ALGORITHM 666
CHABIS: A Mathematical Software Package for Locating and Evaluating Roots of Systems of Nonlinear Equations

MICHAEL N. VRAHATIS
University of Patras

CHABIS is a mathematical software package for the numerical solution of a system of n nonlinear equations in n variables. First, CHABIS locates at least one solution of the system within an n-dimensional polyhedron. Then, it applies a new generalized method of bisection to this n-polyhedron in order to obtain an approximate solution of the system according to a predetermined accuracy. In this paper we briefly describe the user interface to CHABIS and present several details of its implementation, as well as an example of its usage.

Categories and Subject Descriptors: G.1.5 [**Numerical Analysis**]: Roots of Nonlinear Equations—*systems of equations*; G.4 [**Mathematics of Computing**]: Mathematical Software

General Terms: Algorithms

Additional Key Words and Phrases: BLAS utilities, existence of the solution, generalized method of bisection, isolation of a root, localization of a root, solution of nonlinear systems, topological degree of a mapping

1. INTRODUCTION

In this paper we describe a set of nine subprograms, collectively called CHABIS (*cha*racteristic *bis*ection), which solves systems of nonlinear (algebraic and/or transcendental) equations of the form

$$F(X) = (0, 0, \dots, 0) \tag{1.1}$$

where $F = (f_1, f_2, \dots, f_n)$ is a continuous mapping of a bounded region $\mathscr{D} \subset R^n \to R^n$.

First, CHABIS locates at least one root of system (1.1) within an n-dimensional polyhedron, which is called the *characteristic n-polyhedron* [7]. CHABIS then obtains an approximate solution of system (1.1) according to a predetermined accuracy, using a new generalized method of bisection. This method of bisection

Author's present address: Department of Mathematics, White Hall, Cornell University, Ithaca, NY 14853.

is based on the refinement of a characteristic n-polyhedron and is called *characteristic bisection* [7].

CHABIS does not require computations of the topological degree; it requires only that the algebraic signs of the function evaluations be correct, so that it can be applied to problems with imprecise function values. Also, CHABIS can be applied to nondifferentiable continuous functions F and does not involve derivatives of F or approximations of such derivatives. CHABIS is useful primarily for small systems of equations (approximately 10 or less variables) for which a single solution is desired. It employs heuristics and requires BLAS (basic linear algebra subprograms) utilities [2–6]. Specifically, it requires the subprograms SCOPY, SNRM2, and ISAMAX (see [6, CALGO Algorithm 539, pp. 16–18, 21–23, 26–29]).

A complete description of the numerical methods employed in CHABIS along with its theoretical development is given in [7]. Here we describe the user interface to CHABIS and give implementation details, as well as an example of its usage.

2. INPUT PARAMETERS AND TERMINATION CRITERIA

For the construction of a characteristic n-polyhedron, CHABIS creates an initial n-dimensional polyhedron [7], which is a scaled translation of the unit n-cube. To create this initial n-polyhedron, CHABIS must be supplied with an arbitrary point XO in R^n that determines an initial guess of the solution and n arbitrary stepsizes in each coordinate direction $\mathrm{H} = (h_1, h_2, \ldots, h_n)$ in such a way that the corresponding initial n-polyhedron will form a domain in which CHABIS attempts to locate and evaluate a root of system (1.1). On the edges of the initial n-polyhedron, CHABIS strives to evaluate roots of the components $f_1, f_2, \ldots, f_n$ of F. If CHABIS evaluates one such root, it then creates and uses two points from both sides of the root to construct vertices of a characteristic n-polyhedron. Of course, it is not necessary for CHABIS to compute these roots with a high accuracy; CHABIS computes the roots using a predetermined accuracy DELTA, which is an input parameter and permits the user to obtain the roots with the accuracy of his or her choice. CHABIS determines if the input value of DELTA is less than the machine precision EPSMCH, and if so, it changes DELTA to the value 0.0625. EPSMCH is computed within CHABIS.

When a characteristic n-polyhedron is constructed, CHABIS bisects it in order to obtain an approximate solution. Convergence occurs if CHABIS produces a point AS such that $\| \mathrm{F(AS)} \|_\infty \leq$ EPSILO, where EPSILO is a predetermined precision not less than the machine precision EPSMCH. CHABIS determines if the input value of EPSILO is less than the value of EPSMCH, and if so, it changes the value of EPSILO to that of EPSMCH. Convergence may also occur if CHABIS produces a refined characteristic n-polyhedron such that the length of its longest diagonal [7] is less than $2.0 \cdot n \cdot$EPSILO.

Another termination criterion is created by CHABIS: CHABIS computes the maximum number of iterations of the bisection portion of the algorithm and terminates if CHABIS determines that the iterations have reached this upper limit. The maximum number of iterations depends on the value of EPSILO and the length of the longest diagonal of the constructed characteristic n-polyhedron.

The bisection portion of CHABIS is evoked when a characteristic n-polyhedron is constructed; but the user can permit CHABIS to evaluate an approximate solution even if the construction of a characteristic n-polyhedron fails. To do this, the user must assign the value 1 to the input parameter ICON.

Moreover, termination occurs if CHABIS determines one of the following conditions:

(1) The input value of n is less than 2,
(2) an input stepsize is less than or equal to 0, or
(3) the input length of a workspace array (see below) does not have the proper value.

3. DESCRIPTION OF THE SOFTWARE

CHABIS consists of a set of nine subprograms, one of which is called by the user driver program. Namely, CHABIS consists of the subroutines INTSUB, CHAPOL, and GENBIS, and six FORTRAN-callable subprograms for various numerical operations that resemble the BLAS utilities [2–6] and that are thus called *BLAS-like utilities*. The user must only call the subroutine INTSUB; all the other subprograms are evoked by INTSUB. The user must also provide the function FNC, which calculates components of the function of system (1.1).

INTSUB is an interface subroutine between the user driver program and the subroutines CHAPOL and GENBIS through which a single work-space array WA is passed. INTSUB evokes CHAPOL and GENBIS in such a way that the addresses of the corresponding arrays within the subroutines are computed in WA. In this way, their arrays can appear with variable dimensions (adjustable arrays), and the calling sequence of INTSUB can be shortened and simplified. Note that INTSUB has 13 parameters in the calling sequence, while each subroutine, that is, CHAPOL and GENBIS, has 22 parameters. The purpose of CHAPOL (*cha*racteristic *pol*yhedron) is to construct a characteristic n-polyhedron. The purpose of GENBIS (*gen*eralized *bis*ection) is to refine the constructed characteristic n-polyhedron in order to find an approximate solution of system (1.1). The BLAS-like utilities include determination of whether the infinity norm of a vector is smaller than a constant; they include vector comparisons and vector scaling, as well as transfer of the sign of a variable. Complete details for each one of the above subprograms are given in the CHABIS documentation.

CHABIS is evoked by the following FORTRAN statement:

```
   CALL INTSUB(FNC, N, XO, H, DELTA, EPSILO, ICON, INF1,
 +                  AS, VAS, INF2, WA, LWA)
```

where

FNC is the user-supplied function that evaluates components of the given function. FNC should be declared in an external statement in the user-calling program and should be written as follows:

```
REAL FUNCTION FNC(X, IFLAG)
INTEGER IFLAG
REAL X(N)
```

```
------------------------------------------------------------------------------------
Calculate the IFLAGth component of the function at X.
------------------------------------------------------------------------------------
RETURN
END
```

N	is a positive integer input variable that defines the number of equations and variables.
XO	is an input array of length N that defines the initial guess of the solution.
H	is an input array of length N, with positive entries, that determines the stepsizes in each coordinate direction.
DELTA	is a positive input variable that determines the accuracy of the computation of the roots, of the components of the given function, which are located on the edges of the initial n-polyhedron and used for the construction of a characteristic n-polyhedron. If DELTA is less than the machine precision EPSMCH, DELTA takes the value of 0.625E−1. The value of EPSMCH is computed within INTSUB.
EPSILO	is a nonnegative input variable. Termination occurs when the algorithm estimates that the infinity norm of the function values at an approximate solution is at most EPSILO. If EPSILO is less than the machine precision EPSMCH, EPSILO becomes equal to EPSMCH.
ICON	is an integer input variable. If ICON is equal to 1, then GENBIS is evoked even if a characteristic n-polyhedron has not been constructed.
INF1	is an integer output variable set as follows:

INF1 = 0	Improper input parameters—CHAPOL and GENBIS have not been evoked.
INF1 = 1	A characteristic n-polyhedron has been constructed.
INF1 = 2	A characteristic n-polyhedron has not been constructed.
INF1 = 3	More than two vertices of the constructed characteristic n-polyhedron are located on the same edge of the initial n-polyhedron.
INF1 = 4	An approximate solution according to the precision EPSILO has been found during the construction of the characteristic n-polyhedron. GENBIS has not been evoked.

AS	is an output array of length N that determines the final approximate solution.
VAS	is an output array of length N that specifies the function values at AS.
INF2	is an integer output variable set as follows:

INF2 = 0	GENBIS has not been evoked.
INF2 = 1	The solution has been found within the required accuracy of EPSILO.

INF2 = 2	Iterations have reached their upper limit. The answer may not be accurate.
INF2 = 3	The length of the longest diagonal has been found to be less than 2.0 * N * EPSILO.

WA	is a work-space array of length LWA.
LWA	is a positive integer input variable not less than the value of (2*N + (6*N + 1)*2**N).

CHABIS contains about 1,200 lines of code, 50 percent of which are comments. The total storage required for CHABIS is $6n + (6n + 1)2^n$ locations, where n determines the dimension of problem (1.1). CHABIS is coded in ANSI standard FORTRAN (1977) [1] and has been tested on the University of Patras UNIVAC 1100/60 system, on the Cornell University IBM 3090-600E (supercomputer mainframe), IBM 4381, and VAX 8530, and on a SPERRY IT IBM PC compatible.

4. EXAMPLE

We give an example that demonstrates how CHABIS is used to solve a system of nonlinear equations. Suppose we wish to evaluate an approximate solution of the following system [7]:

$$\begin{aligned} f_1(X) &= x_1^2 - 4x_2 = 0 \\ f_2(X) &= x_2^2 - 2x_1 + 4x_2 = 0. \end{aligned} \tag{4.1}$$

This system has the solutions $S_1 = (0, 0)$ and $S_2 = (1.695 \ldots, 0.718 \ldots)$. The sign of the Jacobian at S_1 is -1, while the sign of the Jacobian at S_2 is $+1$. So, the topological degree [7] relative to any region containing both solutions must be equal to 0. We know that the topological degree relative to a characteristic n-polyhedron is ± 1 [7]; thus, we must choose a region that does not contain both solutions. Generally, we are not in a position to know this beforehand; it is only known if we compute the topological degree. In this case, the topological degree will be 0, so no conclusions can be drawn as to the exact number of solutions of system (4.1). We generally prefer to permit CHABIS to strive for the construction of a characteristic n-polyhedron instead of the calculation of the exact value of the topological degree. This is because the calculation of the exact value of the topological degree is quite a time-consuming procedure, and as we shall see below, we can obtain an approximate solution of system (4.1) even if the construction of a characteristic n-polyhedron fails.

To use CHABIS for the computation of an approximate solution of system (4.1), we need an initial guess of the solution XO, the stepsizes H, and the values of DELTA, EPSILO, and ICON; for example, take the values XO = (−2000, −2000), H = (4000, 4000), DELTA = 0.0625, EPSILO = 10^{-6}, and ICON = 1. The following FORTRAN program can be used to evoke CHABIS for the solution of this system:

```
*
*       Example of CHABIS usage.
*
```

```
      PROGRAM MAIN
      IMPLICIT REAL(A - H, O - Z), INTEGER(I - N)
      PARAMETER (N = 2, LWA = 2*N + (6*N + 1)*2**N)
      DIMENSION XO(N), H(N), AS(N), VAS(N), WA(LWA)
      EXTERNAL FNC
      COMMON / BLK1 / NFCALL
*
*     Set the values of DELTA, EPSILO, and ICON.
*
      DATA DELTA, EPSILO, ICON / 0.625E - 1, 1.0E - 6, 1 /
*
*     Set the starting values.
*
      DATA XO / 2* - 2000.0E0 /
      DATA H / 2*4000.0E0 /
      PRINT 9999, ( XO(J ), H(J ), J = 1, N )
*
*     Call the interface subroutine INTSUB.
*
      NFCALL = 0
      CALL  INTSUB( FNC, N, XO, H, DELTA, EPSILO, ICON, INF1,
     +                   AS, VAS, INF2, WA, LWA )
      IF ( INF1 .EQ. 0 ) THEN
        PRINT 9998
      ELSEIF ( INF1 .EQ. 2 .AND. ICON .NE. 1 ) THEN
        PRINT 9997, N
      ELSE
        PRINT 9996, DELTA, EPSILO, (AS(J), VAS(J), J = 1, N)
      ENDIF
      NFCALL = INT( REAL( NFCALL ) / REAL( N ) )
      PRINT 9995, INF1, INF2, NFCALL
      STOP
*
 9999 FORMAT (//2X, 'INITIAL GUESS :', 17X, 'STEPSIZES :'//
     +        2( F16.7, 16X, F16.7 / ) )
 9998 FORMAT (//5X, '*** IMPROPER INPUT PARAMETERS ***'//)
 9997 FORMAT (//5X, '*** THE CHARACTERISTIC', I2,
     +        '-POLYHEDRON HAS NOT BEEN COMPLETED ***'//)
 9996 FORMAT (//2X, 'DELTA = ', F20.18 //2X, 'EPSILO = ', F20.18
     +        ////2X, 'FINAL APPROXIMATE SOLUTION :', 5X,
     +        'VERIFICATION OF THE SOLUTION :' //9(F24.18, 9X,
     +        F24.18/ ))
 9995 FORMAT (//2X, 'EXIT PARAMETERS : INF1 =', I1, 1X, ', INF2 ='
     +        , I1//2X, 'NUMBER OF FUNCTION CALLS : NFCALL = ',
     +        I4 )
      END
* ------------------------------------------------------------------------
      REAL FUNCTION FNC( X, IFLAG )
      INTEGER IFLAG, NFCALL
      REAL X(2)
      COMMON / BLK1 / NFCALL
      NFCALL = NFCALL + 1
      GO TO ( 1, 2 ), IFLAG
    1 FNC = X(1)** 2 - 4.0E0*X(2)
      RETURN
    2 FNC = X(2)** 2 - 2.0E0*X(1) + 4.0E0*X(2)
      RETURN
      END
```

The output results of the above program indicate that even when using the above starting values and obtaining INF1 = 2, which means that a characteristic 2-polyhedron has not been constructed, we still obtain the solution $S_1 = (0, 0)$, utilizing 5 function evaluations. Now, if we change the above XO to XO = (0.1, 0.1) and H to H = (2000, 2000), we can obtain a constructed characteristic 2-polyhedron and the solution S_2, utilizing 70 function evaluations.

ACKNOWLEDGMENTS

I wish to express my appreciation and thanks to Michael A. Saunders and the referees for their constructive and valuable comments. Particularly, I would like to thank Michael A. Saunders for his encouragement to make use of the BLAS utilities and to write the BLAS-like utilities to enhance the readability of CHABIS. I also wish to express my appreciation and thanks to Martha K. Preston for her patient proofreading.

REFERENCES

1. AMERICAN NATIONAL STANDARDS INSTITUTE. ANSI FORTRAN X3.9-1978. ANSI, New York, 1978. (Also known as FORTRAN 77.)
2. DONGARRA, J. J., DU CROZ, J. J., HAMMARLING, S., AND HANSON, R. J. An extended set of Fortran basic linear algebra subprograms. *ACM Trans. Math. Softw. 14*, 1 (Mar. 1988), 1–17.
3. DONGARRA, J. J., DU CROZ, J. J., HAMMARLING, S., AND HANSON, R. J. Algorithm 656: An extended set of Fortran basic linear algebra subprograms: Model implementation and test programs. *ACM Trans. Math. Softw. 14*, 1 (Mar. 1988), 18–32.
4. HANSON, R. J., AND KROGH, F. T. Algorithm 653: Translation of Algorithm 539: PC-BLAS, basic linear algebra subprograms for Fortran usage with the INTEL 8087, 80287 numeric data processor. *ACM Trans. Math. Softw. 13*, 3 (Sept. 1987), 311–317.
5. LAWSON, C. L., HANSON, R. J., KINCAID, D. R., AND KROGH, F. T. Basic linear algebra subprograms for Fortran usage. *ACM Trans. Math. Softw. 5*, 3 (Sept. 1979), 308–323.
6. LAWSON, C. L., HANSON, R. J., KINCAID, D. R., AND KROGH, F. T. Algorithm 539: Linear algebra subprograms for Fortran usage [F1]. *ACM Trans. Math. Softw. 5*, 3 (Sept. 1979), 324–325. (Also in Collected Algorithms from ACM (CALGO) Algorithm 539.)
7. VRAHATIS, M. N. Solving systems of nonlinear equations using the nonzero value of the topological degree. *ACM Trans. Math. Softw. 14*, 4 (Dec. 1988), 312–329.

Received September 1983; revised June 1987, February 1988, and June 1988; accepted June 1988