http://doi.org/10.35784/iapgos.5684

received: 15.12.2023 | revised: 21.03.2024 | accepted: 19.06.2024 | available online: 30.06.2024

AN ADAPTIVE DIFFERENTIAL EVOLUTION ALGORITHM WITH A BOUND ADJUSTMENT STRATEGY FOR SOLVING NONLINEAR PARAMETER IDENTIFICATION PROBLEMS

Watchara Wongsa, Pikul Puphasuk, Jeerayut Wetweerapong
Khon Kaen University, Faculty of Science, Department of Mathematics, Khon Kaen, Thailand

Abstract. *Real-world parameter identification problems require determining the bounds that cover the unknown solutions. This paper presents an adaptive differential evolution algorithm with a bound adjustment strategy (ADEBAS) for solving nonlinear parameter identification problems. The adjustment strategy detects the parameter-bound violations of mutant vectors during the evolution process and gradually extends the bounds. The algorithm adaptively uses two mutation strategies and two ranges of crossover rate to balance the population diversity and convergence speed. Experimental results show that ADEBAS can solve 24 nonlinear regression tasks from the National Institute of Standards and Technology benchmark with accurate estimation and reliability. It also outperforms the compared methods on real-world parameter identification problems.*

Keywords: parameter identification, differential evolution algorithm, bound adjustment strategy

ADAPTACYJNY RÓŻNICZKOWY ALGORYTM EWOLUCYJNY ZE STRATEGIĄ DOSTOSOWYWANIA GRANIC DO ROZWIĄZYWANIA NIELINIOWYCH PROBLEMÓW IDENTYFIKACJI PARAMETRÓW

Streszczenie. *Problemy identyfikacji parametrów w świecie rzeczywistym wymagają określenia granic, które pokrywają nieznane rozwiązania. W artykule przedstawiono adaptacyjny różniczkowy algorytm ewolucyjny ze strategią dostosowywania granic (ADEBAS) do rozwiązywania nieliniowych problemów identyfikacji parametrów. Strategia dostosowywania wykrywa naruszenia granic parametrów zmutowanych wektorów podczas procesu ewolucji i stopniowo rozszerza granice. Algorytm adaptacyjnie wykorzystuje dwie strategie mutacji i dwa zakresy szybkości krzyżowania, aby zrównoważyć różnorodność populacji i szybkość zbieżności. Wyniki eksperymentów pokazują, że ADEBAS może rozwiązać 24 zadania regresji nieliniowej z benchmarku National Institute of Standards and Technology z dokładnym oszacowaniem i niezawodnością. Przewyższa również porównywane metody w rzeczywistych problemach identyfikacji parametrów.*

Słowa kluczowe: parameter identification, differential evolution algorithm, bound adjustment strategy

Introduction

The parameter identification problems aim to determine the unknown parameters of the mathematical models that fit the observed data. Their applications include the parameter estimation of nonlinear models [3, 12], robot dynamics [6, 8], solar photovoltaic modeling [1, 5], and nonlinear dynamic systems [16, 23]. The problems minimize the residual sum of squares between observed and approximated values. Since the objective functions are nonlinear and multimodal and have many local optima, gradient-based methods are unsuitable for determining the globally optimal parameters. Thus, global optimization algorithms that explore the search space to find the best solutions are applied to solve the problems. These methods include Genetic algorithm (GA) [22], Particle swarm optimization (PSO) [9], Simulated annealing (SA) [4], and Differential evolution algorithm (DE) [18]. In practice, the problems require the parameter bounds that cover the solutions. To set these bounds, users must know the behavior of the models. Our research presents a bound adjustment strategy by setting small initial bounds and gradually extending them. We combine this approach with the adaptive DE algorithm to solve the National Institute of Standards and Technology (NIST) nonlinear regression benchmark and real-world parameter identification applications consisting of photovoltaic (PV) models [17, 27] and soil water retention models [2, 24, 26]. The PV models are essential for developing photovoltaic devices that generate electricity from solar energy, and the soil water retention model involves studying several hydraulic processes in soils for calculating water content values. These models use mathematical equations with input data and unknown parameters. Accurate parameter estimation is crucial for designing and optimizing the systems.

The remainder of the paper is organized as follows. Section 1 reviews related work about methods for solving parameter identification problems. Section 2 describes the proposed DE with a bound adjustment strategy and adaptive mutation and crossover strategies (ADEBAS). Section 3 presents the performance of ADEBAS on NIST nonlinear regression problems. Section 4 compares the performance of ADEBAS with those of other methods on real-world parameter identification problems. Section 5 provides insight discussion. Finally, the last section gives the conclusion.

1. Literature review

This section reviews the classic DE algorithm and the population-based global optimization methods for solving continuous functions and parameter identification problems.

1.1. Differential evolution algorithm

Storn and Price [18] presented the differential evolution algorithm (DE) for continuous optimization. DE's concept includes mutation, crossover, and selection operations. The algorithm initials NP random population vectors $x_i = [x_{ij}]$ for $i = 1, 2, ..., NP$ and $j = 1, 2, ..., D$ where D is the dimension of the search space and indicates the best vector x_b and the best value f_b. The mutation operation randomly selects three population vectors to generate the mutant vector x_m for each target vector x_i by

$$x_m = x_{r_1} + F \cdot (x_{r_2} - x_{r_3}) \tag{1}$$

where r_1, r_2, and r_3 are distinct indices and different from index i and F is the scaling factor. Next, the crossover operation creates the trial vector x_c by exchanging some components between target and mutant vectors with the crossover rate CR as follows:

$$x_{c_j} = \begin{cases} x_{m_j} & rand_j < CR \text{ or } j = IC \\ x_j & otherwise \end{cases} \tag{2}$$

where $rand_j$ is a uniform random number in [0,1] for each

$j = 1, 2, ..., D$ and IC is a randomly fixed index from 1 to D. Then, the selection operation compares the fitness values of x_c and x_i. The vector x_c replaces x_i when x_c is better than x_i. The algorithm also updates x_b and $f(x_b)$. These three operations repeat until reaching the stopping condition.

1.2. The population-based global optimization methods

Price [14] proposed the controlled random search (CRS) for minimization problems. The algorithm initials population vectors and creates a candidate solution vector x_c by randomly choosing population vectors $x_1, x_2, ..., x_{n+1}$ and reflects the vector x_{n+1} through the centroid of x_1 to x_n. The vector x_c replaces the worst population vector when it is better. This process repeats until reaching the stopping condition. CRS gives solutions close to global minimums and uses a small number of function evaluations compared with the Becker and Lago method on four benchmark problems. Tvrdík et al. [19] improved the CRS algorithm by combining the classic mutation from the DE algorithm to solve nonlinear regression problems. The algorithms give more successful results than the Levenberg-Marquardt method on the NIST dataset.

Puphasuk and Wetweerapong [15] presented the DEASC algorithm that improves DE by using crossover rates in the ranges of $[0, 0.1]$ and $[0.9, 1]$ according to the success in generating a better solution in the selection. The algorithm randomly uses the scaling factor within $[0.5, 0.7]$. The results show that the algorithm can solve eight benchmark problems reliably. Wang et al. [21] proposed the JAYA algorithm for parameter estimation of the soil water retention model. The algorithm uses the worst and best population vectors to generate a new vector. Experimental results show that it is more accurate than DE and PSO methods. Li et al. [10] proposed an adaptive DE called MADE to estimate the parameters of photovoltaic models. The algorithm combines SHADE with the Nelder – Mead simplex method and uses the mutation strategy from JADE. Experimental results show that MADE performs better than the seven compared methods. Hu et al. [7] presented the reinforcement learning-based differential evolution (RLDE) that mixes the DE algorithm with a reinforcement learning strategy. The strategy learns to choose the scaling factor values within $[0.1, 0.9]$. RLDE performs well in terms of accuracy and reliability when compared with nine algorithms on parameter estimation problems of different PV models. Liang et al. [11] proposed the self-adaptive ensemble-based differential evolution called SEDE to estimate the parameters of solar photovoltaic models. SEDE uses three mutation strategies depending on the periods of function evaluations and randomly chooses three combinations of scaling factor and crossover rate values. The results obtained by SEDE are more accurate and stable than the 15 compared algorithms. Wang et al. [20] proposed an adaptive DE called HDE for parameter estimation of solar photovoltaic models. The algorithm uses four pairs of the scaling factor and crossover rate values with the learning rate. HDE utilizes two mutation strategies. Experimental results show that HDE overall outperforms SEDE and other compared methods.

2. Adaptive differential evolution algorithm with a bound adjustment strategy (ADEBAS)

We propose an adaptive differential evolution algorithm with a bound adjustment strategy (ADEBAS) for solving nonlinear parameter identification problems. The ADEBAS algorithm uses three strategies: parameter bound adjustment, adaptive mutation, and adaptive crossover strategies, which can be described as follows.

2.1. Bound adjustment strategy

The algorithm initials the parameter bound in a small range and gradually extends the bounds when needed. This study experiments with using $[0,1]$ and $[0, 0.1]$ as initial bounds for general nonlinear regression problems. For real-world applications, the small initial bounds must satisfy the conditions of the model parameters. For each dimension $j = 1, 2, ..., D$, the counters CL_j and CU_j count the bound violations of a mutant vector x_m when the j^{th} component x_{m_j} is out of bounds. The strategy extends lower or upper bounds L_j or U_j for each generation by $L_j = -CL_j$ when $x_{m_j} < L_j$ or $U_j = CU_j$ when $x_{m_j} > U_j$. Note that if an increment of a counter is large for the current generation, it usually will be small for the next generations.

2.2. Adaptive mutation strategy

The strategy generates the scaling factor F in the range $[0.5, 0.7]$ for creating a mutant vector using classic or sorting mutations according to the probabilities $pm1$ and $pm2$ calculated from their success in selection. The probabilities are initialed to 0.5 and are updated by the weighted formula of success counters. The classic mutation is expressed as eq. (1). The sorting mutation is as follows:

$$x_m = x^*_{r_1} + F \cdot (x^*_{r_2} - x^*_{r_3}) \tag{3}$$

where $f(x^*_{r_1}) \le f(x^*_{r_2}) \le f(x^*_{r_3})$. The $x^*_{r_1}$, $x^*_{r_2}$, and $x^*_{r_3}$ are sorted vectors of x_{r_1}, x_{r_2}, and x_{r_3} by function values.

2.3. Adaptive crossover strategy

The strategy creates the crossover rate CR in the ranges of $[0, 0.1]$ and $[0.9, 1]$ for generating a trial vector according to the probabilities $pc1$ and $pc2$ calculated from their success in selection. The probabilities are initialed to 0.5 and are updated by the weighted formula of success counters. A trial vector x_c is generated by

$$x_{c_j} = \begin{cases} x_{m_j} & rand_j < CR \text{ or } j = IC \\ x_j & otherwise \end{cases} \tag{4}$$

where $rand_j$ is a uniform random number in $[0,1]$ for each $j = 1, 2, ..., D$ and IC is a randomly fixed index from 1 to D. The pseudo-code of the ADEBAS algorithm is shown in Algorithm 1.

3. Performance of ADEBAS on NIST nonlinear regression problems

The accuracy and reliability of algorithms are tested on the nonlinear regression problems from the National Institute of Standards and Technology (NIST) dataset [28]. It includes the number of parameters, the number of observations, and the reported solutions with 11 decimal places. The accuracy λ of an algorithm is calculated by

$$\lambda = \begin{cases} 0 & \dfrac{|f(x_b) - c|}{c} \ge 1 \\ 11 & \dfrac{|f(x_b) - c|}{c} \le 10^{-11} \\ -\log(\dfrac{|f(x_b) - c|}{c}) & otherwise \end{cases} \tag{5}$$

where $f(x_b)$ denotes the best value obtained from the algorithm and c denotes the certified value for a problem. We select 24 benchmark problems of varying difficulty levels from the NIST dataset to evaluate the performance of the proposed method.

This experiment evaluates two small initial bounds for the ADEBAS algorithm to solve NIST problems. The ADEBAS1 and ADEBAS2 algorithms use the initial bounds $[0,1]$ and $[0,0.1]$, respectively. They set the population size $NP = 10D$, the initial counters of $nm1$, $nm2$, $nc1$, and $nc2$ to 0, and stopping condition $\log(f(x_w)/f(x_b)) < 10^{-10}$ or the maximum number of function evaluations $maxnf = 80000D$. Each algorithm is run 100 times for each problem. We record and report the averages of parameter bounds L_j, U_j for each j, the successful run NS, the average accuracy $Mean\lambda$, and the average number of function evaluations $Meannf$ that the algorithm gives $\lambda > 4$ before exceeding $maxnf$.

Algorithm 1. *ADEBAS algorithm*

```
1:  Set the control parameters $pm1, nm1, nm2, pc1, nc1, nc2$
2:  Set the initial bounds $L_j$, $U_j$ in a small range and the counters $CL_j$, $CU_j$
3:  Generate population vectors $x_i$ for $i = 1,2,...,NP$ and find the best vector $x_b$ and its best value $f(x_b)$
4:  Set the number of function evaluations $nf = 0$
5:  **While** stopping condition is not satisfied **do**
6:      Set the switches $sl = 0$ and $su = 0$
7:      **for** $i = 1:NP$ **do**
8:          Random a scaling factor $F$ in $[0.5, 0.7]$
9:          **if** $rand(0,1) < pm1$ **then** generate a mutant vector $x_m$ by eq. (1)
10:         **else** generate a mutant vector $x_m$ by eq. (3)
11:         **end if**
12:         **for** $j = 1:D$ **do**
13:             **if** $x_{m_j} < L_j$ **then** $CL_j = CL_j + 1$
14:                 **if** $sl = 0$ **then** $L_j = -CL_j$ and set $sl = 1$
15:                 **end if**
16:                 $x_{m_j} = L_j + (U_j - L_j) \cdot rand(0,1)$
17:             **end if**
18:             **if** $x_{m_j} > U_j$ **then** $CU_j = CU_j + 1$
19:                 **if** $su = 0$ **then** $U_j = CU_j$ and set $su = 1$
20:                 **end if**
21:                 $x_{m_j} = L_j + (U_j - L_j) \cdot rand(0,1)$
22:             **end if**
23:         **end for** $j$
24:         **if** $rand(0,1) < pc1$ **then** random $CR$ in $[0, 0.1]$
25:         **else** random $CR$ in $[0.9, 1]$
26:         **end if**
27:         Generate a trial vector $x_c$ by eq. (4)
28:         $nf = nf + 1$
29:         **if** $f(x_c) < f(x_i)$ **then** $x_i = x_c$ and $f(x_i) = f(x_c)$
30:             **if** $x_m$ is created by eq. (1) **then** $nm1 = nm1 + 1$
31:             **else** $nm2 = nm2 + 1$
32:             **end if**
33:             **if** $nm1 + nm2 \geq 100$ **then** adjust $nm1 = nm1 + 10$, $nm2 = nm2 + 10$ and update
                    $pm1 = 0.9pm1 + 0.1nm1/(nm1 + nm2); nm1 = 0, nm2 = 0$
34:             **end if**
35:             **if** $CR$ is in $[0, 0.1]$ **then** $nc1 = nc1 + 1$
36:             **else** $nc2 = nc2 + 1$
37:             **end if**
38:             **if** $nc1 + nc2 \geq 100$ **then** adjust $nc1 = nc1 + 10$, $nc2 = nc2 + 10$ and update
                    $pc1 = 0.9pc1 + 0.1nc1/(nc1 + nc2); nc1 = 0, nc2 = 0$
39:             **end if**
40:             **if** $f(x_c) < f(x_b)$ **then** $x_b = x_c$ and $f(x_b) = f(x_c)$
41:             **end if**
42:         **end if**
43:     **end for** $i$
44:     Find the worst value $f(x_w)$ of the population vectors
45:     **if** $\log(f(x_w)/f(x_b)) < 10^{-10}$ **then** stop
46:     **end if**
47: **end while**
48: Report $[L_j, U_j], x_b, f(x_b)$, and $nf$
```

Table 1. The bounds obtained by ADEBAS1 for the NIST problems

	Problem					
Parameter	Enso	Gauss1	Gauss2	Gauss3	Thurber	Hahn1
α_1	[-31.5, 33.9]	[0, 351.5]	[0, 135]	[0, 132.5]	[0, 1391]	[-1015.3, 456.2]
α_2	[-17.5, 16.2]	[0, 243.2]	[0, 7]	[0, 4.5]	[0, 1535.7]	[-153.7, 514.8]
α_3	[-17.7, 15.4]	[0, 130.7]	[0, 120]	[0, 336]	[0, 770.9]	[-112.2, 62.2]
α_4	[-111.3, 110.9]	[0, 151.1]	[0, 217.5]	[0, 561.5]	[0, 297.6]	[-18.3, 20.6]
α_5	[-16.5, 17.5]	[0, 180.1]	[0, 413.5]	[0, 200.5]	[0, 12.5]	[-183.7, 222.4]
α_6	[-17.5, 16.6]	[0, 131.5]	[0, 129.5]	[0, 123]	[0, 17.9]	[-22.9, 33.5]
α_7	[-119.8, 119.9]	[0, 177.7]	[0, 208]	[0, 225]	[0, 30.2]	[-7.4, 10.1]
α_8	[-17, 15.7]	[0, 216.4]	[0, 116.5]	[0, 100]		
α_9	[-16, 16]					
	Lanczos2	Lanczos3	Kirby2	Mgh17	Rat43	Mgh09
α_1	[0, 4.6]	[0, 4.6]	[-155.4, 261.1]	[-2.5, 7.2]	[0, 710.5]	[0, 1.7]
α_2	[0, 16.1]	[0, 17.3]	[-114.1, 71.2]	[-4.6, 5.5]	[0, 358.6]	[0, 5.4]
α_3	[0, 4.6]	[0, 4.9]	[-16.9, 20.7]	[-4.8, 5.9]	[0, 55.6]	[0, 6.5]
α_4	[0, 17.0]	[0, 16.7]	[-22.8, 35.1]	[-7.8, 13.6]	[0, 127.8]	[0, 5.9]
α_5	[0, 4.8]	[0, 4.7]	[-5, 6.5]	[-5.9, 9.4]		
α_6	[0, 17.1]	[0, 17.6]				
	Roszman1	Mgh10	Chwirut1	Chwirut2	Rat42	Bennett5
α_1	[-6.8, 7.8]	[0, 381.3]	[0, 2.9]	[0, 2.9]	[0, 81.2]	[-2520.7, 1]
α_2	[-2.9, 2.2]	[0, 6184]	[0, 2.0]	[0, 1.8]	[0, 21.1]	[0, 153.4]
α_3	[-20.7, 1220.3]	[0, 346]	[0, 2.6]	[0, 2.5]	[0, 13.2]	[0, 82.9]
α_4	[-213.4, 33.6]					
	Misra1a	Misra1b	Misra1c	Misra1d	Danwood	Boxbod
α_1	[0, 231.4]	[0, 345]	[0, 643.4]	[0, 444.4]	[0, 4.7]	[0, 220.1]
α_2	[0, 19.8]	[0, 11.2]	[0, 12.1]	[0, 10.2]	[0, 5.7]	[0, 42.2]

4. Performance of ADEBAS on real-world parameter identification problems

In this section, we test the performance of ADEBAS1 on real-world problems: parameter identifications on three photovoltaic models and a curve model of soil water retention.

4.1. Parameter estimation of photovoltaic models

Parameter estimation of photovoltaic (PV) models [25] is nonlinear regression consisting of three models: single diode, double diode, and PV module models as shown in Fig. 1. The datasets, lower and upper bounds, and constant parameters are as in the original paper. In all models, V_L is cell output voltage, I_L is cell output current, and q, K, T are constant parameters. The equations of these models are the following.

(4.1a) The single diode model:

$$f(V_L,I_L,a)=I_{ph}-I_{sd}[e^{\frac{q(V_L+R_SI_L)}{nKT}}-1]-\frac{V_L+R_SI_L}{R_{sh}}-I_L \qquad (6)$$

where $a=(I_{ph},I_{sd},R_S,R_{sh},n)$ is a vector of parameters to be estimated.

(4.1b) The double diode model:

$$f(V_L,I_L,a)=$$

$$=I_{ph}-I_{sd1}[e^{\frac{q(V_L+R_SI_L)}{n_1KT}}-1]-I_{sd2}[e^{\frac{q(V_L+R_SI_L)}{n_2KT}}-1]-\frac{V_L+R_SI_L}{R_{sh}}-I_L \qquad (7)$$

where $a=(I_{ph},I_{sd1},I_{sd2},R_S,R_{sh},n_1,n_2)$ is a vector of parameters to be estimated.

(4.1c) The PV module model:

$$f(V_L,I_L,a)=I_{ph}-I_{sd}[e^{\frac{q(\frac{V_L}{N_s}+R_S\frac{I_L}{N_p})}{nKT}}-1]-\frac{\frac{V_L}{N_s}+R_S\frac{I_L}{N_p}}{R_{sh}}-I_L \qquad (8)$$

where $a=(I_{ph},I_{sd},R_S,R_{sh},n)$ is a vector of parameters to be estimated and N_p, N_s are constants.

The objective function for this problem is the root mean square error (RMSE)

$$S(a)=\sqrt{\frac{1}{N}\sum_{i=1}^{N}f(V_L(i),I_L(i),a)^2} \qquad (9)$$

where N is the number of experimental data. The dataset for the models 4.1a and 4.1b contains 25 $(V_L(i),I_L(i))$ data while the dataset for the model 4.1c contains 26 data.

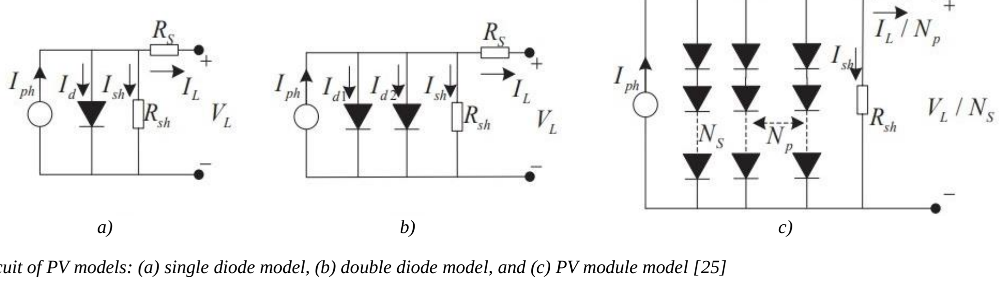

Fig. 1. Equivalent circuit of PV models: (a) single diode model, (b) double diode model, and (c) PV module model [25]

Table 2 (part 1). The bounds obtained by ADEBAS2 for the NIST problems

	Problem					
Parameter	Enso	Gauss1	Gauss2	Gauss3	Thurber	Hahn1
α_1	[-6.2, 20.9]	[0, 145.0]	[0, 129.6]	[0, 128.8]	[0, 1384.2]	[-605.4, 190.3]
α_2	[-7.2, 9.6]	[0, 1.0]	[0, 1.1]	[0, 3.1]	[0, 1530.9]	[-47.4, 328.8]
α_3	[-6.6, 7.4]	[0, 102.0]	[0, 123.6]	[0, 137.9]	[0, 741.8]	[-46.1, 9.7]
α_4	[-80.6, 79.3]	[0, 226.0]	[0, 201.7]	[0, 229.0]	[0, 292.7]	[-3.5, 7.0]
α_5	[-7.5, 7.5]	[0, 398.0]	[0, 93.4]	[0, 86.2]	[0, 7.5]	[-23.5, 114.0]
α_6	[-7.3, 6.8]	[0, 150.0]	[0, 124.8]	[0, 141.9]	[0, 11.4]	[-6.5, 12.0]
α_7	[-82.9, 84.3]	[0, 97.0]	[0, 209.7]	[0, 227.3]	[0, 7.6]	[-4.1, 3.6]
α_8	[-6.9, 7.1]	[0, 115.0]	[0, 90.1]	[0, 91.2]		
α_9	[-7.8, 6.7]					

Table 2 (part 2). The bounds obtained by ADEBAS2 for the NIST problems

	Problem					
	Lanczos2	Lanczos3	Kirby2	Mgh17	Rat43	Mgh09
α_1	[0, 3.2]	[0, 3.3]	[-580.9, 120.3]	[-2.5, 7.2]	[0, 710.4]	[0, 1.0]
α_2	[0, 11.0]	[0, 8.8]	[-45.7, 62.0]	[-4.6, 5.5]	[0, 368.8]	[0, 2.1]
α_3	[0, 3.2]	[0, 3.1]	[-4.2, 5.2]	[-4.8, 5.9]	[0, 53.0]	[0, 1.7]
α_4	[0, 11.2]	[0, 9.6]	[-6.8, 11.1]	[-7.8, 13.6]	[0, 127.8]	[0, 1.4]
α_5	[0, 3.1]	[0, 3.3]	[-2.8, 2.5]	[-5.9, 9.4]		
α_6	[0, 11.1]	[0, 8.8]				
	Roszman1	Mgh10	Chwirut1	Chwirut2	Rat42	Bennett5
α_1	[-3.7, 3.7]	[0, 376.0]	[0, 1.1]	[0, 1.2]	[0, 81.2]	[-2522.6, 0.1]
α_2	[-2.4, 1.4]	[0,6183.9]	[0, 0.9]	[0, 0.9]	[0, 21.3]	[0, 47.0]
α_3	[-8.5, 1219.9]	[0, 346.0]	[0, 0.8]	[0, 0.9]	[0, 13.1]	[0, 28.5]
α_4	[-209.9, 30.0]					
	Misra1a	Misra1b	Misra1c	Misra1d	Danwood	Boxbod
α_1	[0, 221.7]	[0, 344.8]	[0, 643.4]	[0, 444.2]	[0, 4.7]	[0, 219.8]
α_2	[0, 20.5]	[0, 12.3]	[0, 12.1]	[0, 11.7]	[0, 5.7]	[0, 41.9]

4.2. Parameter estimation of the soil water retention model

Parameter estimation of the soil water retention model [21] fits the experimental data to the Van Genuchten model expressed by

$$f(h,a)=\theta_r+(\theta_s-\theta_r)\left[\frac{1}{1+\alpha|h|^n}\right]^{1-\frac{1}{n}} \tag{10}$$

where $a=(\theta_r,\theta_s,\alpha,n)$ is a vector of parameters to be estimated and h is the soil water potential. The datasets for soil samples ID 3020 and 4440 are as in the UNSODA database [13]. The objective function S is the residual sum of square (RSS) defined by

$$S(a)=\sum_{i=1}^{N}\big(y(i)-f(h(i),a)\big)^2 \tag{11}$$

where N is the number of experimental data and $y(i)$ is the measured water content.

For each problem, we apply ADESBAS1 to perform 30 independent runs using the same setting in Section 3 without the maximum number of function evaluations to obtain the best solutions. The setting and the results of compared methods are as in the original papers [17, 20, 21]. The best values are highlighted in boldface.

Table 3. The solutions obtained by ADEBAS1 for the NIST problems

			Problem			
Parameter	Enso	Gauss1	Gauss2	Gauss3	Thurber	Hahn1
α_1	10.51075	98.77821	99.01835	98.94036	1288.13968	1.07764
α_2	3.07621	0.01050	0.01099	0.01095	1491.07924	-0.12269
α_3	0.53280	88.97661	86.96292	87.20030	583.23836	0.00408
α_4	2.24507	112.53846	130.15053	129.69893	75.41664	-0.14263E-05
α_5	-0.66870	21.21447	21.55227	21.48436	0.96630	-0.00576
α_6	0.05096	83.50780	86.96294	87.20028	0.39797	0.00024
α_7	0.72101	133.94070	130.15053	129.69893	0.04973	-0.1231E-07
α_8	-0.74212	20.30470	21.55228	21.48435		
α_9	0.16909					
	Lanczos2	Lanczos3	Kirby2	Mgh17	Rat43	Mgh09
α_1	0.09625	0.08682	1.67451	0.37541	699.64150	0.19281
α_2	1.00573	0.95498	-0.13927	1.70575	5.27713	0.19128
α_3	0.86425	0.84401	0.00260	1.70575	0.75963	0.12306
α_4	3.00783	2.95160	-0.00172	0.00464	1.27925	0.13606
α_5	1.55290	1.58257	0.21664E-04	0.00464		
α_6	5.00288	4.98636				
	Roszman1	Mgh10	Chwirut1	Chwirut2	Rat42	Bennett5
α_1	0.20197	0.00561	0.19028	0.16658	72.46223	-2520.10955
α_2	-0.61954E-05	6181.34635	0.00613	0.00517	2.61808	46.72229
α_3	1204.45561	345.22363	0.01053	0.01215	0.06736	0.93242
α_4	-181.34269					
	Misra1a	Misra1b	Misra1c	Misra1d	Danwood	Boxbod
α_1	238.94213	337.99745	636.42725	437.36971	0.76886	213.80942
α_2	0.00055	0.00039	0.20813E-03	0.3022E-3	3.86041	0.54724

Table 4 (part 1). The solutions obtained by ADEBAS2 for the NIST problems

			Problem			
Parameter	Enso	Gauss1	Gauss2	Gauss3	Thurber	Hahn1
α_1	10.51075	98.77820	99.01833	98.94037	1288.13968	1.07764
α_2	3.07621	0.01050	0.01099	0.01095	1491.07936	-0.12269
α_3	0.53280	71.99449	88.93990	86.45054	583.23844	0.00409
α_4	-44.31107	178.99805	127.08649	130.70240	75.41666	-0.14263E-05
α_5	-1.62315	18.38938	21.82083	21.38346	0.96630	-0.00576
α_6	-0.52554	100.48991	84.98591	87.95002	0.39797	0.24053E+03
α_7	26.88761	67.48111	133.21457	128.69543	0.04973	-0.1231E-06
α_8	0.21232	23.12978	21.28373	21.58526		
α_9	1.49668					
	Lanczos2	Lanczos3	Kirby2	Mgh17	Rat43	Mgh09
α_1	0.09625	0.08682	1.67451	0.37541	699.64151	0.19281
α_2	1.00573	0.95498	-0.13927	0.30359	5.27712	0.19128
α_3	1.55290	1.58257	0.00260	0.16757	0.75963	0.12306
α_4	5.00288	4.98636	-0.00172	0.01731	1.27925	0.13606
α_5	0.86425	0.84401	0.21664E+04	0.01768		
α_6	3.00783	2.95160				

Table 4 (part 2). The solutions obtained by ADEBAS2 for the NIST problems

	Problem					
	Roszman1	Mgh10	Chwirut1	Chwirut2	Rat42	Bennett5
α_1	0.20197	0.00561	0.19028	0.16658	72.46224	-2522.78247
α_2	-0.61954E-05	6181.34637	0.00613	0.00517	2.61808	46.73353
α_3	1204.45566	345.22364	0.01053	0.01215	0.06736	0.93223
α_4	-181.34272					
	Misra1a	Misra1b	Misra1c	Misra1d	Danwood	Boxbod
α_1	238.94213	337.99746	636.42726	437.36971	0.76886	213.80941
α_2	0.55016E-03	0.39039E-3	0.20814E-03	0.3022E-3	3.86041	0.54724

Table 5. Performance of the ADEBAS1 with initial bounds [0,1] and ADEBAS2 with initial bounds [0,0.1] on NIST nonlinear regression problems

Problem	ADEBAS1			ADEBAS2		
	NS	*Meannf*	*Mean*λ	NS	*Meannf*	*Mean*λ
Chwirut1	100	3854	11.0	100	3405	11.0
Chwirut2	100	3900	11.0	100	3385	11.0
Danwood	100	1898	11.0	100	1941	11.0
Boxbod	100	2981	10.4	100	3078	10.4
Lanczos2	100	67375	10.0	100	63223	10.0
Lanczos3	100	68453	10.9	100	55980	10.9
Mgh09	100	6402	11.0	100	5770	11.0
Mgh10	100	60594	11.0	100	60456	11.0
Mgh17	100	19012	10.9	100	16620	10.9
Misra1a	93	4919	10.4	89	5082	10.4
Misra1b	100	4806	11.0	100	4954	11.0
Misra1c	100	6518	11.0	100	6563	11.0
Misra1d	100	5283	11.0	100	5385	11.0
Rat42	100	4396	11.0	100	4557	11.0
Rat43	100	11641	11.0	100	11842	11.0
Roszman1	100	14540	10.9	100	14709	10.9
Thurber	100	40146	10.8	99	40488	10.8
Kirby2	100	16336	10.9	99	15296	10.9
Enso	100	93115	10.6	99	112634	10.6
Hahn1	100	39804	10.9	100	37776	10.9
Bennett5	100	238597	8.9	24	238206	10.1
Gauss1	99	41888	10.6	1	34480	10.5
Gauss2	2	43560	10.2	83	45336	10.3
Gauss3	2	60680	10.4	72	51215	10.5

Table 6. Performance comparison of ADEBAS1 with five compared methods on the single diode model

Parameter	ADEBAS1	SDE-FMP [17]	HDE [20]	RLDE [20]	SEDE [20]	MADE [20]
I_{ph}	0.76078	0.76078	0.76078	0.76080	0.76078	0.76078
I_{sd}	3.23021E-07	0.32302	3.23021E-07	3.23126E−07	3.23021E-07	3.32321E-07
R_S	0.03638	0.03638	0.03638	0.03638	0.03638	0.03638
R_{sh}	53.71853	53.71855	53.71853	53.71852	53.71853	53.71850
n	1.48118	1.48118	1.48118	1.48118	1.48118	1.48118
Meannf (SD)	35173 (918.26)	50000 (NA)	50000 (NA)	50000 (NA)	50000 (NA)	50000 (NA)
Min $f(x_b)$	**9.86022E-04**	**9.86022E-04**	**9.86022E-04**	**9.86022E-04**	**9.86022E-04**	**9.86022E-04**
Mean $f(x_b)$	**9.86022E-04**	**9.86022E-04**	**9.86022E-04**	**9.86022E-04**	**9.86022E-04**	**9.86278E-04**
Max $f(x_b)$	**9.86022E-04**	**9.86022E-04**	**9.86022E-04**	**9.86022E-04**	**9.86022E-04**	**9.89440E-04**

Table 7. Performance comparison of ADEBAS1 with five compared methods on the double diode model

Parameter	ADEBAS1	SDE-FMP [17]	HDE [20]	RLDE [20]	SEDE [20]	MADE [20]
I_{ph}	0.76097	0.76078	0.76078	0.76080	0.76078	0.76078
I_{sd1}	2.72621E-07	0.74935	7.49349E-07	2.26000E-07	7.49347E-07	2.60559E-07
I_{sd2}	3.69642E-03	0.22597	2.2597E-07	7.4923E-07	2.2597E-07	2.6901E-07
R_S	3.68220E-02	0.03674	0.03674	0.03674	0.03674	0.03654
R_{sh}	146.98959	55.48544	55.4854	55.4847	55.4854	54.3179
n_1	1.46465	2.00000	2.00000	2.00000	2.00000	1.899
n_2	16.34144	1.45102	1.451	0.749	1.451	1.466
Meannf (SD)	98508(6054.87)	50000 (NA)	50000 (NA)	50000 (NA)	50000 (NA)	50000 (NA)
Min $f(x_b)$	**9.50372E-04**	9.82485E-04	9.82485E-04	9.82485E-04	9.82485E-04	9.82611E-04
Mean $f(x_b)$	**9.50373E-04**	9.84970E-04	9.84154E-04	9.84571E-04	9.82729E-04	9.85248E-04
Max $f(x_b)$	**9.50373E-04**	9.87351E-04	9.86022E-04	9.86951E-04	9.86022E-04	9.87459E-04

Table 8. Performance comparison of ADEBAS1 with five compared methods on the PV module model

Parameter	ADEBAS1	SDE-FMP [17]	HDE [20]	RLDE [20]	SEDE [20]	MADE [20]
I_{ph}	1.03051	1.03051	1.03051	1.03051	1.03051	1.03051
I_{sd}	3.48226E-06	3.48226	3.48226E-06	3.48231E-06	3.48226E-06	3.48225E-06
R_S	0.83422	0.83422	0.03337	0.03337	0.03337	0.03337
R_{sh}	27.27728	27.27729	27.27728	27.27729	27.27728	27.27729
n	1.35119	1.35119	1.35119	1.35119	1.35119	1.35119
$Meannf$ (SD)	32471(907.47)	50000 (NA)	50000 (NA)	50000 (NA)	50000 (NA)	50000 (NA)
Min $f(x_b)$	**2.42507E-03**	**2.42507E-03**	**2.42507E-03**	**2.42507E-03**	**2.42507E-03**	**2.42507E-03**
Mean $f(x_b)$	**2.42507E-03**	**2.42507E-03**	**2.42507E-03**	**2.42507E-03**	**2.42507E-03**	**2.42507E-03**
Max $f(x_b)$	**2.42507E-03**	**2.42507E-03**	**2.42507E-03**	**2.42507E-03**	**2.42507E-03**	**2.42507E-03**

Table 9. Performance comparison of ADEBAS1 with three compared methods on the Van Genuchten model

Soil sample ID	Parameter	ADEBAS1	JAYA [21]	DE [21]	PSO [21]
3020	θ_r	0.19166	0.19166	0.19166	0.19220
	θ_s	0.44901	0.44901	0.44901	0.44900
	α	6.24379E-05	0.01797	0.01796	0.01780
	n	2.40869	2.40862	2.40869	2.43342
$Meannf$ (SD)		13466(1098.02)	500000 (NA)	500000 (NA)	500000 (NA)
Min $f(x_b)$		**0.11171E-04**	0.11712E-04	0.11724E-04	0.11732E-04
Mean $f(x_b)$		**0.11171E-04**	NA	NA	NA
Max $f(x_b)$		**0.11171E-04**	NA	NA	NA
4440	θ_r	0.10356	0.10357	0.10356	0.10356
	θ_s	0.34817	0.34817	0.34817	0.34817
	α	0.79567E-11	0.02033	0.02033	0.02033
	n	6.56023	6.56040	6.56023	6.56023
$Meannf$ (SD)		24965 (1613.70)	500000 (NA)	500000 (NA)	500000 (NA)
Min $f(x_b)$		**1.72068E-03**	**1.72068E-03**	1.72070E-03	1.72070E-03
Mean $f(x_b)$		**1.72068E-03**	NA	NA	NA
Max $f(x_b)$		**1.72068E-03**	NA	NA	NA

5. Discussion

Section 3 presents the performance of ADEBAS on the NIST dataset. Tables 1 and 2 show the final bounds obtained by ADEBAS1 and ADEBAS2. Our approach generates the parameter bounds that contain the solutions [28]. Tables 3 and 4 present the solutions obtained by ADEBAS1 and ADEBAS2. Table 5 shows that ADEBAS1 and ADEBAS2 give NS = 100 for 20 and 16 out of 24 problems, respectively. The two variants of ADEBAS algorithms obtain high $Mean\lambda$ for almost all problems. ADEBAS1 cannot solve Gauss2 and Gauss3 because they require a very small initial bound for the second parameter. In contrast, ADEBAS2 cannot solve Gauss1 because it requires a broader initial bound for the 7th parameter to overcome getting trapped in a local solution. In practice, we may encounter a parameter estimation problem that contains a parameter requiring either a small or a broad initial bound.

Section 4 shows the performance comparison of ADEBAS1 and other methods on real-world problems. Tables 6 and 8 show that the ADEBAS1 gives solutions close to that of the compared methods for single diode and PV module models. However, ADEBAS1 achieves smaller $Meannf$ values. Table 7 shows that ADEBAS1 gives better solutions than compared methods for the double diode model. Table 9 shows that the $Meannf$ values obtained by the proposed method are less than the compared methods for the soil water retention model. Moreover, ADEBAS1 provides a better solution than the compared methods for soil sample ID 3020.

6. Conclusions

This research presents an adaptive differential evolution algorithm with a bound adjustment strategy called ADEBAS for solving nonlinear parameter identification problems. The algorithm gradually extends the bounds to search for the optimal solutions by detecting the parameter-bound violation of the mutant vectors. ADEBAS adaptively uses two mutation strategies and two ranges of crossover rate based on the probabilities calculated from their success in the selection. The proposed method can generate the bounds that cover the optimal solutions and provide reliability and high accuracy for solving 24 NIST problems. In addition, ADEBAS performs very well in real-world parameter identification applications and achieves the best new solution for the double diode model.

Acknowledgement

The authors would like to thank the Faculty of Science, Khon Kaen University for financial support.

References

[1] Alam D. F. et al.: Flower pollination algorithm based solar PV parameter estimation. Energy Conversion and Management 101, 2015, 410–422 [http://dx.doi.org/10.1016/j.enconman.2015.05.074].
[2] Askary R., Najarchi M., Mazaheri H.: Estimating SWRC parameters and unsaturated hydraulic permeability by improved Jaya and hybrid improved jaya optimization algorithms. Earth Science Informatics 15(4), 2022, 2155–2169 [https://doi.org/10.1007/s12145-022-00862-z].
[3] Barati R.: Parameter estimation of nonlinear Muskingum models using Nelder-Mead simplex algorithm. Journal of Hydrologic Engineering 16(11), 2011, 946–954 [http://doi.org/10.1061/(ASCE)HE.1943-5584.0000379].

[4] Brooks S. P., Morgan B. J.: Optimization using simulated annealing. Journal of the Royal Statistical Society Series D: The Statistician 44(2), 1995, 241–257.
[5] Chen X. et al.: Teaching–learning–based artificial bee colony for solar photovoltaic parameter estimation. Applied energy 212, 2018, 1578–1588 [https://doi.org/10.1016/j.apenergy.2017.12.115].
[6] Gautier M., Janot A., Vandanjon P. O.: A new closed-loop output error method for parameter identification of robot dynamics. IEEE Transactions on Control Systems Technology 21(2), 2012, 428–444 [https://doi.org/10.1109/TCST.2012.2185697].
[7] Hu Z., Gong W., Li S.: Reinforcement learning-based differential evolution for parameters extraction of photovoltaic models. Energy Reports 7, 2021, 916–928 [https://doi.org/10.1016/j.egyr.2021.01.096].
[8] Jin J., Gans N.: Parameter identification for industrial robots with a fast and robust trajectory design approach. Robotics and Computer-Integrated Manufacturing 31, 2015, 21–29 [https://doi.org/10.1016/j.rcim.2014.06.004].
[9] Kennedy J., Eberhart R.: Particle swarm optimization. Proceedings of ICNN'95-international conference on neural networks, 1995, 1942–1948.
[10] Li S., Gong W., Yan X., Hu C., Bai D., Wang L.: Parameter estimation of photovoltaic models with memetic adaptive differential evolution. Solar Energy 190, 2019, 465–474 [https://doi.org/10.1016/j.solener.2019.08.022].
[11] Liang J. et al.: Parameters estimation of solar photovoltaic models via a self-adaptive ensemble-based differential evolution. Solar Energy 207, 2020, 336–346 [https://doi.org/10.1016/j.solener.2020.06.100].
[12] Mohan S.: Parameter estimation of nonlinear Muskingum models using genetic algorithm. Journal of hydraulic engineering 123(2), 1997, 137–142 [https://doi.org/10.1061/(ASCE)0733-9429(1997)123:2(137)].
[13] Nemes A. D. et al.: Description of the unsaturated soil hydraulic database UNSODA version 2.0. Journal of hydrology 251(3-4), 2001, 151–162.
[14] Price W. L.: A controlled random search procedure for global optimisation. The Computer Journal 20(4), 1977, 367–370.
[15] Puphasuk P., Wetweerapong J.: An enhanced differential evolution algorithm with adaptation of switching crossover strategy for continuous optimization. Foundations of Computing and Decision Sciences 45(2), 2020, 97–124 [https://doi.org/10.2478/fcds-2020-0007].
[16] Salhi H., Kamoun S.: A recursive parametric estimation algorithm of multivariable nonlinear systems described by Hammerstein mathematical models. Applied Mathematical Modelling 39(16), 2015, 4951–4962 [https://doi.org/10.1016/j.apm.2015.03.050].
[17] Singsathid P., Wetweerapong J., Puphasuk P.: Parameter estimation of solar PV models using self-adaptive differential evolution with dynamic mutation and pheromone strategy. Computer Science 19(1), 2024, 13–21.
[18] Storn R., Price K.: Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. Journal of global optimization 11, 1997, 341–359 [http://doi.org/10.1023/A:1008202821328].
[19] Tvrdík J., Křivý I., Mišík L.: Adaptive population-based search: application to estimation of nonlinear regression parameters. Computational statistics & data analysis 52(2), 2007, 713–724 [https://doi.org/10.1016/j.csda.2006.10.014].
[20] Wang D. et al.: Heterogeneous differential evolution algorithm for parameter estimation of solar photovoltaic models. Energy Reports 8, 2022, 4724–4746 [https://doi.org/10.1016/j.egyr.2022.03.144].
[21] Wang L., Huang C., Huang L.: Parameter estimation of the soil water retention curve model with Jaya algorithm. Computers and Electronics in Agriculture 151, 2018, 349–353 [https://doi.org/10.1016/j.compag.2018.06.024].
[22] Whitley D.: A genetic algorithm tutorial. Statistics and computing 4, 1994, 65–85.
[23] Xu L.: The damping iterative parameter identification method for dynamical systems based on the sine signal measurement. Signal Processing 120, 2016, 660–667 [https://doi.org/10.1016/j.sigpro.2015.10.009].
[24] Yang X., You X.: Estimating parameters of van Genuchten model for soil water retention curve by intelligent algorithms. Applied Mathematics & Information Sciences 7(5), 2013, 1977–1983.
[25] Yu K. et al.: A performance-guided JAYA algorithm for parameters identification of photovoltaic cell and module. Applied Energy 237, 2019, 241–257 [https://doi.org/10.1016/j.apenergy.2019.01.008].
[26] Zhang J., Wang Z., Luo X.: Parameter estimation for soil water retention curve using the salp swarm algorithm. Water 10(6), 2018, 815 [https://doi.org/10.3390/w10060815].
[27] Zhou J. et al.: Parameters identification of photovoltaic models using a differential evolution algorithm based on elite and obsolete dynamic learning. Applied Energy 314, 2022, [https://doi.org/10.1016/j.apenergy.2022.118877].
[28] National Institute of Standards and Technology. Nonlinear regression, 2003 [https://www.itl.nist.gov/div898/strd/nls/nls_info.shtml].

M.Sc. Watchara Wongsa
e-mail: Watchara_w@kkumail.com

Watchara Wongsa completed M.Sc. degree in Applied Mathematics from Khon Kaen University, Thailand in 2014. He is a Ph.D. student in Applied Mathematics, Khon Kaen University. His research area is optimization.

https://orcid.org/0000-0001-6320-149x

Ph.D. Pikul Puphasuk
e-mail: ppikul@kku.ac.th

Pikul Puphasuk completed M.Sc. degree in Mathematics from Khon Kaen University, Thailand in 2002 and Ph.D. degree in Applied Mathematics from Suranaree University of Technology, Thailand in 2009. She is an associate professor at Department of Mathematics, Khon Kaen University. Her research areas include computational sciences, numerical analysis and optimization.

https://orcid.org/0000-0001-9069-1703

Ph.D. Jeerayut Wetweerapong
e-mail: wjeera@kku.ac.th

Jeerayut Wetweerapong completed M.Sc. degree in Mathematics from West Virginia University, US in 1995 and Ph.D. degree in Mathematics from Khon Kaen University, Thailand in 2012. He is an assistant professor at Department of Mathematics, Khon Kaen University. He has been doing research in field of scientific computing and optimization.

https://orcid.org/0000-0001-5053-3989