

# Resource-Aware Parameterizations of EDA

Sylvain Gelly, Olivier Teytaud, and Christian Gagné \*

June 22, 2006

## Abstract

Bibtex:

```
@inproceedings{tsm2,  
author={Sylvain Gelly and Olivier Teytaud and Christian Gagn},  
title={Resource-aware parametrization of EDAs, proceedings of CEC'06,  
http://www.lri.fr/~sim$teytaud/tsm2.pdf}, year=2006}
```

This paper presents a framework for the theoretical analysis of Estimation of Distribution Algorithms (EDA). Using this framework, derived from the VC-theory, we propose non-asymptotic bounds which depend on: 1) the population size, 2) the selection rate, 3) the families of distributions used for the modelling, 4) the dimension, and 5) the number of iterations. To validate these results, optimization algorithms are applied to a context where bounds on resources are crucial, namely Design of Experiments, that is a black-box optimization with very few fitness-values evaluations.

## 1 Introduction

The optimization of an industrial process where we only have a black-box or a Turing-computable function  $f$  is an interesting problem with direct applications. Usual optimization methods lead to the following troubles: i) in many case, huge cost for each call to the black box (cost in time or cost in money); ii) difficulty or impossibility to get the first and *a fortiori* the second derivatives; iii) difficulty or impossibility of an analytical representation.

A possible solution is the following Surrogate Model (SM) algorithm:

1. Inputs: a sample size  $p$ , a fitness  $f$  to be optimized,
2. Randomly generate  $p$  points uniformly in  $D$ ; evaluate  $f(\cdot)$  at these  $p$  points; add them in the archive;
3. For  $n = 0$  to  $\infty$ :
  - Learn an approximate fitness  $\hat{f}$  on the archive;
  - Define  $x = \arg \min_D \hat{f}$  and add  $(x, f(x))$  to the archive.

This solution looks in particular attractive for very time-consuming cost functions. Then, the complexity is mainly the number of functions evaluations, and it is natural to accept a complex algorithm, involving a difficult task, such as learning an approximate fitness at each generation.

SM has been applied to a variety of problem of increasing size, such the combination of genetic algorithm / neural network / support vector machine for hydrological design [1], the design in aviation and aerospace [2, 3, 4, 5, 6], crash-tests [7], car design [7], dynamic system parameterization [8], and chemical design [9]. In spite of that, there are very few theoretical results about SM. The state of the art mainly concerns local models from the point of view of local convergence. Newton-inspired models can be seen as SM with a polynomial of degree 2 approximation of the fitness, in particular when they do not use derivatives (see [10] and references therein). Various versions of SM often take advantage of two types of extensions w.r.t this traditional degree-2 gradient/Hessian based approximations:

---

\*The authors are from the Équipe TAO (INRIA Futurs), LRI, UMR 8623 (CNRS - Université Paris-Sud), Bat. 490, Université Paris Sud, 91405 Orsay CEDEX, France. Contact author: olivier.teytaud@lri.fr. O. Teytaud and S. Gelly are partially supported by the Pascal Network of Excellence. C. Gagné is supported by postdoctoral fellowships from the ERCIM (Europe) and the FQRNT (Québec).

1. SM that build a model based on previous points and not on the Hessian and gradient;
2. SM that use more complicated approximations, on the whole space and not only at the neighborhood of the last iteration.

Taylor’s expansions are a good argument in favor of the initial Newton-inspired model, as they show that, in smooth cases, polynomials of degree 2 can approximate well locally. But the fact that the first type of extension works has already been pointed out in a famous paper by Powell [10, 11], for the case of 1-dimensional non-linear equation solving. Indeed, Newton-Raphson with finite differences has order 2, which means an average order of  $\sqrt{2}$  (as finite-differences are consuming two points per iteration). In comparison, the secant method (which is a SM) has average order 1.618..., and thus is faster. No such argument has been provided for the above second type of extension.

Various models of EDA have been defined and studied in the literature, for instance: UMDA [12], the compact Genetic Algorithm [13], or the Population-Based Incremental Learning [14]. We will here consider Estimation of Distribution Algorithms by Learning (EDAL) as following, for the minimization of a fitness :

1. Inputs: a population size  $p$ , a selection rate  $\eta = q/p$ , a family of sets  $F \subset 2^D$  with finite VC-dimension  $V$  that contains  $\emptyset$  and  $D$ , a fitness  $f$  to be minimized;
2. Define  $D_0 = D$ ;
3. For  $n = 0$  to  $\infty$ :
  - Randomly generate  $p$  points uniformly in  $D_n$ ;
  - Select  $D_{n+1} \in F$  included in  $D_n$  satisfying the constraint  $C$  (defined below). Among the possibly many possibilities, choose one of the  $D_{n+1}$  such that the proportion of the population that lies in  $D_{n+1}$  is the closest to  $\eta$ .

Definition of the constraint  $C$ :  $D_{n+1}$  satisfies the constraint  $C$  if it is consistent with examples, i.e. with  $(x_{(i)})_{i \in [1, p]}$  the sorted population,  $C$  holds if  $1 \leq i \leq q \Rightarrow x_{(i)} \in D_{n+1}$  and  $p \geq i > q \Rightarrow x_{(i)} \in D_n \setminus D_{n+1}$ . We assume that the probability of having two points with the same fitness is 0, to avoid dealing with case of equalities. Note that  $D_{n+1}$  is well-defined as we have assumed that the empty set was in  $F$ .

EDA have been studied in [15, 16, 17, 18], but has not yet been theoretically studied as a generalization of SM. In our formalism above, SM appear as the limit case  $D_{n+1} = \{\arg \min \hat{f}\}$ , i.e.  $\eta = 0$ . Also, most theoretical results prove a given behavior in the case of simple, well-defined, discrete problems. These results are interesting, as discrete problems are very important, and simple problems give raise to more accurate analysis, but we do think that there is place for general results, independent of the problem (discrete or continuous). This is made difficult by the variety of methods that can be used for learning : the approximation method can be a neural networks [7], a RBF, or a SVM [19, 1]; ensemble methods can be included [20] or not; the model can be local [10] or global [7]; it can be updated during the local search [21] or at a global level; the real-model can be used only for the best individuals [22, 20], or for randomly selected individuals [23], or for the most uncertain individuals [24], or for the potentially best individuals depending on error bounds [25, 26, 21]; the evolution according to the SM can be controlled with a given frequency [27], or only after convergence [28], or some more specific method [23, 29, 30]; meta-models can also be used for defining “informed operators”, what introduces a large variety of algorithms [31, 32]. For dealing with various tools for learning, we use VC-dimension, a classical tool from learning theory. Mainly, the advantage is that both lower and upper bounds exist depending only on the VC-dimension. The interested reader is referred to [33, 34] for a survey of statistical learning.

The goal of this paper is to show theoretical and practical results on these algorithms. The framework “difficult optimization” implies for us: i) no access to derivatives, ii) we do not use derivability. Statement ii) is stronger than the previous statement as even with no access to derivatives, the derivatives could exist and one could sometimes use approximations of derivatives (e.g. approximations of the gradient and the Hessian by finite differences or “surrogate” interpolation).

Our framework is as follows. First, as SM or EDA are often aimed at global optimization, we are interested to study its global convergence. Second, we look for *non-asymptotic* bounds on the precision. This provides bounds that depends on available resources (in particular execution time), and that can lead to the *a priori* optimization of the selection rate and the population size depending on resources. All the  $O(\cdot)$  provided as bounds in the sequel will have *non-asymptotic* constants. They are with respect to some confidence threshold, but they are non-asymptotic. This element is central in the wide area of research initiated by VC-theory in learning. We consider that this non-asymptotic approach is important in optimization as well, as usual methods such as the asymptotic order of convergence do not illustrate the efficiency of algorithms such as genetic algorithms, EDA, SM, for finite number

of individuals. We consider that this point is a strong element in favor of the importance of our results. Without such restrictions, much better convergence rates exist in the literature. In particular, the number of correct digits sometimes doubles at each iteration, or increases more than linearly; but the constant is only asymptotic. Even when the derivative and the Hessian are available, the fast convergence rates only occur asymptotically. For this reason, perhaps the right term for our results is not “convergence rates”, but “bounds on precision”. Some important works have already been done about SM from the convergence rate point of view (see e.g. [10] and references therein). These results consider surrogate models using approximations in the neighborhood of the best point and convergence rates, whereas we consider here global convergence, and bounds instead of rates.

The remaining of the paper is organized as follows: definitions and notation in Section 2, followed by theoretical results in Section 3, then experiments in Section 4, before finishing on some discussions in Section 5.

## 2 Definitions and Notations

$f$  is a deterministic function with domain  $D = [0, 1]^d$ .  $F$  is a subset of  $2^D$ . We note  $x^* \in \operatorname{argmin}_D f$  (assume that  $\operatorname{argmin} f$  is non-empty). In all the sequel, *Card Set* is the number of elements in the set *Set* and  $\mu(\cdot)$  is Lebesgue’s measure. We note  $A = \Omega(B)$  when  $B = O(A)$ . All the constants hidden in the  $O(\cdot)$  (or  $\Omega$ ) only depend on  $F$  and the confidence level.

## 3 Theoretical results

We consider in the sequel an optimization problem on the continuous domain  $D = [0, 1]^d$ . Indeed, we could consider discrete spaces as well, the main tool, VC-dimension, being devoted to learning in general and not only to learning in continuous spaces. However, this would lead to the fact that two outcomes could be exactly identical, what leads to rounding effects in the shrinking property (Equation 1), that would make the writing much more tricky. We therefore stay in the continuous case, but we use tools that are general.

There are strong limitations to the theorem below : assumptions H2 and H3 are too restrictive and could be replaced by much less restrictive assumptions. The goal of this work is only to show, thanks to VC-theory, a general non-asymptotic result for EDAL that is parameterized by all quantities of interest : VC-dimension (which reflects the dimension, as explained after the theorem), population size, number of iterations, selection rate. Tighter and more general results will be a further work.

### Theorem:

Consider  $x$  a random variable uniform in  $D$ . Consider the EDA defined in the introduction (EDAL).

For any  $\epsilon \in [0, 1]$ , and  $D' \subset D$  with non-zero measure, define  $Q_\epsilon(D')$  the set of  $p \in D'$  such that  $P(f(x) < f(p) | x \in D') \leq \epsilon$ . We note for short  $Q_\epsilon = Q_\epsilon(D)$ .

Then for some universal constant  $K > 0$ , the following holds.

Assuming H1, *relevance of the set of distributions*:  $\forall \epsilon \in ]0, 1[, Q_\epsilon \in F$ .

Assuming H2, *the  $Q_\epsilon$  are well nested*: for some  $\epsilon'$ ,  $\forall \epsilon \in ]0, 1[, \forall f \in F, \mu(f \cap Q_\epsilon) \geq (1 - \epsilon')\mu(Q_\epsilon) \Rightarrow \exists \epsilon'' \in ]0, 1[, Q_{\epsilon''}(f) = Q_{\epsilon''}$

Assuming H3, *the population size is sufficient*:  $\zeta = K\sqrt{(V \ln(p) - \ln(1 - \sqrt[n]{1 - \delta}))/p}$  is such that  $\zeta \leq \epsilon'$ .

Then, with probability  $1 - \delta$ ,

$$\mu(D_N) \leq \alpha^{Np} \text{ and } Q_0 \subset D_n$$

where the average convergence rate  $\alpha$  is  $\alpha = (\eta + \zeta)^{1/p}$ .

### Proof:

We will indeed prove the following by induction for  $n \leq N$ : with probability at least  $(1 - \delta)^{(n/N)}$ ,

$$\mu(D_n) \leq \alpha^{np} \tag{1}$$

$$Q_\eta(D_n) \in F \text{ and } Q_\eta(D_n) = Q_t \text{ for some } t > 0. \tag{2}$$

Equation 1 is clear for  $n = 0$ . Equation 2 comes from the fact that  $Q_\eta(D_0) = Q_\eta$  and H1 that implies that  $Q_\eta$  is in  $F$ .

We now turn our attention to the induction. We assume Equations 1 and 2 for  $n$ , and we show them for  $n + 1$ . This means that we assume Equations 1 and 2, and that we must show equations 3 and 4 :

$$\mu(D_{n+1}) \leq \alpha^{(n+1)p} \tag{3}$$

$$Q_\eta(D_{n+1}) \in F \text{ and } Q_\eta(D_{n+1}) = Q_t \text{ for some } t > 0. \quad (4)$$

First, let's show Equation 3.

$Q_\eta(D_n)$  is in  $F$  by Equation 2.

Standard theorems about VC-dimension (see e.g. [34]) ensure that for some universal constant  $K$  with probability at least  $\sqrt[n]{1-\delta}$ ,

$$\forall f \in F, |\hat{P}(f|D_n) - P(f|D_n)| \leq \zeta/2 \quad (5)$$

where  $\hat{P}$  is the empirical distribution associated to the generation of  $p$  examples, i.e. the average of  $p$  Dirac masses centered at an i.i.d. sample of the uniform law restricted to  $D_n$ .

We will derive Equation 3 mainly by using Equation 5.

1. Equation 2 implies that  $\exists f \in F; P(f|D_n) = \eta$  and  $f$  satisfies constraint  $C$ .
2. By construction,  $D_{n+1}$  verifies that  $|\hat{P}(D_{n+1}|D_n) - \eta| \leq |\hat{P}(f|D_n) - \eta|$ .
3. By Equation 5 and point 1 above,  $\hat{P}(f|D_n) \leq \eta + \zeta/2$ .
4. By points 2 and 3 above,  $\hat{P}(D_{n+1}|D_n) \leq \eta + \zeta/2$ .
5. By points 4 and Equation 5, we have proved that  $P(D_{n+1}|D_n) \leq \eta + \zeta$ .

We just have to multiply the confidence rate  $(1-\delta)^{n/N}$  by  $(1-\delta)^{1/N}$  (the confidence of Equation 5) to get the confidence rate  $(1-\delta)^{(n+1)/N}$ .

This leads to the induction for Equation 3. We only have to prove Equation 4.

Thanks to Equation 5 and H3, H2 applies with  $f = D_{n+1}$  and  $\epsilon = \epsilon'$ . This proves Equation 4.

We have proved Equations 3 and 4; the induction is complete.

The proof is complete.

### Some classical VC-dimensions ([34]):

To illustrate the possibility of our model, we consider an application in a simple case. If we consider ellipsoids in a continuous domain, which are a natural approximation of the  $Q_\epsilon$  for smooth enough functions, then the VC-dimension is quadratic in the dimension. If we consider a multi-modal problem, and therefore use finite unions of ellipsoids, the VC-dimension is then bounded by  $O(l \times d^2)$  where  $l$  is the number of ellipsoids (which can be controlled by adding the constraint that the algorithm chooses the smallest possible number of ellipsoids as  $D_{n+1}$ ). Note that under a nice conditioning of the fitness, leading to balls instead of ellipsoids, the VC-dimension decreases to  $O(d)$ .

This leads to the following remarks about the population size:

- $\alpha$  lower than 1 implies that  $p/\log(p)$  is  $\Omega(ld^2)$ . Roughly, this mean that the required population size is at least linear in the number of optima and increases quadratically as the dimension.
- $\alpha$  lower than 1 implies that  $p$  is  $\Omega(-\log(1 - \sqrt[n]{1-\delta}))$  ( $p$  increases at least linearly in  $\log(\delta/N)$ ). To reduce risk, increase the population size (linear in the logarithm of the risk). If you have more time, increase the population (linearly in the logarithm of the number of iterations).
- $\eta$  and  $\epsilon'$  must fulfill H2. This hypothesis looks like a conditioning hypothesis.

The optimal  $p$  and  $\eta$  can be computed from the theorem below, depending on the total number of function evaluations required.

## 4 Experiments

For the experiments, we used a slightly modified version of the algorithm. We generate only one point at each generation, and we proceed as follows for the definition of  $D_{n+1}$ :

- Learn  $\hat{f}$  an approximate fitness on all the domain thanks to all points in the archive;
- In the case of an EDA:
  - If the size of the archive is  $n$ , choose a threshold  $t$  such that  $t - \inf \hat{f} = f(x) - f(x^*)$ , where  $x$  is the  $(n/5)^{th}$  best point of the archive, and  $x^*$  is the best point in the archive;
  - $D_{n+1}$  is  $\{x; f(x) \leq t\}$ .

- In the case of a SM:  $D_{n+1}$  is  $\{\operatorname{argmin} \hat{f}\}$ .

Additionally, for the sake of robustness, when 5 divides  $n$  or when  $n \leq 10$ , the newly generated point is chosen by quasi-random on all  $D$  instead of generating points uniformly on  $D_n$ . The quasi-random sequence is the Niederreiter sequence (see [35] for an explanation); the reader can here only think of a deterministic sequence which is more “uniform” on the domain than a random sequence.

The generation of  $x$  uniformly in  $D_m$  has been approximated as follows: i) set  $\lambda = 1$ , ii) generate  $x$  uniformly in  $D_m$ , iii) set  $x = \lambda x + (1 - \lambda)\operatorname{argmin} g_m$ , and iv) if  $x \notin F_m$  then decrease slowly  $\lambda$  and go back to point ii).

We compared below the following optimization algorithms:

- LBFSGBR: LBFSGSB with restart [36, 37, 38];
- RO: Random optimizer (i.e. a naive random search);
- SMMP: SM using a Multi-layer Perceptron as learner;
- EMMP: EDA-Model using a Multi-layer Perceptron as learner;
- SMRBFN: SM using a Radial Basis Function (RBF) network as learner;
- EMRBFN: EDA-Model using a RBF network as learner;
- SMSMO: SM using a Sequential Minimal Optimization (SMO) [39] SVM as learner;
- EMSMO: EDA-Model using a SMO SVM as learner;
- $(\mu_W, \lambda)$ -CMAES: Covariance Matrix Adaptation Evolution Strategy [40] using weighted recombination, a population of 3 parents in dimension 2 and 5 parents in dimension 10 and generating two times the number of parents as children at each generation ( $\lambda = 2\mu$ );
- (1, 5)-CMAES: Covariance Matrix Adaptation Evolution Strategy [40] using one parent and five children at each generation;
- CDFO: Coin-DFO<sup>1</sup>, [10];
- PO: PDS-Optimizer<sup>2</sup> [11].

All learning methods used in the SM and EDA come from Weka [41] and are used with their default parameterization, while CMA-ES come from the Open BEAGLE C++ framework<sup>3</sup> [42]. The second parameterization of CMAES ((1, 5)-CMAES) has been tested here given that it would be more adapted for the small number of iterates under consideration, than the default weighted recombination  $(\mu_W, \lambda)$  strategy implemented in Open BEAGLE.

Given they are population-based optimization algorithms, we acknowledge that CMA-ES and evolution strategies in general are not specifically designed for dealing with optimization problems that have strong limitations on the number of evaluation points. It is not expected that they will perform very well in the present context, which doesn’t mean at all that they are not excellent optimization algorithms in less frugal optimization context. Nevertheless, we think that is still very interesting to compare their performance with other algorithm, in the context of optimization with very few points.

The benchmark functions used for the experiments are taken from [43]. A continuous EDA has already been tested on this benchmark in [44], but with much larger numbers of function evaluations. The first functions of the benchmark are the most simple, while the following ones are multi-modal and difficult. All these results are averaged among 33 runs. Note that performing better than random search for this criterion is not obvious: non-random algorithms can neglect some parts of the domain, due to their prior, and therefore have a often larger probability of very bad result, what has a strong impact on the average result.

## 4.1 Dimension 2 with 16 points

Table 1 shows results obtained on the different functions from [43], working in dimension 2 and limiting the number of evaluation points for each optimization run to 16. From these results, when can conclude that EMSMO was:

---

<sup>1</sup><http://www.coin-or.org>

<sup>2</sup><http://csmr.ca.sandia.gov/opt++>

<sup>3</sup><http://beagle.gel.ulaval.ca>

Table 1: Results in dimension 2 with 16 evaluation points for the 14 functions taken from [43]. Smaller values mean better results.

|     | LFBGSBR  | RO       | SMMP    | EMMP     | SMRBFN  | EMRBFN   | SMSMO   | EMSMO   | $(\mu_W, \lambda)$ -CMAES | (1, 5)-CMAES | CDFO     | PO       |
|-----|----------|----------|---------|----------|---------|----------|---------|---------|---------------------------|--------------|----------|----------|
| f1  | -450.0   | 266.4    | -254.3  | -256.8   | -254.3  | -261.3   | -254.3  | -258.2  | 244.3                     | 1314.4       | 2478.9   | 9803.7   |
| f2  | -450.0   | 492.2    | -112.1  | -139.4   | -112.1  | -132.9   | -112.1  | -121.4  | 635.0                     | 1191.2       | 2555.0   | 12188.8  |
| f3  | 7.45e+03 | 4.35e+07 | 7.4e+05 | 7.17e+05 | 7.4e+05 | 7.32e+05 | 7.4e+05 | 7.3e+05 | 4.83e+07                  | 3.3e+08      | 1.92e+09 | 7.81e+09 |
| f4  | 8744.8   | 1230.2   | -6.2    | -24.6    | -6.2    | -36.9    | -6.2    | -26.1   | 498.4                     | 1810.3       | 3272.7   | 11680.5  |
| f5  | -238.0   | 2161.6   | 5290.0  | 3631.0   | 5290.0  | 3674.7   | 5290.0  | 3753.7  | 1827.6                    | 238.2        | 6525.0   | 8204.5   |
| f6  | 1.56e+05 | 9.75e+05 | 1.3e+04 | 1.23e+04 | 1.3e+04 | 1.19e+04 | 1.3e+04 | 1.3e+04 | 4.86e+06                  | 2.02e+06     | 2.72e+09 | 2.23e+10 |
| f7  | 341.5    | -41.9    | -139.6  | -139.7   | -140.2  | -119.4   | -139.6  | -140.0  | -138.7                    | 105.1        | 317.5    | 680.3    |
| f8  | -119.2   | -119.5   | -119.1  | -119.1   | -119.3  | -119.2   | -119.2  | -119.1  | -120.1                    | -119.8       | -118.4   | -119.7   |
| f9  | -303.5   | -317.6   | -325.4  | -325.4   | -325.4  | -325.5   | -325.4  | -325.5  | -319.5                    | -313.5       | -303.8   | -307.4   |
| f10 | -283.2   | -313.0   | -307.0  | -311.4   | -307.0  | -310.4   | -307.0  | -309.5  | -316.3                    | -301.8       | -281.6   | -301.9   |
| f11 | 92.8     | 92.0     | 91.9    | 91.9     | 91.9    | 91.9     | 91.9    | 91.9    | 92.0                      | 92.0         | 94.0     | 92.0     |
| f12 | 2960.8   | 856.4    | -414.2  | -414.2   | -414.2  | -414.2   | -414.2  | -414.3  | 950.0                     | 1360.1       | 10853.1  | 1209.4   |
| f13 | 4447.3   | -39.6    | -124.1  | -124.6   | -124.1  | -124.2   | -124.1  | -124.5  | -83.0                     | 120399.0     | -3.8     | 6883.1   |
| f14 | -299.0   | -299.1   | -299.1  | -299.1   | -299.1  | -299.1   | -299.1  | -299.2  | -299.1                    | -299.1       | -299.0   | -299.0   |

Table 2: Results in dimension 10 with 16 evaluation points.

|     | RO       | SMMP     | EMMP     | SMRBFN   | EMRBFN   | SMSMO    | EMSMO    | $(\mu_W, \lambda)$ -CMAES | CDFO     | PO       |
|-----|----------|----------|----------|----------|----------|----------|----------|---------------------------|----------|----------|
| f1  | 25787.8  | 27762.7  | 26206.0  | 27861.1  | 27161.5  | 27758.7  | 26997.0  | 22893.9                   | 27942.5  | 58656.9  |
| f2  | 40818.5  | 23940.5  | 23940.5  | 23940.5  | 23940.5  | 23940.5  | 23680.3  | 39807.5                   | 64534.5  | 269006.2 |
| f3  | 7.25e+08 | 3.24e+08 | 3.24e+08 | 3.24e+08 | 3.17e+08 | 3.24e+08 | 3.21e+08 | 7.38e+08                  | 1.60e+09 | 4.25e+09 |
| f4  | 49004.6  | 29912.4  | 29912.4  | 29912.4  | 29583.6  | 29912.4  | 29779.3  | 52509.2                   | 80719.2  | 207297.9 |
| f5  | 19431.3  | 25269.6  | 23040.0  | 25269.6  | 21819.0  | 25269.6  | 22217.9  | 18824.9                   | 25574.8  | 31949.5  |
| f6  | 1.46e+10 | 7.21e+09 | 7.16e+09 | 7.21e+09 | 7.05e+09 | 7.21e+09 | 6.90e+09 | 1.56e+10                  | 1.40e+10 | 7.43e+10 |
| f7  | 3705.4   | 1089.2   | 1089.9   | 1090.1   | 1112.1   | 1089.3   | 1089.5   | 1543.3                    | 4706.5   | 5073.1   |
| f8  | -118.8   | -118.8   | -118.8   | -118.8   | -118.8   | -118.8   | -118.8   | -118.8                    | -118.6   | -118.7   |
| f9  | -181.9   | -196.8   | -195.5   | -191.9   | -193.3   | -198.2   | -198.7   | -182.4                    | -186.8   | -125.9   |
| f10 | -118.9   | -145.1   | -144.2   | -143.8   | -144.7   | -144.2   | -146.3   | -114.4                    | -70.5    | 9.7      |
| f11 | 105.5    | 105.4    | 105.5    | 105.2    | 105.3    | 105.4    | 105.3    | 105.0                     | 110.0    | 105.9    |
| f12 | 3.11e+05 | 3.00e+05 | 2.83e+05 | 3.00e+05 | 2.80e+05 | 3.00e+05 | 2.88e+05 | 2.97e+05                  | 5.11e+05 | 4.45e+05 |
| f13 | 1.68e+05 | 193.1    | 4170.1   | 805.64   | 13373    | 634.14   | 784.82   | 2.10e+05                  | 113.13   | 2.03e+06 |
| f14 | -295.2   | -295.1   | -295.1   | -295.1   | -295.2   | -295.1   | -295.1   | -295.1                    | -295.0   | -295.1   |

- Better than RO for 11 functions among 14;
- Better than LFBGSBR for 9 functions among 14 (9 among the most difficult functions);
- Better than both CMAES for 11 functions among 14.

In comparison:

- $(\mu_W, \lambda)$ -CMAES outperformed RO for 9 functions among 14;
- The second parameterization of CMA-ES, (1, 5)-CMAES, was indeed not better;
- LFBGSBR only outperformed RO for 5 functions among 14. Note however that LFBGSBR was the best algorithm for 4 functions, that were among the easiest ones (functions 1,2,3 and 5).
- CDFO (using local models) was clearly outperformed, what suggests that local models are not suitable for global optimization with a small number of iterates.

## 4.2 Dimension 10 with 16 points

Table 2 shows the results on the same benchmarks [43], this time in dimension 10, using a maximum of 16 evaluation points. This experiments show that **EDA are very efficient for this small number of function evaluations**. Statistically, the EDA-model with SMO as learner is

- Better than RO (random-search) for 11 functions among 14;
- Better than  $(\mu_W, \lambda)$ -CMAES for 10 functions among 14;

Table 3: Results in dimension 2 with 64 evaluation points.

|     | LBFGBSR | RO      | SMMP     | EMMP     | SMRBFN   | EMRBFN   | SMSMO    | EMSMO    | (1,5)-CMAES | CDFO     | PO       |
|-----|---------|---------|----------|----------|----------|----------|----------|----------|-------------|----------|----------|
| f1  | -450.0  | -250.0  | -254.3   | -306.4   | -254.3   | -315.2   | -254.3   | -309.5   | -132.9      | 3544.4   | 7652.6   |
| f2  | -450.0  | -135.9  | -112.1   | -218.3   | -112.1   | -242.0   | -112.1   | -247.8   | -329.6      | 2796.2   | 13191.1  |
| f3  | -449.96 | 3.7e+06 | 7.39e+05 | 5.99e+05 | 7.39e+05 | 6.87e+05 | 7.39e+05 | 5.18e+05 | 1.06e+07    | 2.16e+09 | 5.18e+09 |
| f4  | 8411.7  | -141.0  | -6.2     | -191.5   | -6.2     | -186.1   | -6.2     | -227.3   | 979.4       | 3732.2   | 13215.5  |
| f5  | -310.0  | 754.6   | 2223.3   | 1329.1   | 2223.3   | 1144.4   | 2223.3   | 1430.9   | -293.8      | 7254.9   | 8290.1   |
| f6  | 502.34  | 56998   | 1416.7   | 1316.1   | 1416.7   | 1371.3   | 1416.7   | 1359.1   | 3.29e+05    | 1.96e+09 | 8.07e+09 |
| f7  | -31.4   | -97.3   | -139.6   | -140.6   | -140.3   | -140.5   | -139.6   | -139.9   | 68.6        | 305.7    | 414.5    |
| f8  | -119.8  | -120.0  | -119.9   | -120.0   | -119.9   | -119.9   | -119.9   | -119.9   | -120.0      | -118.5   | -119.9   |
| f9  | -320.2  | -322.9  | -325.4   | -325.9   | -325.4   | -325.6   | -325.4   | -325.8   | -318.1      | -304.1   | -322.3   |
| f10 | -311.1  | -319.2  | -307.0   | -318.8   | -307.0   | -318.2   | -307.0   | -317.6   | -315.9      | -289.7   | -319.7   |
| f11 | 92.6    | 91.3    | 91.9     | 91.6     | 91.9     | 91.6     | 91.9     | 91.5     | 91.7        | 93.8     | 91.2     |
| f12 | -222.2  | -57.1   | -414.2   | -418.9   | -414.2   | -414.5   | -414.2   | -416.4   | 291.6       | 11952.5  | 109.1    |
| f13 | -6.7    | -127.7  | -129.8   | -129.8   | -129.8   | -129.8   | -129.8   | -129.8   | 55891.3     | -3.1     | -123.2   |
| f14 | -299.0  | -299.2  | -299.1   | -299.3   | -299.1   | -299.2   | -299.1   | -299.2   | -299.2      | -299.0   | -299.0   |

Results also show that **EDA (or SM with a well-shaped diversification) are a very good extension of SM**. The EDA-model with  $X=\{SMO,RBF,MP\}$  as learner is better than the SM with  $X$  as learner in {12 cases among 14, 10 cases among 13, 5 cases among 11}. The case of MP is less important as the MP has not been efficient, either for EDA or for SM. This is probably due to the fact that MP need more examples for being accurate, whereas SMO is aimed at generalizing well from a small sample. Finally, we can also conclude that CoinDFO, which uses a local model, is much less efficient here.

### 4.3 Dimension 2 with 64 points

Using 64 points in dimension 2, results presented in Table 3 show us that:

- For 9 functions among 14 (the most difficult functions), each of the EDA (EMMP, EMRBFN, EMSMO) outperformed LBFGBSR;
- LBFGBSR is the best for 5 functions (among the most simple);
- PDS-optimizer becomes efficient and is the best for two functions (f10 and f11). However it is outperformed by each of the EDA for all other functions;
- EMMP has been the best algorithm for 5 functions (among the most difficult): f7, f9, f12, f13, f14;
- For function f9 and f13, the 6 best algorithms are the SM and the EDA. This is indeed due to the quasi-random exploration that works well for f9 and f13. for f4, EMMP was only outperformed by another EDA, EMSMO.

This mean that even in dimension 2, for 64 function-evaluations, what is not the case of a very small number of function evaluations in front of the dimension, the SM and EDA remain efficient at least for difficult functions.

### 4.4 Dimension 10 with 64 points

And finally, Table 4 presents the results in dimension 10, using 64 evaluation points. From this, we can observe that:

- LBFGBSR was the best algorithm for 5 functions, among the easiest ones (f1,f2,f3,f6,f12);
- For function f4, the EDA and SM all outperform random search, and are the only algorithms that outperform random search;
- For functions f5 and f11, (1,5)-CMAES was the best algorithm;
- Each of the EDA (EMMP, EMRBF, EMSMO) outperformed LBFGBSR for 8 functions among 14 (among the most difficult);
- EMSMO outperformed (1,5)-CMAES for 10 functions among 14 and random search for 10 functions among 14;
- EMRBF outperformed (1,5)-CMAES for 11 functions among 14 and random search for 10 functions among 14;

Table 4: Results in dimension 10 with 64 evaluation points.

|     |          |          |          |          |          |          |          |          |           |          |          |
|-----|----------|----------|----------|----------|----------|----------|----------|----------|-----------|----------|----------|
| f1  | -450.0   | 19027.6  | 27714.5  | 20882.8  | 27801.9  | 18681.0  | 27721.6  | 19176.8  | 20386.5   | 27408.5  | 64250.6  |
| f2  | 11596.0  | 23569.0  | 23940.0  | 23565.0  | 23940.0  | 23083.0  | 23940.0  | 23015.0  | 34077.0   | 62891.0  | 191383.0 |
| f3  | 1.05e+08 | 3.12e+08 | 3.24e+08 | 2.87e+08 | 3.24e+08 | 2.68e+08 | 3.24e+08 | 2.82e+08 | 3.38e+08  | 1.54e+09 | 5.84e+09 |
| f4  | 1.63e+05 | 30932    | 28816    | 27355    | 28816    | 26548    | 28816    | 25896    | 48585     | 79352    | 1.99e+05 |
| f5  | 12594.5  | 16732.0  | 13442.4  | 13227.9  | 13442.4  | 13253.9  | 13442.4  | 13140.5  | 11442.3   | 26048.7  | 31594.0  |
| f6  | 3.24e+08 | 6.25e+09 | 7.21e+09 | 6.03e+09 | 7.21e+09 | 5.65e+09 | 7.21e+09 | 5.95e+09 | 1.65e+10  | 1.40e+10 | 6.99e+10 |
| f7  | 1181.9   | 2924.4   | 1088.8   | 1088.8   | 1088.9   | 1103.7   | 1088.8   | 1088.9   | 4820.9    | 4795.8   | 5670.0   |
| f8  | -118.6   | -119.0   | -119.0   | -119.0   | -119.0   | -119.0   | -119.0   | -119.0   | -119.0    | -118.6   | -118.9   |
| f9  | -155.5   | -209.3   | -225.4   | -215.7   | -209.3   | -198.6   | -216.8   | -209.3   | -164.5    | -185.6   | -132.7   |
| f10 | -53.8    | -145.7   | -161.4   | -159.9   | -148.2   | -159.7   | -157.5   | -167.3   | -101.9    | -63.1    | -12.6    |
| f11 | 106.8    | 103.9    | 104.6    | 104.4    | 104.6    | 104.5    | 104.5    | 104.7    | 103.7     | 109.9    | 104.5    |
| f12 | 1.58e+05 | 1.86e+05 | 3.00e+05 | 2.22e+05 | 3.00e+05 | 2.08e+05 | 3.00e+05 | 2.05e+05 | 2.49e+05  | 5.47e+05 | 4.31e+05 |
| f13 | 5748.8   | 65952.7  | 32.1     | 113.8    | 184.5    | 257.5    | 158.1    | 178.9    | 2242855.6 | 113.1    | 787898.4 |
| f14 | -295.1   | -295.4   | -295.1   | -295.3   | -295.1   | -295.3   | -295.1   | -295.3   | -295.4    | -295.0   | -295.1   |

- EMMP outperformed (1,5)-CMAES for 8 functions among 14 and random search for 10 functions among 14;
- For a comparison, LBFSGBR outperformed random search for 8 functions among 14 and (1,5)-CMAES only outperformed random-search for 6 functions among 14.

Once again, EDA have been the most stable algorithms.

## 5 Conclusion

We provided a common framework for EDA and SM, based on generations correlated with the shape of nearly optimal points. The main features are resource-aware bounds, depending on the VC-dimension, the population size, the selection rate.

The hypothesis on the shape of distributions’ supports is that it must be able of approximating the neighborhood of the set of optima; this avoids too-fast local convergence. For locally quadratic functions (i.e. functions with non-degenerated Taylor’s expansions), this suggests mutations that have a covariance (ellipsoid). This is very related to what is done in CMAES [45].

The superiority of SM and EDA, in front of naive methods, increases as the dimension increases with respect to the number of points. This is explained by the fact that the curse of dimensionality is handled by the learning. In particular, many works in the last 10 years have shown that the huge number of weights (e.g. in neural networks), which may be the consequence of a huge dimension and an ability to approximate complex fitnesses, does not necessarily lead to a huge sample complexity (i.e. a large amount of points for learning). See [33] and references therein for more details.

We consider that the main contributions of this paper are:

- Demonstration of the relevance of VC-theory and statistical learning theory in general for the study of Estimation of Distribution Algorithms;
- Illustration of the relevance of EDA for Design of Experiments, i.e. optimization in very frugal contexts;
- Idea that EDA are a natural extension of SM, in which noise-introduction or various heuristics are clearly formalized.

The mathematical proofs introduce the use of statistical learning for the study of EDA, but they are based on assumptions that are (i) much too restrictive (ii) unnecessary. Comparable properties can be derived with sharper constants, and much less restrictive assumptions (in particular hypothesis H2).

## References

- [1] S. Yan and B. Minsker, “A dynamic meta-model approach to genetic algorithm solution of a risk-based ground-water remediation design model,” in *Proc. of the 2004 World Water & Environmental Resources Congress*, 2004.

- [2] A. R. Center, "Aerodynamic design using neural networks, the amount of computation needed to optimize a design is reduced," NASA Tech Briefs Online, Tech. Rep., 2003.
- [3] B. Grossman, "Surrogate models in aircraft design," in *Proc. of the 1st Int. Work. on Surrogate Modelling and Space Mapping for Engineering Optimization*, 2000.
- [4] D. Hopkins, T. Lavelle, and S. Patnaik, "Neural network and regression methods demonstrated in the design optimization of a subsonic aircraft," NASA Glen Research Center, Research & Technology, Tech. Rep., 2002.
- [5] M. Husken, Y. Jin, and B. Sendhoff, "Structure optimization of neural networks for evolutionary design optimization," in *Workshops of the GECCO 2002*, 2002, pp. 13–16.
- [6] A. Keane and P. Nair, "Computational approaches for aerospace design : The pursuit of excellence," 2005.
- [7] M. V. Grieken, "Optimisation pour l'apprentissage et apprentissage pour l'optimisation," Ph.D. dissertation, Université Paul Sabatier, 2004.
- [8] J. Kleijnen, "Sensitivity analysis of simulation experiments: regression analysis and statistical design," *Mathematics and Computers in Simulation*, vol. 34, no. 3-4, pp. 297–315, 1992.
- [9] S. Ibric, M. Jovanovic, Z. Djuric, J. Parojcic, S. Petrovic, L. Solomun, and B. Stupar, "Artificial neural networks in the modeling and optimization of aspirin extended release tablets with Eudragit L100 as matrix substance," *AAPS Pharm. Sci. Tech.*, vol. 4, no. 1, 2003.
- [10] A. Conn, K. Scheinberg, and L. Toint, "Recent progress in unconstrained nonlinear optimization without derivatives," 1997. [Online]. Available: [citeseer.ist.psu.edu/conn97recent.html](http://citeseer.ist.psu.edu/conn97recent.html)
- [11] M. J. D. Powell, "Unconstrained minimization algorithms without computation of derivatives," *Bollettino della Unione Matematica Italiana*, vol. 9, pp. 60–69, 1974.
- [12] H. Mühlenbein and T. Mahnig, "Evolutionary computation and Wright's equation." *Theoretical Computer Science*, vol. 287, no. 1, pp. 145–165, 2002.
- [13] G. R. Harik, F. G. Lobo, and D. E. Goldberg, "The compact genetic algorithm," *IEEE Trans. on Evolutionary Computation*, vol. 3, no. 4, p. 287, November 1999.
- [14] S. Baluja, "Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning,," School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, Tech. Rep. CMU-CS-94-163, 1994. [Online]. Available: [citeseer.ist.psu.edu/baluja94population.html](http://citeseer.ist.psu.edu/baluja94population.html)
- [15] S. Droste, "Not all linear functions are equally difficult for the compact genetic algorithm," in *Proc. of the Genetic and Evolutionary Computation Conference (GECCO 2005)*, 2005, pp. 679–686.
- [16] J. L. Shapiro, "The sensitivity of PBIL to its learning rate, and how detailed balance can remove it," in *Foundations of Genetic Algorithms VII*, 2002.
- [17] —, "Drift and scaling in estimation of distribution algorithms," *Evolutionary Computation*, vol. 13, no. 1, 2005.
- [18] H. Mühlenbein and R. Höns, "The estimation of distributions and the minimum relative entropy principle," *Evolutionary Computation*, vol. 13, no. 1, pp. 1–27, 2005.
- [19] K. Abboud, "Algorithmes évolutionnaires assistés par des méthodes d'apprentissage en grande dimension," Ph.D. dissertation, Ecole Polytechnique, France, 2004.
- [20] Y. Jin and B. Sendhoff, "Reducing fitness evaluations using clustering techniques and neural networks ensembles," in *Genetic and Evolutionary Computation Conference*, ser. LNCS, vol. 3102. Springer, 2004, pp. 688–699.
- [21] Y. S. Ong, K. Lum, P. B. Nair, D. Shi, and Z. Zhang, "Global convergence unconstrained and bound constrained surrogate-assisted evolutionary search in aerodynamic shape design," in *Congress on Evolutionary Computation, Special Session on Design Optimisation with Evolutionary Computation(CEC'03)*, Canberra, Australia, 2003, pp. 1856–1863.

- [22] H.-S. Kim and S.-B. Cho, "An efficient genetic algorithms with less fitness evaluation by clustering," in *Proceedings of IEEE Congress on Evolutionary Computation*. IEEE, 2001, pp. 887–894.
- [23] Y. Jin, M. Olhofer, and B. Sendhoff, "On evolutionary optimization with approximate fitness functions," in *Proceedings of the Genetic and Evolutionary Computation Conference*. Morgan Kaufmann, 2000, pp. 786–792.
- [24] J. Branke and C. Schmidt, "Fast convergence by means of fitness estimation," *Soft Computing Journal*, 2005.
- [25] M. Emmerich, A. Giotis, M. Özdenir, T. Bäck, and K. Giannakoglou, "Metamodel-assisted evolution strategies," in *Parallel Problem Solving from Nature*, ser. Lecture Notes in Computer Science, no. 2439. Springer, 2002, pp. 371–380.
- [26] H. Ulmer, F. Streicher, and A. Zell, "Model-assisted steady-state evolution strategies," in *Proceedings of Genetic and Evolutionary Computation Conference*, ser. LNCS 2723, 2003, pp. 610–621.
- [27] L. Bull, "On model-based evolutionary computation," *Soft Computing*, vol. 3, pp. 76–82, 1999.
- [28] A. Ratle, "Accelerating the convergence of evolutionary algorithms by fitness landscape approximation," in *Proc. of the 5<sup>th</sup> Conference on Parallel Problems Solving from Nature*, T. Bäck, G. Eiben, M. Schoenauer, and H.-P. Schwefel, Eds. Springer-Verlag, LNCS 1498, 1998, pp. 87–96.
- [29] Y. Jin, M. Olhofer, and B. Sendhoff, "Managing approximate models in evolutionary aerodynamic design optimization," in *Proceedings of IEEE Congress on Evolutionary Computation*, vol. 1, May 2001, pp. 592–599.
- [30] —, "A framework for evolutionary optimization with approximate fitness functions," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 5, pp. 481–494, 2002.
- [31] K. Rasheed and H. Hirsh, "Informed operators: Speeding up genetic-algorithm-based design optimization using reduced models," in *Proc. of the 2000 Genetic and Evolutionary Conference (GECCO 2000)*, D. W. . al., Ed., 2000, pp. 628–635.
- [32] K. Abboud, X. Merlot, and M. Schoenauer, "Deterministic function learning within evolutionary algorithms," in *GECCO 2002 Workshop Program*, 2002, pp. 148–151.
- [33] M. Antony and P. Bartlett, *Neural network learning : Theoretical Foundations*. Cambridge University Press, 1999.
- [34] L. Devroye, L. Györfi, and G. Lugosi, *A probabilistic Theory of Pattern Recognition*. Springer, 1997.
- [35] H. Niederreiter, *Random Number Generation and Quasi-Monte Carlo Methods*, P. SIAM, Ed., 1992.
- [36] R. H. Byrd, J. Nocedal, and R. B. Schnabel, "Representation of quasi-newton matrices and their use in limited memory methods," *Mathematical Programming*, vol. 63, no. 4, pp. 129–156, 1994.
- [37] R. H. Byrd, P. Lu, and J. Noceda, "A limited memory algorithm for bound constrained optimization," *SIAM Journal on Scientific and Statistical Computing*, vol. 16, no. 5, pp. 1190–1208, 1995.
- [38] C. Zhu, R. H. Byrd, and J. Nocedal, "L-bfgs-b, fortran routines for large scale bound constrained optimization," *ACM Transactions on Mathematical Software*, vol. 23, no. 4, pp. 550–560, 1997.
- [39] J. Platt, "Fast training of support vector machines using sequential minimal optimization," in *Advances in Kernel Methods – Support Vector Learning*, B. Schölkopf, C. Burges, and A. Smola, Eds. Cambridge, MA: MIT Press, 1999, pp. 185–208.
- [40] N. Hansen and A. Ostermeier, "Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation," in *Proc. of the IEEE Conference on Evolutionary Computation (CEC 1996)*. IEEE Press, 1996, pp. 312–317.
- [41] I. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, 2005.
- [42] C. Gagné and M. Parizeau, "Open BEAGLE: A new versatile C++ framework for evolutionary computations," in *Late breaking papers of the GECCO 2002*, July 9-13 2002, pp. 161–168.

- [43] P. N. Suganthan, N. Hansen, J. J. Liang, K. Deb, Y.-P. Chen, A. Auger, and S. Tiwari, "Problem definitions and evaluation criteria for the cec 2005 special session on real-parameter optimization, Tech. Rep. AND KanGAL Report #2005005, IIT Kanpur, India, 2005. [Online]. Available: [http://public.cranfield.ac.uk/sims\\_staff/wcat/cec2005/sessions/](http://public.cranfield.ac.uk/sims_staff/wcat/cec2005/sessions/)
- [44] B. Yuan and M. Gallagher, "Experimental results for the special session on real-parameter optimization at cec 2005: A simple, continuous eda," in *proceedings of CEC*, 2005.
- [45] N. Hansen and A. Ostermeier, "Completely derandomized self-adaptation in evolution strategies," *Evolutionary Computation*, vol. 11, no. 1, 2003.