ELSEVIER

Nuclear Instruments and Methods in Physics Research A 455 (2000) 698–705

NUCLEAR INSTRUMENTS & METHODS IN PHYSICS RESEARCH
Section A

www.elsevier.nl/locate/nima

An efficient conjugate directions method without linear minimization

A.I. Manevich[a], E. Boudinov[b,*]

[a]Department of Theoretical Mechanics and Strength of Materials, Ukrainian University of Chemical Technology, 8, Gagarina Avenue, 49640 Dniepropetrovsk, Ukraine
[b]NIKHEF, Postbus 41882, NL-1009 DB Amsterdam, The Netherlands

Received 10 May 1999; received in revised form 9 May 2000; accepted 21 May 2000

Abstract

An algorithm of conjugate directions for unconstrained minimization is presented, called Single-Step Conjugate Directions Method (SSCDM) that differs from the known algorithms in the following points: (1) only one step is made on each iteration, without the linear searches of a minimum; (2) each step includes simultaneous displacements along all conjugate directions found on previous steps (instead of consequent steps along each conjugate vector); (3) each new conjugate direction is determined by the general procedure of construction of conjugate vectors, without assumption that minima along the conjugate directions were reached on previous steps. These features of the algorithm eliminate high sensitivity to computation errors pertinent to usual methods of conjugate directions and simultaneously significantly reduce the running time. Performance of the algorithm is demonstrated by applying it to a set of standard test-functions and solving a problem of high-energy physics with a Monte-Carlo-type objective function. © 2000 Elsevier Science B.V. All rights reserved.

PACS: 02.60; 02.70

Keywords: Numerical methods; Computational- and mathematical-techniques

1. Introduction

The most powerful methods of unconstrained minimization – the conjugate directions and variable metric methods – were proposed in the 1950s and 1960s [1,2]. During the last decades these methods have been improved (see for example [3]) and various program packages have been elaborated, for example [4]. But in spite of large efforts and achievements, the efficiency of these algorithms sometimes turns out to be insufficient. When calculation of the objective function and/or its gradient is very time consuming it is very important to minimize the number of steps, and an improvement of the unconstrained minimization algorithms is of evident practical interest.

It is well known that the main cause of possible failure or slowing down of the conjugate directions and variable metric methods is the degeneracy of the conjugate vectors matrix and the covariance matrix, respectively. This degeneracy is not an accidental drawback, but is a difficulty yielding from

*Corresponding author.
E-mail addresses: ugxtu@dicht.dnepropetrovsk.ua (A.I. Manevich), edouard.boudinov@cern.ch (E. Boudinov).

PII: S0168-9002(00)00568-4

the essence of the problem. For functions having narrow valleys (only for such functions the minimization is a serious problem) the conjugate vectors matrix and covariance matrix are actually very close to the degenerative matrices (the more close, the more narrow are the valleys). And even for quadratic objective functions the best algorithms turn out to be very sensitive to the rounding errors and their theoretical high rate of convergence is not realized.

The main source of error accumulation is the one-dimensional minimization along a given line. Most of the unconstrained minimization algorithms are constructed as a sequence of linear minimizations in some directions, and are based on the assumption that minima along the lines were found on previous steps. According to the Fletcher–Reeves conjugate gradients method [2], each new conjugate vector is sought from the conjugacy condition with respect to only the last conjugate vector; conjugacy conditions with respect to other conjugate vectors are not imposed as it is assumed that they are satisfied automatically by reaching the linear minima on each step. But these conjugacy conditions appear to be violated because of inevitable errors in locating the linear minima. The violation of the conjugacy conditions increases as further steps are performed. This results in the degeneracy of governing matrices. We would like to emphasise the fact that the line minimization inaccuracy itself is not responsible for the high sensitivity of the method to the computational errors. Such sensitivity is mainly caused by this line minimization inaccuracy through the use of the above-mentioned procedure of conjugate vector set construction. Therefore, in order to decrease the governing matrix distortions, one needs an algorithm, which produces the conjugate vector set without requirement that the line minimizations be reached on previous steps. In addition, if some simple procedure can ensure the decrease of the objective function, then there is no need for the linear minimization at all. Note that the linear minimization was needed to construct the conjugate vector set and to decrease the objective function.

Until now such an improvement has not been introduced to computational practice although in the 1980s a corresponding modification of the conjugate directions method was proposed and its efficiency was demonstrated [5]. Note that these methods may have an advantage over the variable metric methods for certain class of objective functions (in particular, in the cases when reliable estimation of second derivatives is impossible, for example, for Monte-Carlo-type functions).

In this paper a modified algorithm of conjugate directions, called the Single-Step Conjugate Direction Method (SSCDM), is presented. On each step (iteration) it determines the new conjugate direction by the general procedure: this vector should be conjugate to all already found conjugate vectors. This procedure does not depend on the line minimization. This enables us to facilitate and improve the determination of the direction and length of each step. Only one step is made per iteration, but this step has non-zero projections on all conjugate vectors found on previous steps and a non-zero projection on the newly found vector, and each of these projections has a Newton-like step length. Such a procedure ensures considerable reduction of sensitivity to computational errors.

In Section 2 we describe the basic algorithm which is sufficient for quadratic functions. Modifications required for non-quadratic functions are presented in Section 3. In Section 4 the performance of the algorithm is demonstrated by applying it to a set of standard test functions, and the algorithm efficiency is compared to that of the variable metric methods. The algorithm was also applied to an analysis in high-energy physics experiment with a Monte-Carlo-type objection function, for which the determination of second derivatives and linear searches are of no use.

2. Description of the method

In the case of a quadratic function the use of conjugate directions eliminates the correlations between the variables and reduces them for general (non-quadratic) functions, making it possible to locate the minimum by independent steps along each conjugate direction. In the conjugate directions methods the construction of conjugate vectors is based on the difference of the gradients in two sequential points x_{k+1} and x_k:

$$\Delta g_k = g_{k+1} - g_k \quad (g_k \equiv g(x_k)) \tag{1}$$

which is equal to $\boldsymbol{A}(x_{k+1} - x_k)$ for the quadratic function with Hesse matrix $\boldsymbol{A}$, where x is the vector of variables and k is the step number. For quadratic functions, any vector orthogonal to Δg_k is conjugate to the vector $\Delta x_k = x_{k+1} - x_k$ with respect to the second derivatives (Hesse) matrix.

In the conjugate gradients methods [2] each step Δx_k is performed only along the conjugate vector p_k: the point x_{k+1} is taken as the local minimum of the function along the direction p_k. After the kth step is made, the new conjugate vector p_{k+1} is sought as a linear combination of the gradient g_{k+1} and the vector p_k from the orthogonality condition with respect to vector $e_k = \Delta g_k$. As it is assumed that the minima along p_i $(i = 1, \dots, k)$ have been reached on corresponding steps, the condition that vector p_{k+1} and vectors p_i $(i = 1, \dots, k-1)$ are conjugate is not imposed explicitly (they must be satisfied for quadratic functions automatically).

In practice this scheme exhibits inherent shortcomings. Even very small errors in the determination of the linear minima and the rounding errors lead to gradual distortion of the conjugate vector set making the procedure inefficient.

In the proposed algorithm at each step the new conjugate vector is determined by the general procedure using the requirement that this vector and any of those previously found conjugate vectors should be conjugate. This new conjugate vector is sought as a linear combination of the gradient g_k and p_i $(i = 1, \dots, k-1)$ (lower index corresponds to the conjugate vector's number, upper index – the step number; it is assumed that $k \le N$, where N is the number of variables):

$$p_k = -g_k + \sum_{i=1}^{k-1} \beta_i^{(k)} p_i. \tag{2}$$

Each step includes displacements along all conjugate directions that were found on the previous steps and along the vector p_k:

$$x_{k+1} = x_k + \sum_{i=1}^{k} \alpha_i^{(k)} \frac{p_i}{\|p_i\|}. \tag{3}$$

Coefficients $\alpha_i^{(k)}$ $(i = 1, \dots, k-1)$ are taken as Newton-like steps, they can be easily determined using the gradients – their projections on corresponding conjugate vectors – in the last two points:

$$\alpha_i^{(k)} = -C_i^{(k)} g_{p_i}(x_k)$$

$$C_i^{(k)} = C_i^{(k-1)} \frac{g_{p_i}(x_{k-1})}{g_{p_i}(x_{k-1}) - g_{p_i}(x_k)}, \quad i = 1, \dots, k-2$$

$$C_{k-1}^{(k)} = -\alpha_{k-1}^{(k-1)} \frac{1}{g_{p_{k-1}}(x_{k-1}) - g_{p_{k-1}}(x_k)} \tag{4}$$

where $g_{p_i}(x_k)$ denotes the derivative along p_i at the point x_k; $\alpha_{k-1}^{(k-1)}$ is the component of the preceding step along the vector p_{k-1}, it was determined in the same way as the coefficient $\alpha_k^{(k)}$ on the current step. The coefficient $\alpha_k^{(k)}$ (the displacement along p_k) cannot be calculated according to Eqs. (4) since the derivative along this conjugate vector is known only at one point. It can be any non-zero displacement, for example, a constant one, and on the following step the displacement along p_k will be specified by Eqs. (4).

It is easily seen that the displacements along p_i $(i = 1, \dots, k-1)$ lead to minima along these conjugate vectors for quadratic functions. The result is that for quadratic functions the minima along p_k would be reached on the $(k+1)$th step; for non-quadratic functions these minima are gradually improved as the algorithm performs further steps.

Once each step includes simultaneous displacements along all conjugate directions, the condition of conjugacy of p_k with respect to p_i $(i = 1, \dots, k-1)$ does not coincide with the condition of orthogonality p_k and Δg_{k-1}. The new conjugate vector p_k and the previous step $\Delta x_{k-1} = x_k - x_{k-1}$ are not conjugated as well. Vectors e_i $(i = 1, \dots, k-1)$ to which p_k must be orthogonal, are determined by the recurrent formula

$$e_{k-1} = \frac{\|p_{k-1}\|}{\alpha_{k-1}^{(k-1)}} \left(\Delta g_{k-1} - \sum_{i=1}^{k-2} \alpha_i^{(k-1)} \frac{e_i}{\|p_i\|} \right). \tag{5}$$

(This expression follows for quadratic functions from Eq. (3) after multiplication by $\boldsymbol{A}$ from the left and substitution k by $k-1$.) So the coefficients $\beta_i^{(k)}$ in Eq. (2) are determined by the following conditions ((x, y) denotes the scalar multiplication of vectors x and y):

$$\beta_i^{(k)} = \frac{(g_k, e_i)}{(p_i, e_i)}. \tag{6}$$

Eqs. (2)–(6) along with the definition (1) give the algorithm which is able to locate the minimum of a quadratic function with very high accuracy after $N+1$ steps. A detailed derivation of Eqs. (2)–(6) can be found in Appendix B of Ref. [6].

3. Ensuring convergence for non-quadratic functions

The basic procedure described above should certainly be modified to ensure convergence for non-quadratic functions. If we maintain the set of conjugate vectors at $k > N$ and continue the descent along these vectors, the search can be expected to be inefficient for non-quadratic functions since the Hesse matrix does not remain constant, and conjugacy of these vectors is not preserved. In order to account for a changing Hesse matrix the set of conjugate vectors should be renewed. We used the following two alternative strategies to renew the conjugate vector set.

3.1. Sequential substitution of each conjugate vector with the new one

This strategy was proposed and discussed in Ref. [5]. When $k > N$ one may on each step substitute some conjugate vector with a new conjugate vector, preserving the other conjugate vectors. But one has to take into account the possible degeneracy (linear dependency) of the conjugate vector set due to the influence of non-quadratic terms and rounding errors. The protection from degeneracy can be based on the result proved in Ref. [7] (Powell's criterion): the determinant of the matrix whose columns are vectors, scaled so that $(p_i, Ap_i) = 1$, takes its maximum value if and only if the vectors p_i are mutually conjugate. Therefore, at $k > N$ the new conjugate vector $\tilde{p}_{k^*}$ should substitute the old vector p_{k^*}, $k^* = k \bmod N$, only under the condition that $\det(p_i)$ increases. Computation of the determinant can be avoided by decomposing the vector $\tilde{p}_{k^*}$ in terms of the vectors p_i, ($i = 1, \ldots, N$):

$$\tilde{p}_{k^*} = \sum_{i=1}^{N} v_i p_i. \tag{7}$$

Assuming that the p_i are normalized conjugate vectors in the sense that $(p_i, e_j) = \delta_{ij}$, where δ_{ij} is the Kronecker symbol, we have $v_i = (\tilde{p}_{k^*}, e_i)$. When introducing $\tilde{p}_{k^*}$ instead of the vector p_{k^*} (which is eliminated from the set of conjugate vectors at determination of p_k by Eq. (2)) the change of the determinant depends only upon v_{k^*}. If the vector $\tilde{p}_{k^*}$ is also scaled, i.e. $(\tilde{p}_{k^*}, e_k) = 1$, the choice between $\tilde{p}_{k^*}$ and p_{k^*} is carried out as follows: if $v_{k^*} < 1$, then the vector p_{k^*} remains; if $v_{k^*} > 1$, then p_{k^*} is replaced with the vector $\tilde{p}_{k^*}$.

Taking into consideration that the vector e_k can be determined only after the step which contains the displacement along $\tilde{p}_{k^*}$ is made, one ought to make this step first, then calculate the vector e_k and afterwards use the criterion presented.

3.2. Total renewal of the conjugate vector set

The simplest procedure is to construct the new set of conjugate vectors after several (1–5) cycles (a cycle consists of N steps) rather than to repair the old one. But the efficiency of such an algorithm depends upon the choice of the direction of the first new conjugate vector and the length of the step along this vector. A too small step does not guarantee a sufficient accuracy in the determination of conjugate vectors; a too large step can remove the point from an optimum zone (near the minimum) and therefore slow down the convergence.

In Ref. [5] the first new conjugate vector in the cycle was chosen along the anti-gradient at the last point. Our experience shows that it is preferable to take this vector along the line connecting the two best points $x^{(s-1)}$, $x^{(s)}$, obtained on the current and previous cycles (one may expect this direction to be close to the direction of a possible valley, similar to the "precipitous step" in the Gelfand algorithm [8]). The initial step length along this direction in the presented algorithm was taken equal to the Newton-like step (determined using the derivatives along the chosen direction in two points).

3.3. Other resources to improve the efficiency of the algorithm

The effective strategy of the search may be different far from the optimum and in the vicinity of the optimum. For numerical differentiation far from the optimum, one may employ one-sided differences, which require a smaller number of function calls. Near the optimum, the function gradient should be calculated with high accuracy and therefore central differences for the gradient calculation are preferable. Also, near the optimum, the updating strategy for the conjugate vector set described in Section 3.2 should not be used, because in the vicinity of the optimum the quadratic approximation is sufficient, and frequent changes in the set of conjugate vectors would hinder to attain the optimum with high accuracy in calculations given by Eqs. (4).

The idea of different search strategies far and near the optimum is realized in the proposed algorithm. A criterion of reaching a vicinity of the optimum was chosen as follows:

$$g_k \le \varepsilon_2, \quad \varepsilon_2 \approx (10\text{–}100)\varepsilon_1 \tag{8}$$

where ε_1 determines the required accuracy of the optimum in the gradient (the minimum is assumed to be found when $g_k \le \varepsilon_1$).

The following safeguards against "unreasonable" steps, as they can occur in highly non-quadratic functions, were introduced in the basic procedure:

(a) A restriction on the distances of displacements along each conjugate vector:

$$\alpha_i^{(k)} \le \Delta_{\max}^{(k)} \tag{9}$$

where $\Delta_{\max}^{(k)}$ decreases depending on k according to a certain rule, for example,

$$\Delta_{\max}^{(k)} = \frac{\Delta_{\max}^{(0)}}{1+\theta k}, \quad \theta = 0.05\text{–}0.1. \tag{10}$$

(b) A restriction on the change of the coefficients $C_i^{(k)}$ in Eqs. (4):

$$\eta_1\, C_i^{(k-1)} \le C_i^{(k)} \le \eta_2 C_i^{(k-1)},$$

$$\eta_1 = 0.1\text{–}0.5;\ \eta_2 = 2\text{–}5. \tag{11}$$

Apart from these modifications, the algorithm includes some other safeguards and adjustment parameters: changing the step length $\alpha_k^{(k)}$ along subsequent conjugate vectors p_k (see the explanation for the step length along this vector following Eqs. (4)), for example, multiplying an initial length by some factor; restriction on the minimal length of the steps when conjugate vectors are calculated to prevent loss of accuracy, this length varying with k in some way, for example, similar to Eq. (10).

The program, using this algorithm, is not complicated. It consists of less than 300 short Fortran lines.

The program includes some switches that enable one to use different variants of the algorithm as well as default algorithm parameters for functions of different types – the basic algorithm is used in the case of quadratic functions and the algorithm modified according to Sections 3.1 and 3.2 in the case of general functions.

4. Performance of the method (numerical experiments)

The algorithm was applied to some standard test-functions (both quadratic and non-quadratic) and to a problem of high-energy physics with a Monte-Carlo-type objective function.

All computations were performed with double precision. The gradient was calculated numerically in all problems. Wherever it was possible, results were compared with those of the variable metric method VMM (in the MINUIT [4] package).

The algorithm stops when the gradient is less than ε_1 (see Eqs. (8)), but one may use additional criteria, for example, using the increment of the function or the function value itself. The latter criterion is used for the problem considered in Section 4.3.

4.1. Quadratic functions

We considered the following function of N variables (here the lower index is the number of the variable):

$$f(x_1,\ldots,x_N) = \sum_{i=1}^{N} \frac{x_i^2}{2^{i-1}} + \sum_{i=1}^{N-1} \frac{x_i x_{i+1}}{2^i}$$

with start point $x^{(0)} = (1, \dots, 1)$ and minimum $f(0, \dots, 0) = 0$.

Results for various values of N obtained in SSCDM and a comparison with those of the variable metric method are given in Table 1.

The convergence of SSCDM for quadratic functions turns out to be close to the theoretical limit – for functions of N variables minima were reached after $N+1$ steps with high accuracy (may be, plus 1–4 steps at large N exceeding approximately 40–60). To compare this algorithm with that of VMM, the last one was required to reach approximately the same accuracy of the function at the minimum.

Both algorithms are very effective for quadratic functions. It is clear that the most efficient method for quadratic functions is the Newton's method. As the VMM-algorithm (in the MINUIT package) calculates on the first iteration the Hesse matrix and then does Newton steps, its efficiency is close to that of the Newton's method. But the SSCDM-algorithm does not compare unfavorably with these methods. If we compare the total numbers of function calls in SSCDM and VMM (for calculations of derivatives of quadratic functions using central differences, one call of the gradient is equivalent to $2N$ calls of the function) we can see that for not very large N ($N \leq 25$) these numbers are less in SSCDM, for large N VMM has some advantage. But accuracy, as a rule, is higher in SSCDM.

4.2. Non-quadratic functions

We considered the following test-functions:

NQ1 (*Rosenbrock's curved valley*, $N = 2$):

$$f(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2,$$

start point $x^{(0)} = (-1.2, 1)$, minimum $f(1,1) = 0$.

NQ2 (*Fletcher and Powell's helical valley*, $N = 3$):

$$f(x) = 100\{(x_3 - 10\Psi(x_1, x_2)^2) + (\sqrt{x_1^2 + x_2^2} - 1)^2\} + x_3^2$$

where

$$2\pi\Psi(x_1, x_2) = \begin{cases} \arctan(x_2/x_1) & (x_1 > 0) \\ \pi + \arctan(x_2/x_1) & (x_1 < 0) \end{cases}$$

start point $x^{(0)} = (-1, 0, 0)$, minimum $f(1,0,0) = 0$.

NQ3 (*Powell's quadratic function*, $N = 4$):

$$f(x) = (x_1 + 10x_2)^2 + 5(x_3 - x_4)^2 + (x_2 - 2x_3)^4 + 10(x_1 - x_4)^4$$

start point $x^{(0)} = (3, -1, 0, 1)$, minimum $f(0,0,0,0) = 0$.

NQ4 (*Wood's function*, $N = 4$):

$$f(x) = 100(x_2 - x_1^2)^2 + (x_1 - 1)^2 + 90(x_4 - x_3^2)^2 + (1 - x_3)^2 + 10.1[(x_2 - 1)^2 + (x_4 - 1)^2] + 19.8(x_2 - 1)(x_4 - 1)$$

start point $x^{(0)} = (-3, -1, -3, -1)$, minimum $f(1,1,1,1) = 0$.

NQ5 ($N = 4$):

$$f(x) = \sum_{i=1}^{10} (e^{-0.2i} + 2e^{-0.4i} - x_1 e^{-0.2x_2 i} - x_3 e^{-0.2x_4 i})^2$$

Table 1
Results for a quadratic function

Number of variables N	This method			VMM	
	Number of steps	Number of funct. calls	Accuracy in F	Number of funct. calls	Accuracy in F
10	11	242	0.2×10^{-25}	448	10^{-25}
20	21	882	0.2×10^{-27}	975	10^{-22}
30	31	1922	0.1×10^{-27}	1658	0.4×10^{-25}
40	43	3526	0.2×10^{-21}	2383	0.1×10^{-21}

start point $x^{(0)} = (0.5,0,2.5,3)$, minimum $f(1,1,2,2) = 0$.

Computations were carried out with total renewal of the conjugate vector sets after 1–2 cycles and with the first step in "the direction of valley". For non-quadratic functions one cannot speak about any theoretical limit on the required number of steps to reach the minimum of the function. We required both algorithms to locate the minimum with a reasonable accuracy and with a minimum number of the function calls. The results obtained by SSCDM and VMM are presented in Table 2 (The following parameters of the SSCDM-algorithm were assumed: $\varepsilon_1 = 10^{-4} - 10^{-6}$; $\varepsilon_2 = 10^{-1} - 10^{-4}$; initial step length $0.1 - 0.2$; $\Delta^{(0)}_{\max} = 0.4 - 2.0$; $\eta_1 = 0.2 - 0.5$;$\eta_2 = 2.0 - 5.0$, step in the calculation of derivatives $H1 = 0.00002 - 0.0001$.

Algorithm SSCDM solved all test-functions successfully. As a rule, it finds minima with better accuracy than VMM. The accuracy depended on the value ε_1 and might be easily improved by reducing ε_1 (with a small increase of the number of steps).

In all test-problems the errors in the determination of the optimal values for the function variables by SSCDM were smaller than 10^{-5} (with the exception of function NQ3).

As for the number of function calls by SSCDM, it is often less than that for VMM. Besides, VMM does many other operations with the covariance matrix (for example, calculation of its determinant and eigenvalues), and sometimes its accuracy is insufficient. SSCDM is a simpler algorithm and nevertheless it is rather reliable and stable.

It is worthwhile to note that the efficiency of the SSCDM-algorithm for non-quadratic functions (as well as that of VMM) depends upon the choice of several control (adjustment) parameters (ε_1,ε_2,$\Delta^{(0)}_{\max}$,η_1,η_2, the differentiation step $H1$, the reduction coefficient θ for maximal and minimal step length). A good choice of these parameters for non-quadratic functions remains a matter of art and intuition of a researcher and can be based on preliminary information about the behaviour of the function. A bad choice of these parameters can result in a pronounced increase of the number of steps, but nevertheless optima are found usually by the SSCDM-algorithm with the required accuracy.

4.3. A Monte-Carlo-type objective function

In high-energy physics different types of detectors are used to study the physical processes. Simulated events are needed to understand the behaviour of the detector and to be able to determine the errors. The simulation of events with a detailed treatment of the detector is usually carried out with Monte-Carlo programs, such as DELSIM [9], GEANT [10].

It is very time consuming to generate large Monte-Carlo samples. Therefore, to explore the physical processes at a level where the detector details are not important, fast techniques are used [11]. In some cases, the parameterised response of a detector is used in Monte-Carlo programs. The parameterisation uses some quantities, which reflect the detector characteristics such as resolutions, detector symmetries, background condition, etc.

Table 2
Results for non-quadratic test-functions

	This method			VMM	
Function	Number of steps	Number of funct. calls	Accuracy in *F*	Number of funct. calls	Accuracy in *F*
NQ1	57	218	0.6×10^{-14}	214	0.4×10^{-7}
NQ2	55	272	0.1×10^{-13}	188	0.9×10^{-6}
NQ3	37	247	0.5×10^{-9}	285	0.1×10^{-6}
NQ4	57	374	0.1×10^{-12}	785	0.8×10^{-8}
NQ5	79	595	0.2×10^{-12}	485	0.2×10^{-9}

One can vary these parameters and fit them to the full detector simulation. This parameterisation can then be used to study other physical parameters, for example, efficiency, purity, acceptance.

The algorithm SSCDM was applied to the Monte-Carlo function used in Refs. [6,12] to investigate the systematics on the particle identification by the DELPHI ring-imaging Cherenkov detectors [13]. The function was a χ^2-function as a result of applying the least-squares method to the Monte-Carlo response and the real data. The function $F = \chi^2/(N-3)$ has three variables and equals approximately to unity at the minimum ($N = 4$ is the number of experimental quantities to be compared with the Monte-Carlo ones). The step in the variables in the numerical calculation of the derivatives, $H1$, was chosen to be 0.2 in order to have a reasonable behaviour of the derivatives. The number of simulated events in the sample was chosen rather large in order to obtain the required accuracy in the variables.

The minimum of $F(7.8 \pm 0.1, 7.6 \pm 0.1, 1.4 \pm 0.1) \simeq 1$ was successfully reached after 6–12 steps from different starting points: $F(7,9,2) = 45, F(9,7,2) = 54, F(7,7,1) = 16, F(9,9,2) = 37$. The parameters found at minimum have variations due to the limited statistics of the Monte-Carlo function.

5. Conclusions

The modified conjugate direction method (SSCDM) is presented which has essential distinctions from the known methods and gives an efficient, reliable and rather simple technique for unconstrained minimization of functions. The high sensitivity to computation errors pertinent to the usual conjugate directions methods is greatly reduced due to the use of the general procedure for the construction of conjugate vectors. Simultaneously, the number of function calls and the running time are strongly decreased due to the absence of the linear minimization on each iteration.

Convergence of SSCDM for quadratic functions turns out to be close to the theoretical limit – the minimum is reached after $N+1$ steps with very high accuracy, where N is the number of variables. The stable updating scheme for conjugate vectors, the different strategies of search far from and near the optimum, separate control for step components and some other features of the algorithm also ensure rapid convergence for non-quadratic functions.

The current version is written in FORTRAN. Due to its simplicity this algorithm can be easily rewritten in C/C++ and included in various program packages.

Acknowledgements

It is a pleasure to acknowledge the interest and encouragement of P. van Dam, P. Kluit and J. Timmermans. Their questions and criticisms saved the authors from being on different conjugate directions with the readers.

References

[1] R. Fletcher, M.J. Powell, Computer J. 6 (1963) 163.

[2] R. Fletcher, C.M. Reeves, Computer J. 7 (2) (1964) 149.

[3] R. Fletcher, Computer J. 13 (3) (1970) 317.

[4] F. James, MINUIT, Function Minimization and Error Analysis, Reference Manual, Version 94.1, CERN Program Library Long Writeup D506, CERN Geneva, Switzerland, 1994.

[5] A.I. Manevich, P.I. Polyanchikov, Tech. Cybernet. 6 (1984) 41 (in Russian).

[6] E. Boudinov, Ph.D. Thesis, University of Amsterdam, June 2000.

[7] M.J.D. Powell, Computer J. 7 (2) (1964) 155.

[8] I.M. Gelfand, M.L. Tsetlin, Principle of nonlocal search in automatic optimization systems, Reports of USSR Academy of Sciences, Vol. 137 (2), 1961, pp. 295–298 (in Russian).

[9] DELPHI Collaboration: DELPHI event generation and detector simulation – User Guide, DELPHI Note, 1989, pp. 89–67.

[10] GEANT, Detector Description and Simulation Tool, CERN Program Library Long Writeup W5013.

[11] J. Cuevas et al., Nucl. Instr. and Meth. A 274 (1989) 459.

[12] P. Abreu et al., (DELPHI Collaboration), Measurement of the strange quark forward–backward asymmetry around the Z peak, CERN-EP 99-134, 1999, E. Phys. J. C. accepted for publication.

[13] W. Adam et al., Nucl. Instr. and Meth. A 343 (1994) 68.