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»

Катруца Александр Михайлович

Дискретное квадратичное программирование с релаксацией при отборе признаков

010656 — Математические и информационные технологии

Выпускная квалификационная работа магистра

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

Москва
2016

Abstract

This study investigates the performance of binary quadratic programming relaxations in the side-chain prediction and feature selection problems. These problems are formulated initially as binary quadratic optimization problems which are NP-hard. To find their approximate solutions we use convex relaxations. The first relaxation changes binary variables for continuous using convex hull of the initial feasible set. The second relaxation treats non-convexity of the objective function. Semidefinite programming relaxation, shift spectrum relaxation and spectral relaxation give the convex approximations of the objective function. Also we consider linear programming relaxation to show the influence of the quadratic term.

In side-chain prediction experiments we use the subset of proteins from the PDB. We compare different approach to approximate the non-convex objective function for side-chain prediction problem and conclude what approach is the best. Also we use the specific learning procedure to get the appropriate energy function, which is the crucial one for high quality of side-chain prediction. This procedure requires the set of incorrect side-chain structures for every protein with correct side-chain structure. The shift spectrum relaxation shows the best quality for the reasonable time. The integer linear programming relaxation demonstrates slightly worse quality, but requires much less time.

In feature selection experiments we use synthetic data sets with extremely high correlated features and target vector and real data set. We illustrate the performance of the proposed approach in the feature selection problem on these datasets. In addition we compare proposed approach with other feature selection methods according to different evaluation criteria.

Contents

1 Introduction **4**

2 Problem Statement **7**

2.1 Feature selection problem 8

2.2 Side-chain prediction problem 11

2.2.1 Energy function 15

2.2.2 Decoys generation 18

3 Computational Experiment **19**

3.1 Feature selection 19

3.2 Side-chain prediction 24

3.2.1 Learning energy function 26

3.2.2 Side-chain structure optimization 27

4 Conclusion **33**

1. INTRODUCTION

This investigation consider the side-chain prediction problem and feature selection problem from the quadratic programming perspective. The feature selection problem aims to reduce the dimensionality of the data fitting problem and remove noisy, irrelevant and multicollinear features. To take into account the features similarity and features relevance we propose to formulate feature selection problem in the form of the quadratic programming, which has the single global optimum due to convexity. This approach does not require parameter vector estimation and considers only relations between features and target vector. Previously this kind of feature selection methods were called *filter methods* [1]. To evaluate the features similarity and relevance, authors use correlation coefficients [2], mutual information [3] or statistical tests [4]. The proposed method is applicable for any chosen similarity measure between features and target vector even if this similarity measure does not imply the convexity of the optimization problem. In this case convex relaxations can be used [5]. The quality of the selected feature subset is evaluated according to external criteria: variance inflation factor [6], Akaike information criterion [7] and adjusted coefficient of determination [8].

The protein folding problem is one of the most important problem in biochemistry. The goal of the protein folding problem is to recover the 3D coordinates of the protein atoms from the protein amino acid sequence. The solution of this problem gives a powerful tool to design new drugs and proteins with specific properties, which significantly affects the parmaceutical industry [9]. Protein folding problem usually is broken down into two steps: backbone modeling and side-chain prediction. To solve backbone modeling part one can use *homology modeling* [10] or *protein threading* [11], which give relatively good backbone structure prediction. This study is devoted to the side-chain prediction problem and quadratic programming approach to solve it. The aim of the prediction protein side-chain is to predict the dihedral angles of the chemical bonds between atoms from the side-chain. Experiments show that every amino acid has the set of the most probable dihedral angles. These most probable angles correspond to some side-chain states which are called *rotamers*. To use this experimental fact in prediction procedure, biologists compose rotamer libraries of such states [12, 13, 14]. These libraries allow formalizing initial side-chain prediction problem as binary optimization problem, which is proved to be NP-hard [15]. To solve binary optimization problem, grid search methods, Monte-Carlo methods, genetic

algorithm and graph-based technique are widely used. The paper [16] studies the mutation stability and uses grid search to search optimum parameters. It proposes *FoldX* algorithm which shows poor results in side-chain prediction because it is not originally designed for this problem. The paper [17] proposes *Rosetta* algorithm, which uses Monte-Carlo simulation to find solution of the corresponding optimization problem. Also, this paper represents energy function as a linear combination of the specific evergy terms taken from other papers. In [18] the *Sccomp* algorithm is introduced. Sccomp uses the genetic algorithm to solve the optimization problem. This paper proposes its own version of energy function which is represented as linear combination of different interaction energies. More details see in the original paper. The graph-based approach to solve optimization problem is preseneted in the paper [19], where the *SCWRL4* algorithm is introduced. This algorithm is based on the construction graph from the protein, elimination some edges and apply tree decomposition technique. The authors use its own defined energy function. More details see in the original paper. The main drawback of the considered approaches is that they find only local optimum not global one. Therefore, they do not guarantee that obtained side-chain structure is the best. Below we describe our approach to solve sise-chain prediction problem which is based on the convex relaxation of the initial binary optimization problem. The convexity guarantees the global optimum of the relaxaed optimization problem. Also we use a learning procedure to define energy function such that proteins with correct side-chains have the smaller energy than proteins with incorrect side-chains.

We model the protein energy by the quadratic function dependent on rotamer states of side-chains. This function takes into account both the side-chain interactions energy in the quadratic term and self-energy of every rotamer in the linear term. Therefore, based on the *minimum energy principle* this function should be appropriate to predict the optimum protein side-chain structure. We relax binary quadratic optimization problem to continues convex one. To make this relaxation, we use semidefinite programming relaxation, shift spectrum relaxation and spectral relaxation of the objective function and use convex hull of the non-convex feasible set from the initial problem. After that, we have to optimize convex function over the convex feasible set. Below we review algorithms which solve this problem efficiently.

Previously, quadratic programming was widely used in different domain applications such as control [20], portfolio optimization [21], signal and image processing [22] and sequential pro-

gramming approach [23]. The quadratic programming framework is widely used in various fields, because of there exists fast, memory-efficient and scalable algorithms to solve these problems. Therefore, we propose to formulate feature selection problem and side-chain prediction problem in the quadratic programming form and use efficient algorithms to solve them. If the objective function is the indefinite quadratic form, then we consider some convex relaxations which give approximate solutions of the initial non-convex problem. We consider the semidefinite programming relaxation [24], Lagrange relaxation [25], shift spectrum relaxation and spectral relaxation. The authors show that Lagrange relaxation is dual to the semidefinite programming relaxation [26], so we use only semidefinite programming relaxation in this study. Moreover, we need to get constraints to the quadratic optimization problem that provide the convex feasible set. These constraints should be defined according to the domain in which the quadratic programming solution is interpreted.

2. PROBLEM STATEMENT

Consider the general form of the binary quadratic programming problem:

$$\mathbf{x}^* = \arg\min_{\mathbf{x}\in\mathbb{B}^n} \mathbf{x}^\intercal \mathbf{Q}\mathbf{x}, \tag{1}$$

where $\mathbb{B}^n = \{0,1\}^n$ is a set of n dimensional binary vectors, $\mathbf{Q} \in \mathbb{R}^{n\times n}$ is a matrix. In general case the problem (1) is NP-hard. To solve this problem efficiently, one needs to convert it to the convex optimization problem:

$$\mathbf{x}^* = \arg\min_{\mathbf{x}\in\mathcal{C}} f(\mathbf{x}), \tag{2}$$

where $\mathcal{C} \subseteq \mathbb{R}^n$ is a convex set, $f : \mathcal{C} \to \mathbb{R}$ is a convex function. To solve the problem (2) and get approximation of the initial problem (1) one has to define a convex set $\mathcal{C}$ and a convex function f. The convex function f is the convex approximation of the objective function in the problem (1). The convex set $\mathcal{C}$ is defined according to the domain in which solution of the problem (2) is interpreted, see subsection 2.1 and 2.2, where we provide different definitions of the set $\mathcal{C}$. Below we discuss how function f can be defined.

Consider two approaches to define function f.

1. Shift spectrum:

$$f(\mathbf{x}) = \mathbf{x}^\intercal \hat{\mathbf{Q}}\mathbf{x} = \mathbf{x}^\intercal \mathbf{Q}\mathbf{x} - \lambda_{\min}\|\mathbf{x}\|_2^2, \tag{3}$$

   where $\hat{\mathbf{Q}} = \mathbf{Q} - \lambda_{\min}\mathbf{I}_n \succeq 0$, $\mathbf{I}_n$ is an identity matrix $n \times n$, $\lambda_{\min}$ is a minimum eigenvalue of the matrix $\mathbf{Q}$ and $\mathbf{x} \in \mathbb{R}^n$.

2. Semidefinite programming relaxation:

$$f(\mathbf{X}) = \mathrm{Tr}(\mathbf{Q}\mathbf{X}), \tag{4}$$

   where $\mathbf{X} \in \mathcal{S}_+^n$ is a symmetric, non-negative definite matrix such that

$$\mathbf{X} - \mathbf{x}\mathbf{x}^\intercal \succeq 0$$

   or using Schur complement:

$$\begin{bmatrix} \mathbf{X} & \mathbf{x} \\ \mathbf{x}^\intercal & 1 \end{bmatrix} \succeq 0. \tag{5}$$

The semidefinite programming relaxation approach requires additional constraints (5) to the problem (2).

2.1. Feature selection problem

Let $\mathbf{X} = [\boldsymbol{\chi}_1, \ldots, \boldsymbol{\chi}_n] \in \mathbb{R}^{m \times n}$ be the design matrix, where $\boldsymbol{\chi}_i \in \mathbb{R}^m$ is an i-th feature. Let $\mathbf{y} \in \mathbb{R}^m$ be the target vector. Denote by $\mathcal{J} = \{1, \ldots, n\}$ a feature index set. Let $\mathcal{A} \subseteq \mathcal{J}$ be a feature index subset. The data fitting problem is to find a parameter vector $\mathbf{w}^* \in \mathbb{R}^n$ such that:

$$\mathbf{w}^* = \arg\min_{\mathbf{w} \in \mathbb{R}^n} S(\mathbf{w}, \mathcal{A} | \mathbf{X}, \mathbf{y}, \mathbf{f}), \tag{6}$$

where S is an error function, which validates the quality of any parameter vector $\mathbf{w}$ with given target vector $\mathbf{y}$ and a function $\mathbf{f}$. The function $\mathbf{f}$ is the parameter function, which gives target vector $\mathbf{y}$ approximation.

In this study we use linear parameter function:

$$\mathbf{f}(\mathbf{X}, \mathcal{A}, \mathbf{w}) = \mathbf{X}_{\mathcal{A}} \mathbf{w},$$

where $\mathbf{X}_{\mathcal{A}}$ is the design matrix which consists of the feature vectors with indices from the set $\mathcal{A}$, and quadratic error function

$$S(\mathbf{w}, \mathcal{A} | \mathbf{X}, \mathbf{y}, \mathbf{f}) = \|\mathbf{f}(\mathbf{X}, \mathcal{A}, \mathbf{w}) - \mathbf{y}\|_2^2.$$

The features $\boldsymbol{\chi}_i, i \in \mathcal{J}$ may be noisy, irrelevant or multicollinear that leads to additional error in estimation of the optimum vector $\mathbf{w}^*$ and unstability of this vector. One can use feature selection methods to remove named features from the design matrix $\mathbf{X}$. The feature selection procedure reduces the dimensionality of the problem (6) and improves stability of the optimum vector $\mathbf{w}^*$.

The feature selection problem is

$$\mathcal{A}^* = \arg\min_{\mathcal{A} \subseteq \mathcal{J}} \mathcal{Q}(\mathcal{A} | \mathbf{X}, \mathbf{y}), \tag{7}$$

where $\mathcal{Q} : \mathcal{A} \to \mathbb{R}$ is a quality criterion, which validates the quality of the considered feature index subset $\mathcal{A} \subseteq \mathcal{J}$.

The problem (7) does not require any estimation of the optimum parameter vector $\mathbf{w}^*$, but uses only relations between the features $\boldsymbol{\chi}_i, i = 1, \ldots, n$ and the target vector $\mathbf{y}$.

Let $\mathbf{x} \in \mathbb{B}^n$ be an indicator vector such that $x_i = 1$ if and only if $i \in \mathcal{A}$. So the problem (7) can be rewrite in the following form:

$$\mathbf{x}^* = \arg\min_{\mathbf{x} \in \mathbb{B}^n} Q(\mathbf{x} | \mathbf{X}, \mathbf{y}), \tag{8}$$

where $Q : \mathbb{B}^n \to \mathbb{R}$ is another form of the criterion $\mathcal{Q}$ with domain $\mathbb{B}^n$ instead of $\mathcal{A}$.

The main idea of the proposed approach is to represent the criterion Q in the form of quadratic function:

$$Q(\mathbf{x}) = \mathbf{x}^\intercal \mathbf{Q} \mathbf{x} - \mathbf{b}^\intercal \mathbf{x}, \tag{9}$$

where $\mathbf{Q} \in \mathbb{R}^n$ is a matrix of pairwise feature similarity, $\mathbf{b} \in \mathbb{R}^n$ is a vector of feature relevance to the target vector.

The most frequently used pairwise feature similarities between features $\boldsymbol{\chi}_i$ and $\boldsymbol{\chi}_j$ are Pearson correlation coefficient [27], mutual information [28] and hessian of the error function [29]. The Pearson correlation coefficient is defined as:

$$\rho_{ij} = \frac{\operatorname{cov}(\boldsymbol{\chi}_i, \boldsymbol{\chi}_j)}{\sqrt{\operatorname{Var}(\boldsymbol{\chi}_i)\operatorname{Var}(\boldsymbol{\chi}_j)}},$$

where $\operatorname{cov}(\boldsymbol{\chi}_i, \boldsymbol{\chi}_j)$ is a covariance between features $\boldsymbol{\chi}_i$ and $\boldsymbol{\chi}_j$, $\operatorname{Var}(\cdot)$ is a variance of every feature. The sample correlation coefficient is calculated as

$$\hat{\rho}_{ij} = \frac{\sum\limits_{k=1}^{m} (\boldsymbol{\chi}_{ik} - \overline{\boldsymbol{\chi}}_i)(\boldsymbol{\chi}_{jk} - \overline{\boldsymbol{\chi}}_j)}{\sqrt{\sum\limits_{k=1}^{m} (\boldsymbol{\chi}_{ki} - \overline{\boldsymbol{\chi}}_i)^2 \sum\limits_{k=1}^{m} (\boldsymbol{\chi}_{kj} - \overline{\boldsymbol{\chi}}_j)^2}}. \tag{10}$$

In this case the elements of the matrix $\mathbf{Q} = [q_{ij}]$ are equal to the absolute values of the corresponding sample correlation coefficients:

$$q_{ij} = |\hat{\rho}_{ij}| \tag{11}$$

and the elements of the vector $\mathbf{b} = [b_i]$ are equal to absolute values of the sample correlation coefficient between every feature and the target vector:

$$b_i = |\hat{\rho}_{iy}|. \tag{12}$$

It means that we want to minimize the number of correlated features and maximize the number of features correlated to the target vector.

The mutual information between features $\boldsymbol{\chi}_i$ and $\boldsymbol{\chi}_j$ is defined as

$$I(\boldsymbol{\chi}_i, \boldsymbol{\chi}_j) = \int\int p(\boldsymbol{\chi}_i, \boldsymbol{\chi}_j) \log \frac{p(\boldsymbol{\chi}_i, \boldsymbol{\chi}_j)}{p(\boldsymbol{\chi}_)p(\boldsymbol{\chi}_j)} d\boldsymbol{\chi}_i d\boldsymbol{\chi}_j. \tag{13}$$

The sample mutual information is calculated based on estimation of the probability distribution in the equation (13).

In this case the elements of the matrix $\mathbf{Q} = [q_{ij}]$ are equal to the value of the corresponding sample mutual information:

$$q_{ij} = I(\boldsymbol{\chi}_i, \boldsymbol{\chi}_j)$$

and the elements of the vector $\mathbf{b} = [b_i]$ are equal the sample mutual information of every feature and the target vector:

$$b_i = I(\boldsymbol{\chi}_i, \mathbf{y}).$$

The third way to define the feature similarity is the hessian matrix

$$\mathbf{H} = [h_{ij}] = \frac{\partial S(\mathbf{w}, \mathbf{X}|\mathbf{y}, \mathbf{f})}{\partial w_i \partial w_j},$$

where $S(\mathbf{w}, \mathcal{A}|\mathbf{X}, \mathbf{y}, \mathbf{f})$ is the considered error function. It raises from the Taylor series expansion of the error function and shows how much h_{ij} corresponding to the $\boldsymbol{\chi}_i$ and $\boldsymbol{\chi}_j$ affects to the total approximation error. If the effect to the total error is small, then the corresponding pair of features $\boldsymbol{\chi}_i$ and $\boldsymbol{\chi}_j$ can be excluded from the active set $\mathcal{A}^*$ without significant error increasing.

Thus, we can write the problem (7) in the form (1):

$$\mathbf{x}^* = \arg\min_{\mathbf{x} \in \mathbb{B}^n} \mathbf{x}^\intercal \mathbf{Q} \mathbf{x} - \mathbf{b}^\intercal \mathbf{x}, \tag{14}$$

or

$$\tilde{\mathbf{x}}^* = \arg\min_{\tilde{\mathbf{x}} \in \mathbb{B}^{n+1}} \tilde{\mathbf{x}}^\intercal \tilde{\mathbf{Q}} \tilde{\mathbf{x}}^\intercal, \tag{15}$$

where $\tilde{\mathbf{Q}} = \begin{bmatrix} 0 & -\frac{1}{2}\mathbf{b}^\intercal \\ -\frac{1}{2}\mathbf{b} & \mathbf{Q} \end{bmatrix}$ and $\tilde{\mathbf{x}} = \begin{bmatrix} 1 \\ \mathbf{x} \end{bmatrix}$. We define *the weight of the i-th feature* $\boldsymbol{\chi}_i$ as the i-th element of the vector $\mathbf{x}^*$.

When the matrix $\mathbf{Q}$ is defined through correlation coefficient or mutual information, then it is symmetric and non-negative definite. Therefore, the main problem is to define the convex set $\mathcal{C}$ to reduce the problem (15) to the problem (2). In the formulation (14), the meaningful definition of the convex set $\mathcal{C}$ is the following:

$$\mathcal{C} = \{\mathbf{x} \in \mathbb{R}^n | \|\mathbf{x}\|_* \leq 1, x_i \geq 0, i = 1, \ldots, n\},$$

where we can use any norm which induces convex unit ball, i.e l_1, l_2 or l_∞ norms or others. In the formulation (15) the convex set $\mathcal{C}$ can be defined as:

$$\mathcal{C} = \{\tilde{\mathbf{x}} \in \mathbb{R}^{n+1} | \|\tilde{\mathbf{x}}\|_* \leq 2; x_i \geq 0, i = 1, \ldots, n; \mathbf{c}^\intercal \tilde{\mathbf{x}} = 1\},$$

where we can use any norm which induces convex unit ball, i.e l_1, l_2 or l_∞ norms or others, $\mathbf{c} \in \mathbb{B}^{n+1}$ is a constant vector such that

$$\mathbf{c} = [c_i] = \begin{cases} 1 & \text{if } i = 0 \\ 0 & \text{if } i = 1, \ldots, n. \end{cases}$$

With the linear equality constraint

$$\mathbf{c}^\mathsf{T} \tilde{\mathbf{x}} = 1$$

we make the zero element of the vector $\tilde{\mathbf{x}}^*$ equal to 1. This constraint allows to extract solution of the problem (14) from the solution of the problem (15) by taken the elements of $\hat{\mathbf{x}}$ indexing from 1 to n.

In these definitions of the convex set $\mathcal{C}$ the solution $\mathbf{x}^*$ represents probability (may be non-normalized) of the belongings every feature to the set $\mathcal{A}^*$. The set $\mathcal{A}^*$ is formed by the thresholding the solution $\mathbf{x}^*$ in the following way. We choose the threshold τ and for every element $\mathbf{x}^*$ check if $x_i^* > \tau, i \in \mathcal{J}$ then $i \in \mathcal{A}^*$.

2.2. Side-chain prediction problem

The side-chain prediction problem is to define the correct dihedral angles in the residues for every amino acid in the protein. To formalize this problem we use two facts from the biology: the stable conformation of the protein corresponds to the minimum energy, and every amino acid in protein has the most probable set of dihedral angles which is called rotamer states. So, the formal statement of the side-chain prediction problem is the following. Let N be a number of amino acids in protein, which have more than one rotamer state. Denote by $n_i,\ i = 1, \ldots, N$ a number of rotamer states for the i-th amino acid. Let $\mathbf{x} \in \mathbb{B}^n, n = \sum\limits_{i=1}^{N} n_i$ be a binary vector, which represents the correspondence between the amino acids and their rotamer states. Particularly, the vector $\mathbf{x} \in \mathbb{B}^n$ consists of N subvectors $\mathbf{x}_i \in \mathbb{B}^{n_i}, i = 1, \ldots, N$ such that every subvector $\mathbf{x}_i$ has the single non-zero element equal to one:

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_N \end{bmatrix}, \qquad \|\mathbf{x}_i\|_1 = 1,\ i = 1, \ldots, N$$

From the biology point of view, it means that every amino acid has to have the single rotamer state. Therefore, we have to add the constraint that guarantees this requirement.

Now we formulate *binary side-chain prediction problem* which is based on the *minimum energy principle*. The most stable state of the amino acid has the smallest energy:

$$\begin{gathered}\mathbf{x}^* = \arg\min_{\mathbf{x}\in\mathbb{B}^n} E(\mathbf{x}), \\ \text{s.t. } \mathbf{A}\mathbf{x} = \mathbf{1}_N,\end{gathered} \tag{16}$$

where $E : \mathbb{B}^n \to \mathbb{R}$ is an energy function, which maps the protein state represented by rotamer states of the corresponding amino acids to the real number, and $\mathbf{1}_N$ is an $N \times 1$ vector of ones. The constraint guarantees that every amino acid has the single rotamer state. To provide this requirement, the matrix $\mathbf{A} \in \mathbb{B}^{N\times n}$ has the following structure:

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

The paper [15] proved that the problem (16) is NP-hard. Therefore, to find approximate solution we need to relax the initial problem (16) and consider the approximate side-chain prediction problem. The first obvious relaxation uses continuous variables instead of binary ones. The *continuous side-chain prediction problem* is formulated as:

$$\begin{gathered}\mathbf{y}^* = \arg\min_{\mathbf{y}\in[0,1]^n} \hat{E}(\mathbf{y}) \\ \text{s.t. } \mathbf{A}\mathbf{y} = \mathbf{1}_N\end{gathered} \tag{17}$$

The energy function $\hat{E} : [0,1]^n \to \mathbb{R}$ is a convex hull of the energy function E. The solution $\mathbf{y}^*$ can be interpreted as the probability of every rotamer state to be an optimum for corresponding amino acid. Also, consider the subvectors $\mathbf{y}_i^*$ such that

$$\mathbf{y}^* = \begin{bmatrix} \mathbf{y}_1^* \\ \vdots \\ \mathbf{y}_N^* \end{bmatrix}, \quad \mathbf{y}_i^* \in \mathbb{R}^{n_i},\ i = 1, \ldots, N.$$

To restore the binary approximation $\hat{\mathbf{x}}^*$ from the $\mathbf{y}^*$, one should consider every subvector $\mathbf{y}_i^*$ and replace the maximum element of every subvector $\mathbf{y}_i^*$ by 1 and the other elements by 0:

$$\hat{\mathbf{x}}^* = \begin{bmatrix} \hat{\mathbf{x}}_1^* \\ \vdots \\ \hat{\mathbf{x}}_N^* \end{bmatrix}, \quad \hat{\mathbf{x}}_i^* \in \mathbb{B}^{n_i},\ i = 1, \ldots, N, \quad \hat{x}_{ij}^* = \begin{cases} 1, & j = \arg\max\limits_{k=1,\ldots,n_i} y_{ik}^*, \\ 0, & \text{otherwise.} \end{cases} \tag{18}$$

We represent the optimized energy function in the quadratic form with a matrix $\mathbf{Q} \in \mathbb{R}^{n \times n}$ and a linear term $\mathbf{b} \in \mathbb{R}^n$:

$$\hat{E}(\mathbf{y}) = \mathbf{y}^\intercal \mathbf{Q}\mathbf{y} + \mathbf{b}^\intercal \mathbf{y}. \tag{19}$$

The choice of such energy function is based on the idea that the optimal rotamer state for every amino acid has to have minimum side-chain interaction energy and minimum self-energy. The linear term $\mathbf{b}^\intercal \mathbf{y}$ represents the self-energy, and the quadratic term $\mathbf{y}^\intercal \mathbf{Q}\mathbf{y}$ — side-chain interaction energy. The matrix $\mathbf{Q}$ is the matrix of pairwise energy between different rotamer states of the protein:

$$\mathbf{Q} = [q_{ij}], \quad q_{ij} = \mathcal{E}(r_i, r_j), \tag{20}$$

where $\mathcal{E}$ is an energy function, which maps protein in rotamer states r_i and r_j to the corresponding energy. More details how define this function see in Section 2.2.1.

The linear term $\mathbf{b}$ is a self-energy of every rotamer states. The self-energy includes the entropy of every rotamer state e_i extracted from the rotamer library and interaction energy between side-chain in rotamer state and protein backbone $\mathcal{E}(r_i)$. The entropy of rotamer state is computed as:

$$e_i = \min\left(50, -\log\frac{p_0}{p_i}\right),$$

where p_0 is a maximum probability among rotamers for considered residue, and p_i is a probability of the i-th rotamer state for considered residue. The entropy and side-chain — backbone interaction energy can be used simultaneously or separately. In section 3.2 we discuss the performance of every approach to define the linear term.

So, now we have the following optimization problem:

$$\begin{aligned} \mathbf{y}^* = &\arg\min_{\mathbf{y} \in [0,1]^n} \mathbf{y}^\intercal \mathbf{Q}\mathbf{y} + \mathbf{b}^\intercal \mathbf{y} \\ \text{s.t. } &\mathbf{A}\mathbf{y} = \mathbf{1}_N. \end{aligned} \tag{21}$$

Because of the matrix $\mathbf{Q}$ indefiniteness, the problem (21) is not convex, and it can not be solved efficiently with global minimum guarantee. To treat this issue, we use the convex relaxations of the objective function described in the section 2 to find some approximation of the initial problem solution. Thus, we consider the following optimization problems:

- semidefinite programming relaxation (SDP):

$$\begin{aligned} \mathbf{y}^* = & \underset{\mathbf{y}\in[0,1]^n, \mathbf{Y}\in\mathcal{S}_+}{\arg\min} \operatorname{Tr}(\mathbf{Q}\mathbf{Y}) + \mathbf{b}^\intercal\mathbf{y} \\ \text{s.t. } & \mathbf{A}\mathbf{y} = \mathbf{1}_N \\ & \begin{bmatrix} \mathbf{Y} & \mathbf{y} \\ \mathbf{y}^\intercal & 1 \end{bmatrix} \succeq 0 \end{aligned} \tag{22}$$

- shift spectrum relaxation (SS):

$$\begin{aligned} \mathbf{y}^* = & \underset{\mathbf{y}\in[0,1]^n}{\arg\min}\, \mathbf{y}^\intercal\hat{\mathbf{Q}}\mathbf{y} + \mathbf{b}^\intercal\mathbf{y} \\ \text{s.t. } & \mathbf{A}\mathbf{y} = \mathbf{1}_N, \end{aligned} \tag{23}$$

  where $\hat{\mathbf{Q}} = \mathbf{Q} - \lambda_{\min}\mathbf{I}_n \succeq 0$, $\mathbf{I}_n$ is an identity matrix $n \times n$, $\lambda_{\min}$ is a minimum eigenvalue of the matrix $\mathbf{Q}$.

- spectral relaxation (SR):

$$\begin{aligned} \tilde{\mathbf{y}}^* = & \underset{\tilde{\mathbf{y}}\in[0,1]^{n+1}}{\arg\min}\, \tilde{\mathbf{y}}^\intercal\tilde{\mathbf{Q}}\tilde{\mathbf{y}} \\ \text{s.t. } & \|\tilde{\mathbf{y}}\|_2^2 = N + 1, \end{aligned} \tag{24}$$

  where $\tilde{\mathbf{Q}} = \begin{bmatrix} 0 & \frac{1}{2}\mathbf{b}^\intercal \\ \frac{1}{2}\mathbf{b} & \mathbf{Q} \end{bmatrix}$ and $\tilde{\mathbf{y}} = \begin{bmatrix} 1 \\ \mathbf{y} \end{bmatrix}$. In this problem statement we relax the constraint on the single rotamer per amino acid. Therefore, we expect that this relaxation gives the worst quality but be the fastest one among the quadratic problems. The reason of high speed of this relaxation is that it has analytical solution, which is the scaled eigenvector of the matrix $\tilde{\mathbf{Q}}$ corresponding to the smallest eigenvalue.

- linear programming (LP):

$$\begin{aligned} \mathbf{y}^* = & \underset{\mathbf{y}\in[0,1]^n}{\arg\min}\, \mathbf{b}^\intercal\mathbf{y} \\ \text{s.t. } & \mathbf{A}\mathbf{y} = \mathbf{1}_N, \end{aligned} \tag{25}$$

  where $\mathbf{b} \in \mathbb{R}^n$ is a self-energy vector. This statement is the simplest model for side-chain prediction, where interactions between rotamers are not considered.

The relevance vector can be defined as entropy, backbone energy interaction or their sum. These approaches to define linear term are compared in the computational experiment for both linear and quadratic programming problem statements.

Also in addition to the constraint $\mathbf{Ay} = \mathbf{1}_N$ one can use other constraints, which improve guarantee that every amino acid has the single rotamer state, but not violate the convexity of the feasible set, for example:

$$\operatorname{diag}(\mathbf{Y}) = \mathbf{y}. \tag{26}$$

2.2.1. Energy function

In this section we describe the way to define the energy function $\mathcal{E}$ of the protein which is expected to be the most appropriate to predict side-chain. The correct energy function $\mathcal{E}$ is crucial for high quality of side-chain prediction procedure.

The main property of the correct energy function is that the true protein structure has the smallest energy:

$$\mathcal{E}(\mathbf{x}^*) < \mathcal{E}(\mathbf{x}),$$

where $\mathbf{x}^*$ represents the protein with a *correct* side-chain structure and $\mathbf{x}$ represents any *incorrect* side-chain structure of the considered protein. The paper [30] proposes the approach to learn energy function which is based on the data collected from the correct and incorrect protein structures. Below we shortly describe this approach.

The energy function $\mathcal{E}$ is represented in the form:

$$\mathcal{E}(\mathbf{x}) = \sum_{k=1}^{M} \sum_{l=k}^{M} \int_0^{r_{\max}} n^{kl}(r) U^{kl}(r) dr, \tag{27}$$

where M is the number of atom types, which are prior known, and every atom has one of these M types, $r_{\max}$ is the maximum interaction radius, $n^{kl}(r)$ is the number density of atom pairs at a distance r between all pairs of atoms of types k and l, and U^{kl} is a scoring potential. The indices of k-type atoms we denote by A_k. The number density $n^{kl}(r)$ is computed as follows:

$$n^{kl}(r) = \frac{1}{\sqrt{2\pi\sigma^2}} \sum_{i \in A_k, j \in A_l} e^{-(r - r_{ij})^2 / 2\sigma^2}, \tag{28}$$

where σ is the standard deviation, which takes into account possible inaccuracies in protein structure, r_{ij} is the distance between i-th and j-th atoms, which is computed from the given protein.

To find the unknown scoring potential $U^{kl}(r)$ we decompose it and number density $n^{kl}(r)$ in a orthonormal polynomial basis:

$$\begin{aligned} U^{kl}(r) &= \sum_q w_q^{kl} \psi_q(r) \\ n^{kl}(r) &= \sum_q x_q^{kl} \psi_q(r), \end{aligned} \tag{29}$$

where $r \in [0, r_{\max}]$, $\psi_q(r)$ are orthonormal basis functions, particularly shifted rectangular functions. The coefficients w_q^{kl} and x_q^{kl} are the expansion coefficients of $U^{kl}(r)$ and $n^{kl}(r)$. Substitution expansions (29) to the equation (27) gives the following linear approximation:

$$\mathcal{E}(\mathbf{x}) \approx \sum_{k=1}^{M} \sum_{l=1}^{M} \sum_{q}^{Q} w_q^{kl} x_q^{kl} = \langle \mathbf{w}, \mathbf{x} \rangle, \tag{30}$$

where $\mathbf{w} \in \mathbb{R}^{Q \times M \times (M+1)/2 + N_e}$ is unknown *scoring vector* and $\mathbf{x} \in \mathbb{R}^{Q \times M \times (M+1)/2 + N_e}$ is a *structure vector* which is computed directly from the given protein, N_e is a number of amino acids types, which are in the considered proteins. The last N_e elements of the vector $\mathbf{x}$ correspond to the cumulative entropy for every amino acid type.

Assume we have N_p proteins with the correct side-chain structure, which is represented by the structure vector $\mathbf{x}_0^j,\ j = 1, \ldots, N_p$ and for every j-th protein we can generate D_j proteins with incorrect side-chain structure, which are represented by the structure vectors $\mathbf{x}_i^j,\ i = 1, \ldots, D$. We denote these proteins with incorrect side-chain as *decoys*. After that, we can state the following optimization problem to find the optimum scoring vector $\mathbf{w}^*$ which gives the smallest energy for every of N_p correct protein side-chain structures compared to incorrect structures for every protein and guarantees the positiveness of the last N_e elements corresponding to the entropy terms:

$$\begin{aligned} \mathbf{w}^* = & \underset{\mathbf{w} \in \mathbb{R}^{Q \times M \times (M+1)/2 + N_e}}{\arg\min} \frac{1}{2} \|\mathbf{w}\|_2^2 + \sum_{j=0}^{N_p} \sum_{i=0}^{D_j} C_{ij} \xi_{ij} \\ \text{s.t. } & y_{ij}(\langle \mathbf{w}, \mathbf{x}_i^j \rangle - b^j) - 1 + \xi_{ij} \geq 0,\ j = 0, \ldots, N_p,\ i = 0, \ldots, D_j \\ & \xi_{ij} \geq 0,\ j = 0, \ldots, N_p,\ i = 0, \ldots, D_j, \end{aligned} \tag{31}$$

where class labels y_{ij} are defined as

$$y_{ij} = \begin{cases} -1, & i = 0,\ j = 1, \ldots, N_p \\ +1, & i \neq 0,\ j = 1, \ldots, N_p. \end{cases}$$

and C_{ij} are parameters which penalize the violation of the first inequality constraint and computed as:

$$C_{ij} = \begin{cases} C \cdot \frac{D_j+1}{2}, & y_{ij} = -1 \\ C \cdot \frac{D_j+1}{2 \cdot D_j}, & y_{ij} = +1, \end{cases}$$

where C is defined according to cross-validation step.

The positiveness of the last N_e elements of the scoring vector $\mathbf{w}^*$ is provided by the additional set of structure vectors which is defined as:

$$\mathbf{x}_0^0 = \mathbf{0}$$

$$\mathbf{x}_i^0 = [x_{ik}^0] = \begin{cases} 1, & k = Q \cdot M \cdot \frac{M+1}{2} + i \\ 0, & \text{otherwise} \end{cases} \qquad i = 1, \ldots, N_e;$$

and corresponding class labels:

$$y_{i0} = \begin{cases} -1, & i = 0 \\ +1, & i \neq 0. \end{cases}$$

The parameters C_{ij} for this set of structure vectors are scaled in 1000 times to improve the importance of the positiveness the last N_e elements of the scoring vector $\mathbf{w}^*$.

The main assumption lies in the base of the problem statement (31) is that the single optimum scoring vector $\mathbf{w}^*$ gives the smallest energy for correct side-chain structure for every considered protein. Visualization of this idea is provided in Fig. 1.

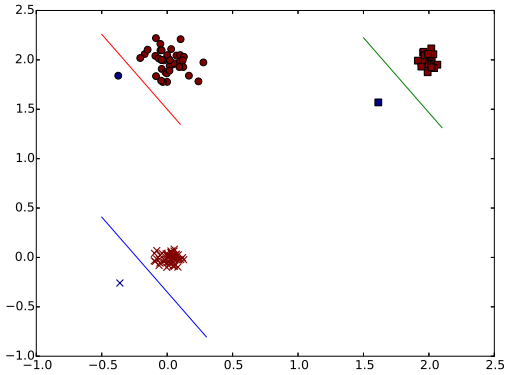


Figure 1: Illustration of the main assumption about energy function

2.2.2. Decoys generation

In this section we describe our approach to generate the incorrect side-chain structure of the protein. This step is very important to get appropriate energy function for the side-chain prediction problem. We tried different approaches and chose the most reasonable, which we provide below. The native rotamer state is the state from the rotamer library such that dihedral angles of the original side-chain equal to the dihedral angles of this rotamer state. If the library has not such state then the native state is the rotamer state, which has the smallest RMSD between original side-chain. The definition of the RMSD is provided in the equation (32). The structure vector generation is the computation of the density numbers $n^{kl}(r)$ and its basis decomposition x_q^{kl} according to the equations (28), (29).

Algorithm 1: Decoys generation procedure

Data: Protein from PDB

Result: Structured vectors $\mathbf{x}_i$

```
i = 0;
State every amino acid in the native rotamer state and generate structure vector x_i;
i = i + 1;
for every amino acid, which has more than one rotamer states do
    if there are no rotamer states, which differ from the native ones in dihedral angles then
        Set rotamer state, which has the maximum RMSD;
    if there is the single rotamer state, which differs from the native one then
        Set this rotamer state;
    if there are more than one rotamer states, which differ from the native one in dihedral angles then
        Set the most probable rotamer state;
    Generate structure vector x_i;
    i = i + 1;
```

This procedure seems to be the most appropriate, because we want to find energy function, which guarantees the minimum for the correct side-chain and increases if any side-chain is incorrect.

3. COMPUTATIONAL EXPERIMENT

In this section we describe experiments to show the performance of our method in feature selection problem and side-chain prediction problem. We compare our approach with previously published approaches for feature selection problem as well as for side-chain prediction problem. Also, in experiments about side-chain prediction we investigate different approaches to define linear term and study their performance.

3.1. Feature selection

In this section we provide the experiments on the synthetic and real data sets to show the performance of the considered approach in feature selection problem.

Data. We use the synthetic data sets generated according to procedure proposed in [31] to investigate performance of the considered methods from the multicollinearity problem point of view. The following types of data sets are considered:

- inadequate correlated — Fig. 2(a);
- adequate random — Fig. 2(b);
- adequate redundant — Fig. 2(c);
- adequate correlated — Fig. 2(d).

Also we use the real dataset of diesel fuels NIR spectra [32].

Evaluation criteria. To validate the selected feature subset we use the following criteria widely used in papers.

Variance inflation factor. To diagnose multicollinearity, the paper [33] uses the variance inflation factor VIF_j. The VIF_j shows a linear dependence between the j-th feature and the other features. To compute VIF_j estimate the parameter vector $\mathbf{w}^*$ according to the problem (6) assuming $\mathbf{y} = \boldsymbol{\chi}_j$ and extracting j-th feature from the index set $\mathcal{A} = \mathcal{A} \backslash j$. The VIF_j is computed with the following equation:

$$\mathrm{VIF}_j = \frac{1}{1 - R_j^2},$$

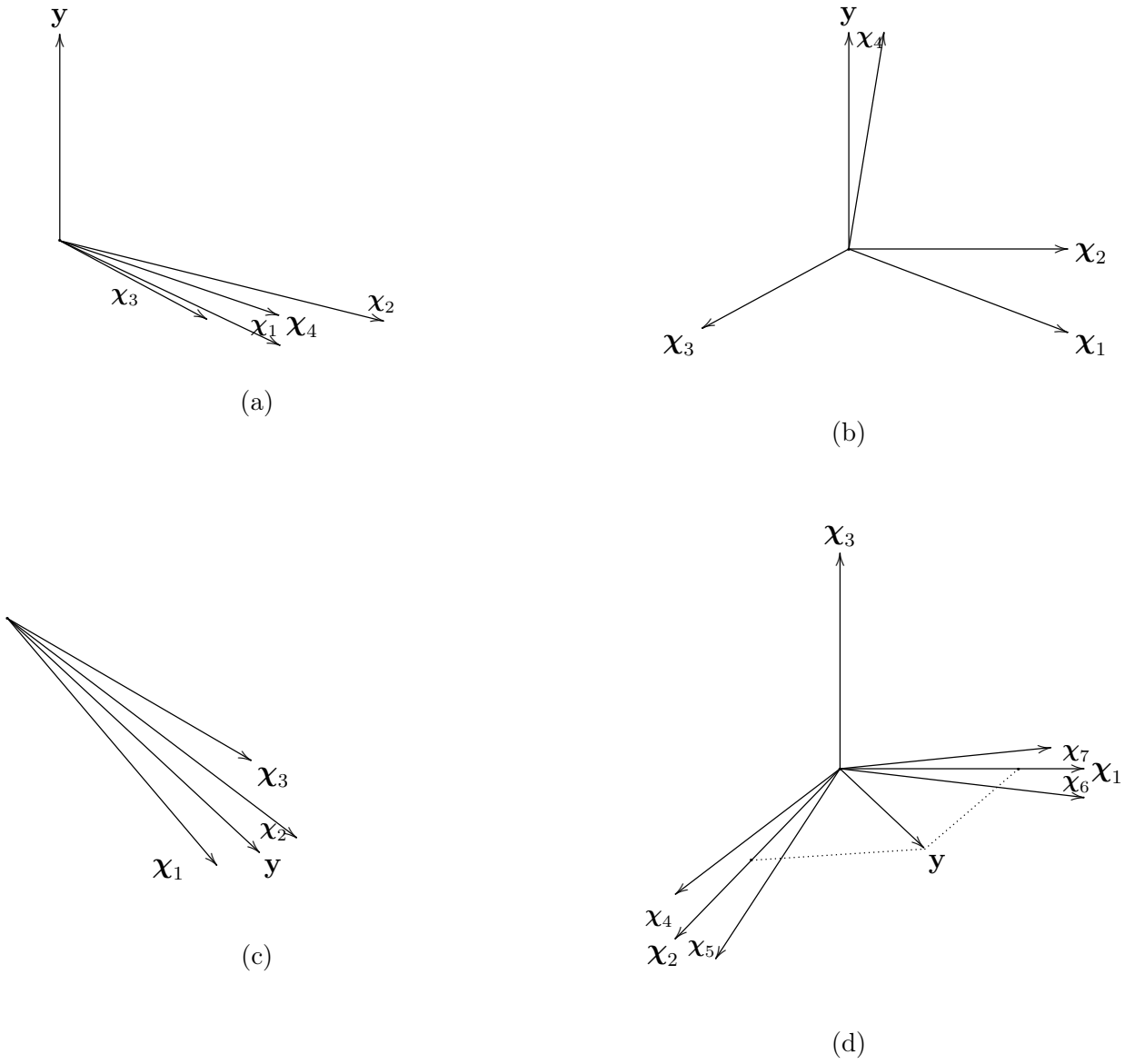


Figure 2: Synthetic test data sets configuration: (a) inadequate correlated, (b) adequate random, (c) adequate redundant, (d) adequate correlated.

where $R_j^2 = 1 - \frac{\text{RSS}}{\text{TSS}}$ is the coefficient of determination and $\text{RSS} = \sum_{i=1}^{m} \|\boldsymbol{\chi}_j - \mathbf{X}_{\mathcal{A}}\mathbf{w}^*\|_2^2$,
$\text{TSS} = \sum_{i=1}^{m}(\chi_{ji} - \overline{\boldsymbol{\chi}}_j)$, $\overline{\boldsymbol{\chi}}_j = \frac{1}{m}\sum_{i=1}^{m}\chi_{ji}$.

The paper [33] states that if $\text{VIF}_j \gtrsim 5$ then the associated element of the vector $\mathbf{w}^*$ is poorly estimated because of multicollinearity. Denote by VIF the maximum value of VIF_j for all $j \in \mathcal{A}$:

$$\text{VIF} = \max_{j \in \mathcal{A}} \text{VIF}_j.$$

Stability. To estimate the stability R of the parameter $\mathbf{w}$ estimation based on the selected feature subset $\mathcal{A}$, we use the logarithm of the the matrix $\mathbf{X}^\intercal\mathbf{X}$ condition number:

$$R = \ln \frac{\lambda_{\min}}{\lambda_{\max}},$$

where the $\lambda_{\max}$ and $\lambda_{\min}$ are the maximum and minimum non-zero eigenvalues of the matrix $\mathbf{X}^\intercal\mathbf{X}$.

The larger R is, the more stable parameter estimation.

Complexity. To measure complexity C of the selected feature subset $\mathcal{A}$ we use the cardinality of this subset, i.e.

$$C = |\mathcal{A}|.$$

The less complexity is, the better selected subset.

Mallow's C_p. The Mallow's C_p criterion [34] trades off the residual norm $r = \|\mathbf{y} - \mathbf{X}_{\mathcal{A}}\mathbf{w}^*\|_2^2$ and the number of features p. The Mallow's C_p defined as

$$C_p = \frac{r_p}{r} - m + 2p,$$

where r_p is similar to r, but computed with p features only. In terms of this criterion the smaller C_p is, the better feature subset.

BIC. Information criterion BIC [35] defined as

$$\mathrm{BIC} = r + p \log m.$$

The smaller value of BIC is, the better model fits the target vector.

Considered criteria are summarized in the table 1.

Performance analysis. This paragraph is devoted to performance analysis and comparisons quadratic programming approach with other feature selection methods. We use the synthetic datasets dataset of diesel fuel NIR spectra.

In figures below we choose the correlation coefficient (11) to generate matrix $\mathbf{Q}$ and correlation between features and target vector (12) to generate linear term $\mathbf{b}$. Fig. 3 shows the number of selected features versus the chosen threshold τ for every kind of synthetic test data sets.

Table 1: A list of the criteria to evaluate the selected feature subset

Name	*Formula*	*Meaning*
VIF	$\mathrm{VIF} = \max_{j \in \mathcal{A}} \frac{1}{1-R_j^2}$	Indicator of the the multicollinear features existence
Stability	$R = \ln \frac{\lambda_{\min}}{\lambda_{\max}}$	An indicator of the model stability
Complexity	$C = \lvert\mathcal{A}^*\rvert$	The number of the selected features
Mallow's C_p	$C_p = \frac{r_p}{r} - m + 2p$	A trade-off between accuracy and number of features
BIC	$\mathrm{BIC} = r + p \log m$	A trade-off between residues norm and number of features

Fig. 3(a) shows that all features have the same and very small weights, which means that these features are irrelevant. Fig. 3(c) shows that all features have the same weights, but opposite to fig. 3(a) these weights are much bigger, which means that all features are relevant and any feature can be selected. Fig. 3(d) shows that subsets of features have the same weights ans are excluded simultaneously. Every that subset corresponds to orthogonal feature and features, which are correlated to it.

Fig. 4 show the dependence of the error function S on the threshold τ. These figures also clearly represents the structure of the test data sets, which means that quadratic programming feature selection extracts such patterns from the dataset. The main reason of good representation is the choice of the correlation coefficient as the function to generate matrix $\mathbf{Q}$ and linear term $\mathbf{b}$.

Tables 2, 3, 4 show that the proposed approach is appropriate for every test except adequate correlated set in contrast with other feature selection methods. In this case we use a correlation coefficient as a similarity measure between features and target vector. Therefore, we can not take into account the feature significance in estimation parameter vector $\mathbf{w}$. Because of that the proposed approach does not show good quality for the adequate correlated datatset. To treat this problem, we can fix the linear term $\mathbf{b}$ definition and add the significance of every feature in estimation parameter vector $\mathbf{w}$ to the correlation with the target vector.

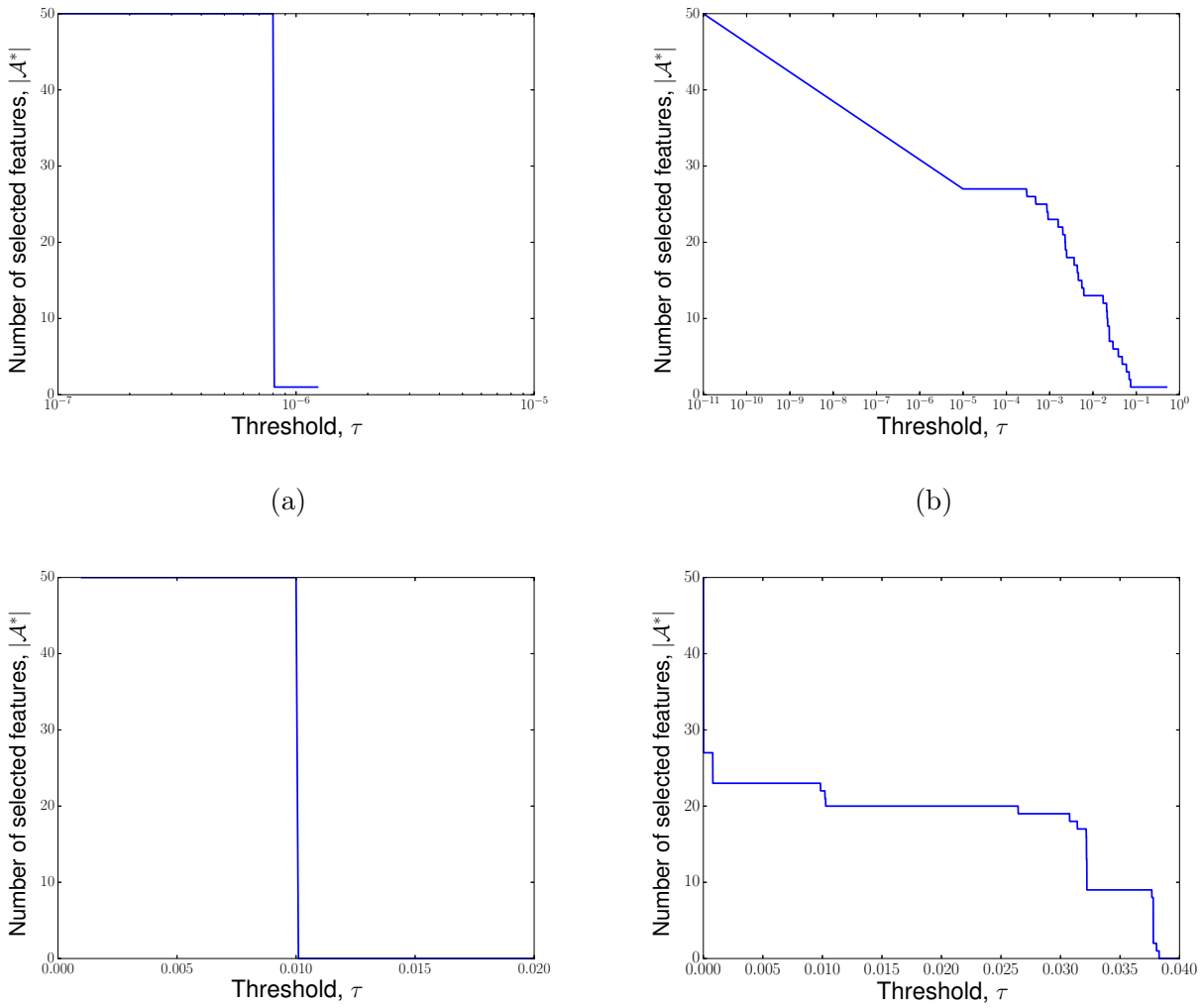


Figure 3: Dependence the cardinality of the active index set $\mathcal{A}$ on the threshold τ for: (a) inadequate correlated data set, (b) adequate random data set, (c) adequate redundant data set, (d) adequate correlated data set.

Now we provide the similar analysis for NIR spectra of diesel fuel dataset in Fig. 5, where we compare dependence of residual norm on the number of the selected features based on correlation coefficient and mutual information similarity measures. These plots show that correlation coefficient similarity measure is better to identify the minimum number of features which give appropriate quality.

Table 5 compares the considered approach with other feature selection methods on the NIR spectra of diesel fuel. This table shows that quadratic programming approach is comparable with other feature selection methods, which estimate the parameter vector $\mathbf{w}$ and select features

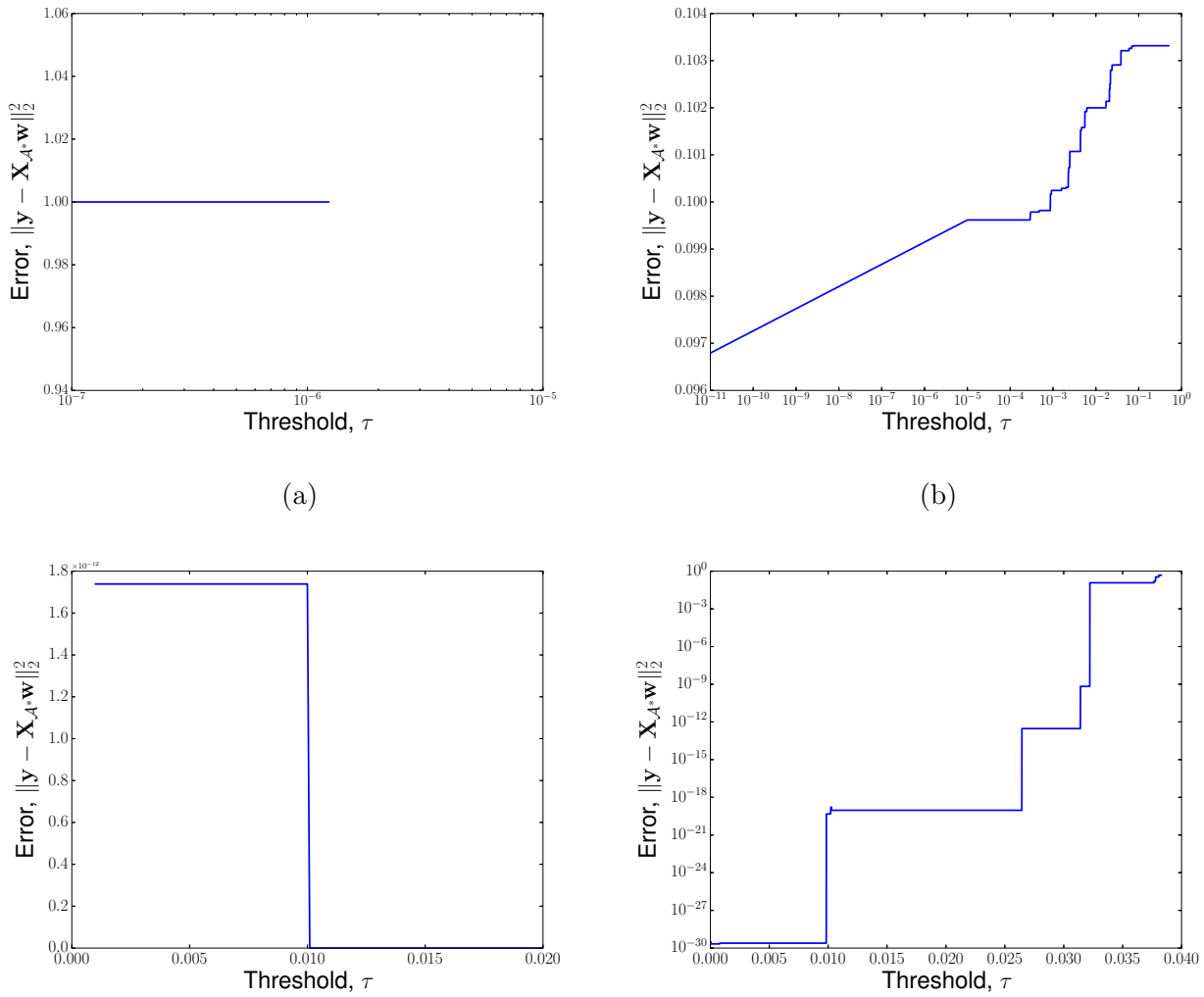


Figure 4: Dependence error function S on the threshold τ for considered types of test data sets: (a) inadequate correlated data set, (b) adequate random data set, (c) adequate redundant data set, (d) adequate and correlated data set.

simultaneously.

3.2. Side-chain prediction

In this subsection we present the learning energy function procedure and the performance analysis of the proposed approach to the side-chain prediction problem.

Table 2: Evaluation criteria on the inadequate correlated data set — Fig. 2(a)

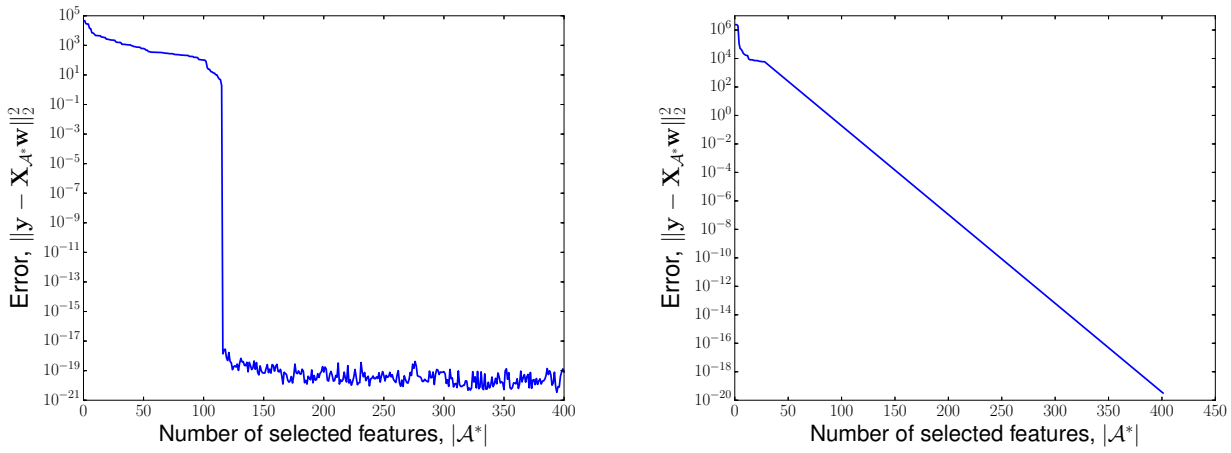
Method	C_p	RSS	R	VIF	BIC
QP(ρ)	-997	—	—	—	—
LARS	-997	—	—	—	—
Genetic	-997	—	—	—	—
Lasso	-997	1	-6.57	16.6	310.48
Ridge	-997	1	-6.69	16.6	346.39
Stepwise	-997	1.68	-6.69	16.6	347.01
Elastic Net	-997	1	-6.58	16.6	310.48

Table 3: Evaluation criteria for the adequate and random data sets — Fig. 2(b)

Method	C_p	RSS	R	VIF	BIC
QP(ρ)	-997	$1.2 \cdot 10^{-9}$	0	0.24	6.9
Lasso	$7 \cdot 10^{6}$	$8.50 \cdot 10^{-4}$	0	0.25	6.9
Elastic Net	$8.76 \cdot 10^{-4}$	$8.76 \cdot 10^{-4}$	0	0.25	6.9
Ridge	$7.97 \cdot 10^{9}$	0.97	0	0.25	7.88
LARS	-997	$1.3 \cdot 10^{-10}$	-0.78	0.32	8.29
Genetic	-997	$1.36 \cdot 10^{-10}$	-3.31	0.9	52.5
Stepwise	-997	$1.33 \cdot 10^{-10}$	-3.36	0.89	53.88

Table 4: Evaluation criteria for the adequate and redundant data set — Fig. 2(c)

Method	C_p	RSS	R	VIF	BIC
QP(ρ)	-997	$8.5 \cdot 10^{-11}$	0	0.25	6.9
Lasso	$5.16 \cdot 10^{8}$	$8.5 \cdot 10^{-4}$	0	0.24	6.9
Ridge	$5.9 \cdot 10^{11}$	0.97	-27.13	$2.9 \cdot 10^{9}$	346.36
Elastic Net	$5.16 \cdot 10^{8}$	$8.5 \cdot 10^{-4}$	-25.01	$2.5 \cdot 10^{9}$	41.45
Genetic	-997	$1.67 \cdot 10^{-12}$	-27.11	$2.87 \cdot 10^{9}$	345.39
Stepwise	-997	$1.73 \cdot 10^{-12}$	-27.13	$2.9 \cdot 10^{9}$	345.39
LARS	-997	$1.65 \cdot 10^{-12}$	-27.13	$2.9 \cdot 10^{9}$	345.39



(a) (b)

Figure 5: Dependence of resudual norm on the number of selected features for (a) correlation coefficient and (b) mutual information similarity measures

Table 5: Evaluation criteria for the diesel NIR spectra dataset

Method	C_p	r	R	VIF	BIC
QP (ρ)	-110	$1.37 \cdot 10^{-18}$	-25.7	$6.43 \cdot 10^{6}$	548.38
Genetic	-110.88	$7.68 \cdot 10^{-30}$	-24	$8.13 \cdot 10^{5}$	534.19
LARS	$3.22 \cdot 10^{21}$	$2.07 \cdot 10^{-7}$	-28.3	$7.94 \cdot 10^{7}$	529.47
Lasso	$2.5 \cdot 10^{28}$	1.61	-27.72	$1.03 \cdot 10^{21}$	1712.92
ElasticNet	$2.51 \cdot 10^{28}$	1.61	-27.72	$1.03 \cdot 10^{21}$	1712.92
Stepwise	$3.66 \cdot 10^{29}$	23.56	-36.78	$1.94 \cdot 10^{22}$	1919.23
Ridge	$1.59 \cdot 10^{28}$	1.02	-36.22	$1.07 \cdot 10^{22}$	$1.79 \cdot 10^{3}$

3.2.1. Learning energy function

Here we describe the used data, learning energy function procedure and validate the obtained scoring vector $\mathbf{w}^*$.

Data. To learn energy function we use the set of non-homologous proteins. The size of train set is 2500 proteins and the size of test set is 865 proteins. To get the structure vectors $\mathbf{x}_i^j$ we use the following parameters in the equations (27), (28) and (29): maximum interaction radius $r_{\max} = 10$, standard deviation $\sigma = 2$, number of atom types $M = 20$, expansion order $Q = 5$.

Therefore, according to equation (30) the dimension of the optimization problem (31) is equal to 1072. These parameters are chosen according to biological sense and carried out experiments.

Scoring vector training and validation. We train the scoring vector on the training set with different parameters C and validate the obtained scoring vector on the test set. The problem (31) is very similar to the classical SVM optimization problem. Therefore, to solve it we modify the procedure SMO proposed in the paper [36] for the SVM problem. Unfortunately, this method requires a lot of time to converge (a few days for $C = 500$) and the convergence time increases with increasing parameter C. Because of that, we can not try more parameters C in experiments. This procedure solves the dual problem and then restore the solution of the primal problem from the dual solution. The obtained learning curve is shown in Fig. 6. The optimum parameter $C = 500$. Also Fig. 7 shows the typical convergence process. Due to the requirements that the last N_e elements of the scoring vector $\mathbf{w}^*$ have to be non-negative, the convergence process starts from the sharp quality increasing and after that the process makes the last N_e elements of the scoring vector $\mathbf{w}^*$ non-negative with slight quality decreasing.

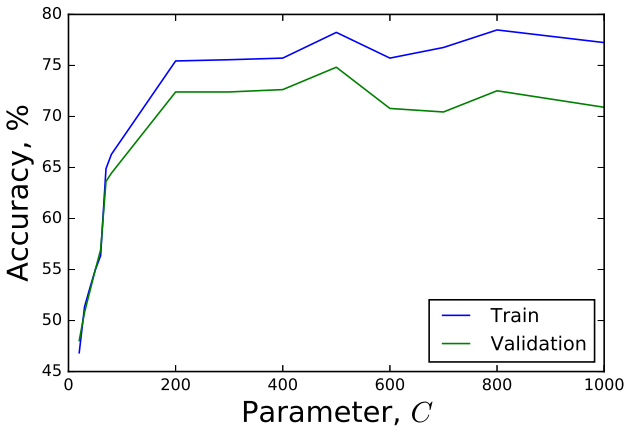


Figure 6: Accuracy for the different parameter C

3.2.2. Side-chain structure optimization

Data. To compare different approaches to solve side-chain prediction problem, we use the dataset *SCWRL4* from the paper [19]. This dataset consists of 379 proteins. The histogram of dimension distribution is shown in Fig. 8(a). To make a preliminary test, we select 32 proteins from the SCWRL4 dataset and denote this subset as *subSCWRL4*. The histogram of

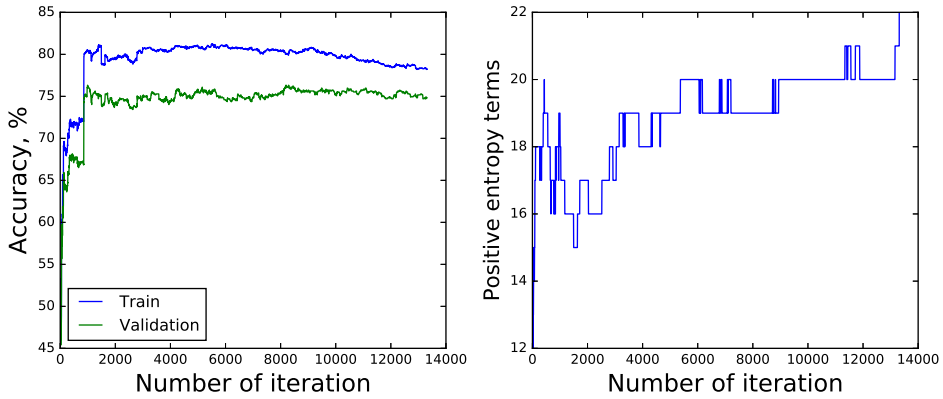
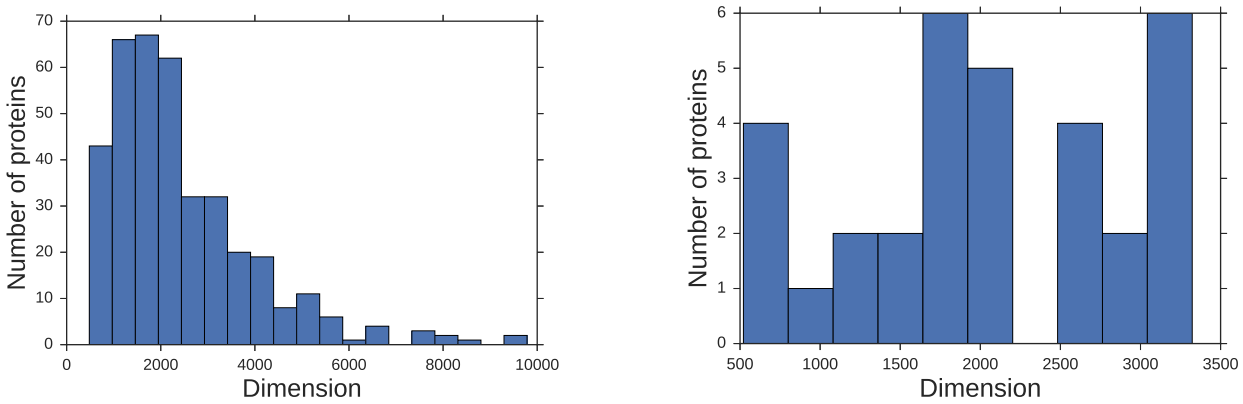


Figure 7: Typical convergence scheme of the learning procedure (here $C = 500$)

the dimension distribution for the subSCWRL4 dataset is shown in Fig. 8(b).

Also, we use the rotamer library [37] to generate possible protein conformations. We use the rotamer states from this library, which has probability greater that 0.01. This threshold probability has a significant influence on the dimensions of optimization problems discussed later. The less threshold probability is, the larger dimension.



(a) SCWRL4 dataset

(b) subSCWRL4 dataset

Figure 8: Distributions of the dimension for considered datasets

Evaluation criteria. To compare the predicted protein with the original one we use the following criteria, which are given by biology community:

- ratio of the correct predicted χ_1 angles with tolerance 40°:

$$\overline{\chi}_1 = \frac{1}{N}\sum_{i=1}^{N}[|\hat{\chi}_1 - \chi_1| < 40^\circ \vee ||\hat{\chi}_1 - \chi_1| - 360^\circ| < 40^\circ],$$

where χ_1 is a dihedral angle in the original protein, $\hat{\chi}_1$ is a dihedral angle in the predicted protein and $[a] = \begin{cases} 1 & \text{if } a \text{ is True} \\ 0 & \text{if } a \text{ is False.} \end{cases}$

- ratio of the correct predicted χ_1 and χ_2 angles with tolerance $40°$:

$$\chi_{12} = \frac{1}{N}\sum_{i=1}^{N}[|\hat{\chi}_1-\chi_1| < 40° \vee ||\hat{\chi}_1-\chi_1|-360°| < 40°] \cdot [|\hat{\chi}_2-\chi_2| < 40° \vee ||\hat{\chi}_2-\chi_2|-360°| < 40°],$$

where χ_2 is a dihedral angle in the original protein, $\hat{\chi}_2$ is a dihedral angle in the predicted protein and $\chi_1, \hat{\chi}_1$ are the same as in $\overline{\chi}_1$.

- root mean square deviation (RMSD)

$$\text{RMSD} = \sqrt{\frac{1}{N_A}\sum_{i=1}^{N_A}(r_i - \hat{r}_i)^2}, \tag{32}$$

where r_i is a position of the i-th atom in the original protein, $\hat{r}_i$ is a position of the i-th atom in the predicted protein and N_A is a number of atoms in the considered protein.

Performance analysis. In this paragraph we present the performance of the proposed approach in side-chain prediction problem. We compare different approaches to solve side-chain prediction problem from the simplest one to more complicated approaches.

1. Linear programming based on the entropy term
2. Linear programming based on side-chain —back-bone interaction energy
3. Linear programming based on entropy term and side-chain —back-bone energy
4. Quadratic programming based on side-chain — side-chain interaction energy
5. Quadratic programming based on side-chain — side-chain interaction energy and linear term based on 1 or 2 or 3.

To solve the problem (22) we use the splitting conic solver [38, 39]. To solve the problem (23) with different linear term definitions we use CVX, a package for specifying and solving convex programs [40, 41] with MOSEK solver [42]. The problem (24) has analytical solution, which is the scaled eigenvector corresponding to the minimum eigenvalue, so to find these eigenvalue

and eigenvector we use the algorithm from the paper [43]. The problem (25) is solved with MATLAB Optimization Toolbox [44].

Tab. 6 shows performance of the considered approaches on the subSCWRL4 dataset. It shows that quadratic objective function with linear term as a sum of entropy and backbone interaction energy and shift spectrum relaxation gives the best quality of prediction for the not the smallest but reasonable time. At the same time, experiments demonstrate that quadratic term improves the prediction quality for shift spectrum relaxation on more than 3 % but requires more time. In addition, the most important part of the linear term is the entropy which significantly improves the quality of every considered approach. The next observation is that spectral relaxation, as it was expected, gives the poorest quality but is the fastest among the quadratic problem relaxations. SDP relaxation is poorly scalable, so we test it only on proteins, which corresponding problem dimension is less than 1800. The quality of this relaxation is not so good to require so much time. Thus, the main conclusion is that the best approach is shift spectrum relaxation with linear term as sum entropy and backbone interaction energy. Also the linear programming approach gives worse quality but is much faster. In both cases entropy plays important role in high quality.

Table 6: Performance of different approaches for subSCWRL4 dataset, energies computed for parameters learned with $C = 500$

Algorithm	$\overline{\chi}_1$	χ_{12}	RMSD, Å	Time, s
QP + entropy (24)	52.42%	32.77%	1.74	2.57
QP + backbone (24)	54.29%	39.94%	1.83	2.16
QP + entropy and backbone (24)	63.69%	49.30%	1.67	2.25
LP backbone (25)	68.33%	54.31%	1.37	1.57
SDP ($n < 1800$) (22)	71.28%	60.56%	1.43	$\sim 10^4$
QP + backbone (23)	74.93%	60.48%	1.30	164.89
LP entropy (25)	77.40%	66.84%	1.42	1.61
LP entropy and backbone (25)	78.33%	66.92%	1.37	1.61
QP + entropy (23)	80.06%	68.68%	1.29	176.87
QP + entropy and backbone (23)	81.91%	70.78%	1.20	181.98

Tab. 7 shows the performance of the considered approaches on the SCWRL4 dataset. The ranking of the considered approaches on subSCWRL4 and SCWRL4 datasets is the same. Also the values of evaluation criteria for the SCWRL4 dataset are slightly bigger than for subSCWRL4 dataset in case of every considered approach.

Table 7: Performance of different approaches for SCWRL4 dataset, energies computed for parameters learned with $C = 500$

Algorithm	$\overline{\chi}_1$	χ_{12}	RMSD, Å	Time, s
QP + backbone (24)	53.91%	38.61%	1.89	30.49
QP + entropy (24)	56.59%	37.67%	1.73	32.36
QP + entropy and backbone (24)	65.54%	51.1%	1.64	27.26
LP backbone (25)	69.36%	54.89%	1.53	17.77
SDP ($n < 1800$) (22)	72.43%	60.56%	1.41	$\sim 10^6$
QP + backbone (23)	75.13%	59.94%	1.33	$3.41 \cdot 10^3$
LP entropy (25)	78.14%	67.24%	1.40	17.22
LP entropy and backbone (25)	78.7%	67.4%	1.35	18.34
QP + entropy (23)	80.29%	68.31%	1.30	$3.24 \cdot 10^3$
QP + entropy and backbone (23)	82.58%	70.97%	1.19	$3.29 \cdot 10^3$

To compare our best approach (23) with other methods proposed previously, we use the results and dataset from the paper [45]. The dataset consists of 240 proteins and the corresponding dimensional distribution is shown in Fig. 9.

Table 8 presents the results of different algorithms on the considered data set. The time is mentioned in the paper [45] only in the order of magnitude. The algorithms are sorted in descending order according to the time and dehidral angles accuracy measures. This table shows that our approach is not the best, but in the top of the algorithm list. The possible explanation of this ordering is that our energy function is not so good as we expect. Therefore, we will try to use other energy functions or improve the current energy finction learning procedure. One more possible explanation is that shift spectrum changes the problem too much to get better quality.

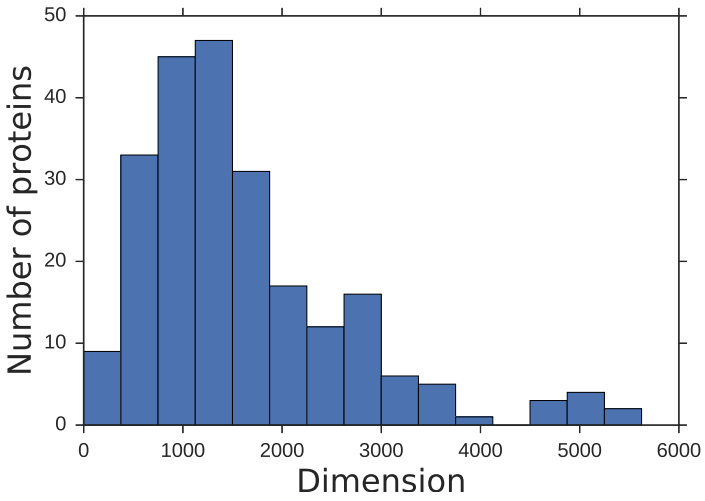


Figure 9: Dimension distribution of the protein data set from the paper [45]

Table 8: Comparison our best algorithms with other methods from papers

Algorithm	$\overline{\chi}_1$	χ_{12}	Time per protein, s
SCWRL4 [19]	85.2%	72.0%	$\sim 1-10$
QP + entropy and backbone (23)	82.6%	70.6%	4.6
Rosetta [17]	83.3%	68.2%	$\sim 1-10$
Sccomp-S [18]	82.3%	59.6%	$\sim 100-1000$
Sccomp-I [18]	81.3%	57.7%	$\sim 1-10$
FoldX [16]	70.4%	49.7%	$\sim 100-1000$

4. CONCLUSION

This study investigates the side-chain prediction and feature selection problems from the quadratic programming point of view. The feature selection and side-chain prediction problems are stated in the form of the binary quadratic programming problems. Then we relax binary problems to continuous ones and in the case of non-convexity of the objective function we propose some types of convex relaxations like spectral relaxation, shift spectrum relaxation and semidefinite programming relaxation. The shift spectrum relaxation shows the best quality in the side-chain prediction problem among proposed in this study. It means that the quadratic term improves the prediction quality. The linear programming approaches gives slightly worse quality but for less time. Also the entropy is significantly improves both linear and quadratic approaches. The potential improvement of the considered approach is fine-tuning energy function, decreasing relaxation influence and speed up optimization step.

In the feature selection problem experiments, the quadratic programming approach demonstrates the high performance in the feature dependence detection and solving multicollinearity problem. We compare quadratic programming approach with other feature selection methods and show that our approach is better to identify multicollinear and redundant features.

References

[1] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *The Journal of Machine Learning Research*, 3:1157–1182, 2003.

[2] Lei Yu and Huan Liu. Feature selection for high-dimensional data: A fast correlation-based filter solution. In *ICML*, volume 3, pages 856–863, 2003.

[3] Hanchuan Peng, Fuhui Long, and Chris Ding. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(8):1226–1238, 2005.

[4] Huiqing Liu, Jinyan Li, and Limsoon Wong. A comparative study on feature selection and classification methods using gene expression profiles and proteomic patterns. *Genome informatics*, 13:51–60, 2002.

[5] Zhi-Quan Luo, Wing-Kin Ma, Anthony Man-Cho So, Yinyu Ye, and Shuzhong Zhang. Semidefinite relaxation of quadratic optimization problems. *IEEE Signal Processing Magazine*, 27(3):20, 2010.

[6] Trevor A Craney and James G Surles. Model-dependent variance inflation factor cutoff values. *Quality Engineering*, 14(3):391–403, 2002.

[7] Wei Pan. Akaike's information criterion in generalized estimating equations. *Biometrics*, 57(1):120–125, 2001.

[8] Lonnie Magee. R^2 measures based on Wald and likelihood ratio joint significance tests. *The American Statistician*, 44(3):250–253, 1990.

[9] JM Yon. Protein folding: a perspective for biology, medicine and biotechnology. *Brazilian Journal of Medical and Biological Research*, 34(4):419–435, 2001.

[10] Torsten Schwede, Jürgen Kopp, Nicolas Guex, and Manuel C Peitsch. Swiss-model: an automated protein homology-modeling server. *Nucleic acids research*, 31(13):3381–3385, 2003.

[11] Susana Cristobal, Adam Zemla, Daniel Fischer, Leszek Rychlewski, and Arne Elofsson. A study of quality measures for protein threading models. *BMC bioinformatics*, 2(1):1, 2001.

[12] Roland L Dunbrack. Rotamer libraries in the 21 st century. *Current opinion in structural biology*, 12(4):431–440, 2002.

[13] Marc De Maeyer, Johan Desmet, and Ignace Lasters. All in one: a highly detailed rotamer library improves both accuracy and speed in the modelling of sidechains by dead-end elimination. *Folding and Design*, 2(1):53–66, 1997.

[14] Simon C Lovell, J Michael Word, Jane S Richardson, and David C Richardson. The penultimate rotamer library. *Proteins: Structure, Function, and Bioinformatics*, 40(3):389–408, 2000.

[15] Tatsuya Akutsu. NP-hardness results for protein side-chain packing. *Genome informatics*, 8:180–186, 1997.

[16] Raphael Guerois, Jens Erik Nielsen, and Luis Serrano. Predicting changes in the stability of proteins and protein complexes: a study of more than 1000 mutations. *Journal of molecular biology*, 320(2):369–387, 2002.

[17] Brian Kuhlman and David Baker. Native protein sequences are close to optimal for their structures. *Proceedings of the National Academy of Sciences*, 97(19):10383–10388, 2000.

[18] Eran Eyal, Rafael Najmanovich, Brendan J Mcconkey, Marvin Edelman, and Vladimir Sobolev. Importance of solvent accessibility and contact surfaces in modeling side-chain conformations in proteins. *Journal of computational chemistry*, 25(5):712–724, 2004.

[19] Georgii G Krivov, Maxim V Shapovalov, and Roland L Dunbrack. Improved prediction of protein side-chain conformations with SCWRL4. *Proteins: Structure, Function, and Bioinformatics*, 77(4):778–795, 2009.

[20] Alberto Bemporad, Manfred Morari, Vivek Dua, and Efstratios N Pistikopoulos. The explicit solution of model predictive control via multiparametric quadratic programming. In *American Control Conference, 2000. Proceedings of the 2000*, volume 2, pages 872–876. IEEE, 2000.

[21] E Ammar and HA Khalifa. Fuzzy portfolio optimization a quadratic programming approach. *Chaos, Solitons & Fractals*, 18(5):1045–1054, 2003.

[22] Daniel P Palomar and Yonina C Eldar. *Convex optimization in signal processing and communications*. Cambridge university press, 2010.

[23] Tor A Johansen, Thor I Fossen, and Svein P Berge. Constrained nonlinear control allocation with singularity avoidance using sequential quadratic programming. *Control Systems Technology, IEEE Transactions on*, 12(1):211–216, 2004.

[24] Tetsuya Fujie and Masakazu Kojima. Semidefinite programming relaxation for nonconvex quadratic programs. *Journal of Global Optimization*, 10(4):367–380, 1997.

[25] Marshall L Fisher. The Lagrangian relaxation method for solving integer programming problems. *Management science*, 50(12_supplement):1861–1871, 2004.

[26] Stephen Boyd and Lieven Vandenberghe. Semidefinite programming relaxations of nonconvex problems in control and combinatorial optimization. In *Communications, Computation, Control, and Signal Processing*, pages 279–287. Springer, 1997.

[27] Mark A Hall. *Correlation-based feature selection for machine learning*. PhD thesis, The University of Waikato, 1999.

[28] Pablo A Estévez, Michel Tesmer, Claudio A Perez, and Jacek M Zurada. Normalized mutual information feature selection. *Neural Networks, IEEE Transactions on*, 20(2):189–201, 2009.

[29] Yann LeCun, John S Denker, Sara A Solla, Richard E Howard, and Lawrence D Jackel. Optimal brain damage. In *NIPS*, volume 89, 1989.

[30] Petr Popov and Sergei Grudinin. Knowledge of native protein–protein interfaces is sufficient to construct predictive models for the selection of binding candidates. *Journal of chemical information and modeling*, 55(10):2242–2255, 2015.

[31] AM Katrutsa and VV Strijov. Stress test procedure for feature selection algorithms. *Chemometrics and Intelligent Laboratory Systems*, 142:172–183, 2015.

[32] Near infrared spectra of diesel fuels, bp50. http://www.eigenvector.com/data/SWRI/index.html.

[33] Ranjit Kumar Paul. Multicollinearity: Causes, effects and remedies. Technical report, Working paper, unknown date. Accessed Apr. 23, 2013, http://pb8.ru/7hy, 2006.

[34] Steven G Gilmour. The interpretation of mallows's c_p-statistic. *The Statistician*, pages 49–56, 1996.

[35] Allan DR McQuarrie and Chih-Ling Tsai. *Regression and time series model selection*. World Scientific, 1998.

[36] John Platt et al. Sequential minimal optimization: A fast algorithm for training support vector machines. 1998.

[37] Maxim V Shapovalov and Roland L Dunbrack. A smoothed backbone-dependent rotamer library for proteins derived from adaptive kernel density estimates and regressions. *Structure*, 19(6):844–858, 2011.

[38] B. O'Donoghue, E. Chu, N. Parikh, and S. Boyd. Conic optimization via operator splitting and homogeneous self-dual embedding. *Journal of Optimization Theory and Applications*, 2016.

[39] B. O'Donoghue, E. Chu, N. Parikh, and S. Boyd. SCS: Splitting conic solver, version 1.2.5. https://github.com/cvxgrp/scs, April 2016.

[40] Michael Grant and Stephen Boyd. CVX: Matlab software for disciplined convex programming, version 2.1. http://cvxr.com/cvx, March 2014.

[41] Michael Grant and Stephen Boyd. Graph implementations for nonsmooth convex programs. In V. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pages 95–110. Springer-Verlag Limited, 2008. http://stanford.edu/~boyd/graph_dcp.html.

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

[43] Richard B Lehoucq and Danny C Sorensen. Deflation techniques for an implicitly restarted arnoldi iteration. *SIAM Journal on Matrix Analysis and Applications*, 17(4):789–821, 1996.

[44] The MathWorks, Inc., Natick, Massachusetts, United States. *MATLAB and Optimization Toolbox, Global Optimization Toolbox Release 2014b*.

[45] Lenna X Peterson, Xuejiao Kang, and Daisuke Kihara. Assessment of protein side-chain conformation prediction methods in different residue environments. *Proteins: Structure, Function, and Bioinformatics*, 82(9):1971–1984, 2014.