the next section, we give a formula for generic likelihood-based loss functions and then describe the scientific benchmarks that are available in optE.

4.1. Loss function models

To jointly optimize an energy function's performance at the scientific benchmarks, we require success at each benchmark to be reported as a single number, which is called the loss. For scientific benchmarks based on recapitulating experimental observations, a common method of defining the loss is, given the weights, the negative log probability of predicting the observed data. If the observed data are assumed to be independently sampled, the loss is the sum of the negative log-probability over all observations. Thinking of the loss as a function of the weights for a fixed set of observations, it is called the negative log-likelihood of the weights. An ideal prediction protocol will generate predictions according to the Boltzmann distribution for the energy function.[1] Therefore, the probability of an observation o is

$$p(o|w) = \mathrm{e}^{-E(o|w)/kT} / Z(w) \quad [6.2]$$

$$Z(w) = \sum_{a \in \{A \cup o\}} \mathrm{e}^{-E(a|w)/kT} \quad [6.3]$$

where the partition function, $Z(w)$, includes o and all possible alternatives, A. Because of the vast size of conformation space, computing Z is often intractable; this is a common problem for energy-based loss functions (LeCun & Jie, 2005). To address this problem, we rely on loss functions that do not consider all possible alternatives.

4.1.1 Recovering native sequences

Within the Rosetta community, we have observed that improvements to the energy function, independently conceived to fix a particular aspect of Rosetta's behavior, have produced improvements in sequence recovery when redesigning naturally occurring proteins (Kuhlman & Baker, 2000;

[1] This assumption needs to be checked, for example, by comparing the distribution of structural features against the Boltzmann distributions defined by the energy function.

Morozov & Kortemme, 2005). Therefore, we attempt to increase sequence recovery to improve the energy function.

The standard sequence-recovery benchmark (described in Section 5.2) looks at the fraction of the amino acids recovered after performing complete protein redesign using Rosetta's Monte Carlo optimization technique. To turn this benchmark into a loss function like Eq. (6.2) would require us to compute the exact solution to the NP-Complete sidechain optimization problem (Pierce & Winfree, 2002) for the numerator, and, for an N residue protein, to repeat that computation 20^N times for each possible sequence for the denominator. This is not feasible.

Instead, we approached the benchmark as a one-at-a-time optimization, maximizing the probability of the native amino acid at a single position given some fixed context. This introduces a split between the energy function whose weights are being modified to fit the native amino acid into a particular environment and the energy function which is holding that environment together. Upweighting one term may help distinguish the native amino acid from the others, but it might also cause the rest of the environment to relax into some alternate conformation in which the native amino acid is no longer optimal. To build consistency between the two energy functions, we developed an iterative protocol (additional details given in Section 4.2.1) that oscillates between loss-function optimization and full-protein redesign. Briefly, the protocol consists of a pair of nested loops. In the outer loop, the loss function is optimized to produce a set of candidate weights. In the inner loop, the candidate weights are mixed in various proportions with the weights from the previous iteration through the outer loop, and for each set of mixed weights, complete protein redesign is performed. The redesigned structures from the last iteration through the inner loop are then used to define new loss functions for the next iteration through the outer loop.

We define the loss function for a single residue by the log-likelihood of the native amino acid defined by a Boltzmann distribution of possible amino acids at that position. We call this the p_{NatAA} loss function.

$$p_{\mathrm{NatAA}}(w) = \frac{e^{-E(\mathrm{nat}|w)/kT}}{\sum_{\mathrm{aa}} e^{-E(\mathrm{aa}|w)/kT}} \qquad [6.4]$$

$$L_{p_{\mathrm{NatRot}}}(w) = -\ln p_{\mathrm{NatAA}}(w) \qquad [6.5]$$

where $E(\mathrm{nat}\,|\,w)$ is the energy of the best rotamer for the native amino acid and $E(\mathrm{aa}\,|\,w)$ is the energy of the best rotamer for amino acid aa. Rotamers

are sampled from Roland Dunbrack's backbone-dependent rotamer library from 2002 (Dunbrack, 2002), with extra samples taken at $\pm\sigma$ for both χ_1 and χ_2. The energies for the rotamers at a particular position are computed in the context of a fixed surrounding. The contexts for the outer-loop iteration i are the designed structures from iteration $i-1$; the first round's context comes from the initial crystal structures.

4.1.2 Recovering native rotamers

As is the case for the full-fledged sequence-recovery benchmark, the rotamer-recovery benchmark (described in Section 5.1) would be intractably expressed as a generic log-likelihood loss function as given in Eq. (6.2). Instead, we again approach the recovery test as a one-at-a-time benchmark to maximize the probability of the native rotamer at a particular position when considering all other rotamer assignments.

For residue j, the probability of the native rotamer is given by

$$p_{\text{NatRot}}(w) = \frac{e^{-E(\text{nat}|w)/kT}}{\sum_{i \in \text{rots}} e^{-E(i|w)/kT}} \quad [6.6]$$

where $E(\text{nat}\,|\,w)$ is the energy of the native rotamer, and the set rots contains all other rotamers built at residue j. We define a loss function, $L\text{p}_{\text{NatRot}}$, for residue j as the negative log of this probability.

4.1.3 Decoy discrimination

Benchmarking the ability of Rosetta to correctly predict protein structures from their sequences is an incredibly expensive task. In the high-resolution refinement benchmark (described in Section 5.4), nonnative structures, *decoys*, are generated using the energy function being tested so that each decoy and each near-native structure will lie at a local minimum, but this takes ~20 K CPU hours. Within optE, we instead test the ability of Rosetta to discriminate near-native structures from decoys looking only at static structures; as optE changes the weights, the property that each structure lies at a local minimum is lost.

Given a set N of relaxed, near-native structures for a protein, and a set D of relaxed decoy structures, we approximate the probability of the native structure for that protein as

$$p_{\text{NatStruct}}(w) = \frac{1}{\sum_j n_j} \sum_{j \in N} e^{-\sigma n_j E_j(w)/kT} / Z(w) \quad [6.7]$$

$$Z(w) = \sum_{j \in D} e^{-\left(\sigma E_j(w)/kT\right) - d_j\left(R \ln \alpha - \sum_k d_k\right)} - \sum_{j \in N} e^{-\sigma E_j(w)/kT} \quad [6.8]$$

and similarly define a decoy-discrimination loss function, $L_{\text{pNatStruct}}$, as the negative log of this probability. Here, n_j is the "nativeness" of conformation j, which is 1 if j is below 1.5 Å Cα Root Mean Squared Deviation (RMSD) from the crystal structure and 0 if it is above 2 Å RMSD. The nativeness decreases linearly from 1 to 0 in the range between 1.5 and 2. Similarly, d_j is the "decoyness" of conformation j, which is 0 if j is below 4 Å RMSD and 1 otherwise. σ is a dynamic-range-normalization factor that prevents the widening of an energy gap between the natives and the decoys by scaling all of the weights; it is defined as $\sigma = \sigma(D,w)/\sigma_0$ where $\sigma(D,w)$ is the computed standard deviation for the decoy energies for a particular assignment of weights and σ_0 is the standard deviation of the decoy energies measured at the start of the simulation.

In the partition function, the $R \ln \alpha - \sum_k d_k$ term approximates the entropy of the decoys, an aspect that is otherwise neglected in a partition function that does not include all possible decoy conformations. This term attempts to add shadow decoys to the partition function, and the number of extra decoys added scales exponentially with the length of the chain, R. We chose 1.5 as the scale factor, α, which is relatively small given the number of degrees of freedom (DOFs) each residue adds. Counting torsions alone, there are between three (glycine) and seven (arginine) extra DOFs per residue. Our choice of a small α is meant to reflect the rarity of low-energy conformations. To normalize between runs which contain differing numbers of far-from-native decoys, we added the $-\sum_k d_k$ term; doubling the number of decoys between two runs should not cause the partition function to double in value.

4.1.4 ΔΔG of mutation

The full benchmark for predicting ΔΔGs of mutation (described in Section 5.3) is computationally expensive, and so, similar to the decoy-discrimination loss function, we define a $\Delta\Delta G$ loss function which relies on static structures. This loss function is given by:

$$L_{\Delta\Delta G}(w) = \left(\Delta\Delta G_{\text{exp}} - \left(\min_{\text{mut} \in \text{muts}} E(\text{mut}|w) - \min_{\text{wt} \in \text{wts}} E(\text{wt}|w)\right)\right)^2 \quad [6.9]$$

where muts is a set of structures for the mutant sequence, wts is a set of structures for the wild-type sequence, and the experimentally observed $\Delta\Delta G$ is

defined such that it is positive if the mutation is destabilizing and negative if it is stabilizing. Note that this loss function is convex if there is only one structure each in the muts and wts sets. This is very similar to the linear least-squares fitting, except that it is limited to a slope of one and a y-intercept of zero. The slope can be fit by introducing a scaling parameter to weight optimization, as described in Section 4.2.2 below.

4.2. Loss function optimization

OptE uses a combination of a particle swarm minimizer (Chen, Liu, Huang, Hwang, & Ho, 2007) and gradient-based minimization to optimize the loss function. Nonconvexity of the loss function prevents perfect optimization, and independent runs sometimes result in divergent weight sets that have similar loss function values. In spite of this problem, optE tends to converge on very similar weight sets (for an example, see the online appendix).

4.2.1 Iterative protocol

As described above, training with the p_{NatAA} loss function effectively splits the energy function into two which we attempt to merge with an iterative procedure where we oscillate between loss-function optimization in an outer loop and complete protein redesign in an inner loop. Between rounds of the outer loop, the weights fluctuate significantly, and so, in each iteration of the inner loop, we create a weight set that is a linear combination of the weights resulting from round i's loss-function optimization and the weight set selected at the end of round $i-1$. The weight set used for design during outer-loop iteration i, inner-loop iteration j is given by

$$w(i,j) = \alpha w_l(i) + (1-\alpha) w(i-1) \quad [6.10]$$

$$\alpha = \frac{1}{i+j} \quad [6.11]$$

where $w_l(i)$ is the weight set generated by minimizing the loss function in round i, and $w(i-1)$ is the final weight set from round $i-1$. In the first round, w(6.1) is simply assigned $w_l(1)$. This inner loop is exited if the sequence-recovery rate improves over the previous round or if six iterations through this loop are completed. The weight set $w(i,j)$ for the last iteration through the inner loop taken as the weight set $w(i)$ for round i and is written to disk. The set of designed structures from this iteration are taken to serve as the context for the p_{NatAA} loss function in the next round.

4.2.2 Extra capabilities

OptE provides extra capabilities useful for exploring weight space. For instance, it is possible to weigh the contributions of the various loss functions differently. Typically, we upweight the decoy-discrimination loss function by 100 when optimizing it along with the p_{NatAA} and p_{NatRot} loss functions. We typically train with ~100 native-set/decoy-set pairs, compared to several thousand residues from which we can define p_{NatAA} and p_{NatRot} loss functions. Upweighting the $p_{NatStruct}$ loss function prevents it from being drowned out.

OptE also offers the ability to fit two or more terms with the same weight, or to obey an arithmetic relationship as specified in an input text file (see the online appendix). This feature was used to scale the *Score12* terms by a linear factor but otherwise keep them fixed to optimize the $\Delta\Delta G$ of the mutation loss function (Kellogg et al., 2011). Finally, OptE allows the definition of restraints for the weights themselves to help hold them to values the user finds reasonable. This is often useful because the loss functions often prefer negative weights.

4.3. Energy function deficiencies uncovered by OptE

OptE is particularly good at two tasks: refitting reference energies (described in Section 4.5) and uncovering areas where the existing Rosetta energy function falls short. Rosetta's efficiency in searching through conformation space means it often finds decoys with lower energies than the native. Such decoys surely point to flaws in Rosetta's energy function, however, it is not always easy to see why the natives are not at lower energy than these decoys. OptE allows efficient hypothesis-driven testing: hypothesize what kind of term is absent from the Rosetta energy function, implement that term, and test that term in optE using the $p_{NatStruct}$ loss function. If the value of the loss function improves after the new term is added, that is strong evidence the term would improve the energy function. There are caveats: because the $p_{NatStruct}$ loss function relies on Rosetta-relaxed native structures, some of the features present in the crystal structures might have already been erased before optE gets started, and optE might fail to identify terms that would improve the energy function.

Using optE, we found two terms that improved the decoy-discrimination loss function over *Score12*. The first was a carbon-H-bond potential added to Rosetta (but not included as part of *Score12*) to model RNA (Das, Karanicolas, & Baker, 2010). This potential was derived from a set of protein crystal structures and a set of decoy protein structures as the log of the difference in the

probability of an oxygen being observed at a particular distance in crystal structures versus Rosetta-generated structures. OptE identified this term as strongly improving decoy discrimination. We followed this lead by splitting the potential into backbone/backbone, backbone/sidechain, and sidechain/sidechain contributions. Here, optE preferred to set the weight on the sidechain/sidechain and backbone/sidechain components to zero while keeping the weight on the backbone/backbone interactions high. This left exactly one interaction: the Hα hydrogen interacting with the carbonyl oxygen. This contact is observed principally in β-sheets and has been reported previously as giving evidence for a carbon H-bond (Fabiola, Krishnaswarmy, Nagarajan, & Pattabhi, 1997; Taylor & Kennard, 1982).

The original CH-bond potential proved to be a poor addition to the energy function: when used to generate new structures or to relax natives, previously observed deep minima at low RMSD (<1.5 Å) from the crystal structure were lost and were instead replaced by broad, flat, near-native minima that reached out as far as 4 Å. To improve the potential, Song et al. (2011) iteratively adjusted the parameters to minimize the difference between the observed and predicted structures. This iterative process resulted in better Hα—O distance distributions; unexpectedly, it also resulted in better distance distributions for other atom-pairs in β-sheets.

A second term identified by optE was a simple Coulombic electrostatic potential with a distance-dependent dielectric (Yanover & Bradley, 2011). After an initial signal from optE suggested the importance of this term, we again separated the term into backbone/backbone, backbone/sidechain, and sidechain/sidechain components. Again, the backbone/backbone portion showed the strongest signal. From there, we separated each term into attractive and repulsive components, and optE suggested that the repulsive backbone/backbone interaction contributed the most toward improved decoy discrimination. OptE also identified which low-energy decoys in the training set were problematic in the absence of the electrostatic term. By hand, we determined that the loops in these decoys contained extremely close contacts between backbone carbonyl oxygens: they were only 3.4 Å apart, which is closer than is commonly observed in crystal structures.

4.4. Limitations

Though our original goal was to fit all the weights simultaneously, optE has not proven exceptionally useful at that task. There are a number of factors that contribute to optE's failures here. For one, the loss function that we use

is not convex, and therefore its optimization cannot be guaranteed. This sometimes leads to a divergence of the weight sets in independent trajectories, which makes interpreting the results somewhat tricky.

The biggest problem facing optE, however, is its inability to see all of the conformation space. This is true for all of the loss functions we try to optimize, but the $p_{NatStruct}$ loss function, in particular, can see only the decoys we give it, and typically we cannot computationally afford to give optE more than a few hundred decoys per protein. Thus, a weight set optE generates may reflect artifacts of our decoy selection. For example, optE typically tries to assign a negative weight to the Ramachandran term (described briefly in Section 6.2), suggesting that this term is overoptimized in our decoys compared to native structures; optE finds that the easiest way to discriminate natives from decoys is turning the weight negative. In general, the weights that optE produces are not good for protein modeling (see the online appendix for benchmark results for an optE-generated weight set). However, using the protocol described in the next section, optE is fantastic at refitting reference energies.

4.5. A sequence-profile recovery protocol for fitting reference energies

Dissatisfyingly, the p_{NatAA} loss function produced weight sets that overdesigned the common amino acids (e.g., leucine) and underdesigned the rare amino acids (e.g., tryptophan). To address this shortcoming, we created an alternative protocol within optE for fitting only the amino acid reference energies, keeping all other weights fixed. This protocol does not use any of the loss functions described above; instead, it adjusts the reference energies directly based on the results of complete protein redesign.

The protocol iteratively performs complete protein redesign on a set of input protein structures and adjusts the reference energies upwards for amino acids it over designs and downwards for those it under designs, where the target frequencies are taken from the input set. After each round, optE computes both the sequence-recovery rate and the Kullback–Leibler (KL) divergence of the designed sequence profile against the observed sequence profile, given by $-\sum_{aa}\ln(p_{aa}/q_{aa})$, where p_{aa} is the naturally occurring frequency of amino acid aa in the test set and q_{aa} is the frequency of amino acid aa in the redesigned structures. The final reference energies chosen are those that maximize the sum -0.1 KL-divergence + seq. rec. rate. This protocol is used to refit reference energies after each of the three energy function changes described in Section 6.

The training set we use, which we call the "HiQ54," is a new collection of 54 nonredundant, monomeric proteins from the PDB through 2010 that have 60–200 residues (avg. = 134) and no tightly bound or large ligands. All were required to have both resolution and MolProbity score (Chen et al., 2010) at or below 1.4, very few bond length or angle outliers, and deposited structure-factor data. The HiQ54 set is available in the online appendix and at http://kinemage.biochem.duke.edu/.

5. LARGE-SCALE BENCHMARKS

Scientific benchmarking allows energy function comparison. The tests most pertinent to the Rosetta community often aim toward recapitulating observations from crystal structures. In this section, we describe a curated set of previously published benchmarks, which together provide a comprehensive view of an energy function's strengths and weaknesses. We continually test the benchmarks on the RosettaTests server to allow us to immediately detect changes to Rosetta that degrades its overall performance (Lyskov & Gray, 2012).

5.1. Rotamer recovery

One of the most direct tests for an energy function is its ability to correctly identify the observed rotamers in a crystal structure against all other possible rotamers while keeping the backbone fixed. Variants of the rotamer recovery test have long been used to evaluate molecular structure energy functions (Liang & Grishin, 2002; Petrella, Lazaridis, & Karplus, 1998), including extensive use to evaluate the Rosetta energy function (Dantas et al., 2007; Dobson, Dantas, Baker, & Varani, 2006; Jacak et al., 2012; Kortemme et al., 2003), the ORBIT energy function (Sharabi, Yanover, Dekel, & Shifman, 2010), and the SCWRL energy function (Shapovalov & Dunbrack, 2011).

Here, we test rotamer recovery in four tests combining two bifurcated approaches to the task: discrete versus continuous rotamer sampling and one-at-a-time versus full-protein rotamer optimization. The discrete, one-at-a-time rotamer optimization protocol is called *rotamer trials*. It builds rotamers, calculates their energies in the native context, and compares the lowest-energy rotamer against the observed crystal rotamer. The continuous, one-at-a-time rotamer optimization protocol is called *rt-min* (Wang, Schueler-Furman, & Baker, 2005). It similarly builds rotamers in the native context but minimizes each rotamer before comparing the lowest-energy rotamer against the crystal rotamer. The discrete, full-protein optimization protocol is called *pack rotamers*. This builds rotamers for all positions and then

seeks to find the lowest-energy assignment of rotamers to the structure using a Monte Carlo with simulated annealing protocol (Kuhlman & Baker, 2000) where at a random position, a random rotamer is substituted into the current environment, its energy is calculated, and the substitution is accepted or rejected based on the Boltzmann criterion. The rotamers in the final assignment are compared against the crystal rotamers. The continuous, full-protein rotamer optimization task is called *min pack*. It is similar to the pack rotamers protocol except that each rotamer is minimized before the Boltzmann decision. A similar protocol has been described before (Ding & Dokholyan, 2006).

Recovery rates are measured on a set of 152 structures, each having between 50 and 200 residues and a resolution less than 1.2 Å. Rotamers are considered recovered if all their χ dihedrals are less than 20° from the crystal χ dihedrals, taking into account symmetry for the terminal dihedrals in PHE, TYR, ASP, and GLU. For the discrete optimization tests, rotamers are built at the center of the rotamer wells, with extra samples included at $\pm\sigma_i$ from $\bar{\chi}_i$ for χ_1 and χ_2. For the continuous optimization test, samples are only taken at $\bar{\chi}_i$ but can move away from the starting conformation through minimization.

5.2. Sequence recovery

In the sequence-recovery benchmark, we perform complete-protein fixed-backbone redesigns on a set of crystal structures, looking to recapitulate the native amino acid at each position. For this chapter, we used the test set of 38 large proteins from Ding and Dokholyan (2006). Sequence recovery was performed with the discrete, full-protein rotamer-and-sequence optimization protocol called `PackRotamers`, described above. Rotamer samples were taken from the given rotamer library (either the 2002 or the 2010 library), and extra samples were chosen at $\pm\sigma_i$ for χ_1 and χ_2. The multi-cool annealer-simulated annealing protocol (Leaver-Fay, Jacak, Stranges, & Kuhlman, 2011) was employed instead of Rosetta's standard simulated annealing protocol. We measured the sequence-recovery rate and the KL-divergence of the designed amino acid profile from the native amino acid profile: the sequence-recovery rate should be high, and the KL-divergence should be low.

5.3. $\Delta\Delta G$ prediction

The $\Delta\Delta G$ benchmark consists of running the high-resolution protocol described in Kellogg et al. (2011), on a curated set of 1210 point mutations for which crystal structures of the wild-type protein are available. The

protocol predicts a change in stability induced by the mutation by comparing the Rosetta energy for the wild-type and mutant sequences after applying the same relaxation protocol to each of them. The Pearson correlation coefficient of the measured versus predicted $\Delta\Delta G$s is used to assess Rosetta's performance.

5.4. High-resolution protein refinement

Rosetta's protocol to predict protein structures from their sequence alone runs in two phases: a low-resolution phase (*ab initio*) relying on fragment insertion (Simons, Kooperberg, Huang, & Baker, 1997) to search through a relatively smooth energy landscape (Simons et al., 1999), and a high-resolution phase (*relax*) employing sidechain optimization and gradient-based minimization. This *abrelax* protocol offers the greatest ability to broadly sample conformation space in an attempt to find nonnative conformations that Rosetta prefers to near-native conformations.

Unfortunately, the *abrelax* protocol requires significantly more sampling than could be readily performed to benchmark a change to the energy function. The problem is that most low-resolution structures produced by the first *ab initio* phase do not yield low-energy structures in the second *relax* phase, so finding low-energy decoy conformations requires hundreds of thousands of trajectories. To reduce the required amount of sampling, Tyka et al. (2010) curated four sets of low-resolution decoys for 114 proteins by taking the low-resolution structures that generated low-energy structures after being put through high-resolution refinement. The benchmark is then to perform high-resolution refinement on these low-resolution structures with the new energy function. Each of the low-resolution sets was generated from different sets of fragments; some sets included fragments from homologs, while others included fragments from the crystal structure of the protein itself. This gave a spectrum of structures at varying distances from native structures.

These sets are used as input to the relax protocol. The decoy energies and their RMSDs from the crystal structure are used to assess the ability of Rosetta to discriminate natives from low-energy decoys. This is reported in two metrics by the benchmark: the number of proteins for which the probability of the native structure given by the Boltzmann distribution exceeds 80% ($\mathrm{pNat} = \exp(E(\mathrm{nat}))/(\exp(E(\mathrm{nat})) + \Sigma\exp(E(d)))$ with $E(\mathrm{nat})$ representing the best energy of any structure under 2 Å RMSD from

the crystal structure, and *d* representing any structure greater than 2 Å RMSD. pNat should not be confused with the $p_{NatStruct}$ loss function in optE, and the number of proteins for which the lowest-energy near-native conformation has a lower energy than the lowest-energy decoy conformation. For the benchmarks reported here, we relaxed 6000 decoys randomly sampled with replacement from each of the four sets, resulting in 24,000 decoys per protein. The statistical significance of the difference between two runs can be assessed with a paired *t*-test, comparing pNat for each of the 114 targets.

The proteins included in this test set include many where the crystal structure of the native includes artifacts (e.g., loop rearrangements to form crystal contacts), where the protein coordinates metal ions or ligands, or where the protein forms an obligate multimer in solution. For this reason, perfect performance at this benchmark is not expected, and interpreting the results is somewhat complicated; an improvement in the benchmark is easier to interpret than a degradation.

5.5. Loop prediction

The loop-prediction benchmark aims to test Rosetta's accuracy at *de novo* protein loop reconstruction. For this, we used the kinematic closure (KIC) protocol, which samples mechanically accessible conformations of a given loop by analytically determining six dihedral angles while sampling the remaining loop torsions probabilistically from Ramachandran space (Mandell, Coutsias, & Kortemme, 2009).

We used the benchmark set of 45 12-residue loops as described in Mandell et al. (2009). For each loop, we generated 8000 structures and calculated their Cα loop RMSD to the native.

In some cases, KIC generates multiple clusters of low-energy conformations of which one cluster is close to the native structure, while the other(s) can be several Ångstroms away. Because the Rosetta energy function does not robustly distinguish between these multiple clusters, we considered not just a single structure, but the five lowest-energy structures produced. Of these five, we used the lowest-RMSD structure when calculating benchmark performance. Overall loop reconstruction accuracy is taken as the median Cα loop RMSD of all best structures across the entire 45-loop dataset. The first and third quartile, though of lesser importance than the median, should be examined as well, as they offer a picture of the rest of the distribution.

6. THREE PROPOSED CHANGES TO THE ROSETTA ENERGY FUNCTION

In this final section, we describe three changes to Rosetta's energy function. After describing each change and its rationale, we present the results of the benchmarks described above.

6.1. Score12′

We used the sequence-profile-recovery protocol in optE to fit the reference energies for *Score12* to generate a new energy function that we called *Score12′*. Refitting the *Score12* reference energies in Rosetta3 (Leaver-Fay, Tyka, et al., 2011) was necessary because the components of *Score12* in Rosetta3 differed in several ways from those in Rosetta2. First, in Rosetta2, there was a disagreement between the energy function used during sequence optimization and the one used throughout the rest of Rosetta. In the sequence optimization module ("the packer"), the knowledge-based Ramachandran potential was disabled, and the weight on the $P(\mathrm{aa}\,|\,\varphi, \psi)$ was doubled from 0.32 to 0.64. In Rosetta3, there is no schism between the energy functions used in the packer and elsewhere. Second, the Lennard–Jones and implicit solvation (Lazaridis & Karplus, 1999) terms were extended from 5.5 to 6 Å and spline-smoothed to be zero and flat at 6 Å. Previously, they ramped linearly from 5 to 5.5 Å, causing discontinuities in the derivatives, which—when combined with gradient-based minimization—created peaks in atom-pair radial distributions (W. Sheffler & D. Baker, unpublished observations). Also, the Lennard–Jones potential now starts counting the contribution of the repulsive component at the bottom of the well, instead of at the x-intercept. This change eliminates a derivative discontinuity at the x-intercept that forms if the attractive and repulsive weights differ (as they do in *Score12*). Third, in Rosetta2, interactions were inappropriately omitted because for certain types of residue pairs, the Cβ–Cβ interaction threshold was too short.

Table 6.2 gives the sequence-recovery rates for Rosetta2 and Rosetta3 using *Score12* and using reference weights trained with the p_{NatAA} loss function (Rosetta3-p_{NatAA}) and with the sequence-profile recovery protocol (Rosetta3-sc12′). Though the p_{NatAA} objective function achieves satisfactory sequence-recovery rates, it overdesigns leucine and lysine and never designs tryptophan. *Score12′*, on the other hand, does an excellent job recapitulating the sequence profile in the testing set while also outperforming the Rosetta3-

Table 6.2 Sequence recovery rates (% Rec.), KL-divergence of the designed sequence profiles from the native sequence profile (KL-div.), and amino acid profiles measured on the Ding & Dokholyan-38 set

	% Rec.	KL-div.	A	C	D	E	F	G	H	I	K	L	M	N	P	Q	G	S	T	V	W	Y
Test set	–	–	8.9	1.2	6.6	6.7	4.3	8.3	2.2	5.2	6.3	7.9	2.4	4.5	4.5	3.4	4.7	5.2	5.7	6.8	1.5	3.6
Rosetta2-sc12	38.0	0.019	6.3	1.2	6.2	6.3	5.5	7.6	1.9	6.3	5.2	10.3	1.9	4.1	6.0	4.2	5.3	5.2	5.0	5.6	2.0	3.9
Rosetta3-sc12	32.6	0.141	6.4	0.0	7.7	8.7	5.4	7.4	6.3	4.7	6.8	8.8	1.7	3.2	1.3	3.5	6.3	7.2	4.2	4.2	2.5	4.0
Rosetta3-p_{NatAA}	36.7	0.391	9.8	0.0	6.4	6.2	0.7	8.3	0.8	5.7	11.8	13.5	0.2	2.7	7.6	0.2	3.1	4.7	6.3	10.6	0.0	1.5
Rosetta3-sc12′	37.0	0.008	8.4	0.7	6.6	7.5	3.9	8.2	2.1	5.0	6.4	8.0	2.2	3.5	3.7	3.3	6.1	6.4	5.8	6.4	1.8	4.1

The Rosetta3-sc12 energy function represents the *Score12* reference energies taken directly from Rosetta2. The Rosetta3-p_{NatAA} weight set keeps the same weights as *Score12*, except that the reference energies were fit by optimizing the p_{NatAA} loss function; the Rosetta3-sc12′ (*Score12′*) reference energies were generated using the sequence-profile optimization protocol.

p_{NatAA} energy function at sequence recovery. The rotamer-recovery, high-resolution refinement, and loop-prediction benchmarks were not run for this proposed change to the energy function as fixed-sequence tasks are unaffected by changes to the reference energies.

6.2. Interpolating knowledge-based potentials with bicubic splines

Three knowledge-based potentials in Rosetta are defined on the ϕ,ψ map: the Ramachandran term which gives $E_{\text{rama}}(\phi,\psi\,|\,\text{aa}) = -\ln p(\phi,\psi\,|\,\text{aa})$, the `p_aa_pp` term (sometimes called the design term) which gives an energy from the log of the probability of observing a particular amino acid at a given ϕ,ψ, and the rotamer term (almost always called the Dunbrack term, or `fa_dun`), which gives $E_{\text{dun}}(\chi\,|\,\phi,\psi,\text{aa})$. These terms use bins on the ϕ,ψ map to collect the data that define these potentials and use bilinear interpolation between the bins to define a continuous function.

The `p_aa_pp` term in *Score12* is given by

$$E_{\text{paapp}}(\text{aa}|\phi,\psi) = -\ln\frac{p(\text{aa}|\phi,\psi)}{p(\text{aa})}. \qquad [6.12]$$

For any particular ϕ and ψ, the energy is given as the negative log of the bilinearly interpolated $p(\text{aa}\,|\,\phi,\psi)$ divided by $p(\text{aa})$. The Ramachandran term similarly defines the energy for the off-grid-point ϕ and ψ values as the negative log of the bilinearly interpolated $p(\phi,\psi\,|\,\text{aa})$. Both the `p_aa_pp` and the Ramachandran term place their bin centers every 10° starting from 5°.

The Dunbrack term in *Score12* is given by

$$E_{\text{dun}}(\chi|\phi,\psi,\text{aa}) = -\ln(p(\text{rot}|\phi,\psi,\text{aa})) + \sum_i \left(\frac{\chi_i - \bar{\chi}_i(\phi,\psi|\text{aa,rot})}{\sigma_i(\phi,\psi|\text{aa,rot})}\right)^2 \qquad [6.13]$$

where the rotamer bin, rot, which gives the probability of the rotamer, $p(\text{rot}\,|\,\phi,\psi,\text{aa})$, is computed from the assigned χ dihedrals, and both $\bar{\chi}_i(\phi,\psi|\text{aa,rot})$ and $\sigma_i(\phi,\psi\,|\,\text{aa,rot})$ are the measured mean χ values and standard deviations for the rotamer. This effectively models the probability for the sidechain conformation as the product of the rotamer probability and several (height-unnormalized) Gaussians. The 2002 library gives the $p(\text{rot}\,|\,\phi,\psi,\text{aa})$, $\bar{\chi}_i(\phi,\psi|\text{aa,rot})$, and $\sigma_i(\phi,\psi\,|\,\text{aa,rot})$ every 10°. Given a particular assignment of ϕ and ψ, the values for $p(\text{rot}\,|\,\phi,\psi,\text{aa})$, $\bar{\chi}_i(\phi,\psi|\text{aa,rot})$, and $\sigma_i(\phi,\psi\,|\,\text{aa},$

rot) are bilinearly interpolated from the four surrounding bin centers. The Dunbrack term divides the ϕ,ψ plane into 10° bins, starting from 0°.

Bilinear interpolation leaves derivative discontinuities every 5° in the ϕ,ψ plane. These discontinuities frustrate the minimizer causing pileups at the bin boundaries. Looking at the ϕ,ψ distribution for nonhelical residues, the grid boundaries are unmistakable (Fig. 6.4B). Indeed, *Score12* predicts 23% of all ϕ,ψ pairs of lying within 0.05° of a grid boundary.

We propose to fix this problem by using bicubic splines to interpolate between the grid points. We fit bicubic splines with periodic boundary conditions for both the Ramachandran and `p_aa_pp` terms on the energies, interpolating in energy space. For the Dunbrack energy, we fit bicubic splines for the $-\ln(p(\text{rot}\,|\,\phi,\psi,\text{aa}))$ portion, but, to avoid increasing our memory footprint too much, continued to use bilinear interpolation for the χ-mean and χ-standard deviations. We refit the reference energies using the

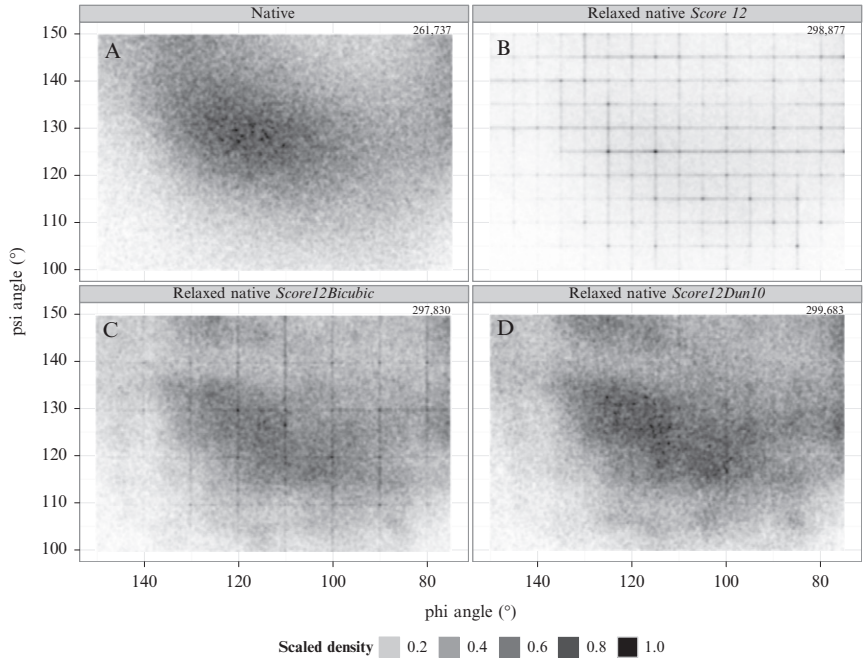

Figure 6.4 Backbone torsion angles in the beta-region with B-factors less than 30. (A) The distribution for the top 8000; counts in upper right. (B) In *Score12*, density accumulates on the 5° bins due to derivative discontinuities caused by bilinear interpolation. (C) *Score12Bicubic* has only a few remaining artifacts on the 10° bin boundaries due to the continued use of bilinear interpolation for parts of the Dunbrack energy. (D) Score12Dun10 has very few remaining artifacts.

sequence-profile recovery protocol to create a new energy function that, for this chapter, we refer to as *Score12Bicubic*; Fig. 6.4C shows that bicubic-spline interpolation dramatically reduces the pileups. Accumulation on the 10°-grid boundaries starting at 5° produced by the Ramachandran and `p_aa_pp` terms is completely gone. Modest accumulation on the 10°-grid boundaries starting at 0° persists because bicubic splines were not used to interpolate the χ-mean and χ-standard deviations.

6.3. Replacing the 2002 rotamer library with the extended 2010 rotamer library

In 2010, Shapovalov and Dunbrack (Shapovalov & Dunbrack, 2011) defined a new rotamer library that differs significantly from the 2002 library in the way the terminal χ is handled for eight amino acids: ASP, ASN, GLU, GLN, HIS, PHE, TYR, and TRP. Because these terminal χ dihedrals are about bonds between sp3-hybridized atoms and sp2-hybridized atoms, there are no well-defined staggered conformations. Instead of modeling the probability landscape for the terminal χ within a particular bin as a Guassian, the new library instead provides a continuous probability distribution over all the bins. This last χ is in effect nonrotameric, though the rest of the χ in the sidechain are still rotameric; these eight amino acids can be called *semi-rotameric*. In the 2002 library, there were discontinuities in both the energy function and the derivatives when crossing over these grid boundaries. Once a rotamer boundary is crossed, an entirely different set of $\bar{\chi}$ and σ_i are used to evaluate the rotamer energy. Further, Rosetta's treatment of the terminal χ probability distributions as Gaussians means that Rosetta structures display Gaussian distributions (the gray lines in Fig. 6.5) that do not resemble the native distributions (the black lines in Fig. 6.5).

With the 2010 library, the energy for a rotameric residue is computed in the same way as for the 2002 library, and the energy for one of the semi-rotameric amino acids is computed as

$$E_{\text{dun}}(\chi|\phi,\psi,\text{aa}) = -\ln(p(\text{rot}|\phi,\psi,\text{aa})p(\chi_T|\text{rot},\phi,\psi,\text{aa}))\sum_{i<T}\left(\frac{\chi_i-\bar{\chi}_i}{\sigma_i}\right)^2 \quad [6.14]$$

where T denotes the terminal χ, and where $\bar{\chi}_i(\phi,\psi|\text{aa},\text{rot})$ and $\sigma_i(\phi,\psi\,|\,\text{aa},\text{rot})$ from Eq. (6.13) have been abbreviated as $\bar{\chi}_i$ and σ_i, though they retain their dependence on the rotamer and amino acid and are a function of ϕ and ψ. The 2010 library provides data for $p(\chi_T|\,\text{rot},\phi,\psi,\text{aa})$ every 10° for ϕ and ψ, and

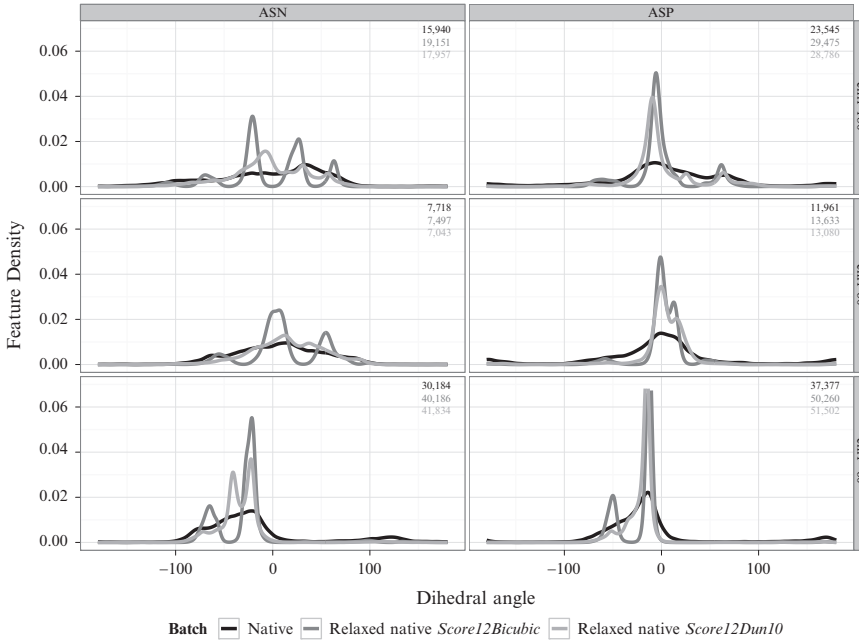

Figure 6.5 ASN/ASP χ_2 distribution by χ_1 bin. Comparison of χ_2 distributions for the semirotameric amino acids ASN and ASP, broken down by χ_1 rotamer; counts in upper right. Rosetta's implementation of the 2002 library produces Gaussian-like distributions for χ_2 in relaxed natives (gray), though the native distributions do not resemble Gaussians (black). Using the 2010 library (light gray), the distributions improve considerably though they remain too peaky in places.

every 5° or every 10° for χ_T depending on whether the amino acid, aa, is symmetric about χ_T. We fit tricubic splines to interpolate $-\ln(p(\text{rot}\,|\,\phi,\psi,\text{aa})\ p(\chi_T|\,\text{rot},\phi,\psi,\text{aa}))$ in ϕ,ψ and χ_T. As in the 2002 library, $\bar{\chi}_i$ and σ_i are interpolated bilinearly from the four surrounding grid points. We refit the reference energies using the sequence-profile recovery protocol to create a new energy function that, for this chapter, we refer to as *Score12Dun10*. *Score12Dun10* builds on top of *Score12Bicubic*.

6.4. Benchmark results

The results of the benchmarks, given in Table 6.3, show that *Score12′* is a clear improvement over *Score12*, substantially improving sequence recovery, and that *Score12Bicubic* is a clear improvement over *Score12′*, behaving as well as *Score12′* on most benchmarks and giving a slight, but statistically insignificant improvement at the high-resolution refinement benchmark ($p = 0.07$).

Score12Dun10 shows mixed results: at the rotamer-recovery benchmarks, it shows a clear improvement over *Score12Bicubic*; the improvement can be most clearly seen in the rotamer-recovery rates for the semirotameric amino acids (Table 6.4).

Score12Dun10 performed worse at the high-resolution refinement benchmark than *Score12Bicubic*. The two principal metrics for this benchmark are slightly worse: the number of proteins where the lowest energy near-native structure has a lower energy than the lowest energy decoy (#(eNat < eDec); 104 vs. 105), and the number of proteins where the probability of the native structure calculated by the Boltzmann distribution is greater than 80% (#(pNat > 0.8); 67 vs. 60). To estimate the significance of these results, we compared the distribution of ($pNat_{sc12Dun10}$–$pNat_{sc12Bicubic}$) for each of the 114 targets against the null hypothesis that this distribution had a mean of 0, using a two-tailed *t*-test. This gave a *p*-value for the difference of 0.01. Because this benchmark includes proteins whose accurate prediction is unlikely given protocol limitations (disulfides are not predicted), and crystal artifacts (loops which adopt conformations supported only by crystal contacts), its results are more difficult to interpret. We therefore restricted our focus to 29 proteins in the set which are absent of these issues (these are listed in the online appendix) and repeated the comparison of ($pNat_{sc12Dun10}$–$pNat_{sc12Bicubic}$). Here too, *Score12Dun10* showed a statistically significant degradation relative to *Score12Bicubic*, with a *p*-value of 0.04. The mean difference of the pNat statistic for this subset (0.06) was similar to the mean difference over the entire set (0.05).

For the $\Delta\Delta G$ benchmark, the differences between the four tested energy functions were very slight. *Score12′*'s performance was somewhat degraded relative to *Score12*, though this should be weighed against the dramatic improvement *Score12′* showed at sequence recovery. The other two energy functions performed in the same range as *Score12′*.

At the loop-modeling benchmark, the differences between the three methods were slight. To estimate the significance of the differences, we took 100 bootstrap samples and measured the three RMSD quartiles. The differences in median RMSDs between *Score12* and *Score12Bicubic* ($p < 0.74$), and *Score12Bicubic* and *Score12Dun10* ($p < 0.11$) were not statistically significant. However, the third quartile improved for both *Score12Bicubic* and *Score12Dun10*. In two cases (see the online appendix), KIC simulations using *Score12Dun10* correctly identified near-native structures by their lowest energy, whereas the *Score12* simulations did not.

Table 6.3 Benchmark results for score12 and for the three proposed energy function modifications

	Rotamer recovery benchmark				Seq. rec. bench		$\Delta\Delta G$ bench	High-res. refinement benchmark			Loop-modeling benchmark		
Energy function	Pack rots (%)	Min pack (%)	Rot. trials (%)	Rt-min (%)	% Rec.	KL-div.	*R*-value	# (pNat > 0.8)	ΣpNat	# (eNat < eDec)	First quart. (Å)	Med. (Å)	Third quart. (Å)
Score12	66.19	69.07	71.49	73.12	32.6	0.019	0.69	67	74.6	104	0.468	0.637	1.839
Score12′	–	–	–	–	37.0	0.008	0.67	–	–		–	–	–
Score12Bicubic	66.24	67.51	71.52	73.15	37.6	0.010	0.68	68	77.9	105	0.499	0.644	1.636
Score12Dun10	67.82	70.50	72.60	74.23	37.6	0.009	0.67	60	72.0	104	0.461	0.677	1.463

Score12′ differs from *Score12* only in its reference energies, which have no effect on rotamer-recovery, high-resolution refinement, or loop modeling, and so data for these benchmarks are not given.

Table 6.4 Percentage rotamer recovery by amino acid

	R	K	M	I	L	S	T	V	N	D	Q	E	H	W	F	Y
Score12	24.7	31.1	51.3	84.0	86.7	71.8	92.9	94.4	55.2	59.2	21.8	28.5	51.8	78.7	84.5	79.9
Score12Bicubic	25.7	31.8	51.1	85.2	86.8	71.5	92.9	94.4	54.8	58.7	20.5	28.7	52.0	80.1	83.3	79.9
Score12Dun10	26.7	31.7	49.6	85.4	87.5	72.5	92.6	94.3	56.8	60.4	30.7	33.6	55.0	85.0	85.4	82.9

Rotameric amino acids are listed on the left; semirotameric amino acids on the right.

It is not immediately clear why the 2010 rotamer library causes a degradation in Rosetta's ability to discriminate native structures from decoys. A possible reason for this might be pointed to in the distribution of off-center χ angles. Structures refined with the new library have χ-angles more tightly distributed around the reported $\bar{\chi}_i$. The 2010 library, which was generated using more stringent data filters than the 2002 library, reports smaller standard deviations on average: for example, the mean σ_1 for leucine for rotamers in all ϕ, ψ bins with probability $>5\%$ is 10.8° (with a median of 13.9°) in the 2002 library, but is down to 7.9° (with a median of 7.2°) in the 2010 library. (For all leucine rotamers, the 2002 library reports a 13.1° mean, and a 9.9° median; the 2010 library reports a 9.2° mean, and a 9.9° median.) By decreasing the weight on the `fa_dun` term or by merely weakening the "off-rotamer penalty" (the $\sum_i (\chi - \bar{\chi}/\sigma)^2$ component of Eq. 6.14), the distributions may broaden and performance at the high-resolution refinement benchmark might improve. Encouragingly, decreasing the `fa_dun` weight down to one-half of its *Score12* weight does not substantially worsen rotamer recovery for the 2010 library (see the online appendix). There is still significant work, however, before we are ready to conclude that the new library should be adopted for general use in Rosetta.

7. CONCLUSION

We have described three tools that can be used to evaluate and improve macromolecular energy functions. Inaccuracies in the energy function can be identified by comparing features from crystal structures and computationally generated structures. New or reparameterized energy terms can be rapidly tested with optE to determine if the change improves structure prediction and sequence design. When a new term is ready to be rigorously tested, we can test for unintended changes to feature distributions by relying upon the existing set of feature analysis scripts, refit reference energies for protein design using the sequence-profile recovery protocol in optE, and measure the impact of the new term on a wide array of modeling problems by running the benchmarks curated here. Of the three changes we benchmarked in this paper, we recommend that the first two should be adopted. In the context of Rosetta, this means using *Score12Bicubic* rather than the current *Score12*.

ACKNOWLEDGMENTS

Support for A. L. F., M. J. O., and B. K. came from GM073151 and GM073960. Support for J. S. R. came from NIH R01 GM073930. Thanks to Steven Combs for bringing the bicubic-spline implementation to Rosetta.

REFERENCES

Bower, M. J., Cohen, F. E., & Dunbrack, R. L., Jr. (1997). Prediction of protein side-chain rotamers from a backbone-dependent rotamer library: A new homology modeling tool. *Journal of Molecular Biology*, *267*, 1268–1282.

Chen, V. B., Arendall, W. B., Headd, J. J., Keedy, D. A., Immormino, R. M., Kapral, G. J., et al. (2010). MolProbity: All-atom structure validation for macromolecular crystallography. *Acta Crystallographica. Section D: Biological Crystallography*, *66*, 12–21.

Chen, H.-M., Liu, B.-F., Huang, H.-L., Hwang, S.-F., & Ho, S.-Y. (2007). Sodock: Swarm optimization for highly flexible protein-ligand docking. *Journal of Computational Chemistry*, *28*, 612–623.

Dantas, G., Corrent, C., Reichow, S. L., Havranek, J. J., Eletr, Z. M., Isern, N. G., et al. (2007). High-resolution structural and thermodynamic analysis of extreme stabilization of human procarboxypeptidase by computational protein design. *Journal of Molecular Biology*, *366*, 1209–1221.

Das, R., Karanicolas, J., & Baker, D. (2010). Atomic accuracy in predicting and designing noncanonical RNA structure. *Nature Methods*, *7*, 291–294.

Ding, F., & Dokholyan, N. V. (2006). Emergence of protein fold families through rational design. *PLoS Computational Biology*, *2*, e85.

Dobson, N., Dantas, G., Baker, D., & Varani, G. (2006). High-resolution structural validation of the computational redesign of human U1A protein. *Structure*, *14*, 847–856.

Dunbrack, R. L., Jr. (2002). Rotamer libraries in the 21st century. *Current Opinion in Structural Biology*, *12*, 431–440.

Dunbrack, R. L., Jr., & Karplus, M. (1993). Backbone dependent rotamer library for proteins: Application to side chain prediction. *Journal of Molecular Biology*, *230*, 543–574.

Fabiola, G. F., Krishnaswarmy, S., Nagarajan, V., & Pattabhi, V. (1997). C-H· · ·O hydrogen bonds in β-sheets. *Acta Crystallographica. Section D: Biological Crystallography*, *53*, 316–320.

Fleishman, S. J., Leaver-Fay, A., Corn, J. E., Strauch, E.-M., Khare, S. D., Koga, N., et al. (2011). RosettaScripts: A scripting language interface to the Rosetta macromolecular modeling suite. *PLoS One*, *6*, e20161.

Gilis, D., & Rooman, M. (1997). Predicting protein stability changes upon mutation using database-derived potentials: Solvent accessibility determines the importance of local versus non-local interactions along the sequence. *Journal of Molecular Biology*, *272*, 276–290.

Guerois, R., Nielsen, J. E., & Serrano, L. (2002). Predicting changes in the stability of proteins and protein complexes: A study of more than 1000 mutations. *Journal of Molecular Biology*, *320*, 369–387.

Hamelryck, T., Borg, M., Paluszewski, M., Paulsen, J., Frellsen, J., Andreetta, C., et al. (2010). Potentials of mean force for protein structure prediction vindicated, formalized and generalized. *PLoS One*, *5*, e13714.

Jacak, R., Leaver-Fay, A., & Kuhlman, B. (2012). Computational protein design with explicit consideration of surface hydrophobic patches. *Proteins*, *80*, 825–838.

Jacobson, M. P., Kaminski, G. A., Friesner, R. A., & Rapp, C. S. (2002). Force field validation using protein side chain prediction. *The Journal of Physical Chemistry B*, *106*, 11673–11680.

Jorgensen, W. L., Maxwell, D. S., & Tirado-Rives, J. (1996). Development and testing of the OPLS all-atom force field on conformational energetics and properties of organic liquids. *Journal of the American Chemical Society*, *118*, 11225–11236.

Keedy, D. A., Arendall III, W. B., Chen, V. B., Williams, C. J., Headd, J. J., Echols, N., et al. (2012). 8000 Filtered Structures, 2012 http://kinemage.biochem.duke.edu/databases/top8000.php.

Kellogg, E. H., Leaver-Fay, A., & Baker, D. (2011). Role of conformational sampling in computing mutation-induced changes in protein structure and stability. *Proteins: Structure, Function, and Bioinformatics*, *79*, 830–838.

Khatib, F., Cooper, S., Tyka, M., Xu, K., Makedon, I., Popovic, Z., et al. (2011). Algorithm discovery by protein folding game players. *Proceedings of the National Academy of Sciences of the United States of America*, *109*, 5277–5282.

Kortemme, T., Morozov, A. V., & Baker, D. (2003). An orientation-dependent hydrogen bonding potential improves prediction of specificity and structure for proteins and protein-protein complexes. *Journal of Molecular Biology*, *326*, 1239–1259.

Kuhlman, B., & Baker, D. (2000). Native protein sequences are close to optimal for their structures. *Proceedings of the National Academy of Sciences of the United States of America*, *97*, 10383–10388.

Kuhlman, B., Dantas, G., Ireton, G., Varani, G., Stoddard, B., & Baker, D. (2003). Design of a novel globular protein fold with atomic-level accuracy. *Science*, *302*, 1364–1368.

Lazaridis, T., & Karplus, M. (1999). Effective energy function for proteins in solution. *Proteins: Structure, Function, and Genetics*, *35*, 133–152.

Leaver-Fay, A., Jacak, R., Stranges, P. B., & Kuhlman, B. (2011). A generic program for multistate protein design. *PLoS One*, *6*, e20937.

Leaver-Fay, A., Tyka, M., Lewis, S. M., Lange, O. F., Thompson, J., Jacak, R., et al. (2011). ROSETTA3: An object-oriented software suite for the simulation and design of macromolecules. *Methods in Enzymology*, *487*, 545–574.

LeCun, Y., & Jie, F. (2005). Loss functions for discriminative training of energy-based models. In: *Proceedings of the 10th international workshop on artificial intelligence and statistics, Society for AI and Statistics (AISTATS'05)*.

Liang, S., & Grishin, N. (2002). Side-chain modeling with an optimized scoring function. *Protein Science*, *11*, 322–331.

Lyskov, S., & Gray, J. J. (2012). RosettaTests. http://rosettatests.graylab.jhu.edu.

Mandell, D. J., Coutsias, E. A., & Kortemme, T. (2009). Sub-angstrom accuracy in protein loop reconstruction by robotics-inspired conformational sampling. *Nature Methods*, *6*, 551–552.

Meiler, J., & Baker, D. (2003). Rapid protein fold determination using unassigned NMR data. *Proceedings of the National Academy of Sciences of the United States of America*, *100*, 15404–15409.

Miyazawa, S., & Jernigan, R. L. (1985). Estimation of effective interresidue contact energies from protein crystal structures: Quasi-chemical approximation. *Macromolecules*, *18*, 534–552.

Morozov, A. V., & Kortemme, T. (2005). Potential functions for hydrogen bonds in protein structure prediction and design. *Advances in Protein Chemistry*, *72*, 1–38.

Murphy, P. M., Bolduc, J. M., Gallaher, J. L., Stoddard, B. L., & Baker, D. (2009). Alteration of enzyme specificity by computational loop remodeling and design. *Proceedings of the National Academy of Sciences of the United States of America*, *106*, 9215–9220.

Novotný, J., Bruccoleri, R., & Karplus, M. (1984). An analysis of incorrectly folded protein models. Implications for structure predictions. *Journal of Molecular Biology*, *177*, 787–818.

Park, B., & Levitt, M. (1996). Energy functions that discriminate X-ray and near native folds from well-constructed decoys. *Journal of Molecular Biology*, *258*, 367–392.

Petrella, R. J., Lazaridis, T., & Karplus, M. (1998). Protein sidechain conformer prediction: A test of the energy function. *Folding and Design*, *3*, 353–377.

Pierce, N., & Winfree, E. (2002). Protein design is NP-hard. *Protein Engineering*, *15*, 779–782.

Ponder, J. W., & Richards, F. M. (1987). Tertiary templates for proteins. Use of packing criteria in the enumeration of allowed sequences for different structural classes. *Journal of Molecular Biology*, *193*, 775–791.

Potapov, V., Cohen, M., & Schreiber, G. (2009). Assessing computational methods for predicting protein stability upon mutation: Good on average but not in the details. *Protein Engineering, Design & Selection*, *22*, 553–560.

Richardson, J. S., Keedy, D. A., & Richardson, D. C. (2013). The Plot Thickens: More Data, More Dimensions, More Uses. In M. Bansai & N. Srinivasan (Eds.), *Biomolecular Forms and Functions: A Celebration of 50 Years of the Ramachandran Map* (pp. 46–61). World Scientific.

Rohl, C. A., Strauss, C. E. M., Misura, K. M. S., & Baker, D. (2004). Protein structure prediction using Rosetta. *Methods in Enzymology*, *383*, 66–93.

Shapovalov, M. V., & Dunbrack, R. L. (2011). A smoothed backbone-dependent rotamer library for proteins derived from adaptive kernel density estimates and regressions. *Structure*, *19*, 844–858.

Sharabi, O. Z., Yanover, C., Dekel, A., & Shifman, J. M. (2010). Optimizing energy functions for protein-protein interface design. *Journal of Computational Chemistry*, *32*, 23–32.

Sheffler, W., & Baker, D. (2009). RosettaHoles: Rapid assessment of protein core packing for structure prediction, refinement, design and validation. *Protein Science*, *18*, 229–239.

Simons, K., 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. *Journal of Molecular Biology*, *268*, 209–225.

Simons, K. T., Ruczinski, I., Kooperberg, C., Fox, B. A., Bystroff, C., & Baker, D. (1999). Improved recognition of native-like protein structures using a combination of sequence-dependent and sequence-independent features of proteins. *Proteins*, *34*, 82–95.

Sippl, M. J. (1990). Calculation of conformational ensembles from potentials of mean force: An approach to the knowledge-based prediction of local structures in globular proteins. *Journal of Molecular Biology*, *213*, 859–883.

Song, Y., Tyka, M., Leaver-Fay, A., Thompson, J., & Baker, D. (2011). Structure guided forcefield optimization. *Proteins: Structure, Function, and Bioinformatics*, *79*, 1898–1909.

Taylor, R., & Kennard, O. (1982). Crystallographic evidence for the existence of the C-H···O, C-H···N and C-H···Cl hydrogen bonds. *Journal of the American Chemical Society*, *104*, 5063–5070.

Tyka, M. D., Keedy, D. A., André, I., Dimaio, F., Song, Y., Richardson, D. C., et al. (2010). Alternate states of proteins revealed by detailed energy landscape mapping. *Journal of Molecular Biology*, *405*, 607–618.

Wang, C., Bradley, P., & Baker, D. (2007). Protein-protein docking with backbone flexibility. *Journal of Molecular Biology*, *373*, 503–519.

Wang, C., Schueler-Furman, O., & Baker, D. (2005). Improved side-chain modeling for protein-protein docking. *Protein Science*, *14*, 1328–1339.

Weiner, S. J., Kollman, P. A., Case, D. A., Singh, U. C., Ghio, C., Alagona, G., et al. (1984). A new force field for molecular mechanical simulation of nucleic acids and proteins. *Journal of the American Chemical Society*, *106*, 765–784.

Wickham, H. (2010). A layered grammar of graphics. *Journal of Computational and Graphical Statistics*, *19*, 3–28.

Wickham, H. (2011). The split-apply-combine strategy for data analysis. *Journal of Statistical Software*, *40*, 1–29.

Wilkinson, L. (1999). *The grammar of graphics*. New York: Springer.

Word, J. M., Lovell, S. C., Richardson, J. S., & Richardson, D. C. (1999). Asparagine and glutamine: Using hydrogen atom contacts in the choice of side-chain amide orientation. *Journal of Molecular Biology*, *285*, 1735–1747.

Yanover, C., & Bradley, P. (2011). Extensive protein and DNA backbone sampling improves structure-based specificity prediction for C2H2 zinc fingers. *Nucleic Acids Research*, *39*, 4564–4576.