Benchmarking Parameter-Free AMaLGaM on Functions With and Without Noise

Peter A. N. Bosman Peter.Bosman@cwi.nl
Centrum Wiskunde & Informatica, P.O. Box 94079, 1090 GB Amsterdam, The Netherlands

Jörn Grahl grahl@uni-mainz.de
Johannes Gutenberg University Mainz, Department of Information Systems & Business Administration, Jakob Welder-Weg 9, D-55128 Mainz, Germany

Dirk Thierens Dirk.Thierens@cs.uu.nl
Department of Information and Computing Sciences, Utrecht University, 3508 TB Utrecht, The Netherlands

doi:10.1162/EVCO_a_00094

Abstract

We describe a parameter-free estimation-of-distribution algorithm (EDA) called the adapted maximum-likelihood Gaussian model iterated density-estimation evolutionary algorithm (AMaLGaM-IDEA, or AMaLGaM for short) for numerical optimization. AMaLGaM is benchmarked within the 2009 black box optimization benchmarking (BBOB) framework and compared to a variant with incremental model building (iAMaLGaM). We study the implications of factorizing the covariance matrix in the Gaussian distribution, to use only a few or no covariances. Further, AMaLGaM and iAMaLGaM are also evaluated on the noisy BBOB problems and we assess how well multiple evaluations per solution can average out noise. Experimental evidence suggests that parameter-free AMaLGaM can solve a wide range of problems efficiently with perceived polynomial scalability, including multimodal problems, obtaining the best or near-best results among all algorithms tested in 2009 on functions such as the step-ellipsoid and Katsuuras, but failing to locate the optimum within the time limit on skew Rastrigin-Bueche separable and Lunacek bi-Rastrigin in higher dimensions. AMaLGaM is found to be more robust to noise than iAMaLGaM due to the larger required population size. Using few or no covariances hinders the EDA from dealing with rotations of the search space. Finally, the use of noise averaging is found to be less efficient than the direct application of the EDA unless the noise is uniformly distributed. AMaLGaM was among the best performing algorithms submitted to the BBOB workshop in 2009.

Keywords

Estimation-of-distribution algorithms, Gaussian distribution, maximum likelihood, adaptive computation, optimization, black box optimization, benchmarking.

1 Introduction

Estimation-of-distribution algorithms (EDAs) are an important strand of research on black box optimization (BBO). In BBO, either no prior knowledge on the function to be optimized is available, or the function cannot be expressed in closed form, for example, when solutions are evaluated using simulations. Still, the optimization problem can have exploitable structural features. One type of structure is dependencies between

Manuscript received: September 15, 2010; revised: September 6, 2011 and August 30, 2012; accepted: September 27, 2012.

problem variables. Identifying and using dependencies during optimization can lead to more efficient search. EDAs provide a principled way of doing this in a stochastic manner.

EDAs belong to the class of evolutionary algorithms (EAs) and as such maintain a population of solutions that are subject to selection and variation. The first population is usually generated randomly. Each solution's fitness is evaluated and the better solutions are selected for variation. Selection pushes the population into promising regions of the search space. New candidate solutions are generated by estimating a probability distribution from the selected solutions and randomly drawing new samples from this distribution. These newly generated solutions replace parts of the old population or the entire old population. The new population is evaluated and the better solutions are kept. This process is iterated until a convergence criterion is met.

EDAs mark an important contribution to BBO research. The main operator is the estimation (also called learning) of a probability distribution and resampling the probability distribution to generate offspring. This classifies the EDA as a model-based optimization technique. Probability distributions provide a principled way of modeling dependencies between problem variables, allowing EDAs to successfully exploit problem dependencies during the search. As a result, they are often more flexible and efficient at performing BBO than other EAs. Different EDAs have been proposed that use different probability distributions. For an overview, we refer the interested reader to the literature (Lozano et al., 2006; Pelikan et al., 2006).

The BBOB (black box optimization benchmarking) framework for real-valued optimization was introduced in 2009. BBOB is an extensive benchmark consisting of both an experimental setup (Hansen et al., 2009a) and carefully chosen functions (Finck et al., 2009; Hansen et al., 2009b). The reader should be familiar with the BBOB framework prior to reading the experimental results provided in this article.

In this article, we give a detailed description of an EDA for real-valued optimization, benchmark it within the BBOB framework, and discuss our findings. The EDA uses a Gaussian probability distribution and is known as the adapted maximum-likelihood Gaussian model iterated density-estimation evolutionary algorithm (Bosman et al., 2008; Bosman, 2009; AMaLGaM-IDEA, or AMaLGaM for short).

This article is organized as follows. In Section 2, we describe AMaLGaM and three variants using (1) a fully multivariate Gaussian distribution, (2) a Bayesian factorization of the Gaussian distribution, or (3) a univariately factorized Gaussian distribution. Section 3 explains how the Gaussian distribution can be learned incrementally over multiple generations rather than anew in each generation, the goal of which is to reduce the required population size. Section 4 describes our approach to removing all parameters and introduces a restart mechanism to improve the EDA's performance on multimodal problems. We provide pseudocode in Section 5 and present the results on the non-noisy and noisy functions in the BBOB framework in Section 6. The article ends with concluding remarks and an outlook on future research.

2 AMaLGaM

AMaLGaM represents the current state in a line of research on using Gaussian distributions in EDAs for numerical optimization. Some initial publications used maximum-likelihood (ML) parameter estimates (Bosman and Thierens, 2000; Larrañaga et al., 2000). Studies of the behaviour of such models revealed important shortcomings and led to the development of AMaLGaM. In the following, we review the basic principles of Gaussian distributions and the adaptations incorporated into AMaLGaM. For a more

in-depth discussion of the design of various components, we refer the interested reader elsewhere (Bosman and Thierens, 2000; Grahl et al., 2006; Bosman et al., 2007; Bosman et al., 2008; Bosman, 2009).

2.1 Gaussian Probability Distribution

We introduce a random variable X_i for each real-valued problem variable x_i, $i \in \{0, 1, \ldots, l-1\}$ where l is the problem dimensionality. Let $\boldsymbol{v}$ be a vector of $|\boldsymbol{v}|$ unique indices, $\boldsymbol{v}_i \in \{0, 1, \ldots, l-1\}$. Using $\boldsymbol{v}$ we can distinguish between the case of having a distribution over all variables (i.e., $\boldsymbol{v} = \{0, 1, \ldots, l-1\}$) and a distribution over a subset of all variables, which is important in the context of factorizations (see below). The Gaussian probability distribution $P^{\mathcal{N}}_{(\boldsymbol{\mu}_{\boldsymbol{v}}, \boldsymbol{\Sigma}^{\boldsymbol{v}})}(X_{\boldsymbol{v}})$ for a vector of random variables $X_{\boldsymbol{v}} = (X_{v_0}, X_{v_1}, \ldots, X_{v_{|\boldsymbol{v}|-1}})$ is parameterized by a vector $\boldsymbol{\mu}_{\boldsymbol{v}}$ of means and a symmetric covariance matrix $\boldsymbol{\Sigma}^{\boldsymbol{v}}$ comprising a total of $\frac{1}{2}|\boldsymbol{v}|^2 + \frac{3}{2}|\boldsymbol{v}|$ parameters to be estimated. It is defined as follows:

$$P^{\mathcal{N}}_{(\boldsymbol{\mu}_{\boldsymbol{v}}, \boldsymbol{\Sigma}^{\boldsymbol{v}})}(X_{\boldsymbol{v}} = \boldsymbol{x}) = \frac{(2\pi)^{-\frac{|\boldsymbol{v}|}{2}}}{(\det \boldsymbol{\Sigma}^{\boldsymbol{v}})^{\frac{1}{2}}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_{\boldsymbol{v}})^T (\boldsymbol{\Sigma}^{\boldsymbol{v}})^{-1} (\boldsymbol{x}-\boldsymbol{\mu}_{\boldsymbol{v}})}. \tag{1}$$

2.2 Maximum-Likelihood Estimates

ML parameter estimation is a common way of estimating parameters of probability distributions. It effectively minimizes the empirical error and maximizes the fit between the probability distribution and all given data (Vapnik, 1995). Let $\boldsymbol{\mathcal{S}}$ denote a vector of data that in AMaLGaM will correspond to the selected solutions. An ML estimation for parameters of the Gaussian probability distribution is obtained from $\boldsymbol{\mathcal{S}}$ if $\boldsymbol{\mu}_{\boldsymbol{v}}$ and $\boldsymbol{\Sigma}^{\boldsymbol{v}}$ are estimated by the sample average and sample covariance matrix, respectively (Anderson, 1958; Tatsuoka, 1971):

$$\hat{\boldsymbol{\mu}}_{\boldsymbol{v}} = \frac{1}{|\boldsymbol{\mathcal{S}}|} \sum_{j=0}^{|\boldsymbol{\mathcal{S}}|-1} (\boldsymbol{\mathcal{S}}_j)_{\boldsymbol{v}}, \quad \hat{\boldsymbol{\Sigma}}^{\boldsymbol{v}} = \frac{1}{|\boldsymbol{\mathcal{S}}|} \sum_{j=0}^{|\boldsymbol{\mathcal{S}}|-1} ((\boldsymbol{\mathcal{S}}_j)_{\boldsymbol{v}} - \hat{\boldsymbol{\mu}}_{\boldsymbol{v}})((\boldsymbol{\mathcal{S}}_j)_{\boldsymbol{v}} - \hat{\boldsymbol{\mu}}_{\boldsymbol{v}})^T. \tag{2}$$

2.3 Random Sampling

To draw new random samples for a single Gaussian-distributed random variable with a mean parameter of 0 and a variance parameter of 1, elementary algorithms exist (see, e.g., Knuth, 1981, Section 3.4.1, Algorithm C).

To draw random samples from a multivariate Gaussian $P^{\mathcal{N}}_{(\boldsymbol{\mu}_{\boldsymbol{v}}, \boldsymbol{\Sigma}^{\boldsymbol{v}})}$ the Cholesky decomposition of the covariance matrix can be computed $\boldsymbol{L}^{\boldsymbol{v}} \boldsymbol{L}^{\boldsymbol{v},*} = \boldsymbol{\Sigma}^{\boldsymbol{v}}$. Then, the vector $\boldsymbol{\mu} + \boldsymbol{L}^{\boldsymbol{v}}(z_0, z_1, \ldots, z_{|\boldsymbol{v}|-1})$, where z_i is a single-dimensional Gaussian random variable with mean 0 and variance 1, is a $|\boldsymbol{v}|$-dimensional random sample drawn from $P^{\mathcal{N}}_{(\boldsymbol{\mu}_{\boldsymbol{v}}, \boldsymbol{\Sigma}^{\boldsymbol{v}})}$.

2.4 Gaussian Factorization Selection

Initial EDAs that used the Gaussian distribution employed a univariate factorization (Rudlof and Köppen, 1996; Sebag and Ducoulombier, 1998), which corresponds to modeling no dependencies. Let $\boldsymbol{\mathcal{X}} = (X_0, X_1, \ldots, X_{l-1})$ be a vector containing all random variables. The univariate factorization is then defined as

$$P(\boldsymbol{\mathcal{X}}) = \prod_{i=0}^{l-1} P(X_i). \tag{3}$$

Estimating a univariate Gaussian factorization entails the estimation of l means and variances. Sampling a new solution is also straightforward and entails sampling the associated one-dimensional Gaussian distribution for each of the l variables.

To incorporate dependencies, the full covariance matrix can be used instead, that is, use Equation (1) with $X_{\boldsymbol{v}} = \boldsymbol{\mathcal{X}}$ and Equation (2) with $\boldsymbol{v} = \{0, 1, \ldots, l-1\}$. Note that the univariate factorization is equivalent to having a diagonal covariance matrix. Alternatively, it is possible to determine and use the most important dependencies only. To this end, one can estimate a Bayesian factorization. The vector of random variables indicated by $X_{\boldsymbol{\pi}_i}$ on which X_i is conditioned in the Bayesian factorization is called the vector of parents of X_i. A Bayesian factorization can now be written as

$$P(\boldsymbol{\mathcal{X}}) = \prod_{i=0}^{l-1} P(X_i | X_{\boldsymbol{\pi}_i}). \tag{4}$$

As described elsewhere (Pelikan et al., 1999; Mühlenbein and Mahnig, 1999; Bosman and Thierens, 2000; Larrañaga et al., 2000) greedy algorithms are the standard approach to compute Bayesian factorizations because this problem itself is NP-hard. The factorization expresses a subset of all dependencies between the variables. The density contours of a (factorized) Gaussian probability distribution are ellipsoids. Depending on the dependencies modeled by the factorization, these ellipsoids can be aligned along any axis. If there is no dependency between a set of random variables (i.e., the univariate factorization), the projected density contours in those dimensions are aligned along the main axes. In any case, a Gaussian distribution is only capable of modeling linear dependencies.

To estimate the conditional distributions $P^{\mathcal{N}}(X_i | X_{\boldsymbol{\pi}_i})$ when constructing Bayesian factorizations, let $\boldsymbol{W}^{\boldsymbol{v}}$ be the inverse of the symmetric covariance matrix for variables $X_{\boldsymbol{v}} = (X_{v_0}, X_{v_1}, \ldots, X_{v_{|\boldsymbol{v}|-1}})$, that is, $\boldsymbol{W}^{\boldsymbol{v}} = (\boldsymbol{\Sigma}^{\boldsymbol{v}})^{-1}$. Matrix $\boldsymbol{W}^{\boldsymbol{v}}$ is commonly called the precision matrix. An ML estimate of $P^{\mathcal{N}}(X_i | X_{\boldsymbol{\pi}_i})$ can be expressed in terms of Equation (2) (Bosman and Thierens, 2000):

$$\hat{P}^{\mathcal{N}}(X_i = x_i \mid X_{\boldsymbol{\pi}_i} = x_{\boldsymbol{\pi}_i}) = \frac{1}{(\breve{\sigma}_i \sqrt{2\pi})} e^{\frac{-(x_i - \breve{\mu}_i)^2}{2\breve{\sigma}_i^2}} \tag{5}$$

where $\breve{\sigma}_i = \frac{1}{\sqrt{\hat{\boldsymbol{W}}_{00}^{(i,\boldsymbol{\pi}_i)}}}$ and $\breve{\mu}_i = \frac{\hat{\boldsymbol{\mu}}_i \hat{\boldsymbol{W}}_{00}^{(i,\boldsymbol{\pi}_i)} - \sum_{j=0}^{|\boldsymbol{\pi}_i|-1} (x_{(\boldsymbol{\pi}_i)_j} - \hat{\boldsymbol{\mu}}_{(\boldsymbol{\pi}_i)_j}) \hat{\boldsymbol{W}}_{(j+1)0}^{(i,\boldsymbol{\pi}_i)}}{\hat{\boldsymbol{W}}_{00}^{(i,\boldsymbol{\pi}_i)}}$.

Equation (5) is a single-dimensional Gaussian distribution. Sampling from Bayesian factorizations is thus straightforward once all computations have been performed.

2.5 AVS

The smaller the variance, the smaller the area of exploration for the EDA. The variance in the Gaussian distribution is stored in the covariance matrix $\boldsymbol{\Sigma}$. Using ML estimates directly, the variance may decrease too rapidly for the EDA to have sufficient time for exploration. As a remedy, the variance can be scaled (adaptively) beyond its ML estimate (Ocenasek et al., 2004; Yuan and Gallagher, 2005; Grahl et al., 2006). The technique used in AMaLGaM is called adaptive variance scaling (AVS; Grahl et al., 2006).

A distribution multiplier $c^{\text{Multiplier}}$ is maintained and upon sampling new solutions, the covariance matrix used is $c^{\text{Multiplier}}\boldsymbol{\Sigma}$ instead of just $\boldsymbol{\Sigma}$. If the best fitness value improves in one generation, then the current size of the variance allows for progress. A further enlargement of the variance may allow further improvement in the next generation. To fight the variance-diminishing effect of selection, the size of $c^{\text{Multiplier}}$ is scaled by $\eta^{\text{INC}} \geq 1$. If, on the other hand, the best fitness does not improve, the range of exploration may be too large to be effective and the variance multiplier should be decreased by a factor of $\eta^{\text{DEC}} \in [0, 1]$. For symmetry, $\eta^{\text{DEC}} = 1/\eta^{\text{INC}}$.

The AVS technique is also used to detect termination. To this end, $c^{\text{Multiplier}} < 1$ is allowed if no improvement has been found in more than a predefined number of subsequent generations while $c^{\text{Multiplier}} = 1$. When this happens, the mean of the distribution is set to the best solution. This allows the EDA to focus on a single peak: the one to which the currently best solution belongs. If an improvement is found during this phase, $c^{\text{Multiplier}}$ is reset to 1. Termination is enforced if $c^{\text{Multiplier}} < 10^{-10}$.

It has previously been observed that $\eta^{\text{DEC}} = 0.9$ gives good results (Grahl et al., 2006). The number of subsequent generations before $c^{\text{Multiplier}} < 1$ is allowed, that is, the maximum no-improvement-stretch NIS^{MAX}, is set to $25 + l$, which was empirically observed to give good results. This number varies with l since convergence speed typically decreases for increasing l.

2.6 SDR

AVS increases $c^{\text{Multiplier}}$ when fitness values improve. However, improved fitness values do not always mean that the variance needs to be enlarged. This is the case when the Gaussian kernel is already near the optimum. Increasing the variance will then only slow down convergence, as the EDA is unnecessarily forced to explore a larger area of the search space. It is therefore sensible to attempt to separate two cases: traversing a slope, and searching around an optimum.

To this end, the standard deviation ratio (SDR) approach (Bosman et al., 2007) relates the distance of the improvements to the mean in parameter space. If improvements mostly take place far away from the mean, then AVS is used. If most of the improvements are obtained near the mean, then the EDA with ML parameters is focused around the optimum and variance enlargement is not required.

Let $\overline{\boldsymbol{x}}^{\text{avg-imp}}$ denote the average of all new samples that were an improvement over the set of selected solutions in the same generation. The distance between $\overline{\boldsymbol{x}}^{\text{avg-imp}}$ and the current mean is decomposed along the l principal axes of the Gaussian distribution. If along any of these l axes the distance exceeds the standard deviation more than a threshold $\theta^{\text{SDR}} \in [0, \infty]$ times, further enlargement of $c^{\text{Multiplier}}$ is triggered. Otherwise, the distribution multiplier remains unchanged. Note that this trigger is independent of the sample range and has a fixed, predefined notion of being "close" to the mean. From previous work, $\theta^{\text{SDR}} = 1$ is known to give good results (Bosman et al., 2007).

2.7 AMS

Estimating the distribution using the selected solutions of the current generation only, the density contours can become aligned with directions that contain only solutions of similar quality (Hansen, 2006; Yunpeng et al., 2007; Bosman et al., 2008). Methods that scale the covariance matrix, such as SDR-AVS, do not help much as they almost solely increase search effort in the direction perpendicular to the direction of improvement.

One remedy is to use the anticipated mean shift (AMS; Bosman et al., 2008). The AMS is computed as the difference between the means of subsequent generations, that is, $\hat{\boldsymbol{\mu}}^{\text{Shift}}(t) = \hat{\boldsymbol{\mu}}(t) - \hat{\boldsymbol{\mu}}(t-1)$. A part, specifically $\alpha^{\text{AMS}}100\%$, of the newly sampled solutions is then moved in the direction of the AMS, that is, $\boldsymbol{x} \leftarrow \boldsymbol{x} + \delta^{\text{AMS}}\hat{\boldsymbol{\mu}}^{\text{Shift}}(t)$. The rationale is that solutions changed by AMS are further down the slope (assuming minimization). Selecting those solutions as well as solutions not changed by AMS aligns the distribution better with the direction of improvement. In a population of size n (where $\lfloor \tau n \rfloor$ solutions are selected, n^{elitist} solutions are maintained and $n - n^{\text{elitist}}$ new solutions are generated), proportioning the selected solutions perfectly between unaltered and AMS-altered solutions requires that $\alpha^{\text{AMS}}(n - n^{\text{elitist}}) = \frac{1}{2}\tau n$ and thus

$\alpha^{\text{AMS}} = \frac{1}{2}\tau\frac{n}{n-n^{\text{elitist}}}$. From previous work, $n^{\text{elitist}} = 1$ and $\delta^{\text{AMS}} = 2$ are known to give good results (Bosman, 2009).

2.8 AMS-SDR-AVS = AMaLGaM

AMaLGaM combines AMS, SDR, and AVS. Traversing a slope is further sped up by multiplying the movement of solutions in the direction of the AMS by the same multiplier used for the covariance matrix, that is, $\boldsymbol{x} \leftarrow \boldsymbol{x} + c^{\text{Multiplier}}\delta^{\text{AMS}}\hat{\boldsymbol{\mu}}^{\text{Shift}}(t)$. The same multiplier is used as for enlarging the values in the covariance matrix, because the multiplier relates to finding improvements far away from the mean, which is indicative that larger movements in the direction of the AMS are beneficial.

3 iAMaLGaM

Distribution estimation in AMaLGaM is done from scratch each generation. However, subsequent generations have much in common and therefore the required population size can be reduced by incremental learning, that is, combining the distribution estimated from the currently selected solutions with the distribution used in the previous generation. In iAMaLGaM, a memory-decay approach is taken to this end.

Most of the parameters are stored in the covariance matrix $\boldsymbol{\Sigma}$. The covariance matrix is also the main reason why the population size has to grow faster with l than when only the variances are considered. More samples are required for the maximum-likelihood estimate of the covariance matrix to ensure that it is conditioned well.

A memory for the covariance matrix leads to a smaller population size requirement. Although a reduction in population size also reduces the reliability of the estimate for the mean, a memory for the mean is undesirable. If the mean needs to shift much before centering over a (local) optimum, a memory will slow this shift down. A memory for the AMS *is* desirable. With less reliable mean estimates, the AMS may oscillate around its optimal direction. A memory can smooth these oscillations out. The equations for incrementally estimating the covariance matrix and the AMS are

$$\begin{aligned}\hat{\boldsymbol{\Sigma}}(t) &= (1-\eta^{\boldsymbol{\Sigma}})\hat{\boldsymbol{\Sigma}}(t-1) + \eta^{\boldsymbol{\Sigma}}\frac{1}{|\boldsymbol{\mathcal{S}}|}\sum_{i=0}^{|\boldsymbol{\mathcal{S}}|-1}(\boldsymbol{\mathcal{S}}_i - \hat{\boldsymbol{\mu}}(t))(\boldsymbol{\mathcal{S}}_i - \hat{\boldsymbol{\mu}}(t))^T \\ \hat{\boldsymbol{\mu}}^{\text{Shift}}(t) &= (1-\eta^{\text{Shift}})\hat{\boldsymbol{\mu}}^{\text{Shift}}(t-1) + \eta^{\text{Shift}}(\hat{\boldsymbol{\mu}}(t) - \hat{\boldsymbol{\mu}}(t-1)).\end{aligned} \tag{6}$$

Values for the learning-rate parameters were determined empirically in previous work. Specifically, the function class for each η parameter is $1 - \exp(\alpha_0|\boldsymbol{\mathcal{S}}|^{\alpha_1}/l^{\alpha_2})$ where $|\boldsymbol{\mathcal{S}}|$ is the selection size and l the number of variables. The α_i parameters were determined by means of regression on the best results found on a set of test problems up to dimensionality $l = 40$ (Bosman, 2009). The resulting parameter settings are shown in Table 1.

4 Parameter-Free (i)AMaLGaM

EDAs and other EAs typically have many parameters. Removing them greatly eases their use although it may come at the cost of losing the optimal settings for specific problems. Typically, and this has also been the case for AMaLGaM (see, e.g., Bosman, 2009), guidelines are derived from testing the algorithm on a set of problems and taking the worst-case or average-case for parameter settings. In real-valued optimization, this set of problems typically consists of various forms of unimodal problems, most

Table 1: Univariate, Bayesian, and full parameter settings.

	Univariate		Bayesian		Full	
	$\eta^{\mathbf{\Sigma}}$	η^{Shift}	$\eta^{\mathbf{\Sigma}}$	η^{Shift}	$\eta^{\mathbf{\Sigma}}$	η^{Shift}
α_0	-0.40	-0.31	-0.33	-0.52	-1.1	-1.2
α_1	0.15	0.27	1.5	0.70	1.2	0.31
α_2	-0.034	0.067	1.1	0.65	1.6	0.50

Table 2: (i)AMaLGaM parameters.

Parameter	Part	Definition
τ	General	Selection percentile (select $\tau \cdot 100\%$ best solutions)
n	General	Population size
$\mathbf{\Sigma}$	Distribution	Covariance matrix of the Gaussian distribution
$\boldsymbol{\mu}$	Distribution	Mean vector of the Gaussian distribution
η^{INC}	AVS	Distribution-multiplier increase factor
η^{DEC}	AVS	Distribution-multiplier decrease factor
NIS^{MAX}	AVS	Threshold on subsequent generations without improvement before $c^{\text{Multiplier}} < 1$ is allowed
θ^{SDR}	SDR	Threshold on ratio of distance of improvements and standard deviation for triggering AVS
α^{AMS}	AMS	Percentile of generated solutions to apply AMS to
δ^{AMS}	AMS	Factor of mean shift for moving generated solutions
n^{elitist}	AMS/General	Number of elitist solutions (generate $n - n^{\text{elitist}}$ anew)
$\eta^{\mathbf{\Sigma}}$	Incremental	Learning rate for covariance matrix
η^{Shift}	Incremental	Learning rate for mean shift

of which are also included in BBOB, and typically includes the Rosenbrock function (which is bimodal) as well, which is also in BBOB. In this section, we summarize the guidelines that have been developed for all parameters in iAMaLGaM on the basis of which parameter-free versions can be designed. First, we summarize all parameters in iAMaLGaM, as shown in Table 2.

In AMaLGaM, the parameters $\mathbf{\Sigma}$ and $\boldsymbol{\mu}$ for the Gaussian distribution are estimated with maximum likelihood. In iAMaLGaM these estimations are updated using memory-decay with empirically determined functions for the learning rates $\eta^{\mathbf{\Sigma}}$ and η^{Shift} as described in Section 3. Guidelines for setting all parameters in AMS, SDR, and AVS are given above and are based on previous work. The parameters for estimation and variation are thereby defined. For the selection step, truncation selection with a truncation percentile of $\tau = 0.35$ (i.e., select the 35% best solutions) is known to give good results (Bosman, 2009).

What remains is an important parameter: the population size n. To overcome the burden of tuning its value, an EA can be restarted after convergence with an exponentially increasing population size (see e.g., Auger and Hansen, 2005). With a small population the EA is expected to terminate quickly, wasting only few evaluations if the optimum is not found. However, for real-valued optimization with a Gaussian EDA,

enlarging the population size may not always be the most effective approach, because a single Gaussian essentially represents a centered search. Therefore, increasing the population size does not necessarily mean that the probability of locating the global optimum far away from the mean of the Gaussian kernel significantly increases. The population size may have to be greatly increased before initialization favorably places solutions well inside the global optimum at better values than solutions at other local optima. Therefore, (i)AMaLGaM uses a different approach (Bosman, 2009).

The number of solutions is increased upon each restart by alternating between two approaches: a single run with a larger population and several parallel runs. To maximize the joint global effect of the parallel runs, they are started in separate regions obtained from clustering the search space. When increasing the number of parallel runs, the subpopulation size is also increased slightly so as to increase the robustness of the more localized searches. Let t^{Restart} denote the number of restarts so far (i.e., the first run has $t^{\text{Restart}} = 0$) and let m denote the number of parallel executions of subpopulation size n^{Sub} (i.e., $mn^{\text{Sub}} = n$). The restart mechanism in (i)AMaLGaM is:

- if t^{Restart} is even, then $n^{\text{Sub}} = \left(1 + \lfloor t^{\text{Restart}}/2 \rfloor\right) n^{\text{Base}}$, $m = 2^{\lfloor t^{\text{Restart}}/2 \rfloor}$;
- if t^{Restart} is odd, then $n^{\text{Sub}} = 2^{1+\lfloor t^{\text{Restart}}/2 \rfloor} n^{\text{Base}}$, $m = 1$.

Here, n^{Base} is a baseline population size. Previous work suggests the following values:

- Full covariance matrix

  AMaLGaM: $n^{\text{Base}} = 17 + 3l^{1.5}$, iAMaLGaM: $n^{\text{Base}} = 10l^{0.5}$
- Bayesian factorization

  AMaLGaM: $n^{\text{Base}} = 12 + 8l^{0.7}$, iAMaLGaM: $n^{\text{Base}} = 7l^{0.5}$
- Univariate factorization

  AMaLGaM: $n^{\text{Base}} = 10l^{0.5}$, iAMaLGaM: $n^{\text{Base}} = 4l^{0.5}$

5 Pseudocode

For technical completeness, pseudocode is presented for AMaLGaM itself (see Algorithm 1) as well as its parameter-free version (see Algorithm 2), which can be seen as a wrapper.

The external functions in the pseudocode differ depending on the type of distribution factorization or whether incremental learning is used. Function SetBasePopulationSize refers to the base population settings provided in Section 4. Function EstimateDistributionParameters refers to the ML estimates provided in Section 2.2 combined with the factorization of choice in Section 2.4 and the definition of the mean shift in Section 2.7. With incremental learning the covariance matrix and the mean shift to be used in the distribution are to be computed according to Equation (6) in Section 3. In that case, the proper settings for $\eta^{\mathbf{\Sigma}}$ and η^{Shift} are also required, which can be found in Section 3. Finally, computing the SDR (function ComputeSDR) requires computing the standard deviation ratio associated with the point of average improvement. For the univariately factorized case, the SDR is given by $\max_{0 \leq i \leq l-1}\{|(x_i^{\text{avg-imp}} - \hat{\mu}_i)/\sigma_i|\}$. For the Bayesian factorized case it is $\max_{0 \leq i \leq l-1}\{|(x_i^{\text{avg-imp}} - \breve{\mu}_i)/\breve{\sigma}_i|\}$. For the case of the full

Algorithm 1 Pseudocode for (i)AMaLGaM

(i)AMaLGaM
$c^{\text{Multiplier}} \leftarrow 1$
$n^{\text{AMS}} \leftarrow \alpha^{\text{AMS}}(n-1)$
NIS $\leftarrow 0$
$t \leftarrow 0$
do
  $\boldsymbol{\mathcal{S}} \leftarrow$ the best $\lfloor \tau n \rfloor$ solutions in $\boldsymbol{\mathcal{P}}$ (truncation selection)
  EstimateDistributionParameters()
  $\boldsymbol{\mathcal{P}}_0 \leftarrow$ the best solution in $\boldsymbol{\mathcal{S}}$
  for $i \leftarrow 1 \ldots n-1$ **do**
    $\boldsymbol{\mathcal{P}}_i \leftarrow$ DrawRandomSample()
  for n^{AMS} randomly chosen solutions $\boldsymbol{\mathcal{P}}_i$ $(1 \leq i \leq n-1)$ **do**
    $\boldsymbol{\mathcal{P}}_i \leftarrow \boldsymbol{\mathcal{P}}_i + c^{\text{Multiplier}} \delta^{\text{AMS}} \hat{\boldsymbol{\mu}}^{\text{Shift}}(t)$
  if any $\boldsymbol{\mathcal{P}}_i$ better than $\boldsymbol{\mathcal{P}}_0$ $(1 \leq i \leq n-1)$
  then
    NIS $\leftarrow 0$
    if $c^{\text{Multiplier}} < 1$ **then** $c^{\text{Multiplier}} \leftarrow 1$
    $\boldsymbol{x}^{\text{avg-imp}} \leftarrow$ average of all $\boldsymbol{\mathcal{P}}_i$ better than $\boldsymbol{\mathcal{P}}_0$ $(1 \leq i \leq n-1)$
    ComputeSDR()
    if SDR $> \theta^{\text{SDR}}$ **then** $c^{\text{Multiplier}} \leftarrow \eta^{\text{INC}} c^{\text{Multiplier}}$
  else
    if $c^{\text{Multiplier}} \leq 1$ **then** NIS $\leftarrow$ NIS $+ 1$
    if $(c^{\text{Multiplier}} > 1)$ **or** $(\text{NIS} \geq \text{NIS}^{\text{MAX}})$
    then $c^{\text{Multiplier}} \leftarrow \eta^{\text{DEC}} c^{\text{Multiplier}}$
    if $(c^{\text{Multiplier}} < 1)$ **and** $(\text{NIS} < \text{NIS}^{\text{MAX}})$ **then** $c^{\text{Multiplier}} \leftarrow 1$
  $t \leftarrow t + 1$
while optimum not found, maximum number of evaluations not reached and
    $c^{\text{Multiplier}} \geq 10^{-10}$

covariance matrix, let $\boldsymbol{L}\boldsymbol{L}^*$ be the Cholesky decomposition of $\hat{\boldsymbol{\Sigma}}$; then the SDR is given by $\max_{0 \leq i \leq l-1}\{|(\boldsymbol{L}^{-1}(\boldsymbol{x}^{\text{avg-imp}} - \hat{\boldsymbol{\mu}}))_i|\}$.

6 BBOB Results

6.1 Without Noise

BBOB experiments were performed with random inital solutions in $[-5, 5]^l$ and at most $10^6 l$ function evaluations. Here we describe the results on functions without noise.

6.1.1 AMaLGaM-Full

The results for the parameter-free version of AMaLGaM with a full covariance matrix are given in Table 3 and Figure 1.

Table 3 shows that compared to the best results obtained in the BBOB-2009 workshop, AMaLGaM's median performance over all benchmark functions becomes less favorable with increasing problem size. The scalability of AMaLGaM in terms of function evaluations is worse than the best known results. Figure 1 underlines this result. The difference in scalability is specifically clear for the Sphere function.

Algorithm 2 Pseudocode for the parameter-free version of (i)AMaLGaM

(i)AMaLGaM-Free
1 $n^{\text{Base}} \leftarrow$ SETBASEPOPULATIONSIZE()
2 $t^{\text{Restart}} \leftarrow 0$
3 **do**
4 **if** $(t^{\text{Restart}}$ **mod** $2) = 0$
5 **then**
6 $n^{\text{Sub}} \leftarrow (1 + \lfloor t^{\text{Restart}}/2 \rfloor)\, n^{\text{Base}}$
7 $m \leftarrow 2^{\lfloor t^{\text{Restart}}/2 \rfloor}$
8 **else**
9 $n^{\text{Sub}} \leftarrow 2^{1+\lfloor t^{\text{Restart}}/2 \rfloor} n^{\text{Base}}$
10 $m \leftarrow 1$
11 Run (i)AMaLGaM with population size n^{Sub} and m parallel runs, starting from the clustering of $n^{\text{Sub}}m$ randomly generated solutions into m clusters and using $\eta^{\text{DEC}} = 0.9, \eta^{\text{INC}} = 1/\eta^{\text{DEC}}, \theta^{\text{SDR}} = 1, \tau = 0.35, \alpha^{\text{AMS}} = \frac{1}{2}\tau(n^{\text{Sub}}/(n^{\text{Sub}} - 1)), \delta^{\text{AMS}} = 2$ and $\text{NIS}^{\text{MAX}} = 25 + l$
12 $t^{\text{Restart}} \leftarrow t^{\text{Restart}} + 1$
13 **while** optimum not found and maximum number of evaluations not reached

Table 3: ERT loss ratio (see Figure 1) compared to the respective best result from BBOB-2009 for budgets given in the first column. The last row $\text{RL}_{\text{US}}/\text{D}$ gives the number of function evaluations in unsuccessful runs divided by dimension. Shown are the smallest, 10th percentile, 25th percentile, 50th percentile (median), and 90th percentile values (smaller values are better).

#FEs/D	Best	10%	25%	Median	75%	90%
2	1.0	1.9	2.2	3.1	4.2	8.5
10	1.7	3.1	3.4	3.9	5.6	11
100	1.0	1.6	2.9	6.1	8.0	30
1e3	1.1	2.1	2.7	8.4	21	77
1e4	1.0	1.4	3.3	7.8	41	66
1e5	1.0	1.4	3.3	5.2	32	1.5e2
1e6	1.0	1.4	3.3	5.3	28	1.7e2
$\text{RL}_{\text{US}}/\text{D}$	1e6	1e6	1e6	1e6	1e6	1e6

Left: f_1–f_{24} in 5D, maxFE/D = 1.00e6.

#FEs/D	Best	10%	25%	Median	75%	90%
2	1.0	3.7	11	31	40	40
10	4.8	7.4	15	1.2e2	2.0e2	2.0e2
100	4.2	6.5	10	27	35	2.7e2
1e3	1.0	1.7	4.6	20	66	2.4e2
1e4	1.0	1.7	2.3	12	86	6.2e2
1e5	1.0	1.2	2.2	19	4.5e2	1.1e3
1e6	1.0	1.2	2.2	12	2.6e2	3.1e3
1e7	1.0	1.2	2.2	12	3.5e2	2.7e4
$\text{RL}_{\text{US}}/\text{D}$	1e6	1e6	1e6	1e6	1e6	1e6

Right: f_1–f_{24} in 20D, maxFE/D = 1.00e6.

On most benchmark functions up to $l = 40$, AMaLGaM identifies the optimum within the highest measured precision. Its perceived scalability is polynomial, even on multimodal problems. It is known from a related recent experimental study that includes some of the BBOB functions and a dimensionality up to $l = 200$, that the perceived scalability remains polynomial (Bosman, 2009), namely $\mathcal{O}(l^{1.91})$ for Sphere, $\mathcal{O}(l^{1.94})$ for Ellipsoid, $\mathcal{O}(l^{1.83})$ for Sharp ridge, and $\mathcal{O}(l^{2.42})$ for Rosenbrock. These results let us conclude that AMaLGaM's scalability is robust. It was found experimentally that on a set of linear and quadratic functions the perceived scalability of AMaLGaM with a full covariance matrix is approximately quadratic (Bosman, 2009).

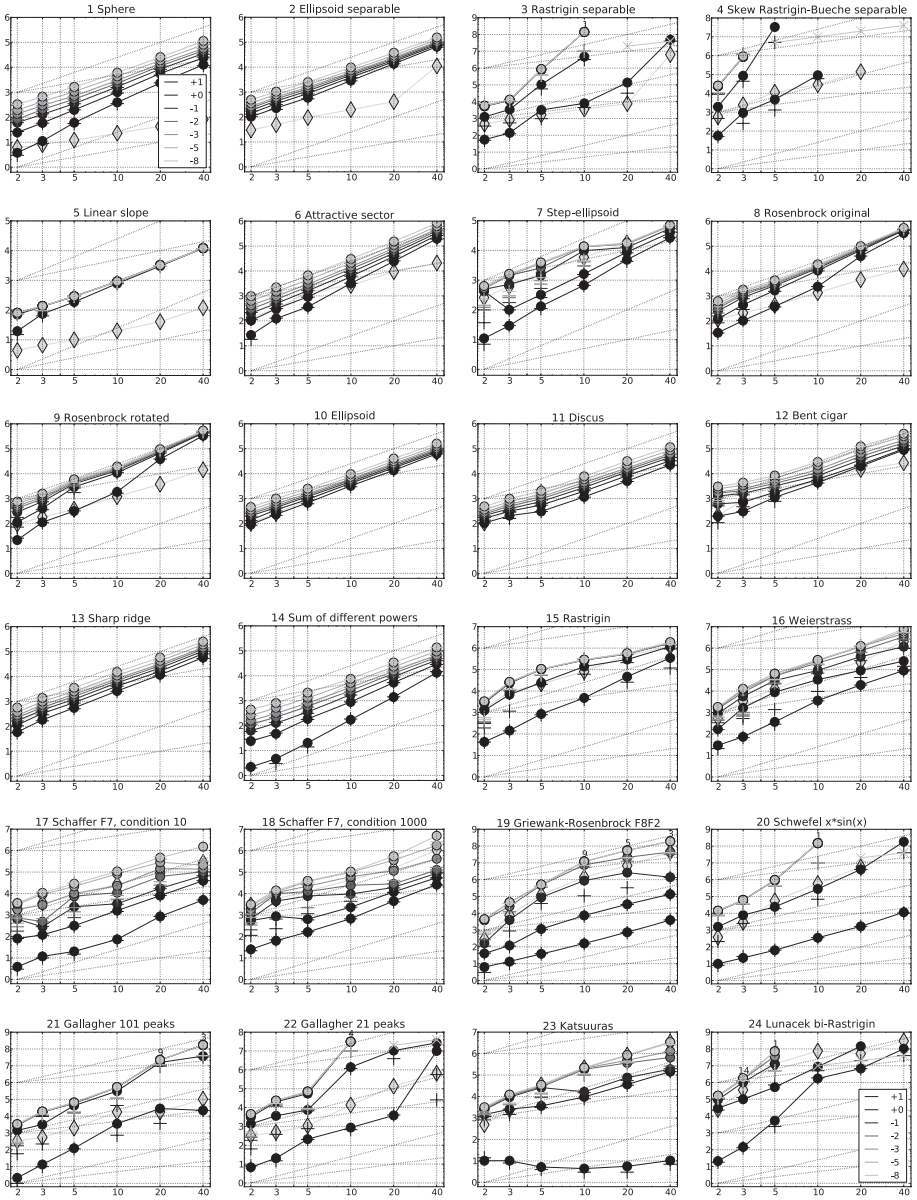

Figure 1: Expected running time (ERT, •) to reach $f_{\text{opt}} + \Delta f$ and median number of f-evaluations from successful trials (+), for $\Delta f = 10^{\{+1,0,-1,-2,-3,-5,-8\}}$ (the exponent is given in the legend of f_1 and f_{24}) versus dimension in log-log presentation. For each function and dimension, ERT(Δf) equals to #FEs(Δf) divided by the number of successful trials, where a trial is successful if $f_{\text{opt}} + \Delta f$ was surpassed. The #FEs(Δf) are the total number (sum) of f-evaluations while $f_{\text{opt}} + \Delta f$ was not surpassed in the trial, from all (successful and unsuccessful) trials, and f_{opt} is the optimal function value. Crosses (×) indicate the total number of f-evaluations, #FEs($-\infty$), divided by the number of trials. Numbers above ERT symbols indicate the number of successful trials. Y axis annotations are decimal logarithms. The thick light line with diamonds shows the single best results from BBOB-2009 for $\Delta f = 10^{-8}$. Additional grid lines show linear and quadratic scaling.

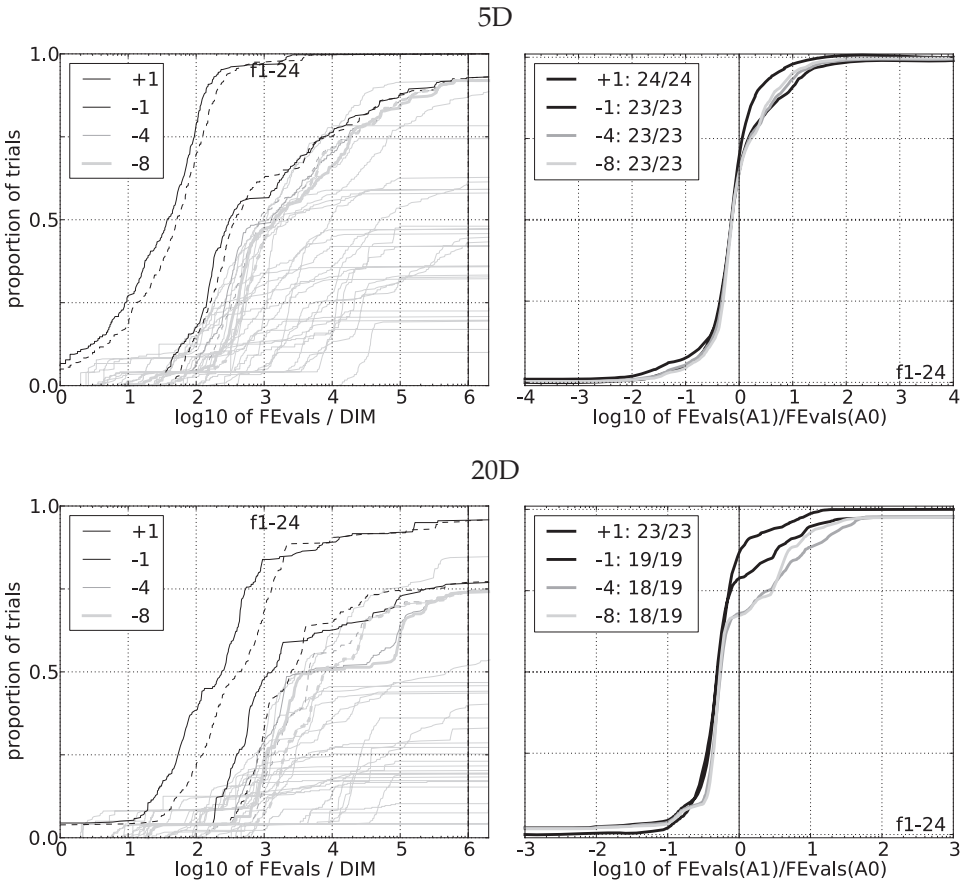

Figure 2: Empirical cumulative distributions (ECDF) of run lengths and speedup ratios in 5D (top) and 20D (bottom) over all functions. Left subcolumns: ECDF of the number of function evaluations divided by dimension D (FEvals/D) to reach a target value $f_{\text{opt}} + \Delta f$ with $\Delta f = 10^k$, where $k \in \{1, -1, -4, -8\}$ is given by the first value in the legend, for iAMaLGaM-Full (solid) and AMaLGaM-Full (dashed). Light lines show the ECDF of FEvals for target value $\Delta f = 10^{-8}$ of algorithms benchmarked during BBOB-2009. Right subcolumns: ECDF of FEval ratios of iAMaLGaM-Full divided by AMaLGaM-Full, all trial pairs for each function. Pairs where both trials failed are disregarded, pairs where one trial failed are visible in the limits being >0 or <1. The legends indicate the number of functions that were solved in at least one trial (iAMaLGaM-Full first).

AMaLGaM's scalability on other functions is not always approximately quadratic and $10^6 l$ evaluations are not always enough to obtain the optimum within the highest precision. The hardest problems for AMaLGaM are 4 and 24. However, the results for lower precision again suggest polynomial scalability. This suggests that AMaLGaM-Full is a robust algorithm able to overcome rotation and, using restarts, multimodality without losing polynomial scalability. Figure 2 shows the percentile of successfully tackled benchmark problems. AMaLGaM efficiently solves a wide range of problems and is among the most successful algorithms in the BBOB-2009 workshop.

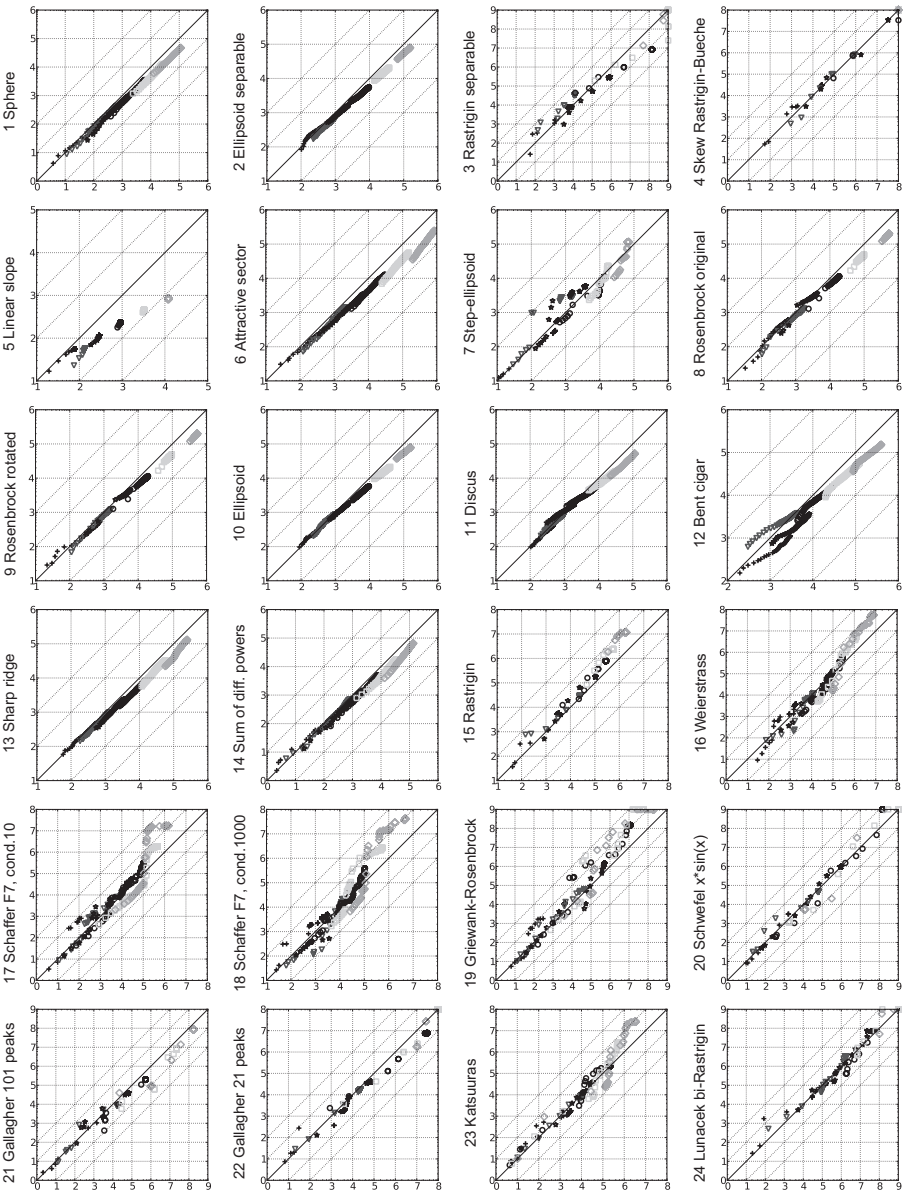

Figure 3: Expected running time (ERT in log10 of number of function evaluations) of iAMaLGaM-Full versus AMaLGaM-Full for 46 target values $\Delta f \in [10^{-8}, 10]$ in each dimension for functions f_1–f_{24}. Markers on the upper or right edge indicate that the target value was never reached by iAMaLGaM-Full or AMaLGaM-Full, respectively. Markers represent dimension: 2:+, 3:∇, 5:$\bigstar$, 10:$\circ$, 20:$\square$, 40:$\diamond$.

6.1.2 The Impact of Incremental Learning

We also tested AMaLGaM with incremental model building. Results for comparing non-incremental model building to incremental model building using a full covariance matrix (i.e., (i)AMaLGaM-Full) are shown in Figures 2 and 3. Further experiments were

performed to study incremental model building using factorized Gaussian distributions. The impact of incremental model building was found to be similar.

Figure 3 shows that incremental model building impacts the performance of AMaLGaM, but not uniformly positively or negatively. On unimodal problems such as Sphere, Ellipsoid, and Linear slope, results for iAMaLGaM are better and the difference increases with dimensionality and target quality. This is in agreement with the aforementioned study in which the perceived scalability of iAMaLGaM-Full remains polynomial, but slightly better than for AMaLGaM-Full (Bosman, 2009), namely $\mathcal{O}(l^{1.81})$ for Sphere, $\mathcal{O}(l^{1.86})$ for Ellipsoid, $\mathcal{O}(l^{1.72})$ for Sharp ridge, and $\mathcal{O}(l^{2.08})$ for Rosenbrock.

For multimodal problems, for example, Rastrigin, Weierstrass, and Griewank-Rosenbrock, nonincremental AMaLGaM performs better. This is most likely due to its larger base-population size. iAMaLGaM is sometimes unable to find the optimum within the predetermined limit of function evaluations because it starts from a smaller base population size. Hence, if memory resources are not very important, a larger base-population size helps in optimizing multimodal problems.

AMaLGaM and iAMaLGaM solve a similar percentile of problems, as shown in Figure 2. If nothing is known about the problem, that is, in a true BBO setting, unless larger population sizes are problematic, the nonincremental version may be preferred because when considering all problems it obtains slightly better results using fewer function evaluations.

6.1.3 The Impact of Factorization

We compared AMaLGaM-Full with AMaLGaM using a Bayesian factorization with few parents, or a univariate factorization (see Section 2.4). We allowed at most five parents per variable in the Bayesian factorization. The results comparing the full covariance matrix with the Bayesian factorization are shown in Figures 4 and 5. The results comparing the full covariance matrix with the univariate factorization are shown in Figures 6 and 7.

Without a full covariance matrix, not all pairwise dependencies can be processed and not all ellipsoid density contours can be aligned with an arbitrary direction. This can lead to an inefficient representation of promising areas of the search space and therefore to inefficient exploitation of the structure of the landscape. For low dimensional problems having $l \leq 5$, this problem is absent for the Bayesian factorization, since five parents are allowed. The cumulative results in Figure 5 show little difference between the two algorithms for $l = 5$. As the problem dimensionality is increased, the proportion of dependencies that can be expressed decreases, making the Bayesian-factorization approach increasingly similar to the univariate-factorization approach regarding performance. Whereas for $l = 5$ the Bayesian factorization is still capable of solving most problems, the univariate factorization can solve a smaller number of problems, as shown in Figure 6. The only significant difference between using a Bayesian factorization and a univariate factorization for higher-dimensional problems appears when the number of dependencies in the problem is not too high, which is the case for function 8, Rosenbrock. Here the use of the Bayesian factorization outperforms the use of the univariate factorization and even remains more efficient than the use of the full covariance matrix.

Computing a Bayesian factorization is time-consuming even using a greedy heuristic. Although less costly than working with a full covariance matrix, the expected added benefit quickly vanishes with an increase in problem dimensionality unless the number of dependencies between problem variables is known to be small. Computing Bayesian

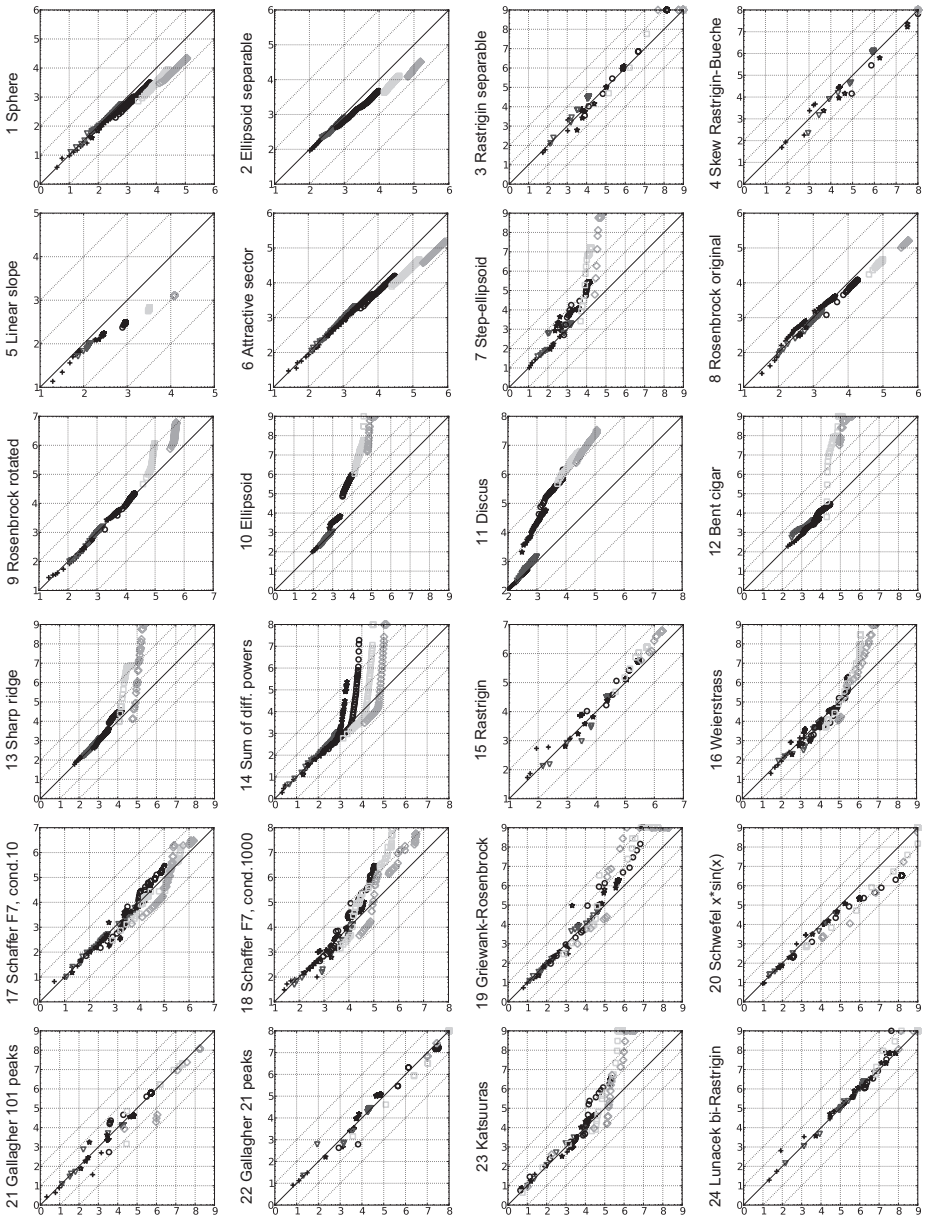

Figure 4: Expected running time (ERT in log10 of number of function evaluations) of AMaLGaM-Bayesian versus AMaLGaM-Full for 46 target values $\Delta f \in [10^{-8}, 10]$ in each dimension for functions f_1–f_{24}. Markers on the upper or right edge indicate that the target value was never reached by AMaLGaM-Bayesian or AMaLGaM-Full, respectively. Markers represent dimension: 2:+, 3:∇, 5:★, 10:○, 20:□, 40:◇.

factorizations for Gaussian distributions is thus arguably not worth the effort and one can focus on either using the full covariance matrix or using the univariate factorization. The results in Figure 7 show that for problems with no dependencies, the use of univariate factorization is more efficient. Polynomial scalability was also found to be favorable

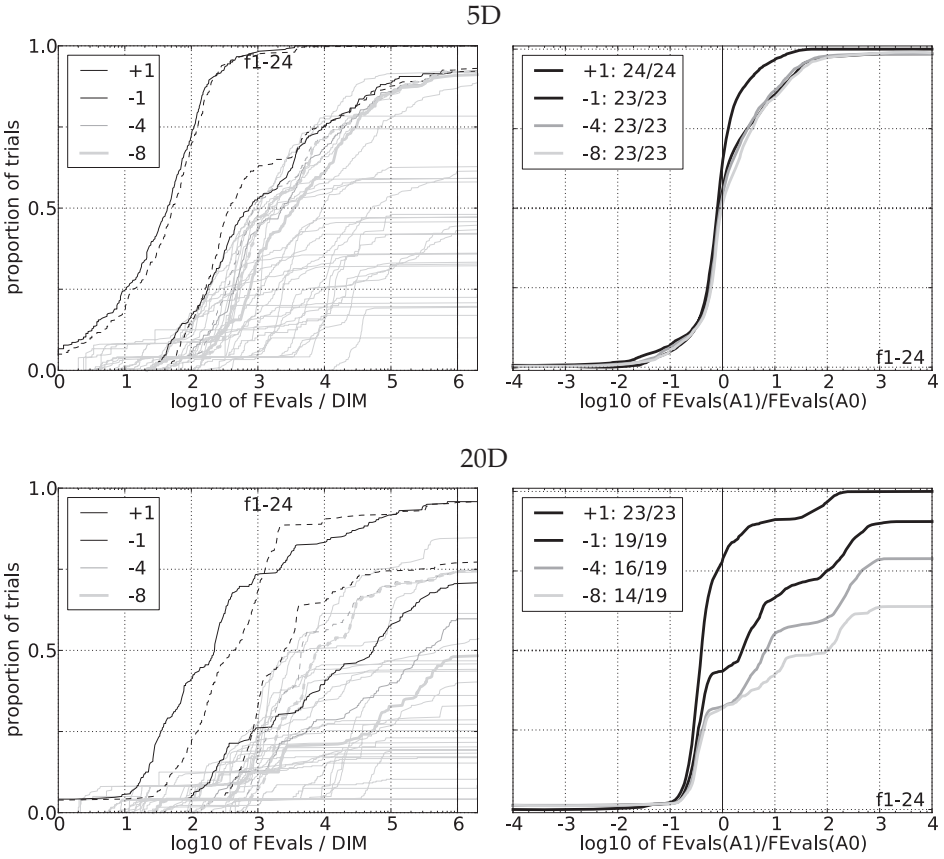

Figure 5: Empirical cumulative distributions (ECDF) of run lengths and speedup ratios in 5D (top) and 20D (bottom) over all functions. Left subcolumns: ECDF of the number of function evaluations divided by dimension D (FEvals/D) to reach a target value $f_{\text{opt}} + \Delta f$ with $\Delta f = 10^k$, where $k \in \{1, -1, -4, -8\}$ is given by the first value in the legend, for AMaLGaM-Bayesian (solid) and AMaLGaM-Full (dashed). Light lines show the ECDF of FEvals for target value $\Delta f = 10^{-8}$ of algorithms benchmarked during BBOB-2009. Right subcolumns: ECDF of FEval ratios of AMaLGaM-Bayesian divided by AMaLGaM-Full, all trial pairs for each function. Pairs where both trials failed are disregarded, pairs where one trial failed are visible in the limits being >0 or <1. The legends indicate the number of functions that were solved in at least one trial (AMaLGaM-Bayesian first).

(Bosman, 2009), namely $\mathcal{O}(l^{1.18})$ for Sphere, $\mathcal{O}(l^{1.18})$ for Ellipsoid separable, and $\mathcal{O}(l^{0.93})$ for Sharp ridge. The use of a univariate factorization is also much faster in terms of actual computation time. Polynomial scalability then is slightly above quadratic, compared to halfway between cubic and quartic for the full covariance matrix. This makes problem dimensionality and actual time for a function evaluation the most important discriminators in choosing the most appropriate EDA. Considering only function evaluations, the use of the full covariance matrix is superior.

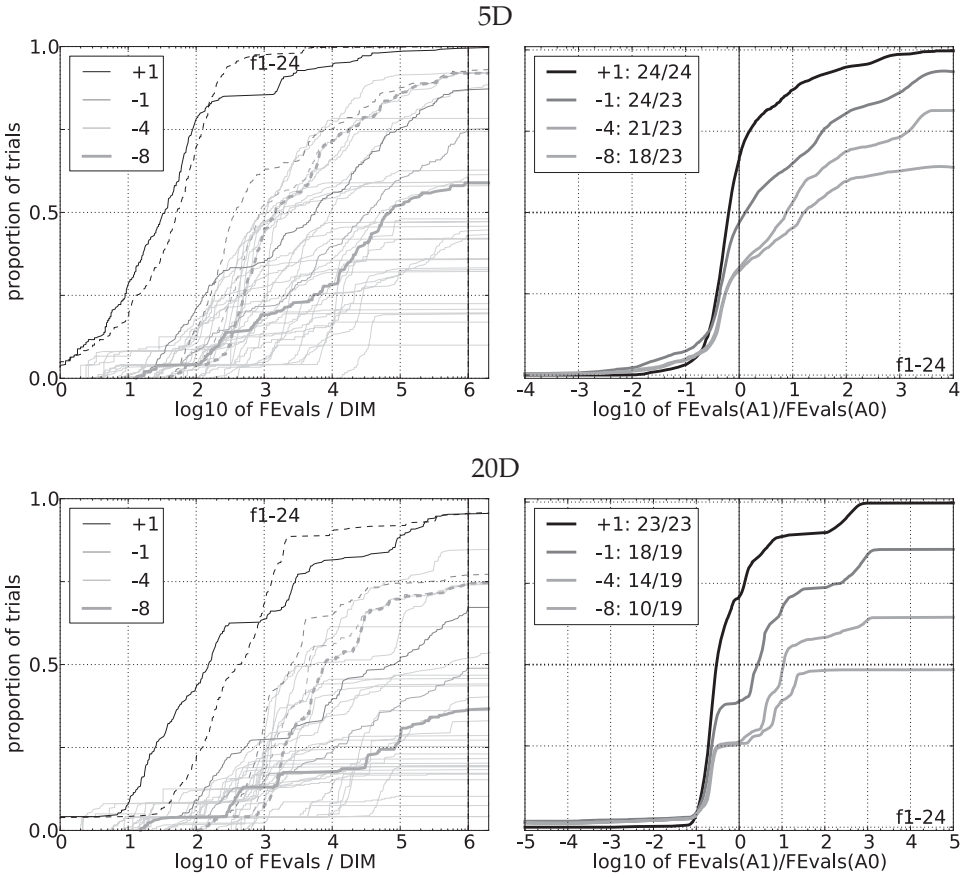

Figure 6: Empirical cumulative distributions (ECDF) of run lengths and speedup ratios in 5D (top) and 20D (bottom) over all functions. Left subcolumns: ECDF of the number of function evaluations divided by dimension D (FEvals/D) to reach a target value $f_{\text{opt}} + \Delta f$ with $\Delta f = 10^k$, where $k \in \{1, -1, -4, -8\}$ is given by the first value in the legend, for AMaLGaM-Univariate (solid) and AMaLGaM-Full (dashed). Light lines show the ECDF of FEvals for target value $\Delta f = 10^{-8}$ of algorithms benchmarked during BBOB-2009. Right subcolumns: ECDF of FEval ratios of AMaLGaM-Univariate divided by AMaLGaM-Full, all trial pairs for each function. Pairs where both trials failed are disregarded, pairs where one trial failed are visible in the limits being >0 or <1. The legends indicate the number of functions that were solved in at least one trial (AMaLGaM-Univariate first).

6.2 With Noise

6.2.1 AMaLGaM-Full

The results for running parameter-free AMaLGaM with a full covariance matrix and no adaptations to specifically tackle noise are given in Table 4 and Figure 8.

Table 4 shows that, similar to the case without noise, AMaLGaM compared to the best results from BBOB 2009 become less favorable with increasing dimensionality, although the decline appears to be smaller.

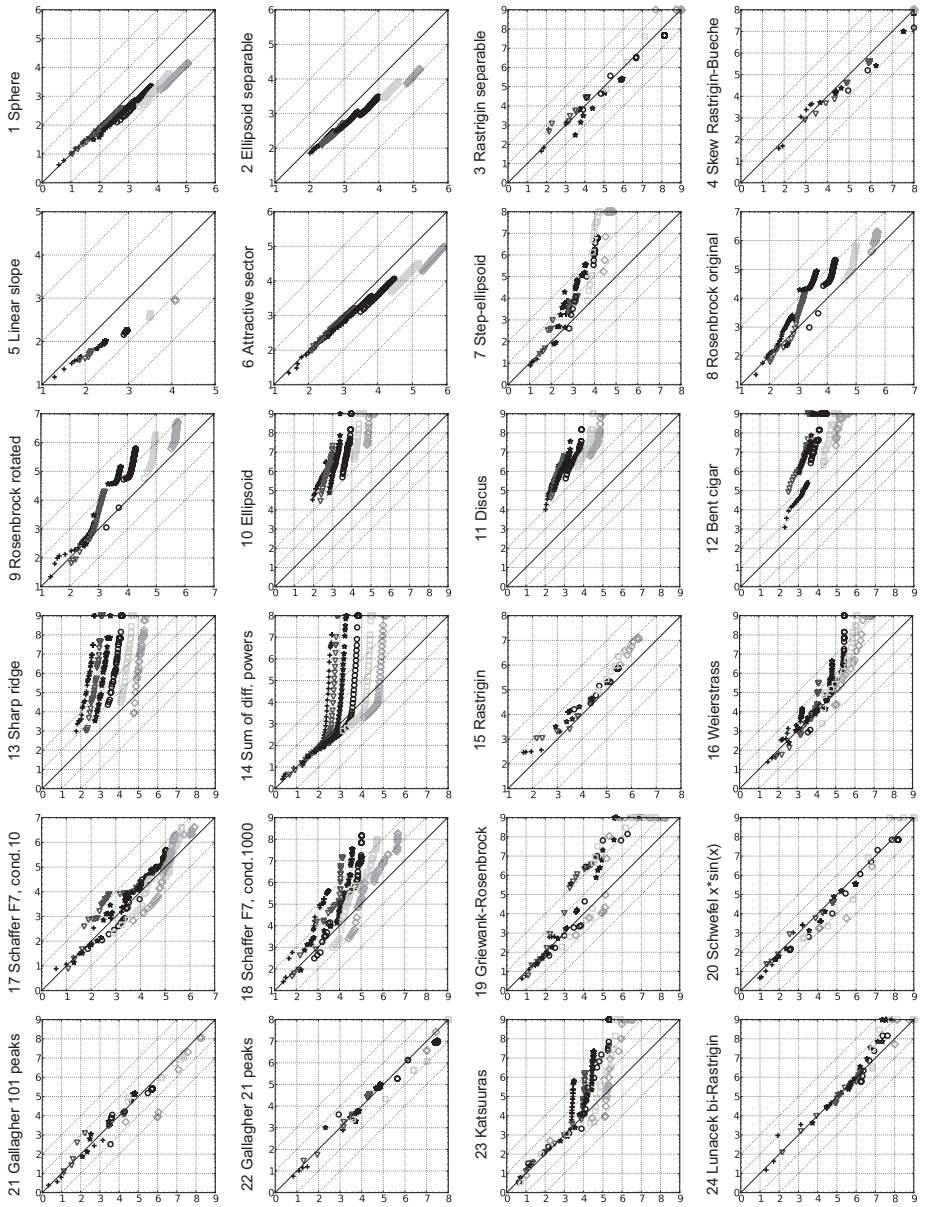

Figure 7: Expected running time (ERT in log10 of number of function evaluations) of AMaLGaM-Univariate versus AMaLGaM-Full for 46 target values $\Delta f \in [10^{-8}, 10]$ in each dimension for functions f_1–f_{24}. Markers on the upper or right edge indicate that the target value was never reached by AMaLGaM-Univariate or AMaLGaM-Full, respectively. Markers represent dimension: 2:+, 3:∇, 5:$\bigstar$, 10:$\circ$, 20:$\square$, 40:$\diamond$.

Figure 8 indicates that the perceived polynomial scalability is upheld even in the presence of noise, though at a higher degree. More evaluations are required and the optimum is not always found. AMaLGaM is robust and again among the best algorithms submitted to the BBOB-2009 workshop, as shown in Figure 9.

Table 4: ERT loss ratio (see Figure 1) compared to the respective best result from BBOB-2009 for budgets given in the first column. The last row RL_{US}/D gives the number of function evaluations in unsuccessful runs divided by dimension. Shown are the smallest, 10th percentile, 25th percentile, 50th percentile (median), and 90th percentile values (smaller values are better).

#FEs/D	Best	10%	25%	Median	75%	90%
2	1.0	1.2	1.6	2.1	5.6	10
10	1.1	1.4	1.8	3.4	5.0	28
100	1.0	1.0	2.1	5.5	17	2.8e2
1e3	1.0	4.2	8.7	18	49	2.6e3
1e4	1.0	1.4	7.2	18	30	2.5e4
1e5	1.0	1.4	4.0	16	51	2.4e2
1e6	1.0	1.4	8.2	18	80	2.4e2
RL_{US}/D	1e6	1e6	1e6	1e6	1e6	1e6

Left: f_{101}–f_{130} in 5D, maxFE/D = 1.00e6.

#FEs/D	Best	10%	25%	Median	75%	90%
2	1.0	1.0	3.8	23	40	40
10	1.0	2.4	10	1.3e2	2.0e2	2.0e2
100	1.0	1.0	1.4	8.1	28	1.0e3
1e3	1.0	1.0	2.3	13	32	2.0e4
1e4	1.0	2.3	5.5	14	56	1.0e5
1e5	1.0	1.2	5.5	19	1.1e2	1.0e6
1e6	1.0	1.1	5.5	23	1.3e2	7.1e2
1e7	1.0	1.1	4.2	29	1.3e2	1.3e3
RL_{US}/D	1e6	1e6	1e6	1e6	1e6	1e6

Right: f_{101}–f_{130} in 20D, maxFE/D = 1.00e6.

Problems with uniform noise are the hardest for AMaLGaM. Here its scalability declines strongly. The noise strength becomes more severe near the optimum in the case of uniform noise. This makes it harder for the algorithm to zoom in on the region of the best function values and an increasing number of additional generations is required to overcome decision errors. Note that Cauchy noise is applied sparingly, and not on every sample. This is much less problematic for AMaLGaM.

6.2.2 The Impact of Noise Averaging

EDAs, including AMaLGaM, are not specifically designed to handle noise. Function evaluations are assumed to be without noise so that selection always returns the better solutions. The lack of explicit noise modeling (in the distribution) is not necessarily a drawback of EDAs as arguably modeling noise can already be done in the selection phase, possibly including a memory for previous generations to get a more reliable ordering of solutions before selecting the better ones for probabilistic modeling.

Here we consider one of the simplest adaptations aimed at getting more robust results in the presence of noise: any evaluation of a solution is the sample average of multiple evaluations of the actual, noisy, function. This potentially smoothes out noise, but the question is whether such costly evaluation benefits the performance of AMaLGaM. We compare AMaLGaM with a full covariance matrix to the same EDA in which evaluations are averaged over 10 trials. The results are given in Figures 9 and 10. AMaLGaM with noise averaging is allowed 10 times more function evaluations. The resulting increase in computation time prohibited us from obtaining results for $l = 40$.

AMaLGaM is hardly influenced on functions with moderate noise; see Figure 10. The ERT for the algorithm with averaging lies on the first diagonal to the top of the main diagonal, that is, indicating a performance decrease of a factor 10, exactly the number of trials used for averaging. For moderate Cauchy noise, the algorithm that employs averaging actually gets worse. This can be seen for Cauchy noise in general. Since the sample average of a Cauchy distribution follows the same distribution as a single sample drawn from the same Cauchy distribution, we would expect the performance of AMaLGaM with averaging over 10 trials in the presence of Cauchy noise to be exactly

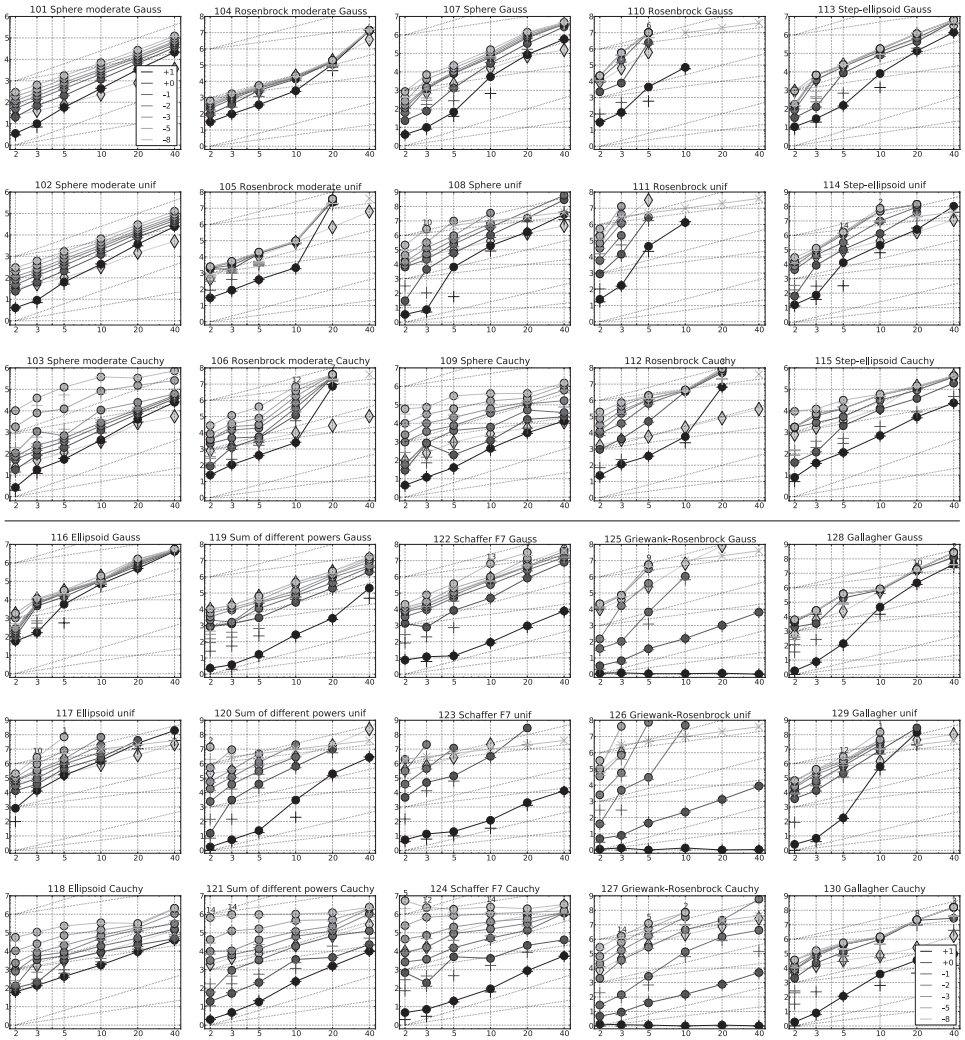

Figure 8: Expected running time (ERT, •) to reach $f_{\text{opt}} + \Delta f$ and median number of f-evaluations from successful trials (+), for $\Delta f = 10^{\{+1,0,-1,-2,-3,-5,-8\}}$ (the exponent is given in the legend of f_{101} and f_{130}) versus dimension in log-log presentation. For each function and dimension, ERT(Δf) equals to #FEs(Δf) divided by the number of successful trials, where a trial is successful if $f_{\text{opt}} + \Delta f$ was surpassed. The #FEs(Δf) are the total number (sum) of f-evaluations while $f_{\text{opt}} + \Delta f$ was not surpassed in the trial, from all (successful and unsuccessful) trials, and f_{opt} is the optimal function value. Crosses (×) indicate the total number of f-evaluations, #FEs($-\infty$), divided by the number of trials. Numbers above ERT symbols indicate the number of successful trials. Y axis annotations are decimal logarithms. The thick light line with diamonds shows the single best results from BBOB-2009 for $\Delta f = 10^{-8}$. Additional grid lines show linear and quadratic scaling.

10 times worse. However, Cauchy noise is applied seldom in the BBOB framework. When averaging over multiple trials, the number of times the evaluation procedure for a single solution contains Cauchy noise strongly increases. As a result, the search actually becomes more noisy with noise averaging.

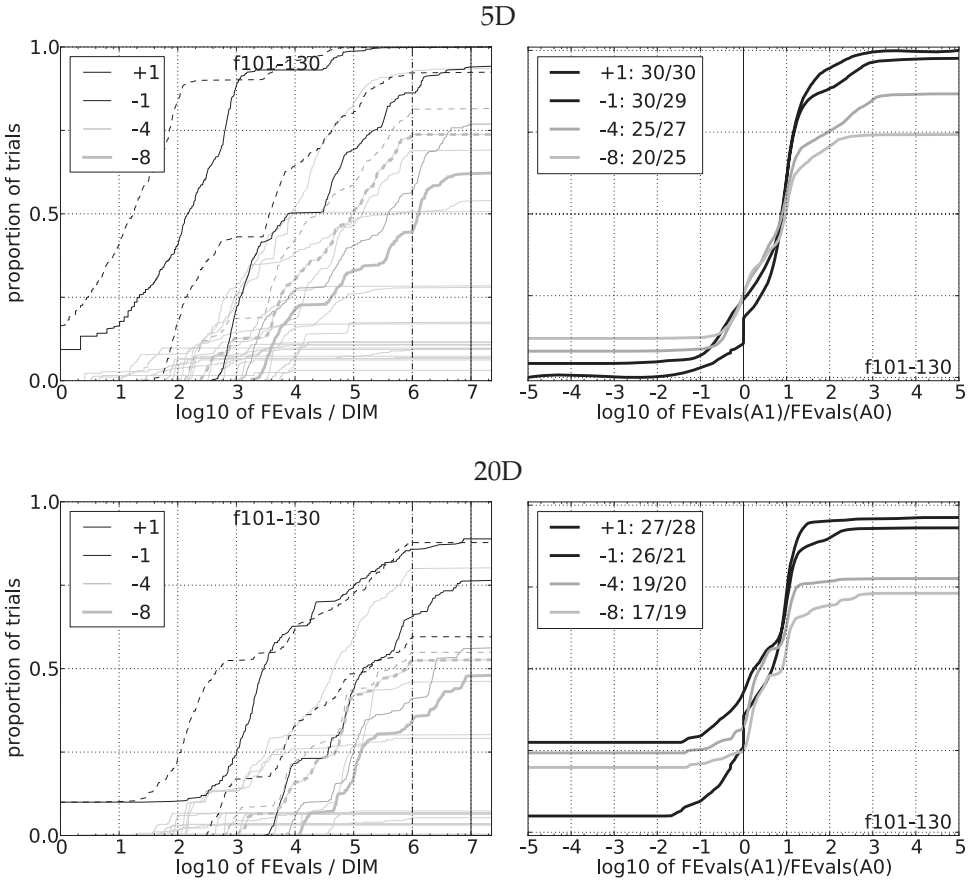

Figure 9: Empirical cumulative distributions (ECDF) of run lengths and speedup ratios in 5D (top) and 20D (bottom) over all functions. Left subcolumns: ECDF of the number of function evaluations divided by dimension D (FEvals/D) to reach a target value $f_{\text{opt}} + \Delta f$ with $\Delta f = 10^k$, where $k \in \{1, -1, -4, -8\}$ is given by the first value in the legend, for AMaLGaM-Full-avg10 (solid) and AMaLGaM-Full (dashed). Light lines show the ECDF of FEvals for target value $\Delta f = 10^{-8}$ of algorithms benchmarked during BBOB-2009. Right subcolumns: ECDF of FEval ratios of AMaLGaM-Full-avg10 divided by AMaLGaM-Full, all trial pairs for each function. Pairs where both trials failed are disregarded, pairs where one trial failed are visible in the limits being >0 or <1. The legends indicate the number of functions that were solved in at least one trial (AMaLGaM-Full-avg10 first).

For Gaussian noise, averaging over 10 trials and no averaging yield similar results. In some cases, such as Griewank-Rosenbrock, averaging even surpasses no averaging. For uniform noise, which in the case of the BBOB framework is more severe, there is almost always an improvement with the use of averaging, especially for higher dimensionality. The uniform distribution obeys the conditions of the central limit theorem and therefore the sample average, in the limit, follows a Gaussian distribution (with shrinking variance as more trials are used in the averaging). From the results on Gaussian noise we already know that AMaLGaM is capable of handling this type of noise and

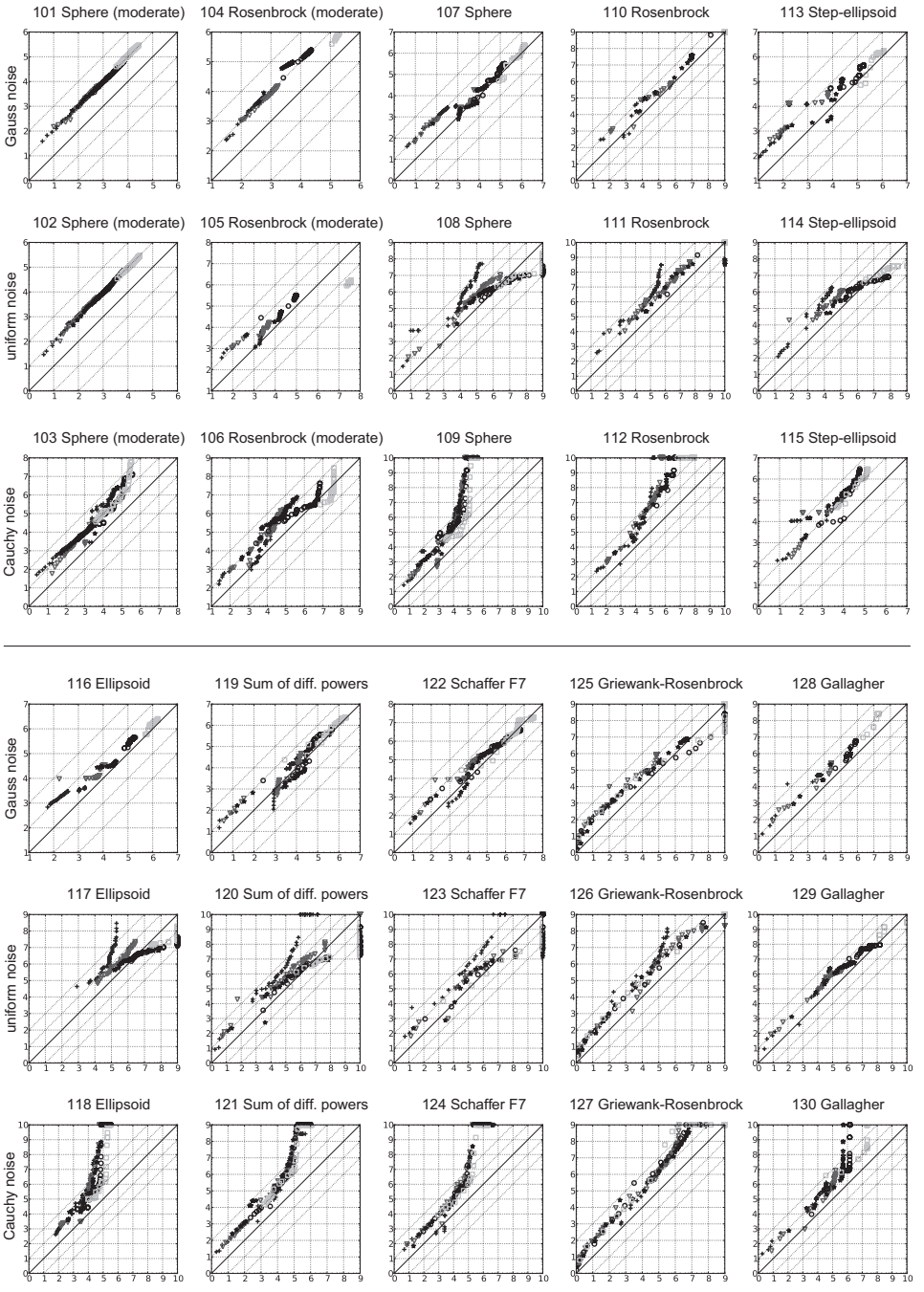

Figure 10: Expected running time (ERT in log10 of number of function evaluations) of AMaLGaM-Full-avg10 versus AMaLGaM-Full for 46 target values $\Delta f \in [10^{-8}, 10]$ in each dimension for functions f_{101}–f_{130}. Markers on the upper or right edge indicate that the target value was never reached by AMaLGaM-Full-avg10 or AMaLGaM-Full, respectively. Markers represent dimension: 2:+, 3:∇, 5:★, 10:○, 20:□, 40:◇.

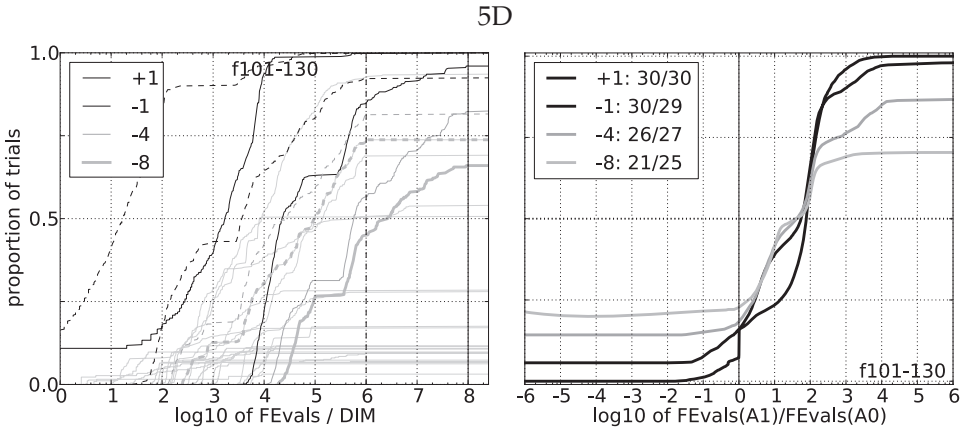

Figure 11: Empirical cumulative distributions (ECDF) of run lengths and speedup ratios in 5D (top) and 20D (bottom) over all functions. Left subcolumns: ECDF of the number of function evaluations divided by dimension D (FEvals/D) to reach a target value $f_{\text{opt}} + \Delta f$ with $\Delta f = 10^k$, where $k \in \{1, -1, -4, -8\}$ is given by the first value in the legend, for AMaLGaM-Full-avg100 (solid) and AMaLGaM-Full (dashed). Light lines show the ECDF of FEvals for target value $\Delta f = 10^{-8}$ of algorithms benchmarked during BBOB-2009. Right subcolumns: ECDF of FEval ratios of AMaLGaM-Full-avg100 divided by AMaLGaM-Full, all trial pairs for each function. Pairs where both trials failed are disregarded, pairs where one trial failed are visible in the limits being >0 or <1. The legends indicate the number of functions that were solved in at least one trial (AMaLGaM-Full-avg100 first).

that the impact of noise averaging becomes more substantial with increasing problem dimensionality. Hence, it is a matter of how severe the noise is. The larger the noise variance, the more need for noise averaging and the bigger the impact.

Regarding the entire benchmark, noise averaging does not improve overall competence, as shown in Figure 9. It slows the EDA down by almost the same factor as the number of trials used to average over, especially for large dimensionality. This is underlined by the results obtained using 100 trials, as shown in Figure 11. Given the number of required evaluations and the available time, experiments were performed up to $l = 10$. The gap between AMaLGaM with and without noise averaging increases to a factor of ≈100.

Tackling noise more efficiently can be important because many real-life problems are noisy. One issue is to detect the frequency of noise. Infrequent noise may lead to an increase in stochasticity as a result of averaging, which may be detected and used to reduce or disable noise averaging. In general, noise averaging was found only to be useful for uniform noise or for severe-enough Gaussian noise and for large-enough problem dimensionalities. A second issue that may be detected is whether the noise comes from a model that satisfies the conditions for the central limit theorem to hold and whether the variance for an evaluation decreases. We further suppose that noise can be addressed more effectively if it is explicitly modeled in EDAs. A separate model could be maintained which explicitly models uncertainty. This model could be used to guide selection and make fewer decision errors regarding which solution is better.

7 Conclusions

We have given an overview of a real-valued EDA called AMaLGaM that is based on the Gaussian distribution and benchmarked it on non-noisy and noisy functions within the BBOB framework. We tested variants in which the Gaussian distribution was factorized and in which incremental learning over multiple generations was used.

Experimental evidence suggests that AMaLGaM is a robust algorithm that solves the benchmark functions with similar polynomial scalability in terms of required number of function evaluations for similar (quadratic or linear) functions, regardless of rotations. Combined with a restart scheme that increases the population size and runs multiple instances of AMaLGaM in parallel, the algorithm is robust to noise and multimodality. In the presence of noise, AMaLGaM still appears to obtain polynomial scalability, albeit of higher degree. AMaLGaM was among the best performing algorithms submitted to the BBOB-2009 workshop.

Averaging evaluations over multiple trials to smooth out noise was found to make the EDA less efficient unless the problem is of high dimensionality and the noise satisfies the conditions of the central limit theorem. For future work, we have pointed out that explicitly modeling noise in EDAs may be beneficial.

References

Anderson, T. W. (1958). *An introduction to multivariate statistical analysis*. New York: John Wiley.

Auger, A., and Hansen, N. (2005). A restart CMA evolution strategy with increasing population size. In *Proceedings of the IEEE Congress on Evolutionary Computation, CEC-2005*, pp. 1769–1776.

Bosman, P. A. N. (2009). On empirical memory design, faster selection of Bayesian factorizations and parameter-free Gaussian EDAs. In *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO-2009*, pp. 389–396.

Bosman, P. A. N., Grahl, J., and Rothlauf, F. (2007). SDR: A better trigger for adaptive variance scaling in normal EDAs. In *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO-2007*, pp. 492–499.

Bosman, P. A. N., Grahl, J., and Thierens, D. (2008). Enhancing the performance of maximum-likelihood Gaussian EDAs using anticipated mean shift. In Rudolph, G. et al., editors, *Parallel Problem Solving from Nature—PPSN X*, pages 133–134. Berlin: Springer-Verlag.

Bosman, P. A. N., and Thierens, D. (2000). Expanding from discrete to continuous estimation of distribution algorithms: The IDEA. In *Parallel problem solving from nature. Lecture Notes in Computer Science*, Vol. 1917 (pp. 767–776). Berlin: Springer-Verlag.

Finck, S., Hansen, N., Ros, R., and Auger, A. (2009). Real-parameter black-box optimization benchmarking 2009: Presentation of the noiseless functions. Technical Report 2009/20, Research Center PPE.

Grahl, J., Bosman, P. A. N., and Rothlauf, F. (2006). The correlation-triggered adaptive variance scaling IDEA. In *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO-2006*, pp. 397–404.

Hansen, N. (2006). The CMA evolution strategy: A comparing review. In J. A. Lozano, P. Larrañaga, I. Inza, and E. Bengoetxea (Eds.), *Towards a new evolutionary computation. Advances in estimation of distribution algorithms* (pp. 75–102). Berlin: Springer-Verlag.

Hansen, N., Auger, A., Finck, S., and Ros, R. (2009a). Real-parameter black-box optimization benchmarking 2009: Experimental setup. Technical Report RR-6828, INRIA.

Hansen, N., Auger, A., Finck, S., and Ros, R. (2009b). Real-parameter black-box optimization benchmarking 2009: Noiseless functions definitions. Technical Report RR-6828, INRIA.

Knuth, D. E. (1981). *Semi-numerical algorithms*. *The art of computer programming*, Vol. 2. Reading, MA: Addison Wesley.

Larrañaga, P., Etxeberria, R., Lozano, J. A., and Peña, J. (2000). Optimization in continuous domains by learning and simulation of Gaussian networks. In *Proceedings of the Optimization by Building and Using Probabilistic Models OBUPM Workshop at the Genetic and Evolutionary Computation Conference, GECCO-2000*, pp. 201–204.

Lozano, J. A., Larrañaga, P., Inza, I., and Bengoetxea, E. (2006). *Towards a new evolutionary computation. Advances in estimation of distribution algorithms*. Berlin: Springer-Verlag.

Mühlenbein, H., and Mahnig, T. (1999). FDA—A scalable evolutionary algorithm for the optimization of additively decomposed functions. *Evolutionary Computation*, 7(4):353–376.

Ocenasek, J., Kern, S., Hansen, N., and Koumoutsakos, P. (2004). A mixed Bayesian optimization algorithm with variance adaptation. In *Parallel problem solving from nature. Lecture notes in computer science*, Vol. 3242 (pp. 352–361). Berlin: Springer-Verlag.

Pelikan, M., Goldberg, D. E., and Cantú-Paz, E. (1999). BOA: The Bayesian optimization algorithm. In *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO-1999*, pp. 525–532.

Pelikan, M., Sastry, K., and Cantú-Paz, E. (2006). *Scalable optimization via probabilistic modeling: From algorithms to applications*. Berlin: Springer-Verlag.

Rudlof, S., and Köppen, M. (1996). Stochastic hill climbing with learning by vectors of normal distributions. In T. Furuhashi (Ed.), *Proceedings of the First Online Workshop on Soft Computing, WSC1*, pp. 60–70.

Sebag, M., and Ducoulombier, A. (1998). Extending population-based incremental learning to continuous search spaces. In *Parallel problem solving from nature. Lecture notes in computer science*, Vol. 1498 (pp. 418–427). Berlin: Springer-Verlag.

Tatsuoka, M. M. (1971). *Multivariate analysis: Techniques for educational and psychological research*. New York: Wiley.

Vapnik, V. (1995). *The nature of statistical learning theory*. Berlin: Springer-Verlag.

Yuan, B., and Gallagher, M. (2005). On the importance of diversity maintenance in estimation of distribution algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO-2005*, pp. 719–726.

Yunpeng, C., Xiaomin, S., Hua, X., and Peifa, J. (2007). Cross entropy and adaptive variance scaling in continuous EDA. In *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO-2007*, pp. 609–616.