# Enhancing the Performance of Maximum–Likelihood Gaussian EDAs Using Anticipated Mean Shift

Peter A.N. Bosman[1], Jörn Grahl[2], and Dirk Thierens[3]

[1] Centre for Mathematics and Computer Science, P.O. Box 94079, 1090 GB Amsterdam, The Netherlands, *Peter.Bosman@cwi.nl*
[2] University of Mannheim, Mannheim, Germany, *joern.grahl@bwl.uni-mannheim.de*
[3] Department of Information and Computing Sciences, Utrecht University, Utrecht, The Netherlands, *Dirk.Thierens@cs.uu.nl*

**Abstract.** Many Estimation–of–Distribution Algorithms use maximum–likelihood (ML) estimates. For discrete variables this has met with great success. For continuous variables the use of ML estimates for the normal distribution does not directly lead to successful optimization in most landscapes. It was previously found that an important reason for this is the premature shrinking of the variance at an exponential rate. Remedies were subsequently successfully formulated (i.e. Adaptive Variance Scaling (AVS) and Standard–Deviation Ratio triggering (SDR)). Here we focus on a second source of inefficiency that is not removed by existing remedies. We then provide a simple, but effective technique called Anticipated Mean Shift (AMS) that removes this inefficiency.

## 1 Introduction

Estimation–of–Distribution Algorithm (EDAs) are a specific type of Evolutionary Algorithm (EA). EDAs are characterized by the way in which new solutions are generated. The information in all selected solutions is combined at once. To this end, an interim representation that compresses and summarizes this information is used: a probability distribution over the solution space. New solutions are generated by sampling the distribution.

Efficient optimization is guaranteed under suitable conditions [14]. In practice it is however impossible to meet these conditions because arbitrarily complex distributions are required. Hence, practical techniques are required. In this paper, we focus on optimization of numerical functions using continuous distributions. The use of the normal distribution or combinations thereof is the most commonly adopted choice. It has already been so since the first EDAs in continuous spaces were introduced [4, 11, 17, 18]. An important question is how efficient EDAs are in the continuous domain using such practical distributions.

Recently, it was shown that without precaution, premature convergence is likely to occur with these approaches, even on slope–like regions of the search space [7–9]. The main reason for this is that the variance decreases too fast at an exponential rate. The current state of the art exists of techniques that

attempt to remedy premature convergence (e.g. adaptive variance scaling [2, 8, 15]). Here we show that another source of inefficiency however exists that cannot be removed by these remedies. The use of ML estimates results in a distribution that describes the set of selected solutions well. On a slope however, it is not the set of selected solutions that is interesting, but it is the direction of descent. Efficient sampling along the direction of descent is therefore not guaranteed, even if the covariance matrix is scaled. We shall illustrate this problem further in this paper and present a remedy that we call AMS (Anticipated Mean Shift). The use of AMS improves performance, even if no covariances are estimated. Also, the resulting EDA still only uses ML estimates, which are a well–understood and sensible way of estimating parameters from data. We call the new EDA AMaLGaM–IDEA (Adapted Maximum–Likelihood Gaussian Model — Iterated Density–Estimation Evolutionary Algorithm) or just AMaLGaM for short. We compare the results of AMaLGaM with CMA–ES, currently the most efficient evolution strategy for continuous optimization.

## 2 Maximum–Likelihood Estimations, AVS and SDR

### 2.1 Maximum–Likelihood Estimations

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. The normal distribution $P^{\mathcal{N}}_{(\boldsymbol{\mu_v}, \boldsymbol{\Sigma^v})}(X_{\boldsymbol{v}})$ for a vector of random variables $X_{\boldsymbol{v}} = (X_{v_0}, X_{v_1}, \ldots, X_{v_{|\boldsymbol{v}|-1}})$ is parametrized by a vector $\boldsymbol{\mu_v}$ of means and a symmetric covariance matrix $\boldsymbol{\Sigma^v}$:

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

In an EDA, the distribution parameters are estimated from the vector of selected solutions $\boldsymbol{\mathcal{S}}$. Maximum–likelihood (ML) estimation is a principled and commonly–adopted approach. ML estimates for the mean and covariance matrix are given by the sample average and sample covariance matrix respectively:

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

To reduce the effort in learning the joint distribution, factorizations are commonly used. A factorization factors the joint distribution into a product of smaller joint (possibly conditional) distributions [12]. Learning effort is reduced the most using the well–known univariate factorization in which all variables are independent, i.e. the distribution is written as $\prod_{i=0}^{l-1} P(X_i)$. In the case of the normal distribution this means that all covariances are zero. Allowing for all possible dependencies implies use of the full covariance matrix. As an intermediate choice, a greedy algorithm can be used to determine and use only the most important dependencies. To this end, Bayesian factorizations are typically used in EDAs [13, 16]. To briefly recall Bayesian factorizations, recall that the vector of random variables indicated by $X_{\boldsymbol{\pi}_i}$ on which $X_i$ is conditioned is called the vector of parents of $X_i$ and that the distribution is written $\prod_{i=0}^{l-1} P(X_i | X_{\boldsymbol{\pi}_i})$. Let

$\boldsymbol{W^j}$ be the inverse of the symmetric covariance matrix, i.e. $\boldsymbol{W^j} = (\boldsymbol{\Sigma^j})^{-1}$. ML estimates of $P^{\mathcal{N}}(X_i|X_{\boldsymbol{\pi}_i})$ can be expressed in terms of Equation 2 [4]:

$$\hat{P}^{\mathcal{N}}(X_i{=}x_i \mid X_{\boldsymbol{\pi}_i}{=}x_{\boldsymbol{\pi}_i}) = \frac{1}{(\breve{\sigma}_i\sqrt{2\pi})} e^{\frac{-(x_i-\breve{\mu}_i)^2}{2\breve{\sigma}_i^2}} \tag{3}$$

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

Because Equation 3 has the form of a single–dimensional normal distribution, sampling from the Bayesian factorization is again straightforward once all relevant computations have been performed. Depending on the independencies expressed by the factorization, the density ellipsoids can be aligned with any axis. Use of the complete covariance matrix corresponds to a Bayesian factorization in which each $X_i$ is conditioned on all $X_j$ with $j > i$.

The full covariance matrix requires the most data to learn properly because all covariances need to be estimated. Although this argument advocates the univariate factorization, the use of it in an EDA brings about important limitations. The ellipsoid–shaped density contours can only be aligned with the main axes. This means that a function such as the Ellipsoid function ($\sum_{i=0}^{l-1} 10^{6\frac{i}{l-1}} x_i^2$) can be optimized efficiently. However, a rotated version of the same function introduces strong dependencies between the variables because each quadratic form is scaled differently. The contours of the function can no longer be matched by the contours of the univariately factorized normal distribution and optimization fails. Hence, a full covariance matrix is required to ensure rotation–invariance [10].

### 2.2 AVS

To remedy the problem of the prematurely vanishing variance, the variance can be scaled beyond its ML estimate [15]. One successful scheme for doing so is called adaptive variance scaling (AVS) [8]. This scheme allows the EDA to solve problems that it couldn't solve without scaling the variance.

In AVS, a variance multiplier $c^{\text{AVS}}$ is maintained. For sampling, $c^{\text{AVS}}\hat{\boldsymbol{\Sigma}}$ is used instead of $\hat{\boldsymbol{\Sigma}}$. If the best fitness value improves, then the current size of the variance allows for progress. Hence, a further enlargement of the variance may allow further improvement in the next generation. To fight the variance–diminishing effect of selection, $c^{\text{AVS}}$ is scaled by $\eta^{\text{INC}} > 1$. If there is no improvement, the exploration range may be too large and $c^{\text{AVS}}$. is decreased by a factor $\eta^{\text{DEC}} \in [0, 1]$. For symmetry, $\eta^{\text{DEC}} = 1/\eta^{\text{INC}}$. As the objective of the AVS scheme is to enlarge the variance to prevent premature convergence, $c^{\text{AVS}} \geq 1$ is enforced.

### 2.3 SDR

With AVS, improvements increase $c^{\text{AVS}}$. If the mean is already near the optimum, no further variance enlargement is necessary however. Let $\overline{\boldsymbol{x}^{\text{IMP}}}(t)$ denote the average of improvements in generation $t$. Further enlargement of $c^{\text{AVS}}$ in generation $t + 1$ is triggered whenever $\overline{\boldsymbol{x}^{\text{IMP}}}(t)$ lies further away from $\hat{\boldsymbol{\mu}}(t)$ than

a single standard deviation. To this end, the standard–deviation ratio (SDR) needs to be computed. The SDR is the ratio $a/b$ of the distance to the mean of $a$) $\overline{\boldsymbol{x}^{\mathrm{IMP},i}}(t)$ and $b$) the contour line of one standard deviation in the same direction. The SDR is independent of the sample range and has a fixed, predefined notion of being "close" to the mean [2].

## 3 Anticipated Mean Shift

### 3.1 Motivation

Most EDAs have been benchmarked using initialization ranges (IRs) centered around the optimum. An EDA based on the normal distribution with ML estimates focuses its search by contracting the region of exploration towards the mean. Hence, problems and the search bias of the EDA are favorably matched, leading to possibly overenthousiastic conclusions. This is already known to be the case for other contractive operators such as intermediate recombination [5]. Hence, it is important to specifically investigate the non–symmetric case.

A simple opposite of a symmetric function is the linear slope. Previous research focused on the one–dimensional case [2, 9]. Here, we consider two dimensions, i.e. $f(\boldsymbol{x}) = x_0 + x_1$. Use of the univariate factorization on this problem corresponds to the same situation of a single dimension studied earlier. We therefore focus on the case in which covariances are estimated also. The direction $\boldsymbol{u}$ of steepest descent obeys $u_0 = u_1$ and $u_i \leq 0$. Thus, it is most efficient to have the density ellipsoids parallel to and elongated along the line $x_0 = x_1$. Conversely, the worst alignment is parallel to and elongated along $x_0 = -x_1$.

Figure 1 shows the density contours in the case of the full covariance matrix for the first six subsequent generations. The density contours shown are the 95% error ellipses. When ML estimates are used only, the normal distribution quickly contracts. Initially, the population is spread uniformly in a square. On a two–dimensional slope the selected solutions form a triangle. Fitting a normal distribution with ML results in density contours aligned in the worst way. Scaling the covariance matrix almost solely increases search effort in the futile direction perpendicular to the best direction.

This effect was first noted in [19]. The same study proposed a first remedy. The remedy employs minimization of cross–entropy in which both the selected solutions and the population are used. Although the problem at hand was alleviated by this remedy, the resulting scaling behavior was reported in that same study to be inferior to AVS when symmetric initialization is used. Also, the well–known ML estimates can no longer be used. Here, we provide a simple, yet elegant and intuitive alternative way to overcome the inefficiency at hand that ultimately leads even to improvements over the use of SDR–AVS alone in the case of symmetric initialization.

### 3.2 Technique

The difference of the means in two subsequent generations indicates the direction in which the solutions are moved to obtain better fitness. Let $\hat{\boldsymbol{\mu}}^{\mathrm{Shift}}(t)$ denote for generation $t$ the mean shift for generations $t-1$ and $t$:

$$\hat{\boldsymbol{\mu}}^{\mathrm{Shift}}(t) = \hat{\boldsymbol{\mu}}(t) - \hat{\boldsymbol{\mu}}(t-1) \tag{4}$$

Note that our definition of mean shift differs from the one used in the mean–shift clustering algorithm that was studied in relation to EDAs elsewhere [6]. A straightforward anticipation of the mean shift that is required to obtain further improvements in generation $t+1$ is $\hat{\boldsymbol{\mu}}^{\text{Shift}}(t)$. It is therefore sensible to alter $100\alpha\%$ of all newly sampled solutions $\boldsymbol{x}$ in generation $t$ by moving them a certain fraction $\delta$ in the direction of the previously observed the mean shift, i.e.:

$$\boldsymbol{x} \leftarrow \boldsymbol{x} + \delta\hat{\boldsymbol{\mu}}^{\text{Shift}}(t) \tag{5}$$

We call this operation Anticipated Mean Shift (AMS).

When centered over an optimum, $\hat{\boldsymbol{\mu}}(t) \approx \hat{\boldsymbol{\mu}}(t-1)$ and therefore $\hat{\boldsymbol{\mu}}^{\text{Shift}}(t) \approx \mathbf{0}$, leaving the original approach unchanged. On a slope, AMS causes an important adjustment of $\hat{\boldsymbol{\Sigma}}$ that is estimated still using only ML. Solutions are selected from three sets: I) previously selected solutions (i.e. elitist solutions), II) new solutions *without* AMS and III) new solutions *with* AMS. Since set II is generated from a model that was estimated with ML from set I, these two sets share a similar region. Set III is further down the slope. If selection now selects solutions from both regions, the density contours are re–aligned, see Figure 1. Note that if the mean is nearing a peak and AMS overshoots the optimum, the mean shift in the next generation will be much smaller because the mean shift will be caused again mostly by the non–anticipated solutions. This thus resets the approach.

**Number of adaptations (setting $\alpha$)** We assume that the best $\tau n$ solutions are selected, where $n$ is the population size. Moreover, the selected solutions survive and $(1-\tau)n$ new solutions are generated to refill the population.

On a slope, all of the $\alpha(1-\tau)n$ altered solutions will be better and get selected. Now, if $\tau \geq \alpha$ only the altered solutions are selected, leaving the orientation of the density contours unchanged. For a change to occur, the selected solutions must consist of both unaltered and altered solutions. Ideally, these proportions are equally sized, which gives $\alpha(1-\tau)n = \frac{1}{2}\tau n$ and thus $\alpha = \frac{\tau}{2-2\tau}$. As using information about the anticipated mean shift is still only predictive, we want to alter no more than 50% of the newly sampled solutions, i.e. $\alpha \leq 0.5$. This restricts the selection percentile: $\alpha \leq 0.5 \quad \Leftrightarrow \quad \frac{\tau}{2-2\tau} \leq 0.5 \quad \Leftrightarrow \quad \tau \leq 0.5$.

**Adaptation length (setting $\delta$)** On a slope, set III in generation $t$ constitutes 50% of the selected solutions in generation $t+1$. The other 50% comes from sets I and II. The mean of the latter two sets is $\hat{\boldsymbol{\mu}}(t)$. The mean of set III is $\hat{\boldsymbol{\mu}}(t) + \delta\hat{\boldsymbol{\mu}}^{\text{Shift}}(t)$. For the suggested value of $\alpha$, the mean of the selected set in generation $t+1$ is[4] $\hat{\boldsymbol{\mu}}(t+1) = \frac{1}{2}\left(\hat{\boldsymbol{\mu}}(t) + \hat{\boldsymbol{\mu}}(t) + \delta\hat{\boldsymbol{\mu}}^{\text{Shift}}(t)\right) = \hat{\boldsymbol{\mu}}(t) + \frac{\delta}{2}\hat{\boldsymbol{\mu}}^{\text{Shift}}(t)$. The mean shift in generation $t+1$ is then $\hat{\boldsymbol{\mu}}^{\text{Shift}}(t+1) = \hat{\boldsymbol{\mu}}(t+1) - \hat{\boldsymbol{\mu}}(t) = \frac{\delta}{2}\hat{\boldsymbol{\mu}}^{\text{Shift}}(t)$. Hence, for any $\delta < 2$ the mean shift is expected to become smaller. Because the newly estimated mean falls in between the two sets, an ML estimate captures also the variance *between* the two sets. This causes the density to be aligned more favorably with the direction of descent. With repetition, the re–aligned density can result in a larger mean–shift. Hence a value of $\delta = 2$ suffices. The illustrations in Figure 1 were obtained using $\delta = 2$.

[4] Equality only holds for an infinite population size, it is an approximation otherwise.

## 4 Combining SDR, AVS and AMS: AMaLGaM

On a slope it makes sense to accelerate the search. The AVS scheme provides a principled way to achieve this. If improvements occur far away from the mean in subsequent generations, $c^{\text{AVS}}$ is enlarged. This relation between $c^{\text{AVS}}$ and improvements allows $c^{\text{AVS}}$ to be seen as a general accelerator. We therefore rename the variance multiplier $c^{\text{AVS}}$ to distribution multiplier $c^{\text{Multiplier}}$. Not only do we use $c^{\text{Multiplier}}\hat{\boldsymbol{\Sigma}}$ instead of $\hat{\boldsymbol{\Sigma}}$ upon sampling the distribution, we also use

$$\boldsymbol{x} \leftarrow \boldsymbol{x} + c^{\text{Multiplier}}\delta\hat{\boldsymbol{\mu}}^{\text{Shift}}(t) \tag{6}$$

upon applying AMS. This accelerates descent on a slope. In Figure 1 the effect of combining AVS with AMS can be seen when traversing the slope in two dimensions. The distribution gets rotated and elongated along the direction of improvement much faster than without the use of the distribution multiplier (note the difference in scale on both axes).

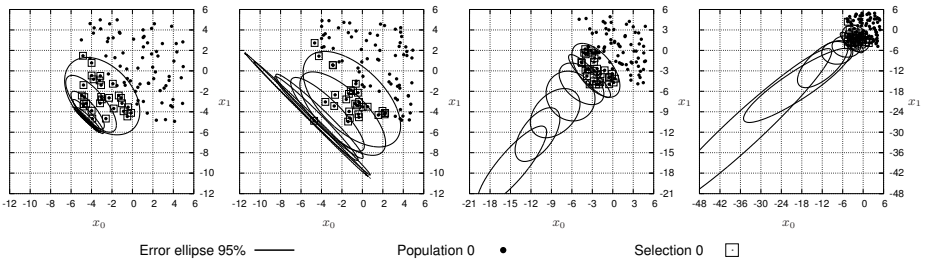



**Fig. 1.** Estimated normal distribution in the first 6 generations of typical runs with (from left to right): ML estimates, SDR–AVS, AMS and AMaLGaM on the two–dimensional slope $f(\boldsymbol{x}) = x_0 + x_1$ with IR $[-5;5]\times[-5;5]$. The density contours are the 95% error ellipses. Also shown are the population and selection in generation 0.

The combination of SDR, AVS and AMS adaptively changes both the covariance matrix and the mean–shift. It prevents premature convergence due to inefficient sampling that results from fitting only the set of selected solutions without considering the direction of descent. We name this composite AMS–SDR–AVS technique AMaLGaM (Adapted Maximum–Likelihood Gaussian Model). Pseudo–code may be found in a technical report [3].

## 5 Guidelines and comparison with CMA–ES

It is important to compare results with literature. It is equally important to have guidelines to use in subsequent applications and research. We therefore first derive guidelines and then use them to compare AMaLGaM with CMA–ES, currently the most efficient evolution strategy for continuous optimization.

### 5.1 Guidelines

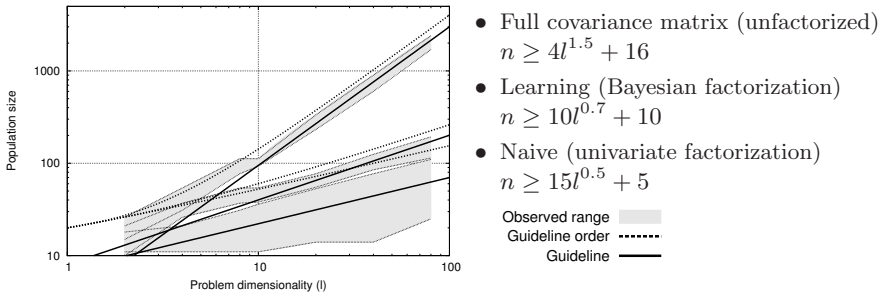
To derive guidelines, we use 10 benchmark functions to be minimized taken from literature [8, 10]. A function is considered to be optimized if the best solution has reached a certain value–to–reach (VTR). The VTR for all functions except the ridge functions is $10^{-10}$. For the two ridge functions the VTR is $-10^{10}$.

| Name | Definition |
|---|---|
| Sphere | $\sum_{i=0}^{l-1} x_i^2$ |
| Ellipsoid | $\sum_{i=0}^{l-1} 10^{6\frac{i}{l-1}} x_i^2$ |
| Cigar | $x_0^2 + \sum_{i=1}^{l-1} 10^6 x_i^2$ |
| Tablet | $10^6 x_1^2 + \sum_{i=1}^{l-1} x_i^2$ |
| Cigar Tablet | $x_0^2 + \sum_{i=1}^{l-2} 10^4 x_i^2 + 10^8 x_{l-1}^2$ |

| Name | Definition |
|---|---|
| Two Axes | $\sum_{i=0}^{\lfloor l/2 \rfloor - 1} 10^6 x_i^2 + \sum_{i=\lfloor l/2 \rfloor - 1}^{l-1} x_i^2$ |
| Different Powers | $\sum_{i=0}^{l-1} \lvert x_i \rvert^{2+10\frac{i}{l-1}}$ |
| Rosenbrock | $\sum_{i=0}^{l-2} \left(100 \cdot (x_i^2 - x_{i+1})^2 + (x_i - 1)^2\right)$ |
| Parabolic Ridge | $-x_1 + 100 \sum_{i=1}^{l-1} x_i^2$ |
| Sharp Ridge | $-x_1 + 100\sqrt{\sum_{i=1}^{l-1} x_i^2}$ |

We determined the optimal population size for AMaLGaM in the naive variant (i.e. univariate factorization), the learning variant (i.e. Bayesian factorization) and the full covariance matrix variant (i.e. unfactorized). For the full covariance matrix variant we used the functions as provided above as well as their rotated variants. With rotation each pair of variables in a solution is rotated 45 degrees before function evaluation takes place (for more details, see [3]).

IRs of $[-7.5; 7.5]$ (symmetric around optimum), $[-10, 5]$ (asymmetric) and $[-115, -100]$ (far–away) were used. We combined all scalability plots and determined on the basis thereof a guideline for the population size. For each variant, a minimal population size of 20 for $l = 1$ was determined. The guidelines and the combined scalability plots are presented in Figure 2.




- Full covariance matrix (unfactorized) $n \geq 4l^{1.5} + 16$
- Learning (Bayesian factorization) $n \geq 10l^{0.7} + 10$
- Naive (univariate factorization) $n \geq 15l^{0.5} + 5$

**Fig. 2.** Observed and guideline population size that leads to the minimum number of evaluations for AMaLGaM to reach the VTR, averaged over 100 independent runs. The gray areas are the observed population sizes for all problems.

Because AMaLGaM solves problems that can't be solved if only SDR–AVS or ML–estimates are used, no comparison is presented here with SDR–AVS or ML–estimates. It was found though, that AMaLGaM requires on average 0.67 times the evaluations of SDR–AVS. Hence, not only does AMaLGaM enlarge the class of problems that can be solved by the EDA, it also improves its efficiency. We also note that using the full covariance matrix no significant difference could be detected in solving rotated and unrotated versions of the problems. Hence, AMaLGaM can be said to be robust to rotations of the search space. More details on these additional results may be found in the technical report [3].

### 5.2 Comparisons

We used the guidelines defined in Section 5.1 and ran AMaLGaM 100 independent times on each of the benchmark problems. For the parameter settings of the CMA–ES, we used the guidelines provided in the literature also [10]. To prevent biased results from symmetric initialization, we used the far–away IR.

**Scalability** We computed a least–squares fit to $\alpha l^{\beta} + \gamma$ for the number of required evaluations. The fit was always found to be highly accurate. The results are summarized in Figure 3.

| Function | Algorithm | $\beta$ | $\alpha$ | $\gamma$ |
|---|---|---|---|---|
| Sphere | AMaLGaM–N | 1.23 | $2.74{\cdot}10^{2}$ | $9.46{\cdot}10^{0}$ |
| | AMaLGaM–L | 1.34 | $2.46{\cdot}10^{2}$ | $1.63{\cdot}10^{2}$ |
| | AMaLGaM–F | 2.05 | $1.09{\cdot}10^{2}$ | $4.08{\cdot}10^{2}$ |
| | CMA–ES | 0.94 | $2.38{\cdot}10^{2}$ | $3.17{\cdot}10^{2}$ |
| Ellipsoid | AMaLGaM–N | 1.24 | $3.33{\cdot}10^{2}$ | $8.20{\cdot}10^{0}$ |
| | AMaLGaM–L | 1.36 | $2.90{\cdot}10^{2}$ | $1.31{\cdot}10^{2}$ |
| | AMaLGaM–F | 2.09 | $1.14{\cdot}10^{2}$ | $4.87{\cdot}10^{2}$ |
| | CMA–ES | 1.92 | $6.40{\cdot}10^{1}$ | $1.79{\cdot}10^{3}$ |
| Cigar | AMaLGaM–N | 1.25 | $3.40{\cdot}10^{2}$ | $-2.30{\cdot}10^{1}$ |
| | AMaLGaM–L | 1.35 | $3.20{\cdot}10^{2}$ | $4.48{\cdot}10^{1}$ |
| | AMaLGaM–F | 2.08 | $1.30{\cdot}10^{2}$ | $4.14{\cdot}10^{2}$ |
| | CMA–ES | 0.90 | $7.18{\cdot}10^{2}$ | $-2.16{\cdot}10^{2}$ |
| Tablet | AMaLGaM–N | 1.22 | $2.95{\cdot}10^{2}$ | $1.12{\cdot}10^{2}$ |
| | AMaLGaM–L | 1.32 | $2.77{\cdot}10^{2}$ | $1.80{\cdot}10^{2}$ |
| | AMaLGaM–F | 2.04 | $1.13{\cdot}10^{2}$ | $5.24{\cdot}10^{2}$ |
| | CMA–ES | 1.64 | $1.17{\cdot}10^{2}$ | $1.59{\cdot}10^{3}$ |
| Cigar tablet | AMaLGaM–N | 1.22 | $3.54{\cdot}10^{2}$ | -4.14e-01 |
| | AMaLGaM–L | 1.34 | $3.21{\cdot}10^{2}$ | $8.52{\cdot}10^{1}$ |
| | AMaLGaM–F | 2.07 | $1.23{\cdot}10^{2}$ | $4.93{\cdot}10^{2}$ |
| | CMA–ES | 1.40 | $2.16{\cdot}10^{2}$ | $1.48{\cdot}10^{3}$ |

| Function | Algorithm | $\beta$ | $\alpha$ | $\gamma$ |
|---|---|---|---|---|
| Two axes | AMaLGaM–N | 1.27 | $3.06{\cdot}10^{2}$ | $4.62{\cdot}10^{1}$ |
| | AMaLGaM–L | 1.37 | $2.86{\cdot}10^{2}$ | $1.18{\cdot}10^{2}$ |
| | AMaLGaM–F | 2.10 | $1.11{\cdot}10^{2}$ | $5.08{\cdot}10^{2}$ |
| | CMA–ES | 2.00 | $7.91{\cdot}10^{1}$ | $1.68{\cdot}10^{3}$ |
| Different powers | AMaLGaM–N | 1.39 | $1.49{\cdot}10^{2}$ | $1.98{\cdot}10^{2}$ |
| | AMaLGaM–L | 1.41 | $1.70{\cdot}10^{2}$ | $1.94{\cdot}10^{2}$ |
| | AMaLGaM–F | 2.09 | $7.75{\cdot}10^{1}$ | $3.78{\cdot}10^{2}$ |
| | CMA–ES | 1.65 | $1.55{\cdot}10^{2}$ | $1.14{\cdot}10^{3}$ |
| Rosenbrock | AMaLGaM–N | 1.55 | $5.94{\cdot}10^{3}$ | $-7.59{\cdot}10^{3}$ |
| | AMaLGaM–L | 1.70 | $2.42{\cdot}10^{2}$ | $1.43{\cdot}10^{3}$ |
| | AMaLGaM–F | 2.57 | $5.58{\cdot}10^{1}$ | $2.35{\cdot}10^{3}$ |
| | CMA–ES | 1.92 | $7.25{\cdot}10^{1}$ | $2.52{\cdot}10^{3}$ |
| Parabolic ridge | AMaLGaM–N | 1.02 | $2.00{\cdot}10^{2}$ | $1.57{\cdot}10^{2}$ |
| | AMaLGaM–L | 1.13 | $2.75{\cdot}10^{2}$ | $1.14{\cdot}10^{2}$ |
| | AMaLGaM–F | 2.01 | $1.06{\cdot}10^{2}$ | $3.38{\cdot}10^{2}$ |
| | CMA–ES | 1.01 | $4.29{\cdot}10^{2}$ | $3.43{\cdot}10^{2}$ |
| Sharp ridge | AMaLGaM–N | 0.95 | $1.70{\cdot}10^{2}$ | $2.02{\cdot}10^{2}$ |
| | AMaLGaM–L | 1.08 | $1.57{\cdot}10^{2}$ | $2.20{\cdot}10^{2}$ |
| | AMaLGaM–F | 1.87 | $7.33{\cdot}10^{1}$ | $3.35{\cdot}10^{2}$ |
| | CMA–ES | 0.78 | $2.80{\cdot}10^{3}$ | $-9.00{\cdot}10^{3}$ |

**Fig. 3.** Scalability regression coefficients on all benchmark problems averaged over 100 independent runs using the guidelines. The IR is $[-115, -100]$ for each variable.

**Comparing naive, learning and full covariance matrix** The naive variant scales better than the Bayesian variant, which in turn scales better than the variant that uses the full covariance matrix. However, this only holds for functions that fit the model used. The naive method for instance cannot solve problems with many dependencies (e.g. rotated versions of the benchmark problems).

In additional experiments (for details, see the technical report [3]) it was found that the scalability of AMaLGaM does not change significantly when moving from asymmetric initialization to far–away initialization. This leads to the conclusion that AMaLGaM is also robust to translations, a property that earlier EDAs with ML estimates and even adaptive variance scaling do not have.

**Comparing AMaLGaM and CMA** The scalability of CMA–ES ranges between the different variants of AMaLGaM. For some functions (e.g. Sphere), CMA–ES has a better scalability than even the naive variant of AMaLGaM. For other functions (e.g. Two axes) it has a scalability similar to the full variant of AMaLGaM. The scalability results of AMaLGaM are less variable, causing CMA–ES to be better on some functions and AMaLGaM to be better on other functions. CMA–ES has the upper hand in the comparison, especially if rotation invariance is desired. This requires use of the full covariance matrix. AMaLGaM then however has a scalability that is at most similar (e.g. Two axes).

**Runtime** The number of required evaluations is important, especially if evaluations are time–consuming. The overall running time is however also important. With higher model complexity comes a larger learning and sampling time. Use of the full covariance matrix requires $\mathcal{O}(l^3)$ time. Assuming bounded complexity for the Bayesian network, the same asymptotic bound holds for the learning case

with the commonly used greedy algorithm [4, 16]. Hence, room for improvement exists to increase benefits from learning over using the full covariance matrix. Modelling time for the univariate factorization is only $\mathcal{O}(l)$. Detailed run–times per benchmark function and per algorithm are given in the technical report [3].

## 6 Summary, Discussion and Future Work

Using maximum–likelihood (ML) estimates for the normal distribution in an EDA, premature convergence is likely to occur. Optimization is only performed properly if the initialization range brackets the optimum. Optimization then mainly proceeds by contraction. Methods of adaptive variance scaling (AVS) provide a way to control the rate of contraction and turn it into expansion. Because ML estimates shape the density similar to the configuration of the selected solutions, the density contours can however be misaligned with the direction of descent. The variance then needs to be scaled to excessively large values to still make progress. We have proposed a simple, yet effective approach called anticipated mean shift (AMS) that removes this inefficiency. AMS advances sampled solutions in the direction of the mean shift of the previous generation. We analyzed this technique and provided rational settings for its parameters. We called the resulting EDA Adapted Maximum–Likelihood Gaussian Model — Iterated Density–Estimation Evolutionary Algorithm (AMaLGaM–IDEA or AMaLGaM for short). An experimental scalability analysis showed that AMaLGaM is robust to rotations and translations of the search space and is competitive with CMA–ES under certain conditions. AMaLGaM therefore makes an important step in the progression of continuous EDAs for numerical optimization.

Adaptivity in real–valued optimization has long been acknowledged to be important [1]. Its use in ES has led to the development of CMA–ES. Both AMaLGaM and CMA–ES adapt a Gaussian model using various techniques. The view upon the Gaussian model is different however. In CMA–ES directions are modelled and thus the Gaussian mainly serves as a mutation operator. In EDAs the region of interest is directly modelled and thus the Gaussian mainly serves as a recombination operator. The type of adaptation required is therefore different. It is important to research and take note of results along both lines.

The practical applicability of AMaLGaM and CMA–ES depends on the problem dimensionality. Using the full covariance matrix, only problems of relatively small dimensionality can be tackled due to the high required computing time. This leaves only methods that consider a few dependencies or no dependencies at all (i.e. the naive AMaLGaM). Certainly, if there are many strong dependencies in the problem, the algorithm can't find the optimum. Still, due to its simplicity, speed, and effectiveness the naive AMaLGaM can well serve as a baseline EDA to be used for future comparison and for applications with many variables.

One important direction of future work that we are currently pursuing is a reduction of the required population size. To ensure the full covariance matrix is well–conditioned for inversion, the required population size is quite large. This requires many samples in the generation–wise ML estimate. CMA–ES on the other hand convolutes the covariance matrix over multiple generations. This reduces the required population size and directly leads to less function evaluations.

## References


1. H.-G. Beyer and K. Deb. On self–adaptive features in real–parameter evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 5(3):250–270, 2001.
2. P. A. N. Bosman, J. Grahl, and F. Rothlauf. SDR: A better trigger for adaptive variance scaling in normal EDAs. In D. Thierens et al., editors, *Proc. of the Genetic and Evol. Comp. Conf. — GECCO–2007*, pages 492–499. ACM Press, 2007.
3. P. A. N. Bosman, J. Grahl, and D. Thierens. Adapted maximum–likelihood Gaussian models for numerical optimization with continuous EDAs. CWI technical report SEN–E0704, 2007.
4. P. A. N. Bosman and D. Thierens. Expanding from discrete to continuous estimation of distribution algorithms: The IDEA. In M. Schoenauer et al., editors, *Parallel Problem Solving from Nature — PPSN VI*, pages 767–776. Springer–Verlag, 2000.
5. D.-B. Fogel and H.-G. Beyer. A note on the empirical evaluation of intermediate recombination. *Evolutionary Computation*, 3(4):491–495, 1996.
6. M. Gallagher and M. Frean. Population–based continuous optimization, probabilistic modelling and mean shift. *Evolutionary Computation*, 13(1):29–42, 2005.
7. C. González, J. A. Lozano, and P. Larrañaga. Mathematical modelling of UMDAc algorithm with tournament selection. behaviour on linear and quadratic functions. *International Journal of Approximate Reasoning*, 31(3):313–340, 2002.
8. J. Grahl, P. A. N. Bosman, and F. Rothlauf. The correlation–triggered adaptive variance scaling IDEA. In M. Keijzer et al., editors, *Proc. of the Genetic and Evol. Comp. Conf. — GECCO–2006*, pages 397–404. ACM Press, 2006.
9. J. Grahl, S. Minner, and F. Rothlauf. Behaviour of UMDAc with truncation selection on monotonous functions. In D. Corne et al., editors, *Proceedings of the IEEE Congress on Evol. Comp. — CEC–2005*, pages 2553–2559. IEEE Press, 2005.
10. 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*, 11(1):1–18, 2003.
11. P. Larrañaga, R. Etxeberria, J. A. Lozano, and J. Peña. Optimization in continuous domains by learning and simulation of Gaussian networks. In M. Pelikan et al., editors, *Proc. of the OBUPM Workshop at the Genetic and Evol. Comp. Conf. — GECCO–2000*, pages 201–204. Morgan Kaufmann, 2000.
12. S. L. Lauritzen. *Graphical Models*. Clarendon Press, 1996.
13. J. A. Lozano, P. Larrañaga, I. Inza, and E. Bengoetxea. *Towards a New Evolutionary Computation. Advances in Estimation of Distribution Algorithms.* Springer–Verlag, 2006.
14. H. Mühlenbein and R. Höns. The estimation of distributions and the minimum relative entropy principle. *Evolutionary Computation*, 13(1):1–27, 2005.
15. J. Ocenasek, S. Kern, N. Hansen, S. Müller, and P. Koumoutsakos. A mixed Bayesian optimization algorithm with variance adaptation. In X. Yao et al., editors, *Par. Prob. Solv. from Nature — PPSN VIII*, pages 352–361. Springer–Verlag, 2004.
16. M. Pelikan, K. Sastry, and E. Cantú-Paz. *Scalable Optimization via Probabilistic Modeling: From Algorithms to Applications*. Springer–Verlag, 2006.
17. S. Rudlof and M. Köppen. Stochastic hill climbing with learning by vectors of normal distributions. In T. Furuhashi, editor, *Proceedings of the First Online Workshop on Soft Computing — WSC1*, pages 60–70. Nagoya Univ., 1996.
18. M. Sebag and A. Ducoulombier. Extending population-based incremental learning to continuous search spaces. In A. E. Eiben et al., editors, *Parallel Problem Solving from Nature — PPSN V*, pages 418–427. Springer–Verlag, 1998.
19. C. Yunpeng, S. Xiaomin, X. Hua, and J. Peifa. Cross entropy and adaptive variance scaling in continuous EDA. In D. Thierens et al., editors, *Proc. of the Genetic and Evol. Comp. Conf. — GECCO–2007*, pages 609–616. ACM Press, 2007.