A new method to compute second derivatives

Hugo D. Scolnik
e-mail: scolnik@fd.com.ar

M. Juliana Gambini
e-mail: jgambini@dc.uba.ar

Dto. de Computación FCEN, UBA.

July 24, 2001

Abstract

In thisarticle we consider the problem of computing approximations to the second derivatives of functions of n variables using finite differences. We show how to derive different formulas and how to comput the errors of those approximations as functions of the increment h, both for first and second derivatives. Based upon those results we describe the methods of Gill and Murray and the one of gradient difference. On the other hand we introduce a new algorithm which use conjugate directions methods for minimizing functions without derivatives and the corresponding numerical comparisons with the other two methods. Finally, numerical experiences are given and the corresponding conclusions are discussed.

Keywords: algorithm, finite difference, numerical approximation.

1 Introduction

The aim of this article is to develop a new method to find the second derivatives from a given function $f(x)$, $x \in R^n$ using finite difference approximations. When derivatives of f are very difficult to compute or too hard to evaluate, the methods to calculate the minimun of $f(x)$, such as quasi Newton or Newton, use finite difference approximations of gradient vector and hessian matrix of the function, and the success of the method depends directly on these approximations for the derivatives. The finite difference equations are obteined from the Taylor series expansion of f, as will see below, is give by

$$\frac{\partial^2 f(x_0)}{\partial x_i \partial x_j} \cong \frac{f(x_0 + h_i e_i + h_j e_j) - f(x_0 + h_i e_i) - f(x_0 + h_j e_j) + f(x_0)}{h_i h_j},$$

where the most important factor to consider is the proper selection of h_i and h_j. When h_i and h_j are too small, arithmetic errors are produced and consequently non satisfactory approximations of derivatives are obtained. The idea behind our method is to find the best size step possible to minimize an error function. In order to do that, we suppose that h depends on several parameters, $h = h(\alpha_1, ..., \alpha_k)$. We take a set of different test functions $f_1, f_2, ...f_m$ whose hessian matrices in several points,$Hv_1, Hv_2, ..., Hv_m$ are known. $Hc_1, Hc_2, ..., Hc_m$ are the hessian matrices calculated with the increment h. Then we define the following objective function:

$$F(\alpha_1, .., \alpha_k) = \sum_{i=1}^{m} \frac{\|Hv_i - Hc_i(\alpha_1, .., \alpha_k)\|_F}{\|Hv_i\|_F},$$

and solve the problem to finding

$$\arg\min_{\alpha} F(\alpha)$$

using minimization methods without derivatives to obtain the best size step.

2 Classic Methods to approximate derivatives

2.1 First derivatives

Let $f : R^n \to R$, $x_0 \in R^n$, $h \in R$, we can approximate $\nabla f(x_0\ )$ using only values of the function f in the following way. We will call $g(x_0) = \nabla f(x_0)$, or $g(x_0) = (g_1(x_0), ..., g_n(x_0))$ where $g_i\ (x_0) = \frac{\partial f}{\partial x_i}(x_0)$

Forward-Difference The forward-difference equation for first derivatives is

$$g_i(x_0) \cong \frac{f(x_0 + he_i) - f(x_0)}{h}.$$

There are two sources of error in this equation, truncation error and roundoff error. In order to find the truncation error, we use the Taylor series expansion,

$$f(x_0 + he_i) = f(x_0) + hg^t(x_0).e_i + \frac{1}{2}h^2 e_i^t H(x_0) e_i + O(h^3).$$

This can be written as

$$g^t(x_0).e_i = \frac{f(x_0 + he_i) - f(x_0)}{h} - \frac{1}{2}he_i^t H(x_0) e_i + O(h^2).$$

So the truncation error is

$$E_t = \frac{1}{2}he_i^t H(x_0)e_i = \frac{1}{2}hH_{ii}(x_0),$$

where $H(x_0)$is the hessian matrix evaluated in x_0.

The roundoff error has various sources. If h is too small, then we will obtain a poor accuracy. To show that, we consider the characteristic error of the machine's floating-point format, which is the least non negative number tol such as: $1 + tol \neq 1$ and consequently multiplying by $|x_0|$ we get

$$|x_0| + |x_0|\, tol \neq |x_0|\,.$$

From the above equation we deduce that the condition $h \geq |x_0|\, tol$ is required to hold. If we took $h < |x_0|\, tol$ it would result $f(x_0+h) = f(x_0)$ and the approximation would be always zero.

Suppose now that h satisfies the above condition and E_f is the fractional accuracy with which $f(x_0)$ is computed. $\Psi(x_0)$ is the value in floating point format of $f(x_0)$

$$\frac{|f(x_0) - \Psi(x_0)|}{|f(x_0)|} < E_f$$

then,

$$\left| \frac{f(x_0+he_i) - f(x_0)}{h} - \frac{\Psi(x_0+he_i) - \Psi(x_0)}{h} \right| < \frac{2E_f\,|f(x_0)|}{h}.$$

We see that if h is too small, the total error may be large. Varying h to minimize $\Upsilon(h)$ gives the optimal choice of h.

$$\Upsilon(h) = \frac{E_f\,|f(x_0)|}{h} + h\,|H_{ii}(x_0)|$$

the best solve of h is given by

$$h \sim \sqrt{\frac{E_f\,|f(x_0)|}{|H_{ii}(x_0)|}}.$$

At best

$$ert \sim \frac{\sqrt{E_f}\sqrt{|f(x_0)|\,|H_{ii}(x_0)|}}{|g_i(x_0)|} \sim \sqrt{E_f}$$

where we assume that f, H_{ii}, g_i all share the same characteristic length scale.

By observing $\Upsilon(h)$, we see that a large size step h will induce a large truncation error and a small size step h will induce a large roundoff error. So a good choice of h will be that both errors have the same order of magnitude. This idea was developed in [8].

Symmetrized form If you can afford two function evaluations for each derivative calculation it is better to use de symmetrized form:

$$g_i(x_0) \cong \frac{f(x_0+he_i) - f(x_0-he_i)}{2h}.$$

In order to deduce that, we use the Taylor series expansion

$$f(x_0+he_i) = f(x_0) + hg^t(x_0).e_i + \frac{1}{2}h^2 e_i^t H(x_0) e_i + \frac{h^3}{6}\frac{\partial^3 f(x_0)}{\partial^3 x_i} + O(h^4) \tag{1}$$

$$f(x_0-he_i) = f(x_0) - hg^t(x_0).e_i + \frac{1}{2}h^2 e_i^t H(x_0) e_i - \frac{h^3}{6}\frac{\partial^3 f(x_0)}{\partial^3 x_i} + O(h^4) \tag{2}$$

substracting equations (1) and (2), we have

$$f(x_0+he_i) - f(x_0-he_i) \cong 2hg^t(x_0).e_i + 2\frac{h^3}{6}\frac{\partial^3 f(x_0)}{\partial^3 x_i},$$

then

$$E_t \sim h^2 \left| \frac{\partial^3 f(x_0)}{\partial^3 x_i} \right|.$$

The roundoff error is the same that the other case, so the resultant error for $g_i(x_0)$ is

$$\Upsilon(h) = \frac{E_f\,|f(x_0)|}{h} + h^2 \left| \frac{\partial^3 f(x_0)}{\partial^3 x_i} \right|,$$

the best value of h is

$$h \sim \sqrt[3]{E_f}\sqrt[3]{\frac{f(x_0)}{\frac{\partial^3 f(x_0)}{\partial^3 x_i}}},$$

and

$$ert \sim \frac{(E_f)^{\frac{2}{3}}(f(x_0))^{\frac{2}{3}}(\frac{\partial^3 f(x_0)}{\partial^3 x_i})^{\frac{1}{3}}}{|g_i(x_0)|} \sim (E_f)^{\frac{2}{3}}.$$

We assume again that $f(x_0)$,$\frac{\partial^3 f(x_0)}{\partial^3 x_i}$ y $g_i(x_0)$ all share the same characteristic length scale.

2.2 Second Derivatives

Given $f : R^n \to R$, $x_0 \in R^n$, $h \in R$, we can compute the second derivatives of f using only values of the function in the following way.

2.2.1 Forward Difference

For $i \neq j$

$$\frac{\partial^2 f(x_0)}{\partial x_i \partial x_j} = \frac{\partial}{\partial x_i}(f_{x_j}),$$

like we see in the previous section

$$\frac{\partial}{\partial x_i}(f_{x_j}(x_0)) = \frac{f_{x_j}(x_0 + he_i) - f_{x_j}(x_0)}{h}$$

$$= \frac{1}{h}\left[\frac{f(x_0 + he_i + he_j) - f(x_0 + he_i)}{h} - \frac{f(x_0 + he_j) - f(x_0)}{h}\right],$$

then

$$\frac{\partial^2 f(x_0)}{\partial x_i \partial x_j} \cong \frac{f(x_0 + he_i + he_j) - f(x_0 + he_i) - f(x_0 + he_j) + f(x_0)}{h^2}, \qquad (3)$$

and for $i = j$

$$\frac{\partial^2 f(x_0)}{\partial^2 x_j} \cong \frac{f(x_0 + 2he_j) - 2f(x_0 + he_j) + f(x_0)}{h^2}.$$

We will call H_{ij} the approximation for $\frac{\partial^2 f(x_0)}{\partial x_i \partial x_j}$. In order to compute the truncation error we use the Taylor series expansion with a different notation: for $t \in R^n$

$$f(x_0 + ht) = f(x_0) + h\sum_{i=1}^{n}\frac{\partial f(x_0)}{\partial x_i}t_i + \frac{h^2}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{\partial^2 f(x_0)}{\partial x_i \partial x_j}t_i t_j +$$

$$+\frac{h^3}{6}\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\frac{\partial^3 f(x_0)}{\partial x_i \partial x_j \partial x_k}t_i t_j t_k + O(h^4).$$

So we obtain:

$$f(x_0 + he_i) = f(x_0) + hg_i(x_0) + \frac{h^2}{2}\frac{\partial^2 f(x_0)}{\partial^2 x_i} + \frac{h^3}{6}\frac{\partial^3 f(x_0)}{\partial^3 x_i} + O(h^4) \qquad (4)$$

$$f(x_0 + he_j) = f(x_0) + hg_j(x_0) + \frac{h^2}{2}\frac{\partial^2 f(x_0)}{\partial^2 x_j} + \frac{h^3}{6}\frac{\partial^3 f(x_0)}{\partial^3 x_j} + O(h^4) \qquad (5)$$

$$f(x_0+he_i+he_j) = f(x_0) + hg_i(x_0) + hg_j(x_0) + \frac{h^2}{2}\frac{\partial^2 f(x_0)}{\partial^2 x_i} + \frac{h^2}{2}\frac{\partial^2 f(x_0)}{\partial^2 x_j} +$$

$$h^2\frac{\partial^2 f(x_0)}{\partial x_i \partial x_j} + \frac{h^3}{6}(\frac{\partial^3 f(x_0)}{\partial^3 x_i} + \frac{\partial^3 f(x_0)}{\partial^3 x_j} + 3\frac{\partial^3 f(x_0)}{\partial^2 x_i \partial x_j} + 3\frac{\partial^3 f(x_0)}{\partial x_i \partial^2 x_j}) + O(h^4). \qquad (6)$$

Making the replacement (4), (5) y (6) in (3) we obtain

$$H_{ij} \cong \frac{\partial^2 f(x_0)}{\partial x_i \partial x_j} + \frac{h}{2}(\frac{\partial^3 f(x_0)}{\partial^2 x_i \partial x_j} + \frac{\partial^3 f(x_0)}{\partial x_i \partial^2 x_j})$$

from

$$E_t = \frac{h}{2}\underbrace{(\frac{\partial^3 f(x_0)}{\partial^2 x_i \partial x_j} + \frac{\partial^3 f(x_0)}{\partial x_i \partial^2 x_j})}_{K}.$$

The roundoff error is

$$Er \sim \frac{E_f\,|f(x_0)|}{h^2},$$

And varying h to minimize the sum of errors yields

$$h \sim \sqrt[3]{\frac{E_f\,|f(x_0)|}{K}}$$

Symmetrized form For $i \neq j$, using the equations in the previous section

$$\frac{\partial}{\partial x_i}(f_{x_j}(x_0)) = \frac{f_{x_j}(x_0+he_i) - f_{x_j}(x_0-he_i)}{2h} =$$

$$= \frac{1}{2h}\left[\begin{array}{c} \frac{f(x_0+he_i+he_j)-f(x_0+he_i-he_j)}{2h} - \\ \frac{f(x_0-he_i+he_j)-f(x_0-he_i-he_j)}{2h} \end{array}\right],$$

then

$$\frac{\partial^2 f(x_0)}{\partial x_i \partial x_j} \cong \frac{f(x_0+he_i+he_j) - f(x_0+he_i-he_j)}{4h^2} +$$

$$\frac{+f(x_0-he_i-he_j) - f(x_0-he_i+he_j)}{4h^2},$$

and for $i = j$

$$\frac{\partial^2 f(x_0)}{\partial^2 x_j} \cong \frac{f(x_0+he_j) - 2f(x_0) + f(x_0-he_j)}{h^2}.$$

Truncation errror: Using (4),(5) y (6),we obtain

$$H_{ij} \cong \frac{\partial^2 f(x_0)}{\partial x_i \partial x_j} + \frac{O(h^4)}{4h^2}.$$

3 Other Methods

The algorithm developed in this work has been compared with the following methods:

3.1 Gill-Murray

Given $f : R^n \to R$ this method uses the following equation:

$$\frac{\partial^2 f(x_0)}{\partial x_i \partial x_j} \cong \frac{f(x_0 + h_i e_i + h_j e_j) - f(x_0 + h_i e_i) - f(x_0 + h_j e_j) + f(x_0)}{h_i h_j},$$

where

$$\begin{array}{c} h_i = \sqrt[3]{tol}\max\{0.1, |x_{0i}|\} \\ h_j = \sqrt[3]{tol}\max\{0.1, |x_{0j}|\} \end{array}.$$

As we show in Section 2 , a good selection for the increment h is

$$h \sim \sqrt[3]{\frac{E_f\,|f(x_0)|}{K}},$$

where $K = \frac{\partial^3 f(x_0)}{\partial^2 x_i \partial x_j} + \frac{\partial^3 f(x_0)}{\partial x_i \partial^2 x_j}$ and $E_f \simeq E_m$ where E_m (or tol) is the error characteristic of the machine's floting point format.

As

$$x_c = \sqrt[3]{\frac{f(x_0)}{K}}$$

is the curvature scale of the function f which is not available, then in the absence of any other information it is often assumed $x_c = x_{0i}$ where i is the coordinate which increases, except near $x_{0i} = 0$. Then it takes $h = \max\{0.1, |x_{0i}|\}$.

3.2 Gradient difference method

The subrutine NUGRAD computes the gradient of the function $f : R^n \to R$ evaluated in x_0, then we obtain $g = (g_1(x_0), ..., g_n(x_0))$ and we use the finite difference equation to compute the matrix M, where

$$M_{ij} = \frac{g_i(x_0 + h_j e_j) - g_i(x_0)}{h_j},$$

and the increment h_j is the same that we used to compute g_j. Nevertheless this is not an accuracy approximation for the hessian matrix of the $f \in C^2$ function, since M is probably not a symmetric matrix, then we use, $H = \frac{M+M^t}{2}$ as stated in the following lemma:

Lemma:

If $A \in R^{nxn}$is non-singular, and

$$B = \frac{A + A^t}{2},$$

then

$$\|A - B\|_F = \min_X \|X - A\|_F \,, \forall X \; symmetric.$$

Proof:

B is a symmetric matrix:

$$B^t = \left(\frac{A + A^t}{2}\right)^t = \frac{A^t + A}{2} = B.$$

Let

$$F(x_{11}, ..., x_{1n}, x_{21}, ..., x_{nn}) = \|X - A\|_F$$

or $F(x_{11}, ..., x_{1n}, x_{21}, ..., x_{nn}) = \sum_{i=1}^n \sum_{j=1}^n (x_{ij} - a_{ij})^2$ then, since $x_{ij} = x_{ji}$

$\frac{\partial F}{\partial x_{ij}} = 2(x_{ij} - a_{ij}) + 2(x_{ij} - a_{ji})$so $\frac{\partial F}{\partial x_{ij}} = 0 \Leftrightarrow 2x_{ij} - a_{ij} - a_{ji} = 0$ and we have $x_{ij} = \frac{a_{ij}+a_{ji}}{2}$

The stationary point is $X = B$ which is a minimun, since the norm is a convex function.

4 Proposed Method

This algorithm compute second derivatives of a function $f : R^n \to R$, $f \in C^2$ using only values function.

As we show in section 2, the algorithm uses the finite differences equation:

$$\frac{\partial^2 f(x_0)}{\partial x_i \partial x_j} \cong \frac{f(x_0 + h_i e_i + h_j e_j) - f(x_0 + h_i e_i) - f(x_0 + h_j e_j) + f(x_0)}{h_i h_j}.$$

The most important issue in this new technique is the choice of h_i, h_j. The proposed algorithm provides a new way to do this selection as explain the following procedure:

We choose an arbitrary amount of test functions, in this case we take thirteen $\{f_1, f_2, ...f_{13}\}$ whose hessian matrices are known $\{Hv^1_, Hv^2, ..., Hv^{13}\}$. These functions have to be representative of all used functions (polinomials, trigonometrics, exponencials, etc.) and we build an error function in the following way:

We suppose that h_i dependes on several parameters $\{\alpha_1, \alpha_2, \alpha_3, \alpha_4\}$

$$h_i = h(\alpha_1, \alpha_2, \alpha_3, \alpha_{4,} x_i, f(x)),$$

in order to choose the function $h(\alpha_1, \alpha_2, \alpha_3, \alpha_{4,} x_i, f(x))$ we have tried several function combinations looking for one combination that is invariant the change of scale variables. We show here some of them

$$x_i = a10^b$$
$$f(x) = c10^d$$

1)$h_i = \alpha_1^b.\alpha_2^d.\alpha_3 + \alpha_4.|x_i|$
2)$h_i = (\alpha_1^b + \alpha_2^d)^{\alpha_3} + \alpha_4.|x_i|$
3)$h_i = (\alpha_1^b - \alpha_2^d).\alpha_3 + \alpha_4.\max(|x_i|, 0.1)$
4)$h_i = (\alpha_1.b \pm \alpha_2.d)\alpha_3^d + \alpha_4.\max(|x_i|, 0.1)$
5)$h_i = (\alpha_1|x_i|)^b. + (\alpha_2|f(x)|)^d.\alpha_3 + \alpha_4.|x_i|$
6)$h_i = sen(\alpha_1^b.\alpha_2^d.\alpha_3) + \alpha_4.\max(|x_i|, 0.1)$
The best results were obtained using the following equation:

$$h_i = (\alpha_1^b + \alpha_2^d).\alpha_3 + \alpha_4.\max(|x_i|, 0.1).$$

We compute the hessian matrices using finite differences with this equation for h_i, then we obtain the hessian matrices, which depend on the same parameters

$$\begin{array}{c} Hc^1(\alpha_1, \alpha_2, \alpha_3, \alpha_4) \\ Hc^2(\alpha_1, \alpha_2, \alpha_3, \alpha_4) \\ \cdots \\ Hc^{13}(\alpha_1, \alpha_2, \alpha_3, \alpha_4) \end{array}.$$

Then we build the objective function in the following way:

We take ten matrices$\{X_1, ..., X_{10}\}$. Each of them contains, at column m, the point where we evaluate the hessian matrix of the function m. It should be emphazised that each hessian matrix is evaluated in ten different points, which gives a wide spectrum of initial values and also a wide spectrum for the values of the derivatives. In the columns of the matrices X_i there are values near zero, near one thousand, etc.

We take

$$F_k(\alpha_1, \alpha_2, \alpha_3, \alpha_4) = \sum_{m=1}^{13} \sum_i \sum_j \left| \frac{Hv_{ij}^m(x_k) - Hc_{ij}^m(\alpha_1, \alpha_2, \alpha_3, \alpha_4, x_k)}{Hv_{ij}^m(x_k)} \right|.$$

Here $Hv_{ij}^m(x_k)$ means that the hessian matrices are evaluated at the points which are contained in the matrix X_k. The objective function is

$$F(\alpha_1, \alpha_2, \alpha_3, \alpha_4) = \sum_{k=1}^{10} F_k(\alpha_1, \alpha_2, \alpha_3, \alpha_4) =$$

$$\sum_{k=1}^{10} \sum_{m=1}^{13} \sum_i \sum_j \left| \frac{Hv_{ij}^m(x_k) - Hc_{ij}^m(\alpha_1, \alpha_2, \alpha_3, \alpha_4, x_k)}{Hv_{ij}^m(x_k)} \right|,$$

then we minimize this function using the Powell method. It results an optimal α^*, which is used to compute the size step for the finite difference equation.

$$h_i = ((\alpha_1^*)^b + (\alpha_2^*)^d).\alpha_3^* + \alpha_4^*.\max(|x_i|, 0.1).$$

This method has been checked with a lot of different functions and the results are exposed in the Numerical Results section.

5 Conclusion

In this work, a new method to compute second derivatives (M3) has been depeloped. It has been checked and compared to other two methods, Gill-Murray(M1) and Gradient difference(M2).

This algorithms have been checked in two ways:

1-forward finite difference form

2-symmetrized finite difference form

The results are the following:

With the first form, M3 gives the best accuracy approximation in 80,6% of all experiments. The number of functions evaluations is the same for M1 and M3, while M2 is more expensive.

With the second form, M3 gives the best accuracy approximation in all experiments, except for cuadratic functions, in which M1 gives best results.

We have observed that whenM3 was the best method , M2 is better than M1, but uses a lot of function evaluations.

Using the symmetrized form, errors are smaller, since this formula has a higher order of accuracy, as we have seen in Section 2, but a large amount of function evaluations are neccesary.

If the matrix dimension is large, the roundoff error is larger since there are more arithmetic errors.

The contribution of the new method proposed in this work, is that in the most of cases, it yields an accuracy approximation with a smaller number of function evaluation. This work shows that finding the minimun of an error function can be an efficient way for modifications to numerical recipes and it was used to construct an algorithm which combines accuracy aproximation and low computacional cost.

6 Numerical Results

Here we show same experimental results.

Cuadratic Function

Forward Difference

	$M1$	$M2$	$M3$
error	$0.2294d-14$	$0.5163d-01$	$0.1874d-07$
num. de eval	7	9	7

Symmetrized Form

	$M1$	$M2$	$M3$
error	$0.2294d-14$	$0.8695d-05$	$0.1264d-08$
num. de eval	9	15	9

Gaussian Function(n=3)

Forward Difference

	$M1$	$M2$	$M3$
error	$0.4367d-01$	$0.15105d+01$	$0.3288d-03$
num. de eval	13	16	13

Symmetrized Form

	$M1$	$M2$	$M3$
error	$0.9412d-02$	$0.1224d-03$	$0.9118d-06$
num. de eval	19	15	19

Biggs function (n=6)

Forward Difference

	$M1$	$M2$	$M3$
error	$0.8514d-02$	$0.3362d-00$	$0.2580d-02$
num. de eval	43	49	43

Rosenbrock extended function (n=12)

Symmetrized Form

	$M1$	$M2$	$M3$
error	$0.1313d + 03$	$0.22003d + 01$	$0.3959d - 01$
num. de eval	289	325	289

References

[1] Fletcher,R. (1987), Practical Methods of Optimization. Wiley, New York

[2] Gill, P.E, Murray, W. and Pitfield, R.A. (1972). The implementation of two modified Newton algorithms for unconstrined optimization. NPL Report NAC11.

[3] Gill,P.E , Murray,W., Saunders,M.A. and Wright, M.H(1980), Computing the Finite-Difference Approximations to Derivatives for Numerical Optimization.Technical Report, Departament of Operations Research-Standford University.

[4] Gill,P.E , Murray,W., Saunders,M.A. and Wright, M.H(1991) Numerical Linear Algebra and Optimization, vol.1.Addison Wesley.

[5] Gill,P.E , Murray,W., Saunders,M.A. and Wright, M.H(1993) Practical Optimization. Academic Press.

[6] Powell, M.J.D.(1964) An efficient method for finding the minimun of a function of several variables without calculating derivatives. Computer Journal, vol.7, pp 155-162.

[7] Press,(1992), Numerical Recipes, Academic Press.

[8] G.W.Stewart(1967) A modification of Davidon's minimization method to accept difference approximations to derivates,J.Ass.Comput.Mach.,14,72-83.

[9] G.R.Walsh(1975), Methods of Optimization, Wiley, London.