Particle Swarm CMA Evolution Strategy for the Optimization of Multi-Funnel Landscapes

Christian L. Müller, Benedikt Baumgartner, Ivo F. Sbalzarini

Abstract— **We extend the Evolution Strategy with Covariance Matrix Adaptation (CMA-ES) by collaborative concepts from Particle Swarm Optimization (PSO). The proposed Particle Swarm CMA-ES (PS-CMA-ES) algorithm is a hybrid real-parameter algorithm that combines the robust *local* search performance of CMA-ES with the *global* exploration power of PSO using multiple CMA-ES instances to explore different parts of the search space in parallel. Swarm intelligence is introduced by considering individual CMA-ES instances as lumped particles that communicate with each other. This includes non-local information in CMA-ES, which improves the search direction and the sampling distribution. We evaluate the performance of PS-CMA-ES on the IEEE CEC 2005 benchmark test suite. The new PS-CMA-ES algorithm shows superior performance on noisy problems and multi-funnel problems with non-convex underlying topology.**

I. Introduction

Optimization problems have to be differentiated into convex (unimodal) and non-convex (multimodal) problems, rather than linear and nonlinear problems [1]. Since their first formulation in the 1960s by Rechenberg and Schwefel [2], Evolution Strategies (ES) have been among the most successful gradient-less optimization paradigms for non-convex, real-valued functions. A particularly successful example is the Evolution Strategy with Covariance Matrix Adaptation (CMA-ES) [3], [4]. During exploration of the parameter space, CMA-ES attempts to learn a second-order model of the underlying objective function, similar to the approximation of the inverse Hessian matrix in Quasi-Newton methods [5]. Recent advances include a pure local restart CMA-ES (LR-CMA-ES) [6] and a CMA-ES with wide initial sample distribution and iteratively increasing population size (IPOP-CMA-ES) [7], achieving excellent performance on non-convex global optimization problems [8]. Hansen and Kern [4], however, point out that on *multi-funnel* functions, where local optima cannot be interpreted as perturbations to an underlying convex (unimodal) topology (cf. Fig. 1), performance can strongly decrease. Prominent examples for such objective functions with multiple funnels are the potential energy surfaces of gases and biomolecules [9], [10]. This connection between the effectiveness of optimization algorithms and the problem classes they can solve is commonly known as the *no free lunch theorem for optimization* [11].

C. L. Müller and I. F. Sbalzarini are with the Institute of Theoretical Computer Science and the Swiss Institute of Bioinformatics, ETH Zurich, CH-8092 Zürich, Switzerland (phone: +41-44-6325512, +41-44-6326344; fax: +41-44-6321562; e-mail: christian.mueller@inf.ethz.ch, ivos@ethz.ch). B. Baumgartner is with the Robotics and Embedded Systems Group, Department of Informatics, Technische Universität München (e-mail: Benedikt.Baumgartner@mytum.de).

There exist, however, only few general metrics that allow classification of non-convex problems into subclasses, a fundamental prerequisite to judge the performance of search heuristics such as ES. The *dispersion metric*, introduced by Lunacek and Whitley [12], attempts such a classification by quantifying the notion of underlying unimodality. Using this metric it was observed that CMA-ES is more successful on low dispersion functions [12]. This behavior of CMA-ES could be due to the fact that CMA-ES was originally developed as a *local* search strategy, whereas the concept of multi-funnel functions is based on *global* information. Such global information is, e.g., used in Particle Swarm Optimization (PSO) as introduced in 1995 by Kennedy and Eberhart [13]. PSO is a population-based optimization method in which particles "fly" over the objective function landscape and employ a *cooperative* strategy to move toward one another based on knowledge of local *and* global best particle positions. Many PSO schemes, however, also suffer from reduced performance on multi-funnel landscapes [14]. We hypothesize that this could be due to a lack in local search performance. This has motivated the recent, independent introduction of a hybrid method [15], the Particle Swarm Guided ES (PSGES), which combines ES for local and PSO for global search. Empirical results have shown, that PSGES enhances the classical ES considerably, but is inferior to IPOP-CMA-ES and, on average, to LR-CMA-ES on almost all multimodal functions. This could be due to a lack in adaptivity of the local search in PSGES.

In order to improve performance on multi-funnel problems, we introduce a combination of PSO with the adaptive CMA-ES, the Particle Swarm CMA-ES (PS-CMA-ES). Our hybrid algorithm considers multiple CMA-ES instances in parallel. Each instance forms a separate swarm particle that communicates with all other swarm particles. The different CMA-ES instances concurrently explore different regions of the search space and exchange information about their current status. By virtue of the collective swarm knowledge, each CMA-ES instance adapts its search direction and distribution. We expect that a combination of the favorable local search properties of CMA-ES and the global exploration power of the PSO paradigm is beneficial, and we test this hypothesis on the IEEE CEC 2005 suite of standard test functions. These benchmarks show that the performance of PS-CMA-ES is superior to other tested CMA variants on noisy test functions, strongly multimodal problems, and multi-funnel problems.

This paper is organized as follows. The next section is dedicated to a short description of CMA-ES, PSO, and the

978-1-4244-2959-2/09/$25.00 © 2009 IEEE

dispersion metric. In Section III the Particle Swarm CMA-ES algorithm will be outlined. In Section IV we determine standard values for the new strategy parameters and assess the performance of PS-CMA-ES on the IEEE CEC 2005 test suite. Section V discusses the results and concludes this work.

II. Prerequisites

A. The CMA Evolution Strategy

We consider a standard CMA-ES [3], [4] with weighted intermediate recombination, step size adaptation, and a combination of rank-μ update and rank-one update [5]. At each iteration of the algorithm, the members of the new population are sampled from a multivariate normal distribution $\mathcal{N}$ with mean $\mathbf{m} \in \mathbb{R}^n$ and covariance $\mathbf{C} \in \mathbb{R}^{n \times n}$. The sampling radius is controlled by the overall standard deviation (step size) σ. Let $\mathbf{x}_k^{(g)}$ the k^{th} individual at generation g. The new individuals at generation $g+1$ are sampled as:

$$\mathbf{x}_k^{(g+1)} \sim \mathbf{m}^{(g)} + \sigma^{(g)} \mathcal{N}\Big(\mathbf{0}, \mathbf{C}^{(g)}\Big) \quad k = 1, \ldots, \lambda\,. \tag{1}$$

The λ sampled points are ranked in order of ascending fitness, and the μ best are selected. The mean of the sampling distribution given in Eq. 1 is updated using *weighted intermediate recombination* of these selected points:

$$\mathbf{m}^{(g+1)} = \sum_{i=1}^{\mu} w_i \mathbf{x}_{i:\lambda}^{(g+1)}\,, \tag{2}$$

with

$$\sum_{i=1}^{\mu} w_i = 1, \quad w_1 \geq w_2 \geq \ldots \geq w_\mu > 0\,, \tag{3}$$

where the w_i are positive weights, and $\mathbf{x}_{i:\lambda}^{(g+1)}$ denotes the i^{th} ranked individual of the λ sampling points $\mathbf{x}_k^{(g+1)}$. We use the standard CMA-ES implementation, where the sample weights are decreased super-linearly as $w_i = \log(\frac{\lambda-1}{2} + 1) - \log(i)$. We adapt the covariance matrix for the next generation using a combination of rank-μ and rank-one update, thus:

$$\mathbf{C}^{(g+1)} = (1 - c_{\text{cov}})\mathbf{C}^{(g)} + \frac{c_{\text{cov}}}{\mu_{\text{cov}}} \underbrace{\mathbf{p}_c^{(g+1)} \mathbf{p}_c^{(g+1)^T}}_{\text{rank-one update}} + c_{\text{cov}}\left(1 - \frac{1}{\mu_{\text{cov}}}\right) \underbrace{\sum_{i=1}^{\mu} w_i \mathbf{y}_{i:\lambda}^{(g+1)} \left(\mathbf{y}_{i:\lambda}^{(g+1)}\right)^T}_{\text{rank-}\mu\text{ update}}, \tag{4}$$

with $\mu_{\text{cov}} \geq 1$ weighting between rank-μ and rank-one update, $c_{\text{cov}} \in [0, 1]$ the learning rate for the covariance matrix update, and $\mathbf{y}_{i:\lambda}^{(g+1)} = (\mathbf{x}_{i:\lambda}^{(g+1)} - \mathbf{m}^{(g)})/\sigma^{(g)}$. The evolution path $\mathbf{p}_c^{(g+1)}$ and the step size $\sigma^{(g)}$ are determined by elaborate adaptation formulae [5].

B. Particle Swarm Optimization

In standard PSO each particle is described by its position $\mathbf{p}$ and its velocity $\mathbf{v}$ in search space ($\mathbb{R}^n$) [13]. While moving through the search space, particles evaluate the fitness function and update their velocity according to an update rule that incorporates both local and global information. The local information of each particle is given by the best solution it has found so far ($\mathbf{p}_{\text{l,best}}$) and the global information corresponds to the best solution any member of the swarm has encountered so far ($\mathbf{p}_{\text{g,best}}$). The velocities of all particles at time $t+1$ are updated from the old velocities and the positions at time t as:

$$\begin{aligned} \mathbf{v}^{(t+1)} = \mathbf{v}^{(t)} \quad & + \\ & c_1 r_1 \left(\mathbf{p}_{\text{l,best}}^{(t)} - \mathbf{p}^{(t)}\right) + \\ & c_2 r_2 \left(\mathbf{p}_{\text{g,best}}^{(t)} - \mathbf{p}^{(t)}\right)\,. \end{aligned} \tag{5}$$

The new positions are computed using an explicit Euler integrator with a time step of 1, thus:

$$\mathbf{p}^{(t+1)} = \mathbf{p}^{(t)} + \mathbf{v}^{(t+1)}\,. \tag{6}$$

The uniform random numbers $r_1, r_2 \in [0, 1]$ introduce a stochastic element to the algorithm, and c_1 and c_2 weight the influence of local vs. global information [13]. As each CMA-ES instance already exploits local information, we set $c_1 = 0$ in PS-CMA-ES. The prefactors c_2 and r_2 are replaced by a more elaborate weighting rule as outlined in Section III.

C. The dispersion metric

We quantify the dispersion of a constrained fitness (objective) function f by considering a random sample of size s_v in search space and a target percentage p. The dispersion $\text{dis}(s_b, s_v, f)$ of f is calculated as the average pair-wise distance between the best $s_b = ps_v$ samples in the search space. A given $p = \frac{s_b}{s_v}$ corresponds to a certain fitness threshold. In order to be able to compare the dispersion values of objective functions with differently constrained search spaces, all distances are normalized to the $[0, 1]^n$ hypercube. Note that the dispersion for $p = 100\%$ corresponds to the hypercube line picking problem [16]. The evolution of the dispersion value is monitored with decreasing p. In order to restrict the number of distance computations for functions in up to $n = 100$ dimensions, p (and hence the fitness threshold) is decreased by fixing the value s_b to 100 and increasing the sample size $s_v = 100 \cdot 2^0, \ldots, 100 \cdot 2^{12}$ [12], corresponding to $p = 100\%, \ldots, 0.0024\%$. The dispersion difference $\Delta_{\text{dis}}(f) = \text{dis}(100, 100 \cdot 2^{12}, f) - \text{dis}(100, 100 \cdot 2^0, f)$ is used as an indicator to compare functions [12]. A negative value of $\Delta_{\text{dis}}(f)$ implies that the best fitness values of f are localized in a small sub-region of the search space, a $\Delta_{\text{dis}}(f)$ value around 0 that the best fitness values of f are either spread over the entire search space or localized in distinct remote funnels. An example of a high-dispersion double-funnel landscape in 2 dimensions is shown in Fig. 1.

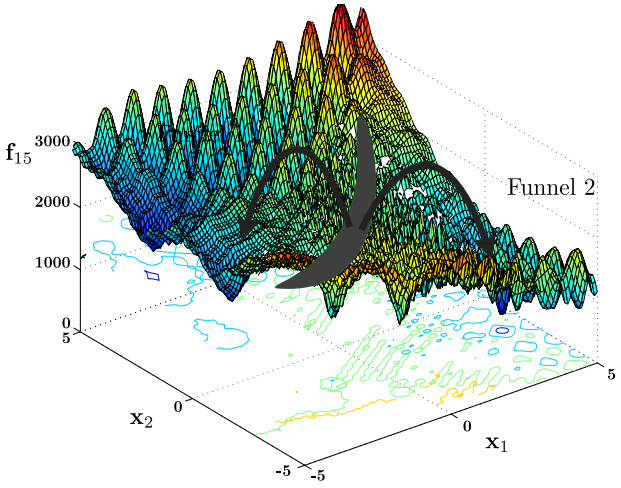

Fig. 1. Two-dimensional version of the highly dispersive function f15 from the CEC benchmark test suite [17]. The global topology is a double funnel separated by the central ridge region (in gray). The global and several local minima are contained in funnel 1, several deep local minima in funnel 2. This topology is hard since a search heuristic can be trapped in the broad funnel 2.

III. Particle Swarm CMA-ES

We hypothesize that exchange of information between parallel CMA-ES runs enhances the performance of CMA-ES on multimodal functions, in particular functions with multiple funnels. Global swarm information is included both in the covariance adaptation mechanism of CMA-ES and in the placement of the population mean.

A. Adapting the Covariance Matrix

We adjust the covariance matrix of the CMA-ES such that it is more likely to sample good candidates, considering the current global best position $\mathbf{p}_{\text{g,best}} \in \mathbb{R}^n$ in the swarm. This is achieved by mixing the CMA covariance matrix from Eq. (4) with a PSO covariance matrix that is influenced by global information:

$$\mathbf{C}^{(g+1)} = c_{\text{p}}\,\mathbf{C}^{(g+1)}_{\text{CMA}} + (1-c_{\text{p}})\,\mathbf{C}^{(g+1)}_{\text{PSO}}\,, \tag{7}$$

where the mixing weight $c_{\text{p}} \in [0,1]$ is a new strategy parameter. $\mathbf{C}^{(g+1)}_{\text{CMA}}$ follows the original adaptation rule from Eq. (4). $\mathbf{C}^{(g+1)}_{\text{PSO}}$ is a rotated version of $\mathbf{C}^{(g)}_{\text{CMA}}$ such that the principal eigenvector $\mathbf{b}_{\text{main}}$ of $\mathbf{C}^{(g)}_{\text{CMA}}$ is aligned with the vector $\mathbf{p}_{\text{g}} = \mathbf{p}_{\text{g,best}} - \mathbf{m}^{(g)}$ that points from the current mean $\mathbf{m}^{(g)}$ toward the global best position $\mathbf{p}_{\text{g,best}}$. $\mathbf{C}^{(g)}_{\text{CMA}}$ can be decomposed as $\mathbf{C}^{(g)}_{\text{CMA}} = \mathbf{B}\,\mathbf{D}^2\,\mathbf{B}^T$, such that the rotated covariance matrix can be constructed by rotating the eigenvectors (columns of $\mathbf{B}$), yielding the orthogonal matrix $\mathbf{B}^{(g)}_{\text{rot}} = \mathbf{R}\,\mathbf{B} \in \mathbb{R}^{n\times n}$ of the rotated eigenvectors. $\mathbf{C}^{(g+1)}_{\text{PSO}}$ is then given by:

$$\mathbf{C}^{(g+1)}_{\text{PSO}} = \mathbf{B}^{(g)}_{\text{rot}} \left(\mathbf{D}^{(g)}\right)^2 \left(\mathbf{B}^{(g)}_{\text{rot}}\right)^T. \tag{8}$$

The rotation matrix $\mathbf{R} \in \mathbb{R}^{n\times n}$ is uniquely and efficiently computed using *Givens Rotations* [18]. The Givens rotation matrix $\mathbf{G}$ describes a unique rotation of a vector onto one axis. An n-dimensional rotation is performed as a sequence of two-dimensional rotations for all possible pairs (i,j) of axes [19]:

$$\mathbf{R}^{(n\times n)} = \prod_{i=1}^{n-1}\prod_{j=i+1}^{n} \mathbf{R}_{ij}\,. \tag{9}$$

$\mathbf{R}_{ij}$ is an $n\times n$ matrix that describes the rotation in the plane spanned by the axes (i,j). It can be considered as a rank-two correction to the identity:

$$\mathbf{R}_{ij,\mathbf{R}_{\text{plane}}} = \begin{pmatrix} 1 & \dots & 0 & \dots & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & & \vdots & & \vdots \\ 0 & \dots & \mathbf{R}_{\text{plane}}(1,1) & \dots & \mathbf{R}_{\text{plane}}(1,2) & \dots & 0 \\ \vdots & & \vdots & \ddots & \vdots & & \vdots \\ 0 & \dots & \mathbf{R}_{\text{plane}}(2,1) & \dots & \mathbf{R}_{\text{plane}}(2,2) & \dots & 0 \\ \vdots & & \vdots & & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & \dots & 0 & \dots & 1 \end{pmatrix} \begin{matrix} 1 \\ \\ i \\ \\ j \\ \\ n \end{matrix}, \tag{10}$$
$$\begin{matrix} 1 & & i & & j & & n \end{matrix}$$

where the 2×2 matrix $\mathbf{R}_{\text{plane}} = \mathbf{G}_{\text{p}}^T\,\mathbf{G}_{\text{b}}$. $\mathbf{G}_{\text{p}}$ is the Givens rotation of the elements i and j of $\mathbf{p}_{\text{g}}$ and $\mathbf{G}_{\text{b}}$ the Givens rotation of the elements i and j of $\mathbf{b}_{\text{main}}$. The complete procedure to compute the rotation matrix $\mathbf{R} \in \mathbb{R}^{n\times n}$ is summarized in Algorithm 1.

Input: Two n-dimensional vectors $\mathbf{b}_{\text{main}}$ and $\mathbf{p}_{\text{g}}$
Result: Rotation matrix $\mathbf{R}$, such that $\mathbf{R}\,\mathbf{b}_{\text{main}} = a\,\mathbf{p}_{\text{g}}$
Initialization: $\mathbf{p} = \mathbf{p_g}$, $\mathbf{b} = \mathbf{b_{main}}$
for $i = (n-1), -1, 1$ **do**
  for $j = n, -1, (i+1)$ **do**
    $\mathbf{p_{plane}} = \begin{pmatrix} p(i) \\ p(j) \end{pmatrix}$ $\quad$ $\mathbf{b_{plane}} = \begin{pmatrix} b(i) \\ b(j) \end{pmatrix}$
    $\mathbf{G_p} = \mathbf{Givens}(\mathbf{p_{plane}})$ $\quad$ $\mathbf{G_b} = \mathbf{Givens}(\mathbf{b_{plane}})$
    $\mathbf{p} \leftarrow \mathbf{R}_{ij,\mathbf{G_p}}\,\mathbf{p}$ $\quad$ $\mathbf{b} \leftarrow \mathbf{R}_{ij,\mathbf{G_b}}\,\mathbf{b}$
    $\mathbf{R_p} \leftarrow \mathbf{R}_{ij,\mathbf{G_p}}\,\mathbf{R_p}$ $\quad$ $\mathbf{R_b} \leftarrow \mathbf{R}_{ij,\mathbf{G_b}}\,\mathbf{R_b}$
  end
end
$\mathbf{R} = \mathbf{R_p}^T\mathbf{R_b}$

Algorithm 1: Efficient computation of the n-dimensional rotation matrix using Givens rotations

B. Biasing the Mean Value

In order to enable individual swarm particles of a PS-CMA-ES to escape local minima, we also bias the mean value in addition to rotating the covariance matrix in direction of the global best solution. After the recombination step of each CMA-ES generation, the updated mean value for the next generation $g+1$ is biased as:

$$\mathbf{m}^{(g+1)} \leftarrow \mathbf{m}^{(g+1)} + \mathbf{bias}\,. \tag{11}$$

Note that the bias changes the evolution path $\mathbf{p}_c^{(g+1)}$ update for future generations, since the path will be computed with respect to the biased mean value. The biasing rules can be used to include prior knowledge about the structure of the problem, e.g. the correlation length of the fitness landscape. In our benchmark tests, we discriminate the following 3 exploration scenarios that are coupled to the step size σ of each individual CMA-ES instance, a natural measure for its mode of exploration:

1) The CMA-ES instance that produced $\mathbf{p}_{\mathrm{g,best}}$ and all CMA-ES instances with step sizes $\sigma > ||\mathbf{p}_{\mathrm{g}}||$ are not biased at all, thus: **bias** $= 0$. Using no bias in these cases avoids catapulting the global best member out of the funnel with the potential global optimum and prevents very explorative runs from overshooting the target.
2) CMA-ES instances that have converged to local minima far from $\mathbf{p}_{\mathrm{g,best}}$ are characterized by a ratio $\sigma/||\mathbf{p}_{\mathrm{g}}||$ that is smaller than $t_{\mathrm{c}}||\mathbf{p}_{\mathrm{g}}||$, $t_{\mathrm{c}} < 1$. These instances are strongly biased in order to allow them to escape from the current funnel, thus: **bias** $= b\,\mathbf{p}_{\mathrm{g}}$. Using a large bias prevents these instances from converging back into the same local minimum again.
3) CMA-ES instances that are not converged, and thus still exploring the space, are given a bias equal to the step size to distance ratio: **bias** $= \sigma/||\mathbf{p}_{\mathrm{g}}||\,\mathbf{p}_{\mathrm{g}}$. Using such a small bias prevents clustering of swarm members and preserves the explorative power of the method.

This set of biasing rules, summarized in Algorithm 2, introduces two new strategy parameters, the convergence threshold t_{c} and the biasing factor b.

Input: Mean value $\mathbf{m}^{(g+1)}$ and global best direction $\mathbf{p}_{\mathrm{g}}$
Result: Biased mean value $\mathbf{m}^{(g+1)}$
if $\sigma < ||\mathbf{p}_g||$ **then**
  if $\frac{\sigma}{||\mathbf{p}_g||} \leq t_c||\mathbf{p}_g||$ **then**
    bias $= b\,\mathbf{p}_{\mathrm{g}}$
  else
    bias $= \frac{\sigma}{||\mathbf{p}_{\mathrm{g}}||}\,\mathbf{p}_{\mathrm{g}}$
  end
else
  bias $= 0$
end
$\mathbf{m}^{(g+1)} = \mathbf{m}^{(g+1)} +$ **bias**

Algorithm 2: **Standard rules for biasing the mean value**

C. Strategy Parameters

In PS-CMA-ES, all swarm members communicate with each other. A swarm of size S thus requires $S(S-1)$ communication steps. The PSO updates (broadcasting the global best solution, rotating the covariance matrices, and biasing the mean values of all CMA instances) must, however, not be performed at each iteration. Too frequent updates would prevent the CMA-ES instances from evolving and learning the local covariance matrix. Too infrequent updates would lead to premature convergence with several CMA-ES instances stopping in local minima before the first swarm information is exchanged. Clearly, a problem-specific tradeoff has to be found for the communication interval I_{c} between PSO updates, constituting the main strategy parameter of the PS-CMA-ES. In the limit of $I_{\mathrm{c}} \to \infty$, PS-CMA-ES is equivalent to S parallel standard CMA-ES runs.

Other strategy parameters are the swarm size S, the biasing parameters t_{c} and b, and the mixing weight c_{p} in Eq. (7). Both t_{c} and b have been considered random variables at first, but the setting $t_{\mathrm{c}} = 0.1$ and $b = 0.5$ was found a more robust choice on most test functions. The weight c_{p} can, e.g., be randomized, self-adapted, or determined by a preliminary grid search. The swarm size could be chosen so as to reflect the dimensionality of the problem or also using a grid search. It determines the overall population size and the computational overhead of the algorithm. Therefore, S should be chosen as low as possible, but as high as necessary to significantly increase exploration power. In the limit case $S = 1$, PS-CMA-ES is equivalent to standard CMA-ES.

IV. Benchmarks

A. Evaluation Criteria

1) Test Functions: We test the presented PS-CMA-ES algorithm on the 25 benchmark functions provided by Suganthan et al. [17] during the CEC 2005 Special Session on Real-Parameter Optimization. This CEC 2005 test suite defines a standard benchmark for real-parameter optimization algorithms along with standardized evaluation criteria and testing procedures. It allows comparing the performance of different optimization algorithms. Functions f1 to f5 of the benchmark are unimodal (U) and f6 to f12 are basic multimodal (BM). Functions f13 and f14 are expanded (E) and f15 to f25 are hybrid (H) test functions that are formed by combining several standard test functions (Table I). The multi-funnel functions f6, f11–f13, and f15–f25, where local optima cannot be interpreted as perturbations to an underlying convex (unimodal) function (cf. Fig. 1), are identified by an "x" in column "M" (Table I). In order to prevent exploitation of search space symmetry, all problems are shifted and many of them are rotated. Moreover, the global optimum of each function is different from the common zero value. Functions f4 and f17 are corrupted by addition of white noise. We compute the dispersion of all test functions for $n = 10, 30$, and 50 dimensions as outlined in Section II-C. The means (and ranks) of $\Delta_{\mathrm{dis}}(f)$ are summarized in Table I. The standard deviations of the $\Delta_{\mathrm{dis}}(f)$ estimates are between 0.02 and 0.03 for all functions and dimensions.

2) Benchmark Settings: According to the evaluation criteria of the CEC 2005 test suite [17], we benchmark the PS-CMA-ES in $n = 10, 30$, and 50 dimensions. Each minimization problem is repeated 25 times with uniformly random initialization. The maximum allowed number of function evaluations (MAX_FES) is defined by the problem dimension

Prob.	Kind	M	$\Delta_{\text{dis}}(f)$, n=10	$\Delta_{\text{dis}}(f)$, n=30	$\Delta_{\text{dis}}(f)$, n=50
f1	U		-0.73727 (23)	-0.60151 (23)	-0.53034 (23)
f2	U		-0.59705 (19)	-0.31114 (11)	-0.21999 (7)
f3	U		-0.39001 (6)	-0.25648 (8)	-0.26603 (9)
f4	U		-0.58483 (18)	-0.28354 (9)	-0.21695 (6)
f5	U		-0.57241 (16)	-0.35923 (14)	-0.30067 (12)
f6	BM	x	-0.66969 (21)	-0.58908 (21)	-0.51453 (21)
f7	BM		- (-)	- (-)	- (-)
f8	BM		**0.0056344 (1)**	**0.0011693 (1)**	**-0.00090142 (1)**
f9	BM		-0.52925 (11)	-0.491 (16)	-0.45898 (16)
f10	BM		-0.61472 (20)	-0.54152 (19)	-0.47772 (20)
f11	BM	x	**-0.00885 (2)**	**-0.00045 (2)**	**0.00442 (2)**
f12	BM	x	**-0.38210 (5)**	**0.18593 (4)**	**-0.20403 (5)**
f13	E	x	-0.68613 (22)	-0.59441 (22)	-0.51738 (22)
f14	E		**-0.11396 (3)**	**-0.056395 (3)**	**-0.05234 (3)**
f15	H	x	**-0.30746 (4)**	**-0.19188 (5)**	**-0.14521 (4)**
f16	H	x	-0.58337 (17)	-0.24801 (7)	-0.27069 (10)
f17	H	x	-0.55598 (14)	-0.21134 (6)	-0.24537 (8)
f18	H	x	-0.44166 (9)	-0.3482 (12)	-0.33211 (13)
f19	H	x	-0.42928 (7)	-0.35427 (13)	-0.33423 (14)
f20	H	x	-0.43469 (8)	-0.36426 (15)	-0.33649 (15)
f21	H	x	-0.54107 (12)	-0.53763 (18)	-0.47505 (19)
f22	H	x	-0.48245 (10)	-0.28543 (10)	-0.30002 (11)
f23	H	x	-0.55078 (13)	-0.54313 (20)	-0.4707 (17)
f24	H	x	-0.56082 (15)	-0.50595 (17)	-0.47219 (18)
f25	H	x	- (-)	- (-)	- (-)

TABLE I

MEAN DISPERSIONS

$\Delta_{\text{dis}}(f) = \text{dis}(100, 100 \cdot 2^{12}, f) - \text{dis}(100, 100 \cdot 2^{0}, f)$ AND RANKS (IN BRACKETS) FOR ALL FUNCTIONS IN THE CEC 2005 BENCHMARK TEST SUITE FOR $n = 10, 30,$ AND 50. F7 AND F25 ARE UNBOUNDED AND HAVE NO DEFINED DISPERSION. THE TEST SUITE CONTAINS UNIMODAL (U), BASIC MULTIMODAL (BM), EXPANDED (E), AND HYBRID (H) FUNCTIONS. FUNCTIONS F8, F11–F12, AND F14–F15 (IN BOLD) ARE HIGHLY DISPERSIVE. MULTI-FUNNEL FUNCTIONS ARE IDENTIFIED BY AN "X" IN COLUMN "M".

as: MAX_FES $= 10^4\, n$. A run is terminated before reaching MAX_FES if the function error value drops below 10^{-8}.

3) Performance Measures: Auger and Hansen stated that some algorithms may have a small probability of success, but converge fast, while others may be slower, but with a larger probability of success [7]. We thus measure algorithm performance using two orthogonal quantities [17], the **Success Rate** $=$ **#successful runs**/**#runs** and the **Success Performance** $=$ **mean(FES for successful runs)** $\cdot$ **#runs**/**#successful runs**. The *Success Rate* is an estimator for the probability of success, while the *Success Performance* is an estimator for the speed of convergence. A run is counted successful if the function error reaches a given accuracy (specified in the CEC 2005 benchmark) before the maximum allowed number of function evaluations (MAX_FES) is reached.

4) Computational Cost: We quantify the computational cost of the PS-CMA-ES algorithms using the three measures defined in the CEC 2005 benchmark [17]: $T0$ is the computing time for standard operations, $T1$ is the time needed to evaluate function f3 200 000 times in a certain dimension, and $\hat{T}2$ is the mean time over five executions of the complete algorithm with 200 000 evaluations of function f3 each.

B. PS-CMA-ES Strategy Parameters

We use grid search to determine good values for the strategy parameters introduced in Section III-C. This determination of strategy parameters can be considered as an unbiased *training* of the algorithm. The strategy parameters for single CMA-ES instances are set according to Hansen [5]. Only the initial step-size σ is varied between 20% and 50% of the constrained region of the minimization problem. Table II summarizes all strategy parameters tested in the grid search. All 144 combinations are tested for functions f1–f25 in $n = 10$ dimensions. The benchmark requires 25 repetitions per problem [17], leading to a total of $25 \cdot 25 \cdot 144 = 90\,000$ PS-CMA-ES runs. The grid search suggests an initial step size of $\sigma = 0.2$, a swarm size of at least $S = 10$, a mixing weight of $c_{\text{p}} = 0.5 - 0.7$, and swarm updates every 200 – 250 generations. Although there may be better configurations for individual test cases, these settings performed well on most problems. In the following, we use the standard setting $\sigma = 0.2$, $S = 15$, $c_{\text{p}} = 0.7$, and $I_{\text{c}} = 200$. This provides robust performance and good tradeoffs over the different test problems. We test the *generality* of this parameter set by using it also for the benchmarks in 30 and 50 dimensions, without repeating the grid search.

Parameter	Tested values
step size σ	0.2, 0.3, 0.4, 0.5
swarm size S	6, 10, 15
mixing weight c_{p}	0.3, 0.5, 0.7
communication interval I_{c}	150, 200, 250, ∞

TABLE II

VALUES OF THE STRATEGY PARAMETERS TESTED IN THE GRID SEARCH.

C. Benchmark Results and Comparison to Related Algorithms

We compare the performance of PS-CMA-ES on the CEC 2005 benchmark to two variants of CMA-ES, LR-CMA-ES [6] and IPOP-CMA-ES [7], as well as to the particle swarm guided ES PSGES [15]. The performance results of the reference algorithms are taken from the corresponding publications [6], [7], [15]. LR-CMA-ES and IPOP-CMA-ES data are available for $n = 10, 30, 50$, PSGES data only for $n = 10$. We strictly follow the performance evaluation criteria prescribed by the CEC 2005 benchmark [17], but only report the key observations in the following.

1) PS-CMA-ES in $n = 10$ dimensions: PS-CMA-ES successfully solves 11 test functions as summarized in Table III. From the other algorithms, only IPOP-CMA-ES is able to solve the same number of function, whereas both LR-CMA-ES and PSGES solve 8 problems [8], [15]. Moreover, PS-CMA-ES is the only CMA-ES variant that solves the multi-funnel, hybrid composition function f15 (Fig. 1). While PS-CMA-ES fails to solve f3 and needs considerably more function evaluations than LR-CMA-ES and IPOP-CMA-ES to solve f1, f2, and f4–f7, it outperforms the others on f9–f10 and on the highly dispersive functions f11–f12 in terms

of success rate, success performance, and mean function error. Fine tuning of the strategy parameters for individual cases enabled PS-CMA-ES to solve 12 test problems in $n = 10$. In particular, the hybrid composition function f16 was solved. To our knowledge, this makes PS-CMA-ES the first algorithm reported to solve any of the functions f16–f25.

	min	med	max	mean	std	Suc.Rate	Suc.Perf.
f1	2.03e+04	2.16e+04	2.31e+04	2.18e+04	8.04e+02	1.00	2.18e+04
f2	3.12e+04	3.32e+04	3.47e+04	3.30e+04	9.27e+02	1.00	3.30e+04
f3	-	-	-	-	-	0.00	-
f4	3.23e+04	3.45e+04	3.69e+04	3.45e+04	1.34e+03	1.00	3.45e+04
f5	9.38e+04	-	-	9.56e+04	2.28e+03	0.24	3.98e+05
f6	7.05e+04	8.12e+04	8.49e+04	8.01e+04	4.11e+03	1.00	8.01e+04
f7	2.16e+04	2.39e+04	2.60e+04	2.39e+04	1.18e+03	1.00	2.39e+04
f8	-	-	-	-	-	0.00	-
f9	6.78e+03	7.68e+03	8.28e+03	7.57e+03	3.68e+02	1.00	7.57e+03
f10	7.38e+03	8.43e+03	9.33e+03	8.39e+03	5.00e+02	1.00	8.39e+03
f11	3.56e+04	-	-	5.02e+04	1.81e+04	0.40	1.25e+05
f12	2.45e+04	2.81e+04	2.96e+04	2.78e+04	1.59e+03	1.00	2.78e+04
f13	-	-	-	-	-	0.00	-
f14	-	-	-	-	-	0.00	-
f15	6.55e+04	-	-	7.21e+04	9.29e+03	0.08	9.01e+05

TABLE III

PS-CMA-ES PERFORMANCE MEASURES FOR SUCCESSFULLY SOLVED PROBLEMS. PROB.: PROBLEM NUMBER; COLUMNS 2–6: NUMBER OF FUNCTION EVALUATIONS (MINIMUM, MEDIAN, MAXIMUM, MEAN, AND STANDARD DEVIATION) TO REACH REQUIRED ACCURACY; COLUMNS 7–8: SUCCESS RATE AND SUCCESS PERFORMANCE.

Prob.	PS-CMA-ES	LR-CMA-ES	IPOP-CMA-ES	PSGES
f1	7.84e-09 (4)	5.14e-09 (2)	5.20e-09 (3)	0.00e+00 (1)
f2	7.66e-09 (4)	5.31e-09 (3)	4.70e-09 (2)	0.00e+00 (1)
f3	1.59e+00 (3)	4.94e-09 (1)	5.60e-09 (2)	3.17e+00 (4)
f4	7.71e-09 (3)	1.79e+06 (4)	5.02e-09 (2)	1.36e-14 (1)
f5	3.73e-05 (3)	6.59e-09 (2)	6.58e-09 (1)	1.05e+02 (4)
f6	**1.93e-05 (3)**	**5.41e-09 (2)**	**4.87e-09 (1)**	**1.59e-01 (4)**
f7	8.07e-09 (3)	4.91e-09 (2)	3.31e-09 (1)	7.39e-03 (4)
f8	2.03e+01 (3)	2.00e+01 (2)	2.00e+01 (1)	2.09e+01 (4)
f9	7.46e-09 (1)	4.49e+01 (4)	2.39e-01 (2)	3.46e+00 (3)
f10	7.93e-09 (1)	4.08e+01 (4)	7.96e-02 (2)	1.46e+01 (3)
f11	**1.33e-01 (1)**	**3.65e+00 (3)**	**9.34e-01 (2)**	**1.35e+01 (4**
f12	**7.86e-09 (1)**	**2.09e+02 (3)**	**2.93e+01 (2)**	**3.60e+02 (4)**
f13	**3.95e-01 (1)**	**4.94e-01 (2)**	**6.96e-01 (3)**	**8.21e-01 (4)**
f14	3.47e+00 (2)	4.01e+00 (3)	3.01e+00 (1)	5.00e+00 (4)
f15	**8.12e+01 (1)**	**2.11e+02 (2)**	**2.28e+02 (3)**	**3.26e+02 (4)**
f16	**8.97e+01 (1)**	**1.05e+02 (2)**	**9.31e+04 (4)**	**2.01e+02 (3)**
f17	**1.03e+02 (1)**	**5.49e+02 (3)**	**1.23e+02 (2)**	**3.03e+03 (4)**
f18	**4.72e+02 (2)**	**4.97e+02 (3)**	**3.32e+02 (1)**	**7.15e+02 (4)**
f19	**4.67e+02 (2)**	**5.16e+02 (3)**	**3.26e+02 (1)**	**6.69e+02 (4)**
f20	**4.94e+02 (3)**	**4.42e+02 (2)**	**3.00e+02 (1)**	**7.05e+02 (4)**
f21	**5.57e+02 (3)**	**4.04e+02 (1)**	**5.00e+02 (2)**	**8.89e+02 (4)**
f22	**5.87e+02 (1)**	**7.04e+02 (2)**	**7.29e+02 (3)**	**8.11e+02 (4)**
f23	**6.43e+02 (2)**	**7.91e+02 (3)**	**5.59e+02 (1)**	**1.08e+03 (4)**
f24	**4.03e+02 (2)**	**8.65e+02 (4)**	**2.00e+02 (1)**	**4.19e+02 (3)**
f25	**4.03e+02 (2)**	**4.42e+02 (4)**	**3.74e+02 (1)**	**4.15e+02 (3)**
Rank	2.12 (53/25)	2.64 (66/25)	1.8 (45/25)	3.44 (86/25)

TABLE IV

RANKING OF THE METHODS BASED ON THE MEAN FUNCTION ERROR AFTER 10^5 FUNCTION EVALUATIONS FOR $n = 10$. MULTI-FUNNEL CASES ARE IN BOLD.

Based on the mean (over all 25 repetitions of each problem) function value error after MAX_FES=10^5, we rank the different methods for performance comparison [15]. The results are given in Table IV. PS-CMA-ES outperforms PSGES on f13–f25, and it outperforms LR-CMA-ES on f13–f19 and f22–f25. Moreover, PS-CMA-ES performs better than IPOP-CMA-ES on the multi-funnel functions f13 and f22, and especially on the set f15–f17 (f16 is a rotated version of f15 and f17 is a noisy version of f16). The incorporation of global information from the particle swarm seems to be favorable on hybrid problems.

2) PS-CMA-ES in $n = 30$ and $n = 50$ dimensions: Based on the mean function errors reported in Tables V and VI, PS-CMA-ES outperforms the other CMA-ES variants in 30 and 50 dimensions. On the noisy functions f4 and f17, it is orders of magnitude better than IPOP-CMA-ES and LR-CMA-ES. On the expanded and hybrid functions f13–f25, PS-CMA-ES is superior to both other algorithms, except for f15, f21, and f23. PS-CMA-ES is, however, dominated on the unimodal functions without noise, the multi-funnel function f6, and the functions f7 and f8. For $n = 30$, PS-CMA-ES solves f1, f2, f7, f9, f10, and f12. For $n = 50$, PS-CMA-ES solves f1, f7, f9, and f10 with a success rate of 1 and f12 with a success rate of 0.2. These results are outmatched by IPOP-CMA-ES, which solves 11 problems for $n = 30$ and 7 for $n = 50$, respectively, due to its powerful performance on unimodal functions.

In summary, PS-CMA-ES is a good choice on noisy or multi-funnel functions. Although the number of solved

Prob.	PS-CMA-ES	LR-CMA-ES	IPOP-CMA-ES
f1	8.79e-09 (3)	5.28e-09 (1)	5.42e-09 (2)
f2	9.26e-09 (3)	6.93e-09 (2)	6.22e-09 (1)
f3	8.00e+04 (3)	5.18e-09 (1)	5.55e-09 (2)
f4	8.47e-04 (1)	9.26e+07 (3)	1.11e+04 (2)
f5	3.98e+02 (3)	8.30e-09 (1)	8.62e-09 (2)
f6	**1.35e+01 (3)**	**6.31e-09 (2)**	**5.90e-09 (1)**
f7	9.33e-09 (3)	6.48e-09 (2)	5.31e-09 (1)
f8	2.10e+01 (3)	2.00e+01 (1)	2.01e+01 (2)
f9	8.85e-09 (1)	2.91e+02 (3)	9.38e-01 (2)
f10	8.98e-09 (1)	5.63e+02 (3)	1.65e+00 (2)
f11	**3.91e+00 (1)**	**1.52e+01** (3)	**5.48e+00 (2)**
f12	**7.89e+01 (1)**	**1.32e+04** (2)	**4.43e+04 (3)**
f13	**2.11e+00 (1)**	**2.32e+00 (2)**	**2.49e+00 (3)**
f14	1.29e+01 (1)	1.40e+01 (3)	1.29e+01 (2)
f15	**2.10e+02 (2)**	**2.16e+02** (3)	**2.08e+02 (1)**
f16	**2.61e+01 (1)**	**5.84e+01** (3)	**3.50e+01 (2)**
f17	**5.17e+01 (1)**	**1.07e+03** (3)	**2.91e+02 (2)**
f18	**8.16e+02 (1)**	**8.90e+02** (2)	**9.04e+02 (3)**
f19	**8.16e+02 (1)**	**9.03e+02** (2)	**9.04e+02 (3)**
f20	**8.16e+02 (1)**	**8.89e+02** (2)	**9.04e+02 (3)**
f21	**7.11e+02 (3)**	**4.85e+02** (1)	**5.00e+02 (2)**
f22	**5.00e+02 (1)**	**8.71e+02** (3)	**8.03e+02 (2)**
f23	**7.99e+02 (3)**	**5.35e+02** (2)	**5.34e+02 (1)**
f24	**2.10e+02 (1)**	**1.41e+03** (3)	**9.10e+02 (2)**
f25	**2.10e+02 (1)**	**6.91e+02** (3)	**2.11e+02 (2)**
Rank	1.76 (44/25)	2.24 (56/25)	2.00 (50/25)

TABLE V

RANKING OF THE METHODS BASED ON THE MEAN FUNCTION ERROR AFTER $3 \cdot 10^5$ FUNCTION EVALUATIONS FOR $n = 30$. MULTI-FUNNEL CASES ARE IN BOLD.

Prob.	PS-CMA-ES	LR-CMA-ES	IPOP-CMA-ES
f1	9.09e-09 (3)	6.20e-09 (2)	5.87e-09 (1)
f2	9.79e-04 (3)	7.96e-09 (2)	7.86e-09 (1)
f3	3.28e+05 (3)	6.04e-09 (1)	6.14e-09 (2)
f4	1.58e+03 (1)	4.46e+08 (3)	4.68e+05 (2)
f5	1.18e+03 (3)	3.27e+00 (2)	2.85e+00 (1)
f6	**2.98e+01 (3)**	**7.12e-09 (1)**	**7.13e-09 (2)**
f7	9.43e-09 (3)	7.49e-09 (2)	7.22e-09 (1)
f8	2.11e+01 (3)	2.00e+01 (1)	2.01e+01 (2)
f9	9.37e-09 (1)	5.67e+02 (3)	1.39e+00 (2)
f10	9.23e-09 (1)	1.48e+03 (3)	1.72e+00 (2)
f11	**1.22e+01 (2)**	**3.41e+01 (3)**	**1.17e+01 (1)**
f12	**2.36e+03 (1)**	**8.93e+04 (2)**	**2.27e+05 (3)**
f13	**4.00e+00 (1)**	**4.70e+00 (3)**	**4.59e+00 (2)**
f14	2.25e+01 (1)	2.39e+01 (3)	2.29e+01 (2)
f15	**2.64e+02 (3)**	**2.50e+02 (2)**	**2.04e+02 (1)**
f16	**2.27e+01 (1)**	**7.09e+01 (3)**	**3.09e+01 (2)**
f17	**6.16e+01 (1)**	**1.05e+03 (3)**	**2.34e+02 (2)**
f18	**8.36e+02 (1)**	**9.06e+02 (2)**	**9.13e+02 (3)**
f19	**8.36e+02 (1)**	**9.11e+02 (2)**	**9.12e+02 (3)**
f20	**8.36e+02 (1)**	**9.01e+02 (2)**	**9.12e+02 (3)**
f21	**7.18e+02 (2)**	**5.00e+02 (1)**	**1.00e+03 (3)**
f22	**5.00e+02 (1)**	**9.10e+02 (3)**	**8.05e+02 (2)**
f23	**7.24e+02 (2)**	**6.37e+02 (1)**	**1.01e+03 (3)**
f24	**2.14e+02 (1)**	**8.43e+02 (2)**	**9.55e+02 (3)**
f25	**2.14e+02 (1)**	**4.77e+02 (3)**	**2.15e+02 (2)**
Rank	1.76 (44/25)	2.2 (55/25)	2.04 (51/25)

TABLE VI

RANKING OF THE METHODS BASED ON THE MEAN FUNCTION ERROR AFTER $5 \cdot 10^5$ FUNCTION EVALUATIONS FOR $n = 50$. MULTI-FUNNEL CASES ARE IN BOLD.

problems is reduced compared to the other CMA-ES variants, PS-CMA-ES shows more robust performance with respect to the mean function error. On f4, f9, f10, f12, and f17 PS-CMA-ES is orders of magnitude better than the competitors.

3) PS-CMA-ES performance and function dispersion: We observe that the computed function dispersion values correlate neither with PS-CMA-ES nor with IPOP-CMA-ES success performance. Performance on the most dispersive function f8 is, e.g., similar for all tested algorithms. This could be due to the fact that f8 has a globally flat structure with one narrow, deep funnel at the boundary. This funnel is almost impossible to detect in higher dimensions. There is also a discrepancy on the function pair f9/f10. Function f9 is a shifted Rastrigin function with moderate dispersion and a clearly unimodal global topology. Its rotated version f10, however, has a considerably lower dispersion. PS-CMA-ES performs equally well on both functions, which is not predicted by the dispersion metric. On the pair f15/f16 (f16 is a rotated version of f15), the situation is similar. With standard parameter settings, PS-CMA-ES is able to solve f15, but not f16. This is not predicted by the dispersion metric, which has a lower value for f16 than for f15. There is also a discrepancy on the functions f11 and f12 (modified Weierstrass and Schwefel), which are highly dispersive, but are well optimized by PS-CMA-ES. These discrepancies could be explained by the fact that the set of highly dispersive functions is overlapping, but not identical with the set of multi-funnel functions. The hybrid functions f16–f24, e.g., have multiple remote funnels, but appear as lowly dispersive as the unimodal functions f1–f5.

4) Computational cost: The computational cost of PS-CMA-ES is reported in Table VII and compared to a standard CMA-ES implementation. We implemented both algorithms in Fortran 90 and all tests were run on a computer with 1 GB RAM and 2 Dual-Core Intel Xeon processors at 3 GHz, running MacOS 10.4.11. For PS-CMA-ES, a swarm of size of $S = 4$ was used. It can be seen that $\hat{T}2$ of PS-CMA-ES is comparable to the computational cost of a standard CMA-ES up to $n = 30$ dimensions. In higher dimensions, the computational cost of PS-CMA-ES increases due to the multi-dimensional matrix rotations.

Algorithm | CMA-ES / PS-CMA-ES				
	$T0$	$T1$	$\hat{T}2$	$(\hat{T}2 - T1)/T0$
$n = 10$	9.53e-2	3.02e-1	3.96e+0 3.87e+0	3.84e+1 3.75e+1
$n = 30$		2.26e+0	1.39e+1 1.55e+1	1.22e+2 1.39e+2
$n = 50$		6.49e+0	3.53e+1 5.04e+1	3.02e+2 4.61e+2

TABLE VII

COMPUTATIONAL COST FOR STANDARD CMA-ES (FIRST ROWS) AND PS-CMA-ES (SECOND ROWS).

V. DISCUSSION AND CONCLUSIONS

We have presented a hybrid Particle Swarm CMA Evolution Strategy, the PS-CMA-ES. Each CMA instance is considered an individual swarm particle. Global knowledge is included in the CMA sampling distribution using two mechanisms: (1) rotation of the covariance matrix such that the longest eigenvector points in the direction of the global best, and (2) shifting the mean of the sampling distribution toward the global best. We have described a computationally efficient algorithm for uniquely rotating the covariance matrix in high-dimensional spaces. The presented method adds five strategy parameters: the swarm size S, the mixing weight c_p, the biasing parameters t_c and b, and the communication interval I_c. The biasing rules for the sampling mean provide additional means of accounting for prior knowledge about the optimization problem at hand. We have determined, in an unbiased way, standard values for all strategy parameters that provide good average performance on the wide range of test functions represented in the CEC 2005 benchmark suite, hence rendering PS-CMA-ES practically parameter free. Using these standard parameters, we have evaluated the performance of the PS-CMA-ES algorithm and compared it to two competitive variants of CMA-ES, namely the LR-CMA-ES and the IPOP-CMA-ES [7], [6], as well as to PSGES, another hybrid particle swarm evolution strategy [15].

Our benchmarks have shown the superior performance of PS-CMA-ES on noisy test functions (f4 and f17), strongly multimodal problems (Rastrigin function, f9, f10), and multi-funnel problems (Schwefel problem, f12, hybrid composition functions). Our results for the dispersion metric also suggest that this metric is not sufficient to detect the absence of a

global convex function topology as is, e.g., the case for the hybrid test functions. The dispersion metric can, thus, not fully explain when and why PS-CMA-ES or IPOP-CMA-ES are advantageous. We believe that PS-CMA-ES benefits from the increased global exploration power introduced by the swarm communication. Analyzing the search space covering of the different algorithms will be subject to future research.

With standard parameter settings, PS-CMA-ES presented no advantage over the reference algorithms on unimodal functions and several basic multimodal functions. This could potentially be improved by tuning the strategy parameters of PS-CMA-ES specifically for these cases, as indicated by our parameter sweep for $n = 10$ dimensions. We observed that runs without global communication and smaller swarm sizes would perform better on these problems. This was expected since the standard CMA-ES approximates well the fitness landscape of unimodal functions and communication could only distract its convergence. For larger swarm sizes, there is only little time for individual CMA-ES instances to converge within the maximum allowed number of function evaluations.

The computational cost of PS-CMA-ES was assessed and compared to standard CMA-ES. We found that the communication overhead was negligible for all practical swarm sizes. The main computational cost is caused by the rotation of the covariance matrix. The rotation algorithm presented in Section III requires $2n\,(n\,-\,1)\,+\,1$ matrix multiplications at each PSO update in order to construct $\mathbf{R}$ for each swarm member. Hence, the computational cost increases quadratically with the number of dimensions and linearly with swarm size. This can, to a certain extent, be relaxed by choosing the smallest possible swarm size and performing PSO updates less frequently (controlled through the strategy parameter I_{c}). In addition, alternative biasing schemes that do not require n-dimensional rotations could be investigated. It is also noteworthy that PS-CMA-ES presents an intrinsically parallel algorithm. This would, e.g., allow leveraging of the computational performance of multi-core platforms, in particular when the swarm size is chosen as an integer multiple of the number of processing cores. PS-CMA-ES thus provides a straightforward way to benefit from the anticipated future increase in the number of cores per chip by using this parallelism to increase the exploration power of the search.

While PS-CMA-ES seems to perform well using the standard parameter settings determined in this work, its performance could be further improved by refined parameter choices. In particular, it could be advantageous to couple the communication interval and the swarm size to the problem dimension.

In summary, we believe that PS-CMA-ES is a good candidate to supplement CMA-ES with global information via PSO-like operations. The increased exploration power will be particularly beneficial on noisy or multi-funnel problems. Such problems frequently occur in, e.g., polymer physics, molecular dynamics, and systems biology. As the proliferation of multi-core platforms is expected to leverage much of the computational overhead, PS-CMA-ES could prove a valuable tool in these and other domains of application.

ACKNOWLEDGMENTS

The authors thank Prof. Utschick, Technical University of Munich, for his help and for serving as the official supervisor of BB. CLM and BB thank Dr. Grégory Paul, Birte Schrader, and Jo Arne Helmuth for valuable discussions and *Nachlader* for the album "Bock auf Aphorismen".

REFERENCES

[1] R. T. Rockafellar, "Lagrange multipliers and optimality," *SIAM Rev.*, vol. 35, no. 2, pp. 183–238, 1993.

[2] I. Rechenberg, *Evolutionsstrategie; Optimierung technischer Systeme nach Prinzipien der biologischen Evolution.* Stuttgart–Bad Cannstatt: Frommann-Holzboog, 1973.

[3] N. Hansen, S. D. Müller, and P. Koumoutsakos, "Reducing the Time Complexity of the Derandomized Evolution Strategy with Covariance Matrix Adaptation (CMA-ES)," *Evolutionary Computation*, vol. 11, no. 1, pp. 1–18, 2003.

[4] N. Hansen and S. Kern, "Evaluating the CMA Evolution Strategy on Multimodal Test Functions," in *Lecture Notes in Computer Science*, ser. Parallel Problem Solving from Nature - PPSN VIII, vol. 3242. Berlin, Heidelberg: Springer, 2004, pp. 282–291.

[5] N. Hansen, "The CMA Evolution Strategy: A Tutorial," 2007.

[6] A. Auger and N. Hansen, "Performance Evaluation of an Advanced Local Search Evolutionary Algorithm," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2005)*, vol. 2, 2005, pp. 1777–1784.

[7] ——, "A Restart CMA Evolution Strategy with Increasing Population Size," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2005)*, vol. 2, 2005, pp. 1769–1776.

[8] N. Hansen, "Compilation of Results on the 2005 CEC Benchmark Function Set," Computational Laboratory (CoLab), Institute of Computational Science, ETH Zurich, Tech. Rep., 2006.

[9] J. P. K. Doye, M. A. Miller, and D. J. Wales, "The double-funnel energy landscape of the 38-atom Lennard-Jones cluster," *J. Chem. Phys.*, vol. 110, no. 14, pp. 6896–6906, Apr 1999.

[10] D. J. Wales, "Energy landscapes and properties of biomolecules," *Phys. Biol.*, vol. 2, no. 4, pp. S86–S93, Dec 2005.

[11] D. Wolpert and W. Macready, "No free lunch theorems for optimization," *"Evolutionary Computation, IEEE Transactions on"*, vol. 1, no. 1, pp. 67–82, Apr 1997.

[12] M. Lunacek and D. Whitley, "The Dispersion Metric and the CMA Evolution Strategy," in *GECCO '06: Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation*. New York, USA: ACM Press, 2006, pp. 477–484.

[13] J. Kennedy and R. Eberhart, "Particle swarm optimization," *Proc. IEEE, International Conference on Neural Networks*, vol. 4, pp. 1942–1948, 1995.

[14] A. M. Sutton, D. Whitley, M. Lunacek, and A. Howe, "PSO and multi-funnel landscapes: how cooperation might limit exploration," in *GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation*. New York, NY, USA: ACM, 2006, pp. 75–82.

[15] C.-T. Hsieh, C.-M. Chen, and Y.-P. Chen, "Particle Swarm Guided Evolution Strategy," in *Genetic and Evolutionary Computation Conference (GECCO '07)*, H. Lipson, Ed., London, England, 7-11 July, 2007, pp. 650–657.

[16] E. W. Weisstein. Hypercube line picking. From MathWorld – A Wolfram web resource. [Online]. Available: http://mathworld.wolfram.com/HypercubeLinePicking.html

[17] 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, Tech. Rep., May 2005.

[18] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed. Johns Hopkins University Press, 1996.

[19] G. Rudolph, "On Correlated Mutations in Evolution Strategies," in *Parallel Problem Solving from Nature 2 (Proc. 2nd Int. Conf. on Parallel Problem Solving from Nature, Brussels 1992)*, R. Männer and B. Manderick, Eds. Amsterdam: Elsevier, 1992, pp. 105–114.