Stochastic Minimization with Adaptive Memory

R. Brunelli and **G. P. Tecchiolli**
Istituto per la Ricerca Scientifica e Tecnologica
I-38050 Povo, Trento, ITALY

Abstract— **In this paper a non-deterministic minimization algorithm is presented. A common feature of random search algorithms is that little or no use is made of information on the local structure of the function to be minimized. While this can be justified when the function has a very complicated microstructure, it results in an unnecessary loss of efficiency when the landscape is smooth but anisotropic. To overcome this deficiency we propose a random minimization algorithm with adaptive memory: the algorithm decides by itself how much of the information gathered through the process of minimizing the function can be successfully used to guide the search. Extensive experiments (minimization of quadratic forms, computation of the minimum eigenvalue of positive definite quadratic forms of high dimensionality, eigenvalue computation in Hilbert spaces and fitting of data by superposition of gaussians) show that efficiency is increased and that the algorithm is able to adapt quickly to the current landscape.**

Keywords: **random search, optimization methods.**

1. Introduction

A wide class of problems to be solved in the scientific community can be formulated as the minimization of functions. As an overall best algorithm does not seem to exist, it may be useful to develop a minimization strategy able to adapt itself to different minimization problems. In this paper we will investigate a variant of the stochastic minimization algorithm presented in [3], [21] (the research was done within MAIA, the integrated Artificial Intelligence project under development at IRST [16]). A common feature of random search algorithms is that little or no use of information on the local structure of the function to be minimized is made: the algorithm explores the landscape *blindly*. While this can be useful when the function has a very complicated microstructure, it results in an unnecessary loss of efficiency when the landscape is smooth but anisotropic. A very simple case is given by a long narrow valley such as in the following function:

$$f(x, y) = x^2 + (100\ y)^2 \tag{1}$$

The minimum can be found easily using conjugate gradient techniques. Other techniques, such as steepest descent, widely used because of its simplicity, perform badly in the presence of a long narrow valley [7]. The main reason being that the information used by the algorithm is strictly local (the gradient) and no information is gathered and used during the minimization process itself. The nondeterministic algorithm proposed in [3], [21] exhibits the same shortcomings. In the next section we will show how the random search can be supplemented with a simple, dynamic memory mechanism, enabling the algorithm to maintain a more global view of the function to be minimized and to use it as a predictive tool in the search for the minimum. After describing the memory mechanism, several problems are considered to demonstrate the increased efficiency of the algorithm over the basic variant. Comparison with classic minimization techniques, such as the conjugate gradient and Levenberg-Marquardt methods are also given to gauge the absolute performance of the algorithm.

2. Stochastic Minimization without adaptive memory

The algorithm, which belongs to the general class investigated by Solis and Wets [20], was first proposed by Caprile and Girosi [2] and refined by the authors [21]. A genetic version by Caprile, Girosi and Poggio, which incorporates ideas from the paradigm of *genetic algorithm* [11], [8] appeared in [3]. The algorithm is characterized by a random search whose scope is limited by an adaptive hyper-parallelepiped. The domain of the function to be minimized is decomposed as the cartesian product of n linear subspaces and the number of independently adaptive parallelepiped axis corresponds to the cardinality of the partition. The estimate of the minimum is then given by the sum of n vector components, one for each of the partitions. Each component is modified in turn by the addition of a random vector whose components have a magnitude limited by the corresponding semi-axis. If the function decreases by moving to the perturbed position, the perturbed point is taken as the new estimate and the axis is doubled. If the function does not decrease, a new point is considered by adding to the original point the reversed perturbation (*double shot*). Should this succeed, the probe is moved and the corresponding axis is doubled. In case of failure, the point is left unmoved and the axis is halved. Let us recall two of the variants described in [21]:

- *BATCH* : the cardinality of the partition is 1. The domain of the function is not decomposed: the adaptive hyper-parallelepiped changes its volume but not its shape.
- *SINGLE* : the cardinality of the partition equals the dimensionality of the function domain (coordinate-wise search). The hyper-parallelepiped is adapted along all of the coordinate axes.

The algorithm has been successfully applied to the approximation of functions using an expansion in radial basis functions [2], [3] and to the computation of the minimum eigenvalue of high dimension quadratic forms [21].

It is apparent that the algorithm uses only local information on the minimization landscape. While this is useful in escaping local minima when the function exhibits a *fractal* microstructure, it results in severe inefficiency whenever the downhill region of space surrounding the probe point is a small fraction of the parallelepiped volume. As pointed out in [21] the problem becomes less severe reducing the search volume, but this adversely affects the speed of the algorithm. The expected behaviour in such cases is similar to that of steepest descent when applied to a long narrow valley landscape. In the next section we will show how information can be gathered by the algorithm during the minimization process to bias the random search and increase the efficiency of the algorithm.

3. Stochastic minimization with adaptive memory

The random search algorithm presented in the previous paragraph makes no use of the already effected movements. The introduction of the *double shot* strategy introduces only a very limited, strictly local memory, yet it was found to be extremely effective in improving the search performance [18], [21]. It is quite natural to look for a way of introducing a more global memory. Another important issue is that the algorithm should adapt to the effective amount of information which can actually be used. The mechanism we devised is a sort of adaptive memory. The algorithm keeps track of the successful (downhill) steps and uses them to give an estimate of the direction it should take. As the memory should be adaptive (to cope with varied landscapes) both the number of steps to be considered and the way they are combined to obtain an estimate of the direction should exhibit a dynamic behaviour controlled by the shape of the function to be minimized. We experimented with a simple memory and prediction scheme. The evolution of the minimization process can be naturally timed by the evaluations of the function to be minimized. All of the successful steps are then marked with *time* and stored in a circular buffer. The size of the buffer represents an upper limit on the amount of usable memory. The information stored in the buffer is used for the computation of a biased step $D(t_i)$ in the following way:

$$D(t_i) = \alpha \frac{1}{W} \sum_k \Delta \mathbf{x}_k e^{-\Delta t_{ik}^2/(2\sigma^2)} \qquad (2)$$

$$W = \sum_k e^{-\Delta t_{ik}^2/(2\sigma^2)} \qquad (3)$$

The two variables that characterize the memory of the algorithm are:

- the *amplification* factor α
- the *span* of the memory σ

Due to the shape of the gaussian function the memory can be considered, in practice, as having limited support. Other functions instead of the gaussian could be used (see Section 4.4 where exponential and gaussian memory shapes are compared).

Memory, as represented by the vector (α, σ), is increased or decreased according to whether it helps in minimizing the function or not. The algorithm is described with a flow chart in Fig. 1.

The proposed algorithm belongs to the class of *local search methods* (see [20] for a basic introduction). Iterated application of the algorithm with starting points uniformly distributed over the search domain becomes a method for *global optimization.*

The next sections will present four minimization tasks on which the algorithm has been successfully applied.

4. Performance of the algorithm

4.1 Quadratic forms

Minimization of a positive quadratic form is probably the simplest minimization problem. Nevertheless, minimizing quadratic forms that exhibit an high ratio R of the maximum to the minimum eigenvalues (also called the conditioning number) may prove difficult for the simplest algorithms such as gradient descent. The analysis of the

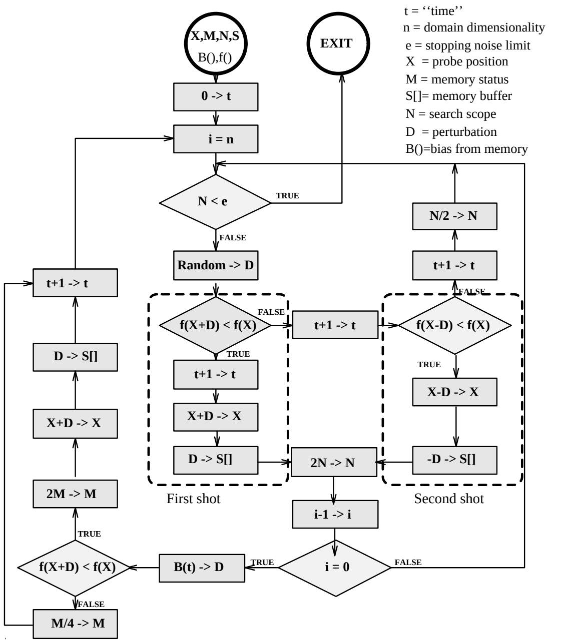

Fig. 1. Flow chart of the random search algorithm with memory.

Caprile-Girosi algorithm presented in [21], shows that the expected performance of this algorithm is also very poor in such cases. The asymmetry of the function to be minimized has a severe impact on the possibility of getting a lower value by randomly looking in a neighborhood.

Typical trajectories of the probe point (using Eq. 1) for the algorithm with and without memory are reported in Fig. 2. The probe without memory moves very slowly because the scope of the random search is kept very small to ensure an high probability of generating a downhill step. The trajectory exhibited by the algorithm with memory is remarkably different: the algorithm accumulates information on the promising direction and finally, when enough information is gathered, a major downhill step can be effected. When the available information is no longer useful the memory is reset and the cycle is restarted.

The correlation between the amount of memory and speed of minimization can be seen in Fig. 3. The different variants of the algorithm have been compared on 30 quadratic forms, obtained from Eq. 1 by random rotation in the plane and the results are reported in Fig. 4.

The performance of Conjugate Gradient (in the Polak-Ribiere form [17]) is added to provide an absolute gauge. Adaptive step size methods converge at a *linear rate* depending on the dimension:

$$N = Kn \qquad (4)$$

where n represents the dimension of the search space, N is the number of function evaluations necessary to get to the minimum with a prescribed accuracy and K is a constant that depends on the algorithm. An heuristic justification is given by Solis and Wets [20]. Different random search techniques can then be compared by means of the linear constant they exhibit. The comparison of the componentwise (SINGLE) modality of our algorithm to the algorithm of Solis and Wets is reported in Fig. 5 for different conditioning numbers R of the quadratic forms (randomly oriented with respect to the coordinate axes). The minimum

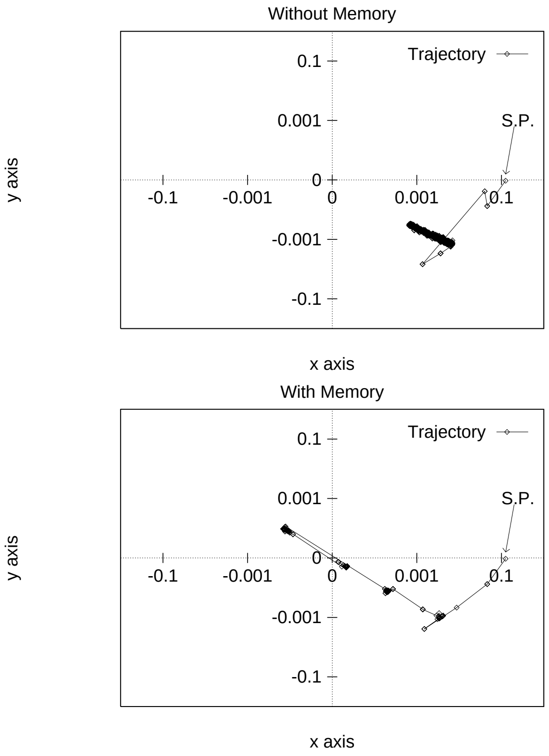

Fig. 2. Typical trajectories exhibited by the algorithm with and without memory during the minimization of $f(x,y) = x^2 + (100y)^2$. Each visited point is marked with a square and linked to the next point of the trajectory with a line segment. The starting point S.P. and the number of iterations is the same for both algorithms.

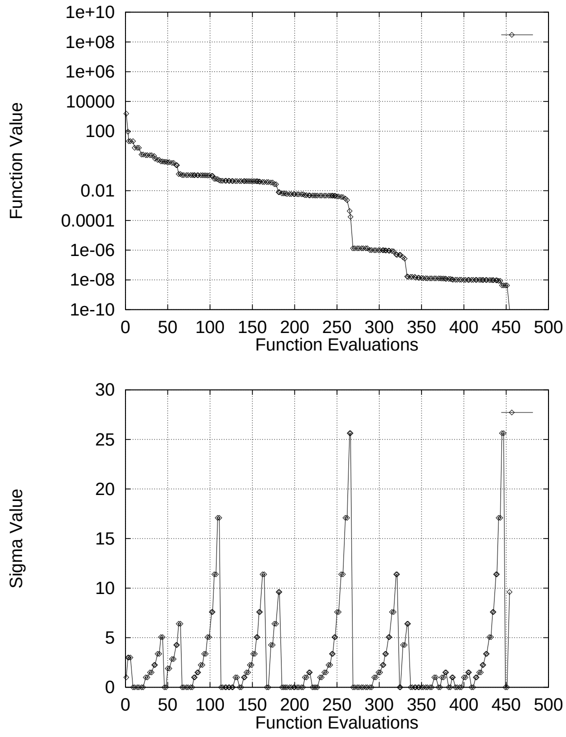

Fig. 3. The minimum value estimate (TOP) and the corresponding amount of memory (BOTTOM) used by the algorithm as represented by σ.

of the quadratic forms was at the origin $\mathbf{O}$: the iterations were stopped when $\|\mathbf{x}_k\| \leq 10^{-3}$ where $\mathbf{x}_k$ represents the location of the probe at the k-th iteration. The proposed algorithm appears to be superior when highly asymmetrical quadratic forms are considered.

4.2 Computation of the Rayleigh coefficient

It is natural to ask how well random minimization can perform whenever a not strictly quadratic form must be minimized. A typical benchmark for minimization routines is the computation of the Rayleigh coefficient ρ of a positive definite quadratic form $\mathbf{A}$ (corresponding to its minimum eigenvalue):

$$\rho = \min_{\mathbf{x}} \mathcal{R}(\mathbf{x}) \tag{5}$$

where

$$\mathcal{R}(\mathbf{x}) = \frac{\mathbf{x}^T \mathbf{A} \mathbf{x}}{\mathbf{x}^T \mathbf{x}} \tag{6}$$

Random search with memory has been tested on this task using high dimension forms ($n = 100$) and its performance is reported in Fig. 6.

4.3 Fitting by superposition of gaussians

The problem of fitting a distribution of data by the superposition of gaussians is common in the field of data analysis, statistical pattern recognition [6], computer vision [10], [19], and recently in artificial intelligence [15].

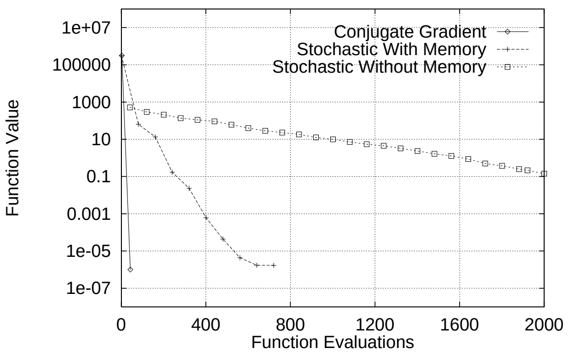

Fig. 4. Average performance of the algorithms in the minimization of a quadratic form. The plot reports the value of the function versus the number of function evaluations (average values over 30 quadratic forms obtained from Eq. 1 by random rotation in the plane). The BATCH variant was used for the stochastic algorithm.

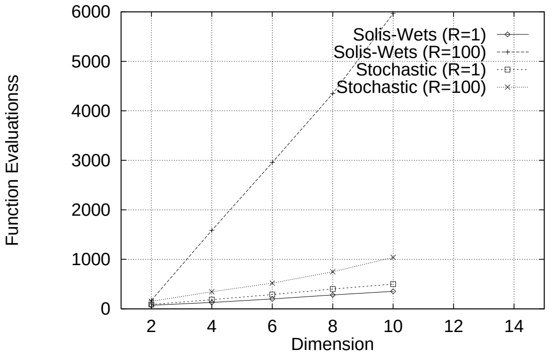

Fig. 5. Comparison of the Solis-Wets algorithm to the SINGLE modality of our algorithm. The plot reports the number of function evaluations necessary to get to the location of the minimum with a prescribed accuracy $||\mathbf{x}|| \leq 10^{-3}$. R represents the ratio maximum/minimum eigenvalue.

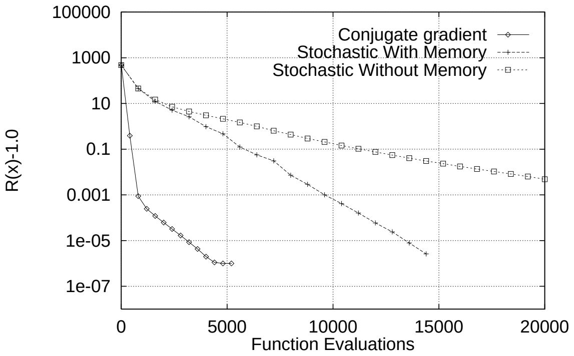

Fig. 6. Average performance of the algorithms in the computation of the Rayleigh coefficient $R(x)$ in $\mathbf{R}^{100}$ with the eigenvalues of the quadratic form evenly spaced from $\lambda_{min} = 1$ to $\lambda_{max} = 1000$. The plot reports the distance from the minimum eigenvalue. The SINGLE variant was used for the stochastic algorithm.

The problem can be formalized in the following way. Let

$$y(x, \mathbf{B}, \mathbf{E}, \mathbf{G}) = \sum_{k=1}^{K} B_k exp \left[- \left(\frac{x - E_k}{G_k} \right)^2 \right] \qquad (7)$$

be the model used to fit the data set $\{(x_i, y_i)\}$. It is then necessary to minimize the following function:

$$\Delta^2(\mathbf{B}, \mathbf{E}, \mathbf{G}) = \sum_i (y_i - y(x_i, \mathbf{B}, \mathbf{E}, \mathbf{G}))^2 \qquad (8)$$

A possible strategy, the Levenberg-Marquardt method [13], [17], is based on the use of gradient information. This method is a mix of first order (gradient) and second order (Hessian matrix) strategies. More recently, a strategy based on a generalization of simulated annealing to continuous problems [19] has been proposed and successfully applied to this kind of problem. In this section we will provide a comparison between the classical LM method and

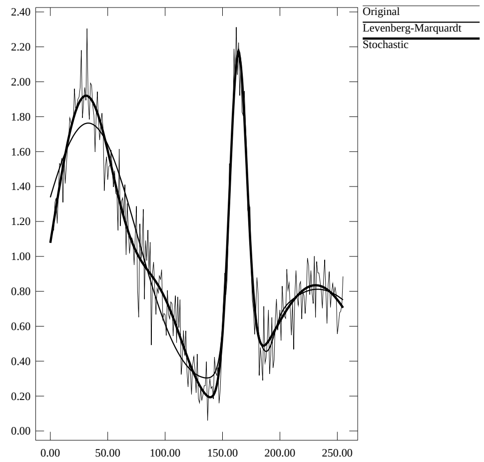

Fig. 7. A typical example used for testing the different strategies.

the two variants of the stochastic algorithm (with and without adaptive memory, within the basic SINGLE mode). To provide a fair basis for the comparison, the algorithms were compared on the same, synthetic data. Multimodal density distributions were generated by choosing at random a set of parameters for a given number of gaussians and by adding random noise to the resulting linear combination. The algorithms were then started with evenly spaced gaussians, whose coefficients were set to the y value at the center position and whose σs (i.e. $G_k/\sqrt{2}$) were set equal to half the width of the center spacing. A set of 100 distributions, specified with $n = 256$ values and normalized to n, was generated for each of three experiments (using 2, 3, 4 gaussians). The performance of the algorithms is gauged by the average Δ^2 and the number of iterations used in the computation.

The computational burden of a single iteration for the two variants of the stochastic algorithm is approximately the same, and smaller than that of the LM method (whose iterations comprise the computation of the Hessian matrix limited to the first order derivatives of $y(x_i, \mathbf{B}, \mathbf{E}, \mathbf{G})$ and the inversion of a $3 * n$ matrix, n being the number of gaussians used for the interpolation).

Algorithm	$K = 2$		$K = 3$		$K = 4$	
	Iter.	Δ^2	Iter.	Δ^2	Iter.	Δ^2
Random	700	7.3	1250	5.3	1600	4.5
Ran. & mem.	500	7.4	900	5.2	1100	4.6
L-M	20	12.8	20	8.7	25	9.1

TABLE I

Iterations number and average Δ^2 for the different algorithms.

Inspection of Table I reveals that the variant with memory produces solutions of the same quality of the algorithm without memory with a lower number of iterations. On the average, both variants of the stochastic algorithm provide better solutions than the LM method. This is due to the latter algorithm getting stuck into wrong local minima. The higher computational weight of random search is balanced by the increased robustness of the algorithm to spurious local minima. In the next section we briefly introduce a task in which the computation of gradient information can be considered unfeasible. In such a case, algorithms based on gradient (or Hessian) information are useless while stochastic algorithms can still be fruitfully applied.

4.4 Searching for the minimum eigenvalue of a singular differential operator

Another successful, more complex, application of the proposed algorithm is given by the computation of the minimum eigenvalue λ_{min} (and the companion eigenfunction ψ_{min}) of a differential operator H (usually self-adjoint and definite positive) in a Hilbert space $\mathcal{H}$.

In fact, this problem is equivalent to the minimization of the Rayleigh functional $R : \mathcal{H} \to \mathbf{R}$ given by:

$$R[\psi] = \frac{(\psi, H\psi)_{\mathcal{H}}}{(\psi, \psi)_{\mathcal{H}}} \tag{9}$$

with $\psi \in \mathcal{H}$ and $(\cdot,\cdot)_{\mathcal{H}}$ is the hermitean product of $\mathcal{H}$. This task can be performed numerically by choosing a subset $\psi_1, \ldots, \psi_n$ of an orthonormal basis of $\mathcal{H}$: the higher n the more accurate the computations will be. Any $\psi \in \mathcal{H}$ can then be approximated by the following expansion:

$$\psi \approx \sum_{i=1}^{n} w_i \psi_i \tag{10}$$

and the functional $R[\psi]$ by the matrix $A_{ij} = (\psi_i, H\psi_j)$ so that:

$$\lambda_{min} = \min_{\psi\in\mathcal{H}} R[\psi] \approx \min_{C^n} \frac{x^\dagger A x}{x^\dagger x} \tag{11}$$

where $x^\dagger$ is the hermitean conjugate of x.

The problem is then reduced to the form described in §4.2 and the algorithm described therein could be applied although more efficient techniques are available [9]. However, when the operator H exhibits some pathology (e.g. it contains poles), traditional techniques usually fail either because of numerical instabilities or because of the prohibitively high value of n necessary to describe accurately the behaviour of H in the neighborhood of the poles. As an example, let us consider the following differential operator:

$$H(\frac{d}{dx}, x, m) = -\frac{d^2}{dx^2} + \frac{1}{4}\left[1 - \frac{1}{sn^2(x,m)}\right] \tag{12}$$

where $m \in [0,1]$, sn is the Jacobian sine-amplitude function [1]:

$$\mathrm{sn}(x,m) = \sin[\theta(x,m)] \tag{13}$$

and $\theta(x,m)$ (the so called *amplitude*) is obtained by inverting

$$x = \int_0^\theta \frac{dt}{(1 - m\sin^2 t)^{1/2}} \tag{14}$$

We recall that $\mathrm{sn}(x,m)$ is a periodic function with period $4K(m)$ where $K(m)$ is:

$$K(m) = \int_0^{\pi/2} \frac{dt}{(1 - m\sin^2 t)^{1/2}} \tag{15}$$

The resulting eigenvalue problem:

$$H(\frac{d}{dx}, x, m) = \lambda(m)\psi(x,m) \tag{16}$$

is of relevance in theoretical physics. It describes the energy spectrum of a free particle on a 2-dimensional Riemannian manifold which stems from a field theory called *nonlinear σ-model* [5]. In this context we are interested in finding the minimum eigenvalue λ as a function of m. An analytical solution of Eq. 16 is unknown [4] and a numerical approach is needed.

Because $\mathrm{sn}(x,m)$ has poles in the points $x = 0, x = 2K(m)$ many traditional techniques can solve the problem only with low accuracy or within a very limited range of m. The direct application of our algorithm to the minimization of the associated Rayleigh functional of Eq. 9 would present the same problems. These difficulties can be removed by the following transformation:

$$\psi(x) = H_1(\frac{d}{dx}, x, m) = \sqrt{\mathrm{sn}(x,m)}\,(1 - \mathrm{scd}(x,m)\frac{d}{dx})\,\phi(x) \tag{17}$$

where:

$$\begin{aligned}
\mathrm{cn}(x,m) &= \cos(\theta(x,m)) \\
\mathrm{dn}(x,m) &= \sqrt{1 - m\sin^2\theta(x,m)} \\
\mathrm{scd}(x,m) &= \mathrm{sn}(x,m)\,\mathrm{cn}(x,m)\,\mathrm{dn}(x,m)
\end{aligned}$$

so that Eq. 16 becomes:

$$H_2(\frac{d}{dx}, x, m)\phi(x) = \lambda(m) H_1(\frac{d}{dx}, x, m)\phi(x) \tag{18}$$

where:

$$\begin{aligned}
H_2 \quad (& \frac{d}{dx}, x, m) = \sqrt{\mathrm{sn}(x,m)}\{[(2+m) - 3m\,\mathrm{sn}^2(x,m)] + \\
- \quad & [1 + 25(m+1) - 63m\,\mathrm{sn}^2(x,m)]\mathrm{scd}(x,m)\frac{d}{dx} + \\
- \quad & [-8 + 20(m+1)\mathrm{sn}^2(x,m) - 28m\,\mathrm{sn}^4(x,m)]\frac{d^2}{dx^2} + \\
+ \quad & 4\mathrm{scd}(x,m)\frac{d^3}{dx^3}\}
\end{aligned}$$

We have then transformed the eigenvalue problem of Eq. 16 into the generalized eigenvalue problem of Eq. 18 which has third-order differential operators but no poles. This new problem can not be solved with the traditional techniques because they rely on the particular form of the Rayleigh functional which is no longer preserved. On the contrary, stochastic search can still be applied to the minimization of the following functional:

$$T[\phi] = \parallel H_2\phi - \frac{(H_1\phi, H_2\phi)}{(H_1\phi, H_1\phi)} H_1\phi \parallel \tag{19}$$

because it is known that $T[\phi] = 0$ if and only if ϕ is a solution of 18 [9], [12] To find λ_{min} as function of m we can use the following algorithm:

1. choose the set of the harmonic functions as a basis for the Hilbert Space (Fourier basis):

$$\xi_j(x) = e^{-i\frac{2\pi j}{4K(m)}x} \quad j = 1, \ldots, n$$

   so that:

   (a) the vector $\mathbf{w}$ with complex components w_j (j=1..n) represents the function:

$$\phi(x) = \sum_{j=1}^{n} w_j \xi_j(x) \tag{20}$$

(b) we can efficiently compute the values of ϕ at points

$$x_j = \frac{2\pi}{n}\frac{j}{4K(m)} \qquad (21)$$

using a Fast Fourier transform algorithm [14],

(c) the derivative terms contained in H_1 and H_2 can be computed directly in the Fourier Space, because, from Eq. 20, we have

$$\frac{d}{dx}\phi(x) = \sum_{j=1}^{n} w_j \frac{2\pi}{4K(m)} j\xi_j(x) \qquad (22)$$

This allows the fast computation of $H_1\phi$ and $H_2\phi$ without evaluating the matrix representations of H_1 and H_2,

2. Given the set $\{m_i\}$ such that $0 < m_1 < m_2, ... < m_k \leq 1$, compute $\phi(x, m_k)$ with algorithm of Fig. 1. The starting point for m_1 is given by the known solution for $m = 0$ of Eq. 18, $\phi(x) = 1$ (and $\lambda_{min} = -2$), while for m_k with $k > 1$ the starting point is given by $\phi(x, m_{k-1})$.

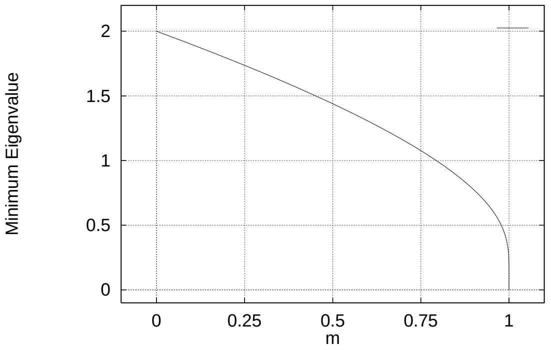

Fig. 8. Plot of $-\lambda_{min}(m)$ as computed by our algorithm.

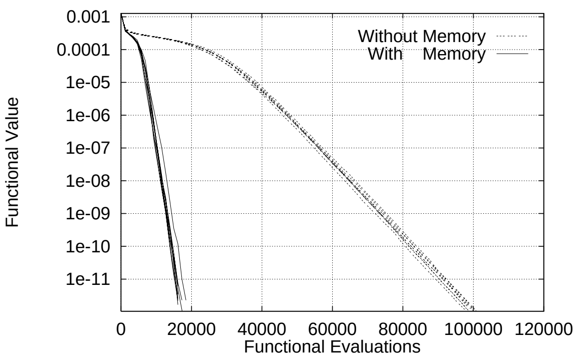

Fig. 9. Convergence of the stochastic algorithms, with and without memory, on 8 different runs.

We tested this algorithm using $n = 64$ (there is no significant increase in accuracy using higher values of n). The plot of λ_{min} as a function of m in the interval $[0, 1]$ is reported in Fig. 8. In Fig. 9 the dynamics of the algorithm

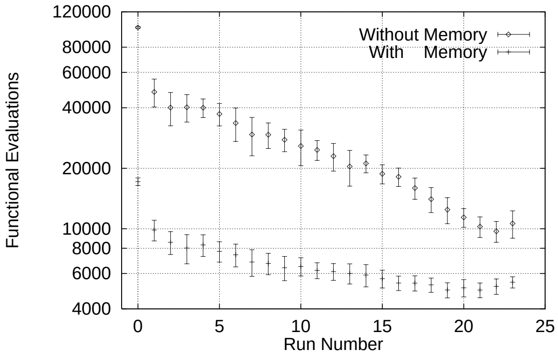

Fig. 10. Number of functional evaluations for convergence at different values of m using as starting point the previously computed configuration. Data are gathered from 64 different sweeps over m.

with(out) memory are reported. The number of evaluations of T to reduce $T(\phi)$ under a threshold of 10^{-12} is plotted in Fig. 10 (a single function evaluation takes a few milliseconds on a Data General AViiON 300 workstation). Note the improvement of the algorithm with memory.

In these tests a variant of the algorithm (B. Caprile and F. Girosi, personal communication) was used: the algorithm makes use of the direction of the *polarization* vector (D of eq. 2) to effect the first random choice:

$$\|\text{random}\| \frac{D}{\| D \|} \rightarrow D \qquad (23)$$

An appropriate name for the modification would be *copysign*. This turns out to be a (minor) improvement as the test $f(x + D) < f(x)$ (see *First shot* block of Fig. 1) is more frequently true thereby reducing computation. The distribution of the number of successful first-shot trials minus the number of successful second-shot trials is clearly symmetric around 0 if no sign information is used while it shifts towards an higher value of first-shot trials using the copysign variant (see Fig 11).

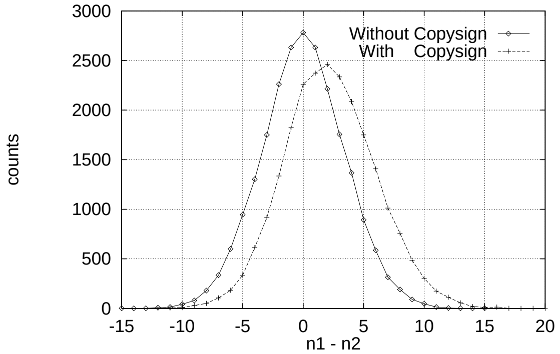

Fig. 11. Improvement using the copysign variant. Frequency distribution of $n1 - n2$ where $n1$ represents the number of successful first shots while $n2$ represents the number of successful second shots.

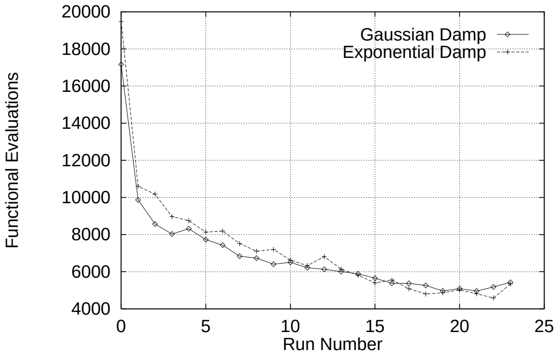

Fig. 12. Comparison between the number of functional evaluations with gaussian and exponential damping functions. The results on a complete sweep on 24 values of m in [0,1] are shown.

Finally, we should note that the choice of gaussian damping for the memory mechanism is not critical. We experimented with a pure exponential damping getting similar results. This suggests that the dynamic mechanism used to maintain the *necessary depth* is sufficiently strong to cope with the effects deriving from the choice of different damping functions (see Fig. 12).

5. Conclusions

A random search algorithm with adaptive memory has been presented which is characterized by the use of an adaptive gaussian memory for biasing the exploration. The algorithm has been compared to its variant without memory and to traditional techniques in several minimization tasks. Of particular interest is the application to the computation of the minimum eigenvalue of a singular differential operator on a Hilbert space for which traditional techniques perform badly.

Acknowledgements

The authors would like to thank Prof. T. Poggio and B. Caprile for their suggestions. G. T. would like to thank warmly E. Onofri and V. Fateev for stimulating discussions.

References

[1] M. Abramowitz and I. A. Stegun, editors. *Handbook of mathematical functions.* Dover, 1970.

[2] B. Caprile and F. Girosi. A Nondeterministic Minimization Algorithm. A.I. Memo No. 1254, Massachusetts Institute of Technology, 1990.

[3] B Caprile, F. Girosi, and T. Poggio. A Nondeterministic Method for Multivariate Functions Minimization. In E. R. Caianiello, editor, *Parallel Architectures and Neural Networks,* pages 37–44. World Scientific, 1991.

[4] A. Erdelyi and al. *Higher trascendental functions,* volume 3. McGraw-Hill, 1953.

[5] V. A. Fateev, E. Onofri, and Al. Zamolodchikov. Integrable deformations of o(3)-σ-model. Technical Report 92-46, Laboratoire de Physique Theorique Hautes Energies, Universite' Paris 6, 1992.

[6] K. Fukunaga. *Introduction to Statistical Pattern Recognition.* Academic Press, 1972.

[7] P. E. Gill, W. Murray, and M. H. Wright. *Practical Optimization.* Academic Press Inc., London, 1981.

[8] D. E. Goldberg. *Genetic Algorithms.* Addison-Wesley, Reading, MA, USA, 1989.

[9] G. H. Golub and C. F. Van Loan. *Matrix computations.* Johns Hopkins University Press, 1989.

[10] R. C. Gonzalez and P. Wintz. *Digital Image Processing.* Addison-Wesley, 1987.

[11] J. H. Holland. *Adaptation in Natural and Artificial Systems.* University of Michigan Press, Ann Arbor, 1975.

[12] B. Kagstrom and A. Ruhe. *Matrix pencils,* volume 973 of *Lecture Notes in Mathematics.* Springer-Verlag, 1983.

[13] D. W. Marquardt. *J. Soc. Ind. Appl. Math.,* 11:431–441, 1963.

[14] H. J. Nussbaumer. *Fast Fourier Transform and Convolution Algorithms.* Springer-Verlag, 1982.

[15] T. Poggio and F. Girosi. Regularization algorithms for learning that are equivalent to multilayer networks. *Science,* 247:978–982, 1990.

[16] T. Poggio and L. Stringa. A Project for an Intelligent System: Vision and Learning. *Internation Journal of Quantum Chemistry,* 42:727–739, 1992.

[17] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipes in C.* Cambridge University Press, 1988.

[18] G. Schrack and M. Choit. Optimized Relative Step Size Random Searches. *Mathematical Programming,* 10:230–244, 1976.

[19] W. Snyder, G. Bilbro, A. Logenthiran, and S. Rajala. Optimal thresholding - a new approach. *PRL,* 11:803–810, 1990.

[20] F. J. Solis and R. J-B. Wets. Minimization by Random Search Techniques. *Mathematics of Operations Research,* 6:19–30, 1981.

[21] G. Tecchiolli and R. Brunelli. On random minimization of functions. *Biological Cybernetics,* 65:501–506, 1991.