On Simplified Global Nonlinear Function for Fitness Landscape: A Case Study of Inverse Protein Folding

Yun Xu, Changyu Hu, Yang Dai, Jie Liang*

Department of Bioengineering, University of Illinois at Chicago, Chicago, IL, United States of America

Abstract

The construction of fitness landscape has broad implication in understanding molecular evolution, cellular epigenetic state, and protein structures. We studied the problem of constructing fitness landscape of inverse protein folding or protein design, with the aim to generate amino acid sequences that would fold into an *a priori* determined structural fold which would enable engineering novel or enhanced biochemistry. For this task, an effective fitness function should allow identification of correct sequences that would fold into the desired structure. In this study, we showed that nonlinear fitness function for protein design can be constructed using a rectangular kernel with a basis set of proteins and decoys chosen *a priori*. The full landscape for a large number of protein folds can be captured using only 480 native proteins and 3,200 non-protein decoys via a finite Newton method. A blind test of a simplified version of fitness function for sequence design was carried out to discriminate simultaneously 428 native sequences not homologous to any training proteins from 11 million challenging protein-like decoys. This simplified function correctly classified 408 native sequences (20 misclassifications, 95% correct rate), which outperforms several other statistical linear scoring function and optimized linear function. Our results further suggested that for the task of global sequence design of 428 selected proteins, the search space of protein shape and sequence can be effectively parametrized with just about 3,680 carefully chosen basis set of proteins and decoys, and we showed in addition that the overall landscape is not overly sensitive to the specific choice of this set. Our results can be generalized to construct other types of fitness landscape.

Citation: Xu Y, Hu C, Dai Y, Liang J (2014) On Simplified Global Nonlinear Function for Fitness Landscape: A Case Study of Inverse Protein Folding. PLoS ONE 9(8): e104403. doi:10.1371/journal.pone.0104403

Editor: Yang Zhang, University of Michigan, United States of America

Received December 26, 2013; **Accepted** July 14, 2014; **Published** August 11, 2014

Funding: This work was supported by NSF grants DBI 1062328 and DMS-0800257, and NIH grants GM079804 and GM086145. This work was also funded by the Chicago Biomedical Consortium with support from the Searle Funds at the Chicago Community Trust and the UIC Research Open Access Article Publishing (ROAAP) Fund. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

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

* Email: jliang@uic.edu

Introduction

Protein design has been the focus of many experimental, theoretical, and computational studies [1–9]. Despite significant challenges, important progresses have been made, with profound implications in biotechnology and biomedicine [10–15].

Here we studied the problem of designing a protein sequence that is compatible with an *a priori* specified three-dimensional template protein fold. This problem was first formulated 30 years ago [16,17]. Also known as the inverse protein folding problem, it addresses the fundamental problem of designing proteins to facilitate engineering of proteins with enhanced or novel biochemical functions.

A key component for designing a protein sequence is a fitness function: it can detect if a solution has been found, and can also guide the search of viable sequences. An ideal fitness function can characterize the properties of fitness landscape of many proteins simultaneously. Such a fitness function would be useful for designing novel proteins and novel functions, as well as for studying the global evolution of protein structure and protein functions.

The development of a fitness function for protein design is closely related to the development of a scoring function for protein structure predictions, protein folding, and protein-protein/ligand docking [18–23]. There are many different approaches in constructing the fitness function. Several studies employ a linear fitness function in the form of weighted linear sum of pairwise contacts, with sometimes additional solvation terms derived from exposed surface area [2,3,5]. Such functions can be obtained from statistical analysis of a database of protein structures [24], or from perceptron learning/linear programming [21,25,26], or by gradient descent [27,28]. Another approach is to use a force field such as those used in molecular dynamics simulations [6,29–31]. However these functions often do not provide global characterization of the overall fitness landscape for protein design. They also often have poor performance in blind test when challenged with the task of designing simultaneously many different proteins [32], or are so complex that they can not be used in high-throughput test. Inaccurate fitness functions can lead to low success rates in protein design [33].

A promising alternative approach is to use nonlinear function to capture the complex design of fitness landscape. In the study of [32], a nonlinear Gaussian kernel function was constructed by maximizing soft margins between native proteins and decoy non-proteins. This fitness function significantly outperforms linear functions in a blind test of identifying 201 native proteins from 3 million challenging protein-like decoys [32]. However, it is parametrized by about 350 native proteins and 4,700 non-protein decoys and its form is rather complex. It is computationally expensive to evaluate the fitness of a candidate sequence. Although obtaining a good answer at high computational cost is acceptable for some tasks, it is difficult to incorporate a complex function in a search algorithm. It is also difficult to characterize global landscape properties of protein sequence design using a complex function.

In this study, we demonstrated how to significantly improve nonlinear function for characterizing fitness landscape of protein design. Using a rectangular kernel with proteins and decoys chosen *a priori*, we obtained a nonlinear kernel function via a finite Newton method. The total number of native proteins and decoy conformations included in the function was reduced to about 3,680. In the blind test of sequence design to discriminate 428 native sequences from 11 million challenging protein-like decoy sequences, this fitness function misclassified only 20 native sequences (correct rate 95%), which far outperform statistical function [34] (87 misclassification, correct rate 57%) and linear optimal functions [26,28] (44–58 misclassification, correct rate 78%–71%) both of which were tested on a smaller scale to discriminate 201 native sequence from 3 million challenging protein-like decoy sequences. It is also comparable to the results of 18 misclassification (correct rate 91%) using far more complex nonlinear fitness function with >5,000 terms [32].

This paper is organized as follows. We first describe our theory and methods for sequence design. We then discuss computational details. Results of a blind test are then presented. We conclude with discussion and remarks.

Theory and Methods

We use a d-dimensional vector $\boldsymbol{c} \in \mathbb{R}^d$ to represent both the sequence and structure of a protein [35]. One possible choice is the vector of the number count of non-bonded pairwise contacts of each of the $\binom{20+2-1}{2} = 210$ contact types [24] between the 20 types of amino acid residues in a protein structure. Once the structural conformation of a protein $\boldsymbol{s}$ and its amino acid sequence $\boldsymbol{a}$ is given, the contact definition $f: (\boldsymbol{s},\boldsymbol{a}) \mapsto \mathbb{R}^d$ fully determines the contact vector $\boldsymbol{c}$.

Inequality criterion

In protein design, the native amino acid sequence $\boldsymbol{a}$ of a protein should have better fitness score on the native structure $\boldsymbol{s}$ of this protein than any other competing sequences taken from proteins of different fold. This leads to the requirement that the native sequence $\boldsymbol{a}_N$ mounted on its native structure $\boldsymbol{s}_N$ should have the best fitness score (lowest "energy") compared to a set of decoys $\mathcal{D} = \{D | \boldsymbol{c}_D = f(\boldsymbol{s}_N, \boldsymbol{a}_D) \text{ for all } \boldsymbol{a}_D\}$ derived from mounting unrelated alternative sequences $\boldsymbol{a}_D$ on the native protein structure $\boldsymbol{s}_N$:

$$H(\boldsymbol{c}_N) < H(\boldsymbol{c}_D) \quad \text{for all } D \in \mathcal{D}, \tag{1}$$

where $\boldsymbol{c}_D = f(\boldsymbol{s}_N, \boldsymbol{a}_D)$ is the contact vector of a decoy sequence $\boldsymbol{a}_D$ mounted on its native protein structure $\boldsymbol{s}_N$, and $\boldsymbol{c}_N = f(\boldsymbol{s}_N, \boldsymbol{a}_N)$ is the contact vector of a native sequence $\boldsymbol{a}_N$ from the set of native training proteins $\mathcal{N}$ mounted on the native structure $\boldsymbol{s}_N$. Here $\mathcal{D}$ is a set of sequence decoys mounted on native protein structures. $H(\boldsymbol{c}_N)$ and $H(\boldsymbol{c}_D)$ are the energy score for native sequence structure pair and for non-native sequence structure pair, respectively. Equivalently, the native sequence will have the highest probability to fit into its native structure, and other sequences will have lower probability. This is the same principle described in [3–5].

A commonly used form for fitness function $H(\boldsymbol{c})$ is the weighted linear sum of pairwise contacts [24,26,36–38]:

$$H(\boldsymbol{c}) = \boldsymbol{w} \cdot \boldsymbol{c}, \tag{2}$$

here "·" represents inner product of two vectors. For such a linear function, the basic requirement for protein fitness is then:

$$\boldsymbol{w} \cdot (\boldsymbol{c}_N - \boldsymbol{c}_D) < 0. \tag{3}$$

We can further require that the difference in fitness must be greater than a constant $d > 0$:

$$\boldsymbol{w} \cdot (\boldsymbol{c}_N - \boldsymbol{c}_D) + d < 0. \tag{4}$$

Geometric views of inequality requirement

There is a natural geometric view of the inequality requirement. Each of the inequalities divides the space of $\mathbb{R}^d$ into two halves separated by a hyperplane. The hyperplane is defined by the normal vector $(\boldsymbol{c}_N - \boldsymbol{c}_D)$ and its distance $d/\|\boldsymbol{c}_N - \boldsymbol{c}_D\|$ from the origin. The weight vector $\boldsymbol{w}$ must be located in the half-space opposite to the direction of the normal vector $(\boldsymbol{c}_N - \boldsymbol{c}_D)$. This half-space can be written as $\boldsymbol{w} \cdot (\boldsymbol{c}_N - \boldsymbol{c}_D) + d < 0$. When there are many inequalities to be satisfied simultaneously, the intersection of the half-spaces forms a convex polyhedron [39]. If the weight vector is located in the interior of the polyhedron, all inequalities are satisfied. Fitness function with such weight vector $\boldsymbol{w}$ can discriminate a native protein from all decoys.

For each native protein i, there is one convex polyhedron $\mathcal{P}_i$ formed by the set of inequalities associated with its decoys. If the scoring function can discriminate simultaneously n native contact vectors from a union of sets of decoys, the weight vector $\boldsymbol{w}$ must be located in the interior of a smaller convex polyhedron $\mathcal{P}$ that is the intersection of the n convex polyhedra: $\boldsymbol{w} \in \operatorname{Int} \mathcal{P} = \operatorname{Int} \bigcap_{i=1}^{n} \mathcal{P}_i$.

There is another geometric view of the inequality requirements. The relationship $\boldsymbol{w} \cdot (\boldsymbol{c}_N - \boldsymbol{c}_D) + d < 0$ for all decoys and native protein sequences can be regarded as a requirement that all points $\{\boldsymbol{c}_N - \boldsymbol{c}_D\}$ are located on one side of a hyperplane, which is defined by its normal vector $\boldsymbol{w}$ and its distance $d/\|\boldsymbol{w}\|$ to the origin. We can show that such a hyperplane exists if and only if the origin is not contained within the convex hull of the set of points $\{\boldsymbol{c}_N - \boldsymbol{c}_D\}$ [32]. This second geometric view is dual to the first geometric view.

Relation to support vector machines

There may exist multiple $\boldsymbol{w}$'s if $\mathcal{P}$ is not empty. We can use the formulation of a support vector machine to find a $\boldsymbol{w}$. Let all vectors $\boldsymbol{c}_N \in \mathbb{R}^d$ form a native training set and all vectors $\boldsymbol{c}_D \in \mathbb{R}^d$ form a decoy training set. Each vector in the native training set is labeled as -1 and each vector in the decoy training set is labeled as $+1$. Then solving the following support vector machine problem will provide an optimal solution to inequalities (3):

$$\begin{aligned} \text{Minimize} \quad & \frac{1}{2}\|\boldsymbol{w}\|^2 \\ \text{subject to} \quad & \boldsymbol{w} \cdot \boldsymbol{c}_N + b \le -1 \\ & \boldsymbol{w} \cdot \boldsymbol{c}_D + b \ge 1. \end{aligned} \tag{5}$$

Note that a solution of the above problem satisfies the system of inequalities (3), since subtracting the second inequality from the first inequality in the constraint conditions of (5) will give us $\boldsymbol{w}\cdot(\boldsymbol{c}_N - \boldsymbol{c}_D) \le -2 < 0$.

Nonlinear fitness function

However, it is possible that no weight vector $\boldsymbol{w}$ exists, *i*.e., the interior of the final convex polyhedron $\mathrm{Int}\mathcal{P} = \mathrm{Int} \bigcap_{i=1}^{n} \mathcal{P}_i$ may be an empty set. First, for a specific native protein i, there may be severe restriction from some inequality constraints, which makes $\mathcal{P}_i$ an empty set. Some decoys are very difficult to discriminate due to perhaps deficiency in protein representation. In these cases, it is impossible to adjust the weight vector so the native structure has a better fitness score than the decoy. Second, even if a weight vector $\boldsymbol{w}$ can be found for each native protein, *i*.e., $\boldsymbol{w}$ is contained in a nonempty polyhedron, it is still possible that the intersection of the interior of n nonempty polyhedra is an empty set, *i*.e., no weight vector can be found that can make all native proteins simultaneously the fittest against decoys.

A fundamental reason for this failure is that the functional form of linear sum of pairwise interaction is too simplistic. To resolve this issue, we obtain nonlinear fitness function for sequence design using an alternative functional form [32]:

$$H(\boldsymbol{c}) = \sum_{D \in \mathcal{D}} \alpha_D K(\boldsymbol{c}, \boldsymbol{c}_D) - \sum_{N \in \mathcal{N}} \alpha_N K(\boldsymbol{c}, \boldsymbol{c}_N) + b, \tag{6}$$

where $\alpha_D \ge 0$ and $\alpha_N \ge 0$ are coefficients to be determined. This functional form is reminiscent of the linear fitness function $H(\boldsymbol{c}) = \boldsymbol{w}\cdot\boldsymbol{c}$, which can be written alternatively as an expansion around positive and negative contact vectors, as used in perceptron learning: $\boldsymbol{w} = -\sum_{N\in\mathcal{N}} \alpha_N \boldsymbol{c}_N + \sum_{D\in\mathcal{D}} \alpha_D \boldsymbol{c}_D$. A convenient kernel function K is:

$$K(\boldsymbol{c}_i, \boldsymbol{c}_j) = e^{-\gamma \|\boldsymbol{c}_i - \boldsymbol{c}_j\|^2} \text{ for any vectors } \boldsymbol{c}_i \text{ and } \boldsymbol{c}_j \in \mathcal{N} \cup \mathcal{D}, \tag{7}$$

where γ is a constant. The fitness function $H(\mathbf{c})$ can be written compactly as:

$$\begin{aligned} H(\boldsymbol{c}) &= \sum_{D\in\mathcal{D}} \alpha_D e^{-\gamma\|\boldsymbol{c}-\boldsymbol{c}_D\|^2} - \sum_{N\in\mathcal{N}} \alpha_N e^{-\gamma\|\boldsymbol{c}-\boldsymbol{c}_N\|^2} + b \\ &= K(\boldsymbol{c}, A) D_s \boldsymbol{\alpha} + b, \end{aligned} \tag{8}$$

where A is the matrix of training data: $A = (\boldsymbol{c}_1^T, \cdots, \boldsymbol{c}_{|\mathcal{D}|}^T, \boldsymbol{c}_{|\mathcal{D}|+1}^T, \cdots, \boldsymbol{c}_{|\mathcal{D}|+|\mathcal{N}|}^T)^T$, and the entry $K(\boldsymbol{c}, \boldsymbol{c}_j)$ of $K(\boldsymbol{c}, A)$ is $e^{-\gamma\|\boldsymbol{c}-\boldsymbol{c}_j\|^2}$. D_s is the diagonal matrix with $+1$ and -1 along its diagonal representing the membership class of each point $A_i = \boldsymbol{c}_i^T$. Here $\boldsymbol{\alpha}$ is the coefficient vector: $\boldsymbol{\alpha} = (\alpha_1, \cdots, \alpha_{|\mathcal{D}|}, \alpha_{|\mathcal{D}|+1}, \cdots, \alpha_{|\mathcal{D}|+\mathcal{N}|})^T$.

Intuitively, the fitness landscape has smooth Gaussian hills of height α_D centered on location $\boldsymbol{c}_D$ of decoy contact vector $D \in \mathcal{D}$, and has smooth Gaussian cones of depth α_N centered on the location $\boldsymbol{c}_N$ of native contact vector $N \in \mathcal{N}$. Ideally, the value of the fitness function will be -1 for contact vectors $\boldsymbol{c}_N$ of native proteins, and will be $+1$ for contact vectors $\boldsymbol{c}_D$ of decoys.

Optimal nonlinear fitness function

To obtain such a nonlinear function, our goal is to find a set of parameters $\{\alpha_D, \alpha_N\}$ such that $H(\boldsymbol{c})$ has fitness value close to -1 for native proteins, and has fitness values close to $+1$ for decoys. There are many different choices of $\{\alpha_D, \alpha_N\}$. We use an optimality criterion developed in statistical learning theory [40–42]. First, we note that we have implicitly mapped each protein and decoy from $\mathbb{R}^d, d = 210$ to another high dimensional space where the scalar product of a pair of mapped points can be efficiently calculated by the kernel function $K(.,.)$. Second, we find the hyperplane of the largest margin distance separating proteins and decoys in the space transformed by the nonlinear kernel [40–43]. That is, we search for a hyperplane with equal and maximal distance to the closest native protein sequence and the closest decoys. Such a hyperplane has good performance in discrimination [40]. It can be found using support vector machine by obtaining the parameters $\{\alpha_D\}$ and $\{\alpha_N\}$ from solving the following primal form of quadratic programming problem:

$$\begin{array}{ll} \displaystyle\min_{\boldsymbol{\alpha}\in\mathbb{R}_+^m,\, b\in\mathbb{R},\, \xi\in\mathbb{R}^m} & \dfrac{C}{2}\boldsymbol{e}\cdot\boldsymbol{\xi} + \dfrac{1}{2}\boldsymbol{\alpha}\cdot\boldsymbol{\alpha} \\ \text{subject to} & D_s(K(A,A)D_s\boldsymbol{\alpha} + b\boldsymbol{e}) + \boldsymbol{\xi} \;\ge\; \boldsymbol{e} \\ & \qquad\qquad\qquad\qquad\quad \boldsymbol{\xi} \;\ge\; \mathbf{0}, \end{array} \tag{9}$$

where m is the total number of training points: $m = |\mathcal{D}| + |\mathcal{N}|$, C is a regularizing constant that limits the influence of each misclassified conformation [40–43], and the $m \times m$ diagonal matrix of signs D_s with $+1$ or -1 along its diagonal indicating the membership of each point A_i in the classes $+1$ or -1; and $\boldsymbol{e}$ is an m-vector with 1 at each entry. The variable ξ_i is a measurement of error for each input vector with respect to the solution: $\xi_i = 1 + y_i H(\boldsymbol{c}_i)$, where $y_i = -1$ if i is a native protein, and $y_i = +1$ if i is a decoy.

Rectangle kernel and reduced support vector machine (RSVM)

The use of nonlinear kernels on large datasets typically demands a prohibiting size of the computer memory in solving the potentially enormous unconstrained optimization problem. Moreover, the representation of the landscape surface using a large data set requires costly storage and computing time for the evaluation of a new unseen contact vector $\boldsymbol{c}$. To overcome these difficulties, the reduced support vector machines (RSVM) developed by Lee and Mangasarian [44] use a very small random subset of the training set to build a rectangular kernel matrix, instead of the use of the conventional $m \times m$ kernel matrix $K(A,A)$ in equation (9). This model can achieve about 10% improvement on test accuracy over conventional support vector machine with random data sets of sizes between $1-5\%$ of the original data [44]. The small subset can be regarded as a basis set in our study. Suppose that the number of contact vectors in our basis set is $\bar{m}$, with $\bar{m} \ll m$. We denote $\bar{A}$ as an $\bar{m} \times d$ matrix, and each contact vector from the basis set is represented by a row vector of $\bar{A}$. The resulting kernel matrix $K(A, \bar{A})$ from A and $\bar{A}$ has size $m \times \bar{m}$. Each entry of this rectangular kernel matrix is calculated by $K(\boldsymbol{c}_i, \bar{\boldsymbol{c}}_j)$, where $\boldsymbol{c}_i^T$ and $\bar{\boldsymbol{c}}_j^T$ are rows from A and $\bar{A}$ respectively. The RSVM is formulated as the following quadratic program:

$$\begin{array}{ll} \displaystyle\min_{\bar{\boldsymbol{\alpha}}\in\mathbb{R}_+^{\bar{m}},\, b\in\mathbb{R},\, \xi\in\mathbb{R}^{\bar{m}}} & \dfrac{C}{2}\boldsymbol{\xi}\cdot\boldsymbol{\xi} + \dfrac{1}{2}(\bar{\boldsymbol{\alpha}}\cdot\bar{\boldsymbol{\alpha}} + b^2) \\ \text{subject to} & D_s(K(A,\bar{A})\bar{D}_s\bar{\boldsymbol{\alpha}} + b\mathbf{e}) + \boldsymbol{\xi} \;\ge\; \boldsymbol{e} \\ & \qquad\qquad\qquad\qquad\quad \boldsymbol{\xi} \;\ge\; \mathbf{0}, \end{array} \tag{10}$$

where $\bar{D}_s$ is the $\bar{m} \times \bar{m}$ diagonal matrix with $+1$ or -1 along its diagonal, indicating the membership of each point $\bar{A}_i$ in the classes $+1$ or -1; and $\boldsymbol{e}$ is an m-vector with 1 at each entry. As shown in

[44], the zero level set surface of the fitness function is given by

$$\begin{aligned} H(\boldsymbol{c}) &= K(\boldsymbol{c},\bar{A})\bar{D}_s\bar{\boldsymbol{\alpha}}+b \\ &= \sum_{\boldsymbol{c}_D\in\bar{A}} \bar{\alpha}_D e^{-\gamma\|\boldsymbol{c}-\boldsymbol{c}_D\|^2} - \sum_{\boldsymbol{c}_N\in\bar{A}} \bar{\alpha}_N e^{-\gamma\|\boldsymbol{c}-\boldsymbol{c}_N\|^2} + b = 0, \end{aligned} \tag{11}$$

where $(\bar{\boldsymbol{\alpha}},b)\in\mathbb{R}^{\bar{m}+1}$ is the unique solution to (10). This surface discriminates native proteins against decoys. Besides the rectangular kernel matrix, the use of 2-norm for the error $\boldsymbol{\xi}$ and an extra term b^2 in the objective function of (10) distinguish this formulation from conventional support vector machine.

Smooth Newton method

In order to solve equation (10) efficiently, an equivalent unconstrained nonlinear program based on the implicit Lagrangian formulation of (10) was proposed in [45], which can be solved using a fast Newton method. We modified the implicit Lagrangian formulation and obtain the unconstrained nonlinear program for the imbalance RSVM in equation (10). The Lagrangian dual of (10) is now [46]:

$$\min_{\bar{\boldsymbol{\alpha}}\in\mathbb{R}^{\bar{m}}_+} \frac{1}{2}\bar{\boldsymbol{\alpha}}\cdot(Q+\bar{D}_s(K(A,\bar{A})^T K(A,\bar{A})+\boldsymbol{e}\boldsymbol{e}^T\bar{D}_s)\bar{\boldsymbol{\alpha}}-\boldsymbol{e}\cdot\bar{\boldsymbol{\alpha}}, \tag{12}$$

where $Q=I/C\in\mathbb{R}^{\bar{m}\times\bar{m}}$, and $I\in\mathbb{R}^{\bar{m}\times\bar{m}}$ is unit matrix. Note that $\mathbb{R}^{\bar{m}}_+$ is the set of nonnegative $\bar{\boldsymbol{m}}$-vectors. Following [45], an equivalent unconstrained piecewise quadratic minimization problem of the above positively constrained optimization can be derived as follows:

$$\begin{aligned} \min_{\bar{\boldsymbol{\alpha}}\in\mathbb{R}^{\bar{m}}} L(\bar{\boldsymbol{\alpha}}) = \min_{\bar{\boldsymbol{\alpha}}\in\mathbb{R}^{\bar{m}}} \frac{1}{2}\bar{\boldsymbol{\alpha}}\cdot Q\bar{\boldsymbol{\alpha}}-\boldsymbol{e}\cdot\bar{\boldsymbol{\alpha}}+ \\ \frac{1}{2}\beta(\|(-\beta\bar{\boldsymbol{\alpha}}+Q\bar{\boldsymbol{\alpha}}-\boldsymbol{e})_+\|^2-\|Q\bar{\boldsymbol{\alpha}}-\boldsymbol{e}\|^2). \end{aligned} \tag{13}$$

Here, β is a sufficiently large but bounded positive parameter to ensure that the matrix $\beta I-Q$ is positive definite, where $I\in\mathbb{R}^{\bar{m}\times\bar{m}}$ is a unit matrix, and the plus function $(\cdot)_+$ replaces negative components of a vector by zeros. This unconstrained piecewise quadratic problem can be solved by the Newton method in a finite number of steps [45]. The Newton method requires the information of the gradient vector $\nabla L(\bar{\boldsymbol{\alpha}})\in\mathbb{R}^{\bar{m}}$ and the generalized Hessian $\partial^2 L(\bar{\boldsymbol{\alpha}})\in\mathbb{R}^{\bar{m}\times\bar{m}}$ of $L(\bar{\boldsymbol{\alpha}})$ at each iteration. They can be calculated using the following formulae [45]:

$$\begin{aligned} \nabla L(\bar{\boldsymbol{\alpha}}) &= (Q\bar{\boldsymbol{\alpha}}-\boldsymbol{e})+\frac{1}{\beta}(Q-\beta I)((Q-\beta I)-\boldsymbol{e})_+-\frac{1}{\beta}Q(Q\bar{\boldsymbol{\alpha}}-\boldsymbol{e}) \\ &= \frac{(\beta I-Q)}{\beta}((Q\bar{\boldsymbol{\alpha}}-\boldsymbol{e})-((Q-\beta I)\bar{\boldsymbol{\alpha}}-\boldsymbol{e})_+), \end{aligned} \tag{14}$$

and

$$\partial^2 L(\bar{\boldsymbol{\alpha}}) = \frac{\beta I-Q}{\beta}(Q+diag((Q-\beta I)\bar{\boldsymbol{\alpha}}-\boldsymbol{e})_*(\beta I-Q)), \tag{15}$$

where $diag(\cdot)$ denotes a diagonal matrix and $(\boldsymbol{\alpha})_*$ denotes the step function, *i.e.*, $(\alpha_i)_*=1$ if $\alpha_i>0$; and $(\alpha_i)_*=0$ if $\alpha_i\leq 0$.

The main step of the Newton method is to solve iteratively the system of linear equations

$$-\nabla L(\bar{\boldsymbol{\alpha}}^i)+\partial^2 L(\bar{\boldsymbol{\alpha}}^i)(\bar{\boldsymbol{\alpha}}^{i+1}-\bar{\boldsymbol{\alpha}}^i)=0, \tag{16}$$

for the unknown vector $\bar{\boldsymbol{\alpha}}^{i+1}$ with given $\bar{\boldsymbol{\alpha}}^i$.

We present below the algorithm, whose convergence was proved in [45]. We denote $\partial^2 L(\bar{\boldsymbol{\alpha}}^i)^{-1}$ as the inverse of the Hessian $\partial^2 L(\bar{\boldsymbol{\alpha}}^i)$.

Start with any $\bar{\boldsymbol{\alpha}}^0\in\mathbb{R}^{\bar{m}}$. For $i=0,1...$:

(i) Stop if $\nabla L(\bar{\boldsymbol{\alpha}}^i-\partial^2 L(\bar{\boldsymbol{\alpha}}^i)^{-1}\nabla L(\bar{\boldsymbol{\alpha}}^i))=0$.

(ii) $\bar{\boldsymbol{\alpha}}^{i+1}=\bar{\boldsymbol{\alpha}}^i-\lambda_i\partial^2 L(\bar{\boldsymbol{\alpha}}^i)^{-1}\nabla L(\bar{\boldsymbol{\alpha}}^i)=\bar{\boldsymbol{\alpha}}^i+\lambda_i\boldsymbol{d}_i$, where $\lambda_i=\max\{1,\frac{1}{2},\frac{1}{4},\cdots\}$ is the Armijo step size [47] such that

$$L(\bar{\boldsymbol{\alpha}}^i)-L(\bar{\boldsymbol{\alpha}}^i+\lambda_i\mathbf{d}^i)\geq-\delta\lambda_i\nabla L(\bar{\boldsymbol{\alpha}}^i)\cdot\boldsymbol{d}^i, \tag{17}$$

for some $\delta\in(0,\frac{1}{2})$, and $\boldsymbol{d}^i$ is the Newton direction

$$\boldsymbol{d}^i=\bar{\boldsymbol{\alpha}}^{i+1}-\bar{\boldsymbol{\alpha}}^i=-\partial^2 L(\bar{\boldsymbol{\alpha}}^i)^{-1}\nabla L(\bar{\boldsymbol{\alpha}}^i), \tag{18}$$

obtained by solving (16).

(iii) $i=i+1$. Go to (**i**).

Computational Experiments

Determination of count vector by alpha shape

Since protein molecules are formed by thousands of atoms, their shapes are complex. In this study, we use the count vector $\boldsymbol{c}$ of pairwise contact interactions derived from the edge simplexes of the alpha shape of a protein structure, where only nearest neighbor atoms in physical contacts are identified. The advantages of this approach are elaborated in [48]. We refer to references [49,50] for further theoretical and computational details.

Relationship between number of contacts and length of protein

We found that there is a relationship between the number of total contacts of a protein and the length of the protein. A linear regression on the relationship between the number of total contacts and the length of the protein gives the following equation,

$$N_{contacts}=3.090\cdot L_{protein}-76.182, \tag{19}$$

where $N_{contacts}$ is the number of contacts for a protein, and $L_{protein}$ is the number of the protein residues. To eliminate the influence of the length of protein, we normalize the number of contacts for each type of pair-wise contact of a protein using equation (19).

Generating sequence decoys by threading

We followed Maiorov and Crippen [51] and used gapless threading to generate a large number of decoys for a simplified test of protein design. We threaded the sequence of a larger protein through the structure of a smaller protein, and obtained sequence decoys by mounting a fragment of the native sequence from the large protein to the full structure of the small protein. We therefore had a set of sequence decoys $(\boldsymbol{s}_N,\boldsymbol{a}_D)$ for each native protein $(\boldsymbol{s}_N,\boldsymbol{a}_N)$ (Fig 1). Because all native contacts were retained, such sequence decoys are quite challenging. This is unlike folding decoys generated by gapless threading [32].

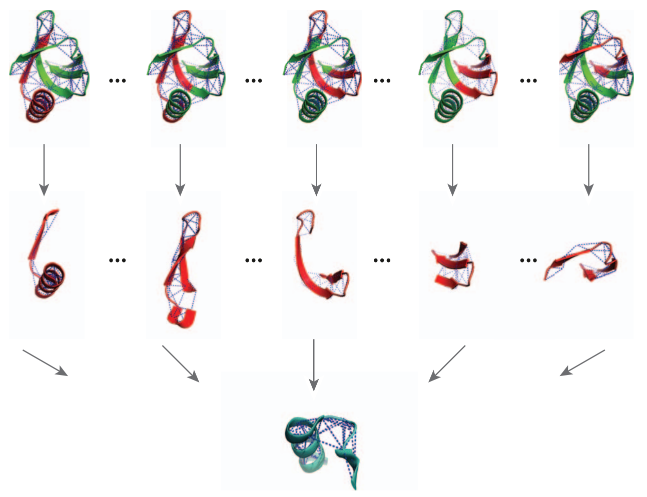
Figure 1. Decoy generation by gapless threading. Sequence decoys can be generated by threading the sequence of a larger protein to the structure of an unrelated smaller protein.
doi:10.1371/journal.pone.0104403.g001

Dataset

We used the list of 1,515 protein chains compiled from the PISCES server [52]. Protein chains in this data set have pairwise sequence identity $<20\%$, With its structural resolution by crystallography and has a resolution $\leq$ 1.6 Å, and the R-factor $\leq$ 0.25. We removed incomplete proteins (i.e. those with missing residues), and proteins with uncertain residues (those denoted as ASX, GLX, XLE, and XAA). We further removed proteins with less than 46 and more than 500 amino acids. In addition, we removed protein chains with more than 30% extensive inter-chain contacts. The remaining set of 1,228 proteins are then randomly divided into two sets. One set includes 800 proteins and the other one includes 428 proteins. Using the sequence threading method, we generated 36,823,837 non-protein decoys, together with 800 native proteins as the training set, and 11,144,381 decoy non-proteins with 428 native proteins as the test set.

Selection of matrix A for iterative training

We used only a subset of the 36 million decoys and native structures so they could fit into the computer memory during training. These structures formed the data matrix A, which was used to construct the kernel matrix $K(A,\bar{A})$. We used a heuristic iterative approach to construct matrices A and $\bar{A}$ during each iteration.

Initially, we randomly selected 10 decoys from the set of decoys $\mathcal{D}_j$ for each of the j-th native protein. We have then $m \approx 8{,}000$ decoys for the 800 native proteins. We further chose only 1 decoy from the selected 10 decoys for each native protein j. These 800 decoys were combined with the 800 native proteins to form the initial matrix A. The contact vectors of a subset of 480 native proteins (60% of the original 800 proteins) and 320 decoys (40% of the 800 selected decoys) were then randomly chosen to form $\bar{A}$. An initial fitness function $H(\boldsymbol{c})$ was then obtained using A and $\bar{A}$. The fitness values of all 36 million decoys and the 800 native proteins were then evaluated using $H(\boldsymbol{c})$. We further used two iterative strategies to improve upon the fitness function $H(\boldsymbol{c})$.

[**Strategy 1**] In the i-th iteration, we selected the subset of misclassified decoys from $\mathcal{D}_j$ associated with the j-th native protein and sorted them by their fitness value in descending order, so the misclassified decoys with least violation, namely, negative but smallest absolute values in $H(\boldsymbol{c})$, are on the top of the list. If there is less than 10 misclassified decoys, we add top decoys that were misclassified in the previous iteration for this native protein, if they exist, such that each native protein has 10 decoys.

A new version of the matrix A was then constructed using these 8,000 decoys and the corresponding 800 native proteins. To obtain the updated $\bar{A}$, from these 8,800 contact vectors, we randomly selected 480 native proteins (60%) and 3,200 unpaired decoy non-proteins (40%) to form $\bar{A}$.

The iterative training process was then repeated until there is no improvement in the classification of the 36 million decoys and the 800 native proteins from the training set. Typically, the number of iterations was about 10. In subsequent studies, we experimented with different percentage of selected decoys, ranging from 10% to 100% to examine the effect of the size of $\bar{A}$ on the effectiveness of the fitness function $H(\boldsymbol{c})$.

[**Strategy 2**] In the i-th iteration, we selected the top 10 correctly classified decoys sorted by their fitness value in ascending order for each native protein, namely, those correctly classified decoy with positive but smallest absolute values are selected. These contact vectors of 8,000 selected decoys are combined with the 800 native proteins to form the new data matrix A.

To construct $\bar{A}$, we first selected the most challenging native proteins by taking the top 80 correctly classified native proteins (10%) sorted by their fitness value in descending order, namely, those that are negative but with smallest absolute values in $H(\boldsymbol{c})$. We then randomly took 400 native proteins (50%) from the rest of the native protein set, so altogether we have 480 native proteins (60%). Similarly, we selected the top one decoy that is most challenging from the 10 chosen decoys in A for each native protein, namely, the top decoy that is correctly classified with positive but smallest value of $H(\boldsymbol{c})$. We then randomly selected 3 decoys for each native protein from the remaining decoys in A to obtain 3,200 decoy non-proteins (40%). The matrix $\bar{A}$ is then constructed from the selected 480 native proteins and 3,200 decoy non-proteins. The iterative training process was repeated until there was no improvement in classification of the 36 million decoys and 800 native proteins in the training set. Typically, the number of iteration was about 5.

In the subsequent studies, we evaluated our method with different choices of challenging native proteins. The selection ranges from the top 10% to 60% most challenging native proteins. The choice of the challenging decoys was also varied, where we experimented with choosing the top one to the top four most challenging decoys for each native protein, while the number randomly selected decoys varies from three to zero.

Learning parameters

There are two important parameters: the constant γ in the kernel function $e^{-\gamma\|\boldsymbol{c}_i-\boldsymbol{c}\|^2}$, and the cost factors C, which is used during training so errors on positive examples were adjusted to outweigh errors on negative examples. Our experimentation showed that $\gamma = 5.0 \times 10^{-5}$ and $C = 1.0 \times 10^4$ are reasonable choices.

Running time

The algorithm was implemented in the C language. It called Lapack [53] and used LU decomposition to solve the system of linear equations. It also called an SVD routine to determine the 2-norm of a matrix for calculating $\beta = 1.1(1/C + \|DA - \boldsymbol{e}\|_2^2)$. Once matrices A and $\bar{A}$ were specified, the fitness function $H(\boldsymbol{c})$ can be derived in about 2 hours and 10 minutes on a 2 Dual Core AMD Opteron(tm) Processors of 1,800 MHz with 4 Gb memory for an

A of size $8{,}800 \times 210$ and an $\bar{A}$ of size $3{,}680 \times 210$. The evaluation of the fitness of 14 million decoys took 2 hours and 10 minutes using 144 CPUs of a Linux cluster (2 Dual Core AMD Opteron(tm) Processors of 1.8 GHz with 2 Gb memory for each node). Because of the large size of the data set, the bottleneck in computation is disk IO.

Results

Performance in discrimination

We used the set of 428 natives proteins and 11,144,381 decoys for testing the designed fitness function. We took the sequence $\boldsymbol{a}$ for a protein such that $\boldsymbol{c}=f(\boldsymbol{s}_N,\boldsymbol{a})$ has the best fitness value as the predicted sequence. If it is not the native sequence $\boldsymbol{a}_N$, then the design failed and the fitness function did not work for this protein.

Sequence decoys obtained by gapless threading were quite challenging, since all native contacts of the protein structures were maintained, and decoy sequences were from real proteins. In a previous study, we showed that no linear fitness function can be found that would succeed in the challenging task of identifying all 440 native sequences in the training set [32]. Because we are unaware of any other development of design fitness functions amenable for high-throughput tests, and frequently no distinctions were made between protein folding potential and protein design fitness function, we compared our fitness function with several well-established scoring functions developed for protein folding.

We also use the F_β score to evaluate the performance of predictions. F_β is defined as:

$$F_\beta=(1+\beta^2)\frac{Precision \times Recall}{\beta^2 \times Precision + Recall},$$

where TP is the number of true positives, FP the number of false positives, FN the number of false negatives, $Precision$ is calculated as $\frac{TP}{(TP+FP)}$, and $Recall$ is calculated as $\frac{TP}{(TP+FN)}$. When $\beta > 1$, recall is emphasized over precision. When $\beta < 1$, precision is emphasized over recall. Because of the imbalanced nature of the data set with much more decoys than native proteins, we assign more weight on the small set of native proteins, with β set to 10. The F_β scores are than calculated accordingly.

Here we succeeded in obtaining a simplified nonlinear fitness function for protein design that are capable of discriminating 796 of the 800 native sequences (Table 1). It also succeeded in correctly identifying 95% (408 out of 428) of the native sequences in the independent test set. Results for other methods were taken from literature obtained using much smaller and less challenging data set. Overall, the performance of our method is better than results obtained using the optimal linear scoring function taken as reported in [26] and in [28], which succeeded in identifying 78% (157 out of 201) and 71% (143 out of 201) of the test set, respectively. Our results are also better than the Miyazawa-Jernigan statistical potential [34] (success rate 58%, 113 out of 201). This performance is also comparable with a more complex nonlinear fitness function, with $>5{,}000$ terms reported in [32], which succeeded with a correct rate of 91% (183 out of 201).

Effect of the size of the basis set $\bar{A}$ using Strategy 1

The matrix $\bar{A}$ contains both proteins and decoys from A and its size is important in discrimination of native proteins from decoys. In our fitness function, Gaussian kernels centered around these selected contact vectors were used as basis set to interpolate the global landscape of protein design.

We examined the effects of different sizes of $\bar{A}$ using Strategy 1. For a data matrix A consisting of 800 native proteins and 8,000 sequence decoys derived following the procedure described earlier, we tested different choice of $\bar{A}$ on the performance of discrimination. With the data matrix A, we fixed the selection of the 480 native proteins (60%), and experimented with random selection of different number of decoys, ranging from 800 (10%) to 8,000 (100%) to form different $\bar{A}$s.

The results of classifying both the training set of 800 native proteins with 36 million decoys and the test set of 428 native proteins with 11 million decoys are shown in Table 2. When 60% (480) native proteins and 100% (8,000) decoys are included, there are only 5 native proteins misclassified in the training set and 24 native proteins in the test set.

Effect of the size of the pre-selection of dataset using Strategy 2

We examined the effects of different choices in constructing matrix $\bar{A}$ using Strategy 2. We varied our selection of the most challenging native proteins from the top 10% to 60%, and varied selection of the most challenging decoys from the top one to the top four decoys for each native protein, as describe earlier. Results are shown in Table 3. We found that the performance of the discrimination of both the training set and test set have little changes when either native proteins selection rate is changed from 10% to 60%, or decoys selection rate is changed from the top 1 to the top 4. Overall, these results suggest that for the blind test developed here, a fitness function with good discrimination can be achieved with about 480 native proteins and 3,200 decoys, along

Table 1. The number of misclassification compared with other methods.

Method	Training set	Training set	Test set	Test set
	800 / 36 M	**440 / 14 M**	**428 / 11 M**	**201 / 3 M**
Nonlinear function	4 / 988	NA	20 / 218	NA
Tobi *et. al.*	NA	192 / 39,583	NA	44 / 53,137
Bastolla *et. al.*	NA	134 / 47,750	NA	58 / 29,309
Miyazawa & Jernigan	NA	173 / 229,549	NA	87 / 80,716

The number of misclassification using simplified nonlinear fitness function, optimal linear scoring function taken as reported in [26,28], and Miyazawa-Jernigan statistical potential [34] for both native proteins and decoys (separated by "/") in the test set and the training set. The simplified nonlinear function is formed using a basis set of 3,680 (480 native+3,200 decoy) contact vectors derived using Strategy 2.
doi:10.1371/journal.pone.0104403.t001

Table 2. Effects of the size of basis set $\bar{A}$ on performance of discrimination using Strategy 1.

		Training set		Test set	
Select decoys rate	Iteration	Native / Decoy	F_β	Native / Decoy	F_β
		800 / 36 M		428 / 11 M	
0%	4	21 / 1,374	0.958	26 / 387	0.931
2%	5	19 / 1,029	0.964	27 / 219	0.933
5%	5	17 / 1,303	0.963	21 / 317	0.944
8%	5	13 / 1,246	0.969	23 / 274	0.941
10%	5	14 / 922	0.972	24 / 216	0.940
20%	6	16 / 902	0.969	28 / 250	0.930
30%	6	10 / 1,037	0.975	29 / 304	0.926
40%	10	16 / 812	0.970	27 / 199	0.933
50%	10	13 / 1,112	0.971	25 / 269	0.936
60%	12	15 / 802	0.972	27 / 237	0.932
70%	9	13 / 947	0.973	24 / 256	0.939
80%	8	11 / 1,078	0.973	28 / 278	0.929
90%	9	12 / 690	0.977	27 / 170	0.934
100%	5	5 / 2,681	0.962	24 / 609	0.931

The number of misclassifications of both native proteins and decoys (separated by "/") with select native proteins rate 60% in both training set and test set are listed. Misclassifications as well as the F_β scores in two tests using different number of native proteins and decoys are listed (see text for details).
doi:10.1371/journal.pone.0104403.t002

with 400 pre-selected native proteins and 800 pre-selected top-1 decoys. Our final fitness function used in Table 1 is constructed using a basis set of 3,680 contact vectors. We also observed that the average number of iterations is about 5 using Strategy 2, which is much faster than Strategy 1.

We found that using Strategy 2 (Table 3) leads to overall better performance compare to using Strategy 1 (Table 2). Specifically, the fitness function formed by pre-selecting the top 1 decoys and top 50% native proteins using Strategy 2 works well to discriminating native proteins from decoys.

Table 3. Effect of the size of the pre-selection of dataset using Strategy 2.

			Training Set		Test set	
Pre-select native proteins top	Pre-select decoys top	Iteration	Native / Decoy	F_β	Native / Decoy	F_β
			800 / 36 M		428 / 11 M	
0%	1	6	8 / 1,010	0.978	25 / 212	0.938
2%	1	5	5 / 1,079	0.981	24 / 266	0.939
5%	1	5	5 / 1,038	0.981	24 / 247	0.939
8%	1	5	5 / 1,093	0.981	24 / 249	0.939
10%	1	5	5 / 997	0.982	24 / 242	0.939
20%	1	6	9 / 625	0.981	26 / 174	0.936
30%	1	6	9 / 689	0.980	24 / 211	0.940
40%	1	6	8 / 869	0.980	25 / 218	0.937
50%	1	5	4 / 988	0.983	20 / 218	0.949
60%	1	5	6 / 1,039	0.980	24 / 280	0.938
10%	1	5	5 / 997	0.982	24 / 242	0.939
10%	2	5	6 / 1,270	0.977	22 / 372	0.941
10%	3	7	9 / 934	0.978	22 / 247	0.944
10%	4	5	5 / 1,071	0.981	24 / 210	0.944

Test results using Strategy 2 with different sizes of the pre-selected native proteins, which range from 0% to 60% while the pre-selected decoys are fixed as the top 1 level, and with different pre-selected decoys, which ranges from the top 1 s to the top 4 s while the pre-selected native proteins are fixed at 10%. Misclassifications as well as the F_β scores in two tests using different number of native proteins and decoys are listed (see text for details).
doi:10.1371/journal.pone.0104403.t003

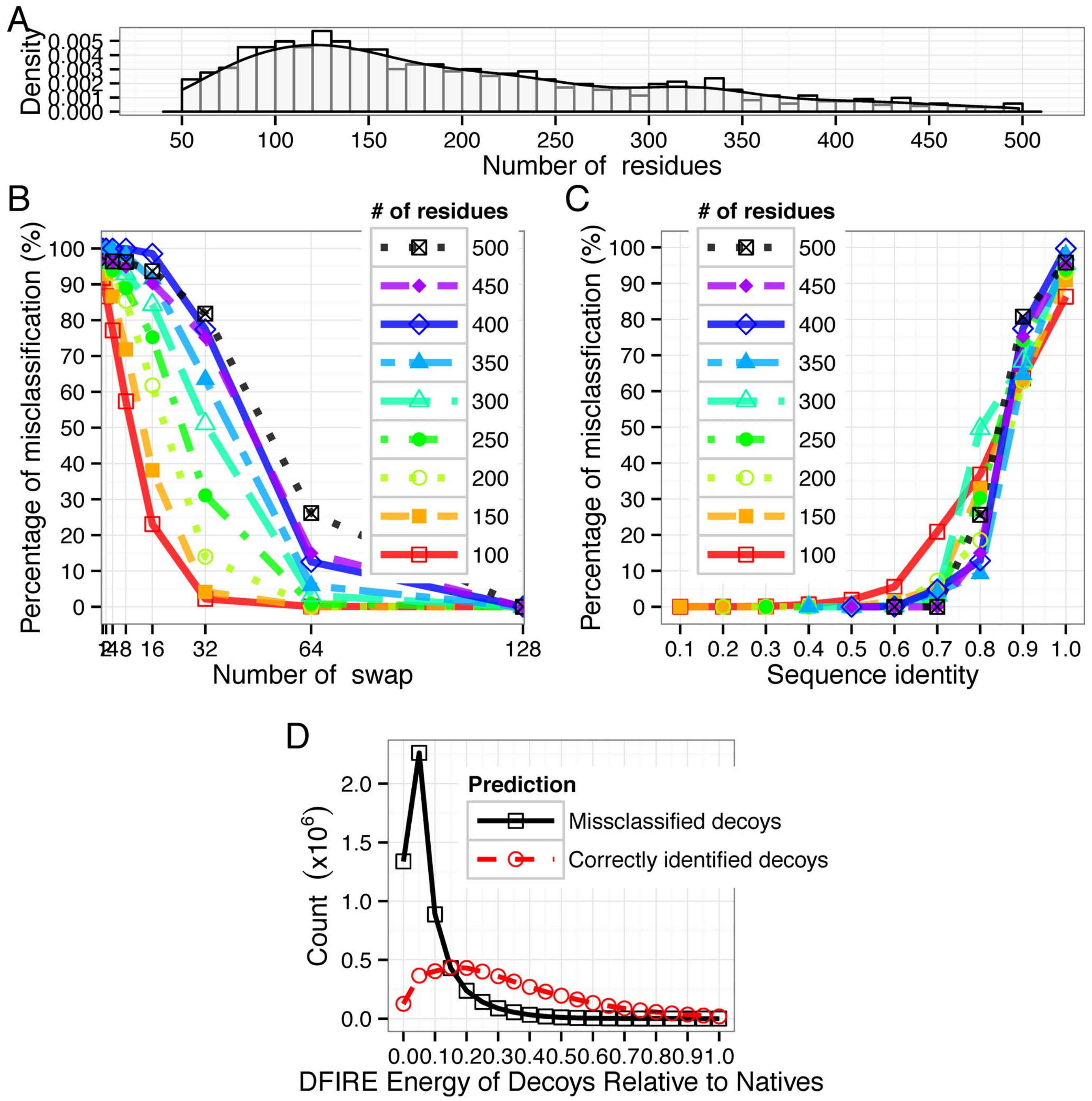

Figure 2. Discriminating a different decoy set using the nonlinear fitness function. Sequence decoys in this set are generated by swapping residues at different positions. (A). The length distribution of the 1,227 native proteins in the set; (B). The relationship between the number of swaps N and the percentage of misclassified decoys grouped by protein length binned with a width of 50 residues shown in different curves. (C). The relationship between the sequence identity binned with width 0.1 and the percentage of misclassification grouped by protein length shown in different curves. The fitness function was derived using strategy 2, with top 50% pre-selected native proteins, and top 1 pre-selected decoys. (D). Misclassified sequence decoys have overall lower DFIRE energy values than correctly classified sequence decoys and therefore are more native-like. The x-axis is the net DFIRE energy difference of decoys to native proteins, and the y-axis is the number count of decoys at different net DFIRE energy differences. The solid black line represents decoys misclassified by our fitness function and the dashed red line represents decoys correctly classified by our fitness function.
doi:10.1371/journal.pone.0104403.g002

Furthermore, our method is robust. The overall performance using either Strategy 1 or Strategy 2 is stable when decoy selection rate changes from 5% to 90%. Using the F_β score as the criterion, we found that using Strategy 2 gives significantly more accurate result than using Strategy 1.

Discrimination against a different decoy set

To further assess our fitness function, we examine how well decoys generated by a different approach can be discriminated using our nonlinear fitness function. We selected 799 training proteins and 428 test proteins for this test. Figure 2A shows the

Table 4. 20 native proteins in the test set are misclassified using Strategy 2.

Molecular name		Classification	Ligand(s)	PDBID	Chain	Fitness value
Catalase	◦	Oxidoreductase	1 HEM and 3 SO_4	1gwe	A	0.1085
Streptavidin	◦	Biotin binding	1 BTN and 2 GOL	2f01	A	0.1407
Acutohaemonlysin	◦	Toxin	2 IPA	1mc2	A	0.1728
Endonuclease I	◦	Hydrolase	1 Mg and 2 Cl	2pu3	A	0.1900
cytochrome c, putative	◦	Electron transport	2 SO_4, 1 Na and 2 HEM	2czs	A	0.2664
Cytochrome F	◦	Electron transport	1 HEME C	1e2w	A	0.6023
Bowman-Birk type trypsin inhibitor	□	Hydrolase inhibitor	None	2fj8	A	0.8463
Uncharacterized protein with erredoxin-like fold	◦	Structural genomics Unknown function	1 Unkown ligand	3e8o	A	1.1592
General secretion pathway protein G	◦	Protein transport	1 Zn	1t92	A	1.3175
ARF GTPase-activating protein git1	Δ	Signaling protein	None	2w6a	A	1.6581
Cystatin B	Δ	Protein binding	None	2oct	A	1.8043
SNAP-25A	□	Transport protein	None	1n7s	D	1.9074
Lin2189 protein	◦	Structural genomics Unknown function	2 GOL	3b49	A	2.0142
Fibritin	□	Chaperone	None	2ibl	A	2.1211
Oxalate oxidase 1	◦	Oxidoreductase	1 Mn, 1 GLV	2et1	A	2.9975
Alpha-2-macroglobulin receptor-associated protein	◦	Lipid transport endocytosis chaperone	2 Ca, 1 Na and 3 MPD	2fcw	B	3.5660
Recombination endonuclease VII	◦	Plasma protein	1 Zn and 7 SO_4	1e7l	A	3.7397
Hypothetical protein	◦	Isomerase	1 BEZ	1gyx	A	4.2697
YDCE	Δ					
Syntaxin 1a	Δ	Transport protein	None	1n7s	B	5.0204
Bacteriophage t4 short tail fibre	◦	Structural protein	1 CIT, 2 SO_4 and 1 Zn	1ocy	A	8.0264

The number of ligands bound to the protein are listed. The molecules are sorted by the fitness value. 14 of them (marked by "◦") have ligand(s) bound to the protein. 4 of them (marked by "Δ") have $>20\%$ contacts due to inter chain interactions. The fitness function definitively failed for only 3 proteins (marked by "□"). For the remaining 17 proteins, the contacts between organic compounds and metal ions with the protein and inter chain interactions may provide additional stability beyond the intra-residue interactions captured in the descriptors.
doi:10.1371/journal.pone.0104403.t004

length distribution of these 1,227 proteins. To generate these new decoys, we fixed the composition of each of these proteins and permute its sequence by carrying out N swaps between random residues, with $N=1,2,4,8,16,32,64$, and 128. The resulting decoys all have the same amino acid composition as the original native proteins, but have progressively more point mutations. We generate 1,000 random sequence decoys at each swap N for each protein. We call this Decoy Set 2.

Our results show that as expected, the number of misclassified decoys decreases rapidly as the number of swaps increases. When N increase from 1 to 32, The percentage of misclassified decoys for protein of length ≤ 250 is about 30% or less. Less than 30% of the decoys of all lengths are misclassified when $N=64$, with the rate of misclassification much smaller than 10% among those with length <350 (Fig 2B). Only 62 decoys are misclassified among 1,227,000 decoys when $N\geq 128$ (Fig 2B).

It is informative to examine the number of misclassified decoys and the sequence identity of the decoys with their corresponding native proteins at different protein lengths. Figure 2C shows that the percentage of misclassified decoys decreases rapidly with the sequence identity to the native proteins. When decoys have a sequence identity of $\leq 60\%$ with the native protein, $<10\%$ of the decoys are misclassified, and all decoys can be discriminated against at 40% identity for proteins of length ≥ 150. For proteins of length ≤ 150, most decoys with $\leq 50\%$ sequence identity can be corrected discriminated against. These observations are consistent with current understanding of protein structures, where most proteins with $\geq 70\%$ sequence identity belong to the same family [54], and these with $\geq 30\%$ sequence identity have similar structure [55].

To examine whether misclassified decoy sequences are actually more native-like and therefore more likely to potentially adopt the correct structures than those correctly classified as non-natives, we selected $5.5M$ misclassified decoys and $4.3M$ correctly classified decoys from all decoys in Decoy Set 2, and examined their energy values. We use the DFIRE energy function that was independently developed in [56,57]. These decoys all have values of net DFIRE energy difference of decoys to native proteins within the interval of [0.0, 1.0] kcal/mol. Our results (Fig 2D) show that overall, misclassified decoys have much lower average DFIRE energy values, indicating that they are potentially more native-like than those correctly classified as decoys.

Discussion

In this study, we have developed a simplified nonlinear kernel function for fitness landscape of protein design using a rectangular kernel and a fast Newton method. The results in a blind test are encouraging. They suggest that for a simplified task of designing simultaneously 428 proteins from a set of 11 million decoys, the search space of protein shape and sequence can be effectively parametrized with just about 3,680 basis set of contact vectors. It is likely that the choice of matrix A is important. We showed that once A is carefully chosen, the overall design landscape is not overly sensitive to the specific choice of the basis set contact vectors for $\bar{A}$.

The native protein list in both training and test sets come from the PISCES server, which has the lowest pair-wise identity (20%), finer resolution cutoff (1.6 Å), and lower R-factor cutoff (0.25). This native dataset is better than previous study [32] dataset derived from the WHATIF database, which has looser constraints: pair-wise sequence identity $<30\%$, resolution cutoff <2.1 Å, and R-factor cutoff <2.1. We compared our results with classic studies of Tobi *et. al.* [26], Bastolla *et. al.* [28] and Miyazawa and Jernigan [34]. Although the training set and test set are different, we observed that our simplified nonlinear function detected 95% (208) native proteins from 11 million decoys and only misclassified 218 decoys as native proteins, which outperformed Tobi *et. al.* [26] (78% correct rate for native proteins, 53,137 misclassification for decoys), Bastolla *et al.* [10] (71% correct rate for native proteins, 29,309 misclassification for decoys), and Miyazawa and Jernigan [34] methods (57% correct rate for native proteins, 80,716 misclassification for decoys) on much smaller blind test set of 201 native proteins and 3 million decoys.

As protein length is linearly correlated with the total number of contacts, we found that length corrections is important for improving fitness function. For example, the rate of misclassification is 7.2% in an earlier study without length correction (14 out 494 natives) [58], while this rate is now improved to 4.7% in the current study with length correction (20 out of 428 misclassified).

We developed two strategies to search for improving fitness landscape. Strategy 1 mostly uses misclassified decoys in the next iteration of construction of matrix A. On average, 10 iterations is necessary to arrive at a good fitness function, which has excellent performance of only 5 misclassification for the training data set. The misclassification rate in the test set is comparable to other fitness function [26,28,34]. Strategy 2 selected the most challenging decoys by the fitness value landscape in the matrix A for the next iteration. We pre-selected certain percentage of the number of native proteins and certain number of decoys before generating the basis set matrix $\bar{A}$. Overall, Strategy 2 performs better than Strategy 1, not only in reducing both native proteins and decoys misclassifications in the blind test set, but also can speed up the search process in deriving the final fitness function with the number of iteration reduced from 10 to 5 iterations. With Strategy 2, the updated fitness landscape is only adjusted by challenging decoys, it can identify the most challenging decoys and native proteins, leading to improved the fitness landscape in the next iteration.

Our final fitness landscape can correctly classify most of the native proteins, except 4 proteins (1ft5 chain A, 1gk9 chain A, 2p0s chain A, 2qud chain A) in the training set and 20 proteins in the test set (Table. 4). Among misclassified proteins, 4 of which have $>20\%$ contacts due to inter chain interactions. In addition, 14 misclassified proteins contain metal ions and organic compounds. We note that the interactions between these organic compounds, metal ions and rest of the protein are not reflected in the protein description. It is likely that substantial unaccounted interactions with other protein chains, DNA, or co-factors contributed to the misclassifications. The conformations of these proteins may be different upon removal of these contacts. Altogether, 21 of the 24 misclassified proteins have explanations, and the fitness function truly failed only for 3 proteins.

In protein folding studies, it is well known that contact maps of decoys formed by gapless threading have considerable higher energy than the native contact map, and these protein folding decoys are not as challenging as decoys generated by other methods such as Monte Carlo simulation. Results showed in Figure 2 demonstrated that these sequence decoys are challenging, and our nonlinear fitness function works well.

The representation of protein structures will likely have important effects on the success of protein design. The approach of the reduced nonlinear function is general and applicable when alternative representations of protein structures are used, *e.g.*, adding solvation terms, including higher-order interactions.

Conclusions

We showed that a simplified nonlinear fitness function for protein design can be can be obtained using a simplified nonlinear kernel function via a finite Newton method. We used a rectangular kernel with a basis set of native proteins and decoys chosen *a priori*.

We succeeded in predicting 408 out of the 428 (95%) native proteins and misclassified only 218 out of 11 million decoys in a large blind test set. Although the test sets used is different, as other method were based on relatively small (201 native proteins and 3 million decoys) blind test set. Our result outperforms statical linear scoring function (87 out of the 201 misclassifications, 57% correct rate) and optimized linear function (between 44 and 58 misclassifications out of the 201, 78% and 71% correct rate). The performance is also comparable with results obtained from a far more complex nonlinear fitness function with $>5{,}000$ terms (18 misclassifications, 91% correct rate). Our results further suggest that for the task of global sequence design of 428 selected proteins, the search space of protein shape and sequence can be effectively parametrized with just about 3,680 carefully chosen basis set of native proteins and non-native protein decoys.

The rectangle kernel matrix with a finite Newton method works well in constructing fitness landscape. In addition, we showed that the overall landscape is not overly sensitive to the specific choice of the dataset.

Overall, our strategy of reduced kernel can be generalized to constructing other types of fitness function.

Author Contributions

Conceived and designed the experiments: YX CH YD JL. Performed the experiments: YX CH. Analyzed the data: YX CH JL. Contributed reagents/materials/analysis tools: YX CH. Wrote the paper: YX CH YD JL.

References

1. Desmet J, De Maeyer M, Hazes B, Lasters I (1992) The dead-end elimination theorem and its use in protein side-chain positioning. Nature 356: 539–542.
2. Yue K, Dill KA (1992) Inverse protein folding problem: designing polymer sequences. Proceedings of the National Academy of Sciences of the United States of America 89: 4163–4167.

3. Shakhnovich EI, Gutin AM (1993) Engineering of stable and fast-folding sequences of model proteins. Proceedings of the National Academy of Sciences of the United States of America 90: 7195–7199.
4. Li H, Helling R, Tang C, Wingreen N (1996) Emergence of preferred structures in a simple model of protein folding. Science (New York, NY) 273: 666–669.
5. Deutsch J, Kurosky T (1996) New Algorithm for Protein Design. Physical review letters 76: 323–326.
6. Dahiyat BI, Mayo SL (1997) De novo protein design: fully automated sequence selection. Science (New York, NY) 278: 82–87.
7. Kleinberg JM (1999) Efficient algorithms for protein sequence design and the analysis of certain evolutionary fitness landscapes. Journal of computational biology : a journal of computational molecular cell biology 6: 387–404.
8. Hill RB, Raleigh DP, Lombardi A, DeGrado WF (2000) De novo design of helical bundles as models for understanding protein folding and function. Accounts of chemical research 33: 745–754.
9. Siegel JB, Zanghellini A, Lovick HM, Kiss G, Lambert AR, et al. (2010) Computational design of an enzyme catalyst for a stereoselective bimolecular Diels-Alder reaction. Science (New York, NY) 329: 309–313.
10. Bolon DN, Mayo SL (2001) Enzyme-like proteins by computational design. Proceedings of the National Academy of Sciences of the United States of America 98: 14274–14279.
11. Röthlisberger D, Khersonsky O, Wollacott AM, Jiang L, DeChancie J, et al. (2008) Kemp elimination catalysts by computational enzyme design. Nature 453: 190–195.
12. Jiang L, Althoff EA, Clemente FR, Doyle L, Röthlisberger D, et al. (2008) De novo computational design of retro-aldol enzymes. Science (New York, NY) 319: 1387–1391.
13. Lazar GA, Dang W, Karki S, Vafa O, Peng JS, et al. (2006) Engineered antibody Fc variants with enhanced effector function. Proceedings of the National Academy of Sciences of the United States of America 103: 4005–4010.
14. Joachimiak LA, Kortemme T, Stoddard BL, Baker D (2006) Computational design of a new hydrogen bond network and at least a 300-fold specificity switch at a protein-protein interface. Journal of molecular biology 361: 195–208.
15. Shifman JM, Choi MH, Mihalas S, Mayo SL, Kennedy MB (2006) Ca2+/calmodulin-dependent protein kinase II (CaMKII) is activated by calmodulin with two bound calciums. Proceedings of the National Academy of Sciences of the United States of America 103: 13968–13973.
16. Drexler KE (1981) Molecular engineering: An approach to the development of general capabilities for molecular manipulation. Proceedings of the National Academy of Sciences of the United States of America 78: 5275–5278.
17. Pabo C (1983) Molecular technology: Designing proteins and peptides. Nature 301: 200–200.
18. Huang SY, Zou X (2006) An iterative knowledge-based scoring function to predict protein-ligand interactions: I. Derivation of interaction potentials. Journal of computational chemistry 27: 1866–1875.
19. Huang SY, Zou X (2008) An iterative knowledge-based scoring function for protein-protein recognition. Proteins 72: 557–579.
20. Huang SY, Grinter SZ, Zou X (2010) Scoring functions and their evaluation methods for proteinligand docking: recent advances and future directions. Physical chemistry chemical physics : PCCP 12: 12899–12908.
21. Wagner M, Meller JA, Elber R (2004) Large-scale linear programming techniques for the design of protein folding potentials. Mathematical programming 101.
22. Májek P, Elber R (2009) A coarse-grained potential for fold recognition and molecular dynamics simulations of proteins. Proteins 76: 822–836.
23. Ravikant DVS, Elber R (2010) PIE-efficient filters and coarse grained potentials for unbound protein-protein docking. Proteins 78: 400–419.
24. Miyazawa S, Jernigan RL (1985) Estimation of Effective Interresidue Contact Energies From Protein Crystal-Structures - Quasi-Chemical Approximation. Macromolecules 18: 534–552.
25. Vendruscolo M, Najmanovich R, Domany E (2000) Can a pairwise contact potential stabilize native protein folds against decoys obtained by threading? Proteins 38: 134–148.
26. Tobi D, Shafran G, Linial N, Elber R (2000) On the design and analysis of protein folding potentials. Proteins 40: 71–85.
27. Bastolla U, Vendruscolo M, Knapp EW (2000) A statistical mechanical method to optimize energy functions for protein folding. Proceedings of the National Academy of Sciences of the United States of America 97: 3977–3981.
28. Bastolla U, Farwer J, Knapp EW, Vendruscolo M (2001) How to guarantee optimal stability for most representative structures in the Protein Data Bank. Proteins 44: 79–96.
29. Jacak R, Leaver-Fay A, Kuhlman B (2012) Computational protein design with explicit consideration of surface hydrophobic patches. Proteins 80: 825–838.
30. Pokala N, Handel TM (2005) Energy functions for protein design: adjustment with protein-protein complex affinities, models for the unfolded state, and negative design of solubility and specificity. Journal of molecular biology 347: 203–227.
31. Liang S, Grishin NV (2004) Effective scoring function for protein sequence design. Proteins 54: 271–281.
32. Hu C, Li X, Liang J (2004) Developing optimal non-linear scoring function for protein design. Bioinformatics (Oxford, England) 20: 3080–3098.
33. Li Z, Yang Y, Zhan J, Dai L, Zhou Y (2013) Energy functions in de novo protein design: current challenges and future prospects. Annual review of biophysics 42: 315–335.
34. Miyazawa S, Jernigan RL (1996) Residue-residue potentials with a favorable contact pair term and an unfavorable high packing density term, for simulation and threading. Journal of molecular biology 256: 623–644.
35. Mintseris J, Weng Z (2003) Atomic contact vectors in protein-protein recognition. Proteins 53: 629–639.
36. Tanaka S, Scheraga HA (1976) Medium- and Long-Range Interaction Parameters between Amino Acids for Predicting Three-Dimensional Structures of Proteins. Macromolecules 9: 945–950.
37. Vendruscolo M, Domany E (1998) Pairwise contact potentials are unsuitable for protein folding. The Journal of chemical physics 109: 11101.
38. Lu H, Skolnick J (2001) A distance-dependent atomic knowledge-based potential for improved protein structure selection. Proteins 44: 223–232.
39. Edelsbrunner H (1987) Algorithms in combinatorial geometry. Springer Verlag.
40. Vapnik V (1999) The Nature of Statistical Learning Theory (Information Science and Statistics). Springer, 2nd edition.
41. Burges CJC (1998) A Tutorial on Support Vector Machines for Pattern Recognition. Data mining and knowledge discovery 2: 121–167.
42. Schölkopf B (2002) Learning with kernels: support vector machines, regularization, optimization, and beyond. The MIT Press.
43. Vapnik VN, Chervonenkis AJ (1974) Theory of Pattern Recognition. (1974).
44. Lee YJ, Mangasarian OL (2001) RSVM: Reduced support vector machines. In: Proceedings of the First SIAM International Conference on Data Mining. pp. 1–17.
45. Fung G, Mangasarian OL (2003) Finite Newton method for Lagrangian support vector machine classification. Neurocomputing 55: 39–55.
46. Mangasarian OL (1994) Nonlinear programming. Society for Industrial Mathematics.
47. Nocedal J, Wright SJ (1999) Numerical optimization. Springer Verlag.
48. Li X, Hu C, Liang J (2003) Simplicial edge representation of protein structures and alpha contact potential with confidence measure. Proteins 53: 792–805.
49. Edelsbrunner H (1993) The union of balls and its dual shape. In: Proceedings of the ninth annual symposium on Computational geometry. New York, NY, USA, pp. 218–231.
50. Liang J, Edelsbrunner H, Fu P, Sudhakar P (1998) Analytical shape computation of macromolecules: I. Molecular area and volume through alpha shape. Proteins: Structure Function and Genetics.
51. Maiorov VN, Crippen GM (1992) Contact potential that recognizes the correct folding of globular proteins. Journal of molecular biology 227: 876–888.
52. Wang G, Dunbrack RL (2003) PISCES: a protein sequence culling server. Bioinformatics (Oxford, England) 19: 1589–1591.
53. Anderson E, Bai Z, Bischof C (1999) LAPACK users' guide. Society for Industrial Mathematics.
54. Holm L, Ouzounis C, Sander C, Tuparev G, Vriend G (1992) A database of protein structure families with common folding motifs. Protein science : a publication of the Protein Society 1: 1691–1698.
55. Rost B (1999) Twilight zone of protein sequence alignments. Protein engineering, design & selection: PEDS 12: 85–94.
56. Yang Y, Zhou Y (2008) Ab initio folding of terminal segments with secondary structures reveals the fine difference between two closely related all-atom statistical energy functions. Protein science 17: 1212–1219.
57. Yang Y, Zhou Y (2008) Specific interactions for ab initio folding of protein terminal regions with secondary structures. Proteins 72: 793–803.
58. Hu C, Li X, Liang J (2003) On design of optimal nonlinear Kernel potential function for protein folding and protein design. arXivorg.