

AD-A130 760

DOCUMENTATION FOR FCALC AND FDCORE(U) STANFORD UNIV CA
SYSTEMS OPTIMIZATION LAB P E GILL ET AL JUN 83
SOL-83-6 ARO-18424.10-MA N00014-75-C-0267

1/1

UNCLASSIFIED

F/G 9/2

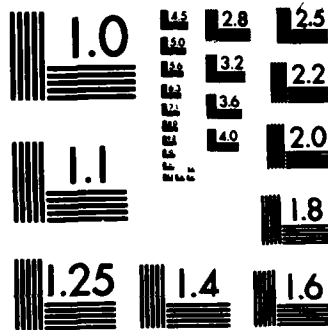
NL

END

FILMED

14

DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

ARD 18424.10-MA

(12)

AD A 130760



Systems
Optimization
Laboratory

DTIC FILE COPY

Department of Operations Research
Stanford University
Stanford, CA 94305

JUL 28 1983

A

This document has been approved
for public release and its
distribution is unlimited

83 07 27 00 8

**SYSTEMS OPTIMIZATION LABORATORY
DEPARTMENT OF OPERATIONS RESEARCH
STANFORD UNIVERSITY
STANFORD, CALIFORNIA 94305**

Documentation for FCALC and FDCORE[†]

by

**Philip E. Gill, Walter Murray,
Michael A. Saunders and Margaret H. Wright**

TECHNICAL REPORT SOL 83-6

June 1983

3 JUL 28 1983
A

[†]FCALC and FDCORE are available from the Office of Technology Licensing, 105 Encina Hall, Stanford University, Stanford, California, 94305.

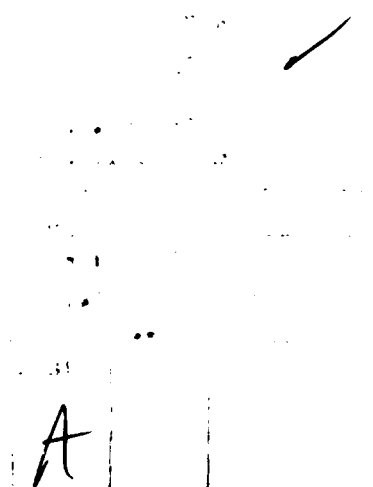
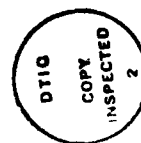
Research and reproduction of this report were partially supported by the National Science Foundation Grants MCS-7926009 and ECS-8012974; Department of Energy Contract DE-AM03-76SF00326, PA# DE-AT03-76ER72018; Office of Naval Research Contract N00014-75-C-0267; and Army Research Office Contract DAAG29-81-K-156.

Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the author(s) and do NOT necessarily reflect the views of the above sponsors.

Reproduction in whole or in part is permitted for any purposes of the United States Government. This document has been approved for public release and sale; its distribution is unlimited.

TABLE OF CONTENTS

1.	Purpose	1
2.	Description	2
3.	Specification of FDCALC	4
4.	Input Parameters of FDCALC	5
5.	Output Parameters of FDCALC	6
6*	Specification of FDCORE	7
7*	Input Parameters of FDCORE	8
8*	Input/Output Parameters of FDCORE	9
9.	Auxiliary Subprograms and Labelled Common	12
10.	Error Recovery	13
11.	Implementation Information	14
12.	Example Program and Output	16
13.	References	21



* Sections marked with an asterisk need not be read by those calling FDCALC directly.

Documentation for FDCALC and FDCORE†

**Philip E. Gill, Walter Murray,
Michael A. Saunders and Margaret H. Wright
Systems Optimization Laboratory
Department of Operations Research
Stanford University
Stanford, California 94305**

**Version 2.1
June 1983**

ABSTRACT

This report documents two Fortran subroutines -- FDCALC and FDCORE -- that compute a set of forward-difference intervals to be used in minimizing a smooth function. The primary subroutine FDCORE, which is called repeatedly by FDCALC, produces a suitable interval for a univariate function, as well as approximations to the first and second derivatives.

†FDCALC and FDCORE are available from the Office of Technology Licensing, 105 Encina Hall, Stanford University, Stanford, California, 94305.

The material contained in this report is based upon research supported by the U.S. Department of Energy Contract DE-AC03-76SF00326, PA No. DE-AT03-76ER72018; National Science Foundation Grants MCS-7926009 and ECS-8012974; the Office of Naval Research Grant N00014-75-C-0267; and the U.S. Army Research Office Contract DAAG29-79-C-0110.

1. PURPOSE

When minimizing a smooth function whose derivatives are not available, it is common to use finite-difference approximations in a gradient-based method. However, certain "standard" choices for the finite-difference intervals may produce unnecessarily inaccurate results. The subroutines documented in this report are an implementation of the method of Gill et al. (1983), which is designed to compute "good" intervals for forward-difference approximations to the gradient.

Our implementation is based on a reverse-communication control structure. The "core" subroutine FDCORE (which treats a single variable) must be called repeatedly by an outer subroutine in order to obtain a set of intervals for multivariate optimization. The outer subroutine FDCALC described in this report should be suitable for many applications. If it is not, the user may develop an outer subroutine that caters for the special needs of his problem.

The method used by FDCORE requires at least three evaluations of the function for each component of the gradient, even for a well scaled problem. An optimization algorithm that required this number of function evaluations at every iteration would be uncompetitive with alternative non-derivative methods. Fortunately, in practice several factors make it possible to obtain adequate gradient approximations with only one function evaluation per component. First and most important, it is our experience that, for many functions, the finite-difference intervals generated by the method of FDCORE do not vary significantly from one iteration to the next. Second, these intervals do not usually differ widely from the "optimal" intervals. Finally, finite-difference gradient methods can generally make satisfactory progress as long as the overall gradient vector has a reasonable level of accuracy; it is not essential for each component of the gradient to have close-to-maximal accuracy at every iterate.

Based on these observations, we suggest that FDCALC (or its equivalent) should be executed at a "typical" point (usually, the initial point of the minimization). The set of intervals obtained should then be used to compute forward-difference approximations at subsequent iterates.

We emphasize that these subroutines are *not* intended to compute the most accurate possible estimate of the gradient at a single point. The general problem of finding approximate derivatives by finite differences is known as *numerical differentiation*. A recent discussion of methods for numerical differentiation is given by Lyness (1977) (for other references, see Gill, Murray and Wright, 1981). Many automatic differentiation routines require a significant number of function evaluations — often, at least ten per derivative — and hence are not appropriate within a finite-difference gradient method.

2. DESCRIPTION

We shall briefly summarize the procedure of FDCORE applied to a *univariate* function f at the point x . (In order to obtain a set of intervals for the function $F(x)$, where x is an n -vector, the procedure is applied to each component of x , keeping the other components fixed.) Roughly speaking, the method is based on the fact that the bound on the relative truncation error in the forward-difference approximation tends to be an increasing function of h , while the relative condition error bound is generally a decreasing function of h .

The "best" interval h_r is given by

$$h_r = 2\sqrt{\frac{\epsilon_A}{|\Phi|}}, \quad (1)$$

where Φ is an estimate of $f''(x)$, and ϵ_A is an estimate of a good bound on the *absolute error* associated with computing the function (for a discussion of ϵ_A , see Chapter 8 of Gill, Murray and Wright, 1981). Given an interval h , Φ is defined by the second-order difference approximation

$$\Phi = \frac{f(x+h) - 2f(x) + f(x-h)}{h^2}.$$

The decision as to whether a given value of Φ is acceptable involves $\hat{C}(\Phi)$, the following bound on the *relative condition error* in Φ :

$$\hat{C}(\Phi) \equiv \frac{4\epsilon_A}{h^2|\Phi|}. \quad (2)$$

(When Φ is zero, $\hat{C}(\Phi)$ is taken as an arbitrarily large number.)

The procedure selects the interval h_* (to be used in computing Φ) from a sequence of trial intervals $\{h_k\}$. The initial trial interval is taken as $10\bar{h}$, where

$$\bar{h} = 2(\eta + |x|)\sqrt{\frac{\epsilon_A}{(\omega + |f|)}}. \quad (3)$$

(The quantities η and ω are discussed later.) The value of $\hat{C}(\Phi)$ for a trial value h_k is defined as "acceptable" if it lies in the interval $[.001, .1]$. In this case, h_* is taken as h_k , and the current value of Φ is used to compute h_r from (1). If $\hat{C}(\Phi)$ is unacceptable, the next trial interval is chosen so that the relative condition error bound will either decrease or increase, as required. If the bound on the relative condition error is too large, a *larger* interval is used as the next trial value in an attempt to reduce the condition error bound. On the other hand, if the relative condition error bound is too small, h_k is reduced.

The procedure will fail to produce an acceptable value of $\hat{C}(\Phi)$ in two situations. Firstly, if $f''(x)$ is extremely small, then $\hat{C}(\Phi)$ may never become small, even for a very large value of the

2. DESCRIPTION

FDCORE/3

interval. Alternatively, $\hat{C}(\Phi)$ may never exceed .001, even for a very small value of the interval. This usually implies that $f''(x)$ is extremely large, and occurs most often near a singularity.

As a check on the validity of the estimated first derivative, the procedure provides a comparison of the forward-difference approximation computed with h_r and the central-difference approximation computed with h_d . If these values do not display some agreement, neither can be considered reliable.

The parameters η and ω in (3) are local variables ETA and OMEGA of subroutine FDCORE; they are both set to 1 with DATA statements, and may be changed if appropriate (see Section 10). Other aspects of the algorithm may also be adjusted by changing the values of local variables in FDCORE. In particular, KMAX defines the maximum number of trial intervals, BNDLO and BNDUP define the range of acceptable values for $\hat{C}(\Phi)$, and RHO defines the factor by which h_k is changed at each iteration.

3. SPECIFICATION OF FDCALC

SUBROUTINE FDCALC(FUN, MSGVLV, N, EPSA, X,
NUMF, NUMOK, INFO,
FX, GRAD, HCNTRL, HESSD, HFORW)

EXTERNAL	FUN
INTEGER	MSGVLV, N, NUMF, NUMOK
INTEGER	INFO(N)
REAL	EPSA, FX
REAL	X(N), GRAD(N), HCNTRL(N), HESSD(N), HFORW(N)

(The specification of a parameter as REAL should be interpreted as *working precision*, which may be DOUBLE PRECISION in some circumstances.)

4. INPUT PARAMETERS OF FDCALC

FUN is a user-provided subroutine that must define the function for which the gradient is to be approximated. **FUN** must be declared as **EXTERNAL** in the routine that calls **FDCALC**. The specification of **FUN** is:

```
SUBROUTINE FUN ( N, X, FX )  
  INTEGER      N  
  REAL         FX  
  REAL         X(N)
```

where

N is the number of variables;

FX should be set to the value of the function evaluated at **X**;

X is the vector of **N** variables at which the function is to be evaluated.

MSGLVL determines the level of printout from **FDCALC**. If **MSGLVL** = 0, there is no printout from **FDCALC**; if **MSGLVL** = 1, a summary is printed of the results for each variable. If **MSGLVL** = 2, a full debug printout is produced by **FDCALC** and **FDCORE**.

N is the number of variables (**N** must be positive).

EPSA is a positive number that should be a good bound on the *absolute error* associated with computing the function F at x . In general, **EPSA** should not be less than $\epsilon_M(1 + |F(x)|)$, where ϵ_M is the machine precision. A more detailed discussion of **EPSA**, and of methods for computing it, is given in Chapter 8 of Gill, Murray and Wright (1981).

X is an array of length **N** that contains the vector of variables at which the set of intervals should be computed.

5. OUTPUT PARAMETERS OF FDCALC

- NUMF** gives the total number of function evaluations required to compute the final set of intervals.
- NUMOK** gives the number of variables for which the intervals were acceptable. If $\text{NUMOK} = N$, satisfactory intervals were obtained for all the variables.
- INFO** is an integer array of length N whose j -th component is set to the final value of **INFORM** from **FDCORE** for the j -th variable. The meaning of each value is explained under **INFORM** in Section 8.
- FX** is the value of the function evaluated at the input vector X .
- GRAD** is an array of length N whose j -th component contains the best estimate of the first partial derivative for the j -th variable.
- HCNTRL** is an array of length N whose j -th component is the best interval found for computing a central-difference approximation to the partial derivative for the j -th variable.
- HESSD** is an array of length N whose j -th component is the estimate of the second partial derivative with respect to the j -th variable (i.e., the j -th diagonal element of the Hessian matrix $\nabla^2 F(x)$).
- HFORW** is an array of length N whose j -th component is the best interval found for computing a forward-difference approximation to the partial derivative for the j -th variable.

6. SPECIFICATION OF FDCORE

SUBROUTINE FDCORE(DEBUG, EPSA, FX, X,
IENTRY, INFORM, ITER,
CDEST, CDSAVE, ERRBND, FBACK, FDOPT,
FDSAVE, FFORW, H, HOPT, HPHI, HSAVE,
OLDH, SDEST, SDSAVE)

LOGICAL	DEBUG
INTEGER	IENTRY, INFORM, ITER
REAL	EPSA, FX, X, CDEST, CDSAVE, ERRBND, FBACK, FDOPT, FDSAVE, FFORW, H, HOPT, HPHI, HSAVE, OLDH, SDEST, SDSAVE

(The specification of a parameter as REAL should be interpreted as *working precision*, which may be DOUBLE PRECISION in some circumstances.)

* This section need not be read by those calling FDCALC directly.

7* INPUT PARAMETERS OF FDCORE

- DEBUG** is a logical variable that should be set to **.TRUE.** if a detailed printout is desired. If **DEBUG** is **.FALSE.**, there is no printout from **FDCORE**.
- EPSA** is a positive number that should be a good bound on the absolute error associated with computing the function at **X**. In general, **EPSA** should not be less than $\epsilon_M(1 + |f(x)|)$, where ϵ_M is the machine precision. A more detailed description of **EPSA**, and techniques for computing it, are given in Chapter 8 of Gill, Murray and Wright (1981). *The value of EPSA must not be changed until after the final exit from FDCORE for a given point.*
- FX** must be set to the value of the function at the point where the derivative is to be approximated. *The value of FX must not be altered until after the final exit from FDCORE for a given point.*
- X** must be set to the value of x (the point where the derivative is to be approximated). *The value of X must not be altered until after the final exit from FDCORE for a given variable.*

* This section need not be read by those calling **FDCALC** directly.

8. INPUT/OUTPUT PARAMETERS OF FDCORE

Many of the following parameters are included in the calling sequence of FDCORE simply to allow communication between iterations. For such a parameter, the description below includes only the meaning of the parameter on exit from FDCORE.

IENTRY controls the logic of FDCORE. On entry to FDCORE, IENTRY designates the portion of FDCORE to be executed. IENTRY *must be set to zero before the first call of FDCORE for each variable*. Thereafter, IENTRY should not be altered by the calling routine. On exit from FDCORE, IENTRY indicates the action to be taken by the calling routine, and thus the value of IENTRY should be tested after each call of FDCORE. If IENTRY = 5 on exit from FDCORE, the procedure has terminated (and the parameter INFORM described below indicates the reason for termination). If $\text{IENTRY} \leq 4$, FDCORE should be called again. If IENTRY = 4, a new value must be assigned to FFORW before calling FDCORE. If $\text{IENTRY} < 4$, new values must be assigned to FFORW and FBACK before calling FDCORE.

INFORM indicates the final result of FDCORE, as follows.

Value	Definition
0	The algorithm terminated successfully. The forward-difference estimate FDOPT and the central-difference estimate computed with HPHI agree to at least half a decimal digit.
1	The function appears to be constant. The variable HOPT is set to the value $\bar{h}$ (equation (3)) corresponding to a well scaled problem, and HPHI is set to $10\bar{h}$; FDOPT, SDEST and ERBND are set to zero.
2	The function appears to be linear or odd. The variables HOPT and HPHI are set to the smallest interval with acceptable bounds on the relative condition error in the forward- and backward-difference estimates. The variable FDOPT is set to the corresponding forward-difference estimate, and SDEST is set to zero.
3	The second derivative of the function appears to be so large that it cannot be reliably estimated. The variables HOPT and HPHI are set to the smallest trial interval; FDOPT and SDEST are set to the corresponding finite-difference estimates.

* This section need not be read by those calling FDCALC directly.

- 4 The algorithm terminated with an apparently acceptable estimate of the second derivative. However, the forward-difference estimate with HOPT and the central-difference estimate with HPHI do not agree to half a decimal place. This value of INFORM usually occurs when the first derivative is small. In this case, although the derivative estimates will probably have poor *relative* accuracy, the *interval* is still likely to be acceptable.

- ITER** is the count of the iteration within FDCORE.
- CDEST** is usually the value of the central-difference derivative estimate with the input value of H. When IENTRY = 4 on entry to FDCORE, the value of CDEST is not altered.
- CDSAVE** is the central-difference estimate computed with HSAVE if HSAVE is positive. Otherwise, CDSAVE is undefined.
- ERRBND** is a bound on the estimated error in the final forward-difference approximation. (When INFORM = 1, ERRBND is set to zero.)
- FBACK** must be defined by the calling routine as follows. If IENTRY < 4 on exit from FDCORE, FBACK must be set to the value of $f(X - H)$ before the next call of FDCORE. FBACK need not be defined by the calling routine when IENTRY = 0 on entry to FDCORE, or when IENTRY = 4 or 5 on exit from FDCORE.
- FDOPT** is the "best" forward-difference estimate of the first derivative after the final exit from FDCORE. If INFORM = 0 or 4, FDOPT is the forward-difference estimate computed with HOPT. Otherwise, FDOPT is defined as described above under INFORM.
- FDSAVE** is the value of the forward-difference approximation computed with HSAVE when HSAVE is positive. Otherwise, FDSAVE is undefined.
- FFORW** must be defined by the calling routine as follows. If IENTRY < 5 on exit from FDCORE, FFORW must be set to the value of $f(X + H)$ before the next call of FDCORE. FFORW need not be defined by the calling routine when IENTRY = 0 on entry to FDCORE, or when IENTRY = 5 on exit from FDCORE.
- H** is the current value of the difference interval. On entry to FDCORE (after the first call), H is the interval for which the new function values have been evaluated. On exit from FDCORE, H is the interval for which new function values should be computed.
- HOPT** is the final estimate of the "optimal" forward-difference interval when FDCORE terminates with INFORM = 0 or 4. The values assigned to HOPT for other values of INFORM are described above under INFORM.

8. INPUT/OUTPUT PARAMETERS OF FDCORE

FDCORE/11

- HPHI** is the interval used to compute the final estimate of the second derivative when FDCORE terminates with **INFORM** = 0 or 4. The values assigned to **HPHI** for other values of **INFORM** are described above under **INFORM**.
- HSAVE** is set to -1 in FDCORE when **IENTRY** = 0. Thereafter, a positive value of **HSAVE** on exit from FDCORE is the smallest interval for which the bounds on the relative condition error in the forward- and backward-difference approximations are acceptable.
- OLDH** is the value of **H** from the previous call of FDCORE.
- SDEST** is the best estimate of the second derivative (the one used to compute **HOPT**) when **INFORM** = 0 or 4. The values assigned to **SDEST** for other values of **INFORM** are described above under **INFORM**.
- SDSAVE** is the estimate of the second derivative computed with **HSAVE** if **HSAVE** is positive on exit from FDCORE.

9. AUXILIARY SUBPROGRAMS AND LABELLED COMMON

The subroutine **FDCORE** requires two auxiliary subprograms **MCHPAR** and **QUOTNT**. The subroutine **MCHPAR** sets the machine-dependent parameters in the **WMACH** array, which is stored in the labelled **COMMON** block **SOLMCH** (see Section 11). **QUOTNT** is a function subprogram that computes the quotient of two real numbers, safeguarded against overflow.

The subroutines **FDCORE** and **FDALC** use the labelled **COMMON** block **SOLMCH**, whose contents are defined in Section 11.

10. ERROR RECOVERY

Whenever the value of **INFORM** from **FDCORE** is non-zero for no apparent reason, the user should re-run **FDCORE** with the parameter **DEBUG** set to **.TRUE.**, in order to obtain a detailed printout of each iteration within **FDCORE**. The main quantities of interest are **FDCERR**, the estimated condition error in the forward-difference approximation, and **SDCERR**, the estimated condition error (2) in the second-derivative estimate.

Termination**Recommended Action****INFORM = 1**

This value occurs when the estimated relative condition error in the first derivative approximation is unacceptably large for every value of the finite-difference interval. If this happens when the function is not constant, the initial interval may be too small; in this case, the constants **ETA** and **OMEGA** in (3) may be changed in the **DATA** statements of **FDCORE**. This error may also occur if the function evaluation includes an inordinately large constant term, or if **EPSA** is too large.

INFORM = 2

In this case, the estimated relative condition error in the second derivative approximation remained large for every trial interval, but the estimated error in the first derivative approximation was acceptable for at least one interval. This usually means that the function is linear or odd. If it is not, the user should check whether the estimated relative condition error in the second derivative (**SDCERR** in the printout) appears to be decreasing as the trial intervals increase. If so, it may be worthwhile to alter the **ETA** and **OMEGA** in (3) so that the initial trial interval is larger, or to allow more iterations by increasing **KMAX** (see Section 2).

INFORM = 3

This value occurs when the relative condition error estimate in the second derivative remained very small for every trial interval. This usually occurs when the second derivative is extremely large (e.g., near a singularity). If this is not the case, the user should check whether the value of **SDCERR** in the debug printout is increasing as the trial intervals decrease. If so, it may be worthwhile to alter **ETA** and **OMEGA** so that the initial interval is smaller, or to increase **KMAX** (see Section 2). This error may also occur when the given value of **EPSA** is not a good estimate of a bound on the absolute error in the function (i.e., **EPSA** is too small).

INFORM = 4

The usual reason that the forward- and central-difference estimates fail to agree is that the first derivative is small. If the first derivative is not small, it may be helpful to execute the procedure at a different point.

11. IMPLEMENTATION INFORMATION

Both FDCORE and FDCALC have been written in ANSI (1966) Fortran and tested on an IBM 3081 computer using the WATFIV Compiler (Version 1, Level 6). All subroutines are PFORT-compatible (Ryder, 1974). Before the first call of FDCALC or FDCORE, the subprogram MCHPAR must be called to assign various machine-dependent parameters. These parameters are stored in the array WMACH(15), which is stored in the labelled COMMON block SOLMCH.

The version of MCHPAR provided by the Systems Optimization Laboratory contains the parameters associated with double precision on a machine in the IBM 370 series (a listing of the IBM version of MCHPAR is given in Section 12). *The user must substitute a version of MCHPAR that is appropriate for the machine to be used.* The specification of MCHPAR is

```
SUBROUTINE MCHPAR
  REAL          WMACH
  COMMON  /SOLMCH/ WMACH(15)
```

The first eleven components of the REAL array WMACH must be set in MCHPAR. The components of WMACH are defined as follows.

Value	Definition
WMACH(1)	is NBASE, the base of floating-point arithmetic.
WMACH(2)	is NDIGIT, the number of NBASE digits of precision.
WMACH(3)	is EPSMCH, the floating-point precision.
WMACH(4)	is RTEPS, the square root of EPSMCH.
WMACH(5)	is FLMIN, the smallest positive floating-point number.
WMACH(6)	is RTMIN, the square root of FLMIN.
WMACH(7)	is FLMAX, the largest positive floating-point number.
WMACH(8)	is RTMAX, the square root of FLMAX.
WMACH(9)	is UNDFLW, which specifies whether or not underflow is checked for in certain computations (not relevant to FDCORE).
WMACH(10)	is NIN, the file number for the input stream.
WMACH(11)	is NOUT, the file number for the output stream.

The values of NBASE, NDIGIT, EPSMCH, FLMIN and FLMAX for several machines are given in the following table, for both single and double precision; RTEPS, RTMIN and RTMAX may be computed using Fortran statements. The values NIN and NOUT depend on the machine installation.

For each precision, we give two values for EPSMCH, FLMIN and FLMAX. The first value is a Fortran decimal approximation of the exact quantity; use of this value in MCHPAR should cause no difficulty except in extreme circumstances. The second value is the exact mathematical representation.

Table of machine-dependent parameters

Variable	IBM 360/370 Single	CDC 6000/7000 Single	DEC 10/20 Single	Univac 1100 Single	DEC VAX Single
NBASE	16	2	2	2	2
NDIGIT	6	48	27	27	24
EPSMCH	9.54E-7 16^{-5}	7.11E-15 2^{-47}	7.46E-9 2^{-27}	1.50E-8 2^{-26}	1.20E-7 2^{-23}
FLMIN	1.0E-78 16^{-65}	1.0E-293 2^{-975}	1.0E-38 2^{-129}	1.0E-38 2^{-129}	1.0E-38 2^{-128}
FLMAX	1.0E+75 $16^{63}(1-16^{-6})$	1.0E+322 $2^{1070}(1-2^{-48})$	1.0E+38 $2^{127}(1-2^{-27})$	1.0E+38 $2^{127}(1-2^{-27})$	1.0E+38 $2^{127}(1-2^{-24})$

Variable	IBM 360/370 Double	CDC 6000/7000 Double	DEC 10/20 Double	Univac 1100 Double	DEC VAX Double
NBASE	16	2	2	2	2
NDIGIT	14	96	62	61	56
EPSMCH	2.22D-16 16^{-13}	2.53D-29 2^{-95}	2.17D-19 2^{-62}	8.68D-19 2^{-60}	2.78D-17 2^{-55}
FLMIN	1.0D-78 16^{-65}	1.0D-293 2^{-975}	1.0D-38 2^{-129}	1.0D-308 2^{-1025}	1.0D-38 2^{-128}
FLMAX	1.0D+75 $16^{63}(1-16^{-14})$	1.0D+322 $2^{1070}(1-2^{-96})$	1.0D+38 $2^{127}(1-2^{-62})$	1.0D+307 $2^{1023}(1-2^{-61})$	1.0D+38 $2^{127}(1-2^{-56})$

12. EXAMPLE PROGRAM AND OUTPUT

This section contains a listing and the computed results from a sample main program that calls FDCALC to obtain a set of intervals for the following four-variable function

$$F(x) = \sum_{i=1}^4 f_i(x_i),$$

where

$$\begin{aligned} f_1(t) &= 2t^3 + 4t; \\ f_2(t) &= \exp(10t); \\ f_3(t) &= 10^{-4}t^3 + t; \\ f_4(t) &= 2t^3 - 2.5t^2 - t. \end{aligned}$$

(Note that F is the sum of univariate functions.) The analytic first and second derivatives of the functions $\{f_i\}$ are:

$$\begin{aligned} f_1' &= 6t^2 + 4, & f_1'' &= 12t; \\ f_2' &= 10 \exp(10t), & f_2'' &= 100 \exp(10t); \\ f_3' &= .0002t + 1, & f_3'' &= .0002; \\ f_4' &= 6t^2 - 5t - 1, & f_4'' &= 12t - 5. \end{aligned}$$

The point at which derivatives are to be estimated is $\bar{x} = (1, .25, 10, 1 + \sqrt{\epsilon_M})^T$, where ϵ_M is the machine precision. The corresponding values of F and $\{f_i\}$ (to fifteen figures) are

$$\begin{aligned} F(\bar{x}) &= 26.8924939607035; \\ f_1(x_1) &= 6.0; & f_2(x_2) &= 12.1824939607035; \\ f_3(x_3) &= 10.01; & f_4(x_4) &= -1.5. \end{aligned}$$

The exact gradient at this point (to 15 figures) is

$$\nabla F(\bar{x}) = (10, \quad 121.82493960703, \quad 1.002, \quad 7\sqrt{\epsilon_M} + 6\epsilon_M)^T,$$

and the exact Hessian diagonals are

$$12, \quad 1218.2493960703, \quad 2 \times 10^{-4}, \quad \text{and} \quad 7 + 12\sqrt{\epsilon_M}.$$

The computed results were obtained on an IBM 3081 computer with double precision arithmetic ($\epsilon_M \approx 2.2 \times 10^{-16}$).

12. EXAMPLE PROGRAM AND OUTPUT

FDCORE/17

```

1 C *****
2 C SAMPLE MAIN PROGRAM FOR TESTING SUBROUTINES FDCALC AND FDCORE
3 C WITH 4-VARIABLE EXAMPLE.
4 C
5 C DOUBLE PRECISION VERSION 2.1. JUNE 1983.
6 C
7 C SYSTEMS OPTIMIZATION LABORATORY, DEPARTMENT OF OPERATIONS RESEARCH,
8 C STANFORD UNIVERSITY, STANFORD, CALIFORNIA 94305.
9 C
10 C *****
11 C
12 C     INTEGER          MSGVLV, N, NUNF, NUNOK
13 C     INTEGER          INFO(4)
14 C     DOUBLE PRECISION EPSA, EPSMCH, FX
15 C     DOUBLE PRECISION GRAD(4), HCNTRL(4), HESSD(4), HFORM(4), X(4)
16 C
17 C THE ARRAY WMACH(15) IN THE LABELLED COMMON AREA SOLMCH CONTAINS
18 C MACHINE-DEPENDENT QUANTITIES NEEDED BY FDCALC AND FDCORE.
19 C THE WMACH ARRAY IS INITIALIZED BY CALLING THE SUBROUTINE MCHPAR.
20 C
21 C     DOUBLE PRECISION WMACH
22 C     COMMON /SOLMCH/ WMACH(15)
23 C     EXTERNAL          FUN
24 C
25 C     DOUBLE PRECISION DABS, DSQRT
26 C
27 C THE SUBROUTINE MCHPAR INITIALIZES THE WMACH ARRAY IN THE
28 C LABELLED COMMON AREA SOLMCH.
29 C
30 C     CALL MCHPAR
31 C     EPSMCH = WMACH(3)
32 C
33 C     N      = 4
34 C
35 C MSGVLV = 1 WILL GIVE A SUMMARY PRINTOUT FOR EACH VARIABLE.
36 C
37 C     MSGVLV = 1
38 C
39 C SET THE POINT AT WHICH THE DERIVATIVES ARE TO BE ESTIMATED.
40 C
41 C     X(1) = 1.00+0
42 C     X(2) = .250+0
43 C     X(3) = 10.00+0
44 C     X(4) = 1.00+0 + DSQRT(EPSMCH)
45 C
46 C COMPUTE EPSA, A BOUND ON THE ABSOLUTE PRECISION OF THE FUNCTION.
47 C
48 C     CALL FUN ( N, X, FX )
49 C     EPSA = 10.00+0*EPSMCH*(1.00+0 + DABS(FX))
50 C
51 C     CALL FDCALC( FUN, MSGVLV, N, EPSA, X,
52 C     *          NUNF, NUNOK, INFO,
53 C     *          FX, GRAD, HCNTRL, HESSD, HFORM )
54 C
55 C     STOP
56 C
57 C
58 C END OF MAIN.
59 C     END

```

```

60      SUBROUTINE FUN  ( N, X, FX )
61      INTEGER          N
62      DOUBLE PRECISION  FX
63      DOUBLE PRECISION  X(N)
64 C
65 C *****
66 C
67 C SAMPLE 4-VARIABLE FUNCTION ( A SUM OF UNIVARIATE FUNCTIONS) FOR
68 C TESTING FDCORE AND FDCALC.
69 C
70 C *****
71 C
72      INTEGER          I
73      DOUBLE PRECISION  FCMP(4)
74      DOUBLE PRECISION  DEXP
75 C
76      FCMP(1) = 2.00+0*X(1)**3 + 4.00+0*X(1)
77      FCMP(2) = DEXP(10.00+0*X(2))
78      FCMP(3) = X(3) + 1.00-4*X(3)**2
79      FCMP(4) = 2.00+0*X(4)**3 - 2.50+0*X(4)**2 - X(4)
80 C
81      FX = 0.00+0
82      DO 100 I = 1, N
83          FX = FX + FCMP(I)
84 100 CONTINUE
85 C
86      RETURN
87 C
88 C END OF FUN
89      END

```


12. EXAMPLE PROGRAM AND OUTPUT

FDCORE/19

```

605      SUBROUTINE MCHPAR
606 C
607      DOUBLE PRECISION WMACH
608      COMMON /SOLMCH/ WMACH(15)
609 C
610 C      MCHPAR MUST DEFINE THE RELEVANT MACHINE PARAMETERS AS FOLLOWS.
611 C      WMACH(1) = NBASE = BASE OF FLOATING-POINT ARITHMETIC.
612 C      WMACH(2) = NDIGIT = NO. OF BASE WMACH(1) DIGITS OF PRECISION.
613 C      WMACH(3) = EPSMCH = FLOATING-POINT PRECISION.
614 C      WMACH(4) = RTEPS = SQRT(EPSMCH).
615 C      WMACH(5) = FLMIN = SMALLEST POSITIVE FLOATING-POINT NUMBER.
616 C      WMACH(6) = RTMIN = SQRT(FLMIN).
617 C      WMACH(7) = FLMAX = LARGEST POSITIVE FLOATING-POINT NUMBER.
618 C      WMACH(8) = RTMAX = SQRT(FLMAX).
619 C      WMACH(9) = UNDFLW = 0.0 IF UNDERFLOW IS NOT FATAL, +VE OTHERWISE.
620 C      WMACH(10) = NIN = STANDARD FILE NUMBER OF THE INPUT STREAM.
621 C      WMACH(11) = NOUT = STANDARD FILE NUMBER OF THE OUTPUT STREAM.
622 C
623      INTEGER          NBASE, NDIGIT, NIN, NOUT
624      DOUBLE PRECISION DSQRT
625 C
626      NBASE = 16
627      NDIGIT = 14
628      WMACH(1) = NBASE
629      WMACH(2) = NDIGIT
630      WMACH(3) = WMACH(1)**(1 - NDIGIT)
631      WMACH(4) = DSQRT(WMACH(3))
632      WMACH(5) = WMACH(1)**(-62)
633      WMACH(6) = DSQRT(WMACH(5))
634      WMACH(7) = WMACH(1)**61
635      WMACH(8) = DSQRT(WMACH(7))
636      WMACH(9) = 0.00+0
637      NIN = 5
638      NOUT = 6
639      WMACH(10) = NIN
640      WMACH(11) = NOUT
641 C
642 C----- IN MATFIV, ALLOW UP TO 100 UNDERFLOWS.
643 C----- CALL TRAPS ( 0,0,100 )
644      RETURN
645 C
646 C      END OF MCHPAR
647      END

```

***** OUTPUT FROM FDCALC. DETAILS FOR VARIABLE 1

INFORM AND ERROR BOUND	0	1.717994D-06
ESTIMATED GRADIENT AND HESSIAN DIAGONALS	1.000000D 01	1.200000D 01
BEST INTERVALS FOR FORWARD AND CENTRAL DIFFERENCES	1.431662D-07	1.884864D-06
FDCALC NEEDED 3 FUNCTION EVALUATIONS FOR THIS VARIABLE		

***** OUTPUT FROM FDCALC. DETAILS FOR VARIABLE 2

INFORM AND ERROR BOUND	0	1.731047D-05
ESTIMATED GRADIENT AND HESSIAN DIAGONALS	1.218249D 02	1.218304D 03
BEST INTERVALS FOR FORWARD AND CENTRAL DIFFERENCES	1.428867D-08	1.178040D-07
FDCALC NEEDED 5 FUNCTION EVALUATIONS FOR THIS VARIABLE		

***** OUTPUT FROM FDCALC. DETAILS FOR VARIABLE 3

INFORM AND ERROR BOUND	0	7.013683D-09
ESTIMATED GRADIENT AND HESSIAN DIAGONALS	1.002000D 00	2.000000D-04
BEST INTERVALS FOR FORWARD AND CENTRAL DIFFERENCES	3.506842D-05	1.036675D-03
FDCALC NEEDED 7 FUNCTION EVALUATIONS FOR THIS VARIABLE		

***** OUTPUT FROM FDCALC. DETAILS FOR VARIABLE 4

INFORM AND ERROR BOUND	4	1.312046D-06
ESTIMATED GRADIENT AND HESSIAN DIAGONALS	7.580661D-07	6.999000D 00
BEST INTERVALS FOR FORWARD AND CENTRAL DIFFERENCES	1.874620D-07	1.884864D-06
FDCALC NEEDED 3 FUNCTION EVALUATIONS FOR THIS VARIABLE		

ABNORMAL EXIT FOR THIS VARIABLE BECAUSE....

THE FORWARD AND CENTRAL ESTIMATES ARE NOT CLOSE.

TOTAL NUMBER OF FUNCTION EVALUATIONS = 19 NUMOK = 3

13. REFERENCES

- Gill, P. E., Murray, W., Saunders, M. A. and Wright, M. H. (1983). Computing forward-difference intervals for numerical optimization, *SIAM J. Sci. Stat. Comput.* 4, pp. 310-321.
- Gill, P. E., Murray, W. and Wright, M. H. (1981). *Practical Optimization*, Academic Press, London and New York.
- Lyness, J. N. (1977). "Has numerical differentiation a future?" *Proceedings of the 7th Manitoba Conference on Numerical Mathematics*, pp. 107-129.
- Ryder, B. G. (1974). The PFORT verifier, *Software-Practice and Experience* 4, pp. 359-377.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER SOL 83-6	2. GOVT ACCESSION NO. AD-A130 760	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Documentation for FCALC and FDCORE		5. TYPE OF REPORT & PERIOD COVERED Technical Report
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Philip E. Gill, Walter Murray, Michael A. Saunders and Margaret H. Wright		8. CONTRACT OR GRANT NUMBER(s) N00014-75-C-0267 DAAG29-81-K-0156
9. PERFORMING ORGANIZATION NAME AND ADDRESS Department of Operations Research - SOL Stanford University Stanford, CA 94305		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS NR-047-143
11. CONTROLLING OFFICE NAME AND ADDRESS Office of Naval Research - Dept. of the Navy 800 N. Quincy Street Arlington, VA 22217		12. REPORT DATE June
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) U.S. Army Research Office P.O. Box 12211 Research Triangle Park, NC 27709		13. NUMBER OF PAGES 21 pp.
		15. SECURITY CLASS. (of this report) UNCLASSIFIED
15a. DECLASSIFICATION/DOWNGRADING SCHEDULE		
16. DISTRIBUTION STATEMENT (of this Report) This document has been approved for public release and sale; its distribution is unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Nonlinear optimization Fortran subroutine Finite-difference approximation nonderivative method		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) This report documents two Fortran subroutines -- FDCALC and FDCORE -- that compute a set of forward-difference intervals to be used in minimizing a smooth function. The primary subroutine FDCORE, which is called repeatedly by FDCALC, produces a suitable interval for a univariate function, as well as approximations to the first and second derivatives. ↑		

END

FILMED

9-83

DTIC