IOP Conf. Series: Materials Science and Engineering **210** (2017) 012060 doi:10.1088/1757-899X/210/1/012060

Chaos-enhanced Stochastic Fractal Search algorithm for Global Optimization with Application to Fault Diagnosis

Tuan A Z Rahman[1], N A Abdul Jalil[1], A As'arry[1] and R K Raja Ahmad[2]

[1]Sound and Vibration Research Group, Department of Mechanical and Manufacturing Engineering, Faculty of Engineering, Universiti Putra Malaysia (UPM) 43400 Serdang, Selangor, Malaysia
[2]Department of Electrical and Electronics Engineering, Faculty of Engineering, Universiti Putra Malaysia (UPM) 43400 Serdang, Selangor, Malaysia

E-mail: tuanzahidi@upm.edu.my

Abstract. Support vector machine (SVM) has been known as one-state-of-the-art pattern recognition method. However, the SVM performance is particularly influenced byits parameter selection. This paper presents the parameter optimization of an SVM classifier using chaos-enhanced stochastic fractal search (SFS) algorithm to classify conditions of a ball bearing. The vibration data for normal and damaged conditions of the ball bearing system obtained from the Case Western Reserve University Bearing Data Centre. Features based on time and frequency domains were generated to characterize the ball bearing conditions. The performance of chaos-enhanced SFS algorithms in comparison to their predecessor algorithm is evaluated. In conclusion, the injection of chaotic maps into SFS algorithm improved its convergence speed and searching accuracy based on the statistical results of CEC 2015 benchmark test suites and their application to ball bearing fault diagnosis.

1. Introduction

Vibration-based condition monitoring (CM) currently plays an important role in manufacturing industries to continuous surveillanceof rotating machinery compared to the conventional methods. Reducing the productivity costs by minimizing themachine's downtime is one of its crucial benefits in order to enhance goods production.As vibration-based condition monitoring area can be treated as pattern recognition, support vector machines (SVMs) approach is commonly used. However, the SVMs performance is really dependable to its kernel and parameters selection. This problem can be formulated as an optimization problem. Thus, many optimization algorithms were applied to overcome this essential problem, systematically.

Zhang *et al.* proposed a hybrid method of the barebones differential evolution (BBDE) algorithm for SVMs parameters tuning [1]. Several others evolutionary algorithms (EAs) have been employed to optimize the SVMs parameters such as genetic algorithm (GA) [2], particle swarm optimization (PSO)[3], ant colony optimization (ACO) [4]and artificialimmunization (AIA) algorithms [5].

In this paper, four chaos-enhanced stochastic fractal search (SFS) algorithms are presented. The performance of these improved variants of SFS algorithms was evaluated using modern benchmark test suites (CEC 2015) and the SVMs parameters tuning for fault diagnosis as its engineering application.

2. Enhanced Stochastic Fractal Search algorithm with chaos

Stochastic fractal search (SFS) optimization algorithm was developed by Salimi to imitate a growth process [6]. The particles in the process try to expand its growing in searching space. This metaheuristic algorithm shows promising results with short computational time. The main advantage of this algorithm is less starting tuning parameters to initiate the searching process.There are two main equations from the original SFS algorithm have been modified with the introduction of a chaotic variable named as α as follow:

$$GW = Gaussian(\mu_{BP}, \sigma) + (\varepsilon \times BP - \alpha \times P_i) \quad (1)$$

and,

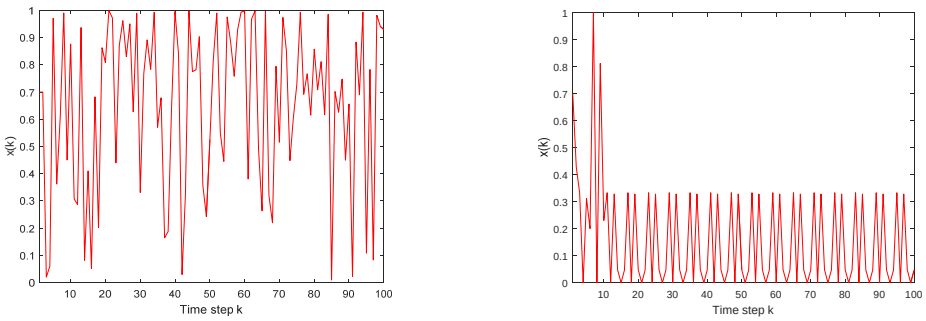
$$P'_i(j) = P_r(j) - \alpha \times (P_t(j) - P_i(j)) \quad (2)$$

Noted that, eq. 1 was part of Diffusion Process while eq. 2 in First Updating Process in the original SFS algorithm. This study investigated two different one-dimension non-invertible chaotic maps influence toward the algorithm performance as proposed in previous studies by Saremi*et al.*[7] and Mitic*et al.*[8]. These new chaotic equations will force the particles to move towards the current best optimal solution in a chaotic manner. Table 1 tabulates the mathematical description of the proposed addition of chaotic maps while Figure 1 shows the graphical presentation of these maps over 100 generations. Each of the chaotic maps has the starting point 0.7 and normalized to a range of [0, 1].

Table 1. The mathematical description of chaotic maps

No	Map Name	Equation
1	Chebyshev	$x_{i+1} = \cos(i \cos^{-1}(x_i))$
2	Gauss/Mouse	$x_{i+1} = \begin{cases} 1, for x_i = 0 \\ \frac{1}{\mathrm{mod}(x_i, 1)}, otherwise \end{cases}$

Four chaos-enhanced stochastic fractal search (CFS) algorithms were developed with the implementation of two different chaotic maps at two different parts of SFS algorithm. The initial value of 0.95 was selected for parameter α as suggested in [8]. Table 2 shows the combination of chaotic maps for CFS algorithms. More details regarding the CFS algorithms can be found in [9].



a) Chebyshev map b) Gauss/Mouse map

Figure 1. Visualization of chaotic maps used

Table 2. Chaos-enhanced stochastic fractal search (CFS) algorithms

Algorithm	Diffusion Process	Updating Process
Original SFS	Random [0, 1]	Random [0, 1]
CFS01	Chebyshev map	Chebyshev map
CFS02	Chebyshev map	Gauss/Mouse map
CFS03	Gauss/Mouse map	Chebyshev map
CFS04	Gauss/Mouse map	Gauss/Mouse map

In this study, 4 modern benchmark test functions from CEC 2015 were used to evaluate the performance of CFS algorithms. Two different levels of problem dimension, *D* were investigated which are 10 and 30. The number of population (Start Point), *NP* was fixed to 100. Maximum Diffusion Number (*MDN*) equal to 1 with the first Gaussian walk is utilized. Note that, all calculations were performed in MATLAB 2015b software that runs on a desktop PC Intel ® Core ™ i5 of CPU 3.30 GHz with 4GBs RAM and Window 7 (64 bit) operating system. The full results of CFS algorithms evaluation using modern benchmark functions are tabulated in Table 3a and 3b. Non-parametric statistical analysis was conducted based on a total of 50 simulation runs.

3. Support vector machines

Support vector machines (SVMs) can be simplest discussed using a description of linear discriminant analysis. A hyperplane (i.e. a straight line in two dimensions) will be created to separate two classes of data that generalized best (maximum margin). The hyperplane equation as follow,

$$D(x) = \langle w, x \rangle = 0 \tag{3}$$

where*x* is the input vector and w is the vector of free weights. Each data point x_k in the training set is assigned to a class on the basis of the separating condition given by,

$$x_k \in C_1 \Rightarrow D(x_k) = \left\langle w, x_k \right\rangle \geq 1$$

$$x_k \in C_2 \Rightarrow D(x_k) = \left\langle w, x_k \right\rangle \geq -1 \tag{4}$$

whereC_1 and C_2 are classes, with respective class labels (1) and (-1), or more concisely as,

$$D\left(x_k\right) = y_k \left\langle w, x_k \right\rangle \geq 1 \tag{5}$$

wherey_k is the class label. The distance of each point in the training set from the separating hyperplane is,

$$\frac{D\left(x_k\right)}{\|w\|} \tag{6}$$

An interval that contains the separating hyperplane but excluding all data is called margin, τif,

$$y_k \frac{D\left(x_k\right)}{\|w\|} \geq \tau \tag{7}$$

is satisfied for all *k*. The optimal margin is illustrated in Figure 2. Note that the parameterization of the hyperplane is currently arbitrary. This can be fixed by specifying,

$$\|w\|\tau=1 \tag{8}$$

and this converts eq. (7) into the separation condition of eq. (5).

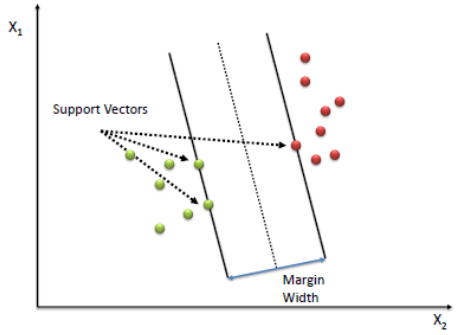


Figure 2. Optimal separating hyperplane

The objective of SVMs is to maximize the margin so that the hyperplane will be placed at the furthest point from data and can be achieved by minimizing $\|w\|$. An appropriate objective function is,

$$Q(w)=\frac{1}{2}\|w\|^2-\sum_{i=1}^{N}\alpha_i y_i\left(\langle w,x_i\rangle-1\right) \tag{9}$$

where the parameters α_i are Lagrange multipliers. Further discussion can be found in [10] regarding convex and the dual formulation of the problem. The analysis so far only considers linear hyperplane. The technique can be made nonlinear by the use of kernel trick which involves quadratic, polynomial and radial basis function (RBF) kernels.

4. Application of CFS algorithms to fault diagnosis

Theapplication of vibration signals is relatively usual in condition monitoring and damage detection area. Signal processing techniques is an important expertise in order to extract diagnosis information from the selected raw vibration data. Features extraction and selection phase are performed to characterize the condition before pattern recognition methods were employed to identify the damages.

Seed faults bearing data from Case Western Reserve University was used in this study [11]. Figure 3 shows the schematic diagram of the experimental setup. Three faults which are outer race fault (OR), inner race (IR) fault and ball (B) fault were introduced to the drive end bearing. The defects sizes are 0.007 inches and 0.021 inches. Each original signalwas divided into 10 signals for each condition of fault types and sizes. Four features (two time-domain and two frequency-domain features) were calculated based on [1].

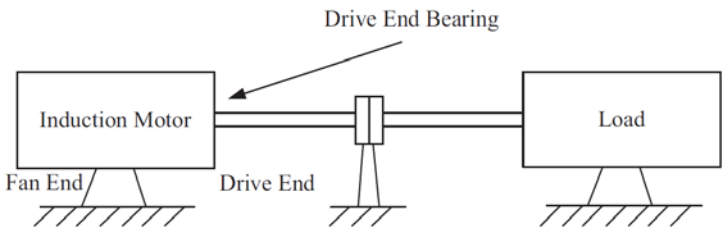


Figure 3. The schematic diagram of the experimental setup (adapted from[1])

Table 3a. Statistical result tested on CEC 2015 benchmark functions (200 Generations)

Function	Dimension		SFS	CFS 01	CFS 02	CFS 03	CFS 04
F1	10	Mean	2.6221e+04	722.9056	**216.5546**	706.9905	316.8680
		SD	1.2390e+04	676.7885	**387.5189**	520.7234	1.1308e+03
	30	Mean	4.3292e+06	**2.9398e+06**	2.9917e+06	2.9521e+06	3.4427e+06
		SD	1.6492e+06	1.6966e+06	1.9840e+06	**1.3451e+06**	1.7882e+06
F3	10	Mean	19.5146	20.2153	**18.8614**	19.0280	20.0611
		SD	3.5328	0.0544	4.8138	4.7194	**0.0471**
	30	Mean	20.7685	20.7608	20.6563	20.7641	**20.6341**
		SD	0.0610	0.0598	0.0648	**0.0566**	0.0806
F6	10	Mean	309.5684	52.8969	**7.0392**	45.0227	8.6524
		SD	77.9991	19.4256	**4.1259**	18.4530	5.5534
	30	Mean	1.3456e+05	1.3207e+04	1.1302e+04	1.1861e+04	**1.0792e+04**
		SD	6.6003e+04	5.9933e+03	7.5751e+03	**5.1988e+03**	7.6252e+03
F10	10	Mean	280.5511	219.9898	**217.5860**	220.0394	217.6341
		SD	20.2745	1.3676	**0.9332**	1.2873	1.0027
	30	Mean	5.8437e+04	5.8325e+03	5.7653e+03	**5.0251e+03**	8.5475e+03
		SD	2.2595e+04	1.8679e+03	7.7974e+03	**1.7204e+03**	1.3749e+04
Algorithm ranking based on average mean value computed via Friedman test			4.5975 (5)	3.2900 (4)	**1.9400 (1)**	3.1950 (3)	1.9775 (2)

Table 3b. Non-Parametric test results based on Table 3a

No. of function	Dimension	Algorithm	Friedman test		Wilcoxon signed rank test			
			Mean rank	*p*	SFS-CFS 01	SFS-CFS 02	SFS-CFS 03	SFS-CFS 04
					p	*p*	*p*	*p*
F1	10	SFS	5.0000	<0.05 7.1986e-31	7.5569e-10	7.5569e-10	7.5569e-10	7.5569e-10
		CFS 01	3.2800					
		CFS 02	1.7200					
		CFS 03	3.2600					
		CFS 04	1.7400					
	30	SFS	4.0000	<0.05 1.2016e-05	2.2093e-05	3.3554e-04	4.0002e-05	**0.0036**
		CFS 01	2.5200					
		CFS 02	2.7400					
		CFS 03	2.6800					

		CFS 04	3.0600					
F3	10	SFS	4.0000	<0.05 1.3843e-26	**0.9040**	1.4705e-07	**0.1463**	2.0113e-07
		CFS 01	3.9600					
		CFS 02	1.6000					
		CFS 03	3.8000					
		CFS 04	1.6400					
	30	SFS	3.8000	<0.05 1.5408e-19	**0.4486**	1.6775e-08	**0.9116**	4.2535e-09
		CFS 01	3.7000					
		CFS 02	2.0000					
		CFS 03	3.8600					
		CFS 04	1.6400					
F6	10	SFS	5.0000	<0.05 5.9277e-38	7.5569e-10	7.5569e-10	7.5569e-10	7.5569e-10
		CFS 01	3.6000					
		CFS 02	1.4600					
		CFS 03	3.4000					
		CFS 04	1.5400					
	30	SFS	5.0000	<0.05 4.1620e-22	7.5569e-10	7.5569e-10	7.5569e-10	7.5569e-10
		CFS 01	2.9400					
		CFS 02	2.2600					
		CFS 03	2.5600					
		CFS 04	2.2400					
F10	10	SFS	5.0000	<0.05 7.4688e-32	7.5569e-10	7.5569e-10	7.5569e-10	7.5569e-10
		CFS 01	3.1800					
		CFS 02	1.7800					
		CFS 03	3.4200					
		CFS 04	1.6200					
	30	SFS	4.9800	<0.05 1.9589e-23	7.5569e-10	7.5569e-10	7.5569e-10	8.0311e-10
		CFS 01	3.1400					
		CFS 02	1.9600					
		CFS 03	2.5800					
		CFS 04	2.3400					

Multiclass SVMs based on one-against-one technique is employed to classify the bearing conditions. One-third of each class fault is used for training the SVM and the remaining data forthe testing purposed. The training and testing data were selected randomly and repeated for 30 times before average classification error is calculated as the objective function. CFS algorithms were evaluated to optimize two SVMs parameters which are the soft margin/penalty parameter, C and the scaling factor for RBF-kernel, γ. Then, the obtained parameters were used to generate an average classification error of 10,000 SVMs runs in verification stage. The performances of each CFS algorithm are compared to their predecessor algorithm. Figure 4plotted the features data based on 1st and 2nd principle components for visualization purpose. All data have seen clearly separated between normal and damaged conditions. For Inner Race (IR021) and Outer Race (OR021) of fault size 0.021 inch, the data can be divided in 3-Dimension view. The original normalized four features data were used to generate the SVMs classifier model.

The initial parameters of SFS and CFS algorithms were set as follow; starting point (NP) = 100 particles, number of maximum iterations (G) = 50 generations, problem dimension (D) = 2, searching range of [0 150] for the soft margin/penalty (C) and [0 10] for the scaling factor of RBF-kernel (γ).Maximum Diffusion Number (MDN) was set as 1 while 1st Gaussian Walk is selected.

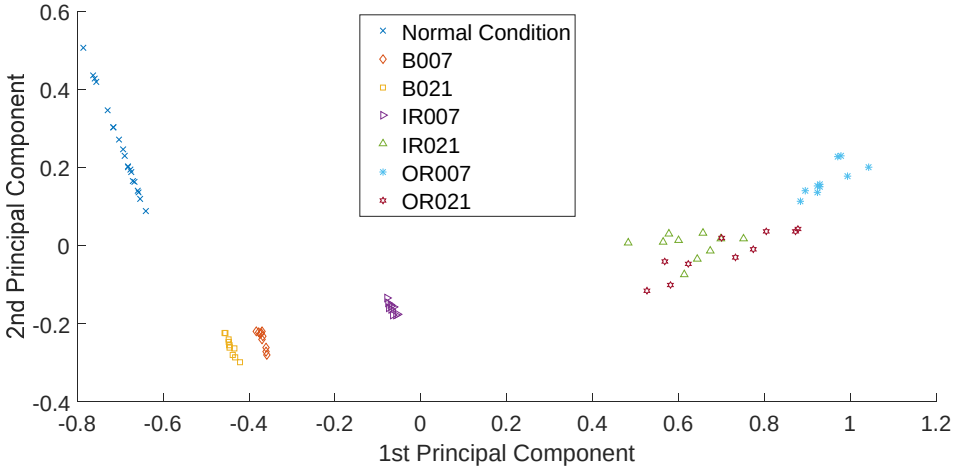


Figure 4. The visualization of features data

5. Results and discussion

Table 4 tabulates the results of bearing fault classification using SVMs-based SFS algorithms.The second and third column shows the obtained scaling factor, γ and soft margin, C respectively. The average of percentage classification accuracy for 10,000 runs is shown in the fourth column with its standard deviation. Based on classification error in the fifth column, CFS 04 algorithm performance was better than other CFS and its predecessor algorithms. The CFS 04 algorithm has achieved 99.992% classification accuracy with the lowest standard deviation of 0.18% in comparison with others. On the other hand, the addition of Chebyshev map in Diffusion and First Updating Processes of SFS algorithm has slightly deteriorated its performance.

The benchmark test suites of CEC 2015 and bearing fault classification results indicated that CFS 04 algorithm shows better searching accuracy compared to the SFS and its other chaos-enhanced algorithms.

Table 4. CFS algorithms performance in comparison to SFS

Algorithm	Scaling Factor (γ)	Soft Margin, (*C*)	Accuracy (%)	Classification Error
SFS	1.8438	0.3771	99.971±0.340	2.8750×10^{-4}
CFS 01	5.3945	4.0529	99.746±1.240	2.5375×10^{-3}
CFS 02	2.3719	1.0194	99.986±0.240	1.3750×10^{-4}
CFS 03	1.6727	1.1077	99.984±0.260	1.5833×10^{-4}
CFS 04	1.9870	1.1471	**99.992±0.180**	**7.0833×10^{-5}**

6. Conclusion

In this study, four variants of SFS algorithm enhanced with chaos is introduced. Their searching accuracy and convergence speed performance were evaluated using modern benchmark test suites of CEC 2015 and engineering application to ball bearing fault diagnosis. CFS algorithm with Gauss/Mouse map in Diffusion and First Updating Processes show superiority performance in comparison to its predecessor and other CFS algorithms.

Acknowledgment

The first author would like to acknowledge Universiti Putra Malaysia (UPM) and Ministry of Higher Education (MOHE) Malaysia for the PhD scholarship funding.

References

[1] X. Zhang, D. Qiu, and F. Chen, Support vector machine with parameter optimization by a novel hybrid method and its application to fault diagnosis, *Neurocomputing,* vol. 149, Part B, pp. 641-651, 2/3/ 2015.

[2] A. C. Lorena and A. C. P. L. F. de Carvalho, Evolutionary tuning of SVM parameter values in multiclass problems, *Neurocomputing,* vol. 71, pp. 3326-3334, 10// 2008.

[3] S.-W. Lin, K.-C. Ying, S.-C. Chen, and Z.-J. Lee, Particle swarm optimization for parameter determination and feature selection of support vector machines, *Expert Systems with Applications,* vol. 35, pp. 1817-1824, 11// 2008.

[4] X. Zhang, X. Chen, and Z. He, An ACO-based algorithm for parameter optimization of support vector machines, *Expert Systems with Applications,* vol. 37, pp. 6618-6628, 9// 2010.

[5] I. Aydin, M. Karakose, and E. Akin, A multi-objective artificial immune algorithm for parameter optimization in support vector machine, *Applied Soft Computing,* vol. 11, pp. 120-129, 1// 2011.

[6] H. Salimi, Stochastic Fractal Search: A powerful metaheuristic algorithm, *Knowledge-Based Systems,* vol. 75, pp. 1-18, 2// 2015.

[7] S. Saremi, S. Mirjalili, and A. Lewis, Biogeography-based optimisation with chaos, *Neural Computing and Applications,* vol. 25, pp. 1077-1097, 2014/10/01 2014.

[8] M. Mitić, N. Vuković, M. Petrović, and Z. Miljković, Chaotic fruit fly optimization algorithm, *Knowledge-Based Systems,* vol. 89, pp. 446-458, 11// 2015.

[9] T. A. Z. Rahman and M. O. Tokhi, Enhanced stochastic fractal search algorithm with chaos, in *2016 7th IEEE Control and System Graduate Research Colloquium (ICSGRC)*, 2016, pp. 22-27.

[10] N. Cristianini and J. Shawe-Taylor, *An introduction to support Vector Machines: and other kernel-based learning methods*: Cambridge University Press, 2000.

[11] Case Western Reserve University bearing data centre website, 2017 http://csegroups.case.edu/bearingdatacenter/pages/welcome-case-western-reserve-university-bearing-data-center-website