The Experimental Study of Population-based Parameter Optimization Algorithms on Rule-based Ecological Modelling

Hongqing Cao
School of Earth and Environmental Sciences,
University of Adelaide, 5005
Australia
hongqing.cao@adelaide.edu.au

Friedrich Recknagel
School of Earth and Environmental Sciences,
University of Adelaide, 5005
Australia
friedrich.recknagel@adelaide.edu.au

Philip T. Orr
Seqwater, PO Box 16146,
City East Qld, 4002
Australia
porr@seqwater.com.au

Abstract—**This study investigates six population-based algorithms for the parameter optimization (PO) within the hybrid methodology developed for modelling algal abundance by rule-based models. These PO algorithms include: (1) Hill Climbing (2) Simulated Annealing (3) Genetic Algorithm (4) Differential Evolution (5) Covariance Matrix Adaptation Evolution Strategy and (6) Estimation of Distribution Algorithm. The effectiveness of algorithms is tested on the *Cylindrospermopsis* abundance data from Wivenhoe Reservoir in Queensland (Australia). We provide a systematic analysis and comparison of different parameter optimization algorithms as well as the resulting predictive rule models.**

Keywords-evolutionary algorithm; genetic programming; parameter optimization; population-based algorithm; ecological modelling

I. INTRODUCTION

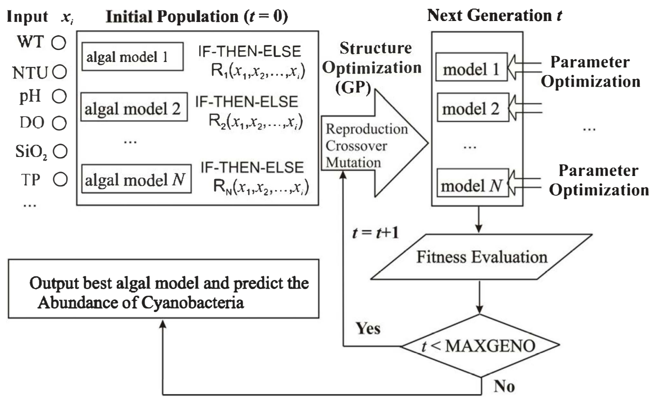
Cyanobacterial blooms, perhaps of increasing intensity and extent, are a recurrent feature of Australian rivers, reservoirs and lakes [1], [2]. The cyanobacteria causing these blooms may produce toxins that may cause death if ingested in sufficient amounts by domestic animals and wildlife. They also pose significant health risks to humans using the water [3], [4]. Harmful algal blooms have caused massive economic cost to Australia annually [5]. There is an immediate need to develop an advanced modelling tool for cyanobacterial blooms to facilitate informed decision-making for water managers.

Traditionally models of phytoplankton dynamics are expressed as process-based mathematical models of eutrophication based on ordinary differential equations (ODEs) [6], [7] which allow simulations of food web dynamics and nutrient cycles over time. However these ODEs are typically calibrated and validated for one site only. With the rapid development of computing technology, extensive study of machine learning techniques has been conducted in ecological modelling. Artificial neural network [8]-[10] is one of the well-established technologies in machine learning and a mainstream technology for data-driven modelling. Despite its great advantage to approximate nonlinear multivariate functions with high accuracy, a major drawback of this approach is that it is difficult to extract and explain the knowledge about the relationship between the model inputs and outputs. On the contrary, fuzzy logic models [11], [12] can make up for this to some extent. They can improve the model interpretability by introducing some fuzzy rules, but the acquisition of the fuzzy rules largely depends on expert empirical knowledge.

In recent years, the study of evolutionary algorithms (EAs) has gained interest of many researchers in a wide range of fields [13]. Since 2003, our group led by Dr Friedrich Recknagel have initiated pioneering work on modelling algal blooms by using EAs [14]-[16]. Recently we developed a hybrid evolutionary algorithm (HEA) to build an algal abundance model with a single IF-THEN-ELSE rule structure [14]. The most advantage of such rule models lies in its high interpretability. Meanwhile from our research, we surprisingly found that when developing predictive rule model for algal abundance, in many runs, the modelling algorithm can find the threshold values for some key water quality parameters, like water temperature, pH value etc. This gives us a hint that the calibration of the random constants in the rule is crucial work. Previously we used a genetic algorithm (GA) based on multi-parent crossover [17] to optimise the model parameters contained in a rule and have not compared with other optimisation algorithms. In this study we extend previous work by systematically studying six population-based algorithms for the parameter optimization within the hybrid methodology which include: (1) Hill Climbing (HC) (2) Simulated Annealing (SA) (3) Genetic Algorithm (GA) (4) Differential Evolution (DE) (5) Covariance Matrix Adaptation Evolution Strategy (CMA-ES) and (6) Estimation of Distribution Algorithm (EDA). The effectiveness of all algorithms is tested on the *Cylindrospermopsis* (one of the Cyanobacterial populations) abundance data from Wivenhoe Reservoir in Queensland (Australia). We provide a systematic analysis and comparison of different parameter optimization algorithms as well as the resulting predictive rule models.

II. A HYBRID EVOLUTIONARY ALGORITHM FOR EVOLVING IF-THEN-ELSE RULES

We propose a hybrid evolutionary algorithm (HEA) to evolve the rule model for predicting the algal abundance. There are two layers in the HEA in which the first layer searches for model structures using genetic programming (GP) [18], [19], whereas the inner layer searches for the optimal continuous parameters of the model by using some population-based optimization algorithms. Fig. 1 illustrates the conceptual diagram of the HEA based on water-quality input data and cyanobacteria output data. The details of the GP and the population-based algorithms are described in what follows.



Input x_i
WT
NTU
pH
DO
SiO_2
TP
...
Initial Population (t = 0)
algal model 1
IF-THEN-ELSE $R_1(x_1,x_2,\ldots,x_i)$
algal model 2
IF-THEN-ELSE $R_2(x_1,x_2,\ldots,x_i)$
algal model N
IF-THEN-ELSE $R_N(x_1,x_2,\ldots,x_i)$
Structure Optimization (GP)
Reproduction
Crossover
Mutation
Next Generation t
model 1
model 2
model N
Parameter Optimization
Parameter Optimization
$t = t+1$
Fitness Evaluation
Output best algal model and predict the Abundance of Cyanobacteria
Yes
$t <$ MAXGENO
No

Figure 1. The conceptual diagram of HEA for evolving the IF-THEN-ELSE rule models.

A. *Structure Optimization of Rule Models*

Model representation

We use GP as the main technique to evolve the rule model structure, which typically operates on parse trees instead of bit strings in traditional GA. In our case, the rule model with IF-THEN-ELSE structure is represented as a vector of multiple trees in GP with the form of (Tree1, Tree2, Tree3) where Tree1 denotes the IF condition branch, Tree2 and Tree3 denote the result branches of THEN branch and ELSE branch respectively. Fig. 2 shows an example of a rule model for predicting the abundance of *Cylindrospermopsis*. The chromosome representation (Tree1, Tree2, Tree3) in GP for this model is illustrated in Fig. 3.

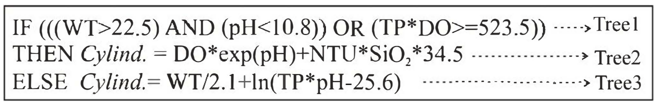
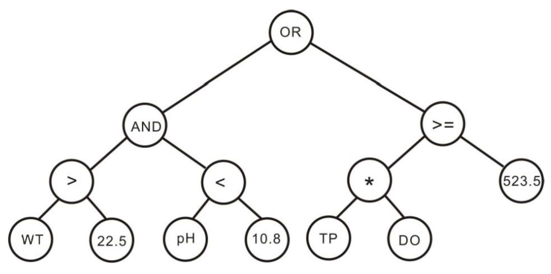
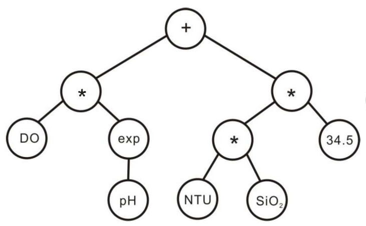

IF (((WT>22.5) AND (pH<10.8)) OR (TP*DO>=523.5)) $\cdots\cdots\rightarrow$ Tree1
THEN *Cylind.* = DO*exp(pH)+NTU*SiO_2*34.5 $\cdots\cdots\rightarrow$ Tree2
ELSE *Cylind.*= WT/2.1+ln(TP*pH-25.6) $\cdots\cdots\rightarrow$ Tree3

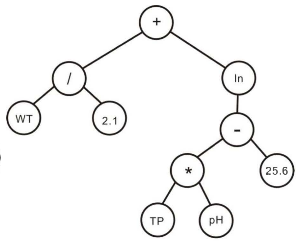
Figure 2. An example of a rule model for predicting the abundance of *Cylindrospermopsis.*



(a)

This work was supported by Australian Research Council (ARC Grant No: LP0990453) and two water industries in Australia (SA water and Seqwater).





(b) (c)

Figure 3. The chromosome representation of tree structure in GP for the rule model shown in Fig. 2: (a) Tree1 (IF branch); b) Tree2 (THEN branch); (c) Tree 3 (ELSE branch).

Genetic operators

We use the GP standard genetic operators of crossover and mutation to recombine the rules in the population and produce new models. The crossover is always performed between the same types of trees (Tree1/Tree2/Tree3). Firstly we do the crossover for Tree1, Tree2 and Tree3 in sequence. For each parent, randomly choose a node within its tree as a crossover point, swap the subtrees rooted at the crossover points and produce two new trees, then use either of them as the offspring trees. Afterwards we ensemble the three offspring trees for Tree1, Tree2 and Tree3 as a complete rule and take it as the crossover offspring of the two parents.

Similarly the mutation is performed on the Tree1, Tree2 and Tree3 of one parent in sequence. Whatever which type of the tree is performed, the mutation always begins by randomly selecting a node within the tree as the mutation point, replacing the subtree rooted at the mutation point with a randomly generated new subtree, thus producing an offspring tree. After the mutations for Tree1, Tree2 and Tree3 all are completed, we ensemble the three mutation offspring trees as a complete rule and take it as the mutation offspring of the parent.

Fitness evaluation

The whole data are divided into two parts: training data and testing data. To improve the robustness of the model, we use bootstrap method [20] to choose the training data points and testing data points randomly with a predefined ratio. The goodness of a model in the population is evaluated by its fitness value which is calculated as the root mean squared error (RMSE) between the measured training data and the predicted data.

It is noticeable that the training data set and testing data set can vary in each run due to the essence of the bootstrap sampling method. To make a fair comparison, we validate the best-evolved rule in each run on the whole data and calculate its total RMSE as the comparable value among different runs and different algorithms.

B. *Parameter Optimization of Rule Models*

In this study six population-based global optimization techniques are applied to optimize the random constants contained in a rule model and their performance and efficiency are compared based on the experimental results. Those methods include HC, SA, GA, DE, CMA-ES and EDA. Among those, except SA [21], [22] and GA [23], other methods, such as DE, CMA-ES and EDA, which are considered as the most competitive representatives of evolutionary computation, so far have not been applied to ecosystem models yet.

To make fair comparison, for each algorithm we define the stopping rule as when the total number of fitness evaluation *evalnum* reaches a maximum number N_{eval} (=1000). The population size for all the algorithms is set as 50. Below we give brief descriptions for each method as well as the settings of some key control parameters in the algorithm.

HC

Hill climbing (HC) originally is a local search technique which starts with a random (potentially poor) solution, and iteratively makes small changes to the solution, each time improving it a little. When the algorithm cannot see any improvement anymore, it terminates. In this study we modify the simple HC by enabling to do hill-climbing iteratively, each time with an initial starting point from the best solution in last run. The pseudo-code of HC is shown as follows. The neighbourhood size $N_{neighbors}$ in the algorithm is set as 50 in this study.

Pseudo-code of HC

```
BEGIN
  check all constants contained in current rule;
  evalnum ← 0;
  best rule ← current rule;
  best fitness ← fitness of current rule;
  while (evalnum < $N_{eval}$)
  {
    n ← 0;
    current rule ← best rule;
    current fitness ← best fitness;
    while(n < $N_{neighbors}$)
    {
      produce a temporary rule by performing
        Gaussian mutations on the constants
        randomly chosen from Tree1,Tree2 and
        Tree3 respectively;
      calculate the fitness of the temporary rule;
      evalnum ← evalnum + 1;
      if(temporary fitness is better than best
         fitness)
      {
        best fitness ← temporary fitness;
        best rule ← current rule;
      }
      n ← n + 1;
    }
  }
  output the best rule and best fitness;
END
```

SA

Simulated annealing (SA) is a random-search technique which exploits an analogy between the way in which a metal cools and freezes into a minimum energy crystalline structure (the annealing process) and the search for a minimum in a more general system. It forms the basis of an optimization technique for combinatorial and other problems. We set the three key control parameters in the algorithm as follows: initial temperature T_0 = 100, the epoch length L = 100 (the number of trials allowed at each temperature level), and the maximum allowable number of Markov chains K = 10000. The pseudo-code of SA is shown as follows:

Pseudo-code of SA

```
BEGIN
  check all constants contained in current rule;
  evalnum ← 0;
  best rule ← current rule;
  best fitness ← fitness of current rule;
  T ← $T_0$;
  frozen ← false;
  totaln ← 0;
  while((!frozen || totaln < K) && evalnum < $N_{eval}$)
  {
    cooling ← false;
    k ← 0;
    while( k < L)
    {
      produce a temporary rule by performing
        Gaussian mutations on the constants
        randomly chosen from Tree1, Tree2 and Tree3
        respectively;
      calculate the fitness of the temporary rule;
      evalnum ← evalnum + 1;
      ΔE ← temporary fitness - current fitness;
      if (ΔE < 0)
      {
        cooling ← true;
        current rule ← temporary rule;
        current fitness ← temporary fitness;
        best fitness ← temporary fitness;
        best rule ← current rule;
      }
      else if ($e^{\frac{(-1)*\Delta E}{T}}$ > random(0,1))
      {
        current rule ← temporary rule;
        current fitness ← temporary fitness;
      }
      k ← k + 1;
      totaln ← totaln + 1;
    }
    if (cooling)
      T ← 0.9 * T;
    else frozen ← true;
  }
  output the best rule and best fitness;
END
```

GA

A genetic algorithm (GA) [24] is a search heuristic that mimics the process of biological evolution and the mechanisms of natural selection and genetic variation. Suitable encodings are used to represent possible solutions to a problem, and the search is guided by using some genetic operators (crossover, mutation etc.) and the principle of "survival of the fittest". Due to the merits of self-adaptation, self-organization, self-learning, intrinsic parallelism and generality, GAs have been applied successfully in a wide range of economic, engineering and scientific computations [25]. The GA we use in this study is a simple and general GA whose effectiveness has been demonstrated in our previous studies [17], [26]. The novelty of the algorithm is the design of the multi-parent crossover. The pseudo-code of GA is shown as follows:

Pseudo-code of GA

```
BEGIN
  t ← 0;
  evalnum ← 0;
  randomly initialize a parameter population P(0);
  calculate the fitness of the individuals in P(0);
  evalnum ← evalnum + 1(for each calculation);
  while(evalnum < $N_{eval}$)
  {
    sort population P(t) in terms of fitness;
    randomly choose M individuals from P(t) to do
      multi-parent crossover;
    calculate the fitness of the new produced
      individual;
    evalnum ← evalnum + 1;
    if(new fitness is better than the worst fitness)
     replace the worst one with the new individual;
    P(t) ← P(t+1) ;
    t ← t + 1;
  }
  output the best rule and best fitness;
END
```

DE

Differential evolution (DE) is one of the most recent Evolutionary Algorithms (EAs) [27] for solving real-parameter optimization problems proposed by Price and Storn [28]. It is an effective, robust and simple global optimization algorithm which extracts the differential information (i.e., distance and direction information) from the current population of solutions to guide its further search. In this way no separate probability distribution has to be used which makes the scheme completely self-organizing. According to frequently reported comprehensive studies [29], [30], DE outperforms many other optimization methods in terms of convergence speed and robustness over common benchmark functions and real-world problems. The Pseudo-code of DE is shown as follows. The five DE schemes I used in the paper are DE-rand1, DE-rand2, DE-best1, DE-best2, and DE-randtobest1 (see [31] for more details).

Pseudo-code of DE

```
BEGIN
  t ← 0;
  evalnum ← 0;
  randomly initialize a parameter population P(0);
  calculate the fitness of the individuals in P(0);
  evalnum ← evalnum + 1(for each calculation);
  while(evalnum < $N_{eval}$)
  {
    sort population P(t) and get the best
      individual bestp(t);
    for(i =0; i<Popsize; i++)
    {
     produce a new offspring $p_i*(t)$ for individual
       $p_i$(t) by randomly choosing one of the five
       alternative DE schemes;
     calculate the fitness of $p_i*(t)$;
     evalnum ← evalnum + 1;
     if (fitness($p_i*(t)$) is better than fitness($p_i$(t))
       $p_i$(t+1) ← $p_i*(t)$;
     else $p_i$(t+1) ← $p_i$(t) ;
    }
    t ← t+1;
  }
  output the best rule and best fitness;
END
```

CMA-ES

The covariance matrix adaptation evolution strategy (CMA-ES) is one of the most powerful evolutionary algorithms for real-valued optimization [32], [33] with many successful applications (for an overview see [34]). The main advantages of the CMA-ES lie in its invariance properties, which are achieved by carefully designed variation and selection operators, and in its efficient (self-) adaptation of the mutation distribution. The general scheme of CMA-ES approach is shown as follows:

Main scheme of the CMA-ES approach

```
set λ; // number of samples per iteration, at least two, generally > 4
initialize m, σ, C = I, $p_\sigma$ = 0, $p_c$ = 0;
                  // initialize state variables
 while not terminate    // iterate
   for i in {1...λ} // sample λ new solutions and evaluate them
     $x_i$ = sample_multivariate_normal
          (mean = m, covariance_matrix = $\sigma^2 C$);
     $f_i$ = fitness($x_i$);
   $x_{1...\lambda}$ ← $x_{s(1)...s(\lambda)}$ with s(i)= argsort($f_{1...\lambda}$, i);
                         // sort solutions
   m' = m;              // we need later $m - m'$ and $x_i - m'$
   m ← update_m($x_1$, ..., $x_\lambda$);// move mean to better solutions
   $p_\sigma$ ← update_ps($p_\sigma$, $\sigma^{-1}C^{-1/2}$(m - m'));
                        // update isotropic evolution path
   $p_c$ ← update_pc($p_c$, $\sigma^{-1}$(m - m'),||$p_\sigma$||);
                        // update anisotropic evolution path
   C ← update_C(C, $p_c$,($x_1$ - m')/σ,...,($x_\lambda$ - m')/σ);
                        // update covariance matrix
   σ ← update_sigma(σ,||$p_\sigma$||);
                      // update step-size using isotropic path length
return m or $x_1$.
```

EDA

Estimation of distribution algorithm (EDA) [35], [36] is a relatively new branch of EAs. Unlike other EAs, EDAs do not use crossover or mutation. Instead, they explicitly extract global statistical information from the selected solutions and build a probability model of promising solutions based on the extracted information. New solutions are sampled from the model thus built and fully or in part replace solutions in the current population. The general scheme of the EDA approach for continuous domains is shown as follows:

Main scheme of the EDA approach

```
$D_0$ ← Generate M individuals (the initial
      population) randomly;
l ← 1;
do
{
  $D^{se}_{l-1}$ ← Select N ≤M individuals from $D_{l-1}$
    according to a selection method;
  $p_l(x) = p(x|D^{se}_{l-1})$ ← Estimate the probability
    distribution of an individual being among the
    selected individuals;
  $D_l$ ← Sample M individuals (the new population)
    from $p_l(x)$;
  l ← l + 1;
}
until Termination criterion is met.
```

Different continuous EDAs have been proposed for the global continuous optimization problem [37]-[40]. In this study we use an improved IDEA (Iterated Density Evolutionary Algorithm) [40] called AMaLGaM-IDEA (Iterated Density Evolutionary Algorithm with Adapted Maximum-Likelihood Gaussian Models) [41] which uses the combination of SDR (Standard-Deviation Ratio), AVS (Adaptive Variance Scaling), and AMS (Anticipated Mean Shift) to adaptively change both the covariance and the mean-shift to prevent inefficient sampling. In addition the truncation selection, rejection sampling and elitist replacement are used in the algorithm.

Parameter set encoding and fitness evaluation

For all above six parameter optimization (PO) algorithms, the encoding of the parameter set is the same. Regarding a specific rule model, we first check all the constants contained in the Tree1, Tree2 and Tree3, including counting the number of constants l and recording their positions. Each individual in the parameter population can then be represented as an l-dimensional row vector (c_1, c_2, …, c_l) where each component c_i for i =1, 2, …, l is encoded as a floating number and generated randomly ranging from 0 to a predefined maximum integer during the initialization of the parameter population.

Before the fitness evaluation of an individual in the parameter population, we first return to the original rule model and replace all constants with the corresponding components of the row vector (i.e. the individual) and then follow the same procedure as in Section II (A) to calculate the fitness.

III. EXPERIMENTS

A. *Study Data*

In this study we use the *Cylindrospermopsis* abundance data from Wivenhoe Reservoir in Queensland (Australia) to test and compare the performance of different parameter optimization algorithms based on hybrid evolutionary modelling framework. A number of limnological variables have been collected over an 11-year period (1998–2009) as shown in Table I. A simple linear interpolation has been used to fill missing values to produce a complete daily time series for this period. In order to develop 7-days-ahead predictive models for *Cylindrospermopsis* abundance by using our modelling algorithm, we shift the interpolated daily data for 7 days and use as input data.

TABLE I. PARAMETERS USED AS INPUT AND OUTPUT VARIABLES IN THE EVOLUTIONARY MODELLING WITH HEA

Division	**Variables**	**Unit**	**Mean ± STDEV**
Input Variables	Electrical Conductivity (EC)	uS/cm	354.90±72.64
	Turbidity (NTU)	NTU	12.92±56.76
	Water Temperature (WT)	°C	22.66±3.70
	Dissolved Oxygen (DO)	mg/L	8.67±1.25
	pH (pH)		8.21±0.32
	Reactive SiO_2 (SiO_2)	mg/L	3.24±2.69
	Total Nitrogen (TN)	mg/L	0.48±0.078
	Total Phosphorus (TP)	mg/L	0.019±0.012
Output Variable	*Cylindrospermopsis (Cylind.)*	cells/mL	10108±20698

B. *Paramter Settings and Measures*

All the experiments were performed on a Hydra supercomputer (IBM eServer 1350 Linux) provided by eResearch SA with a peak speed of 1.2 TFlops by using the programming language C++. The GP parameter settings of HEA for structure optimization are: Popsize = 100, maximum tree depth = 4, maximum number of generations = 80. Besides total RMSE, we also calculate the total R^2 value between the predicted and measured values on whole data set to measure the algorithm performance.

C. *Results and Discussions*

Table II shows the statistical results of six PO algorithms in 100 runs. In terms of the RMSE and R^2 values, the rank of the average performance of those six algorithms is: HC, SA, DE, GA, EDA, and CMA-ES. In terms of computational efforts, HC and CMA-ES are the two most time-consuming algorithms, whereas the average running time for other four algorithms are very close.

TABLE II. THE STATISTICAL RESULTS OF SIX PO ALGORITHMS IN 100 RUNS

No.	**Algo.**	**Total RMSE (Mean±STDEV)**	**Total R^2**	**Run-Time (h)**
A1	HC	14320.77±1280.43	0.53	3.3
A2	SA	14467.44±915.88	0.51	2.8
A3	GA	15674.24±573.03	0.43	2.4
A4	DE	15217.4±510.57	0.46	2.6
A5	CMA-ES	18772.74±846.83	0.19	3.5
A6	EDA	16373.12±1005.84	0.38	2.7

Table III shows the best rule models obtained by six PO algorithms in 100 runs. It is interesting to notice that all those rule models clearly indicate the threshold values of some water quality parameters (i.e. WT, TP, NTU, SiO_2, DO, EC) or their ratios (i.e. SiO_2/pH) in the IF condition branch favoured by the growth of *Cylindrospermopsis*. This demonstrates that the HEA has the most advantage of automatically discovering the understandable IF-THEN-ELSE rules hidden in the measured data and can be regarded as a powerful intelligent data mining tool for ecological data.

Among the six best models found by different PO algorithms, the HC and SA models are significantly superior to other models in terms of their much lower RMSE and higher R^2 value (>0.65). The secondary are the DE, EDA and GA models and their RMSE and R^2 values are very close. It seems that the CMA-ES model is the worst one with the largest RMSE (>15000) and the lowest R^2 value (=0.46).

TABLE III. THE BEST MODELS OBTAINED BY SIX PO ALGORITHMS IN 100 RUNS

Algo.	**Best model**

HC	IF((SiO_2<6.41)AND((WT*SiO_2)>126.63)AND(EC>=257.98)
	AND(NTU>2.37))
	THEN *Cylind.* = (2*NTU+exp(pH)-498.51)*16.65
	ELSE *Cylind.* = (WT-ln(|TN|)-21.73)*(WT-TN/DO-20.54)*
	(NTU*5.38+296.45)
	(RMSE = 11566.96 R^2= 0.69)

SA	IF ((TP<=0.028)AND((SiO_2/pH)<=0.73))
	THEN *Cylind.* = (NTU+exp(SiO_2)+WT*15.93-355.93)*
	(WT*21.91-445.93)
	ELSE *Cylind.* = (2*WT+152.23)*(WT*19.64-452.54)
	(RMSE = 12053.33 R^2= 0.66)

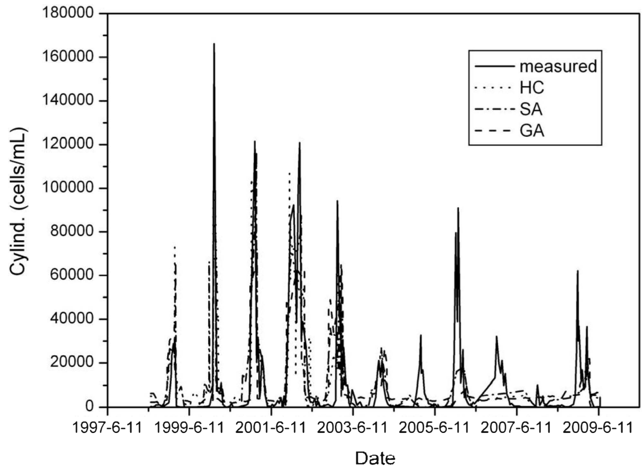
GA	IF ((SiO_2<=6.42)AND((NTU>=310.7)OR(WT>=25.71)))
	THEN *Cylind.* = ((((WT*pH)+(SiO_2*44.84))*20.76)*SiO_2)
	ELSE *Cylind.* = WT*(EC*TN+4.01)
	(RMSE = 14134.61 R^2 = 0.53)

DE	IF ((NTU<=268.45)AND(((WT*NTU)<=66.42)OR (WT<=25.82)))
	THEN *Cylind.* = (WT+9.31)*pH*WT
	ELSE *Cylind.* = SiO_2*10059.73
	(RMSE = 13860.57 R^2 = 0.55)

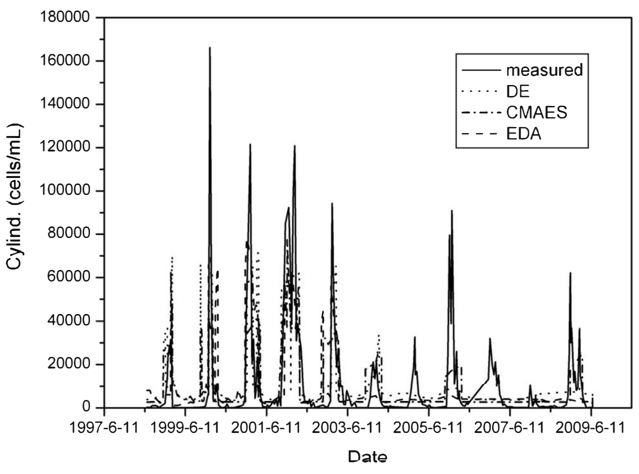
CMA-ES	IF (WT*ln(|WT|)<=82.6)
	THEN *Cylind.* =NTU*96.25+2612.5
	ELSE *Cylind.* = (WT+NTU+WT*SiO_2+78.5)*131.27
	(RMSE = 15234.23 R^2 = 0.46)

EDA	IF ((DO<4.46)OR(DO>9.57)OR(SiO_2<=4.46)OR ((WT-SiO_2)<19.19))
	THEN *Cylind.* = exp(pH)
	ELSE *Cylind.* = exp(pH)*14.74-2*exp(DO)
	(RMSE = 14029.40 R^2 = 0.54)

The modelling results of the six best models validated on the whole data are illustrated in Fig. 4. One feature of the *Cylindrospermopsis* measured data is that there are regular summer peaks annually during the years from 1998 to 2009. Among those peaks, the most prominent ones are the six experiencing in the years of 2000, 2001, 2002, 2003, 2006 and 2009 respectively. It is encouraging result that for all the six models, the predicted timings of these prominent peaks correspond very well with the measured data, which coincides with our aim of giving early warning of cyanobacterial blooms by developing these models. Desirably both the HC and SA models can forecast the magnitudes of the summer peaks in successive four years from 2000 to 2003 very well, whose predicted values almost overlap with the measured data. In contrast, other models under-estimate these peaks in most cases. Due to the fact that all the six models fail to predict their magnitudes reasonably, it seems challenging work to predict the summer peaks in 2006 and 2009.



(a)



(b)

Figure 4. The validation results on whole data for the best models in 100 runs using six PO algorithms: (a) HC, SA, GA; (b) DE, CMA-ES, EDA.

Finally the results of the experiments are analysed using a one-way ANOVA test combined with Tukey's HSD Post Hoc Test for multiple comparisons using a 95% confidence level as suggested by [42]. The data we use for statistical analysis are the total RMSE values in 100 runs for six PO algorithms. The calculated F-value is 337.4 which is significant compared with the threshold value 2.2899. The statistical results for Tukey's HSD Post Hoc Test are shown in Table IV. From the table, we conclude the dominance order of six PO algorithms performed on the *Cylindrospermopsis* data is:

HC→SA→DE→GA→EDA→CMA-ES.

TABLE IV. RESULTS FOR TUKEY'S HSD POST HOC TEST

Pairs	**(A1, A2)**	**(A1, A3)**	**(A1, A4)**	**(A1, A5)**	**(A1, A6)**
Q-value	-1.63	-15.13	-10.02	-49.79	-22.95
Signi.? (Y/N)	N	Y	Y	Y	Y
Order	–	A1→A3	A1→A4	A1→A5	A1→A6
Pairs	**(A2, A3)**	**(A2, A4)**	**(A2, A5)**	**(A2, A6)**	**(A3, A4)**
Q-value	-13.50	-8.39	-48.16	-21.32	5.11
Signi.? (Y/N)	Y	Y	Y	Y	Y
Order	A2→A3	A2→A4	A2→A5	A2→A6	A4→A3
Pairs	**(A3, A5)**	**(A3, A6)**	**(A4, A5)**	**(A4, A6)**	**(A5, A6)**
Q-value	-34.66	-7.82	-39.77	-12.93	26.84
Signi.? (Y/N)	Y	Y	Y	Y	Y
Order	A3→A5	A3→A6	A4→A5	A4→A6	A6→A5
Total Order	A1→A2→A4→A3→A6→A5				

IV. CONCLUSIONS AND FUTURE WORK

This paper proposes a new methodology in freshwater ecosystem modelling. It mainly includes the following three main features:

(1) The explainable single rules with IF-THEN-ELSE structure are developed to model the abundance of *Cylindrospermopsis*;

(2) A hybrid evolutionary algorithm (HEA) is designed to perform structural and parameter optimization of rule models simultaneously;

(3) Six population-based algorithms are applied to perform the parameter optimization (PO) task within the hybrid methodology including HC, SA, GA, DE, CMA-ES, and EDA.

The effectiveness of the methodology is tested on the *Cylindrospermopsis* abundance data from Wivenhoe Reservoir in Queensland (Australia). Six different parameter optimization (PO) methods are applied to each test data set and their results are compared and analysed. Based on the experimental results, we draw some conclusions as follows:

(a) Whatever which PO method is applied in the hybrid EA, our modelling algorithm can always find multiple IF-THEN-ELSE rules, which not only clearly indicate various environmental conditions favoured by *Cylindrospermopsis* but also are capable of forecasting their abundances reasonably. More interestingly, our algorithm is able to propose a variety of alternative rule models with distinctive IF condition branches. This result is very encouraging as it could help ecologists to design new experiments in the lab to testify the competing hypothesis (models) which the human brains can hardly imagine.

(b) Both the experiments and their statistical analysis suggest that when comparing different PO methods, HC and SA always perform the best. The secondary is DE, GA and EDA, which are three comparable methods, whereas CMA-ES performs the worst. This result looks somewhat surprising. We surmise that this is mainly caused by the high complexity and uncertainty of model parameters contained in a rule model. Unlike some benchmark functions, the number of the random constants in a rule is arbitrary and their value ranges can be largely different. Hence it is hard for some sophisticated methods with complex mathematical computation (i.e. CMA-ES) to capture the data relativity and find the global optima with small population size. We observed that the default population size λ in CMA-ES is less than 10 in most cases of our study. On the contrary, some classical optimization techniques, like HC and SA, have the advantage to explore the irregular search space by multiple starting points and iterative random search. They seem to work better when optimizing the random constants with arbitrary number for these rule models which are evolved based on the real-world ecological data.

In order to further improve the predictive accuracy of rule models we consider two possibilities to extend the single rule model structure in our future work. One is to use an ensemble of multiple single rules to model the algal abundance. Another is to design a more complicated rule set model with multi-level embedded IF-THEN-ELSE structures. It will be interesting to compare the advantages and disadvantage of these three different model structures: a single rule, an ensemble of multiple single rules, and a rule set with embedded single rules, when performing on ecological data from real-world freshwater ecosystems.

REFERENCES

[1] Gustaaf M. Hallegraeff, "Harmful algal blooms in the Australian region," Marine pollution bulletin, vol. 25, no. 5-8, pp. 186-190, 1992.

[2] T. H. Donnelly, M. R. Grace, and B. T. Hart, "Algal blooms in the Darling-Barwon River, Australia." Water, Air, and Soil Pollution, vol. 99, no. 1-4, pp. 487-496, 1997.

[3] S. E. Shumway, "A review of the effects of algal blooms on shellfish and aquaculture." Journal of the World Aquaculture Society, vol. 21, no. 2, pp. 65-104, 1990.

[4] S. J. Pittman and K. M. Pittman, "Short-term consequences of a bentic cyanobacterial bloom (Lyngbya majuscule Gomont) for fish and penaeid prawns in Moreton Bay (Queesland, Australia)," Estuarine Coastal and Shelf Science, vol. 63, pp. 619-632, 2005.

[5] Atech Group, "Cost of algal bloom," LWRRDC Occasional Paper 26/99. Land and Water Resources Research and Development Corporation, Canberra, ACT, Australia, pp.1-42, 2000.

[6] R. A. Park, "A generalized model for simulating lake ecosystems," Simulation, vol. 23, no. 2, pp. 33-50, 1974.

[7] J. Benndorf and F. Recknagel, "Problems of application of the ecological model SALMO to lakes and reservoirs having various trophic states," Ecological Modelling, vol. 17, no. 2, pp. 129-145, 1982.

[8] H. Pei and J. Ma, "Study on the algal dynamic model for West Lake, Hangzhou," EcologicalModelling, vol. 148, pp. 67-77, 2002.

[9] B. Wei, N. Sugiura, and T. Maekawa, "Use of artificial neural network in the prediction of algal blooms," Water Research, vol. 35, no. 8, pp. 2022-2028, 2001.

[10] M. Scardi, "Advances in neural network modelling of phytoplankton primary production," Ecological Modelling, vol. 146, pp. 33-45, 2001.

[11] Q. Chen and A. Mynett, "Integration of data mining techniques and heuristic knowledge in fuzzy logic modelling of eutrophication in Taihu Lake," Ecological Modelling, vol. 162, pp. 55-67, 2003.

[12] H. R. Maier, T. Sayed, and B. J. Lence, "Forecasting cyanobacterium Anabaena spp. in the River Murray, South Australia, using B-spline neurofuzzy models," Ecological Modelling, vol. 146, pp. 85-96, 2001.

[13] T. Back, U. Hammel, and H.-P. Schwefel, "Evolutionary computation: comments on the history and current state," IEEE Trans. Evol. Comput., vol. 1, no. 1, pp. 5-16, 1997.

[14] H. Cao, F. Recknagel, B. Kim, and N. Takamura, "Hybrid evolutionary algorithm for rule set discovery in time-series data to forecast and explain algal population dynamics in two lakes different in morphometry and eutrophication," in F. Recknagel, editor, Ecological Informatics, 2nd Edition, Chapter 17, 2006, Springer-Verlag, Berlin, pp. 347-367.

[15] A. Welk, F. Recknagel, H. Cao, W. S. Chan, and A. Talib, "Rule-based agents for forecasting algal population dynanics in freshwater lakes discovered by hybrid evolutionary algorithms." Ecological Informatics, vol. 3, pp. 46-54, 2008.

[16] W. S. Chan, F. Recknagel, H. Cao, and H. D. Park, "Elucidation and short-term forecasting of Microcystin concentrations in Lake Suwa (Japan) by means of artificial neural networks and evolutionary algorithms," Water Research, vol. 41, pp. 2247-2255, 2007.

[17] J. Yu, H. Cao, Y. Chen, L. Kang, and H. Yang, "A new approach to estimation of the electrocrystallization parameters," Journal of Electroanalytical Chemistry, vol. 474, no. 1, pp. 69-73, 1999.

[18] J. R. Koza, Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press: Cambridge, MA, 1992.

[19] J. R. Koza, Genetic Programming II: Automatic Discovery of Reusable Programs. MIT Press: Cambridge, MA, 1994.

[20] A. C. Davison and D. V. Hinkley, Bootstrap Methods and Their Application. Cambridge University Press, 1997.

[21] G. C. Hurtt and R. A. Armstrong, "A pelagic ecosystem model calibrated with BATS and OWSI data," Deep-Sea Research, vol. 46, pp. 27–61, 1999.

[22] R. J. Matear, "Parameter optimization and analysis of ecosystem models using simulated annealing: a case study at station P," Journal of Marine Research, vol. 53, pp. 571–607, 1995.

[23] J. J. Vallino, “Improving marine ecosystem models: Use of data assimilation and mesocosm experiments,” Journal of Marine Research, vol. 58, pp. 117–164, 2000.

[24] M. Mitchell, An Introduction to Genetic Algorithm, MIT Press, Cambridge, MA, 1996.

[25] D. E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning. Addison Wesley, Reading, MA, 1989.

[26] Jingxian Yu, Hongqing Cao, and Yanbin He, “A new tree structure coding for equivalent circuit and evolutionary estimation of parameters,” Chemometrics and Intelligent Laboratory Systems, vol. 85, no. 1, pp. 27-39, 2007.

[27] T. Bäck, D. B. Fogel, and Z. Michalewicz, Eds., Evolutionary Computation 2: Advanced Algorithms and Operators. Bristol, U.K.: Institute of Physics, 2000.

[28] R. Storn and K. Price, “Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces,” Journal of Global Optimization, vol.11, pp. 341-359, 1997.

[29] O. Hrstka and A. Ku?cerová, “Improvement of real coded genetic algorithm based on differential operators preventing premature convergence,” Advance in Engineering Software, vol. 35, pp. 237–246, 2004.

[30] J. Vesterstroem and R. Thomsen, “A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems,” in Proc. Congr. Evol. Comput., vol. 2, 2004, pp. 1980–1987.

[31] R. Storn, “On the usage of differential evolution for function optimization,” in Proc. Biennial Conf. North Amer. Fuzzy Inf. Process. Soc., 1996, pp. 519–523.

[32] Hansen, N., S. D.M¨uller, and P. Koumoutsakos, “Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES),” Evolutionary Computation, vol. 11, no. 1, pp. 1–18, 2003.

[33] N. Hansen, “The CMA Evolution Strategy: A Comparing Review,” in J.A. Lozano, P. Larrañaga, I. Inza and E. Bengoetxea (Eds.). Towards a New Evolutionary Computation. Advances in Estimation of Distribution Algorithms. Springer, 2006, pp. 75-102.

[34] N. Hansen, References to CMA-ES Applications, 2005. Available: http://www.bionik.tu-berlin.de/user/niko/cmaapplications.pdf.

[35] H. Mühlenbein and G. Paaß, “From recombination of genes to the estimation of distributions I. Binary parameters,” in Lecture Notes in Computer Science 1411: Parallel Problem Solving from Nature – PPSN IV, 1996. pp. 178-187.

[36] P. Larran˜aga and J.A. Lozano (eds.), Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation, Kluwer Academic Publishers, Boston, MA, 2002.

[37] P. Larra˜naga, R. Etxeberria, J. A. Lozano, and J. M. Pe˜na, “Optimization in continuous domains by learning and simulation of Gaussian networks,” in Proceedings of the 2000 Genetic and Evolutionary Computation Conference Workshop Program, 2000, pp. 201–204.

[38] S. Rudlof and M. K¨oppen, “Stochastic hill climbing with learning by vectors of normal distributions,” in Proceedings of the First On-line Workshop on Soft Computing (WSC1), Nagoya, Japan, 1996, pp. 60–70.

[39] M. Pelikan and H. M¨uhlenbein, “The bivariate marginal distribution algorithm,” in R. Roy, T. Furuhashi, and P. K. Chawdhry (editors), Advances in Soft Computing - Engineering Design and Manufacturing , London, Springer, 1999, pp. 521–535.

[40] P. A. N. Bosman, and D. Thierens, “Expanding from discrete to continuous estimation of distribution algorithms: The IDEA,” in Lecture Notes in Computer Science 1917: Parallel Problem Solving from Nature – PPSN VI, 2000, pp. 767-776.

[41] P. A. N. Bosman, J. Grahl, and D. Thierens, “Adapted maximum-likelihood Gaussian models for numerical optimization with continuous EDAs,” Centre for Mathematics and Computer Science (CWI), Amsterdam, The Netherlands, Tech. Rep. SEN-E0704, Dec. 2007.

[42] P. L. Lanzi, D. Loiacono, S. W. Wilson, and D. E. Goldberg DE, “Prediction update algorithms for XCSF:RLS, kalman filter, and gain adaptation,” in Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation, New York, USA, 2006, pp. 1505–1512.