Some nonlinear optimization methods

Jos van Trier

ABSTRACT

Although most nonlinear optimization methods use linear approximations to find search directions in the model space, their search for the minimum is nonlinear: the nonlinear objective function is recalculated at each step in the search. These nonlinear methods can be divided in three classes: methods that only use function evaluations, methods that also use first derivatives or gradients, and methods that require function, gradient and second derivative information. I tested several methods on Rosenbrock's function, a test function often used in optimization literature.

INTRODUCTION

My main reason for testing optimization methods is to find out how well suited they are for problems with a small set of parameters. Therefore I use Rosenbrock's function, a function of only two parameters, to test each method.

I will not describe any new theory in this paper; I will just give a brief summary of each method, without deriving the mathematics. A extensive survey of optimization methods can be found in Gill et al (1981).

OPTIMIZATION

Linear vs. Nonlinear

Traditionally, inverse problems in geophysics are formulated in terms of a linear relationship between model and data:

$$\mathbf{G}\mathbf{m} = \mathbf{d}, \tag{1}$$

where $\mathbf{m}$ is the model vector, $\mathbf{d}$ the data vector and $\mathbf{G}$ a matrix. The conjugate-gradient method was originally designed to solve this problem, because existing methods for inverting the matrix $\mathbf{G}$ were expensive for many model parameters. Its main advantage over previous methods, though, is that it is an iterative method. In each iteration, the squared difference between data $\mathbf{d}$ and modeled data $\mathbf{G}\mathbf{m}$ is minimized. Although the exact solution can be found after n iterations (with n the number of model parameters), the error is often small enough to end the search after only a few iterations.

This *linear* conjugate-gradient method is described in Claerbout's tutorial on conjugate gradients (Claerbout (1985)). The conjugate gradient method can also be used in cases when there is no linear relationship between model and data, i.e., when it is impossible to construct the matrix **G**. Although in each iteration linear approximations to the gradient are used, the search for the minimum is nonlinear: the nonlinear function is recalculated at each step in the search.

Several other nonlinear methods use this approach. In the next section section I will discuss some of them. All the methods discussed here are so-called direction methods; I will not consider random-search methods, such as the Monte Carlo method.

Nonlinear Optimization

Optimization methods find the minimum of a objective function F. If we take some particular point **P** as the origin of the coordinate system with coordinates **x**, then the function can be approximated by its Taylor series

$$\begin{aligned} F(\mathbf{x}) &= F(\mathbf{P}) + \sum_i \frac{\partial F}{\partial x_i} x_i + \frac{1}{2} \sum_{i,j} \frac{\partial^2 F}{\partial x_i \partial x_j} x_i x_j + \ldots \\ &\approx c - \mathbf{b} \cdot \mathbf{x} + \frac{1}{2} \mathbf{x} \cdot \mathbf{A} \cdot \mathbf{x}, \end{aligned} \tag{2}$$

where **x** is the n-dimensional vector $(x_1, x_2, \ldots, x_n)^T$, **A** is the second derivative or Hessian matrix, **b** is the vector opposite to the gradient vector and c is a constant. Many optimization methods use this approximation; they differ in how they calculate the gradient vector and/or the Hessian matrix, and using this distinction, we can divide them into three classes:

1. methods that require function, gradient and second derivative information;
2. methods that require gradient information or first derivatives;
3. search methods that use function evaluations only.

The different sorts of optimization are listed here in reverse order of simplicity. In terms of theory however, the first method is easier to explain than the last one, and therefore the different methods are discussed in this order in the following sections.

Second-derivative optimization

Using equation (2), we can find the minimum $\mathbf{x}_m$ of F by setting its derivative with respect to **x** to zero:

$$\mathbf{x}_m = \mathbf{A}^{-1} \cdot \mathbf{b} \tag{3}$$

If the matrix **A** is known at every point, the problem reduces to the linear problem discussed in the previous section, and a linear conjugate-gradient method can be used to solve for $\mathbf{x}_m$. For quadratic functions, $\mathbf{x}_m$ will be the true minimum; for non-quadratic functions, the matrix **A** can be recalculated at $\mathbf{x}_m$, and the procedure is repeated.

If the matrix $\mathbf{A}$ is not known, it is possible to build up iteratively $\mathbf{A}^{-1}$, if first derivatives of the function F can be calculated at arbitrary points. Quasi-Newton methods use this approach.

First-derivative optimization: conjugate gradients

At each iteration i, the conjugate-gradient method finds the minimum in a particular search direction $\mathbf{p}_i$. The search direction $\mathbf{p}_i$ is a linear combination of the gradient vector $\mathbf{g}_i$ and the previous search direction:

$$\mathbf{p}_i = -\mathbf{g}_i + \beta_i \, \mathbf{p}_{i-1}. \tag{4}$$

β_i is chosen such that the search directions are conjugate (i.e., $\mathbf{p}_i \cdot \mathbf{A} \cdot \mathbf{p}_j = 0$ for all $i \neq j$):

$$\beta_i = \frac{\mathbf{g}_i \cdot \mathbf{g}_i}{\mathbf{g}_{i-1} \cdot \mathbf{g}_{i-1}} = \frac{(\mathbf{g}_i - \mathbf{g}_{i-1}) \cdot \mathbf{g}_i}{\mathbf{g}_{i-1} \cdot \mathbf{g}_{i-1}} \tag{5}$$

Note that the matrix $\mathbf{A}$ need not to be known to calculate β_i. After n iterations, we have done n searches in conjugate directions, and, if the function is quadratic, the minimum is located. The search is started with a steepest descent direction: $\mathbf{p}_0 = -\mathbf{g}_0$. The first expression in equation (5) is used by Fletcher and Reeves (1964), the second by Polak and Ribiere (see Polak (1971)). Although both definitions are equivalent for quadratic functions, the latter is reputed to be better when more than n iterations are done (Press et al (1986)).

The gradients can be calculated in two ways: analytically by using exact expressions for the gradient of F, and empirically by using a finite difference approximation to the gradient.

Non-derivative optimization

An optimization method that uses only function values to locate the minimum is described by Powell (1964). Powell's algorithm iteratively builds up conjugate directions. Each iteration consists of searches down n linearly independent directions $\xi_1, \xi_2, \ldots, \xi_n$. These directions are initially chosen to be the coordinate directions. At the end of each iteration, one direction is replaced by new one. After n iterations, all the directions are conjugate, and, again if the function is quadratic, the minimum is located. In summary, an iteration of the basic algorithm is as follows (we start from $\mathbf{x}_0$, the current approximation to the minimum):

a. for $r = 1, 2, \ldots, n$, find the minimum $\mathbf{x}_r$ in direction ξ_r, and replace $\mathbf{x}_{r-1}$ by $\mathbf{x}_r$;

b. for $r = 1, 2, \ldots, n$, replace ξ_r by ξ_{r-1};

c. replace ξ_n by $(\mathbf{x}_n - \mathbf{x}_0)$ and $\mathbf{x}_0$ by the minimum in this direction.

This basic procedure can be modified for cases in which nearly dependent directions are chosen. This is especially important when the objective function has long, twisty valleys.

RESULTS

The problem that I want to optimize (Van Trier (1987)), is formulated such that no information on the derivatives of the objective function is available. I will therefore concentrate on non-derivative and gradient methods, where gradients can be calculated by finite differences.

The methods are tested on Rosenbrock's function, a function that is frequently used in optimization text books and articles. The function is a function of two variables:

$$F(x_1, x_2) = 100\,(x_2 - x_1^2)^2 + (1 - x_1)^2, \tag{6}$$

and has the desirable (at least for test cases) feature that its minimum (at (1,1)) lies in a narrow, banana-shaped valley. In all the examples, we start the search from the point (-1.2,1.0), another tradition in optimization literature.

To illustrate the problems involved with a narrow valley, we start with a steepest descent method. This method, which was not mentioned before, is a simple search method that uses the gradient as search direction in each iteration. The result is shown in Figure 1. The search algorithm does a line search with increasing step length. Once the minimum is bracketed, it is located by a quadratic interpolation. The parameters for the search algorithm are kept the same for the different methods discussed here.

Convergence of the steepest descent method is poor: at each search the minimum is "overshot," resulting in a zig-zag pattern with slowly decreasing width (Figure 2). The objective function hardly changes any more after 100 iterations, and the optimization is stopped.

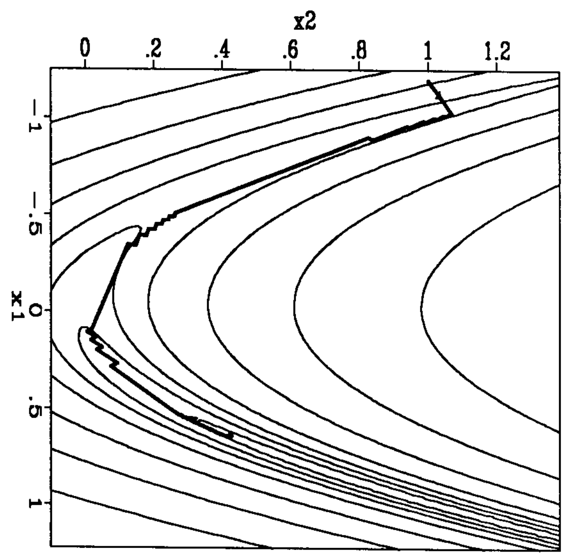

FIG. 1. Steepest descent. The search was stopped after 100 iterations.

The conjugate-gradient method performs better (Figure 3); the search directions are chosen to be almost parallel to the center of the valley. The method still requires a considerable number of iterations (25) to converge, because each time after n iterations, a steepest descent direction is used to start the next round of n iterations. It is possible to use the last search direction after n iterations instead of a steepest descent one, although search directions are then no longer conjugate. Figure 4 displays the result of this version of the conjugate-gradient method. The minimum is almost located after 19 iterations, but then the optimization overshoots the minimum, because the old gradient direction is used in the next iteration. It returns to the minimum however, and after 25 iterations it has found it.

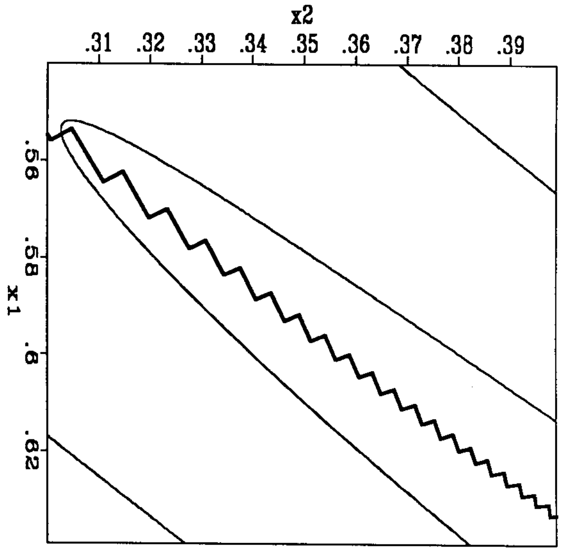

FIG. 2. Part of Figure 1 is plotted on a larger scale. The plot clearly shows the zig-zag pattern typical of steepest descent methods.

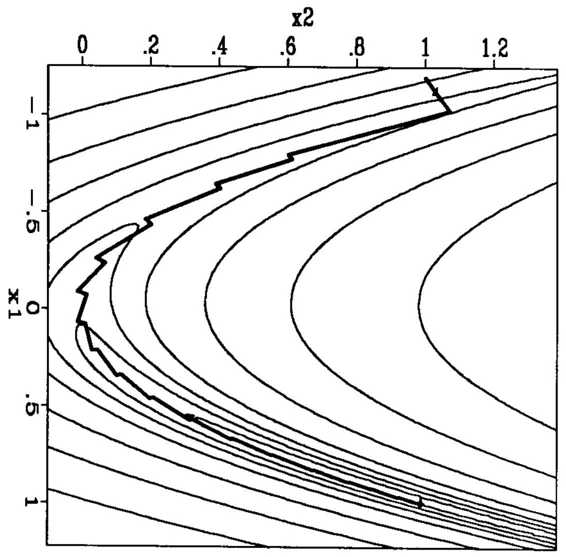

FIG. 3. Fletcher-Reeves conjugate-gradient method after 25 iterations. Analytic expressions for the gradient are used in the algorithm, and each iteration is started with a steepest descent direction.

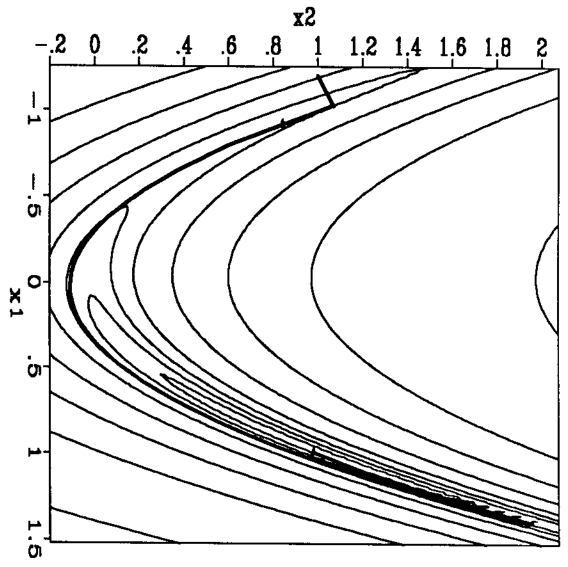

FIG. 4. Fletcher-Reeves conjugate-gradient algorithm which uses the previous search direction at the start of each iteration. Again the gradient is calculated analytically, and 25 iterations are done. Note the different scale of the plot.

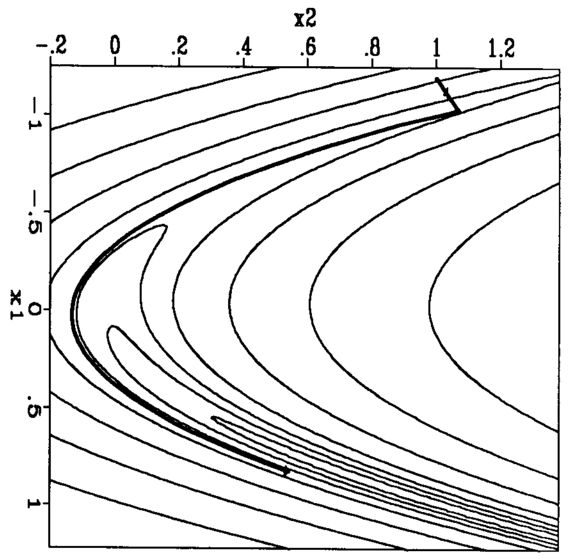

FIG. 5. Fletcher-Reeves conjugate-gradient with finite difference approximations for the gradient. As in Figure 4 the optimization was stopped after 25 iterations.

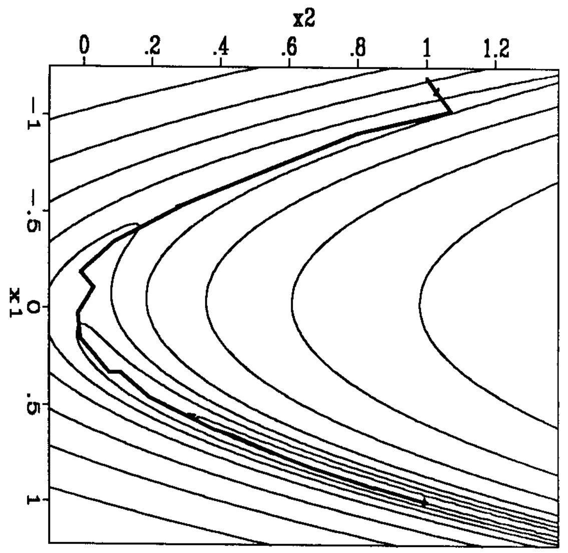

FIG. 6. Polak-Ribiere conjugate gradient algorithm with finite difference approximations for the gradients, and using the previous search direction in the next iteration. Only 12 iterations are needed to find the minimum.

Figure 5 shows the result of the conjugate-gradient method which uses finite difference approximations for the gradient. The total number of iterations is again 25. In the first iterations the convergence is about the same as when exact expressions for the gradient are used (Figure 4). For later iterations the convergence of the finite difference method gets worse than that of the exact gradient method, because the finite difference interval gets too large and thus the approximation too crude.

I also tested the Polak-Ribiere version of the conjugate-gradient method (Figure 6). The optimization converged after only 12 iterations.

Finally, the search path of the non-derivative method is shown in Figure 7. This method has the fastest convergence: 11 iterations were required to reach the minimum. It should be noted however, that the method needs about n times more function evaluations than the conjugate-gradient method, *if* gradients can be calculated theoretically. Gradients calculated by finite differences, though, need *n* function evaluations as well. Conjugate-gradient methods are then as expensive as Powell's method.

CONCLUSIONS

Although no firm conclusions can be drawn from tests on a two-parameter function, the non-derivative method of Powell seems to be well suited for small parameter problems. The method is simple in the sense that it does not require any derivative information. If gradient information is available, the Polak-Ribiere method is preferred because it is cheaper than Powell's method and it has the same convergence. Care should be taken with calculating the gradient by finite differences: the convergence of the optimization strongly depends on the choice of the finite difference interval. For large parameter problems, the conjugate-gradient method is likely to be more efficient: the search in all the parameter directions at each iteration of Powell's method can be costly when some of these directions are not well

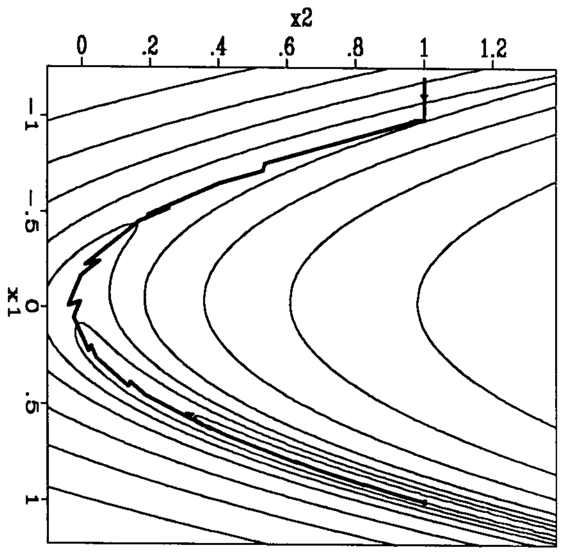

FIG. 7. Non-derivative optimization. 11 iterations were done. The search path consists of 22 line segments because two line searches are done in each iterations.

resolved.

REFERENCES

Claerbout, J., 1985, Conjugate gradient for beginners: SEP-report, **44**, 161–167.

Fletcher, R., and Reeves, C.M., 1964, Function minimization by conjugate gradient: Computer Journal **7**, 149–154.

Gill, P. E., Murray, W., and Wright, M., 1981, Practical Optimization: Academic Press.

Polak, E., 1971, Computational methods in optimization: Academic Press.

Powell, M. J. D., 1964, An efficient method for finding the minimum of a function of several variables without calculating derivatives: Computer Journal, **7**, 155–162.

Press, W. H., Flannery, B. P., Teukolsky, S. A., and Vetterling, W. T., 1986, Numerical recipes: Cambridge University Press.

Van Trier, J., 1987, Velocity analysis by nonlinear optimization of phase-contoured shot profiles: this report.