8.3 Check Code for Computing Derivatives

A. Purpose

Subroutine DCKDER checks the mutual consistency of code for computing values of a (possibly vector valued) function with code for computing first derivatives (*e.g.*, gradient vector or Jacobian matrix) of the function. In particular this is expected to be useful for persons using the software of Chapters 8.2, 9.2, and 9.3.

B. Usage

B.1 Program Prototype, Double Precision

INTEGER MODE, M, N, LDFJAC, IMAX, JMAX

DOUBLE PRECISION X(≥N)**, FVEC**(≥M)**, FJAC**(LDFJAC,≥N)**, TSTMAX, TEST**(LDFJAC,≥N)

```
   Assign values to M, N, and LDFJAC.
   Set X() to the value of the vector x at which
      consistency is to be tested.
   Compute FJAC(,) as the M × N Jacobian matrix
      of first partial derivatives of FVEC with respect
      to x, evaluated at X().
   MODE = 1
10 continue
```

CALL DCKDER(MODE, M, N, X, FVEC, FJAC, LDFJAC, TEST, IMAX, JMAX,TSTMAX)

```
   if(MODE .eq. 2) then
      Compute FVEC() as an M-vector of function
         values evaluated at X().
      go to 10
   end if
   Here the process is completed. Results are in
   TEST(,), IMAX, JMAX, and TSTMAX.
```

B.2 Argument Definitions

MODE [inout] On initial entry set MODE = 1. DCKDER will return a number of times with MODE = 2. The calling code should compute FVEC() as a function of X() and call DCKDER again, not altering MODE. When DCKDER is finished, it returns with MODE = 3.

M [in] Number of terms in FVEC() and number of rows of data in FJAC(,).

N [in] Number of terms in X() and number of columns of data in FJAC(,).

X() [inout] Initially must contain the vector **x** around which the testing will be done. Contains perturbed **x** on each return with MODE = 2. On final return with MODE = 3, contains the original **x** exactly restored.

FVEC() [in] On entries with MODE = 2, the user stores function values in FVEC(), *i.e.*, FVEC(i) = f_i.

FJAC(,) [in] On the initial entry with MODE = 1, the user stores the Jacobian matrix in FJAC(,), *i.e.*, FJAC(i,j) = $\partial f_i/\partial x_j$.

LDFJAC [in] Declared first dimension of the arrays FJAC(,) and TEST(,). Require LDFJAC ≥ M.

TEST(,) [inout] Array with the same dimensions as FJAC(). On final return with MODE = 3, TEST(i,j) contains the consistency measure for FJAC(i,j) for all i = 1, ..., M, and j = 1, ..., N. This quantity is computed as the signed difference: FJAC(i,j) minus a central finite difference approximation to $\partial f_i/\partial x_j$. The user does not need to store anything in TEST(,) before the initial entry with MODE = 1. On intermediate returns with MODE = 2, TEST(,) contains saved intermediate quantities, and thus the user must not alter the contents of TEST(,) on these returns.

IMAX, JMAX, TSTMAX [inout] On final return with MODE = 3, these quantities are set so that

TSTMAX = *abs*(TEST(IMAX, JMAX)) = max{*abs*(TEST(i,j)) : i = 1, ..., M; j = 1, ..., N}. The user does not need to store anything in these variables before the initial entry with MODE = 1. On intermediate returns with MODE = 2, these variables contain saved intermediate quantities, and thus the user must not alter their contents on these returns.

B.3 Modifications for Single Precision

For single precision usage change the DOUBLE PRECISION statements to REAL and change the name DCKDER to SCKDER. It is recommended that one use the double precision rather than the single precision version of this package for better reliability, except possibly on computers such as the Cray Y/MP that have precision of about 14 decimal places in single precision.

C. Examples and Remarks

The program DRDCKDER illustrates the use of DCKDER. Results are shown in ODDCKDER. This example was run using double precision IEEE arithmetic which has precision of $\epsilon \approx 0.22 \times 10^{-15}$. If third derivatives have about the same magnitude as function values, and the relative error in function evaluations is about machine precision, then the magnitude of entries of TEST(,) should be about $\epsilon^{2/3}$ times the magnitude of the function values. For example in our sample case we have $|f_1| = 0.0646$ and so we would expect the magnitude of terms in the first row of TEST(,) to be about $(0.37 \times 10^{-10}) \times 0.0646 \approx 0.24 \times 10^{-11}$, with the stated assumptions on the size of third derivatives and the error in function evaluations. If the relative error in the computation of function values is larger than the machine precision, or the magnitudes of third derivatives are larger than the magnitudes of the function values, the values in TEST(,) will be larger as discussed in Section D, even when the code being tested is correct.

D. Functional Description

Let $\mathbf{x}$ denote the vector given initially in X(). Assume FJAC(,) contains the $m \times n$ Jacobian matrix of $\partial f_i/\partial x_j$ evaluated at $\mathbf{x}$. Let ϵ denote the machine precision and ω denote the underflow limit. These are given by D1MACH(4) and D1MACH(1) respectively (R1MACH() in single precision) (See Chapter 19.1). Define $\alpha = (3\epsilon)^{1/3}$ and $\sigma = \max(10^5\omega/\alpha,\ \epsilon^2)$. For each value of j from 1 to n, compute $h_j = \alpha x_j$ if $|x_j| > \sigma$, or $h_j = \alpha\sigma$ if $0 < |x_j| \le \sigma$, or $h_j = \alpha$ if $x_j = 0$. Let $\mathbf{e}_j$ denote the n-vector that is all zeros except for the j^{th} component which is one.

For each i and j compute

$$\text{TEST}(i,j) = \text{FJAC}(i,j) - \frac{f_i(\mathbf{x} + h_j\mathbf{e}_j) - f_i(\mathbf{x} - h_j\mathbf{e}_j)}{2h_j}$$

The error in this central difference approximation to the derivative $\partial f_i/\partial x_j$ is

$$h_j^2 M_3/6 + \delta/h_j$$

where M_3 denotes the magnitude of $\partial^3 f_i/\partial^3 x_j$ evaluated at some point on the line segment from $\mathbf{x} - h_j\mathbf{e}_j$ to $\mathbf{x} + h_j\mathbf{e}_j$, and δ is a bound on the error in computing f_i. The optimal step length to minimize this error estimate is

$$(3\delta/M_3)^{1/3}$$

If we assume $M_3 \approx |f_i|$ and $\delta \approx \epsilon|f_i|$, so $\delta/M_3 \approx \epsilon$, throughout the relevant interval, then the optimal step length is $(3\epsilon)^{1/3}$ and the error bound is $\frac{1}{2}(3\epsilon)^{2/3}|f_i| \approx 1.04\epsilon^{2/3}|f_i|$. These formulas, and useful insights into the computational use of finite differences, are given in Section 8.6 of [1].

References

1. Philip E. Gill, Walter Murray, and Margaret H. Wright, **Practical Optimization**, Academic Press, New York (1981) 401 pages. Sixth printing, 1987.

E. Error Procedures and Restrictions

Require MODE = 1 or 2 on any entry to DCKDER. If not, DCKDER will call the error message processor of Chapter 19.2 with an error level of 0 and return with MODE unchanged.

F. Supporting Information

The source language is ANSI Fortran 77.

Entry	**Required Files**
DCKDER	AMACH, DCKDER, ERFIN, ERMSG, IERM1, IERV1
SCKDER	AMACH, ERFIN, ERMSG, IERM1, IERV1, SCKDER

Designed and programmed by C. L. Lawson and F. T. Krogh, JPL, 1991.

DRDCKDER

```
c     program DrDCKDER
c>> 2007-01/02 DRDCKDER Krogh  Put commas around ':' in formats.
c>> 1996-06-28 DrDCKDER Krogh  Format changes for conversion to C.
c>> 1994-11-02 DrDCKDER Krogh  Changes to use M77CON
c>> 1992-04-15 DrDCKDER CLL
c>> 1992-01-13 C. L. Lawson, JPL.
c  DRDCKDER..  Demo driver for DCKDER.  Checks derivative calculation.
c     ------------------------------------------------------------------
c--D replaces "?": Dr?CKDER, ?CKDER, ?TRG11
c     ------------------------------------------------------------------
      integer I,IMAX, J, JMAX, M,N,LDFJAC,MODE, NMAX
      parameter(LDFJAC = 5, NMAX = 5)
      double precision FVEC(15),FJAC(LDFJAC,NMAX)
      double precision TEST(LDFJAC,NMAX), TSTMAX, X(NMAX)
      data M, N / LDFJAC, NMAX /
```

```
      data X / 0.13d0, 0.14d0, 0.15d0, 0.16d0, 0.17d0 /
c     ------------------------------------------------------------------
      print*,'Program DrDCKDER..  Demo driver for DCKDER.'
      call DTRG11(N, X, FVEC ,FJAC, 2)
      MODE = 1
   10 continue
      call DCKDER(MODE, M, N, X, FVEC, FJAC, LDFJAC,
     *            TEST, IMAX, JMAX, TSTMAX)
      if(MODE .eq. 2) then
         call DTRG11(N, X, FVEC ,FJAC, 1)
         go to 10
      endif
c
      call DTRG11(N, X, FVEC ,FJAC, 1)
      print'(/11x,''X(J) ='',5g11.3,:,/(17x,5g11.3))',(X(J),J=1,N)
      print'(/1x,''  I    FVEC(I)    ....................'',
     *   ''FJAC(I,J)......................''/)'
      do 20 I = 1,M
         print'(1x,i3,1x,g11.3,1x,5g11.3,:,/(17x,5g11.3))',
     *         I,FVEC(I),(FJAC(I,J),J=1,N)
   20 continue
c
      print'(/1x,''TEST(,):''/)'
      do 30 I = 1,M
         print'(1x,i3,13x,5g11.3,:,/(17x,5g11.3))',I,(TEST(I,J),J=1,N)
   30 continue
      print'(/1x,''IMAX ='',i3,'',    JMAX ='',i3,'',    TSTMAX ='',
     *   g11.3)', IMAX, JMAX, TSTMAX
      stop
      end
c     ==================================================================
      subroutine DTRG11(N, X, FVEC ,FJAC, IFLAG)
c     Trigonometric test case No. 11 from MINPACK test set developed by
c     J. J. More', B. S. Garbow, and K. E. Hillstrom, Argonne National
c     Laboratories, 1980.
c     ------------------------------------------------------------------
      integer I, IFLAG, J, N
      double precision FJAC(N,N), FVEC(N), SUM, TEMP, X(N)
c     ------------------------------------------------------------------
      if (IFLAG .eq. 1) then
c                                            Compute function vector.
         SUM = 0.0d0
         do 10 J = 1, N
            FVEC(J) = cos(X(J))
            SUM = SUM + FVEC(J)
   10    continue
         do 20 J = 1, N
            FVEC(J) = dble(N+J) - sin(X(J)) - SUM - dble(J)*FVEC(J)
   20    continue
      elseif (IFLAG .eq. 2) then
c                                            Compute Jacobian matrix.
         do 40 J = 1, N
            TEMP = sin(X(J))
            do 30 I = 1, N
               FJAC(I,J) = TEMP
   30       continue
            FJAC(J,J) = dble(J+1)*TEMP - cos(X(J))
   40    continue
      endif
```

```
return
end
```

ODDCKDER

```
Program DrDCKDER..   Demo driver for DCKDER.

          X(J) =   0.130       0.140       0.150       0.160       0.170

  I     FVEC(I)    .....................FJAC(I,J)........................

  1   -0.646E-01  -0.732       0.140       0.149       0.159       0.169
  2   -0.633E-01   0.130      -0.572       0.149       0.159       0.169
  3   -0.591E-01   0.130       0.140      -0.391       0.159       0.169
  4   -0.516E-01   0.130       0.140       0.149      -0.191       0.169
  5   -0.405E-01   0.130       0.140       0.149       0.159       0.295E-01

TEST(,):

  1                0.173E-09   0.185E-10   0.217E-09   0.693E-10   0.134E-10
  2                0.315E-09  -0.783E-10   0.217E-09   0.693E-10   0.134E-10
  3                0.315E-09   0.185E-10   0.273E-09   0.693E-10   0.134E-10
  4                0.315E-09   0.185E-10   0.217E-09   0.222E-09   0.134E-10
  5                0.315E-09   0.185E-10   0.217E-09   0.693E-10   0.128E-09

IMAX =   2,     JMAX =   1,     TSTMAX =   0.315E-09
```