APPENDIX B

Solving and Analyzing Dynamical Systems Using XPPAUT

G. Bard Ermentrout

Most of the examples and exercises in the book have been designed to be solvable with the ordinary differential equations solution and analysis package XPPAUT. One reason that we emphasize the use of XPPAUT rather than one of the other available packages is that XPPAUT is distributed at no cost and runs under both Unix and Windows environments. XPPAUT will also run under the new Macintosh operating system OSX with the appropriate Xwindows server. The second reason is that XPPAUT incorporates the bifurcation package AUTO, which is not included in other packages. The Windows version of XPPAUT, Winpp, uses a different bifurcation package, as explained below.

XPPAUT can be obtained from the web site of Bard Ermentrout, the developer. The site contains instructions for the installation of XPPAUT on various platforms, as well as a very useful tutorial. The web site is:

`http://www.math.pitt.edu/~bard/xpp/xpp.html`.

In addition, there is a full–length book describing the details of XPPAUT available (Ermentrout 2002). The tutorial in this appendix will introduce the reader to the main tools available in XPPAUT that are necessary to solve most of the exercises in this book.

B.1 Basics of Solving Ordinary Differential Equations

B.1.1 Creating the ODE File

Consider the simple linear differential equation system

$$\frac{dx}{dt} = ax + by,$$
$$\frac{dy}{dt} = cx + dy, \tag{B.1}$$

where a, b, c, d are parameters. We will explore the behavior of this two-dimensional system using XPPAUT (even though it is easy to obtain a closed–form solution). To analyze a differential equation using XPPAUT, you must create an input file that tells the program the names of the variables and parameters, and defines the equations. By convention, these files have the file extension `ode`, and we will call them ODE files. Here is an ODE file for system (B.1):

```
# linear2d.ode
#
# right hand sides
dx/dt=a*x+b*y
dy/dt=c*x+d*y
#
# parameters
par a=0,b=1,c=-1,d=0
#
# some initial conditions
init x=1,y=0
#
# we are done
done
```

We have included some comments indicated by lines starting with #; these are not necessary but can make the file easier to understand. The rest of the file is fairly straightforward. The values given to the parameters are optional; by default they are set to zero. The `init` statement is also optional. The minimal file for this system is

```
dx/dt=a*x+b*y
dy/dt=c*x+d*y
par a,b,c,d
done
```

In contrast to the more elaborate file, with the minimal file all parameters and intital conditions are set to zero. Use a text editor to type in the first file exactly as it is shown. Name the file `linear2d.ode` and save it. Note also that XPPAUT accepts other notation

for equations, and you should not be surprised to see the more compact version used in Appendix C or in ODE files you might find on the XPPAUT web site or in the XPPAUT user's manual. For example, the minimal file could be written

```
x'=a*x+b*y
y'=c*x+d*y
par a,b,c,d
done
```

That's it! You have written an ODE file. The minimal steps are as follows:

- Use an editor to open a text file.
- Write the differential equations in the file; one per line.
- Use the `par` statement to declare all the parameters in your system. Optionally define initial conditions with the `init` statement.
- End the file with the statement `done`.
- Save and close the file.

ODE FILE NOTES: The equation reader is case-insensitive, so that `AbC` and `abC` are treated as identical. In statements declaring initial conditions and parameters, **do not** put spaces between the variable and the "=" sign and the number. XPPAUT uses spaces as a delimiter. Always write `a=2.5` and **never** write `a = 2.5`.

B.1.2 Running the Program

Run XPPAUT by typing

```
xpp linear2d.ode
```

The name of the exacutable, here `xpp`, might be different for your system. Use the name of your executable, along with all of the desired command line options (see on-line help for details). (*If you are using Winpp, click on the* **Winpp** *icon; then choose the file from the file selection dialog box.*)

Six windows will appear on the screen, or they may be iconified (depending on the command line options). If any of the windows appear "dead" or blank, iconify them manually and then uniconify them. Next time run XPPAUT without the `-iconify` command line option.

Menu commands will appear like this, `Command`, and single–letter keyboard shortcuts will appear like this: **A**. *Do not use the CapsLock key*; all shortcuts are lowercase. Every command can be accessed by a series of keystrokes. To make sure key clicks are interpreted correctly, click on the title bar of the window for which the shortcut is intended.

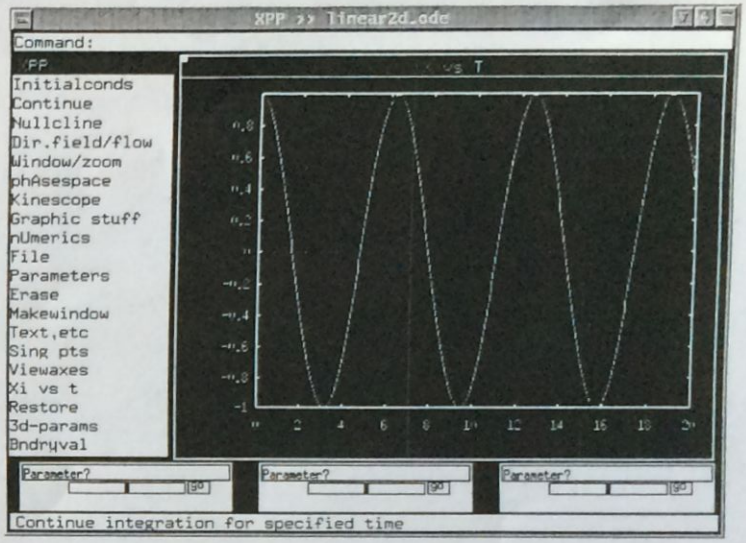

Figure B.1 The main XPPAUT window

B.1.3 The Main Window

The **Main Window** contains a large region for graphics, menus, and various other regions and buttons. It is illustrated in Figure B.1. Commands are given either by clicking on the menu items in the left column with the mouse or tapping keyboard shortcuts. After a while, as you become more used to XPPAUT, you will use the keyboard shortcuts more and more. Both the full commands and the keyboard shortcuts are included here. In general, the keyboard shortcut is the first letter of the command unless there is ambiguity (such as `Nullcline` and `nUmerics`), and then, it is just the capitalized letter (**N** and **U**, respectively). Unlike Windows keyboard shortcuts, the letter key alone is sufficient, and it is not necessary to press the Alt key at the same time. The top region of the **Main Window** is for typed input such as parameter values. The bottom of the **Main Window** displays information about various things as well as a short description of the highlighted menu item. The three little boxes with the words `parameter` are sliders to let you change parameters and initial data.

In addition to the **Main Window**, there are several other windows that appear. The **Equation Window**, shown in Figure B.2, allows you to see the differential equations that you are solving. We will describe the other windows as the tutorial progresses.

Quitting the Program

To exit XPPAUT, click `File` `Quit` `Yes` (**F** **Q** **Y**).

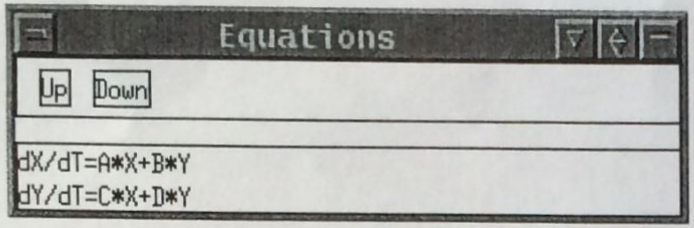

Figure B.2 The Equation Window.

B.1.4 Solving the Equations, Graphing, and Plotting.

Here, we will solve the ODEs, use the mouse to select different initial conditions, save plots of various types, and create files for printing.

Computing the Solution

In the **Main Window** you should see a box with axis numbers. The title in the window should say `X vs T`, which tells you that the variable `X` is along the vertical axis and `T` along the horizontal. The plotting range is from 0 to 20 along the horizontal and −1 to 1 along the vertical axis. When a solution is computed, this view will be shown. Click on `Init Conds` `Go` (**I** **G**) in the **Main Window**. A solution will be drawn followed by a beep. As one would expect given the differential equations, the solution looks like a few cycles of a cosine wave.

Changing the View

To plot `Y` versus `T` instead of `X`, just click on the command `Xi vs t` **X** and choose `Y` by backspacing over `X`, typing in `Y`, and typing **Enter**.

Many times you may want to plot a phase plane instead, that is, `X` vs. `Y`. To do this, click on `Viewaxes` `2D` (**V** **2**), and a dialog box will appear. Fill it in as follows:

X-axis: X	Xmax: 1
Y-axis: Y	Ymax: 1
Xmin: -1	Xlabel:
Ymin: -1	Ylabel:

Click on `OK` when you are done. (Note that you could have filled in the labels if you had wanted, but for now, there is no reason to.) You should see a nice elliptical orbit in the window. This is the solution in the phase plane (cf. Figure B.3).

Phase Plane Shortcuts

There is a very simple way to view the phase plane or view variables versus time. Look at the **Initial Data Window** (Figure B.4). You will see that there are little boxes next to the

ICs en haut à gauche

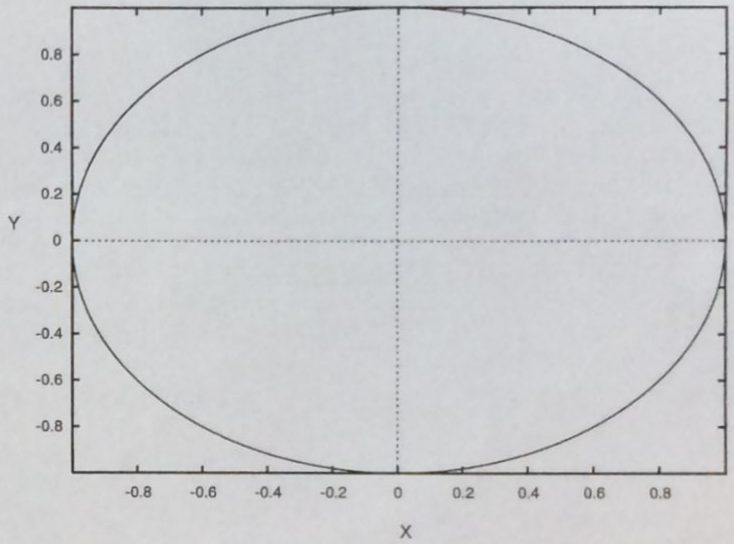

Figure B.3 Phase plane (X vs Y) for the linear 2D problem (B.1).

variable names. Check the two boxes next to `X` and `Y`. Then at the bottom of the **Initial Data Window**, click the `XvsY` button. This will plot a phase plane and automatically fit the window to contain the entire trajectory. This is a shortcut and does not give you the control that the menu command does. (For example, the window is always fit to the trajectory, and no labels are added or changed. Nor can you plot auxiliary quantities with this shortcut.) To view one or more variables against time, just check the variables you want to plot (up to 10) and click on the `XvsT` button in the **Initial Data Window**.

You should have a phase plane picture in the window. (If not, get one by following the above instructions or using the shortcut.) Click on `Init Conds` `Mouse` (**I** **M**.) Use the mouse to click somewhere in the window. You should see a new trajectory drawn. This, too, is an ellipse. Repeat this again to draw another trajectory. If you get tired of repeating this, try `Init Conds` `mIce` (**I** **I**), which, being "mice," is many mouses. Keep clicking in the window. When you are bored with this, click either outside the window or tap the escape key, **Esc**.

Click on `Erase` and then `Restore` (**E** **R**). Note that all the trajectories are gone except the latest one. XPPAUT stores only the latest one. There is a way to store many of them, but we will not explore that for now.

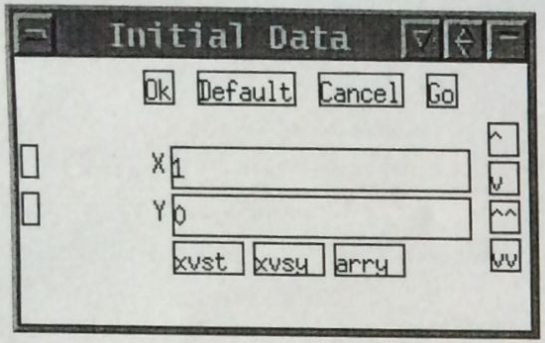

Figure B.4 The initial conditions window.

B.1.5 Saving and Printing Plots

XPPAUT does not directly send a picture to your printer. Rather, it creates a PostScript file that you can send to your printer. If you do not have PostScript capabilities, then you probably will have to use the alternative method of getting hard copy outlined below. (Note that Microsoft Word supports the import of PostScript and Encapsulated PostScript, but can only print such pictures to a PostScript printer. You can download a rather large program for Windows called GhostView which enables you to view and print PostScript on nonPostScript printers. Linux and other UNIX distributions usually have a PostScript viewer included.)

Here is how to make a PostScript file. Click on Graphics Postscript (G P), and you will be asked for three things: (i) Black and White or Color (ii) Landscape or Portrait; (iii) and the Fontsize for the axes. Accept all the defaults for now by just clicking Enter. Finally, you will be asked for a file name. The File Selector box is shown in Figure B.5. You can move up or down directory trees by clicking on the <>; choose files by clicking on them; scroll up or down by clicking on the up/down arrows on the left or using the arrow keys and the PageUp/PageDown keys on the keyboard; change the wild card; or type in a file name. For now, you can just click on Ok and a PostScript plot will be created and saved. The file will be called `linear2d.ode.ps` by default, but you can call it anything you want.

Once you have the PostScript file, you can type

```
lpr filename
```

on UNIX. In Windows, if your computer is hooked up to a PostScript printer, then you can print from a viewing application or type

```
copy filename lpt1
```

from the command line (if available).

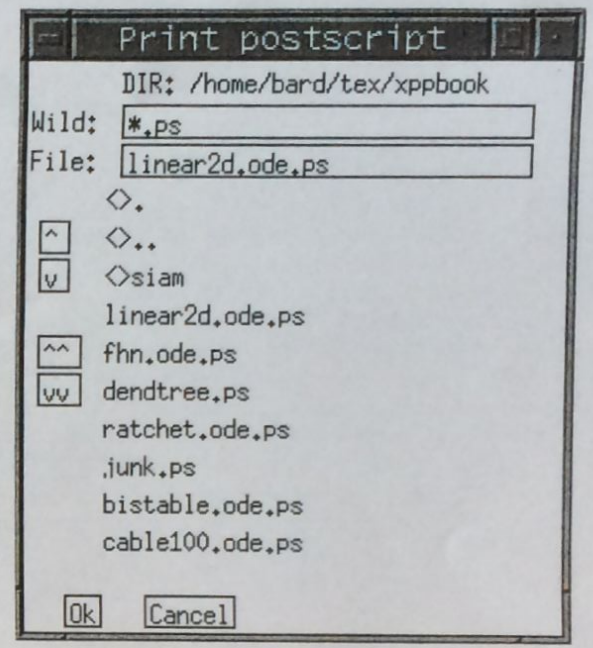

Figure B.5 File selector.

Other Ways to get Hard Copy

Another way to get hard copy that you can import into documents is to grab the image from the screen. In Windows, click on Alt+PrtSc after making the desired window active. Paste this into the MSPaint accessory and then use the tools in Paint to cut out what you want. Pasting into Microsoft Word is useful for generating reports with added text. Alternatively, you can download a number of programs that let you capture areas of the screen. In the UNIX environment, you can capture a window using `xv`, an excellent utility that is free and available for most UNIX versions. All of the screen shots in this tutorial were captured with `xv`. Finally, you can capture the screen (or a series of screen images) with the Kinescope Capture command and then write these to disk with the Kinescope Save command. This produces a series of GIF files that are usable by many software packages.

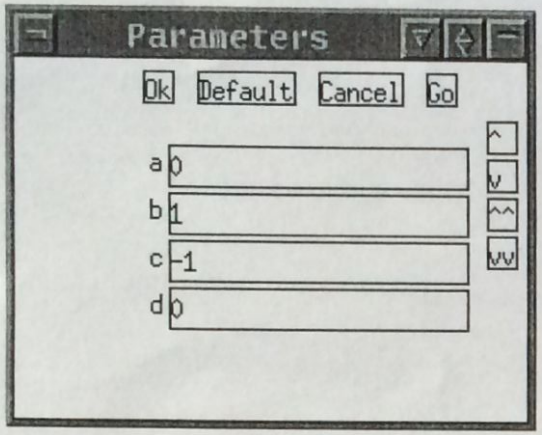

Figure B.6 The parameter window.

Getting a Good Window

If you have computed a solution and do not have a clue about the bounds of the graph, let XPPAUT do all the work. Click on `Window/zoom` `(F)it`, and the window will be resized to a perfect fit. The shortcut is **W** **F** and you will likely use it a lot!

B.1.6 Changing Parameters and Initial Data

There are many ways to vary the parameters and initial conditions in XPPAUT. We have already seen how to change the initial data using the mouse. This method works for any n-dimensional system as long as the current view is a phase plane of two variables. Here are two other ways to change the initial data:

- From the main menu click on `Init Conds` `New` and manually input the data at the prompts. You will be prompted for each variable in order. (For systems with hundreds of variables, this is not a very good way to change the data!)
- In the **Initial Data Window** you can edit the particular variable you want to change. Just click in the window next to the variable and edit the value. Then click on the `Go` button in the **Initial Data Window**. If there are many variables, you can use the little scroll buttons on the right to go up and down a line or page at a time. If you click the mouse in the text entry region for a variable, you can use the **PageUp**, etc., keys to move around. Clicking **Enter** rolls around in the displayed list of initial conditions. The `Default` button returns the initial data to those with which the program started. If you do not want to run the simulation, but have set the initial data, *you must* click on the `Ok` button in the **Initial Data Window** for the new initial data to be recognized.

  There are many ways to change parameters as well. Here are three of them:

- From the **Main Window**, click on `Parameters`. In the command line of the **Main Window**, you will be prompted for a parameter name. Type in the name of a parameter that you want to change. Click on `Enter` to change the value and `Enter` again to change another parameter. Click on `Enter` a few times to get rid of the prompt.
- In the **Parameter Window** (shown in Figure B.6) type in values next to the parameter you want to change. Use the scroll buttons or the keyboard to scroll around. As in the **Initial Data Window**, there are four buttons across the top. Click on `Go` to keep the values and run the simulation; click on `Ok` to keep the parameters without running the simulation. Click on `Cancel` to return to the values since you last pressed `Go` or `Ok`. The `Default` button returns the parameters to the values when you started the program.
- Use the little sliders (Figure B.7). We will attach the parameter `d` to one of the sliders. Click on one of the unused parameter sliders. Fill in the dialog box as follows:

Parameter: d
Value: 0
Low: -1
High: 1

  and click `Ok`. You have assigned the parameter `d` to one of the sliders and allowed it to range between -1 and 1. Grab the little slider with the mouse and move it around. Watch how `d` changes. Now click on the tiny `go` button in the slider. The equations will be integrated. Move the slider some more and click on the `go` button to get another solution.

B.1.7 Looking at the Numbers: The Data Viewer

In addition to the graphs that XPPAUT produces, it also gives you access to the actual numerical values from the simulation. The **Data Viewer** shown in Figure B.8 has many buttons, some of which we will use later in the book. The main use of this is to look at the actual numbers from a simulation. The independent variable occupies the leftmost column, and the dependent variables fill in the remaining windows. Click on the top of the **Data Viewer** to make it the active window. The arrow keys and the **PageUp**,

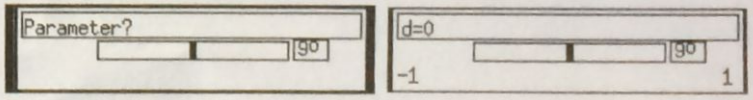

Figure B.7 Left: Unused parameter slider. Right: parameter slider used for d.

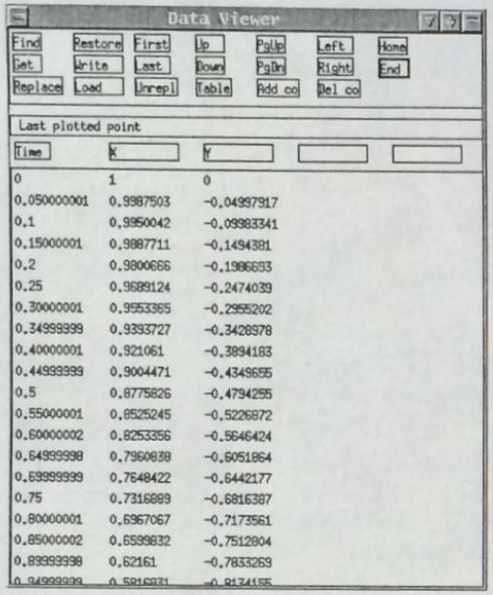

Figure B.8 The Data Viewer.

PageDown, **Home**, and **End** keys (as well as their corresponding buttons) do all the obvious things. Left and right keys scroll horizontally, a useful feature if you have many variables. Here we mention three buttons of use:

`Find` brings up a dialog box prompting you for the name of a column and a value. If you click on `Ok`, XPPAUT will find the entry that is closest and bring that row to the top. You can find the maximum and minimum, for example, of a variable.
`Get` loads the top line of the **Data Viewer** as initial data.
`Write` writes the entire contents of the browser to a text file that you specify.

B.1.8 Saving and Restoring the State of Simulations

Often you will have a view, a set of parameters, and initial data that you want to keep. You can save the current state of XPPAUT by clicking on `File` `Write set` (**F** **W**) in the **Main Window**. This brings up a file selection box. Type in a file name; the default extension is `.set`. The resulting file is an ASCII file that is human and computer readable. The first and last few lines look like this:

```
## Set file for linear2d.ode on Fri Aug  4 13:53:31 2000
2   Number of equations and auxiliaries
4   Number of parameters
```



```
# Numerical stuff
1    nout
40    nullcline mesh

.......

RHS etc ...
dX/dT=A*X+B*Y
dY/dT=C*X+D*Y
```

Once you quit XPPAUT, you can start it up again and then use the `File` `Read set` to load up the parameters, etc., that you saved.

Now you should quit the program. We will look at a nonlinear equation next, find fixed points, and draw some nullclines and direction fields. To quit, click on `File` `Quit` `Yes` (**F** **Q** **Y**).

B.1.9 Important Numerical Parameters

XPPAUT has many numerical routines built into it, and thus there are many numerical parameters that you can set. These will be dealt with in subsequent sections of the book where necessary. However, the most common things you will want to change are the total amount of time to integrate and the step size for integration. You may also want to change the method of integration from the default fixed–step Runge–Kutta algorithm. To alter the numerical parameters, click on `nUmerics` (**U**), which produces a new menu. This is a top–level menu, so you can change many things before going back to the main menu. To go back to the main menu, just click on the `[Esc]-exit` or tap **Esc**. There are many entries in the numerics menu. The following four are the most commonly used:

`Total` sets the total amount of time to integrate the equations. (Shortcut: **T**.)
`Dt` sets the size of the time step for the fixed step size integration methods and sets the output times for the adaptive integrators. (Shortcut: **D**.)
`Nout` sets the number of steps to take before *plotting* an output point. Thus, to plot every fourth point, change `Nout` to 4. For the variable step size integrators, this should be set to 1.
`Method` sets the integration method. There are currently 13 available. (Shortcut: **M**.) They are described in the user manual.

When you are done setting the numerical parameters, just click on `Esc-exit` or tap the Esc key.

B.1.10 Command Summary: The Basics

Initialconds | Go computes a trajectory with the initial conditions specified in the **Initial Data Window** (I G).

Initialconds | Mouse computes a trajectory with the initial conditions specified by the mouse. Initialconds | m(I)ce lets you specify many initial conditions (I M or I I).

Erase erases the screen (E).

Restore redraws the screen (R).

Viewaxes | 2D lets you define a new 2D view (V 2).

Graphic stuff | Postscript allows you to create a PostScript file of the current graphics (G P).

Kinescope | Capture allows you to capture the current view into memory, and Kinescope | Save writes this to disk.

Window/zoom | (F)it fits the window to include the entire solution (W F).

File | Quit exits the program (F Q).

File | Write set saves the state of XPPAUT(F R).

File | Read set restores the state of XPPAUT from a saved .set file (F R).

B.2 Phase Planes and Nonlinear Equations

Here we want to solve a nonlinear equation. We will choose a planar system, since there are many nice tools available for analyzing two-dimensional systems. A classic model is the FitzHugh–Nagumo equations, which are used as a model for nerve conduction. The equations are

$$\frac{dv}{dt} = Bv(v-\beta)(\delta - v) - Cw + I_{\text{app}}, \qquad \text{(B.2)}$$
$$\frac{dw}{dt} = \epsilon(v - \gamma w),$$

with parameters $I_{\text{app}}, B, C, \beta, \delta, \epsilon, \gamma$. Here we will use $I_{\text{app}} = 0, B = 1, C = 1, \beta = .1, \delta = 1, \gamma = 0.25$, and $\epsilon = .1$. Let us write an ODE file for this:

```
# Fitzhugh-Nagumo equations
dv/dt=B*v*(v-beta)*(delta-v)-Cw+Iapp
dw/dt=epsilon*(v-gamma*w)
par Iapp=0,B=1,C=1,beta=.1,delta=1,gamma=.25,epsilon=.1
@ xp=V,yp=w,xlo=-.25,xhi=1.25,ylo=-.5,yhi=1,total=100
@ maxstor=10000
done
```


We have already seen the first four lines: (i) lines beginning with a # are comments, (ii) the next two lines define the differential equations, and (iii) the line beginning with par defines the parameters and their default values. The penultimate line beginning with the @ sign is a directive to set some of the options in XPPAUT. These could all be done within the program, but this way everything is all set up for you. Details of these options are found in the user manual. For the curious, these options set the x-axis (xp) to be the v variable, the y-axis (yp) to be the w variable, the plot range to be $[-.15, 1.25] \times [-.5, 1]$, and the total amount of integration time to be 100. The last option, @ maxstor=10000, is a very useful one. XPPAUT allocates enough storage to keep 4000 time points. You can make it allocate as much as you want with this option. Here we have told XPPAUT to allocate storage for 10000 points. Type this in and save it as fhn.ode.

B.2.1 Direction Fields

Run the file by typing xpp fhn.ode. The usual windows will pop up. One of the standard ways to analyze differential equations in the plane is to sketch the *direction fields*. Suppose that the differential equation is

$$\frac{dx}{dt} = f(x,y), \qquad \frac{dy}{dt} = g(x,y).$$

The phase plane is divided into a grid, and at each point (x,y) in the grid a vector is drawn with (x,y) as the base and $(x+sf(x,y), y+sg(x,y))$ as the terminal point, where s is a scaling factor. This so-called direction field gives you a hint about how trajectories move around in the plane. XPPAUT lets you quickly draw the direction field of a system. Click on Dir.field/flow | (D)irect Field (D D) and then accept the default of 10 for the grid size by clicking Enter. A bunch of vectors will be drawn on the screen, mainly horizontal. They are horizontal because ϵ is small so that there is little change in the w variable. The length of the vectors is proportional to the magnitude of the flow at each point. At the head of each vector is a little bead. If you want to have pure direction fields that do not take into account the magnitude of the vector field, just click on Dir.field | (S)caled Dir. Fld (D S) and use the default grid size. (We prefer pure direction fields, but this is a matter of taste.)

Click on Initialconds | m(I)ce to experiment with a bunch of different trajectories. Note how the vectors from the direction field are tangent to the trajectories. See Figure B.9.

B.2.2 Nullclines and Fixed Points

As discussed in earlier chapters, a powerful technique for the analysis of planar differential equations and related to the direction fields is the use of *nullclines*. Nullclines are curves in the plane along which the rate of change of one or the other variable is zero. The x-nullcline is the curve where $dx/dt = 0$, that is, $f(x,y) = 0$. Similarly, the y-nullcline is the curve where $g(x,y) = 0$. The usefulness of these curves is that they break the plane

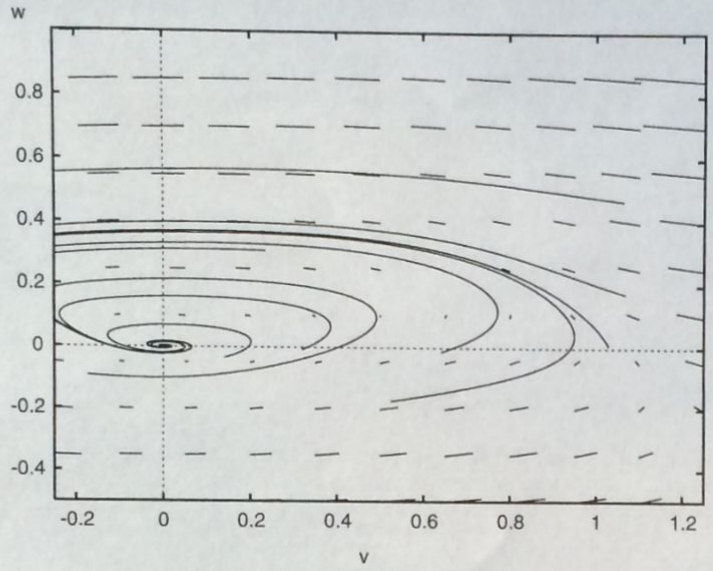

Figure B.9 Direction fields and some trajectories for the FitzHugh–Nagumo equations.

up into regions along which the derivatives of each variable have a constant sign. Thus, the general direction of the flow is easy to determine. Furthermore, any point where they intersect represents a fixed point of the differential equation.

XPPAUT can compute the nullclines for planar systems. To do this, just click on `Nullcline` `New` (**N** **N**). You should see two curves appear: a red one representing the V-nullcline and a green one representing the W-nullcline. The green one is a straight line, and the red is a cubic. They intersect just once: There is a single fixed point. Move the mouse into the phase plane area and hold it down as you move it. At the bottom of the **Main Window** you will see the x and y coordinates of the mouse. The intersection of the nullclines appears to be at $(0, 0)$.

The stability of fixed points is determined by linearizing the system of equations about them and then finding the eigenvalues of the resulting linear matrix. XPPAUT will do this for you quite easily. XPPAUT uses Newton's method to find the fixed points and then numerically linearizes the system about them to determine stability. To use Newton's method, a decent guess needs to be provided. For planar systems this is easy to do; it is just the intersection of the nullclines. In XPPAUT fixed points and their stability are found using the `Sing pts` command, since "singular points" is a term sometimes used for fixed points or equilibrium points. Click on `Sing pts` `Mouse` (**S** **M**) and move the mouse to near the intersction of the nullclines. Click

the button, and a message box will appear on the screen. Click on `No`, since we do not need the eigenvalues. A new window will appear that contains information about the fixed points. The stability is shown at the top of the window.

The nature of the eigenvalues follows: **c+** denotes the number of complex eigenvalues with positive real part; **c-** is the number of complex eigenvalues with negative real part; **im** is the number of purely imaginary eigenvalues; **r+** is the number of positive real eigenvalues; and **r-** is the number of negative real eigenvalues. Recall that a fixed point is linearly stable if all of the eigenvalues have negative real parts. Finally, the value of the fixed points is shown under the line. As can be seen from this example, there are two complex eigenvalues with negative real parts: the fixed point is $(0, 0)$. (XPPAUT reports a very small nonzero fixed point due to numerical error.) Integrate the system using the mouse, starting with initial conditions near the fixed point. (In the **Main Window**, tap **I** **I**.) Note how solutions spiral into the origin, as is expected when there are complex eigenvalues with negative real parts.

For nonplanar systems of differential equations you must provide a direct guess. Type your guess into the **Initial Data Window** and click on `Ok` in the **Initial Data Window**. Then from the **Main Window**, click on `Sing Pts` `Go` (**S** **G**).

Change the parameter `I` from 0 to 0.4 in the **Parameter Window** and click on `Ok` in the **Parameter Window**. In the **Main Window** erase the screen and redraw the nullclines: `Erase` `Nullclines` `New` (**E** **N** **N**). The fixed point has moved up. Check its stability using the mouse (`Sing pts` `Mouse`). The fixed point should be $(0.1, 0.4)$. Use the mouse to choose a bunch of initial conditions in the plane. All solutions go to a nice limit cycle. That is, they converge to a closed curve in the plane representing a stable periodic solution.

We can make a nice picture that has the nullclines, the direction fields, and a few representative trajectories. Since XPPAUT keeps only the last trajectory computed, we will "freeze" the solutions we compute. We can freeze trajectories automatically or one at a time, and we will do the former. Click on `Graphic stuff` `(F)reeze` `(O)n freeze` (**G** **F** **O**) to permanently save computed curves. Up to 26 can be saved in any window. Frist we use the mouse to compute a bunch of trajectories. Draw the direction fields by clicking `Dir.field/flow` `(D)irect Field` (**D** **D**).

We can label the axes as follows: Click on `Viewaxes` `2D` (**V** **2**), and the 2D view dialog will come up. Change nothing but the labels (the last two entries), and put `V` as the **Xlabel** and `w` as the **Ylabel**. Click on `Ok` to close the dialog. Finally, since the axes are confusing in the already busy picture, click on `Graphic stuff` `aXes opts` (**G** **X**) and in the dialog box change the 1's in the entries **X-org(1=on)** and **Y-org(1=on)** to 0's to turn off the plotting of the X and Y axes. Click `Ok` when you are done.

To create a PostScript file, follow `Graphic stuff` `(P)ostscript` (**G** **P**) and accept all the defaults. Name the file whatever you want and click on **Ok** in the file selection box. Figure B.10 shows the version that we made. Yours will be slightly different. If you want to play around some more, turn off the automatic freeze option,

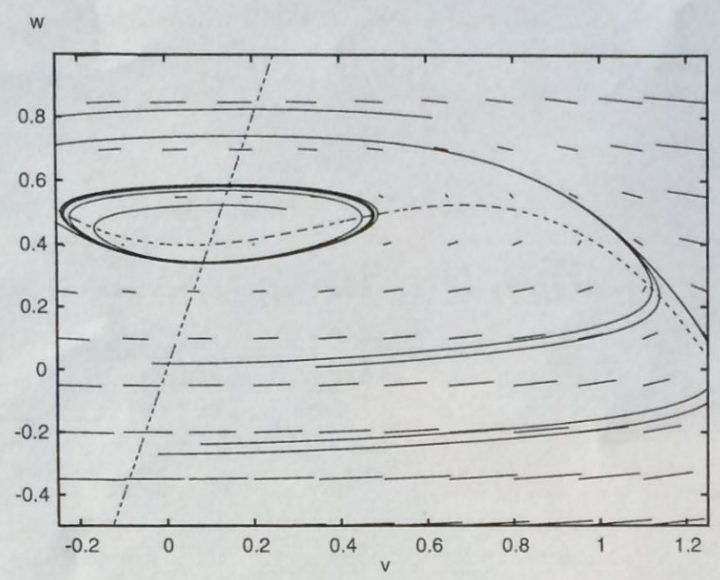

Figure B.10 Nullclines, direction fields, trajectories for `Iapp=0.4` in the Fitzhugh-Nagumo equations.

`Graphic stuff` `Freeze` `Off freeze` (G F O), and delete all the frozen curves, `Graphic stuff` `Freeze` `Remove all` (G F R).

B.2.3 Command Summary: Phase Planes and Fixed Points

`Nullcline` `New` draws nullclines for a planar system (N N).

`Dir.field/flow` `(D)irect Field` draws direction fields for a planar system (D D).

`Sing pts` `Mouse` computes fixed points for a system with initial guess specified by the mouse (S M).

`Sing pts` `Go` computes fixed points for a system with initial guess specified by the current initial conditions (S G).

`Graphic stuff` `Freeze` `On Freeze` will permanently keep computed trajectories in the current window (G F O).

`Graphic stuff` `Freeze` `Off Freeze` will toggle off the above option (G F O).

`Graphic stuff` `Freeze` `Remove all` deletes all the permanently stored curves (G F R).

`Graphic stuff` `aXes opts` lets you change the axes (G X).

`Viewaxes` `2D` allows you to change the 2D view of the current graphics window and to label the axes (V 2D).

B.3 Bifurcation and Continuation

Once we have found the critical points of a system of interest, we can then embark on a continuation and bifurcation analysis of the solutions, as we mentioned in Appendix A. Continuation analysis describes how solutions to differential equations evolve over parameters, while bifurcation analysis refers, in particular, to how solutions appear and disappear as parameters are varied. One of the main strengths of XPPAUT is that it provides a convenient interface to many of the features found in the AUTO package for continuation/bifurcation analysis. AUTO remains one of the best such packages. However, the stand-alone versions of AUTO require coding compilation of the equations with the FORTRAN computer language. Note that AUTO currently only is available for the Unix version of XPPAUT. The Windows version uses a continuation package called LOCBIF, and slight differences from the procedures outlined below are explained in the user's manual on the web site. While AUTO is powerful even as implemented in XPPAUT, it is not foolproof. The results you obtain should be viewed with a critical eye. Bifurcation analysis is discussed in more depth in an excellent book by Kuznetsov (Kuznetsov 1998). It is useful to understand that the AUTO features available in XPPAUT are independent of the other tools, but that parameters and fixed points are exchanged back and forth. Bifurcation diagrams can be imported into XPPAUT for plotting.

B.3.1 General Steps for Bifurcation Analysis

- AUTO bifurcation analysis must start from a fixed (or singular) point or from a periodic orbit. Get rid of transients and find a stable fixed point by by integrating several times: Click `Initialconds` `Go` and then click `Initialconds` `Last` several times. For limit cycles, get a good estimate of the period and integrate one full period only.
- Bring up the **AUTO window** by selecting `File` `Auto` (F A).
- Use the `Parameter` function to choose the parameters that will be varied.
- Use the `Axes` `Hilo` function to select the parameters to be plotted and the range over which they will be varied. A one–parameter bifurcation diagram must be completed before a second parameter can be varied.
- Use the `Numerics` function to define direction, step size, etc.

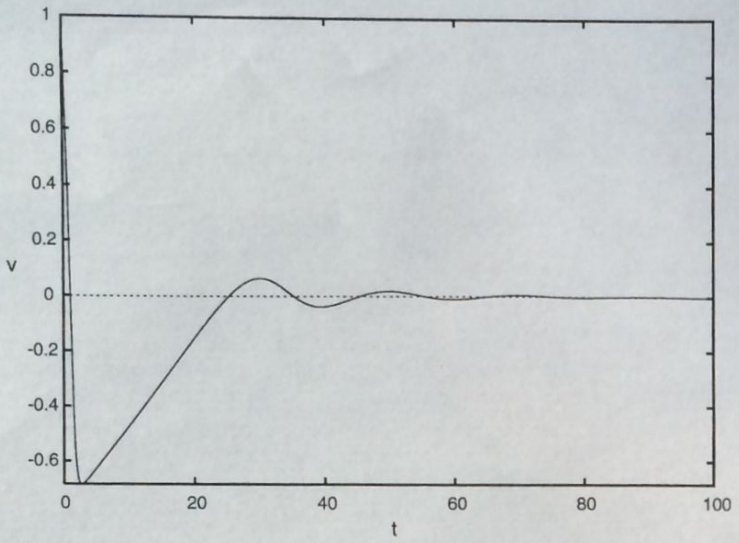

Figure B.11 FH-N system with `Iapp=0`.

- Run the analysis for Steady state or Periodic.

It is more difficult for AUTO to start from a periodic solution, and so some further assistance is required. First, a good approximation of the period must be determined from the data browser (use Find in conjunction with a large number to determine the maximum) or by measuring the peak-to-peak period with the mouse. Integrate over just that period by adjusting nUmerics Total to the period length. After starting AUTO and selecting parameters and bounds, choose Run Periodic. AUTO may still fail, and further adjustments to numerical parameters or a better approximation of the period may be necessary.

B.3.2 Hopf Bifurcation in the FitzHugh–Nagumo Equations

We will continue with the FitzHugh-Nagumo example and explore the bifurcation structure of this system. Using the FH-N equations from earlier, add the lines

```
v(0)=1,
w(0)=1,
```

to the ODE file to set initial conditions. Now we are ready to begin the analysis:

1. Start up XPPAUT with `Iapp=0`. Next, run the ODE file, ploting `x vs t` as discussed above. You should see v oscillate a bit and go to zero, as seen in Figure B.11. Now run the simulation with `Iapp=1.5`. You should see the periodic solution shown in Figure B.12. We want to understand how this change ocurrs as the parameter `Iapp` is changed.
2. To set up the bifurcation analysis run the simulation again with `Iapp=0`: Click Initialconds Go and then click Initialconds Last several times. This will run the simulation until it is really at the steady state.
3. The next step is to bring up the **AUTO window** by selecting File Auto (F A). Once the **AUTO window** is present, make sure that `Iapp` is listed as `Par1` under Parameter. It should already be there if you typed the file in as written. If not, select `Iapp`.
4. Set up the graphics axes with Axes Hilo. Fill in the dialog box as follows:

Y-axis: V
Main Parm: Iapp
Xmin: -0.5
Ymin: -3
Xmax: 0.5
Ymax: 0.5

5. Set up the Numerics, and change only `Par Min=0` and `Par Max=3.5`.

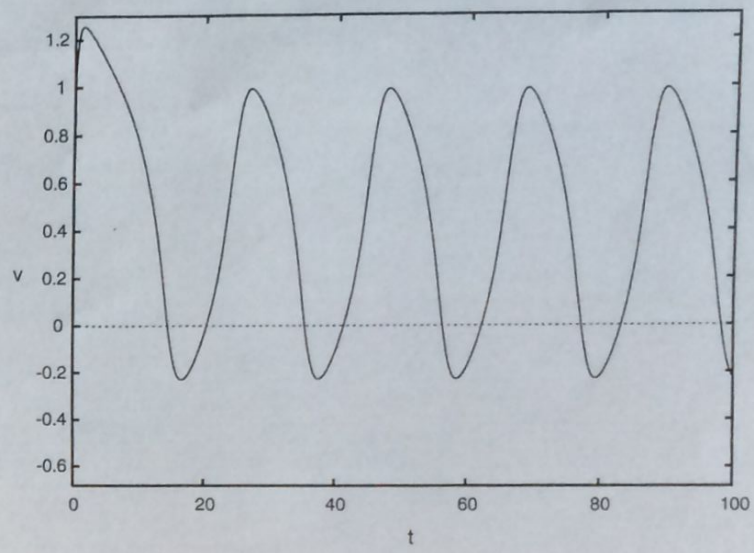

Figure B.12 FH-N system with `Iapp=1.5`.

6. To begin, click Run | Steady state. The beginning of the diagram should appear with four points labeled as in Figure B.13. The bold line represents a stable steady state, and the faint line represtents an unstable steady state.
7. Now choose Grab. This will allow you to see what the four marked points in the diagram represent. A cross appears on the plot at marker 1, and below the plot, a description will be present. Lab is the label for the point (1), Ty is the type of point. EP stands for End point, where we started computing. Iapp is the value of the parameter for that location of the graph.
8. Move the cursor over the (2). Under Ty should be the label HP, for Hopf bifurcation. Point (3) should also be a Hopf bifurcation, and (4) will be the other end point.
9. With the cross back on (2), press **Enter**. This "grabs" that point as the beginning of a new calculation. Click Run again. This time, the pop-up screen is labeled as Hopf Pt, and we will choose Periodic to follow the periodic orbits as Iapp changes. You should get something like Figure B.14. The darkened circles show that the periodic orbits are stable. Open circles represent an unstable periodic solution. Note that there are upper and lower points in the plot for the periodic solution, showing the maximum and minimum values (of v) that the solution attains.
10. By using grab again, we can go to the periodic orbits, and their period will be shown below the plot.
11. We can save the plot using the File menu as discussed above.

B.3.3 Hints for Computing Complete Bifurcation Diagrams

- Be sure to start at a fixed point or clearly defined limit cycle. As discussed above, get rid of transients and find a stable fixed point by integrating several times: Click

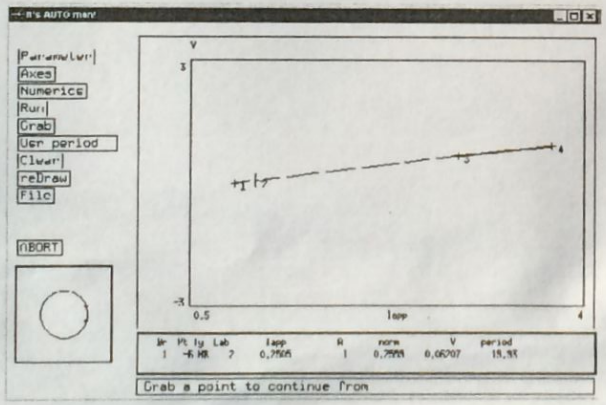

Figure B.13 Initial bifurcation diagram and AUTO window for the FH-N system.

Initialconds | Go and then Initialconds | Last several times. For limit cycles, get a good estimate of the period and integrate one full period only.

- Learn to navigate the diagram efficiently. Tab jumps to special points, Axes | Fit | reDraw will fit the entire diagram to the page, and Axes | Zoom magnifies a given area.
- AUTO will try to follow all branches of fixed points. However, AUTO may need some assistance. Grab special points and Run in different directions by changing the sign of **Ds** in the numerics dialog box.
- Try to find the periodic solution from all Hopf points.
- Be sure also to change **Ds** for two parameter bifurcations.
- An initial **MX** label indicates that auto has failed and you may not have provided a good fixed point or periodic orbit.
- To erase the diagram and start again with different parameters, Grab the initial starting point and destroy the diagram with File | Reset diagram.
- If AUTO fails to continue, try making **dsmin** smaller; for periodic orbits and boudary value problems make **ntst** larger.
- If AUTO clearly misses a bifurcation point, make **dsmin** smaller and recompute.

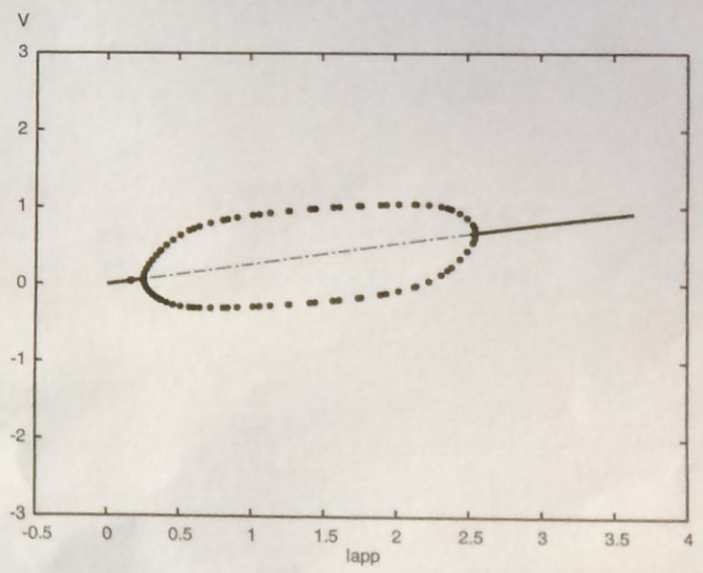

Figure B.14 Final bifurcation diagram the FH-N system.