306 Test Problems for Nonlinear Programming with Optimal Solutions

- User's Guide -

Address: Prof. Dr. K. Schittkowski
Department of Computer Science
University of Bayreuth
D - 95440 Bayreuth

E-mail: klaus.schittkowski@uni-bayreuth.de
Web: http://www.klaus-schittkowski.de

Date: December 10, 2008

Abstract

The availability of nonlinear programming test problems is extremely important to test optimization codes or to develop new algorithms. We describe the usage of the Fortran subroutines for all 306 test problems of two previous collections of the author, see Hock and Schittkowski [4] and Schittkowski [9]. For each test example, we provide at least a local optimal solution. The execution of the codes, the program organization, and some numerical test results are presented to be able to test and compare own implementations.

1 Introduction

Some years ago, the author published two test problem collections for testing nonlinear programming codes, see Hock and Schittkowski [4] and Schittkowski [9]. The problems are widely used and contained also in other test problem collections, for example in the Cute library of Bongartz et al. [1], available through the URL

`http://www.cse.clrc.ac.uk/activity/cute`

The test problem collection of Spellucci [12] is available through the ftp site

`ftp://ftp.mathematik.tu-darmstadt.de/pub/department/software/opti/`

confer also the benchmark test page maintained by Mittelmann

`http://plato.la.asu.edu/bench.html`

In addition, AMPL versions of all test problems of the two collections are available through the links

`http://www.sor.princeton.edu/~rvdb/ampl/nlmodels/hs/index.html`

and

`http://www.sor.princeton.edu/~rvdb/ampl/nlmodels/s/index.html`

see also Fourer et al. [3] for more details about AMPL.

When developing a new version of a sequential quadratic programming algorithm, the test examples were investigated again and used for some numerical tests, see Schittkowski [11]. The purpose of the paper is to outline the usage of the codes and to make them available for public usage (21,800 lines of coding).

We consider the general optimization problem, to minimize an objective function $f(x)$ under nonlinear equality and inequality constraints,

$$x \in I\!R^n : \begin{array}{ll} \min f(x) & \\ a_j^T x + \beta_j \geq 0 \ , & j = 1, \ldots, m_{11}, \\ g_j(x) \geq 0 \ , & j = m_{11}+1, \ldots, m_1, \\ a_j^T x + \beta_j = 0 \ , & j = m_1+1, \ldots, m_{21}, \\ g_j(x) = 0 \ , & j = m_{21}+1, \ldots, m, \\ x_l \leq x \leq x_u & \end{array} \tag{1}$$

where x is an n-dimensional parameter vector. It is supposed that the first m_{11} inequality constraints and that the first $m_{21} - m_1$ equations are linear, whereas the remaining ones

are nonlinear. To facilitate the notation, we set $g_j(x) = a_j^T x + \beta_j$ for $j = 1, \ldots, m_{11}$ and $j = m_1 + 1, \ldots, m_{21}$. Objective function and constraints are supposed to be continuously differentiable on the whole $\mathbb{R}^n$.

The test problems have been used in the past to develop the nonlinear programming code NLPQL [8], a Fortran implementation of a sequential quadratic programming (SQP) algorithm. The design of the numerical algorithm is founded on extensive comparative numerical tests of Schittkowski [7], Schittkowski et al. [10], and Hock, Schittkowski [5]. To complete the numerical tests, an additional random test problem generator was developed for a major comparative study, see [7]. More than 100 test problems based on a finite element formulation are collected for the comparative evaluation in Schittkowski et al. [10].

All these efforts indicate the importance of a qualified set of test examples for debugging, validation, performance evaluation, and quantitative numerical comparisons with alternative codes. Although not collected in a very systematically way, the test problems represent all numerical difficulties we observe in practice, for example

1. badly scaled objective and constraint functions,
2. badly scaled variables,
3. non-smooth model functions,
4. ill-conditioned optimization problems,
5. non-regular solutions at points where the constraint qualification is not satisfied,
6. different local solutions,
7. infinitely many solutions.

Academic test problems allow either an analytical or a numerical investigation of all interesting properties, with nearly no or only limited efforts. On the other hand, nonlinear programming problems based on a *real-life* background are often too complex to serve as test problems, are often not available, are not programmed in a standard form as required for massive tests, or contain round-off and truncation errors, in particular if secondary iterative numerical algorithms are included to compute function and gradient values. The latter argument is crucial, since most non-trivial application problems generate numerical errors in the one or other form. Often gradients are only available by forward differences. However, we can use the academic test problems also in this situation, by adding randomly generated errors and by approximating derivatives numerically. A few corresponding test results are found in Schittkowski [11].

It is important that all test examples come with an optimal solution obtained by analytical investigation or numerical experimentation. Most examples are non-convex,

but we hope that at least in most cases, we are able to provide a global solution. For most test problems, analytical gradients are available. However, we cannot give a guarantee that they are correct and recommend usage of numerical differentiation. About 270 of 306 test problems do not have more than 10 variables.

The Fortran source codes of all test problems are made available through the link

`http://www.klaus-schittkowski.de/home.htm`

The usage of the subroutines is documented in Section 2 together with a detailed example. A driver program is listed that shows how a nonlinear programming code, in this case NLPQLP, would evaluate function and gradient values. The files that are provided by the author, are listed in Section 3 together with a description of the generated output. A main program that executes all 306 test examples within a loop and solves them by the code NLPQLP, see Schittkowski [11], is attached. The code contains also an evaluation of numerical results based on a decision, whether the result of a test run is considered to be a successful one or not. Some numerical tests are included in Section 4 which are helpful for comparing own implementations. An appendix contains a list of all individual results including performance data, i.e. number of function calls, number of iterations, errors in objective function, and constraint violations.

2 Usage of the Fortran Subroutines

A test problem is set up by

CALL TPno(MODE)

where *no* stands for any of the available test problem numbers. This section describes the organization of the FORTRAN subroutines and informs the user how to execute the test problems. Since it is assumed that at least a subset of the problems is used within a series of test runs for different optimization programs, the problems are coded in a very flexible manner. For example, it is possible to compute an arbitrary subset of constraint values. The parameter MODE describes the five possible operations of the subroutine.

MODE=1: The driving program will be provided with all information necessary to initialize an optimization program for the solution of the test problem, i.e. dimension, type and number of constraints, upper and lower bounds, starting point, derivatives of linear constraints, and, in particular, the exact or computed optimal solution.

MODE=2: The objective function $f(x)$ is computed at a current iterate x.

MODE=3: The gradient $\nabla f(x)$ of the objective function will be computed.

MODE=4: A predetermined subset of constraints $g_1(x)$, ..., $g_m(x)$ is evaluated at the actual iterate x.

MODE=5: The gradients of a predetermined subset of nonlinear constraints are computed, i.e. of $\nabla g_1(x)$, ..., $\nabla g_m(x)$.

The information on the test problem is delivered in the following common-blocks which have to be defined in the driving program with fixed array dimensions NMAX=101 and MMAX=50:

COMMON/L1/N,NILI,NINL,NELI,NENL: A call of TPno(1) gives on return the data:

N	Dimension of the problem, i.e. n.
NILI	Number of linear inequality constraints, i.e. m_{11}.
NINL	Number of nonlinear inequality constraints, i.e. $m_1 - m_{11}$.
NELI	Number of linear equality constraints, i.e. $m_{21} - m_1$.
NENL	Number of nonlinear equality constraints, i.e. $m - m_{21}$.

COMMON/L2/X(NMAX): For MODE=1, X will be set to a starting point from which the optimization process is to be started. For MODE>1, X must contain the argument x for which the problem functions or derivatives are to be computed.

COMMON/L3/G(MMAX): For all indices J with INDEX1(J)=.TRUE., G(J) is set to the j-th constraint value $g_j(x)$ (MODE=4).

COMMON/L4/GF(NMAX): Contains the partial derivatives of the objective function on return, i.e., GF(I) is set to $\frac{\partial}{\partial x_i} f(x)$, $i = 1, \ldots, n$ (MODE=3).

COMMON/L5/GG(MMAX,NMAX): For MODE=1, all constant partial derivatives are stored in GG. In particular, the rows 1, ..., m_{11} and $m_1 + 1$, ..., m_{21} of GG store the constant derivatives of the linear constraints. For MODE=5, the j-th row of GG defined by INDEX2(J)=.TRUE. will be replaced by the gradient of the j-th restriction, i.e. GG(J,I) is set to $\frac{\partial}{\partial x_i} g_j(x)$, if this term is not constant. Since all array dimensions of

the common blocks are defined by the exact values of n or m, respectively, we recommend to define GG as a one-dimensional array in the driving program and to use it there in the form GG((I-1)·M+J).

COMMON/L6/FX: For MODE=2, FX contains the objective function value $f(x)$ on return.

COMMON/L9/INDEX1(MMAX): The logical array INDEX1 has to be initialized by the user before calling TPno(4), and defines the restrictions which are to be computed in the case MODE=4. INDEX1 is not changed by the subroutine.

COMMON/L10/INDEX2(MMAX): The logical array INDEX2 has to be initialized by the user before calling TPno(5), and defines the gradients of the nonlinear restrictions which are to be computed in the case MODE=5. INDEX2 is not changed during a call of TPno.

COMMON/L11/LXL(NMAX): The logical array LXL informs about the existence of lower bounds. If there is a lower bound for the i-th variable, LXL(I) is set to .TRUE. during a call of TPno(1). Otherwise, we find LXL(I)=.FALSE..

COMMON/L12/LXU(NMAX): Same for the existence of upper bounds.

COMMON/L13/XL(NMAX): If LXL(I)=.TRUE., XL(I) obtains a lower bound for the i-th variable during a call of TPno(1).

COMMON/L14/XU(NMAX): if LXU(I)=.TRUE., XU(I) is set to an upper bound for the i-th variable during a call of TPno(1).

COMMON/L20/LEX,NEX,FEX,XEX(NMAX) : L20 contains information on the optimal solution of the problem and is set during a call of TPno(1). If LEX=.FALSE., an analytical solution is not known a priori and XEX stores the best computed solution known to the author. Otherwise, we have LEX=.TRUE. and NEX shows the number of known local optimal minima. FEX contains the best objective function value known to the author and XEX the best known solution.

Note that for some test problems, analytical gradients are not available. There is no warranty that the gradients, as far as included, are correct. In some cases, the implementation differs slightly from the printed documentation in [4] and [9] due to misprints in the source.

To give an example, we consider Rosenbrock's post office problem, i.e. test problem TP37 of the first collection, [4], given in the form

$$x = (x_1, x_2, x_3)^T \in I\!R^3 : \quad \begin{array}{ll} \min -x_1 x_2 x_3 & \\ x_1 + 2x_2 + 2x_3 - 72 \leq 0, & \\ x_1 + 2x_2 + 2x_3 \geq 0, & \\ 0 \leq x_i \leq 42, & i = 1, 2, 3 \end{array} \tag{2}$$

We have three variables, i.e. $n = 3$, bounds for all variables and only two linear inequality constraints, i.e. $m_{11} = m_1 = m_{21} = m = 3$. The Fortran source code is:

```
      SUBROUTINE TP37(MODE)
      INTEGER    MODE,NMAX,MMAX,N,NILI,NINL,NELI,NENL,NEX
      REAL*8     X,G,GF,GG,FX,XL,XU,FEX,XEX
      LOGICAL    INDEX1,INDEX2,LXL,LXU,LEX
      PARAMETER (NMAX=101, MMAX=50)
      COMMON     /L1/  N,NILI,NINL,NELI,NENL
     /           /L2/  X(NMAX)
     /           /L3/  G(MMAX)
     /           /L4/  GF(NMAX)
     /           /L5/  GG(MMAX,NMAX)
     /           /L6/  FX
     /           /L9/  INDEX1(MMAX)
     /           /L10/ INDEX2(MMAX)
     /           /L11/ LXL(NMAX)
     /           /L12/ LXU(NMAX)
     /           /L13/ XL(NMAX)
     /           /L14/ XU(NMAX)
     /           /L20/ LEX,NEX,FEX,XEX(NMAX)
      REAL*8 V,DATAN,DSIN,DCOS,V1,V2,V3,V4
      LOGICAL LXL(2),LXU(2),INDEX1(1),INDEX2(1),LEX
      GOTO (1,2,3,4,5),MODE
1     N=3
      NILI=2
      NINL=0
      NELI=0
      NENL=0
      DO 6 I=1,3
      X(I)=10.D0
      LXL(I)=.TRUE.
      LXU(I)=.TRUE.
      XU(I)=42.D0
6     XL(I)=0.D0
      LEX=.TRUE.
      NEX=1
      XEX(1)=24.D0
      XEX(2)=12.D0
      XEX(3)=12.D0
      FEX=-3.456D+3
      GG(1,1)=-1.D0
      GG(1,2)=-2.D0
      GG(1,3)=-2.D0
      GG(2,1)=1.D0
      GG(2,2)=2.D0
```

```
      GG(2,3)=2.D0
      RETURN
 2    FX=-X(1)*X(2)*X(3)
      RETURN
 3    GF(1)=-X(2)*X(3)
      GF(2)=-X(1)*X(3)
      GF(3)=-X(1)*X(2)
      RETURN
 4    IF (INDEX1(1)) G(1)=72.D0-X(1)-2.D0*X(2)-2.D0*X(3)
      IF (INDEX1(2)) G(2)=X(1)+2.D0*X(2)+2.D0*X(3)
 5    RETURN
      END
```

To show how to call subroutine TP37, we list the corresponding Fortran source code executing NLPQLP.

```
      IMPLICIT   NONE
      INTEGER    NMAX, MMAX, MNNMAX, LWA, LIWA, LACTIV
      PARAMETER (NMAX   = 101,
     /           MMAX   = 50,
     /           MNNMAX = NMAX + NMAX + MMAX + 2,
     /           LWA    = 1.5*NMAX*NMAX + 33*NMAX + 9*MMAX + 200,
     /           LIWA   = NMAX + 10,
     /           LACTIV = 2*MMAX + 10)
      REAL*8     X, G, DF, DG, F, XL, XU, FEX, XEX,
     /           U(MNNMAX), C(NMAX,NMAX), D(NMAX), WA(LWA)
      REAL*8     ACC, ACCQP, RHOB, STPMIN
      INTEGER    N, NILI, NINL, NELI, NENL, IWA(LIWA), M, ME, MI,
     /           MNN2, MODE, IPRINT, IOUT, MAXFUN, MAXIT, NEX,
     /           MAX_NM, L, IFAIL, I, J
      LOGICAL    INDEX1, INDEX2, LXL, LXU, LEX, ACTIVE(LACTIV)
      EXTERNAL   QL
      COMMON     /L1/  N,NILI,NINL,NELI,NENL
     /           /L2/  X(NMAX)
     /           /L3/  G(MMAX)
     /           /L4/  DF(NMAX)
     /           /L5/  DG(MMAX,NMAX)
     /           /L6/  F
     /           /L9/  INDEX1(MMAX)
     /           /L10/ INDEX2(MMAX)
     /           /L11/ LXL(NMAX)
     /           /L12/ LXU(NMAX)
     /           /L13/ XL(NMAX)
     /           /L14/ XU(NMAX)
     /           /L20/ LEX,NEX,FEX,XEX(NMAX)
C
C   Optimization settings for NLPQLP
C
      MODE   = 0
      IPRINT = 2
      IOUT   = 6
      MAXFUN = 20
      MAXIT  = 500
      MAX_NM = 30
      RHOB   = 1.0D3
      L      = 1
      STPMIN = 1.0D-8
```

```
      ACC    = 1.0D-12
      ACCQP  = 1.0D-12
C
C   Model parameters and bounds
C
      CALL TP37(1)
      ME = NELI + NENL
      MI = NILI + NINL
      M  = ME + MI
      DO I=1,N
        IF (.NOT.LXL(I)) XL(I) = X(I) - 1.0D+10
        IF (.NOT.LXU(I)) XU(I) = X(I) + 1.0D+10
        IF (X(I).LT.XL(I)) X(I) = XL(I)
        IF (X(I).GT.XU(I)) X(I) = XU(I)
      ENDDO
      DO J=1,M
        INDEX1(J) = .TRUE.
      ENDDO
C
C   Call of NLPQLP with reverse communication
C
      IFAIL = 0
    1 CONTINUE
      IF (IFAIL.EQ.0.OR.IFAIL.EQ.-1) THEN
C
C   Retrieve function values
C
         CALL TP37(2)
         CALL TP37(4)
      ENDIF
      IF (IFAIL.EQ.0.OR.IFAIL.EQ.-2) THEN
C
C   Retrieve derivative values
C
         CALL TP37(3)
         CALL TP37(5)
      ENDIF
      CALL NLPQLP(L,M,ME,MMAX,N,NMAX,M+N+N+2,X,F,G,DF,DG,U,XL,XU,C,D,
     /            ACC,ACCQP,STPMIN,MAXFUN,MAXIT,MAX_NM,RHOB,
     /            IPRINT,MODE,IOUT,IFAIL,WA(M+1),LWA,IWA,LIWA,ACTIVE,
     /            LACTIV,.TRUE.,QL)
      IF (IFAIL.LT.0) GOTO 1
C
C   End
C
      STOP
      END
```

The following output is generated:

```
------------------------------------------------------------------
 START OF THE SEQUENTIAL QUADRATIC PROGRAMMING ALGORITHM
------------------------------------------------------------------

 Parameters:
    N      =       3
```

```
   M      =       2
   ME     =       0
   MODE   =       0
   ACC    =   0.1000D-11
   ACCQP  =   0.1000D-11
   STPMIN =   0.1000D-07
   RHOB   =   0.1000D+04
   MAXFUN =      20
   MAXNM  =      30
   MAXIT  =     500
   IPRINT =       2

 Output in the following order:
   IT    - iteration number
   F     - objective function value
   SCV   - sum of constraint violations
   NA    - number of active constraints
   I     - number of line search iterations
   ALPHA - steplength parameter
   DELTA - additional variable to prevent inconsistency
   KKT   - Karush-Kuhn-Tucker optimality criterion

  IT         F          SCV      NA  I    ALPHA     DELTA     KKT
-----------------------------------------------------------------------
   1 -0.10000000D+04  0.00D+00    2  0  0.00D+00  0.00D+00  0.44D+04
   2 -0.23625000D+04  0.00D+00    1  1  0.10D+01  0.00D+00  0.11D+04
   3 -0.32507304D+04  0.00D+00    1  1  0.10D+01  0.00D+00  0.69D+03
   4 -0.34557650D+04  0.71D-14    1  2  0.54D+00  0.00D+00  0.55D+00
   5 -0.34559928D+04  0.00D+00    1  1  0.10D+01  0.00D+00  0.15D-01
   6 -0.34560000D+04  0.00D+00    1  1  0.10D+01  0.00D+00  0.33D-06
   7 -0.34560000D+04  0.00D+00    1  1  0.10D+01  0.00D+00  0.24D-12

   Objective function value:      F(X)  = -0.34560000D+04
   Solution values:               X     =
      0.24000000D+02  0.12000000D+02  0.12000000D+02
   Distances from lower bounds:   X-XL  =
      0.24000000D+02  0.12000000D+02  0.12000000D+02
   Distances from upper bounds:   XU-X  =
      0.18000000D+02  0.30000000D+02  0.30000000D+02
   Multipliers for lower bounds:  U     =
      0.00000000D+00  0.00000000D+00  0.00000000D+00
   Multipliers for upper bounds:  U     =
      0.00000000D+00  0.00000000D+00  0.00000000D+00
   Constraint values:             G(X)  =
      0.00000000D+00  0.72000000D+02
   Multipliers for constraints:   U     =
      0.14400000D+03  0.00000000D+00
   Number of function calls:      NFUNC =       8
   Number of gradient calls:      NGRAD =       7
   Number of calls of QP solver:  NQL   =       7
```

3 Program Organization

All 306 test problems of the two collections [4] and [9] are available together with a test frame. A decision is made which of the runs is successful, and performance results are

evaluated. With the default tolerances given, all problems can be solved successfully by the code NLPQLP, a new version of the SQP implementation NLPQL of the author [11]. Results of NLPQLP are discussed in the subsequent section.

The following files are provided by the author and can be downloaded from his home page:

1. **PROB.FOR**: Fortran codes of the test problems of the two collections mentioned above.

2. **CONV.FOR**: Interface between the individual test problem codes and an available optimization routine to facilitate the calling procedure and to be able to execute all test problems within a loop. The subroutine is invoked by

   ```
   CALL CONV(MODE)
   ```

   where the test problem number is passed through the common block

   ```
   COMMON/L8/NTP
   ```

3. **TESTP.FOR**: Test program that executes all 306 problems in a loop. The calling sequence for the SQP code NLPQLP is included to give an example. An approximation formula for gradient evaluations by forward differences is included. The code generates the output files listed below.

4. **TEST.DAT**: Output file of the test frame containing numerical results obtained by NLPQLP:

TP1	2	0	0	0	26	19	64	0.00000000D+00	0.44420806D-07	0.444D-07	0.000D+00
TP2	2	0	0	0	21	15	51	0.50426188D-01	0.49412293D+01	0.970D+02	0.000D+00
TP3	2	0	0	0	10	10	30	0.00000000D+00	0.16286892D-17	0.163D-17	0.000D+00
TP4	2	0	0	0	2	2	6	0.26666667D+01	0.26666667D+01	0.000D+00	0.000D+00
TP5	2	0	0	0	8	6	20	-0.19132230D+01	-0.19132230D+01	0.139D-10	0.000D+00
TP6	2	1	1	0	10	9	28	0.00000000D+00	0.18696094D-12	0.187D-12	0.219D-04
TP7	2	1	1	0	13	12	37	-0.17320508D+01	-0.17320508D+01	0.134D-12	0.893D-13
TP8	2	2	2	0	5	5	15	-0.10000000D+01	-0.10000000D+01	0.000D+00	0.533D-04
								.....			

   The following data are listed, see TESTP.FOR for details:

NTP	Test problem number
N	Number of variables
ME	Number of equality constraints
M	Number of constraints
IFAIL	Convergence criterion
NF	Number of objective function evaluations
NDF	Number of gradient evaluations of objective function
NEF	Number of equivalent function evaluations, i.e. NF plus number of function calls needed for gradient approximation
FEX	Exact objective function value
F	Computed objective function value
DFX	Relative error in objective function
DGX	Sum of constraint violations including bound violations

5. **TEST.TEX**: Same as above, but with Latex separators.

6. **RESULT.DAT**: The following summary is shown:

   (a) Tolerance for gradient approximation

   (b) Termination accuracy for NLP routine

   (c) Total number of test runs

   (d) Number of successful test runs

   (e) Number of better solutions

   (f) Number of local solutions

   (g) Number of test runs with error message IFAIL>0

   (h) Tolerance for determining successful return

   (i) Average number of function evaluations

   (j) Average number of gradient evaluations

   (k) Average number of equivalent function calls

7. **TEMP.DAT**: Contains the same data in one row.

4 Numerical Results

Sequential quadratic programming or SQP methods belong to the most powerful nonlinear programming algorithms we know today for solving differentiable nonlinear programming problems of the form (1). The theoretical background is described for example by

Stoer [13] in form of a review, or in Spellucci [12] in form of an extensive text book. From the more practical point of view, SQP methods are also introduced in the books of Papalambros, Wilde [6] and Edgar, Himmelblau [2]. Their excellent numerical performance is evaluated and compared to other methods in Schittkowski [7]. Since many years they belong to the most frequently used algorithms to solve practical optimization problems.

The numerical results of some computational tests are reported in this section. They have been obtained by the code NLPQLP [11], a Fortran implementation of a sequential quadratic programming algorithm. The Fortran subroutine NLPQLP solves smooth nonlinear programming problems and is an extension of the code NLPQL, see Schittkowski [8]. The new version is specifically tuned to run under distributed systems and to apply non-monotone line search in error situations.

Since analytical derivatives are not available for all problems, we approximate them numerically by a 2-point difference formula. The test examples are provided with exact solutions, either known from analytical solutions or from the best numerical data found so far. Since the calculation times are very short, less than one second for solving all 306 test problems, we count only function and gradient evaluations. This is a realistic assumption, since for the practical applications, calculation times for evaluating model functions dominate and the numerical efforts within an optimization code are negligible.

First we need a criterion to decide whether the result of a test run is considered as a successful return or not. Let $\epsilon > 0$ be a tolerance for defining the relative termination accuracy, x_k the final iterate of a test run, and $x^\star$ the supposed exact solution as reported by the two test problem collections. Then we call the output of an execution of NLPQLP a successful return, if the relative error in objective function is less than ϵ and if the sum of all constraint violations less than ϵ^2, i.e., if

$$f(x_k) - f(x^\star) < \epsilon |f(x^\star)| \ , \ \text{if } f(x^\star) \neq 0 \ \ ,$$

or

$$f(x_k) < \epsilon \ , \ \text{if } f(x^\star) = 0 \ \ ,$$

and

$$r(x_k) := \sum_{j=1}^{m_e} |g_j(x_k)| + \sum_{j=m_e+1}^{m} |\min(0, g_j(x_k))| < \epsilon^2 \ \ .$$

We take into account that NLPQLP returns a solution with a better function value than the known one, subject to the error tolerance of the allowed constraint violation. However, there is still the possibility that NLPQLP terminates at a local solution different from the one known in advance. Thus, we call a test run a successful one, if NLPQLP terminates with error message IFAIL=0, and if

$$f(x_k) - f(x^\star) \geq \epsilon |f(x^\star)| \ , \ \text{if } f(x^\star) \neq 0 \ \ ,$$

or

$$f(x_k) \geq \epsilon \ , \ \text{if } f(x^\star) = 0 \ \ ,$$

and

$$r(x_k) < \epsilon^2 \ \ .$$

For our numerical tests, we use $\epsilon = 0.01$, i.e., we require a final accuracy of one per cent. NLPQLP is executed with termination accuracy ACC=10^{-7}, and $MAXIT$=*500*. All problems are executed with the same set of solution tolerances.

When executing NLPQLP, the following results are obtained:

```
*** Local solution:  TP 16
*** Local solution:  TP 25
*** Local solution:  TP 33
*** Local solution:  TP 54
*** Local solution:  TP 55
*** Local solution:  TP 57
*** Local solution:  TP 59
*** Local solution:  TP105
*** Local solution:  TP202
*** Local solution:  TP220
*** Local solution:  TP259
*** Local solution:  TP265
*** Local solution:  TP312
*** Local solution:  TP327
*** Local solution:  TP338
*** Local solution:  TP350
*** Local solution:  TP358
*** Local solution:  TP362
*** Local solution:  TP371


*** NUMERICAL RESULTS ***

Tolerance for gradient approximation:                0.481D-05

Termination accuracy for NLP routine:                  0.1D-06

Total number of test runs:                                 306

Number of successful test runs:                            306
- constraint violation less than squared tolerance and
  either error in objective less than tolerance or
  termination criteria of NLP routine satisfied

Number of better test runs:                                  0
- constraint violation less than squared tolerance and
  error in objective less than -tolerance

Number of local solutions obtained:                         19

Number of runs without satisfying termination accuracy:      0
- as indicated by NLP routine, i.e. IFAIL>0

Tolerance for determining successful return:           0.1D-01
```

```
Average number of function evaluations:                      32
- NLFUNC calls for successful returns

Average number of gradient evaluations:                      20
- NLGRAD calls for successful returns

Average number of equivalent function calls                 557
- additionlly counted function calls for gradient approximations
```

References

[1] Bongartz I., Conn A.R., Gould N., Toint Ph. (1995): *CUTE: Constrained and unconstrained testing environment*, Transactions on Mathematical Software, Vol. 21, No. 1, 123-160

[2] Edgar T.F., Himmelblau D.M. (1988): *Optimization of Chemical Processes,* McGraw Hill

[3] Fourer R., Gay D.M., Kernighan B.W. (2002): *AMPL: A Modeling Language for Mathematical Programming*, Brooks/Cole Publishing Company

[4] Hock W., Schittkowski K. (1981): *Test Examples for Nonlinear Programming Codes,* Lecture Notes in Economics and Mathematical Systems, Vol. 187, Springer

[5] Hock W., Schittkowski K. (1983): *A comparative performance evaluation of 27 nonlinear programming codes*, Computing, Vol. 30, 335-358

[6] Papalambros P.Y., Wilde D.J. (1988): *Principles of Optimal Design,* Cambridge University Press

[7] Schittkowski K. (1980): *Nonlinear Programming Codes,* Lecture Notes in Economics and Mathematical Systems, Vol. 183 Springer

[8] Schittkowski K. (1985/86): *NLPQL: A Fortran subroutine solving constrained nonlinear programming problems,* Annals of Operations Research, Vol. 5, 485-500

[9] Schittkowski K. (1987): *More Test Examples for Nonlinear Programming,* Lecture Notes in Economics and Mathematical Systems, Vol. 182, Springer

[10] Schittkowski K., Zillober C., Zotemantel R. (1994): *Numerical comparison of nonlinear programming algorithms for structural optimization*, Structural Optimization, Vol. 7, No. 1, 1-28

[11] Schittkowski K. (2012): *NLPQLP: A Fortran implementation of a sequential quadratic programming algorithm with distributed and non-monotone line search - User's Guide, Version 4.0*, Report, Department of Computer Science, University of Bayreuth

[12] Spellucci P. (1993): *Numerische Verfahren der nichtlinearen Optimierung,* Birkhäuser

[13] Stoer J. (1985): *Foundations of recursive quadratic programming methods for solving nonlinear programs,* in: Computational Mathematical Programming, K. Schittkowski, ed., NATO ASI Series, Series F: Computer and Systems Sciences, Vol. 15, Springer

APPENDIX: Individual Results

The appendix contains a list of all test problems with the data

TP	test problem number,
N	number of variables,
ME	number of equality constraints,
M	number of constraints,
IFAIL	convergence criterion,
NF	number of objective function evaluations,
NDF	number of gradient evaluations of objective function,
NEF	number of equivalent function evaluations, i.e. NF plus number of function calls needed for gradient approximation,
FEX	exact objective function value,
F	computed objective function value,
DFX	relative error in objective function,
DGX	sum of constraint violations including bound violations.

The performance results are obtained by NLPQLP under the conditions outlined in Section 4.

TP	*N*	*ME*	*M*	*IFAIL*	*NF*	*NDF*	*NEF*	*FEX*	*F*	*DFX*	*DGX*
1	2	0	0	0	26	19	102	0.00000000E+00	0.45240196E-07	0.45E-07	0.00E+00
2	2	0	0	0	20	15	80	0.50426188E-01	0.50426193E-01	0.11E-06	0.00E+00
3	2	0	0	0	10	10	50	0.00000000E+00	0.11652229E-25	0.12E-25	0.00E+00
4	2	0	0	0	2	2	10	0.26666667E+01	0.26666667E+01	0.00E+00	0.00E+00
5	2	0	0	0	8	6	32	-0.19132230E+01	-0.19132230E+01	0.14E-10	0.00E+00

(continued)

TP	*N*	*ME*	*M*	*IFAIL*	*NF*	*NDF*	*NEF*	*FEX*	*F*	*DFX*	*DGX*
6	2	1	1	0	10	9	46	0.00000000E+00	0.19130383E-12	0.19E-12	0.22E-04
7	2	1	1	0	13	12	61	-0.17320508E+01	-0.17320508E+01	-0.36E-13	0.22E-12
8	2	2	2	0	5	5	25	-0.10000000E+01	-0.10000000E+01	0.00E+00	0.53E-04
9	2	1	1	0	6	6	30	-0.50000000E+00	-0.50000000E+00	0.55E-09	0.85E-13
10	2	0	1	0	14	12	62	-0.10000000E+01	-0.10000000E+01	-0.32E-07	0.63E-07
11	2	0	1	0	10	9	46	-0.84984642E+01	-0.84984642E+01	-0.30E-12	0.85E-12
12	2	0	1	0	11	10	51	-0.30000000E+02	-0.30000000E+02	0.00E+00	0.00E+00
13	2	0	1	0	50	50	250	0.10000000E+01	0.10000017E+01	0.17E-05	0.00E+00
14	2	1	2	0	6	6	30	0.13934650E+01	0.13934650E+01	-0.10E-11	0.77E-12
15	2	0	2	0	3	3	15	0.30650000E+01	0.30650000E+01	0.19E-11	0.00E+00
17	2	0	2	0	21	17	89	0.10000000E+01	0.10000000E+01	0.47E-09	0.00E+00
18	2	0	2	0	8	8	40	0.50000000E+01	0.50000000E+01	-0.11E-08	0.39E-07
19	2	0	2	0	7	7	35	-0.69618139E+04	-0.69618139E+04	-0.17E-14	0.36E-14
20	2	0	3	0	5	5	25	0.38198730E+02	0.38198730E+02	0.18E-09	0.00E+00
21	2	0	1	0	3	2	11	-0.99960000E+02	-0.99960000E+02	-0.89E-11	0.00E+00
22	2	0	2	0	7	6	31	0.10000000E+01	0.10000000E+01	-0.23E-11	0.34E-11
23	2	0	5	0	7	7	35	0.20000000E+01	0.20000000E+01	0.27E-13	0.00E+00
24	2	0	3	0	5	5	25	-0.10000000E+01	-0.10000000E+01	0.58E-11	0.10E-10
26	3	1	1	0	17	15	107	0.00000000E+00	0.74439335E-07	0.74E-07	0.64E-04
27	3	1	1	0	37	22	169	0.40000000E+01	0.40000000E+01	0.51E-10	0.14E-12
28	3	1	1	0	5	4	29	0.00000000E+00	0.35239880E-20	0.35E-20	0.66E-10
29	3	0	1	0	13	12	85	-0.22627417E+02	-0.22627417E+02	-0.21E-09	0.69E-08
30	3	0	1	0	14	14	98	0.10000000E+01	0.10000000E+01	0.15E-07	0.00E+00
31	3	0	1	0	12	7	54	0.60000000E+01	0.60000000E+01	-0.51E-08	0.80E-08
32	3	1	2	0	3	3	21	0.10000000E+01	0.10000000E+01	0.59E-11	0.29E-11
34	3	0	2	0	8	8	56	-0.83403245E+00	-0.83403245E+00	-0.12E-09	0.11E-08
35	3	0	1	0	7	7	49	0.11111111E+00	0.11111111E+00	0.11E-11	0.00E+00
36	3	0	1	0	2	2	14	-0.33000000E+04	-0.33000000E+04	-0.55E-11	0.16E-09
37	3	0	2	0	8	8	56	-0.34560000E+04	-0.34560000E+04	0.25E-13	0.00E+00
38	4	0	0	0	100	78	724	0.00000000E+00	0.51825054E-07	0.52E-07	0.00E+00
39	4	2	2	0	14	12	110	-0.10000000E+01	-0.10000000E+01	-0.13E-09	0.13E-09
40	4	3	3	0	6	6	54	-0.25000000E+00	-0.25000000E+00	-0.93E-09	0.43E-09
41	4	1	1	0	8	8	72	0.19259259E+01	0.19259259E+01	0.74E-10	0.46E-13
42	4	2	2	0	10	8	74	0.13857864E+02	0.13857864E+02	-0.31E-08	0.17E-07
43	4	0	3	0	14	11	102	-0.44000000E+02	-0.44000000E+02	-0.17E-10	0.26E-09
44	4	0	6	0	6	6	54	-0.15000000E+02	-0.15000000E+02	-0.16E-11	0.16E-10
45	5	0	0	0	8	8	88	0.10000000E+01	0.10000000E+01	0.00E+00	0.00E+00
46	5	2	2	0	14	12	134	0.00000000E+00	0.55456287E-06	0.55E-06	0.19E-06
47	5	3	3	0	17	13	147	0.00000000E+00	0.46052105E-09	0.46E-09	0.93E-07
48	5	2	2	0	9	8	89	0.00000000E+00	0.27709254E-07	0.28E-07	0.81E-13
49	5	2	2	0	9	6	69	0.00000000E+00	0.22835781E-04	0.23E-04	0.12E-12
50	5	3	3	0	18	14	158	0.00000000E+00	0.31208008E-08	0.31E-08	0.36E-14
51	5	3	3	0	5	3	35	0.00000000E+00	0.62379882E-22	0.62E-22	0.52E-11
52	5	3	3	0	8	6	68	0.53266476E+01	0.53266476E+01	0.40E-10	0.19E-14
53	5	3	3	0	8	7	78	0.40930233E+01	0.40930233E+01	0.74E-08	0.84E-13
56	7	4	4	0	11	9	137	-0.34560000E+01	-0.34560000E+01	-0.10E-07	0.24E-07
60	3	1	1	0	10	9	64	0.32568200E-01	0.32568202E-01	0.64E-07	0.19E-06
61	3	2	2	0	9	8	57	-0.14364614E+03	-0.14364614E+03	0.22E-10	0.12E-12
62	3	1	1	0	14	9	68	-0.26272514E+05	-0.26272514E+05	-0.50E-12	0.55E-15
63	3	2	2	0	8	8	56	0.96171517E+03	0.96171517E+03	0.30E-11	0.14E-09
64	3	0	1	0	149	80	629	0.62998424E+04	0.62998424E+04	-0.46E-10	0.11E-11
65	3	0	1	0	8	8	56	0.95352886E+00	0.95352882E+00	-0.42E-07	0.54E-06

(continued)

TP	*N*	*ME*	*M*	*IFAIL*	*NF*	*NDF*	*NEF*	*FEX*	*F*	*DFX*	*DGX*
66	3	0	2	0	7	7	49	0.51816327E+00	0.51816327E+00	-0.22E-08	0.31E-08
67	3	0	14	0	20	20	140	-0.11620365E+04	-0.11620365E+04	-0.15E-07	0.00E+00
68	4	2	2	0	33	26	241	-0.92042502E+00	-0.92042501E+00	0.95E-08	0.96E-07
69	4	2	2	0	63	40	383	-0.95671289E+03	-0.95671289E+03	0.43E-09	0.15E-10
70	4	0	1	0	37	34	309	0.74984636E-02	0.74984827E-02	0.26E-05	0.00E+00
71	4	1	2	0	5	5	45	0.17014017E+02	0.17014017E+02	-0.26E-08	0.22E-08
72	4	0	2	0	22	22	198	0.72767936E+03	0.72767936E+03	-0.31E-08	0.11E-11
73	4	1	3	0	5	5	45	0.29894378E+02	0.29894378E+02	0.62E-10	0.26E-11
74	4	3	5	0	10	10	90	0.51264981E+04	0.51264981E+04	0.50E-10	0.10E-09
75	4	3	5	0	9	9	81	0.51744129E+04	0.51744127E+04	-0.37E-07	0.18E-12
76	4	0	3	0	6	6	54	-0.46818182E+01	-0.46818182E+01	0.45E-10	0.49E-13
77	5	2	2	0	16	15	166	0.24150513E+00	0.24150513E+00	-0.14E-07	0.35E-07
78	5	3	3	0	8	8	88	-0.29197004E+01	-0.29197004E+01	0.49E-10	0.18E-11
79	5	3	3	0	10	9	100	0.78776821E-01	0.78776822E-01	0.90E-08	0.27E-07
80	5	3	3	0	7	7	77	0.53949848E-01	0.53949847E-01	-0.73E-08	0.72E-08
81	5	3	3	0	8	8	88	0.53949848E-01	0.53949846E-01	-0.28E-07	0.27E-07
83	5	0	6	0	35	5	85	-0.30665539E+05	-0.30665539E+05	-0.27E-11	0.36E-14
84	5	0	6	0	72	44	512	-0.52803351E+07	-0.52803351E+07	-0.29E-10	0.58E-10
85	5	0	38	0	91	56	651	-0.19051553E+01	-0.19051553E+01	0.53E-08	0.16E-06
86	5	0	10	0	6	5	56	-0.32348679E+02	-0.32348679E+02	0.18E-09	0.31E-13
87	6	4	4	0	20	16	212	0.89275977E+04	0.89275977E+04	0.61E-10	0.38E-08
88	2	0	1	0	26	20	106	0.13626568E+01	0.13626873E+01	0.22E-04	0.00E+00
89	3	0	1	0	34	22	166	0.13626568E+01	0.13626873E+01	0.22E-04	0.17E-10
90	4	0	1	0	26	20	186	0.13626568E+01	0.13626872E+01	0.22E-04	0.37E-10
91	5	0	1	0	33	23	263	0.13626568E+01	0.13626873E+01	0.22E-04	0.16E-10
92	6	0	1	0	41	30	401	0.13626568E+01	0.13626872E+01	0.22E-04	0.56E-10
93	6	0	2	0	15	12	159	0.13507596E+03	0.13507596E+03	0.12E-07	0.40E-09
95	6	0	4	0	2	2	26	0.15619514E-01	0.15619525E-01	0.69E-06	0.00E+00
96	6	0	4	0	2	2	26	0.15619513E-01	0.15619525E-01	0.76E-06	0.00E+00
97	6	0	4	0	7	7	91	0.31358089E+01	0.31358089E+01	0.80E-08	0.00E+00
98	6	0	4	0	7	7	91	0.31358089E+01	0.31358089E+01	0.80E-08	0.00E+00
99	7	2	2	0	70	19	336	-0.83107989E+09	-0.83107989E+09	0.71E-11	0.12E-09
100	7	0	4	0	20	14	216	0.68063006E+03	0.68063006E+03	0.10E-09	0.25E-07
101	7	0	6	0	66	39	612	0.18097648E+04	0.18097648E+04	0.43E-12	0.23E-11
102	7	0	6	0	55	35	545	0.91188057E+03	0.91188057E+03	0.12E-09	0.18E-11
103	7	0	6	0	37	26	401	0.54366796E+03	0.54366796E+03	0.75E-10	0.92E-12
104	8	0	6	0	19	17	291	0.39511634E+01	0.39511634E+01	-0.11E-08	0.17E-08
106	8	0	6	0	283	109	2027	0.70492480E+04	0.70492480E+04	0.29E-08	0.58E-10
107	9	6	6	0	8	8	152	0.50550118E+04	0.50550114E+04	-0.70E-07	0.27E-13
108	9	0	13	0	13	13	247	-0.86602540E+00	-0.86602540E+00	-0.65E-09	0.13E-08
109	9	6	10	0	23	18	347	0.53620693E+04	0.53620692E+04	-0.18E-07	0.37E-12
110	10	0	0	0	10	7	150	-0.45778470E+02	-0.45778470E+02	0.17E-09	0.00E+00
111	10	3	3	0	51	51	1071	-0.47761090E+02	-0.47761091E+02	-0.13E-07	0.19E-09
112	10	3	3	0	37	21	457	-0.47761086E+02	-0.47761091E+02	-0.10E-06	0.20E-14
113	10	0	8	0	16	13	276	0.24306209E+02	0.24306209E+02	0.16E-09	0.37E-08
114	10	3	11	0	36	32	676	-0.17688070E+04	-0.17688070E+04	-0.16E-09	0.45E-12
116	13	0	15	0	84	61	1670	0.97588409E+02	0.97591035E+02	0.27E-04	0.36E-14
117	15	0	5	0	16	16	496	0.32348679E+02	0.32348679E+02	0.87E-09	0.00E+00
118	15	0	29	0	19	19	589	0.66482045E+03	0.66482045E+03	0.13E-10	0.29E-10
119	16	8	8	0	12	9	300	0.24489970E+03	0.24489970E+03	0.22E-10	0.68E-13
200	2	0	0	0	26	19	102	0.00000000E+00	0.45240196E-07	0.45E-07	0.00E+00
201	2	0	0	0	5	3	17	0.00000000E+00	0.28814501E-22	0.29E-22	0.00E+00

(continued)

TP	*N*	*ME*	*M*	*IFAIL*	*NF*	*NDF*	*NEF*	*FEX*	*F*	*DFX*	*DGX*
203	2	0	0	0	12	10	52	0.00000000E+00	0.78080431E-08	0.78E-08	0.00E+00
204	2	0	0	0	7	6	31	0.18360100E+00	0.18360120E+00	0.11E-05	0.00E+00
205	2	0	0	0	14	12	62	0.00000000E+00	0.15623221E-07	0.16E-07	0.00E+00
206	2	0	0	0	10	7	38	0.00000000E+00	0.21917300E-10	0.22E-10	0.00E+00
207	2	0	0	0	14	13	66	0.00000000E+00	0.27034254E-07	0.27E-07	0.00E+00
208	2	0	0	0	47	34	183	0.00000000E+00	0.11224337E-07	0.11E-07	0.00E+00
209	2	0	0	0	182	129	698	0.00000000E+00	0.20021064E-07	0.20E-07	0.00E+00
210	2	0	0	0	7	5	27	0.00000000E+00	0.39924439E-05	0.40E-05	0.00E+00
211	2	0	0	0	26	20	106	0.00000000E+00	0.47794459E-08	0.48E-08	0.00E+00
212	2	0	0	0	19	12	67	0.00000000E+00	0.64625833E-09	0.65E-09	0.00E+00
213	2	0	0	0	21	15	81	0.00000000E+00	0.68313237E-04	0.68E-04	0.00E+00
214	2	0	0	0	55	27	163	0.00000000E+00	0.19010374E-03	0.19E-03	0.00E+00
215	2	0	1	0	7	7	35	0.00000000E+00	-0.26592285E-08	-0.27E-08	0.27E-08
216	2	1	1	0	15	11	59	0.10000000E+01	0.99937529E+00	-0.62E-03	0.76E-08
217	2	1	2	0	9	9	45	-0.80000000E+00	-0.80000000E+00	-0.28E-13	0.89E-13
218	2	0	1	0	14	14	70	0.00000000E+00	0.00000000E+00	0.00E+00	0.67E-04
219	4	2	2	0	18	17	154	-0.10000000E+01	-0.10000000E+01	-0.13E-08	0.10E-08
221	2	0	1	0	25	25	125	-0.10000000E+01	-0.10000000E+01	0.00E+00	0.00E+00
222	2	0	1	0	7	6	31	-0.15000000E+01	-0.15000000E+01	-0.30E-12	0.34E-12
223	2	0	2	0	9	9	45	-0.83403245E+00	-0.83403245E+00	-0.17E-13	0.33E-12
224	2	0	4	0	4	4	20	-0.30400000E+03	-0.30400000E+03	0.49E-12	0.00E+00
225	2	0	5	0	7	7	35	0.20000000E+01	0.20000000E+01	0.27E-13	0.00E+00
226	2	0	2	0	7	7	35	-0.50000000E+00	-0.50000000E+00	-0.85E-10	0.85E-10
227	2	0	2	0	6	5	26	0.10000000E+01	0.10000000E+01	-0.55E-10	0.34E-10
228	2	0	2	0	7	7	35	-0.30000000E+01	-0.30000000E+01	-0.47E-09	0.84E-08
229	2	0	0	0	30	24	126	0.00000000E+00	0.20418966E-09	0.20E-09	0.00E+00
230	2	0	2	0	5	4	21	0.37500000E+00	0.37500000E+00	-0.12E-09	0.46E-10
231	2	0	2	0	47	35	187	0.00000000E+00	0.11785198E-07	0.12E-07	0.00E+00
232	2	0	3	0	5	5	25	-0.10000000E+01	-0.10000000E+01	-0.34E-11	0.28E-11
233	2	0	1	0	18	14	74	0.00000000E+00	0.47203178E-09	0.47E-09	0.00E+00
234	2	0	1	0	13	13	65	-0.80000000E+00	-0.79999992E+00	0.96E-07	0.00E+00
235	3	1	1	0	21	18	129	0.40000000E-01	0.39999999E-01	-0.35E-07	0.41E-07
236	2	0	2	0	6	6	30	-0.58903436E+02	-0.58903436E+02	-0.15E-09	0.00E+00
237	2	0	3	0	10	10	50	-0.58903436E+02	-0.58903436E+02	-0.15E-09	0.00E+00
238	2	0	3	0	14	13	66	-0.58903436E+02	-0.58903436E+02	-0.15E-09	0.00E+00
239	2	0	1	0	6	6	30	-0.58903436E+02	-0.58903436E+02	-0.15E-09	0.00E+00
240	3	0	0	0	5	3	23	0.00000000E+00	0.13985074E-13	0.14E-13	0.00E+00
241	3	0	0	0	34	25	184	0.00000000E+00	0.95883248E-08	0.96E-08	0.00E+00
242	3	0	0	0	11	11	77	0.00000000E+00	0.11959946E-10	0.12E-10	0.00E+00
243	3	0	0	0	11	7	53	0.79657853E+00	0.79657853E+00	0.56E-08	0.00E+00
244	3	0	0	0	18	17	120	0.00000000E+00	0.16968604E-09	0.17E-09	0.00E+00
245	3	0	0	0	20	19	134	0.00000000E+00	0.27020613E-07	0.27E-07	0.00E+00
246	3	0	0	0	26	19	140	0.00000000E+00	0.50593466E-07	0.51E-07	0.00E+00
247	3	0	0	0	5	2	17	0.00000000E+00	0.00000000E+00	0.00E+00	0.00E+00
248	3	1	2	0	16	13	94	-0.80000000E+00	-0.80000000E+00	-0.14E-10	0.46E-10
249	3	0	1	0	14	14	98	0.10000000E+01	0.10000000E+01	0.15E-07	0.00E+00
250	3	0	2	0	2	2	14	-0.33000000E+04	-0.33000000E+04	-0.55E-11	0.16E-09
251	3	0	1	0	8	8	56	-0.34560000E+04	-0.34560000E+04	0.25E-13	0.00E+00
252	3	1	1	0	39	25	189	0.40000000E-01	0.40000000E-01	0.83E-08	0.29E-04
253	3	0	1	0	6	6	42	0.69282032E+02	0.69282032E+02	0.44E-08	0.00E+00
254	3	2	2	0	9	9	63	-0.17320508E+01	-0.17320508E+01	-0.13E-15	0.15E-04
255	4	0	0	0	20	12	116	0.00000000E+00	0.99255199E-09	0.99E-09	0.00E+00

(continued)

TP	*N*	*ME*	*M*	*IFAIL*	*NF*	*NDF*	*NEF*	*FEX*	*F*	*DFX*	*DGX*
256	4	0	0	0	36	27	252	0.00000000E+00	0.53112636E-07	0.53E-07	0.00E+00
257	4	0	0	0	42	29	274	0.00000000E+00	0.37845098E-08	0.38E-08	0.00E+00
258	4	0	0	0	120	90	840	0.00000000E+00	0.11588756E-07	0.12E-07	0.00E+00
260	4	0	0	0	121	91	849	0.00000000E+00	0.24100045E-08	0.24E-08	0.00E+00
261	4	0	0	0	35	32	291	0.00000000E+00	0.58764398E-07	0.59E-07	0.00E+00
262	4	1	4	0	7	7	63	-0.10000000E+02	-0.10000000E+02	0.21E-12	0.14E-10
263	4	2	4	0	14	14	126	-0.10000000E+01	-0.10000000E+01	0.00E+00	0.13E-04
264	4	0	3	0	12	9	84	-0.44000000E+02	-0.44000000E+02	-0.55E-09	0.11E-07
266	5	0	0	0	8	7	78	0.99597447E+00	0.99597447E+00	-0.45E-08	0.00E+00
267	5	0	0	0	35	32	355	0.00000000E+00	0.15955254E-04	0.16E-04	0.00E+00
268	5	0	5	0	22	11	132	0.00000000E+00	0.12107648E-10	0.12E-10	0.00E+00
269	5	3	3	0	8	7	78	0.40930233E+01	0.40930233E+01	0.74E-08	0.84E-13
270	5	0	1	0	9	8	89	-0.10000000E+01	-0.10000000E+01	0.50E-11	0.17E-06
271	6	0	0	0	25	7	109	0.00000000E+00	0.25001599E-19	0.25E-19	0.00E+00
272	6	0	0	0	40	36	472	0.00000000E+00	0.56556526E-02	0.57E-02	0.00E+00
273	6	0	0	0	55	30	415	0.00000000E+00	0.32808299E-07	0.33E-07	0.00E+00
274	2	0	0	0	6	5	26	0.00000000E+00	0.24144027E-25	0.24E-25	0.00E+00
275	4	0	0	0	5	4	37	0.00000000E+00	0.13975558E-06	0.14E-06	0.00E+00
276	6	0	0	0	9	8	105	0.00000000E+00	0.13184431E-10	0.13E-10	0.00E+00
277	4	0	4	0	5	5	45	0.50761905E+01	0.50761930E+01	0.50E-06	0.00E+00
278	6	0	6	0	8	8	104	0.78385281E+01	0.78385284E+01	0.32E-07	0.14E-15
279	8	0	8	0	8	8	136	0.10605950E+02	0.10605953E+02	0.36E-06	0.00E+00
280	10	0	10	0	11	11	231	0.13375428E+02	0.13375428E+02	0.72E-08	0.44E-13
281	10	0	0	0	204	80	1804	0.00000000E+00	0.54124747E-04	0.54E-04	0.00E+00
282	10	0	0	0	177	125	2677	0.00000000E+00	0.19166854E-08	0.19E-08	0.00E+00
283	10	0	0	0	7	2	47	0.00000000E+00	0.21232321E-81	0.21E-81	0.00E+00
284	15	0	10	0	65	31	995	-0.18400000E+04	-0.18400000E+04	-0.45E-11	0.10E-07
285	15	0	10	0	45	20	645	-0.82520000E+04	-0.82520000E+04	-0.18E-12	0.25E-09
286	20	0	0	0	165	97	4045	0.00000000E+00	0.14971611E-04	0.15E-04	0.00E+00
287	20	0	0	0	27	27	1107	0.00000000E+00	0.14817404E-06	0.15E-06	0.00E+00
288	20	0	0	0	65	39	1625	0.00000000E+00	0.48370155E-06	0.48E-06	0.00E+00
289	30	0	0	0	8	7	428	0.00000000E+00	0.10087287E-09	0.10E-09	0.00E+00
290	2	0	0	0	19	17	87	0.00000000E+00	0.10788892E-06	0.11E-06	0.00E+00
291	10	0	0	0	69	41	889	0.00000000E+00	0.11976172E-04	0.12E-04	0.00E+00
292	30	0	0	0	195	73	4575	0.00000000E+00	0.91355897E-04	0.91E-04	0.00E+00
293	50	0	0	0	245	95	9745	0.00000000E+00	0.34198691E-05	0.34E-05	0.00E+00
294	6	0	0	0	72	67	876	0.00000000E+00	0.50711975E-07	0.51E-07	0.00E+00
295	10	0	0	0	114	107	2254	0.00000000E+00	0.37424249E-07	0.37E-07	0.00E+00
296	16	0	0	0	31	31	1023	0.00000000E+00	0.15280757E-02	0.15E-02	0.00E+00
297	30	0	0	0	33	33	2013	0.00000000E+00	0.29172836E-02	0.29E-02	0.00E+00
298	50	0	0	0	33	33	3333	0.00000000E+00	0.48981547E-02	0.49E-02	0.00E+00
299	100	0	0	0	33	33	6633	0.00000000E+00	0.98482202E-02	0.98E-02	0.00E+00
300	20	0	0	0	41	21	881	-0.20000000E+02	-0.20000000E+02	0.85E-14	0.00E+00
301	50	0	0	0	101	51	5201	-0.50000000E+02	-0.50000000E+02	0.51E-14	0.00E+00
302	100	0	0	0	201	101	20401	-0.10000000E+03	-0.10000000E+03	0.11E-12	0.00E+00
303	20	0	0	0	20	14	580	0.00000000E+00	0.30618142E-09	0.31E-09	0.00E+00
304	50	0	0	0	8	2	208	0.00000000E+00	0.31312251E-27	0.31E-27	0.00E+00
305	100	0	0	0	8	2	408	0.00000000E+00	0.49116799E-26	0.49E-26	0.00E+00
307	2	0	0	0	30	21	114	0.12436000E+03	0.12436218E+03	0.18E-04	0.00E+00
308	2	0	0	0	14	11	58	0.77319906E+00	0.77319906E+00	-0.45E-08	0.00E+00
309	2	0	0	0	14	10	54	-0.39871708E+01	-0.39871708E+01	-0.19E-08	0.00E+00
310	2	0	0	0	15	9	51	0.00000000E+00	0.29846170E-09	0.30E-09	0.00E+00

(continued)

TP	*N*	*ME*	*M*	*IFAIL*	*NF*	*NDF*	*NEF*	*FEX*	*F*	*DFX*	*DGX*
311	2	0	0	0	18	10	58	0.00000000E+00	0.33026215E-09	0.33E-09	0.00E+00
314	2	0	0	0	9	7	37	0.16904000E+00	0.16904271E+00	0.16E-04	0.00E+00
315	2	0	3	0	11	10	51	-0.80000000E+00	-0.80000000E+00	-0.48E-08	0.15E-07
316	2	1	1	0	7	6	31	0.33431458E+03	0.33431458E+03	-0.15E-07	0.00E+00
317	2	1	1	0	9	7	37	0.37246661E+03	0.37246661E+03	-0.11E-07	0.49E-12
318	2	1	1	0	10	8	42	0.41275005E+03	0.41275005E+03	0.97E-08	0.25E-14
319	2	1	1	0	12	9	48	0.45240440E+03	0.45240440E+03	-0.92E-08	0.13E-12
320	2	1	1	0	17	11	61	0.48553146E+03	0.48553146E+03	0.52E-08	0.14E-09
321	2	1	1	0	15	10	55	0.49611237E+03	0.49611237E+03	-0.84E-08	0.12E-09
322	2	1	1	0	21	14	77	0.49996001E+03	0.49996001E+03	0.40E-08	0.65E-11
323	2	0	2	0	9	7	37	0.37989446E+01	0.37989445E+01	-0.15E-07	0.31E-08
324	2	0	2	0	8	8	40	0.50000000E+01	0.50000000E+01	-0.11E-08	0.39E-07
325	2	1	3	0	8	7	36	0.37913414E+01	0.37913414E+01	0.13E-07	0.62E-11
326	2	0	2	0	8	7	36	-0.79807821E+02	-0.79807821E+02	0.19E-08	0.34E-11
328	2	0	0	0	24	17	92	0.17441520E+01	0.17441520E+01	0.35E-08	0.00E+00
329	2	0	3	0	7	7	35	-0.69618139E+04	-0.69618139E+04	0.35E-08	0.60E-13
330	2	0	1	0	10	10	50	0.16205833E+01	0.16205833E+01	0.14E-07	0.34E-09
331	2	0	1	0	9	7	37	0.42580000E+01	0.42583847E+01	0.90E-04	0.00E+00
332	2	0	2	0	707	93	1079	0.11495015E+03	0.11495059E+03	0.39E-05	0.00E+00
333	3	0	0	0	27	20	147	0.43200001E-01	0.43270406E-01	0.16E-02	0.00E+00
334	3	0	0	0	33	24	177	0.82148773E-02	0.82149184E-02	0.50E-05	0.00E+00
335	3	2	2	0	20	19	134	-0.44721370E-02	-0.44721406E-02	-0.80E-06	0.14E-07
336	3	2	2	0	23	16	119	-0.33789573E+00	-0.33789570E+00	0.83E-07	0.66E-09
337	3	0	1	0	11	8	59	0.60000000E+01	0.60000000E+01	-0.13E-09	0.15E-09
339	3	0	1	0	12	11	78	0.33616797E+01	0.33616797E+01	-0.22E-08	0.00E+00
340	3	0	1	0	8	8	56	-0.54000000E-01	-0.54000000E-01	0.45E-08	0.35E-13
341	3	0	1	0	13	12	85	-0.22627417E+02	-0.22627417E+02	-0.12E-09	0.69E-08
342	3	0	1	0	13	12	85	-0.22627417E+02	-0.22627417E+02	-0.12E-09	0.69E-08
343	3	0	2	0	5	5	35	-0.56847825E+01	-0.56847825E+01	0.13E-09	0.28E-12
344	3	1	1	0	10	9	64	0.32568200E-01	0.32568202E-01	0.72E-07	0.19E-06
345	3	1	1	0	29	20	149	0.32568200E-01	0.32568200E-01	0.15E-07	0.77E-08
346	3	0	2	0	5	5	35	-0.56847825E+01	-0.56847825E+01	0.13E-09	0.28E-12
347	3	1	1	0	2	2	14	0.17374625E+05	0.17374625E+05	0.15E-07	0.29E-11
348	3	1	1	0	36	22	168	0.36970840E+02	0.36970840E+02	-0.17E-08	0.91E-12
351	4	0	0	0	34	26	242	0.31857175E+03	0.31857175E+03	-0.38E-08	0.00E+00
352	4	0	0	0	14	7	70	0.90323433E+03	0.90323433E+03	0.20E-08	0.00E+00
353	4	1	3	0	2	2	18	-0.39933673E+02	-0.39933673E+02	-0.12E-07	0.43E-12
354	4	0	1	0	20	14	132	0.11378385E+00	0.11378386E+00	0.12E-06	0.00E+00
355	4	1	1	0	389	185	1869	0.69675463E+02	0.69675463E+02	0.12E-08	0.24E-07
356	4	0	5	0	10	10	90	0.23811648E+01	0.23811648E+01	-0.15E-07	0.42E-10
357	4	0	0	0	11	11	99	0.35845660E+00	0.35845711E+00	0.14E-05	0.00E+00
359	5	0	14	0	3	3	33	-0.52804168E+07	-0.52803351E+07	0.15E-04	0.18E-14
360	5	0	2	0	67	41	477	-0.52803351E+07	-0.52803351E+07	-0.22E-08	0.00E+00
361	5	0	6	0	4	4	44	-0.77641212E+06	-0.77641212E+06	0.61E-08	0.36E-11
364	6	0	4	0	30	26	342	0.60600199E-01	0.60600245E-01	0.76E-06	0.25E-05
365	7	0	5	0	15	15	225	0.23313708E+02	0.23313708E+02	0.21E-07	0.14E-08
366	7	0	14	0	17	16	241	0.70430560E+03	0.70430560E+03	0.16E-08	0.11E-13
367	7	2	5	0	16	12	184	-0.37412960E+02	-0.37412960E+02	0.12E-07	0.21E-10
368	8	0	0	0	9	9	153	-0.74997564E+00	-0.75000000E+00	-0.32E-04	0.00E+00
369	8	0	6	0	34	33	562	0.70492480E+04	0.70492480E+04	0.29E-08	0.99E-12
370	6	0	0	0	38	31	410	0.22876701E-02	0.22877124E-02	0.18E-04	0.00E+00
372	9	0	12	0	27	20	387	0.13390093E+05	0.13390093E+05	0.89E-08	0.80E-10

(continued)

TP	*N*	*ME*	*M*	*IFAIL*	*NF*	*NDF*	*NEF*	*FEX*	*F*	*DFX*	*DGX*
373	9	6	6	0	16	15	286	0.13390093E+05	0.13390093E+05	0.89E-08	0.28E-13
374	10	0	35	0	57	38	817	0.23326400E+00	0.23326347E+00	-0.23E-05	0.67E-08
375	10	9	9	0	46	25	546	-0.15161000E+02	-0.15161044E+02	-0.29E-05	0.68E-08
376	10	1	15	0	17	17	357	-0.44300879E+04	-0.44300879E+04	-0.76E-08	0.17E-12
377	10	3	3	0	2	2	42	-0.79500098E+03	-0.79498973E+03	0.14E-04	0.12E-08
378	10	3	3	0	51	51	1071	-0.47761091E+02	-0.47761091E+02	0.28E-08	0.26E-09
379	11	0	0	0	69	52	1213	0.40137700E-01	0.40137741E-01	0.10E-05	0.00E+00
380	12	0	3	0	316	202	5164	0.31682215E+01	0.31682997E+01	0.25E-04	0.00E+00
381	13	1	4	0	7	7	189	0.10149000E+01	0.10148975E+01	-0.24E-05	0.22E-11
382	13	1	4	0	12	11	298	0.10383100E+01	0.10383119E+01	0.18E-05	0.21E-07
383	14	1	1	0	123	54	1635	0.72856600E+06	0.72859365E+06	0.38E-04	0.22E-15
384	15	0	10	0	57	25	807	-0.83102590E+04	-0.83102590E+04	0.31E-08	0.91E-09
385	15	0	10	0	52	25	802	-0.83152859E+04	-0.83152859E+04	-0.53E-08	0.38E-09
386	15	0	11	0	62	32	1022	-0.81643688E+04	-0.81643688E+04	-0.52E-08	0.32E-08
387	15	0	11	0	64	33	1054	-0.82501417E+04	-0.82501417E+04	0.13E-08	0.87E-10
388	15	0	15	0	47	29	917	-0.58210842E+04	-0.58210842E+04	-0.42E-08	0.54E-08
389	15	0	15	0	41	24	761	-0.58097197E+04	-0.58097197E+04	-0.38E-08	0.95E-10
391	30	0	0	0	175	54	3415	0.00000000E+00	0.84925674E-08	0.85E-08	0.00E+00
392	30	0	45	0	15	15	915	-0.16960671E+07	-0.16960671E+07	0.18E-07	0.14E-13
393	48	2	3	0	90	89	8634	0.86337998E+00	0.86338003E+00	0.54E-07	0.11E-06
394	20	1	1	0	181	114	4741	0.19166667E+01	0.19166673E+01	0.29E-06	0.81E-08
395	50	1	1	0	473	227	23173	0.19166668E+01	0.19166667E+01	-0.51E-07	0.12E-07