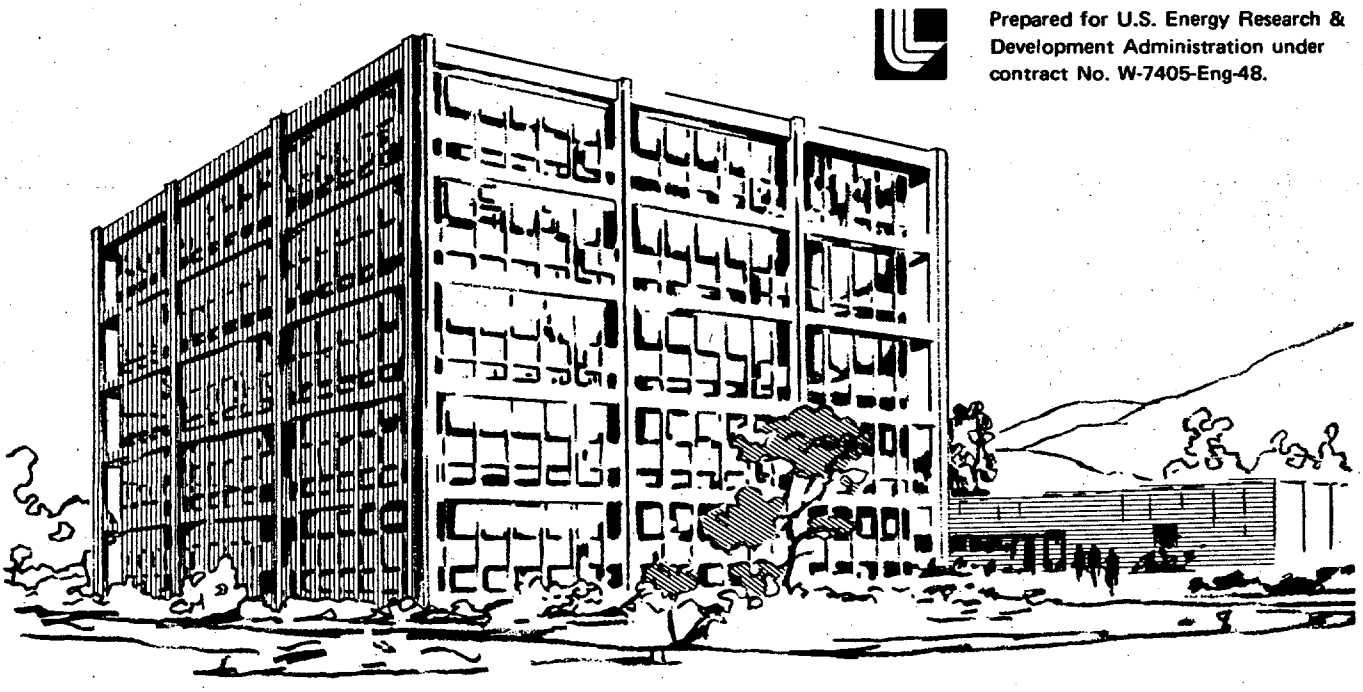
UCID-30149
COMPUTER
DOCUMENTATION

Lawrence Livermore Laboratory

PRELIMINARY DOCUMENTATION OF GEARBI: SOLUTION OF ODE SYSTEMS WITH BLOCK-ITERATIVE TREATMENT OF THE JACOBIAN

A. C. Hindmarsh

December 1976



Prepared for U.S. Energy Research & Development Administration under contract No. W-7405-Eng-48.

CONTENTS

PRELIMINARY DOCUMENTATION OF GEARBI: SOLUTION OF ODE SYSTEMS WITH BLOCK-ITERATIVE TREATMENT OF THE JACOBIAN

A. C. Hindmarsh

ABSTRACT

This report is a preliminary version of documentation for the GEARBI package. This package solves systems of ordinary differential equations (initial value problem), with emphasis on stiff systems, in which the Jacobian matrix has a regular block structure. The package takes advantage of that structure and uses a block-iterative (block-SOR) method to solve the linear algebraic systems that arise. In all other respects, GEARBI is much the same as the GEAR package (UCID-30001, Rev. 3). An earlier version of GEARBI has been used for atmospheric kinetics-transport problems in one and two dimensions.

INTRODUCTION

The GEARBI package is concerned with systems of ordinary differential equations (ODE's) of the form

$$\dot{y} = dy/dt = f(y,t) \ , \qquad (1)$$

in which y and f are vectors of length N, and in which an initial value $y_0 = y(t_0)$ is given. The older and heavily used GEAR package [1] is concerned with the same problem, but has severe limitations for large stiff problems, which GEARBI is intended to overcome. For the efficient solution of stiff problems, it is necessary to construct and use the

Jacobian matrix,

$$J = \partial f/\partial y = (\partial f_i/\partial y_j)_{i,j=1}^{N} . \qquad (2)$$

In GEAR, J is treated as a full N x N matrix; hence, the limitation for large N. In contrast, GEARBI assumes that J has a certain regular block structure, one that is present in many applications of interest, and attempts to take maximum advantage of that structure, in economy of both storage and computation.

To begin with, consider a system of p partial differential equations in a space variable r (in any number of dimensions) and time t,

$$\partial u_i/\partial t = L(u_i) + R_i(u_1,\ldots,u_p) \quad (i = 1,2,\ldots,p). \qquad (3)$$

Here L is a fixed linear spatial differential operator which involves only u_i in the i^{th} equation, while R_i may involve all p variables but no spatial derivatives. Both L and the R_i may also be functions of r and t. (The R_i might be chemical kinetic rate functions, for example.) Suppose that the numerical method of lines [2] with finite differencing is to be applied to (3). The result is a system of ODE's of the form (1). If the total number of nodes in the differencing scheme is q, then the components of y are the approximations u_{ij} to u_i at the nodes $r_j (j = 1,\ldots,q)$, or in q zones, and the number of components is N = pq. If $\bar{L}_j$ represents the chosen (linear) difference analog of L at r_j, then the ODE representing (3) at r_j is

$$du_{ij}/dt = \bar{L}_j(u_{i1},\ldots,u_{iq}) + R_i(u_{1j},\ldots,u_{pj}) \qquad (4)$$

The Jacobian of the system (4) is a blocked matrix,

$$J = \begin{bmatrix} D_1 & a_{12}I_p & a_{13}I_p & \cdots & a_{1q}I_p \\ a_{21}I_p & D_2 & a_{23}I_p & \cdots & \\ \cdots & & & & \\ & & & & \\ a_{q1}I_p & a_{q2}I_p & & \cdots & D_q \end{bmatrix} \qquad (5)$$

That is, it consists of p x p blocks, with q blocks in each direction along the matrix. Here D_j, the block at the i^{th} diagonal block position, is a p x p matrix depending only on $u_{1j},\ldots,u_{qj}$,

$$D_j = \left[\frac{\partial \bar{L}_j(u_{\alpha 1},\ldots,u_{\alpha q})}{\partial u_{\alpha j}} \delta_{\alpha\beta} + \frac{\partial R_\alpha(u_{1j},\ldots,u_{pj})}{\partial u_{\beta j}} \right]_{\alpha,\beta=1}^{p} \qquad (6)$$

where $\delta_{\alpha\beta}$ is the Kronecker δ. The off-diagonal blocks are multiples of the p x p identity matrix I_p. The coefficients are

$$\alpha_{\gamma\delta} = \frac{\partial \bar{L}_\gamma(u_{i1},\ldots,u_{iq})}{\partial u_{i\delta}} \quad (\gamma,\delta = 1,\ldots,q;\ \gamma \neq \delta) \qquad (7)$$

(the above expression is independent of i, by linearity) and these form a q x q matrix A. (Take the diagonal elements of A to be zero.) The D_j (j=1,...,q) and A describe completely the Jacobian J. In fact, GEARBI assumes only the structure (5), whether or not it arises from a problem of the form (3).

Here we will regard the D_j as full p x p matrices, although in practice they may be sparse and their structure could be utilized with appropriate sparse matrix algorithms. The matrix A, however, is regarded as large and sparse, as it represents the geometry of the region of interest in r-space, the manner in which it is covered by q nodes

or zones, and the nature of the finite differencing in (4). For simplicity, and for maximum efficiency in the case of a regular geometry, we assume that the sparseness structure of A is described in terms of the diagonal lines containing its nonzero elements. Thus, for example, for a rectangle in two dimensions with a 5-point discrete operator, A will have four nonzero diagonal lines - one immediately below the main diagonal, one further down, and the other two in symmetric positions above the main diagonal. We assume in general that the nonzero diagonals of A lie in symmetric positions relative to the main diagonal. Of course, only the nonzero diagonal lines of A are stored.

The assumptions in (3) that L is linear and fixed are made because they are relevant in many applications and because they simplify both the data structure and the algorithm to be used. The same alogrithm, and the GEARBI package, can be easily extended to more general problems, by replacing $a_{\gamma\delta}I_p$ above by an arbitrary diagonal p x p matrix, or, most generally of all, an arbitrary p x p matrix. However, storage considerations become much more restrictive in that case.

An alternative approach in the case of more general problems (e.g. nonlinear L or component-dependent transport coefficients) is to approximate the Jacobian J by a matrix which <u>does</u> satisfy the assumptions made here, and supply that approximation where called for. This amounts to solving a nonlinear equation by Newton's method with somewhat inaccurate slope data. The limit will be accurate, but may require more iterations. (One must <u>not</u> approximate similarly in the computation of the function f, or the right-hand sides of (3), however, as this would mean solving a different problem.)

The solution by GEARBI of the initial value problem for (1) or (4) is done (as in GEAR) by implicit linear multistep methods, of either the Adams or the BDF (backward differentiation formula) type. In either case, the algorithm is both variable-order and variable-step-size. The solution of the implicit step equation by a Newton-like (or chord) method leads to the solution of linear systems where coefficient matrix is

$$P = I - h\beta_0 J \ , \tag{8}$$

where I is the N x N identity matrix, h is the step size in t, β_0 is a scalar associated with the selected method and current order, and J is the Jacobian (2). Clearly the Newton matrix P has the same block structure as J does, and we will use the notation in (5) for P as well as J. The solution of a linear system

$$Px = b \tag{9}$$

is then done in GEARBI by a block-iterative method based on the given block structure.

The specific method used here is block successive overrelaxation (block-SOR) and can be summarized as follows. If x and b are regarded as segmented with segments x_i and b_i of length p, then we begin by estimating x as x^0 according to

$$D_i \, x_i^0 = b_i \ , \tag{10}$$

i.e., by ignoring the elements of A. Then subsequent iterates x^ν are generated according to

$$D_i x_i^{\nu} = \omega b_i + (1 - \omega) D_i x_i^{\nu-1} - \omega \sum_{j<i} a_{ij} x_j^{\nu} - \omega \sum_{j>i} a_{ij} x_j^{\nu-1} \quad (i=1,2,\ldots q) , \tag{11}$$

where ω is the overrelaxation parameter. For maximum efficiency in both (10) and (11), the solution of the p x p systems, with coefficient matrices D_i, is done by computing LU decompositions of these blocks, saving these, and backsolving as necessary, using subroutines DEC and SOL [3]. The parameter ω is chosen adaptively as the iteration proceeds [4]. A convergence test is applied at each iteration, based on the local error to be allowed in the ODE solution. Details of these methods will be given in a later report.

Historically, the GEARBI package was originally developed in late 1972, for use in studies of stratospheric kinetics-transport at LLL [5]. The original version of GEARBI has been considerably modified and improved in the present version. The principal modifications are in the use of a more convenient user interface (especially the new driver routine), new and faster full linear system solvers, and a better algorithm for selecting ω. Most of the changes correspond to improvements that have been made to the GEAR package since 1972.

In 1973, an extended version of GEARBI, called GEARBIL, was also written. This was based on the same algorithm as in GEARBI (as it existed then), but with all of the large arrays stored in Large Core Memory (LCM) on the CDC 7600. That package has been (and is) used with considerable success in a wide variety of atmospheric modeling problems, e.g., [5], [6], with values of p

up to 20 and values of q up to about 1500. In the near future, an updated version of GEARBIL, based on the present GEARBI, will be generated.

AVAILABILITY

The GEARBI package is written in reasonably standard Fortran, in single precision, with no language features which are peculiar to LLL. At LLL, the source code is available from the NMG Mathematical Software Library, by way of the access routine MSLAR, using the teletype command

NMG MSLAR READ DRIVBI END (return)

The package has also been cleared for unlimited release outside LLL. A listing is available on request.

The structure of the GEARBI package is quite analogous to that of GEAR. It consists of nine subroutines: DRIVBI, INTERP, STIFBI, COSET, PSETBI, SOLBI, OMEGA, DEC, and SOL. The routines INTERP, COSET, DEC, and SOL are identical to the routines by the same names in GEAR. Subroutines DRIVBI, STIFBI, and PSETBI are similar to DRIVE, STIFF, and PSET in GEAR. SOLBI and OMEGA are devoted to the block-SOR linear system solution performed by GEARBI.

Most of the routines contain OPTIMIZE cards, for maximum efficiency under CHAT (or ORDER). These must be removed for use on other compilers.

A demonstration program is also available, for purposes of illustration. (See the last section below for a description of the demonstration problem.) The source code for this program, <u>together with</u> the GEARBI package, is obtainable with the command

NMG MSLAR READ DEMO DRIVBI END (return)

USAGE

As with the GEAR package, the usage of GEARBI requires the user to make calls to a driver routine in the package and to supply subroutines which describe the function f and the Jacobian J. The driver subroutine is DRIVBI, and the user supplied routines are called DIFFUN, PDBD, and ASET. What follows is an abbreviated description of the call sequences for these routines. They are also rather fully described in comment cards in the source code of DRIVBI.

The calling sequence to DRIVBI is as follows:

CALL DRIVBI (N, TØ, HØ, YØ, TOUT, EPS, MF, INDEX, MATP)

The arguments N, TØ ($=t_0$), HØ (initial step size), YØ ($=y_0$), TOUT (next output value of t), and EPS (local error tolerance) have meanings identical to those in GEAR, as described in [1].

The method flag MF is similar, but has only 6 values, not 8 - 10, 11, 13, 20, 21, and 23. Its first digit, METH, is 1 for Adams and 2 for BDF. Its second digit is 0 for functional iteration, 1 for modified Newton iteration with user-supplied Jacobian, and 3 for modified Newton iteration with a diagonal Jacobian approximation. These differ from the values in GEAR in the absence of MITER=2 (internally generated Jacobian) and in the special meaning of MITER=1 for the block structure assumed here. Also, to keep the storage requirements down, the order is restricted to 5 for both method types.

The flag INDEX has the same input meanings as for GEAR (1 for first call, 0 for normal continuation, etc.). Its output values are also the same except that INDEX=-3 means that the problem was halted either because of repeated failures of the corrector convergence test, or because of repeated failures of the block-iterative (inner) convergence test (the printed error messages will indicate which).

The argument MATP is new with GEARBI; it is not present in GEAR. It is an integer vector containing the parameters needed to describe the structure of the Jacobian matrix, as shown in (5). Its components are as follows:

MATP(1) = p = the size of the blocks in J, and the number of partial differential equations in (3) (if that is the application).

MATP(2) = q = the number of blocks in each direction along J, and the number of points in the discretization of each equation in (3).

MATP(3) = NA = the number of diagonal lines in the q x q matrix $A = (a_{ij})$ which contain nonzero elements and which lie strictly below the main diagonal. The number of such lines strictly above the main diagonal is assumed to be the same, and they are assumed to lie in symmetric positions.

MATP(3+k) = IDA(k) (k=1,2,...,NA) is the row index in A at which the k^{th} nonzero lower diagonal starts in column 1, $2 \le IDA(k) \le q$. Thus if a lower diagonal line of a_{ij} with $i - j = \ell$ is labeled as the k^{th} lower diagonal, then $IDA(k) = \ell + 1$. Its reflection is the line of a_{ij} with $j - i = \ell = IDA(k) - 1$. (IDA is the name of a vector internal to the package; the notation IDA(k) is used here in place of MATP(3 + k) only for convenience.)

The allocation of problem-dependent storage is done in DRIVBI, using labeled Common blocks. The source code, as supplied, contains dimensions which allow for $p \le 20$, $N \le 500$, $p*N \le 2500$, $2*NA*q \le 800$, and $NA \le 9$. For problem sizes exceeding any of these limits, alter the dimensions in DRIVBI according to the instructions in comment cards there.

The right-hand side of the ODE system (1), f(y,t) is supplied by the user in the form of Subroutine DIFFUN(N,T,Y,YDOT). It has exactly the same parameters as in the GEAR package.

The Jacobian matrix (when MITER=1) is supplied by two routines, Subroutine PDBD(T,YI,I,DI,MP) and Subroutine ASET(T,Y,AA,MQ). Here T is t, MP is p, and MQ is q. PDBD is to compute the diagonal block D_I of the Jacobian (see (6)). It is to store D_I into the MP by MP array DI. The argument YI is the I^{th} segment of length p in the y vector, i.e., the p variables associated with the I^{th} node or zone. (By assumption, DI does not depend on any other component of y.) ASET is to compute the off-diagonal coefficients a_{ij}, or those which lie on the diagonal lines described by MATP. It is to store these into the MQ by 2*NA array AA, as follows: The first NA columns of AA (each of length MQ) are to be set to the lower diagonal lines of A in the order indicated by IDA, with elements in row i of A stored in the i^{th} position in the appropriate columns of AA. Thus lower elements a_{ij} with $i - j = \ell$ = IDA(k) - 1 go into AA(i,k). The remaining NA columns of AA are to be set to the upper diagonal lines, with the reflection of the k^{th} lower line stored in column NA + k. Thus upper elements a_{ij} with $j - i = \ell$ = IDA(k)-1 go into AA(i,NA+k).

The call to ASET is made soon after entering the package routine PSETBI, and is thereby executed each time the Newton matrix P, defined by (8), is updated. However, if the A matrix is <u>constant</u>, then ASET need be called only once. In that case, the call can be removed from PSETBI and inserted into DRIVBI, to just below statement number 17.

As suggested earlier, it may be convenient to supply, through PDBD and ASET, a matrix J that only approximates the true value of

$\partial f/\partial y$. This would allow the GEARBI package, without modification, to solve the problem (3) even if, for example, L were mildly nonlinear, or if the off-diagonal coefficients given by (7) were slightly different from one component to another, or if some of the coefficients were small enough to be neglected entirely in J, thereby making the structure of A significantly simpler than it would otherwise be.

If the matrices D_j given by (6) are sparse (say, with at least half of the elements being zero), and if the sparseness structure is the same for all j, then efficiency could quite possibly be improved considerably by substituting appropriate sparse matrix routines for DEC and SOL.

DEMONSTRATION PROBLEM

A demonstration program, which uses GEARBI to solve a simple problem with a known analytic solution, has been written to validate the package. To date, no other testing of GEARBI has been done, other than the actual use of earlier versions of the package.

The demonstration problem used here is based on a simple 2-dimensional advective transport problem. Consider a square g x g grid, with the g^2 nodes indexed rowwise by (i,j) with $0 \le i,j < g$. At each node consider two functions u_{ij} and v_{ij}, given by the (identical) ODE's

$$\dot{u}_{ij} = \alpha_1 u_{i-1,j} + \alpha_2 u_{i,j-1} - 2u_{ij}$$

$$\dot{v}_{ij} = \alpha_1 v_{i-1,j} + \alpha_2 v_{i,j-1} - 2v_{ij} \qquad (12)$$

in which $0 \le i,j < g$ and any term with a negative subscript is dropped. If the initial values are $u_{00} = v_{00} = 1$, and all other u_{ij} and v_{ij} are zero, then the true solution is

$$u_{ij} = v_{ij} = \alpha_1^i \alpha_2^j t^{i+j} e^{-2t} / i!j! \quad . \tag{13}$$

We now form a vector y of size $N = 2g^2$ out of the u_{ij} and v_{ij} by proceeding thorugh the nodes in the reverse of the usual order. Thus for $0 \leq i,j < g$, set

$$k = (g-i) + (g-1-j)g$$

$$y_{2k-1} = u_{ij}$$

$$y_{2k} = v_{ij} \quad .$$

Then (12) becomes

$$\dot{y} = My \tag{14}$$

where M is an upper triangular matrix with an appropriate block structure. The diagonal blocks are all equal to $-2I_2$ (I_2 = 2×2 identity matrix). The blocks just above the main diagonal, except in block positions (i,i+1) with $i \equiv 0 \pmod g$, are all $\alpha_1 I_2$. The blocks in the block positions (i,i+g) are all $\alpha_2 I_2$. All other elements of M are zero.

The particular case used here is with g = 10 (N = 200), $\alpha_1 = 1.5$, and $\alpha_2 = .7$. These choices of α_1 and α_2 make M fail to be diagonally dominant, and the choice of ordering of the variables makes the block iteration process nontrivial. (With the natural ordering, M would be lower triangular, and the process would give exact solutions to (9) in one iteration.) We wish to integrate the problem from t = 0 to t = 1000 and take output at powers of 10 starting at $t = 10^{-2}$.

This problem is suitable for solution by GEARBI if we take p = 2 and $q = g^2 = 100$. The diagonal blocks of the Jacobian, M, are $-2I_2$, and the 100×100 matrix A has its nonzero elements on two lines above the main diagonal. Because of the symmetry assumed by GEARBI, we must also include

the reflections of these two lines. Thus NA is 2, and we take IDA(1) = 2, IDA(2) = g+1 = 11. The demonstration program supplied with the GEARBI package solves the problem for each of the six values of MF, and checks the computed answers against those in (13).

REFERENCES

[1] A.C. Hindmarsh, GEAR: Ordinary Differential Equation System Solver, LLL Report UCID-30001, Rev. 3, December 1974.

[2] R.F. Sincovec and N.K. Madsen, Software for Nonlinear Partial Differential Equations, ACM-Trans. Math. Software, 1 (1975), 232-260.

[3] A.C. Hindmarsh, et al., DEC/SOL: Solution of Dense Systems of Linear Algebraic Equations, LLL Report UCID-30137, June 1976.

[4] L.A. Hageman, The Estimation of Acceleration Parameters for the Chebyshev Polynomial and the Successive Overrelaxation Iteration Methods, Bettis Atomic Power Laboratory Report WAPD-TM-1038, 1972; LLL implementation due to N.K. Madsen.

[5] J.S. Chang, A.C. Hindmarsh, and N.K. Madsen, Simulation of Chemical Kinetics Transport in the Stratosphere, in Stiff Differential Systems, Plenum Press, New York, 1974, R.A. Willoughby, ed.

[6] M.C. MacCracken and G.D. Sauter, editors, Development of an Air Pollution Model of the San Francisco Bay Area, LLL Report UCRL-51920, 2 volumes, 1975.

Printed in the United States of America
Available from
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Road
Springfield, VA 22161
Price: Printed Copy $; Microfiche $3.00

Page Range	Domestic Price	Page Range	Domestic Price
001–025	$ 3.50	326–350	10.00
026–050	4.00	351–375	10.50
051–075	4.50	376–400	10.75
076–100	5.00	401–425	11.00
101–125	5.50	426–450	11.75
126–150	6.00	451–475	12.00
151–175	6.75	476–500	12.50
176–200	7.50	501–525	12.75
201–225	7.75	526–550	13.00
226–250	8.00	551–575	13.50
251–275	9.00	576–600	13.75
276–300	9.25	601–up	*
301–325	9.75		

*Add $2.50 for each additional 100 page increment from 601 to 1,000 pages; add $4.50 for each additional 100 page increment over 1,000 pages.