Министерство образования и науки Российской Федерации
Московский физико-технический институт (государственный университет)
Факультет управления и прикладной математики
Вычислительный центр им. А. А. Дородницына РАН
Кафедра «Интеллектуальные системы»

Карасиков Михаил Евгеньевич

Построение ранжирующей функции для прогнозирования третичной структуры белка

03.04.01 — Прикладные математика и физика

Выпускная квалификационная работа магистра

Научный руководитель:
д. ф.-м. н. Стрижов Вадим Викторович

Москва
2017

Построение ранжирующей функции для прогнозирования третичной структуры белка

Карасиков Михаил Евгеньевич

Аннотация

Дизайн белков ставит задачу определения первичной структуры белка с заданными свойствами и является одной из труднейших задач вычислительной биологии. В формальной постановке данная задача упрощается до задачи обратного фолдинга, которая ставится следующим образом. Найти первичную структуру белка, соответствующую третичной структуре с заданным скелетом. В данной работе проводится комплексное исследование задачи обратного фолдинга и других задач вычислительной биологии. Придерживаясь гипотезы о том, что белок сворачивается в состояние минимальной энергии, задача обратного фолдинга формулируется как задача комбинаторной оптимизации. В работе предлагается метод оценки качества белковых структур, позволяющий свести задачу в начальной постановке к задаче булевого квадратичного программирования с ограничениями. Проводится анализ методов выпуклой релаксации возникающей NP-трудной задачи и сравнение их с другими методами дискретной оптимизации. Особое внимание уделяется задаче построения скоринговых потенциалов, аппроксимирующих свободную энергию белковых структур, для прогнозирования близости их к целевым структурам. В работе исследуются методы машинного обучения для построения скоринговых потенциалов, и предлагается новый метод оценки качества модельных структур белков по конформациям их скелетов без боковых цепей. Для обучения используются модельные данные. Полученные скоринговые потенциалы являются эффективно вычислимыми парно-сепарабельными по аминокислотам функциями. Проведенный вычислительный эксперимент и сравнение предложенного потенциала с другими потенциалами на реальных и модельных данных показал, что предложенный скоринговый потенциал достигает качества и производительности лучших существующих методов. Разработанный скоринговый потенциал предлагается использовать для отбора наилучших моделей из множества моделей, полученного при решении задачи обратного фолдинга.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Proteins play an important role in fundamental biological processes as formation of new molecules, transport functions, control of chemical reactions, and protecting through binding to specific foreign particles such as viruses and bacteria. This importance has caused a lot of research on their functions and understanding the mechanisms involved in those processes. The function of a protein is defined by its corresponding gene. A particular function of a protein essentially depends on its shape or structure. Hence protein folding plays an essential functional role in living cells and takes an important part in the problem of protein design (Huang *et al.*, 2016). However, its biological investigation requires conducting expensive chemical experiments, which could be replaced with relatively inexpensive and fast methods of computational biology and molecular modeling techniques for prediction of unknown protein structures (Kmiecik *et al.*, 2016).

1.2 Problems of structural biology

Protein is a polypeptide macromolecule that consists of a sequence of amino acids covalently linked by peptide bonds. Each amino acid is an organic compound that contains amine $-NH_2$ and carboxylic acid $-COOH$ functional groups, common to all amino acids, and side chain group specific to each amino acid. An amino acid consists of carbon, hydrogen, oxygen, and nitrogen atoms as well as other elements in the side chain. For example, Cysteine consists of:

$$[\underbrace{\mathrm{N}, \mathrm{C}_\alpha, \mathrm{C}, \mathrm{H}, \mathrm{O}}_{\text{backbone part}}, \underbrace{\mathrm{H}_\alpha, \mathrm{C}_\beta, \mathrm{H}_{\beta_1}, \mathrm{H}_{\beta_2}, \mathrm{S}_\gamma, \mathrm{H}_\gamma}_{\text{side chain}}].$$

According to Wagner and Musso (1983), more than 500 different amino acids have been already discovered. However, only 20 of them are generally represented in the human genome. These are

$$\{\mathrm{Ala}, \mathrm{Arg}, \mathrm{Asn}, \mathrm{Asp}, \mathrm{Cys}, \mathrm{Glu}, \mathrm{Gln}, \mathrm{Gly}, \mathrm{His}, \mathrm{Ile},$$
$$\mathrm{Leu}, \mathrm{Lys}, \mathrm{Met}, \mathrm{Phe}, \mathrm{Pro}, \mathrm{Ser}, \mathrm{Thr}, \mathrm{Trp}, \mathrm{Tyr}, \mathrm{Val}\} =: \mathcal{A}.$$

Sometimes we will include into the set $\mathcal{A}$ two other amino acids of rather frequent occurrence $\{\text{Sec}, \text{Mse}\}$ and say that $\mathcal{A}$ is of cardinality $|\mathcal{A}| = 22$.

Each protein is defined by its *primary structure* — a sequence of amino acids $\boldsymbol{a} \in \mathcal{A}^m$, where m is a length of the protein. Under certain environmental conditions, a protein can fold into a certain shape, which is called a *tertiary structure* — a three-dimensional arrangement of atoms in the protein. While this folded structure strictly depends on the environmental conditions (solvent, concentration, temperature, etc.) we will disregard this dependency in our study for simplicity because of its negligible significance in experiments carried out under the same protocol conditions. When a protein consists of two or more protein chains, its folded atomic arrangement in $\mathbb{R}^3$ is called a *quaternary structure*. In this paper, we will not distinguish tertiary and quaternary structure. We will often call them *structure*. Also, we will often call the primary structure just the *sequence*.

1.2.1 Protein folding

A lot of progress has been recently made in protein structure prediction, a computational problem also known as the *protein folding problem*, which consists in determining the protein structure given a protein sequence. Most of the proposed methods for protein structure prediction first generate a set of proposed protein models and then rank them by a certain quality assessment (QA) method to select the best protein model (see figure -1). In such a problem setting, these QA methods deal only with static structures at the atomic level, that is, the step of sampling the protein structure conformations is fully omitted.

It is generally assumed that the best protein model must satisfy the Gibbs free energy minimization principle. Hence the Gibbs free energy serves as a scoring potential or measure of the closeness of a protein model to the actual (native) protein conformation. However, rigorous estimation of the Gibbs free energy requires exhaustive sampling of a huge number of conformational states (Cecchini *et al.*, 2009; Tyka *et al.*, 2006), and in most of the practical cases is computationally intractable.

If the native protein conformation is known, one can measure the quality of a protein model using other straightforward and more similarity measures (RMSD, GDT-TS, TM-score, etc.) (Zemla, 2003) that, in contrast to the Gibbs free energy, can be efficiently computed to score protein models when comparing them to the native structure.

Protein quality assessment is aimed at devising scoring potentials that predict the closeness of a protein model to the native protein conformation. Usually, QA methods are trained to predict the quality of protein models in line with some similarity measures, which often involves machine learning. Therefore, being aimed at scoring the protein models according to their closeness to the native structure, protein quality assessment is a crucial problem for tertiary structure prediction methods.

1.2.2 Rotamer prediction

The problem of side chain (or *rotamer*) prediction is to predict the folding of side chains given a protein backbone with the known sequence. Apparently, this problem can be reduced to the problem of protein folding. Hence it is simpler than the latter. However, usually, the

problem of protein folding requires solving the side chain prediction problem at the last stage (see figure -1). Hence general strategy for protein folding is as follows.

1. Predict a *coarse-grained* protein model — backbone without side chains.

2. Refine the predicted on the first stage coarse-grained model (i.e. predict the folding of the side chains).

As for the protein folding problem, side chains are modeled to minimize the total energy of the molecule. Although an optimization problem can be already formally stated, we have to discuss the search space first.

For simplification, the continual conformational space of the side chain of each amino acid $a_i \in \mathcal{A}$ is restricted to the finite set $\mathcal{R}_i$, $|\mathcal{R}_i| < \infty$, of possible conformations $r_i \in \mathcal{R}_i$, which are called *rotamers*. Then, the whole search space is the Cartesian product of sets of rotamers

$$\mathcal{R}^m := \mathcal{R}_1 \times \cdots \times \mathcal{R}_m, \quad |\mathcal{R}_i| < \infty, \quad i = 1, \ldots, m, \tag{1.1}$$

for each side chain. Possible rotamers $\mathcal{R}_i$ themselves depend on the type of the corresponding amino acid $a_i \in \mathcal{A}$:

$$\mathcal{R}_i = \mathcal{R}(a_i). \tag{1.2}$$

This helps to restrict the conformational search space and to reduce the initial problem to the problem of choosing the best in the energy sense rotamers (possible conformations). The rotamer sets are usually constructed according to rotamer libraries (Shapovalov and Dunbrack, 2011) that can be either backbone-independent or backbone-dependent. In (1.2) a rotamer space depends on the type of amino acid but not on the positions of other amino acids. Backbone-dependent rotamer libraries are built with respect to backbone dihedral angles of adjacent amino acids driven by the conformation of a given backbone:

$$\mathcal{R}_i = \mathcal{R}_{\text{bd}}(a_i, \boldsymbol{b}_{i-1}, \boldsymbol{b}_i, \boldsymbol{b}_{i+1}), \tag{1.3}$$

where $\boldsymbol{b}_i, \boldsymbol{b}_{i-1}, \boldsymbol{b}_{i+1} \in \mathbb{R}^{3\times 3}$ define backbone parts (positions of heavy backbone atoms $\mathrm{N}, \mathrm{C}_\alpha, \mathrm{C}$) of amino acids a_i, a_{i-1}, and a_{i+1} respectively:

$$\boldsymbol{b}_i = [\boldsymbol{b}_i^{\mathrm{N}}, \boldsymbol{b}_i^{\mathrm{C}_\alpha}, \boldsymbol{b}_i^{\mathrm{C}}] \in \mathbb{R}^{3\times 3}, \ \ i = 1, \ldots, m. \tag{1.4}$$

Since an objective (energy potential) and the search space are defined, a certain optimization problem is to be solved. This problem can be formulated as follows:

$$E(\boldsymbol{a}^0, \boldsymbol{b}^0, \boldsymbol{r}) \to \min_{\boldsymbol{r}\in\mathcal{R}^m}, \tag{1.5}$$

where $\boldsymbol{a}^0 \in \mathcal{A}^m$ defines the given protein sequence, $\boldsymbol{b}^0 \in \mathbb{R}^{m\times 3\times 3}$ — given protein backbone conformation (backbone parts for all amino acids in the protein — spatial arrangement of $3m$ heavy backbone atoms), and $\boldsymbol{r} \in \mathcal{R}^m = \mathcal{R}_1 \times \cdots \times \mathcal{R}_m$ is the vector of rotamers (conformations of the side chains). For rotamers, the energy potential $E(\boldsymbol{a}^0, \boldsymbol{b}^0, \boldsymbol{r})$ is calculated according to a given force field model after the reconstruction of the whole protein structure using certain rotamers $\boldsymbol{r} \in \mathcal{R}^m$.

Finally, all the missed side chains are reconstructed in accordance to the found optimal rotamers. According to Miao *et al.* (2011), to measure the quality of side chain folding, the reconstructed structure is compared to the native one and the four following criteria are estimated:

- root mean square deviation (RMSD) between corresponding atoms in aligned reconstructed and native structures,
- number of atomic pairs in contact, with the distance between them of less than 60% of the sum of van der Waals radii of atoms in the pair,
- the ratio of amino acids with side chain torsion angles χ_1, predicted with 40-degree accuracy compared to the native structure (Shapovalov and Dunbrack, 2011),
- the ratio of amino acids with side chain torsion angles χ_1 и χ_2, predicted with 40-degree accuracy each (Shapovalov and Dunbrack, 2011).

1.2.3 Protein design

Protein design is one of the most important tools in protein engineering, and at the same time, it is one of the most complicated problems of computational biology. It is aimed at determining a protein sequence that possesses given chemical properties and biological function. There are two approaches to design new protein molecules: from scratch (*de novo protein design*) or composing a new structure from already known templates (protein redesign) (Khoury *et al.*, 2014). In this thesis, we will consider approaches for *de novo protein design* because they are more general, and they do not suffer from the limits imposed by the obtained database of protein structures. Since no restrictions are imposed on the primary structure, there is a high probability that the solution will be a new, previously unknown molecule. Methods for *de novo protein design* are based on the search through all possible primary structures and on the selection of the best possible sequences according to certain heuristic criteria.

Since the initial formulation of the protein design problem is vague and neither formal nor strict, this problem is often simplified to the so-called problem of inverse folding (see figure 1-1), which is to find a protein sequence that folds into the target protein structure. This simplification is admissible because the protein structure and functions of this protein are often in a strong relationship. According to Huang *et al.* (2016), this formulation is also called as fixed-backbone protein design problem. Assuming the hypothesis that proteins fold into the lowest energy states, the protein design is performed in two main stages (Huang *et al.*, 2016).

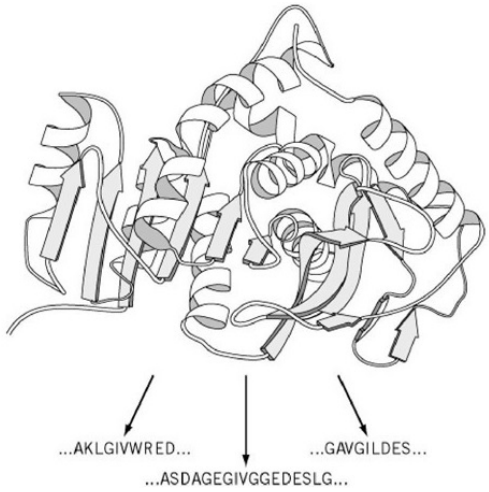


Figure 1-1: Inverse folding illustration

- First, the most probable sequences are to be sampled.
- Second, they must be verified for folding into the target shape. This makes the problem of tertiary structure prediction essentially important here, since these sampled at the first stage sequences are almost always new, and their actual tertiary structure is not yet discovered.

Despite the advances in protein design methods, their current state is far from being used in biomedicine and bioengineering. It is always required to perform laborious experiments and manually contribute to the produced results. Another bottleneck here is a procedure of assessment since the results always require performing expensive experimental validation.

The setting of the protein design problem naturally states an optimization problem. Indeed, the aim is to find a primary structure that would be likely to fold into the target tertiary structure. Hence, if we could know folds of all the stable protein structures, the answer would be a sequence (or sequences) whose folded tertiary structure is the closest to the target one. However, since the sequence space grows exponentially when increasing the length of the structure, and the number of possible sequences is 20^m, where m is a sequence length (roughly, $m \sim 100$), it is impossible to predict or compute folds for each possible sequence. Even though not all the possible sequences have a stable native state to fold into, we will leave this aspect out of the scope of the current thesis.

Another approach is to consider this problem as a problem of searching for a protein sequence whose native state would be the closest to the target protein structure in the energy sense. In other words, a protein sequence that would be likely to fold into the target tertiary structure. Hence the optimization problem can be stated as follows:

$$E(\boldsymbol{a}, \boldsymbol{b}^0, \boldsymbol{r}) - \min_{\boldsymbol{b}, \boldsymbol{r}} E(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{r}) \to \min_{\boldsymbol{a}, \boldsymbol{r}}, \tag{1.6}$$

where

- $\boldsymbol{a} \in \mathcal{A}^m$ — protein sequence,
- $\boldsymbol{r} \in \mathcal{R}^m = \mathcal{R}_1 \times \cdots \times \mathcal{R}_m$ — vector of rotamers ($\mathcal{R}_i,\ i = 1, \ldots, m$ — rotamer sets),
- $\boldsymbol{b} \in \mathbb{R}^{m \times 3 \times 3}$ — protein backbone conformation (positions of backbone parts (1.4)),
- $\boldsymbol{b}^0 \in \mathbb{R}^{m \times 3 \times 3}$ — conformation of the target backbone structure.

Note that problem (1.6) can be reformulated in the following manner:

$$\begin{aligned} \min_{\boldsymbol{a}, \boldsymbol{r}} \left[E(\boldsymbol{a}, \boldsymbol{b}^0, \boldsymbol{r}) - \min_{\boldsymbol{b}, \boldsymbol{r}} E(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{r}) \right] \\ = \min_{\boldsymbol{a}} \left[\min_{\boldsymbol{r}} E(\boldsymbol{a}, \boldsymbol{b}^0, \boldsymbol{r}) - \min_{\boldsymbol{b}, \boldsymbol{r}} E(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{r}) \right], \end{aligned} \tag{1.7}$$

and hence distinctly estimating the minimum over rotamers $\boldsymbol{r}$ for energy function E as a function of $\boldsymbol{r}$:

$$E(\boldsymbol{a}, \boldsymbol{b}) := \min_{\boldsymbol{r} \in \mathcal{R}^m} E(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{r}), \tag{1.8}$$

problem (1.6) can be simplified as follows:

$$E(\boldsymbol{a}, \boldsymbol{b}^0) - \min_{\boldsymbol{b} \in \mathbb{R}^{m \times 3 \times 3}} E(\boldsymbol{a}, \boldsymbol{b}) \to \min_{\boldsymbol{a} \in \mathcal{A}^m} . \tag{1.9}$$

We should mention that the energy function (1.8) does not depend on the conformation of its protein side chains. Protein models in such a reduced representation without defined side chains are called *coarse-grained protein models*. Hence we will call functions of form (1.8) *coarse-grained* energy functions. It is seen from definition (1.8) that the coarse-grained energy function is supposed to score protein backbones as they would belong to proteins with side chains folded into the energetically optimal conformation.

Usage of an accurate energy function is crucial for solving the protein design problem because this function should rank protein sequences according to the closeness of their native tertiary structures to the target one. Energy functions are usually composed of physics-based and statistical potentials. While physics-based energy functions are derived from the physical knowledge and quantum mechanical simulations, which are very hard to compute, statistical potentials are based on conformations of known structures in the database. There are also so-called knowledge-based potentials that are built from the observed structures, with a difference that in contrast to the statistical potentials, which are constructed only from known native conformations, for building knowledge-based potentials, often additional simulated non-native (decoy) structures are used.

Usually, energy potentials are devised to be pairwise decomposable. That is, the total energy is calculated as a sum of pairwise terms for each atom or residue pair. Nevertheless, this does not make problem (1.6) simpler in the general case unless the second term $\min_{\boldsymbol{b}, \boldsymbol{r}} E(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{r})$ is still present in the objective (1.6). Note that classical energy potentials were initially devised to score different conformations of proteins with identical sequences for performing simulations. However, protein design deals with a vast amount of different protein sequences simultaneously. That is why the second term in problem (1.6) can not be omitted when using classical energy potentials (Cecchini *et al.*, 2009; Tyka *et al.*, 2006).

1.3 Formal definitions

For the sake of simplicity, here and further it is assumed without loss of generality that all the protein structures under consideration are of identical fixed length m. To sum up the problem statements above and to simplify further narration, let us put them in a category theoretic (or set-theoretic) language. First, let us introduce the following notation:

- $\mathcal{A}^m$ — the set of all protein sequences,
- $\mathbb{S}^m_{\text{b}} = \mathbb{R}^{m \times 3 \times 3}$ — protein backbone conformation space,
- $\mathcal{R}^m$, $|\mathcal{R}^m| < \infty$ — the set of all rotamers (1.1),
- $S^m_{\text{t}} \subset \mathcal{A}^m \times \mathbb{S}^m_{\text{b}} \times \mathcal{R}^m$ — the set of all tertiary structures,
- $S^m_{\text{r}} \subset \mathcal{A}^m \times \mathbb{S}^m_{\text{b}}$ — the set of all coarse-grained protein structures,

- $\boldsymbol{a} \in \mathcal{A}^m$ — a primary structure,
- $\boldsymbol{r} \in \mathcal{R}^m$ — a vector of rotamers,
- $\boldsymbol{b} \in \mathbb{S}_{\mathrm{b}}^m$ — a protein backbone — positions of backbone parts (1.4),
- $(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{r})$ — a tertiary structure,
- $(\boldsymbol{a}, \boldsymbol{b})$ — a coarse-grained structure,
- π_{tr} — the coarse-graining projection mapping:

$$\pi_{\mathrm{tr}}(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{r}) = (\boldsymbol{a}, \boldsymbol{b}) \in S_{\mathrm{r}}^m \quad \forall (\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{r}) \in S_{\mathrm{t}}^m, \tag{1.10}$$

- $\pi_{\mathrm{tb}},\ \pi_{\mathrm{rb}}$ — the backbone projection mappings:

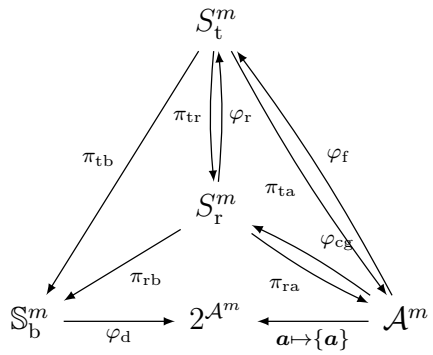
$$\pi_{\mathrm{tb}}(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{r}) = \pi_{\mathrm{rb}}(\boldsymbol{a}, \boldsymbol{b}) = \boldsymbol{b} \in \mathbb{S}_{\mathrm{b}}^m \quad \forall (\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{r}) \in S_{\mathrm{t}}^m, \tag{1.11}$$

- $\pi_{\mathrm{ta}},\ \pi_{\mathrm{ra}}$ — the sequence projection mappings:

$$\pi_{\mathrm{ta}}(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{r}) = \pi_{\mathrm{ra}}(\boldsymbol{a}, \boldsymbol{b}) = \boldsymbol{a} \in \mathcal{A}^m \quad \forall (\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{r}) \in S_{\mathrm{t}}^m, \tag{1.12}$$

- $\varphi_{\mathrm{r}} = \pi_{\mathrm{tr}}^{-1}$ — the rotamer reconstruction (predict rotamers given coarse-grained structure),
- $\varphi_{\mathrm{f}} = \pi_{\mathrm{ta}}^{-1}$ — the protein folding (predict protein structure given protein sequence),
- $\varphi_{\mathrm{cg}} = \pi_{\mathrm{ra}}^{-1}$ — the coarse-grained protein folding (predict coarse-grained protein structure given protein sequence),
- φ_{d} — protein design (predict protein sequence given backbone structure),

Now, let us consider commutative diagram (1.13).



(1.13)

Existence of the inverses of the projection mappings $\pi_{\mathrm{tr}},\ \pi_{\mathrm{ra}},\ \pi_{\mathrm{ta}}$ follows from the physical interpretation of their definitions (protein folding defines isomorphism between the sets $\mathcal{A}^m$, S_{r}^m, and S_{t}^m). For other protein lengths the mappings are from this section can be reciprocally generalized.

1.4 Project aims and contribution

The goal of this thesis project is to conduct a comprehensive study of the protein design problem and other related problems of computational biology: protein folding, protein quality assessment, protein side chain prediction when considering rather mathematical and computational component than biological aspects.

We put the emphasis on the training the protein scoring potentials that similarly to the Gibbs free energy predict the closeness of a protein model structure to a native structure (correctly folded primary structure of the protein sequence). We investigate the machine learning methods for the training the scoring potentials and propose a novel method that requires only backbone conformation of proteins without their side chain atom positions. The proposed method can be used to assess the quality of coarse-grained protein models, what makes it generally particularly efficient for applying to the protein design problem with a coarse-grained energy function (see problem (1.9)), since the reduced optimization problem is vastly smaller than the initial one (see problem (1.6)). The proposed energy function is a single-model QA method, which uses only structural features along with the explicit representation of solvent on a grid preserving the smoothness of the scoring potential that makes it also applicable to molecular mechanics or dynamics, for example. The learning procedure requires artificially generated non-native (decoy) structures for the training and provides high-quality residue-pairwise scoring potentials. This makes the total scoring potential efficient to compute and also pairwise decomposable what is so important for applying it to the protein design problem. It can also be easily extended for taking into account conformations of protein side chains, which would make it applicable to rotamer prediction problem as well.

We provide two modifications of the proposed scoring potential. One is designed for the protein design problem, another one — for protein folding. A computational experiment on the data from real experiments and competitions for the protein structure prediction proved that the modification of the scoring function for protein folding achieves the state-of-the-art quality and surpasses state-of-the-art methods on some datasets. We propose to use the modification of the scoring function for protein folding problem to validate the solution of the protein design problem and to select those models from the generated set when solving the protein design problem that are most probable to fold into the target shape.

The modification of the scoring function for protein design is designed in such a way to enable the reduction of protein design optimization problem (1.9) to the problem of quadratic boolean programming (BQP). We analyze the techniques for relaxing the arising NP-hard BQP problem (linear relaxation, semi-definite programming relaxation, Lagrange relaxation) into the problems of convex optimization and compare them to general methods for discrete optimization: greedy optimization algorithm and simulated annealing.

Although a lot of algorithms have been proposed to solve the rotamer prediction problem, they are either computationally inefficient or do not have any proved deterministic theoretical guarantees for the quality of the solution. Usage of pairwise decomposable energy function enables the reduction of the arising optimization problem (1.5) for rotamer prediction to a BQP problem as in the case of protein design.

1.5 Structure of the thesis

The thesis is organized as follows: in chapter 2, we give a comprehensive review of related studies and literature. Then, in chapter 3, we study the methods that are involved in our study. In sections 3.2 and 3.5, we propose the methods and other involved instruments and put the considered problems of structural biology in a formal way giving mathematical problem statements as optimization problems in sections 3.4 and 3.3. In chapter 4, we show the results from the performed computational experiments and discuss the results of the thesis project. Finally, in chapter 5, we emphasize the achievements, findings, and observed limitations.

Chapter 2

Related work

2.1 Protein quality assessment

There are two types of QA methods. Single-model QA methods consider atoms only in the assessed model without any additional information about other models in the sampled set. On the contrary, consensus-model QA methods decide on the quality of individual models based on their statistics in the set. This makes their range of applicability somewhat narrower. For example, single-model QA methods can be also used for conformational sampling and structure refinement. They are also proved to achieve a better performance compared to consensus-model QA methods on unbalanced protein model sets and in cases where the protein models within the assessed set are very similar (Ray *et al.*, 2012).

Some protein structure prediction methods use an additional level of simplification (Kmiecik *et al.*, 2016). More precisely, they work with a reduced (coarse-grained) representation of amino acids instead of the fully atomic representation. This allows to significantly enhance their performance at the first stage of protein structure prediction workflow. However, this representation also requires a corresponding scoring function for the coarse-grained quality assessment.

Among recently proposed single-model QA methods there are, generally, two main approaches to design a scoring potential: physics-based and knowledge-based (data-driven) approaches (Faraggi and Kloczkowski, 2014; Liu *et al.*, 2014). Physics-based potentials are constructed according to some physical knowledge of the configuration and interactions in the system. Usually, these potentials (often called force fields) decompose the total energy into a sum of additive contributions that can represent stretching of bonds and angles, dihedral potentials, electrostatic and van der Waals forces, etc.

Alongside with physics-based methods, there are so-called knowledge-based ones that deduce the essential energies of molecular interactions from biological structural and evolutionary data. The corresponding scoring potentials are typically derived using the information from known tertiary structures found in structural databases either training the scoring potential using phenomenological machine learning or estimating probabilities of the certain conformations (statistical methods).

There have been many proposed QA methods using either physics-based or knowledge-based approaches. However, QA methods that combine these two recently appeared to be

the most promising. Here we observe protein QA methods that use the most common and representative techniques. The SELECTpro method (Randall and Baldi, 2008) is derived from an energy function comprised by physical, statistical, and predicted structural terms. The total energy is a weighted sum of introduced energy terms with weights maximizing the sum of the GDT-TS (Zhang and Skolnick, 2007) of the lowest models on the training set built from CASP6 (Moult *et al.*, 2016) protein domains. Along with the conventional potentials (van der Waals forces and electrostatic interactions, etc.) and side chain hydrogen bonding potential, the authors of SELECTpro also propose to use β-strand pairing derived from a reduced representation and introduce penalties for the mismatch of the observed structural features to the predicted ones (secondary structure, solvent accessibility, contact map). In the reduced representation, only N, C_α, C, O, H, C_β atoms are represented, which requires positions of only heavy backbone N, C_α, C atoms as input, because positions of all the rest atoms can be derived from the given ones. For glycines, a pseudo C_β is calculated reciprocally. Finally, SELECTpro (Randall and Baldi, 2008) computes the all-atom representation from the reduced one using energy terms for the reduced representation and then it performs the quality assessment for all-atom representation using all the proposed energy terms.

Statistical potentials are derived according to the Boltzmann assumption, which says that the energy of a protein structure is proportional to the negative logarithm of the probability of its conformational state (Liu *et al.*, 2014). For example, the popular RWplus (Zhang and Zhang, 2010) scoring function combines pairwise distance-dependent and orientation-dependent contributions. For each pair of atom types (α, β), it computes the number of atom pairs $N_{\text{obs}}(\alpha, \beta, r)$ within a distance in the r to $r + \Delta r$ interval for a given model and the number of expected pairs $N_{\text{exp}}(\alpha, \beta, r)$ to estimate the likelihood of the protein model. Orientation-dependent statistical potentials consider the many-body interactions by describing both distance and relative orientation of interacting atom groups. Consequently, they typically outperform traditional only distance-dependent methods. To assign the orientation-dependent potential, the authors of RWplus (Zhang and Zhang, 2010) for each residue type except glycine and alanine, define a local frame based on three side chain atoms being centered in the position corresponding to one of them. The relative orientation of two local frames is then represented by five parameters, two pairs of spherical angles and a torsion angle. Finally, the total orientation-dependent packing energy is calculated using the same statistics technique as when calculating the distance-dependent potential.

ORDER_AVE is another orientation-dependent statistical potential, which assesses the quality of protein models purely at the reduced representation (Liu *et al.*, 2014). It uses joint probability distribution of four parameters to describe the geometric relationship between pairs of heavy atoms (representatives) and their corresponding bonded neighbors. These parameters are three angles that describe the relative orientation of four atoms, and an atomic distance between the pair of the representatives. ORDER_AVE treats local, i.e. when two corresponding residues have a small sequence separation, and nonlocal interactions separately with the overall energy given as a weighted sum of these two.

Nowadays more and more research is devoted to methods for training of scoring potentials using machine learning techniques. For example, the ProQ2 (Ray *et al.*, 2012) scoring function, which is one of the best protein QA methods according to experiments on CASP11 (Moult *et al.*, 2016) tests sets, is trained using support vector machine (SVM) in the space of structural and sequence-based features calculated from the model. As structural fe-

atures, this potential uses contacts between 13 different atom types, residue-residue contacts, and surface accessibility by grouping amino acids into six different groups. The sequence-based features (secondary structure, surface accessibility, and sequence profiles) are derived using information predicted from sequence. ProQ2 proved to achieve a considerable performance of the protein quality assessment but it requires usage of many external programs that make it difficult to distribute (Uziela and Wallner, 2016). Another scoring method by Faraggi and Kloczkowski (Faraggi and Kloczkowski, 2014) uses physics-based electrostatic potentials and other knowledge-based potentials as features and trains a neural network to predict the TM-score (Zhang and Skolnick, 2007) similarity measure between protein models and native structures. The Wang_SVM (Liu *et al.*, 2016) scoring function was trained with support vector machine (SVM) using as features the protein's sequence and also several protein descriptors predicted by external utilities. These are secondary structure and solvent accessibility predicted with SSPRO, residue-residue contact probabilities predicted with NNcon, and evolutionary information predicted with PSI-BLAST. The Qprob (Cao and Cheng, 2016) scoring function was trained by estimating the distribution of features provided by other QA methods (RWplus (Zhang and Zhang, 2010), ModelEvaluator (Wang *et al.*, 2009), DOPE (Shen and Sali, 2006), RF_CB_SRS_OD (Rykunov and Fiser, 2010)), structural features (compact score, surface score, the exposed mass, the exposed surface), and features predicted from the protein's sequence (secondary structure, solvent accessibility). Following the similar strategy, in (Jing *et al.*, 2016) a learning-to-rank technique (ranking SVM (Joachims, 2002)) was applied to train a scoring function using predictions of other QA methods (DFIRE (Zhou and Zhou, 2002), DOPE (Webb and Sali, 2014), GOAP (Zhou and Skolnick, 2011), RWplus (Zhang and Zhang, 2010), etc.) as features. Thus, machine learning techniques have been widely applied to train scoring potentials for QA methods (Ray *et al.*, 2012; Faraggi and Kloczkowski, 2014; Liu *et al.*, 2016; Cao and Cheng, 2016; Jing *et al.*, 2016) and currently knowledge-based methods appear to demonstrate a better performance than statistical and physics-based techniques.

Although plenty of QA methods have been proposed (Randall and Baldi, 2008; Zhang and Zhang, 2010; Liu *et al.*, 2014; Ray *et al.*, 2012; Faraggi and Kloczkowski, 2014; Liu *et al.*, 2016; Cao and Cheng, 2016; Jing *et al.*, 2016), often these either miss components related to meaningful energy terms, which must be taken into account when computing the total Gibbs free energy (solvation-related terms (Zhang and Zhang, 2010; Liu *et al.*, 2014; Jing *et al.*, 2016), hydrogen bond energy (Zhang and Zhang, 2010; Liu *et al.*, 2014; Ray *et al.*, 2012; Faraggi and Kloczkowski, 2014; Liu *et al.*, 2016; Cao and Cheng, 2016; Jing *et al.*, 2016)), or they require features predicted from the sequence, which breaks the continuity and smoothness of the scoring potential (Randall and Baldi, 2008; Ray *et al.*, 2012; Faraggi and Kloczkowski, 2014; Liu *et al.*, 2016; Cao and Cheng, 2016; Jing *et al.*, 2016). For example, hydrogen bonds (Hubbard and Kamran Haider, 2001) confer directionality and specificity to the intra-molecular interactions in structures. They provide structural organization of distinct protein folds because suitable interactions in the folded structure are typically achieved by the maximal number of hydrogen-bonding groups. This makes hydrogen bond terms important to include in scoring potentials when ranking the protein models. Also, most of QA methods require all-atom protein models as input, and thus their performance critically depends on the accuracy of side chain packing, i.e., positions of the side chain atoms, which can be modeled using SCWRL4 (Krivov *et al.*, 2009) as in (Cao and

Cheng, 2016). Another problem of many QA methods is that sequence-dependent features often introduce non-smooth terms that violate the smoothness of total scoring potential.

2.2 Protein folding

Khoury *et al.* (2014) reviewed methods for protein structure prediction and refinement (side chain prediction). They have also noted that methods for protein structure refinement appeared to degrade models predicted by methods of protein structure prediction. Therefore, the rotamer prediction problem remains an important unsolved problem of computational biology.

The Critical Assessment of Techniques for Protein Structure Prediction (CASP) (Moult *et al.*, 2005, 2009, 2011, 2014, 2007, 2016, 2003) started in 1994 and since then it aims to assess the quality of a number of competing methods for protein structure prediction. The assessment can be split into two parts. For template-based modeling, the tertiary structure is being predicted given a template structure. This structure is used to infer information about topology and secondary structure of the target protein. Methods for template-based modeling are aimed at identifying structures from the protein data bank (PDB) that can be aligned with the given sequence (sequence alignment). In the case of free modeling, the tertiary structure is predicted for proteins that do not have distinguishable templates in the PDB. This case appears to be very hard and thus most of the methods perform poorly (Moult *et al.*, 2014). The majority of methods for free modeling work by selecting at first a proper template from structures of the PDB. Duan and Kollman (1998) proposed using methods of molecular dynamics to predict the protein structures. Since then a number of methods were developed but all of them can be applied to simulate only small molecules Khoury *et al.* (2014).

To assess the methods for protein structure prediction, their predictions are compared to native structures. There have been lots of different metrics proposed. However, the most robust and efficient are GDT-TS and TM-score metrics. GDT-TS and TM-score were designed to assess the quality of protein structure templates and predicted full-length models being size-independent and robust to local prediction structural errors (Zemla, 2003).

2.3 Protein design

Saven (2013); Khoury *et al.* (2014); Huang *et al.* (2016) review advances and challenges in protein structure prediction and *de novo* protein design. Huang *et al.* (2016) discuss the current challenges of computational biology. They also note that up to now almost all *de novo* designed protein structures are very stable in contrast to native proteins that are unstable to few changes in sequences. Hence, it is supposed that *de novo* protein design helps to design more robust proteins (Huang *et al.*, 2016).

Samish *et al.* (2011); Khoury *et al.* (2014) reviewed modern techniques for protein re-design and *de novo* design in the assumption that structure energy function is pairwise

decomposable, i.e. the total energy is calculated as a sum of one-body and two-body terms:

$$E(\boldsymbol{a}, \boldsymbol{r}) = \sum_{i=1}^{m} \varepsilon_i(a_i, r_i) + \sum_{i=1}^{m} \sum_{j=1}^{m} \gamma_{ij}(a_i, r_i, a_j, r_j). \tag{2.1}$$

One-body term $\varepsilon_i(a_i, r_i)$ represents the energy of interaction of side chain atoms in rotamer r_i of amino acid a_i with fixed backbone atoms of the protein. Two-body term $\gamma_{ij}(a_i, r_i, a_j, r_j)$ is calculated as the sum of all interactions between pairs of side chain atoms of amino acids a_i and a_j in rotamers r_i and r_j, respectively. They also discussed the foldability criteria mentioning that decreasing the energy of the target structure by varying the primary structure is not always correlated with the improved foldability since proteins with different sequences can have native states of different energies Samish *et al.* (2011).

Fung *et al.* (2008); Samish *et al.* (2011) discussed methods for solving arising combinatorial optimization problems. A number of techniques for solving these optimization problems were proposed. Generally, they can be divided into two groups: stochastic (Monte Carlo, simulated annealing, graph search algorithm A^*, etc.) and deterministic (dead-end elimination (DEE), mean field algorithm, graph decomposition, linear programming, etc.) approaches. While stochastic techniques work fast but without any theoretical guarantees, deterministic algorithms often provide the exact solution (e.g. DEE) but they might be computationally inefficient when the length of the protein is large since the optimization problem they deal with is NP-hard in the general case.

Research conducted by Riazanov *et al.* (2016) showed that usage of conventional protein energy function in protein design results in predicting sequences that consist entirely of the same amino acids Cys. This result seems to be expected as the optimal solution corresponds to sequences that aim to maximize the number of disulfide bridges, which are very energetically favorable. However, conventional potentials do not validate the proposed sequences to fold into the target structure. Hence, the question of constraining the proposed sequences to make conventional energy more relevant in protein design remains open. In this paper, we propose to introduce an energy correction terms that make amino acids of different types more evenly distributed along the proposed sequences.

Chapter 3

Methods

In this chapter, we will use the notation introduced in section 1.3.

First, in section 3.1 we will consider measures for estimating the similarity between pairs of protein structures with identical amino acid sequences.

Then, in section 3.2 we construct the SBROD (Smooth Backbone-Reliant Orientation-Dependent) scoring potential assessing the quality of coarse-grained protein models. This method comes in two modifications. The first one is used for protein quality assessment when solving the protein folding problem (ranking protein models with identical sequences). The second one is applied to the protein design problem (assessing protein models with different sequences).

In sections 3.3 and 3.4 we formally set the protein design and rotamer prediction problems. Then, in section 3.5 we consider basic general methods for solving arising problems of discrete optimization. Furthermore, assuming that the energy potential is pairwise decomposable (the SBROD scoring function for the protein design problem and any other pairwise decomposable energy potential for the rotamer prediction problem), we reduce the arising optimization problems to the boolean quadratic programming problems.

Finally, we study convex relaxation techniques for approximate solving the boolean quadratic programming problem in section 3.6.

3.1 Protein similarity measures

In this section, we will consider measures for estimating the similarity between protein tertiary structures with identical amino acid sequences. When calculating for a protein model against the native structure, these measures provide estimation on the quality of considered protein model, but in contrast to the Gibbs free energy, they can be efficiently computed.

Let

$$P = (\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{r}) \in S_{\mathrm{t}}^m \subset \mathcal{A}^m \times \mathbb{S}_{\mathrm{b}}^m \times \mathcal{R}^m \tag{3.1}$$

be the tertiary structure of a native protein, and let

$$P' = (\boldsymbol{a}, \boldsymbol{b}', \boldsymbol{r}') \in \mathcal{A}^m \times \mathbb{S}_{\mathrm{b}}^m \times \mathcal{R}^m \tag{3.2}$$

be an arbitrary tertiary structure (model structure) with the same sequence as native struc-

ture P:

$$\pi_{\text{ta}}(P') = \pi_{\text{ta}}(P) = \boldsymbol{a}. \tag{3.3}$$

First, let us consider several metrics for estimating the similarity between coarse-grained model structure P' and native structure P.

1. RMSD (root-mean-square deviation) of atomic positions

$$S^*_{\text{RMSD}}(P', P) = \min_{\boldsymbol{t}\in\mathbb{R}^3,\, \mathbf{S}\in\text{SO}(3)} \left(\frac{1}{3m} \sum_{i=1}^{m} \sum_{\theta\in\{\text{N},\text{C}_\alpha,\text{C}\}} \|\boldsymbol{b}_i^\theta - \mathbf{S}\boldsymbol{b}'^\theta_i + \boldsymbol{t}\|^2 \right)^{\frac{1}{2}} \in [0, \infty) \tag{3.4}$$

— square root of the sum of squared distances between corresponding heavy backbone atoms (1.4) in superimposed pair of structures.

2. TM-score (template modeling score) (Zemla, 2003)

$$S^*_{\text{TM}}(P', P) = \max_{\boldsymbol{t}\in\mathbb{R}^3,\, \mathbf{S}\in\text{SO}(3)} \frac{1}{m} \sum_{i=1}^{m} \left(1 + \frac{\|\boldsymbol{b}_i^{\text{C}_\alpha} - \mathbf{S}\boldsymbol{b}'^{\text{C}_\alpha}_i + \boldsymbol{t}\|^2}{d_0^2} \right)^{-1} \in (0, 1], \tag{3.5}$$

where d_0 is scale factor (typically, $d_0 = 5\text{Å}$).

3. GDT-TS (global distance test: total score) (Zemla, 2003)

$$S^*_{\text{GDT-TS}}(P', P) = \max_{\boldsymbol{t}\in\mathbb{R}^3,\, \mathbf{S}\in\text{SO}(3)} \frac{1}{4m} \sum_{i=1}^{m} \sum_{j=1}^{4} \mathbb{1}\left[\|\boldsymbol{b}_i^{\text{C}_\alpha} - \mathbf{S}\boldsymbol{b}'^{\text{C}_\alpha}_i + \boldsymbol{t}\| < c_j \right] \in [0, 1], \tag{3.6}$$

where $c_{1,2,3,4} = 1, 2, 4, 8\text{Å}$ — cutoff distances between superimposed residues, and $\mathbb{1}[\cdot]$ — the truth predicate.

4. GDT-HA score (global distance test: high accuracy) (Zhang and Skolnick, 2007) is calculated as the GDT-TS, where cutoffs are twice discounted:

$$S^*_{\text{GDT-HA}}(P', P) = \max_{\boldsymbol{t}\in\mathbb{R}^3,\, \mathbf{S}\in\text{SO}(3)} \frac{1}{4m} \sum_{i=1}^{m} \sum_{j=1}^{4} \mathbb{1}\left[\|\boldsymbol{b}_i^{\text{C}_\alpha} - \mathbf{S}\boldsymbol{b}'^{\text{C}_\alpha}_i + \boldsymbol{t}\| < \frac{c_j}{2} \right] \in [0, 1]. \tag{3.7}$$

The histograms for similarity measured on CASP[5-10] datasets (Moult *et al.*, 2003, 2005, 2007, 2009, 2011, 2014) (see data description in section 3.2.2) are depicted in figure -2. One can see that while the RMSD metric is the most intuitive, it is not robust since there are few big outliers in the histogram. In contrast to RMSD, histograms for other metrics are more even, which makes them preferable to RMSD.

To estimate the similarity between full tertiary structures (with defined rotamers), first, side chain atomic positions are reconstructed from the rotamers. Then, four criteria described by Miao *et al.* (2011) can be used:

- RMSD for the whole tertiary structures (taking into account all the atoms in full atomic representation),

- percentage of correct χ_1 — the ratio of amino acids with side chain torsion angles χ_1, predicted with 40-degree accuracy compared to the native structure (Shapovalov and Dunbrack, 2011),

- percentage of correct both χ_1 and χ_2 — the ratio of amino acids with side chain torsion angles χ_1 и χ_2, predicted with 40-degree accuracy each (Shapovalov and Dunbrack, 2011).

Description of the geometry of amino acids and definition of dihedral angles χ_1 and χ_2 can be found in (Shapovalov and Dunbrack, 2011).

Finally, one can use another external criterion to assess the quality of a protein model: a number of clashed atom pairs, where an atom pair is called to be in clash if the distance between these atoms is less than 60% of the sum of their van der Waals radii. This metric does not require the native structure but indicates how many obvious mistakes are made when reconstructing a protein model.

3.2 Protein quality assessment

In this section, we construct the SBROD (Smooth Backbone-Reliant Orientation-Dependent) scoring function to assess the quality of coarse-grained protein models. The workflow of SBROD consists in two stages. First, features are to be extracted from each protein model in the dataset. Then, the predicted score of a given protein model is computed as a weighted sum of features extracted at the first stage. Figure 3-1 schematically shows the training workflow. Here, four groups of features are extracted, as we describe in more detail below. These features are then either scaled individually setting the maximal absolute value of each feature in the training set to be 1.0 (Scaler in figure 3-1) or normalized by setting the ℓ^2 norm of each sample with at least one non-zero component to be 1.0 (Normalizer in figure 3-1). Finally, the Ridge Regression model is trained on these features. We should also note that we have tried more sophisticated models at this stage including Lasso (Tibshirani, 1996), Elastic Net (Zou and Hastie, 2005), Bayesian Regression (Neal, 1996), Ranking SVM (Joachims, 2002) in combination with PCA and Random Projections (Ailon and Chazelle, 2009) for dimensionality reduction, as well as their different modifications and combination via stacking (der Laan *et al.*, 2007). However, they did not provide any significant advantages over Ridge Regression model regarding the quality of the predictions. The input parameters for the proposed workflow will be discussed further.

3.2.1 Feature extraction

We built the feature space by concatenating four groups of structural features, which have their physical interpretation and correspond to particular inter-atomic interactions:

- residue-residue pairwise interactions,

- atom-atom pairwise interactions,

- interactions corresponding to hydrogen bonds,

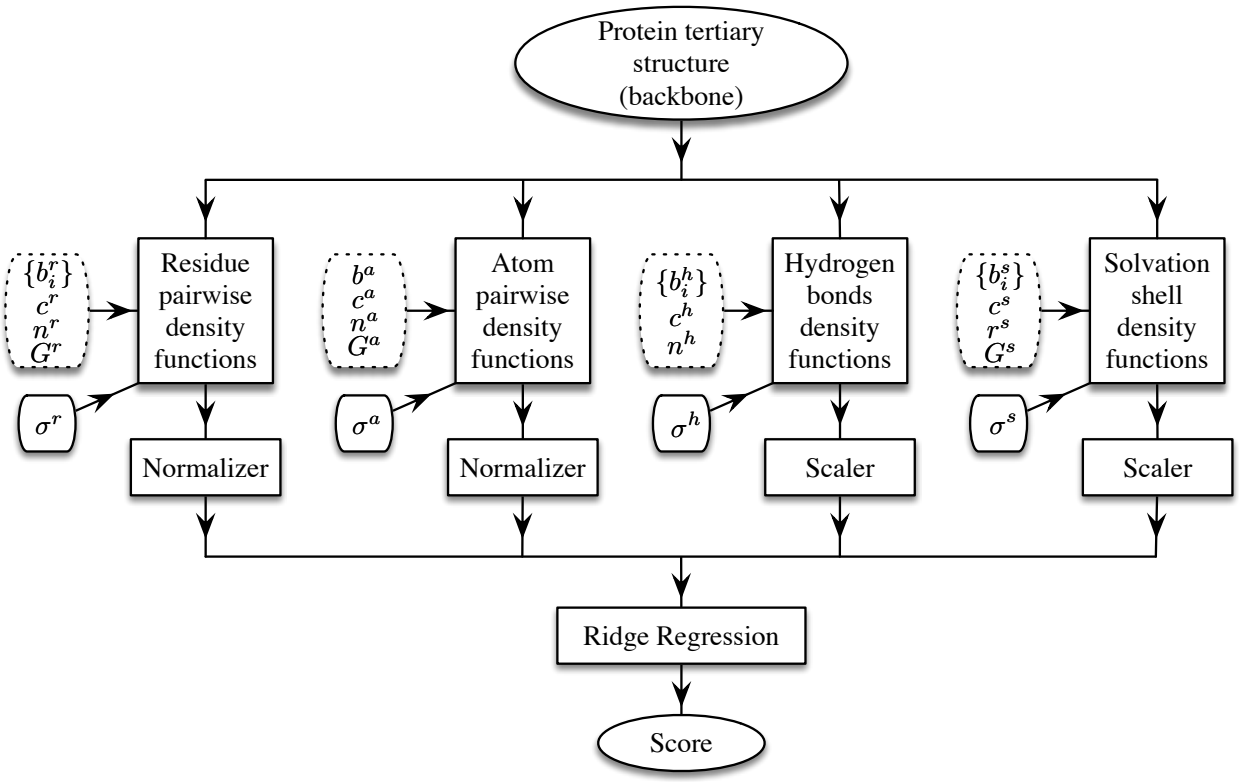


Figure 3-1: Workflow of the proposed QA method SBROD. The dotted blocks correspond to structural parameters, which are to be chosen on the stage of training

- solvent-solvate interactions from the generated hydration shell.

The procedure of feature extraction is the same for each group of features. We iterate over specific groups of atoms in the protein and estimate representing them descriptors — parameters describing the relative configuration of the considered atom group (a pair of atoms, a pair of residues, etc.), where the number of descriptors depends on the group of features. Then, we compute *number density* function of the estimated descriptors. We also apply smoothing with the truncated Gaussian kernel before computing number density function at the testing stage. We do not use the smoothing when training due to the huge dimension of the feature space.

The descriptors we use are shown in figure 3-2. Superscripts d correspond to the distances between pairs of alpha carbon atoms, θ denotes angle between two vectors, and $\varphi_{k,l}$ is the dihedral angles between two planes composed of atoms $\mathrm{C}_\alpha^k, \mathrm{C}_\alpha^l, \mathrm{C}_\beta^l$ and $\mathrm{C}_\alpha^l, \mathrm{C}_\alpha^k, \mathrm{C}_\beta^k$. While the interval allowed for angles θ is $[0, \pi]$, and for φ is $[0, 2\pi)$, the distance r can generally fall into $[0, \infty)$. However, we restrict it to the segment $[0, c]$ introducing a cutoff distance c. This simplification significantly increases the performance of the feature extraction procedure and makes generated features less noisy, since the interactions between objects on a distance larger than the cutoff distance are assumed to be negligible. Finally, to compute the number density functions (or histograms) we coarse-grain the intervals for each descriptor into b bins of equal width and accumulate the integrals of the smoothing kernel, centered in estimated descriptors, over these bins. Each procedure for feature extraction takes as input

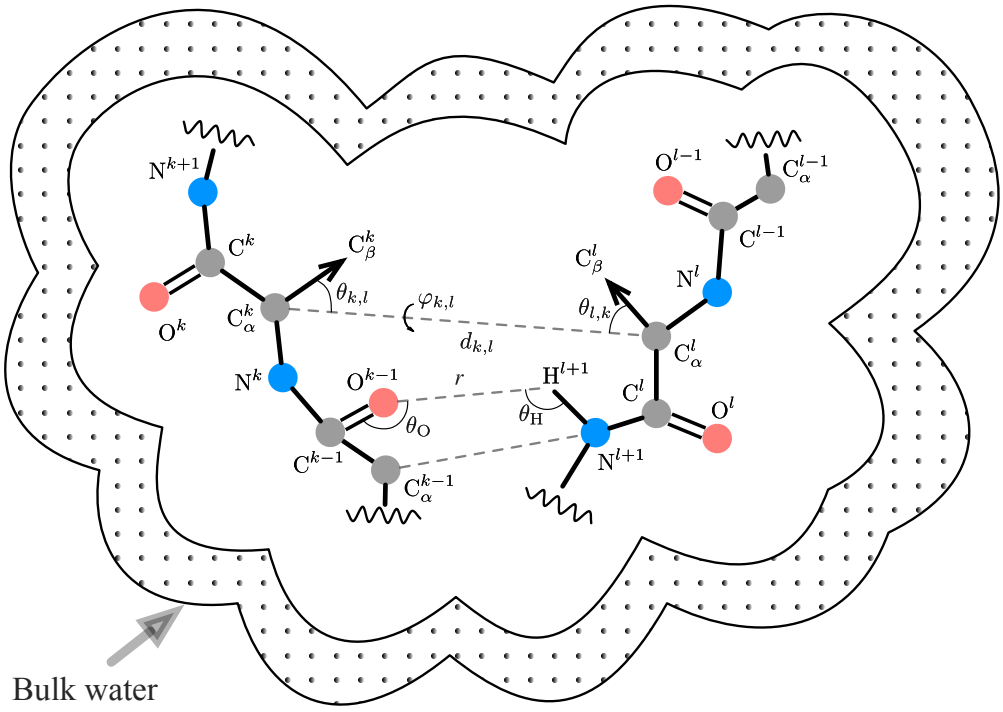


Figure 3-2: Geometrical features proposed

its parameters, which are shown in figure 3-1 from the left side for each of four blocks. Below we describe the four types of structural features with the corresponding parameters.

Residue pairwise features

We start with the description of the reduced representation of amino acids. More precisely, we assign to each amino acid a *rigid frame* such that its origin corresponds to the C_α atom, the z-axis points towards C_β, and the x-axis point towards nitrogen N. We treat all amino acids independently and accumulate their number density functions for each pair of amino acid types

$$G^r := \{\text{Ala}, \text{Arg}, \ldots, \text{Val}, \text{Sec}, \text{Mse}\} = \mathcal{A}. \tag{3.8}$$

Overall, we use 22×22 combinations of residues. We iterate over each pair of amino acids of the certain types within the cutoff distance $c^r + R_1 + R_2$, where R_1 and R_2 — side-chain radii for the considered amino acid types, and we describe the relative orientation of the corresponding rigid frames. For the cutoff distance, we use the value of $c^r = 5$Å. We use four descriptors, as it is shown in Figure 3-2. For a pair of amino acids k and l, these are the distance between the centers of the rigid frames $d_{k,l}$, the dihedral angle $\varphi_{k,l}$, and two angles $\theta_{k,l} = C^k_\beta C^k_\alpha C^l_\alpha$ and $\theta_{l,k} = C^l_\beta C^l_\alpha C^k_\alpha$. Then, we accumulate the number density functions as

follows:

$$d(i_1, i_2, i_3, i_4) = \sum_{(k,l)} \mathbb{1} \Bigg[\begin{aligned} &\frac{c^r}{b_1^r}(i_1 - 1) \leqslant d_{k,l} < \frac{c^r}{b_1^r} i_1, \\ &\frac{2\pi}{b_2^r}(i_2 - 1) \leqslant \varphi_{k,l} < \frac{2\pi}{b_2^r} i_2, \\ &\frac{\pi}{b_3^r}(i_3 - 1) \leqslant \theta_{k,l} < \frac{\pi}{b_3^r} i_3, \\ &\frac{\pi}{b_3^r}(i_4 - 1) \leqslant \theta_{l,k} < \frac{\pi}{b_3^r} i_4 \Bigg], \end{aligned} \tag{3.9}$$

where the sum is taken over all pairs (k, l) of amino acids of considering types within cutoff distance specified above, b_t^r — number of bins for the t-th descriptor, $i_t = 1, \ldots, b_t^r$ — bins into which the t-th descriptor's interval was coarse-grained.

Backbone atomic pairwise features

We use amino acid-specific backbone atom types. More precisely, we define types of heavy backbone atoms by their elements, which can be C, N, and O, and the corresponding amino acid types, so that

$$G^a := \{(\text{Ala}, \text{C}), \ldots, (\text{Ala}, \text{O}), (\text{Arg}, \text{C}), \ldots, (\text{Arg}, \text{O}), \ldots\} = \mathcal{A} \times \{\text{C}, \text{N}, \text{O}\}. \tag{3.10}$$

Overall, we use 22×3 backbone atom types. We iterate over each pair of atoms of certain types within the cutoff distance c^a and describe their relative position by a distance d between them. We use $c^a = 7$Å as the cutoff distance. Then, to accumulate the number density functions, we use the following equation:

$$d(i) = \sum_{(k,l)} \mathbb{1} \left[\frac{c^a}{b^a}(i - 1) \leqslant d_{k,l} < \frac{c^a}{b^a} i \right], \tag{3.11}$$

where the sum is taken over all pairs (k, l) of atoms of considering types within the cutoff distance specified above, b^a — number of bins, $i = 1, \ldots, b^a$ — bins into which the distance interval was coarse-grained.

Hydrogen bonds features

The third type of structural features is hydrogen bonds. For these, we use directional descriptors. To accumulate number density function for hydrogen bonds we, iterate over each pair of a donor atom N and an acceptor atom O within a cutoff distance c^h in the protein backbone. We use $c^h = 6$Å as the cutoff distance. To describe the directionality of this type of interactions, we use three descriptors. These are the distance d between the hydrogen atom H and the oxygen atom O, the donor angle $\theta_{\text{H}} = \text{NHO}$, and the acceptor angle $\theta_{\text{O}} = \text{HOC}$, as it is shown in Figure 3-2. Then, we accumulate the number density functions $d(i_1, i_2, i_3)$ similarly to the residue-residue features case.

Solvation features

To explicitly take into account the solvation effect, we construct a regular mesh of water molecules around the protein at a distance no further than $c^s = 20$Å. We describe each water molecule with the position of its oxygen atom and represent the solvent-solvate interactions using the reduced representation of amino acids. More precisely, for each pair (residue, water molecule) within a certain cutoff distance c^s, which is equal to 15Å, we calculate two descriptors. These are the distance d between the C_α atom of the amino acid and the oxygen O of the water molecule, and the angle θ between the z-axis of the residue's rigid frame and the vector $C_\alpha O$ pointing to the water oxygen. Then, we accumulate the number density function $d(i_1, i_2)$ similarly to the previous cases.

Smoothed density functions

To make the scoring function smooth, we calculate the number density function for a certain bin as follows. Instead of accumulating the number of hits of estimated descriptors into this bin, we accumulate the integrals of the truncated Gaussian kernel, centered in estimated descriptors, over this bin. The truncated Gaussian kernel with the support width of a half bin's width is chosen to preserve sparsity of the features. According to the proposing technique, the formula for accumulation of number density functions is changing as follows:

$$d(i) = \int_{\frac{c^a}{b^a}(i-1)}^{\frac{c^a}{b^a}i} G\left(x - d_{k,l};\, \sigma^a, \frac{c^a}{2b^a}\right) dx, \tag{3.12}$$

where G_h^σ is the truncated Gaussian kernel with support width $2h$:

$$G(x;\, \sigma, h) = \frac{\mathbb{1}[-h \leqslant x \leqslant h] f_\sigma(x)}{\int_{-h}^{h} f_\sigma(\xi) d\xi}, \quad f_\sigma(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}}. \tag{3.13}$$

Formulas for accumulating the residue pairwise, hydrogen-bonds, and water solvation number density functions are changing analogously.

There is one additional technique we use when processing hydration shell. By default, it is constructed by generating a grid of water molecules around the protein and then, removing those water molecules that are inside the protein (i.e. too close to the protein's atoms). This method leads to abrupt appearing and disappearing of water molecules when the protein structure fluctuates and hence affects the smoothness of the solvation features. We propose to overcome this issue through introducing smooth weights for the solvent that are close to the protein model and multiply the calculated integrals of the shifted kernel over the bins by these introduced weights. As long as both multipliers are continuous in the tertiary structure, the extracted features appear to be continuous as well. The weight w for a pair solvent-solvate equals to 0 for distances d between atoms shorter than minimal threshold $d_{\min}$

and rises to 1 when increasing the distance:

$$w = \begin{cases} 0, & d < d_{\min}, \\ \frac{d - d_{\min}}{\Delta}, & d_{\min} \leqslant d < d_{\min} + \Delta, \\ 1, & \text{otherwise.} \end{cases} \tag{3.14}$$

3.2.2 Machine learning

To train a scoring function, we apply classical machine learning technique Ridge Regression described below. It is used to find the direction in feature space, along which protein quality assessment is performed the best.

The problem of training a scoring function can be formulated as follows. Let $\mathfrak{D}$ be a set of protein domains

$$\mathfrak{D} = \{\mathcal{D}_1, \ldots, \mathcal{D}_n\}, \tag{3.15}$$

where $\mathcal{D}_i$ denotes a set of protein decoy models and their native structure $P_0^{(i)} \in S_{\text{t}}^m$:

$$\mathcal{D}_i = \{P_0^{(i)}, \ldots, P_{t_i}^{(i)}\} \subset \mathcal{A}^{m_i} \times \mathbb{S}_{\text{b}}^{m_i} \times \mathcal{R}^{m_i}. \tag{3.16}$$

Training set

First, we train the SBROD scoring function on server predictions from the earlier CASP[5-10] and their corresponding NMA decoy models (Hoffmann and Grudinin, 2017), to avoid overfitting and use the remaining CASP11 dataset for comparison the SBROD scoring function to state-of-the-art scoring functions. Then, the final SBROD scoring function is trained on the whole CASP[5-11] dataset and its corresponding NMA decoy models.

CASP decoy models. CASP[5-10] datasets (Moult *et al.*, 2003, 2005, 2007, 2009, 2011, 2014) were downloaded from the official CASP website at `http://predictioncenter.org/download_area`. These are the structures predicted by different servers participated in the CASP (Critical Assessment of protein Structure Prediction) experiments.

NMA decoy models. For each target structure in the CASP training set, we applied a tool for normal mode analysis developed by Hoffmann and Grudinin (2017). We varied 100 first normal modes of NMA model for each native structure in the training set to generate for each native structure per 300 decoy models within RMSD range $[0.5, 6]$.

Actual scores

Although there are multiple ways to measure the similarity between decoy protein models and native structures, the most accepted one in the protein structure prediction community is the GDT-TS (see section 3.1). The GDT-TS of a protein model is an average percent of residues from the scored model that can be superimposed with corresponding residues from the native structure under selected distance cutoffs $1, 2, 4, 8$Å. We use the TM-score utility developed by Zhang and Skolnick (2007) to compute GDT-TS (global distance test total

score by Zemla, 2003) of protein models. The computed GDT-TS of a protein model $P_j^{(i)}$ against its corresponding native structure $P_0^{(i)}$ is denoted by

$$S^*_{\text{GDT-TS}}(P_j^{(i)}, P_0^{(i)}) \in [0,1] \tag{3.17}$$

and treated as the actual score of the protein model $P_j^{(i)}$. Thus, function

$$S^*_{\text{GDT-TS}} : \bigcup_{m=1}^{\infty} \mathcal{A}^m \times \mathbb{S}_{\text{b}}^m \times \mathcal{R}^m \times S_{\text{t}}^m \to [0,1] \tag{3.18}$$

ranks all the decoy protein models $P_j^{(i)} \in \mathcal{D}_i$ for each $i = 1, \ldots, n$, such that

$$S^*_{\text{GDT-TS}}(P_0^{(i)}, P_0^{(i)}) = 1 \text{ for all the native structures } P_0^{(i)},\ i = 1, \ldots, n. \tag{3.19}$$

Ranking model

Let

$$\mathbf{f} : \bigcup_{m=1}^{\infty} \mathcal{A}^m \times \mathbb{S}_{\text{b}}^m \times \mathcal{R}^m \to \mathbb{R}^k \tag{3.20}$$

be a feature extractor described in section 3.2.1. We train a model of ranking

$$S_{\mathbf{w},\mathbf{f}} : \bigcup_{m=1}^{\infty} \mathcal{A}^m \times \mathbb{S}_{\text{b}}^m \times \mathcal{R}^m \to \mathbb{R} \tag{3.21}$$

minimizing the regularized empirical loss:

$$R(\mathbf{w}, \mathbf{b}) + \sum_{i=1}^{n} \sum_{j=0}^{t_i} L\left(S_{\mathbf{w},\mathbf{f}}(P_j^{(i)}) + b_i,\ S^*_{\text{GDT-TS}}(P_j^{(i)}, P_0^{(i)})\right) \to \min_{\mathbf{w},\,\mathbf{b}}. \tag{3.22}$$

We add additional parameters $b_i \in \mathbb{R},\ i = 1, \ldots, n$, since we are interested in capacity of the scoring function $S_{\mathbf{w},\mathbf{f}}(P)$ to rank the protein models, and hence the loss function should not depend on the per-domain shifts of the scoring function $S_{\mathbf{w},\mathbf{f}}(P)$.

We use a linear model of the ranking function $S_{\mathbf{w},\mathbf{f}}$:

$$S_{\mathbf{w},\mathbf{f}}(P) = \langle \mathbf{w}, \mathbf{f}(P) \rangle, \tag{3.23}$$

with squared loss function and ridge regularization:

$$L(x,y) = (x-y)^2, \quad R(\mathbf{w}, \mathbf{b}) = \alpha \left(\|\mathbf{w}\|_2^2 + \frac{1}{\beta^2} \|\mathbf{b}\|_2^2 \right). \tag{3.24}$$

Therefore, training optimization problem (3.22) can be rewritten as follows:

$$\alpha\left(\|\mathbf{w}\|_2^2+\frac{1}{\beta^2}\|\mathbf{b}\|_2^2\right)+\sum_{i=1}^{n}\sum_{j=0}^{t_i}\left(\left\langle\mathbf{w},\mathbf{f}(P_j^{(i)})\right\rangle+b_i-S^*_{\text{GDT-TS}}(P_j^{(i)},P_0^{(i)})\right)^2\to\min_{\mathbf{w},\mathbf{b}},\quad(3.25)$$

Note, that this can be turned into standard ridge regression form

$$\alpha\|\tilde{\mathbf{w}}\|_2^2+\sum_{i=1}^{n}\sum_{j=0}^{t_i}\left(\left\langle\tilde{\mathbf{w}},\tilde{\mathbf{f}}(P_j^{(i)})\right\rangle-S^*_{\text{GDT-TS}}(P_j^{(i)},P_0^{(i)})\right)^2\to\min_{\tilde{\mathbf{w}}},\quad(3.26)$$

where

$$\tilde{\mathbf{f}}(P_j^{(i)})=\Big[\mathrm{f}_1(P_j^{(i)}),\ldots,\mathrm{f}_k(P_j^{(i)}),\underbrace{0,\ldots,0,\beta}_{i},0,\ldots,0\Big]^{\mathsf{T}}\in\mathbb{R}^{k+n},\quad(3.27)$$

$$\tilde{\mathbf{w}}=[w_1,\ldots,w_k,b_1,\ldots,b_n]^{\mathsf{T}}\in\mathbb{R}^{k+n}.\quad(3.28)$$

Regression model

To adapt the proposed ranking model for protein design problem, i.e. to make the trained scoring function applicable for the protein design problem, it is enough to omit the domains shifting parameters b_j in formula (3.22). Without shifting parameters the ranking model transforms into ordinary ridge regression model that is trained to predict the actual score $S^*_{\text{GDT-TS}}$ (3.18). Therefore, once the actual score of each native structure is the same, the training is performed to satisfy condition (3.37) as well as to optimize the overall quality of quality assessment.

Optimization. The optimization problem (3.26) was reduced to a linear equation system, and it was solved by the Conjugate Gradient iteration implemented in SciPy python library (Jones *et al.*, 2001), which is adapted to sparse matrices of huge dimension.

Cross-validation. To estimate the best parameters of feature extraction procedure $\mathbf{f}$: b_i^r, c^r, b^a, b^a, etc. (see figure 3-1), and also the best regularization parameters α and β (see equation (3.26)), we used 3-fold cross-validation on the CASP[5-10] datasets. k-fold cross-validation is the standard technique to estimate the free parameters of the model as follows. The original dataset is randomly partitioned into k equal sized parts. Of the k parts, a single part is retained as the validation set for testing the model, trained on the remaining $k-1$ parts. The cross-validation process is then repeated k times with each of the k parts used exactly once as the validation data. The k results from the folds are then averaged to produce a single estimation serving as a criterion of picking the best free parameters of the model. Thus, all the training data is used for both training and validation, and none of the test data is used here, which helps to avoid overfitting. As a result, the regularization parameters were chosen to be $\alpha=5$, $\beta=50$. The optimal parameters of the feature generation procedure were mentioned above in subsection 3.2.1.

3.3 Rotamer prediction

In this section, we will formally set the rotamer prediction problem. Let protein backbone $\boldsymbol{b}^0 \in \mathbb{S}_{\text{b}}^m$ and sequence $\boldsymbol{a}^0 \in \mathcal{A}^m$ be given, where m is the protein length. Let

$$E: \ \mathcal{A}^m \times \mathbb{S}_{\text{b}}^m \times \mathcal{R}^m \to \mathbb{R} \tag{3.29}$$

denote a protein energy. Let us consider a problem of discrete programming

$$E(\boldsymbol{a}^0, \boldsymbol{b}^0, \boldsymbol{r}) \to \min_{\boldsymbol{r} \in \mathcal{R}^m}, \tag{3.30}$$

which is called the rotamer prediction problem (predict folding of side-chains given a coarse-grained protein model). The search space of problem (3.30) is of huge cardinality $|\mathcal{R}|^m$, which is $\sim 10^{300}$ for typical protein of length $m = 300$ and number of rotamers per side chain $|\mathcal{R}| = 10$. Hence exhaustive search cannot be applied to find optimal solution and solve it. To find approximate solution of this problem, algorithms of discrete optimization (see sections 3.5.1 and 3.5.2) can be applied.

To assess the quality of a solution of rotamer prediction problem (3.30), the quality criteria described in section 3.1 are used. If energy function $E(\boldsymbol{a}^0, \boldsymbol{b}^0, \boldsymbol{r})$ is pairwise decomposable, that is, equal to the sum of symmetrical pairwise energy potentials of rotamer interactions:

$$E(\boldsymbol{a}^0, \boldsymbol{b}^0, \boldsymbol{r}) = \sum_{k=1}^{m} \sum_{l=1}^{m} E_{kl}(r_k, r_l), \tag{3.31}$$

where r_k and r_l — rotamers of amino acids a_k and a_l respectively, then the rotamer prediction problem (3.30) is equivalent to the following discrete optimization problem:

$$\sum_{k=1}^{m} \sum_{l=1}^{m} E_{kl}(r_k, r_l) \to \min_{(r_1, \ldots, r_m)}. \tag{3.32}$$

Problem (3.32) is set as problem (3.54). Therefore, methods described in section 3.5 can be used to solve it or find an approximate solution.

3.4 Protein design problem

In this section, we will formally set the protein design problem. Let protein backbone $\boldsymbol{b}^0 \in \mathbb{S}_{\text{b}}^m$ be given, where m is the protein length. Let

$$E: \ \mathcal{A}^m \times \mathbb{S}_{\text{b}}^m \times \mathcal{R}^m \to \mathbb{R} \tag{3.33}$$

denote a protein energy. Let us consider a problem of discrete programming

$$E(\boldsymbol{a}, \boldsymbol{b}^0, \boldsymbol{r}) - \min_{\boldsymbol{b} \in \mathbb{S}_b^m, \boldsymbol{r} \in \mathcal{R}^m} E(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{r}) \to \min_{\boldsymbol{a} \in \mathcal{A}^m, \boldsymbol{r} \in \mathcal{R}^m}, \tag{3.34}$$

which is to predict a protein sequence $\boldsymbol{a} \in \mathcal{A}^m$ that would be likely to fold into the target tertiary backbone structure $\boldsymbol{b}^0 \in \mathbb{R}^{m\times 3\times 3}$ (see problem (1.6)). As we showed in section 1.2.3, problem (3.34) can written as follows

$$E(\boldsymbol{a}, \boldsymbol{b}^0) - \min_{\boldsymbol{b}\in\mathbb{S}_b^m} E(\boldsymbol{a}, \boldsymbol{b}) \to \min_{\boldsymbol{a}\in\mathcal{A}^m}, \tag{3.35}$$

where

$$E(\boldsymbol{a}, \boldsymbol{b}) := \min_{\boldsymbol{r}\in\mathcal{R}^m} E(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{r}). \tag{3.36}$$

One can see that problem statement (3.35) involves solving rotamer prediction problem to find coarse-grained energy function (3.36) for each possible sequence $\boldsymbol{a} \in \mathcal{A}^m$ (number of these sequences is typically $\sim 20^m$), what makes this problem statement intractable. However, if the energy function is the same for all the native structures, that is, the following condition is satisfied

$$\min_{\boldsymbol{b}\in\mathbb{S}_{\mathrm{b}}^m,\, \boldsymbol{r}\in\mathcal{R}^m} E(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{r}) = \mathrm{const}(\boldsymbol{a}), \tag{3.37}$$

then problem (3.34) can be rewritten as follows:

$$E(\boldsymbol{a}, \boldsymbol{b}^0, \boldsymbol{r}) \to \min_{\boldsymbol{a}\in\mathcal{A}^m,\, \boldsymbol{r}\in\mathcal{R}^m}, \tag{3.38}$$

which is called the protein design problem. The search space of problem (3.38) is of a huge cardinality $\sim 200^{300}$ for a typical protein of length $m = 300$ and number of rotamers per side chain $|\mathcal{R}_i| = 10$. Hence exhaustive search cannot be applied to find optimal solution and solve it. To find approximate solution of this problem, algorithms of discrete optimization (see sections 3.5.1 and 3.5.2) can be applied.

On the other hand, if the energy function in (3.38) is devised to predict the quality of backbone folding of coarse-grained protein structure:

$$S:\ \mathcal{A}^m \times \mathbb{S}_{\mathrm{b}}^m \to \mathbb{R}, \tag{3.39}$$

where scores of all native protein structures are the same (the best backbone folding score does not depend on a sequence), for example, trained to predict GDT-TS (see section 3.1), then the protein design problem can be simplified as follows:

$$S(\boldsymbol{a}, \boldsymbol{b}^0) \to \min_{\boldsymbol{a}\in\mathcal{A}^m}. \tag{3.40}$$

The search space of problem (3.40) is significantly reduced in comparison to problem (3.38). Therefore, usage of the scoring function that assesses coarse-grained protein models is preferable for solving the protein design problem.

The main issue of the protein design problem is the assessment of quality of a solution of problem (3.40), that is, finding a quality criterion for assessing a method for protein design $\hat{\varphi}_{\mathrm{d}}$ either approximate or precise:

$$\hat{\varphi}_{\mathrm{d}}(\boldsymbol{b}^0) \approx \operatorname*{Arg\,min}_{\boldsymbol{a}\in\mathcal{A}} S(\boldsymbol{a}, \boldsymbol{b}^0). \tag{3.41}$$

In this paper, we propose the following quality criteria:

$$Q(\hat{\varphi}_{\mathrm{d}}, \boldsymbol{b}) = \frac{1}{|\varphi_{\mathrm{d}}(\boldsymbol{b})|} \sum_{\boldsymbol{a} \in \varphi_{\mathrm{d}}(\boldsymbol{b})} \|\boldsymbol{b} - (\pi_{\mathrm{rb}} \circ \hat{\varphi}_{\mathrm{cg}})(\boldsymbol{a})\|\,, \tag{3.42}$$

where

$$\hat{\varphi}_{\mathrm{cg}} : \ \mathcal{A}^m \to \mathcal{A}^m \times \mathbb{S}^m_{\mathrm{b}} \tag{3.43}$$

is a coarse-grained protein folding algorithm. This criterion checks how well protein sequences predicted by protein design method $\hat{\varphi}_{\mathrm{d}}$ fold into a target backbone $\boldsymbol{b}$ according to a protein folding algorithm $\hat{\varphi}_{\mathrm{cg}}$. Hence criterion (3.42) requires a protein folding algorithm, which is usually is imprecise as well and serves as an approximation.

If energy function $S(\boldsymbol{a}, \boldsymbol{b}^0)$ is pairwise decomposable, that is, equal to the sum of symmetrical pairwise energy terms:

$$S(\boldsymbol{a}, \boldsymbol{b}^0) = \sum_{k=1}^{m} \sum_{l=1}^{m} E_{kl}(a_k, a_l), \tag{3.44}$$

where a_k and a_l — types of amino acids a_k and a_l respectively, then protein design problem (3.40) is equivalent to the following discrete optimization problem:

$$\sum_{k=1}^{m} \sum_{l=1}^{m} E_{kl}(a_k, a_l) \to \min_{(a_1, \ldots, a_m)}. \tag{3.45}$$

Problem (3.45) is set as problem (3.54). Therefore, methods described in section 3.5 can be used to solve it or find an approximate solution.

3.4.1 Amino acid distribution

One can see that protein design problem (3.45) does not contain any restrictions on the proposed sequences. Though, in nature, not each protein sequence is able to fold into a stable tertiary structure under normal conditions. Hence we are interested in finding protein sequences that are of a similar pattern to those that have been already discovered in nature. One possible constraint that can be introduced is a restriction on the distribution of different amino acid types (we want to encourage the presence of the whole variety of amino acids in determined sequences and to penalize those sequences that contain too many amino acids of a certain type, which is not common in discovered proteins). In this section, we introduce an energy modification to take into account a certain amino acid distribution. To do that, let us rewrite the protein design problem in probabilistic context and introduce marginal distribution for amino acids, which is a distribution on a set of all sequences.

First, let us write the probabilistic problem statement for the protein design problem:

$$p(\boldsymbol{a}|\boldsymbol{b}^0) \to \max_{\boldsymbol{a}}, \tag{3.46}$$

where probability for protein $\boldsymbol{a}$ to fold into conformation $\boldsymbol{b}^0$ is defined by Boltzmann distri-

bution

$$p(\boldsymbol{b}^0|\boldsymbol{a}) = \exp\left(-\frac{E(\boldsymbol{a},\boldsymbol{b}^0)}{T}\right) \tag{3.47}$$

(we put the normalizing constant to energy function) and energy function $E(\boldsymbol{a},\boldsymbol{b}^0)$ is pairwise decomposable, i.e. can be represented in form (3.44).

Marginal distribution for amino acids

Let us follow the unigram language model when the order of amino acids is irrelevant. This model sets multinomial distribution over amino acid sequences.

$$p(a_1,\ldots,a_m) = m!\prod_{a\in\mathcal{A}}\frac{p_a^{m_a}}{m_a!}, \tag{3.48}$$

where

$$m_a = \sum_{i=1}^{m}\mathbb{1}[a_i = a],\ a\in\mathcal{A}. \tag{3.49}$$

However, energy correction defined by this model will not be pairwise decomposable. Hence, let us approximate multinomial distribution (3.48) as follows:

$$p(a_1,\ldots,a_m) = C\prod_{a\in\mathcal{A}}\mathcal{N}\left(m_a|\ mp_a, m\sigma_a^2\right), \tag{3.50}$$

where $\mathcal{N}\left(x|\ \mu,\sigma^2\right) = \frac{1}{\sqrt{2\pi\sigma^2}}e^{-\frac{(x-\mu)^2}{2\sigma^2}}$ is the probability density function of the Gaussian distribution, p_a and σ_a for each $a\in\mathcal{A}$ are constant parameters not dependent on $a_1,\ldots,a_m$, and C is a normalizing constant. Lemma 3.4.1 proves that this approximation provides pairwise decomposable energy correction. Therefore, the total energy function is preserved to be pairwise decomposable and hence the protein design optimization problem still can be reduced to BQP programming problem.

Lemma 3.4.1 (Energy corrections) *Let energy $E(\boldsymbol{a},\boldsymbol{b}^0)$ in (3.47) be pairwise decomposable:*

$$E(\boldsymbol{a},\boldsymbol{b}^0) = \sum_{k=1}^{m}\sum_{l=1}^{m}E_{kl}(a_k,a_l), \tag{3.51}$$

and marginal distribution $p(a_1,\ldots,a_m)$ be defined by (3.50). Then, after introducing energy correction terms as follows:

$$E'_{kl}(a_k,a_l) := \begin{cases}\frac{T}{2m}\cdot\frac{1-2p_{a_k}}{\sigma^2_{a_k}}, & a_k = a_l;\\ -\frac{T}{2m}\cdot\left(\frac{p_{a_k}}{\sigma^2_{a_k}}+\frac{p_{a_l}}{\sigma^2_{a_l}}\right), & a_k\neq a_l,\end{cases} \tag{3.52}$$

problem (3.46) is equivalent to minimization of the total energy with the introduced energy

correction terms:

$$\sum_{k=1}^{m}\sum_{l=1}^{m} E_{kl}^{total}(a_k, a_l) = \sum_{k=1}^{m}\sum_{l=1}^{m} [E_{kl}(a_k, a_l) + E'_{kl}(a_k, a_l)] \to \min_{(a_1,\ldots,a_m)}. \tag{3.53}$$

See the proof in the appendix.

3.5 Pairwise decomposable energy minimization

In this section we will consider the problem of minimization a pairwise decomposable objective — the sum of pairwise functions of integer variable:

$$\sum_{k=1}^{m}\sum_{l=1}^{m} E_{kl}(y_k, y_l) \to \min_{\substack{y_1\in\{1,\ldots,n_1\}\\ \cdots\cdots\cdots\\ y_m\in\{1,\ldots,n_m\}}}, \tag{3.54}$$

where $n_i \in \mathbb{N},\ i = 1, \ldots, m$.

The problem (3.54) is a problem of discrete optimization, hence algorithms for discrete optimization can be applied. First, we will consider such algorithms for discrete optimization as greedy optimization in section 3.5.1 and simulated annealing in section 3.5.2. Then, we will reduce the pairwise decomposable energy minimization problem (3.54) to the boolean quadratic programming problem (3.66).

3.5.1 Greedy optimization algorithm

An iterative algorithm is called greedy when it makes a locally optimal choice on each iteration when solving the decision problem. The latter heuristic is aimed at finding a decent global optimal solution in a reasonable time. Although there are no theoretical guarantees on the quality of found solution in general case, in some special cases (as finding a maximum-weight basis of a matroid) this paradigm provides the exact solution.

For the discrete optimization problem (3.54), it is reasonable to propose greedy optimization algorithm 3.5.1, which follows the greedy heuristic in a hope to find a global minimum of function

$$f(y_1, \ldots, y_m) = \sum_{k=1}^{m}\sum_{l=1}^{m} E_{kl}(y_k, y_l). \tag{3.55}$$

Algorithm 3.5.1 Simple Greedy Search

Input: $(y_1, \ldots, y_m)$, $y_k \in \{1, \ldots, n_k\}$, $k = 1, \ldots, m$ *# starting point*
Output: $(y_1, \ldots, y_m)$ *# approximate solution to problem* (3.54)

repeat
  $n \leftarrow 0$
  for all $k = 1, \ldots, m$ **do**
    $\hat{y}_k := \arg\min\limits_{y \in \{1, \ldots, n_k\}} f(y_1, \ldots, y_{k-1}, y, y_{k+1}, \ldots, y_m)$
    if $\hat{y}_k \neq y_k$ **then**
      $y_k := \hat{y}_k$
      $n := n + 1$
until $n > 0$
return $(y_1, \ldots, y_m)$

On each iteration, it tries all possible values for the optimized variable and chooses one that minimizes the objective the best.

Note that on each stage of algorithm 3.5.1 an optimization problem is to be solved, which is to minimize objective over one variable. Now let us consider the function $f(y_1, \ldots, y_m)$ as a function of vector argument $f(\boldsymbol{y}) = f(y_1, \ldots, y_m)$. Hence, we can make the following interpretation of the optimization problems that are to be solved on each stage of algorithm 3.5.1. It is minimization of the objective on the neighborhood $N_1(\boldsymbol{y})$ of current approximation, where

$$N_t(\boldsymbol{y}) = \left\{ \boldsymbol{y}' \in \times_{i=1}^{m} \{1, \ldots, n_i\} \;\middle|\; \|\boldsymbol{y} - \boldsymbol{y}'\|_0 \leqslant t \right\}, \tag{3.56}$$

where $\|\boldsymbol{y}\|_0 = |\{k \in \{1, \ldots, m\} \mid y_k \neq 0\}|$ is a cardinality function — the number of non-zero elements in a vector. One can notice that an obvious possible modification would be expanding the neighborhood. This brings us to algorithm 3.5.2.

Algorithm 3.5.2 Greedy Search

Input: $t \in \mathbb{N}$, $\boldsymbol{y} \in \times_{i=1}^{m} \{1, \ldots, n_i\}$ *# starting point*
Output: $\boldsymbol{y} \in \times_{i=1}^{m} \{1, \ldots, n_i\}$ *# appr. sol. to problem* (3.54)

repeat
  $\boldsymbol{y}_{\text{old}} := \boldsymbol{y}$
  $\boldsymbol{y} := \arg\min\limits_{\boldsymbol{y}' \in N_t(\boldsymbol{y}_{\text{old}})} f(\boldsymbol{y}')$
until $\boldsymbol{y} \neq \boldsymbol{y}_{\text{old}}$
return $\boldsymbol{y}$

Algorithm 3.5.2 is more flexible than 3.5.1. However, the search space cardinality of the optimization problem on each iteration is

$$C_{\text{total}}(t) = \sum_{I \in \binom{\{1, \ldots, m\}}{t}} \prod_{i \in I} n_i. \tag{3.57}$$

In the extreme case $t = m$, it turns into exhaustive search with the complexity of

$$C_{\text{total}}(m) = \prod_{i=1}^{m} n_i. \tag{3.58}$$

3.5.2 Simulated annealing

Simulated annealing is a probabilistic technique aimed at approximating the global minimum of a given function. Initially, it was proposed by Khachaturyan *et al.* (1981); Semenovskaya *et al.* (1985) as a technique for solving problems where finding a better approximate optimal solution is more important than finding a local optimum. It explores the solution space iteratively. On each iteration, it considers a neighboring point $\boldsymbol{y}' \in N_t(\boldsymbol{y})$ (see (3.56)) of the current approximate solution $\boldsymbol{y}$ and decides whether to accept new point as a new approximate solution or not. The same iterations are repeated until a given computational time is exhausted. Similarly to greedy search 3.5.2, it accepts as new approximate solution points from the neighborhood of current approximate solution at which the objective function is lower than the value of the objective function in current approximate solution. However, simulated annealing leaves the probability to accept neighboring points leading to the objective function growth in order not to get stuck in the local optima. This probability $\mathsf{P}(\boldsymbol{y} \to \boldsymbol{y}', T)$ depends on the values of the objective function in current candidate points $f(\boldsymbol{y})$, $f(\boldsymbol{y}')$ and a temperature T — global time varying parameter. It is common to use the following probability function:

$$\mathsf{P}(\boldsymbol{y} \to \boldsymbol{y}', T) = \begin{cases} 1, & f(\boldsymbol{y}') < f(\boldsymbol{y}), \\ \exp\left(-\frac{f(\boldsymbol{y}') - f(\boldsymbol{y})}{T}\right), & \text{otherwise.} \end{cases} \tag{3.59}$$

The pseudocode for the simulated annealing algorithm is listed bellow.

Algorithm 3.5.3 Simulated Annealing

Input: t, $k_{\max} \in \mathbb{N}$, $\boldsymbol{y} \in \times_{i=1}^{m}\{1, \ldots, n_i\}$ *# starting point*
Output: $\boldsymbol{y} \in \times_{i=1}^{m}\{1, \ldots, n_i\}$ *# appr. sol. to problem* (3.54)

1: **for** $k = 1, \ldots, k_{\max}$ **do**
2: $\quad T := \frac{k}{k_{\max}}$
3: $\quad$ Pick randomly a neighbor $\boldsymbol{y}' \in N_t(\boldsymbol{y}) \subset \times_{i=1}^{m}\{1, \ldots, n_i\}$
4: $\quad$ **if** $\xi \leqslant \mathsf{P}(\boldsymbol{y} \to \boldsymbol{y}', T)$, $\xi \sim U[0, 1]$ **then**
5: $\quad\quad \boldsymbol{y} := \boldsymbol{y}'$ *# update approximate solution with probability defined in* (3.59)
6: **return** $\boldsymbol{y}$

Here $U[0, 1]$ is a uniform distribution that has the following probability density function:

$$p(x) = \begin{cases} 1, & 0 \leqslant x \leqslant 1, \\ 0, & \text{otherwise.} \end{cases} \tag{3.60}$$

3.5.3 Reduction to the BQP

Let us derive the reduction of the pairwise decomposable energy minimization problem (3.54) to the boolean quadratic programming problem (3.66).

Let $\mathbf{Q}$ be the energy block matrix as follows:

$$\mathbf{Q} = \begin{bmatrix} [E_{11}(y_1, y_1)] & [E_{12}(y_1, y_2)] & \cdots & [E_{11}(y_1, y_m)] \\ [E_{21}(y_2, y_1)] & [E_{22}(y_2, y_2)] & \cdots & [E_{2m}(y_2, y_m)] \\ \vdots & \vdots & \ddots & \vdots \\ [E_{m1}(y_m, y_1)] & [E_{m2}(y_m, y_2)] & \cdots & [E_{mm}(y_m, y_m)] \end{bmatrix}, \tag{3.61}$$

of blocks

$$[E_{kl}(y_k, y_l)] = \begin{bmatrix} E_{kl}(1,1) & E_{kl}(1,2) & \cdots & E_{kl}(1,n_l) \\ E_{kl}(2,1) & E_{kl}(2,2) & \cdots & E_{kl}(2,n_l) \\ \vdots & \vdots & \ddots & \vdots \\ E_{kl}(n_k,1) & E_{kl}(n_k,2) & \cdots & E_{kl}(n_k,n_l) \end{bmatrix}, \tag{3.62}$$

Then the problem (3.54) is equivalent to the following one:

$$\begin{aligned} \underset{\boldsymbol{x}}{\text{minimize}} \quad & \boldsymbol{x}^\intercal \mathbf{Q} \boldsymbol{x} \\ \text{subject to} \quad & \boldsymbol{x} = [\boldsymbol{x}^{1\intercal}, \dots, \boldsymbol{x}^{m\intercal}]^\intercal \\ & \boldsymbol{x}^k \in \{0,1\}^{n_k}, \quad k = 1, \dots, m, \\ & \|\boldsymbol{x}^k\|_0 = 1, \quad k = 1, \dots, m. \end{aligned} \tag{3.63}$$

The dimensionality of this problem is

$$n = \sum_{k=1}^{m} n_k. \tag{3.64}$$

Let us introduce the binary matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$:

$$\mathbf{A} = \begin{bmatrix} 1 & \cdots & 1 & 0 & \cdots & 0 & \cdots\cdots & 0 & \cdots & 0 \\ 0 & \cdots & 0 & 1 & \cdots & 1 & \cdots\cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \cdots\cdots & \vdots & \ddots & \vdots \\ \underbrace{0 \cdots 0}_{n_1} & & & \underbrace{0 \cdots 0}_{n_2} & & & \cdots\cdots & \underbrace{1 \cdots 1}_{n_m} & & \end{bmatrix}. \tag{3.65}$$

Then, the problem (3.63) can be written as problem (3.66).

The proposed reduction enables usage of convex relaxation techniques (section 3.6) for solving the initial decomposable energy minimization problem (3.54).

3.6 Boolean quadratic programming

- $\mathcal{S}^n = \{X \in \mathbb{R}^{n \times n} \mid X = X^\intercal\}$ — space of symmetric matrices,

- $\mathcal{S}_+^n = \{X \in \mathbb{R}^{n\times n} \mid X = X^\intercal \succeq 0\}$ — space of symmetric positive semidefinite matrices,

In this section, we investigate the optimization problem

$$\begin{aligned} &\underset{\boldsymbol{x}\in\{0,1\}^n}{\text{minimize}} && \boldsymbol{x}^\intercal \mathbf{Q}\boldsymbol{x} \\ &\text{subject to} && \mathbf{A}\boldsymbol{x} = \mathbf{1}_m, \end{aligned} \tag{3.66}$$

where

- $\mathbf{Q} \in \mathcal{S}^n$ — symmetric real matrix,
- $\mathbf{A} \in \{0,1\}^{m\times n}$ — boolean matrix of form (3.65),
- $\mathbf{1}_m = [\underbrace{1,\ldots,1}_{m}]^\intercal$ — vector of all ones.

Problem (3.66) is NP-hard in general case (even if the matrix $\mathbf{Q}$ is positive semidefinite: $\mathbf{Q} \in \mathcal{S}_+^n$) since the problem MAXCUT, which is known to be NP-complete, can be reduced to it. Hence problem (3.66) cannot be solved efficiently in a reasonable time in the assumption that P $\neq$ NP. In this section, we will discuss possible techniques for relaxing the non-convex problem (3.66) into convex optimization problem.

3.6.1 Continuous relaxation

First, let us regularize the matrix of the quadratic form $\mathbf{Q}$:

$$\hat{\mathbf{Q}} := \underbrace{\mathbf{Q} - \lambda_{\min}(\mathbf{Q}) \cdot \mathbf{I}_n}_{\in\mathcal{S}_+^n}, \tag{3.67}$$

where $\lambda_{\min}(\mathbf{Q})$ is the smallest eigenvalue of the $\mathbf{Q}$ matrix:

$$\lambda_{\min}(\mathbf{Q}) = \min\{\lambda \in \mathbb{R} \mid \det(\mathbf{Q} - \lambda\mathbf{I}_n) = 0\}. \tag{3.68}$$

Now, since $\boldsymbol{x} \in \{0,1\}^n$, and hence

$$\boldsymbol{x}^\intercal\mathbf{I}_n\boldsymbol{x} = \boldsymbol{x}^\intercal\boldsymbol{x} = \sum_{i=1}^n x_i^2 = \sum_{i=1}^n x_i = \mathbf{1}_n^\intercal\boldsymbol{x}, \tag{3.69}$$

the BQP problem (3.66) can be rewritten as follows:

$$\begin{aligned} &\underset{\boldsymbol{x}\in\{0,1\}^n}{\text{minimize}} && \boldsymbol{x}^\intercal\hat{\mathbf{Q}}\boldsymbol{x} + \lambda_{\min}(\mathbf{Q})\mathbf{1}_n^\intercal\boldsymbol{x} \\ &\text{subject to} && \mathbf{A}\boldsymbol{x} = \mathbf{1}_m. \end{aligned} \tag{3.70}$$

Here the matrix of the quadratic form is positive semidefinite. Therefore, relaxing the constraints $\boldsymbol{x} \in \{0,1\}^n$ to convex constraints $\boldsymbol{x} \in [0,1]^n$, we relax non-convex problem (3.70)

to a convex one:

$$\begin{aligned} \underset{\boldsymbol{x}}{\text{minimize}} \quad & \boldsymbol{x}^\intercal \hat{\mathbf{Q}} \boldsymbol{x} + \lambda_{\min}(\mathbf{Q}) \mathbf{1}_n^\intercal \boldsymbol{x} \\ \text{subject to} \quad & \mathbf{A}\boldsymbol{x} = \mathbf{1}_m, \\ & \mathbf{0}_n \leqslant \boldsymbol{x} \leqslant \mathbf{1}_n. \end{aligned} \tag{3.71}$$

Note that the optimal value of relaxation (3.71) provides the lower bound on the optimal value of initial BQP problem (3.66).

Problem (3.71) is convex, and more precisely, it is a convex quadratic programming problem, and hence it can be solved efficiently (Boyd and Vandenberghe, 2004; Gill and Wong, 2015). There are many well-developed packages to solve quadratic programming problems, such as Gurobi (Gurobi Optimization, 2016), MOSEK (ApS, 2015), etc.

Rounding strategy

To get an upper bound on the optimal value of initial BQP problem (3.66), one needs to perform the so-called rounding procedure, when the obtained optimal solution of relaxation (3.71) is projected onto the feasible set. Rounding procedure is highly important because it links the optimal solution of the relaxation and an approximate solution of the initial non-convex problem. Though it is complicated to derive the projection in general case, we will do that for constraint matrices of special form (3.65). In this case, the following rounding strategy can be proposed:

$$\hat{x}_{\sum_{i=1}^{t-1} n_i + k} := \begin{cases} 1, & k = \hat{\arg\max}_{j=1,\ldots,n_t} x_{\sum_{i=1}^{t-1} n_i + j}, \\ 0, & \text{otherwise}, \end{cases} \qquad k = 1, \ldots, n_t, \; t = 1, \ldots, m, \tag{3.72}$$

where

$$\hat{\arg\max}_j x_j = \min \left(\operatorname{Arg\,max}_j x_j \right). \tag{3.73}$$

It is clear that formula (3.72) defines a projection of a point $\boldsymbol{x} \in [0,1]^n$ onto the search space $V = \{\boldsymbol{x} \in \{0,1\}^n \,|\, \mathbf{A}\boldsymbol{x} = \mathbf{1}_m\}$, that is,

$$\|\boldsymbol{x} - \hat{\boldsymbol{x}}\| = \min_{\boldsymbol{x}' \in V} \|\boldsymbol{x} - \boldsymbol{x}'\|.$$

We also can randomly sample neighboring points $\boldsymbol{x}'$ around the found optimal solution of relaxation (3.71):

$$\boldsymbol{x}' \sim \mathcal{N}(\boldsymbol{x}, \operatorname{diag}(x_1 - x_1^2, \ldots, x_n - x_n^2)), \tag{3.74}$$

and then perform rounding procedure (3.72). This sampling technique generates new points from the distribution taking into account the optimal solution of the relaxation. These generated points are projected onto the feasible set, and the obtained rounded approximations might correspond to the objective values that are lower than initially obtained upper bound. Though these rounding and sampling procedures are heuristics, often they help to tighten the gap between the upper and lower bounds significantly (D'Aspremont and Boyd, 2003).

3.6.2 Lagrangian relaxation

The Lagrangian relaxation technique is based on the fact that the dual of any optimization problem is always convex and its solution provides a lower bound for the primal problem. First, let us derive the Lagrangian relaxation for non-convex problem (3.66).

$$\begin{aligned} L(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{u}) &= \boldsymbol{x}^\mathsf{T}\mathbf{Q}\boldsymbol{x} + \sum_{i=1}^{n} \lambda_i (x_i^2 - x_i) + \boldsymbol{u}^\mathsf{T}(\mathbf{A}\boldsymbol{x} - \mathbf{1}_m) = \\ &= \boldsymbol{x}^\mathsf{T} \left(\mathbf{Q} + \operatorname{diag}(\boldsymbol{\lambda})\right) \boldsymbol{x} - \boldsymbol{\lambda}^\mathsf{T}\boldsymbol{x} + \boldsymbol{u}^\mathsf{T}(\mathbf{A}\boldsymbol{x} - \mathbf{1}_m) = \\ &= \boldsymbol{x}^\mathsf{T} \underbrace{\left(\mathbf{Q} + \operatorname{diag}(\boldsymbol{\lambda})\right)}_{\mathbf{P}(\boldsymbol{\lambda})} \boldsymbol{x} - \underbrace{\left(\boldsymbol{\lambda}^\mathsf{T} - \boldsymbol{u}^\mathsf{T}\mathbf{A}\right)}_{\boldsymbol{q}^\mathsf{T}(\boldsymbol{\lambda}, \boldsymbol{u})} \boldsymbol{x} - \underbrace{\boldsymbol{u}^\mathsf{T}\mathbf{1}_m}_{r(\boldsymbol{u})}. \end{aligned} \tag{3.75}$$

Then, the dual function is calculated minimizing the Lagrangian over primal variables $\boldsymbol{x}$:

$$\begin{aligned} g(\boldsymbol{\lambda}, \boldsymbol{u}) &= \inf_{\boldsymbol{x} \in \mathbb{R}^n} L(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{u}) = \inf_{\boldsymbol{x} \in \mathbb{R}^n} \left[\boldsymbol{x}^\mathsf{T}\mathbf{P}(\boldsymbol{\lambda})\boldsymbol{x} - \boldsymbol{q}^\mathsf{T}(\boldsymbol{\lambda}, \boldsymbol{u})\boldsymbol{x} - r(\boldsymbol{u})\right] = \\ &= \begin{cases} -\frac{1}{4}\boldsymbol{q}^\mathsf{T}(\boldsymbol{\lambda}, \boldsymbol{u})\mathbf{P}^+(\boldsymbol{\lambda})\boldsymbol{q}(\boldsymbol{\lambda}, \boldsymbol{u}) - r(\boldsymbol{u}), & \mathbf{P}(\boldsymbol{\lambda}) \succeq 0 \text{ and } \boldsymbol{q}(\boldsymbol{\lambda}, \boldsymbol{u}) \perp \ker \mathbf{P}(\boldsymbol{\lambda}), \\ -\infty, & \text{otherwise}, \end{cases} \end{aligned} \tag{3.76}$$

where $\mathbf{P}^+(\boldsymbol{\lambda})$ is the Moore–Penrose pseudoinverse of matrix $\mathbf{P}(\boldsymbol{\lambda})$. Hence, optimal value of the dual problem would be the solution of problem (3.77),

$$\begin{aligned} \underset{\boldsymbol{\lambda} \in \mathbb{R}^n,\, \boldsymbol{u} \in \mathbb{R}^m}{\text{maximize}} \quad & -\frac{1}{4}\boldsymbol{q}^\mathsf{T}(\boldsymbol{\lambda}, \boldsymbol{u})\mathbf{P}^+(\boldsymbol{\lambda})\boldsymbol{q}(\boldsymbol{\lambda}, \boldsymbol{u}) - r(\boldsymbol{u}) \\ \text{subject to} \quad & \mathbf{P}(\boldsymbol{\lambda}) \succeq 0, \\ & \boldsymbol{q}(\boldsymbol{\lambda}, \boldsymbol{u}) \perp \ker \mathbf{P}(\boldsymbol{\lambda}). \end{aligned} \tag{3.77}$$

To simplify the constraints in problem (3.77), we need the Schur lemma.

Definition 3.6.1 (Schur complement) *Let* $\mathbf{X}$ *be a symmetric block matrix given by*

$$\mathbf{X} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{B}^\mathsf{T} & \mathbf{C} \end{bmatrix} \in \mathcal{S}^n. \tag{3.78}$$

Then, if $\det A \neq 0$*, the matrix*

$$\mathbf{S} = \mathbf{C} - \mathbf{B}^\mathsf{T}\mathbf{A}^+\mathbf{B} \tag{3.79}$$

is called Schur complement of the block $\mathbf{A}$ *in* $\mathbf{X}$*.*

Lemma 3.6.1 (Schur (Boyd and Vandenberghe, 2004)) *Let* $\mathbf{X}$ *be a symmetric block matrix given by*

$$\mathbf{X} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{B}^\mathsf{T} & \mathbf{C} \end{bmatrix},$$

where $\mathbf{B} = [\boldsymbol{b}_1, \ldots, \boldsymbol{b}_p] \in \mathbb{R}^{m \times p}$*. Let* $\mathbf{S}$ *be the Schur complement of the block* $\mathbf{A}$ *in* $\mathbf{X}$*. Then, the following two statements are true.*

1. $\mathbf{X} \succ 0$ *if and only if* $\mathbf{A} \succ 0$ *and* $\mathbf{S} \succ 0$*.*

2. $\mathbf{X} \succeq 0$ *if and only if* $\mathbf{A} \succeq 0$, $\mathbf{S} \succeq 0$, *and* $\boldsymbol{b}_i \perp \ker \mathbf{A}$, $i = 1, \ldots, p$.

Proof Let us consider the following quadratic optimization problem

$$\boldsymbol{u}^{\mathsf{T}}\mathbf{A}\boldsymbol{u} + 2\boldsymbol{v}^{\mathsf{T}}\mathbf{B}^{\mathsf{T}}\boldsymbol{u} + \boldsymbol{v}^{\mathsf{T}}\mathbf{C}\boldsymbol{v} \to \inf_{\boldsymbol{u}}. \tag{3.80}$$

This problem is bounded if and only if $\mathbf{A} \succeq 0$ and $\mathbf{B}\boldsymbol{v} \perp \ker \mathbf{A}$. The optimal solution then would be $\boldsymbol{u} = -\mathbf{A}^{+}\mathbf{B}\boldsymbol{v}$, and the optimal value:

$$\inf_{\boldsymbol{u}} \begin{bmatrix}\boldsymbol{u}\\ \boldsymbol{v}\end{bmatrix}^{\mathsf{T}} \underbrace{\begin{bmatrix}\mathbf{A} & \mathbf{B}\\ \mathbf{B}^{\mathsf{T}} & \mathbf{C}\end{bmatrix}}_{\mathbf{X}} \begin{bmatrix}\boldsymbol{u}\\ \boldsymbol{v}\end{bmatrix} = \begin{cases} \boldsymbol{v}^{\mathsf{T}}(\underbrace{\mathbf{C} - \mathbf{B}^{\mathsf{T}}\mathbf{A}^{+}\mathbf{B}}_{\mathbf{S}})\boldsymbol{v}, & \mathbf{A} \succeq 0 \text{ and } \mathbf{B}\boldsymbol{v} \perp \ker \mathbf{A}, \\ -\infty, & \text{otherwise.} \end{cases} \tag{3.81}$$

Note that problem (3.80) has the only optimal solution if and only if matrix $\mathbf{A}$ is strictly positive definite: $\mathbf{A} \succ 0$. Now proof becomes straightforward. ∎

Now, according to the Schur lemma, problem (3.77) is equivalent to the following optimization problem:

$$\begin{aligned} &\underset{\boldsymbol{\lambda}\in\mathbb{R}^n,\, \boldsymbol{u}\in\mathbb{R}^m}{\text{maximize}} \quad \gamma - r(\boldsymbol{u}) \\ &\text{subject to} \quad \begin{bmatrix}\mathbf{P}(\boldsymbol{\lambda}) & \frac{1}{2}\boldsymbol{q}(\boldsymbol{\lambda},\boldsymbol{u})\\ \frac{1}{2}\boldsymbol{q}^{\mathsf{T}}(\boldsymbol{\lambda},\boldsymbol{u}) & -\gamma\end{bmatrix} \in \mathcal{S}_{+}^{n+1}. \end{aligned} \tag{3.82}$$

This is a problem of semidefinite programming with $2n$ variables. The optimal value of relaxation (3.82) provides a lower bound on the optimal value of initial BQP problem (3.66). It can be efficiently solved by modern solvers for cone optimization (Boyd and Vandenberghe, 2004). The numerical solution can be computed using the MOSEK (ApS, 2015) solver.

3.6.3 Semidefinite relaxation

In this section, we will consider the relaxation of the BQP problem (3.66) into a semidefinite programming (SDP) problem (Boyd and Vandenberghe, 1997; Ghaoui, 2013; D'Aspremont and Boyd, 2003).

The semidefinite programming problem is aimed at finding the optimal value of a linear objective function over the cone of positive semidefinite matrices $\mathcal{S}_{+}^{n}$ or over its intersection with an affine space, which is a set of solutions of a system of linear equations

$$\{\boldsymbol{x} \in \mathbb{R}^n \mid \mathbf{A}\boldsymbol{x} = \boldsymbol{b}\}. \tag{3.83}$$

SDP belongs to a cone programming subfield and can be solved efficiently by interior point methods (Boyd and Vandenberghe, 2004).

Note that using the cyclic property of trace, one can derive

$$\boldsymbol{x}^{\mathsf{T}}\mathbf{Q}\boldsymbol{x} = \operatorname{Tr}(\boldsymbol{x}^{\mathsf{T}}\mathbf{Q}\boldsymbol{x}) = \operatorname{Tr}(\mathbf{Q}\boldsymbol{x}\boldsymbol{x}^{\mathsf{T}}). \tag{3.84}$$

Also note that boolean constraints $\boldsymbol{x} \in \{0,1\}^n$ can be equivalently replaced with $x_i^2 - x_i = 0$. Therefore, introducing a new variable $\mathbf{X} = \boldsymbol{x}\boldsymbol{x}^{\mathsf{T}} \in \mathcal{S}_{+}^{n}$, the BQP problem (3.66) can be

rewritten as follows:

$$\begin{aligned} \underset{\boldsymbol{x},\,\mathbf{X}}{\text{minimize}} \quad & \operatorname{Tr}(\mathbf{QX}) \\ \text{subject to} \quad & \mathbf{A}\boldsymbol{x} = \mathbf{1}_m, \\ & X_{ii} - x_i = 0, \;\; i = 1, \ldots, n, \\ & \mathbf{X} = \boldsymbol{x}\boldsymbol{x}^\intercal. \end{aligned} \tag{3.85}$$

In SDP relaxation the non-convex rank constraint $X = \boldsymbol{x}\boldsymbol{x}^\intercal$ is replaced with a convex semidefiniteness constraint $\mathbf{X} \succeq \boldsymbol{x}\boldsymbol{x}^\intercal$ but first, let us introduce additional constraints that would tighten the convex search space in semidefinite relaxation. First, note that from the first constraint one can derive the following $\mathbf{A}\boldsymbol{x}\boldsymbol{x}^\intercal = \mathbf{1}_m\boldsymbol{x}^\intercal$, and hence $\mathbf{AX} = \mathbf{1}_m\boldsymbol{x}^\intercal$. Second, we add constraints $\mathbf{X} \in [0, 1]^{n\times n}$. Now, let us relax the non-convex rank constraint $\mathbf{X} = \boldsymbol{x}\boldsymbol{x}^\intercal$ and write the semidefinite relaxation:

$$\begin{aligned} \underset{\boldsymbol{x},\,\mathbf{X}}{\text{minimize}} \quad & \operatorname{Tr}(\mathbf{QX}) \\ \text{subject to} \quad & \mathbf{A}\boldsymbol{x} = \mathbf{1}_m, \\ & \mathbf{AX} = \mathbf{1}_m\boldsymbol{x}^\intercal, \\ & 0 \leqslant X_{ij} \leqslant 1, \;\; i = 1, \ldots, n, \;\; j = 1, \ldots, n, \\ & X_{ii} - x_i = 0, \;\; i = 1, \ldots, n, \\ & \begin{bmatrix} \mathbf{X} & \boldsymbol{x} \\ \boldsymbol{x}^\intercal & 1 \end{bmatrix} \in \mathcal{S}_+^{n+1}, \end{aligned} \tag{3.86}$$

where the semidefiniteness constraint $\mathbf{X} \succeq \boldsymbol{x}\boldsymbol{x}^\intercal$ is written in matrix form using lemma 3.6.1.

Again, as in case of continuous relaxation (see section 3.6.1), the optimal value of relaxation (3.86) provides a lower bound on the optimal value of initial BQP problem (3.66).

Problem (3.86) is the SDP problem, which can be solved efficiently by modern solvers for cone optimization (Boyd and Vandenberghe, 2004). The numerical solution can be computed using the MOSEK (ApS, 2015) solver.

Rounding strategy

To get the upper bound on the optimal value of initial BQP problem (3.66), one needs to come up with a rounding procedure to project the optimal solution of relaxation (3.86) onto the feasible set. The same rounding strategy as described for continuous relaxation in section 3.6.1 can be used (see formula (3.72)). The matrix $\mathbf{X} - \boldsymbol{x}\boldsymbol{x}^\intercal \in \mathcal{S}_+^n$ composed of the optimal solution of the SDP relaxation problem is positive semidefinite, and hence it is a covariance matrix. Therefore, one can also randomly sample neighboring points $\boldsymbol{x}'$ near the found optimal solution of relaxation (3.86):

$$\boldsymbol{x}' \sim \mathcal{N}(\boldsymbol{x}, \mathbf{X} - \boldsymbol{x}\boldsymbol{x}^\intercal), \tag{3.87}$$

and then perform rounding procedure (3.72).

Chapter 4

Results and Discussion

4.1 Protein quality assessment

We measured the performance of the SBROD method on the very recent CASP11 Stage1 and CASP11 Stage2 datasets (Moult *et al.*, 2016). We downloaded CASP11 Stage1 and Stage2 server prediction datasets from the official CASP website at http://predictioncenter.org/download_area/CASP11 and merged them with the published target structures. As a result, we obtained 84 and 83 sets of model structures with the corresponding target structures (84 and 83 domains) for CASP11 Stage1 and CASP11 Stage2 sets, respectively. The actual GDT-TS were computed using the TM-score utility by Zhang and Skolnick (2007).

Being trained on the CASP[5-10] datasets (Moult *et al.*, 2003, 2005, 2007, 2009, 2011, 2014), the SBROD method proved to have the performance close to state-of-the-art QA methods when testing on the CASP11 dataset (Moult *et al.*, 2016).

To compare the performance of the SBROD scoring function and other protein QA methods, for each QA method S, we evaluate the predicted scores $S(P) \in \mathbb{R}$ of each model in each decoy domain $\mathcal{D}$ from the test set, and then estimate the four different performance measures for each decoy domain $\mathcal{D}$ and corresponding native structure P:

- the score loss

$$\text{Loss}(P, \mathcal{D}; S) = \left| \max_{P' \in \mathcal{D}} S^*_{\text{GDT-TS}}(P', P) - S^*_{\text{GDT-TS}}(\arg\max_{P' \in \mathcal{D}} S(P'), P) \right|, \tag{4.1}$$

- Pearson correlation coefficient between predicted scores $S(P')$ of the decoy models from the domain $\mathcal{D}$ and the actual scores $S^*_{\text{GDT-TS}}(P', P)$,

- Spearman's rank correlation coefficient — Pearson correlation coefficient between ranks of scores $\text{rg}S(P')$ and $\text{rg}S^*_{\text{GDT-TS}}(P', P)$, $P' \in \mathcal{D}$,

- Kendall rank correlation coefficient.

Finally, we calculate the average of the estimated performance measures over all the decoy domains in the test set.

4.1.1 Dependence on smoothness

To measure the dependence on the level of smoothing in the feature extraction procedure, we trained distinct SBROD scoring function on the CASP[5-9] datasets using the feature generation procedure described above, without smoothing: $\sigma^a = \sigma^r = \sigma^h = \sigma^s = 0$. Then, we estimated the performance of this trained scoring function on the validation set CASP10 (Stage1 and Stage2 together), with different levels of smoothing in the feature extraction procedure, changing the parameters of truncated Gaussian kernels σ^a, σ^r, σ^h, σ^s. The results are shown in figure 4-1. One can see that the use of the smoothing technique described above

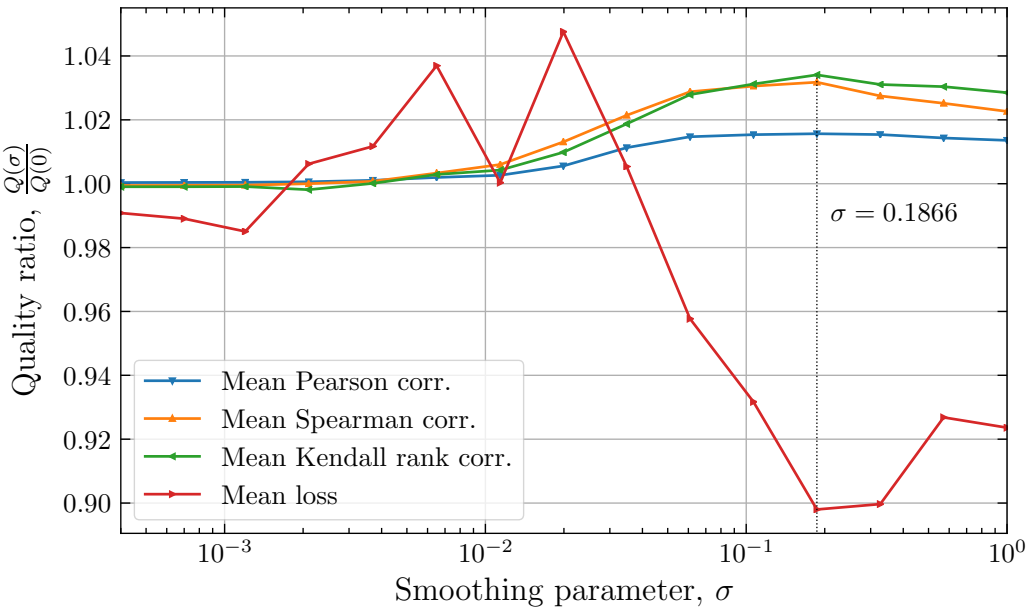


Figure 4-1: The Performance of SBROD on CASP10 dataset (stage1 and stage2 together) for different parameters of smoothing $\sigma^a = \sigma^r = \sigma^h = \sigma^s = \sigma$ being trained on the CASP[5-9] datasets using features without smoothing ($\sigma = 0$)

improves the performance of the scoring potential. The optimal parameter of smoothing according to all the performance measures appeared to be $\sigma = 0.1866$. This value was used in all further experiments on the test sets CASP11 Stage1 and Stage2.

4.1.2 Performance comparison

To compare the performance of the proposed SBROD scoring function and nine state-of-the-art protein QA methods, we used the results obtained by Cao and Cheng (2016). They measured the performance of the QA methods using SCWRL4 (Krivov *et al.*, 2009) (as they stated in a private correspondence) for preprocessing and GDT-TS computed by the LGA utility (Zemla, 2003). As the LGA utility (Zemla, 2003) is not an open-source product but the TM-score utility (Zhang and Skolnick, 2007) is, we have been using the latter. Nevertheless, SBROD is not sensible to side-chains packing and hence to preprocessing by SCWRL4 as well, and the difference between the GDT-TS computed by the TM-score utility (Zhang and Skolnick, 2007) and the LGA utility (Zemla, 2003) is negligible. Therefore, the measurements estimated by Cao and Cheng (2016) are consistent, and they are comparable to the

Table 4.1: Performance on CASP11 Stage1 dataset

QA Method	GDT-TS loss	Pearson	Spearman	Kendall
SBROD (this study)	**0.083**	0.645	0.522	0.388
ProQ2	0.090	0.643	0.506	0.379
ProQ2-refine	0.093	0.653	**0.535**	**0.402**
Qprob	0.097	0.631	0.517	0.389
ModelEvaluator	0.097	0.600	0.470	0.353
VoroMQA	0.108	0.561	0.426	0.318
Wang-SVM	0.109	**0.655**	**0.535**	0.401
Dope	0.111	0.542	0.416	0.316
SBROD (regression)	0.124	0.568	0.441	0.323
RWplus	0.135	0.536	0.433	0.433
RF_CB_SRS_OD	0.162	0.486	0.357	0.357

Table 4.2: Performance on CASP11 Stage2 dataset

QA Method	GDT-TS loss	Pearson	Spearman	Kendall
SBROD (this study)	**0.057**	**0.441**	**0.426**	**0.298**
ProQ2	0.058	0.372	0.366	0.256
Qprob	0.068	0.381	0.387	0.272
VoroMQA	0.069	0.401	0.386	0.269
ProQ2-refine	0.069	0.370	0.375	0.264
ModelEvaluator	0.072	0.324	0.305	0.212
SBROD (regression)	0.074	0.385	0.365	0.255
Dope	0.077	0.304	0.324	0.228
RWplus	0.084	0.295	0.314	0.220
Wang-SVM	0.085	0.362	0.351	0.245
RF_CB_SRS_OD	0.097	0.360	0.350	0.243

performance of the SBROD scoring function measured as described above.

Tables 4.1 and 4.2 list the performance measures computed for the SBROD scoring function (ranking and regression modifications, see section 3.2.2, in combination with the smooth feature extraction procedure, when $\sigma^a = \sigma^r = \sigma^h = \sigma^s = 0.1866$) and for nine other state-of-the-art methods on CASP11 Stage 1 dataset. It can be seen that our method outperforms these state-of-the-art methods on both stages of the CASP11 experiment.

We have also tested the performance of the SBROD scoring and compared it with other QA methods on the MOULDER dataset (Saven, 2013). The results are listed in table 1 Taking into account that other methods were developed to predict the RMSD similarity, we have also tested the performance with the RMSD similarity as the target quality function 2. One can see that on all the investigated datasets (see tables 4.1, 4.2, 1, 2) the proposed QA method SBROD provides the quality comparable to the state-of-the-art methods.

To calculate individual contributions for all the four groups of features, we set to zero all the trained weights w_i (see formula (3.28)) corresponding to feature groups that are not under consideration (see formula (3.26)) and estimated the quality measures on the CASP11 Stage2

Table 4.3: Contribution to the performance for different feature groups on CASP11 Stage2 dataset. Parameters of smoothing $\sigma^a = \sigma^r = \sigma^h = \sigma^s = 0.1866$

Feature groups	GDT-TS loss	Pearson	Spearman	Kendall
All features	0.057	0.441	0.426	0.298
Atomic pairwise	0.069	0.344	0.327	0.224
Residue pairwise	0.078	0.380	0.365	0.253
Solvation shell	0.107	0.267	0.271	0.189
Hydrogen bonds	0.112	0.142	0.126	0.089

test set. Then, we repeated a similar procedure for the other three feature groups. The results are listed in table 4.3. It can be observed from table 4.3 that features corresponding to the atomic backbone representation contribute to the quality of picking the best available model the most. However, features representing pairs of protein residues as rigid frames provide the best correlation performance. Note that knowledge of only the solvation shell without interior protein conformation gives significant prediction quality. Weights corresponding to the hydrogen bonds features provide very poor prediction ability. This might be the case because the information about the hydrogen bonds is already included in other features and can be inferred from the residue pairwise representation. Finally, one can see that usage of all the proposed features provides a significant gain in quality compared to the usage of each feature extraction procedure individually.

4.2 Rotamer prediction

In this computational experiment, we compared different methods for solving the rotamer prediction problem (3.32). More precisely, we compared greedy search algorithm 3.5.1, simulated annealing 3.5.3 with $5 \cdot 10^5$ iterations, continuous relaxation (3.71), Lagrangian relaxation (3.82) (to find a lower bound on the optimal value), semidefinite relaxation (3.86) with sampling of $5 \cdot 10^5$ random approximations near the optimal solution of the relaxation. We also used the greedy search algorithm to improve approximate solutions found by continuous and semidefinite relaxations. To solve numerically relaxation problems, Splitting Conic Solver (SCS) (O'Donoghue *et al.*, 2016) with the CVXPY interface (Diamond and Boyd, 2016) was used with tolerance parameter eps $= 10^{-4}$.

In this section, we show the results for solving the rotamer prediction optimization problem, where a model of the force field proposed by Miao *et al.* (2011) was used.

4.2.1 Data description

To compile the dataset, 40 first protein structures from the test set of SCWRL4 (Krivov *et al.*, 2009) were used. The backbone-dependent rotamer library (Shapovalov and Dunbrack, 2011) was used to define the rotamer space (see definition (1.3)) reducing the continuous set of side chain conformations to a discrete set of rotamers. For each pair of amino acids in a protein, the five most probable according to the rotamer library rotamers were altered, and the total energy was calculated using the forcefield proposed by Miao *et al.* (2011). Then, the pairwise

energy terms were turned into symmetric matrices (3.61) (of sparsity about 99%), and the obtained matrices were truncated to the maximum matrix size of 700×700 to simplify the numerical experiments.

4.2.2 Optimization results

Figure 4-2 shows the overall summary for all the discussed methods. One can see that

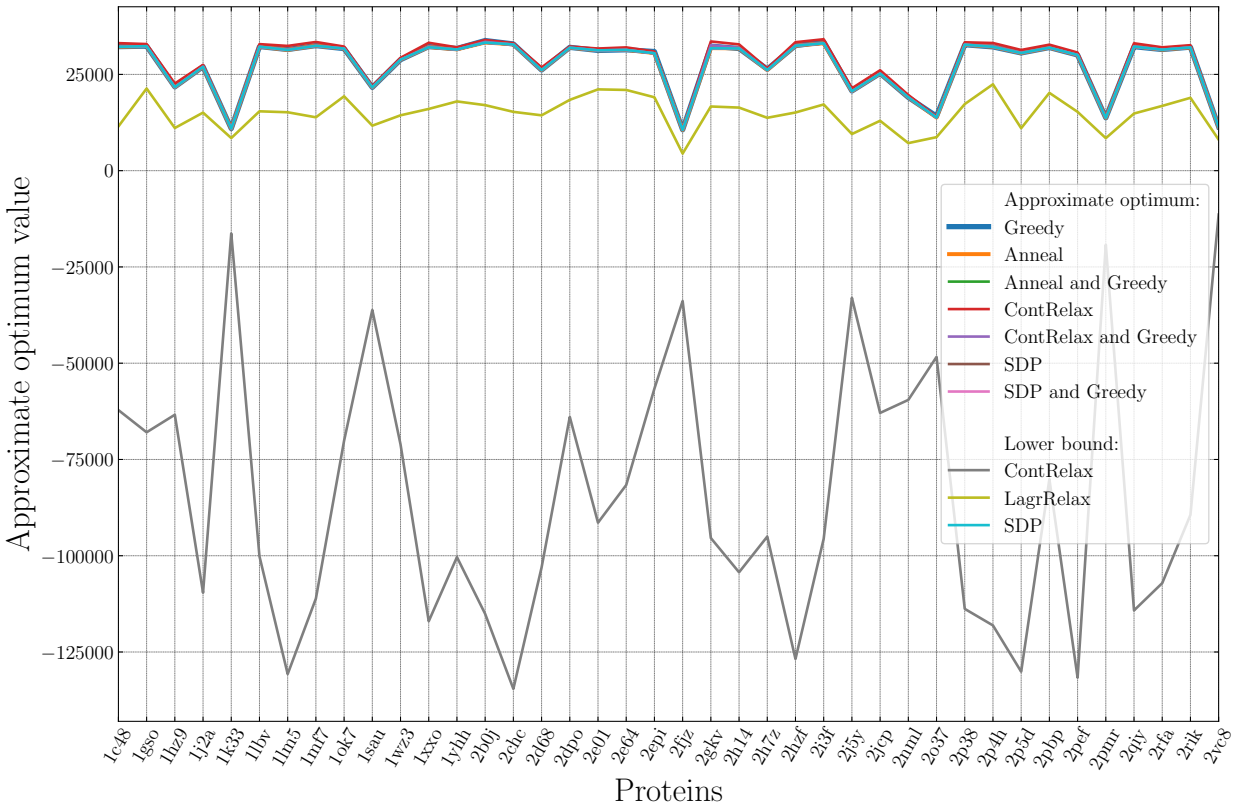


Figure 4-2: Upper and lower bounds on the optimum value of the rotamer prediction optimization problem when solving by different methods

semidefinite relaxation provides the best lower bound on the optimal value of energy, while the lower bound provided by continuous relaxation appears to be extremely poor. The lower bound provided by the Lagrangian relaxation appears to be not that good as one provided by the semidefinite relaxation, which is quite close to the upper bounds found by other algorithms.

Figure 4-3 shows histograms for the normalized (divided by the lower bound provided by the semidefinite relaxation) approximate optimal values of the rotamer prediction problem obtained by algorithms of greedy optimization, simulated annealing, continuous relaxation and semidefinite relaxation followed by rounding procedure (3.72), and continuous and semidefinite relaxations improved by greedy search after the rounding procedure.

Figure 4-4 shows box plots depicting the normalized approximate optimal values obtained by the optimization algorithms considered. The depicted box plots consist of box and

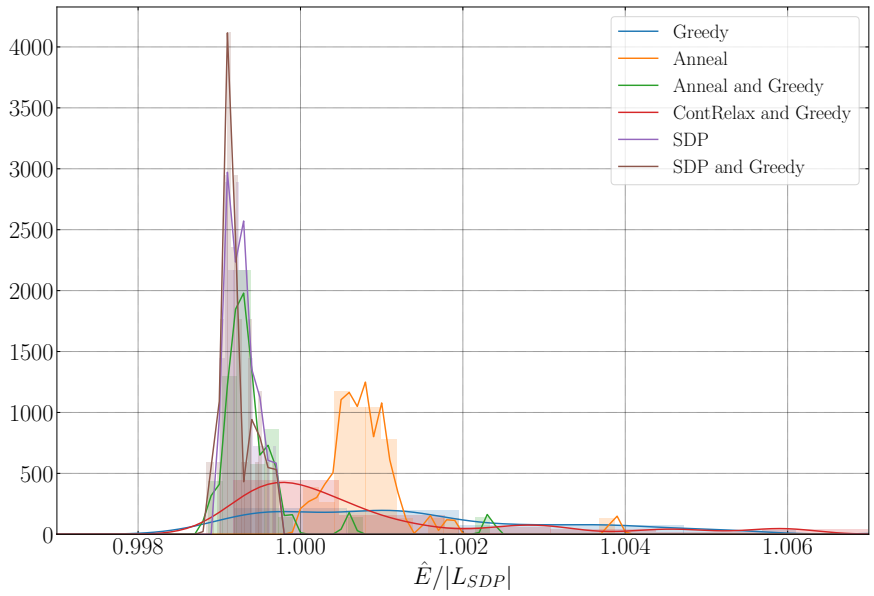


Figure 4-3: Histograms with smoothed curves accumulated for 40 proteins in the dataset for normalized approximate optimal values obtained by different optimization methods

whisker plots. Each box has the bottom and the top at the first and the third quartiles, with the median band inside. Whiskers are plotted to include the data that are still within 1.5IQR from the lower and upper quartile correspondingly, where IQR $= Q_3 - Q_1$ is the interquartile range. Beyond the whiskers, points are considered outliers and marked as individual circles. One can see that rounded approximate solution of the continuous relaxation is the worst solution of the initial problem compared to other methods considered. However, the greedy search algorithm improves it significantly, while usage of just the greedy search algorithm does not lead to the same level of quality. The figure shows that simulated annealing in combination with the greedy search algorithm and semidefinite relaxation followed by sampling and rounding procedures (3.87) and (3.72) achieve the best approximate optimal values, which are very close disregarding few outliers. One can see that greedy search almost does not improve the approximate optimal solutions found by the semidefinite relaxation. This makes SDP the most robust and preferable among the methods considered. Note that normalized quality below $\frac{\hat{E}}{|L_{\text{SDP}}|} = 1$ is actually impossible and such bugs depicted in figures 4-3 and 4-4 are caused by the modest accuracy of the SCS solver (O'Donoghue *et al.*, 2016) that was used to solve numerically the semidefinite relaxation and to find the lower bound on the optimal value.

Figures -3 and -4 in the appendix show individual results for quality and execution time.

4.3 Protein design

The experiment for protein design was conducted on all 352 protein structures from the test set of SCWRL4 (Krivov *et al.*, 2009). We tested the same optimization algorithms using the same computational environment as when conducting the experiment for rotamer

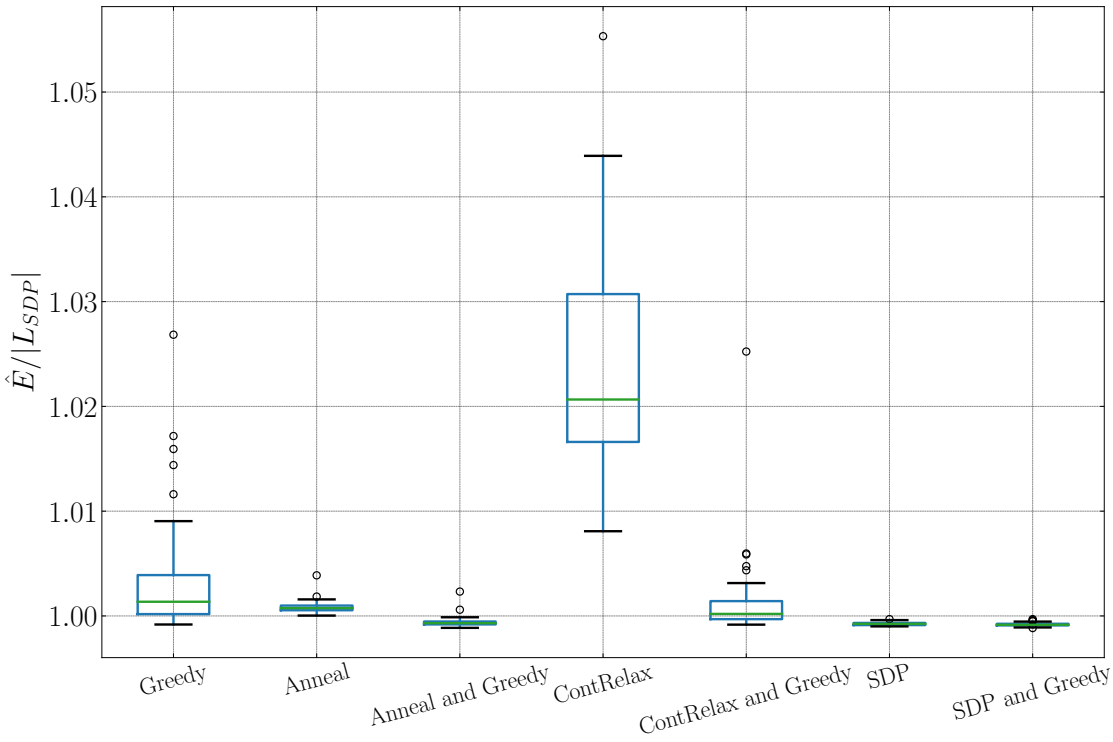


Figure 4-4: Box plots for 40 proteins in the dataset for normalized approximate optimal values obtained by different optimization methods

prediction (see section 4.2). The DFIRE-C_α (Zhang *et al.*, 2004) energy function was used as a coarse-grained potential (3.44). This potential requires only positions of C_α atoms when calculating the energy of atomic pairwise interactions. Due to the high dimension of arising BQP problem (3.66), we restricted the length of predicted protein sequences to make the computational experiments feasible. The matrices $\mathbf{Q}$ were truncated to the maximum size of $20m \times 20m$, where the parameter of the sequence length m was variated from 5 to 30.

4.3.1 Optimization results

We variated the maximal lengths of the predicted protein sequences and measured the quality of approximate optimal values found along with the execution time. This experiment was repeated 352 times for each protein structure in the dataset and then results were averaged for each value of the fixed maximal length. We measured the quality using two metrics. As the first quality criterion, we used the approximate optimum value found by different optimization methods, that is, the upper bound on the actual optimal value for BQP problem (3.66).

Figure 4-5 shows the upper bounds on the optimal energy for different lengths of proteins averaged over 352 runs, for all structures in the dataset. One can see that the continuous relaxation followed by rounding procedure (3.72) provides the worst approximate solution among all the methods in test. Then, greedy search and continuous relaxation followed by the rounding procedure and greedy search show slightly worse results compared to the rest

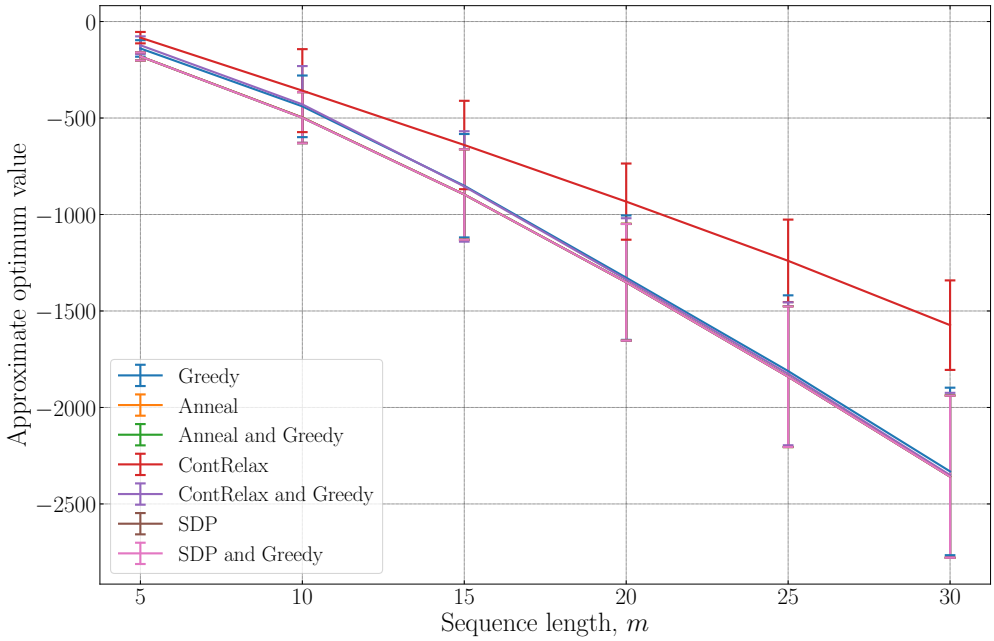


Figure 4-5: Upper bounds on the optimal value averaged over 352 protein structures in the dataset depending on the sequence length

methods. One can conclude from figure 4-5 that the obtained protein design problems are easy to solve since almost all the tested methods provide approximately the same upper bounds. This outcome might have been caused by the specificity of the forcefield used (DFIRE-C_α by Zhang *et al.* (2004)) as it encourages appearing of the Cysteine bridges making the optimal protein sequence to consist of only Cys amino acids.

Figure 4-6 shows average number of common amino acids for the corresponding positions in predicted and the native protein sequences: $\sum_{i=1}^{m} \mathbb{1}[a_i = \hat{a}_i]$, where $\hat{\boldsymbol{a}} \in \mathcal{A}^m$ is the predicted sequence, $\boldsymbol{a} \in \mathcal{A}^m$ is the native sequence. The figure confirms unworthiness of scoring potential DFIRE-C_α for the protein design problem as the precision for the predicted sequences is about 0.03, which is worse than for random guessing. Also, note that the best continuous relaxation shows the best precision whereas it provides the worst approximate solution when comparing the approximate optimum values.

The discussed results appeal to further development of scoring potentials tuned specifically for the protein design problem. As long as the proposed regression modification of the SBROD scoring function was trained to predict GDT-TS of protein backbone structures, it potentially could be applied to the protein design problem instead of basic coarse-grained energy potentials such as DFIRE-C_α. However, such experiments go out of the scope of the current thesis project.

Figures -5 and -6 in the appendix show the quality and execution time for protein design on each individual protein structure truncated to the maximum length $m = 30$. For the sake of compactness, only results for first 80 protein structures in the dataset are presented.

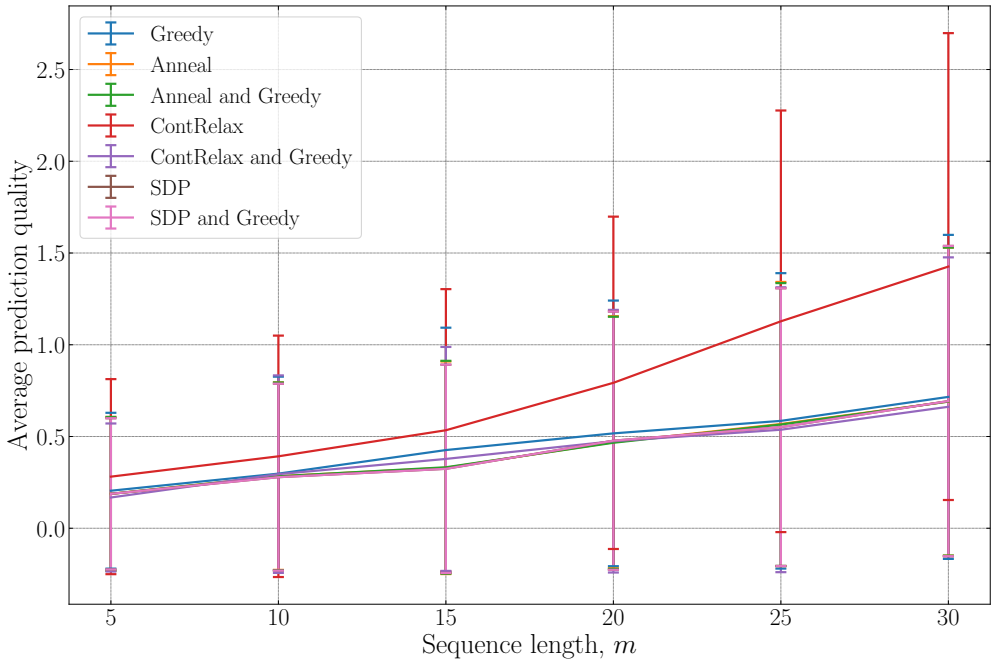


Figure 4-6: Average ratio of correctly predicted amino acids for different protein lengths

4.3.2 Energy correction

The computational experiment conducted by Riazanov *et al.* (2016) revealed that for almost all input backbones, optimal solutions of the protein design problem with C_α-C_α potential (Rajgaria *et al.*, 2006) are sequences that consist entirely of amino acids Cys and Tyr. This finding is rather expected yet undesirable as we discussed in section 2.3. To overcome this issue, we introduced energy correction terms (3.52) that make amino acids of different types more evenly distributed along the sequences predicted. To determine introduced energy corrections (3.52), the parameters p_a and σ_a for amino acids $a \in \mathcal{A}$ were defined for simplicity as follows: $p_a = \frac{1}{|\mathcal{A}|}$, $\sigma_a = 1 \quad \forall a \in \mathcal{A}$.

Figures -7 and -8 in appendix show frequencies for occurrences of amino acids Cys and Glu in predicted sequences depending on the temperature factor $\beta = \frac{1}{T}$ (see formula (3.47)). It can be seen from the figures that when temperature $T = \frac{1}{\beta}$ is low (β close to 1) the predicted protein sequences consist of mostly the Cys amino acids (the distribution is concentrated near 0.8-1.0). On the contrary, there are almost no Glu amino acids in sequences predicted when the energy corrections are small (low temperature). One can conclude from figures -7 and -8 that the frequencies of occurrence for amino acids becomes more evenly distributed and their mean values are shifted close to $p_a = \frac{1}{|\mathcal{A}|} = \frac{1}{20}$ when increasing the temperature T. Figure -9 in the appendix shows mutation of frequencies for each amino acid type depending on the temperature factor $\beta = \frac{1}{T}$.

Chapter 5

Conclusions

In this thesis project, we investigated the main and the most challenging problems of computational biology: protein folding, side chain prediction, and protein design.

The physical nature of protein design makes this problem hard to devise computational techniques for solving it without involving biochemical experiments since the predicted sequences are usually new and their tertiary structures are not yet discovered. However, the performance of computational techniques for protein design can be estimated approximately by testing foldings of the sequences found. This makes the protein folding problem essential for the development of the computational tools for protein design.

We proposed a novel method for protein quality assessment that is a crucial part of methods for protein folding. The proposed method is computationally efficient, and it uses only structural features, what makes its corresponding scoring function smooth. Performed computational experiment and comparison with other state-of-the-art methods for protein quality assessment proved the proposed QA method to achieve the state-of-the-art quality.

The regression modification of the proposed quality assessment method can be potentially applied for solving protein design problem, and this can be a direction for further research.

The proposed QA methods provide pairwise decomposable scoring functions, which enable the reduction of the initial problem to the problem of boolean quadratic programming (BQP). Note that for any energy function the arising problems of discrete optimization can be represented as problems of boolean polynomial programming and then be reduced to the boolean quadratic programming. However, this reduction makes the eventual BQP problem of huge dimension and impossible to solve using current computational facilities. This fact makes the proposed scoring function that includes only terms corresponding to pairwise interactions between amino acids especially important for considered problems of computational biology.

We analyzed the techniques for relaxing the arising NP-hard BQP problems into problems of convex optimization and compared them to other methods for combinatorial optimization: greedy search and simulated annealing algorithms. We introduced additional linear constraint to the semidefinite relaxation that tightened the search space and sustained that performing the computational experiment. Lower bounds provided by SDP relaxation can be used by other algorithms for discrete optimization but the computational cost remains dramatic. It appeals for further development and advances in computational techniques for convex optimization.

Besides that, we proposed pairwise energy correction terms that regulate the contribution of an energy function used to diversify occurrence of different amino acids in the sequences predicted minimizing the total energy of the structure for protein design problem. The performed computational experiment confirmed theoretical assumptions.

Bibliography

Ailon, N. and Chazelle, B. (2009). The Fast Johnson–Lindenstrauss Transform and Approximate Nearest Neighbors. *SIAM Journal on Computing*, **39**(1), 302–322.

ApS, M. (2015). *The MOSEK optimization toolbox for MATLAB manual. Version 7.1 (Revision 28).*

Boyd, S. and Vandenberghe, L. (1997). Semidefinite Programming Relaxations of Non-Convex Problems in Control and Combinatorial Optimization.

Boyd, S. and Vandenberghe, L. (2004). *Convex optimization*. Cambridge university press.

Cao, R. and Cheng, J. (2016). Protein single-model quality assessment by feature-based probability density functions. *Scientific Reports*, **6**, 23990.

Cecchini, M., Krivov, S. V., Spichty, M., and Karplus, M. (2009). Calculation of Free-Energy Differences by Confinement Simulations. Application to Peptide Conformers. *The Journal of Physical Chemistry B*, **113**(29), 9728–9740.

D'Aspremont, A. and Boyd, S. (2003). Relaxations and randomized methods for nonconvex QCQPs.

der Laan, M. J., Polley, E. C., and Hubbard, A. E. (2007). Super learner. *Statistical applications in genetics and molecular biology*, **6**(1).

Diamond, S. and Boyd, S. (2016). {CVXPY}: A {P}ython-Embedded Modeling Language for Convex Optimization. *Journal of Machine Learning Research*, **17**(83), 1–5.

Duan, Y. and Kollman, P. A. (1998). Pathways to a Protein Folding Intermediate Observed in a 1-Microsecond Simulation in Aqueous Solution. *Science*, **282**(5389), 740–744.

Faraggi, E. and Kloczkowski, A. (2014). A global machine learning based scoring function for protein structure prediction. *Proteins: Structure, Function, and Bioinformatics*, **82**(5), 752–759.

Fung, H. K., Welsh, W. J., and Floudas, C. A. (2008). Computational de novo peptide and protein design: Rigid templates versus flexible templates. *Industrial and Engineering Chemistry Research*, **47**(4), 993–1001.

Ghaoui, L. E. (2013). Convex Optimization Lecture Notes for EE 227BT Draft, Fall 2013.

Gill, P. E. and Wong, E. (2015). Methods for convex and general quadratic programming. *Mathematical Programming Computation*, **7**(1), 71–112.

Gurobi Optimization, I. (2016). Gurobi Optimizer Reference Manual.

Hoffmann, A. and Grudinin, S. (2017). NOLB: Nonlinear Rigid Block Normal-Mode Analysis Method. *Journal of Chemical Theory and Computation*, **13**(5), 2123–2134.

Huang, P.-S., Boyken, S. E., and Baker, D. (2016). The coming of age of de novo protein design. *Nature*, **537**(7620), 320–327.

Hubbard, R. E. and Kamran Haider, M. (2001). Hydrogen Bonds in Proteins: Role and Strength. In *eLS*. John Wiley & Sons, Ltd.

Jing, X., Wang, K., Lu, R., and Dong, Q. (2016). Sorting protein decoys by machine-learning-to-rank. *Scientific Reports*, **6**, 31571.

Joachims, T. (2002). Optimizing Search Engines Using Clickthrough Data. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '02, pages 133–142, New York, NY, USA. ACM.

Jones, E., Oliphant, T., Peterson, P., and Others (2001). {SciPy}: Open source scientific tools for {Python}.

Khachaturyan, A., Semenovsovskaya, S., and Vainshtein, B. (1981). The thermodynamic approach to the structure analysis of crystals. *Acta Crystallographica Section A*, **37**(5), 742–754.

Khoury, G. A., Smadbeck, J., Kieslich, C. A., and Floudas, C. A. (2014). Protein folding and de novo protein design for biotechnological applications. *Trends in Biotechnology*, **32**(2), 99–109.

Kmiecik, S., Gront, D., Kolinski, M., Wieteska, L., Dawid, A. E., and Kolinski, A. (2016). Coarse-Grained Protein Models and Their Applications. *Chemical Reviews*, **116**(14), 7898–7936.

Krivov, G. G., Shapovalov, M. V., and Dunbrack, R. L. (2009). Improved prediction of protein side-chain conformations with SCWRL4. *Proteins*, **77**(4), 778–795.

Liu, T., Wang, Y., Eickholt, J., and Wang, Z. (2016). Benchmarking Deep Networks for Predicting Residue-Specific Quality of Individual Protein Models in CASP11. *Scientific Reports*, **6**, 19301.

Liu, Y., Zeng, J., and Gong, H. (2014). Improving the orientation-dependent statistical potential using a reference state. *Proteins*, **82**(10), 2383–2393.

Miao, Z., Cao, Y., and Jiang, T. (2011). RASP: rapid modeling of protein side chain conformations. *Bioinformatics*, **27**(22), 3117–3122.

Moult, J., Fidelis, K., Zemla, A., and Hubbard, T. (2003). Critical assessment of methods of protein structure prediction (CASP)-round V. *Proteins: Structure, Function, and Bioinformatics*, **53**(S6), 334–339.

Moult, J., Fidelis, K., Rost, B., Hubbard, T., and Tramontano, A. (2005). Critical assessment of methods of protein structure prediction (CASP)—Round 6. *Proteins: Structure, Function, and Bioinformatics*, **61**(S7), 3–7.

Moult, J., Fidelis, K., Kryshtafovych, A., Rost, B., Hubbard, T., and Tramontano, A. (2007). Critical assessment of methods of protein structure prediction—Round VII. *Proteins: Structure, Function, and Bioinformatics*, **69**(S8), 3–9.

Moult, J., Fidelis, K., Kryshtafovych, A., Rost, B., and Tramontano, A. (2009). Critical assessment of methods of protein structure prediction—Round VIII. *Proteins: Structure, Function, and Bioinformatics*, **77**(S9), 1–4.

Moult, J., Fidelis, K., Kryshtafovych, A., and Tramontano, A. (2011). Critical assessment of methods of protein structure prediction (CASP)—round IX. *Proteins: Structure, Function, and Bioinformatics*, **79**(S10), 1–5.

Moult, J., Fidelis, K., Kryshtafovych, A., Schwede, T., and Tramontano, A. (2014). Critical assessment of methods of protein structure prediction (CASP) — round x. *Proteins: Structure, Function, and Bioinformatics*, **82**, 1–6.

Moult, J., Fidelis, K., Kryshtafovych, A., Schwede, T., and Tramontano, A. (2016). Critical assessment of methods of protein structure prediction: Progress and new directions in round XI. *Proteins: Structure, Function, and Bioinformatics*, **84**, 4–14.

Neal, R. M. (1996). *Bayesian Learning for Neural Networks*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

O'Donoghue, B., Chu, E., Parikh, N., and Boyd, S. (2016). Conic Optimization via Operator Splitting and Homogeneous Self-Dual Embedding. *Journal of Optimization Theory and Applications*, **169**(3), 1042–1068.

Rajgaria, R., McAllister, S. R., and Floudas, C. A. (2006). A novel high resolution Calpha–Calpha distance dependent force field based on a high quality decoy set. *Proteins: Structure, Function and Genetics*, **65**(3), 726–741.

Randall, A. and Baldi, P. (2008). SELECTpro: effective protein model selection using a structure-based energy function resistant to BLUNDERs. *BMC Structural Biology*, **8**(1), 52.

Ray, A., Lindahl, E., and Wallner, B. (2012). Improved model quality assessment using ProQ2. *BMC Bioinformatics*, **13**(1), 224.

Riazanov, A., Karasikov, M., and Grudinin, S. (2016). Inverse protein folding problem via quadratic programming. In *Information Technology and Systems 2016*, pages 561–568, Repino, St. Petersburg, Russia. MIPT.

Rykunov, D. and Fiser, A. (2010). New statistical potential for quality assessment of protein models and a survey of energy functions. *BMC Bioinformatics*, **11**(1), 128.

Samish, I., Macdermaid, C., Perez-Aguilar, J., and Saven, J. (2011). Theoretical and computational protein design. *Annual Review of Physical Chemistry*, **62**(1), 129–149.

Saven, J. G. (2013). De Novo Computational Protein Design. In *De novo Molecular Design*, pages 467–493. Wiley-VCH Verlag GmbH & Co. KGaA.

Semenovskaya, S. V., Khachaturyan, K. A., and Khachaturyan, A. G. (1985). Statistical mechanics approach to the structure determination of a crystal. *Acta Crystallographica Section A*, **41**(3), 268–273.

Shapovalov, M. V. and Dunbrack, R. L. (2011). A Smoothed Backbone-Dependent Rotamer Library for Proteins Derived from Adaptive Kernel Density Estimates and Regressions. *Structure*, **19**(6), 844–858.

Shen, M.-y. and Sali, A. (2006). Statistical potential for assessment and prediction of protein structures. *Protein Science*, **15**(11), 2507–2524.

Tibshirani, R. (1996). Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, **58**(1), 267–288.

Tyka, M. D., Clarke, A. R., and Sessions, R. B. (2006). An Efficient, Path-Independent Method for Free-Energy Calculations. *The Journal of Physical Chemistry B*, **110**(34), 17212–17220.

Uziela, K. and Wallner, B. (2016). ProQ2: estimation of model accuracy implemented in Rosetta. *Bioinformatics*, **32**(9), 1411–1413.

Wagner, I. and Musso, H. (1983). New Naturally Occurring Amino Acids. *Angewandte Chemie International Edition in English*, **22**(11), 816–828.

Wang, Z., Tegge, A. N., and Cheng, J. (2009). Evaluating the absolute quality of a single protein model using structural features and support vector machines. *Proteins*, **75**(3), 638–647.

Webb, B. and Sali, A. (2014). Comparative Protein Structure Modeling Using MODELLER. *Current protocols in bioinformatics*, **47**, 5.6.1–32.

Zemla, A. (2003). LGA: a method for finding 3D similarities in protein structures. *Nucleic Acids Research*, **31**(13), 3370–3374.

Zhang, C., Liu, S., Zhou, H., and Zhou, Y. (2004). An accurate, residue-level, pair potential of mean force for folding and binding based on the distance-scaled, ideal-gas reference state. *Protein Science : A Publication of the Protein Society*, **13**(2), 400–411.

Zhang, J. and Zhang, Y. (2010). A Novel Side-Chain Orientation Dependent Potential Derived from Random-Walk Reference State for Protein Fold Selection and Structure Prediction. *PLOS ONE*, **5**(10), e15386.

Zhang, Y. and Skolnick, J. (2007). Scoring function for automated assessment of protein structure template quality. *Proteins: Structure, Function, and Bioinformatics*, **68**(4), 1020.

Zhou, H. and Skolnick, J. (2011). GOAP: A Generalized Orientation-Dependent, All-Atom Statistical Potential for Protein Structure Prediction. *Biophysical Journal*, **101**(8), 2043–2052.

Zhou, H. and Zhou, Y. (2002). Distance-scaled, finite ideal-gas reference state improves structure-derived potentials of mean force for structure selection and stability prediction. *Protein Science : A Publication of the Protein Society*, **11**(11), 2714–2726.

Zou, H. and Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society. Series B: Statistical Methodology*, **67**(2), 301–320.

Appendix

Proof of Lemma 3.4.1

$$-\log p(\boldsymbol{a}|\boldsymbol{b}^0) = -\log p(\boldsymbol{b}^0|\boldsymbol{a}) - \log p(\boldsymbol{a}) + \text{const} = \tag{1}$$

$$\overset{\substack{(3.47)\\(3.50)}}{=} \frac{E(\boldsymbol{a},\boldsymbol{b}^0)}{T} - \sum_{a\in\mathcal{A}} \log \mathcal{N}\left(m_a|\, mp_a, m\sigma_a^2\right) + \text{const} = \tag{2}$$

$$= \frac{E(\boldsymbol{a},\boldsymbol{b}^0)}{T} + \sum_{a\in\mathcal{A}} \frac{(m_a - mp_a)^2}{2m\sigma_a^2} + \text{const}\,. \tag{3}$$

Let us consider the sum over all amino acid types separately.

$$\sum_{a\in\mathcal{A}} \frac{(m_a - mp_a)^2}{2m\sigma_a^2} = \sum_{a\in\mathcal{A}} \frac{1}{2m\sigma_a^2} \left(\sum_{i=1}^{m} \left(\mathbb{1}[a_i = a] - p_a\right)\right)^2 = \tag{4}$$

$$= \sum_{a\in\mathcal{A}} \frac{1}{2m\sigma_a^2} \sum_{k=1}^{m}\sum_{l=1}^{m} \left(\mathbb{1}[a_k = a] - p_a\right)\left(\mathbb{1}[a_l = a] - p_a\right) = \tag{5}$$

$$= \sum_{k=1}^{m}\sum_{l=1}^{m} \underbrace{\sum_{a\in\mathcal{A}} \frac{1}{2m\sigma_a^2} \left(\mathbb{1}[a_k = a] - p_a\right)\left(\mathbb{1}[a_l = a] - p_a\right)}_{M(a_k,a_l)}\,. \tag{6}$$

Then,

$$M(a_k,a_l) = \sum_{a\in\mathcal{A}} \frac{1}{2m\sigma_a^2} \left(\mathbb{1}[a_k = a] - p_a\right)\left(\mathbb{1}[a_l = a] - p_a\right) = \tag{7}$$

$$= \sum_{a\in\mathcal{A}} \frac{1}{2m\sigma_a^2} \left(\mathbb{1}[a_k = a]\mathbb{1}[a_l = a] + p_a^2\right) - \frac{p_{a_k}}{2m\sigma_{a_k}^2} - \frac{p_{a_l}}{2m\sigma_{a_l}^2} = \tag{8}$$

$$= \sum_{a\in\mathcal{A}} \frac{p_a^2}{2m\sigma_a^2} + \sum_{a\in\mathcal{A}} \frac{1}{2m\sigma_a^2}\mathbb{1}[a_k = a]\mathbb{1}[a_l = a] - \frac{p_{a_k}}{2m\sigma_{a_k}^2} - \frac{p_{a_l}}{2m\sigma_{a_l}^2} = \tag{9}$$

$$= \frac{\mathbb{1}[a_k = a_l]}{2m\sigma_{a_k}^2} - \frac{p_{a_k}}{2m\sigma_{a_k}^2} - \frac{p_{a_l}}{2m\sigma_{a_l}^2} + \text{const}\,. \tag{10}$$

Therefore, energy corrections $E'_{kl}(a_k,a_l)$ defined by (3.52) make problem (3.53) equivalent

to problem (3.46). This makes clear from continuing derivation (3):

$$-\log p(\boldsymbol{a}|\boldsymbol{b}^0) \overset{(3)}{\overset{(6)}{\overset{(10)}{=}}} \frac{E(\boldsymbol{a},\boldsymbol{b}^0)}{T} + \sum_{k=1}^{m}\sum_{l=1}^{m}\left[\frac{\mathbb{1}[a_k = a_l]}{2m\sigma_{a_k}^2} - \frac{p_{a_k}}{2m\sigma_{a_k}^2} - \frac{p_{a_l}}{2m\sigma_{a_l}^2}\right] + \text{const} = \tag{11}$$

$$\overset{(3.52)}{=} \frac{E(\boldsymbol{a},\boldsymbol{b}^0)}{T} + \sum_{k=1}^{m}\sum_{l=1}^{m}\frac{E'_{kl}(a_k,a_l)}{T} + \text{const} \propto \tag{12}$$

$$\propto E(\boldsymbol{a},\boldsymbol{b}^0) + \sum_{k=1}^{m}\sum_{l=1}^{m}E'_{kl}(a_k,a_l) + \text{const} = \tag{13}$$

$$\overset{(3.51)}{=} \sum_{k=1}^{m}\sum_{l=1}^{m}\left[E_{kl}(a_k,a_l) + E'_{kl}(a_k,a_l)\right] + \text{const}\,. \tag{14}$$

∎

Table 1: Performance on the MOULDER dataset. The SBROD scoring function is trained on the CASP[5-11] datasets. The metrics are calculated with the GDT-TS as a target scoring function. Results are sorted by Pearson correlation

QA Method	GDT-TS loss	Pearson	Spearman	Kendall
SVM_SCORE	0.023	0.938	0.935	0.778
SBROD (this study)	0.041	0.927	0.920	0.755
Native_Overlap_3.5	0.008	0.922	0.922	0.781
PSIPRED_WEIGHT	0.040	0.919	0.910	0.738
PSIPRED_PERCENT	0.050	0.910	0.900	0.723
PROSA_COMB	0.040	0.889	0.900	0.723
MODCHECK	0.052	0.888	0.887	0.708
DOPE_AA	0.034	0.887	0.909	0.743
MP_COMBI	0.050	0.879	0.889	0.708
PROSA_SURF	0.065	0.873	0.873	0.692
ROSETTA	0.042	0.868	0.878	0.694
MP_SURF	0.071	0.855	0.864	0.677
RWplus	0.055	0.847	0.871	0.696
DFIRE	0.040	0.846	0.883	0.709
Xd	0.077	0.838	0.835	0.656
GA341	0.076	0.838	0.913	0.741
SOLVX	0.100	0.815	0.810	0.614
PROSA_PAIR	0.070	0.799	0.818	0.626
DOPE_BB	0.053	0.798	0.826	0.634
MP_PAIR	0.080	0.793	0.812	0.615
FRST	0.100	0.761	0.773	0.588
Anolea_PUC	0.123	0.682	0.667	0.484
Anolea_Z	0.097	0.592	0.645	0.457
GB	0.056	0.584	0.794	0.609
EEF1	0.054	0.535	0.791	0.605
SIFT	0.220	0.226	0.275	0.187
Anolea_PE	0.453	0.093	0.127	0.091

Table 2: Performance on the MOULDER dataset. The SBROD scoring function is trained on the CASP[5-11] datasets. The metrics are calculated with the RMSD as a target scoring function. Results are sorted by Pearson correlation

QA Method	GDT-TS loss	Pearson	Spearman	Kendall
Native_Overlap_3.5	0.099	0.901	0.905	0.751
SVM_SCORE	0.601	0.874	0.881	0.696
DOPE_AA	0.675	0.870	0.872	0.690
SBROD (this study)	0.890	0.866	0.868	0.682
PSIPRED_WEIGHT	0.792	0.858	0.865	0.672
PSIPRED_PERCENT	0.919	0.847	0.855	0.661
DFIRE	0.692	0.847	0.859	0.677
ROSETTA	0.868	0.846	0.843	0.652
RWplus	1.272	0.846	0.852	0.670
PROSA_COMB	0.844	0.835	0.839	0.648
MP_COMBI	1.231	0.816	0.823	0.627
MODCHECK	1.045	0.806	0.827	0.634
PROSA_SURF	1.045	0.803	0.819	0.629
MP_SURF	1.811	0.783	0.809	0.615
DOPE_BB	1.119	0.783	0.787	0.589
GA341	1.604	0.768	0.849	0.654
SOLVX	2.431	0.753	0.762	0.566
Xd	1.741	0.753	0.770	0.585
PROSA_PAIR	1.961	0.748	0.752	0.560
MP_PAIR	1.770	0.730	0.739	0.544
FRST	2.081	0.693	0.699	0.515
Anolea_PUC	2.368	0.673	0.645	0.462
GB	1.562	0.620	0.751	0.562
Anolea_Z	1.903	0.588	0.615	0.433
EEF1	1.386	0.577	0.758	0.567
SIFT	6.052	0.203	0.257	0.175
Anolea_PE	10.748	0.136	0.159	0.115

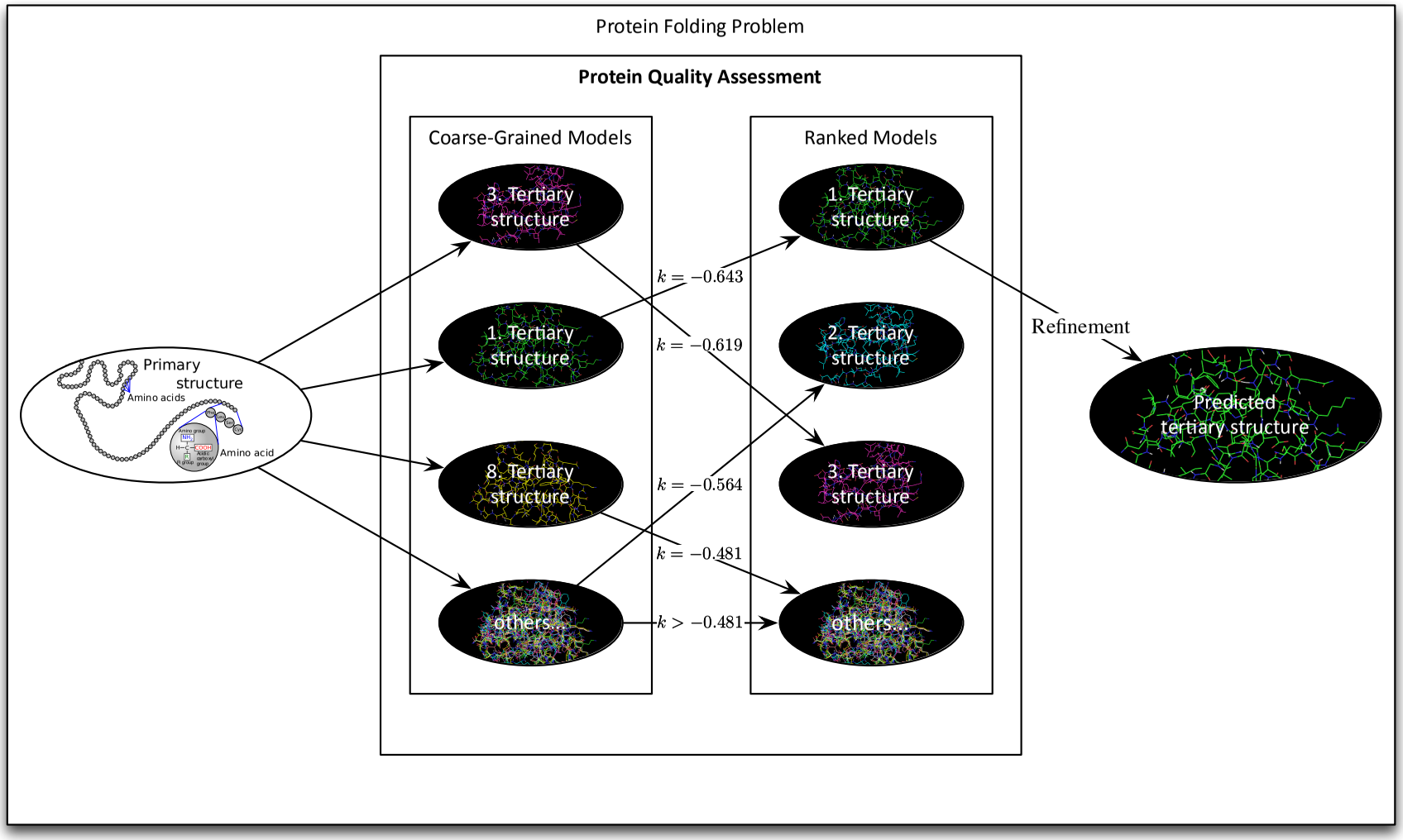


Figure -1: Workflow for the protein folding

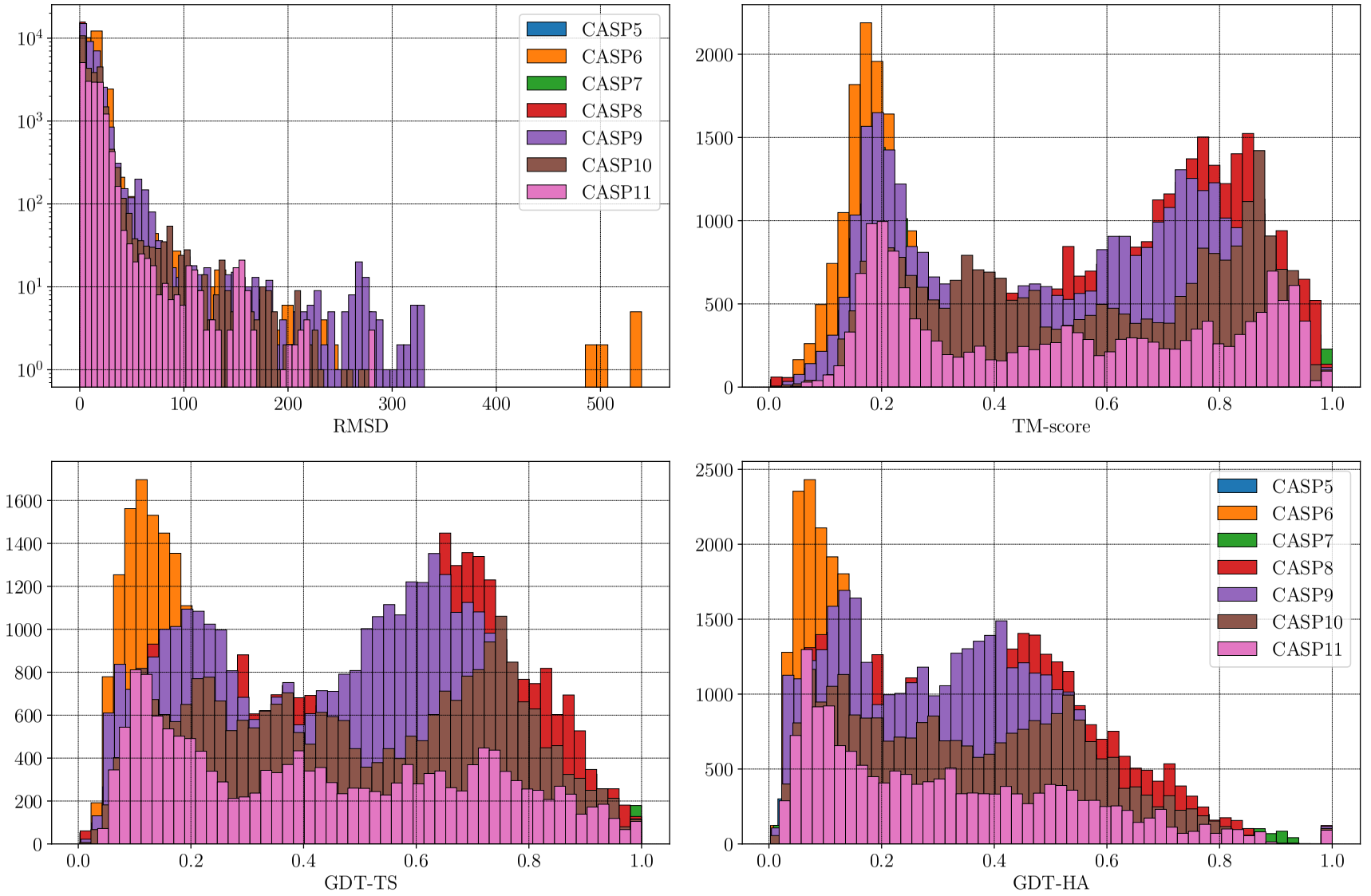


Figure -2: The distribution for protein distance measures

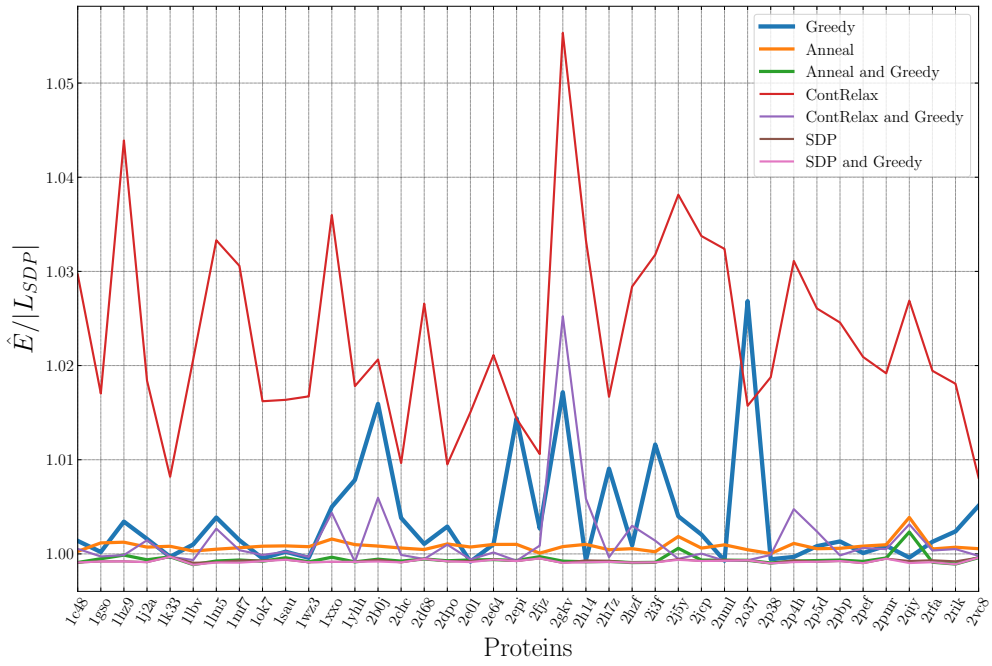


Figure -3: The quality of solving the rotamer prediction problem by different algorithms on truncated matrices of size 700×700. The quality is measured as approximate optimal value divided by the lower bound on the approximate value found by the SDP relaxation

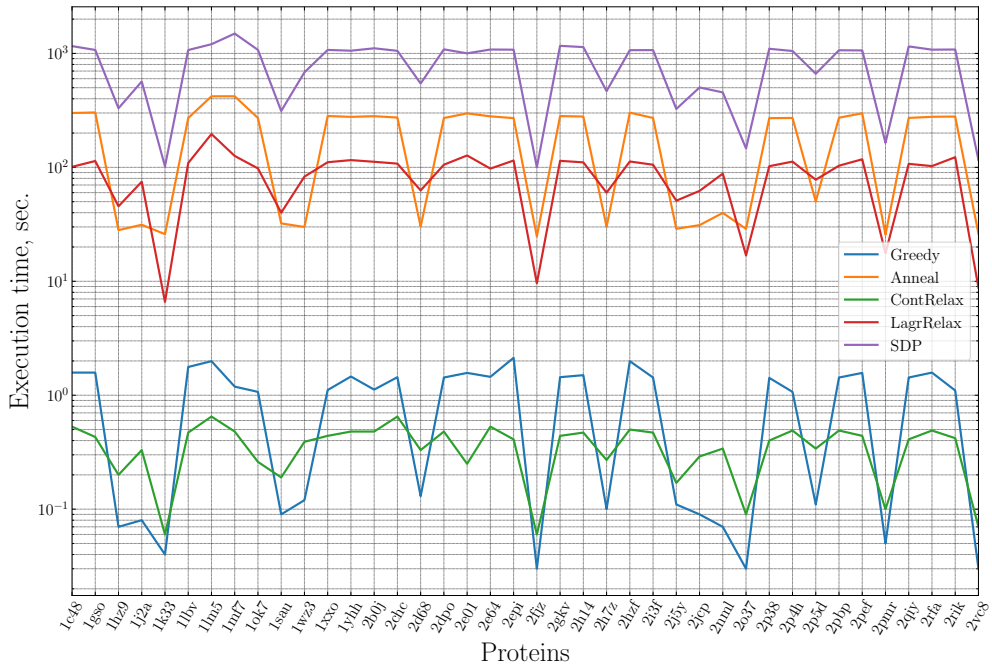


Figure -4: Computational time for solving the rotamer prediction problem by different optimization algorithms depending on truncated matrices of size 700×700

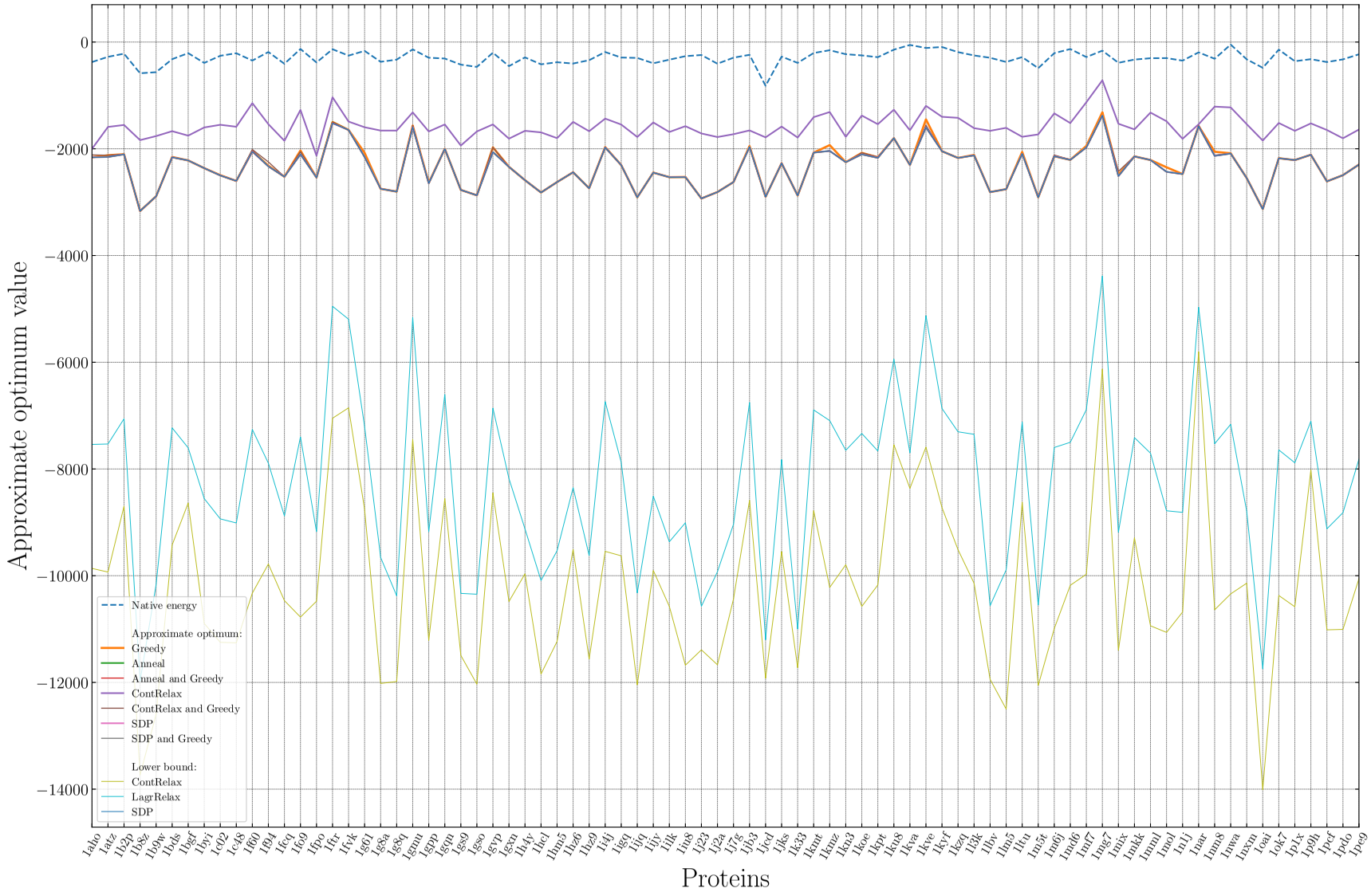


Figure -5: Approximate optimal values of the protein design problem for different algorithms on truncated protein structures of length $m = 30$

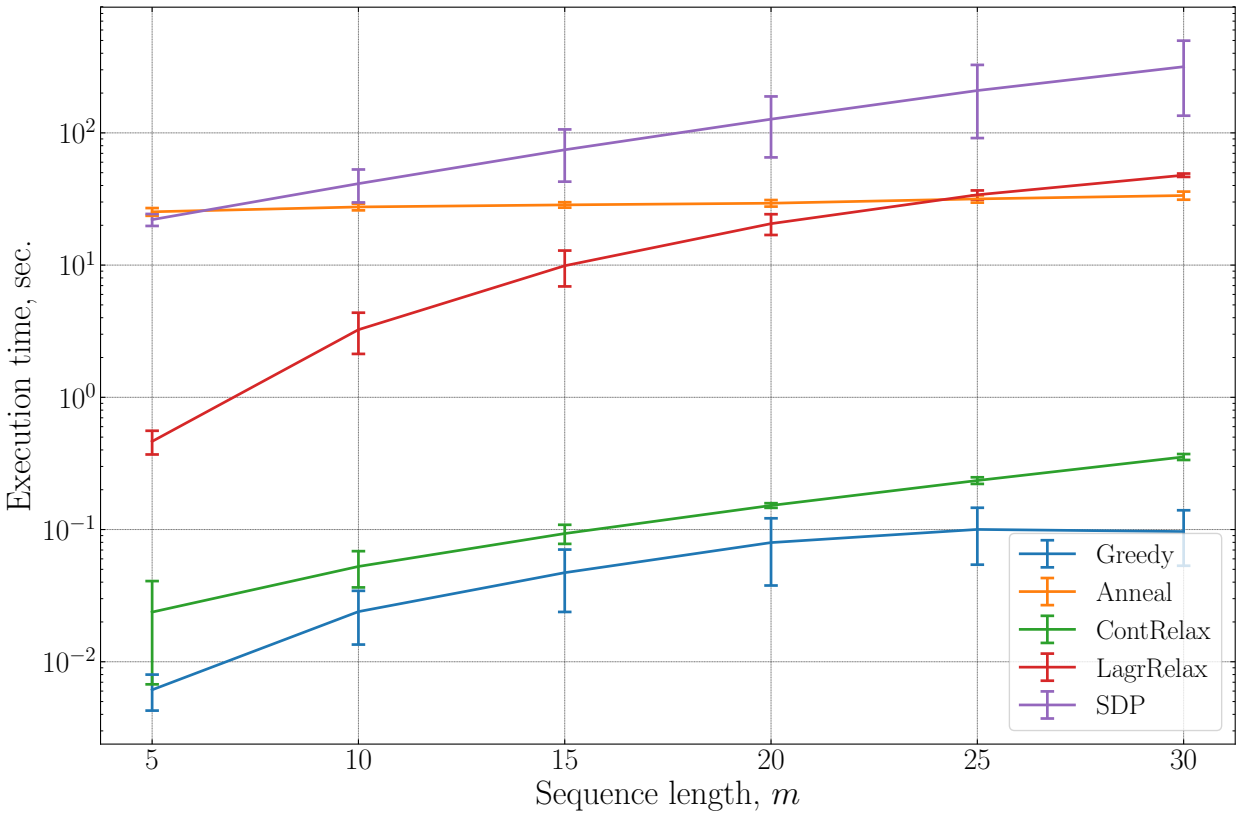


Figure -6: Computational time for solving the protein design problem by different optimization algorithms depending on the length of truncated protein structures. The results are averaged over 352 runs on all the structures in the dataset

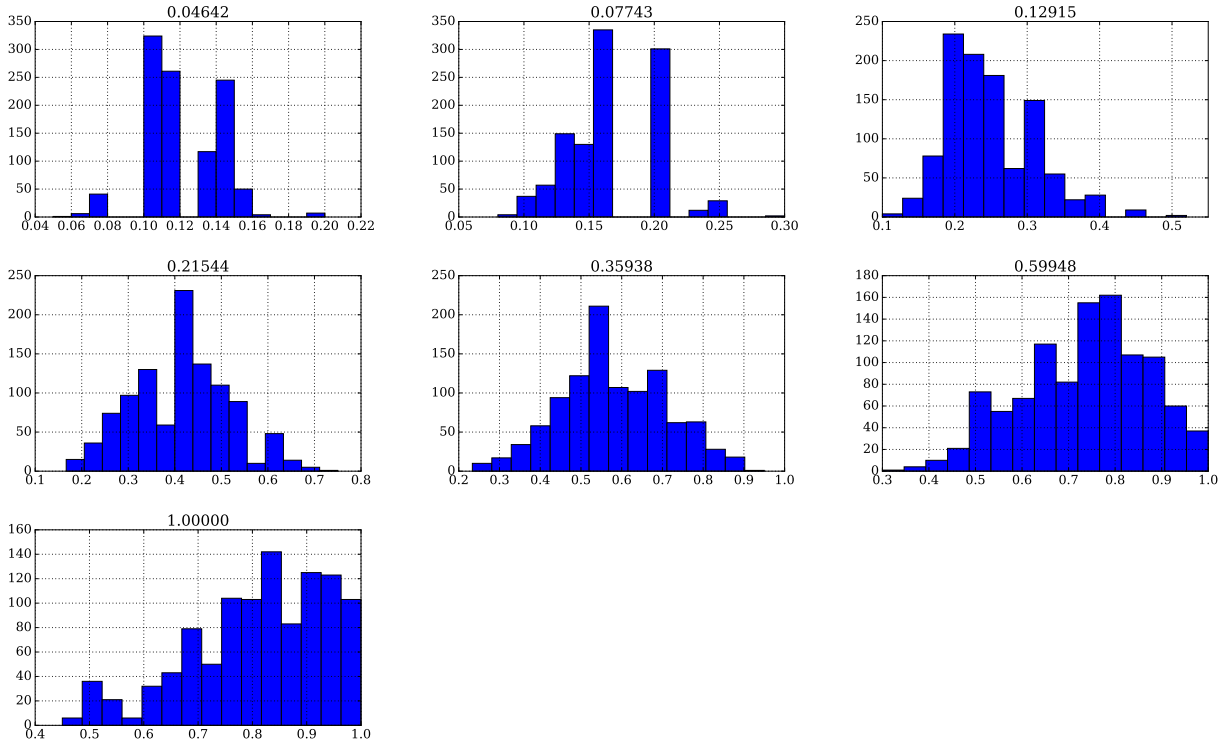


Figure -7: The occurrence frequency of amino acid Cys in the predicted sequences depending on the temperature factor $\beta = \frac{1}{T}$

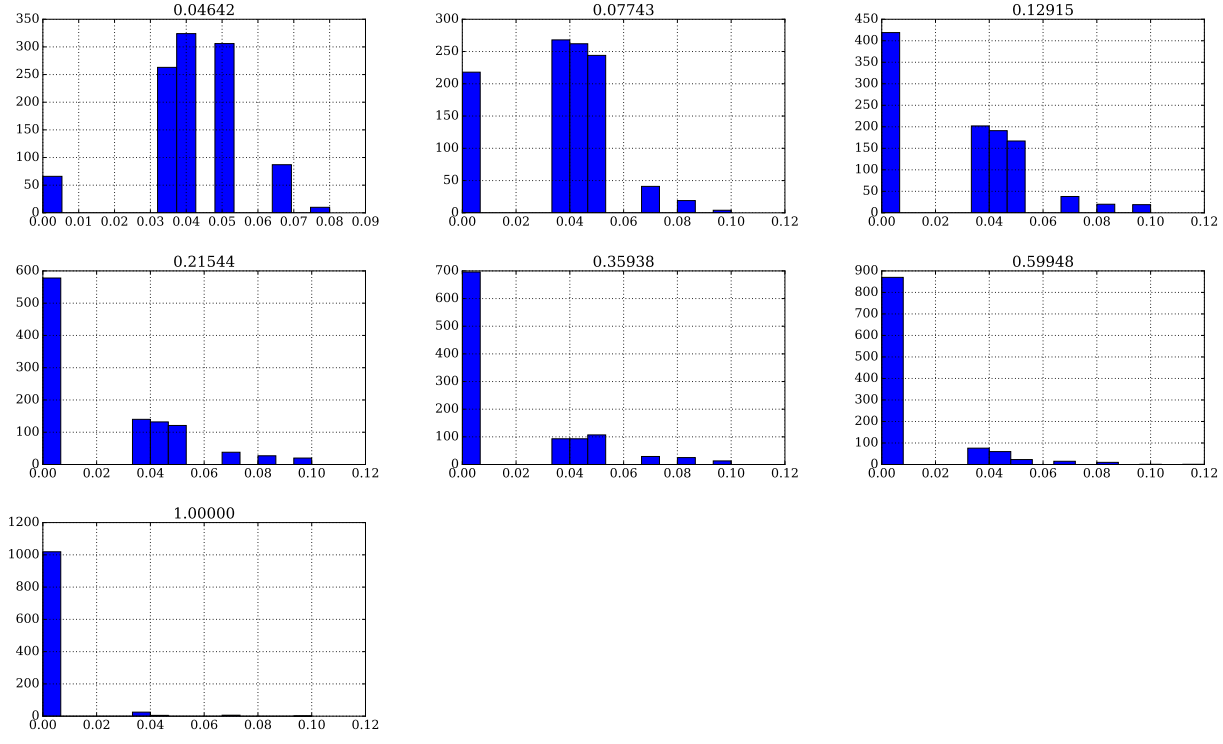


Figure -8: The occurrence frequency of amino acid Glu in the predicted sequences depending on the temperature factor $\beta = \frac{1}{T}$

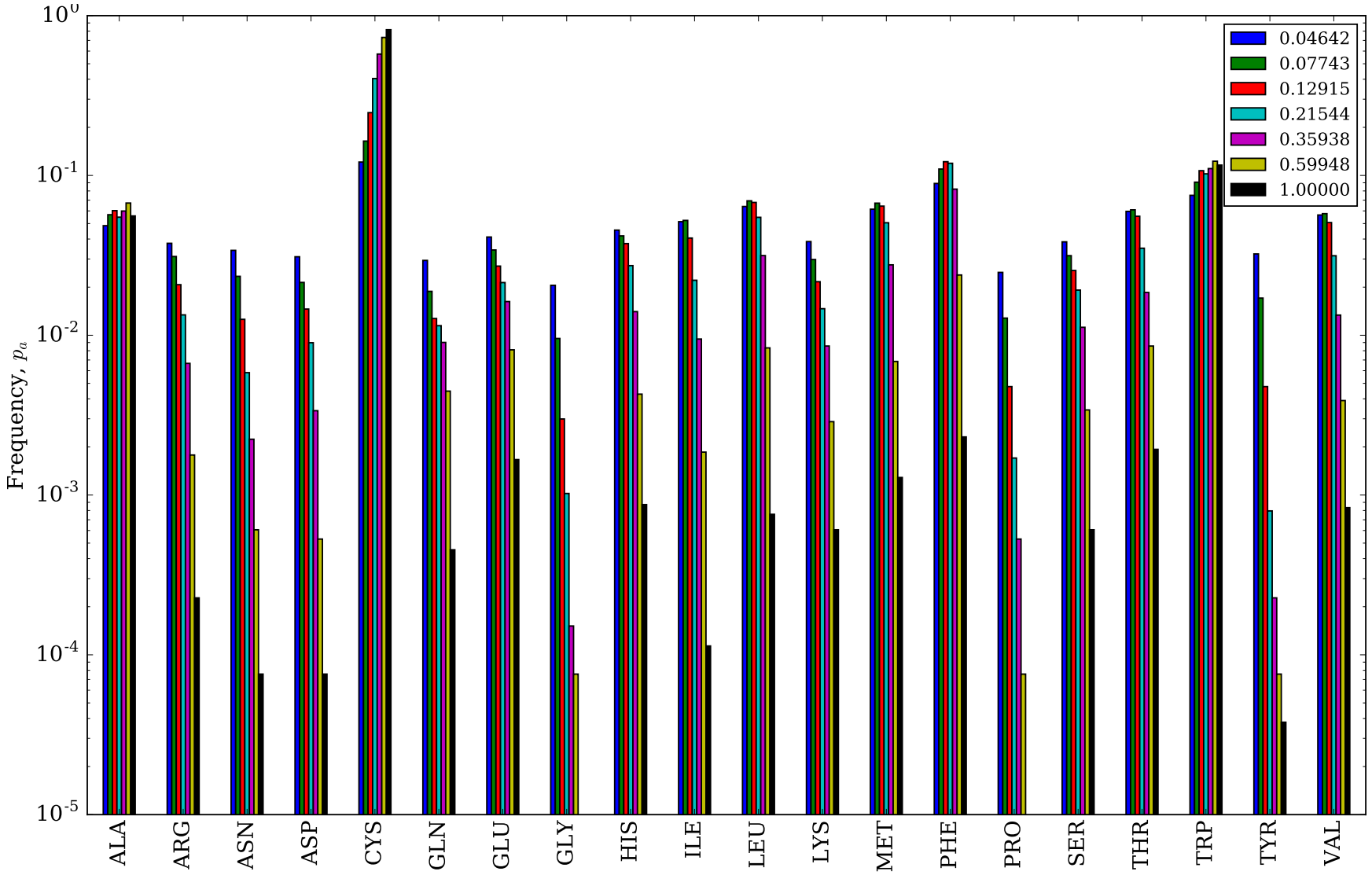


Figure -9: Average occurrence frequency of different amino acids in predicted sequences depending on temperature factor $\beta = \frac{1}{T}$