Improved Hessian Approximation with Modified Quasi-Cauchy Relation for a Gradient-type Method

Wah June Leong[†], Mahboubeh Farid[‡,*] and Malik Abu Hassan[†]

[†] *Department of Mathematics, University Putra Malaysia 43400 Serdang, Selangor, Malaysia*

[‡] *Institute for Mathematical Research, University Putra Malaysia 43400 Serdang, Selangor, Malaysia*

In this work we develop a new gradient-type method with improved Hessian approximation for unconstrained optimization problems. The new method resembles the Barzilai-Borwein (BB) method, except that the Hessian matrix is approximated by a diagonal matrix rather than the multiple of the identity matrix in the BB method. Then the diagonal Hessian approximation is derived based on the quasi-Cauchy relation. To further improve the Hessian approximation, we modify the quasi-Cauchy relation to carry some additional information from the values and gradients of the objective function. Numerical experiments show that the proposed method yields desirable improvement.

Keywords: Diagonal updating; modified quasi-Newton equation; quasi-Cauchy relation; Barzilai-Borwein method

AMS Subject Classification: 90C30; 65K05

1. Introduction

In this paper, we consider the unconstrained optimization problem

$$\min f(x), \quad x \in R^n \tag{1}$$

where $f : R^n \to R$ is a continuously differentiable function with gradient $\nabla f(x) = g(x)$. Secant (or quasi-Newton) method for solving (1) has the following iterative scheme:

$$x_{k+1} = x_k - B_k^{-1} g_k, \quad k = 0, 1, 2, \ldots, \tag{2}$$

where $g_k = \nabla f(x_k)$ denotes the gradient of f at x_k and B_k is an approximation to the Hessian matrix, $\nabla^2 f(x_k)$. To ensure that B has a correct curvature information, B_{k+1} the update of B_k should satisfy the secant or quasi-Newton equation

$$B_{k+1} s_k = y_k \tag{3}$$

where $s_k = x_{k+1} - x_k$ and $y_k = g_{k+1} - g_k$.

Barzilai and Borwein [2] proposed a new gradient method which is derived from a two-point approximation to the secant equation underlying the quasi-Newton

AMO - Advanced Modeling and Optimization. ISSN: 1841-4311

[*]Corresponding author: Mahboubeh Farid (mahboubeh@inspem.upm.edu.my).

methods. By regarding $B_k = \alpha_k I$ and forcing it to satisfy the secant equation, they obtained the following choice of α_k:

$$\alpha_k = \frac{s_{k-1}^T s_{k-1}}{y_{k-1}^T s_{k-1}}. \tag{4}$$

Due to its simplicity, less computational work, low memory requirement and numerical efficiency the BB method has attracted many attentions (see [3] and [7], for examples).

This work is motivated from the application of gradient method to the strictly convex quadratic function $f(x)$ with constant Hessian matrix A. The gradient method with steplength λ_k is the iterative procedure

$$x_{k+1} = x_k - \lambda_k g_k. \tag{5}$$

We can view the gradient method as quasi-Newton method with steplength 1, and the Hessian approximation as a scaled identity, $\alpha_k I$ with $\lambda_k = \alpha_k^{-1}$. The updating scheme (5) on the other hand, can also be viewed as the standard gradient (steepest descent) method, which chooses the steplength α_k such that $\alpha_k = \frac{g_k^T A g_k}{g_k^T g_k}$ or the BB method, which takes the steplength, $\alpha_k = \frac{s_k^T A s_k}{s_k^T s_k}$. Although both α_k are the Rayleigh quotients of A that give points in the interval spectrum of A, their performances vary a lot. Hence, it will be interesting to study the case where the Hessian is approximated by a diagonal matrix with a larger interval spectrum that might better overlap the interval spectrum of the Hessian matrix. Arm with this encouragement, in this paper we attempt to obtain a diagonal updating formula that will be used within the updating scheme (2). The formulation of the diagonal updating will be given in Section 2. It follows by computational results in Section 3 to illustrate the merit of the new algorithm.

2. Diagonal Updating via Modified Quasi-Cauchy Relation

Before we set out to derive the desired diagonal updating formula, recall that by the secant equation (3), we see that y is an approximation to $\nabla^2 f(x)s$. Since y is a vector and the standard mean-valued theorem might not holds for vector-valued functions, we do not know whether there will exist $\bar{x} \in R^n$ such that $y = \nabla^2 f(\bar{x})s$ or not. However, Taylor theorem and mean-valued theorem ensure that the existence of such $\bar{x}$ so that $s^T y = s^T \nabla^2 f(\bar{x}) s$. Hence, it might be reasonable to let our diagonal matrix satisfies only the quasi-Cauchy relation [9]:

$$s_k^T B_{k+1} s_k = s_k^T y_k. \tag{6}$$

Furthermore, without evaluating additional functions or gradients, the only Hessian information we have is y, and this is only approximation information. Thus, in order to better approximate the Hessian matrix, we can incorporate additional information from both functions and gradients to y. Two type of modifications on y are considered: The first type is given by Li and Fukushima [6]) where y is replaced by

$$\hat{y}_k = y_k + \nu_k \|g_k\| s_k \tag{7}$$

with $\nu_k = 1 + \max\{-\frac{s_k^T y_k}{\|s_k\|^2}, 0\}$. In the second type, we consider a modification of Wei et al. [8] where

$$\hat{y}_k = y_k + \omega_k s_k \tag{8}$$

with $\omega_k = \frac{2(f(x_k)-f(x_{k+1}))+(g_{k+1}+g_k)^T s_k}{(s_k^T y_k)^2} y_k y_k^T$.

We can now give our updating formula for the diagonal Hessian approximation. For algorithmic purpose, we choose to derive the updating formula for the inverse Hessian approximation, U_{k+1} by letting it satisfies, instead the modified quasi-Cauchy equation, $\hat{y}_k^T U_{k+1} \hat{y}_k = \hat{y}_k^T s_k$ as follows:

THEOREM 2.1 *Assume that $U_k > 0$ is a positive definite diagonal matrix and U_{k+1} is the updated version of U_k, which is also diagonal. Suppose that $s_k \neq 0$ and $\hat{y}_k$ is defined either by (7) or (8). Then the optimal solution of the following minimization problem:*

$$\begin{aligned} min \quad & \frac{1}{2}\|U_{k+1} - U_k\|_F^2 \\ s.t. \quad & \hat{y}_k^T U_{k+1} \hat{y}_k = \hat{y}_k^T s_k \\ and \;\; & U_{k+1} \text{ is diagonal} \end{aligned} \tag{9}$$

is given by

$$U_{k+1} = U_k + \frac{(\hat{y}_k^T s_k - \hat{y}_k^T U_k \hat{y}_k)}{tr(G_k^2)} G_k, \tag{10}$$

where $G_k = diag\Big(\hat{y}_{k,1}^2, \hat{y}_{k,2}^2, \ldots, \hat{y}_{k,n}^2\Big)$ and $\hat{y}_{k,i}$ is the ith component of the vector $\hat{y}_k$.

Proof Let $\Omega_k = U_{k+1} - U_k$. The Lagrangian function for (9) in term of Ω_k is given by

$$L(\Omega_k, \mu) = \frac{1}{2}\|\Omega_k\|_F^2 + \mu\left(\hat{y}_k^T \Omega_k \hat{y}_k + \hat{y}_k^T U_k \hat{y}_k - \hat{y}_k^T s_k\right) \tag{11}$$

where μ is the Lagrange multiplier associated with the constraint. Differentiating L with respect to each component of Ω_k and setting the results to zero yields

$$\Omega_k = -\mu G_k \tag{12}$$

It follows that $\hat{y}_k^T \Omega_k \hat{y}_k = -\mu \hat{y}_k^T G_k \hat{y}_k = -\mu\, tr(G_k^2)$ and invoking the constraint, we have

$$\mu = \frac{(\hat{y}_k^T U_k \hat{y}_k - \hat{y}_k^T s_k)}{tr(G_k^2)}. \tag{13}$$

Finally, by substituting (13) into (12) we obtain (10). ■

Note that there is no guarantee that the diagonal matrix U_{k+1}, updated through (10) is always positive-definite. Furthermore, it can be shown that (see [5] for

details) if

$$u_{k+1,m} < \frac{u_{k,M}^2}{2u_{k,m}} \tag{14}$$

where $u_{k,m}$, $u_{k,M}$, $u_{k+1,m}$ and $u_{k+1,M}$ are the smallest and largest diagonal component of U_k and U_{k+1}, respectively, then possibility of non-monotonic behavior occurs in the sequence $\{f(x_k)\}$ might be observed. Hence, some strategies are needed to safeguard these difficulties. The details of the algorithm are given as follows:

M-DiaGRAD Algorithm

- **Step 0.** Given an initial point x_0 and a positive definite diagonal matrix U_0. Set $k = 0$.
- **Step 1.** If $\|g_k\| \leq \epsilon$ then stop.
- **Step 2.** If $k = 0$, compute $x_1 = x_0 - \frac{g_0}{\|g_0\|}$. Else if $k \geq 1$, compute $x_{k+1} = x_k - U_k g_k$ and update U_{k+1}.
- **Step 3.** Check whether $U_{k+1} < 0$ or $u_{k+1,m} < \frac{u_{k,M}^2}{2u_{k,m}}$.

  If yes, set $U_{k+1} = \rho I$ where $\rho = \min\left(\frac{2u_{k,m}}{u_{k,M}^2}, \frac{\hat{y}_k^T s_k}{\hat{y}_k^T \hat{y}_k}\right)$. Otherwise retain U_{k+1} that is computed in Step 2.
- **Step 4.** set $k := k + 1$ and return to step 1.

We shall also establish the convergence of the M-DiaGRAD algorithm when applied to the minimization of a strictly convex quadratic function with constant Hessian matrix A. Note that if $U_{k+1} > 0$, then we have

$$y_k^T A^{-1} y_k = y_k^T U_{k+1} y_k \leq u_{k+1,M} \|y_k\|^2.$$

Thus we assume that $s_k^T A s_k \leq u_{k+1,m}^{-1} \|s_k\|^2$.

THEOREM 2.2 *Let $\{x_k\}$ be a sequence generated by the M-DiaGRAD Algorithm and x^* is the unique minimizer of a strictly convex quadratic function f with constant Hessian matrix A. Then either $g_k = 0$ holds for some finite $k \geq 1$ or $lim_{k\to\infty}\|g_k\| = 0$.*

Proof Consider the Taylor expansion of f at x_{k+1}:

$$f(x_k - U_k g_k) = f(x_k) - g_k^T U_k g_k + \frac{1}{2} g_k U_k A U_k g_k. \tag{15}$$

It follows that

$$f(x_{k+1}) \leq f(x_k) - c\|g_k\|^2, \tag{16}$$

where $c = u_{k,m} - \frac{1}{2} u_{k,M}^2 u_{k+1,m}^{-1}$. If $c > 0$ (or $u_{k+1,m} > \frac{u_{k,M}^2}{2u_{k,m}}$), we have $f(x_{k+1}) \leq f(x_k)$ for all k and since f is bounded below, it follows that

$$\lim_{k\to\infty} f(x_k) - f(x_{k+1}) = 0.$$

Else if $c \leq 0$ or $U_{k+1} < 0$, then we let $U_{k+1} = \rho I$ where $\rho = \min\left(\frac{2u_{k,m}}{u_{k,M}^2}, \frac{\hat{y}_k^T s_k}{\hat{y}_k^T \hat{y}_k}\right)$.

Hence, (16) becomes

$$f(x_{k+1}) \leq f(x_k) - \bar{c}\|g_k\|^2.$$

where $\bar{c} = u_{k,M}^{-1} - \left(u_{k,m}^{-2}\rho^{-1}\right)/2$. With the choice of our ρ, we have that $\bar{c} \geq 0$. This also implies that $f(x_{k+1}) \leq f(x_k)$ for all k. As $f(x_k) - f(x_{k+1}) \to 0$, then $\lim_{k\to\infty} \|g_k\| = 0$. ■

3. Numerical Results

We report the performance of M-DiaGRAD algorithm and its comparison to the modified BB method with $\hat{\alpha}_k = \frac{s_{k-1}^T s_{k-1}}{\hat{y}_{k-1}^T s_{k-1}}$. All of these methods belong to a class of gradient methods without line searches. A set of 28 standard unconstrained optimization problems given in Table 1, were used with dimension varying from 10 up to 10000. The full description of these test problems can be found in [1]. All algorithms were coded in MATLAB 7.0 and were executed on a PC with Core Duo CPU. All runs were terminated if $\|g_k\| \leq 10^{-4}$. The routine was also forced to stop when the number of iteration exceed 1000. We implement two versions of the modified BB and M-DiaGRAD algorithms: BB(1) and M-DiaGRAD(1) use $\hat{y}_k$ that is defined by (7) whereas BB(2) and M-DiaGRAD(2) use (8) to compute $\hat{y}_k$. The performances of modified BB and M-DiaGRAD method, relative to iteration, are given using the performance profiling of Dolan and Morè [4]. More detailed are listed in table 1 and table 2. The symbol "-" in tables indicates that the method fails to converge within 1000 iterations. We observed from Figure 1 that the M-DiaGRAD algorithms obtain vast improvement over the BB method with an average of 40% decreases in number of iterations. It is also worth noting that such improvement can be achieved only with an extra n storage and a slight increment in computational costs.

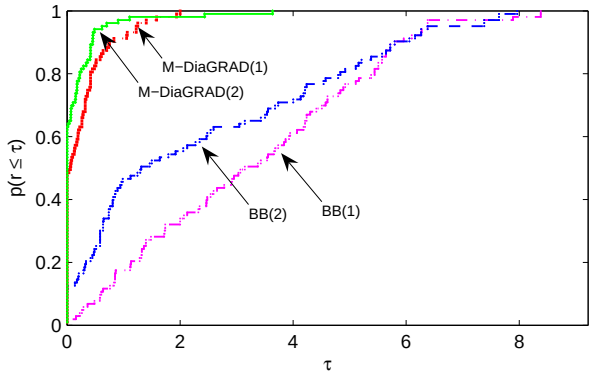

Figure 1. Performance profile based on iterations.

4. Conclusion

We have presented two gradient-type methods for unconstrained optimization based on the modified quasi-Cauchy equations. These methods can be considered as an extension of the BB method with improved Hessian approximation. Using a

simple strategy, the new methods are globally converged when minimizing a strictly convex quadratic function. Numerical experiments on a large number of problems indicate that our methods are very promising.

Acknowledgements

This work is supported in part by the Malaysian Fundamental Research Grant Scheme (no. 05-10-07-383FR) and the first author also acknowledges support by the joint Chinese Academy of Sciences (CAS) and Academy of Sciences for the Developing Worlds (TWAS) Fellowship (no. 3240157252).

References

[1] N. Andrei, An unconstrained optimization test functions collection, Adv. Model. Optim. 10 (2008) 147-161.
[2] J. Barzilai and J.M. Borwein, Two point step size gradient methods, IMA J. Numer. Anal. 8 (1988) 141-148.
[3] Y.H. Dai and L.Z. Liao, R-linear convergence of the Barzilai and Borwein gradient method, IMA J. Numer. Anal. 22 (2002) 1-10.
[4] E.D. Dolan and J.J. More, Benchmarking optimization software with perpormance profiles, Math. Program. 91 (2002) 201-213.
[5] M.A. Hassan, W.J. Leong and M. Farid, A new gradient method via quasi-Cauchy relation which guarantees descent, J. Comput. Appl. Math. (2008) doi:10.1016/j.cam.2008.11.013.
[6] D.H. Li and M. Fukushima, A modified BFGS method and its global convergence in nonconvex minimization, J. Comput. Appl. Math. 129 (2001) 15-35.
[7] M. Raydan, On the Barzilai and Borwein choice of steplength for the gradient method, IMA J. Numer. Anal. 13 (1993) 618-622.
[8] Z. Wei, G.Li and L. Qi, New quasi-Newton methods for unconstrained optimization problems, Appl. Math. Comput. 175 (2006) 1156-1188.
[9] M. Zhu, J.L. Nazareth and H. Wolkowicz, The quasi-Cauchy relation and diagonal updating, SIAM J. Optim. 9 (1999) 1192-1204.

Table 1. BB and MODIFYMONOCAUCHY TYPE1

Test Function (Dimension)	Initial point x_0	BB(1)	BB(2)	M-DiaGRAD(1)	M-DiaGRAD(2)
Extended Freudenstein and Roth (10)	$x_0 = (1, 1, ..., 1)$	-	386	46	22
Extended Freudenstein and Roth (100)	$x_0 = (1, 1, ..., 1)$	-	353	37	22
Extended Freudenstein and Roth (1000)	$x_0 = (1, 1, ..., 1)$	-	408	36	448
Extended Freudenstein and Roth (10000)	$x_0 = (1, 1, ..., 1)$	-	87	29	23
Extended Trigonometric (10)	$x_0 = (0.2, 0.2, ..., 0.2)$	6	-	8	6
Extended Trigonometric (100)	$x_0 = (0.2, 0.2, ..., 0.2)$	46	-	23	21
Extended Trigonometric (1000)	$x_0 = (0.2, 0.2, ..., 0.2)$	78	-	43	59
Extended Trigonometric (10000)	$x_0 = (0.2, 0.2, ..., 0.2)$	-	-	68	53
Extended Beale (10)	$x_0 = (1, 0.8, 1, 0.8, ..., 1, 0.8)$	-	-	172	180
Extended Beale (100)	$x_0 = (1, 0.8, 1, 0.8, ..., 1, 0.8)$	-	-	229	229
Extended Beale (1000)	$x_0 = (1, 0.8, 1, 0.8, ..., 1, 0.8)$	-	-	197	261
Extended Beale (10000)	$x_0 = (1, 0.8, 1, 0.8, ..., 1, 0.8)$	-	-	181	294
Perturbed Quadratic (10)	$x_0 = (0.5, 0.5, ..., 0.5)$	45	25	18	19
Perturbed Quadratic (100)	$x_0 = (0.5, 0.5, ..., 0.5)$	703	87	86	90
Perturbed Quadratic (1000)	$x_0 = (0.5, 0.5, ..., 0.5)$	-	351	322	304
Raydan 1 (10)	$x_0 = (1, 1, ..., 1)$	18	20	14	18
Raydan 1 (100)	$x_0 = (1, 1, ..., 1)$	156	157	64	88
Raydan 1 (1000)	$x_0 = (1, 1, ..., 1)$	-	-	179	184
Raydan 2 (10)	$x_0 = (1, 1, ..., 1)$	7	6	7	6
Raydan 2 (100)	$x_0 = (1, 1, ..., 1)$	13	6	8	6
Raydan 2 (1000)	$x_0 = (1, 1, ..., 1)$	30	6	8	6
Raydan 2 (10000)	$x_0 = (1, 1, ..., 1)$	80	6	8	7
Diagonal 1 (10)	$x_0 = (\frac{1}{n}, \frac{1}{n}, ..., \frac{1}{n})$	36	22	24	20
Diagonal 1 (100)	$x_0 = (\frac{1}{n}, \frac{1}{n}, ..., \frac{1}{n})$	776	523	185	-
Diagonal 2 (10)	$x_0 = (\frac{1}{1}, \frac{1}{2}, ..., \frac{1}{n})$	18	15	20	23
Diagonal 2 (100)	$x_0 = (\frac{1}{1}, \frac{1}{2}, ..., \frac{1}{n})$	68	70	61	64
Diagonal 2 (500)	$x_0 = (\frac{1}{1}, \frac{1}{2}, ..., \frac{1}{n})$	121	-	145	144
Diagonal 5 (10)	$x_0 = (1.1, 1.1, ..., 1.1)$	7	5	7	5
Diagonal 5 (100)	$x_0 = (1.1, 1.1, ..., 1.1)$	14	5	5	5
Diagonal 5 (1000)	$x_0 = (1.1, 1.1, ..., 1.1)$	31	5	5	4
Diagonal 5 (10000)	$x_0 = (1.1, 1.1, ..., 1.1)$	84	5	6	5
Extended Himmelblau (10)	$x_0 = (1, 1, ..., 1)$	152	18	12	12
Extended Himmelblau (100)	$x_0 = (1, 1, ..., 1)$	430	20	12	12
Extended Himmelblau (1000)	$x_0 = (1, 1, ..., 1)$	-	21	12	12
Extended Himmelblau (10000)	$x_0 = (1, 1, ..., 1)$	-	21	12	12
Generlized Rosenbrock (10)	$x_0 = (-1.2, 1, -1.2, 1, ..., -1.2, 1)$	-	-	18	20
Generlized Rosenbrock (100)	$x_0 = (-1.2, 1, -1.2, 1, ..., -1.2, 1)$	-	-	19	24
Generlized Rosenbrock (1000)	$x_0 = (-1.2, 1, -1.2, 1, ..., -1.2, 1)$	-	-	20	28
Generlized Rosenbrock (10000)	$x_0 = (-1.2, 1, -1.2, 1, ..., -1.2, 1)$	-	-	19	26
Generlized PSC1 (10)	$x_0 = (3, 0.1, 3, 0.1, ..., 3, 0.1)$	592	36	23	20
Generlized PSC1 (100)	$x_0 = (3, 0.1, 3, 0.1, ..., 3, 0.1)$	-	38	27	21
Generlized PSC1 (1000)	$x_0 = (3, 0.1, 3, 0.1, ..., 3, 0.1)$	-	39	26	23
Generlized PSC1 (10000)	$x_0 = (3, 0.1, 3, 0.1, ..., 3, 0.1)$	-	40	27	24
Extended PSC1 (10)	$x_0 = (3, 0.1, 3, 0.1, ..., 3, 0.1)$	319	-	15	15
Extended PSC1 (100)	$x_0 = (3, 0.1, 3, 0.1, ..., 3, 0.1)$	959	-	15	14
Extended PSC1 (1000)	$x_0 = (3, 0.1, 3, 0.1, ..., 3, 0.1)$	-	-	13	14
Extended PSC1 (10000)	$x_0 = (3, 0.1, 3, 0.1, ..., 3, 0.1)$	-	-	12	14
Diagonal 3 (10)	$x_0 = (1, 1, ..., 1)$	50	79	19	19
Diagonal 3 (100)	$x_0 = (1, 1, ..., 1)$	469	147	62	85
Diagonal 3 (1000)	$x_0 = (1, 1, ..., 1)$	-	-	253	166
Hager (10)	$x_0 = (1, 1, ..., 1)$	11	11	9	11
Hager (100)	$x_0 = (1, 1, ..., 1)$	70	26	21	22
Hager (1000)	$x_0 = (1, 1, ..., 1)$	685	75	46	45
Hager (10000)	$x_0 = (1, 1, ..., 1)$	-	-	113	128
Generlized Tridiagonal 1 (10)	$x_0 = (2, 2, ..., 2)$	35	29	23	26
Generlized Tridiagonal 1 (100)	$x_0 = (2, 2, ..., 2)$	59	36	23	26
Generlized Tridiagonal 1 (1000)	$x_0 = (2, 2, ..., 2)$	132	37	24	21
Generlized Tridiagonal 1 (10000)	$x_0 = (2, 2, ..., 2)$	370	32	25	24
Extended Three Exponential Terms (10)	$x_0 = (0.1, 0.1, ..., 0.1)$	21	12	10	8
Extended Three Exponential Terms (100)	$x_0 = (0.1, 0.1, ..., 0.1)$	32	12	8	9
Extended Three Exponential Terms (1000)	$x_0 = (0.1, 0.1, ..., 0.1)$	67	12	12	8
Extended Three Exponential Terms (10000)	$x_0 = (0.1, 0.1, ..., 0.1)$	182	12	10	8
Generlized Tridiagonal 2 (10)	$x_0 = (-1, -1, ..., -1)$	160	-	69	29
Generlized Tridiagonal 2 (100)	$x_0 = (-1, -1, ..., -1)$	-	-	47	33
Generlized Tridiagonal 2 (1000)	$x_0 = (-1, -1, ..., -1)$	-	-	55	103
Generlized Tridiagonal 2 (10000)	$x_0 = (-1, -1, ..., -1)$	-	-	112	54
Broydan Tridiagonal (10)	$x_0 = (-1, -1, ..., -1)$	165	-	39	24
Broydan Tridiagonal (100)	$x_0 = (-1, -1, ..., -1)$	61	-	34	28
Broydan Tridiagonal (1000)	$x_0 = (-1, -1, ..., -1)$	280	-	48	27
Broydan Tridiagonal (5000)	$x_0 = (-1, -1, ..., -1)$	525	-	56	36
Almost Perturbed Quadratic (10)	$x_0 = (0.5, 0.5, ..., 0.5)$	43	25	13	15
Almost Perturbed Quadratic (100)	$x_0 = (0.5, 0.5, ..., 0.5)$	687	95	60	48
Almost Perturbed Quadratic (1000)	$x_0 = (0.5, 0.5, ..., 0.5)$	-	282	-	250
Tridiagonal perturbed quadratic (10)	$x_0 = (\frac{1}{n}, \frac{1}{n}, ..., \frac{1}{n})$	112	-	287	75
Tridiagonal perturbed quadratic (100)	$x_0 = (\frac{1}{n}, \frac{1}{n}, ..., \frac{1}{n})$	515	-	547	207
Extended Block Diagonal BD1 (10)	$x_0 = (0.1, 0.1, ..., 0.1)$	14	14	9	9
Extended Block Diagonal BD1 (100)	$x_0 = (0.1, 0.1, ..., 0.1)$	16	14	9	9
Extended Block Diagonal BD1 (1000)	$x_0 = (0.1, 0.1, ..., 0.1)$	30	14	9	9
Extended Block Diagonal BD1 (10000)	$x_0 = (0.1, 0.1, ..., 0.1)$	75	14	10	10
Quadratic QF2 (10)	$x_0 = (1, 1, ..., 1)$	9	-	6	6
Quadratic QF2 (100)	$x_0 = (1, 1, ..., 1)$	9	-	5	5
Quadratic QF2 (1000)	$x_0 = (1, 1, ..., 1)$	9	-	5	5
Quadratic QF2 (10000)	$x_0 = (1, 1, ..., 1)$	10	-	4	4

Table 2. BB and MODIFYMONOCAUCHY TYPE1

Test Function (Dimension)	Initial point x_0	BB(1)	BB(2)	M-DiaGRAD(1)	M-DiaGRAD(2)
Extended Tridiagonal 2 (10)	$x_0 = (1.5, 1.5, ..., 1.5)$	-	-	54	60
Extended Tridiagonal 2 (100)	$x_0 = (1.5, 1.5, ..., 1.5)$	-	-	89	85
Extended Tridiagonal 2 (1000)	$x_0 = (1.5, 1.5, ..., 1.5)$	-	-	93	94
Extended Tridiagonal 2 (10000)	$x_0 = (1.5, 1.5, ..., 1.5)$	-	-	95	78
Penalty 1 (10)	$x_0 = (1, 2, ..., n)$	-	39	24	21
Penalty 1 (100)	$x_0 = (1, 2, ..., n)$	-	72	32	29
Penalty 1 (1000)	$x_0 = (1, 2, ..., n)$	-	142	56	41
Penalty 1 (10000)	$x_0 = (1, 2, ..., n)$	-	146	70	53
Penalty 2 (20)	$x_0 = (1, 2, ..., n)$	683	15	12	11
Penalty 2 (100)	$x_0 = (1, 2, ..., n)$	-	39	21	20
Penalty 2 (400)	$x_0 = (1, 2, ..., n)$	683	61	28	31
Penalty 2 (10000)	$x_0 = (1, 2, ..., n)$	-	109	50	35
EG 2 (10)	$x_0 = (1, 1, ..., 1)$	22	-	13	28
EG 2 (100)	$x_0 = (1, 1, ..., 1)$	691	-	35	35
EG 2 (1000)	$x_0 = (1, 1, ..., 1)$	-	-	42	42
EG 2 (10000)	$x_0 = (1, 1, ..., 1)$	-	-	86	86
Diagonal 4 (10)	$x_0 = (1, 1, ..., 1)$	232	3	5	4
Diagonal 4 (100)	$x_0 = (1, 1, ..., 1)$	712	3	7	4
Diagonal 4 (1000)	$x_0 = (1, 1, ..., 1)$	-	3	7	4
Diagonal 4 (10000)	$x_0 = (1, 1, ..., 1)$	-	3	9	4