AUTO 86 USER MANUAL

Software for Continuation and Bifurcation Problems in Ordinary Differential Equations.

Eusebius Doedel

January 1986

Companion document:

Software for Continuation Problems in Ordinary Differential Equations with Applications

Table of Contents

Acknowledgements.

The author is thankful to J. Alexander (University of Maryland), K. Bar-Eli (Tel-Aviv University), S. Dunbar (University of Nebraska), J. Dockery (University of Utah), B. Hassard (SUNY, Buffalo), M. Küper (Universität Dortmund), J. Rinzel (N.I.H., Washington), A. Schroll (Universität Konstanz), for detailed comments on earlier versions of AUTO.

Also acknowledged are the contributions of students at Concordia University, Montréal, to the development of the interactive graphics program PLAUT, included in the current package. In particular Nguyen Thanh Long developed the 3D capability of PLAUT.

The development of AUTO has been supported in part by the Natural Sciences and Engineering Research Council of Canada (A4274), and La Fondation FCAC, Québec, (EQ1438). Work on AUTO 86 was carried out at Caltech with support from IBM.

1. Files contained in the AUTO Package.

Included on the VAX/VMS tape are:

AUTOS.COM	: The command file for running AUTO (single precision).
AUTOD.COM	: The command file for running AUTO (double precision).
AUTPP2.FOR	: Example: Hopf bifurcation phenomena.
AUTEXP.FOR	: Example: A boundary value problem.
AUTINT.FOR	: Example: A BVP with integral constraint.
AUTDD2.FOR	: Example: A discrete dynamical system.
AUTOPT.FOR	: Example: An algebraic optimization problem.
AUTBVP.FOR	: Example: A BVP with bifurcations.
AUTLIN.FOR	: Example: A linear eigenvalue problem.
AUTPP3.FOR	: Example: Period doubling bifurcations.
AUTWAV.FOR	: Example: Wave phenomena.
AUTTIM.FOR	: Example: A BVP for timing purposes.
AUTPRPS.FOR	: The preprocessor (single precision).
AUTPRPD.FOR	: The preprocessor (double precision).
AUTPRPS.OBJ	: The object code of the preprocessor (single precision).
AUTPRPD.OBJ	: The object code of the preprocessor (double precision).
AUTLIBS.FOR	: The library of subroutines (single precision).
AUTLIBD.FOR	: The library of subroutines (double precision).
AUTLIBS.OLB	: The object code of the library (single precision).
AUTLIBD.OLB	: The object code of the library (double precision).
PLAUT.COM	: The command file for the interactive plotting program.
PLAUT.FOR	: The interactive plotting program.
PLAUT.EXE	: The load module of the plotting program.
TCONV.COM	: The command file for single/double conversion.
TCONV.FOR	: A utility program for single/double conversion.
TCONV.EXE	: The load module of the conversion program.
SVAUT.COM	: A command file for saving output files.
RPAUT.COM	: A command file for replacing output files.
APAUT.COM	: A command file for appending output files.
AUTO.TEX	: The TEXsource file of this manual.

2. Notes on Installing AUTO.

2.1 Required software.

AUTO requires the IMSL subroutine EIGRF for computing the eigenvalues of a general real matrix. All calls to EIGRF pass through the subroutine EIG in the section 'General Support Routines' of the library files AUTLIBS and AUTLIBD. If your system does not have the IMSL library, then an alternate eigenvalue solver may have to be substituted. To properly declare IMSL as a library, a small change may be necessary in the command files AUTOS.COM and AUTOD.COM. Note that AUTOS.COM uses IMSL/single precision, while AUTOD.COM uses IMSL/double precision.

The plotting program PLAUT requires the Tektronix PLOT10 software for use on Tektronix graphics terminals. Several graphics terminals from other manufacturers are PLOT10 compatible.

If your system does not have PLOT10 compatible graphics, then it will be necessary to make changes in PLAUT.FOR. However, PLAUT has been written so that adaption to another environment may be quite simple. Only the first few subroutines at the beginning of PLAUT will normally require change. All calls to external graphics package software are channeled through these subroutines. Necessary modifications for use of PLAUT with IBM/GDDM graphics and with CALCOMP and NORPAK have already been indicated in PLAUT.

2.2 Portability.

All programming is in Fortran 77. This version of AUTO has been tested thoroughly on a VAX 750/VMS, an IBM 4341/VM/CMS, and on an FPS 164. However, portability problems may surface on other systems. A good check of proper performance of the package on your system is to run all the example problems described later in this manual. This may also serve as a 'tutorial' on using AUTO.

The double precision version of AUTO assumes the availability of the data type COMPLEX*16 (double precision complex). In addition to the use of a few COMPLEX*16 variables, there are also references to the COMPLEX*16 functions DREAL, DIMAG, CDABS, CDLOG. If your Fortran compiler does not have this data type then appropriate changes will have to be made.

2.3 Output of the example problems.

Due to minor adjustments in new releases of this version, or due to minor system differences, the output produced by the example problems may differ slightly from the output given in this manual. Major differences may indicate a compatibility problem. Contact the author in this case.

Output reproduced in this manual was actually obtained on an IBM 4341 running under VM/CMS, using double precision arithmetic. To get the output on a VAX/VMS system to agree accurately with the output in this manual it is necessary to run the examples in double precision. The program TCONV can be used to automatically convert from single to double precision and conversely. For example, to change AUTPP2.FOR from single precision (as on the tape) to double precision, type @TCONV PP2. Then the first example can be run using double precision by typing @AUTOD PP2 , rather than @AUTOS PP2 , which runs the single precision version.

2.4 Restrictions on problem size.

For certain calculations there are restrictions on the problem size, i.e., on the number of differential or algebraic equations, even though the preprocessor generates the necessary workspace for each application of AUTO.

For one-parameter computations with user-supplied Jacobian there are no restrictions at all. For two-parameter computations and for calculations without Jacobian the problem size may not exceed the value 25.

If necessary the maximum problem size can be increased by changing array dimensions in the COMMON block BLWIF (Interface Workspace) in AUTLIBS and AUTLIBD. The first occurrence of this COMMON block has been documented.

2.5 Compatibility with previous versions.

This version is upwardly compatible with the preceding version of September 1984, with the following exceptions:

(1) The user has to supply an additional subroutine USZR. This addition allows plotting and restart data to be written at selected values of free parameters. See the Documentation Section and the Examples for details.

(2) In connection with the above change, a new constant (NUZR) has been inserted in the COMMON block BLLIM. All user subroutines containing this COMMON block must be modified accordingly.

3. Documentation for AUTO.

3.1 General description of program capabilities.

This program can do a limited bifurcation analysis of algebraic systems of the form

$$F(U, PAR) = 0, \qquad F, U \in R^n, \tag{1}$$

and of systems of ordinary differential equations of the form

$$U'(t) = F(U(t), PAR), \qquad F, U \in R^n. \tag{2}$$

Here PAR denotes one or more free parameters.

It can also do certain continuation and evolution computations for the diffusive system

$$U_t = DU_{xx} + F(U, PAR), \qquad F, U \in R^n, \tag{3}$$

where D denotes a diagonal matrix of diffusion constants.

Specifically the program has the following capabilities for (1):

- ⋆ Trace out branches of solutions and compute the eigenvalues of the Jacobian along these branches.
- ⋆ Locate bifurcation points and automatically compute bifurcating branches.
- ⋆ Locate Hopf bifurcation points and continue these in two parameters.
- ⋆ Locate limit points (= turning points, folds, saddle-node bifurcations) and continue these in two parameters.

* Do all of the above for fixed points of the discrete dynamical system

$$U^{(k+1)} = F(U^{(k)}, PAR).$$

* Find extrema of an objective function along solution branches and successively continue such extrema in more parameters.

For the ordinary differential equation (2) the program has the following capabilities:

* Compute branches of stable or unstable periodic solutions and compute the Floquet multipliers, that determine stability, along these branches. Starting data for the computation of periodic orbits are generated automatically at Hopf bifurcation points.
* Accurately locate limit points, period doubling bifurcations, bifurcations to tori, and ordinary bifurcations along branches of periodic solutions. Branch switching is possible at ordinary and period doubling bifurcations.
* Continue limit points in two parameters. The 2-parameter continuation of orbits of fixed period is also possible. This allows the approximate computation of curves of homoclinic or heteroclinic orbits if the period is large.
* Compute curves of solutions to (2) on $[0, 1]$, subject to general nonlinear boundary or integral conditions. The boundary conditions need not be separated, i.e., they may involve both U(0) and U(1) simultaneously. The side conditions may also depend on parameters. The number of boundary conditions plus the number of integral conditions need not equal the dimension of the differential equation, provided there is a corresponding number of additional parameters.
* Determine limit points and bifurcation points along solution branches to the boundary value problem described above. Branch switching is possible at bifurcation points. Curves of limit points can be computed in two parameters.

For the diffusive system (3) the program can do the following:

- ⋆ Trace out branches of spatially homogeneous solutions. This simply amounts to a bifurcation analysis of the algebraic system (1). However AUTO uses a related system instead, in order to enable the detection of bifurcations to wave train solutions of given wave speed. More precisely, the bifurcations to wave trains are detected as Hopf bifurcations along fixed point branches of the related ordinary differential equations

$$(2') \qquad \begin{aligned} U'(z) &= V(z), \\ V'(z) &= -D^{-1}\, cV(z) + F(U(z), PAR)\ , \end{aligned}$$

  where $z = x - ct$, with the wave speed c specified by the user.

- ⋆ Trace out the branches of wave solutions to (3) that emanate from a Hopf bifurcation point of (2′). The wave speed c is still fixed along such a branch, but the wave length L (= 'period' of solution to (2′)) will normally vary. If the wave length L becomes 'large', i.e., if a homoclinic orbit of (2′) is approached, then the wave tends to a solitary wave solution of (3).
- ⋆ Trace out branches of waves of fixed wave length L in two parameters. For example the wave speed c may be chosen as one of these parameters. If L large, then such a continuation results in a branch of approximate solitary wave solutions to (3).
- ⋆ Do time evolution calculations for (3), given periodic initial data on the interval $[0, L]$. Actually, the initial data must be specified on $[0, 1]$ and L must be set separately because of internal scaling. The spatial domain is then a ring of circumference L. The initial data may be specified analytically or they may be obtained from a previous computation of wave trains, solitary waves, or from a previous evolution calculation. Conversely, if an evolution calculation results in a stationary wave, then this wave can be used as starting data for a wave continuation calculation.

Note that the system (2′) is just a special case of (2) and that its fixed point analysis is a special case of (1). One advantage of the built-in capacity of AUTO to deal with problem (3) is that the user need only specify F, D, and c or L. Another advantage is the compatibility of output data for restart purposes. This allows switching back and forth between evolution calculations and wave computations.

The discretization used throughout is orthogonal collocation with 2,..,7 collocation points per mesh interval. The mesh automatically adapts to the solution so that a measure of the local discretization error is equidistributed. The number of mesh intervals and the number of collocation points remain constant during any given run, although these may be increased (not decreased) at restart points. Time evolution computations of (3) are adaptive in space and in time. Discretization in time is not very accurate: only implicit Euler. Indeed, time integration of (3) has only been included as a convenience and it is not very efficient: Every time step is treated as a boundary value problem.

Three subroutines that take much of the computation time for differential equations have been programmed so that they may benefit from vectorization ('pipelining'). These subroutines are CONPAR, CONRHS, and INFPAR in the section 'Vectorized Subroutines for the Linear Equation Solver BRBD'.

Another subroutine that takes a good proportion of the computation time is SETUBV. It sets up the coefficients of the linear system in Newton's method. Unfortunately in the current design of AUTO this subroutine does not lend itself easily to vectorization.

3.3 Description of output.

AUTO generates four output files:

⋆ The first file, unit 7, contains names and values of user supplied constants as well as the complete bifurcation diagram. This information is written by the subroutines STHD and STPLAE in the section ' Output (Algebraic Problems)', and by STPLBV in the section ' Output (Boundary Value Problems) '.

⋆ The second file, unit 8, contains the data for plotting selected solutions. This file is written by WRTBV8 in the section 'Output (Boundary Value Problems)'. Restart information for the multi-parameter continuation of limit points, Hopf bifurcation points, etc., is also written in unit 8 by WRTSP8 in the section 'Output (Algebraic Problems)'.

⋆ AUTO prints a summary of the computation in unit 6 (typically the user terminal, when in timesharing mode). This information is printed only at solution points where AUTO also writes plotting and restart information in unit 8, i.e., at bifurcation points, limit points, plotting points, and end points. The type of solution point is indicated by the following codes:

EP : End point of branch, normal termination.

MX : End point of branch, abnormal termination (No convergence).

UZ : Zero of a function defined in USZR (Defined below).

BP : Bifurcation point.

LP : Limit point.

HB : Hopf bifurcation point.

PD : Period doubling bifurcation.

TR : Bifurcation to an invariant torus.

⋆ Messages, eigenvalues, Floquet multipliers, and debug output, if any, are written in unit 9. It is often important to inspect the output in unit 9, for example to verify whether the type of bifurcation has been determined correctly.

3.4 Remarks on the detection of bifurcations.

It is important to be aware of the following:

- ⋆ Degenerate (multiple) bifurcations cannot be detected in general. Furthermore, bifurcations that are close to each other may not be noticed when the pseudo-arclength step size (defined below) is not sufficiently small.
- ⋆ Hopf bifurcation points may go unnoticed if no clear crossing of the imaginary axis takes place. This may happen when other real or complex eigenvalues are near the imaginary axis and when the pseudo-arclength step is large compared to the rate of change of the critical eigenvalue pair. An often occuring case is a Hopf bifurcation close to a limit point.
- ⋆ Similarly, Hopf bifurcations may go undetected if switching from real to complex conjugate, followed by crossing of the imaginary axis, occurs rapidly with respect to the pseudo-arclength step size.
- ⋆ Secondary periodic bifurcations may not be detected for very similar reasons.
- ⋆ For periodic solutions one should check the unit 9 output to see when the Floquet multiplier at z=1 is reasonably accurate and whether points labeled as LP, BP, PD, or TR have indeed been diagnosed correctly.

3.5 How to run AUTO.

First compile the user subroutines and load and run these with the object code of the preprocessor AUTPRP(S/D). This preprocessing will result in the main program being written in unit 1. Compile this main program and load and run it with the user subroutines and with the object module of AUTLIB(S/D) and IMSL(S/D) as libraries. Here 'S/D' stands for single/double precision.

In the VAX/VMS version of AUTO the above procedure has already been set up in the command files AUTOS.COM and AUTOD.COM.

3.6 User supplied subroutines.

The user is required to supply the following subroutines. For details on the exact form of user-supplied subroutines see the example problems.

- ⋆ A subroutine FUNC that defines the function F(U,PAR) in (1),(2),(3), and its derivatives with respect to U and PAR.
- ⋆ A subroutine INIT that sets various constants needed by AUTO. All required constants are given default values in the library subroutine DFINIT. The purpose of the user supplied INIT is to modify some or all default values (INIT is called after DFINIT). In most applications several of the constants will require values that differ from the default values. The user then specifies the appropriate common blocks from DFINIT in INIT in order to be able to make the changes. A description of each of the constants can be found below.
- ⋆ A subroutine STPNT that sets the values of the problem parameters in the parameter array PAR and a corresponding solution of (1) or (2) in the array U. The parameter array PAR may also be initialized in INIT. Thus AUTO assumes that an exact solution is known for some choice of parameters. The starting solution should not be a bifurcation point. One need only supply STPNT for a first run: it is not needed for subsequent restarts from computed solutions in unit 8. When starting from a fixed point or steady state, the arguments of STPNT are (NDIM , U , PAR). If one restarts from an analytically known periodic orbit, or from an analytically known solution of a boundary value problem, or from user-interpolated data points approximating such an exact solution, then the arguments of STPNT are (NDIM , U , PAR , T). Here T denotes the independent time (or space) variable, which takes values in the interval $[0, 1]$. See the example problems supplied with AUTO for details. When restarting from an analytically known periodic orbit, one must specify the period in PAR(11).

- ⋆ A subroutine USZR that defines functions of parameters of which zeroes are to be determined. This subroutine can be used to get restart data or plotting data at specified values of the continuation parameter.
- ⋆ For boundary value problems: Subroutines BCND and ICND that define the boundary and integral conditions respectively.
- ⋆ For optimization problems: A subroutine FOPT that defines the objective function.
- ⋆ The IMSL subroutine EIGRF for computing eigenvalues of a general real matrix. EIGRF is called in the subroutine EIG which is located in the section 'General Support Routines'. It may be replaced by an appropriate alternate subroutine that computes the eigenvalues of a general real matrix.

3.7 Listing of COMMON blocks of program constants.

Default values assigned to program constants are listed below. These values may be changed in the user-supplied subroutine INIT, provided that the corresponding COMMON blocks below also appear.

```
COMMON /BLBCN/ NDIM , IPS , IRS , ILP , ICP(20) , PAR(20)
COMMON /BLCDE/ NTST , NCOL , IAD , ISP , ISW , IPLT , NBC , NINT
COMMON /BLTHT/ THETAL(20) , THETAU
COMMON /BLDLS/ DS , DSMIN , DSMAX , IADS
COMMON /BLEPS/ EPSL(20) , EPSU , EPSS
COMMON /BLLIM/ NMX , NUZR , RL0 , RL1 , A0 , A1
COMMON /BLMAX/ NPR , MXBF , IID , ITMX , ITNW , NWTN , JAC
```

3.8 Description of program constants.

Basic Constants (COMMON block BLBCN):

NDIM : Dimension of the system of algebraic or differential equations.

IPS : This important constant defines the type of problem:

IPS=0 : Algebraic bifurcation problem. Hopf bifurcations will not be detected and stability properties will not be indicated in unit 7.

IPS=1 : Algebraic bifurcation problem with detection of Hopf bifurcation points (where the Jacobian DFDU has purely imaginary eigenvalues). The sign of the 'point number' in unit 7 (second integer per output row) will be used to indicate stability: $-$= stable , $+$ = unstable.

IPS=$-$1: Fixed points of the discrete dynamical system $U^{(k+1)} = F(U^{(k)}, PAR)$, with detection of Hopf bifurcations (where DFDU has a complex conjugate pair of eigenvalues of modulus 1). The sign of the 'point number' in unit 7 (second integer per output row) will be used to indicate stability: $-$= stable , $+$ = unstable.

IPS= 2 : Computation of periodic solutions. Starting data may be a Hopf bifurcation point from a previous run with IPS=1, a periodic orbit from a previous calculation with IPS=2 or 3, or an analytically known periodic orbit specified in the user subroutine STPNT. The sign of the 'point number' in unit 7 (second integer per output row) will be used to indicate stability: $-$= stable , $+$ = unstable or unknown.

IPS= 3 : Two-parameter continuation of orbits of fixed period. Starting data may be a periodic orbit from a previous run with IPS=2 or 3, or an analytically known periodic orbit specified in STPNT. The sign of the 'point number' in unit 7 (second integer per output row) will be used to indicate stability: $-$= stable , $+$ = unstable or unknown.

IPS= 4 : A boundary value problem. Integral constraints are also allowed.

IPS= 5 : Algebraic optimization problems.

IPS=11: Spatially uniform solutions of the system (3) with detection of bifurcations to traveling waves. This is similar to the choice IPS=1, but for equations (2') rather than (2). The wave speed must be provided in PAR(10) and a free problem parameter must be designated in ICP(1) (ICP and PAR are described below). PAR(15) through PAR(20) must be used to define the diffusion constants. Note that this limits the dimension to 6.

IPS=12: Continuation of traveling waves. This is similar to the choice IPS=2, but for equations (2') rather than (2). There are two free parameters: the wave length and a problem parameter. AUTO uses PAR(10) for the wave speed and PAR(11) for the wave length. PAR(15) through PAR(20) are used for the diffusion constants. When starting from an analytically known traveling wave, one must specify the wave on the interval [0,1].

IPS=13: Continuation of traveling waves of fixed wave length. This is similar to the choice IPS=3, but for equations (2') rather than (2). Two free parameters must be designated, e.g., the wave speed and a problem parameter. AUTO uses PAR(10) for the wave speed and PAR(11) for the wave length. PAR(15) through PAR(20) are used for the diffusion constants.

IPS=14: Time evolution computation for (3) on a periodic space interval. The initial data must be specified in the interval [0,1]. The actual length of the interval must be specified in PAR(11). AUTO uses PAR(14) for the independent time variable. PAR(15) through PAR(20) are used for the diffusion constants. DS, DSMIN, and DSMAX (described below) govern the stepsize in time: Pseudo-arclength in (U,T) is used. Starting data may be provided analytically in STPNT or as restart labels from a previous run with IPS=12, 13 or 14.

IRS :

IRS=0 : A new problem: No restart data. STPNT must be supplied.

IRS>0 : Computation to be restarted at point with label IRS. The restart information generated by a previous run of AUTO in unit 8 must be attached as unit 3. Some constants may be modified at a restart point, e.g., a a new continuation parameter may chosen, and NTST and NCOL may be changed provided that the product NTST*NCOL is not decreased.

ILP :

ILP=0 : No detection of limit points.

ILP=1 : Accurate location of limit points. Required if 2 parameter continuation of limit points is intended. The choice ILP=1 may give difficulty for 'vertical' branches and near homoclinic orbits.

PAR : An array of problem parameters. The free parameter(s) in PAR must be designated in ICP. The first 9 entries of PAR are always available to the user. The other entries have restricted use: For wave calculations PAR(10) contains the wave speed. For periodic solutions PAR(11) contains the period. For wave calculations PAR(11) contains the wave length. For time integration of diffusive systems PAR(11) contains the length of the (periodic) space interval. For time integration of diffusive systems PAR(14) monitors the independent time variable. For wave calculations and for time integration of diffusive systems PAR(15)-PAR(20) contain the diffusion constants. In certain applications AUTO needs PAR(10)-PAR(20) internally. Thus PAR(10)-PAR(20) should not be used, except for the purposes described above.

ICP : Must be used to designate the free (bifurcation) parameters:

- Often there is only a single free parameter in which case ICP(1) should specify its index, i.e., PAR(ICP(1)) is free.
- For the 2-parameter continuation of limit points and Hopf bifurcation points one should use ICP(2) to indicate the second parameter.
- For boundary value problems for which the number of free parameters NFPAR $\equiv$ NBC + NINT−NDIM +1 is greater than 1, one should set ICP(1),$\cdots$,ICP(NFPAR) to indicate which components of PAR are free.
- For the continuation of limit points for boundary value problems ICP(NFPAR+1) must be used to indicate the extra parameter required for such a continuation.
- For optimization problems one should not set ICP(1). Rather, ICP(2) must be used to designate the free parameter, because PAR(ICP(1)) is used by AUTO to monitor the value of the objective function. Limit points with respect to the objective function are stored as usual in unit 8. These are local extrema of the objective function. A 2-parameter curve of such local extrema can be computed by re-attaching unit 8 to unit 3, setting ICP(3) to indicate the second free parameter, setting IRS to the restart label, and by restarting. Limit points found in this second run are local extrema over two parameters. This procedure can be repeated indefinitely, by successively setting additional free parameters in ICP and by restarting at a limit point found in the preceding run.

Discretization Constants (COMMON block BLCDE):

NTST : The number of mesh intervals to be used for discretization. NTST remains fixed during any particular run. However the location of the mesh points can be made to adapt to the solution by setting the constant IAD.

NCOL : The number of Gauss collocation points per mesh interval, ($2 \leq$ NCOL ≤ 7). NCOL remains fixed during any given run.

IAD :

IAD=0: Fixed mesh (Not recommended). This option can easily result in spurious solutions.

IAD>0: Adapt the mesh every IAD steps along the branch.

ISP :

ISP=0: For all equations: No detection of bifurcation points. For periodic solutions: No Floquet

ISP=1: For algebraic equations: Detection of bifurcations. For differential equations: No detection of bifurcations. For periodic solutions: Floquet multipliers computed.

ISP=2: For all equations: Detection of bifurcations. For periodic solutions: Floquet multipliers computed.

ISP=3: If IPS=2,3 then the choice ISP=3 is similar to ISP=2 except that the first two (rather than one) closest Floquet multipliers to z=1 are excluded from stability and bifurcation considerations. Furthermore, limit points will be determined with respect to the period.

If IPS=2 or 3 then the choice ISP=2,3 should be used with care, due to potential inaccuracy in the computation of the linearized Poincaré map and possible rapid variation of the Floquet multipliers. The linearized Poincaré map always has a multiplier at z=1. If this multiplier becomes inaccurate, then the automatic detection of potential secondary periodic bifurcations (if ISP=2,3) will be discontinued and a warning message will be printed in unit 9.

ISW : Normally ISW=1.

If ISW=−1 and IPS= 2,3 or 4, and if the integer assigned to IRS denotes a bifurcation point in the unit 3 restart file, then the restart procedure will attempt to switch branches rather than continue computation of the given branch. For period doubling bifurcations the value of NTST, as supplied by the user in INIT, must be set to at least twice the value of NTST used in the original computation that detected the period doubling (More accurately: the value of NTST*NCOL should at least double).

If ISW=2 and IPS=−1,0,1,2,3 or 4, and if IRS>0 is the label of a limit point or a Hopf bifurcation point in unit 3, then the program will try to trace out a curve of such points in two parameters. The choice of second parameter must be specified in ICP(2), while ICP(1) remains the index of the first parameter. For boundary value problems with more than one free parameter one should use ICP(NBC+NINT−NDIM+2) to indicate the additional parameter freed for the computation of the curve of limit points.

IPLT : Gives the user some control over the output in unit 7, viz., what is to be written for the second real number per line:

IPLT = 0: L_2 - norm.

0 < IPLT ≤ NDIM: Maximum of the IPLT'th component.

−NDIM ≤ IPLT <0: Minimum of the IPLT'th component.

NDIM < IPLT ≤ 2*NDIM: Integral of the (IPLT−NDIM)'th solution component.

Note that

⋆ L_2-norm and integrals are 'exact' (modulo discretization error).

⋆ For differential equations the maximum and the minimum are only approximate.

NBC : The number of boundary conditions (For boundary value problems only).

NINT : The number of integral conditions (For boundary value problems only).

Definition of Inner Product (COMMON block BLTHT):

THETAL:

THETAU: Certain constants defining inner product and norm in (U,PAR) - space, viz.,

$$\| (U, PAR) \|^2 = \theta_u^2 \int_0^1 U^2 dt + \theta_{\lambda,1}^2 PAR(ICP(1))^2 + \\ + \cdots + \theta_{\lambda,NFPAR}^2 PAR(ICP(NFPAR))^2.$$

THETAL is a vector: one θ_λ for each 'free' parameter. NFPAR denotes the number of free parameters. For periodic solutions (IPS=2) one can include the period in the definition of pseudo-arclength step by setting THETAL(2)=1.0. Note that the default value of THETAL(2) is 0.0, so that the period is not automatically included. The default zero value is often convenient for computation near orbits of infinite period.

Stepsize along Solution Branches (COMMON block BLDLS):

DS : The pseudo-arclength stepsize along a branch in (U,PAR)-space. Note that the settings of THETAU and THETAL influence the definition of 'stepsize'. The sign of DS determines the direction of computation.

DSMIN: The minimum stepsize to be used when IADS>0. (DSMIN>0)

DSMAX: The maximum stepsize to be used when IADS>0. (DSMAX>0)

IADS :

IADS=0: Fixed stepsize along branches.

IADS>0: Adapt the stepsize every IADS steps. If the Newton iteration converges rapidly, then the magnitude of DS will be increased but never exceed DSMAX in absolute value. If a step fails then it will be retried with half the stepsize. This will be done repeatedly until the step is successful or until the absolute value of DS reaches DSMIN. When the stepsize is adaptive, the value assigned to DS is used at the starting or restart point and at the start of every bifurcating branch. See the subroutine ADPTDS for the precise criteria used in

Tolerances (COMMON block BLEPS):

EPSL : Relative tolerance for PAR in Newton's method.

EPSU : Relative tolerance for U in Newton's method.

EPSS : Relative tolerance for arclength in detecting bifurcations.

Limits (COMMON block BLLIM):

NMX : Maximum number of steps to be taken along any branch.

NUZR : If NUZR>0, then zeroes are to be found of functions defined in the user supplied subprogram USZR. The number of functions defined in USZR must equal NUZR. NUZR can not exceed 20 in value.

RL0 : Left limit of the bifurcation diagram for PAR(ICP(1)).

RL1 : Right limit of the bifurcation diagram for PAR(ICP(1)).

A0 : Lower limit of the bifurcation diagram (L_2-norm, or other measure as selected through IPLT).

A1 : Upper limit of the bifurcation diagram (L_2-norm, or other measure as selected through IPLT). The computation of any branch is discontinued when these limits are reached.

Maxima (COMMON block BLMAX):

NPR : Print complete plotting and restart information in unit 8, every NPR steps along a branch of solutions.

MXBF : For algebraic problems only: MXBF sets the maximum number of bifurcating branches to be traced out. Only the first MXBF bifurcating branches will be traced out. All subsequently detected bifurcations will be noted, but the corresponding bifurcating branches will not be computed. If MXBF is negative then these bifurcations will only be traced out in one direction. If MXBF is positive then both directions are traced out.

IID : Controls output printed in unit 9:

IID=0: Minimal output.

IID>0: Additional output printed.

IID=2: Recommended value of IID.

IID>4: If IPS=2,3, or 4: Very extensive output from the linear equation solver BRBD. This option

ITMX : The maximum number of iterations allowed in the accurate location of bifurcation and limit points.

ITNW : The maximum number of iterations allowed in the Newton-Chord method. If this maximum is reached, then the step will be tried again with half the stepsize. This will be repeated until convergence, or until the minimum stepsize is reached. In the latter case the computation of the branch will be discontinued and a message will be printed on unit 9.

NWTN : For differential equations only: After NWTN full Newton iterations the Jacobian is frozen, i.e., the Chord method is used.

JAC : Used to indicate whether derivatives are supplied by the user or whether these are to be obtained by differencing:

JAC=0: No derivatives are given in the user-supplied subroutines.

JAC=1: Derivatives are given in the user-supplied subroutines. The user supplied subroutines concerned are: FUNC, BCND, ICND and FOPT.

3.9 Default values of program constants.

Constant	Default	Constant	Default
NDIM	2	IPS	1
IRS	0	ILP	0
ICP(i)	i	NTST	10
NCOL	4	IAD	3
ISP	1	ISW	1
IPLT	0	NBC	2
NINT	0	THETAL(1)	1.0
THETAL(*)	0.0	PAR(*)	0.0
THETAU	1.0	DS	0.01
DSMIN	0.001	DSMAX	1.0
IADS	1	EPSL	1.0e−4
EPSU	1.0e-4	EPSS	1.0e−4
NMX	100	NUZR	0
RL0	−1.0e+6	RL1	1.0e+6
A0	−1.0e+6	A1	1.0e+6
NPR	20	MXBF	5
IID	2	ITMX	8
ITNW	5	NWTN	3
JAC	1		

4. Example Problem: AUTPP2.
(Hopf Bifurcation Phenomena.)

This is a system of ordinary differential equations that models a 2-species predator-prey model with 'harvesting'. The equations are:

$$u_1' = p_2 u_1(1-u_1) - u_1 u_2 - p_1(1-e^{-p_3 u_1}),$$
$$u_2' = -u_2 + p_4 u_1 u_2.$$

4.1 Listing of AUTPP2.

```
      SUBROUTINE FUNC(NDIM,U,ICP,PAR,IJAC,F,DFDU,DFDP)
C     ---------- ----

C This subroutine evaluates the right hand side of the first order
C system and the derivatives with respect to (U(1),U(2)) and with respect
C to the free parameters.
C Input parameters:
C       NDIM   -   Dimension of U and F.
C       U      -   Vector containing U.
C       PAR    -   Array of parameters in the differential equations.
C       ICP    -   PAR(ICP(1)) is the initial 'free' parameter.
C                  PAR(ICP(2)) is a secondary 'free' parameter,
C                  for subsequent 2-parameter continuations.
C       IJAC   -   =1 if the Jacobians DFDU and DFDP are to be returned,
C                  =0 if only F(U,PAR) is to be returned in this call.

C Values to be returned:
C       F      -   F(U,PAR)  the right hand side of the ODE.
C       DFDU   -   The derivative (Jacobian) with respect to U.
C                  DFDU(i,j) must be given the value of d F(i) / d U(j) .

C       DFDP   -   The derivative with respect to the 'free' parameters:
C                  DFDP(i,ICP(j)) =  d F(i) /d PAR(ICP(j)).
```

```
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION U(NDIM),PAR(20)
      DIMENSION F(NDIM),DFDU(NDIM,NDIM),DFDP(NDIM,20)

       E=DEXP(-PAR(3)*U(1))
CSGLE  E= EXP(-PAR(3)*U(1))

       F(1)=PAR(2)*U(1)*(1-U(1)) - U(1)*U(2) - PAR(1)*(1-E)
       F(2)=-U(2) + PAR(4)*U(1)*U(2)

       IF(IJAC.EQ.0)RETURN

C Derivatives of F(1) with respect to U(1) and U(2).

       DFDU(1,1)=PAR(2)*(1-2*U(1)) - U(2) - PAR(1)*PAR(3)*E
       DFDU(1,2)=-U(1)

C Derivatives of F(2) with respect to U(1) and U(2).

       DFDU(2,1)=PAR(4)*U(2)
       DFDU(2,2)=PAR(4)*U(1) - 1

C Derivatives with respect to PAR(1).

       DFDP(1,1)=E-1
       DFDP(2,1)=0.0

C Derivatives with respect to PAR(2).

       DFDP(1,2)=U(1)*(1-U(1))
       DFDP(2,2)=0.0

      RETURN
      END
```

```
      SUBROUTINE STPNT(NDIM,U,PAR)
C     ---------- -----

C In this subroutine the steady state starting point must be defined.
C (Used when not restarting from a previously computed solution).
C The problem parameters (PAR) may be initialized here or else in INIT.

C       NDIM   -   Dimension of the system of equations.
C       U      -   Vector of dimension NDIM.
C                  Upon return U should contain a steady state solution
C                  corresponding to the values assigned to PAR.
C       PAR    -   Array of parameters in the differential equations.

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION U(NDIM),PAR(20)

C Initialize the problem parameters.

       PAR(1)=0.0
       PAR(2)=3.0
       PAR(3)=5.0
       PAR(4)=3.0

C Initialize the steady state.

       U(1)=0.0
       U(2)=0.0

      RETURN
      END
```

```
      SUBROUTINE INIT
C     ---------- ----

C In this subroutine the user should set those constants that require
C values that differ from the default values assigned in DFINIT.
C (See the main documentation for the default assignments).

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      COMMON /BLBCN/ NDIM,IPS,IRS,ILP,ICP(20),PAR(20)
      COMMON /BLCDE/ NTST,NCOL,IAD,ISP,ISW,IPLT,NBC,NINT
      COMMON /BLDLS/ DS,DSMIN,DSMAX,IADS
      COMMON /BLLIM/ NMX,NUZR,RL0,RL1,A0,A1
      COMMON /BLMAX/ NPR,MXBF,IID,ITMX,ITNW,NWTN,JAC

       NDIM=2    IPS=1    IRS=0    ILP=1

C Set the principal bifurcation parameter to be PAR(1).

       ICP(1)=1

C Set the second free parameter to be PAR(2).
C (For 2-parameter continuation).

       ICP(2)=2

       RL0=0.0    DS=0.02
       RL1=1.2    DSMIN=0.01
       A0=0.0     DSMAX=0.05
       A1=2.0

       NTST=15    ISW=1
       NMX=50     NUZR=0
       NPR=50

      RETURN
      END
```

```
      FUNCTION USZR(I,NUZR,PAR)
C     -------- ----

C This subroutine can be used to obtain plotting and restart data
C at certain values of free parameters.

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION PAR(20)

C Initially, for the steady state analysis, set NUZR=0 in INIT.
C Then the functions specified below will be ignored.

C During the second run of this test problem, when computing the branch
C of periodic solutions, set NUZR=4 in INIT.
C In this example, output will then be written in unit 8 for the values
C of PAR(11) specified below.
C Note that PAR(11) is normally reserved.  It is used by AUTO to keep
C track of the period (See main documentation).

      GOTO(1,2,3,4)I

1     USZR=PAR(11) - 11.0
      RETURN

2     USZR=PAR(11) - 14.0
      RETURN

3     USZR=PAR(11) - 20.0
      RETURN

4     USZR=PAR(11) - 30.0
      RETURN

      END
```

4.2 Steady states and bifurcations.

To determine the steady state solution structure of the example problem in the example file AUTPP2.FOR, simply type

@AUTOS PP2 .

The summary output in unit 6 will be:

BR	PT	TY	LAB	PAR(1)	L2-NORM	U(1)	U(2)
1	1	EP	1	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
1	14	BP	2	6.000000E-01	0.000000E+00	0.000000E+00	0.000000E+00
1	26	EP	3	1.200000E+00	0.000000E+00	0.000000E+00	0.000000E+00

BR	PT	TY	LAB	PAR(1)	L2-NORM	U(1)	U(2)
2	11	BP	4	8.219312E-01	3.334338E-01	3.334338E-01	0.000000E+00
2	13	LP	5	8.329293E-01	4.111538E-01	4.111538E-01	0.000000E+00
2	35	EP	6	-2.548660E-02	1.008371E+00	1.008371E+00	0.000000E+00

BR	PT	TY	LAB	PAR(1)	L2-NORM	U(1)	U(2)
2	47	EP	7	7.029627E-04	2.036092E+00	-2.036092E+00	0.000000E+00

BR	PT	TY	LAB	PAR(1)	L2-NORM	U(1)	U(2)
3	11	HB	8	6.715938E-01	4.948663E-01	3.333333E-01	3.657615E-01
3	46	EP	9	6.406314E-03	2.012212E+00	3.333333E-01	1.984411E+00

BR	PT	TY	LAB	PAR(1)	L2-NORM	U(1)	U(2)
3	23	EP	10	1.213453E+00	1.009410E+00	3.333333E-01	-9.527837E-01

Above BR = branch number, PT = the number of steps taken along the branch, TY = the type of point (see the main documentation for an explanation of the codes used under this heading), LAB = the label of the solution point (every labeled point has restart information written in unit 8).

The output in unit 7 is similar to that in unit 6 printed above, but much more extensive: *every* solution point has a corresponding line printed in unit 7.

To save the unit 7,8,9 output files under the names P.PP2, Q.PP2, and D.PP2, respectively, type

@SVAUT PP2 .

In order to further compute the branch that terminates at label 10 in the above computation, make the following changes in the user-supplied subroutine INIT:

IRS	from	0	to	10	(restart label),
RL1	from	1.2	to	2.0	(diagram limit),

and then type

@AUTOS PP2 .

Now the output will be

```
BR    PT TY LAB     PAR(1)         L2-NORM          U(1)           U(2)
3    24 EP  11  1.642974E+00  2.025584E+00  3.333333E-01 -1.997969E+00
```

Append the new unit 7,8, and 9 output files to the previous output files, stored as P.PP2, Q.PP2, and D.PP2, respectively, by typing:

@APAUT PP2 .

Normally one will use the interactive plotting program PLAUT after each run in order to graphically inspect the results obtained so far. In this example, however, we shall illustrate use of PLAUT at the end of Section 4.5.

4.4 Periodic solutions from a Hopf bifurcation.

To compute the branch of periodic solutions that emanates from the Hopf bifurcation point found in Section 4.2 above, make the following changes in INIT:

IRS	from	10	to	8	(restart label),
IPS	from	1	to	2	(periodic solutions),
NUZR	from	0	to	4	(user defined output points),
ILP	from	1	to	0	(turn off detection of limit points).

Restart by typing

@AUTOS PP2 .

The output will be

BR	PT	TY	LAB	PAR(1)	L2-NORM	MAX U(1)	MAX U(2)	PERIOD
8	5	UZ	12	6.765300E-01	4.904248E-01	4.309411E-01	5.108366E-01	1.100002E+01
8	8	UZ	13	6.921120E-01	4.714642E-01	5.519356E-01	5.960594E-01	1.399913E+01
8	11	UZ	14	7.037223E-01	4.504964E-01	6.224614E-01	5.933895E-01	2.000027E+01
8	13	UZ	15	7.075787E-01	4.381730E-01	6.474360E-01	5.821317E-01	3.000016E+01
8	17	MX	16	7.080557E-01	4.244152E-01	6.518259E-01	5.820363E-01	1.097409E+02

In the above output UZ denotes a zero of one of the functions in the user-supplied subroutine USZR. Here these zeroes correspond to certain values of the period at which plotting and restart information is wanted.

The code MX denotes non-convergence: In this example the periodic orbit terminates in a heteroclinic orbit, so that the continuation will necessarily fail at some point while approaching the infinite period endpoint. (Homoclinic orbits can often be computed to very large period, but heteroclinic orbits are normally more difficult to approach.)

Append the new output file to the accumulated previous output:

@APAUT PP2 .

4.5 Restarting at a computed periodic solution.

In order to approach the heteroclinic orbit in the preceding section more closely, we can restart the computation at a computed orbit using increased accuracy. To do this, make the following changes in INIT:

IRS	from	8	to	15	(restart label),
NTST	from	15	to	25	(number of mesh intervals).

Suppose we want output at period 150. For this purpose make a change in USZR, e.g., change the string

PAR(11)−30.0 to PAR(11)−150.0

Restart by typing

@AUTOS PP2 .

The new output is:

BR	PT	TY	LAB	PAR(1)	L2-NORM	MAX U(1)	MAX U(2)	PERIOD
8	20	UZ	17	7.080728E-01	4.248802E-01	6.518227E-01	5.823078E-01	1.499747E+02
8	26	MX	18	7.080727E-01	4.243443E-01	6.519776E-01	5.822629E-01	2.316212E+02

Again, the computation must fail when the period becomes too large. Note that the period at the end point (label 18) is now 231.6, as compared to 109.7 at the end point (label 16) in Section 4.4.

Append the new output file to the accumulated previous output:

@APAUT PP2 .

4.6 An example of using the plotting program PLAUT.

Now use the program PLAUT to plot the output accumulated in the output files P.PP2 and Q.PP2 (Normally one will graphically inspect the output at the end of each run). To run the interactive graphics program, type

@PLAUT PP2 .

Type HELP for information. Type D3 to select a convenient 'default option', which avoids a lot of questions being asked. Type DP ('differential plot') to indicate that you want the bifurcation diagram to show the stability properties of the solutions. Type SY ('symbols') to get special symbols to mark bifurcation points, etc. Type AX to select the axes to be plotted. Upon prompting, enter 1 and 3 to indicate that you want to plot the first column of real numbers versus the third column of real numbers from unit 7. As seen from the output given above (and from the contents of unit 7) this choice will result in a plot with the bifurcation parameter PAR(1) as horizontal axis, and the maximum of the first solution component as vertical axis. Now type BD0 to actually get the plot. After it has appeared you can make a blow-up by typing BD. You will be asked to enter limits. Enter, e.g., 0.5 0.9 0.3 0.8 . The blow-up will appear on the screen.

To plot some periodic orbits type 2D. You will be asked for labels of orbits to be plotted. Enter,

want to plot the first solution component versus time (1 3 will give the second component versus time, 2 3 will give the phase diagram). To get the plot type D ('draw'). After the plot appear you will be asked if you want to plot more axes. E.g., type 2 3 to get the phase diagram. Type D to draw it. After viewing the plot type EX to exit from the 2D mode. Type END to exit from PLAUT.

4.7 A 2-parameter curve of steady state limit points.

To compute the 2-parameter curve of limit points that passes through the limit point (label 5) found above in Section 4.2, make the following changes in INIT:

IRS	from	15	to	5	(restart label),
IPS	from	2	to	1	(algebraic equations),
ISW	from	1	to	2	(2-parameter continuation).

Start the 2-parameter computation by typing

@AUTOS PP2 .

The output is:

```
BR    PT TY LAB    PAR(1)        L2-NORM         U(1)          U(2)          PAR(2)
5    50 EP  19  1.483011E+00  4.111616E-01  4.111616E-01  0.000000E+00  5.341430E+00
```

Normally one will save and plot the output files.

4.8 A 2-parameter curve of Hopf bifurcation points.

To compute the 2-parameter curve of Hopf bifurcation points that passes through the Hopf bifurcation point (label 8) found above in Section 4.2, make the following change in INIT:

IRS from 5 to 8 (restart label),

Start the 2-parameter computation by typing

@AUTOS PP2 .

The output is:

```
BR   PT TY LAB    PAR(1)        L2-NORM         U(1)          U(2)         PAR(2)
8    50 EP  19  1.190073E+00  7.288272E-01  3.333333E-01  6.481342E-01  5.316039E+00
```

4.9 A 2-parameter curve of fixed-period orbits.

Suppose we want to compute the 2-parameter curve of orbits of period 30.0 that passes through the output point with label 15 found above in Section 4.4. To do this make the following changes in INIT:

IRS from 8 to 15 (restart label),

IPS from 1 to 3 (fixed period).

Start the 2-parameter computation by typing

@AUTOS PP2 .

The output is:

```
BR   PT TY LAB    PAR(1)        L2-NORM       MAX U(1)      MAX U(2)      PAR(2)
15   28 EP  19  2.000082E+00  5.743138E-01  6.627835E-01  1.221364E+00  8.625596E+00
```

Suppose we no longer need the old files P.PP2, Q.PP2, and D.PP2. Then we can replace them by the newly generated output, by typing

@RPAUT PP2 .

4.10 Restarting a 2-parameter continuation.

If, e.g., we want to continue the branch of fixed period orbits further, then the following changes can be made in INIT:

IRS	from	15	to	19	(restart label),
RL1	from	2.0	to	2.5	(increase diagram limit).

Start the 2-parameter computation by typing

@AUTOS PP2 .

The output is:

```
BR   PT TY LAB    PAR(1)        L2-NORM       MAX U(1)      MAX U(2)      PAR(2)
15   12 EP  20  2.503168E+00 6.373378E-01 6.654198E-01 1.481185E+00 1.084119E+01
```

Append the unit 7,8, and 9 output files to P.PP2, Q.PP2, and D.PP2 respectively, by typing

@APAUT PP2 .

4.11 Another example of using PLAUT.

To get a plot of the 2-parameter curve of orbits of fixed period accumulated in the plotting files P.PP2 and Q.PP2, type

@PLAUT PP2 .

Type D3 to simplify use of PLAUT. Type AX, and upon prompting for axes, type 1 5. This selects PAR(1) as horizontal axis and PAR(2) as vertical axis, since these are the values printed on the first and fifth column of reals in unit 7 (and in unit 6, as can be seen in the output above). To actually get the plot type BD0. Type END to terminate plotting.

5. Example Problem: AUTEXP.
(A Boundary Value Problem.)

This is a two point boundary value problem of the form

$$u_1' = u_2,$$
$$u_2' = -p_1 e^{u_1},$$

with boundary conditions $u_1(0) = 0, \quad u_1(1) = 0.$

5.1 Listing of AUTEXP.

```
      SUBROUTINE FUNC(NDIM,U,ICP,PAR,IJAC,F,DFDU,DFDP)
C     ---------- ----

C This subroutine evaluates the right hand side of the first order
C system and the derivatives with respect to (U(1),U(2)) and with respect
C to PAR(1)  (For documentation see the example problem AUTPP2).

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION U(NDIM),PAR(20),F(NDIM),DFDU(NDIM,NDIM),DFDP(NDIM,20)

       E=EXP(U(1))

       F(1)=U(2)
       F(2)=-PAR(1)*E

      IF(IJAC.EQ.0)RETURN

       DFDU(1,1)=0.0          DFDU(1,2)=1
       DFDU(2,1)=-PAR(1)*E    DFDU(2,2)=0.0

       DFDP(1,1)=0.0          DFDP(2,1)=-E

      RETURN
      END
```

```
      SUBROUTINE STPNT(NDIM,U,PAR,T)
C     ---------- -----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

C This subroutine must used to generate an initial starting point
C (i.e., when not restarting from a previously computed solution).
C The solution vector U must be given as a function of the independent
C variable T. T takes on values between zero and one.

C       NDIM   -   Dimension of the system of differential equations.
C       U      -   Vector of dimension NDIM.
C                  Upon return, U must contain a solution of the
C                  differential equation evaluated at 'time' T.
C       PAR    -   Array of parameters in the differential equations.
C                  These may be initialized here, or else in INIT.
C                  (STPNT is called after INIT).
C       T      -   Contains a value of the independent variable in [0,1]
C                  where the solution is to be evaluated.

C (In this example PAR is initialized in INIT.)

      DIMENSION U(NDIM),PAR(20)

C (In this problem the starting solution is actually independent of T.)

       U(1)=0.0
       U(2)=0.0

      RETURN
      END
```

```
      SUBROUTINE INIT
C     ---------- ----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      COMMON /BLBCN/ NDIM,IPS,IRS,ILP,ICP(20),PAR(20)
      COMMON /BLCDE/ NTST,NCOL,IAD,ISP,ISW,IPLT,NBC,NINT
      COMMON /BLDLS/ DS,DSMIN,DSMAX,IADS
      COMMON /BLLIM/ NMX,NUZR,RL0,RL1,A0,A1
      COMMON /BLMAX/ NPR,MXBF,IID,ITMX,ITNW,NWTN,JAC

C In this subroutine the user should set those constants that require
C values that differ from the default values assigned in DFINIT.
C (See the main documentation for the default assignments).

C Initialize the continuation parameter.
      PAR(1)=0.0

      NDIM=2       IPS=4        IRS=0       ILP=1
      ICP(1)=1     NTST=5       NBC=2       NINT=0
      NPR=50       JAC=1        NMX=50      NUZR=2
      RL0=0.0      RL1=4.0      A0=0.0      A1=50.0
      DS=0.01      DSMIN=0.001  DSMAX=1.0

      RETURN
      END
```

```
      SUBROUTINE BCND(NDIM,PAR,ICP,NBC,U0,U1,FB,IJAC,DBC)
C     ---------- ----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

C This subroutine defines the boundary conditions.

C Supplied variables:

C       NDIM    :   Dimension of the first order autonomous system.
C       PAR     :   Vector of problem parameters.
C       ICP     :   Vector of indices of the free parameters.
C       NBC     :   Number of boundary conditions.
C       U0      :   Value of the vector function U at t=0.
C       U1      :   Value of the vector function U at t=1.
C       IJAC    :   IJAC=0   :      Derivatives need not be returned.
C                   IJAC=1   :      Derivatives must be returned also.

C Variables to be returned upon completion of call:

C       FB      :   The vector function defining the boundary conditions:

C                               FB ( U0 , U1 , PAR) = 0.

C       DBC     :   The Jacobian of the boundary conditions:

C                   DBC(i,j) :  The derivative of the i'th boundary
C                    condition with respect to the j'th component of U0.

C                   DBC(i,NDIM+j) :     The derivative of the i'th boundary
C                   condition with respect to the j'th component of U1.

C                   DBC(i,2*NDIM+j) :    The derivative of the i'th boundary
C                   condition with respect to PAR(j) (For free parameters).
```

```
      DIMENSION PAR(20),ICP(20),U0(NDIM),U1(NDIM),FB(NBC),DBC(NBC,20)

C Define the two boundary conditions:

       FB(1)=U0(1)
       FB(2)=U1(1)

      IF(IJAC.EQ.0)RETURN

C Set up the derivatives of the boundary conditions:

C With respect to U0   (U0 = U at 'time' T=0).

       DBC(1,1)=1.0      DBC(1,2)=0.0
       DBC(2,1)=0.0      DBC(2,2)=0.0

C With respect to U1   (U1 = U at 'time' T=1).

       DBC(1,3)=0.0      DBC(1,4)=0.0
       DBC(2,3)=1.0      DBC(2,4)=0.0

C With respect to the free parameter (Here PAR(1) ).

       DBC(1,5)=0.0
       DBC(2,5)=0.0

      RETURN
      END
```

```
      SUBROUTINE ICND(NDIM,PAR,ICP,NINT,U,UOLD,UDOT,UPOLD,FI,IJAC,DINT)
C     ---------- ----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

C (This problem has no integral constraints.)

      RETURN
      END

       FUNCTION USZR(I,NUZR,PAR)
C      -------- ----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION PAR(20)

C This subroutine makes it possible to generate plotting and restart
C data at user-selected values of the free parameter(s).
C Plotting and restart data will be written in unit 8 at zeroes of
C functions defined below.

C In this example 2 functions must be defined, because NUZR has been
C given the value 2 in INIT.

      GOTO(1,2)I

1     USZR=PAR(1)-1.0
       RETURN

2     USZR=PAR(1)-3.0
       RETURN

      END
```

5.2 Continuation of a solution curve.

To run the example problem in the file AUTEXP.FOR, type

@AUTOS EXP .

The output will be

BR	PT	TY	LAB	PAR(1)	L2-NORM	MAX U(1)	MAX U(2)
1	1	EP	1	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
1	9	UZ	2	9.999996E-01	3.388826E-01	1.404988E-01	5.493525E-01
1	12	UZ	3	3.000000E+00	1.521270E+00	6.401164E-01	2.319603E+00
1	14	LP	4	3.513831E+00	2.781940E+00	1.186263E+00	4.000000E+00
1	16	UZ	5	3.000000E+00	4.554630E+00	1.975141E+00	6.103383E+00
1	22	UZ	6	9.999999E-01	9.157868E+00	4.089818E+00	1.084694E+01
1	50	EP	7	2.239591E-05	3.677775E+01	1.723320E+01	3.720814E+01

Note that output is written at zeroes of the two functions specified in the user supplied subroutine USZR. These points are marked UZ in the output above. The output marked LP denotes a limit point on the solution branch.

Save the output files under the names P.EXP, Q.EXP, and D.EXP respectively, by typing

@SVAUT EXP .

5.3 Restarting at a computed solution.

Suppose we want to continue the above branch of solutions further, using increased accuracy. For this, make the following changes in INIT:

IRS	from	0	to	7	(restart label),
NTST	from	5	to	20	(increase accuracy),
A1	from	50	to	200	(diagram limit),
DSMAX	from	1.0	to	10.0	(maximum step size),

and then type

@AUTOS EXP .

Now the output will be

```
BR    PT TY LAB     PAR(1)          L2-NORM         MAX U(1)        MAX U(2)
1    50 EP    8  1.187867E-22  1.245716E+02  5.939173E+01  1.215573E+02
```

Append the new output files (units 7,8,9) to the previous output files by typing

@APAUT EXP .

5.4 Another example of using PLAUT.

To plot the results of the computations run the plotting program by, typing

@PLAUT EXP .

Type D3 to set a convenient default option. Then type BD0 to get the bifurcation diagram (These two commands may be concatenated: D3BD0). Since the axes were not specified, PLAUT will choose the first two columns of real numbers from unit 7. This will result in the L_2-norm being plotted versus PAR(1). Labels 1 through 8 will also appear in the diagram.

To plot the first solution component at the labeled points, type 2D. Upon prompting, enter A (all). Thereafter enter D (draw). To exit from the 2D option, type EX. Type END to exit from PLAUT.

5.5 Continuation without Jacobian.

To compute without Jacobian in the example problem AUTEXP.FOR, make the following change in INIT:

JAC from 1 to 0 (no Jacobian).

In this example, the derivatives have actually been specified in the user subroutines FUNC, BCND, and ICND. But with the above change the derivatives could actually have been omitted. This is often convenient for quick runs. Moreover, it is very easy to make a mistake in setting up a Jacobian. This further favors the choice JAC=0. However, computer time will normally increase, and convergence problems may occur more readily.

6. Example Problem: AUTINT.
(A BVP with Integral Constraint.)

The example problem contained in the file AUTINT.FOR consists of two coupled differential equations

$$u_1' = u_2,$$

$$u_2' = -p_1 e^{u_1},$$

with a non-separated boundary condition and an integral constraint:

$$u_1(0) - u_1(1) - p_2 = 0, \qquad \int_0^1 u(t)dt - p_3 = 0.$$

6.1 Listing of AUTINT.

```
      SUBROUTINE FUNC(NDIM,U,ICP,PAR,IJAC,F,DFDU,DFDP)
C     ---------- ----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION U(NDIM),PAR(20),F(NDIM),DFDU(NDIM,NDIM),DFDP(NDIM,20)

       E=EXP(U(1))

       F(1)=U(2)
       F(2)=-PAR(1)*E

      IF(IJAC.EQ.0)RETURN

       DFDU(1,1)=0.0            DFDU(1,2)=1
       DFDU(2,1)=-PAR(1)*E      DFDU(2,2)=0.0

       DFDP(1,1)=0.0            DFDP(2,1)=-E

      RETURN
      END
```

```
      SUBROUTINE STPNT(NDIM,U,PAR,T)
C     ---------- -----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION U(NDIM),PAR(20)

       U(1)=0.0
       U(2)=0.0

      RETURN
      END

      SUBROUTINE INIT
C     ---------- ----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      COMMON /BLBCN/ NDIM,IPS,IRS,ILP,ICP(20),PAR(20)
      COMMON /BLCDE/ NTST,NCOL,IAD,ISP,ISW,IPLT,NBC,NINT
      COMMON /BLDLS/ DS,DSMIN,DSMAX,IADS
      COMMON /BLLIM/ NMX,NUZR,RL0,RL1,A0,A1
      COMMON /BLMAX/ NPR,MXBF,IID,ITMX,ITNW,NWTN,JAC

C Initialize the problem parameters.
       PAR(1)=0.0
       PAR(2)=0.0
       PAR(3)=0.0

C Designate the free parameter to be PAR(1).

       ICP(1)=1

C Designate the second free parameter to be PAR(2).
C (For 2-parameter continuation).

       ICP(2)=2
```

```
C Set the number of boundary conditions and integral conditions.
      NBC=1         NINT=1

      NDIM=2        IPS=4         IRS=0         ISW=1
      NTST=5        NMX=40        NPR=40        NUZR=2
      DS=0.01       DSMIN=0.001   DSMAX=2.0     ILP=1
      RL0=0.0       RL1=20.0      A0=0.0        A1=100.0

      RETURN
      END

      SUBROUTINE BCND(NDIM,PAR,ICP,NBC,U0,U1,FB,IJAC,DBC)
C     ---------- ----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION PAR(20),ICP(20),U0(NDIM),U1(NDIM),FB(NBC),DBC(NBC,1)

C Define the boundary condition.

      FB(1)=U0(1)-U1(1)-PAR(2)

      IF(IJAC.EQ.0)RETURN

C Set the derivatives of the boundary condition.

C With respect to U0(1) and U0(2) (i.e., at t=0).
      DBC(1,1)=1.0      DBC(1,2)=0.0

C With respect to U1(1) and U1(2) (i.e., at t=1).
      DBC(1,3)=-1.0     DBC(1,4)=0.0

C With respect to PAR(1), PAR(2) and PAR(3).
      DBC(1,5)=0.0      DBC(1,6)=-1.0       DBC(1,7)=0.0

      RETURN
      END
```

```
      SUBROUTINE ICND(NDIM,PAR,ICP,NINT,U,UOLD,UDOT,UPOLD,FI,IJAC,DINT)
C     ---------- ----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION U(NDIM),UOLD(NDIM),UDOT(NDIM),UPOLD(NDIM)
      DIMENSION FI(NINT),DINT(NINT,1),ICP(20),PAR(20)

C Supplied variables:

C     NDIM   :  Dimension of the first order autonomous system.
C     PAR    :  Vector of problem parameters.
C     ICP    :  Vector of indices of the free parameters.
C     NINT   :  Number of integral conditions.
C     U      :  Value of the vector function U at 'time' t.

C The following 3 arguments are not normally needed (All 3 are evaluated
C                at the preceding point on the solution branch):
C     UOLD   :  Value of the vector function U at 'time' t.
C     UDOT   :  Derivative of U at 'time' t with respect to arclength.
C     UPOLD  :  Derivative of U at 'time' t with respect to t.

C     IJAC   :  IJAC=0   :      Derivatives need not be returned.
C               IJAC=1   :      Derivatives must be returned also.

C Variables to be returned upon completion of call:

C     FI     :  The vector integrand FI of the integral conditions

C                          1
C                  integral   FI( U(t) , PAR ) dt = 0 .
C                          0

C     DINT   :  The Jacobian of the integrand:
C               DINT(i,j)     :    Derivative of the i'th integrand
C                         with respect to the j'th component of U.
C               DINT(i,NDIM+j):    Derivative of the i'th integrand
C                         with respect to PAR(j) (For free parameters).
```

```
C Evaluate the integrand (Only one integral constraint in this example).

      FI(1)=U(1)-PAR(3)

      IF(IJAC.EQ.0)RETURN

C Evaluate the derivatives of the integrand.

C With respect to U(1) and U(2).
      DINT(1,1)=1.0       DINT(1,2)=0.0

C With respect to PAR(1), PAR(2) and PAR(3).
      DINT(1,3)=0.0       DINT(1,4)=0.0       DINT(1,5)=-1.0

      RETURN
      END

       FUNCTION USZR(I,NUZR,PAR)
C      -------- ----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION PAR(20)

      GOTO(1,2)I

1     USZR=PAR(1)-1.0
       RETURN

2     USZR=PAR(1)-3.0
       RETURN

      END
```

6.2 Continuation of solutions.

To run the example problem in the file AUTINT.FOR, type

@AUTOS INT .

The output will be

BR	PT	TY	LAB	PAR(1)	L2-NORM	MAX U(1)	MAX U(2)
1	1	EP	1	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
1	9	UZ	2	1.000087E+00	2.962759E-01	4.290991E-02	5.004057E-01
1	12	UZ	3	2.999996E+00	9.261274E-01	1.377196E-01	1.510763E+00
1	18	LP	4	1.192384E+01	7.164968E+00	1.292456E+00	9.145748E+00
1	27	UZ	5	2.999998E+00	2.095950E+01	4.438223E+00	2.281422E+01
1	31	UZ	6	9.999938E-01	2.758932E+01	6.045403E+00	2.932943E+01
1	40	EP	7	3.250848E-02	4.542328E+01	1.041347E+01	4.695709E+01

Note the detection of a limit point (LP) and output at zeroes of the functions in USZR (UZ). Save the output as usual.

6.3 A 2-parameter curve of limit points.

To compute the 2-parameter curve of limit points that passes through the limit point (label 4) found above in Section 6.2, make the following changes in INIT:

IRS	from	0	to	4	(restart label),
ISW	from	1	to	2	(2-parameter continuation).

Start the 2-parameter computation by typing

@AUTOS INT .

The output is:

BR	PT	TY	LAB	PAR(1)	L2-NORM	MAX U(1)	MAX U(2)	PAR(2)
4	23	UZ	8	3.000003E+00	1.237293E+01	3.493221E+00	1.274570E+01	7.909235E+00
4	26	UZ	9	9.999998E-01	1.582219E+01	4.998711E+00	1.553697E+01	1.131937E+01
4	40	EP	10	4.944531E-05	4.137523E+01	1.665819E+01	3.801577E+01	3.543773E+01

It is possible to continue the above 2-parameter branch of limit points further by setting IRS to the desired restart label. One can also switch back to 1-parameter continuation if in addition one also resets ISW to 1.

7. Example Problem: AUTDD2.
(A Discrete Dynamical System).

The example problem in the file AUTDD2.FOR is a 2-species discrete predator-prey model of the form

$$u_1^{(k+1)} = p_1 u_1^{(k)}(1 - u_1^{(k)}) - p_2 u_1^{(k)} u_2^{(k)},$$
$$u_2^{(k+1)} = (1 - p_3) u_2^{(k)} + p_2 u_1^{(k)} u_2^{(k)}.$$

7.1 Listing of AUTDD2.

```
      SUBROUTINE FUNC(NDIM,U,ICP,PAR,IJAC,F,DFDU,DFDP)
C     ---------- ----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION U(NDIM),PAR(20)
      DIMENSION F(NDIM),DFDU(NDIM,NDIM),DFDP(NDIM,20)

C Set up the vector function F.

       F(1)=PAR(1)*U(1)*(1-U(1)) - PAR(2)*U(1)*U(2)
       F(2)=(1-PAR(3))*U(2) + PAR(2)*U(1)*U(2)

      IF(IJAC.EQ.0)RETURN

C Set up the derivatives of F.

       DFDU(1,1)=PAR(1)*(1-2*U(1))-PAR(2)*U(2)   DFDU(1,2)=-PAR(2)*U(1)
       DFDU(2,1)=PAR(2)*U(2)           DFDU(2,2)=1-PAR(3) + PAR(2)*U(1)

       DFDP(1,1)=U(1)*(1-U(1))       DFDP(2,1)=0.0
       DFDP(1,2)=-U(1)*U(2)          DFDP(2,2)= U(1)*U(2)

      RETURN
      END
```

```
      SUBROUTINE STPNT(NDIM,U,PAR)
C     ---------- -----

C In this subroutine starting values for PAR and a corresponding
C fixed point U must be given, i.e., U must satisfy:  U = F(U,PAR).
C (Used when not restarting from a previously computed solution).
C The problem parameters (PAR) may also be initialized in INIT.

C       NDIM   -   Dimension of F and U.
C       U      -   Vector of dimension NDIM.
C                  Upon return, U should contain a fixed point of F
C                  corresponding to the values assigned to PAR.
C       PAR    -   Array of parameters in the differential equations.


      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION U(NDIM),PAR(20)

C Set the problem parameters.

       PAR(1)=0.0
       PAR(2)=0.2
       PAR(3)=0.1

C Define the fixed point.

       U(1)=0.0
       U(2)=0.0

      RETURN
      END
```

```
      SUBROUTINE INIT
C     ---------- ----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      COMMON /BLBCN/ NDIM,IPS,IRS,ILP,ICP(20),PAR(20)
      COMMON /BLCDE/ NTST,NCOL,IAD,ISP,ISW,IPLT,NBC,NINT
      COMMON /BLDLS/ DS,DSMIN,DSMAX,IADS
      COMMON /BLLIM/ NMX,NUZR,RL0,RL1,A0,A1
      COMMON /BLMAX/ NPR,MXBF,IID,ITMX,ITNW,NWTN,JAC

       IPS=-1       NDIM=2       ICP(1)=1      ICP(2)=2
       IRS=0        ISW=1        ILP=0         NMX=100
       RL0=0.0      RL1=5.0      A0=-1.0       A1=5.0
       DS=0.02      DSMIN=0.01   DSMAX=0.1     NPR=100

      RETURN
      END

       FUNCTION USZR(I,NUZR,PAR)
C      -------- ----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION PAR(20)

C This example has no functions of which zeroes are to be determined.
C (Since the default assignment NUZR=0 is used).

       USZR=0.0

      RETURN
      END
```

7.2 Fixed points and bifurcations.

To run the example problem in AUTDD2.FOR, type

@AUTOS DD2 .

The output will be

```
BR    PT TY LAB     PAR(1)        L2-NORM         U(1)          U(2)
1      1 EP   1  0.000000E+00  0.000000E+00  0.000000E+00  0.000000E+00
1     13 BP   2  1.000000E+00  0.000000E+00  0.000000E+00  0.000000E+00
1     53 EP   3  5.000001E+00  0.000000E+00  0.000000E+00  0.000000E+00

BR    PT TY LAB     PAR(1)        L2-NORM         U(1)          U(2)
2     17 BP   4  1.999975E+00  4.999934E-01  4.999934E-01  0.000000E+00
2     28 HB   5  2.999995E+00  6.666661E-01  6.666661E-01  0.000000E+00
2     49 EP   6  5.095086E+00  8.037327E-01  8.037327E-01  0.000000E+00

BR    PT TY LAB     PAR(1)        L2-NORM         U(1)          U(2)
2     57 EP   7  1.649611E-01  5.058569E+00 -5.058569E+00  0.000000E+00

BR    PT TY LAB     PAR(1)        L2-NORM         U(1)          U(2)
3     58 EP   8  4.017190E+00  5.067700E+00  5.000001E-01  5.042973E+00

BR    PT TY LAB     PAR(1)        L2-NORM         U(1)          U(2)
3     58 EP   9 -1.718925E-02  5.067700E+00  5.000001E-01 -5.042973E+00
```

Note that a Hopf bifurcation (HB) has been detected. In this case it does not define the bifurcation of an invariant circle, but rather it is a period doubling bifurcation. This can be verified by inspecting the eigenvalues of the Jacobian which are printed in unit 9. Branch switching at Hopf bifurcations has not been implemented (To deal with a period doubling one can of course set up the subroutine FUNC for the second iterate of the map). However, it is possible to compute a 2-parameter curve of Hopf bifurcations. This is done in the following section.

The program PLAUT can be used to plot the bifurcation diagram, showing the stability properties of the solutions. Use of PLAUT in this application is very similar to its use for the example problem AUTPP2.FOR in Section 4.6.

To save the unit 7, 8, and 9 output files from the above run under the names P.DD2, Q.DD2, and D.DD2 respectively, type

@SVAUT DD2 .

7.3 A 2-parameter curve of Hopf bifurcation points.

To compute the 2-parameter curve of Hopf bifurcation points that passes through the Hopf bifurcation point (label 5) found above in Section 7.2, make the following changes in INIT:

IRS	from	0	to	5	(restart label),
ISW	from	1	to	2	(2-parameters).

Start the 2-parameter computation by typing

@AUTOS DD2 .

The output is:

```
BR   PT TY LAB    PAR(1)         L2-NORM         U(1)           U(2)          PAR(2)
5  100 EP  10  3.000000E+00  6.666667E-01  6.666667E-01 -7.664940E-34  9.960003E+00
```

In this simple example the period doubling bifurcation turns out to be independent of PAR(2), i.e., PAR(1) is constant along the branch.

Limit points on a branch of fixed points of a discrete dynamical system can also be continued in two parameters. The procedure for doing this is identical to the corresponding procedure for Hopf bifurcations illustrated above.

8. Example Problem: AUTOPT.
(Algebraic Optimization Problem.)

This example problem illustrates a simple optimization problem: Find the maximum sum of coordinates on the unit sphere in R^5. Coordinate 1 is treated as the state variable. Coordinates 2-5 are treated as control parameters.

8.1 Listing of AUTOPT.

```
      SUBROUTINE FUNC(NDIM,U,ICP,PAR,IJAC,F,DFDU,DFDP)
C     ---------- ----

C Defines the 'State Equation' F(U,PAR)=0.

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION U(NDIM),ICP(20),PAR(20)
      DIMENSION F(NDIM),DFDU(NDIM,NDIM),DFDP(NDIM,20)

C The 'state variable'.
       X1=U(1)

C The 'control variables'.
       X2=PAR(1)       X3=PAR(2)       X4=PAR(3)       X5=PAR(4)

C The 'state function' (Only one in this example).
       F(1)=X1*X1 + X2*X2 + X3*X3 + X4*X4 + X5*X5 - 1

      IF(IJAC.EQ.0)RETURN

C The derivatives of the state function.
       DFDU(1,1)=2*X1
       DFDP(1,1)=2*X2       DFDP(1,2)=2*X3
       DFDP(1,3)=2*X4       DFDP(1,4)=2*X5

      RETURN
      END
```

```
      SUBROUTINE STPNT(NDIM,U,PAR)
C     ---------- -----

C In this subroutine a starting point (U,PAR) must be defined.
C (U must satisfy the state equation F(U,PAR)=0 ).

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION U(NDIM),PAR(20)

C        NDIM    :   Dimension of the state equation.
C        U       :   The state vector.
C        PAR     :   The vector of control parameters.

C Give a starting point on the sphere :

      X1=1.0     X2=0.0     X3=0.0     X4=0.0     X5=0.0

C Set the state variable:

      U(1)=X1

C Set the control variables:

      PAR(1)=X2
      PAR(2)=X3
      PAR(3)=X4
      PAR(4)=X5

      RETURN
      END
```

```
      SUBROUTINE FOPT(NDIM,U,ICP,PAR,IJAC,FS,DFDU,DFDP)
C     ---------- ----

C In this subroutine the objective function must be defined.

C Supplied variables:
C        NDIM    :   Dimension of the state equation.
C        U       :   The state vector.
C        ICP     :   Indices of the control parameters.
C        PAR     :   The vector of control parameters.
C        IJAC    :   IJAC=0 :  derivatives need not be returned,
C                    IJAC=1 :  derivatives must be returned also.

C Values to be returned:
C        FS      :   The value of the objective function.
C        DFDU    :   DFDU(j) should be given the value of the derivative
C                    of FS with respect to the j'th component of U.
C        DFDP    :   DFDP(j) should be given the value of the derivative
C                    of FS with respect to the control parameter PAR(j).

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION U(NDIM),ICP(20),PAR(20),DFDU(NDIM),DFDP(20)

       X1=U(1)
       X2=PAR(1)       X3=PAR(2)       X4=PAR(3)       X5=PAR(4)

       FS=X1 + X2 + X3 + X4 + X5

      IF(IJAC.EQ.0)RETURN

       DFDU(1)=1.0
       DFDP(1)=1.0     DFDP(2)=1.0     DFDP(3)=1.0     DFDP(4)=1.0

      RETURN
      END
```

```
      SUBROUTINE INIT
C     ---------- ----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      COMMON /BLBCN/ NDIM,IPS,IRS,ILP,ICP(20),PAR(20)
      COMMON /BLCDE/ NTST,NCOL,IAD,ISP,ISW,IPLT,NBC,NINT
      COMMON /BLDLS/ DS,DSMIN,DSMAX,IADS
      COMMON /BLLIM/ NMX,NUZR,RL0,RL1,A0,A1
      COMMON /BLMAX/ NPR,MXBF,IID,ITMX,ITNW,NWTN,JAC

      NDIM=1        IPS=5         ILP=1        IRS=0
      RL0=-2.0      RL1=3.0       A0=-2.0      A1=3.0
      DS=0.01       DSMIN=0.001   DSMAX=0.2    IID=2
      NMX=25        NPR=25

C Define the parameters that are to be freed successively.

       ICP(2)=1      ICP(3)=2      ICP(4)=3      ICP(5)=4

C Limits points are to be located (ILP=1).
C These will correspond to local extrema of the objective function.
C In the first run there is no restart point (IRS = 0) and PAR(ICP(2))
C will be used as the free problem parameter.
C Note that ICP(1) is reserved for optimization problems:
C  ICP(1) may not be used to designate problem parameters.

C For subsequent restarts, the label of a limit point (=local extremum)
C from the preceding run should be assigned to IRS.
C The unit 8 output file of this preceding run must be attached as
C restart file (unit 3).
C For every successive restart an additional free parameter must be
C specified in ICP(3), ICP(4),...etc.
C (All of these may be set in the very first run, as done above.)

      RETURN
      END
```

```
      FUNCTION USZR(I,NUZR,PAR)
C     -------- ----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION PAR(20)

C This function is not used in this example.

      USZR=0.0

      RETURN
      END
```

8.2 Locating extrema of the objective function.

An initial 1-parameter optimization is obtained by typing

@AUTOS OPT .

The output is

BR	PT	TY	LAB	FOPT	L2-NORM	U(1)	PAR(1)
1	1	EP	1	1.000000E+00	1.000000E+00	1.000000E+00	0.000000E+00
1	12	LP	2	1.414214E+00	7.071100E-01	7.071100E-01	7.071036E-01
1	25	EP	3	3.484888E-01	5.110576E-01	-5.110576E-01	8.595465E-01

Above FOPT denotes the value of the objective function. The free parameter is PAR(1). The limit point (label 2) is a limit point with respect to FOPT. It therefore corresponds to a one parameter extremum of the objective function.

Save the unit 7,8, and 9 output files under the names P.OPT, Q.OPT and D.OPT respectively, by typing

@SVAUT OPT .

To compute an entire curve of such extrema, make the following change in INIT:

IRS from 0 to 2 (restart label),

and restart, by typing

@AUTOS OPT .

The output is:

BR	PT	TY	LAB	FOPT	L2-NORM	U(1)	PAR(1)	PAR(2)
2	10	LP	4	1.732051E+00	5.773614E-01	5.773614E-01	5.773614E-01	5.773281E-01
2	25	EP	5	3.759085E-01	2.732147E-01	-2.732147E-01	-2.732147E-01	9.223382E-01

Save the new output files, or perhaps replace the old output files by typing

@RPAUT OPT .

The contents of the old files is now lost. The limit point found above (label 4) is a 2-parameter e compute a curve of these, change the value of IRS from 2 to 4, and restart as before. The output is:

BR	PT	TY	LAB	FOPT	L2-NORM	U(1)	PAR(1)	PAR(2)	PAR(3)
4	11	LP	6	2.000000E+00	5.000055E-01	5.000055E-01	5.000055E-01	5.000055E-01	4.999838E-01
4	25	EP	7	8.151979E-01	5.980741E-02	-5.980741E-02	-5.980741E-02	-5.980741E-02	9.946203E-01

The limit point (label 6) is now a 3-parameter extremum of the objective function. To compute a curve of these, replace the output files, change the value of IRS from 4 to 6, and restart.

The output will be:

```
BR    PT TY LAB      FOPT         L2-NORM          U(1)          PAR(1)          PAR(2)
6     10 LP   8  2.236068E+00  4.472162E-01  4.472162E-01  4.472162E-01  4.472162E-01
6     25 EP   9  7.790896E-01  5.377738E-02 -5.377738E-02 -5.377738E-02 -5.377738E-02

                                                          PAR(3)          PAR(4)
                                                       4.472162E-01  4.472033E-01
                                                      -5.377738E-02  9.941994E-01
```

The final limit point (label 8) denotes the extremum of the objective function over all four parameters. In this example the value of the control variables at the optimum should be equal to the value of the state variable. That this is not exactly the case in the output above is due to the preset level of accuracy of the computation. If more accuracy is required, then one has to change the tolerances EPSU, EPSL, and EPSS. See the main documentation for a definition of these quantities.

If a restart point specified by IRS does not denote a limit point (=local optimum), then AUTO will continue the given branch further, leaving the number of control variables unchanged.

9. Example Problem: AUTBVP.

(A Boundary Value Problem with Bifurcations.)

This is a two point boundary value problem of the form

$$u_1' = u_2,$$
$$u_2' = -(p_1\pi)^2 u_1 + u_1^2,$$

with boundary conditions $u_1(0) = 0, \quad u_1(1) = 0.$

The user supplied subroutines for defining this problem are very similar to those for example problem AUTEXP. The listing below therefore has little documentation. Generally it is advisable to provide derivatives, but in this and the following examples the derivatives will not be given (JAC=0) in order to abbreviate the listings.

9.1 Listing of AUTBVP.

```
      SUBROUTINE FUNC(NDIM,U,ICP,PAR,IJAC,F,DFDU,DFDP)
C     ---------- ----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION U(NDIM),PAR(20),F(NDIM),DFDU(NDIM,NDIM),DFDP(NDIM,20)

       PI=4*DATAN(1.0D 00)

       F(1) = U(2)
       F(2) = -( PAR(1)*PI )**2 * U(1) + U(1)**2

      RETURN
      END
```

```
      SUBROUTINE INIT
C     ---------- ----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      COMMON /BLBCN/ NDIM,IPS,IRS,ILP,ICP(20),PAR(20)
      COMMON /BLCDE/ NTST,NCOL,IAD,ISP,ISW,IPLT,NBC,NINT
      COMMON /BLDLS/ DS,DSMIN,DSMAX,IADS
      COMMON /BLLIM/ NMX,NUZR,RL0,RL1,A0,A1
      COMMON /BLMAX/ NPR,MXBF,IID,ITMX,ITNW,NWTN,JAC

       ICP(1)=1

       PAR(1)=0.0

       NDIM=2       IPS=4        IRS=0       ILP=1
       ISP=2        ISW=1        NBC=2       NINT=0
       DS=0.2       DSMIN=0.001  DSMAX=0.2   NTST=5
       RL0=0.0      RL1=5.0      A0=-50.0    A1=50.0
       NMX=100      NPR=100      JAC=0

C Choose the integral over [0,1] of the first solution component
C as the 'principal' vertical axis in the bifurcation diagram.

       IPLT=3

      RETURN
      END
```

```
      SUBROUTINE STPNT(NDIM,U,PAR,T)
C     ---------- -----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION U(NDIM),PAR(20)

       U(1)=0.0       U(2)=0.0

      RETURN
      END

      SUBROUTINE BCND(NDIM,PAR,ICP,NBC,U0,U1,FB,IJAC,DBC)
C     ---------- ----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION PAR(20),ICP(20),U0(NDIM),U1(NDIM),FB(NBC),DBC(NBC,20)

       FB(1)=U0(1)       FB(2)=U1(1)

      RETURN
      END

      SUBROUTINE ICND(NDIM,PAR,ICP,NINT,U,UOLD,UDOT,UPOLD,F,IJAC,DINT)
C     ---------- ----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

C ( THIS PROBLEM HAS NO INTEGRAL CONSTRAINTS  )

      RETURN
      END
```

```
      FUNCTION USZR(I,NUZR,PAR)
C     -------- ----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION PAR(20)

C (Not used in this example.)
        USZR=0.0

      RETURN
      END
```

9.2 Detecting bifurcation points.

To run the example problem in the file AUTBVP.FOR, type

@AUTOS BVP .

The output will be

BR	PT	TY	LAB	PAR(1)	INTEGRAL U(1)	MAX U(1)	MAX U(2)
1	1	EP	1	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
1	6	BP	2	1.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
1	12	BP	3	2.000001E+00	0.000000E+00	0.000000E+00	0.000000E+00
1	18	BP	4	3.000019E+00	0.000000E+00	0.000000E+00	0.000000E+00
1	24	BP	5	4.000231E+00	0.000000E+00	0.000000E+00	0.000000E+00
1	29	EP	6	5.000231E+00	0.000000E+00	0.000000E+00	0.000000E+00

Note the bifurcation points, indicated as BP.

Save the output files under the names P.BVP, Q.BVP, and D.BVP respectively, by typing

@SVAUT BVP .

9.3 Branch switching at a bifurcation point.

Suppose we want to switch branches at the first bifurcation point. For this make the following changes in INIT:

IRS	from	0	to	2	(restart label),
ISW	from	1	to	−1	(switch branches),
NPR	from	100	to	20	(output frequency),
DSMAX	from	0.2	to	10.0	(maximum step size),

and then type

@AUTOS BVP .

Now the output will be

```
BR   PT TY LAB    PAR(1)       INTEGRAL U(1)   MAX U(1)       MAX U(2)
2    20      7  2.216474E+00  3.063972E+01  4.340622E+01  1.920102E+02
2    28 EP   8  2.721186E+00  5.134310E+01  6.973634E+01  3.596425E+02
```

Append the new output files (units 7,8,9) to the previous output files by typing

@APAUT BVP .

Sofar, the first bifurcating branch has been traced out in one direction only. To compute in the other direction, make the following change in INIT:

DS	from	0.2	to	−0.2	(reverse direction),

and restart, by typing

@AUTOS BVP .

This time the output will be

```
BR   PT TY LAB    PAR(1)       INTEGRAL U(1)   MAX U(1)       MAX U(2)
3    18 EP   9 -8.923502E-03 -7.254633E+00  6.668941E-21  3.307986E+01
```

One can append the output files to the previous output and use PLAUT to plot the results.

10. Example Problem: AUTLIN.
(A Linear Eigenvalue Problem.)

This is a linear two point boundary value problem of the form

$$u_1' = u_2,$$
$$u_2' = (p_1\pi)^2 u_1,$$

with boundary conditions $u_1(0) - p_2 = 0$ and $u_1(1) = 0$. The purpose of this example is to show how one can determine the dependence of an eigenvalue on a second parameter (Here p_2). For this purpose it will be convenient if we add an integral condition and another free parameter to the above equations, namely, $\int_0^1 u_1(t)^2 dt - p_3 = 0$.

To compute a branch of solutions to these equation we shall always need two free parameters. Note that p_3 simply 'measures' the L_2-norm of the first solution component. The user supplied subroutines for defining this problem are very similar to those for example problems AUTBVP and AUTINT.

10.1 Listing of AUTLIN.

```
      SUBROUTINE FUNC(NDIM,U,ICP,PAR,IJAC,F,DFDU,DFDP)
C     ---------- ----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION U(NDIM),PAR(20),F(NDIM),DFDU(NDIM,NDIM),DFDP(NDIM,20)

       PI=4*DATAN(1.0D 00)

       F(1) = U(2)       F(2) = -( PAR(1)*PI )**2 * U(1)

      RETURN
      END
```

```
      SUBROUTINE STPNT(NDIM,U,PAR,T)
C     ---------- -----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION U(NDIM),PAR(20)

       U(1)=0.0       U(2)=0.0

      RETURN
      END

      SUBROUTINE INIT
C     ---------- ----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      COMMON /BLBCN/ NDIM,IPS,IRS,ILP,ICP(20),PAR(20)
      COMMON /BLCDE/ NTST,NCOL,IAD,ISP,ISW,IPLT,NBC,NINT
      COMMON /BLDLS/ DS,DSMIN,DSMAX,IADS
      COMMON /BLLIM/ NMX,NUZR,RL0,RL1,A0,A1
      COMMON /BLMAX/ NPR,MXBF,IID,ITMX,ITNW,NWTN,JAC

       ICP(1)=1       ICP(2)=3
       PAR(1)=0.0     PAR(2)=0.0     PAR(3)=0.0

       NDIM=2         IPS=4          IRS=0          ILP=0
       ISP=2          ISW=1          NBC=2          NINT=1
       NTST=5         DS=0.2         DSMIN=0.001    DSMAX=0.2
       NMX=100        NUZR=1         NPR=100        JAC=0
       RL0=0.0        RL1=5.0        A0=0.0         A1=5.0

      RETURN
      END
```

```
      SUBROUTINE BCND(NDIM,PAR,ICP,NBC,U0,U1,FB,IJAC,DBC)
C     ---------- ----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION PAR(20),ICP(20),U0(NDIM),U1(NDIM),FB(NBC),DBC(NBC,20)

       FB(1)=U0(1)-PAR(2)       FB(2)=U1(1)

      RETURN
      END

      SUBROUTINE ICND(NDIM,PAR,ICP,NINT,U,UOLD,UDOT,UPOLD,FI,IJAC,DINT)
C     ---------- ----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION U(NDIM),UOLD(NDIM),UDOT(NDIM),UPOLD(NDIM)
      DIMENSION FI(NINT),DINT(NINT,1),ICP(20),PAR(20)

       FI(1)=U(1)*U(1)-PAR(3)

      RETURN
      END


      FUNCTION USZR(I,NUZR,PAR)
C     -------- ----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION PAR(20)

        USZR=PAR(3)-1.0

      RETURN
      END
```

10.2 Detecting eigenvalues.

To run the example problem in the file AUTLIN.FOR, type

@AUTOS LIN .

The output will be

BR	PT	TY	LAB	PAR(1)	L2-NORM	MAX U(1)	MAX U(2)	PAR(3)
1	1	EP	1	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
1	6	BP	2	1.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
1	12	BP	3	2.000001E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
1	18	BP	4	3.000019E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
1	24	BP	5	4.000231E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
1	29	EP	6	5.000231E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00

Note the the bifurcation points, indicated as BP. These are the eigenvalues of the linear eigenvalue problem.

Save the output files under the names P.LIN, Q.LIN, and D.LIN respectively, by typing

@SVAUT LIN .

10.3 Branch switching at an eigenvalue.

Suppose we want to switch branches at the first eigenvalue. For this, make the following changes in INIT:

IRS	from	0	to	2	(restart label),
ISW	from	1	to	-1	(switch branches),
DSMAX	from	0.2	to	0.5	(maximum step size),

and then type

@AUTOS LIN .

The output will be

```
BR    PT TY LAB    PAR(1)         L2-NORM        MAX U(1)       MAX U(2)       PAR(3)
2     9 UZ   7  1.000000E+00  3.296909E+00  1.414166E+00  4.442884E+00  1.000000E+00
2    13 EP   8  1.000000E+00  5.296909E+00  2.272039E+00  7.138066E+00  2.581257E+00
```

Note that the branch is 'vertical' (constant PAR(1)), because we deal with a linear eigenvalue problem. Also recall that PAR(3) measures the L_2-norm of the first solution component. At label 9 (UZ) this norm equals 1.0 and restart information is written in unit 8 as specified in the user supplied subroutine USZR.

Append the new output files (units 7,8,9) to the previous output files by typing

@APAUT LIN .

10.4 A 2-parameter curve of eigenvalues.

To determine how the first eigenvalue depends on the parameter p_2 make the following changes in INIT:

IRS	from	2	to	7	(restart label),
ISW	from	-1	to	1	(no branch switching),
ICP(2)	from	3	to	2	(new free parameter).

The free parameters are now PAR(1) and PAR(2). The value of PAR(3) (the L_2-norm) will now remain constant and equal to 1.0.

Start the computation by typing

@AUTOS LIN .

This time the output will be

```
BR    PT TY LAB    PAR(1)         L2-NORM        MAX U(1)       MAX U(2)       PAR(2)
2    14 EP   9  1.715464E+00  5.021822E+00  1.354154E+00  7.298193E+00 -1.055625E+00
```

The plotting program PLAUT can be used to draw the computed curve of eigenvalues in the PAR(1)-PAR(2) plane. Further computation of the curve is possible by restarting at label 9 with enlarged diagram limits. Computation of the curve in reverse direction can be done by sign reversal of DS.

11. Example Problem: AUTPP3.
(Period-Doubling Bifurcations.)

This is a 3-species predator prey model with harvesting of the form

$$u_1' = u_1(1-u_1) - p_4 u_1 u_2,$$
$$u_2' = -p_2 u_2 + p_4 u_1 u_2 - p_5 u_2 u_3 - p_1(1-e^{-p_6 u_2})$$
$$u_3' = -p_3 u_3 + p_5 u_2 u_3.$$

The solution structure of these equations in considered in *J. Math. Biol. 20, 1984, 1-14.* We use this example here to show how one can switch branches at a period-doubling bifurcation.

11.1 Listing of AUTPP3.

```
      SUBROUTINE FUNC(NDIM,U,ICP,PAR,IJAC,F,DFDU,DFDP)
C     ---------- ----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION U(NDIM),PAR(1)
      DIMENSION F(NDIM),DFDU(NDIM,NDIM),DFDP(NDIM,1)

       F(1)= U(1)*(1-U(1)) - PAR(4)*U(1)*U(2)
       F(2)=-PAR(2)*U(2)   + PAR(4)*U(1)*U(2) - PAR(5)*U(2)*U(3)
     *                     - PAR(1)*(1-EXP(-PAR(6)*U(2)))
       F(3)=-PAR(3)*U(3)   + PAR(5)*U(2)*U(3)

       RETURN
       END
```

```
      SUBROUTINE STPNT(NDIM,U,PAR)
C     ---------- -----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION U(NDIM),PAR(1)

       U(1)=1.0         U(2)=0.0         U(3)=0.0

       PAR(1)=0.0       PAR(2)=0.25      PAR(3)=0.5
       PAR(4)=5.6       PAR(5)=3.0       PAR(6)=5.0

      RETURN
      END

      SUBROUTINE INIT
C     ---------- ----
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      COMMON /BLBCN/ NDIM,IPS,IRS,ILP,ICP(20),PAR(20)
      COMMON /BLCDE/ NTST,NCOL,IAD,ISP,ISW,IPLT,NBC,NINT
      COMMON /BLDLS/ DS,DSMIN,DSMAX,IADS
      COMMON /BLLIM/ NMX,NUZR,RL0,RL1,A0,A1
      COMMON /BLMAX/ NPR,MXBF,IID,ITMX,ITNW,NWTN,JAC

       NDIM=3       IPS=1        IRS=0
       ILP=1        ICP(1)=1     ICP(2)=4
       NTST=10      ISP=2        ISW=1
       NMX=200      NPR=200      JAC=0
       DS=0.01      DSMIN=0.002  DSMAX=0.02
       RL0=0.0      RL1=2.0      A0=0.0       A1=2.0

      RETURN
      END
```

```
      FUNCTION USZR(I,NUZR,PAR)
C     -------- ----
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION PAR(20)

      USZR=0.0

      RETURN
      END
```

11.2 Preliminary Computations.

Run this example problem by typing

@AUTOS PP3 .

The output will be

BR	PT	TY	LAB	PAR(1)	L2-NORM	U(1)	U(2)	U(3)
1	1	EP	1	0.000000E+00	1.000000E+00	1.000000E+00	0.000000E+00	0.000000E+00
1	55	BP	2	1.070000E+00	1.000000E+00	1.000000E+00	0.000000E+00	0.000000E+00
1	102	EP	3	2.010000E+00	1.000000E+00	1.000000E+00	0.000000E+00	0.000000E+00

BR	PT	TY	LAB	PAR(1)	L2-NORM	U(1)	U(2)	U(3)
2	56	EP	4	1.355913E+00	2.006196E+00	1.998260E+00	-1.782607E-01	0.000000E+00

BR	PT	TY	LAB	PAR(1)	L2-NORM	U(1)	U(2)	U(3)
2	34	HB	5	6.662994E-01	5.372057E-01	5.306268E-01	8.381665E-02	0.000000E+00
2	72	HB	6	7.537876E-02	1.860007E-01	9.073341E-02	1.623690E-01	0.000000E+00
2	75	BP	7	3.634700E-02	1.795044E-01	6.666139E-02	1.666676E-01	0.000000E+00
2	78	EP	8	-1.480074E-02	1.758605E-01	3.578249E-02	1.721817E-01	0.000000E+00

BR	PT	TY	LAB	PAR(1)	L2-NORM	U(1)	U(2)	U(3)
3	6	EP	9	-1.140621E-02	1.874545E-01	6.666660E-02	1.666667E-01	5.400919E-02

BR	PT	TY	LAB	PAR(1)	L2-NORM	U(1)	U(2)	U(3)
3	5	HB	10	6.293088E-02	1.820036E-01	6.666660E-02	1.666667E-01	-3.005146E-02
3	14	HB	11	1.702538E-01	2.348361E-01	6.666660E-02	1.666667E-01	-1.514126E-01
3	137	EP	12	1.799887E+00	2.002271E+00	6.666660E-02	1.666667E-01	-1.994208E+00

Note the four Hopf bifurcation points, indicated as HB.

Save the output files under the names P.PP3, Q.PP3, and D.PP3 respectively, by typing

@SVAUT PP3 .

In order to compute the branch of periodic solutions that bifurcates from the Hopf bifurcation point with label 11, make the following changes in INIT.

IRS	from	0	to	11	(restart label),
IPS	from	1	to	2	(periodic solutions),
NMX	from	200	to	15	(maximum number of steps),

and then type

@AUTOS PP3 .

The output will be

```
BR    PT TY LAB     PAR(1)         L2-NORM        MAX U(1)       MAX U(2)       MAX U(3)
11     6 PD  13  1.955108E-01  2.638959E-01  1.493848E-01  2.247852E-01 -1.020317E-01
11    15 EP  14  2.919909E-01  3.781098E-01  3.769431E-01  3.481973E-01 -6.753894E-02

                                                                          PERIOD
                                                                     2.096959E+01
                                                                     2.803399E+01
```

Note the detection of a period doubling bifurcation (label 13). Append the new output files (units 7,8,9) to the previous output files by typing

@APAUT PP3 .

11.3 Branch switching at a period doubling bifurcation.

To switch branches at the period-doubling bifurcation found above, make the following changes in INIT:

IRS	from	11	to	13	(restart label),
ISW	from	1	to	−1	(switch branches),
NTST	from	10	to	20	(increase accuracy).

The doubling of NTST is required for branch switching at period-doubling bifurcations (See main documentation).

Restart, by typing

@AUTOS PP3 .

The output will be

```
BR    PT TY LAB    PAR(1)        L2-NORM       MAX U(1)      MAX U(2)      MAX U(3)
12     6 PD  15  1.975329E-01  2.672741E-01  1.740751E-01  2.423755E-01 -8.949440E-02
12    15 EP  16  2.532501E-01  3.397171E-01  3.889997E-01  3.773087E-01 -4.833423E-02

                                                                          PERIOD
                                                                      4.195837E+01
                                                                      4.992349E+01
```

Note the detection of a second, cascading, period-doubling bifurcation. Append the new output files (units 7,8,9) to the previous output files by typing

@APAUT PP3 .

The branch bifurcating from the second period doubling (label 15) can be determined by repeating the above procedure. The value of NTST will have to be doubled again for this computation.

11.4 Another example of using PLAUT: 3D plots.

To plot the contents of the accumulated output files P.PP3 and Q.PP3, type

@PLAUT PP3 .

Suppose we want a 3D plot of the last computed orbit with label 16. To do this first enter 3D mode by typing 3D. You will be asked for labels. Enter 16. When prompted for axes, enter 2 3 4. This will result in a plot of the three solution components of the periodic orbit. (The first output component written per line in unit 8 is the independent time variable. The solution components occupy positions 2, 3, and 4 per output line.) To actually get the plot, using a choice of default settings, type D4. A 3D plot of the orbit will appear om the screen.

To get another plot of the same orbit from a different view point type VU. You will be asked for a new view point. Enter, e.g., 0.5 0.5 1.0 (Only the relative size of the coordinates matters here). Type D to get the plot.

Type PSL and then D to get the same plot, but with the projections shown as solid curves rather than dotted curves.

For user-selected projections, type PJU. Upon prompting enter, e.g., 3. Then type D to have the plot drawn. Now only the projection onto the horizontal plane will be shown.

To rotate the orbit around the z-axis, type RZ. Upon request enter an angle, e.g. 15 (degrees). Type D to get the rotated plot. Note that coordinate plane intercepts are marked and that portions of the curve outside the positive octant appear as dashed curves.

To scale the current plot, enter SC. Upon prompting enter, e.g., 1.4 1.3 1.5 as scaling factors for the three coordinates. Type D to get the new plot.

For more information on the 3D option type HELP. Type END to exit from 3D. Once more type END to exit from PLAUT.

12. Example Problem: AUTWAV.

(Wave Phenomena.)

This is a system of two partial differential equations that models an enzyme catalyzed reaction. The equations are:

$$\frac{\partial u_1}{\partial t} = \frac{\partial^2 u_1}{\partial x^2} - p_1\left(p_4 R - (p_2 - u_1)\right),$$
$$\frac{\partial u_2}{\partial t} = \beta\frac{\partial^2 u_2}{\partial x^2} - p_1\left(p_4 R - p_7(p_3 - u_2)\right),$$

where

$$R = \frac{u_2}{p_5 + u_2}\frac{u_1}{1 + u_1 + p_6 u_1^2}.$$

12.1 Listing of AUTWAV.

```
      SUBROUTINE FUNC(NDIM,U,ICP,PAR,IJAC,F,DFDU,DFDP)
C     ---------- ----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION U(NDIM),PAR(20)
      DIMENSION F(NDIM),DFDU(NDIM,NDIM),DFDP(NDIM,20)

C Define the equations (excluding the diffusion terms).

       R= U(2)/(PAR(5)+U(2)) * U(1)/(1+U(1)+PAR(6)*U(1)*U(1))

       F(1)=-PAR(1)*( PAR(4)*R - (PAR(2)-U(1)) )
       F(2)=-PAR(1)*( PAR(4)*R - PAR(7)*(PAR(3)-U(2)) )

      RETURN
      END
```

```
      SUBROUTINE STPNT(NDIM,U,PAR)
C     ---------- -----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION U(NDIM),PAR(20)

C Initialize the problem parameters.

       PAR(1) = 3.0
       PAR(2) = 145
       PAR(3) = 539.23
       PAR(4) = 210
       PAR(5) = 3.4
       PAR(6) = 0.023
       PAR(7) = 0.2

C Define a spatially uniform stationary state
C for the above choice of parameters.

       U(1) = 59.702
       U(2) = 112.752

C Set the wave speed (PAR(10) is reserved for this purpose).

       PAR(10) = 0.05

C Set the diffusion constants.
C (PAR(15)-PAR(20) are reserved for this purpose.)

       PAR(15) = 1.0
       PAR(16) = 5.0

      RETURN
      END
```

```
      SUBROUTINE INIT
C     ---------- ----
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      COMMON /BLBCN/ NDIM,IPS,IRS,ILP,ICP(20),PAR(20)
      COMMON /BLCDE/ NTST,NCOL,IAD,ISP,ISW,IPLT,NBC,NINT
      COMMON /BLDLS/ DS,DSMIN,DSMAX,IADS
      COMMON /BLLIM/ NMX,NUZR,RL0,RL1,A0,A1
      COMMON /BLMAX/ NPR,MXBF,IID,ITMX,ITNW,NWTN,JAC

      NDIM=2     IPS=1      IRS=0      ILP=1       ICP(1)=3
      NTST=20    ISP=0      ISW=1      NMX=75      ICP(2)=10
      JAC=0      DS=0.5     DSMIN=0.1  DSMAX=10.0  NUZR=0
      RL0=0.0    RL1=700.0  A0=0.0     A1=1000.0   NPR=75

      RETURN
      END

       FUNCTION USZR(I,NUZR,PAR)
C      -------- ----
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION PAR(20)

       GOTO(1,2,3)I

1     USZR=PAR(3)-610.0
       RETURN

2     USZR=PAR(3)-630.0
       RETURN

3     USZR=PAR(11)-22.0
       RETURN

       END
```

12.2 Preliminary computations.

To determine the spatially uniform steady state solution structure of the example problem in the example file AUTWAV.FOR, type

@AUTOS WAV .

The summary output in unit 6 will be:

BR	PT	TY	LAB	PAR(3)	L2-NORM	U(1)	U(2)
1	1	EP	1	5.392300E+02	1.275827E+02	5.970200E+01	1.127520E+02
1	22	LP	2	6.661437E+02	1.818407E+02	4.690848E+01	1.756861E+02
1	37	LP	3	6.308909E+02	7.490197E+01	3.233447E+01	6.756320E+01
1	43	HB	4	6.622018E+02	3.255911E+01	1.798752E+01	2.713935E+01
1	47	EP	5	7.002895E+02	2.173036E+01	8.906369E+00	1.982133E+01

Note the proximity of two limit points and a Hopf bifurcation point. This implies that the full partial differential equation will have traveling wave solutions for sufficiently large wave speed. The results of the above computations will not be needed below. The computation was done for illustrative purpose only. Thus the output files need not be saved.

Note that AUTWAV.FOR has already been set up for the detection and continuation of traveling wave solutions. Specifically, the wave speed (PAR(10)) and the two diffusion constants (PAR(15) and PAR(16)) have been initialized in STPNT.

Now make the following change in INIT:

IPS from 1 to 11 (look for bifurcations to waves).

Run AUTO again, by typing

@AUTOS WAV .

Now the unit 6 output will be:

BR	PT	TY	LAB	PAR(3)	L2-NORM	U(1)	U(2)
1	1	EP	1	5.392300E+02	1.275827E+02	5.970200E+01	1.127520E+02
1	22	LP	2	6.661437E+02	1.818407E+02	4.690848E+01	1.756861E+02
1	37	LP	3	6.308909E+02	7.490197E+01	3.233447E+01	6.756320E+01
1	40	HB	4	6.355912E+02	5.497349E+01	2.741207E+01	4.765148E+01
1	48	EP	5	7.046887E+02	2.143781E+01	8.037118E+00	1.987421E+01

U(3)	U(4)
0.000000E+00	0.000000E+00
0.000000E+00	0.000000E+00
0.000000E+00	0.000000E+00
0.000000E+00	0.000000E+00
0.000000E+00	0.000000E+00

The limit points are as before, but the Hopf bifurcation takes place at a different location. In fact, the Hopf bifurcation now signals a bifurcation to a traveling wave of wave speed PAR(10) (in this example PAR(10)=0.05).

Save the output files under the names P.WAV, Q.WAV, and D.WAV, by typing

@SVAUT WAV .

12.3 Traveling waves from a Hopf bifurcation point.

In order to compute the branch of waves of fixed wave speed PAR(10)=0.05 that emanates from the Hopf bifurcation point at label 4 above, make the following changes in INIT:

IRS	from	0	to	4	(restart label),
IPS	from	11	to	12	(traveling waves),
NUZR	from	0	to	3	(user defined output points),
ILP	from	1	to	0	(turn off detection of limit points),

and then type

@AUTOS WAV .

This computation and those that follow require a fair amount of CPU time. It is therefore perhaps better to submit a batch job by typing instead

SUBMIT AUTOS / PAR=WAV .

The output on unit 6 will be

BR	PT	TY	LAB	PAR(3)	L2-NORM	MAX U(1)	MAX U(2)	MAX U(3)	MAX U(4)	PERIOD
4	5	UZ	6	6.355923E+02	5.520906E+01	2.843928E+01	5.094883E+01	3.026960E-01	9.157546E-01	2.200000E+01
4	9	UZ	7	6.361881E+02	5.738212E+01	3.142086E+01	6.071621E+01	2.072390E+00	3.965566E+00	2.200000E+01
4	51	UZ	8	6.300000E+02	6.184102E+01	4.997210E+01	7.841665E+01	1.695000E+01	1.879296E+01	1.170068E+01
4	57	UZ	9	6.100000E+02	9.350632E+01	5.675413E+01	1.204616E+02	1.607223E+01	2.411035E+01	1.497530E+01
4	61	UZ	10	6.049190E+02	1.194360E+02	5.616753E+01	1.470836E+02	1.549190E+01	2.470734E+01	2.200004E+01
4	75	EP	11	6.067428E+02	1.646100E+02	5.580192E+01	1.560172E+02	1.328344E+01	2.422343E+01	1.155263E+03

Note the plotting and restart output at zeroes of the functions supplied in the subroutine USZR. In particular, note the detection of three different traveling waves of wave length (='period') equal to 22.0.

Append the new unit 7,8, and 9 output files to the previous output files stored as P.WAV, Q.WAV, and D.WAV, respectively, by typing:

@APAUT WAV .

Actually there are several limit points and bifurcations along the branch of traveling waves computed above. The bifurcation behavior in this example is quite complicated. Details can be found in *A Numerical Analysis of Wave Phenomena in a Reaction Diffusion Model, in: Oscillations in Biology and Chemistry, H.G. Othmer, ed., Lecture Notes in Biomathematics, Springer Verlag.*

12.4 Waves traveling around a ring.

To compute a 2-parameter curve of traveling waves of fixed wave length, make the following changes in INIT:

IRS	from	4	to	7	(restart label),
IPS	from	12	to	13	(fixed wave length),
NMX	from	75	to	25	(maximum number of steps).

Thus we want to restart at output label 7, found above, which has wave length 22.0. The continuation parameters will be PAR(3) and the wave speed PAR(10) as specified in INIT. Start the 2-parameter computation by typing

@AUTOS WAV .

The output is:

```
BR    PT TY LAB     PAR(3)         L2-NORM        MAX U(1)       MAX U(2)       MAX U(3)
7     25 EP  12  6.302433E+02  1.041504E+02  5.441612E+01  1.434382E+02  2.357942E+01

                                                MAX U(4)       PAR(10)
                                              3.128534E+01  5.891823E-01
```

Along the computed branch the wave length is constant and equal to 22.0. Thus each of the solutions computed can be thought of as a wave of a certain wave speed (= value of PAR(10)) that travels around a ring of circumference 22.0.

Normally one will save and plot the output files of this run. However, we shall not need the results again in this example.

12.5 Time integration to test stability.

There is no information on the asymptotic stability of any of the computed traveling waves. To test stability we can use a computed wave as initial data in a time integration of the partial differential equation. For example, to test the stability of the traveling wave with label 7 found above in Section 12.3, make the following change in INIT:

IPS	from	13	to	14	(time integration),
NMX	from	25	to	80	(maximum number of steps),
NPR	from	75	to	10	(regular output points).

Start the time integration by typing

@AUTOS WAV .

The output is:

BR	PT	TY	LAB	TIME	L2-NORM	MAX U(1)	MAX U(2)
4	10		12	1.066297E+01	5.742977E+01	3.159066E+01	6.219886E+01
4	20		13	1.342745E+01	5.627367E+01	3.601162E+01	7.017847E+01
4	30		14	1.741691E+01	6.131912E+01	4.882608E+01	8.571494E+01
4	40		15	2.617667E+01	5.474263E+01	4.733966E+01	7.114598E+01
4	50		16	4.402828E+01	5.318968E+01	4.664799E+01	6.766422E+01
4	60		17	5.903789E+01	5.055587E+01	4.437825E+01	5.745706E+01
4	70		18	1.204313E+02	5.053738E+01	4.386391E+01	5.595132E+01
4	80	EP	19	2.204313E+02	5.053728E+01	4.391177E+01	5.599934E+01

Evidently, the traveling wave that we used as initial data is unstable. During the time integration there is first a period of transition followed by stationary behavior. In particular the L_2-norm clearly becomes constant (Maxima are only approximate: hence their continued variation). This long time behavior can possibly be a spatially uniform state, a traveling wave, or a stationary wave. A plot of the last three labels (17, 18, and 19) will show that the long time behavior is in fact a stationary wave (or 'pattern') with three 'humps'.

If one wants to keep the old files then these should now be renamed. Otherwise replace the accumulated output in the files P.WAV, Q.WAV, and D.WAV, by the new output, by typing

@RPAUT WAV .

12.6 A curve of stationary waves.

Suppose we want to compute an entire branch of stationary waves using the stationary wave found above as starting point. To do this make the following changes in INIT:

IPS	from	14	to	13	(fixed wave length),
IRS	from	7	to	19	(restart label),
NMX	from	80	to	10	(maximum number of steps).

An additional change is required. Recall that the time integration in the preceding section started with a traveling wave of wave speed PAR(10)=0.05. After time integration the final state was a stationary wave (zero wave speed). However, AUTO does not monitor such a change during time integration: The wave speed for our restart label 19 still equals 0.05 in the restart file. Thus it will be necessary to edit this restart file (Q WAV). The data set for label 19 is at the end of this file. In particular, the last three lines of the file contain the values of the 20 entries of the parameter array PAR. The 10th value is PAR(10) and equals 0.05. Change this value to 0.0 and replace the file.

Start the 2-parameter computation by typing

@AUTOS WAV .

The output is now:

```
BR    PT TY LAB     PAR(3)         L2-NORM        MAX U(1)       MAX U(2)       MAX U(3)
19    10 EP  20  6.631607E+02  3.276587E+01  2.209973E+01  3.072665E+01  3.637322E+00

                                                MAX U(4)       PAR(10)
                                              3.106581E+00 -7.289878E-07
```

Note that the wave speed (PAR(10)) is still treated as a free parameter, even though it is actually zero along the branch. This allows an extra constraint to be imposed, namely an integral phase condition. The latter is necessary, because the stationary waves would otherwise not be unique since they are invariant under translation.

13. Timing of AUTO: AUTTIM.

The example AUTTIM.FOR defines a simple first order system of ordinary differential equations with boundary conditions. The dimension of the system is variable: NDIM may be assigned any even value within the limits stated in Section 2.4 of this manual. If NDIM=2 then the example is identical to AUTEXP.FOR without derivatives.

AUTTIM.FOR can be used to get a well defined set of timings of AUTO. During each run of AUTTIM there should be 10 decompositions and 10 backsubstitutions in the linear equation solver BRBD of AUTLIB. Timings are printed at the end of output unit 9. In addition to the total CPU time the timings of certain individual subroutines will also be printed there. In particular the vectorized subroutines CONPAR, CONRHS, and INFPAR will be clocked.

The timings will give the user some idea of the CPU time required by AUTO for computations involving differential equations. They will also be useful for comparing the performance of AUTO on various machines.

To enable the timing, the user will have to insert appropriate installation dependent calls in the subroutines AUTIM0 and AUTIM1. These subroutines can be found in the library files AUTLIBS and AUTLIBD.

13.1 Listing of AUTTIM.

```
      SUBROUTINE FUNC(NDIM,U,ICP,PAR,IJAC,F,DFDU,DFDP)
C     ---------- ----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION U(NDIM),PAR(20),F(NDIM),DFDU(NDIM,NDIM),DFDP(NDIM,20)

       NDIM2=NDIM/2

       DO 1 I=1,NDIM2
         I1=2*(I-1)+1
         I2=I1+1
         E=DEXP(U(I1))
CSGLE    E= EXP(U(I1))
         F(I1)=U(I2)
         F(I2)=-PAR(1)*E
 1     CONTINUE

      RETURN
      END

      SUBROUTINE STPNT(NDIM,U,PAR,T)
C     ---------- -----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION U(NDIM),PAR(10)

      DO 1 I=1,NDIM
        U(I)=0.0
 1    CONTINUE

      RETURN
      END
```

```
      SUBROUTINE INIT
C     ---------- ----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      COMMON /BLBCN/ NDIM,IPS,IRS,ILP,ICP(20),PAR(20)
      COMMON /BLCDE/ NTST,NCOL,IAD,ISP,ISW,IPLT,NBC,NINT
      COMMON /BLDLS/ DS,DSMIN,DSMAX,IADS
      COMMON /BLEPS/ EPSL(20),EPSU,EPSS
      COMMON /BLLIM/ NMX,NUZR,RL0,RL1,A0,A1
      COMMON /BLMAX/ NPR,MXBF,IID,ITMX,ITNW,NWTN,JAC

C Only NDIM, NTST, and NCOL need to be varied to get different timings.
       NDIM=2        NTST=5        NCOL=2

       PAR(1)=0.0    DS=0.01       IADS=0
       IPS=4         NBC=NDIM      NINT=0
       NMX=10        NPR=10        JAC=0

      RETURN
      END

      SUBROUTINE BCND(NDIM,PAR,ICP,NBC,U0,U1,FB,IJAC,DBC)
C     ---------- ----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION PAR(10),ICP(10),U0(NDIM),U1(NDIM),FB(NBC),DBC(NBC,10)

       NDIM2=NDIM/2
       DO 1 I=1,NDIM2
         I1=2*(I-1)+1
         I2=I1+1
         FB(I1)=U0(I1)
         FB(I2)=U1(I1)
 1     CONTINUE

      RETURN
      END
```

```
      SUBROUTINE ICND(NDIM,PAR,ICP,NINT,U,UOLD,UDOT,UPOLD,FI,IJAC,DINT)
C     ---------- ----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

C (This problem has no integral constraints.)

      RETURN
      END

       FUNCTION USZR(I,NUZR,PAR)
C      -------- ----

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
CSGLE IMPLICIT REAL (A-H,O-Z)

      DIMENSION PAR(20)

      USZR=0.0

      RETURN
      END
```

13.2 Timing results.

The timing problem AUTTIM.FOR was run with various choices of NDIM, NCOL, and NTST. See the main documentation for a definition of these constants. Derivatives are not supplied in AUTTIM.FOR since JAC=0. Thus AUTO generates these by differencing. CPU time will decrease somewhat if derivatives are specified in the subroutines FUNC and BCND of AUTTIM.FOR.

Results of the timings are given below. The quantity listed is the total CPU time used by AUTO, in seconds, not including the preprocessing time. The calculations always include 10 decompositions and 10 backsubstitutions in the linear equation solver BRBD of AUTO.

IBM 4341 under VM/CMS OPT=3

Double Precision, NDIM=2

	NCOL=2	NCOL=4	NCOL=6
NTST= 5	1.20	1.84	2.75
NTST= 10	1.71	2.93	4.64
NTST= 20	2.71	5.12	8.53
NTST= 40	4.73	9.49	16.3
NTST= 80	8.75	18.4	32.0

IBM 4341 under VM/CMS OPT=3

Double Precision, NDIM=4

	NCOL=2	NCOL=4	NCOL=6
NTST= 5	1.96	3.82	6.83
NTST= 10	3.42	6.69	12.4
NTST= 20	5.54	12.4	23.6
NTST= 40	10.3	24.0	46.4
NTST= 80	19.9	47.2	91.6

FPS 164 with I/O to an IBM 4341, OPT=4.

Double Precision, NDIM=2

	NCOL=2	NCOL=4	NCOL=6
NTST= 5	0.43	0.68	1.05
NTST= 10	0.62	1.06	1.65
NTST= 20	1.02	1.82	2.88
NTST= 40	1.79	3.33	5.31
NTST= 80	3.35	6.33	10.1

FPS 164 with I/O to an IBM 4341, OPT=4.

Double Precision, NDIM=4

	NCOL=2	NCOL=4	NCOL=6
NTST= 5	0.73	1.55	2.95
NTST= 10	1.13	2.35	4.35
NTST= 20	1.93	4.00	7.21
NTST= 40	3.51	7.16	12.7
NTST= 80	6.66	13.5	23.9

In the above timings AUTO was run in its entirety on an FPS 164 Array Processor except for a dummy main program on the host processor an IBM 4341. Inner loops in the vectorized subroutines CONPAR, CONRHS, and INFPAR were replaced by calls to AP software. All output was written to a disk attached to the IBM 4341. The writing of output into IBM files takes up a non-negligible proportion of the timings reported above.

14. Notes on using AUTO.

1. Running AUTO without preprocessing.

AUTO has a preprocessor that generates the main program with the correct amount of array space and the required subroutine calls. This preprocessing takes some computer time and frequently it is desirable to bypass it. This can be done as follows:

First make changes in the user supplied subroutines INIT and STPNT in order to *read* parameter and constant values that one wants to change between successive runs. The READ can be be from Fortran Unit 5. Create a file FOR005.DAT and enter the values to be read in it. When doing this take into account that INIT is executed before STPNT. Now just run AUTO as before. If one wants another run with different parameter or constant values then make these changes in FOR005.DAT. Run AUTO again using the EXE file from the preceding run. Thus the entire preprocessing and loading can be omitted this time.

It is important to note that an EXE file can only be re-used for a 'similar task'. For example, a new EXE file must be generated if any one of the following is done:

(i) If IRS is changed from zero to nonzero or conversely. Changes from nonzero to nonzero are allowed.

(ii) If the value of NTST or NCOL is increased. Decreasing these values is ok.

(iii) For Hopf bifurcation problems, when changing from restart from a Hopf bifurcation point to restart from an orbit and conversely.

(iv) When IPS is changed.

(v) When ISW is changed from (1,−1) to 2 and conversely.

2. Efficient use of AUTO.

The main purpose of AUTO remains the computation of branches of periodic solutions. This is probably where the program is reasonably efficient. The bifurcation analysis of algebraic systems generally requires much less computer time and efficiency of implementation has not been an important consideration there. This applies in particular to the two-parameter continuation of limit points and Hopf bifurcation points.

It is easy to use up a lot of computer time when investigating an equation with free parameters. However, the user has significant control over the efficiency of computation. In particular the choice of NTST, NCOL, DS, DSMIN, DSMAX, and NMX, are important for this purpose. Smooth, slowly varying solutions can often be computed with small values of NTST, NCOL, e.g., NCOL=2, NTST=5. Sometimes ITNW, NWTN, THETAU, THETAL, EPSU, EPSL, EPSS, can also be adjusted to increase the efficiency of a computation. Understanding the significance of these constants is therefore useful. This information can be found in the Documentation section of this manual and in the Companion Document *Software for Continuation Problems in Ordinary Differential Equations with Applications.*

4. Spurious solutions.

Usuallly solutions are structurally correct, while there is still convergence. Parasitic (spurious) solutions are not normally obtained, because the adaptive mesh strategy (when IAD>0) to some extent provides a safeguard against computing such solutions. When they do occur, they are often easy to recognize by the jagged appearance of the solution branch. The latter may occur near homoclinic and heteroclinic orbits.

5. Detection of bifurcation points and limit points.

It is generally better to initially disable the detection of limit points and bifurcation points when computing periodic solutions. In fact, by default these are disabled: ILP=0 and ISP=1.

For example, near a homoclinic orbit a branch becomes vertical and hence all orbits are nearly singular. Also, equations may have perfectly vertical solution branches. In these cases AUTO may try to find one singular point after another, thereby using up unnecessary computer time. For remarks on the detection of bifurcation points see also Section 3.4.

6. The accuracy of Floquet multipliers.

AUTO extracts approximations to the Floquet multipliers from the Jacobian of the linearized system that arises in Newtons method. See the Companion Document for details on how this is done. The procedure is very efficient: multipliers are obtained at negligible extra cost. However, the algorithm has inherent limitations. When all multipliers are of moderate size then they are generally accurate. But Floquet multipliers easily become very large (or small), in which case the algorithm no longer yields accurate approximations. A good check is to verify whether the multiplier at z=1 is still present and reasonably accurate. Note that the orbits will normally retain accuracy even after the computation of the Floquet multipliers breaks down.