(12)

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AIM 1254 (C.B.I.P. 1254)	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) A Nondeterministic Minimization Algorithm		5. TYPE OF REPORT & PERIOD COVERED memorandum
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Bruno Caprile and Federico Girosi		8. CONTRACT OR GRANT NUMBER(s) S1-801534-2 DACA76-85-C-0010 N00014-85-K-0124
9. PERFORMING ORGANIZATION NAME AND ADDRESS Artificial Intelligence Laboratory 545 Technology Square Cambridge, Massachusetts 02139		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Advanced Research Projects Agency 1400 Wilson Blvd Arlington, Virginia 22209		12. REPORT DATE September 1990 13. NUMBER OF PAGES 8
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Office of Naval Research Information Systems Arlington, Virginia 22217		15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Distribution of this document is unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

None

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

optimization
learning
stochastic algorithms

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The problem of minimizing a multivariate function is recurrent in many disciplines as Physics, Mathematics, Engeneering and, of course, Computer Science. Both deterministic and nondeterministic algorithms have been devised to perform this task. It is common practice to use the nondeterministic algorithms when the function to be minimized is *not smooth* or depends on

(continued on back)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Block 20 continued:

binary variables, as in the case of combinatorial optimization. In this paper we describe a simple nondeterministic algorithm which is based on the idea of adaptive noise, and that proved to be particularly effective in the minimization of a class of multivariate, *continuous valued, smooth functions*, associated with some recent extension of regularization theory by Poggio and Girosi (1990). Results obtained by using this method and a more traditional gradient descent technique are also compared.

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
ARTIFICIAL INTELLIGENCE LABORATORY
and
CENTER FOR BIOLOGICAL INFORMATION PROCESSING
WHITAKER COLLEGE

A.I. Memo No. 1254
C.B.I.P. Paper No. 58

September 1990

A Nondeterministic Minimization Algorithm

Bruno Caprile and Federico Girosi

Abstract

The problem of minimizing a multivariate function is recurrent in many disciplines as Physics, Mathematics, Engeneering and, of course, Computer Science. Both deterministic and nondeterministic algorithms have been devised to perform this task. It is common practice to use the nondeterministic algorithms when the function to be minimized is *not smooth* or depends on *binary variables*, as in the case of combinatorial optimization. In this paper we describe a simple nondeterministic algorithm which is based on the idea of adaptive noise, and that proved to be particularly effective in the minimization of a class of multivariate, *continuous valued, smooth functions*, associated with some recent extension of regularization theory by Poggio and Girosi (1990). Results obtained by using this method and a more traditional gradient descent technique are also compared.

This paper describes research done mainly at I.R.S.T. in Trento, Italy, and also within the Center for Biological Information Processing, in the Department of Brain and Cognitive Sciences, and at the Artificial Intelligence Laboratory. This research is sponsored by a grant from the Office of Naval Research (ONR), Cognitive and Neural Sciences Division; by the Artificial Intelligence Center of Hughes Aircraft Corporation (S1-801534-2). Support for the A. I. Laboratory's artificial intelligence research is provided by the Advanced Research Projects Agency of the Department of Defense under Army contract DACA76–85–C–0010, and in part by ONR contract N00014-85-K-0124.

1 Introduction

In many areas of science, problems of interest, both practical and theoretical, consist in determining a configuration of a set of parameters that is "optimal" with respect to some cost function. Therefore, a variety of minimization algorithms have been developed for finding the global minimum of a multivariate function. In this short note we describe an algorithm that we have successfully used to solve a class of minimization problems that arise in the study of the problem of learning from examples.

The cost functions that we consider, an explicit form of which will be presented in the next section, are defined in high dimensional spaces and usually show many local minima. Standard descent techniques, as gradient descent or conjugate gradient, (Polak, 1971; Acton, 1970; Press et al., 1987), are of limited usefulness in this case, since they cannot escape local minima. Moreover, although the analytical expression of the gradient is available, its computation can be very time consuming, and it would be preferable to avoid it. Therefore, it may be better to use nondeterministic techniques, of the Metropolis type, (Metropolis et al., 1953), that usually do not require the computation of derivatives, and are more suited to deal with the problem of local minima. A successful nondeterministic method is simulated annealing (Kirpatrick et al., 1983), which however requires the difficult choice of an annealing schedule, and has a high computational cost. The algorithm that we developed is nondeterministic, does not require any annealing and does not have a high computational cost. Albeit we are not guaranteed to found the global minimum, experimental results show that, for the class of functions we are considering, good local minima are usually found.

The algorithm we implemented is similar in spirit to many heuristic minimization algorithms, like A^* (Hart et al., 1968; Shapiro, 1987), the algorithm of Kernighan and Lin (1970, 1973), and the algorithms described by Pearl (1984). Our algorithm is also similar in spirit to some "evolutionary" optimization algorithms, like the ones described for example by Wang (1987) and Fogel (1988). It is interesting to notice that most of these algorithms have been developed and used to solve *combinatorial* optimization problems, like the Traveling Salesman Problem (Lin and Kernighan, 1973), and rarely used to minimize smooth functions. We tested our algorithm in a variety of cases, all belonging to the class of minimization problems which will be described in the next section, and we compared the results with the ones given by a

standard technique of gradient descent with adaptive step. In all cases considered it turns out that the nondeterministic algorithm finds the best local minima. Preliminary experiments suggest that this could hold true not only for the particular class of functions we are considering, but also for a wider class of problems.

2 Regularization Networks and Minimization Problems

Recently, Poggio and Girosi (1989, 1990) described how standard mathematical techniques can be used to approach the problem of learning. In fact, learning an input-output mapping from a set of examples can be regarded as synthesizing an approximation of a multi-dimensional function, that is solving the problem of surface reconstruction from sparse data. Poggio and Girosi used a variational approach, based on regularization theory, (Tikhonov and Arsenin, 1977; Morozov, 1984; Bertero, 1986) and showed that this problem is equivalent to find the optimal "weights" of a particular type of network with one layer of hidden units, called *regularization network*. Therefore the problem of learning becomes equivalent to the following minimization problem:

Minimization problem for regularization networks: *Let $H[f]$ be the functional*

$$H[f] = \sum_{i=1}^{N} (y_i - f(\mathbf{x}_i))^2 + \lambda \| P_{\mathbf{y}} f \|^2 , \tag{1}$$

where $\{(\mathbf{x}_i, y_i) \in R^d \times R\}_{i=1}^N$ is a given sparse data set, $P_{\mathbf{y}}$ is a linear operator that is radially symmetric in the variable $\mathbf{y} = \mathbf{W}\mathbf{x}$, $\mathbf{W}$ is a $d \times d$ matrix, $\| \cdot \|$ is a norm on the function space which $P_{\mathbf{y}} f$ belongs to, and λ is a positive real number. Having defined the function

$$f^*(\mathbf{x}) = \sum_{\alpha=1}^{n} c_\alpha G(\|\mathbf{x} - \mathbf{t}_\alpha\|_{\mathbf{W}}^2) , \tag{2}$$

find $\{c_\alpha\}_{\alpha=1}^n$, $\{\mathbf{t}_\alpha\}_{\alpha=1}^n$ and $\mathbf{W}$ such that $H[f^*]$ is minimum.

Here G is the Green's function of $\hat{P}P$, $\hat{P}$ indicating the adjoint of the operator P, and the function f^* that minimizes the functional H is the solution of the approximation problem. Equations (1) and (2) define the class of cost functions which our algorithm has been applied to.

3 The Algorithm

This section is dedicated to a rather detailed presentation of the minimization algorithm that we propose. We will describe, first, the elemental step of the procedure, i.e. what can be considered its *core*, and later we will outline how loops of elemental steps are arranged to form the *outer structure* of the algorithm.

In what follows, the outcome of the random extraction of a real number within the interval $[a, b]$ will be indicated with the scripture $random(a, b)$, and the set of the natural numbers greater than 0 and smaller than $n + 1$ will be indicated with the symbol I_n.

Let $g = g(\mathbf{x})$ be a multivariate real function defined in some domain, A, of $\mathcal{R}^n$. Let $\mathbf{x}$ be an internal point of A, and let $\mathcal{P} = \{P_1, \ldots, P_k\}$ be a partition of I_n. Let us set, for each element $P_i \in \mathcal{P}$, a positive number ω_i, and let Ω be the set of all such numbers. The set Ω, for reasons that will be clear in a moment, is called *the noise list.*

An elemental step consists of a series of k perturbations of the current point, each of them obtained by adding *random noise* only to elements that belong to a certain subset of its components. In particular, the j-th perturbation of any series will concern elements belonging to P_j only. In this sense, partition $\mathcal{P}$ realizes a grouping of variables that, for some reason, may be considered "similar". Any perturbation is then accepted - and the current point consequently updated - if, and only if, the value taken by the function g at the perturbed point is smaller than the value of g at the current point. The amplitude of the noise relative to the group of components of the current point that the perturbation has modified is either doubled, when the perturbation is accepted, or it is halved when the converse case occurs. More formally :

- g_c is set equal to $g(\mathbf{x})$;
- for each element P_i of partition $\mathcal{P}$,
  - a n-dimensional vector, $\mathbf{v}$, is generated according to the following

    $$v_j = \begin{cases} random(-\omega_i, \omega_i) & \text{if } j \in P_i \\ 0 & otherwise \end{cases}$$

    where v_j indicates the j-th component of $\mathbf{v}$;
  - g_a is set equal to $f(\mathbf{x} + \mathbf{v})$;
  - if $g_a < g_c$, then
    - g_c is set equal to g_a ;
    - $\mathbf{x}$ is set equal to $\mathbf{x} + \mathbf{v}$;
    - ω_i is set equal to $2\omega_i$;

    else
    - ω_i is set equal to $\omega_i/2$;

Performing a single step of the algorithm may result either in the update of the noise list or in the update of the noise list *and* the current point. The whole algorithm consists, in essence, in performing a series of such steps, keeping track, meanwhile, of the evolution of the current point as well as the evolution of the noise list.

Since elements of the noise list dictate the maximum amplitude of perturbations that will occur in the following step, the noise list undergoes, during the minimization process, a fairly intricated dynamics. However, as the current point approaches a point of minimum for the function, our expectation is that the elements of the noise list become smaller and smaller, although their rates of convergence to zero may differ considerably from each other.

Given these considerations it seems to be sensible to stop the minimization process once that elements of the noise list have reached values that can be considered small, and no further relevant evolution is expected to take place.

How to evaluate whether the values contained in the noise list are small enough is now matter of choice, and we propose the following criterion :

elements of the noise list are small enough if

$$\omega_i \quad < \quad \eta \quad ,\forall \; i \; \in \; I_k \; ,$$

where η is a given threshold.

Once that the process has stopped, and the final current point is returned, there is no way, in general, to ascertain how close this point is to the actual point of global minimum for the function. Hence, the only possibility that we are given consists in restarting the procedure using a different initial point. However, if we believe that the point we are looking for is not very far from the point that we have reached, we may want to restart the procedure *nearby* - both in terms of initial point and noise list - the result we have just reached. A practical way to perform this task is to generate the new noise list by multiplying the elements of the old one by a given constant. The new starting point can be generated, now, by perturbing the old one according to the new noise list.

Results obtained by restarting and performing several times the minimization procedure are then collected, and the point where the function has reached its *best* value is considered the outcome of the whole algorithm.

4 Experimental Results

Our algorithm has been tested on the problem of finding the function, belonging to the class expressed by eq. (1), which minimizes the functional of eq. (2). As poggio and Girosi showed (1989, 1990), solution to this minimization problem best approximates data y_i, $i = 1, \ldots, n$. Each experiment consisted, then, of the following steps:

- a data set containing N elements was generated by sampling at random a given multivariate function $h = h(\mathbf{x})$;
- the number n of eq. (2) was set, and the number of parameters that function f^* depended on consequently fixed;
- a minimum for the function $H[f^*]$ - considered as a function of $\{c_\alpha\}_{\alpha=1}^n$, $\{\mathbf{t}_\alpha\}_{\alpha=1}^n$ and $\mathbf{W}$ - was sought;
- the mean square differences between the function h and its approximation was computed over a randomly generated test set.

In order to evaluate the performances of our algorithm, we decided to compare results obtained by using it with the ones obtained by using the fairly classical method of gradient descent with adaptive step. One hundred iterations of gradient descent and two thousand iterations of our method both guaranteed a good degree of convergence and the same computing time for the two methods.

In Table (1) results showing the *average* behavior of the algorithms, that is the average of the mean square error on several experiments, are collected.

In Table (2) the best performances of the algorithms, that is the lowest mean square approximation error obtained in each group of experiments, are reported: in this case the nondeterministic algorithm performs better of gradient descent, since it has more chances to attain the global minimum.

Number of variables	19	34	49
Gradient Descent	0.11	0.11	0.09
Adaptive Noise	0.18	0.13	0.10

Table 1 : Errors affecting surface reconstruction obtained by using the method that we propose and the gradient descent with adaptive step method are compared. Each column shows the averages of the mean square approximation errors coming from a set of 10 experiments. The function to be reconstructed was $h(x,y) = e^{-(0.8x+1.2y)}cos5xsin3y$ *and the mean square approximation error was computed over a set of 10,000 points. Columns 1, 2, 3 refer to groups of experiments performed by setting number n of eq. (2) to 5, 10, and 15 respectively. Therefore, the functions to be minimized depended on 19, 34, and 49 variables. For each experiment, 100 iterations of gradient descent with adaptive step method, and 2000 iterations of our method were performed.*

Number of variables	19	34	49
Gradient Descent	0.045	0.057	0.044
Adaptive Noise	0.015	0.027	0.038

Table 2 : As Table (1), except that it shows in each column the lowest mean square approximation error over a group of 10 experiments.

Experience we have been collecting by performing hundreds of running of both algorithms leads us to believe that the one exhibited in the experiments reported is their own typical behavior.

5 Remarks

The first aspect of oour algorithm to be noticed is that it does not require any computation of the value of the gradient. This crucial, of course, whenever an explicit expression of the gradient is either unavailable or computationally expensive. For example, computation of the value of the gradient for the functions we have performed experiments with, was 10 or 20 times more time consuming than computation of the value of the function.

Another aspect of the method is that, unlike many minimization methods, it does not rest on any particular assumptions about the function to be minimized and the structure of its minima. By giving credit to those perturbations that yield a decrease of the value of the function, we perform an estimate of the gradient that, though very rough, may capture the essential trend of the function. In situations of interest of us, this line of action may be recommended, since what we want to avoid is to spend time in computing a quantity - the gradient - that may end up telling little about location of points of minimum for the function.

Let us finally remark that the introduction of the partition list allows us to group the variables according to their typical magnitudes, which, due to the different role variables play, may differ considerably.

Acknowledgements. The authors are indebted with Tomaso Poggio for his suggestions and his constant support. One of the authors (F.G.) is also grateful to Shimon Edelman for helpful discussions and remarks. Cesare Furlanello read the manuscript and made useful comments.

References

[1] F.S. Acton. *Numerical Methods that Work.* Harper and Row, New York, 1970. Chapter 17.

[2] M. Bertero. Regularization methods for linear inverse problems. In C. G. Talenti, editor, *Inverse Problems.* Springer-Verlag, Berlin, 1986.

[3] W.H. Press et al. *Numerical Recipes: tha Art of Scientific Computing.* Cambridge University Press, Cambridge, 1987.

[4] D.B. Fogel. An evolutionar approach to the travelling salesman. *Biological Cybernetics*, 60(2):139-144, 1988.

[5] P.E. Hart, N.J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Trans. on Systems, Man and Cybernetics*, SSC-4(2):100–107, 1968.

[6] B.W. Kernighan and S. Lin. An efficient heuristic procedure for partitioning graphs. *The Bell System Technical Journal*, 49(2):291–307, February 1970.

[7] B.W. Kernighan and S. Lin. Heuristic solution of a signal design optimization problem. *The Bell System Technical Journal*, 52(7):1145–1159, September 1973.

[8] S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi. Optimization by simulated annealing. *Science*, 220:219–227, 1983.

[9] S. Lin and B.W. Kernighan. An effective heuristic for the travelling salesman problem. *OR*, 21:498–516, 1973.

[10] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller. Equation of state calculations by fast computing machines. *J. Phys. Chem*, 21:1087, 1953.

[11] V.A. Morozov. *Methods for solving incorrectly posed problems.* Springer-Verlag, Berlin, 1984.

[12] J. Pearl. *Heuristics: Intelligent Search Strategies for computer problem solving.* Addison-Wesley, Reading, MA, 1984.

[13] T. Poggio and F. Girosi. A theory of networks for approximation and learning. A.I. Memo No. 1140, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1989.

[14] T. Poggio and F. Girosi. Extension of a theory of networks for approximation and learning: dimensionality reduction and clustering. A.I. Memo 1167, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1990.

[15] T. Poggio and F. Girosi. Networks for approximation and learning. *Proceedings of the IEEE*, 78(9), September 1990.

[16] T. Poggio and F. Girosi. A theory of networks for learning. *Science*, 247:978–982, 1990.

[17] E. Polak. *Computational Methods in Optimization*. Academic Press, New York, 1971.

[18] S.C. Shapiro. *Encyclopedia of Artificial Intelligence*, volume 1. John Wiley and Sons, New York, 1987.

[19] A. N. Tikhonov and V. Y. Arsenin. *Solutions of Ill-posed Problems*. W. H. Winston, Washington, D.C., 1977.

[20] Q. Wang. Optimization by simulating molecular evolution. *Biological Cybernetics*, 57(1/2):95–101, 1987.