Dynamic Data Clustering Using Stochastic Approximation Driven Multi-Dimensional Particle Swarm Optimization

Serkan Kiranyaz[1], Turker Ince[2], and Moncef Gabbouj[1,*]

[1] Tampere University of Technology, Tampere, Finland
{serkan.kiranyaz,moncef.gabbouj}@tut.fi
[2] Izmir University of Economics, Izmir, Turkey
turker.ince@ieu.edu.tr

Abstract. With an ever-growing attention Particle Swarm Optimization (PSO) has found many application areas for many challenging optimization problems. It is, however, a known fact that PSO has a severe drawback in the update of its global best (*gbest*) particle, which has a crucial role of guiding the rest of the swarm. In this paper, we propose two efficient solutions to remedy this problem using a stochastic approximation (SA) technique. For this purpose we use simultaneous perturbation stochastic approximation (SPSA), which is applied only to the *gbest* (not to the entire swarm) for a low-cost solution. Since the problem of poor *gbest* update persists in the recently proposed extension of PSO, called multi-dimensional PSO (MD-PSO), two distinct SA approaches are then integrated into MD-PSO and tested over a set of unsupervised data clustering applications. Experimental results show that the proposed approaches significantly improved the quality of the MD-PSO clustering as measured by a validity index function. Furthermore, the proposed approaches are generic as they can be used with other PSO variants and applicable to a wide range of problems.

Keywords: Particle Swarm Optimization, stochastic approximation, multi-dimensional search, gradient descent, dynamic data clustering.

1 Introduction

The particle swarm optimization (PSO) [4,10,11] exhibits certain similarities with the other evolutionary algorithms (EAs) [2]. The common point of all is that EAs are in population based nature and they can avoid being trapped in a local optimum. Thus they can find the optimum solutions; however, this is never guaranteed. In a PSO process, a swarm of particles (or agents), each of which represent a potential solution to an optimization problem, navigate through the search (or solution) space. One major drawback of PSO is the direct link of the information flow between particles and the global-best particle, *gbest*, which primarily "guides" the rest of the swarm and thus resulting in the creation of similar particles with some loss of diversity. Hence this phenomenon increases the probability of being trapped in local optima [7] and it is the main cause of the premature convergence problem especially when the search

* This work was supported by the Academy of Finland, project No. 213462 (Finnish Centre of Excellence Program (2006 - 2011).

C. Di Chio et al. (Eds.): EvoApplications 2010, Part I, LNCS 6024, pp. 336–343, 2010.

space is of high dimensions [10] and the problem to be optimized is multi-modal [7]. This makes it clear that at any iteration of a PSO process, *gbest* is the most important particle; however, it has the poorest update equation, i.e. when a particle becomes *gbest*, it resides on its personal best position (*pbest*) and thus both social and cognitive components are nullified in the velocity update equation. Although it guides the swarm during the following iterations, ironically it lacks the necessary guidance to do so effectively. In that, if gbest is (likely to get) trapped in a local optimum, so the rest of the swarm due to the aforementioned direct link of information flow. This deficiency has been raised in a recent work [5] where an artificial GB particle, the *aGB*, is created at each iteration as an alternative to *gbest*. However, the underlying mechanism for creating the *aGB* particle, the so-called fractional GB formation (FGBF), is not generic, rather problem dependent.

For the problem of finding a root θ^* (either minimum or maximum point) of the gradient equation: $g(\theta) \equiv \frac{\partial L(\theta)}{\partial \theta} = 0$ for some differentiable function $L : R^p \to R^1$, when g is present and L is a uni-modal function, there are powerful deterministic methods for finding the global θ^* such as traditional steepest descent and Newton-Raphson methods. However, in many real problems g cannot be observed directly and/or L is in multi-modal nature, which presents many deceiving local optima. This brought the era of the stochastic optimization algorithms, which can estimate the gradient and may avoid being trapped into a local optimum due to their stochastic nature. One of the most popular stochastic optimization techniques is stochastic approximation (SA), in particular the form that is called "gradient free" SA. Among many SA variants the one and somewhat different SA application is called simultaneous perturbation SA (SPSA) proposed by Spall in [8].

In this paper we shall propose two approaches, one of which drives *gbest* efficiently or simply put, *guides* it with respect to the function (or error surface). The idea behind this is quite simple: since the velocity update equation of *gbest* is quite poor, SPSA as a simple yet powerful search technique is used to *drive* it instead. The second approach has a similar motivation with the FGBF proposed in [5], i.e., an artificial Global Best (*aGB*) particle is created by SPSA this time, which is applied over the personal best (*pbest*) position of the *gbest* particle. The *aGB* particle will then guide the swarm instead of *gbest* if and only if it achieves a better fitness score than the (personal best position of) *gbest*. Note that both approaches *only* deal with the *gbest* particle and hence the internal PSO process remains as is. They are then applied to the multi-dimensional extension of PSO, the MD-PSO technique proposed in [5], which can find the optimum dimension of the solution space. SA-driven (SAD) MD-PSO is then tested and evaluated against the standalone MD-PSO application over several data clustering problems.

2 Proposed Technique: SAD MD-PSO

2.1 SPSA Overview

The goal of the deterministic optimization methods is to minimize a loss function $L : R^p \to R^1$, which is a differentiable function of θ and the minimum (or maximum) point θ^* corresponds to zero-gradient point, i.e.

$$g(\theta) \equiv \left.\frac{\partial L(\theta)}{\partial \theta}\right|_{\theta=\theta^*} = 0 \tag{1}$$

As mentioned earlier, in cases where more than one point satisfies this equation (e.g. a multi-modal problem), then such algorithms may only converge to a local minimum. Moreover, in many practical problems, g is not readily available. This makes the SA algorithms quite popular and they are in the general SA form:

$$\hat{\theta}_{k+1} = \hat{\theta}_k - a_k \hat{g}_k(\hat{\theta}_k) \tag{2}$$

where $\hat{g}_k(\hat{\theta}_k)$ is the estimate of the gradient $g(\theta)$ at iteration k and a_k is a scalar gain sequence satisfying certain conditions [8]. There are two common SA methods: finite difference SA (FDSA) and simultaneous perturbation SA (SPSA). FDSA adopts the traditional Kiefer-Wolfowitz approach to approximate gradient vectors as a vector of p partial derivatives where p is the dimension of the loss function. On the other hand, SPSA has all elements of $\hat{\theta}_k$ perturbed simultaneously using only two measurements of the loss function as,

$$\hat{g}_k(\hat{\theta}_k) = \frac{L(\hat{\theta}_k + c_k\Delta_k) - L(\hat{\theta}_k - c_k\Delta_k)}{2c_k} \begin{bmatrix} \Delta_{k1}^{-1} \\ \Delta_{k2}^{-1} \\ . \\ . \\ . \\ \Delta_{kp}^{-1} \end{bmatrix} \tag{3}$$

where the p-dimensional random variable Δ_k is usually chosen as Bernoulli ± 1 distribution and c_k is a scalar gain sequence satisfying certain conditions [8]. Spall [8] presents conditions for convergence of SPSA (i.e. $\hat{\theta}_k \to \theta^*$) and show that under certain conditions both SPSA and FDSA have the same convergence ability –yet SPSA needs only 2 measurements whereas FDSA needs 2p. This makes SPSA our natural choice for driving gbest in both approaches. SPSA has 5 parameters. Spall [9] recommended to use values for A (the stability constant), α, and γ as 60, 0.602 and 0.101, respectively. However, he also concluded that "*the choice of both gain sequences is critical to the performance of the SPSA as with all stochastic optimization algorithms and the choice of their respective algorithm coefficients*". This especially makes the choice of gain parameters *a* and *c* critical for a particular problem. i.e. Maryak and Chin [6] varied them with respect to the problem whilst keeping the other three (A, α, and γ) as recommended.

2.2 SAD MD-PSO

As mentioned earlier, in this work two distinct SAD MD-PSO approaches are proposed, each of which is only applied in each dimension on the *gbest* of the positional PSO whilst keeping the internal PSO processes (both positional and dimensional)

intact. Since both SPSA and positional PSO are iterative processes, in both approaches SPSA can thus easily be integrated into PSO by using the same iteration count (i.e. $t \equiv k$). The following sub-sections will detail each approach.

A1) First SAD MD-PSO approach: gbest update by SPSA

In this approach, at each iteration *gbest* particle is updated using SPSA. This requires the adaptation of the SPSA elements (parameters and variables) and integration of the internal SPSA part (within the loop) appropriately into the PSO pseudo-code. Due to space limitations, we have to skip the pseudo code details of this approach. Note that such a "plug-in" approach will not change the internal PSO structure and only affects the *gbest* particle's movement. It only costs two extra function evaluations and hence at each iteration the total number of evaluations is increased from *S* to *S*+2 (recall that *S* is the swarm size).

Since the fitness of each particle's current position is computed within the PSO process, it is possible to further diminish this cost to only *one* extra fitness evaluation per iteration. Let $\hat{\theta}_k + c_k\Delta_k = xx_a^d(t)$ and thus $L(\hat{\theta}_k + c_k\Delta_k)$ is known *a priori*. Then naturally, $\hat{\theta}_k - c_k\Delta_k = xx_a^d(t) - 2c_k\Delta_k$, which is the only (new) location where the (extra) fitness evaluation ($L(\hat{\theta}_k - c_k\Delta_k)$) has to be computed. Once the gradient ($\hat{g}_k(\hat{\theta}_k)$) is estimated, then the next (updated) location of the *gbest* will be: $xx_a^d(t+1) = \hat{\theta}_{k+1}$. Note that the difference of this "low-cost" SPSA update is that $xx_a^d(t+1)$ is updated (estimated) *not* from $xx_a^d(t)$, instead from $xx_a^d(t) - c_k\Delta_k$. Note that in a MD-PSO process there is a distinct *gbest* particle, *gbest(d)*. So SPSA is applied individually over the position of each *gbest(d)* if it (re-) visits the dimension d, (i.e. $d = xd_{gbest}(t)$). Therefore, there can be $2(D_{\max} - D_{\min})$ number of function evaluations, indicating a significant cost especially if a wide dimensional range is used. However, this is a theoretical limit, which can only happen if $gbest(i) \neq gbest(j)$ *for* $\forall i, j \in [D_{\min}, D_{\max}], i \neq j$ and all particles altogether visit the particular dimensions in which they are gbest (i.e. $xd_{gbest(d)}(t) = d, \forall t \in [1, iterNo]$). Especially in a wide dimensional range, note that this is highly unlikely due to the dimensional velocity, which makes particles move (jump) from one dimension to another at each iteration. It is straightforward to see that under the assumption of a uniform distribution for particles' movements over all dimensions within the dimensional range, SAD MD-PSO too, would have the same cost overhead as the SAD PSO. Experimental results indicate that the practical overhead cost is only slightly higher than this.

A2) Second SAD PSO approach: aGB formation by SPSA

The second approach replaces the FGBF operation proposed in [5] with the SPSA to create an *aGB* particle. SPSA is basically applied over the *pbest* position of the *gbest* particle. The *aGB* particle will then guide the swarm instead of *gbest* if and only if it achieves a better fitness score than the (personal best position of) *gbest*. SAD MD-PSO pseudo-code as given in Table 1 can then be plugged into the MD-PSO.

Note that in this approach, there are three extra fitness evaluations (as opposed to two in the first one) at each iteration. Yet as in the first approach, it is possible to further diminish the cost by *one* (from three to two fitness evaluations per iteration). In order to create an *aGB* particle for all dimensions in the given range (i.e. $\forall d \in [D_{\min}, D_{\max}]$) SPSA is applied individually over the personal best position of each *gbest(d)* particle and furthermore, the aforementioned competitive selection ensures that $xy^d_{aGB}(t), \forall d \in [D_{\min}, D_{\max}]$ is set to the best of the $xx^d_{aGB}(t+1)$ and $xy^d_{aGB}(t)$. As a result, the SPSA creates one *aGB* particle providing (potential) GB solutions ($xy^d_{aGB}(t+1), \forall d \in [D_{\min}, D_{\max}]$) for all dimensions in the given dimension range. The pseudo-code of the second approach as given in Table 1 can then be plugged in between steps 3.2 and 3.3 of the MD-PSO pseudo-code, given in [5].

Table 1. MD-PSO Plug-in for the second approach at dimension *d*

A2) *SAD MD-PSO Plug-in* (ξ **, *a, c, A,*** α , γ)

1. Create a new *aGB* particle, $\{xx^d_{aGB}(t+1),\ xy^d_{aGB}(t+1)\}$ *for* $\forall d \in [D_{\min}, D_{\max}]$
2. For $\forall d \in [D_{\min}, D_{\max}]$ do:
   - 2.1. *Let k=t,* $\hat{\theta}_k = x\hat{y}^d(t)$ *and L=f*
   - 2.2. *Let* $a_k = a/(A+k)^\alpha$ *and* $c_k = c/k^\gamma$
   - 2.3. *Compute* $L(\hat{\theta}_k + c_k\Delta_k)$ *and* $L(\hat{\theta}_k - c_k\Delta_k)$
   - 2.4. *Compute* $\hat{g}_k(\hat{\theta}_k)$ *using Eq. (3)*
   - 2.5. *Compute* $xx^d_{aGB}(t+1) = \hat{\theta}_{k+1}$ *using Eq. (2)*
   - 2.6. If ($f(xx^d_{aGB}(t+1)) < f(xy^d_{aGB}(t))$) then $xy^d_{aGB}(t+1) = xx^d_{aGB}(t+1)$
   - 2.7. Else $xy^d_{aGB}(t+1) = xy^d_{aGB}(t)$
   - 2.8. If ($f(xy^d_{aGB}(t+1)) < f(xy^d_{gbest(d)}(t))$ then $xy^d_{gbest(d)}(t) = xy^d_{aGB}(t+1)$
3. End For.
4. Re-assign *dbest*: $dbest = \arg\min_{d \in [D_{\min}, D_{\max}]} (f(xy^d_{gbest(d)}(t))$

3 Experimental Results

In order to test each approach of the proposed SAD MD-PSO technique over clustering, we created 8 synthetic data spaces as shown in Figure 1 where white dots (pixels) represent data points. For illustration purposes each data space is formed in 2D; however, clusters are formed with different shapes, densities, sizes and inter-cluster distances to test the robustness of clustering application of the proposed approaches against such variations. Furthermore, recall that the number of clusters determines the (true) dimension of the solution space in a PSO application and hence it is also kept

varying among data spaces to test the converging accuracy to the true (solution space) dimension. As a result, significantly varying complexity levels are established among all data spaces to perform a general-purpose evaluation of each approach.

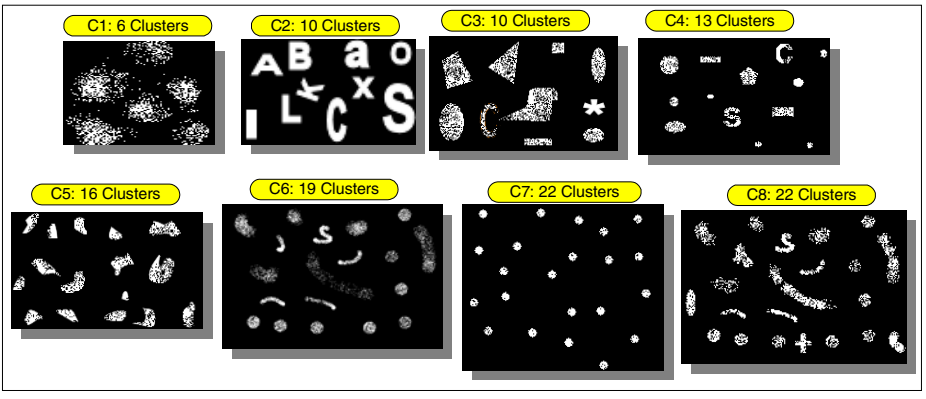

Fig. 1. 2D synthetic data spaces carrying different clustering schemes

The maximum number of iterations is set to 10000 and the use of cut-off error as a termination criterion is avoided since it is not feasible to set a unique ε_C value for all clustering schemes. For MD-PSO, we used the swarm size, *S*=200 and for both SAD MD-PSO approaches, a reduced number is used in order to ensure the same number of evaluation among all competing techniques. *w* is linearly decreased from 0.75 to 0.2 and we again used the recommended values for *A*, α, and γ as 60, 0.602 and 0.101, whereas *a* and c are set to 0.4 and 10, respectively. For each dataset, 20 clustering runs are performed.

For visual evaluation, Figure 2 presents the *worst* and the *best* clustering results of the two competing techniques, standalone vs. SAD MD-PSO, based on the highest (worst) and lowest (best) fitness scores achieved among the 20 runs. In the figure each cluster is represented in one of the three color codes (red, green and blue) for illustration purposes and each cluster centroid (each dimensional component of the *gbest* particle) is shown with a white '**+**'.The clustering results of the best performing SAD MD-PSO approach are shown whilst excluding C1 since results of all techniques are quite close for this data space due to its simplicity. Note first of all that the results of the (standalone) MD-PSO deteriorate severely as the complexity and/or the number of clusters increases. Particularly in the *worst* results, the critical errors such as under-clustering often occur with dislocated cluster centroids. For instance 4 out of 20 runs for C6 results in severe under-clustering with 3 clusters, similar to the one shown in the figure whereas this goes up to 10 out of 20 runs for C8. Although the clusters are the simplest in shape and in density for C7, due to the high solution space dimension (e.g. number of clusters = 22), even the *best* MD-PSO run is not immune to under-clustering errors. In some of the *worst* SAD MD-PSO runs too, one or few under-clusterings do occur; however, they are minority cases in general and definitely not as severe as in MD-PSO runs. It is quite evident from the *worst* and the *best* results in

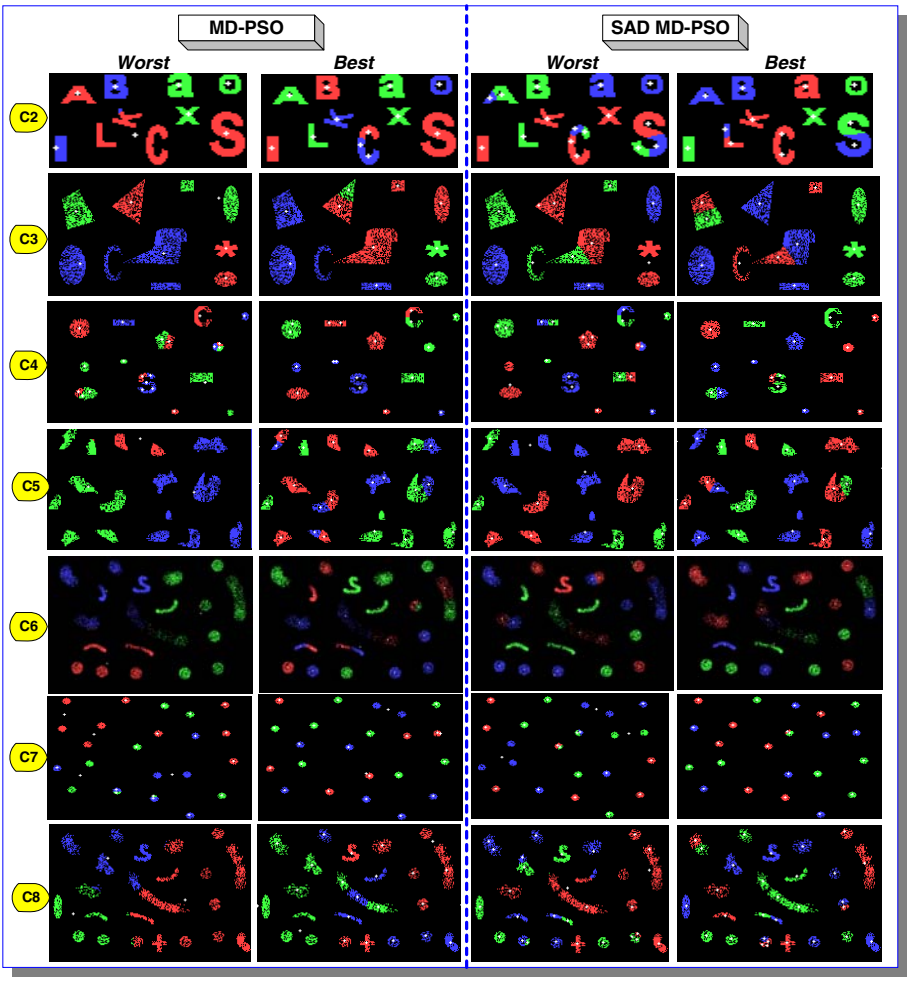

Fig. 2. The worst and the best clustering results using standalone (left) and SAD (right) MD-PSO

the figure that SAD MD-PSO achieves a significantly superior clustering quality and usually converges to a close vicinity of the global optimum solution.

4 Conclusions

In this paper, we draw the focus on a major drawback of the PSO algorithm: the poor *gbest* update. This can be a severe problem, which may cause pre-mature convergence to local optima since *gbest* as the common term in the update equation of all particles, is the primary *guide* of the swarm. SPSA is purposefully adapted to guide (or *drive)* the *gbest* particle (with simultaneous perturbation) towards the "*right*" direction with the gradient estimate of the underlying surface (or function) whilst avoiding local traps due to its stochastic nature. In that, the proposed SAD MD-PSO is not a new

MD-PSO variant or extension, rather a "guided MD-PSO" algorithm, which has an identical process with the MD-PSO as guidance is only provided to *gbest* particle –not the whole swarm.

SAD MD-PSO is then applied to the unsupervised clustering problem within which the (clustering) complexity can be thought of as synonymous to (function) modality and tested over 8 synthetic data spaces in 2D with ground truth clusters. The statistical results obtained from the clustering runs approve the superiority of SAD MD-PSO in terms of global convergence. We have applied a *fixed* set of SPSA parameters and observed that if the setting of the critical parameters, e.g. a and c is appropriate, then a significant performance gain can be achieved by SAD MD-PSO. If not, SAD MD-PSO still outperforms the standalone MD-PSO. This shows that SPSA, even without proper parameter setting still performs *better* than the PSO's native *gbest* update. Furthermore, we have noticed that the performance gap widens especially when the clustering complexity increases since the performance of the standalone MD-PSO operation, without any proper guidance, severely deteriorates. One observation worth mentioning is that the second approach on SAD MD-PSO has a significant overhead cost, which is anyway balanced by using reduced number of particles in the experiments; therefore, the low-cost mode should be used with a limited dimensional range for those applications with high computational complexity.

References

1. Abraham, A., Das, S., Roy, S.: Swarm Intelligence Algorithms for Data Clustering. In: Soft Computing for Knowledge Discovery and Data Mining book, Part IV, October 25, pp. 279–313 (2007)
2. Bäck, T., Schwefel, H.P.: An overview of evolutionary algorithm for parameter optimization. Evolutionary Computation 1, 1–23 (1993)
3. Halkidi, M., Batistakis, Y., Vazirgiannis, M.: On Cluster Validation Techniques. Journal of Intelligent Information Systems 17(2, 3), 107–145 (2001)
4. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proc. of IEEE Int. Conf. on Neural Networks, Perth, Australia, vol. 4, pp. 1942–1948 (1995)
5. Kiranyaz, S., Ince, T., Yildirim, A., Gabbouj, M.: Fractional Particle Swarm Optimization in Multi-Dimensional Search Space. IEEE Trans. on Systems, Man, and Cybernetics (2009) (in print)
6. Maryak, J.L., Chin, D.C.: Global random optimization by simultaneous perturbation stochastic approximation. In: Proc. of the 33rd Conf. on Winter Simulation, Washington, DC, December 9-12, pp. 307–312 (2001)
7. Riget, J., Vesterstrom, J.S.: A Diversity-Guided Particle Swarm Optimizer - The ARPSO, Technical report, Department of Computer Science, University of Aarhus (2002)
8. Spall, J.C.: Multivariate Stochastic Approximation Using a Simultaneous Perturbation Gradient Approximation. IEEE Transactions on Automatic Control 37, 332–341 (1992)
9. Spall, J.C.: Implementation of the simultaneous perturbation algorithm for stochastic optimization. IEEE Trans. on Aerospace and Electronic Systems 34, 817–823 (1998)
10. Van den Bergh, F.: An Analysis of Particle Swarm Optimizers, PhD thesis, Department of Computer Science, University of Pretoria, Pretoria, South Africa (2002)
11. Yan, Y., Osadciw, L.A.: Density estimation using a new dimension adaptive particle swarm optimization algorithm. Journal of Swarm Intelligence 3(4) (2009)