NUMERICAL TEST RESULTS FOR A PROTOTYPE LOCAL MONOTONICITY STRATEGY

by

S. Azarm and P. Papalambros

Report No. UM-MEAM-84-27

June 1984

NUMERICAL TEST RESULTS FOR A PROTOTYPE LOCAL MONOTONICITY STRATEGY

Shapour Azarm
Lecturer

and

Panos Papalambros
Assistant Professor

Dept. of Mechanical Engineering
and Applied Mechanics
The University of Michigan
Ann Arbor

ABSTRACT

A prototype local monotonicity-based strategy codified in the program ACCME (Automated Constraint Criticality by Monotonicity Evaluations) was tested on a subset of the problem bank compiled by Hock and Schittkowski for nonlinear programming. Under similar conditions, two other codes were tested: generalized reduced gradient (GRG2 by Lasdon et al) and sequential quadratic programming (VMCON by Argonne National Laboratory). The results of this study are presented with some discussion. Although ACCME is only a demonstration code lacking numerical refinements, its performance was judged fairly competitive to the other two.

INTRODUCTION

The present work addresses the manner of solving numerically the general nonlinear programming (NLP) problem

$$\text{minimize } f(\underline{x})$$

$$\text{subject to } g_j(\underline{x}) = 0,\ j = 1, \ldots, m \qquad (1)$$

$$g_j(\underline{x}) \leq 0,\ j = (m+1), \ldots, p$$

where f and g_j are scalar objective and constraint functions of the n-vector x. This formulation arises in design optimization models where typically the number of inequality constraints is large. Moreover, many inequality constraints are often active (critical) at the optimum. This motivates development of strategies for identifying active constraints and thus solving the problem with reduced number of degrees of freedom. These active set strategies are sometimes considered particularly attractive for design optimization problems, because for example they may offer better insight for the optimal design or may handle better large numbers of inactive and/or redundant constraints [1].

The method of monotonicity analysis was introduced for identifying active constraints in a global sense, a priori (see e.g. [2] and references therein). The strength but also the difficulty in this method is the analytical, often complicated, algebraic manipulations, which allow global conclusions to be reached. A first attempt to implement the method on the computer was done by Zhou and Mayne [3] in an interactive mode. Fully automated procedures using monotonicity arguments and reduced

gradients were reported by Zhou and Mayne [4] and Azarm and Papalambros [5,6] and extensively studied by Zhou [7] and Azarm [8]. The focus in the former's work was to use monotonicity for improving several existing NLP codes, while in the latter's work was in extending the global procedure, as for example in coupling global and local rules in the active set strategy [5]. The results presented here are based on the work in [6,8] using the program ACCME (Automated Constraint Criticality by Monotonicty Evaluations).

In the following sections we first summarize previous comparative studies and describe the classification of the 40 test problems selected from the compilation by Hock and Schittkowski (abbr. H & S) [18]. The numerical results are presented for ACCME, GRG2 [13] and VMCON[16], followed by comparison and discussion.

PREVIOUS MAJOR COMPARATIVE STUDIES

In the last two decades, a considerable number of papers has been written for comparison of nonlinear constrained optimization programs. Here we summarize the most comprehensive studies.

A paper by Colville [9] is the first major comparative study done in 1968. He used a ranking scheme which was based upon the computer times required for a solution, and the number of test problems solved by a given program from a set of eight test problems. In that study it was pointed out that the number of objective and constraint function evaluations was not a good indication of the performance of a given program. The conclusion of the study [10] was that the large-step gradient methods, such as generalized reduced gradient, performed in a measurable way

better than the other methods tested.

Comprehensive comparative results, for problems in engineering design, were presented by Sandgren [11] in 1977. A variety of constrained optimization techniques, including linear approximation methods, interior and exterior penalty methods, were tested on a set of thirty problems. The ranking criterion was the ability to solve problems within a reasonable amount of computational time. The general conclusion of the study was that generalized reduced gradient methods solved a greater percentage of the test problems and were much faster than the other types of methods tested. Finally, the most recent comprehensive comparative evaluation of nonlinear programming codes was made by Hock and Schittkowski [12] in 1983. They compared the performance of 27 nonlinear constrained optimization programs on a set of 115 test problems mostly from actual realistic applications. The final conclusion of the study was that the sequential quadratic programming methods performed better, followed by the generalized reduced gradient, augmented Lagrangian, and penalty methods. A study of optimization methods for structural design performed by Belegundu [1] is also of interest, since it points out the utility of active set strategies if finite element analysis models are used.

The results of these major comparative studies suggest that algorithms which are based upon the generalized reduced gradient (GRG) and sequential quadratic programming (SQP) techniques are generally better than the other techniques tested. Therefore it is desirable to have some measure of the performance of ACCME in

comparison with nonlinear programming codes which are based on GRG and SQP techniques. GRG2 is one such code, which was developed by Lasdon and Warren [13,14], and is based on the GRG algorithm as suggested by Abadie and Carpentier [15]. VMCON is the other code, which was developed at the Argonne National Laboratory [16] and is based on a SQP algorithm of a type suggested by Powell [17]. Both GRG2 and VMCON have been extensively tested and are considered generally efficient and reliable.

TEST PROBLEMS

All three algorithms, ACCME, VMCON, and GRG2 were tested on a set of 40 test problems selected from Hock and Schittkowski (abbr. H&S) primarily with inequality constraints and representing increasing level of complexity [18]. Since the test problems have different structure, a classification number is defined. Following the practice of H&S with a slightly different notation, we define the following sequence of letters:

OCS-N (2)

The following list gives all possible abbreviations which could replace the letters O,C,S, and N for the tested problems:

O: Information about the Objective function

- O=L: Linear objective function
- O=Q: Quadratic objective function
- O=P: Generalized Polynomial objective function

C: Information about the Constraint functions

- C=B: Upper and lower Bounds on the variables only

C=L: Linear constraint functions

C=Q: Quadratic constraint functions

C=P: Generalized Polynomial constraint functions

C=G: General constraint functions

S: Information about the Starting point

S=F: Feasible starting point

S=I: Infeasible starting point

N: Problem number in H&S

As an example, consider the following problem:

$$\text{minimize } f(x) = x_1 x_2 x_3 \tag{3}$$

subject to

$$x_1^2 + 2x_2^2 + 4x_3^2 - 48 \leq 0$$

starting point: $(1,1,1)^T$

The objective function is Polynomial, the constraint is Quadratic, the starting point is Feasible, and it is problem No. 29 in H&S, therefore we classify this problem by:

$$PQF-29 \tag{4}$$

We now summarize the abbreviations used in Table 1 to describe the test problems:

TP : Test problem number

OCS-N : Classification of the test problem

NV : Number of the variables

NEQ : Number of equality constraints

NC : Total number of constraints, i.e., equalities and inequalities

NACTC : Total number of active inequality and equality constraints

$f(x^*)$: Objective function value at the optimal solution

The test problems considered in this study have 2 to 15 variables with 1 to 31 constraints, from which 7 problems have 1 to 3 equality constraints. In 22 of the tested problems, there are as many variables as there are active constraints (satisfied as equalities at the optimum). The problems were selected on the basis of having primarily inequality constraints. This is obviously the only case of interest in terms of comparison with ACCME.

NUMERICAL RESULTS

The codes were tested in double precision FORTRAN, at the Michigan Terminal System's Amdahl 5860 Computer. The tests were done under similar workload conditions of computer, since measured execution time may vary depending on the load on the machine. All intermediate printouts for the codes were suppressed. Thus the output included only initial data and final results. Tolerances used in various parts of the three codes were uniformly taken to be (10^{-4}).

Numerical partial derivatives required in ACCME and VMCON for the objective and constraint functions, were computed by a combination of forward and central differencing techniques. In the case of GRG2, the central differencing option of that code was used, which puts the method in a relative disadvantage.

ACCME Test Results

Detailed numerical results of the ACCME testing are listed in Table 2. The abbreviations used in the table are described here:

TP : Test problem number

NLSOLV : Index number for equations solver

NLSOLV=1 [19,20]

NLSOLV=2 Newton method [21]

NLSOLV=3 Least Square method [22,23]

NF : Number of objective function evaluations

NG : Number of constraint function evaluations

NDF : Number of gradient evaluations of objective function

NDG : Number of gradient evaluations of constraint function

ET : Execution time in seconds, compilation and loading time is not included

$f(\underset{\sim}{x}_*)$: Objective function value at the optimal solution obtained by ACCME

VMCON : An "X" in this column indicates the ACCME reached a solution using VMCON program

A letter "F" in the table indicates the failure of the algorithm for a specified problem, or execution time greater than ten seconds.

Certain observations concerning the contents of Table 2 are in order:

Execution time is generally one of the factors considered most important in the performance evaluation of a code in a successful run. From Table 2, it is clear that execution time does not always follow the same trends as the number of function evaluations. For example, in problem No. 4, where the number of constraint function evaluations were 525, 385, and 525, the execution times were 0.059, 0.062, and 0.103 seconds respectively for the three different nonlinear equation solvers. This becomes

more obvious in problem No. 7, where the number of function evaluations is the same for different equation solvers while execution time is different. The general observation is that in 26 of the 40 test problems, the execution times did not follow the trends of number of function evaluations. This does support the Collvile Study [7], in the sense that the number of function evaluations is not an accurate measure of algorithm performance.

With regard to the type of equations solver, i.e., 1 (NLSOLV=1), 2 (NLSOLV=2), and 3 (NLSOLV=3) in Table 2, the indication is that using 1 or 2 resulted to reduced execution time compared to method 3 for most of the tested problems. Comparing method 1 with 2, it is readily seen that for larger problems method 2 resulted to reduced execution time for convergence of ACCME compared to 1. Also, in terms of number of function evaluations the second method performed better.

VMCON and GRG2 Test Results

Numerical results of tested problems for VMCON and GRG2 are listed in Table 3. The abbreviations used are the same as in Table 2. In terms of number of objective or constraint function evaluations, VMCON performed much better than GRG2. In fact only in 4 of the test problems, GRG2 had slightly smaller number of function evaluations than VMCON. In terms of gradient evaluations, GRG2 and VMCON performed about the same. VMCON also had smaller execution time than GRG2 for most of the test problems.

Note that in VMCON a sequence of quadratic programming subproblems is solved, where the objective function is an

approximation of the Lagrangian and the constraints are linear approximations to the original constraints. These subproblems generally estimate a search direction which is used in a subsequent one-dimensional minimization for a combination of objective function and constraint infeasibilities. If the quadratic programming subproblems approximate the original problem closely and nonlinearities in the problem are such that obtaining a feasible point is not a difficult task, then VMCON would generally require only a few function evaluations to find the solution. On the other hand, GRG2 uses the active constraints to express the basic variables in terms of the nonbasic ones, thus the original problem is changed to a reduced one for which a search direction is found. An important part of the computational effort in GRG2 is devoted to an attempt to return to the constraint surface at each step.

DISCUSSION

As stated previously, execution time for a given tolerance in a successful run, is one of the main and most common criteria for evaluating the efficiency of a constrained optimization code. Execution times for all the tests are reproduced in Table 4. As seen from the table, ACCME performed competitively compared to VMCON and GRG2. In terms of number of function evaluations, ACCME performed significantly less well compared to the other two codes, Tables 2, 3. In terms of number of unsuccessful runs, ACCME performed almost as well as GRG2 and VMCON, a reason for this being the existence of three different options of nonlinear equation solver available to the user in ACCME.

For some of the test problems, ACCME performed poorly compared to VMCON and GRG2. Some reasons explaining this behavior are described below:

(1) Nonlinear Equation Solver

Iteration procedures in the equation solvers may not converge or may produce some oscillation about the solution if the starting point is not properly chosen. This may happen especially if the starting point is far from the solution. In ACCME, it is quite possible to move from one corner of feasible region to another, thus demanding a large step from equation solver. If the equation solver does not converge, it may cause ACCME to go through a few more steps in order to compensate for that.

(2) One-Dimensional Search

In ACCME the one-dimensional search is done by the Golden Section method. Although the method is quite reliable, the rate of convergence is only linear and convergence may be slow. Also, the fact that this one-dimensional search is done in the space of active inequality and equality constraints, may substantially affect the number of function evaluations.

(3) Descent Method

In ACCME, if no new constraint is found active in a given iteration, the program switches to a descent routine, which is basically a simple gradient method in the space of the active inequality and equality constraints. These constraints may be very nonlinear, therefore causing the descent method to take a large number of very small steps towards the optimum.

(4) Feasible Point Search

Large step moves in ACCME may result to an infeasible point. If the number of consecutive iterations resulting to infeasible points is greater than 2, a default value, then ACCME attempts to find a feasible point. This is carried out in the following order:

(1) Deactivation of all inequalities, if any.

(2) Minimization of the sum of squares of constraint violations using the Davidon-Fletcher-Powell method [24] Fletcher-Powell method [24].

(5) Scaling

If the variables and constraints of a problem which is supplied to ACCME are not properly scaled, this may significantly alter the convergence of the program. A common difficulty is in finding feasible solutions, which is due to large differences in magnitude between values of constraint functions. A "well scaled" set of constraint functions should be balanced with respect to each other, i.e., all the constraints should have "equal weight" in the solution process (see e.g. [25]).

CONCLUSION

Interpretation of test results for NLP codes typically is a source of arguments among researchers not only because of test conditions, but also because of often substantially different motivation in the use of optimization techniques. One tends to classify as good the algorithm having less difficulty in solving problems of one's interest. The work of Hock and Schittkowski is a substantial contribution to attempts at establishing some measure of objectiveness.

Evaluating ACCME as a general NLP software is rather improper. In fact some of the difficulties discussed in the previous section are avoided by using the approach of Zhou and Mayne. There however, only one monotonicity rule is used, instead of two as in ACCME. The second rule appears to be a disadvantage in a purely local strategy. Yet when global information is introduced (which generaly invalidates convergence proofs), the second rule becomes important [5,6]. In spite of this and the fact that ACCME lacks numerical efficiency refinements, its overall performance compares sufficiently favorably to encourage further investigation in the utility of monotonicity-based active set strategies.

ACKNOWLEDGEMENT

This research was supported by NSF Grant No. MEA 83-0158 at the University of Michigan. This support is gratefully acknowledged.

REFERENCES

1. Belegundu, A.D., "A Study of Mathematical Programming Methods for Structural Optimization," Ph.D. Dissertation. Division of Material Engineering, University of Iowa, 1982.

2. Papalambros, P. and Li, H.L., "Notes on the Operational Utility of Monotonicity in Optimization", Trans. ASME, J. of Mech. Transm. and Automation in Design, Vol. 105, No. 2, June 1982, pp. 174-180.

3. Zhou, J., and Mayne, R.W., "Interactive Computing in the Application of Monotonicity Analysis to Design Optimization", Trans. ASME, J. of Mech. Transm. and Automation in Design, Vol. 105, No. 2, June 1982, pp. 181-186.

4. Zhou, J., and Mayne, R.W., "Monotonicity Analysis and the Reduced Gradient Method in Constrained Optimization", Trans. ASME, J. of Mech., Transm. and Automation in Design, Vol. 106, No. 1, March 1984, pp. 90-94.

5. Azarm, S., and Papalambros, P., "A Case for A Knowledge-Based Active Set Strategy", Trans. ASME, J. of Mech., Transm. and Automation in Design, Vol. 106, No. 1, March 1984, pp. 77-81.

6. Azarm, S., and Papalambros, P., "An Automated Procedure for Local Monotonicity Analysis", Trans. ASME, J. of Mech. Transm. and Automation in Design, Vol. 106, No. 1, March 1984, pp. 82-86.

7. Zhou, J., A Class of Monotonicity Analysis Based Algorithms for Nonlinear Constrained Optimization, Ph.D. Dissertation, Mech. and Aero. Engin. Dept., SUNY at Buffalo, 1984.

8. Azarm, S., Local Monotonicity in Optimal Design, Ph.D. Dissertation, Dept. of Mech. Engin. and Appl. Mechanics, The University of Michigan, Ann Arbor, June 1984.

9. Colville, A.R., "A Comparative Study of Nonlinear Programming Codes," IBM New York Scientific Report No. 320-2949, June 1968.

10. Colville, A.R., "A Comparative Study of NLP Codes," Proceedings of Princeton Symposium on Math. Programming, H.W. Kuhn (eds.), Princeton, N.J., 1970, pp. 487-501.

11. Sandgren, E., "The Utility of Nonlinear Programming Algorithms," Ph.D. dissertation, Purdue University, Dec. 1977.

12. Hock, W., Schittkowski, K., "A Comparative Performance Evaluation of 27 NLP Codes," Computing, 30, 1983, pp. 335-358.

13. Lasdon, L.S., Waren, A.D., Jain, A., Ratner, M., "Design and Testing of a Generalized Reduced Gradient Code for Nonlinear Programming," ACM Transactions on Math. Programming, Vol. 4, No. 1, 1978, pp. 34-50.

14. Lasdon, L.S., Waren, A.D., Ratnner, M., GRG2 User's Guide, September, 1980.

15. Abadie, J., and Carpentier, J. "Generalization of the Wolfe Reduced Gradient Method to the case of Nonlinear constraints," in Optimization, R. Fletcher (Ed.), 1969.

16. Crane, R.L., Hillstorm, K.E., Minkoff, M., "Solution of the General Nonlinear Programming Problems with Subroutine VMCON," Argonne National Lab., July, 1980.

17. Powell, M.J.D., "A Fast Algorithm for Nonlinearly Constrained Optimization Calculations," Proceedings of the 1977 Dundee Conference on Numerical Anlysis, Lecture Notes in Mathematics, Vol. 630, Springer-Verlog, Berlin, 1978, pp. 144-157.

18. Hock, W., and Schittkowski, K., Test Examples for Nonlinear Programming Codes, Springer-Verlog, New York, 1980.

19. "Numerical Analysis and Application Software Abstracts," Computing Center Memo 407, The University of Michigan, Computing Center, September, 1979.

20. Brown, K.M., and Conte, S.D., "The Solutions of Simultaneous Nonlinear Equations," Proceedings of the 22nd National Conference of the ACM, Washington, D.C., 1967, pp. 111-114.

21. Gerald, C.F., Applied Numerical Analysis, Addison-Wesley, 1978.

22. More, J.J., Garbow, B.S., Hillstorm, K.E., User Guide for Minpack-1, Applied Math. Division, Argonne National Lab., August, 1980.

23. Powel, M.J.D., "A Hybrid Method for Nonlinear Equations," in Numerical Methods for nonlinear Algebraic Equations, P. Rabinowitz (ed.), Gordon and Breach, 1970.

24. Fletcher, R. and Powell, M., "A Rapidly Convergent Descent Method for Minimization," Computer Journal, 1963, pp. 163-168.

25. Gill, E.G., Murray, W., Wright, M.H., Practical Optimization, Academic Press, 1982.

LIST OF TABLES

Table 1

List of Test Problems

TP	OCS-N	NV	NEQ	NC	NACTC	$f(x_*)$
1	QBF-3	2	0	1	1	0
2	PBF-4	2	0	2	2	2.666
3	LQI-10	2	0	1	1	-1
4	QQI-11	2	0	1	1	-8.498
5	QQF-12	2	0	1	1	-30
6	QPI-13	2	0	3	2	1
7	QQI-14	2	1	2	2	1.393
8	PQI-15	2	0	3	2	306.500
9	QQI-18	2	0	4	1	5
10	PQI-20	2	0	5	2	38.199
11	QQI-22	2	0	2	2	1
12	QQI-23	2	0	9	2	2
13	PLF-24	2	0	5	2	-1
14	PQF-29	3	0	1	1	-22.627
15	QQF-30	3	0	7	2	1
16	QQF-31	3	0	7	1	6
17	QPF-32	3	1	5	3	1
18	PQF-33	3	0	6	3	-4.586
19	LGF-34	3	0	8	3	-0.834
20	PLF-36	3	0	7	3	-3300
21	QQF-43	4	0	3	2	-44
22	QLF-44	4	0	10	4	-15
23	PBI-45	5	0	10	5	1

Table 1 (Cont.)

TP	OCS-N	NV	NEQ	NC	NACTC	$f(x_*)$
24	LGF-66	3	0	8	2	0.518
25	PPI-71	4	1	10	3	17.014
26	LGI-73	4	1	7	4	29.894
27	PGI-74	4	3	13	3	5126.498
28	PGI-75	4	3	13	4	5174.413
29	QLF-76	4	0	7	7	-4.682
30	QQI-83	5	0	16	5	-30665.539
31	QQF-84	5	0	16	5	-5280335.133
32	PLF-86	5	0	15	4	-32.349
33	LQI-95	6	0	16	6	0.0156
34	LQI-96	6	0	16	6	0.0156
35	LQI-97	6	0	16	6	3.136
36	LQI-98	6	0	16	6	3.136
37	LQI-106	8	0	22	6	7049.331
38	QQF-113	10	0	8	6	24.306
39	QGI-114	10	3	31	9	-1768.807
40	PQF-117	15	0	20	11	32.349

Table 2

ACCME Test Results

TP	NLSOLV	NF	NG	NDF or NDG	ET	$f(x_*)$ ACCME	VMCON
1	1	128	266	27	0.037	0	
	2	128	153	27	0.034	0	
	3	128	200	27	0.052	0	
2	1	19	40	6	0.033	-2.666	
	2	19	26	6	0.033	-2.666	
	3	19	31	6	0.044	-2.666	
3	1	120	470	26	0.074	-1	X
	2	102	440	26	0.085	-1	X
	3	200	774	41	0.147	-1	
4	1	191	525	51	0.059	8.498	
	2	191	385	51	0.062	8.498	
	3	191	525	51	0.103	8.498	
5	1	83	283	24	0.038	-30	
	2	83	239	24	0.037	-30	
	3	83	429	24	0.113	-30	
6	1	792	2037	220	0.185	0.999	X
	2	157	205	8	0.048	-1.082	
	3	157	205	8	0.050	1.705	
7	1	4053	4053	2	0.198	1.393	
	2	4053	4053	2	0.190	1.393	
	3	4053	4053	2	0.210	1.393	
8	1	19	40	6	0.035	306.5	

Table 2 (Cont.)

TP	NLSOLV	NF	NG	NDF or NDG	ET	$f(x_*)$ ACCME	VMCON
	2	19	26	6	0.043	306.5	
	3	19	31	6	0.045	306.5	
9	1	120	2094	16	0.132	5	
	2	F	F	F	F	F	
	3	67	872	12	0.366	5	
10	1	25	84	6	0.040	38.198	
	2	85	123	14	0.057	40.198	
	3	25	105	6	0.063	38.198	
11	1	43	75	9	0.035	1	
	2	43	56	9	0.035	1	
	3	43	78	9	0.045	1	
12	1	19	46	6	0.045	2	
	2	19	29	6	0.059	2	
	3	19	49	6	0.066	2	
13	1	19	40	6	0.055	-1	
	2	19	26	6	0.038	-1	
	3	19	38	6	0.060	-1	
14	1	268	749	67	0.096	-22.627	X
	2	453	963	113	0.132	-22.627	X
	3	212	702	53	0.154	-22.627	X
15	1	77	422	10	0.073	1	
	2	98	178	10	0.050	1	
	3	98	374	10	0.091	1	

Table 2 (Cont.)

TP	NLSOLV	NF	NG	NDF or NDG	ET	$f(x_*)$ ACCME	VMCON
16	1	183	1016	12	0.111	6	
	2	183	419	12	0.117	6	
	3	183	703	12	0.160	6	
17	1	47	127	10	0.061	1	
	2	47	67	10	0.044	1	
	3	47	123	10	0.076	1	
18	1	26	54	6	0.044	-4	
	2	26	34	6	0.040	-4	
	3	26	47	6	0.063	-4	
19	1	34	134	8	0.054	-0.834	
	2	34	71	8	0.049	-0.834	
	3	34	66	8	0.054	0	
20	1	34	82	8	0.044	-3300	
	2	34	46	8	0.043	-3300	
	3	34	65	8	0.065	-3300	
21	1	F	F	F	F	F	
	2	1012	2023	182	0.217	-44	
	3	370	5254	44	2.002	-44	X
22	1	53	143	10	0.073	-15	
	2	53	64	10	0.053	-15	
	3	53	75	10	0.061	-15	
23	1	76	226	12	0.070	1	
	2	76	98	12	0.077	1	

Table 2 (Cont.)

TP	NLSOLV	NF	NG	NDF or NDG	ET	$f(x_*)$ ACCME	VMCON
	3	76	120	12	0.074	1	
24	1	57	594	8	0.082	-0.518	
	2	136	656	12	0.120	-0.518	
	3	34	66	8	0.055	0.518	
25	1	121	1675	12	0.156	17.014	
	2	123	557	12	0.091	17.014	
	3	43	106	8	0.082	19.348	
26	1	3259	3423	16	0.350	29.689	
	2	1079	1109	12	0.143	29.687	
	3	3254	3357	16	0.377	29.689	
27	1	7548	9854	8	1.248	5126.497	
	2	128	829	9	0.196	5126.497	
	3	20893	20950	20	2.657	5126.497	X
28	1	17978	19016	8	2.296	5174.411	
	2	11411	11659	6	1.432	5174.411	
	3	27963	28020	19	3.467	5174.411	X
29	1	187	2034	14	0.189	-4.682	
	2	187	591	14	0.110	-4.682	
	3	187	1867	14	0.565	-4.682	
30	1	233	517	20	0.116	-30665.5	
	2	233	284	20	0.097	-30665.5	
	3	233	377	20	0.163	-30665.5	
31	1	76	260	12	0.090	-5280335.0	

Table 2 (Cont.)

TP	NLSOLV	NF	NG	NDF or NDG	ET	$f(x_*)$ ACCME	VMCON
	2	76	112	12	0.081	-5280335.0	
	3	76	155	12	0.126	-5280335.0	
32	1	1265	16227	98	3.324	-28.452	
	2	287	974	22	0.302	-32.349	
	3	288	2231	22	0.128	-32.349	
33	1	85	321	12	0.162	0.0156	
	2	85	99	12	0.138	0.0156	
	3	85	255	12	0.242	0.0156	
34	1	85	321	12	0.163	0.0156	
	2	85	99	12	0.140	0.0156	
	3	85	255	12	0.228	0.0156	
35	1	244	496	12	0.134	4.07	
	2	244	284	18	0.112	4.07	
	3	244	347	18	0.154	4.07	
36	1	244	496	12	0.135	4.07	
	2	244	284	18	0.112	4.07	
	3	244	347	18	0.153	4.07	
37	1	6898	6972	38	1.592	7049.25	X
	2	6277	6298	35	1.361	7049.25	X
	3	15223	15571	38	2.423	7049.25	X
38	1	508	9754	34	1.369	21.989	X
	2	447	1535	31	0.483	21.989	X
	3	938	3307	55	2.169	21.989	X

Table 2 (Cont.)

TP	NLSOLV	NF	NG	NDF or NDG	ET	$f(x_*)$ ACCME	VMCON
39	1	929	15083	76	2.794	-1896.9	
	2	359	1468	26	0.634	-1896.9	
	3	F	F	F	F	F	
40	1	F	F	F	F	F	
	2	2153	17488	95	8.622	32.348	X
	3	481	621	30	1.432	32.348	X

Table 3

VMCON and GRG2 Test results

TP	NF or NG		NDF or NDG		ET		$f(x_*)$	
	VMCON	GRG2	VMCON	GRG2	VMCON	GRG2	VMCON	GRG2
1	10	27	3	3	0.059	0.060	0.0009	0
2	7	20	2	3	0.027	0.060	2.666	2.666
3	37	107	12	11	0.049	0.066	-1	-1
4	64	132	21	11	0.046	0.065	-8.499	-8.498
5	36	89	11	6	0.039	0.062	-30	-30
6	79	67	26	6	0.069	0.068	0.999	0.950
7	16	59	5	6	0.041	0.066	1.393	1.393
8	16	35	5	4	0.033	0.065	306.5	306.3
9	25	277	8	16	0.049	0.086	4.999	4.999
10	58	57	19	5	0.072	0.068	38.198	40.19
11	25	42	8	4	0.086	0.06	0.999	0.999
12	19	65	6	5	0.046	0.079	2	1.999
13	16	24	5	3	0.049	0.067	-1	-1

Table 3 (Cont.)

TP	NF or NG		NDF or NDG		ET		$f(x_*)$	
	VMCON	GRG2	VMCON	GRG2	VMCON	GRG2	VMCON	GRG2
14	45	147	11	11	0.051	0.070	-22.627	-22.627
15	37	20	9	2	0.093	0.073	1	1
16	37	137	9	10	0.086	0.086	5.999	5.999
17	13	65	3	4	0.039	0.076	1	1
18	17	25	4	2	0.044	0.071	-3.999	-4
19	29	166	7	9	0.065	0.089	-0.834	-0.834
20	9	F	2	F	0.048	F	-3300	F
21	56	210	11	13	0.069	0.099	-44	-44
22	31	42	6	3	0.086	0.094	-15	-15
23	45	88	8	6	0.095	0.093	1	1
24	25	110	6	9	0.068	0.099	0.518	0.518
25	26	254	5	13	0.067	0.113	17.014	17.014
26	21	64	4	5	0.054	0.107	29.688	29.687
27	56	303	11	12	0.112	0.166	5126.497	5126.4

Table 3 (Cont.)

TP	NF or NG		NDF or NDG		ET		$f(x_*)$	
	VMCON	GRG2	VMCON	GRG2	VMCON	GRG2	VMCON	GRG2
28	51	214	10	7	0.117	0.146	5174.4	5174.41
29	26	75	5	6	0.075	0.097	-4.682	-4.682
30	25	125	4	6	0.079	0.157	-30665.5	-30665.5
31	19	97	3	6	0.087	0.12	-5280335	-5280330
32	37	136	6	9	0.101	0.144	-32.35	-32.35
33	15	41	2	3	0.079	0.132	0.0156	0.0156
34	15	41	2	3	0.080	0.120	0.0156	0.0156
35	50	127	7	7	0.151	0.135	3.135	3.135
36	50	127	7	7	0.154	0.136	3.135	3.135
37	325	10165	36	43	1.33	0.361	7049.2	7049.2
38	177	424	16	14	0.337	0.162	21.989	21.989
39	F	531	F	18	F	0.309	F	-1896.9
40	209	2019	13	48	1.248	1.346	32.348	32.348

Table 4
Execution Time for ACCME, VMCON, GRG2

TP	ET				
	ACCME			GRG2	VMCON
	NLSOLV=1	NLSOLV=2	NLSOLV=3		
1	0.037	0.034	0.052	0.060	0.059
2	0.033	0.033	0.044	0.060	0.027
3	0.075	0.085	0.147	0.066	0.049
4	0.059	0.062	0.103	0.065	0.046
5	0.038	0.037	0.113	0.062	0.039
6	0.185	0.048	0.050	0.068	0.069
7	0.198	0.190	0.210	0.066	0.041
8	0.035	0.043	0.045	0.065	0.033
9	0.132	F	0.366	0.086	0.049
10	0.040	0.057	0.063	0.068	0.072
11	0.035	0.035	0.045	0.060	0.086
12	0.045	0.059	0.066	0.079	0.046
13	0.055	0.038	0.060	0.067	0.049
14	0.096	0.132	0.154	0.070	0.051
15	0.073	0.050	0.091	0.073	0.093
16	0.111	0.117	0.160	0.086	0.086
17	0.061	0.044	0.076	0.076	0.039
18	0.044	0.040	0.063	0.071	0.044
19	0.054	0.049	0.054	0.089	0.065
20	0.044	0.043	0.065	F	0.048
21	F	0.217	2.002	0.099	0.069
22	0.073	0.053	0.061	0.094	0.086

Table 4 (Cont.)

TP	ET				
	ACCME			GRG2	VMCON
	NLSOLV=1	NLSOLV=2	NLSOLV=3		
23	0.070	0.077	0.074	0.093	0.095
24	0.080	0.120	0.055	0.099	0.068
25	0.156	0.090	0.082	0.113	0.067
26	0.350	0.143	0.377	0.107	0.054
27	1.248	0.198	2.657	0.166	0.112
28	2.296	1.432	3.467	0.146	0.117
29	0.189	0.110	0.565	0.097	0.075
30	0.116	0.097	0.163	0.157	0.079
31	0.090	0.081	0.126	0.120	0.087
32	3.324	0.302	0.128	0.144	0.101
33	0.162	0.138	0.242	0.132	0.079
34	0.163	0.140	0.228	0.120	0.080
35	0.134	0.112	0.154	0.135	0.151
36	0.135	0.112	0.153	0.136	0.154
37	1.592	1.361	2.423	0.361	1.33
38	1.369	0.483	2.169	0.162	0.337
39	2.794	0.634	F	0.309	F
40	F	8.622	1.432	1.346	1.248