Separability Detection Cooperative Particle Swarm Optimizer based on Covariance Matrix Adaptation

Sheng-Fuu Lin, Yi-Chang Cheng, Jyun-Wei Chang, and Pei-Chia Hung
Department of Electrical Engineering
National Chiao Tung University
Hsinchu, Taiwan

***Abstract*—The particle swarm optimizer (PSO) is a population-based optimization technique that can be widely utilized to many applications. The cooperative particle swarm optimization (CPSO) applies cooperative behavior to improve the PSO on finding the global optimum in a high-dimensional space. This is achieved by employing multiple swarms to partition the search space. However, independent changes made by different swarms on correlated variables will deteriorate the performance of the algorithm. This paper proposes a separability detection approach based on covariance matrix adaptation to find non-separable variables so that they can previously be placed into the same swarm to address the difficulty that the original CPSO encounters.**

Keywords- cooperative behavior; particle swarm optimization; covariance matrix adaptation; separability.

I. INTRODUCTION

The particle swarm optimizer (PSO) [1, 2] is a stochastic, population-based optimization learning algorithm. Its learning procedure is based on a population made of individuals with specific behaviors similar to certain biological phenomena. Individuals keep exploring the solution space and exploiting information between individuals while evolution proceeding. In general, by means of exploring and exploiting, the PSO is less likely to be trapped at the local optimum.

As with many stochastic optimization algorithms [1, 3-6], the PSO suffers from the "curse of dimensionality," which implies that its performance deteriorates as the dimensionality of the search space increases. To cope with this difficulty, Potter [3] proposed a cooperative coevolutionary genetic algorithm (CCGA) that partitions the search space by splitting the solution vectors into smaller ones. The mechanism proposed by Potter significantly improves the performance of the original GA. Van den Bergh [5] applies this technique to the PSO and presented several cooperative PSO models named CPSOs. In the CPSOs learning procedure, the search space can be arbitrarily partitioned into different number of subspaces. Each smaller search space is then searched by a separate swarm. The fitness function is evaluated by the context vector, which means the concatenation of particles found by each of the swarms. However, as with the CCGA algorithm, the performance of the CPSO deteriorates when correlated variables are placed into separate populations. In this paper, we call such variables "non-separable." A function f is said to be separable if

$$\begin{aligned}&\arg\min_{(x_1,\ldots,x_n)} f(x_1,\ldots,x_n)\\&=(\arg\min_{x_1} f(x_1,\ldots),\ldots,\arg\min_{x_n} f(\ldots,x_n))\end{aligned},\quad(1)$$

and it is followed by a fact that *f* can be optimized in a sequence of *n* independent 1-*D* optimization processes. This paper proposes a variation on the original CPSO to detect the separability of the variables. To this end, we adopt a mechanism from evolution strategy with covariance matrix adaption (CMA-ES) [8, 9]. The performance of the CPSO after applying separability detection is compared with that of the traditional PSO and CPSO algorithm.

This paper is organized as follows. Section II presents an overview of the PSO and the CPSO. In section III, we describe the proposed separability detection cooperative particle swarm optimizer (SD-CPSO). This is followed by the experiment results presented in section VI. Finally, some directions for the future research are discussed in section V.

II. RELATED WORKS

The PSO is first introduced by Kennedy and Eberhart. It's one of the most powerful methods for solving global optimization problems. The algorithm searches an optimal point in a multi-dimensional space by adjusting the trajectories of its particles. The individual particle updates its position and velocity based on its personal best performance and the global best performance among all particles that denote *y* and respectively. The position $x_{i,d}$ and velocity $v_{i,d}$ of the *d*-th dimension of *i*-th particle are updated as follows:

$$\begin{aligned}v_{i,d}(t+1) &= v_{i,d}(t)+c_1\cdot rand_1\cdot(y_{i,d}(t)-x_{i,d}(t))\\&\quad+c_2\cdot rand_2\cdot(\hat{y}_d(t)-x_{i,d}(t)),\end{aligned}\quad(2)$$

$$x_i(t+1)=x_i(t)+v_i(t+1),\quad(3)$$

where y_i represents the previous best position yielding the best performance of the *i*-th particle; c_1 and c_2 denote the acceleration constants describing the weighting of each particle been pulled toward *y* and $\hat{y}$ respectively; $rand_1$ and $rand_2$ are two random numbers in the range [0, 1].

Let *s* denote the swarm size and *f*() denote the fitness function evaluating the performance yielded by a particle. After (2) and (3) are executed, the personal best position *y* of each particle is updated as follows:

$$y_i(t+1) = \begin{cases} y_i(t+1), & \text{if } f(x_i(t+1)) \geq f(y_i(t)), \\ x_i(t+1), & \text{if } f(x_i(t+1)) < f(y_i(t)), \end{cases} \quad (4)$$

and the global best position is found by:

$$\hat{y}(t+1) = \arg\min_{y_i} f(y_i(t+1)), \quad 1 \leq i \leq s. \quad (5)$$

The CPSO [5, 6] is one of the most significant improvements to the original PSO. Van den Bergh presented a family of CPSOs, including CPSO-S, CPSO-S_K, CPSO-H, CPSO-H_K. Algorithm CPSO-H_K is the hybrid from PSO and CPSO-S_K and it is proposed to address the issue of "pseudominima." A discussion of pseudominima is outside of the scope of this article. The objective of this article is to propose a self-organized technique to assist the CPSO-S_K in finding how the components on a context vector be related.

The concept of CPSO-S is that instead of trying to find an optimal *n*-dimensional vector, the vector is split into *n* parts so that each of *n* swarms optimizes a 1-D vector. The CPSO-S_K is a family of CPSO-S, where a vector is split into *K* parts rather than *n*, where $K \leq n$. *K* also represents the number of swarms. Each of the *K* swarms acts as a PSO optimizer (2)-(5). The main difference between the PSO and the CPSO is that the fitness of a single particle of the CPSO has to be evaluated through global best particles of the other swarms. Let P_j denote the *j*-th swarm and $P_j \bullet x_i$ represents the *i*-th particle in the swarm *j*. The fitness of $P_j.x_i$ is defined as:

$$f(P_j \bullet x_i) = f(P_1 \bullet \hat{y}, \ldots, P_{j-1} \bullet \hat{y}, P_j.x_i, \ldots, P_K \bullet \hat{y}). \quad (6)$$

The CPSO applies cooperative behavior to improve the PSO on find the global optimum in a high-dimensional space. This is achieved by employing multiple swarms to explore the subspaces of the search space separately to reduce the curse of dimensionality. However, there is no absolute criterion that the CPSO is superior than the PSO since independent changes made by different swarms on correlated variables will deteriorate its performance . In addition, in one generation of a *n*-dim CPSO-S operation, the computational cost is *n* times larger than that of a PSO operation.

III. METHODLOGY

This paper proposes an approach to help the CPSO self-organize the swarms composed of non-separable variables. Consider a particular optimization task illustrated in Fig. 1, from which we can see a 2-dim function with a bar-shaped local optimal region and a global optimum lies in it. The task is to find its global optimum by particle swarm optimizer. At first, particles are uniformly distributed in the search space. At this moment, we expect particles to be divided into two swarms, performing separate 1-dim PSO operation on each dimension to speed up the process of particles gathering around the optimal region.

If by any chance particles gather around the optimal region as we expected, as shown in Fig. 2. At this point of time, we prefer particles performing 2-dim PSO operation on the whole search space to reduce the computational cost, which, in this case, represents the number of function evaluations.

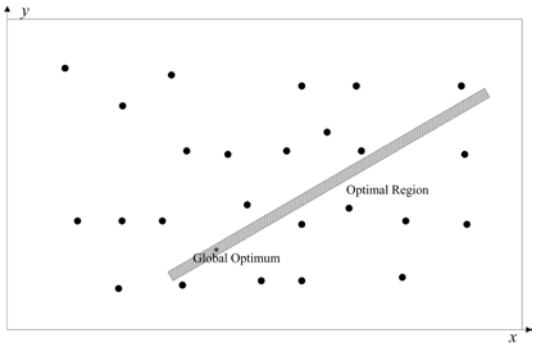


Figure 1. Case with particles uniformly distributed in the search space to find the global optimum lies in a bar-shaped local optimal region.

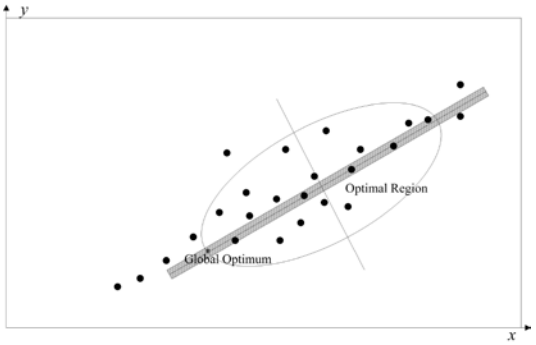


Figure 2. Case with particles gather around the bar-shaped optimal region to find the global optimum.

In order to implement the idea illustrated above, we have to determine the timing of switching between the PSO and the CPSO operation when dealing with a task. In this paper, we think this can be done by determining the separability between variables, and placing non-separable into the same swarm at each generation. If at certain moment, all variables are determined as non-separable, then the PSO operation is taken; otherwise, the CPSO operation is taken.

The separability between variables is found by estimating the covariance matrix of the distribution of particles. The method we adopt is called the covariance matrix adaptation proposed in [8, 9]. In the standard CMA-ES, a population of new search points is generated by sampling a multivariate normal distribution *N* with mean $m \in \mathbb{R}^n$ and covariance matrix $\mathbf{C} \in \mathbb{R}^{n \times n}$. The equation of sampling new search points, for each generation number $g = 0, 1, 2, \ldots$, reads

$$x_i^{(g+1)} \sim m^{(g)} + \sigma^{(g)} N(0, \mathbf{C}^{(g)}) \quad \text{for } i = 1, \ldots, \lambda, \quad (7)$$

where ~ denotes the same distribution on the left and right hand side; $\sigma^{(g)}$ denotes the overall standard deviation, step-size, at generation *g* and λ is the sample size. The new mean $m^{(g+1)}$ of the search distribution is a weighted average of the μ selected points from λ samples $x_1^{(g+1)}, x_2^{(g+1)}, \ldots, x_\lambda^{(g+1)}$:

$$m^{(g+1)} = \sum_{i=1}^{\mu} w_i x_{i:\lambda}^{(g+1)}, \quad (8)$$

with

$$\sum_{i=1}^{\mu} w_i = 1, \quad w_1 \geq w_2 \geq \mathsf{L} \geq w_1 > 0 ,$$

(9)

where w_i are positive weights, and $x_{i:\lambda}^{(g+1)}$ denotes the i-th rank individual out of λ samples from (8). The index i:λ denotes the i-th rank individual and

$$f(x_{1:\lambda}^{(g+1)}) \leq f(x_{2:\lambda}^{(g+1)}) \leq \mathsf{L} \leq f(x_{\lambda:\lambda}^{(g+1)}), \qquad (10)$$

where $f()$ is the objective function to be minimized. The adaption of new covariance matrix $\mathbf{C}^{(g+1)}$ is formed by a combination of rank-μ and rank-one update [10]

$$\mathbf{C}^{(g+1)} = (1-c_{\text{cov}})\mathbf{C}^{(g)} + \underbrace{\frac{c_{\text{cov}}}{\mu_{\text{cov}}} p_c^{(g+1)} \left(p_c^{(g+1)}\right)^T}_{\text{rank-one update}}$$

$$+ c_{\text{cov}}\left(1-\frac{1}{\mu_{\text{cov}}}\right) \times \underbrace{\sum_{i=1}^{\mu} w_i y_{i:\lambda}^{(g+1)} \left(y_{i:\lambda}^{(g+1)}\right)^T}_{\text{rank-}\mu\text{ update}} ,$$

(11)

where $\mu_{\text{cov}} \geq 1$ is the weighting between rank-μ update and rank-one update; $c_{\text{cov}} \in [0,1]$ is the learning rate for the covariance matrix update, and

$$y_{i:\lambda}^{(g+1)} = (x_{i:\lambda}^{(g+1)} - m^{(g)}) / \sigma^{(g)} ,$$

(12)

is a modified formula used to compute the estimated covariance matrix for the selected samples. The evolution path $p_c^{(g+1)}$ for rank-one update is described as follows:

$$p_c^{(g+1)} = (1-c_c) p_c^{(g)} + \sqrt{c_c(2-c_c)\mu_{\text{eff}}} \frac{m^{(g+1)} - m^{(g)}}{\sigma^{(g)}} ,$$

(13)

where $c_c \leq 1$ denotes the backward time horizon and

$$\mu_{\text{eff}} = \left(\sum_{i=1}^{\mu} w_i^2 \right)^{-1} , \qquad (14)$$

denotes the variance effective selection mass. The new step-size $\sigma^{(g+1)}$ is updated according to

$$\sigma^{(g+1)} = \sigma^{(g)} \exp\left(\frac{c_\sigma}{d_\sigma} \left(\frac{\left\| p_\sigma^{(g+1)} \right\|}{E\left\| N(0,\mathbf{I}) \right\|} - 1 \right) \right),$$

(15)

with

$$p_\sigma^{(g+1)} = (1-c_\sigma) p_\sigma^{(g)} + \sqrt{c_\sigma(2-c_\sigma)\mu_{\text{eff}}} \mathbf{C}^{(g)^{-1/2}} \frac{m^{(g+1)} - m^{(g)}}{\sigma^{(g)}} ,$$

(16)

where c_σ is the backward time horizon of evolution path, similar to c_c; d_σ is a damping parameter and $p_\sigma^{(g+1)}$ is the conjugate evolution path for step-size $\sigma^{(g+1)}$. The expectation of the Euclidean norm of a $N(0, \mathbf{I})$ reads

$$E\left\| N(0,\mathbf{I}) \right\| = \sqrt{2}\Gamma\left(\frac{n+1}{2}\right) / \Gamma\left(\frac{n}{2}\right) \approx \sqrt{n} + O(1/n) ,$$

(17)

where $O(\cdot)$ represents high-order terms.

Consider the estimated covariance matrix has the form shown as follows,

$$C = \begin{bmatrix} c_1^2 & c_{12} & \mathsf{L} & c_{1n} \\ \mathsf{M} & c_2^2 & \mathsf{L} & \mathsf{M} \\ \mathsf{M} & \mathsf{M} & & \mathsf{M} \\ c_{1n} & c_{2n} & \mathsf{L} & c_n^2 \end{bmatrix}, \qquad (18$$

)

where n is the number of dimensions, c_{jk} represents the weighted covariance between variables j and k. The separability between dimensions can be obtained from correlation coefficient matrix with its element defined as follows:

$$\rho_{jk} = c_{jk} / c_j c_k ,$$

(19)

We define a parameter ρ_{thres} to determine whether dimension j and k are viewed as separable. If $\rho_{jk} < \rho_{\text{thres}}$ then we say variable j and k are separable. Conventionally, if $|\rho|>0.8$, it implies that there exists a very strong linear relationship between these two variables; $0.8>|\rho|>0.6$ implies strong relationship, and $0.6>|\rho|>0.4$ implies moderate relationship. So, in this paper, we avoid setting ρ_{thres} less than 0.6. The block diagram of the proposed method can be found in Fig. 3.

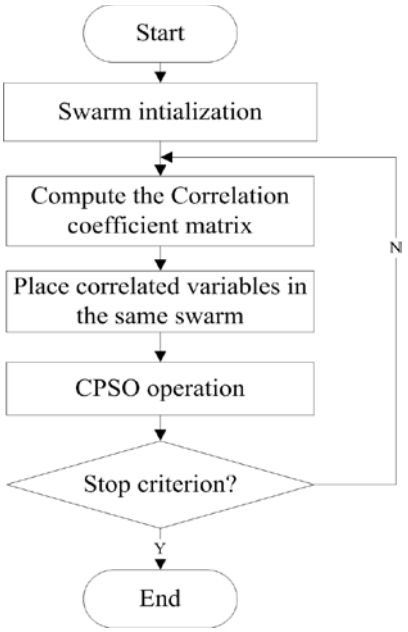


Figure 3. Block diagram of SD-CPSO.

IV. EXPERIMENT RESULTS

In order to compare the performance between different algorithms, a fair time measure must be selected. Here we use the number of function evaluations as a time measure following [5]. The performance of the proposed SD-CPSO is verified by real-parameter minimization tasks, which contains totally nine test functions. By their nature they can be divided into two parts: unimodal and multi-modal functions.

The first two functions are unimodal, followed by seven multimodal functions with three of them have simple global structures (single-funnel functions) and another four have complex global structures (multi-funnel functions). The difference between single- and multi-funnel functions can be illustrated by the following two figures, where Figure 4 shows a visualization of a 2-D Rastrigin's function, from which we can see that in spite of the large amount of local minima, there exists a trend to the global minimum. Figure 5 shows a visualization of a 2-D double Rastrigin's function, from which we can see that there are two funnel-type global trends and a large amount of noisy local minima.

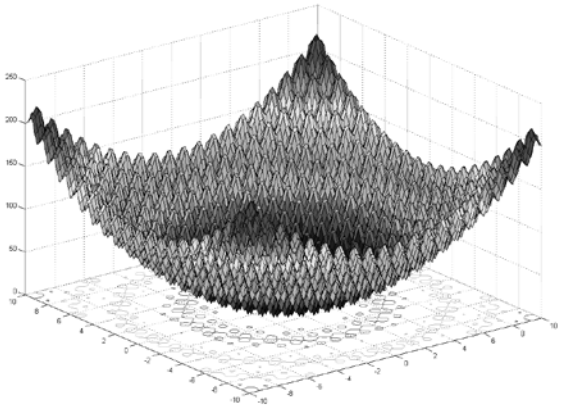


Figure 4. Visualization of a single-funnel, 2-D Rastrigin's function.

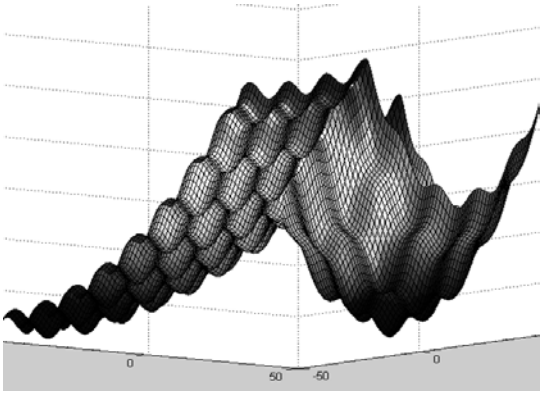


Figure 5. Visualization of a multi-funnel-funnel, 2-D double Rastrigin's function.

The types and names of functions are described in Table I. A detailed definition of test functions can be seen in [11, 12].

All functions are of 50 dimensions and have been adjusted to zero optimal solution respectively. To make sure that there was sufficient correlation between the variables, making it even harder for optimization, all the functions were further tested under 45 degree coordinate rotation.

In the following of this chapter, we will describe the configurations of the algorithms that we use to compare the performance with the proposed SD-CPSO in section 3.A. Experiment result and the discussion will be shown in section 3.B.

A. Algorithms Configuration

The three algorithms for comparison are listed as follows:

- PSO: the origin algorithm.
- CPSO-S: algorithm that splits swarm into each dimension.
- SD-CPSO: the proposed separability detection cooperative particle swarm optimization.

For each algorithm, experiments are executed for 50 times. Denote n the dimension of the optimization task and s the number of particles in one swarm. Parameters of the three algorithms are listed in Table II.

TABLE I. TYPE AND NAME OF THE TEST FUNCTION.

Unimodal Functions
F_1: Sphere Function
F_2: High Conditioned Ellipsoidal Function
Multimodal Functions
F_3: Rosenbrock Function
F_4: Rastrigin Function
F_5: Griewank Function
Multi-Funnel Functions
F_6: Schwefel Function
F_7: Double-Rastrigin Function
F_8: Weierstrass Function
F_9: Michalewicz Function

TABLE II. MS-CMA-ES AND CMA-ES PARAMETERS.

Parameters of Selection operator
$\lambda = 4 + \lfloor 3\ln n \rfloor$
$w_i = \frac{w_i'}{\sum_{j=1}^{\lambda} w_j'}, w_i' = \begin{cases} \ln(\frac{\lambda}{2}+0.5) - \ln i \text{ , for } i=1,\ldots \lfloor \frac{\lambda}{2} \rfloor, \\ 0 \text{ , otherwise,} \end{cases}$
Parameters of Covariance adaptation:
$c_{\text{cov}}=0.7$
$\mu_{\text{cov}}=10$
$c_c = \frac{4+\mu_{\text{eff}}/n}{n+4+2\mu_{\text{eff}}/n}$,
$c_\sigma = \frac{\mu_{\text{eff}}+2}{n+\mu_{\text{eff}}+5}$
$d_\sigma = 1+2\max\left(0, \sqrt{\frac{\mu_{\text{eff}}-1}{n+1}}-1\right)+c_\sigma$
Parameters of PSO operation:
$c_1=c_2=1.49$
$\rho_{\text{thres}}=0.8$
$s=50$

B. Experiment Result

This section presents optimization results. The number of maximum fitness calculation times, initial search range, initial search position and minimum fitness threshold are detailed in Table III. All particles are evenly distributed in the initial search range.

TABLE III. PARAMETERS OF THE EXPERIMENT.

	maximum fitness calculation times	Initial search range	Minimum fitness threshold
f_1	10000	$\mathbf{x} \in [0,100]^d$	1e-6
f_2	10000	$\mathbf{x} \in [0,100]^d$	1e-6
f_3	10000	$\mathbf{x} \in [0,100]^d$	1e-2
f_4	3000	$\mathbf{x} \in [0, 5]^d$	1e-2
f_5	8000	$\mathbf{x} \in [0,600]^d$	1e-2
f_6	4000	$\mathbf{x} \in [0,3]^d$	1e-2
f_7	2000	$\mathbf{x} \in [-20,20]^d$	1e-2
f_8	4000	$\mathbf{x} \in [0,0.5]^d$	1e-2
f_9	5000	$\mathbf{x} \in [0,5]^d$	1e-2

The experimental data is obtained by executing each 50 dimensional test functions until the stopping criterion is met. The procedure was repeated 50 times to compute the average fitness value. In the paper, instead of the actual numeric fitness value, the rank of the minimum average fitness value is defined as the standard of comparison. The reason is that we want to exclude the impact of the different degree of scale on the raw numeric difference between each test function. For example, some functions have very large fitness gap between the best and the second best local minimum, some of them don't even have local minima. Therefore, the numeric difference may not be a good performing index for evaluating algorithms. The experiment result is shown in Table IV as follows.

TABLE IV. AVERAGE FITNESS VALUE.

	CPSO-S	SD-CPSO	PSO
f_1	6.361e-99(1)*	**2.634e-062(3)**	9.653-76(2)
f_2	4.481e-84(1)*	**3.464e-033(3)**	2.876e-75(2)
f_3	18.8764 (3)	**0.8872 (1)***	1.4356(2)
f_4	11.871(1)	**17.721(2)**	26.65(1)*
f_5	9.6198(3)	**0.6893(1)***	6.3769(2)
f_6	469.9(3)	**288.3(2)**	87.36(1)*
f_7	12.57(2)	**7.659(1)***	95.03(3)
f_8	1.2287(2)	**0.6643(1)***	1.254(3)
f_9	5.75(3)	**7.864e-008(1) ***	4.08(2)

The results to be discussed are divided into three parts in accordance with the function types:

1) Unimodal Function:

Under the sphere function f_1, CPSO-S has the best performance, owing to its property of rapid convergence. As to ellipsoid function f_2, at first, PSO is better than the other two algorithms. As shown from the experiment result, all three algorithms are capable of solving unimodal optimization task, and no improvement of performance can be found by applying our method.

2) Multimodal Function:

The SD-CPSO is better than other algorithms under the f_3 and f_5 test functions except for f_4, the Rastrigin's function. We think it might due to the fact that Rastrigin's function is nearly the same after rotation, which makes our effort trying to find a special trend to the global optimum irrelevant. However, the superiority of the proposed SD-CPSO in finding global optima of multimodal functions can be seen in substance.

3) Multi-Funnel Function:

From Table IV we can see that in coping with multi-funnel function optimization tasks, the superiority of the proposed SD-CPSO is obvious. In general, the optimization of multi-funnel function is difficult as we can see especially from the optimization result of the f_6 function. Despite the proposed SD-CPSO has better performance on the optimization tasks of f_7 and f_8 function, the improvement is not very obvious. However, in the optimization of f_9, the Michalewicz's function, the improvement is remarkable. As a result, we will illustrate the optimization results of applying Michalewicz's function in both its unrotated and rotated form in Fig. 7.

Fig. 7(a) represents the result of applying unrotated Michalewicz's function. Michalewicz's function introduces many valleys into the plain, and the function values for points in the space outside the narrow valleys give very little information about the location of the global optimum. Thus, the swarms need to follow through these valleys to find minimums. In its rotated version, these narrow valleys are too correlated to follow through from the perspective of the CPSO. In Fig. 7(b), the SD-CPSO in evidence overcomes the drawback.

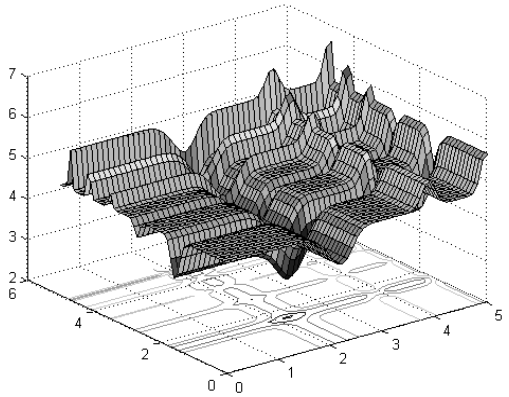
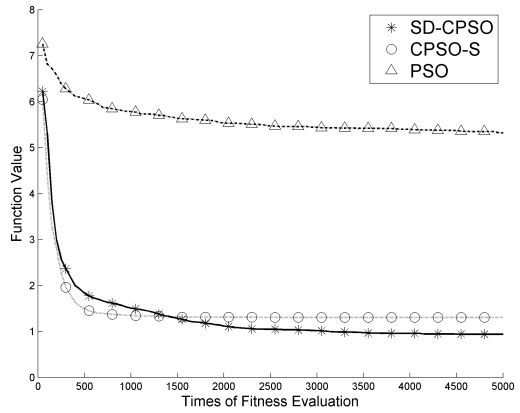
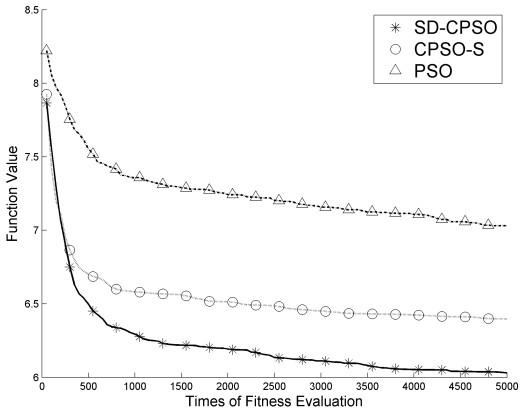


Figure 6. Visualization of a 2-D Michalewicz's function.



(a)



(b)

Figure 7. Experiment results of applying Michalewicz's function in its (a) unrotated form, (b) rotated form.

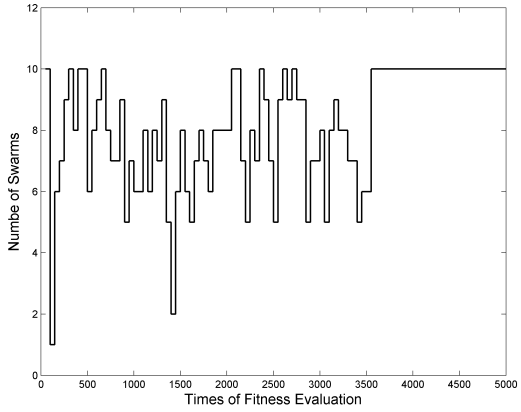


Figure 8. Results of the number of swarms of applying rotated Michalewicz's function.

Fig. 8, on the other hand, illustrates the ability of SD-CPSO self-organizes the decomposition of dimensions. We place the detected non-separable variables to the same swarm in the CPSO operation to alleviate the detrimental effect we encountered when placing independent variables into separate swarms. When particles waver in the valley, the number of swarm decreased for the sake of correlated dimension has being coupled, and when swarms step into the local minimum region, the number of swarm increased to adapt these uncorrelated sphere-liked region.

V. CONCLUSION

In this paper, we propose a self-organization approach to the CPSO. This approach determines the suitable swarm structure of the CPSO by estimating the correlations between variables. Experiments show reasonable performance. The combination of dimensions forming a swarm is detected by covariance matrix adaptation. Future research should be done to investigate the pseudominima caused by the split of swarm.

ACKNOWLEDGMENT

This work is supported in part by the National Science Council, Taiwan. R. O. C. under Grants NSC 99-2221-E-009-148.

REFERENCES

[1] J. Kennedy, “The particle swarm: social adaptation of knowledge,” in *Proc. of 1997 IEEE International Conference on Evolutionary Computation* (ICEC '97), pp. 303-308, 1997.

[2] M. Clerc and J. Kennedy, “The particle swarm - Explosion, stability, and convergence in a multidimensional complex space,” *IEEE Transactions on Evolutionary Computation*, vol. 6, pp. 58-73, 2002.

[3] M. A. Potter and K. A. De Jong, “A cooperative coevolutiouary approach to function optimization,” *in Parallel Problem Solving from Nature - Ppsn Iii - International Conference on Evolutionary Computation, Proceedings*, vol. 866, Y. Davidor, et al., Eds., ed Berlin: Springer-Verlag Berlin, pp. 249-257, 1994.

[4] N. Hansen, et al., “Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES),” *Evolutionary Computation*, vol. 11, pp. 1-18, 2003.

[5] F. van den Bergh and A. P. Engelbrecht, “A cooperative approach to particle swarm optimization,” *IEEE Transactions on Evolutionary Computation*, vol. 8, pp. 225-239, 2004.

[6] Yi-Chang Cheng, Sheng-Fuu Lin, and Chi-Yao Hsu “Q-Value Based Particle Swarm Optimization for Reinforcement Neuro-Fuzzy System Design,” *International Journal on Computer Science and Engineering*, vol. 3, no. 10, pp. 3477-3489, 2011.

[7] C. K. Goh, et al., “A competitive and cooperative co-evolutionary approach to multi-objective particle swarm optimization algorithm design,” *European Journal of Operational Research*, vol. 202, pp. 42-54, 2010.

[8] N. Hansen and A. Ostermeier, “Completely derandomized self-adaptation in evolution strategies,” *Evolutionary Computation*, vol. 9, pp. 159-195, 2001.

[9] N. Hansen and S. Kern, “Evaluating the CMA evolution strategy on multimodal test functions,” *Parallel Problem Solving from Nature - Ppsn* Viii, vol. 3242, pp. 282-291, 2004.

[10] N. Hansen, “The CMA Evolution Strategy: A Tutorial.” 2008. (from http://www.bionik.tu-berlin.de/user/niko/cmatutorial.pdf)

[11] N. Hansen, A. Auger, S. Finck, and R. Ros, “Real-parameter block-box optimization benchmarking 2010: experimental setup.” INRIA Research Report RR-7215.

[12] P. N. Suganthan, N. Hansen, J. J. Liang, K. Deb, Y. P. Chen, A. Auger, and S. Tiwari. “Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization.” Nanyang Technological University, Singapore and KanGAL Report Number 2005005.

AUTHORS PROFILE

Sheng-Fuu Lin was born in Tainan, the Republic of China, in 1954. He received the B.S and M.S. degree in mathematics from National Normal University in 1976 and 1979, respectively, the M.S. degree in computer

science from the University of Maryland in 1985, and the Ph.D. degree in electrical engineering from the University of Illinois, Champaign, in 1988. Since 1988, he has been on the faculty of the Department of Electrical Engineering at National Chiao Tung University, Hsinchu, Taiwan, where he is currently a professor. His research interests include fuzzy systems, genetic algorithms, neural networks automatic target recognition, scheduling, image processing, and image recognition.

Yi-Chang Cheng received the B.S. degree in engineering science from the National Cheng Kung University, Taiwan, R.O.C., in 2005. He is currently pursuing the Ph. D. degree at the department of electrical engineering from the National Chiao Tung University, Taiwan, R.O.C. His research interests include neural networks, fuzzy systems, evolutional algorithms and genetic algorithms.

Jyun-Wei Chang received the B.S. and M.S. degree in electronic engineering from National Kaohsiung University of Applied Sciences, Taiwan, R.O.C. in 2005 and 2007, respectively. He is currently pursuing the Ph.D. degree at the department of electrical engineering from the National Chiao Tung University, Taiwan, R.O.C. His research interests include neural networks, fuzzy systems, and evolutional algorithms.

Pei-Chia Hung received the B.S. degree in engineering science from the National Chiao Tung University, Taiwan, R.O.C., in 2004. He is currently pursuing the Ph. D. degree at the department of electrical engineering from the National Chiao Tung University, Taiwan, R.O.C. His research interests include image processing and image compression.