Computational design of PDZ domains: parameterization and performance of a simple model

David Mignon[1], Nicolas Panel[1], Xingyu Chen and Thomas Simonson*

†Laboratoire de Biochimie (UMR CNRS 7654), Ecole Polytechnique, Palaiseau, France

[1]Joint first authors. *Corresponding author: thomas.simonson@polytechnique.fr

Abstract

Short title: Computational design of PDZ domains

Keywords:

1 Introduction

PDZ domains are small, globular protein domains that help establish protein-protein interaction networks in the cell [**?**]. To do this, they form specific interactions with other, target proteins, usually by recognizing a few amino acids at the C-terminus of the target proteins. Due to their biological importance, PDZ domains have been targeted for protein engineering and/or ligand design. Thus, peptide ligands have been identified or developed that modulate the activity of PDZ domains involved in pathologies like cancer, rabies infection, and cystic fibrosis [**?**]. Engineered PDZ domains have also been used to elucidate principles of protein folding and evolution [**?**], and to test the engineering methods themselves.

One emerging method that has been applied to several PDZ domains is computational protein design (CPD) [1–7]. Starting from a 3D structural model, CPD explores a large space of possible sequences and conformations, to identify protein variants that have certain predefined properties, such as stability or ligand binding. Conformational space is usually defined by a library of sidechain rotamers, which can be discrete or continuous, and by a finite set of backbone conformations or a specific repertoire of allowed backbone deformations. The energy function usually combines physical and empirical terms [8–10]. Both solvent and the unfolded protein state are described implicitly.

Here, we consider a simple but important CPD variant; we optimize selected parameters of the energy function for PDZ proteins, we test the quality of the model, and we use it for two applications. Our CPD variant [11–13] uses an "MMGBSA" energy function, which combines a molecular mechanics protein energy function with a Generalized Born + Surface Area implicit solvent treatment. The folded protein structure is represented by a single, fixed, backbone conformation and a discrete sidechain rotamer library. The unfolded state energy depends only on sequence composition, not an explicit structural model. With this CPD variant, the main adjustable model parameters are the protein dielectric constant ϵ, a small set of atomic surface energy coefficients σ_i, and a collection of amino acid chemical potentials, or "reference energies" $\{E_t^r\}$ that each represent the contribution of a single amino acid of type t to the unfolded state energy.

We specifically optimize the reference energies, using a maximum likelihood formalism and a small set of eight PDZ test proteins. We compare two sets of surface coefficients and two values of the dielectric constant. The resulting parameter sets are tested by generating designned sequences for all eight test proteins and comparing them to natural

sequences, as well as sequences generated with the Rosetta energy function and software [**?**]. We also perform 100-300 nanosecond molecular dynamics simulations for a few designed sequences, to help assess their stability.

We then apply one of the optimized models to two problems. First, we do a series of Monte Carlo simulations of one of the test proteins, Tiam1, where the chemical potential of the hydrophobic amino acid types is gradually increased. As a result, hydrphobic amino acids gradually invade the protein interior, forming a hydrophobic core that is initially smaller, then larger than the natural one. The propensity of each core position to become hydrophobic can be seen as a structure-dependent hydrophobicity index, providing information on the designability of the protein core. The second application consists in designing four Tiam1 positions that are known to be involved in specific target recognition, and that have been experimentally mutated so as to modify the preferred Tiam1 target [**?**], switching its preference from the natural target peptide syndecan-1 to another peptide, Caspr4. We mutate these positions through Monte Carlo (MC) simulations of either the apo-protein or the protein in complex with either the natural peptide ligand, syndecan-1, or the ligand preferred by the quadruple mutant, Caspr4.

2 The unfolded state model

2.1 Maximum likelihood reference energies

We use Monte Carlo to generate a Markov chain of states [14, 15], such that the states are populated according to a Boltzmann distribution. One possible elementary move is a "mutation": we modify the sidechain type $t \to t'$ at a chosen position i in the folded protein, assigning a particular rotamer r' to the new sidechain. At the same time, we perform the reverse mutation in the unfolded protein, $t' \to t$. The corresponding energy change has the form:

$$\Delta E = \Delta E^f - \Delta E^u = \left(E^f(...t'_i, r'_i...) - E^f(...t_i, r_i...)\right) - (E^u(t'_i) - E^u(t_i)) \qquad (1)$$

ΔE measures the stability change due to the mutation. For a particular sequence S, the unfolded state energy has the form:

$$E^u_S = \sum_{i \in S} E^r(t_i). \qquad (2)$$

The type-dependent quantities $E^r(t) \equiv E^r_t$ are essential parameters in the simulation model, referred to as "reference energies". Our goal here is to choose them empirically

so that the simulation produces amino acid frequencies that match a set of target values, for example experimental values in the Pfam database. Specifically, we will choose them so as to maximize the probability, or likelihood of the target sequences.

Let S be a particular sequence. Its Boltzmann probability is

$$p(S) = \frac{1}{Z} \exp(-\beta \Delta G_S), \tag{3}$$

where $\Delta G_S = G_S^f - E_S^u$ is the folding free energy of S, G_S^f is the free energy of the folded form, and Z is a normalizing constant (the partition function). We then have

$$kT \ln p(S) = \sum_{i \in S} E^r(t_i) - G_S^f - kT \ln Z = \sum_{t \in \mathrm{aa}} n_S(t) E_t^r - G_S^f - kT \ln Z, \tag{4}$$

where the sum on the right is over the amino acid types and $n_S(t)$ is the number of amino acids of type t within the sequence S.

We now consider a set $\mathcal{S}$ of N target sequences S; we denote $\mathcal{L}$ the probability of the entire set, which depends on the model parameters E_t^r; we refer to $\mathcal{L}$ as their likelihood [16]. We have

$$kT \ln \mathcal{L} = \sum_S \sum_{t \in \mathrm{aa}} n_S(t) E_t^r - \sum_S G_S^f - NkT \ln Z = \sum_{t \in \mathrm{aa}} N(t) E_t^r - \sum_S G_S^f - NkT \ln Z, \tag{5}$$

where $N(t)$ is the number of amino acids of type t in the whole dataset $\mathcal{S}$. The normalization factor or partition function Z is a sum over all possible sequences R:

$$Z = \sum_R \exp(-\beta \Delta G_R) = \sum_R \exp(-\beta \Delta G_R^f) \Pi_{t \in aa} \exp(\beta n_R(t) E_t^r) \tag{6}$$

In view of maximizing $\mathcal{L}$, we consider the derivative of Z with respect to one of the E_t^r:

$$\frac{\partial Z}{\partial E_t^r} = \sum_R \beta n_R(t) \exp(-\beta \Delta G_R^f) \prod_{s \in aa} \exp(\beta n_R(s) E_s^r) \tag{7}$$

We then have

$$\frac{kT}{Z} \frac{\partial Z}{\partial E_t^r} = \frac{\sum_R n_R(t) \exp(-\beta \Delta G_R)}{\sum_R \exp(-\beta \Delta G_R)} = \langle n(t) \rangle. \tag{8}$$

The quantity on the right is the Boltzmann average of the number $n(t)$ of amino acids t over all possible sequences. In practice, this is the average population of t we would obtain in a long MC simulation. We note that, as usual in statistical mechanics [17], the derivative of $\ln Z$ with respect to one quantity (E_t^r) is equal to the ensemble average of the conjugate quantity ($\beta n_S(t)$).

A necessary condition to maximize $\ln \mathcal{L}$ is that its derivatives with respect to the E_t^r should all be zero. We see that

$$\frac{1}{N}\frac{\partial}{\partial E_t^r} \ln \mathcal{L} = \frac{1}{N} \sum_S n_S(t) - \langle n(t) \rangle = \frac{N(t)}{N} - \langle n(t) \rangle \tag{9}$$

so that

$$\mathcal{L} \text{ maximum} \Longrightarrow \frac{N(t)}{N} = \langle n(t) \rangle, \ \ \forall t \in \text{aa} \tag{10}$$

Thus, to maximize $\mathcal{L}$, we should choose $\{E_t^r\}$ such that a long simulation gives the same amino acid frequencies as the target database.

2.2 Searching for the maximum likelihood

To approach the maximum likelihood $\{E_t^r\}$ values, starting from a current guess $\{E_t^r(n)\}$, we will compare three methods. With the first method, we step along the gradient of $\ln \mathcal{L}$, using the update rule [16]:

$$E_t^r(n+1) = E_t^r(n) + \alpha \frac{\partial}{\partial E_t^r} \ln \mathcal{L} = E_t^r(n) + \delta E \left(n_t^{\text{exp}} - \langle n(t) \rangle_n \right) \tag{11}$$

Here, α is a constant; $n_t^{\text{exp}} = N(t)/N$ is the mean population of amino acid type t in the target database; $\langle \rangle_n$ indicates an average over a simulation done using the current reference energies $\{E_t^r(n)\}$, and δE is an empirical constant with the dimension of an energy, referred to as the update amplitude. This update procedure is repeated until convergence. We refer to this method as the linear update method.

The second method, used previously [11, 12], employs a logarithmic update rule:

$$E_t^r(n+1) = E_t^r(n) + kT \ \ln \frac{\langle n(t) \rangle_n}{n_t^{\text{exp}}} \tag{12}$$

where kT is a thermal energy, set empirically to 0.5 kcal/mol. We refer to this as the logarithmic update method. Both the linear and logarithmic update methods converge to the same optimum, specified by (Eq. 10).

More sophisticated optimization methods usually need to calculate not only the gradient of the cost function but also the function itself. Here, our cost function is $\mathcal{C} = \ln \mathcal{L}$. We can easily compute its gradient, but $\mathcal{C}$ itself is much harder to compute. Therefore, we introduce a simpler, auxiliary cost function,

$$C = \sum_{t \in \text{aa}} (n_t^{\text{exp}} - \langle n(t) \rangle_n)^2 \tag{13}$$

We will use the gradient of $\mathcal{C}$ to define the search direction, as with the linear update rule, but C to select the optimal update amplitude δE along this direction. Specifically, starting from $\{E_t^r(n)\}$, we run MC simulations with $\{E_t^r(n+1)\}$ values obtained from Eq. (11), using 3 or 4 trial δE values. Based on these 3-4 results, we then fit $C(\delta E)$ to a simple parabolic function, and we deduce the δE value that minimizes C along the current search direction. This value is denoted $\delta E(n)$. Finally, we update the $\{E_t^r(n)\}$ values through Eq. (11), using $\delta E(n)$ as the update amplitude. This update procedure is repeated until convergence. We refer to this method as the parabolic update method.

3 Computational methods

3.1 Effective energy function for the folded state

The energy matrix was computed with the following effective energy function for the folded state:

$$E = E_{\text{bonds}} + E_{\text{angles}} + E_{\text{dihe}} + E_{\text{impr}} + E_{\text{vdw}} + E_{\text{Coul}} + E_{\text{solv}} \tag{14}$$

The first six terms in (14) represent the protein internal energy. They were taken from the Amber ff99SB empirical energy function [18], slightly modified for CPD (see below). The last term on the right, E_{solv}, represents the contribution of solvent. We used a "Generalized Born + Surface Area", or GBSA implicit solvent model [19]:

$$E_{\text{solv}} = E_{\text{GB}} + E_{\text{surf}} = \frac{1}{2}\left(\frac{1}{\epsilon_W} - \frac{1}{\epsilon_P}\right) \sum_{ij} q_i q_j \left(r_{ij}^2 + b_i b_j \exp[-r_{ij}^2/4b_i b_j]\right)^{-1/2} + \sum_i \sigma_i A_i \tag{15}$$

Here, ϵ_W, ϵ_P are the solvent and protein dielectric constants; r_{ij} is the distance between atoms i, j and b_i is the "solvation radius" of atom i [19, 20]. A_i is the exposed solvent accessible surface area of atom i; σ_i is a parameter that reflects each atom's preference to be exposed or hidden from solvent. The solute atoms were divided into 4 groups with specific σ_i values (Table 1): unpolar, aromatic, polar, and ionic. Hydrogen atoms were assigned a surface coefficient of 0. Surface areas were computed by the Lee and Richards algorithm [21], implemented in the XPLOR program [22], using a 1.5 Å probe radius. Most of the MC simulations used a protein dielectric of $\epsilon_P = 4$ or 8 (see Results).

In the GB energy term, the atomic solvation radius b_i approximates the distance from i to the protein surface and is a function of the coordinates of all the protein atoms. The

particular b_i form corresponds to a GB variant we call GB/HCT, after its original authors [19], with model parameters optimized for use with the Amber force field [20]. Since b_i depends on the coordinates of all the solute atoms [19], an additional approximation is needed to make the GB energy term pairwise additive and define the energy matrix. We use a "Native Environment Approximation", or NEA, where the solvation radii b_i of a particular group (backbone, sidechain or ligand) are computed ahead of time, with the rest of the system having its native sequence and conformation [12, 23].

The surface energy contribution E_{surf} is not pairwise additive either, because in a protein structure, surface area buried by one sidechain may also be buried by another. To make this energy pairwise, Street et al proposed a simple procedure [24]. The buried surface of a sidechain is computed by summing over the neighboring sidechain and backbone groups. For each neighboring group, the contact area with the sidechain of interest is computed, independently of other surrounding groups. The contact areas are then summed. To avoid overcounting of buried surface area, a scaling factor is applied to the contact areas involving buried sidechains. Previous work showed that a scaling factor of 0.65 works well [20, 23].

The Amber force field ff99SB is slightly modified for CPD, with the original backbone charges replaced by a unified set, obtained by averaging over all amino acid types and adjusting slightly to make the backbone portion of each amino acid neutral [25].

3.2 Reference energies in the unfolded state

In the unfolded state, the energy depends on the sequence composition through a set of reference energies E_t^r (Eq. 2). The values are assigned based on amino acid types t, taking into account also the position of each amino acid in the folded structure, through its buried or solvent-exposed character. Thus, for a given type (Ala, say), there are two distinct E_t^r values: a buried and an exposed value. This is done even though the reference energies are used to represent the unfolded, not the folded state. There are two rationales for this: we assume residual structure is present in the unfolded state, so that amino acids partly retain their buried/exposed character; also, we recognize that the unfolded state model compensates in a systematic way for errors in the folded state energy function, so that the folded structure matters.

Distinguishing buried/exposed positions doubles the number of adjustable E_t^r parameters. Conversely, to reduce the number of adjustable parameters, we group amino acids into homologous classes (Table 2). Within each class c, and for each type of position

(buried or exposed), the reference energies have the form

$$E_t^r = E_c^r + \delta E_t^r \tag{16}$$

Here, E_c^r is an adjustable parameter while δE_t^r is a constant, computed as the molecular mechanics energy difference between amino acid types within the class c, assuming an unfolded conformation where each amino acid interacts only with itself and with solvent. During likelihood maximization, E_c^r is optimized while δE_t^r is held fixed. The δE_t^r values employed are listed in Table 2.

3.3 Experimental sequences

We considered a set of eight PDZ domains, whose PDB codes are listed in Table 3. To define the target amino acid frequencies for likelihood maximization, we collected homologous sequences for each of the eight. We started by a Blast search with the PDB sequence as the query, and retained homologs with a sequence identity, relative to the query, above a certain threshold, around 50% depending on the test protein. Within the retained homologs, sequences that were nearly identical were pruned, keeping just one of the redundant variants. This led to about xxx homologs per test protein. For each set, amino acid frequencies were computed, distinguishing buried and exposed positions. The eight sets of mean frequencies were then themselves averaged, giving the overall target amino acid frequencies (Table 4).

3.4 Structural models and Monte Carlo simulations

Structures were prepared and energy matrices computed using procedures described previously [26, 27]. Briefly, each PDB structure was minimized through 200 steps of conjugate gradient minimization. For each residue pair, interaction energies were computed after 15 steps of energy minimization, with the backbone fixed and only the interactions of the pair with each other and the backbone included. Sidechain rotamers were described by a slightly expanded version of the library of Tuffery et al [28], which has a total of 228 rotamers (sum over all amino acid types). The expansion consists in allowing additional hydrogen orientations for OH and SH groups [23]. This rotamer library was chosen for its simplicity and because it gives very good performance in sidechain placement tests, comparable to the specialized Scwrl4 program (which uses a much larger library) [29, 30]. Energy calculations were done with a modified version of the Xplor program [12, 22].

The Monte Carlo simulations use one- and two-position moves, where either rotamers, types, or both are changed. For two-position moves, the second position is selected among those that have a significant (unsigned) interaction energy with the first one, meaning that there is at least one rotamer conformation where their interaction is 10 kcal/mol or more. In addition, we mostly perform Replica Exchange Monte Carlo (REMC), where several simulations ("replicas" or "walkers") are propagated in parallel, at different temperatures; periodic swaps are attempted between the conformations of two walkers i ,j. The swap is accepted with probability

$$acc(\text{swap}_{ij}) = \text{Min}\left[1, e^{(\beta_i - \beta_j)(\Delta E_i - \Delta E_j)}\right] \tag{17}$$

where β_i, β_j are the inverse temperatures of the two walkers and ΔE_i, ΔE_j are the changes in their folding energies, due to the conformation change [31, 32]. We use eight walkers, with thermal energies kT_i that range from 0.125 to 2 kcal/mol, and are spaced in a geometric progression: T_{i+1}/T_i = constant [31]. Simulations are done with the proteus program [12]; REMC uses an efficient, shared-memory, OpenMP parallelization.

3.5 Rosetta sequence generation

Monte Carlo simulations were also done using the Rosetta program and energy function [**?**], for comparison. The simulations were done using version xxx of Rosetta (freely available online), using the command

```
rosetta -options blablabla
```

Simulations were run until 1000 low energy sequences were identified, corresponding to total run times of about xxx hours on a single core of a recent Intel processor.

3.6 Sequence characterization

Designed sequences were compared to the Pfam alignment for the PDZ family, using the Blosum40 scoring matrix and a gap penalty of -6. Each Pfam sequence was also compared to its own Pfam alignment. For these Pfam/Pfam comparisons, if a test protein T was part of the Pfam alignment, the T/T self comparison was left out, to be more consistent with the designed/Pfam comparisons. If the test protein T was not part of the Pfam alignment, we used Blast to identify its closest Pfam homologue H and left the T/H comparison out, for consistency. The Pfam alignment was either the "seed" alignment

(around 50 sequences) or the "full" alignment, with 14944 sequences. Similarities were computed for protein core residues, defined by their near-complete burial, and listed in Results.

Designed sequences were submitted to the Superfamily library of Hidden Markov Models [33, 34], which attempts to classify sequences according to the SCOP classification [35]. Classification was based on SCOP version 1.75 and version 3.5 of the Superfamily tools. Superfamily executes the hmmscan program, which implements a Hidden Markov model for each SCOP family and superfamily; here hmmscan was executed with an E-value threshold of 10^{-10}, using a total of 15438 models to represent the SCOP database.

3.7 Molecular dynamics simulations

For wildtype Tiam1 and a few designed sequences, to help assess the stability of the designed proteins, we ran MD simulations with explicit solvent. Starting structures were taken from the MC trajectory or the crystal structure (wildtype protein) and slightly minimized through xxx steps of conjugate gradient minimization. The protein was immersed in a large box of water; waters overlapping protein were eliminated, and the solvated system was truncated to the shape of a truncated octahedral box using the Charmm graphical interface or GUI [**?**]. A few sodium or chloride ions were included to ensure overall electroneutrality. Protonation states of histidines were assigned to be neutral, based on visual inspection. MD was done at room temperature and pressure, using a Nose-Hoover thermostat and barostat. Long-range electrostatic interactions were treated with a Particle Mesh Ewald approach [36]. The Amber ff99SB forcefield was used for the protein; the TIP3P model [37] was used for water. Simulations were run for 100–300 nanoseconds, depending on the sequence, using the Charmm and NAMD programs [38, 39].

4 Results

4.1 Optimizing the unfolded state model

4.2 Assessing designed sequence quality

4.3 Varying the size of the Tiam1 hydrophobic core

4.4 Designing specificity positions in Tiam1

5 Discussion

References

[1] Dantas, G., Kuhlman, B., Callender, D., Wong, M., and Baker, D. A large test of computational protein design: Folding and stability of nine completely redesigned globular proteins. *J. Mol. Biol. 332* (2003), 449–460.

[2] Butterfoss, G. L., and Kuhlman, B. Computer-based design of novel protein structures. *Ann. Rev. Biophys. Biomolec. Struct. 35* (2006), 49–65.

[3] Lippow, S. M., and Tidor, B. Progress in computational protein design. *Curr. Opin. Biotech. 18* (2007), 305–311.

[4] Saven, J. G. Computational protein design: engineering molecular diversity, nonnatural enzymes, nonbiological cofactor complexes, and membrane proteins. *Curr. Opin. Chem. Biol. 15* (2011), 452–457.

[5] Feldmeier, K., and Hoecker, B. Computational protein design of ligand binding and catalysis. *Curr. Opin. Chem. Biol. 17* (2013), 929–933.

[6] Tinberg, C. E., Khare, S. D., Dou, J., Doyle, L., Nelson, J. W., Schena, A., Jankowski, W., Kalodimos, C. G., Johnsson, K., Stoddard, B. L., and Baker, D. Computational design of ligand-binding proteins with high affinity and selectivity. *Nature 501* (2013), 212–218.

[7] Nisonoff, P. G. H. M., and Donald, B. R. Algorithms for protein design. *Curr. Opin. Struct. Biol. 39* (2016), 16–26.

[8] Pokala, N., and Handel, T. Energy functions for protein design I: Efficient and accurate continuum electrostatics and solvation. *Prot. Sci. 13* (2004), 925–936.

[9] Samish, I., MacDermaid, C. M., Perez-Aguilar, J. M., and Saven, J. G. Theoretical and computational protein design. *Ann. Rev. Phys. Chem. 62* (2011), 129—149.

[10] Li, Z., Yang, Y., Zhan, J., Dai, L., and Zhou, Y. Energy functions in de novo protein design: Current challenges and future prospects. *Ann. Rev. Biochem 42* (2013), 315–335.

[11] Schmidt am Busch, M., Lopes, A., Mignon, D., and Simonson, T. Computational protein design: software implementation, parameter optimization, and performance of a simple model. *J. Comput. Chem. 29* (2008), 1092–1102.

[12] Simonson, T., Gaillard, T., Mignon, D., Schmidt am Busch, M., Lopes, A., Amara, N., Polydorides, S., Sedano, A., Druart, K., and Archontis, G. Computational protein design: the Proteus software and selected applications. *J. Comput. Chem. 34* (2013), 2472–2484.

[13] Polydorides, S., Michael, E., Mignon, D., Druart, K., Archontis, G., and Simonson, T. Proteus and the design of ligand binding sites. In *Methods in Molecular Biology: Design and Creation of Protein Ligand Binding Proteins*, B. Stoddard, Ed., vol. 1414. Springer Verlag, New York, 2016, p. 0000.

[14] Frenkel, D., and Smit, B. *Understanding molecular simulation, Chapter 3.* Academic Press, New York, 1996.

[15] Grimmett, G. R., and Stirzaker, D. R. *Probability and random processes.* Oxford University Press, 2001.

[16] Kleinman, C. L., Rodrigue, N., Bonnard, C., Philippe, H., and Lartillot, N. A maximum likelihood framework for protein design. *BMC Bioinf. 7* (2006), Art. 326.

[17] Fowler, R. H., and Guggenheim, E. A. *Statistical Thermodynamics.* Cambridge University Press, 1939.

[18] Cornell, W., Cieplak, P., Bayly, C., Gould, I., Merz, K., Ferguson, D., Spellmeyer, D., Fox, T., Caldwell, J., and Kollman, P. A second generation force field for the simulation of proteins, nucleic acids, and organic molecules. *J. Am. Chem. Soc. 117* (1995), 5179–5197.

[19] Hawkins, G. D., Cramer, C., and Truhlar, D. Pairwise descreening of solute charges from a dielectric medium. *Chem. Phys. Lett. 246* (1995), 122–129.

[20] Lopes, A., Aleksandrov, A., Bathelt, C., Archontis, G., and Simonson, T. Computational sidechain placement and protein mutagenesis with implicit solvent models. *Proteins 67* (2007), 853–867.

[21] Lee, B., and Richards, F. The interpretation of protein structures: estimation of static accessibility. *J. Mol. Biol. 55* (1971), 379–400.

[22] Brünger, A. T. X-plor *version 3.1, A System for X-ray crystallography and NMR*. Yale University Press, New Haven, 1992.

[23] Gaillard, T., and Simonson, T. Pairwise decomposition of an MMGBSA energy function for computational protein design. *J. Comput. Chem. 35* (2014), 1371–1387.

[24] Street, A. G., and Mayo, S. Pairwise calculation of protein solvent-accessible surface areas. *Folding and Design 3* (1998), 253–258.

[25] Aleksandrov, A., Polydorides, S., Archontis, G., and Simonson, T. Predicting the acid/base behavior of proteins: A constant-pH Monte Carlo approach with Generalized Born solvent. *J. Phys. Chem. B 114* (2010), 10634–10648.

[26] Schmidt am Busch, M., Mignon, D., and Simonson, T. Computational protein design as a tool for fold recognition. *Proteins 77* (2009), 139–158.

[27] Schmidt am Busch, M., Sedano, A., and Simonson, T. Computational protein design: validation and possible relevance as a tool for homology searching and fold recognition. *PLoS One 5(5)* (2010), e10410.

[28] Tuffery, P., Etchebest, C., Hazout, S., and Lavery, R. A new approach to the rapid determination of protein side chain conformations. *J. Biomol. Struct. Dyn. 8* (1991), 1267.

[29] Krivov, G. G., Shapalov, M. V., and Dunbrack, R. L. Improved prediction of protein side-chain conformations with SCWRL4. *Proteins 77* (2009), 778–795.

[30] Gaillard, T., Panel, N., and Simonson, T. Protein sidechain conformation predictions with an MMGBSA energy function. *Proteins 84* (2016), 803–819.

[31] Kofke, D. A. On the acceptance probability of replica-exchange Monte Carlo trials. *J. Chem. Phys. 117* (2002), 6911–6914.

[32] Earl, D., and Deem, M. W. Parallel tempering: theory, applications, and new perspectives. *Phys. Chem. Chem. Phys. 7* (2005), 3910–3916.

[33] Gough, J., Karplus, K., Hughey, R., and Chothia, C. Assignment of homology to genome sequences using a library of hidden Markov models that represent all proteins of known structure. *J. Mol. Biol. 313* (2001), 903–919.

[34] Wilson, D., Madera, M., Vogel, C., Chothia, C., and Gough, J. The SUPERFAMILY database in 2007: families and functions. *Nucl. Acids Res. 35* (2007), D308––D313.

[35] Andreeva, A., Howorth, D., Brenner, S. E., Hubbard, J. J., Chothia, C., and Murzin, A. G. SCOP database in 2004: refinements integrate structure and sequence family data. *Nucl. Acids Res. 32* (2004), D226–229.

[36] Darden, T. Treatment of long-range forces and potential. In *Computational Biochemistry & Biophysics*, O. Becker, A. Mackerell Jr., B. Roux, and M. Watanabe, Eds. Marcel Dekker, N.Y., 2001, ch. 4.

[37] Jorgensen, W., Chandrasekar, J., Madura, J., Impey, R., and Klein, M. Comparison of simple potential functions for simulating liquid water. *J. Chem. Phys. 79* (1983), 926–935.

[38] Brooks, B., Brooks III, C. L., Mackerell Jr., A. D., Nilsson, L., Petrella, R. J., Roux, B., Won, Y., Archontis, G., Bartels, C., Boresch, S., Caflisch, A., Caves, L., Cui, Q., Dinner, A. R., Feig, M., Fischer, S., Gao, J., Hodoscek, M., Im, W., Kuczera, K., Lazaridis, T., Ma, J., Ovchinnikov, V., Paci, E., Pastor, R. W., Post, C. B., Pu, J. Z., Schaefer, M., Tidor, B., Venable, R. M., Woodcock, H. L., Wu, X., Yang, W., York, D. M., and Karplus, M. CHARMM: The biomolecular simulation program. *J. Comp. Chem. 30* (2009), 1545–1614.

[39] Phillips, J. C., Braun, R., Wang, W., Gumbart, J., Tajkhorshid, E., Villa, E., Chipot, C., Skeel, R. D., Kale, L., and Schulten, K. Scalable molecular dynamics with NAMD. *J. Comput. Chem. 26* (2005), 1781–1802.

Table 1: Atomic surface parameters

=

—

).

Table 2: Amino acid classes

=

—

).

Table 3: PDZ test proteins

=

—

).

Table 4: Amino acid frequencies

=

—

).