Digital IIR Filters Design Using Differential Evolution Algorithm with a Controllable Probabilistic Population Size

Wu Zhu[1], Jian-an Fang[1], Yang Tang[1,2,3,4]*, Wenbing Zhang[1,5], Wei Du[5]

[1] College of Information Science and Technology, Donghua University, Shanghai, China, [2] Research Institute for Intelligent Control and System, Harbin Institute of Technology, Harbin, China, [3] Institute of Physics, Humboldt University, Berlin, Germany, [4] Potsdam Institute for Climate Impact Research, Potsdam, Germany, [5] Institute of Textiles and Clothing, The Hong Kong Polytechnic University, Hong Kong, China

Abstract

Design of a digital infinite-impulse-response (IIR) filter is the process of synthesizing and implementing a recursive filter network so that a set of prescribed excitations results a set of desired responses. However, the error surface of IIR filters is usually non-linear and multi-modal. In order to find the global minimum indeed, an improved differential evolution (DE) is proposed for digital IIR filter design in this paper. The suggested algorithm is a kind of DE variants with a controllable probabilistic (CPDE) population size. It considers the convergence speed and the computational cost simultaneously by nonperiodic partial increasing or declining individuals according to fitness diversities. In addition, we discuss as well some important aspects for IIR filter design, such as the cost function value, the influence of (noise) perturbations, the convergence rate and successful percentage, the parameter measurement, etc. As to the simulation result, it shows that the presented algorithm is viable and comparable. Compared with six existing State-of-the-Art algorithms-based digital IIR filter design methods obtained by numerical experiments, CPDE is relatively more promising and competitive.

Citation: Zhu W, Fang J-a, Tang Y, Zhang W, Du W (2012) Digital IIR Filters Design Using Differential Evolution Algorithm with a Controllable Probabilistic Population Size. PLoS ONE 7(7): e40549. doi:10.1371/journal.pone.0040549

Editor: Jérémie Bourdon, Université de Nantes, France

Received March 27, 2012; **Accepted** June 8, 2012; **Published** July 11, 2012

Funding: This paper was supported by the Alexander von Humboldt Foundation of Germany and the China Postdoctoral Science Foundation (2011M501040). The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing Interests: The authors have declared that no competing interests exist.

* E-mail: tangtany@gmail.com

Introduction

Filtering problem [1,2] is a widely studied research topic in various fields of control and signal processing. The main objective of filtering is synthesizing and implementing a filter network [3] to modify, reshape, or manipulate the frequency spectrum of a signal according to some desired specifications. As one of the most successful filter networks, the well-known Digital infinite-impulse-response (IIR) filter has been extensively used in many practical systems [4–7], such as engineering system, network system, nuclear reactor, biological system, chemical system and electrical networks system. However, it has been recognized now that the IIR filter will generally not guarantee satisfactory performance if its feedback coefficients are chosen not appropriately during the adaptation process [8]. Apart from this disadvantage, the possibility of having a multi-modal and nonlinear error surface is another important design challenge for recursive filters [9,10]. To improve the robustness, in recent years, many heuristic optimization design methods have been developed, such as simulated annealing (SA) [11], ant colony optimization (ACO) [12], particle swarm optimization (PSO) [13], seeker optimization algorithm (SOA) [14], artificial bee colony (ABC) [15,16] and differential evolution (DE) [17], etc.

A SA is usually sensitive to its starting point of the search and requires too many function evaluations to converge to the global minima. The ACO imitates the social behavior of real ant colonies and it has been originally developed for combinatorial optimization problems. But, it may occasionally be trapped into local stagnation or premature convergence resulting in a low optimizing precision or even failure [18]. What's more, the conventional PSO algorithm [19] as shown in several studies can easily fly into the local optima. It also lacks the ability to jump out of the local optima when solving complex multimodal tasks. The SOA simulates the act of human searching and has been widely developed for system identification [20]. Nonetheless, the performance of SOA is also affected by its parameters, and it could not easily escape from premature convergence.

Differential evolution, proposed by Storn and Price [21], is a population-based heuristic search algorithm with dual features of reliability and flexibility. It implements the evolutionary generation-and-test paradigm for global optimization by using the current population information of distance and direction to guide the search. It has many advantages such as simplicity, reliability, high performance and easy implementation, which gives great potential application to IIR design. In the seminal DE algorithm, perturbation is operated by adding a weighted moving vector (the weight F is called scale factor) and modifying the values of some randomly selected coordinates. The perturbed solution, namely the offspring, is then evaluated by means of its objective function and compared with its corresponding parent. If the newly generated solution outperforms its parent, then a replacement occurs; otherwise the parent solution is retained. To provide a rigorous proof for its probabilistic convergence, [22] has modeled the population as a dynamical system in which the probability density function (PDF) of the population vectors changes with time. It was shown therein that the dynamics is asymptotically stable (which implies convergence) at the equilibrium PDF, which is a Dirac delta function placed at the global optima. Later on, various mutation strategies [23] were used for the generation of new solutions to augment the robustness of the underlying algorithm.

In DE, it is often the case that, for optimization problems such as single-objective, multi-objective, large scale, constrained and dynamic problems, the population size is naturally fixed on a constant value; see, e.g., [24]. Unfortunately, it is usually difficult to determine how large the population size is suitable for solving numerical optimization problems. For instance, a definite population size is given in [25] which increases linearly with the problem dimension; yet the sparse and noisy data makes it difficult to accurately estimate the maximum population size. Inspired by this fact, an efficient population utilization strategy for DE (DESAP) [26] was developed to automatically tune its population size from initialization to completion right through the evolutionary search process. Nevertheless, the population utilization method depends on its encoding methodology, which is a restriction for the current population with complex dynamical behaviors [27]. No significant advantages can be observed while using relative encoding. Subsequently, the idea of population adaptation has been applied in solving dynamic optimization problems [28], where multi-population approach (DynDE) is placed onto DE aimed at locating optima faster. Yet such a method requires a determinated topology that may be sensitive to the noise of measurement in some extent [29]. It is worth mentioning that although the population may not be as large as possible, it ought to meet the requirements of given engineering. Therefore, a new reduction method for the population size was shown up for the jDE in order to enhance algorithmic performance [30], where the population size was progressively declining until the arrival of the final budget during the optimization process. Unfortunately, this method can not keep track of the progress of individuals in the sustainable reduction.

Many studies have indicated that various computational predictors or models developed in biology and biomedicine, such as those in identifying DNA-binding proteins [31], predicting G-protein-coupled receptors (GPCRs) and their types [32], identifying nuclear receptors and their subfamilies [33,34], identifying the subcellular localization of proteins from various organisms [35–39], can timely provide very useful insights and informations for both basic research and drug development. These predictors all use the methods of digital signal processing. In view of this, the present study is attempted to addresses an important problem in designing digital IIR filters in hopes that it may become a useful tool for the related information-treating areas. However, most of the developed adaptive population methods have their advantages and disadvantages. So far, it remains open that how to utilize the dynamic population strategy to solve real-world practical problem. We aim at employing a Markov jumping (switching) population updating DE for digital IIR filter, so that the dynamic population can quickly converge to the potential global optimum by taking advantage of the current search information. Thus, the CPDE-based evolutionary method is simulated in digital IIR filter design, and its performance is compared to that of three versions of DE, CMA-ES, GL-25 and SOA. In the community of six digital IIR filter design problems, it is shown empirically that CPDE is capable of producing highly competitive results compared with other EAs.

Results

0.1 Illustration

Application of the IIR filter in system identification has been widely studied since many problems encountered in signal processing can be characterized as a system identification problem (Figure 1) [40,41]. Therefore, in the simulation study, IIR filters are designed for the system identification purpose. In this section, we will utilize a modified DE to adjust the parameters of the filters until the error between the output of the filter and the unknown system is minimized. Subsequently, we provide an overall comparison between the performance of CPDE and several other State-of-the-Art algorithms to verify the effectiveness and usefulness of the proposed method.

Six typical system identification problems [14] make up the test suite used for this comparative study, which are listed in Tables 1 and 2. $H_s(z)$ and $H_f(z)$ specify the system and filter transfer functions, respectively; $x(k)$ indicates the system input; SNR is the Signal to Noise Ratio; $\omega(k)$ is the system noise, which is independent of $x(k)$; $\mathcal{N}(0,1)$ presents the white Gaussian noise (WGN) in zero-mean normal distribution with variance 1. In Table 2, ϕ records all coefficients of six digital IIR filters; Search space is the predefined boundary constraints, that will be analyzed in Section 05; N denotes the data length used in calculating the mean-square-error (MSE). The examples were selected so as to include problems with the following characteristics: unimodal/multi-modal, no noise/noisy. For each algorithm and each test function, 30 independent runs are conducted with 100,000 FES as the termination criterion.

Traditionally,”generation” is a natural form of computational cost for statistical comparison [11–14,17]. However, the population may not be the same in different algorithms. The algorithm with a larger population may obtain a better performance together with much more function evaluations in every generation. Thus, in this paper, the function evaluations (FES) are conducted here to represent its computational cost for algorithm comparison.

In all simulations, the population size of the most EAs is 100 with the exception of EPSDE and SaDE. As suggested in Ref. [42,43], the population size of EPSDE and SaDE is chosen to be 50. Seven existing EA algorithms are shown in Table 3 in detail. CMA-ES represents the state of the art of Evolution Strategies and it is a referent in the continuous optimization field. GL-25 is a hybrid real-coded genetic algorithm which combines the global and local search. EPSDE is an adaptive DE with ensemble of parameters which incorporates a self-organizing method. jDE is a standard DE with adapted parameter setting. SaDE delivers a mutation strategy pool where strategy is self-adapted based on their previous performance. SOA is a novel heuristic stochastic optimization algorithm based on the simulation of the act of human searching. The parameters for these EAs are provided in Table 3.

0.2 Comparison on the Solution Accuracy

In this section, an overall comparison of the performance is provided between the CPDE variant and other six State-of-the-Art EAs (i.e., CMA-ES, GL-25, EPSDE, jDE, SaDE and SOA). We evaluate the performance of seven heuristic algorithms over six typical nonlinear uncertain discrete-time problems. Fig. 2 illustrates the cost function value versus number of evaluations averaged over 30 random runs for the seven algorithms. The subfigures amplify the convergence graphs in clarity. Table 4 reports the experimental results of Examples 1–6, averaged over 30 independent runs with 100,000 FES.

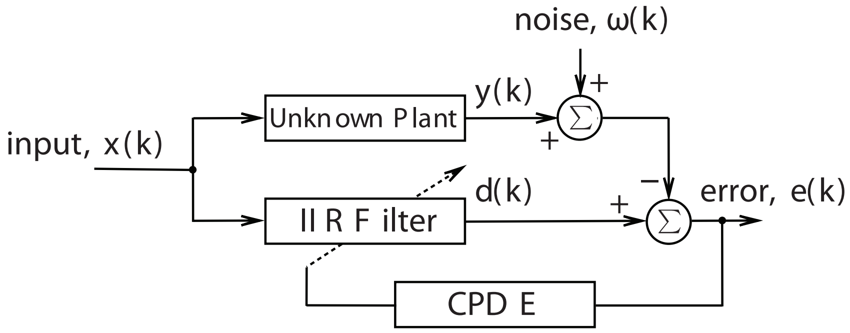

Figure 1. Block diagram of the system identification process using IIR filter designed by CPDE.
doi:10.1371/journal.pone.0040549.g001

From the Table 4 and Fig. 2, the CPDE provides the best performance on the $ex_2 - ex_3$, ex_5 and ex_6, then ranks the second on the ex_1 and ex_4. SOA gives the best performance on the ex_1 and ex_4. The results show that GL-25 and SOA have good ability of convergence speed. Fig. 3 also shows instance of evolution of the parameters of two filters for CPDE.

To be specific, on the three multimodal problems ($ex_1 - ex_2$ and ex_6), CPDE performs much better than CMA-ES, GL-25, EPSDE, jDE, SaDE and SOA on the latter two functions. SOA delivers good accuracy and the highest convergence rate on ex_1, while CPDE outperforms other five methods. To sum up, CPDE is the winner on multimodal functions. This might be due to the fact that CPDE implements the overall adaptive variable population size method, which can help the DE search the optimum as well as maintain a higher convergence speed when dealing with multimodal rotated functions. On the remaining three unimodal functions ($ex_3 - ex_5$), CPDE performs significantly better than six others for ex_3 and ex_5. SOA can provide good accuracy on ex_4, while CPDE achieves the highest convergence rate. The outstanding performance of CPDE is due to its dynamic PDS, which leads to very fast convergence. Overall, the CPDE is the best among the seven methods in the comparison conducted on unimodal functions and expanded multi-modal functions. For a thorough comparison, the t-test has also been carried out in this paper. Table 4 presents the total score on every function of this two-tailed test with a significance level of 0.05 between the CPDE variant and other heuristic algorithms. Rows "+ (Better)," "= (Same)," and "– (Worse)" give the number of functions that the CPDE performs significantly better than, almost the same as, and significantly worse than the compared algorithm on fitness values in 30 runs, respectively. As confirmed in t-test, the CPDE in general offers more improved performance than the other six State-of-the-Art EAs.

Table 1. Problem Illustration.

Inst.	Test Function	$x(k)$	$\omega(k)$
Example 1	$H_s(z) = \frac{0.05 - 0.4z^{-1}}{1 - 1.1314z^{-1} + 0.25z^{-2}}$	a white-noise sequence	
	$H_f(z) = \frac{a_0}{1 + b_1 z^{-1}}$		
Example 2	$H_s(z) = \frac{-0.3 + 0.4z^{-1} - 0.5z^{-2}}{1 - 1.2z^{-1} + 0.5z^{-2} - 0.1z^{-3}}$	a uniformly distributed white-noise sequence, taking values from $(-0.5,0.5)mi$, $SNR = 30$ dB	
	$H_f(z) = \frac{a_0 + a_1 z^{-1}}{1 + b_1 z^{-1} + b_2 z^{-2}}$		
Example 3	$H_s(z) = \frac{1}{1 - 1.2z^{-1} + 0.6z^{-2}}$	a unit-variance white Gaussian pseudonoise sequence	0 and
	$H_f(z) = \frac{a_0}{1 + b_1 z^{-1} + b_2 z^{-2}}$		$\mathcal{N}(0,1)$
Example 4	$H_s(z) = \frac{1.25z^{-1} - 0.25z^{-2}}{1 - 0.3z^{-1} + 0.4z^{-2}}$	a white-noise input, $SNR = 40$ dB	
	$H_f(z) = \frac{a_0 + a_1 z^{-1} + a_2 z^{-2}}{1 + b_1 z^{-1} + b_2 z^{-2}}$		
Example 5	$H_s(z) = \frac{1}{1 - 1.4z^{-1} + 0.49z^{-2}}$	a colored noise by filtering a white Gaussian pseudo-noise sequence with a FIR filter:	
	$H_f(z) = \frac{a_0}{1 + b_1 z^{-1} + b_2 z^{-2}}$	$H_c(z) = (1 - 0.7z^{-1})^2 (1 + 0.7z^{-1})^2$	
Example 6	$H_s(z) = \frac{1}{(1 - 0.6z^{-1})^3}$	a colored noise by filtering a white Gaussian pseudo-noise sequence with a FIR filter:	
	$H_f(z) = \frac{a_0}{1 + b_1 z^{-1} + b_2 z^{-2}}$	in$H_c(z) = (1 - 0.6z^{-1})^2 (1 + 0.6z^{-1})^2$	

doi:10.1371/journal.pone.0040549.t001

Table 2. Parameters Illustration.

Inst.	ϕ	Search Space	N
Example 1	$a_0=-0.311$		1000
	$b_1=-0.906$		
Example 2	$a_0=-0.3948$, $a_1=-0.0742$		100
	$b_1=-0.2230$, $b_2=-0.5739$		
Example 3	$a_0=1$		100
	$b_1=-1.2$, $b_2=0.6$	$[-2,2]$	
Example 4	$a_0=0$, $a_1=1.25$, $a_2=-0.25$		100
	$b_1=-0.3$, $b_2=0.4$		
Example 5	$a_0=1$		100
	$b_1=-1.4$, $b_2=0.49$		
Example 6	$a_0=1.14$		2000
	$b_1=1.6$, $b_2=-0.7$		

doi:10.1371/journal.pone.0040549.t002

Table 3. EA algorithms for comparison.

Algorithm	Parameters	Reference
CMA-ES$_{ii}$	$\sigma=0.25$	[54]
GL-25$_{ii}$	$P_G=25\%,N_F^L=5$	[55]
EPSDE$_{ii}$	$F:0.4-0.9,C_r:0.1-0.9$	[42]
jDE$_{ii}$	$\tau_1=\tau_2=0.1$	[56]
SaDE$_{ii}$	$F:rand_{\mathcal{N}}(0.5,0.3),LP=50$	[43]
SOA	$\omega:0.4-0.9,\mu:0.0111-0.99$	[14]
CPDE	$p_1=1,p_2=5,\zeta=4$	this paper

doi:10.1371/journal.pone.0040549.t003

As mentioned above, the CPDE has shown a very competitive performance in the six filtering problems. In practical engineering, noise exist universally in nature [44]. Therefore, in the past few decades researchers have witnessed significant progress on filtering and control for linear/nonlinear systems with various types of noises among which Gaussian noise is one of the most general signals that has been widely studied [45,46]. Here, we further evaluate the proposed framework on the six expanded stochastic systems, where a zero-mean Gaussian white-noise is added. The maximum number of FES is set to be 100,000 in all runs. Table 5 summarizes the experimental results.

From the Table 5, the CPDE provides the best performance on the ex_2, ex_3-ex_6, then ranks the second on the ex_1. SOA offers the best performance on the ex_1. The results show that GL-25 and jDE have a good ability of convergence speed.

More specifically, on the three multimodal problems (ex_1-ex_2 and ex_6), although it worked slightly weaker on some functions, the CPDE in general offered more improved performance than all the EAs compared. It performs much better on the ex_2 and ex_6, and attains slightly worse performances than the best solutions on the ex_1. On the remaining three unimodal functions (ex_3-ex_5), CPDE performs much better than other EAs. Hence, CPDE exhibits the highest performance in noise-expanded filtering problems, which can efficiently adjust the population structure and guide the evolution process toward more promising solutions.

The t-test is also summarized in Table 5. In fact, CPDE performs better than CMA-ES, GL-25, EPSDE, jDE, SaDE and SOA on 6, 6, 6, 6, 6 and 5 out of 6 test functions, respectively. Thus, CPDE is better than other six competitors in filtering system identification problems. Comparing the results and the cost function graphs, among these seven EA algorithms, the GL-25 can converge to the best solution found so far very quickly though it is easy to stuck in the local optima. The SOA has good global search ability and slow convergence speed. The jDE have good search capability on noise-expanded filtering problems. The CPDE has good local search ability and global search ability at the same time.

0.3 Comparison on Convergence Rate and Successful Percentage

The convergence rate for achieving the global optimum is another key point for testing the algorithm performance. Note that in solving real-world optimization problems, the "function evaluation" overwhelms the algorithm overhead [47]. Hence, the computation times of these algorithms are not provided here. Table 6 shows that CPDE needs the least FES to achieve the acceptable solution on ex_3-ex_4 and ex_6, which reveals that CPDE has a higher convergence rate than other algorithms. Though SOA or GL-25 might outperform CPDE on the other functions, SOA and GL-25 have much worse successful ratio and accuracy than CPDE on the problems for comparison. In addition, CPDE can achieve accepted value with a good convergence speed and accuracy on most of the problems, as shown in Tables 4, 5 and Fig. 2. Tables 4, 5 also show that CPDE yields a highest percentage for achieving acceptable solutions in 30 runs. According to the no free lunch theorem [48], any elevated performance over one class of problems is offset by performance over another class. Hence, one algorithm cannot perform better on convergence speed and accuracy than the others on every optimization problem.

In summary, the CPDE performs best on unimodal problems with or without noise and has a good search ability of multimodal problems. Owing to the controllable probabilistic technique, the CPDE processes capabilities of fast convergence speed, highest successful ratio and the best search accuracy among these EAs.

0.4 Performance of Controllable Probabilistic Approach

In this section, the controllable probabilistic (CP) approach is used to test the search performance of CPDE. In all the experiments, threshold ζ is adjusted in the following. Moreover, the parameter p_2 is also considered, which denotes the number of potential candidates for perturbation.

In this paper, ζ indicates the trigger thresholds, which is used to control the sensitivity of the dynamic CPDE. While ζ is set as one, the population size will be adjusted in each generation. Setting a higher ζ value will result in a lower sensitivity of the CPDE, while a lower ζ value will lead to a higher efficiency of the population adjustment. Notice that the parameter ζ should set to be larger than one. Failure to do this will result in an instant elimination of a newborn individual with poor performance, which may provide some degree of diversity preservation. On the other hand, coefficient p_2 also influences the perturbation process substantially.

Table 7 shows the comparisons between CPDE with other three parameter settings of CPDE over Examples 1–6 with noise perturbation. It indicates that CPDE is not sensitive to the adjustment of parameters. In order to make a balance of the search accuracy and robustness, $\zeta=4$ and $p_2=5$ are used as a representative parameter setting in our paper. This setting will prevent the instant elimination of a newborn individual and keep the CP approach high sensitivity.

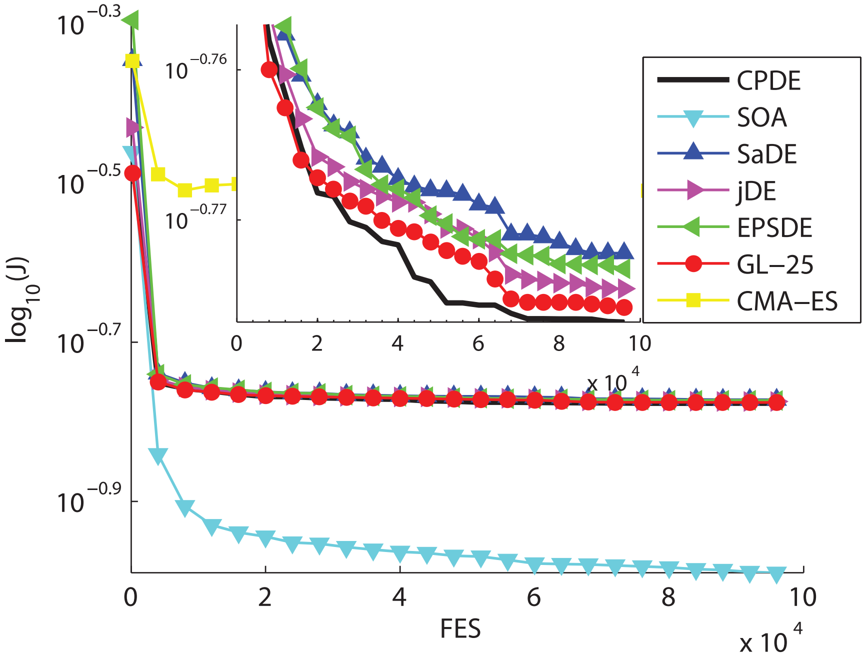

Figure 2. Cost function value versus number of evaluations averaged over 30 random runs for the seven algorithms (a) ex_1. (b) ex_2. (c) ex_3. (d) ex_4. (e) ex_5. (f) ex_6.
doi:10.1371/journal.pone.0040549.g002

Discussion

The CPDE is an improved differential evolution algorithm with a controllable probabilistic population size. When particles are clustered together in a region and trapped into the local basin, CPDE perturbs the population and generates the necessary "fine" individuals to share their up-to-date information. In addition, CPDE removes redundant individual with its entropy and ranking metrics to save computational load. In this paper, a CPDE-based digital filter design method has been proposed, and the benefits of CPDE for designing digital IIR filters have been studied.

An overall comparison between the performance of the CPDE and other six State-of-the-Art EAs (i.e., jDE, SaDE, EPSDE, CMA-ES, GL-25 and SOA) was provided over 6 typical robust

Table 4. Experimental results of Examples 1–6, averaged over 30 independent runs with 100,000 FES.

Inst.		CMA-ES	GL-25	EPSDE	jDE	SaDE	SOA	CPDE
	Mean Error	3.2236E−01	1.6756E−01	1.6856E−01	1.6804E−01	1.6896E−01	**1.0246E−01**	1.6719E−01
Example 1	Std Dev	2.1399E−01	2.2370E−03	2.9506E−03	2.3747E−03	2.2793E−03	**4.2071E−03**	2.1508E−03
	T-test	+	+	+	+	+	−	
	Mean Error	1.4424E−02	6.8069E−03	6.7232E−03	6.6451E−03	6.9145E−03	6.6265E−03	**6.3136E−03**
Example 2	Std Dev	5.9387E−03	4.4369E−04	4.4507E−04	4.1499E−04	4.7206E−04	3.1431E−04	**5.0418E−04**
	T-test	+	+	+	+	+	+	
	Mean Error	1.4517E−01	8.9999E−33	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**0.00E+00**
Example 3	Std Dev	5.1851E−01	3.4254E−32	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**0.00E+00**
	T-test	+	+	≈	≈	≈	≈	
	Mean Error	1.2372E−04	4.6732E−64	1.4751E−71	4.2794E−73	1.4999E−83	3.9896E−100	**0.00E+00**
Example 4	Std Dev	6.7762E−04	2.5596E−63	4.5215E−72	2.0951E−73	6.7257E−83	1.8653E−99	**0.00E+00**
	T-test	+	+	+	+	+	+	
	Mean Error	2.4285E−01	3.9674E−20	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**0.00E+00**
Example 5	Std Dev	6.0577E−01	1.8016E−19	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**0.00E+00**
	T-test	+	+	≈	≈	≈	≈	
	Mean Error	1.8801E+00	1.0167E−01	1.0162E−01	1.0127E−01	1.0199E−01	1.0168E−01	**1.0063E−01**
Example 6	Std Dev	2.9858E+00	1.3400E−03	1.6198E−03	9.8318E−01	1.3437E−03	1.2518E−03	**1.1027E−03**
	T-test	+	+	+	+	+	+	

doi:10.1371/journal.pone.0040549.t004

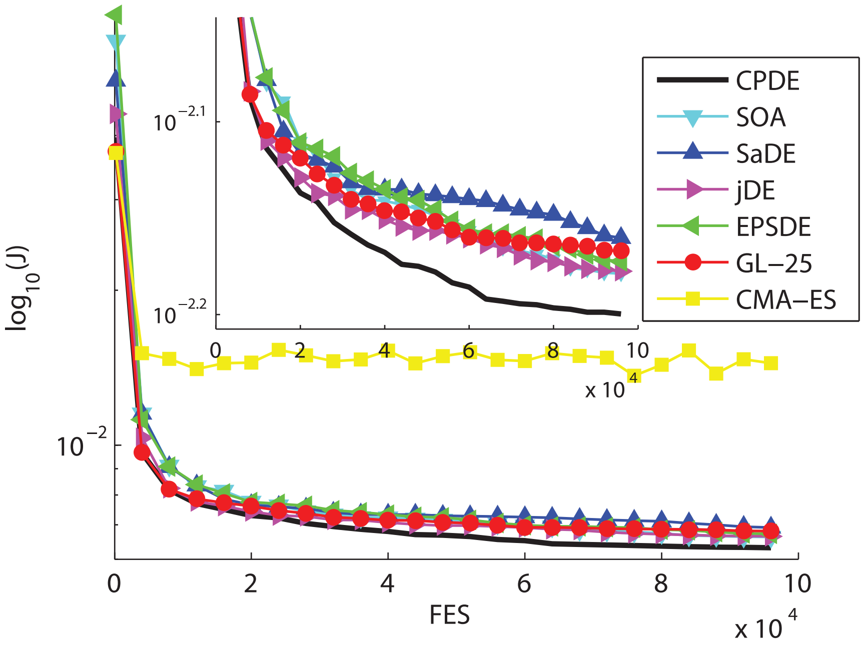

Figure 3. Instance of evolution of the parameters of two filters for CPDE (a) ex_2**. (b)** ex_4**.**
doi:10.1371/journal.pone.0040549.g003

system identification problems, and the result clearly indicated the CPDE achieved a substantially significant improvement on the performance. Furthermore, convergence rate was also validated that the CPDE has good convergence performance to achieve the fixed accuracy level with acceptable generations. Thus, it is believed that the proposed CPDE is capable of rapidly and efficiently adapting the parameters of a wide variety of IIR structures and will become a promising candidate for digital filter design.

Previous work has shown the importance of system identification on digital IIR filter design. Furthermore, CPDE has effectively been applied to estimate the structure of nonlinear uncertain discrete-time system. Therefore, our method is possible to be used to reconstruct the topology structure for on-line

Table 5. Experimental results of Examples 1–6 with noise perturbation, averaged over 30 independent runs with 100,000 FES.

Inst.		**CMA-ES**	**GL-25**	**EPSDE**	**jDE**	**SaDE**	**SOA**	**CPDE**
	Mean Error	1.3351E+00	9.9644E–01	1.0013E+00	9.9526E–01	1.0001E+00	**6.4329E–01**	9.8494E–01
Example 1	Std Dev	2.2756E–02	1.3799E–02	1.1911E–02	1.3331E–02	1.5337E–02	**2.7742E–02**	1.2588E–02
	T-test	+	+	+	+	+	–	
	Mean Error	8.6191E–01	5.2458E–01	5.5239E–01	5.3305E–01	5.4549E–01	5.4789E–01	**5.1608E–01**
Example 2	Std Dev	7.7377E–02	2.7869E–02	1.9928E–02	2.3498E–02	2.2388E–02	2.2444E–02	**1.8810E–02**
	T-test	+	+	+	+	+	+	
	Mean Error	1.0701E+00	5.3043E–01	5.3999E–01	5.3420E–01	5.4608E–01	5.4132E–01	**5.0512E–01**
Example 3	Std Dev	6.8551E–01	2.2359E–02	2.9233E–02	2.3386E–02	2.5721E–01	2.2572E–02	**2.2874E–02**
	T-test	+	+	+	+	+	+	
	Mean Error	8.9973E–01	5.4311E–01	5.5414E–01	5.3367E–01	5.5998E–01	5.3624E–01	**5.2156E–01**
Example 4	Std Dev	7.9829E–02	2.8612E–02	2.5562E–02	2.1445E–02	3.2593E–02	2.6400E–02	**2.1444E–02**
	T-test	+	+	+	+	+	+	
	Mean Error	1.6026E+00	5.3366E−01	5.4711E–01	5.2517E–01	5.4717E–01	5.2705E–01	**5.0945E–01**
Example 5	Std Dev	1.1095E+00	2.8589E–02	2.7466E–02	2.5575E–02	2.4434E–02	3.4826E–02	**1.6459E–02**
	T-test	+	+	+	+	+	+	+
	Mean Error	2.7866E+00	9.8570E–01	9.8841E–01	9.8231E–01	9.8739E–01	9.8765E–01	**9.7624E–01**
Example 6	Std Dev	2.9110E+00	1.3256E–02	9.7794E–03	9.2576E–03	1.3060E–02	1.0416E–02	**7.8179E–02**
	T-test	+	+	+	+	+	+	

doi:10.1371/journal.pone.0040549.t005

Table 6. Convergence speed and algorithm reliability comparisons on Examples 1–6 with noise perturbation; '−' representing no runs reached an acceptable solution.

Inst.		CMA-ES	GL-25	EPSDE	jDE	SaDE	SOA	CPDE
Example 1	Mean Generations	–	313.5	438.6	397.1	380.5	**10.8**	341.7
	Right Percentage(%)	0	86.7	73.3	90	76.7	**100**	**100**
Example 2	Mean Generations	406.7	**295.9**	378.6	370.3	477.3	587.9	356.8
	Right Percentage(%)	56.7	90	70	86.7	73.3	56.7	**96.7**
Example 3	Mean Generations	–	554.1	560.6	454.6	513.9	716.8	**406.5**
	Right Percentage(%)	0	66.7	53.3	66.7	36.7	63.3	**100**
Example 4	Mean Generations	329.2	313.2	467.5	493	508.5	546.6	**290.4**
	Right Percentage(%)	23.3	90	73.3	90	66.7	90	**100**
Example 5	Mean Generations	–	388.1	422.5	**295.4**	516.6	513.1	363.3
	Right Percentage(%)	0	90	80	93.3	66.7	**100**	**100**
Example 6	Mean Generations	–	340.3	510	298.7	364.8	520.4	**291.9**
	Right Percentage(%)	0	90	80	93.3	86.7	**100**	**100**
	Mean Reliability	13.3	85.6	71.6	86.7	67.8	85	**99.45**

doi:10.1371/journal.pone.0040549.t006

adaptive filtering applications [49]. Another possible application is to identify topology and parameters of complex networks [50–52] and biological time series [53] by dynamic population strategy, provided online measurement and increasing/decreasing techniques are feasible. Generally, the suggested technique enables us to identify the unknown parameter of real networks which allows the required control applications (perturbations). Some possible experimental research is now under our investigation in controller design and tuning.

Materials and Methods

0.5 Description of the Problem

Consider the digital IIR filter with the input-output relationship governed by the difference equation:

$$y(k)+\sum_{i=1}^{M} b_i y(k-i)=\sum_{i=0}^{L} a_i x(k-i), \tag{1}$$

where $x(k)$ and $y(k)$ are the filter's input and output, respectively, and $M(\geq L)$ is the filter order. The transfer function of this IIR filter can be written in the following general form:

$$H(z)=\frac{A(z)}{B(z)}=\frac{\sum_{i=0}^{L} a_i z^{-i}}{1+\sum_{i=1}^{M} b_i z^{-i}}. \tag{2}$$

Hence, the design of this filter can be formulated as an optimization problem with the cost function $J(\phi)$ stated as follows:

$$J(\phi)=\frac{1}{N}\sum_{k=1}^{N} e^2(k)=\frac{1}{N}\sum_{k=1}^{N}(d(k)-y(k))^2, \tag{3}$$

where $d(k)$ and $y(k)$ are the desired and actual responses of the filter, respectively; and $e(k)=d(k)-y(k)$ is the filter's error signal; N is the number of samples used to calculate the objective function.

$$\phi=[\mathbf{a}^T\mathbf{b}^T]^T=[a_0 a_1 \ldots a_L b_1 \ldots b_M]^T \tag{4}$$

denotes the filter coefficient vector. The aim is to minimize the cost function $J(\phi)$ by adjusting ϕ. An important consideration during the adaptive process is to maintain the stability of the IIR filter. Not all filters defined by Eq. (1) are feasible or implementable.

Table 7. Effects of ζ and p_2 on search accuracy of CPDE.

Inst.	$\zeta=1,p_2=5$	$\zeta=1,p_2=1$	$\zeta=4,p_2=1$	$\zeta=4,p_2=5$
	Mean Error ± Std Dev	**Mean Error ± Std Dev**	**Mean Error ± Std Dev**	**Mean Error ± Std Dev**
Example 1	9.8904E–011.2771E–02≈	9.8780E–011.1698E–02	9.9147E–01±7.5527E–03+	9.8494E–01±1.2588E–02
Example 2	5.1568E–01±2.8237E–02	5.3038E–01±1.7731E–02+	5.2603E–01±2.1477E–02+	5.1608E–01±1.8810E–02
Example 3	5.1935E–01±1.6915E–02+	5.0609E–01±2.3654E–02	5.2400E–01±2.2003E–02+	5.0512E–01±2.2874E–02
Example 4	5.2926E–01±2.5568E–02	5.2539E–01±2.5586E–02≈	5.2821E–01±2.5189E–02≈	5.2156E–01±2.1444E–02
Example 5	5.1930E–01±2.5434E–02+	5.1788E–01±2.3220E–02+	5.2145E–01±3.0849E–02+	5.0945E–01±1.6459E–02
Example 6	9.7708E–01±1.2567E–02≈	9.7858E–01±1.1414E–02≈	9.7989E–01±8.3052E–03≈	9.7624E–01±7.8179E–02

doi:10.1371/journal.pone.0040549.t007

An efficient way of maintaining stability is to convert the direct form to a lattice form and make sure that all reflection coefficients k_i, $0 \le i \le M-1$, have magnitudes less than one. We will adopt a similar approach as in [11] to guarantee the stability of the IIR filter during adaptation. Thus, the actual filter coefficient vector used in optimization is $\phi = [a_0 a_1 \ldots a_L k_0 \ldots k_{M-1}]^T$. In the circumstances, the coefficient space Ψ is formed by the constraints of $a_i \in [-2,2] (i=1,\ldots,L)$ and the magnitudes of k_i that are less than one. For the sake of simplicity, we adopt the predefined boundary constraints as $[-2,2]$ to compare other existing EAs fairly.

0.6 A Controllable Probabilistic DE

The stochastic system is iterated forward in time using a synchronous DE updating scheme. In our work, a mode-dependent population updating equation with Markovian switching parameters is introduced with the hope to keep track of the progress of individuals and further improve the search abilities. A detailed algorithm design of CPDE can be found in **Documentation S1**.

In CPDE, there are two levels of sub-optimizers, population decreasing strategy (PDS) and population increasing strategy (PIS). The probability of selecting different sub-optimizer to improve the online solution-searching status is completely up to its non-homogeneous Markov chain. For choosing required sub-optimizers adaptively, consider the following probability transition matrix in Eq. (5):

$$\Pi = \begin{bmatrix} \varphi_a & \varphi_b & \varphi_c \\ 1-\varphi_c & 0 & \varphi_c \\ 1-\varphi_b & \varphi_b & 0 \end{bmatrix}, \tag{5}$$

Here, $\mathcal{M}(1)=a$, $\mathcal{M}(2)=b$ and $\mathcal{M}(3)=c$ stand for the population maintaining state, population increasing state and population decreasing state, respectively. The expectations of Markov chain Π are automatically updated by the search environment. It is worthwhile to mention that π_{22} and π_{33} are set to be 0. In this case, the increasing and decreasing operators will not be performed in consecutive generations.

If few trial vectors can outperform the corresponding parent in selection operation, particles may be clustered together and trapped into the local basin. In such a case, the PIS is employed to add new individuals into the population and share their up-to-date information to help the individuals escape the local basin.

$$\delta_2 = a \cdot \left[\frac{(1-\lambda(G-\frac{1}{\lambda}))^2}{(1-\lambda(G-\frac{1}{\lambda}))^2 + (1+\lambda(G-\frac{1}{\lambda}))^2} + b \right], \tag{6}$$

where δ_2 is the number of dimensions for perturbation, which is monotonically decreasing through the evolutionary search. Parameters a and b are the magnification coefficient. Parameters λ and G are here considered as the generation variables. During early stages of the optimization process, much more reproductions will be generated to spread out its particles within the decision space. Nevertheless, solutions of the population tend to concentrate in specific parts of the decision space during the later period of optimization process.

However, in case most of individuals can spawn new promising offspings in the evolutionary process, it then signifies that redundant intermediate particles exist. In this case, we introduce the PDS to remove poor particles to avoid undesirable computational cost and excessive search complexity.

$$\tau = \left(1 - \frac{1}{\mathrm{rank}_i + 1}\right) \times (1 - H_i(D)), \tag{7}$$

where τ denots an overall deletion indicator. The variable $rank_i$ and $H_i(D)$ indicate the rank metric and the entropy metric for individual i, respectively. It can be observed that from Eq. (7), for the individuals that have high rank values (i.e., away from the global best solution) or low entropy values (i.e., located in the crowded regions); these particles will have a higher probability of elimination.

Supporting Information

Documentation S1 A material algorithm design of CPDE.
(PDF)

Author Contributions

Conceived and designed the experiments: WZ JAF YT. Performed the experiments: WZ JAF. Analyzed the data: WZ WD. Contributed reagents/materials/analysis tools: WBZ YT. Wrote the paper: WZ YT.

References

1. Li Z, O'Doherty JE, Hanson TL, Lebedev MA, Henriquez CS, et al. (2009) Unscented kalman filter for brain-machine interfaces. PLoS ONE 4: e6243.
2. Ralph JF, Oxtoby NP (2011) Quantum filtering one bit at a time. Physical Review Letters 107: 274101.
3. Pajevic S, Plenz D (2009) Efficient network reconstruction from dynamical cascades identifies small-world topology of neuronal avalanches. PLoS Computational Biology 5: e1000271.
4. Tang Y, Gao H, Kurths J, Fang J (2012) Evolutionary pinning control and its application in uav coordination. IEEE Transactions on Industrial Informatics: DOI:10.1109/TII.2012.2187911.
5. Luukko J, Rauma K (2008) Open-loop adaptive filter for power electronics applications. IEEE Transactions on Industrial Electronics 55: 910–917.
6. Arenas A, Diaz-Guilera A, Kurths J, Moreno Y, Zhou C (2008) Synchronization in complex networks. Physics Reports 469: 93–153.
7. Liu YH, Kuo CY, Chang CC, Wang CY (2011) Electro-osmotic flow through a two-dimensional screen-pump filter. Physical Review E 84: 036301.
8. Luque SP, Fried R (2011) Recursive filtering for zero offset correction of diving depth time series with gnu r package divemove. PLoS ONE 6: e15850.
9. Tumminello M, Michele LF, Mantegna RN (2007) Kullback-leibler distance as a measure of the information filtered from multivariate data. Physical Review E 76: 031123.
10. Yu D, Parlitz U (2011) Inferring network connectivity by delayed feedback control. PLoS ONE 6: e24333.
11. Chen S, Istepanian RH, Luk BL (2001) Digital iir filter design using adaptive simulated annealing. Digital Signal Processing 11: 241–251.
12. Karaboga N, Kalinli A, Karaboga D (2004) Designing iir filters using ant colony optimisation algorithm. Engineering Applications of Artificial Intelligence 17: 301–309.
13. Krusienski DJ, Jenkins WK (2004) Particle swarm optimization for adaptive iir filter structures. In: Congress on Evolutionary Computation, CEC2004. 2805–2810.
14. Dai C, Chen W, Zhu Y (2010) Seeker optimization algorithm for digital iir filter design. IEEE Transactions on Industrial Electronics 51: 1710–1718.
15. Karaboga N, Cetinkaya MB (2011) A novel and efficient algorithm for adaptive filtering:artificial bee colony algorithm. Turkish Journal of Electrical Engineering & Computer Sciences 19: 175–190.
16. Karaboga N (2009) A new design method based on artificial bee colony algorithm for digital iir filters. Journal of the Franklin Institute 346: 328–348.
17. Karaboga N (2005) Digital iir filter design using differential evolution algorithm. EURASIP Journal on Applied Signal Processing 8: 1269–1276.
18. Dorigo M, Stutzle T (2004) Ant Colony Optimization. MIT Press, Cambridge.
19. Tang Y, Wang Z, Fang J (2011) Feedback learning particle swarm optimization. Applied Soft Computing 11: 4713–4725.
20. Dai C, Chen W, Li L, Zhu Y, Yang Y (2011) Seeker optimization algorithm for parameter estimation of time-delay chaotic systems. Physical Review E 83: 036203.

21. Storn R, Price KV (1995) Differential evolution–A simple and efficient adaptive scheme for global optimization over continuous spaces. ICSI, Bloomington, MN, Tech. Rep. TR-95-012. [Online]. Available: http://http.icsi.berkeley.edu/~storn/litera.html.
22. Ghosh S, Das S, Vasilakos AV, Suresh K (2012) On convergence of differential evolution over a class of continuous functions with unique global optimum. IEEE Transaction on Systems Man and Cybernetics: Part B 42: 107–124.
23. Das S, Suganthan PN (2011) Differential evolution: A survey of the state-of-the-art. IEEE Transactions on Evolutionary Computation 15: 4–31.
24. Chakraborty UK (2008) Advances in Differential Evolution, studies in computational intelligence. Springer, Berlin.
25. Price KV, Chakraborty UK (2008) Eliminating drift bias from the differential evolution algorithm. Advances in Differential Evolution, Studies in Computational Intelligence 143: 33–88.
26. Teng NS, Teo J, Hijazi M (2009) Self-adaptive population sizing for a tune-free differential evolution. Soft Computing 13: 709–724.
27. Osipov GV, Kurths J, Zhou C (2007) Synchronization in Oscillatory Networks. Springer, Berlin.
28. du Plessis MC, Engelbrecht AP (2012) Using competitive population evaluation in a differential evolution algorithm for dynamic environments. European Journal of Operational Research 218: 7–20.
29. Yu D (2010) Estimating the topology of complex dynamical networks by steady state control: Generality and limitation. Automatica 46: 2035–2040.
30. Brest J, Maucec MS (2011) Self-adaptive differential evolution algorithm using population size reduction and three strategies. Soft Computing 15: 2157–2174.
31. Lin WZ, Fang JA, Xiao X, Chou KC (2011) idna-prot: Identification of dna binding proteins using random forest with grey model. PLoS ONE 6: e24756.
32. Xiao X, Wang P, Chou KC (2011) Gpcr-2l: predicting g protein-coupled receptors and their types by hybridizing two different modes of pseudo amino acid compositions. Molecular Biosystems 7: 911–919.
33. Xiao X,Wang P, Chou KC (2012) inr-physchem: A sequence-based predictor for identifying nuclear receptors and their subfamilies via physical-chemical property matrix. PLoS ONE 7: e30869.
34. Wang P, Xiao X, Chou KC (2011) Nr-2l: A two-level predictor for identifying nuclear receptor subfamilies based on sequence-derived features. PLoS ONE 6: e23505.
35. Wu ZC, Xiao X, Chou KC (2011) iloc-plant: a multi-label classifier for predicting the subcellular localization of plant proteins with both single and multiple sites. Molecular BioSystems 7: 3287–3297.
36. Wu ZC, Xiao X, Chou KC (2012) iloc-gpos: A multi-layer classifier for predicting the subcellular localization of singleplex and multiplex gram-positive bacterial proteins. Protein & Peptide Letters 19: 4–14.
37. Xiao X, Wu ZC, Chou KC (2011) A multi-label classifier for predicting the subcellular localization of gramnegative bacterial proteins with both single and multiple sites. PLoS ONE 6: e20592.
38. Chou KC, Wu ZC, Xiao X (2012) iloc-hum: Using accumulation-label scale to predict subcellular locations of human proteins with both single and multiple sites. Molecular Biosystems 8: 629–641.
39. Xiao X, Wu ZC, Chou KC (2011) iloc-virus: A multi-label learning classifier for identifying the subcellular localization of virus proteins with both single and multiple sites. Journal of Theoretical Biology 284: 42–51.
40. Tang Y, Wang Z, Fang J (2009) Pinning control of fractional-order weighted complex networks. CHAOS 19: 013112.
41. Schumann-Bischoff J, Parlitz U (2011) State and parameter estimation using unconstrained optimization. Physical Review E 84: 056214.
42. Mallipeddi R, Suganthana PN, Pan QK, Tasgetiren MF (2011) Differential evolution algorithm with ensemble of parameters and mutation strategies. Applied Soft Computing 11: 1679–1696.
43. Qin AK, Huang VL, Suganthan PN (2009) Differential evolution algorithm with strategy adaptation for global numerical optimization. IEEE Transactions on Evolutionary Computation 13: 398–417.
44. Bostani N, Kessler DA, Shnerb NM, Rappel WJ, Levine H (2012) Noise effects in nonlinear biochemical signaling. Physical Review E 85: 011901.
45. Gao J, Hu J, Tung W (2011) Facilitating joint chaos and fractal analysis of biosignals through nonlinear adaptive filtering. PLoS ONE 6: e24331.
46. Peng HP, Li LX, Yang YX, Sun F (2011) Conditions of parameter identification from time series. Physical Review E 83: 036202.
47. Lu LY, Liu WP (2011) Information filtering via preferential diffusion. Physical Review E 83: 066119.
48. Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Transactions on Evolutionary Computation 1: 67–82.
49. Merlin JC, Rajasekaran S, Mi T, Schiller MR (2012) Reducing false-positive prediction of minimotifs with a genetic interaction filter. PLoS ONE 7: e32630.
50. van Dijk ADJ, van Mourik S, van Ham RCHJ (2012) Mutational robustness of gene regulatory networks. PLoS ONE 7: e30591.
51. Kiss IZ, Rusin C, Kori H, Hudson JL (2007) Engineering complex dynamical structures: Sequential patterns and desynchronization. Science 316: 1886–1889.
52. Achard S, Bullmore E (2007) Efficiency and cost of economical brain functional networks. PLoS Computational Biology 3: e17.
53. Chang X, Liu S, Yu YT, Li YX, Li YY (2010) Identifying modules of coexpressed transcript units and their organization ofsaccharopolyspora erythraea from time series gene expression profiles. PLoS ONE 5: e12126.
54. Hansen N, Ostermeier A (2001) Completely derandomized selfadaptation in evolution strategies. Evolutionary Computation 9: 159–195.
55. Garcia-Martinez C, Lozano M, Herrera F, Molina D, Sanchez AM (2008) Global and local real-coded genetic algorithms based on parent-centric crossover operators. European Journal of Operational Research 185: 1088–1113.
56. Brest J, Greiner S, Boscovic B, Mernik M, Zumer V (2006) Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems. IEEE Transactions on Evolutionary Computation 10: 646–657.