Sampling-based optimization with mixtures

Rémi Bardenet
LAL, LRI (Computer Science Lab)
Paris-Sud XI University
91405 Orsay (France)
bardenet@lri.fr

Balázs Kégl
LAL, LRI (Computer Science Lab)
Paris-Sud XI University & CNRS
91405 Orsay (France)
balazs.kegl@gmail.com

Abstract

Sampling-based Evolutionary Algorithms (EA) are of great use when dealing with a highly non-convex and/or noisy optimization task, which is the kind of task we often have to solve in Machine Learning. Two derivative-free examples of such methods are Estimation of Distribution Algorithms (EDA) and techniques based on the Cross-Entropy Method (CEM). One of the main problems these algorithms have to solve is finding a good surrogate model for the normalized target function, that is, a model which has sufficient complexity to fit this target function, but which keeps the computations simple enough. Gaussian mixture models have been applied in practice with great success, but most of these approaches lacked a solid theoretical founding. In this paper we describe a sound mathematical justification for Gaussian mixture surrogate models, more precisely we propose a proper derivation of an EDA/CEM algorithm with mixture updates using Expectation Maximization techniques. It will appear that this algorithm resembles the recent Population MCMC schemes, thus reinforcing the link between Monte-Carlo integration methods and sampling-based optimization. We will concentrate throughout this paper on continuous optimization.

1 A quick review on existing techniques

1.1 Estimation of Distribution Algorithms

EDAs are optimization algorithms which match the paradigm of evolutionary strategies, summarized in the following repeated steps: generation, selection, reproduction. The particularity of EDAs resides in the fact that the generation and reproduction steps at time t are related to an underlying instrumental probability distribution q_t: the generation sample is drawn according to q_t and the reproduction is merely an update of q_t to make it approximate some hidden target distribution. The example of the Estimation of Multivariate Normal Algorithm (EMNA) is given Figure 1.

EMNA(S, N, M)

1. Initialize $\mu^{(0)}$,$\Sigma^{(0)}$, $q_0 := \mathcal{N}(\mu^{(0)}, \Sigma^{(0)})$, $t := 1$.
2. Sample $x_1, ..., x_N \sim q_{t-1}$ i.i.d.
3. Take the M points with best fitness, and compute their sample mean μ and covariance Σ.
4. Set the new mean $\mu^{(t)}$ and covariance $\Sigma^{(t)}$ for the Gaussian proposal q_t to μ and Σ.
5. Stop if the termination criterion is met, otherwise set $t := t + 1$ and go back to step 2.

Figure 1: The EMNA algorithm: the surrogate model for the fitness function is a single Gaussian.

A mixture version of EMNA called EMDA (Estimation of Mixture of Distributions Algorithm) already was mentioned in [1]: it consists of seeing the distribution q as a mixture of conditionally

Gaussian networks, updating it with a Bayesian-score EM step. The algorithm we derive in Section 2 can be seen as an instance of EMDA. It has the advantage though that the selection step is theoretically justified by a ghost integration goal which is at the core of the Cross-Entropy Method presented in the following section.

1.2 The Cross-Entropy Method

The Cross-Entropy method (CEM; see [2] for a detailed review) is a method originally designed for integration on rare events. It proved to apply quite naturally to optimization. CEM provides a solid mathematical justification to evolutionary sampling methods, rigorously introducing the selection step in the estimation procedure. Imagine we want to compute

$$I := \mathbb{P}_u(A) = \mathbb{E}_u 1_A = \int 1_A(x) f(dx; u) \tag{1}$$

where the expectation is taken with respect to a pdf $f(x; u)$ belonging to some parametric family $\mathcal{F}$ and A is $\mathbb{P}_u$-rare. If we know how to sample from $f(x; u)$, we can compute a crude Monte-Carlo estimate of (1). But as A is rare for $\mathbb{P}_u$, few of the sampled points will happen to fall in A, so we prefer to use importance sampling to reduce the variance of our MC estimator by sampling more points in the region of interest A. Importance sampling in this case consists in writing

$$I = \int 1_A(x) \frac{f(dx; u)}{g(x)} g(x) dx \approx \sum_{i=1}^{N} 1_A(x_i) \frac{f(x_i; u)}{g(x_i)} \tag{2}$$

for some distribution g we chose for easy sampling, whose support contains the support of $f(.; u)$ and $x_1, ..., x_N \sim g$ i.i.d (see [3]).

There is a theoretical answer to the question of the optimal choice of g, as if you take

$$g = \tilde{g} \propto 1_A(x) f(.; u),$$

the variance of your estimator will be 0. Of course, this is of no practical use, since to normalize g you would need the integral of interest I, but you can still try to approximate this $\tilde{g}$ in some sense. In particular, minimizing over $f(.; v) \in \mathcal{F}$ the Kullback-Leibler divergence between $\tilde{g}$ and $f(.; v)$ is equivalent to solve

$$\max_v \int 1_A(x) f(.; u) \log f(.; v), \tag{3}$$

or, taking the empirical counterpart of (3) with eventually a new importance sampling step:

$$\max_v \sum_{i=1}^{N} 1_A(x_i) \frac{f(x_i; u)}{f(x_i; w)} \log f(x_i; v) \tag{4}$$

where the x_i's are drawn independently according to $f(.; w)$.

Let us now turn this remark into an evolutionary optimization algorithm. Imagine we have some fitness function S to maximize over a domain $\mathcal{X}$. The key idea is to think of estimating integrals of the form $\mathbb{P}_u(S(X) \geq \gamma)$. Using the CEM principle to approximate the optimal importance distribution $1_{(S(.) \geq \gamma)} f(.; u)$, the importance sampling paradigm will help us to sample points in $(S(X) \geq \gamma)$. Iteratively repeating the procedure while cleverly adapting γ to keep enough samples in the region of interest should lead us to sample near from the optima of S.

Since we do not care about the value of the integral's estimate, we can get rid of the importance weights in (4) and iteratively optimize our choice of the importance distribution $f(.; v)$ to estimate $\mathbb{P}_{v_{t-1}}(S(X) \geq \gamma_t)$. The core algorithm finally proposed by the authors of [2] is given Figure 2.

Notice that by taking $\mathcal{F}$ to be the family of Gaussians in CEM leads exactly to EMNA, with $\mu = \rho N$.

2 Derivation of the algorithm

2.1 Introduction of Mixtures in the CEM paradigm

The authors of [2] claim that (5) is analytically solvable when $\mathcal{F}$ is an exponential family. It is true, but there is a certain class of more generic distributions which would intuitively allow for a

CEM FOR OPTIMIZATION(S, N, ρ, d)

1 Initialize v_0, set $t := 1$.

2 Sample $x_1, ..., x_N \sim f(.; v_{t-1})$ i.i.d.

3 Compute and order their performances $S(x_1), ..., S(x_N)$ decreasingly.

4 Take γ_t to be the $(1-\rho)$-quantile γ_t of the ordered performances.

5 Solve

$$v_t := \arg\max_v \sum_{i=1}^{N} 1_{(S(.)\geq\gamma_t)}(x_i) \log f(x_i; v) \tag{5}$$

6 If $t \geq d$ and $\gamma_t = \gamma_{t-1} = ... = \gamma_{t-d}$, then stop. Else set $t := t+1$ and go back to step 2.

Figure 2: The CEM algorithm: the goal is to iteratively sample in regions of high fitness.

better fit of disconnected areas $(S \geq \gamma)$, performing better exploration of multimodal landscapes by clustering the data: the mixture distributions. Their simplest form is a weighted sum of distributions belonging to parametric family $\Phi = ((\varphi(., v))_v$:

$$f(.; \mathbf{v}) := \sum_{d=1}^{D} \alpha_d \varphi(.; v_d)$$

where $\sum_d \alpha_d = 1$. In the following subsection, we demonstrate with an EM-flavored technique that they also lead to analytical update formulae, whenever Φ is an exponential family.

2.2 Solving the updates with EM

The problem in its integral form is

$$\max_v \ell(\alpha, v) := \int 1_{(S(X)\geq\gamma)} \times \log\left(\sum_{d=1}^{D} \alpha_d \varphi(x; v_d)\right) \times f(dx; v^{(t-1)}).$$

Let us define

$$\rho_d(x; \alpha, v) := \frac{\alpha_d \varphi(x; \mu_d, \Sigma_d)}{\sum_{d=1}^{D} \alpha_d \varphi(x; \mu_d, \Sigma_d)}$$

and consider the auxiliary quantity

$$L^t(\alpha, v) := \int \sum_{d=1}^{D} 1_{(S(X)\geq\gamma)} \rho_d(x; \alpha, v) \log(\alpha_d \varphi(x; v_d)) f(dx; v^{(t-1)}).$$

Using concavity of the log, it comes

$$L^t(\alpha, v) - L^t(\alpha^{(t-1)}, v^{(t-1)}) \leq \ell(\alpha, v) - \ell(\alpha^{(t-1)}, v^{(t-1)}),$$

so any increase in L^t would mean a bigger-or-equal increase in ℓ. At the same time, maximization of $L^t(\alpha, v)$ leads to a closed formula whenever φ belongs to an exponential family, for example in the Gaussian case, writing $\rho_d^t(x)$ for $\rho_d(x; \alpha^t, \mu^t, \Sigma^t)$, we easily derive

$$\begin{aligned}
\alpha_d^t &= \int 1_{(S(X)\geq\gamma)} \rho_d^{t-1}(x) f(dx; v^{(t-1)}), \\
\mu_d^t &= \frac{1}{\alpha_d^t} \int 1_{(S(X)\geq\gamma)} x \rho_d^{t-1}(x) f(dx; v^{(t-1)}), \\
\Sigma_d^t &= \frac{1}{\alpha_d^t} \int 1_{(S(X)\geq\gamma)} (x - \mu_d^t)(x - \mu_d^t)' \rho_d^{t-1}(x) f(dx; v^{(t-1)})
\end{aligned}$$

whose empirical versions can be directly used as updates in Algorithm 2. Remark that this new algorithm is very similar to Population Monte-Carlo schemes for integration (see [3]).

Function	Expression	Initial range
Sphere	$\sum_{i=1}^{d} x_i^2$	$[-600, 600]^d$
Rastrigin	$10d + \sum_{i=1}^{d} \left(x_i^2 - 10\cos(2\pi x_i)\right)$	$[-5, 5]^d$
Ackley	$20 + e - 20\exp\left(-0.2\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2}\right) - \exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos(2\pi x_i)\right)$	$[-10, 10]^d$

Table 1: The three benchmark functions.

3 Experiments

In this section we experimentally compare EMNA with the proposed algorithm. Starting the optimization with a relatively large number of mixture components (10 for dimension 10, 20 for dimension 50), the algorithm progressively kills them when their mixing proportions go below a certain threshold (10^{-5} in our experiments). We used two different killing strategies. The first strategy, denoted as MN-EDA 1, was to simply continue the procedure with the remaining components without replacing the killed ones. The second strategy, denoted as MN-EDA 2, was to replace the killed components by the ones having the highest mixing proportions, but giving them large initial variances in order to favor exploration around the current detected modes. For each of the three algorithms, 7 independent runs were performed. We plot the mean values of the fitness obtained at the mean of the component with the highest mixing proportion against number of function evaluations. The dotted lines stand for one standard deviation error bars.

As the sample covariance matrix of p points has rank at most p, early degeneracy of the covariance is likely to appear and hinder exploration. This is a well-known problem in EMNA schemes. To avoid it, we followed the softening advice of the authors of [2], taking at each time step $t = 1, 2, ...$ the new covariance matrix to be a weighted sum of the old covariance matrix and the selected sample covariance matrix, the weight of the latter being

$$\beta_t = \beta - \beta\left(1 - \frac{1}{t}\right)^q$$

with $\beta = 0.8$ and $q = 5$. This softening makes the degeneracy appear polynomially with the time rather than exponentially.

We chose common benchmark test functions in the continuous optimization community. We tried to reproduce the conditions of [1], initializing means uniformly over the given initial ranges, taking $N = 2000$ points at each iteration and selecting the best $\mu = 1000$ points to compute the updates. We initialized all variances to 1. The three rows of Figure 3 depict our results on the Sphere, Rastrigin, and Ackley functions, respectively, the latter two being highly multi-modal. The two columns correspond to dimensions $d = 10$ and $d = 50$. The function definitions and initializations are recalled in Table 3. All functions are normalized to present a unique minimum at the origin.

4 Discussion

The presented fitness graphs and spatial and eigenvalue-based diagnoses suggest that EMNA – with the degeneracy-avoiding update – finds the optimal mode after a reasonable number of function evaluations. However, once it has attained it, the covariance matrix eigenvalues do not shrink quickly enough to optimize more precisely and find a better neighborhood of the located optimum. We observe a kind of metastability, where the γ's of successive iterations in the CEM are almost equal.

MN-EDA seems to reach more quickly the best mode with either killing strategy: it automatically selects the best area according to its global model of the surface by focusing on the best components. We observed that after this pre-selection phase, the means of the components quickly concentrated on the best mode. After this step, the behavior is of course very similar to EMNA.

Let us insist on the fact that we used the same N and μ for EMNA and MN-EDA, so updating a mixture costs the same price in function evaluations as updating a single distribution. That is why we think MN-EDA can be considered as an automatic way of tuning the initialization of EMNA.

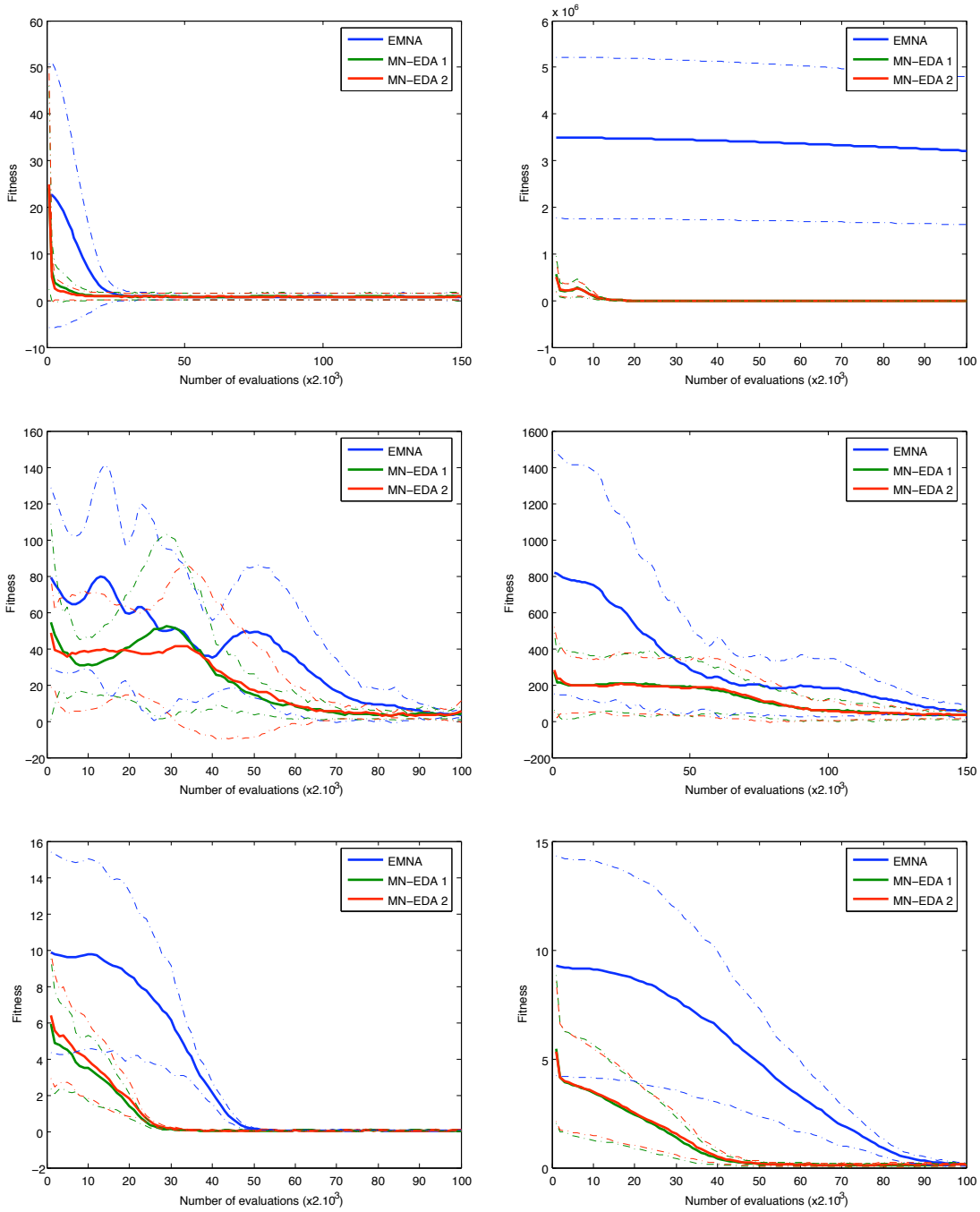

Figure 3: Empirical comparison of EMNA with two different strategies of MN-EDA on the three benchmark functions (Sphere, Rastrigin and Ackley, from top to bottom) in dimensions 10 (left) and 50 (right).

4.1 Two remarks on tails and adaptivity

First, note that heavy-tailed proposals could allow a faster exploration of the landscape when we do not have much prior information on the location of the modes: the updates for Student mixtures exist in a closed form (exponential family property) and can be obtained through a little bit more sophisticated argument than in the Gaussian case.

In our algorithm, we chose to adapt the mixture to the target function by starting with a high number of components and progressively killing the ones with low mixing proportion. We could try other paradigms. In particular, the MCMC community provides us with some interesting techniques to adapt the number of components in a mixture. A first candidate is the Reversible Jump MCMC algorithm [4] in which split-merge moves allow a component with high probability to be split in two or two nearby components to be merged together. Another promising technique is to fit a mixture with an unknown number of components using Dirichlet processes [5, 6].

4.2 Links to CMA-ES

In black-box optimization, the Covariance Matrix Adaptation - Evolutionary Strategies algorithm (CMA-ES) [7] represents the state-of-the-art in real-valued $\mathbb{R}^n$ search spaces. It is a sophisticated version of EMNA, the major difference being that instead of updating the covariance matrix of the selected *points*, it uses the covariance matrix of the selected *steps*. Furthermore, it adds two successful heuristics related to the control of the path length that widen the covariance matrix in the direction of high gradient, which is in contrast with the behavior of EMNA we mentioned earlier. There are several connections we foresee to make with CMA-ES: find a similar probabilistic interpretation, introduce mixtures in it to automatically tune the initialization, and try to mimic its heuristics for MN-EDA, especially the step size control: it would allow us to shrink the covariance matrix eigenvalues faster once we are in the neighborhood of a mode, and consequently solve the problem of metastability we observed in the simulations.

4.3 Potential applications to Machine Learning optimization problems

The presented mixture method not only uses Machine Learning clustering tools for better exploration of the search space, but it could also be applied to difficult optimization tasks that appear in Machine Learning. For example, the authors of [1] claim that EDAs are competitive methods to learn weights in artificial neural networks, so we plan to apply our mixture model to see if we can avoid a poor initialization and local minima of the cost function. With the same flavor, we have in mind applications to the connection design in reservoir computing.

References

[1] P. Larrañaga and J. Lozano, editors. *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*. Springer, 2001.

[2] R. Y. Rubinstein and D. P. Kroese. *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation and Machine Learning*. Springer, 2004.

[3] C. P. Robert and G. Casella. *Monte Carlo Statistical Methods*. Springer-Verlag, New York, 2004.

[4] P. J. Green. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82(4):711–732, 1995.

[5] C. E. Rasmussen. The infinite Gaussian mixture model. In *Advances in Neural Information Processing Systems*, volume 12. The MIT Press, 2000.

[6] R. M. Neal. Markov chain sampling methods for Dirichlet process mixture models. *Journal of Computational and Graphical Statistics*, 9:249–265, 2000.

[7] N. Hansen. The CMA evolution strategy: a comparing review. In J.A. Lozano, P. Larranaga, I. Inza, and E. Bengoetxea, editors, *Towards a new evolutionary computation. Advances on estimation of distribution algorithms*, pages 75–102. Springer, 2006.