Punjab University Journal of Mathematics
(ISSN 1016-2526)
Vol. 52(9)(2020) pp.65-77

Comparative Analysis of Single/multi-step quasi-Newton Methods at Different Delta Values

Farah Jaffar
Department of Mathematics
Shaheed Benazir Bhutto Women University, Peshawar, Pakistan,
Email: farah.maths@gmail.com

Nudrat Aamir
Department of Basic Sciences and Humanities
CECOS University of IT and Emerging Sciences, Peshawar, Pakistan,

Received: 20 January, 2020 / Accepted: 17 July, 2020 / Published online: 07 September, 2020

Abstract.: In this paper the author has considered two-step methods proposed by Ford and Moghrabi with new values of delta. As, it is evident in literature that quasi-Newton methods (single/multi-step) for solution of nonlinear unconstrained problems outperforms all other methods available. In case of multi-step methods it can be noted that different choice of τ values effects the performance of algorithm. Furthermore, it can be noted that variables in scalar and gradient space depends upon delta (δ) value. Therefore, in this paper we have considered different choices of δ values to check the performance of algorithm on 15 test problems. Numerical experiments reveals that performance of algorithm improves when δ is taken as 0.85 and 0.95.

AMS (MOS) Subject Classification Codes: 65K05; 65K99
Key Words: Quasi-Newton method, Multi-step quasi-Newton method, Unit-spaced method, Delta values.

1. Introduction

Major development have been made in the area of numerical methods of unconstrained optimization in 1960 in United Kingdom. Various methods such as steepest descent, Newton method, quasi-Newton method etc for unconstrained nonlinear optimization problems have been developed to find the solution of different optimization problems [11]. Every method has some limitations as well as some strengths. Some methods have linear convergence such as steepest descent, conjugate gradient method and some have quadratic convergence such as Newton method. It's convergence depend on good initial guess. It

also requires the information of first and second derivative at each iteration to find the solution of linear system but, for large dimension it can be very expensive. To overcome these limitations, in 1959 Davidon introduced quasi-Newton methods which have fast convergence rate than Newton method and do not require to compute derivatives at each iteration. It was further developed by Ford and Moghrabi [2, 7] as multi-step quasi-Newton methods and achieved good results as compare to quasi-Newton methods.
We consider unconstrained optimization problems define as

$$minf(x), \quad where \quad f : R^n \rightarrow R$$

There is no requirement of Hessian (M_{i+1}) matrix computation in quasi-Newton method but it focuses on the approximation of Hessian or its inverse Hessian by using updating formulas such as Symmetric Rank-1 (SR1), David-Fletcher-Powell (DFP) and Broyden-Fletcher-Goldfarb-Shanno formula (BFGS) etc. The updated Hessian approximation i.e (B_{i+1}) is required to satisfy the secant condition, given as

$$B_{i+1}s_i = y_i. \tag{1. 1}$$

Where s_i be the step size in the variable space x_i

$$s_i = x_{i+1} - x_i, \tag{1. 2}$$

and y_i is the step size in gradient space $g(x_i)$, such as

$$y_i = g(x_{i+1}) - g(x_i). \tag{1. 3}$$

Yang et.al [1] investigated the performance of four quasi-Newton methods for nonlinear unconstrained optimization problems and found that an updating formula BFGS is superior than the other methods. Ford and Moghrabi [2] introduced a new method known as multi-step method which is the generalization of quasi-Newton method by using interpolating curve. Moghrabi and Kassar [10] investigated new parameterization techniques for multi-step methods for the approximation of Hessian. Ford and Moghrabi [4] also introduced an alternating multi-step methods for unconstrained nonlinear optimization problems and achieved better results. To improve the performance of multi-step quasi-Newton methods Ford [6] introduced implicit updates. These updates are further developed by Ford and Tharmlikit [3] in 2003.

In multi-step quasi-Newton methods, the path *X* is define as a polynomial $x(\tau)$ of degree *m* and it interpolated the following points

$$x(\tau_k) = x_{i-m+k+1}, \quad for \quad k = 0, 1, 2, ..., m.$$

The polynomial $x(\tau)$ will depend on $\{\tau_k\}_{k=0}^m$ values which corresponds to iterates $\{x_{i-m+k+1}\}_{k=0}^m$. The Lagrangian polynomial $\mathcal{L}_k$ is used for interpolating the path *X* as

$$x(\tau) = \sum_{k=0}^{m} \mathcal{L}_k(\tau) x_{i-m+k+1}, \tag{1. 4}$$

and define as

$$\mathcal{L}_k(\tau) \equiv \prod_{j=0, j\neq k}^{m} \frac{\tau - \tau_j}{\tau_k - \tau_j}.$$

Let $g(x(\tau))$ is a function which is approximated by interpolating polynomial $x(\tau)$ based on gradient values from iterates $\{x_{i-m+k+1}\}_{k=0}^{m}$ and obviously on $\{\tau_k\}_{k=0}^{m}$ which are used for the construction of path *X*. Therefore, we have

$$g(x(\tau)) = \sum_{k=0}^{m} \mathcal{L}_k(\tau) g(x_{i-m+k+1}). \tag{1. 5}$$

In order to apply the Newton's equation we use (1. 4) to estimate $\frac{dx(\tau)}{d\tau}$ and (1. 5) to find $\frac{dgx(\tau)}{d\tau}$, we obtain the condition which is satisfied by updated Hessian approximation B_{i+1}

$$\frac{dx(\tau_m)}{d\tau} = \sum_{k=0}^{m} \mathcal{L}'{}_k(\tau_m) x_{i-m+k+1}.$$

$$= r_i, \tag{1. 6}$$

$$\frac{g(x(\tau_k))}{d\tau} \approx \sum_{k=0}^{m} \mathcal{L}'{}_k(\tau_m) g(x_{i-m+k+1}).$$

$$= w_i, \tag{1. 7}$$

where

$$\mathcal{L}'{}_k(\tau_m) = (\tau_k - \tau_m)^{-1} \prod_{j=0, j\neq k}^{m} \frac{\tau - \tau_j}{\tau_k - \tau_j}, (k \neq m)$$

$$\mathcal{L}'{}_k(\tau_m) = \sum_{j=0}^{m-1} (\tau_k - \tau_m)^{-1}.$$

r_i and w_i can be represented in terms of most recent vectors $\{s_{i-j}\}_{i=0}^{m-1}$ and $\{y_{i-j}\}_{i=0}^{m-1}$, such as

$$r_i = \sum_{i=0}^{m-1} s_{i-j} \left(\sum_{k=m-j}^{m} \mathcal{L}'{}_k(\tau_m) \right),$$

$$w_i = \sum_{i=0}^{m-1} y_{i-j} \left(\sum_{k=m-j}^{m} \mathcal{L}'{}_k(\tau_m) \right).$$

In case of multi-step methods, the secant equation (1. 1) will be of the form as

$$B_{i+1} r_i = w_i. \tag{1. 8}$$

In BFGS formulae s_i and y_i are replaced by r_i and w_i to get the updated Hessian approximation B_{i+1} in multi-step methods as

$$B_{i+1} = B_i + \frac{B_i r_i r_i^T B_i}{r_i^T B_i r_i} + \frac{w_i w_i^T}{w_i^T r_i}.$$

Similarly, updated inverse Hessian approximation is define as

$$H_{i+1} = \left(I - \frac{r_i w_i^T}{w_i^T r_i} \right) H_i \left(I - \frac{w_i r_i^T}{w_i^T r_i} \right) + \frac{r_i r_i^T}{w_i^T r_i}.$$

Hence the updated inverse Hessian approximation is required to satisfy

$$H_{i+1} w_i = r_i.$$

the updating conditions for two-step method are

$$r_i = s_i - \frac{\delta^2}{(2\delta+1)} s_{i-1}, \tag{1. 9}$$

$$w_i = y_i - \frac{\delta^2}{(2\delta+1)} y_{i-1}, \tag{1. 10}$$

where

$$\delta = \frac{(\tau_2 - \tau_1)}{(\tau_1 - \tau_0)}.$$

Farzin et.al [9] applied the Symmetric Rank-one (SR1) method by using interpolatory polynomials in multi-step methods and obtained significant results while comparing with single-step method. Ford and Maghrabi [2] also presented fixed-point and accumulative approaches under metric based method, used to determine the parametric values $\tau' s$ by measuring the distance between different iterates in the current interpolation by norm define by positive-definite matrix N.

$$\phi_N = ((z_1 - z_2)^T N (z_1 - z_2))^{\frac{1}{2}}. \tag{1. 11}$$

The matrix N is taken as I, B_i, B_{i+1} where $z_1, z_2 \quad \epsilon \quad R^n$.
Multi-step skipping method is investigated by Ford and Nudrat [5] in which skipping technique was implemented and achieved good results. In the [5] the authors also introduced modified search direction with skipping technique and achieved good results. Waziri et.al [13] presented a new approach of an improved diagonal secant-type method for the solution of large scale nonlinear system. Ford and Maghrabi [2] gave the idea of unit-spacing for two-step and three-step methods to parameterize the interpolating curve. In single-step methods where *m*=1, we can choose the values of $\{\tau_k\}_{k=0}^m$ i.e $\tau_0 = 0$ and $\tau_1 = 1$. In unit-spacing the values of τ can be defined as

$$\tau_k = k - m + 1, \quad for \quad k = 0, 1, 2, ..., m.$$

For unit-spaced method Eq.(1. 1) can be written as

$$B_{i+1} r_i = w_i,$$

as the vectors r_i and w_i are given by

$$r_i = s_i,$$

$$w_i = y_i.$$

In second section different values of delta are proposed. The numerical results of all comparable methods with different delta values of all dimension is shown in section 3 and section 4 included the concluding remarks of proposed research.

Table 1: **Computation of different delta values**

δ	r_j	w_j
0.15	$s_i - (0.01730)s_{i-1}$	$y_j - (0.01730)y_{i-1}$
0.25	$s_i - (0.04166)s_{i-1}$	$y_i - (0.04166)y_{i-1}$
0.5	$s_i - (0.125)s_{i-1}$	$y_i - (0.125)y_{i-1}$
0.75	$s_i - (0.225)s_{i-1}$	$y_i - (0.225)y_{i-1}$
0.85	$s_i - (0.26759)s_{i-1}$	$y_i - (0.26759)y_{i-1}$
0.95	$s_i - (0.31120)s_{i-1}$	$y_i - (0.31120)y_{i-1}$
2.0	$s_i - (0.8)s_{i-1}$	$y_i - (0.8)y_{i-1}$

2. Proposed Delta Values

It is evident from equation (1. 9) and (1. 10) that values of r_i and w_i depends upon delta value. Hence, instead of delta value considered in literature, we propose to investigate the effect of different values of delta on algorithm. These delta values generates different values of r_i and w_i (by using Eq.(1. 9) and Eq.(1. 10)), as presented in Table 1. The algorithms produced by different values of delta are then compared with standard single-step method (where delta is 0) and unit-spaced method (where delta is 1) to check the performance of new algorithms.

3. Numerical Results

The performance of different algorithms is analyzed by first considering total number of function evaluations, lesser the number of total evaluation better the performance of algorithm. However, in case if function evaluations in two different methods are same we will check the number of iterations and then the time spent on experiments per second. All the methods are investigated 20 times and results are obtained on the basis of their averages. In order to compare the performance of algorithms, produced by new delta values, 15 test functions (already present in literature [8]) with four different starting points are considered. There is a possibility of algorithm failure which is controlled by epsilon value. Hence, four different starting points and epsilon values for each test function are reported in table 4-7. In table 4-7 square brackets [,] represents the repetition of starting point for the dimension considered. These test problems are further subdivided into three categories, soft, medium and hard. Soft problems have dimension ranging from 2 to 20, medium dimension ranging from 21 to 60 and dimension of hard problems ranges 61 to 200, as presented in table 3. Analysis of different δ values on the problems of different dimensions has been displayed in table 2 and figure 1.

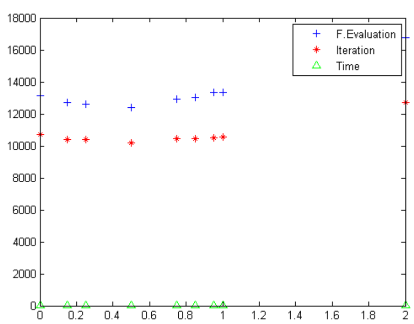
- It is evident from table 2 and figure 1(a) that in soft dimension the δ value 0.5 is out performing all other delta values in case of function evaluations and number of iterations. On the other hand $\delta = 0.15$ is wining over all reported delta values in case of time spent in seconds for experimentation.
- Furthermore, it can be noticed in table 2 and in figure 1(b) that in medium dimension the performance of method $\delta = 0.95$ is better in case of function evaluations and number of iterations than the other methods. While the method $\delta = 0.85$ shows less computational time over all other delta values.
- The results reported in table 2 and figure 1(c) proved that in hard dimension δ value 0.85 is out performing in case of function evaluation than other considered delta values. On the other hand unit-spaced method shows less number of iteration in less computational time over all other methods.
- From table 2 and figure 1(d), the results of combined dimension exhibited same performance as executed by the results of hard dimension problems that is the performance of method where δ = 0.85 is outperforming in function evaluation as compared to standard BFGS and unit-spaced method.

It is evident from the results that the method at the δ value 2.0, failed to perform well in terms of function evaluations, number of iterations as well as in computational time in all categories of problems. $\delta = 0.85$ exhibits reduction in computational time in soft and medium dimension while unit-spaced method also showed less computational time in hard and combined dimension.

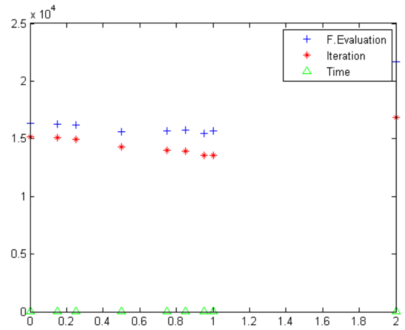
4. Conclusion

Different values of δ are assessed with standard BFGS (single-step method), unit-spaced method and multi-step quasi-Newton methods of different techniques. The experimental results of all methods are obtained at the computational cost of 20 times. The analysis was done on the basis of average results. The problems of different dimensions are analyzed by function evaluations and number of iterations per time. The results revealed that the method at $\delta = 0.5$ outperformed in case of function evaluation and iterations in soft problems. In medium dimension the method at δ value 0.95 exhibited better performance in function evaluation and number of iteration from all other considered delta values. It is evident from experimental results that δ value at 0.85 outperformed the time taken by experiments in the problems of soft and medium dimension and in case of function evaluation in the hard dimension.

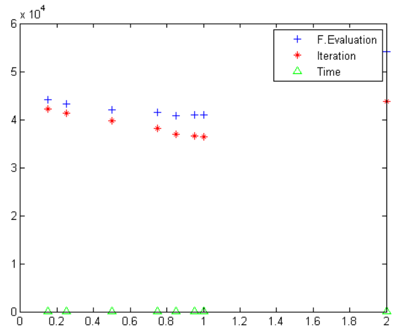
The results also revealed that unit-spaced method is winning from all the reported delta values in computational time and number of iteration in hard and combined dimension. The method $\delta = 2.0$ failed to outperform in all dimension problems in case of function evaluation, iteration and in computational time. The analysis revealed that the δ values 0.85 and 0.95 outperformed from other proposed values of δ in all the problems of different dimension. It can also be easily concluded that all delta values which are less than one are out performed the BFGS method and unit-spaced method in function evaluation, number



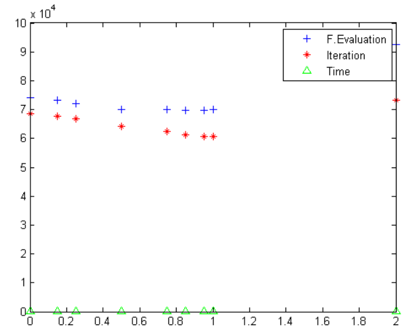
(a)



(b)



(c)



(d)

Figure 1: Results of all dimension problems on different delta value
(a) Soft problems (b) Medium problems
(c) Hard problems (d) Combined problems

of iteration and also in case of time spent in experimentation.

5. Acknowledgments

The author is grateful to her supervisor, Dr. Nudrat Aamir, for her unlimited guidance, encouragement and useful advices she provided throughout this research. The author is also thankful to the reviewers, their valuable suggestions and healthy criticism helped to improve the quality of this work
.

REFERENCES

[1] Yang Ding and Enkeleida Lushi and Qingguo Li, *Investigation of quasi-Newton Methods for Unconstrained Optimization*, Simon Fraser University, Canada 2011.

[2] J. Ford and I. Moghrabi, *Multi-step quasi-Newton Methods for Optimization*, J. Comput. Appl. Math. **50** (1994) 305-323.

[3] J. Ford and S. Tharmlikit, *New Implicit Updates in multi-step quasi-Newton Methods for Unconstrained Optimization*, J. Comput. Appl. Math. **152**, (2003) 133-146.

[4] J. Ford and I. Moghrabi, *Alternating Multi-step quasi-Newton Methods for Unconstrained Optimization*, J. Comput. Appl. Math. **82**, (1997) 105-116.

[5] J. Ford and I. Moghrabi, *Minimum Curvature Multi-step quasi-Newton Methods*, Comput. Math. Appl. **31**, (1996) 179186.

[6] J. Ford, *Implicit Updates in Multi-step quasi-Newton methods*, J. Comput. Appl. Math. **42**, (2001) 1083-1091.

[7] J. Ford and I. Moghrabi, *An Alternative Parametric Choices for Multi-step quasi-Newton methods*, Optim. Methods. Softw. **2**, (1993) 357-370.

[8] K. Hillstrom and J. More and B. Garbow, *Testing Unconstrained Optimization Software*, ACM Trans. Math. Softw. **7**, No.1 (1981) 136140.

[9] F. Modarres and M. AbuHassan and W. Leong, *Multi-steps Symmetric Rank-one Update for Unconstrained Optimization*, World Appl. Sci. J. **7**, No.5 (2009) 610-615.

[10] I. Moghrabi and A. Kassar, *New Parameterizations for Multi-Step Unconstrained Optimization*, J. Korean. Soc. Ind. Appl. Math. **3**, No. 1 (1999) 71-79.

[11] S. Rao, *Engineering Optimization: Theory and Practice*, John Wiley and Sons, Inc.Hoboken, New Jersey, Canada 2009.

[12] P. Toint, *On large Scale Nonlinear Least Squares Calculation*, SIAM J. Sci. Comput. **8**, No.3 (1987) 416-435.

[13] M. Waziri and Z. Majid and H. Aisha, *An Implicit Multi-Step Diagonal Secant-Type Method for Solving Large-Scale Systems of Nonlinear Equations*, Appl. Math. Sci. **6**, No.114 (2012) 5667 - 5676.

Table 2: **Comparison in All Dimension**

Dimension	Method	F.Evaluation	Iteration	Time(sec)	Failure
Soft	BFGS	13112	10693	2.58	0
	Unit-spaced	13332	10561	2.51	0
	δ=0.15	12703	10416	2.29*	0
	δ=0.25	12629	10377	2.59	0
	δ=0.5	12412*	10168*	2.60	0
	δ=0.75	12924	10460	2.45	0
	δ=0.85	13022	10442	2.42	0
	δ=0.95	13318	10509	2.68	0
	δ=2.0	16766	12706	3.13	0
Medium	BFGS	16365	15196	4.67	0
	Unit-spaced	15660	13584	4.58	0
	δ=0.15	16293	15091	4.61	0
	δ=0.25	16183	14969	4.74	0
	δ=0.5	15576	14266	4.60	0
	δ=0.75	15660	13974	4.35	0
	δ=0.85	15781	13909	4.21*	0
	δ=0.95	15482*	13534*	4.30	0
	δ=2.0	21645	16884	5.24	0
Hard	BFGS	44650	42748	44.93	0
	Unit-spaced	40933	36411*	38.03*	0
	δ=0.15	44133	42219	44.23	0
	δ=0.25	43313	41357	43.56	0
	δ=0.5	41991	39787	41.32	0
	δ=0.75	41456	38094	40.40	0
	δ=0.85	40794*	36878	38.22	0
	δ=0.95	40903	36658	38.75	0
	δ=2.0	54205	43727	46.02	0
Combined	BFGS	74127	68637	52.18	0
	Unit-spaced	69925	60556*	44.75*	0
	δ=0.15	73129	67726	51.26	0
	δ=0.25	72125	66703	50.89	0
	δ=0.5	69979	64221	48.53	0
	δ=0.75	70040	62528	47.20	0
	δ=0.85	69597*	61229	44.86	0
	δ=0.95	69703	60701	45.74	0
	δ=2.0	92616	73317	54.39	0

Table 3: **Test Problems and Dimensions** [12]

Function Name	**Dimensions**
Extended Rosenbrock	2, 20, 26, 40, 60, 80, 100, 120
Extended Wood	4, 12, 24, 48, 68, 92, 112, 140
Extended Powell singular	4, 8, 60, 80, 100, 140
Penalty 1	10, 14, 20, 30
Penalty 2	10, 16, 24, 30
Modified Trigonometric Function	16, 32, 64, 95, 128, 150
Broyden Tridiagonal	18, 36, 72, 90, 108, 144, 186
Discrete Boundary Value	20, 38, 60, 90, 120, 136, 188
Discrete Integral Equation	20, 84, 100, 150, 175, 200
Freudenstein and Roth	28, 52, 85, 118, 190
Variably Dimensioned	30, 55, 75, 100, 130, 150
Merged Quadratic	30, 50, 70, 110, 136, 180
Discrete ODE II	33, 44, 66, 88, 110, 176
Discrete ODE I	42, 58, 78, 96, 114, 160
Extended Engvall Function	64, 76, 88, 104, 155, 196

Table 4: **Test problems and dimensions in soft test sets**

	Starting points			
Function Name and dimension	[a]	[b]	[c]	[d]
Rosenbrock (2)	e = 10^{-7}(-1.2, 1.0)	(-120, 100)	(20, -20)	(6.39, -0.221)
Wood (4)	e = 10^{-7}([-3, -1])	([-300, -100])	([-3, 1])	([-300, 100])
Powell singular (4)	e = 10^{-7}(3, -1, 0, 1)	(300, -100, 0, 100)	(2, 2, 3, -1)	(20, 20, 30, -10)
Powell singular (8)	e = 10^{-7}([3, -1, 0, 1])	([300, -100, 0, 100])	([2, 2, 3, -1])	([20, 20, 30, -10])
Penalty 2 (10)	e = 10^{-7}([F])	([0.5])	([-0.1, -0.2, 0.0, 0.1, 0.2])	([-1.0])
Penalty 1 (10)	e = 10^{-7}(1, 2, 3,...., 10)	([5, -5])	([2, 1, 0, -1, -2])	(-10, -20,.., -100)
Wood (12)	e = 10^{-7}([-3, -1])	([-300, -100])	([-30, -10, 0, 10])	([100])
Penalty 1 (14)	e = 10^{-7}(1, 2, 3,.., 14)	([10, 5, 0, -5, -10])	([20, -10])	(-10, -20,....., -140)

Table 5: **Test problems and dimensions in medium set**

		Starting-points			
Function Name and dimension		[a]	[b]	[c]	[d]
Penalty 2 (16)	$e = 10^{-6}$	([0.5])	([-0.2, -0.1, 0.2, 0.1])	([-1.0]) ([F])	
Modified Trigonometric (16)	$e = 10^{-7}$	([-2, -1, 1, 2])	([-2, 1.5,.., -1.5, 2])	([0.1, 1, -0.1, -1])	([2.5, 2, 1.5, 1, 0.5], 2.5)
Broyden tridiagonal (18)	$e = 10^{-7}$	([-1]) ([-100])	([-2, -1, 0, 0, -1, -2])	([-15, -10, -5])	
Penalty 1 (20)	$e = 10^{-7}$	(1, 2, 3,..., 20)	([10, 5, 0, -5, -10])	([20, -10])	(-10, -20,...., -200)
Discrete Boundary Value (20)	$e = 10^{-7}$	([F])	(1, 2, 3,....., 20)	([10, 5, 0, -5, -10])	([25, -25])
Rosenbrock (20)	$e = 10^{-7}$	([-1.2, 1.0])	(1, 2,....,20)	([6.39, -0.221])	(-1, -1, -1, -1, -1, 1,....1)
Discrete integral equation (20)	$e = 10^{-7}$	([F])	([-10, 5, 0, -5, 10])	([5])	([5, -4, 3,.., -3, 4, -5])
Wood (24)	$e = 10^{-7}$	([-3, -1])	([-300])	([10, 0, -10])	([100])
Penalty 2 (24)	$e = 10^{-7}$	([F])	([-0.2, -0.1, 0.1, 0.2])	([-1.0])	([0.5]
Rosenbrock (26)	$e = 10^{-7}$	([-1.2, 1.0])	([F])	([20])	([6.39, -0.221])
Freudenstein and Roth (28)	$e = 10^{-2}$	([2])	([0.5, -2])	([-10])	([1])
Penalty 1 (30)	$e = 10^{-7}$	(1, 2, 3,..., 30)	([-10, -5, 0, 5, 10])	([10, 20, 30,..,100])	(300, 290, 280,..., 10)
Penalty 2 (30)	$e = 10^{-6}$	([0.5])	([-0.2, -0.1, 0.0, 0.1, 0.2])	([F])	([0.5, 0.0, -0.25])
Variably Dimensioned (30)	$e = 10^{-6}$	([F])	([-1, -2,..., -5])	([-10, 5, 0, -5, 10])	([F])
Merged Quadratic (30)	$e = 10^{-3}$	([5.0])	([1, -2, 3, -4, 5])	([-2,1])	([-10, -9, -8,..., -2, -1])
Modified Trigonometric (32)	$e = 10^{-7}$	([-2, -1, 1, 2]	([-2, 1.5,.., -1.5, 2]	([0.1, 1, -0.1, 1]	([2.5, 2,..,0.5], 2.5, 2)
Discrete ODE II (33)	$e = 10^{-7}$	([F])	([1])	([3, -1])	([-10])
Broyden Tridiagonal (36)	$e = 10^{-6}$	([-1])	([-100])	([-2, -1, 0, 0, -1, -2])	([-10, 0, -10])
Discrete boundary value (38)	$e = 10^{-7}$	([F])	([2, -2])	([10, -5])	([F])
Extended Rosenbrock (40)	$e = 10^{-7}$	([-1.2, 1])	([-120, 100])	([1, -2, 3, -4,...., -10])	([20])
Extended Powell Singular (40)	$e = 10^{-7}$	([3, -1, 0, 1])	([5])	([3, -3])	([1, 2,.., 5, -1,...,-5])
Discrete ODE I (42)	$e = 10^{-7}$	([1, -2, 3,.., -6])	([0])	([1, 2])	([-5, -1])
Discrete ODE II (44)	$e = 10^{-7}$	([1])	([3, -1])	([-10])	([-11, 10, -9,..., -1])
Wood (48)	$e = 10^{-7}$	([-3, -1])	([-300, -100])	([1, 0, -1])	([100])
Merged Quadratic (50)	$e = 10^{-3}$	([5.0])	([1, -2, 3, -4. 5])	([-2, 1])	([-10, 9, -8,..., 1])
Freudenstein and Roth (52)	$e = 10^{-2}$	([2])	([0.5, -2])	([-10])	([1])
Variably Dimensioned (55)	$e = 10^{-5}$	([-1, -2, ..-5])	([-10, 5, 0, -5, 10])	([F])	([F])
Discrete ODE I (58)	$e = 10^{-7}$	([0])	([1, 2])	([-5, -1])	([1, -2, .., -6], 1, -2, 3, -4)
Powell singular (60)	$e = 10^{-7}$	([3, -1, 0, 1])	([2, 2, 3, -1])	([-3, 1])	([10])
Discrete boundary value (60)	$e = 10^{-7}$	([F])	([-2, -1, 0, 1, 2])	([10, 0, -10])	([10, -9,..., -1])
Rosenbrock (60)	$e = 10^{-7}$	([-1.2, 1])	([F])	([F])	([6.39, -0.221])
Modified Trigonometric (64)	$e = 10^{-6}$	([-2, -1, 1, 2])	([-2, 1.5,..., -1.5, 2])	([0.1, 1, -0.1, 1])	([2.5, 2,.. 0.5], 2.5,.., 1)

Table 6: **Test problems and dimensions in hard test set**

		Starting-points			
Function Name and dimension		[a]	[b]	[c]	[d]
Extended Engvall Function (64)	e=10^{-5}	([2])	[5, -1])	([-1, 2, -3, 4])*([10])	
Discrete ODE II (66)	e= 10^{-7}	([1])	([3, -1])	([-10])	([F])
Wood (68)	e=10^{-7}	([-3, -1])	([-300, -100])	([-3, 1])	([100])
Merged Quadratic (70)	e= 10^{-7}	([5])	([1, -2, 3, -4, 5])	([-2, 1])	([-10, -9, ..., -1])
Broyden Tridiagonal (72)	e= 10^{-7}	([-1])	([-100])	([-2.0, -1, 0, 0, -1, -2])	([-10, 0, -5])
Variably Dimensioned (75)	e=10^{-5}	([-1, -2, -3])	([-10, 5, 0, -5, 10])	([F])	F
Extended Engvall Function (76)	e= 10^{-5}	([2])	([5, -1])	([-1, 2, -3, 4])	([10])
Discrete ODE I (78)	e=10^{-7}	([0])	([1, 2])	([-5, -1])	([1, -2,..., -6])
Rosenbrock (80)	e= 10^{-7}	([-1.2, -1.0])	([F])	([F])	([F])
Extended Powell Singular (80)	e= 10^{-7}	([3, -1, 0, 1])	([2, 2, 3, -1])	([10])	
Discrete integral equation (84)	e= 10^{-4}	([F])	([-10, 5,.., -10],-10,.., -5)	([5])	([5, -4,...,-5], 5,.., 2)
Freudenstein and Roth (85)	e= 10^{-3}	([2])	([0.5, -2])	([-10])	([1])
Discrete ODE II (88)	e=10^{-7}	([1])	([3, -1])	([-10])	([F])
Extended Engvall Function (88)	e= 10^{-5}	([2])	([5, -1])	([-1, 2, -3, 4])	([10])
Discrete boundary value (90)	e=10^{-7}	([10, 5, 0, -5, -10])	([F])	([1, 2,...., 15])	([25, -25])
Broyden Tridiagonal (90)	e=10^{-7}	([-1])	([-100])	([-2, -1, 0, -1, -2])	([-10, 0, -5])
Wood (92)	e= 10^{-7}	([-3, -1])	([-300, -100])	([30, -10, 0, 10])	([100])
Modified Trigonometric (95)	e= 10^{-5}	([-2, -1, 1, 2],-2, -1, 1)	([-2, 1.5,.., 2])	([0.1, 1.0, -0.1, 1.0])	([2.5, 2.0, 1.5, 1.0, 0.5])
Discrete ODE I (96)	e=10^{-7}	([0])	([1, 2])	([-5, -1])	([1, -2,..., -6])
Rosenbrock (100)	e=10^{-7}	([-1.2, -1])	([F])	([F])*([F])	
Extended Powell Singular (100)	e = 10^{-7}	([3, -1, 0, 1])	([5])	([3, -3])	([1, 2,.., 5, -1,..., -5])]]
Variably Dimensioned (100)	e = 10^{4}	([-1, -2,..., -5])	([-10, 5, 0, -5, 10])	([F])	([F])
Discrete integral equation (100)	e = 10^{4}	([F])	([10, 5, 0, -5, 10])	([5.0])	([5, -4, 3, -2, 1, -1, 2, -3, 4, -5])
Extended Engvall Function (104)	e = 10^{5}	([2])	([5, -1])	([-1, 2, -3, 4])	([10])
Broyden Tridiagonal (108)	e = 10^{6}	([-1])	([-100])	([-2, -1, 0, 0, -1, -2])	([-5, 0, -5])
Discrete ODE II (110)	e = 10^{7}	([1])	([3, -1])	([10])	([F])

Table 7: **Test problems and dimensions in hard test set**

		Starting points			
Function Name and dimension		[a]	[b]	[c]	[d]
Merged Quadratic (110)	e = 10^{3}	([5])	([1, -2, 3, -4, 5])	([-2, 1])	([-10, -9,......, -1])
Wood (112)	e = 10^{7}	([-3, -1])	([-300, -100])	([30, -10, 0, 10])	([100])
Discrete ODE I (114)	e = 10^{7}	([0])	([1, 2])	([-5, -1])	([1, -2,..., -6])
Freudenstein and Roth (118)	e = 10^{2}	([2])	([0.5, -2])	([-10])	([1])
Discrete boundary value (120)	e= 10^{-7}	([10, 5, 0, -5, -10])	([F])	([1, 2, ..., 20])	([25, -25])
Rosenbrock (120)	e=10^{-7}	([-1.2, -1])	([20])	[F]	([6.39, -0.221])
Modified Trigonometric function (128)	e= 10^{-6}	([-2, -1, 1, 2], -2, -1, 1)	([-2, 1.5,.., -1.5, 2])	([0.1, 1, -0.1, 1])	([2.5, 2, 1.5, 1, 0.5], 2.5, 2, 1.5)
Variably Dimensioned (130)	e= 10^{-4}	([-1, -1, -1, -1, -1])	([-10, 5, 0, -5, 10])	([F])	([F])
Discrete ODE II (132)	e=10^{-7}	([1])	([3, -1])	([10])	([F])
Discrete boundary value (136)	e=10^{-7}	([10, 5, 0, -5, -10])	([F])	([1, 2, ..., 17])	([25, -25])
Merged Quadratic (136)	e=10^{-3}	([5])	([1, -2, 3, -4, 5], 1)	([-2, 1])	([-10, -9,......, -1], -10,..., -5)
Wood (140)	e=10^{-7}	([-3, -1])	([-300, -100])	([30, -10, 0, 10])	([100])
Extended Powell Singular (140)	e= 10^{-7}	([3, -1, 0, 1])	([2, 2, 3, -1])	([10])	([3, -1])
Broyden Tridiagonal (144)	e=10^{-7}	([-1])	([-100])	([-2, -1, 0, 0, -1, -2])	([-10, 0, -5])
Modified Trigonometric function (150)	e= 10^{-5}	([-2, -1, 1, 2], -2, -1)	([-2, 1.5, ..., 2], -2,..., -0.5, 1)	([0.1, 1, -0.1, 1], 0.1, 1)	([2.5, 2.0, 1.5, 1.0, 0.5])
Variably Dimensioned (150)	e=10^{-4}	([0.5, 0.5, 0.5])	([-10, 5, 0, -5, 10])	([F])	([F])
Discrete integral equation (150)	e=10^{-4}	([F])	([10, 5, 0, -5, 10])	([5])	([5, -4, 3,.., 2, -3, 4, -5])
Extended Engvall Function (155)	e=10^{-5}	([2])	([5, -1], 5)	([-1, 2, -3, 4], -1, 2, -3)	([10])
Discrete ODE I (160)	e=10^{-7}	([0])	([1, 2])	([-5, -1])	([1, -2,..., -6], 1, -2, 3, -4)
Discrete integral equation (175)	e= 10^{-4}	([F])	([10, 5, 0, -5, 10])	([5])	([5, -4, 3, .., 2, -3, 4, -5], 5, -4, 3, -2, 1)
Discrete ODE II (176)	e=10^{-7}	([1])	([3, -1])	([-10])	([F])
Merged Quadratic (180)	e=10^{-2}	([5])	([1, -2, 3, -4, 5])	([-2, 1])	([-10, -9,......, -1])
Broyden Tridiagonal (186)	e=10^{-6}	([-1])	([-100])	([-2, -1, 0, 0, -1, -2])	([-5, 0, -5])
Discrete boundary value (188)	e= 10^{-7}	([10,.., -5, -10], 10, 5, 0)	([F])	([1, 2, ..., 47])	([25, -25])
Freudenstein and Roth (190)	e= 10^{-2}	([2])	([0.5, -2])	([-10])	([1])
Extended Engvall Function (196)	e= 10^{-5}	([2])	([5, -1])	([-1, 2, -3, 4])	([10])
Discrete integral equation (200)	e= 10^{-4}	([F])	([10, 5, 0, -5, 10])	([5])	([5, -4, 3, -2, 1, -1, 2, -3, 4, -5])