J Mol Model (2015) 21: 261
DOI 10.1007/s00894-015-2806-y

ORIGINAL PAPER

Protein folding optimization based on 3D off-lattice model via an improved artificial bee colony algorithm

Bai Li[1,2] · Mu Lin[3] · Qiao Liu[2,4] · Ya Li[5] · Changjun Zhou[6]

Received: 18 June 2015 / Accepted: 30 August 2015 / Published online: 17 September 2015

Abstract Protein folding is a fundamental topic in molecular biology. Conventional experimental techniques for protein structure identification or protein folding recognition require strict laboratory requirements and heavy operating burdens, which have largely limited their applications. Alternatively, computer-aided techniques have been developed to optimize protein structures or to predict the protein folding process. In this paper, we utilize a 3D off-lattice model to describe the original protein folding scheme as a simplified energy-optimal numerical problem, where all types of amino acid residues are binarized into hydrophobic and hydrophilic ones. We apply a balance-evolution artificial bee colony (BE-ABC) algorithm as the minimization solver, which is featured by the adaptive adjustment of search intensity to cater for the varying needs during the entire optimization process. In this work, we establish a benchmark case set with 13 real protein sequences from the Protein Data Bank database and evaluate the convergence performance of BE-ABC algorithm through strict comparisons with several state-of-the-art ABC variants in short-term numerical experiments. Besides that, our obtained best-so-far protein structures are compared to the ones in comprehensive previous literature. This study also provides preliminary insights into how artificial intelligence techniques can be applied to reveal the dynamics of protein folding.

Keywords Artificial bee colony · Numerical optimization · Off-lattice model · Protein folding · Protein structure optimization

✉ Bai Li
libaioutstanding@163.com

1 College of Control Science and Engineering, Zhejiang University, Hangzhou 310027, China

2 School of Advanced Engineering, Beihang University, Beijing 100191, China

3 College of Biomedical Engineering and Instrument Science, Zhejiang University, Hangzhou 310027, China

4 School of Instrument Science and Opto-electronic Engineering, Beihang University, Beijing 100191, China

5 School of Mathematics and Systems Science & LMIB, Beihang University, Beijing 100191, China

6 Key Laboratory of Advanced Design and Intelligent Computing, Dalian University, Dalian 116622, China

Introduction

Proteins are large biological molecules or macromolecules consisting of amino acid residues that are connected by peptide bonds [1]. Understanding the functional mechanism of proteins, known as the building blocks of life, is essential in biological sciences and presents an enormous impact on enzyme engineering pathology, medicine, and pharmaceutics [2]. The folded structures of proteins in a cell provide the information necessary for studying the functions of proteins at the molecular level [3, 4].

Various experimental techniques have been developed to identify protein structures [5]. Up to February 25, 2013, approximately 81,700 protein structures had been deposited in the Protein Data Bank (PDB) database; most of these structures were determined using X-ray diffraction (88.8 %) and nuclear magnetic resonance spectroscopy (10.5 %) [6]. However, strict laboratory requirements and heavy operating burdens have limited the applications of the experimental techniques, making it difficult to keep track of deposited structures in the same pace of primary structures (i.e., amino-acid sequences) recovery [2, 3, 7]. In 2011, the number of experimentally determined secondary/tertiary structures was two-fold lower than the number of undetermined protein structures [8]. Hence, computation-based techniques have been used to optimize protein structures or to simulate protein folding processes.

Protein folding in computational biology involves calculating protein conformation by arranging a sequence of basic structural elements. Christian Anfinsen's theory, which states that all information needed to predict the native structure of a protein is encoded in its primary sequence, has largely contributed to the development in this research area [9, 10]. Researchers [11, 12] have used this theory as a basis to analyze native structures with minimal free energy. However, solving such a problem remains challenging for complicated protein models [13]. A key issue here is to what extent the trivial mechanisms in protein folding should be omitted [14]. Therefore, models with reduced complexity (e.g., [15–18]) have been developed by utilizing parts of the protein properties to avoid the large computational cost associated with an all-atom model.

In the present study, an off-lattice model introduced by Stillinger et al. [18] is utilized to simplify the protein folding scheme. All the amino acid residues in a sequence can be classified as hydrophobic or hydrophilic on the basis of the K-D method [19]. These "binarized" residues are connected by unbendable but free-to-rotate chemical bonds. Considering potentials among particles, we utilize a predefined criterion to calculate the free energy of a given conformation. Assuming that the native conformation of a protein is the most stable structure with minimal free energy, we can simulate the protein folding process through minimizing the free energy criterion.

The original protein folding scheme can be transformed into a numerical optimization problem by using the off-lattice model. Considering the NP-hard property of such optimization problems [20], researchers have developed various optimization techniques. For example, Kim et al. [13] proposed a conformational space annealing optimizer, Zhang et al. [21] a genetic tabu search algorithm, Chen and Huang [22] a heuristic algorithm, Wang et al. [23] a chaotic artificial bee colony algorithm, and Liang an annealing contour Monte Carlo algorithm, among many others [24–44]. However, previous research studies have focused overly on developing optimization algorithms, leaving the original protein structure folding mission ignored. In fact, large numbers of new optimization methods are being proposed every day, which outperform the relatively old ones. Therefore, the impact of an optimizer is commonly not anticipated to increase with time in the long run. In this sense, we believe that research efforts should be exerted on the concerned biological problem so as to make long-lasting contributions.

Previous publications in this broad research area suffer from three typical drawbacks from the authors' viewpoint. First is the lack of algorithm validation. Only the best-so-far protein structure solutions were reported without sufficient numerical experiments or theoretical analyses to support the proposed optimization algorithm. Second is that artificial protein sequences (e.g., Fibonacci sequences) or real sequences that are relatively short are considered in simulations. At this point, ref. [27] indicated that real sequence formations are far more realistic than those artificial ones, thus simulations on the Fibonacci sequences cannot fully verify the efficacy of the utilized optimization algorithm. Third is the lack of insights into the biological problem: the protein folding mission is regarded as a general numerical optimization problem and few in-depth analysis or discussions about the obtained results are attached. For the assistance of understanding, a list of the aforementioned references is shown in Fig. 1. In addition to the three common drawbacks mentioned above, the obtained solution vectors were seldom made public among the previous publications, rendering their calculated solutions impossible to validate.

The present study aims to overcome the typical limitations mentioned above. A balance-evolution artificial bee colony (BE-ABC) algorithm [31, 45] is applied for protein structure optimization. Compared with the conventional ABC algorithm and other state-of-the-art variants, BE-ABC algorithm utilizes parameters/variables in the algorithm framework as guidelines to adaptively adjust search intensity. Moreover, an overall degradation procedure is used to avoid premature convergence. A benchmark case set for protein structure optimization that focuses only on real protein sequences is systematically formulated. In addition to reporting the best-so-far optimization results, we also strictly compare the short-term convergence performance of BE-ABC algorithm to those of some state-of-the-art ABC variants. Also, innovative insights

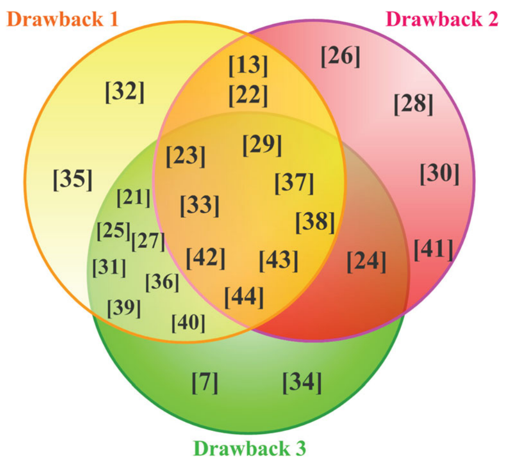

Fig. 1 Comprehensive review on the common drawbacks of publications regarding protein structure optimization using off-lattice models

into the protein folding problem are presented. The details of our optimized best-so-far structures are released for the convenience of validation.

The remainder of this paper is organized as follows: the next section introduces the 3D off-lattice protein model, then the principle of BE-ABC algorithm is briefly presented. Results are presented next followed by further discussions and conclusions.

Three-dimensional off-lattice protein model

The off-lattice protein model was initially developed to consider 2D folding problems and was extended to deal with 3D scenarios where additional torsional energy contributions of each bond are taken into account [14]. In the present study, a 3D off-lattice model is used to fold proteins in a relatively flexible but nontrivial extent.

The remainder of this section concerns how a structural protein can be uniquely determined by a set of bond/torsional angle parameters and how the corresponding free-energy value can be calculated.

Structure formulation

The off-lattice model is a coarse-grained one which holds the viewpoint that the main driving forces to form protein structures are the hydrophobic interactions among amino acid residues [18]. Thus, all 20 types of amino acid residues are classified as hydrophobic residues (amino acids I, V, L, P, C, M, A, and G represented by letter "A") and hydrophilic residues (amino acids D, E, F, H, K, N, Q, R, S, T, W, and Y represented by letter "B") [19, 25, 27]. All the AB-binarized amino acid residues are sequentially connected by unit-length chemical bonds to form a structural chain in the 3D space. This subsection introduces how the unique structure of a chain with N residues can be specified by $(2N-5)$ angle parameters $[\theta_1, \ldots, \theta_{N-2}, \beta_1, \ldots, \beta_{N-3}]_{1\times(2N-5)}$, which include $(N-2)$ bond angles and $(N-3)$ torsional angles.

Locations of amino acid residues in the space are determined as follows [23]:

$$(x_i, y_i, z_i) = \begin{cases} (0,0,0) & i=1 \\ (0,1,0) & i=2 \\ (\cos(\theta_1), \sin(\theta_1)+1, 0) & i=3 \\ \begin{pmatrix} x_{i-1} + \cos(\theta_{i-2})\cdot\cos(\beta_{i-2}), \\ y_{i-1} + \sin(\theta_{i-2})\cdot\cos(\beta_{i-2}), \\ z_{i-1} + \sin(\beta_{i-2}) \end{pmatrix} & 4 \le i \le N \end{cases}. \tag{1}$$

For normalization, the first three amino acid particles are defined in the plane $z=0$. The locations of subsequent particles (i.e., when $i\ge4$) can then be calculated recursively. Specifically, the location of the ith particle is based on that of the $(i-1)$th particle ($i\ge4$). Figure 2 shows how the location of a protein sequence AABA is determined by the angle vector $[\theta_1, \theta_2, \beta_1]$.

Free energy computation

Once a unique structure can be specified by a set of angular parameters, the corresponding free energy of the protein structure can be calculated.

In the 3D off-lattice model, the free energy function *Energy* can be defined as:

$$Energy([\theta_1, \ldots, \theta_{N-2}, \beta_1, \ldots, \beta_{N-3}]) = \sum_{i=1}^{N-2} \frac{1-\cos(\theta_i)}{4} + 4\sum_{i=1}^{N-2}\sum_{j=i+2}^{N} \left[r_{ij}^{-12} - C(\xi_i, \xi_j) r_{ij}^{-6}\right], \tag{2}$$

where ξi reflects the binary property of the ith particle in the sequence, r_{ij} denotes the distance between residues i and j in the 3D space, and $C(\xi i, \xi j)$ represents the interaction between these two particles. If particle i is hydrophilic, then $\xi_i=1$; otherwise, $\xi_i=-1$. r_{ij} and $C(\xi i, \xi j)$ can be defined as follows:

$$r_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}. \tag{3}$$

$$C(\xi_i, \xi_j) = \frac{1}{8}\left(1 + \xi_i + \xi_j + 5\xi_i\xi_j\right). \tag{4}$$

The free energy in Eq. (2) consists of two items that respectively represent intermolecular interactions among residues and intermolecular interactions between the peptide chain and surrounding solvent molecules. The first sum in Eq. (2) runs over the $(N-2)$ angles $\theta_i \in (-180^\circ, 180^\circ]$ of successive bond vectors. This item represents the bending energy and the coupling is ferromagnetic, i.e., it penalizes energy to bend the chain [31]. The second item partially competes with the bending barrier depending on the distance between nonadjacent particles along the chain [14].

One angle vector $[\theta_1, \ldots, \theta_{N-2}, \beta_1, \ldots, \beta_{N-3}]$ represents one candidate structure and $Energy([\theta_1, \ldots, \theta_{N-2}, \beta_1, \ldots, \beta_{N-3}])$ is the corresponding free energy value, with $\theta_i, \beta_j \in (-180^\circ, 180^\circ]$ for any $1\le i\le N-2$, $1\le j\le N-3$.

In this way, an protein folding mission is transformed into a constrained numerical optimization problem. Given a protein sequence, we aim to find the optimal angle vector associated with the minimal free energy value, i.e.,

$$[\theta_1, \ldots, \theta_{N-2}, \beta_1, \ldots, \beta_{N-3}]^* = \arg\left(\min_{-180^\circ < \theta_i, \beta_j < 180^\circ} (Energy([\theta_1, \ldots, \theta_{N-2}, \beta_1, \ldots, \beta_{N-3}]))\right),$$

$$1\le i\le N-2,\ 1\le j\le N-3.$$

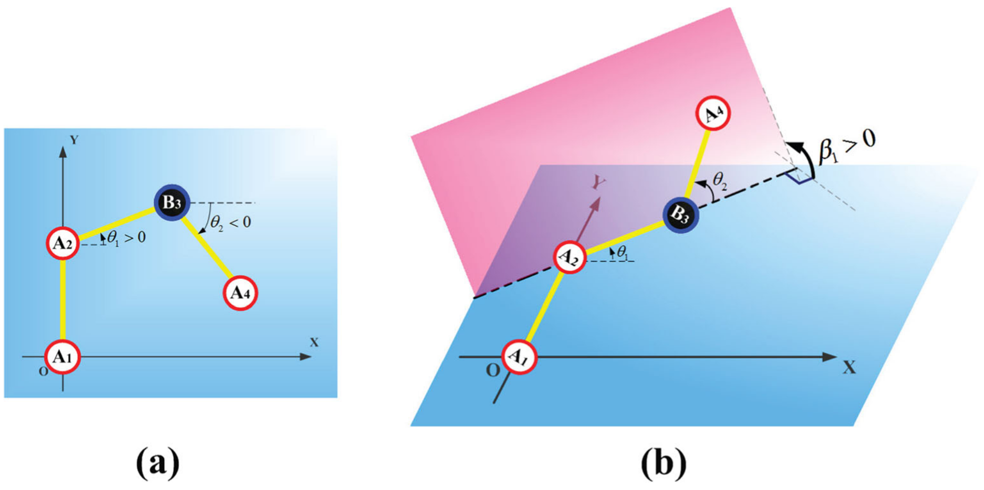

Fig. 2 Schematic of 3D off-lattice protein model for sequence AABA (N=4): **a** Illustration of how $[\theta_1,\theta_2]$ affects protein structure with β_1=0; **b** Illustration of how β_1 effects protein structure while $[\theta_1,\theta_2]$ is temporarily fixed

The following section introduces BE-ABC algorithm to search for that optimal solution.

BE-ABC algorithm

Consistent with Anfinsen's theories of protein structure stability when free energies of native protein structures are at their lowest, this section focuses on how BE-ABC algorithm searches for the solution with minimum free energy.

Motivations

The conventional ABC algorithm is a well-known swarm intelligence optimization method inspired by the foraging behavior of bees [46]. ABC algorithm is featured by the implementation of both local exploitation and global exploration procedures during the optimization process [47]. However, local search accuracy of the ABC algorithm is not satisfactory [48], causing large numbers of remedies proposed for that imperfection. Most of the proposed ABC variants have considered adjusting the local exploitation equations to achieve good search intensity. To that end, three ways have been prevailingly utilized: (i) to adopt a new principle from the outside world (e.g., [23, 49, 50]), (ii) to integrate ABC with other metaheuristics and (iii) to adjust the local search intensity in a self-adaptive manner (e.g., [7, 45, 51]). Commonly it is difficult to intuitively judge whether one ABC variant outperforms another unless statistically significant results based on comparative numerical experiments are available. Generally speaking, the absence of in-depth mathematical analyses may have made this research area as prosperous as it is. In contrast with the first two ways of modification, self-adaptive ABC variants (i.e., methods based on the third way) are intuitively understandable (because no complicated outside-world principles are involved) and remain the original framework of the conventional ABC algorithm. On the other hand, one may

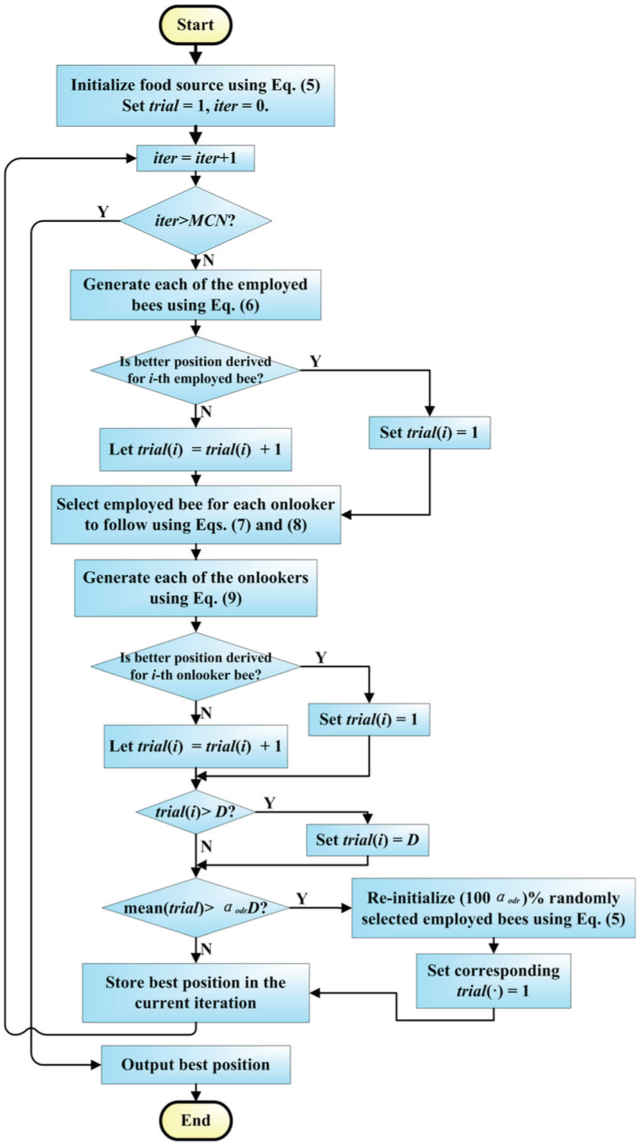
Start
Initialize food source using Eq. (5)
Set trial = 1, iter = 0.
iter = iter+1
iter>MCN?
Y
N
Generate each of the employed bees using Eq. (6)
Is better position derived for i-th employed bee?
Y
N
Let trial(i) = trial(i) + 1
Set trial(i) = 1
Select employed bee for each onlooker to follow using Eqs. (7) and (8)
Generate each of the onlookers using Eq. (9)
Is better position derived for i-th onlooker bee?
Y
N
Set trial(i) = 1
Let trial(i) = trial(i) + 1
trial(i)> D?
Y
N
Set trial(i) = D
mean(trial)> α_{odr}D?
Y
N
Re-initialize (100 α_{odr})% randomly selected employed bees using Eq. (5)
Set corresponding trial(·) = 1
Store best position in the current iteration
Output best position
End

Fig. 3 Flowchart of BE-ABC algorithm

notice that the effect to execute a local/global search may be different during different periods of the entire optimization process, making it sensible to strike an intensity balance between a global search and a local one.

Adjusting the search intensity adaptively is considered in BE-ABC algorithm. Specifically, when evolutions are made with ease around one employed bee, the subsequent search accuracy is intensified to expect further evolutions there. On the contrary, when the search efficiency is bad there, the subsequent search should be executed in a relatively wide scale. Through this, it is possible that a global search appears local and a local exploitation appears global. Then there is no longer any clear gap between the global and local searches during the evolutions. Since a balance between the global and local searches is ideally expected, we name this proposal “balance evolution”.

In more detail, BE-ABC algorithm is different from the conventional ABC algorithm in two aspects. First, adaptively changed multipliers are added in the global and global search equations to manipulate the search intensity. Those multipliers are determined on the basis of the optimization efficiency in a current iteration. Second, the conventional re-initialization procedure is improved by an overall degradation strategy, which aims to re-initialize employed bees simultaneously during one iteration (instead of re-initializing only one in each iteration). In BE-ABC algorithm, the optimization efficiency is evaluated through the unsuccessful trail attempts. In the next subsection, the principle of BE-ABC algorithm is presented.

Algorithm principle

BE-ABC algorithm involves three types of bees: unemployed, employed, and onlooker. Unemployed bees search in a fully random manner in space while employed bees form a cooperative group to search in a global manner. Onlooker bees follow “qualified” employed bees to exploit locally. A qualified employed bee is one that finds a high-quality nectar source [31]. The location of a nectar source resembles a candidate solution to the concerned free energy minimization problem and the nectar quantity is evaluated by the function *Energy*(·). Considering that the search space (wherein all nectar sources exist) is not infinite, we impose bounds on the solution vectors: let $\mathbf{X}=[\theta_1,\ldots,\theta_{N-2},\beta_1,\ldots,\beta_{N-3}]_{1\times(2N-5)}$ represent a general candidate solution, then $\mathbf{Lb}\leq\mathbf{X}\leq\mathbf{Ub}$, where **Ub** and **Lb** denote the upper/lower bound. In this current study, we have $\mathbf{Ub}=(180^\circ,\ldots,180^\circ)_{1\times(2N-5)}$ and $\mathbf{Lb}=(-180^\circ,\ldots,-180^\circ)_{1\times(2N-5)}$. To simplify the presentation, in the remaining of this paper, we refer to **X** as $(X^1,X^2,\cdots,X^{Dim})$, **Ub** as $(U^1,U^2,\cdots,U^{Dim})$ and **Lb** as $(L^1,L^2,\cdots,L^{Dim})$ when $Dim=2N-5$.

Table 1 Details of 13 benchmark amino acid sequences

PDB ID	Length	Sequences *
1CB3	13	BABBBAABBAAAB
1BXL	16	ABAABBAAAAABBABB
1EDP	17	ABABBAABBBAABBABA
2H3S	25	AABBAABBBBBABBBABAABBBBBB
2KGU	34	ABAABBAABABBABAABAABABABABABAAABBB
1TZ4	37	BABBABBAABBAAABBAABBAABABBBABAABBBBBB
1TZ5	37	AAABAABAABBABABBAABBBBAABBBABAABBABBB
1AGT	38	AAAABABABABABAABAABBAAABBABAABBBABABAB
1CRN	46	BBAAABAAABBBBBAABAAABABAAAABBBAAAAAAAABAAABBAB
1HVV	75	BAABBABBBBBBAABABBBABBABBABABAAAAABBBABAABBABBBABBAABBABBAABBBBBAABBBBBABBB
1GK4	84	ABABAABABBBBABBBABBABBBBAABAABBBBBAABABBBABBABBBAABBABBBBBAABABAAABABAABBBBAABABBBBA
1PCH	88	ABBBAAABBBAAABABAABAAABBABBBBBABAAABBBBABABBAABAAAAAABBABBABABABABBABBAABAABBBAABBAAABA
2EWH	98	AABABAAAAAAABBBAAAAAABAABAABBAABABAAABBBAAAABABAAABABBAAABAAABAAABAABBAABAAAAABAAABABBBABBAAABAABA

* An “A” in these sequences denotes a hydrophobic amino acid, and a “B” denotes a hydrophilic amino acid

A bee colony consists of *SN* bees, one half of which are unemployed bees and the other half onlooker bees. Before the optimization starts, those ${}^{SN}/_2$ unemployed bees are randomly initialized in the search space. The following equation shows how the *j*th element of the *i*th employed bee's location X_i is generated:

$$X_i^j \leftarrow L^j + rand(0,1)\cdot\left(U^j - L^j\right), \quad i = 1,2,\ldots,{}^{SN}/_2,\ j = 1,2,\ldots,Dim, \tag{5}$$

where $rand(0,1)$ denotes a random number (range of 0 to 1) obeying uniform distribution. After the initialization procedure, these unemployed bees directly become employed bees. Then an iterative optimization process gets started.

In each cycle of iteration, ${}^{SN}/_2$ employed bees try new positions within the search space. The new positions are generated by sharing locations with each other. For instance, the *i*th employed bee X_i is assumed at $(X_i^1, X_i^2, \cdots, X_i^{Dim})$ and the to-be-computed search location $\mathbf{X}_i^*$ is denoted by $(X_i^{*\,1}, X_i^{*\,2}, \cdots, X_i^{*\,Dim})$. First, as many as *trial*(*i*) out of all *Dim* dimensions in X_i are randomly selected. *trial*(*i*) is an integral scalar (range of 1 to *Dim*) associated with the *i*th employed bee, which is formally introduced below. Second, the element of each selected dimension is mutated through a crossover procedure. The following equation shows how the *i*th employed bee takes usage of the randomly chosen *k*th employed bee's location to update the *j*th dimension:

$$X^{*j}_i \leftarrow X_i^j + rand(-1,1)\cdot\left(X_k^j - X_i^j\right)\cdot\frac{trial(i)}{trial(i)+trial(k)}, \quad k\in\left\{1,2,\ldots,{}^{SN}/_2\right\},\ k\neq i. \tag{6}$$

When $\mathbf{X}_i^*$ is available, $Energy(\mathbf{X}_i^*)$ is immediately compared with $Energy(\mathbf{X}_i)$. When $Energy(\mathbf{X}_i^*) < Energy(\mathbf{X}_i)$, the *i*th employed bee flies to the new location $\mathbf{X}^*$, i.e., X_i is updated as $\mathbf{X}_i^*$. In addition, *trial*(*i*) should be reset to 1. When $Energy(\mathbf{X}_i^*) \geq Energy(\mathbf{X}_i)$, the *i*th employed bee remains at X_i but *trial*(*i*) adds one. Based on the principle by which the principle *trial*(*i*) is calculated, it is notable that *trial*(*i*) records the consecutive unsuccessful trial attempts of the *i*th employed bee.

When all the employed bees have updated their locations, onlooker bees act accordingly. A roulette selection procedure guides those ${}^{SN}/_2$ onlooker bees to choose qualified employed bees to search around. A probability index *P* is calculated according to Eqs. (7) and (8) to reflect the search quality of the employed bees [46].

$$P(i) = \frac{fitness(i)}{\sum_{j=1}^{SN/2} fitness(j)},\ i=1,2,\ldots,{}^{SN}/_2, \tag{7}$$

$$fitness(i) = \begin{cases} \dfrac{1}{1+Energy(\mathbf{X}_i)} & \text{if } Energy(\mathbf{X}_i) \geq 0 \\ 1 - Energy(\mathbf{X}_i) & \text{if } Energy(\mathbf{X}_i) < 0 \end{cases}. \tag{8}$$

Let's take the *i*th onlooker bee for example to see how its search location can be generated. In order to find an employed bee to follow around, a comparison is made between *rand*(0, 1) and *P*(1) first. When $P(1) \geq rand(0,1)$, the *i*th onlooker bee will search around the first employed bee; otherwise, an additional comparison between $rand(0,1)$ (another random number this time) and *P*(2) is performed. If all the *P*(*j*) $(j = 1,2,\ldots,{}^{SN}/_2)$ happen to be smaller than $rand(0,1)$, the process is repeated from the beginning *P*(1) again until one *P*(*j*) that satisfies $P(j) \geq rand(0,1)$ is finally found. That finally found *j*th employed bee is chosen by the *i*th onlooker bee to search around.

Contrary to the employed-bee phase, the crossover and mutation process involves only one (randomly chosen) dimension of the onlooker bee's solution vector. The following equation shows how the *i*th onlooker bee utilizes the *m*th

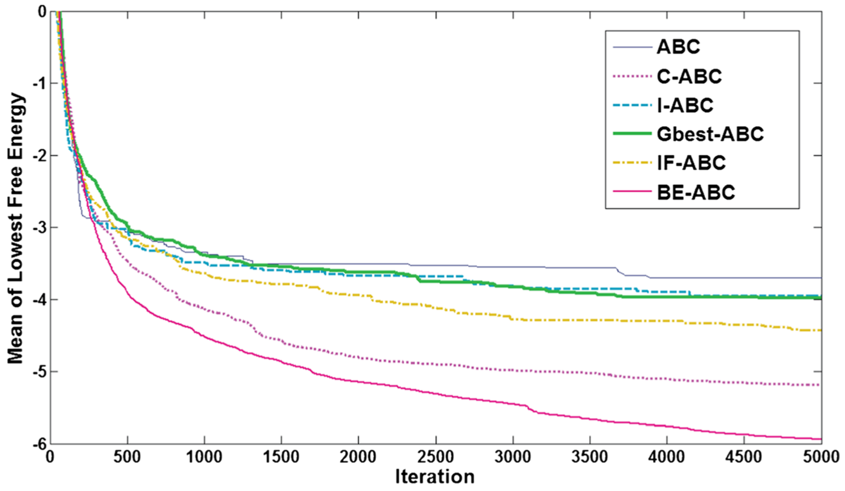

Fig. 4 Convergence curves of BE-ABC and other ABC variants when tested on case 1CB3 (*Dim*= 21, *SN*=40, and *MCN*=5000)

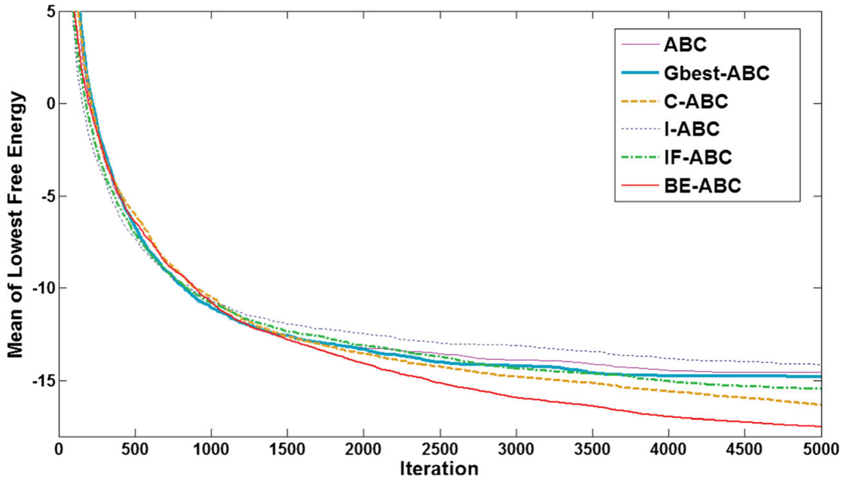

Fig. 5 Convergence curves of BE-ABC and other ABC variants when tested on case 1TZ5 (*Dim*=69, *SN*=40, and *MCN*=5000)

employed bee to generate a new search location around the *j*th in the *k*th dimension:

$$Y_j^k \leftarrow X_j^k + rand(-1,1)\cdot\left(X_m^k - X_j^k\right)\cdot\frac{trial(j)}{trial(j)+trial(m)}, \quad m\in\left\{1,2,\ldots,{}^{SN}\!/_2\right\},\ m\neq j. \tag{9}$$

When the location of the onlooker bee $\mathbf{Y}_i=(X_j^1,\ldots,X_j^{k-1},Y_j^k,X_j^{k+1},\ldots,X_j^{Dim})$ is determined, *Energy*($\mathbf{X}_j$) and *Energy*($\mathbf{Y}_i$) are compared. When *Energy*($\mathbf{Y}_i$)<*Energy*($\mathbf{X}_j$), the *j*th employed bee updates its current location $\mathbf{X}_j$ as $\mathbf{Y}_i$. Also, *trial*(*j*) is reset to 1. When *Energy*($\mathbf{X}_j$)<*Energy*($\mathbf{Y}_i$), *trial*(*j*) adds one. When all the onlooker bees have tried around their chosen employed bees, a re-initialization phase at the end of each iteration is followed.

Two things should be accomplished in the re-initialization phase. First, any *trial*(*i*) that exceeds *Dim* is set to *Dim*. Second, a criterion (i.e., $\frac{2}{SN}\sum_{i=1}^{SN/2} trial(i)$) is proposed to reflect the overall optimization efficiency and is compared with $\alpha_{odr}\cdot Dim$, where $\alpha_{odr}\in(0,1)$ is a user-specified threshold. When $\alpha_{odr}\cdot Dim < \frac{2}{SN}\sum_{i=1}^{SN/2} trial(i)$, the entire employed bee swarm is considered not working efficiently to a degree of α_{odr}. Then, $(100\alpha_{odr})$% randomly chosen employed bees directly become unemployed bees, which will turn to employed bees at the beginning of the next iteration. The corresponding *trial* indices are reset to 1. When $\alpha_{odr}\cdot Dim \geq \frac{2}{SN}\sum_{i=1}^{SN/2} trial\ (i)$, the current iteration cycle is directly terminated.

When the iteration number reaches a user-specified threshold *MCN*, the entire optimization process is accomplished. A flowchart of BE-ABC algorithm is depicted in Fig. 3.

Results

We systematically design a series of simulations on real protein sequences (Table 1) that originated from the PDB database. All simulations involved in this section were carried out in a Matlab R2011b environment and executed on an Intel Core 2 Duo CPU with 2 GB RAM running at 2.53 GHz under Windows XP. In this work, α_{odr} is set to 0.9.

In this section, first, the convergence performance of BE-ABC algorithm is evaluated by comparing it to several ABC variants in short-term numerical experiments. Second, our best-so-far solutions optimized by BE-ABC algorithm are

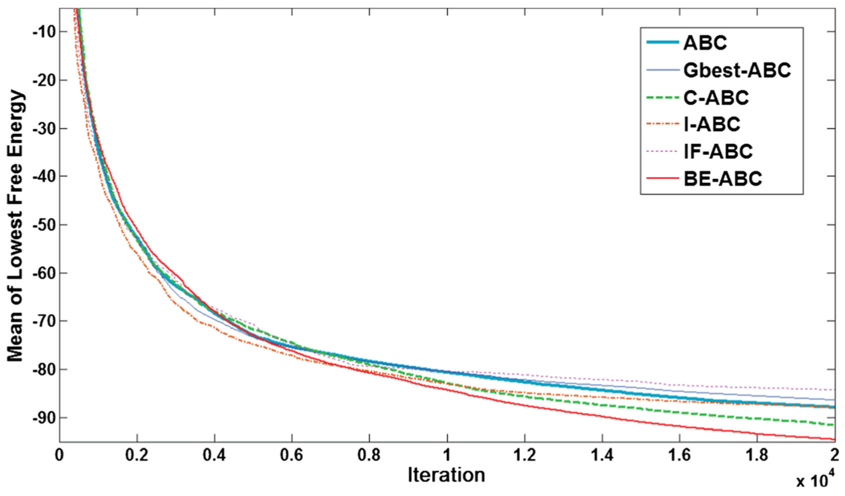

Fig. 6 Convergence curves of BE-ABC and other ABC variants when tested on case 2EWH (*Dim*=191, *SN*=40, and *MCN*=20000)

compared to the ones published in previous works. Third, the dynamic process of protein folding is preliminarily investigated through case studies.

Comparison of optimization performance among ABC variants

The convergence performance of BE-ABC algorithm is evaluated on all 13 real sequences. In order to show its advantage, BE-ABC algorithm was compared to the conventional ABC algorithm and its state-of-the-art variants, namely an improved ABC (I-ABC) [51], Gbest-guided ABC (Gbest-ABC) [48], chaotic ABC (C-ABC) [23], and internal feedback ABC (IF-ABC) [7]. In each of the 13 cases, the mentioned algorithms were tested repeatedly for 30 times under the condition that *SN*=40. *MCN* was set to 5000 for relatively short sequences and 20,000 for sequences with more than 85 amino acids. Figures 4, 5, and 6 depict the evolutionary curves in three representative cases (which represent short, medium and long sequences respectively). Results of all these cases are collected in Table 2.

Comparison of structures optimized by BE-ABC algorithm with previous literature

This subsection compares the best-so-far structures optimized by BE-ABC algorithm with the structures reported in previous publications. Table 3 lists the lowest free energy values for the concerned 13 sequences obtained using different optimization techniques. Figure 7 illustrates the structures obtained using our BE-ABC algorithm. Details of the optimized structures are provided in Table 4.

Folding process investigation

During the optimization process wherein BE-ABC searches for the lowest free energy, lower and lower values are sequentially obtained. Therefore, how the protein structure "migrates" during the optimization process can be observed, which may reveal critical insights into the protein folding process that occurs in nature.

Simulations were conducted on sequences 1BXL and 1AGT as examples. The results are depicted in Figs. 8 and 9.

Discussion

In this section, the simulation results shown in the preceding section are further analyzed.

Performance of BE-ABC algorithm

Through a comprehensive set of short-term numerical experiments, the optimization performance of BE-ABC algorithm

Table 2 Comparative optimization results (average, standard deviations, and time consumptions) obtained using BE-ABC and other ABC variants

PDB ID	ABC			Gbest-ABC			C-ABC			I-ABC			IF-ABC			BE-ABC		
	Mean	S.T.	CPU time (sec)	Mean	S.T.	CPU time (sec)	Mean	S.T.	CPU time (sec)	Mean	S.T.	CPU time (sec)	Mean	S.T.	CPU time (sec)	Mean	S.T.	CPU time (sec)
1CB3	−3.6990	0.3290	104.53	−3.9863	0.4827	102.06	−5.1849	0.8295	137.06	−3.9524	0.2110	103.94	−4.4273	0.5732	165.58	**−5.9417**	0.7821	119.86
1BXL	−9.1286	0.8272	155.72	−9.7543	0.4039	152.68	−11.0939	1.2277	210.62	−9.1779	1.0478	151.81	−10.2340	0.8418	231.33	**−11.6942**	1.1261	196.40
1EDP	−5.6460	0.6331	163.74	−5.8842	1.0872	161.24	−7.3751	1.3471	216.23	−5.3425	1.0685	164.92	−6.0956	0.6221	258.07	**−8.0500**	0.9330	203.00
2H3S	−6.4793	0.4651	314.49	−6.7934	0.6018	306.54	−8.6503	1.4119	408.98	−5.9213	0.5066	321.76	−7.4646	0.7863	486.95	**−10.4618**	1.1263	349.45
2KGU	−18.9879	1.8812	576.79	−18.3398	1.6560	580.76	−20.9629	2.3174	721.00	−17.6510	1.6699	584.91	−18.9293	1.8628	871.51	**−22.7195**	2.0087	691.49
1TZ4	−12.1997	1.3066	625.61	−11.8534	1.7940	627.55	−13.4762	1.9036	789.55	−11.6117	1.5066	611.10	−12.5083	1.8969	1028.60	**−14.9436**	2.2152	730.21
1TZ5	−14.5438	1.3189	649.23	−14.7945	1.5030	647.86	−16.2804	2.6254	811.19	−14.1122	2.1827	597.27	−15.4112	1.6454	1025.52	**−17.4859**	1.3702	769.74
1AGT	−22.2767	1.5450	724.11	−21.5642	1.9416	721.57	−24.9030	2.3336	869.73	−20.3744	2.5419	580.63	−22.7532	1.6269	1061.15	**−25.6024**	2.3415	826.32
1CRN	−37.3081	2.5401	1078.73	−37.7204	2.8849	1041.15	−40.0698	2.9916	1124.21	−37.2869	2.7380	788.46	−37.1658	2.1301	1511.06	**−42.3083**	2.9651	900.06
1HVV	−16.5131	2.4233	2448.23	−20.0426	1.2033	2212.39	−18.5579	2.3441	2652.70	−19.4625	1.6689	2379.26	−18.2009	1.5650	3343.61	**−21.5386**	3.5286	2322.83
1GK4	−22.4414	2.3180	2748.60	−24.4584	3.1366	2819.25	−25.6405	3.0218	3424.27	−24.6870	3.4688	2487.21	−24.2962	2.1232	4071.53	**−27.0410**	3.2358	2789.23
1PCH	−46.7577	1.7855	9676.22	−47.1605	1.3820	9704.07	−45.6608	1.5434	11816.69	−47.5994	2.5702	10029.80	−45.9589	3.7887	16376.52	**−51.6674**	3.4980	12007.39
2EWH	−87.9134	4.2843	10471.80	−86.3458	4.5785	10770.94	−91.6197	5.8237	12899.66	−88.0604	3.0621	10689.62	−84.1826	4.4643	19146.52	**−94.5785**	5.6967	12622.51

The bold value in each row denotes the winner in each case (i.e., the lowest free-energy value among all the six ones)

Table 3 Comparative results of the lowest free energy values for 13 sequences achieved by an improved particle swarm optimization (I-PSO) algorithm [25], a hybrid optimization that combines PSO, genetic algorithm (GA), and Tabu search (TS) named PGATS [35], a multiple-populations GA-PSO (MPGPSO) algorithm [37], ABC algorithm [36], GA-based TS (GATS) algorithm [21, 27], and our BE-ABC algorithm.

PDB ID	BE-ABC	I-PSO [25]	PGATS [35]	MPGPSO [37]	ABC [36]	GATS [27]	GATS [21]
1CB3	**−8.4580**	–	–	–	–	−8.2515	–
1BXL	**−15.9261**	–	–	–	–	–	−15.8246
1EDP	**−13.9276**	–	–	–	–	–	−13.7769
2H3S	**−18.3299**	–	–	–	–	−18.1640	–
2KGU	−28.1423	−20.9633	**−32.2599**	–	−31.9480	–	–
1TZ4	**−39.4901**	–	–	–	–	−39.3444	–
1TZ5	**−45.3233**	–	–	–	–	−45.3019	–
1AGT	**−51.8019**	–	–	–	–	–	−46.0842
1CRN	**−54.7253**	−28.7591	−49.6487	−43.9339	−52.3249	–	–
1HVV	**−47.4484**	–	–	–	–	–	–
1GK4	**−49.4871**	–	–	–	–	–	–
1PCH	**−91.3508**	−46.4964	−49.5729	−38.2766	−63.4272	–	–
2EWH	**−146.8231**	–	–	–	–	–	–

The bold values denote the best results (lowest free-energy value) for each case

is compared to other state-of-the-art ABC variants. The collected data in Table 2 indicate that BE-ABC algorithm outperforms the other ABC variants.

As depicted in Figures 4, 5, and 6, the evolution processes of BE-ABC algorithm do not remain the best among all six ABC variants in some early iterations. The origin of the

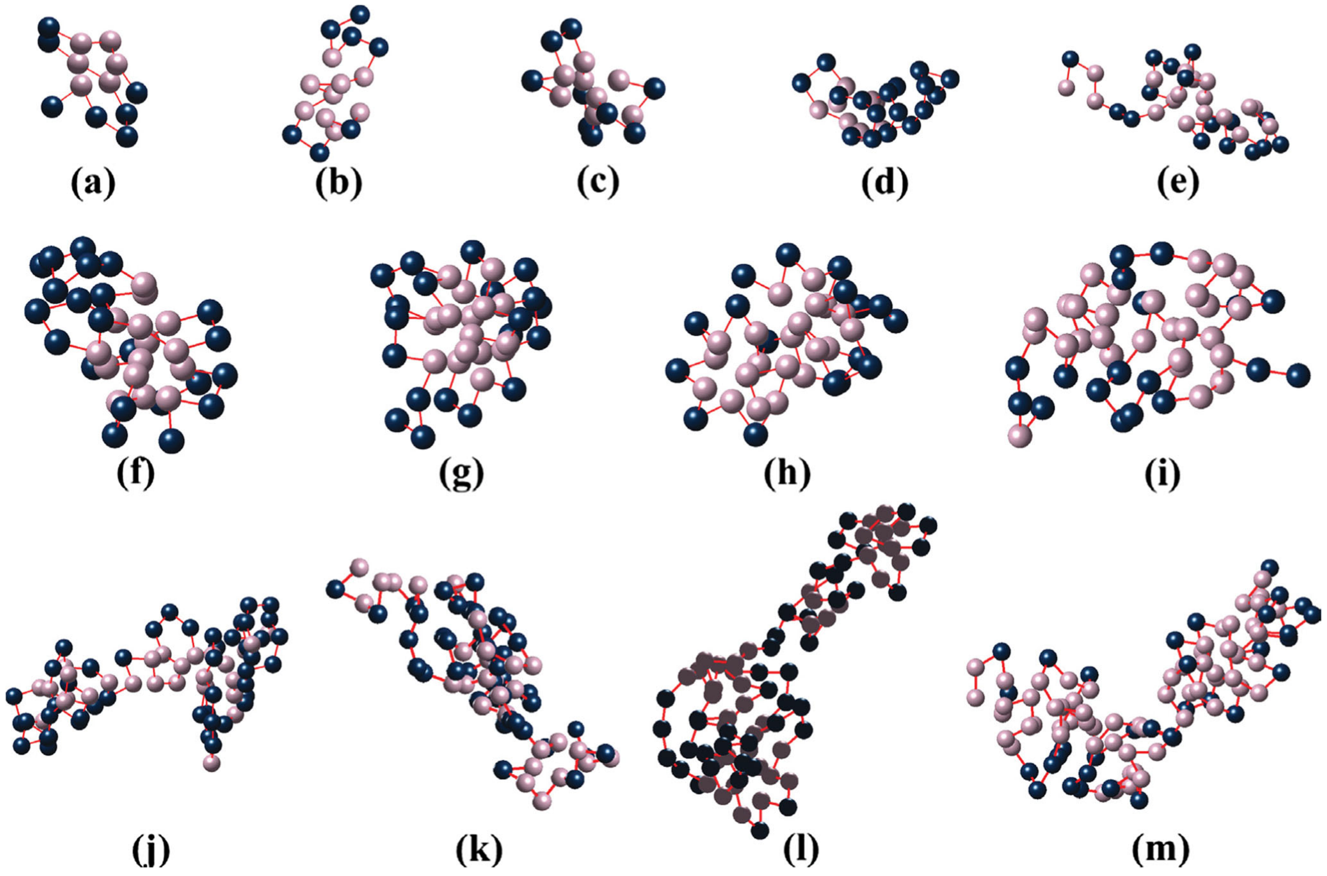

Fig. 7 Best-so-far conformation produced by BE-ABC algorithm for **a** 1CB3; **b** 1BXL; **c** 1EDP; **d** 2H3S; **e** 2KGU; **f** 1TZ4; **g** 1TZ5; **h** 1AGT; **i** 1CRN; **j** 1HVV; **k** 1GK4; (l) 1PCH and (m) 2EWH

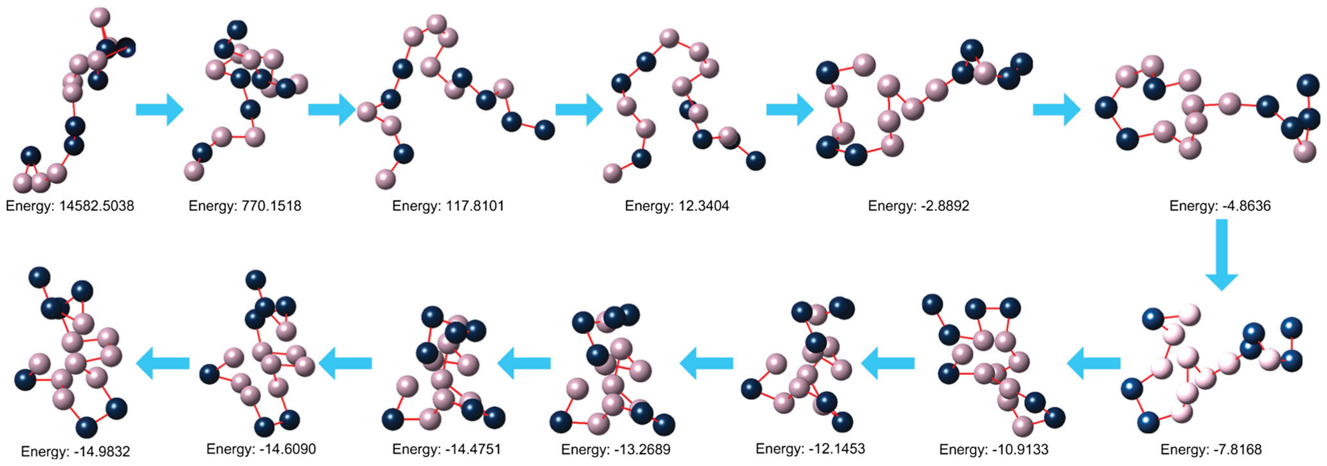

Fig. 8 Folding simulation on case 1BXL

problem might come from the following aspects of the algorithm. First, intuitively speaking, global search should be enhanced in the early iterations when evolutions are easy to make. During that period, elements in the *trial* vector are relatively small, yielding the global search scale of BE-ABC algorithm generally smaller than that of the conventional ABC algorithm under the same circumstance. In a simple instance with *trial*(i)=*trial*(k)=1 regarding Eq. (6), the search scale determined by BE-ABC algorithm shrinks to half of that determined by the conventional ABC algorithm. When *trial*(k) is larger, the shrink becomes severer, which is not good for a global search. Thus the evolution processes based on BE-ABC algorithm commonly lag at the very beginning. Due to the same reason, when making an evolution is not that easy, the benefit of utilizing gradually emerges and that is why BE-ABC algorithm usually catches up with its competitors in the later iterations. The second aspect concerns the overall degradation strategy. Unlike that in the conventional ABC algorithm, BE-ABC algorithm requires that part of the employed bees re-initialized all at once, which happens under some certain circumstance, which is not likely to happen in the early iterations. In contrast, ABC algorithm re-initializes the employed bees one after another when they are judged as inefficient. In other words, the re-initialization procedure in BE-ABC algorithm seldom works during the early iterations. This may cause differences among the evolution curves. However, re-initializing the employed bees one after another (in the conventional ABC algorithm) is not efficacious when evolutions are difficult to make or when *SN* is set large. The long-lasting convergence performance (based on efficient re-initialization ability) of BE-ABC algorithm is clearly illustrated, especially in Fig. 6.

Moreover, Table 2 shows that BE-ABC algorithm usually consumes less time than IF-ABC and C-ABC algorithms do. This is because BE-ABC algorithm neither adds complicated outside-world strategy nor changes the conventional algorithm framework.

Performance of the optimized best-so-far structures

The best solutions obtained through BE-ABC are compared with those presented in previous literature. The blanks in

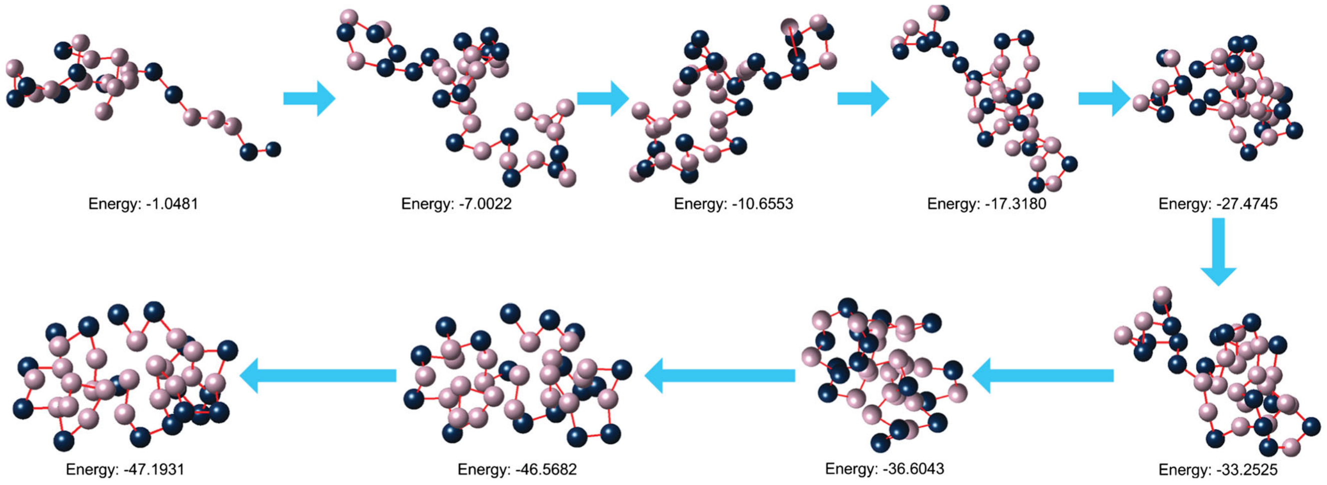

Fig. 9 Folding simulation on case 1AGT

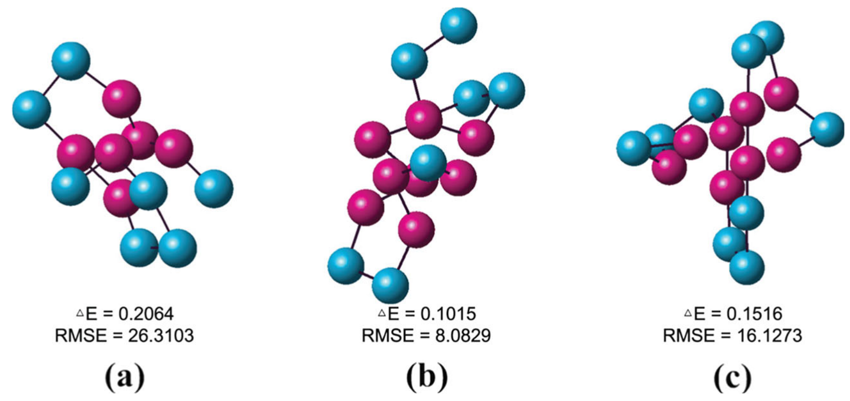

Fig. 10 Best-so-far structures optimized by GATS algorithm: **a** for case 1CB3; **b** for case 1BXL; **c** for case 1EDP. In each case, ΔE denotes the error between free energies obtained by GATS and BE-ABC. The *RMSE* index reflects how close two structures optimized by BE-ABC and GATS are

Table 3 indicate that more cases than those in previous studies have been considered in this present study. In fact, the comparative studies listed in Table 3 constitute all the relevant studies that can be found, to the best of our ability. In nearly all of the cases, BE-ABC algorithm clearly outperforms its competitors.

Here, curious readers may ask how the best solutions were computed. It is a "mystery" how the best solutions in most of the previous publications were computed. In fact, there is not a unified standard that all the publications had followed to generate the best solutions. In our work, we terminated the optimization process when there had been no evolution for up to 5,000,000 consecutive iterations.

Although in some cases our obtained free-energy values are slightly lower than previously published ones, the optimized structures are completely different. The best-so-far structures reported in refs. [21] and [27] for cases 1CB3, 1BXL, and 1EDP are depicted in Fig. 10. Also in that figure, an index ΔE is utilized to reflect the error between their reported free energy and ours; another root of mean square error index *RMSE* (defined in Eq. (10)) measures the difference between two concerned solution vectors.

$$RMSE(\mathbf{X},\mathbf{Y}) = \sqrt{\frac{1}{Dim}\left(\sum_{i=1}^{Dim}(X_i-Y_i)^2\right)}, \tag{10}$$

where $\mathbf{X}=[X_1,X_2,\ldots,X_{Dim}]$ and $\mathbf{Y}=[Y_1,Y_2,\ldots,Y_{Dim}]$.

Commonly in a natural protein structure, the hydrophobic residues tend to form a core with a minimum surface area that encounters water molecules. Thus such a hydrophobic core tends to be surrounded by hydrophilic residues so as to prevent encountering water molecules outside. In Fig. 7 (where the light-colored particles reflect hydrophobic residues while those dark-colored ones reflect hydrophilic residues), that characteristic appearing in nature can be reflected in nearly all of the simulated cases.

Performance of off-lattice model

In this research study, we have also investigated the protein folding process which is expressed on the basis of the off-lattice model. Assuming that the structure optimization process is the natural process for a protein structure to be folded, we observe that all the hydrophobic residues gradually assemble to become one central core surrounded by hydrophilic residues in cases 1BXL and 1AGT.

Although the off-lattice model roughly classifies all the amino acids in a sequence into two groups, some critical characteristics during the natural folding process can be observed in our artificial optimization process, possibly providing an initial guess for protein folding simulation with more complicated models.

Conclusions

In this paper, we have investigated protein folding optimization on the basis of a coarse-grained mode (i.e., the 3D off-lattice model) and an improved ABC algorithm. The underlying contributions of this paper are summarized as follows.

First, this work presents an application of a relatively new metaheuristic optimization technique. The optimization performance of BE-ABC algorithm is comprehensively evaluated by comparing it with several state-of-the-art ABC variants.

Second, 13 real protein sequences, rather than artificial sequences, are chosen for optimization. To the best of our knowledge, none of the previous works have conducted simulations in such a comprehensive manner. The simulated protein sequences and obtained optimization results form a benchmark case set, which can be used as a standard set by future researchers.

Last, the protein folding process is investigated on the basis of the off-lattice model through a viewpoint of evolution. Previous studies used their concerned optimization techniques to solve a numerical problem but quite few of those studies further investigated the efficiency of the off-lattice model. Simulation results in this study indicate that the off-lattice model provides a rough estimation of the folding process of proteins.

Investigations about how the optimized results benefit the area of molecular biology should be made through future studies. Effects are also needed to measure how close the optimized structures are to the existing ones in the PDB database and to reveal more details about the protein folding process.

Acknowledgments The authors are sincerely grateful to the anonymous reviewers, Mr. Xuebin Ji, Mr. Ligang Gong, Dr. Haibin Duan, Dr. Chengdong Pu, Prof. I-Fang Chung, Prof. Raymond Chiong, and Prof. Jaap Heringa for their comments and suggestions. This work is sponsored by the Fundamental Research Funds for the Central Universities under Grant YWF-14-SXXY-006 and Grant YWF-15-SXXY-004, the 6th National College Students' Innovation & Entrepreneurial Training Program under Grant 201210006050 as well as the National Natural Science Foundation of China under Grant 61370005.

Appendix

Table 4 Best-so-far solution optimized by BE-ABC algorithm

PDB ID	Solution vector
1CB3	−14.0618, 24.7751, −37.4574, −10.2442, 20.8918, 14.3826, 0.0344, 22.7245, 71.7223, −26.6440, −4.9327, −21.2546, −149.5309, 171.7070, 176.4815, 176.4340, 89.9305, 3.0125, −32.8040, 28.5409, 2.4127
1BXL	−23.9894, −102.7314, −3.2272, −5.4584, 0.0727, 28.2955, 45.9967, −2.5259, 60.9520, 67.4231, 161.0594, 67.5692, 0.0288, 27.4589, −53.3614, −60.4171, 26.3909, 113.2205, 152.2952, 37.0642, 167.7646, −169.6191, 136.4026, 177.8383, −17.0115, 87.2334, 155.4952
1EDP	−22.9111, 18.2597, 79.1534, 35.3602, −8.2222, −8.6311, 22.0229, 25.7423, −0.2752, 0.1581, 39.3589, −26.6495, 36.4065, −80.1681, 89.5273, 38.9650, 145.9329, −141.4454, −84.7010, −76.9813, −126.1700, 136.2198, 88.6941, 95.9751, 145.3596, 35.9802, −27.0676, −29.0454, −37.9873
2H3S	56.3676, −27.8037, 62.0377, 0.5187, 1.0712, 94.4369, 63.5028, 13.0689, −82.9610, −24.1770, −74.1721, −8.2340, 46.0441, −54.2720, −33.8035, −34.7444, 36.7134, −62.5235, −24.3748, 18.6261, 41.7080, −81.0016, −25.0891, 5.0992, 171.9440, 133.3680, 84.1103, 140.9098, 133.4524, −134.8044, −133.4496, −66.8584, 32.2283, 50.2538, −39.0348, −121.7507, −25.4591, −139.5654, 158.7352, 26.0329, 57.6732, 48.8666, −51.6648, −8.2943, −124.7090
2KGU	−21.8634, −82.1474, −20.4767, −31.8675, 19.7468, 45.8982, −13.9909, 83.0446, 96.7560, −7.8566, −25.2354, 103.6728, −101.1429, 80.5692, −101.6640, 5.5789, −85.8154, −62.8431, 12.2646, −46.3197, 94.6380, −7.0008, −27.7652, −65.9529, 56.7320, −14.0602, 85.2909, −27.0873, 0.1473, −30.9119, −42.7890, 35.0373, 13.1556, 28.7092, 52.1369, 34.4259, −32.7349, −151.3791, 70.2779, −35.3470, −45.5690, 28.6629, 28.1801, 33.9151, 153.1775, 118.6225, −36.0729, −65.1601, −24.4103, 156.0928, 33.2109, 22.1670, −6.4856, 107.0941, −1.5942, 48.9682, −26.7827, 36.6497, 161.8515, −87.2386, −55.3366, −14.6302, −176.6925
1TZ4	−14.7019, 64.8802, −16.5290, −29.9987, 22.4011, −1.1015, 21.1064, 79.4800, −117.8403, −112.5468, 73.6780, −3.6188, −31.2753, 20.4495, 6.9570, 25.9656, 9.6939, −79.9026, 12.0103, −37.0138, −9.7200, 81.3474, 162.9115, 116.2320, 11.2949, −29.7690, 109.2367, −1.2398, 0.9434, −31.3814, 16.1091, 30.7287, −87.6773, −10.3117, 17.4046, −30.3043, −170.3973, −66.1578, −161.4577, 113.4954, 130.5740, 148.5034, 120.3430, −154.0686, −60.0036, −141.5332, −109.0243, −15.0149, 69.8330, 55.6446, 8.9491, 40.5354, −177.8266, 171.7074, 155.2177, −130.3283, 11.2923, −23.7099, −2.7977, −20.9224, −41.0206, 11.3730, 91.8050, 150.6533, 121.9906, −158.9056, −148.1879, −101.5477, 2.9887
1TZ5	−21.5509, −50.6901, −19.9966, 78.0566, −53.4513, −106.9051, −174.9670, −133.1228, −27.9993, 38.3999, −54.8007, −141.8863, 4.5882, −47.9745, 33.4699, −115.0175, 48.8074, 20.5984, −6.3313, 33.0176, 65.4813, 34.2962, 178.3703, 38.0308, −25.9016, −17.7960, 61.1886, −3.8905, −64.6424, 11.4757, −1.1819, 11.9522, 35.3380, 1.5420, −43.5413, 55.2521, 173.3674, 48.4508, 173.2151, −58.1362, −160.4915, −174.3564, −82.2771, −174.7100, −57.7204, 26.4959, −107.9280, 175.4451, 70.0877, −173.8045, 10.4306, 43.0849, 110.4102, −154.1437, −161.6427, 178.4970, 21.7039, 115.1081, 30.5950, −11.2696, 171.0239, −53.3141, −154.9870, 176.5773, −93.0401, −10.8202, −116.9664, −83.6263, 172.1319
1AGT	−2.3116, −132.9355, −152.9745, 116.2728, −167.8692, 36.3193, −62.0157, 5.2481, 117.0098, −69.7927, 44.8072, −44.9432, −94.6170, 62.4969, −40.0646, −86.2814, −6.3002, −1.2781, −9.5921, 140.6445, −79.9387, −80.5554, −3.9163, −78.2688, 89.3200, −21.3175, −0.9789, 7.4211, 5.1962, 135.7684, 2.8577, 48.7351, 28.6067, −67.0007, 58.9994, −42.7972, −53.5891, −47.1174, 30.9531, −2.6463, 69.1991, −163.3194, 2.2289, 45.7841, 20.5006, 22.4734, −58.3811, −72.5963, −23.9633, 46.2442, −33.4880, −30.9297, 57.2125, 142.8207, 42.3729, 13.2871, −5.9342, 29.4574, 158.4325, 50.0175, −133.9704, −47.8558, −14.5316, −66.0909, −5.3917, 151.4913, 35.8287, 152.2362, −173.6238, −138.2699, 172.5047
1CRN	74.6273, 2.8197, 159.4046, 91.7786, −14.3079, −7.4513, 4.3412, 142.2140, 166.4528, 30.6291, 71.2766, −19.6545, −28.1067, −1.9392, 29.1772, 59.4027, −96.1102, −44.7633, −47.9253, 61.3879, −19.8284, −51.2245, −21.6517, −41.5195, 54.1117, −100.4569, −66.7062, −4.0724, 31.3876, 51.7816, −13.2873, 55.6203, −31.8552, 18.6697, −100.6095, 50.6588, −12.4137, −3.7499, −106.7324, 15.1948, −29.6233, 14.9463, 10.0778, −22.7216, −87.1945,

Table 4 (continued)

PDB ID	Solution vector
	18.8110, −23.0831, 27.3022, 68.4818, −178.3041, −6.3629, −13.3074, −55.6663, 42.4080, −14.7205, 23.9507, 31.3978, −47.5298, −166.6089, 2.0765, 39.2503, −14.3898, 10.3680, −94.5755, 152.5378, 123.2597, 36.1458, −32.3986, −171.6447, 175.2720, −78.1063, −68.8631, −8.0533, −170.4521, −157.7137, −70.8431, 35.3547, −177.0941, −117.2142, −160.2187, 84.3092, 148.4590, −66.2448, −130.8496, −168.2827, −100.1141, 6.0774
1HVV	80.6908, −50.5238, 30.4070, −51.0668, 25.2822, 6.1210, −50.8016, 0.4894, 68.6584, 14.2372, −59.3433, −172.4218, 15.0704, 65.2936, 159.7749, 26.0839, −100.6799, 28.5089, −1.6451, −6.8137, 22.8042, −90.4714, 15.4533, 57.6816, −9.7989, −49.6702, 35.6463, −76.7221, 25.5995, −51.9401, −26.5090, 62.4642, 33.4776, 161.0594, −117.9189, −154.3733, −39.2689, 36.3216, −68.0224, 11.1980, 17.5177, −30.3917, 35.7225, 12.9786, 19.5005, 3.5806, 20.7122, −132.8711, 7.2817, −167.2517, 18.8807, −78.0814, −14.4544, −34.9323, 11.7969, −57.2538, 21.1423, 54.0009, 3.5195, −54.4489, 2.1415, 38.4551, −8.4623, 144.5103, 24.5351, 33.2396, −6.4086, −78.8046, 174.9484, 1.5930, 28.1424, 9.5056, −54.1250, 170.7820, −174.8209, −33.8289, 177.1698, 97.0941, 166.9026, 55.7014, 175.9118, 67.7849, 58.1494, 174.5629, −83.8023, 162.5864, −58.8373, −73.0339, 155.9237, −29.3007, 67.6176, −16.2332, −107.3964, 23.8396, 110.4114, 162.7966, −97.6016, 14.6544, 4.8662, −1.1012, −123.5701, −36.9214, 74.8656, −24.7145, −3.4225, −162.2034, 12.4995, −9.8740, −26.4676, 152.7539, 117.5263, 74.2509, 162.1538, −117.3443, 154.0370, −137.8483, −117.6415, −104.3963, −1.0420, 121.8787, 9.0743, −114.3532, −133.1037, −54.1915, −41.0200, 61.4201, 128.1850, 111.7726, 83.7646, 129.3614, 85.8994, −12.5094, 8.1339, −24.0536, −94.8778, −2.5177, 108.1601, 53.6906, 149.2220, 5.0216, 23.1824, 88.4716, −15.7221, −30.1273, −36.9809
1GK4	−21.6894, −14.3276, 48.8168, −45.4543, −1.0648, −124.2028, 37.9826, 66.3801, 22.9687, 5.5179, −27.2564, 65.3296, −153.3232, −16.5857, −8.6788, −91.1073, 16.0011, 12.8929, 30.0995, 2.2852, −63.3065, 39.8721, 9.3686, 27.5157, −0.3128, −20.2697, −45.9781, 6.7297, 4.4989, −6.0233, −96.5715, 13.9484, −90.3138, −62.4654, 44.0272, 102.3733, −66.8061, −23.1234, 6.7663, 30.5414, 3.7864, 3.2423, 0.5144, 12.4356, 6.1785, −12.1507, −7.4363, −29.1272, −7.1087, 87.1612, −10.9189, −6.6633, 11.7076, −55.8428, 24.0808, −27.1981, 0.4037, −72.7388, 111.1732, 72.3373, −53.7392, −19.2691, 2.6709, 81.1208, −40.6180, −133.2265, −42.1346, 7.5862, −58.8290, 8.9371, 22.2074, −33.3028, 13.8379, −13.5482, −86.2635, −13.8697, 43.4693, 55.3598, 121.5120, 16.8944, −10.9632, 36.8455, −54.7074, 172.6659, 74.5327, 0.8122, −48.8131, −27.8977, 30.2474, −24.9637, 77.6013, 19.3923, −115.1565, −13.5718, 102.2621, 44.3140, −30.3100, −139.5109, −158.2457, −49.1128, −124.6835, 1.2227, −19.8960, −75.0098, 5.8541, 89.1655, −23.1566, 57.9790, 152.2665, 178.7988, 73.1711, −163.6388, 44.4483, 122.8186, 138.2718, 112.6628, −142.4871, 66.0475, 143.8586, 76.5748, 17.5478, −86.2162, −56.7060, −91.1800, 13.2051, 72.4476, 124.0226, 105.3506, 146.2343, 103.7664, 174.8075, −54.8102, 50.7264, 109.2054, 170.0172, −30.8748, 59.1876, 36.3606, 58.2695, 61.7742, −47.1014, −155.1102, −32.8831, 43.3692, −161.7924, −59.2082, 26.8005, −28.3700, 80.4676, 143.4945, 102.1201, 156.2368, 171.1537, 76.1097, 78.0965, 4.7715, −70.8766, 34.2546, −69.5999, −25.6245, 0.0602, 103.6680, 124.5616
1PCH	43.5603, −0.5581, 51.0269, 71.5241, −22.8763, −165.5940, 69.4043, −40.8120, −144.3618, 6.0212, 4.9260, 141.2707, 29.9156, −45.9807, −76.6985, 13.2083, 11.0730, −46.6472, −169.2636, 109.4209, −80.6196, −21.3525, 56.0178, 2.7469, −24.9770, 18.8028, −92.8137, −167.1236, −33.6219, 43.3143, 89.5431, 18.6533, −56.8690, 51.3334, 8.6419, −0.3859, −61.6817, 61.0599, −18.0811, 37.7962, −136.6101, 16.2508, 31.1677, 30.5831, 27.6785, 83.5453, −127.2399, −48.2667, −70.0013, 58.6984, −62.6201, −81.5554, −44.2539, −50.3361, 20.6317, 4.2708, −112.4258, −30.9974, 23.5624, 42.3764, 42.1948, 13.3415, 2.5809, −74.9152, −164.8953, −0.8178, 21.5609, 22.5203, 8.5765, 109.2869, 29.3634, 2.1225, 12.4136, −120.3413, −6.1064, 14.6941, 50.0448, 28.3427, −54.4405, 38.3312, −117.3602, 56.3906, 9.8377, −109.8487, −99.7961, −135.9576, −120.7264, 28.1106, 160.7656, 31.9478, −134.0017, −39.1912, −124.0489, −12.1312, −120.0252, 167.0531, −79.3470, 29.8425, 108.8634, 34.5249, 175.9989, −22.5645, 132.2016, 2.7658, −30.3842, −66.1483, 29.3910, 31.0837, 145.7338, 172.0597, 21.1105, 148.7510, −178.3366, 54.9076, 125.5836, −14.3095, −133.9550, −100.7464, −72.4418, −159.6307, −27.8017, 179.2643, 52.3683, 21.1497, 66.1417, −11.4370, 47.2959, 26.2722, 170.4601, 39.8690, −52.3557, 60.4005, −3.9348, 154.2336, 171.8271, 156.5632, 145.9230, 2.8887, −2.3310, −53.9904, 33.5704, 154.8804, 39.9316, 34.0064, −77.7366, 25.3800, −19.0780, −0.2700, 73.7883, 14.3260, 11.3731, −52.1158, −23.9903, 83.6962, 141.4928, −166.4798, 23.4281, 36.1021, −4.6947, −10.3817, 2.6026, −47.5569, −179.6645, 23.5649, 46.0917, 0.9257, −41.4373
2EWH	31.0281, −83.1940, −73.7166, 73.6313, 79.8774, −162.3809, −85.0653, 23.5845, −152.6138, −52.6713, −54.7878, 113.9021, 59.4182, 74.2415, −90.7065, −30.0839, −71.6468, −10.2363, −59.7657, −109.0952, −47.0266, −23.4636, 93.7763, −120.4693, 63.9653, −94.9731, 58.5368, 73.2819, −15.1595, −85.8182, 39.1232, 30.5319, 40.5315, −7.9589, −129.9194, −104.0965, −142.5123, −18.3873, −9.3663, 14.6949, 45.0905, −92.2574, 6.3949, 33.4429, −81.4748, 84.4697, 118.1556, 1.5611, 41.7624, 162.2825, −1.3615, −21.1779, 53.7514, 127.2352, −107.9389, 39.6852, 4.1324, −101.6262, 121.7615, 108.6972, −9.2238, −70.4706, −29.4339, −101.0204, 161.7725, 149.1204, −106.1018, 19.9872, 75.5759, 118.9055, 101.8863, −12.5656, 27.2378, −94.5046, −81.4401, −30.0655, 42.8940, 158.0247, 158.9893, 42.4068, −6.7307, 110.7461, 7.9615, −44.3511, −142.4752, 177.3489, 63.5718, 55.8396, −159.4334, −0.0666, 51.8864, 57.5426, −106.9189, −103.3725, −73.2516, 11.2726, −15.2303, −65.1614, 134.1459, 75.7639, −159.2708, −25.4270, 97.6577, −27.4364, −70.8047, −3.5138, 131.0074, 14.2295, 2.0808, −157.3447, −10.6784, 109.1077, 161.5699, 174.5464, −143.9630, 0.5748, −39.6871, −60.6532, −3.2241, 172.6112, −20.8339, −73.3347, 15.3115, 40.8376, 6.7255, 40.9140, −36.2423, 77.4473, 148.0497, 4.7864, −18.5122, −46.9985, −72.7250, −38.1795, −65.8628, 43.1726, 66.3916, 130.9052, 17.1987, 22.3094, 45.3919, −13.2915, −11.7787, 45.2203, 0.1654, 75.9715, 45.0252, 42.7999, 7.7206, 126.6125, 47.8645, −57.7001, −25.2384, −26.3449, −23.4756, −4.9972, 11.5920,

Table 4 (continued)

PDB ID	Solution vector
	−15.6547, −28.3669, 45.5674, −21.0754, −53.8507, −28.6844, 38.3740, −13.4775, −2.9792, 22.6226, 74.4558, 35.4358, 15.6760, −11.6839, 3.1357, −30.2239, −44.6561, 3.6959, −11.8332, −9.8726, 32.7796, 65.3137, 5.1472, −2.1037, 49.5231, −41.2739, −43.3171, −119.9027, 1.0740, −64.1879, −2.9169, 11.8586, −38.7253, 148.4466

References

1. Dehzangi A, Lyons J, Sharma A, Sattar A (2014) A segmentation-based method to extract structural and evolutionary features for protein fold recognition. IEEE/ACM Trans Comput Biol Bioinf 11(3):510–519
2. Sahu SS, Panda G (2010) A novel feature representation method based on Chou's pseudo amino acid composition for protein structural class prediction. Comput Biol Chem 34(5–6):320–327
3. Hendy H, Khalifa W, Roushdy M, Salem AB (2015) A study of intelligent techniques for protein secondary structure prediction. Int J Inf Mod Anal 4(1):3–12
4. Xu Y, Xu D, Liang J (2007) Computational methods for protein structure prediction and modeling. Vol 1: Basic characterization. Biol Med Phys Biomed Eng. Springer, New York
5. Comellas G, Rienstra CM (2013) Protein structure determination by magic-angle spinning solid-state NMR, and insights into the formation, structure, and stability of amyloid fibrils. Ann Rev Biophys 42:515–536
6. Bagaria A, Jaravine V, Güntert P (2013) Estimating structure quality trends in the protein data bank by equivalent resolution. Comput Biol Chem 46:8–15
7. Li B, Li Y, Gong LG (2014) Protein secondary structure optimization using an improved artificial bee colony algorithm based on off-lattice model. Eng Appl Artif Intell 27:70–79
8. Poole RK (2011) Globins and other nitric oxide-reactive proteins. Academic, San Diego
9. Anfinsen CB (1973) Principles that govern the folding of protein chains. Science 181(4096):223–230
10. Wooley JC and Ye Y (2007) A historical perspective and overview of protein structure prediction. Comput Meth Pro Struct Predic Model. Springer, New York, pp 1–43
11. Gibson KD, Scheraga HA (1967) Minimization of polypeptide energy. I. Preliminary structures of bovine pancreatic ribonuclease Speptide. Proc Natl Acad Sci U S A 58(2):420–427
12. Scott RA, Vanderkooi G, Tuttle RW, Shames PM, Scheraga HA (1967) Minimization of polypeptide energy, iii. Application of a rapid energy minimization technique to the calculation of preliminary structures of gramicidins. Proc Natl Acad Sci U S A 58(6): 2204–2211
13. Kim SY, Lee SB and Lee J (2005) Structure optimization by conformational space annealing in an off-lattice protein model. Phys Rev E 72:011916
14. Bachmann M, Arkın H, and Janke W (2005) Multicanonical study of coarse-grained off-lattice models for folding heteropolymers. Phys Rev E 71:031906
15. Bryant SH, Lawrence CE (1993) An empirical energy function for threading protein sequence through the folding motif. Pro: Struct Func Bioinf 16(1):92–112
16. Levitt M, Warshel A (1975) Computer simulation of protein folding. Nature 253(5494):694–698
17. Dill KA (1985) Theory for the folding and stability of globular proteins. Biochemistry 24(6):1501–1509
18. Stillinger FH, Head-Gordon T, Hirshfeld CL (1993) Toy model for protein folding. Phys Rev E 48(2):1469–1477
19. Mount DW, Mount DW (2001). Bioinformatics: sequence and genome analysis (vol 2). Cold Spring Harbor Laboratory Press, New York
20. Unger R, Moult J (1993) Finding the lowest free energy conformation of a protein is an NP-hard problem: proof and implications. B Math Biol 55(6):1183–1198
21. Zhang X, Wang T, Luo H, Yang JY, Deng Y, Tang J, Yang MQ (2010) 3D Protein structure prediction with genetic tabu search algorithm. BMC Syst Biol 4(1):S6
22. Chen M, Huang WQ (2006) Heuristic algorithm for off-lattice protein folding problem. J Zhejiang Uni SCI B 7(1):7–12
23. Wang Y, Guo GD, Chen LF (2013) Chaotic artificial bee colony algorithm: a new approach to the problem of minimization of energy of the 3D protein structure. Mol Biol 47(6):894–900
24. Liang F (2004) Annealing contour monte Carlo algorithm for structure optimization in an off-lattice protein model. J Chem Phys 120(14):6756–6763
25. Chen X, Lv M, Zhao L and Zhang X (2011) An improved particle swarm optimization for protein folding prediction. Int J Inform Eng Electron Bus 3:1
26. Kim J, Straub JE and Keyes T (2007) Structure optimization and folding mechanisms of off-lattice protein models using statistical temperature molecular dynamics simulation: statistical temperature annealing. Phys Rev E 76: e011913
27. Wang T, Zhang X (2011) A case study of 3D protein structure prediction with genetic algorithm and Tabu search. Wuhan Uni J Nat Sci 16(2):125–129
28. Kalegari DH, Lopes HS (2010) A differential evolution approach for protein structure optimisation using a 2D off-lattice model. Int J Bio-Inspired Comput 2(3):242–250
29. Hsu HP, Mehra V, Grassberger P (2003) Structure optimization in an off-lattice protein model. Phys Rev 68(3):e037703
30. Irbäck A, Peterson C, Potthast F, Sommelius O (1997) Local interactions and protein folding: a three-dimensional off-lattice approach. J Chem Phys 107(1):273–282
31. Li B, Chiong R, Lin M (2015) A balance-evolution artificial bee colony algorithm for protein structure optimization based on a three-dimensional off-lattice model". Comput Biol Chem 54:1–12
32. Li B, Gong L, andYao Y (2013) On the performance of internal feedback artificial bee colony algorithm (IF-ABC) for protein secondary structure prediction. In Proc 6th Int Conf Adv Comput Intell, Hangzhou, China, pp 33–38
33. Zhang X and Cheng W (2008) Protein 3D structure prediction by improved tabu search in off-lattice AB model. In Proc 2nd Int Conf Bioinf Biomed Eng, Shanghai, pp 184–187
34. Jana ND, Sil J (2013) Hybrid particle swarm optimization technique for protein structure prediction using 2D off-lattice model. Swarm Evolut Meme Comput 8298:193–204
35. Zhou C, Hou C, Wei X, Zhang Q (2014) Improved hybrid optimization algorithm for 3D protein structure prediction. J Mol Model 20(7):1–12
36. Li Y, Zhou C, and Zheng X (2014) "The application of artificial bee colony algorithm in protein structure prediction. Bio-Inspired Comput Theories Appl 472:255–258

37. Zhou C, Hu T, Zhou S (2014) Protein structure prediction based on improved multiple populations and GA-PSO. Bio-Inspired Comput Theories Appl 472:644–647
38. Benitez CM, Parpinelli RS and Lopes HS (2013) A heterogeneous parallel ecologically-inspired approach applied to the 3D-off-lattice protein structure prediction problem. In Proc 11th Conf Comput Intell, pp 592–597
39. Lin X and Zhang X (2014) Protein folding structure optimization based on GAPSO algorithm in the off-lattice model. In Proc IEEE Conf Bioinf Biomed, pp 43–49
40. Liu J, Sun Y, Li G, Song B, Huang W (2013) Heuristic-based tabu search algorithm for folding two-dimensional off-lattice model proteins. Comput Biol Chem 47:142–148
41. Jana ND, Sil J and Das S (2015) Improved Bees Algorithm for Protein Structure Prediction Using AB Off-Lattice Model. Mendel 2015. Springer, Heidelberg, pp 39–52
42. Fan J, Duan H, Xie H and Shi H (2014) Improved Biogeography-Based Optimization approach to secondary protein prediction. In 2014 International Joint Conference on Neural Networks (IJCNN), IEEE, pp 4223–4228
43. Doğan B, Ölmez T (2015) Modified Off-lattice AB model for protein folding problem using the vortex search algorithm. Int J Mach Learn Comput 5(4):329–333
44. Kalegari DH and Lopes DH (2013) An improved parallel differential evolution approach for protein structure prediction using both 2D and 3D off-lattice models. In 2013 I.E. Symposium on Differential Evolution (SDE), IEEE, pp 143–150
45. Li B, Gong LG and Yang W (2014) An improved artificial bee colony algorithm based on balance-evolution strategy for unmanned combat aerial vehicle path planning. Sci World J doi:10.1155/2014/232704
46. Karaboga D, Basturk B (2007) A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. J Glob Optim 39(3):459–471
47. Li B, Chiong R, and Gong L G (2014) Search-evasion path planning for submarines using the artificial bee colony algorithm. In 2014 I.E. Congress on Evolutionary Computation, IEEE, pp 528–535
48. Zhu G, Kwong S (2010) Gbest-guided artificial bee colony algorithm for numerical function optimization. Appl Math Comput 217(7):3166–3173
49. Gao W, Liu S, Huang L (2013) A novel artificial bee colony algorithm with Powell's method. Appl Soft Comput 13(9):3763–3775
50. Kang F, Li J, Ma Z (2011) Rosenbrock artificial bee colony algorithm for accurate global optimization of numerical functions. Inf Sci 181(16):3508–3531
51. Li G, Niu P, Xiao X (2012) Development and investigation of efficient artificial bee colony algorithm for numerical function optimization. Appl Soft Comput 12(1):320–332