Design and implementation of bi-level artificial bee colony algorithm to train hidden Markov models for performing multiple sequence alignment of proteins

Soniya Lalwani

Department of Mathematics,
Bal Krishna Institute of Technology,
Kota, India
Email: slalwani.pdf@rtu.ac.in

Abstract: Multiple sequence alignment (MSA) is an NP-complete problem that is a challenging area from bioinformatics. Implementation of hidden Markov model (HMM) is one of the most effective approach for executing MSA, that performs training and testing of the sequence data so as to obtain alignment scores with accuracy. The training of HMM is again an NP-hard problem, hence it requires the implementation of metaheuristic methods. Proposed work presents a bi-level artificial bee colony (BL-ABC) algorithm to train hidden Markov models (HMMs) for MSA of proteins, i.e., BLABC-HMM. The trained stochastic model created by BL-ABC basically yields position-dependent probability matrices at higher prediction ratios. The performance of proposed algorithm is compared with the competitive state-of-the-art algorithms and different variants of particle swarm optimisation (PSO) algorithm on protein benchmark datasets from pfam and BAliBase database, and BLABC-HMM is found yielding better alignment scores and prediction accuracy.

Keywords: hidden Markov model; HMM; proteins; artificial bee colony; ABC; multiple sequence alignment; MSA.

Reference to this paper should be made as follows: Lalwani, S. (2021) 'Design and implementation of bi-level artificial bee colony algorithm to train hidden Markov models for performing multiple sequence alignment of proteins', *Int. J. Swarm Intelligence*, Vol. 6, No. 1, pp.48–64.

Biographical notes: Soniya Lalwani obtained her Post-doctorate from the Science and Engineering Research Board (SERB), DST during September 2016–September 2018. She implemented her Post-doctorate project at the Department of Computer Science, RTU, Kota. She received her PhD from the Department of Mathematics, MNIT, Jaipur. Currently, she is working as an Associate Professor with the Department of Mathematics, BKIT, Kota. She has published more than 35 research papers in reputed journals and 15 research papers in conferences. She has over 14 years of job experience at various research and teaching positions. Her research areas include swarm intelligence: particle swarm optimisation, multi-objective

optimisation, bioinformatics: multiple sequence alignment of DNA/RNA sequences, sequence-structure alignment, ABS algorithm and clinical/medical biostatistics.

1 Introduction

Multiple sequence alignment (MSA) is an intricate and substantial technique useful to discover functional, structural and evolutionary information in biological sequences and species. MSA acts as a strong factor in extrapolation of secondary and tertiary structure, building of phylogenetic tree and distinguishing the conserved domain. The laboratory tests with expensive apparatus are not efficient at time and cost factors. In fact, they are likely to have investigational, machinery and manual errors. Therefore, numerous computational efforts are carried out since last two decades in order to cultivate softwares for executing quality alignments at time and space efficiency. MSA is basically a technique of ordering the sequence molecules by introducing gaps in such a manner that the columns may obtain maximum number of identical molecules (i.e., nucleotides for DNA, RNA and amino acids for protein). MSA is an NP-complete problem therefore several approaches have been established to resolve it. These approaches are approximately categorised in four classes:

1 progressive approach

2 exact approach

3 consistency-based approach

4 iterative approach.

Progressive approaches as the name suggests progressively perform the alignment. The alignment is built up by beginning with the utmost identical sequences and then progressively aligning more distant sequences or groups. ClustalW (Thompson et al., 1994) is the most prevalent software based on progressive approaches. They have the drawback of being reliant on primarily provided alignment and scoring pattern. Additional established approaches in this category include MultAlin (Corpet, 1998), MUSCLE (Edgar, 2004), PileUp (Devereux et al., 1984) and MATCH-BOX (Depiereux et al., 1997). The working of exact approaches follows dynamic programming (DP) method (Needleman and Wunsch, 1970). In DP, the shortest track is explored in a weighted direct acyclic graph. The former results are intended for finest alignment, starting from smaller sub-sequences to construct the best likely alignment. DP is unsuccessful for lengthy and large number of sequences (Lipman et al., 1989; Carillo and Lipman, 1988). Consistency-based approaches aim to attain the maximum consensus optimum pairwise alignment. Most widespread softwares in this category are T-coffee (Notredame et al., 2000) and DIALIGN (Subramanian et al., 2005). Consistency-based approaches have been revealed to be the most outperforming approach with respect to accuracy. But this accuracy expenses a very high complexity of time. An iterative algorithm initiates with a random alignment and iteratively enhances it up, until the algorithmic stopping criteria are met. These approaches include simulated annealing (SA) (Kim et al., 1994), hidden Markov model (HMM) training (Eddy, 1995; Lytynoja

and Milinkovitch, 2003; Rasmussen and Krink, 2003), evolutionary algorithms and swarm intelligence (SI) Blum and Merkle (2008) techniques. In general, iterative algorithms are deficient at speed and consistency, Bucak and Uslan (2011) proposed work belongs from this category.

Proposed algorithm bi-level ABC trains HMM for sequence alignment of protein sequences, by employing ABC algorithm in two levels. In first level, it determines estimation parameters and model length. The optimal results of level 1 containing model length and trained parameters then move towards level 2 for constructing a trained HMM of transition and emission probabilities. The results of proposed algorithm are compared with state-of-the-art algorithms and our previously developed algorithm variants. BLABC-HMM is found outperforming than all compared algorithms, confirmed by statistical testing.

The classification of the work is as follows: Section 2 presents the details of the ABC algorithm and HMM for MSA. Section 3 presents the details of proposed bi-level artificial bee colony (BL-ABC) algorithm and its step-by-step procedure to train the HMM for MSA. Section 4 contain the information about experimental setup and benchmark datasets. The results are discussed in Section 5 and concluded in Section 6.

2 Introduction and objectives

2.1 Artificial bee colony algorithm

The recent development of nature-inspired swarm-intelligence-based metaheuristic algorithms enthused many scientific communities to develop and solve complex optimisation problems by using natural metaphors. Artificial bee colony (ABC) is one such recently developed population-based algorithm used to solve many NP-hard, continuous, large-scale combinatorial and numerical optimisation problems. ABC algorithm was first introduced by Karaboga in 2005, which is inspired from the foraging behaviour of real-honey bees and it is used to solve both continuous and discrete optimisation problems. Karaboga considered an intelligent and foraging behaviour of real honey bees to solve multimodal and multidimensional optimisation problems.

The algorithmic configuration of ABC is based on natural foraging behaviour of real honeybee swarm. Generally there are three kinds of honeybee groups determined towards the food search criteria, namely as follows:

1 *Employed bees:* Every individual exploited food sources are associated with the employed bee memory.

2 *Onlooker bees:* In bee hive with a certain probability onlooker bee analyses the waggle dance performed by employed bee and collects all the information regarding nectar amount of the food sources.

3 *Scouts bee:* Without any assistance, scout bees explores the entire search space and randomly find out the new food sources.

The ABC is uniformly divided into two equal halves, where first half constitutes artificial employed bees and second half constitutes artificial onlooker bees. Since, each and every individual food source is linked to an individual employed bee, therefore the number of employed bees will be equal to the number of food sources. Several

employed bees, who abandon their depleted food sources transforms into the scouts bee and explores the environment in search of new food sources. The search process is carried out in three main steps:

- Initialise
- REPEAT
    a Movement of employed and onlooker bees towards their selected food sources and evaluating there nectar amounts.
    b Movement of scout bees in the environment. to search the new food sources (solutions).
    c Memorise all the best possible food sources (solutions) achieved so far.
- UNTIL (requirements are fulfilled).

Every cycle of the search is processed using three main steps: first, movement of employed and onlooker bee directed towards the food sources and evaluating their fitness value, i.e., nectar amount is calculated and then determining the movement of scouts bee to randomly search the possible food sources. The position of food source defines the possible solution for the optimised problem and the amount of nectar with respect to the quality of solution linked with it. In onlooker bee phase, with certain probability onlooker bee choose their food sources on the basis of the information gain from the employed bee and to accomplish this purpose a fitness-based selection technique is used to place the onlooker bee on food by using 'roulette wheel selection' method. Scouts are bees basically perturbed to find any kind of food sources and this type of behaviour typifies the low average quality (fitness) of food source (solution) and the low search costs.

2.2 HMM for MSA

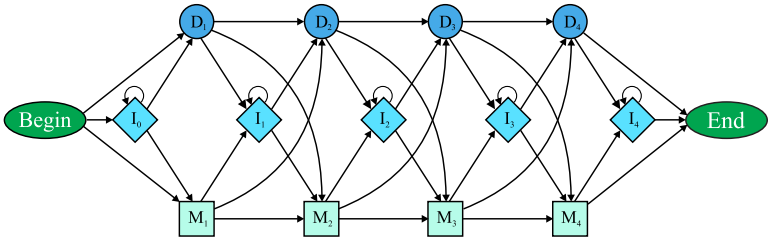
HMM is one of the highly prevalent and accurate modelling techniques for MSA proposed by Krogh et al. (1994). MSAs are modelled by training of HMMs, which is an NP-hard problem. MSA is basically a technique of arranging the elements of a row (i.e., sequence) in such a way that it may result in maximum number of matches in columns. The more the number of matches in a column are, the more increment in alignment score will be. A MSA is presented by Figure 1 for a sub-sequence of PHYLIP nucleotide sequences taken from EMBL-EBI services. The left most part in the figure is the sequence ID of each sequences, which stays unique and right most part contains the length of the aligned sequence. A few columns with symbol '-' are observable in the sequence. This symbol is named as gap which determines the state of sequence, i.e., insertion or deletion. It means a sub-sequence has been inserted or deleted due to the mutation, a genetic process. The columns containing coloured columns show the relation between sequences, i.e., sequences in column colour dark blue are 100% identical sequences, whereas sequences in column colour light blue are 70% identical sequences, and sequences in column colour light green are sequences with less than 70% of similarity or no similarity. The sequences with highest match in the figure are known as the match state sequences, whereas the sequences with gaps are in insertion or deletion state. These states build HMM, by giving the classification of match, insert

and delete states. HMMs are trained to know the states pattern of a specific family, known as profile HMM.

Figure 1 Multiple sequence alignment (see online version for colours)

```
CAA28435.1  ATGT-CTCTGACCAGGAC-TGAGAGGACCA-TCATCCTGT  37
BAA20512.1  AT-GAGTCTCTCT-GATAAGGAC-AAGGCTGCTGTGAAAG  37
CAA23748.1  ATG-GTGCTGTCT-CCTGCCGACA-AGACCAACGTCAAGG  37
CAA24095.1  AT-GGTGCTCTCTGGGGAA-GACAAAAGCA-ACATCAAGG  37
```

Figure 2 Profile HMM structure for MSA (see online version for colours)



Usually HMM training is performed by Baum-Welch algorithm (Rabiner, 1989). Other meta-heuristic available are evolutionary algorithms (Slimane et al., 1996), simulating annealing (Kim et al., 1994) and SI-based algorithms (Rasmussen and Krink, 2003). Structure for developing a profile HMM of length four is presented by Figure 2. It is build-up of three hidden states: match (M), insertion (I), deletion (D), and $o + 2$ additional states. Number of observables states remains o and number of dummy states is two, i.e., start and end. The lowest line (with rectangular shapes) in figure characterises a sequence with four match states. Middle line (with rhombus shapes) displays the insertion states, whereas, top most line (with circular structure) illustrates deletion states. Connection probability between the states is known as transition probability p_{ij}. Each column represents a match state and every match or insert states excretes a symbol η_m, $\eta_m \in$ the set of all 26 amino acids for protein alignment (namely set A_A). There is a delete state consistent to every match state. Delete states are known as silent states because they do not excrete any symbol. Also, the dummy states 'begin' and 'end' do not excrete any symbol, hence delete states. Permissible transition in insertion states can happen from the state insert to insert, hence insert states can contain several adjacent columns.The trained HMM yields the trained sequence sets in form of MSA, known as a profile HMM. Profile HMM is expedient in aligning the sequences of the group. Transition from state t_a to t_b is expressed by transition probability matrix, as below:

$$p_{ab} = P\ (t_b \mid t_a), (1 \leq a \leq b \leq n)$$
$$\text{for} \sum_{c=1}^{3} p_{ac} = 1 \qquad \forall a = 1, 2, \ldots, n \tag{1}$$

here c represents any state out from the three states match (M), insert (I) and delete (D). The emission probability matrix e_{bc} is:

$$e_{bc} = e_b(\eta_c) = P\ (\eta_c \mid t_b), (1 \leq b \leq n, \eta_c \in \mathrm{A}_A)$$

$$\text{where } \sum_{c=1}^{27} e_{ac} = 1 \qquad \forall a = 1, 2, \ldots n \tag{2}$$

here c is the number of emission symbol, which is equal to the number of elements in A_A (27 here).

A trained HMM model ξ is obtained by employing the training set (namely S) containing N aligned sequences, i.e., $S = (S_1, S_2, ..., S_N)$. Transition and emission probability matrices obtained for trained HMM, deliver the maximum values of probabilities of S created from ξ, i.e., $P\ (S \mid \xi)$. The comprehensive training procedure is explicated as follows:

Step 1 Length ascertainment of profile HMM: The frequent strategy concerning the ascertainment of length of HMM remains to approximate the normal length of nonaligned sequences. Model surgery is applied to modify the length after training (Krogh et al., 1994).

Step 2 Update of estimation parameters: Transition and emission probabilities are the estimation parameters that are updated by Baum-Welch algorithm in general. The process is performed by training over the aligned/unaligned sequences.

Step 3 Assessment of alignment quality: Throughout the training process, assessment of model is performed by calculating the log-likelihood score (LLS). It is the measure of evaluating the quality of alignment.

$$LLS\ (S, \xi) = \frac{1}{N} \sum_{i=1}^{N} \log_2 \frac{P\ (S_i \mid \xi)}{P\ (S_i \mid \xi_r)} \tag{3}$$

here ξ_r stands for the null-hypothesis model, a random model. The normalised score for t^{th} method is evaluated through:

$$\mu_{score} = \frac{S(t) - S_{avg}}{SD} \qquad \forall t = 1, 2, ..., M \tag{4}$$

here μ_{score} stands for the normalised score, S_{avg} is the average score of all the M methods' score and $S(t)$ is the score from t^{th} method.

Step 4 Creation of profile HMM: The trained model ξ acts as the profile HMM, that aims the groups of nonaligned sequences. For q nonaligned sequences displayed as: $Q = (Q_1, Q_2, ..., Q_q)$, the structure of procedure is:

1 The trained model ξ yields the maximum probable state track MP_s, attained through Viterbi algorithm, for nonaligned sequences Q_s, $\forall s = 1, 2, ..., q$.

2 This track provides linked emitted symbols, i.e., a gap is excreted by a delete state, whereas an amino acid is excreted by an insert state.

3 Whenever complete probable state paths are obtained for all sequences, the aligned sequences can be obtained.

Step 5 Alignment score evaluation: The quality of an alignment is evaluated at the basis of alignment scores of the alignments, i.e., sum-of-pairs score (SoP_S) and similarity score (S_S). The sequences that do not have reference alignment are evaluated by S_S scheme and the sequences with reference alignments, are evaluated at SoP_S. The S_S scheme is expressed as:

$$S_S = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} score(S_i, S_j) \tag{5}$$

here N is the number of sequences. The formulation of the models of affine gap penalty contain G_P as resultant gap penalty, G_o gap open penalty and G_E as extension gap penalty along with number of gaps G_L for a single gap opening series:

$$G_P = G_o + (G_L - 1) * G_E \tag{6}$$

All the gap penalties of all strings are added for the final gap penalty. Resultant alignment score is formulated by:

$$Resultant\ alignment\ score = Alignment\ score - \sum G_P \tag{7}$$

For a sequence set with N sequences containing 'a' columns in test set and 'b' columns in the reference set, the SoP_S is represented as:

$$SoP_S = \frac{\sum_{i=1}^{a} S_i}{\sum_{j=1}^{b} S_j} \tag{8}$$

here S_j is the alignment score for the j^{th} column in reference set and S_i is the alignment score for the i^{th} column of test set, formulated as:

$$S_i = \sum_{j=1, j\neq k}^{N} \sum_{k=1}^{N} p_{ijk} \tag{9}$$

For the i^{th} column, the function between any two residues from S_{i1}, S_{i2}, ..., S_{im} results in p_{ijk} = 1 if residues S_{ij} and S_{ik} are aligned with each other in the reference alignment, otherwise the results becomes p_{ijk} = 0.

3 Construction of profile HMM by BL-ABC algorithm

The training of HMM is performed in two levels by proposed BL-ABC algorithm. Level 1 employs discrete version of ABC algorithm for determination of model length, whereas level 2 is implemented on the optimal length parameters obtained from level 2. The model length is re-estimated by σ and κ, so as to enhance the diversity and to prevent trapping in local optima. Problem formulated in ABC architecture for HMM contains the following terms:

- *Bee:* Each Bee represents a profile corresponding to an alignment and its transition and emission probability matrix. Hence, every bee has a path built-of match (M), insert (I) and delete (D) states. Insertions comes in form of emitting states and deletions as non-emitting states.
- *Optimal solution:* The parameters providing most suitable transition and emission matrices that produce highest probabilities, are the optimal solutions.

The objective formulation is:

$$F = \max(LLS) \tag{10}$$

where LLS is calculated by equation (3) and is equal to the minimisation of negative of the function f. Further, the fitness is evaluated for the objective function by:

$$Fitness = \begin{cases} \frac{1}{1+F} & \text{if } F \geq 0 \\ 1 + abs(F) & \text{if } F < 0 \end{cases} \tag{11}$$

here abs stands for the absolute value. For t iterations, $drand$ for generating discrete random variable, h employed, h onlooker bees, X position, C cost, F_1 fitness, $n.X$ as the new bee position, $n_c.F$ for the fitness of c^{th} bee, pop as population, E_B as employed bees phase, O_B as observer bees phase and S_B as scout bees phase, step by step process of BL-ABC is as follows:

Step 1 Initially the parameter values are determined for levels 1 and 2.

Step 2 Iterative process gets initiated:

for $c = 1, 2, ..., t$
$E_B()$
$O_B()$
$S_B()$
endfor

Step 3 The process begin for the loop of level 1.

Level 1

Step 4 The loop firstly enters in the employed bees phase for determination of the model length.

$E_B()$

Step 5 Position update.

Here, length of the model for level 1 is initially determined by:

$$m^j_{1l(i)} = l^j_{avg} + int(rand * (l^j_{\max} - l^j_{avg})) \tag{12}$$

here $m^{j}_{1l(i)}$: model length for i^{th} bee and j^{th} training sequence set from first training set; l^{j}_{avg}: average sequence length; and $l^{j}_{\max}$: highest length of sequence from unaligned sequences. One bee is additionally added to the group that serves as the profile acquired from Baum-Welch algorithm. The j^{th} training set of i^{th} bee, presents a profile with specific model length. $e = drand(1, 2, ..., h)$, $e \neq c$, $\forall c = 1, 2, ..., h$

$$n.X = pop_c.X + P * (pop_c.X - pop_e.X) \tag{13}$$

Step 6 Length modification with φ and ψ.

Length modification gets started for enhancing the matching between the model and the inserted sequences. For this, population normalisation is performed by modifying the length of the model related to each profile using the model surgery parameters φ and ψ. If a specific column contains delete in more than half of the paths, then that position is dismissed from the model with the help of φ parameters. Whereas, if a specific x^{th} column contains more than half of the paths, then the average number of insertions namely y is calculated. Now, after x^{th} position, y new positions are created with ψ parameter. The process is repeated until the process completion. The training parameters are updated after each normalisation, subject to the transition and emission probability constraints, as shown in equations (1) and (2).

Step 7 Cost evaluation.

Step 8 Evaluate new bee cost $n.C$ by:

$$n.C = LLS(n.X) \tag{14}$$

Step 9 Now the loop enters into the onlooker bees phase.

$O_B()$

Step 10 Fitness evaluation.

Evaluate the fitness by equation (11), further determine F_{1c}, i.e.,

$$F_{1c} = Fitness(pop_c)$$
$$c = Roulette\ wheel\ selection(F_{1c})$$
$$e = drand(1, 2, ..., h), e \neq c \qquad \forall c = 1, 2, ..., h$$

Step 11 Probability measure.

Evaluate the probability for the selection of food source at pop_c by:

$$prob_c = \frac{n_c.F}{\sum_{e=1}^{h} n_e.F} \tag{15}$$

Step 12 Position update.

Update the new bee position $n.X$ by equation (13).

Step 13 Length modification.

Apply step 6.

Step 14 Cost evaluation.

Step 15 Evaluate the new bee cost $n.C$ by equation (14).

Step 16 Loop now enters into scout bees phase.

$S_B()$

Step 17 Determination of $pop_c.X$.

$$pop_c.X = [drand(1, 2, ..., p)]_{1\times h} \qquad \forall c = 1, 2, ..., h$$

Step 18 Length modification.

Apply step 6.

Step 19 Determination of $pop_c.C$.

$pop_c.C$ is obtained by:

$$pop_c.C = LLS(pop_c.X), \qquad (pop_c.LimitCount > Limit) \tag{16}$$

Level 2

Step 20 Extract the model length for all bees from level 1 and employ it for profile length of level 2.

Step 21 The loop for level 2, enters into the employed bees phase for determination of the model length.

$E_B()$

Step 22 For the predetermined length from level 1 is applied at the profile length, for the optimal parameters of level 2.

$$b = rand(1, 2, ..., h), e \neq c, \qquad \forall c = 1, 2, ..., h$$

$$n.X = pop_c.X + P * (pop_c.X - pop_e.X) \tag{17}$$

Step 23 Update $n.C$ by equation (14).

Step 24 Onlooker bees phase starts.

$O_B()$

Step 25 Evaluate the fitness by equation (11) for F_{1c}, i.e.,

$$F_{1c} = Fitness(pop_c)$$
$$c = Roulettewheelselection(F_{1c})$$
$$e = rand(1, 2, ..., h), e \neq c \qquad \forall c = 1, 2, ..., h$$

Step 26 Evaluate the probability for the selection of food source at pop_c by equation (15).

Step 27 Update the new bee position $n.X$ by equation (13).

Step 28 Evaluate the new bee cost $n.C$ by equation (14).

Step 29 The loop now enters into scout bees phase.

$S_B()$

Step 30 Position determination.

$$pop_c.X = [rand(1, 2, ..., p)]_{1\times h} \qquad \forall c = 1, 2, ..., h$$

Step 31 $pop_c.C$ is obtained by equation (16).

The training dataset (T_s) are separated into two parts for both the levels in order to carry out cross training, whereas if training sets are not available, then same sequences can be used in both the levels. This helps in sustaining the model structure and ensuring the most advantageous compression from HMM. The trained model can now be implemented to find out the best MSA of new nonaligned sequence. The probability $P\ (unaligned\ sequences|HMM)$ is evaluated by Viterbi/forward algorithm by obtaining the ideal track regarding the best alignment. The strategy taken up in Yoon and Vaidyanathan (2008) was to create a typical profile followed by measuring the maximum distance between the matching amino acids/nucleotides, whereas TLPSO-HMM was implemented to train the HMM in Lalwani et al. (2015). Proposed work is different in the profile HMM construction method, i.e., BL-ABC constructs the conventional profile HMM here. This helps in obtaining finer estimation parameters, finer training, better HMM profile as compared to Baum-Welch as shown in next section.

4 Experimental setup

4.1 Benchmark dataset

The performance of BL-ABC has been evaluated on two benchmark datasets from protein families, i.e., sequence set P_1 and P_2. Sequence sets P_1 do not contain the reference sets, whereas P_2 contains them. The details of P_1 and P_2 datasets are presented by in Table 1. Dataset P_1 is drawn from pfam (Sonnhammer et al., 1997), whereas P_2 is drawn from BAliBase database (Thompson et al., 1999). P_1 was randomly spawned by Rose (Stoye and Evers, 1998; Sun et al., 2012). The sequence sets of P_1 are separated into training and validation sets, i.e., T_s and V_s. 150 sequences are taken in T_s, that are divided into half-half parts for levels 1 and 2 respectively. Rest of the sequences are taken as the validation set, whereas, P_2 is not separated into T_s and V_s due to limited quantity sequences. l_{avg}, $l_{\min}$ and $l_{\max}$ stand for average, minimum and maximum sequence lengths respectively. Here, APSI stands for average sequence identity, a measure of similarity between sequences. The scoring scheme employed for dataset P_1 is S_S [equation (5)] and SoP_S [equation (8)] for dataset P_2.

Table 1 Benchmark datasets of protein families

Dataset	*Name*	T_s	V_s	$l_{avg}(l_{\min}, l_{\max})$
P_1	G5	75	127	79 (67, 88)
	CagY_M	75	399	31 (24, 35)
	Interferon	75	225	164 (23, 200)
	Biopterin_H	75	193	170 (13, 359)
Dataset	*Name*	*N*	*APSI (%)*	$l_{avg}(l_{\min}, l_{\max})$
P_2	1aboA	5	<25	59 (49, 80)
	1idy	5	<25	54 (49, 58)
	451c	5	20–40	78 (70, 87)
	1krn	5	>35	78 (66, 82)
	1bbt3	5	<25	176 (149, 192)
	kinase	5	<25	270 (263, 276)
	1pii	4	20–40	252 (247, 259)
	5ptp	5	>35	232 (222, 245)
	gal4	5	<25	362 (335, 395)
	1ajsA	4	<25	370 (358, 387)
	glg	5	20–40	468 (438, 486)
	1taq	5	>35	865 (806, 928)

4.2 Parameter settings

The parameters setting for BL-ABC, performed in MATLAB programming environment is as follows: limit = 20, no. of onlookers = 15, no. of function evaluations = 2,000, population size = 50, and dimension = no. of sequence sets. Similarly, S_S method parameters are: Alignment scores evaluated from BLOSUM62 matrix: G_o = –11 and G_E = –2.

5 Experimental results and discussion

The performance of BL-ABC is tested against the particle swarm optimisation (PSO) algorithm variants and competitive state-of-art algorithms (Rasmussen and Krink, 2003). Tables 2 and 3 outline the simulation outcomes for protein sequence sets P_1 and P_2. The results included in comparison are taken from the several PSO algorithm variants namely standard PSO (SPSO), quantum-behaved PSO (QPSO) and diversity maintained QPSO (DMQPSO), as well as from the state-of-art algorithm, i.e., ClustalW (CW) and Baum-Welch (BW). The result included in comparison with proposed BL-ABC, are taken from Sun et al. (2012).The evaluation criteria are average LLS, S_S and SoP_S and μ_{score} for T_s and V_s. The characters in bold faced letter represent the best results in respective category. Table 2 shows the comparative results of proposed BLABC-HMM at the grounds of LLS and S_S for dataset P_1, whereas Figure 3 presents the μ_{score} comparison. Table 3 presents the similar kind of comparison for dataset P_2.

Table 2 Comparison for HMM LLS and S_S for P_1 dataset

Protein	*Algorithm*	*LLS* (T_S)	*LLS* (V_S)	S_S
G5	SPSO	101.45	141.45	176
	QPSO	154.94	154.94	229
	DMQPSO	173.94	173.94	230
	ClustalW	-	-	189
	Baum-Welch	103.146	78.357	192
	BLABC-HMM	*197.18*	*197.18*	*242*
CagY_M	SPSO	20.090	20.090	–120
	QPSO	28.255	20.255	–106
	DMQPSO	31.649	31.649	–103
	ClustalW	-	-	–142
	Baum-Welch	11.178	12.832	–138
	BLABC-HMM	*36.67*	*36.67*	*–97.03*
Interferon	SPSO	141.736	95.736	3,772
	QPSO	179.549	179.549	4,136
	DMQPSO	188.63	188.63	4,835
	ClustalW	-	-	3,226
	Baum-Welch	158.314	102.652	3,294
	BLABC-HMM	*203.437*	*203.437*	*4,983*
Biopterin_H	SPSO	179.521	162.292	3,924
	QPSO	179.549	179.549	4,328
	DMQPSO	201.284	188.63	4,926
	ClustalW	-	-	4,015
	Baum-Welch	162.431	171.281	4,113
	BLABC-HMM	*242.18*	*222.01*	*5,092*

Table 3 Comparison for HMM LLS and SoP_s for dataset P_2

Protein	*Algorithms*	*LLS*	μ_{score}	SoP_S	μ_{score}
1aboA	SPSO	63.8205	–0.5818	0.6974	–0.5696
	QPSO	84.2931	0.4118	0.7519	0.3254
	DMQPSO	86.1835	0.5036	0.7728	0.6686
	ClustalW	-	-	0.7140	–0.2970
	Baum-Welch	46.3814	–1.4283	0.6418	–1.4826
	BLABC-HMM	*98.3612*	*1.0947*	*0.8146*	*1.3551*
1idy	SPSO	59.7932	–0.4693	0.5658	–1.0009
	QPSO	71.4864	0.1893	0.7763	0.4892
	DMQPSO	80.5751	0.7012	0.8158	0.7688
	ClustalW	-	-	0.705	–0.0156
	Baum-Welch	42.0576	–1.4683	0.5132	–1.3734
	BLABC-HMM	*86.7142*	*1.0470*	*0.8671*	*1.1319*

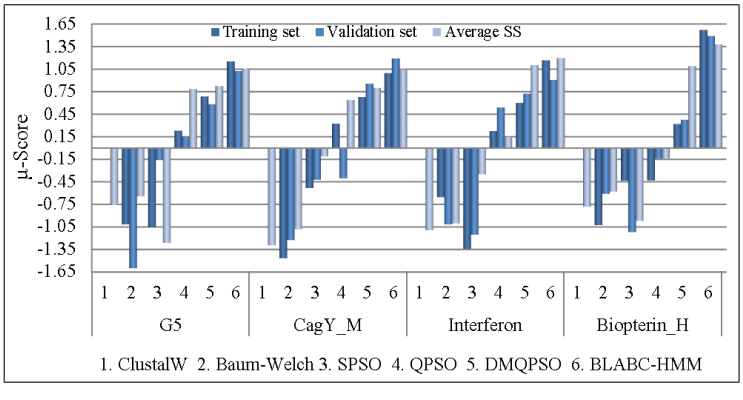
Table 3 Comparison for HMM LLS and SoP_s for dataset P_2 (continued)

Protein	*Algorithms*	*LLS*	μ_{score}	SoP_S	μ_{score}
451c	SPSO	89.1605	0.6481	0.4519	–4.6011
	QPSO	106.3024	1.4801	0.5027	–3.7669
	DMQPSO	90.3075	0.7038	0.6301	–1.6748
	ClustalW	-	-	0.7190	–0.2149
	Baum-Welch	68.3522	–0.3619	0.3989	–5.4715
	BLABC-HMM	*98.9841*	*1.1249*	*0.8042*	*1.1843*
1krn	SPSO	81.9846	0.7806	0.7863	0.5600
	QPSO	103.6417	2.0005	0.9585	1.7790
	DMQPSO	110.5327	2.3886	0.9968	2.0501
	ClustalW	-	-	*1.0000*	*2.0728*
	Baum-Welch	69.0222	0.0505	0.8182	0.7858
	BLABC-HMM	*121.7531*	*3.0206*	*1.0000*	*2.0728*
1bbt3	SPSO	169.2160	4.5337	0.6219	–1.8094
	QPSO	211.4329	6.5828	0.7146	–0.2871
	DMQPSO	236.8514	7.8165	0.7253	–0.1114
	ClustalW	-	-	0.6380	–1.5450
	Baum-Welch	172.3816	4.6874	0.5347	–3.2414
	BLABC-HMM	*303.9867*	*11.0751*	*0.8751*	*2.3486*
kinase	SPSO	211.2745	8.0630	0.3061	–2.8395
	QPSO	356.8937	16.2652	0.5753	–0.9337
	DMQPSO	403.8526	18.9102	0.6053	–0.7214
	ClustalW	-	-	0.7360	0.2039
	Baum-Welch	214.9693	8.2711	0.2268	–3.4008
	BLABC-HMM	*3498.9931*	*324.2690*	*30.8036*	*30.6824*
1pii	SPSO	277.0576	9.7680	0.2738	–7.5259
	QPSO	328.1439	12.2476	0.6372	–1.5582
	DMQPSO	310.1645	11.3749	0.7064	–0.4218
	ClustalW	-	-	*0.8640*	*2.1663*
	Baum-Welch	213.0459	6.6611	0.1647	–9.3175
	BLABC-HMM	*403.6541*	*15.9126*	0.8542	2.0054
5ptp	SPSO	311.5647	13.7120	0.6831	–0.1706
	QPSO	428.8537	20.3184	0.8572	1.0619
	DMQPSO	504.1372	24.5588	0.9074	1.4172
	ClustalW	-	-	*0.9660*	*1.8321*
	Baum-Welch	266.5928	11.1789	0.6053	–0.7214
	BLABC-HMM	*605.4531*	*30.2655*	0.9434	1.6721
gal4	SPSO	389.3147	15.2166	0.3185	–6.7918
	QPSO	484.5218	19.8377	0.5294	–3.3284
	DMQPSO	567.3841	23.8595	0.5784	–2.5238
	ClustalW	-	-	0.4830	–4.0904
	Baum-Welch	347.2819	13.1765	0.2017	–8.7099
	BLABC-HMM	*623.2317*	*26.5702*	*0.7081*	*–0.3939*
1ajsA	SPSO	381.6639	17.6604	0.3245	–2.7092
	QPSO	483.7352	23.4096	0.5914	–0.8198
	DMQPSO	536.2753	26.3690	0.6031	–0.7369
	ClustalW	-	-	0.5710	–0.9642
	Baum-Welch	326.4896	14.5526	0.2864	–2.9789
	BLABC-HMM	*587.0945*	*29.2314*	*0.7107*	*0.0248*

Table 3 Comparison for HMM LLS and SoP_s for dataset P_2 (continued)

Protein	*Algorithms*	*LLS*	μ_{score}	SoP_S	μ_{score}
glg	SPSO	395.1211	15.4984	0.6684	–1.0458
	QPSO	486.5318	19.9352	0.8569	2.0497
	DMQPSO 589.0737	24.9123	0.8895	2.5851	
	ClustalW	-	-	*0.9410*	*3.4308*
	Baum-Welch	380.7306	14.8000	0.5691	–2.6765
	BLABC-HMM	*631.0652*	*26.9504*	0.9245	3.1598
1taq	SPSO	763.7521	39.1818	0.6931	–0.0998
	QPSO	875.6509	45.4846	0.7953	0.6237
	DMQPSO	953.9467	49.8947	0.8504	1.0137
	ClustalW	-	-	*0.9630*	*1.8108*
	Baum-Welch	729.3726	37.2454	0.6453	–0.4382
	BLABC-HMM	*989.6754*	*51.9072*	0.9002	1.3663

Figure 3 Comparison of μ_{score} for P_1 dataset (see online version for colours)



The results for datasets P_1 and P_2 from Tables 2, 3 and Figure 3 show that BLABC-HMM is generally an out-performer than compared algorithms and methods, at the criteria of alignment quality (evaluated by S_S, SoP_S and μ_{score}) and prediction accuracy (evaluated by LLS). The algorithm is yielding higher prediction ratios even at lower APSI scores.

6 Conclusions

Proposed approach develops BL-ABC algorithm for training HMM, in order to perform MSA of proteins. The structure of the model is preserved whereas the best compression is obtained along with improvement in the prediction accuracy. The training set is comprised of training and cross training sets in proposed methodology. First level of the algorithm provides the optimised model length, that is carried forward to the second level for obtaining the optimal parameters for the complete stochastic model. Proposed BL-ABC algorithm is an effective framework for protein MSA as confirmed by the alignment quality and prediction accuracy results of BL-ABC in comparison

to the state-of-art and PSO-based algorithms. Hence, BL-ABC algorithm is proven efficient in building HMM at better prediction accuracy. As a future scope of proposed work, implementation of BLABC-HMM in parallel computing environment for handling highly complex protein sequences can be explored.

Acknowledgements

The author gratefully acknowledges ATU (RTU), TEQIP-III. She is thankful to Dr. Krishna Mohan from BISR, Jaipur, India for his valuable suggestions throughout the work.

References

Blum, C. and Merkle, D. (2008) 'Swarm intelligence', in *Swarm Intelligence in Optimization*, pp.43–85, Springer, Berlin, Heidelberg.

Bucak, I.O. and Uslan, V. (2011) 'Sequence alignment from the perspective of stochastic optimization: a survey', *Turkish Journal Electrical Engineering and Computer Science*, Vol. 19, No. 1, pp.157–173.

Carillo, H. and Lipman, D. (1988) 'The multiple sequence alignment problem in biology', *Societyfor Industrial Applied Mathematics*, Vol. 48, No. 5, pp.1073–1082.

Corpet, F. (1988) 'Multiple sequence alignment with hierarchical clustering', *Nucleic Acids Research*, Vol. 16, No. 22, pp.10881–10890.

Depiereux, E., Baudoux, G., Briffeuil, P., Reginster, I., Xavier, X., Vinals, C. and Feytmans, E. (1997) 'Match-Box_server: a multiple sequence alignment tool placing emphasis on reliability', *Bioinformatics*, Vol. 13, No. 3, pp.249–256.

Devereux, J., Haeberli, P. and Smithies, O. (1984) 'A comprehensive set of sequence analysis programs for the VAX', *Nucleic Acids Research*, Vol. 12, No. 1, Part 1, pp.387–395.

Eddy, S.R. (1995) 'Multiple alignment using hidden Markov models', *Proceedings of the Third International Conference on Intelligent Systems for Molecular Biology*, MenloPark, CA, pp.114–120.

Edgar, R.C. (2004) 'MUSCLE: a multiple sequence alignment method with reduced time and space complexity', *BioMed Central Bioinformatics*, Vol. 5, No. 1, pp.1–19.

Kim, J., Pramanik, S. and Chung, M.J. (1994) 'Multiple sequence alignment using simulated annealing', *Computer Applications in the Biosciences*, Vol. 10, No. 4, pp.419–426.

Krogh, A., Brown, M., Mian, I.S., Sjolander, K. and Haussler, D. (1994) 'Hidden Markov models in computational biology: applications to protein modeling', *Journal of Molecular Biology*, Vol. 235, No. 5, pp.1501–1531.

Lalwani, S., Kumar, R. and Gupta, N. (2015) 'A novel two-level particle swarm optimization approach to train the transformational grammar based hidden Markov models for performing structural alignment of pseudo knotted RNA', *Swarm and Evolutionary Computation*, February, Vol. 20, pp.58–73.

Lipman, D.J., Altschul, S.F. and Kececioglu, J.D. (1989) 'A tool for multiple sequence alignment', *Proceedings of the National Academy of Science*, USA, Vol. 86, pp.4412–4415.

Lytynoja, A. and Milinkovitch, M.C. (2003) 'A hidden Markov model for progressive multiple alignment', *Bioinformatics*, Vol. 19, No. 12, pp.1505–1513.

Needleman, S.B. and Wunsch, C.D.(1970) 'A general method applicable to the search for similarity in the amino acid sequences of two proteins', *Journal of Molecular Biology*, Vol. 48, No. 3, pp.443–453.

Notredame, C., Higgins, D.G. and Heringa, J. (2000) 'T-coffee: a novel method for fast andaccurate multiple sequence alignment', *Journal of Molecular Biology*, Vol. 302, No. 1, pp.205–217.

Rabiner, L.R. (1989) 'A tutorial on hidden Markov models and selected applications in speech recognition', in *Proceedings of the IEEE*, Vol. 77, No. 2, pp.257–285.

Rasmussen, T.K. and Krink, T. (2003) 'Improved hidden Markov model training for multiple sequence alignment by a particle swarm optimization-evolutionary algorithm hybrid', *Biosystems*, Vol. 72, Nos. 1–2, pp.5–17.

Slimane, M., Venturini, G., de Beauville, J.P.A., Brouard, T. and Brandeau, A. (1996) 'Optimizing hidden Markov models with a genetic algorithm', in Alliot, J.M., Lutton, E., Ronald, E., Schoenauer, M. and Snyers, D. (Eds.): *Artificial Evolution*, Vol. 1063, pp.384–396, Springer, Berlin, Heidelberg.

Sonnhammer, E.L., Eddy, S.R. and Durbin, R. (1997) 'PFAM: a comprehensive database of protein families based on seed alignments', *Proteins*, Vol. 28, No. 3, pp.405–420.

Stoye, J. and Evers, D. (1998) 'Rose: generating sequence families', *Bioinformatics*, Vol. 14, No. 2, pp.157–163.

Subramanian, A.R., Menkho, J.W., Kaufmann, M. and Morgenstern, B. (2005) 'DIALIGN-T: an improved algorithm for segment-based multiple sequence alignment', *Bioinformatics*, Vol. 6, No. 1, pp.1–13.

Sun, J., Wu, X., Fang, W., Ding, Y., Long, H. and Xu, W. (2012) 'Multiple sequence alignment using the hidden Markov model trained by an improved quantum-behaved particle swarm optimization', *Information Sciences*, Vol. 182, No. 1, pp.93–114.

Thompson, J.D., Higgins, D.G. and Gibson, T.J. (1994) 'Clustal W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice', *Nucleic Acids Research*, Vol. 22, No. 22, pp.4673–4680.

Thompson, J., Plewniak, F. and Poch, O. (1999) 'BAliBase: a benchmark alignments database for the evaluation of multiple sequence alignment programs', *Bioinformatics*, Vol. 15, No. 1, pp.87–88.

Yoon, B.J. and Vaidyanathan, P.P. (2008) 'Structural alignment of RNAs using Profile-csHMM and its application to RNA homology search: overview and new results', *IEEE-Special Issue on System Biology*, Vol. 53, Special Issue, pp.10–25.