COMPARATIVE STUDY BETWEEN A NEW METHOD OF CONJUGATE ADDRESSES AND QUASI-NEWTON METHODS IN UNCONSTRAINED OPTIMIZATION

Jhonys Enrique Bolaño Ospino[1*], Juan Carlos León García[1], David Enrique Aragon Peña[1], Jose Luis Zabaleta Fernandez[1], Helmer Javier Muegues Rincon[1], Julio Rafael Vega Barrios[1]

[1]Universidad Popular del Cesar, Valledupar, Cesar, Colombia

[1]Email: jhonysbolano@unicesar.edu.co | Email: jcleon@unicesar.edu.co | Email: dearagon@unicesar.edu.co | Email: joseluiszabaleta@unicesar.edu.co | Email: hjmuegues@unicesar.edu.co | Email: juliovegabarrios@unicesar.edu.co
*Corresponding Author: Jhonys Enrique Bolaño Ospino | Email: jhonysbolano@unicesar.edu.co

Summary
This paper shows a comparative study between some Quasi-Newton methods and a new method of conjugate directions in unconstrained optimization, based on a modification of the condition of the secant. The new method of conjugate directions was studied by Guoyin Li et al, where gradient and target function information are used to define the conjugate condition. The Quasi-Newton methods used to perform the numerical comparison are BFGS and DFP. The numerical results show that in most tests, the proposed conjugate address method shows an improvement in execution time.
Keywords: Unconstrained optimization; conjugate condition; conjugate gradient method; Quasi-Newton equation

1. Introduction

The basic idea behind the conjugate gradient method is to build a base of orthogonal vectors and use it to find the solution more efficiently. Such a course of action would generally not be advisable because the construction of an orthogonal base using the Gramm-Schmidt procedure requires, when selecting each new element of the base, to ensure its orthogonality with respect to each of the previously constructed vectors. The great advantage of the conjugate gradient method lies in the fact that when this procedure is used, it is enough to ensure the orthogonality of a new member with respect to the last one that has been constructed, so that this condition is automatically fulfilled with respect to all the previous ones [8 10].

On the other hand, several recent studies have shown that quasi-newton methods are very effective in solving optimization problems without restriction. It is natural to remember that Newton's method is based on second-order Taylor development involving the computation of the Hessian matrix in each iteration. In practice, it is often preferred to approximate this matrix or its inverse with a symmetrical and definite positive matrix through some effective procedure rather than its exact computation. This idea of approximating the Hessian with a symmetrical and definite positive matrix was introduced by Davidon [5]. The class of methods that approximate the Hessian of Newton's method or its inverse using a symmetric and positive defined matrix instead of the corresponding exact value are known as quasi-Newton methods. Usually this approximation matrix is denoted as $\boldsymbol{B_{k+1}}$, here the approximations of Davidon, Fletcher, and Powell (DFP) and Broyden, Fletcher, Goldfarb, and Shanno (BFGS) were used for this calculation. It should be noted that the algorithmic implementation is found in the MATLAB language.

2. Conjugate gradient method

2.1 Basic properties

The conjugate gradient method is a conjugate direction method with a very special property: By generating your set of conjugate vectors, you can find a new vector $\boldsymbol{d_{k+1}}$ using only the previous vector You don't need to know all the previous elements of the conjugate set; it is automatically a conjugate vector. This remarkable property implies that the method requires small storage and computation.$\boldsymbol{d_k}.\ \boldsymbol{d_1}, \boldsymbol{d_2}, \dots, \boldsymbol{d_k}\boldsymbol{d_{k+1}}$

In the conjugate gradient method, each direction is a linear combination of the negative residue, which is the steepest downward direction, for a defined function of and the previous direction. The new conjugate direction is calculated as follows:$\boldsymbol{d_{k+1}}\ \ -\boldsymbol{g_k} \boldsymbol{f} \boldsymbol{R^n} \to \boldsymbol{R}\ \boldsymbol{d_k}$

$\boldsymbol{d_{k+1}}$= (1)$\begin{cases} -\boldsymbol{g_k} & \overset{sik=0}{} \\ -\boldsymbol{g_k} + \boldsymbol{\beta_k d_k}\ \boldsymbol{si}\ \boldsymbol{k \geq 1} \end{cases}$

where is a scalar and =. If f is a quadratic and strictly convex function, i.e. of the form,$\boldsymbol{\beta_k g_k}\ \mathbf{\nabla f(x_k)}$

$$(2)\boldsymbol{f(x) = \frac{1}{2}x^T H x + b^T x}$$

where is a positive defined matrix, the iterative formula of the conjugate gradient method is given by:$\boldsymbol{H}$

$$\boldsymbol{x_{k+1} = x_k + s_k}, \tag{3}$$

with , let us also note that $\boldsymbol{s_k = \alpha_k d_{k+1}}$

,
$$(4)\boldsymbol{s_k = x_{k+1} - x_k}$$

where is the exact minimizer along i.e.$\boldsymbol{\alpha_k d_{k+1}}$
arg min .
$$(5)\boldsymbol{\alpha_k = f(x_k + \alpha d_{k+1})}$$

In general, some ways of calculating the conjugate condition that have had a significant impact on the development of the conjugate gradient method will be presented. Among the most referenced we have the following:$\boldsymbol{\beta_k}$

- Fletcher-Reeves (FR) $\boldsymbol{\beta_k}$= (6) $\frac{g_{k+1}^T g_{k+1}}{g_k^T g_k}$
- Hestenes Boots (HS) $\boldsymbol{\beta_k}$= $\frac{y_k^T g_{k+1}}{y_k^T s_k}$ (7)
- Polak-Ribiere (PR) $\boldsymbol{\beta_k} = \frac{y_k^T g_{k+1}}{g_k^T g_k}$ (8)

2.2 New conjugate condition

Using Taylor's formula for the objective function, we have: $\boldsymbol{f(x)}$

$$\boldsymbol{f(x) \approx f_{k+1} + g_{k+1}^T(x - x_{k+1}) + \frac{1}{2}(x - x_{k+1})^T G_{k+1}(x - x_{k+1})}$$

where represents the Hessian matrix in . Substituting for we have: $\boldsymbol{G_{k+1} x_{k+1} x x_k}$

$$\boldsymbol{f_k \approx f_{k+1} - g_{k+1}^T s_k + \frac{1}{2}s_k^T G_{k+1} s_k}.$$

Consequently,

$$\boldsymbol{s_k^T G_{k+1} s_k \approx 2(f_k - f_{k+1}) + 2g_{k+1}^T s_k}$$
$$(9) = \boldsymbol{2(f_k - f_{k+1}) + (g_k + g_{k+1})^T s_k + s_k^T y_k}$$

Considering as a new approximation of we have to:$\boldsymbol{B_{k+1} G_{k+1}}$

$$\boldsymbol{s_k^T B_{k+1} s_k = s_k^T y_k + v_k} \tag{10}$$

where is defined by:$\boldsymbol{v_k}$

$$\boldsymbol{v_k = 2(f_k - f_{k+1}) + (g_k + g_{k+1})^T s_k} \tag{11}$$

This makes us think of the following new quasi-Newton equation:

$$\boldsymbol{B_{k+1}s_k = y_k^*}, \quad \boldsymbol{y_k^* = y_k + \frac{v_k}{\|s_k\|^2}s_k} \tag{12}$$

Theorem 1: Assume that the function f(x) is smooth enough and the is small enough, then we have.$\boldsymbol{s_k}$

$$\boldsymbol{s_k^T G_{k+1} s_k - s_k^T y_k^* = \frac{1}{3} s_k^T (T_k s_k) s_k + 0(\|s_k\|^4)} \tag{13}$$

y

$$\boldsymbol{s_k^T G_{k+1} s_k - s_k^T y_k = \frac{1}{2} s_k^T (T_k s_k) s_k + 0(\|s_k\|^4)}, \tag{14}$$

where $\boldsymbol{y_k^*}$ is defined in (12) and is the tensor of f in y $\boldsymbol{T_k x_k}$

$$\boldsymbol{s_k^T (T_k s_k) s_k = \sum_{i,j,l=1}^{n} \frac{\partial^3 f(x)}{\partial x^i \partial x^j \partial x^l} s_k^i s_k^j s_k^l}.$$

Proof. Using Taylor's formula for the objective function f(x) we have:

$$\boldsymbol{f_k = f_{k+1} - g_{k+1}^T s_k + \frac{1}{2} s_k^T G_{k+1} s_k - \frac{1}{6} s_k^T (T_k s_k) s_k + 0(\|s_k\|^4)} \tag{15}$$

y

$$\boldsymbol{g_k^T s_k = g_{k+1}^T s_k - s_k^T G_{k+1} s_k + \frac{1}{2} s_k^T (T_k s_k) s_k + 0(\|s_k\|^4)}. \tag{16}$$

So the conclusion follows by the definitions of and .$\boldsymbol{y_k}$ $\boldsymbol{y_k^*}$

Theorem 2: Suppose (12) satisfies, then for any we have:$\boldsymbol{B_k k \geq 1}$

$$\boldsymbol{f_k = f_{k+1} + g_{k+1}^T (x_k - x_{k+1}) + \frac{1}{2} (x_k - x_{k+1})^T B_k (x_k - x_{k+1})} \tag{17}$$

Proof. Consequently (17) is obtained simply by rewriting (10) and (12) the conclusion follows immediately.

2.2.1 Conjugate condition based on the new Quasi-Newton equation

In this subdivision, the new conjugated condition is ours. To this end, the new Quasi-Newton equation is used instead of the original quasi-Newton equation.

$\boldsymbol{B_{k+1} s_k = y_k^*}$

and the search address is calculated as follows:

$\boldsymbol{d_{k+1} = -B_{k+1}^{-1} g_{k+1}}$

thus we obtain that

$$\boldsymbol{d_{k+1}^T y_k^* = d_{k+1}^T (B_{k+1} s_k) = -g_{k+1}^T s_k} \tag{18}$$

it should be clarified that it is defined in (12). Finally, taking (18) and adding the parameter t to it achieves the new conjugate condition:$\boldsymbol{y_k^*}$

$$\boldsymbol{d_{k+1}^T y_k^* = -t g_{k+1}^T s_k} \qquad \boldsymbol{(t \geq 0)} \tag{19}$$

New formula to calculate :$\boldsymbol{\beta_k}$

$$\boldsymbol{\beta_k^{0*}(t) = \frac{g_{k+1}^T y_k^* - t g_{k+1}^T s_k}{d_k^T y_k^*}} \qquad \boldsymbol{t \geq 0} \tag{20}$$

where it is defined in (12), it can be seen that if the objective function f is uniformly convex, then by definition of convexity as we will see later we have: $\boldsymbol{y_k^*}$

$$\begin{aligned} \boldsymbol{d_k^T y_k^*} &= \boldsymbol{d_k^T \left(y_k + \frac{2[f(x_k) - f(x_{k+1})] + \left(g(x_{k+1}) + g(x_k)\right)^T}{\|s_k\|^2} s_k\right)} \\ &= \boldsymbol{d_k^T (g_{k+1} - g_k) + 2\alpha_k^{-1} [f(x_k) - f(x_{k+1})] + (g(x_{k+1}) + g(x_k))^T d_k} \\ &= \boldsymbol{2 d_k^T g_{k+1} + 2\alpha_k^{-1} [f(x_k) - f(x_{k+1})] > 0}, \end{aligned}$$

where the last inequality refers to uniform convexity. Where is well defined as long as f is uniformly convex. However, if f is a general function, it can equal zero. In this case, it is not well defined. So to overcome this problem, the following modification is made.$\boldsymbol{\beta_k^{0*}(t) d_k^T y_k^* \beta_k^{1*}(t)}$

$$\boldsymbol{\hat{y}_k^* = y_k + \frac{max\{v_k, 0\}}{\|s_k\|^2} s_k} \quad (21)$$

y,

$$(22) \quad \boldsymbol{\beta_k^{1*}(t) = \frac{g_{k+1}^T \hat{y}_k^* - t g_{k+1}^T s_k}{d_k^T \hat{y}_k^*}} \quad \boldsymbol{(t \geq 0)}$$

Using this modification, when the linear search used is through Wolfe's strong conditions, as well as the descent condition that will be announced later is well defined as follows:$\boldsymbol{\beta_k^{1*}(t)}$

$\boldsymbol{d_k^T \hat{y}_k^* \geq d_k^T y_k \geq -(1-\delta) g_k^T d_k > 0}$, with $\boldsymbol{0 < \delta < \sigma < 1}$

In addition, following Dai and Liao we arrive at the following formula:

$$\boldsymbol{\beta_k = max\left\{0, \frac{\hat{y}_k^* g_{k+1}^T}{\hat{y}_k^* s_k}\right\} - t \frac{s_k^T g_{k+1}}{\hat{y}_k^{*T} s_k}} \quad (23)$$

3. Numerical Strategy

Table 1.
General Algorithm Conjugate Gradient Method

Step 0 Given $\boldsymbol{x_k \in R^n d_k = -g_k = -\nabla f(x_k)}$.
$\boldsymbol{\varepsilon := 10^{-6}}$
Step 1 if you stop$\|\boldsymbol{g_k}\| \leq \boldsymbol{\varepsilon}$
If you don't find the pitch length because of the strong Wolfe conditions
Step 2 Be $\boldsymbol{x_{k+1} = x_k + \alpha_k d_k}$
Step 3 Find by the corresponding formula and generate $\boldsymbol{\beta_k}$
by $\boldsymbol{d_{k+1} d_{k+1} = -g_k + \beta_k d_k}$
Step 4 Be and get back to the step $\boldsymbol{k := k + 1}$

Table 2.
General Quasi-Newton Method Algorithm

Step 0 Choose a starting point and a symmetrical, defined positive matrix. Be $\boldsymbol{x_1 \in R^n}$ $\boldsymbol{B_1 \varepsilon > 0}$
$\boldsymbol{k = 1}$.

Step 1 If the gradient in satisfies stop$\boldsymbol{g(x) x_k} \|\boldsymbol{g_k}\| \leq \boldsymbol{\varepsilon}$

Step 2 Compute the search address by solving the following linear equation $\boldsymbol{d_k}$
$\boldsymbol{B_k d_k + g_k = 0}$.

Step 3 Research along the direction the step size and calculated$\boldsymbol{d_k \alpha_k}$
$\boldsymbol{x_{k+1} = x_k + \alpha_k d_k}$.

Step 4 Select a new symmetrical and positive defined matrix calculated by the proposed methods that satisfies the equation$\boldsymbol{B_{k+1}}$
$\boldsymbol{B_{k+1} s_k = y_k}$

Step 5 Do and return to step 1$\boldsymbol{k = k + 1}$

4. Results Conclusions

Table 3
Method Results (Modified PR) with PR, BFGS, and DFP Method

PROBLEM	DIMENSION	Modified PR	PR	BFGS	DFP

		NI/NF	CPU TIME (%) approx	NI/NF	TIME CPU (%)	NI/NF	CPU TIME (%)	NI/NF	CPU TIME (%)
Rosenbrock Widespread	20	13/13	59	20/20	-40	86/86	0	51/52	5
	40	16/16	94	20/20	-30	156/156	0	94/95	-3
	60	18/18	25	20/20	-30	222/222	0	147/148	-10
	200	17/17	97	21/21	85	340/341	0	340/341	12
Powell Extendida	20	10/10	74	25/25	-20	30/71	0	16/17	3
	40	15/15	15	25/25	-15	33/13	0	17/18	11
	60	15/15	32	30/30	-10	33/20	0	17/18	8
	200	16/16	97	14/14	75	14/15	0	14/15	-9
Extended Tridiagonal	20	18/18	-21	37/37	-10	15/33	0	11/12	-5
	40	18/18	-7	41/41	-12	15/65	0	12/13	-10
	60	16/16	10	39/39	-13	15/97	0	12/13	-31
	200	17/17	90	20/20	85	13/14	0	13/14	10
Extended Freudenstein &Roth	20	10/10	-22	11/11	-6	7/25	0	7/8	-5
	40	11/10	-27	11/11	-7	7/49	0	9/10	-10
	60	5/5	22	7/7	50	7/73	0	9/10	-50
	200	5/5	99	50/50	92	11/12	0	5/6	4
Fletchcr Function	20	52/52	60	52/52	0	25/54	0	8/9	2
	40	44/44	50	44/44	-12	25/10	0	8/9	4
	60	43/43	12	43/43	25	25/15	0	8/9	2
	200	42/42	60	43/43	55	9/10	0	9/10	3
Rosenbrock Extendida	20	16/16	52	20/20	-12	11/357	0	8/9	20
	40	14/14	48	25/25	-21	11/69	0	8/9	7
	60	16/16	15	18/18	35	11/103	0	8/9	2
	200	27/27	97	30/30	31	19/20	0	19/20	3
Extended White & Holst Function	20	30/30	15	34/34	10	40/123	0	31/32	0
	40	30/30	75	30/30	50	40/282	0	33/34	-3
	60	30/30	85	33/33	75	40/359	0	30/31	-5
	200	33/33	120	50/50	35	33/34	0	33/34	-2

Extended PSC1 Function	20	10/10	-2	10/10	10	8/29	0	6/7	-5
	40	10/10	30	12/12	-10	8/57	0	6/7	-8
	60	10/10	32	10/10	45	8/85	0	6/7	-3
	200	9/9	78	24/24	50	6/7	0	6/7	-4

Table 4
Results of the method (modified PR) with different t-values with the PR method

PROBLEM	DIMENSION	Modified PR t = 0.1		Modified PR t = 0.2		Modified PR t = 0.5		PR	
		NI/NF	CPU TIME (%)	NI/NF	CPU TIME (%)	NI/NF	CPU TIME (%)	NI/NF	CPU TIME (%)
Rosenbrock Extendida	20	16/16	20	16/14	25	16/14	42	20/20	0
	40	14/14	50	14/12	36	14/12	52	20/20	0
	60	16/16	25	16/14	30	16/14	36	20/20	0
	200	27/27	30	27/27	32	27/27	46	30/30	0

According to [6], the Dai Liao method has a good numerical performance by selecting $t = 0.1$. Therefore, here we regulated $t = 0.1$ first and compared the modified PR method with the DFP, PR and BFGS methods.
In this study, the MATLAB package was used to test the chosen problems. The numerical results are listed in Tables 3 and 4, where the items in each column have the following meanings:
Name: the name of the problem in [1];
Dimension: the dimension of the problem;
Modified PR – the test results of the modified PR method;
PR: Conjugate gradient method with Polak-Ribiere conjugate condition (PR)
BFGS: the results of the BFGS method test;
DFP: the results of the DFP method test;
NI: the number of iterations;
NF: the number of evaluations of the function;
CPU Time: Percentage of Execution Time

From Table 3, it can be seen that the results of the modified PR, DFP and PR methods compared to the BFGS in most of the tests, the modified PR shows a significant improvement in the percentage of execution times under the parameter $t = 0.1$. It is noteworthy that the modified PR method is even better as the number of variables increases.
From Table 4 we can infer that a better numerical behavior persists in most of the tests, with the application of the PR method modifying when we use different values of the parameter t.

In the numerical results obtained by comparing the execution times, number of evaluations and number of iterations of the proposed algorithms, it was shown that the gradient method with the proposed modifications was much more convenient since in most of the tests it presented a notable improvement in the variables studied see tables 3 and 4.

BIBLIOGRAPHY

[1] A. Neculai; (2004). Unconstrained optimization test functions
[2] L. Armijo; 16 (1966). 1–3 Minimization of functions having Lipschitz conditions partial derivatives, Pacific J. Math.
[3] Chen X; Sun J; (2002). Global convergence of two-parameter family of conjugate gradient methods without line search, J. Comput. Appl. Math. 146 páginas 37–45.

[4] Y. Dai; 2001. Convergence of Polak–Ribière–Polyak conjugate gradient method wit stepsizes, Manuscript, Institute of Computational Mathematics and Scientific/Engineering Computing, Chinese Academy of Sciences.

[5] W.C. Davidon, Variable metric methods for minimization, Argonne National Labs Report, ANL-5990.

[6] Y. Dai, L.Z. Liao; (2001) New conjugacy conditions and related nonlinear conjugat gradient methods, Appl. Math. Optim. 43 ,87–101.

[7] Y. Dai,Y.Yuan; 1995. Further studies on the Polak–Ribière–Polyak method, Research Report ICM-95-040, Institute of Computational Mathematicsand Scientific/ Engineering Computing, Chinese Academy of Sciences.

[8] Y. Dai; Y.Yuan; 2000. Nonlinear Conjugare Gradient Methods, Science Press of Shanghai, Shanghai.

[9] J.C. Gibert; J. Nocedal; (1992). Global convergence properties of conjugate gradient mato for optimization, SIAM J. Optim. 2 ,21–42.

[10] G. Li; C. Tang y Z. Wei; (2006). New conjugacy condition and related new conjugat gradient methods for unconstrained optimization.

[11] J.J. Moreau; (1965). Proximity and duelito in a Hilbertian space, Bull. Soc. Math. France93, 273–299.

[12] J.J. Morè; B.S. Garbow; K.E. Hillstrome; (1981) - 1995 Testing unconstrained optimization software, AVM Trans. Math. Software 7,17–41.Gradient Related Methods, SIAM,