
≡ Using TraX ≡

**A Tutorial to Accompany TraX,
A Program for the Simulation and Analysis of Dynamical Systems**

version 1.1

by

**Alexander I. Khibnik, Ph.D.
Research Computing Center, Academy of Sciences, USSR**

**The first in a series of Soviet Scientific Software Products.
Edited by Jeffrey A. Millstein, Ph.D., Applied Biomathematics**

distributed by

**Exeter Software
Setauket, NY**

Produced in the United States of America
© Copyright 1990 by Applied Biomathematics
All Rights Reserved

First published 1990

TraX version 1.1 Single User License

The computer software TraX and this documentation are sold with the understanding that they will be used by only one person at a time (like using a book). They may be used by one person on more than one machine (like taking a book home). They may also be used by more than one person if the software is installed on just one computer (like a reference book). If any defects are found in the programs or documentation, Applied Biomathematics will, at its option, make corrections or refund the purchase price. Applied Biomathematics will not be liable for any incidental or consequential damages, loss of revenues, grants or data that may arise from use or misuse of the software.

The program TraX was written by Dr. V.V. Levitin of the Research Computing Center, Soviet Academy of Sciences. In the creation of the program TraX, Drs. A.I. Khibnik and A.M. Molchanov served as dynamical systems advisors to Dr. Levitin. This manual (and the archive tut) was written by Dr. A.I. Khibnik of the Research Computing Center, Soviet Academy of Sciences, and requires the TraX program. The program TraX, the TraX reference manual, and Using TraX, are distributed by Exeter Software. English versions of all manuals distributed with the TraX software were edited by Dr. J.A. Millstein of Applied Biomathematics.

Distributed by:
Exeter Software, Inc.
100 North Country Road
Setauket, New York 11733, USA

Technical information: (516)-751-4350
To place an order: 1-(800)-842-5892.
Facsimile: (516)-751-3435

Table of Contents

<u>TraX for Beginners</u>	1
Typographic conventions used in this tutorial	2
Lesson 1. Starting TraX and navigating through the TraX Archives	3
Lesson 2. How to simulate ordinary differential equations	8
Lesson 3. Interacting with the Parameter window	13
Lesson 4. How to specify equations in TraX	15
Lesson 5. Interacting with Graphic windows in TraX	18
Lesson 6. More about Graphic window functions	21
6.1. Window limits	22
6.2. Using expressions to label axes	22
6.3. Cylindric phase space	23
Lesson 7. The use of colors in TraX	27
Lesson 8. How to simulate difference equations	28
Lesson 9. The microscope facility	38
<u>Advanced TraX</u>	42
Lesson 10. Using functions in equations	43
Lesson 11. The chaotic dynamics of nonlinear systems	53
11.1. Chaotic behavior	53
11.2. The sensitive dependence phenomena.	54
11.3. Homoclinic trajectory.	55
11.4. Next-maximum map.	58
11.5. Poincaré map.	59
11.6. Three-dimensional plot.	60
11.7. Lyapunov exponent.	63
Lesson 12: Phase portraits, bifurcation diagrams and keystroke macros	66
12.1. Phase portrait	66
12.2. Bifurcation diagram	67
Lesson 13. Stochastic modeling in TraX	70
References Cited	75

TraX for Beginners

Welcome to "Using TraX", the TraX tutorial. This tutorial has been specifically written to accompany TraX version 1.1. When you purchased TraX you received two manuals, a Reference Manual and this tutorial called "Using TraX". In attempting to keep things brief and to the point, many of the details of TraX are found only in the Reference Manual. You do not need to study the Reference Manual before using this tutorial but you should at least review the first three sections of the Reference Manual to familiarize yourself with TraX before working on these lessons. Although performing the lessons in this tutorial will teach you most of the important TraX commands and give you a good appreciation of the power of TraX to analyze and simulate dynamical systems, you may, however, want to have the Reference Manual handy while going through the lessons here. Specifically, Appendix A of the Reference Manual which lists all of the key sequences of TraX and is especially useful for TraX beginners.

Typographic conventions used in this tutorial

The following typographic conventions are used throughout this manual:

Typeface	Description
KEY TERMS	Text in <code>Courier</code> typeface indicates a specific term, punctuation or mark that you must type in exactly as shown, although case is not significant. This typeface also indicates the response of the program or what you'll see on the screen.
KEY NAMES	The names of keys or key sequences are shown as they appear on the keyboard, for example, the function key F3 is shown as <code>F3</code> and the two-key combination CTRL-F7 is shown as <code>Ctrl-F7</code> . Whenever a two-key combination appears, the first key (the Control, Alt or Shift key) must be depressed first, and while holding this key down, press the second key and release both keys. Note that <code>␣</code> is the backspace key while <code>←</code> is the left directional arrow.
<code>↵</code>	This is the ENTER key and whenever it is pressed the information entered preceding the ENTER key is sent to the program for processing. This symbol is shaded to emphasize its importance.
<i>f(x), dx/dt, 2π, φ</i>	Mathematical equations are printed in an italicized typeface.

Lesson 1. Starting TraX and navigating through the TraX Archives

In this first lesson you'll learn how to invoke TraX and how to navigate through the TraX Archives. The TraX Archives is a collection of equations which are stored by you, the user. The Archives has a tree-like form and its division into branches or levels as well as the names of the sections are determined by you. In addition, there may be many different Archives although only one Archive can be loaded per TraX session. Multiple Archives, however, are especially useful in multi-user settings, such as in the classroom.

Each level of the Archives stores either an equation, a group of equations or an equation state. An equation state is a set of values which are related to specific equations, these include such things as the values of system parameters or screen attributes which were saved during a TraX session.

The stored equation states allow you to restart a TraX simulation using the related equations from the point where you left off and therefore, you can interrupt a TraX session at almost anytime and save your work quite easily. Another useful application of archived equations and stored states is using them as an example to speed up the development of new models. For example, if you decide to extend or modify a model system which is already stored in the Archives, you can essentially copy your previous work into a new Archive section, edit this model and save it with a new name.

In addition to the Archives mode there is the Investigation mode where you create and edit models and conduct simulations. Keep in mind, however, that the only way to reach the Investigation mode is to pass through the Archives, which is why we're discussing the Archives first. Let's now enter TraX from DOS by following the commands below (we're assuming that the TraX programs and the Archives called tut all reside in a directory on drive C: called, appropriately TRAX (or the TraX programs all reside in a directory on your DOS PATH):

```
C:\TRAX: trax tut ↵
```

This command invokes TraX and automatically loads the Archive named tut, which is short for tutorial. In a few seconds the opening screen should appear (Figure 1). When you see this screen, press any key and you will be placed into the Archives and you'll be, appropriately enough, in the Archives mode. You'll also be located at the first or root level of the Archives (Figure 2a). To travel through the Archives, use the directional arrow keys (↑, ↓, ← and →). The ↑ and ↓ keys move the cursor within a current Archive level and the → and ← keys move you to the next or the previous level, respectively. Now we'll follow the Archive path, beginning from the root level, which goes Differential equations → Predator-prey model → (Figure 2). Now on the screen (and reproduced in Figure 2a) you'll see that some

entries are marked with an asterisk (*). This indicates that this level contains *groups* of equations. Notice however, that in Figure 2b each particular equation is marked by =, and in Figure 2c, stored states related to the equations named Predator-prey model are marked by either a + or a -, indicating whether a plot (picture) is stored (+) or not (-).

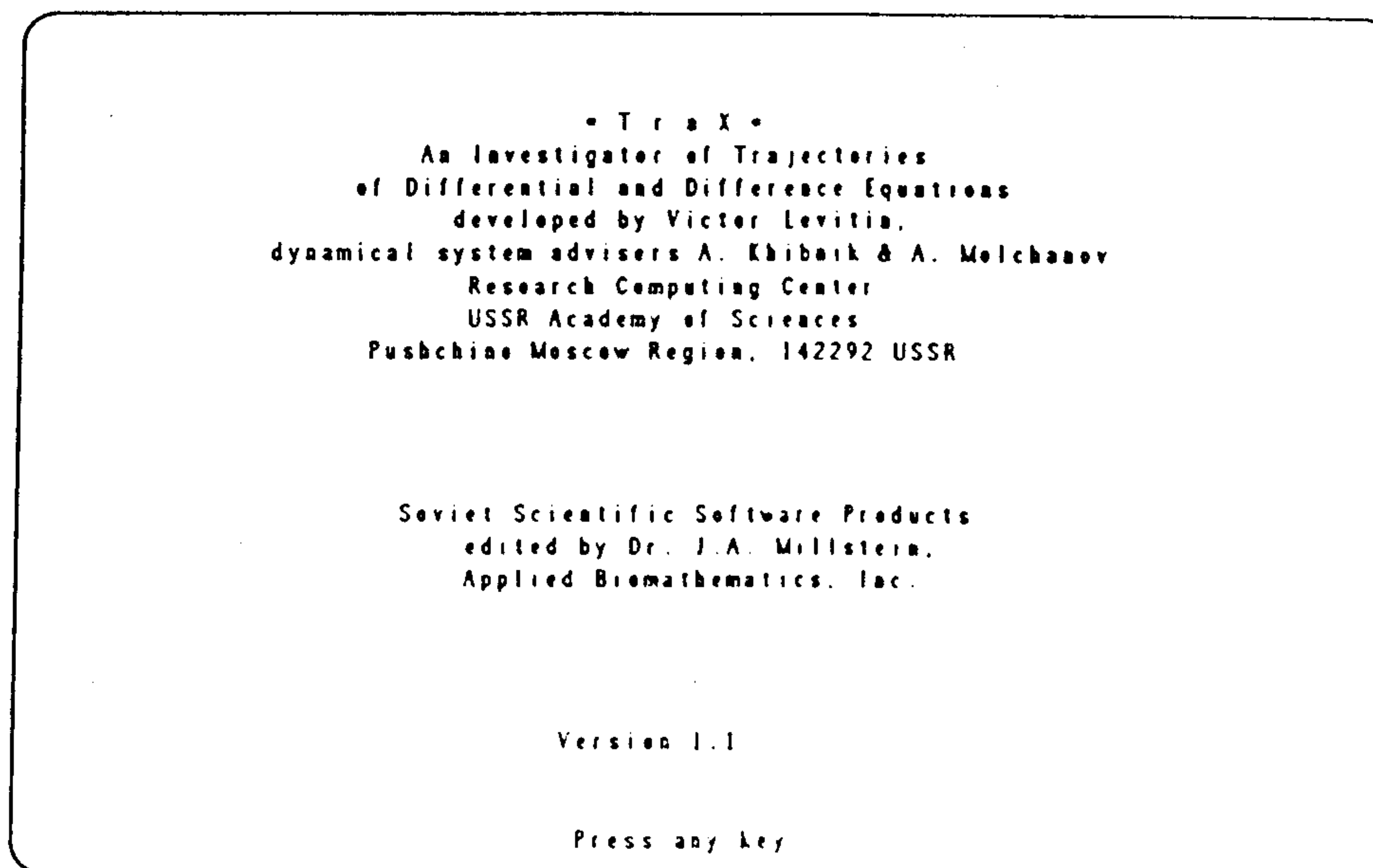


Figure 1. The opening screen of TraX.

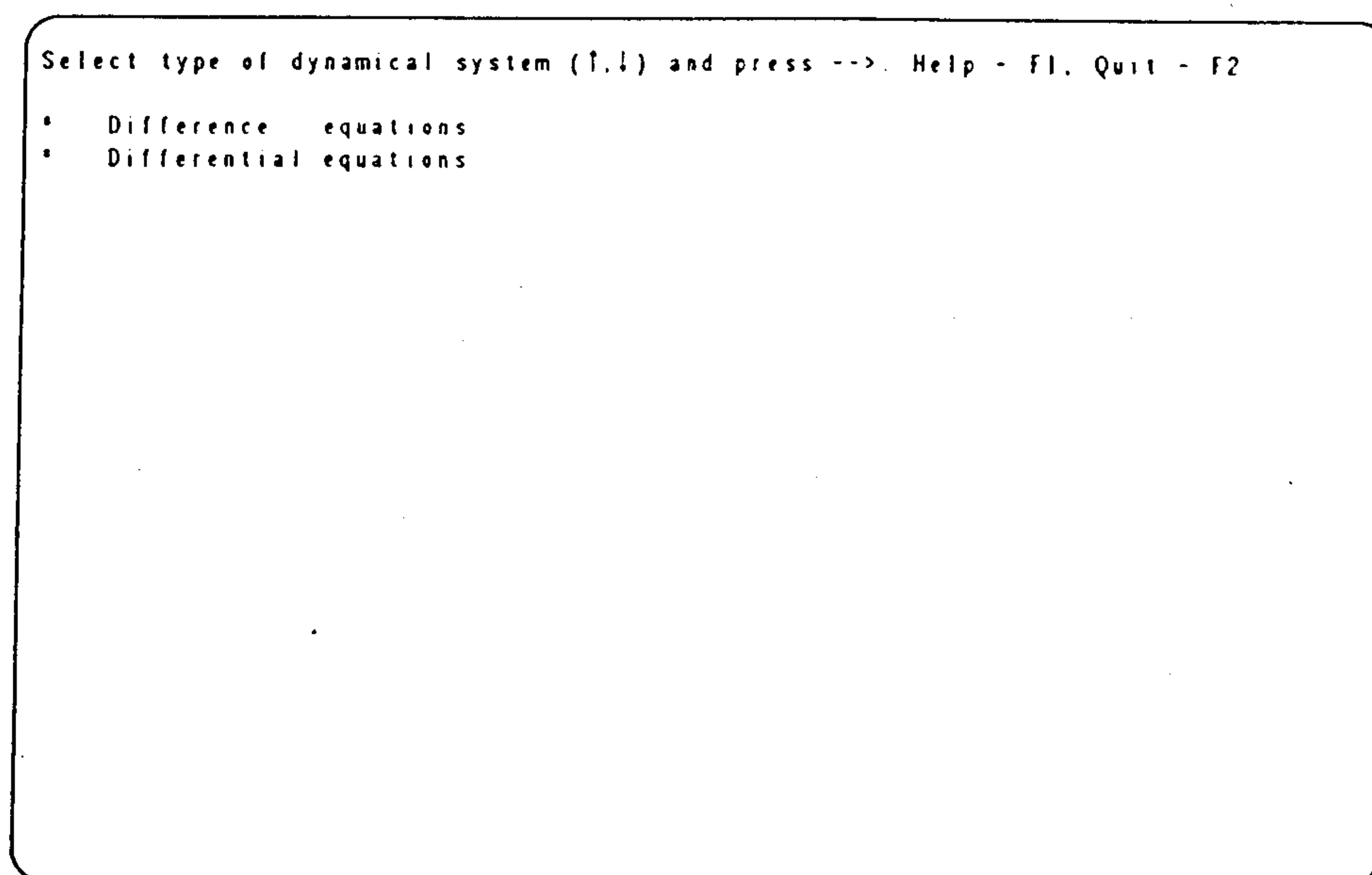


Figure 2a. The structure of the TraX Archives: the two root levels of the archive are marked with an asterisk (*) indicating that they are groups of equations.

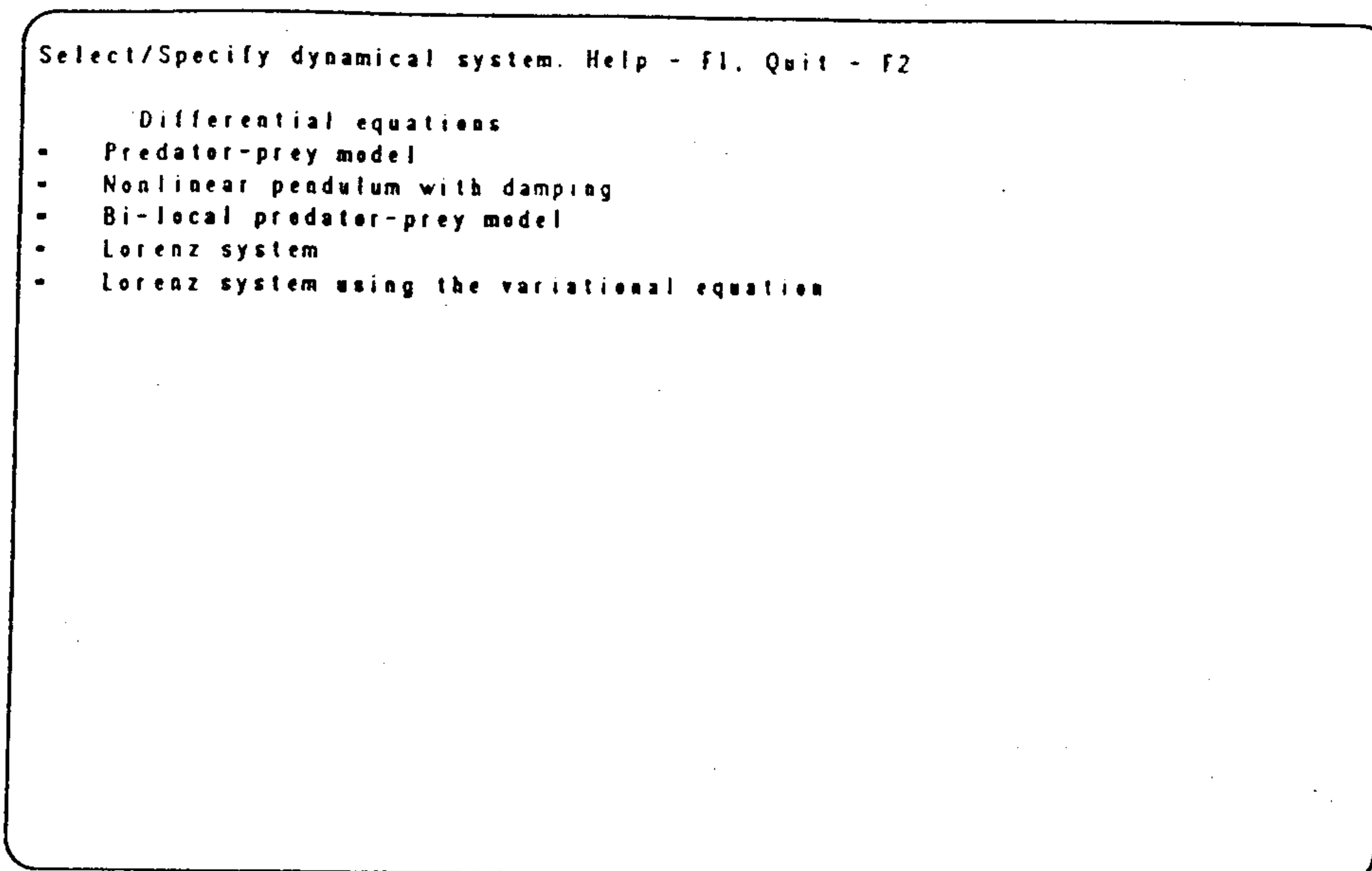


Figure 2b. The structure of the TraX Archives: the second level under the Differential equations. Equation groups are marked by =.

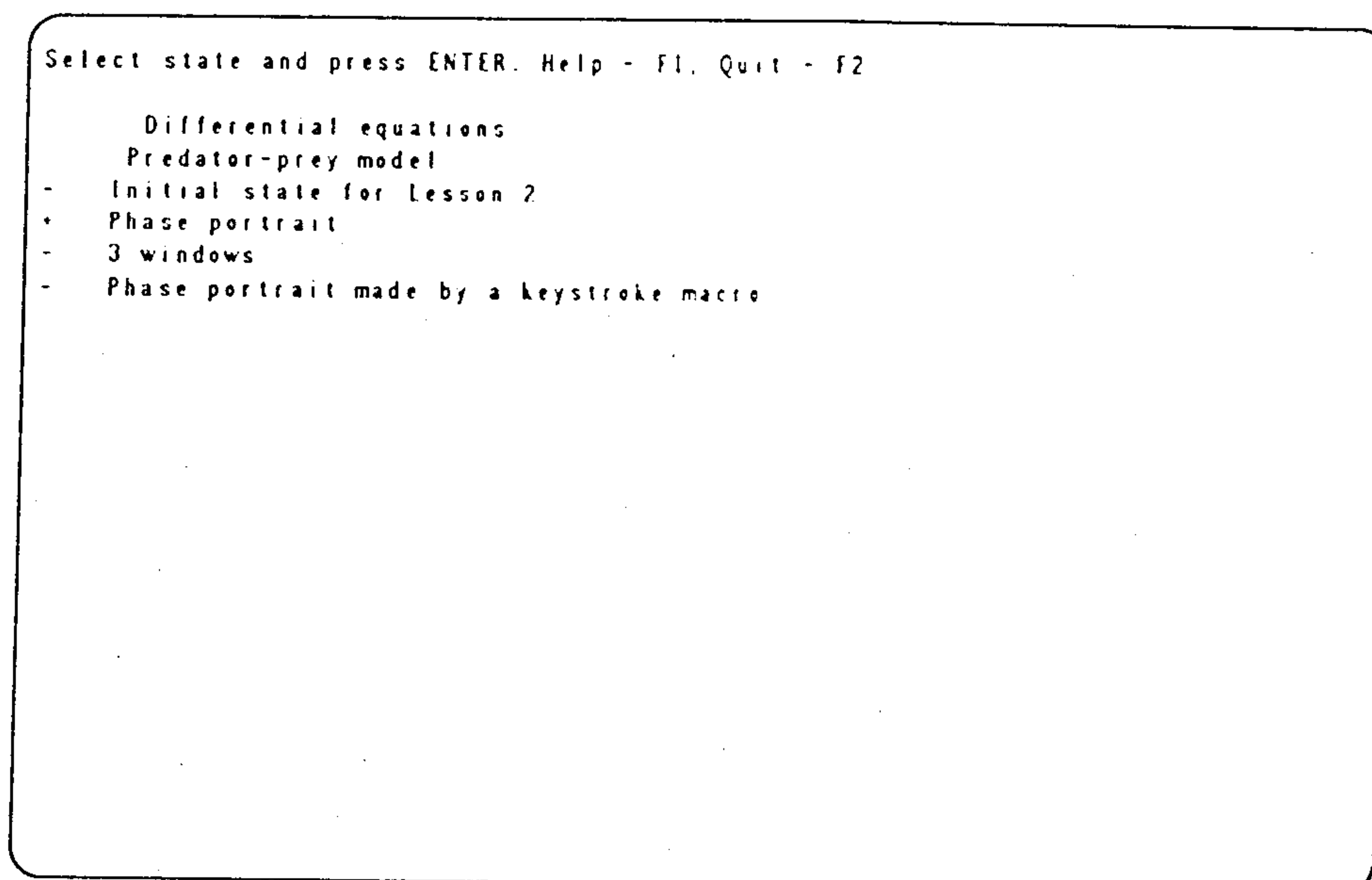


Figure 2c. The structure of the TraX Archives: the third level under the equation group Predator-prey model containing stored states. These are marked with a + or a - to indicate whether they contain a stored picture (+) or not (-).

Within the TraX Archives the two root levels, Differential equations and Difference equations are the only fixed levels within the TraX Archives; they cannot be renamed or deleted. The creation and naming of all other Archive levels,

however, is under your control. Figure 2 illustrates that following the two root levels all the other levels have been ordered according to the lessons in this tutorial, although there are many possible ways of ordering these systems. For example, you could classify the equations by their dimension or by the subject that they deal with.

The equations included in the Archives may be displayed on the screen. For example, to display the equations named Predator-prey model, set the cursor on this entry and press **[Shift-F1]**. The screen should now appear as shown in Figure 3. Press any key to return to the Archives. You should now travel through the Archives to review the content. You will see some equations and states which are used in the following lessons. Also notice the content of the on-line Help system which is accessible by pressing **[F1]**; Help is available throughout TraX although the content of the Help screens depends upon the TraX mode.

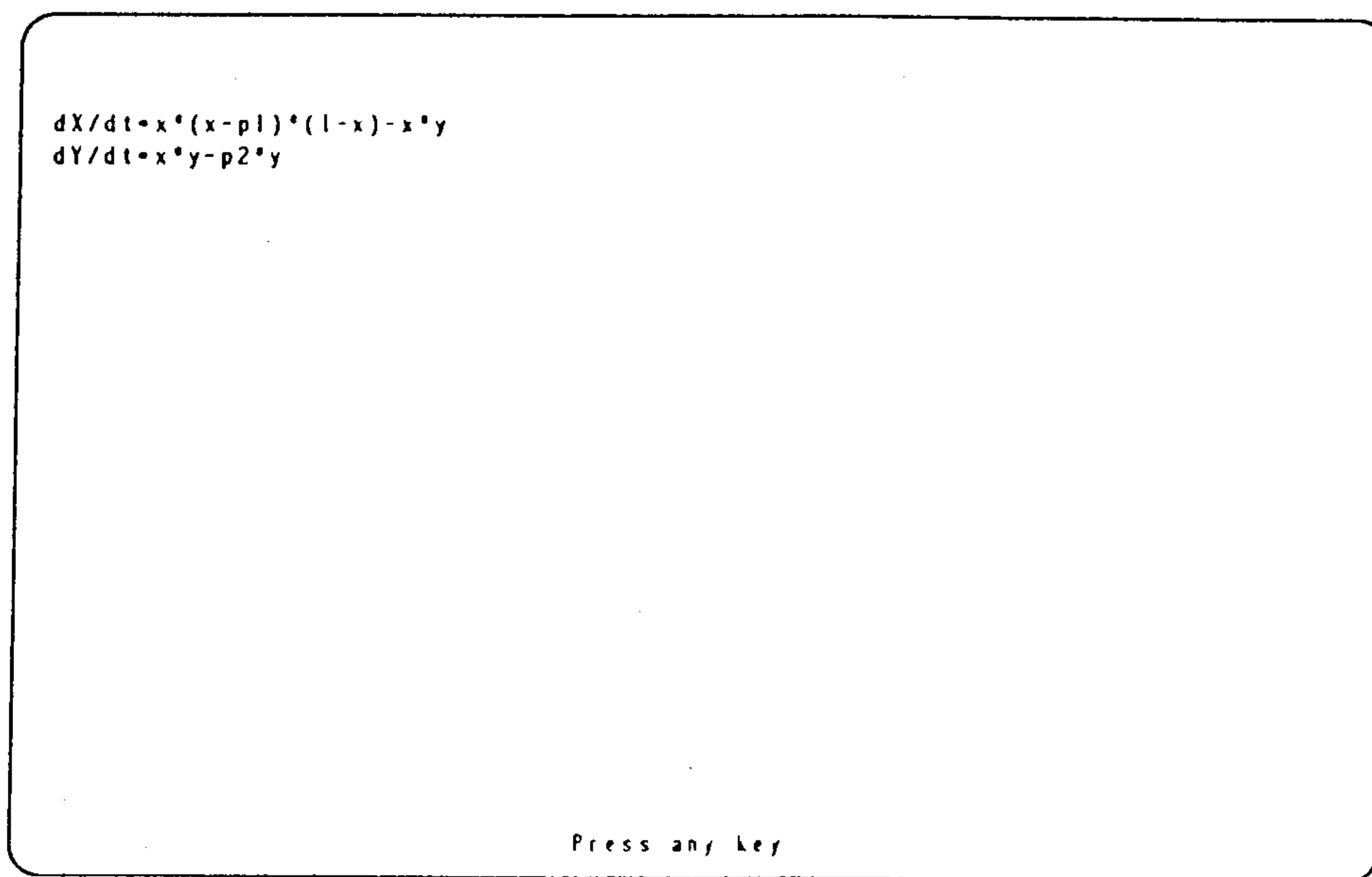


Figure 3. The display of the equations stored in the Archive as Predator-prey model.

Now we'll learn how to introduce a new group of equations, rename them and delete them from the Archives. First, move to a position within the Archives as shown in Figure 2b. Next, press **[Home]** to create a new level or more precisely, a new vertex within the tree. When a prompt appears, enter the name of this new equation group and then press **[Enter]** followed by **[Enter]**. Following this, the name you entered will appear in the list marked by an asterisk (*). Press **[Enter]** to make sure that this group is empty and then press **[Enter]** to return to where you started. To rename this equation group press **[F2]** and edit the name in the standard way. Finally, to delete this equation group

press **[Del]** and after the prompt confirm your intention to delete by pressing **[Y]** (for Yes). After you feel comfortable using the Archives press **[F2]** to quit TraX and return to DOS.

Lesson 2. How to simulate ordinary differential equations

In this lesson, we'll learn to evaluate and draw the trajectories of ordinary differential equations (ODEs). We're assuming that the equations which we'll use for this lesson are already in the TraX Archives (Archive tut). For this lesson we're going to use the predator-prey model of Bazykin (1985):

$$\begin{aligned}\frac{dx}{dt} &= x(x - \alpha)(1 - x) - xy \\ \frac{dy}{dt} &= xy - by\end{aligned}\tag{1}$$

In this system, x represents the density of the prey population, y represents the density of the predator population, and α and b are parameters. To begin, invoke TraX from DOS and in the Archives find the differential equations named Predator-prey model. Press **Shift-F1** to display these equations. Notice that TraX uses specific notation for specifying equations (see Figure 3). For example, x and y represent the state variables x and y , and $p1$ and $p2$ represent the parameters α and b , respectively. After checking the equations, press **→** and enter the list of the related states. Select the state Initial state for Lesson 2 by placing the cursor on this state and press **←**. Now we're ready to begin Lesson 2.

When the Investigation mode is loaded the screen will appear as in Figure 4. There are three areas on the screen: the Parameter window on the right, the Graphic window on the left and the Information line on the top of the screen.

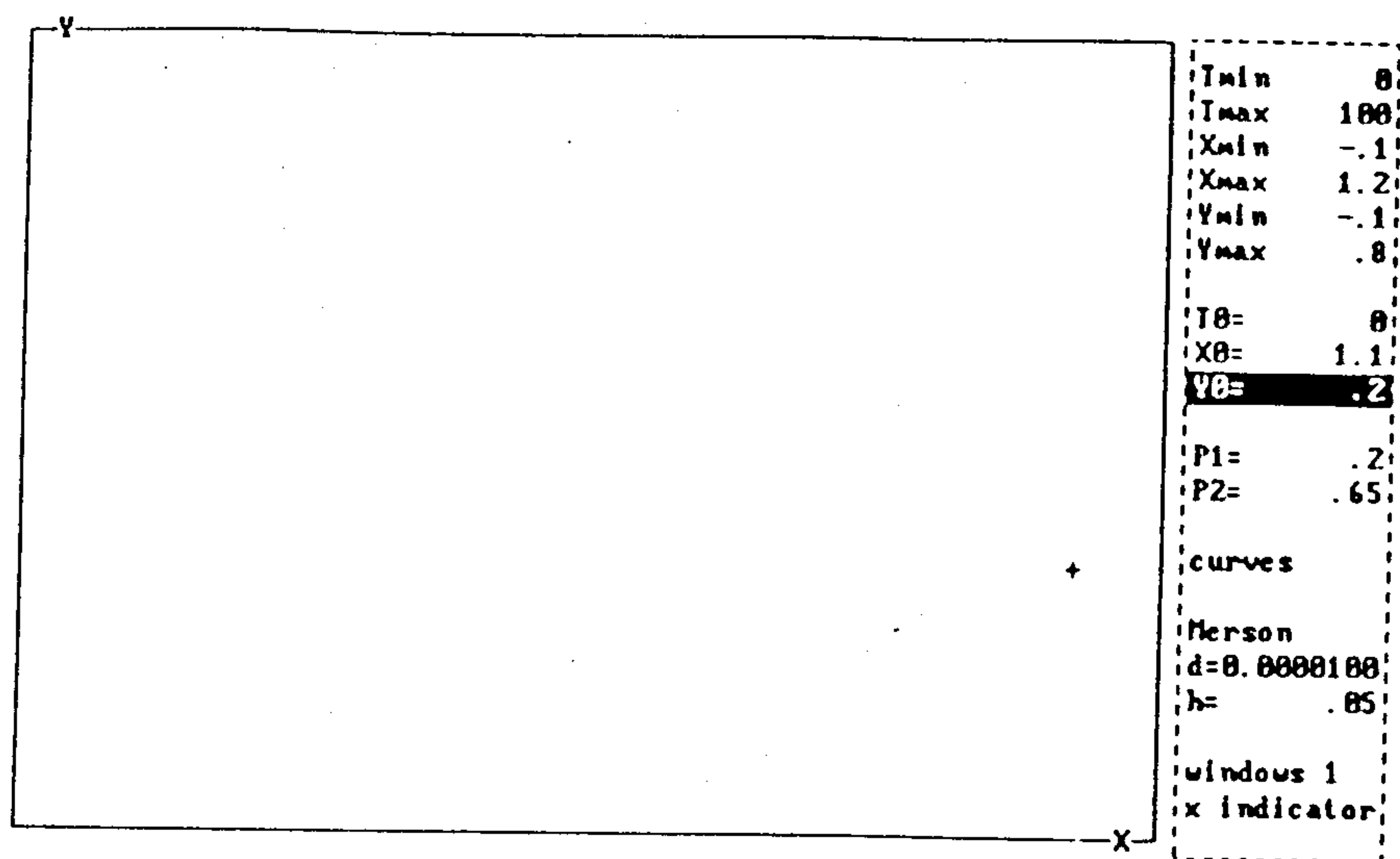


Figure 4. In the Investigation mode, the screen contains the Parameter window (right), one or more Graphic windows (left), and the Information line (top).

The Parameter window exhibits numerical values and indicators (or toggles). There are six entries which set the limits of the graphic windows to visualize the solution curves, these are: $T_{min}=0$, $T_{max}=100$, $X_{min}=-0.1$, $X_{max}=1.2$, $Y_{min}=-0.1$, and $Y_{max}=0.8$. The entry $T_0=0$ represents the initial value of time variable t (i.e., T_0 is regarded as a lower boundary and T_{max} is the upper boundary of the integration interval). The entries $X_0=1.1$ and $Y_0=0.2$ show the current value or position of the initial conditions and the entries $P_1=0.2$ and $P_2=0.65$ display the current parameter values of the model. The other information in the Parameter Window will be discussed in next few lessons so don't worry about it now.

The Graphic window shows a portion of the phase plane $x - y$ which corresponds to the above limits. The position of initial point of the trajectory is marked by a plus sign (+); this is also referred to as the *Initial Point Indicator* or IPI. Now, press **(F8)** to plot the axes and then press **(F10)** to begin evaluating and plotting the trajectory (in TraX this is done simultaneously!). The trajectory will appear in the Graphic window as shown in Figure 5. The elongating bar on the left side of the Graphic window indicates the progress of the computation over the specified integration interval. During computation the limits of the current time interval, as well as current integration step h , are displayed in the Information Line on the top of the screen. The integration step may be varied automatically by the ODE solver to minimize computation time and control accuracy. When the simulation is finished the Information Line will exhibit the final coordinates of the trajectory.

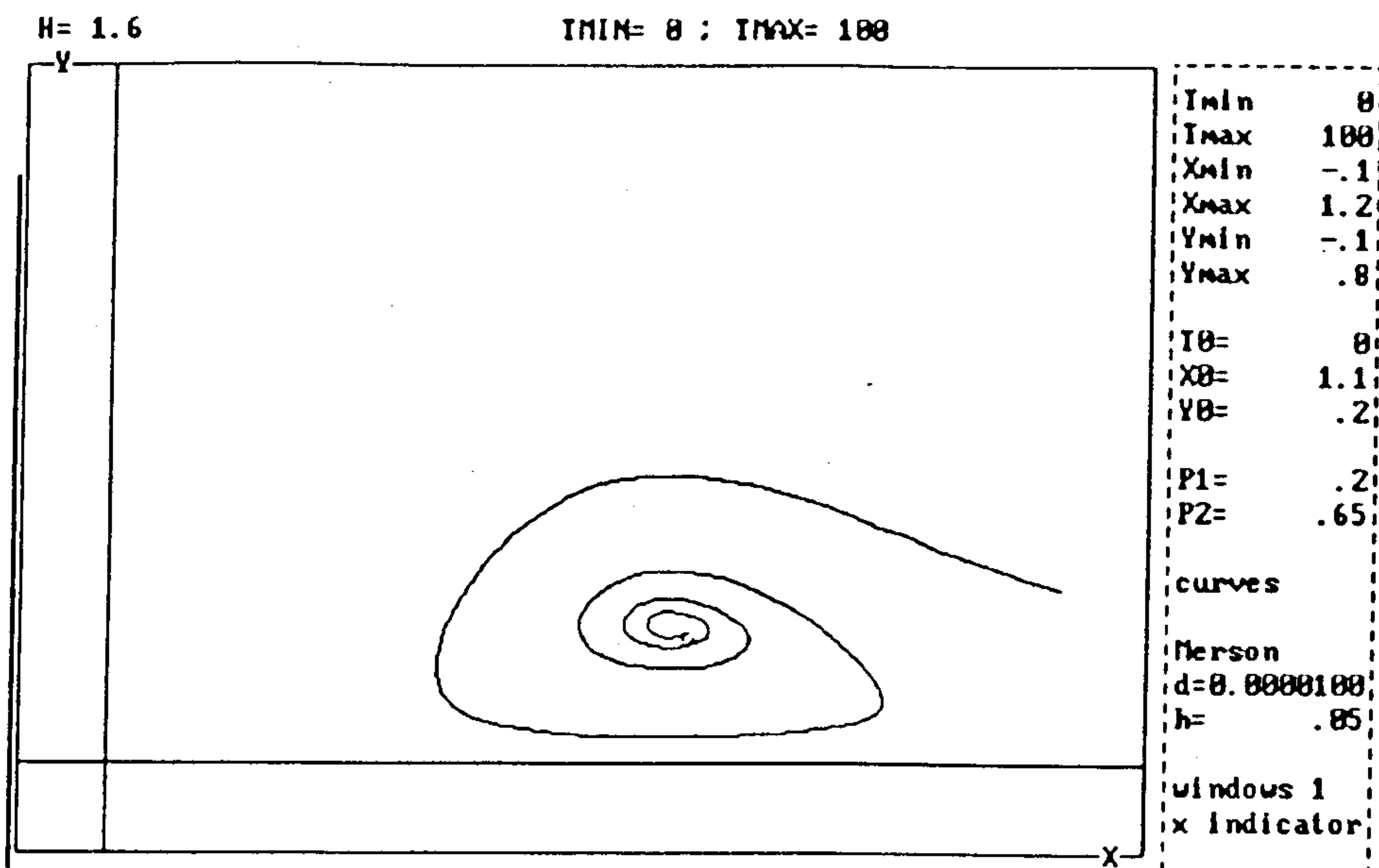


Figure 5. Computing and plotting a trajectory of the predator-prey model (1) with axes plotted.

To interrupt a simulation at any time, press **[Esc]**. Press **[F10]** once more to continue computing the trajectory following an interruption or if the time interval has ended. If the time interval has ended, the simulation will proceed using the same time interval length as before. Thus, **[F10]** starts or continues a simulation. Set some new initial values and compute a new trajectory (**[F10]**). To set a new initial point, move the IPI in the Graphic window by pressing either **[↑]**, **[↓]**, **[←]** or **[→]**. Notice that when you move the IPI the corresponding initial values in the Parameter window are updated. Use **[↔]** to move the IPI more quickly (first specify the direction to move the IPI with the directional arrow keys and then use **[↔]**). When you have set the new initial point, press **[F10]** to begin evaluating the new trajectory.

If the solution does not fall within the graphics window the computation will still proceed, although you will not see the results drawn on the screen. When this happens a special marker (a reverse color .x) will appear in the upper left hand corner of the Graphic window. If the IPI is out of bounds you can still run a simulation but only that part of the trajectory which falls within the window limits will be observed. You can also simulate trajectories in the backward time direction (for ODEs only, however) and in this case, t is decreased starting from T_0 . This type of investigation can give you some idea of the "prehistory" the trajectory. To initiate a simulation in the backward direction press **[F9]**.

You should be aware, however, that a backward or reverse time simulation often leads to a decrease in the integration step because of variables which grow without limit (in absolute values). When no or little progress is being made you can either interrupt the simulation by yourself (**Esc**) or you can wait until TraX aborts the simulation because the minimum step size Hmin is reached (the default value of Hmin is 1.E-07)¹.

Now we'll construct a phase portrait of the model as shown in Figure 6. First clear the graphic window using (**F7**) and re-draw the axes (**F8**). Next, compute several new trajectories beginning with initial points which lie on a vertical line; this means that the initial density of predator is varied but the prey density is held constant. The phase portrait in Figure 6 shows some differences between high and low initial predator densities. This particular case illustrates the asymptotic state the system reaches after a long time. If the initial predator density is low enough, the system will stabilize and reach an equilibrium or a steady-state. And for a large initial density, the system will collapse; the organisms go extinct and the trajectory goes to the origin. There is some critical predator density, however, separating these two cases which gives rise to the so-called separatrix trajectory. This critical density is an important boundary between two regions which correspond to a surviving or an extinct system. You can find the approximate location of the separatrix trajectory by choosing the appropriate initial points to begin the model simulation. It should be noted, however, that the relationship between the critical predator and prey densities has a nonlinear character and this follows from the shape of separatrix trajectory.

¹In the Parameter window this value may appear as 0.1D-06 indicating double precision.

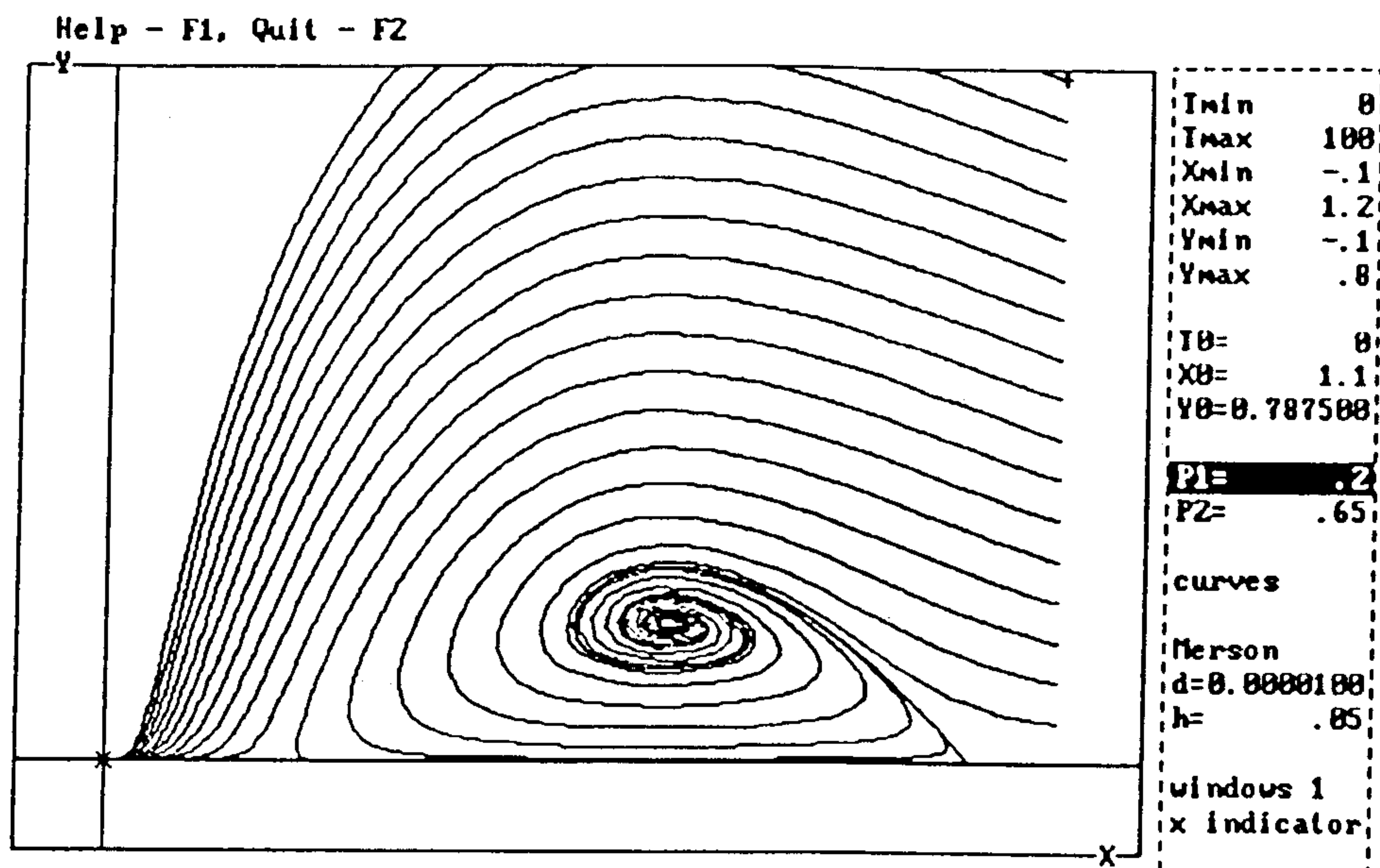


Figure 6. The phase portrait of the predator-prey model (1) plotted by varying Y_0 while holding X_0 constant.

Before finishing this lesson and quitting, save the current state and the plot. Press **F3** to save the state; you will be asked to name the state. Enter Lesson 2, phase portrait, for example, and when the prompt Save the plot? (Y/N) comes up, reply **Y**. Next, leave the Investigation mode by pressing **F2** and you'll then be placed in the Archives. Make sure that name of the state appears in the state list and then press **F2** to quit TraX and return to DOS.

Lesson 3. Interacting with the Parameter window

In this lesson you will learn how to interact with the Parameter window. Let's start with the equation state which we stored in Lesson 2 (*i.e.*, if you have just started TraX, select the equations `Predator-prey model` and the stored state in the Archives, and enter the Investigation mode). The state `Phase portrait` is also suitable and if this state is selected, the saved picture will appear on the screen when you enter the Investigation mode.

You can only change an entry in the Parameter window if it is highlighted (*i.e.*, appears in reverse color), and therefore, to change any entry you must highlight it first. To move the highlight use `PgUp` or `PgDn` (not `↑` and `↓` because these keys move the IPI in the Graphic window).

Let's try, for example, to change the initial value `X0`. Highlight the entry `X0=...` and type the number 0.5. When entering the new value it will appear first on the Information line at the top of your screen, and after you press `↵` the highlighted value will be updated. Notice that the IPI will be moved too. Reset the value of `X0` to 1.1, the value prior to pressing `↵`. You can use `Shift-Esc` to return the parameter value to the previous setting prior to pressing `↵`.

Now let's turn to the entries in the Parameter window which we have not yet discussed (*see* Figure 4). Highlight the entry `curves`. This entry toggles between trajectories drawn as curves and trajectories plotted using unconnected points. If you press `↵`, curves toggles to points. To try this option, clear the Graphic window (`F7`), redraw the axes (`F8`) and initiate the simulation (`F10`). The trajectory will be plotted by points. Press `↵` to return to the curves setting which seems more natural for displaying trajectories of differential equations (or more generally, of continuous-time dynamic systems).

The next entry below, `Merson`, indicates a type of ODE solver (an algorithm). The `Merson` setting corresponds to the fourth-order Runge-Kutta-Merson method with accuracy control. By toggling this entry you can select `Euler` which indicates the second-order Runge-Kutta method, sometimes called the improved Euler method. This method has a constant step size (`h`). Two other entries, `Euler+F0` and `Merson+F0` which appear in the toggle cycle are special modifications to the above methods. They allow you to process a trajectory during its computation, for example, to add random noise after each iteration. These functions will be discussed in detail in the next few lessons. The next entry, `d = 0.0000100`, indicates the absolute tolerance for the Merson solver. You can set a new tolerance value by entering a new value as previously mentioned for `X0`. Note, however, that TraX does not control the relative error.

The entry $h = 0.05$ indicates the initial integration step size. The entry `windows 1` toggles to a second group of windows but we'll leave the discussion of this entry for future lessons. The last entry, `x indicator`, is a toggle to indicate whether or not the current point of the trajectory should be marked by a cross \times (the `x indicator`) or not marked (the `- indicator` case). It should be stressed that the actual number of parameters in the Parameter window menu is greater than the number of lines which can be displayed. You can access any hidden parameter by scrolling the highlight up or down using `PgUp` or `PgDn`. Don't be surprised at the empty lines, they are reserved for high-dimensional systems.

Here we'll briefly comment on several other entries which you can see by pressing `PgDn` to scroll the Parameter window downward (see Figure 7). The entries `Hmin0.1D-06` and `Hmax10` show the minimum and maximum integration step sizes allowed. They are used by the Merson solver which will automatically adjust the step size to achieve the desired level of accuracy. The next two entries `On -1D50` and `Off 1D50` represent the lower and upper boundaries, respectively, of a time interval for which an evaluated trajectory is actually plotted. This option is useful for skipping transient or beginning values and to plot, for example, only the asymptotic solution of some model. The entries `F0 =` and so on are provided for the special processing of trajectories or for specifying rather complex equations. They aren't in use now and you shouldn't worry about them until later. Press `F2` and you will see the prompt `Current state isn't saved. Save - F3, Quit - F2`. Press `F2` to exit without saving and then press `F2` once more to exit TraX and return to DOS.

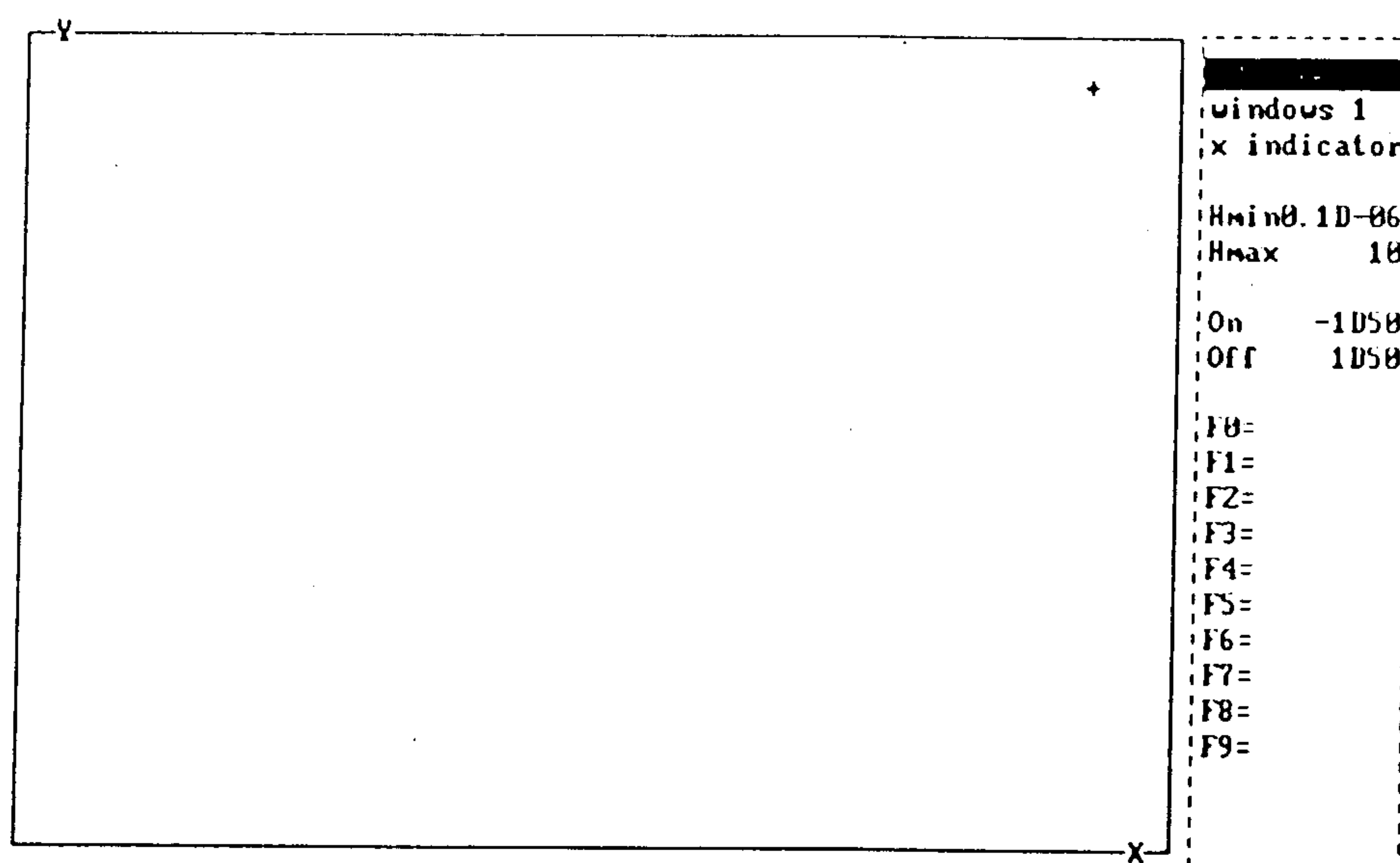


Figure 7. By scrolling downward in the Parameter window you can access hidden parameters and toggles.

Lesson 4. How to specify equations in TraX

In this lesson you'll learn how to specify new equations in TraX. To learn how this is done we'll first consider the differential equations which describe a nonlinear pendulum with damping:

$$\frac{dx}{dt} = y$$

$$\frac{dy}{dt} = -\alpha \cdot \sin(x) - b \cdot y$$

(2)

In this model, α and b are parameters. Run TraX and choose a position in the Archives as shown in Figure 2b. Now press **[Ins]** to start the specification of a new equation. You will be asked first to enter the name of this new equation. Enter an appropriate name and press **[Enter]**. You will then be asked for a sample (*i.e.*, choose an equation or a state as an example from which to copy) to be used for specification and further work with the new equations. When choosing the sample you may move through the Archives as before and select any equations or state. Let's choose the state **Initial** state which was used in Lesson 2 and which is related to the equations **Predator-prey** model (*i.e.*, move the cursor to **Predator-prey** model, press **[Right]** and then move the cursor to **Initial** state). After you have set the cursor on the example system, press **[Enter]**. The screen should now look similar to Figure 8; you are in the TraX Editor.

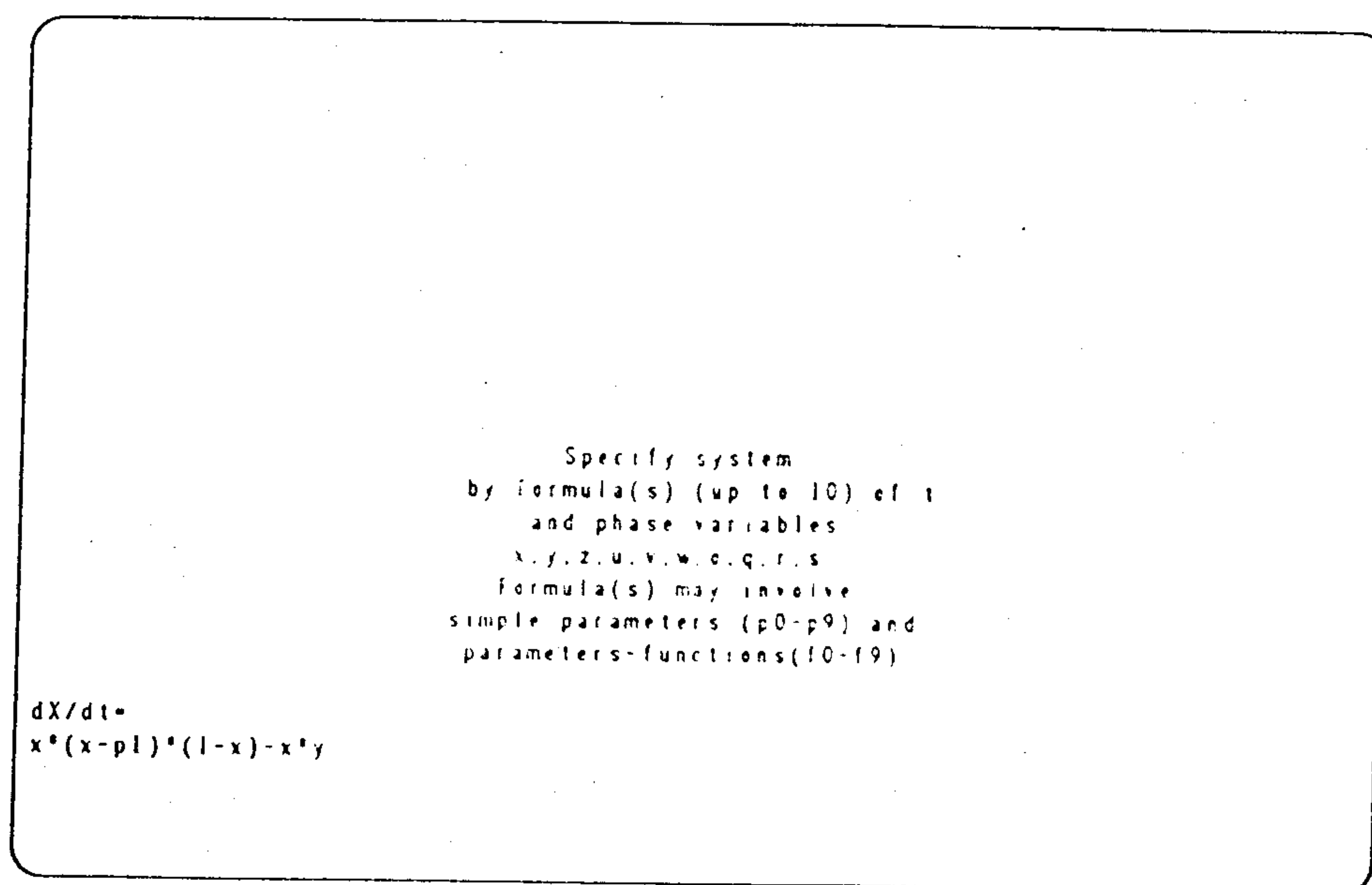


Figure 8. After entering the TraX Editor, you may start the specification of a RHS. We use equations (1) as a sample when specifying equations (2).

In simple cases, the notation of the RHS in TraX is very similar to a natural mathematical notation (compare (1) and Figure 3). The RHS of the selected example (see Figure 8) is now used as a prompt. You should modify this RHS to the RHS of (2). In particular, you can delete the RHS of the example and then specify the new RHS. When editing, the directional arrow keys as well as **Del** and **↵** may be used. Specify the RHS of (2) using x and y for phase variables and $p1$ and $p2$ for parameters. Complete each formula by pressing **↵**. When the prompt $dz/dt=$ appears (Figure 9), press **↵**. This indicates that you've finished specifying the equations. After the last **↵**, TraX goes directly into the Investigation mode. Keep in mind that when you're specifying the RHS, you can use **Shift-Esc** (before the last **↵** is pressed!) to return to the previous formulas to check or edit them.

```

Specify system
by formula(s) (up to 10) of t
and phase variables
x,y,z,u,v,w,e,q,r,s.
Formula(s) may involve
simple parameters (p0-p9) and
parameters-functions(10-19)

dX/dt=
y

dY/dt=
-p1*sin(x)-p2*y

dZ/dt=

```

Figure 9. When dz/dt appears, press **↵** indicating that you have completed the specification of the model.

You can access the Help screen now by pressing **F1**. If Help is invoked at this time, however, only information about the TraX Editor will be displayed. In the Investigation mode all the settings are as in the sample system used. Set them so that they match those in Figure 10 and simulate the model (**F10**). Try some different initial points and investigate the phase portrait. Don't forget that **F10** means simulate in the forward direction and **F9** means simulate in the reverse direction. Also, you should be reminded that after pressing a directional arrow key to move the IPI, pressing **↵** will automatically execute that command again 32 times for accelerated IPI positioning. In the Investigation mode you also can see the equations studied by pressing **Shift-F1**. When you are working with new equations it is helpful to save

the current state prior to leaving the Investigation mode, otherwise all the settings made will be lost. To finish this session, press **F3** and enter the name of the current state (for example state 1 or default state), and then press **F2** to quit.

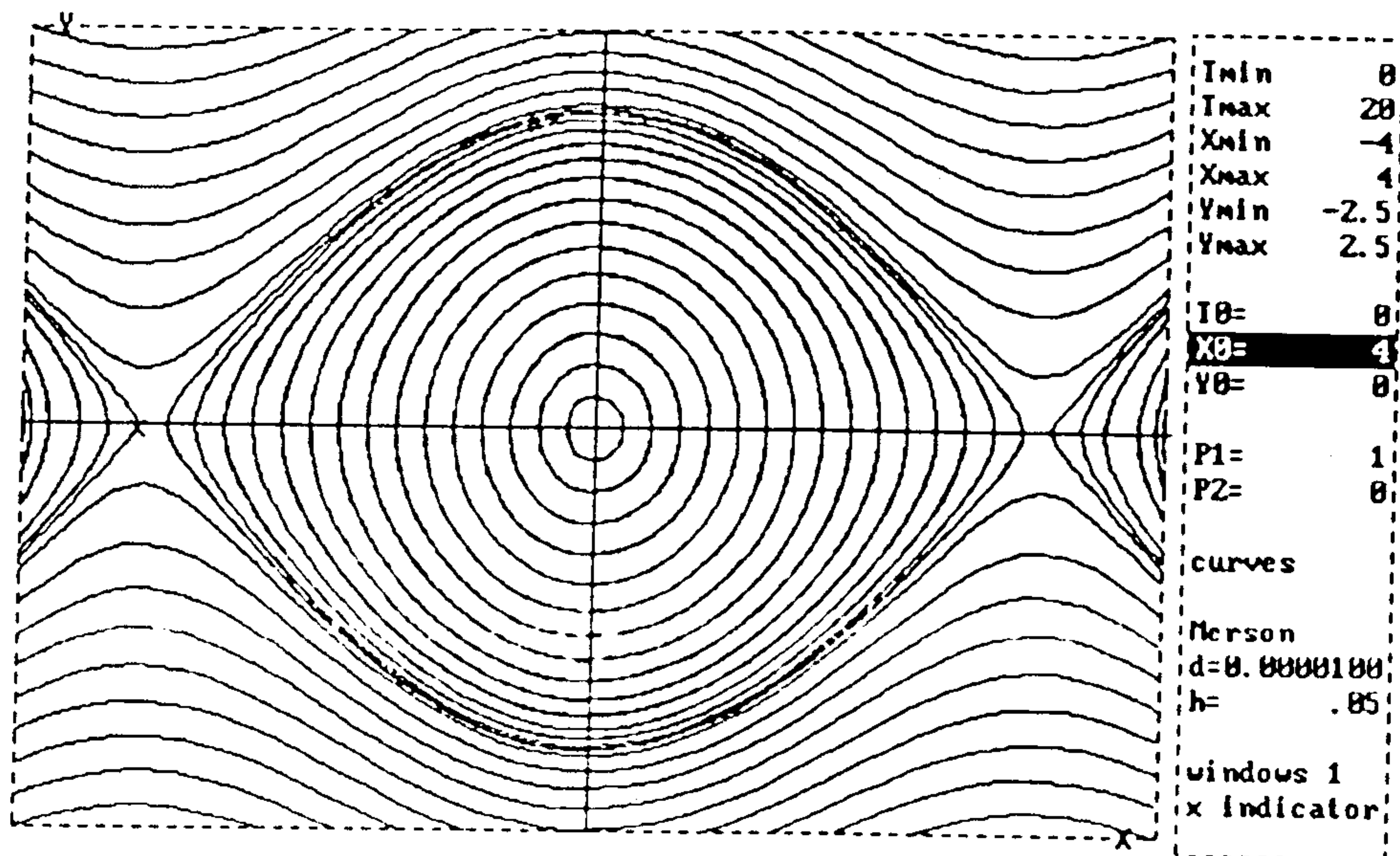


Figure 10. The phase portrait of the nonlinear pendulum equations (2) using $\alpha = 1$ (P1) and $b = 0$ (P0); this is the conservative case, *i.e.*, no damping.

Lesson 5. Interacting with Graphic windows in TraX

In the previous lessons only one graphic window was used for plotting trajectories, and you did not have to worry about specifying this window. In this lesson, however, you will learn to deal with several graphic windows, and in particular, how to specify new windows and position them on the screen.

We'll begin where we left off in Lesson 4. Start TraX and enter the Archives. When the Archives is displayed, select **Differential equations**, press \rightarrow , select the equation **Nonlinear pendulum with damping**, press \rightarrow and then position the cursor on the name of the stored state for equation (2) or select the state **Initial state (phase plane)** and press \rightarrow .

Press \leftarrow and you will enter the Investigation mode. Set the entries **X0** and **Y0** as shown in Figure 10 (use the **PgUp** or **PgDn** to move the highlight and then type in new values and press \leftarrow). Next, highlight the **Parameter window entry windows 1**. This entry specifies an active group of graphic windows. There are two groups of graphic windows in TraX: the first group is visible now and it contains one window labelled **x-y**. To display the second group, press \leftarrow to toggle the entry to **windows 2**. Two graphic windows will appear; the upper window is labelled **t-x** and the lower is labelled **t-y**. Press **F10** to plot the evolution of the time series of the two state variables, **x** and **y** (see Figure 11). Continue the simulation for another time interval by pressing **F10** again. Notice that the graphic windows which have a **t** axis are automatically cleared before the next part of the solution curve is plotted.

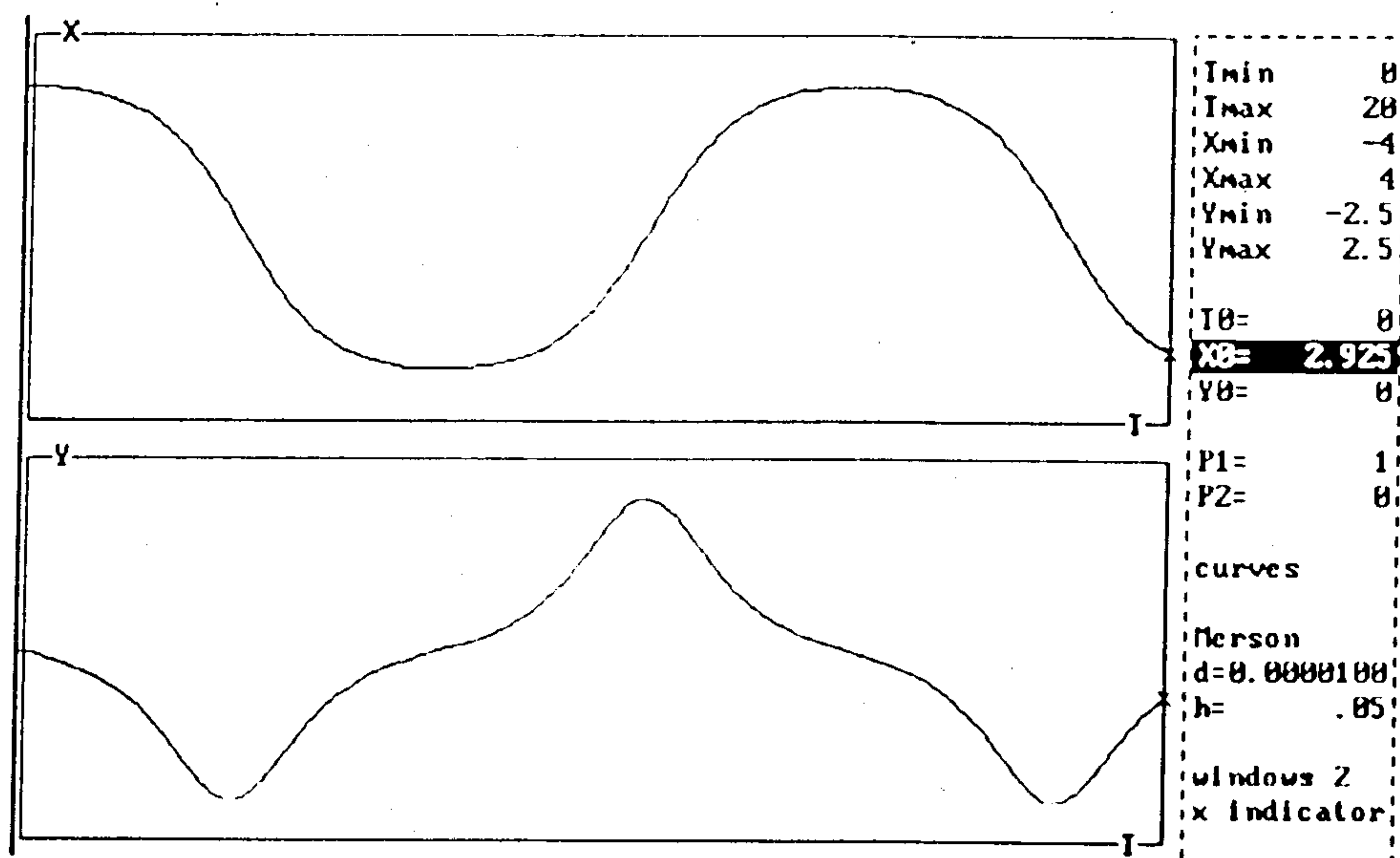


Figure 11. The time series graphs of variables *x* (top) and *y* (bottom) for equations (2). The windows 2 group is used.

Now we'll plot the phase trajectory and the time series together on the same screen (Figure 12). For this, a new window labelled x-y must be added to the existing set of windows. Since there is no free space on the screen for a new window we'll need to resize the existing windows and then reposition them, for example, near the bottom of the screen. Using TraX we can create overlapping windows, but we won't do this here. Press **(F5)** and the frame of the t-y window becomes dotted. This means this window is the active window. Only when a window is active can you move it, resize it, relabel its axes or delete it entirely. Press **(Shift-F7)** to resize or move the active window, which should be the t-y window. The directional arrow keys (**(←)**, **(→)**, **(↑)** or **(↓)**) are used to move the active window and the keys **(Shift-Left)**, **(Shift-Right)**, **(Shift-Up)** and **(Shift-Down)** are used to change the active window size (note that window scaling will not be affected by changing the window size). Experiment with the many possible variations and you will see just how flexible the TraX interface is. After you have finished operating on a window, press **(↵)** to fix it in position.

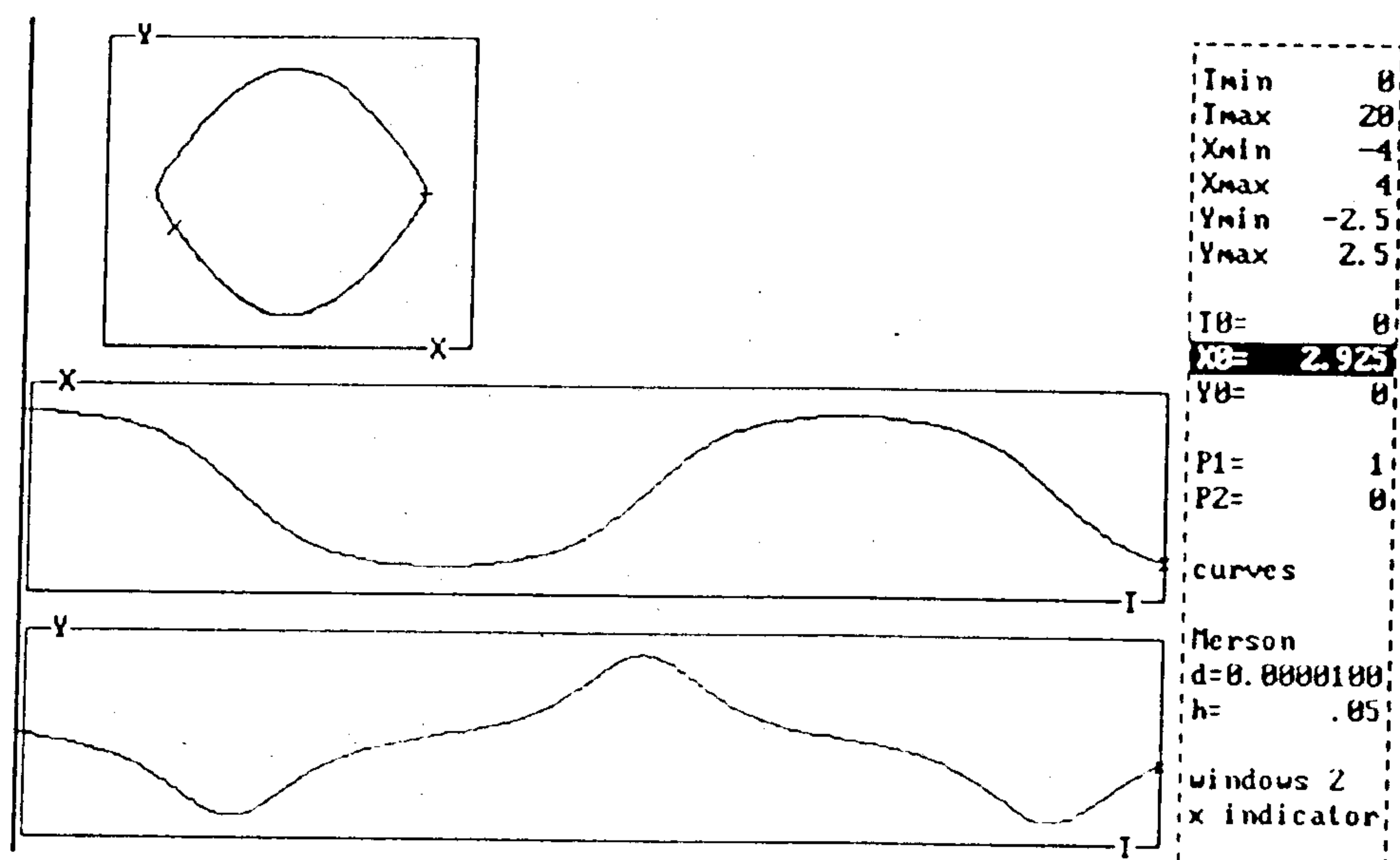


Figure 12. The graphical window x-y for plotting the phase portrait and the windows t-x, t-y to show x and y as a function of t located on the same screen. The windows 2 group is used.

Now, select the window t-y (**(F5)**), edit it (**(Shift-F7)**) and decrease its vertical size, and then move it to the lowest position on the screen like that shown in Figure 12. Complete this operation by pressing **(↵)**. Next, resize and move the t-x window in a similar fashion. To access this window, press **(F5)**; you probably have noticed by now that this key cycles through all the windows making the next window in the list active with each keypress. The final position and size of the window t-x should appear similar to that shown in Figure 12.

Now we'll specify the new x-y window. Press **Ins** and a small window labelled **t-x** will appear in the upper left corner of the screen. Here **t** and **x** are the default names for the abscissa and the ordinate. You should first increase the size of this window and then relabel the axes. Do this now: first resize the window (**Shift-F7**) and then relabel the axes; press **Ctrl-F7** to rename the ordinate or y-axis (rename it **Y**) and press **Ctrl-F8** to rename the abscissa or x-axis (rename it **X**). Your screen should now look like that shown in Figure 12.

To preserve the results of your work on the window's design, save the current state (**F3**). Now you can plot a trajectory in the windows you have designed. Plot the axes (**F8**) and initiate the simulation (**F10**). At the end of the time interval the screen should appear similar to that shown in Figure 12. Continue the simulation by pressing **F10**. Be sure that only the windows having a **t** axis have been cleared before plotting the next part of the solution curve.

Compute some other trajectories using different parameter values. You will see some advantages and shortcomings of using two kinds of plots at one time. You may want to clear only one window and to do this, make the window active (**F5**) and then clear it by pressing **Alt-F7**. You should recall that pressing **F7** clears all of the graphic windows at one time. Now you've learned the basics of the TraX graphic window interface.

Lesson 6. More about Graphic window functions

In this lesson you'll learn to use windows with nonstandard axis labels (*i.e.*, those which are not t , x , or y , etc.). Nonstandard axes will need to be scaled individually and we'll discuss this problem as well. This lesson is an extension of Lesson 5 and we're proceeding with the investigation of the equations (2) which you worked with in the previous lesson. Start from the position shown in Figure 13 (*i.e.*, enter the Archives, select Differential equations, press \rightarrow , select Nonlinear pendulum with damping, press \rightarrow , select the state you saved in the previous lesson or the state Three windows..., press \leftarrow , wait for the Investigation mode to set up and then press F10 to begin the simulation).

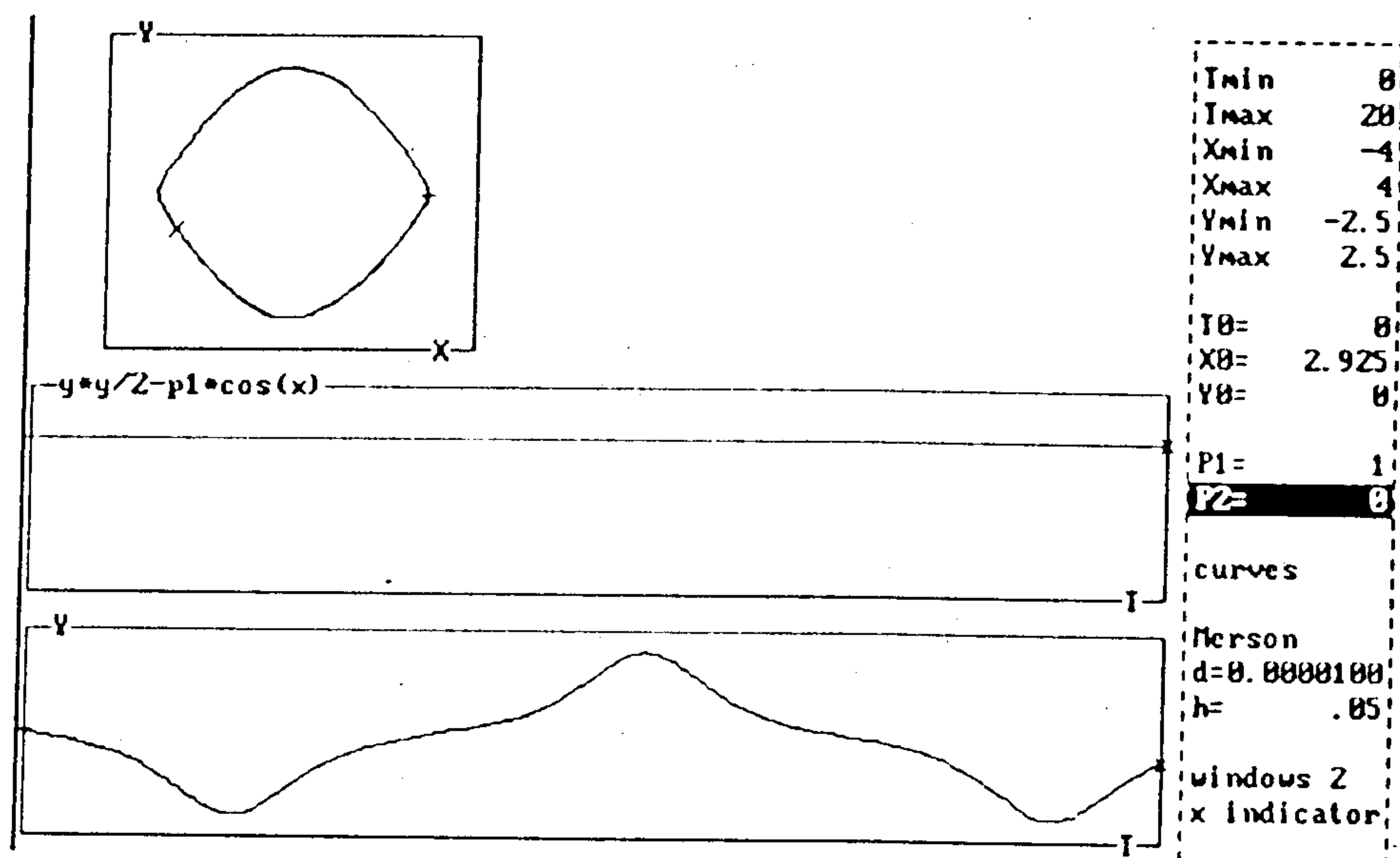


Figure 13a. In the center window, the energy function $y^2/2 - a \cdot \cos(x)$ as a function of t for the solution of equations (2) is plotted: the conservative case of no damping is shown ($b = 0$).

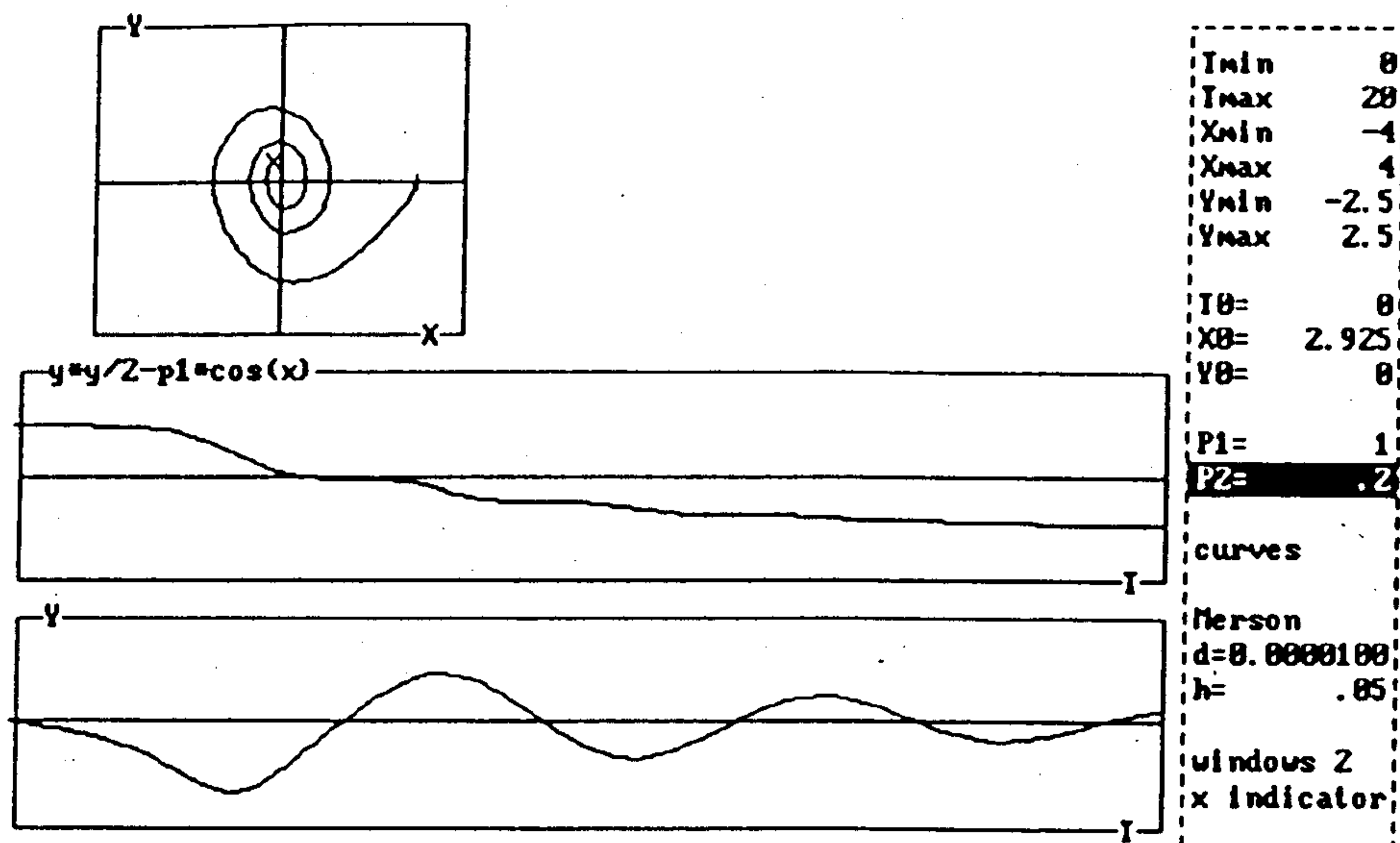


Figure 13b. In the center window, the energy function $y^2/2 - a \cdot \cos(x)$ as a function of t for the solution of equations (2) is plotted: the non-conservative case of damping is shown ($b = 0.2$).

6.1. Window limits

First, examine the entries Tmin, Tmax, Xmin, and so on, in the Parameter window and the limits of the graphic windows as well. To do this, press **(F5)** several times in order to make all the windows, including the Parameter window, active in sequence (one at a time). When the Parameter window is active, you will see the limits for all the variables. When a graphic window becomes active the parameter window is automatically updated and shows only the limits of that active window. You'll recall that all of the settings in the Parameter window can be changed. Thus, by activating the Parameter window you can change some or all of the common or global limits, and by activating another graphic window you get access to its individual or local limits. Play with the limits of these windows but pay particular attention to changes made in the value of a common limit because this will automatically change the limits in all the windows which have the same axis label. The opposite, however, is not true; changing a window's individual limits has no effect on the limits of any other window. One other essential feature of TraX is that no automatic window clearing is performed after rescaling. This is left for you to decide.

6.2. Using expressions to label axes

Now we'll study the energy function:

$$E = \frac{y^2}{2} - a \cdot \cos(x) \quad (3)$$

on the solutions of equations (2). This function is constant when there is no damping ($b = 0$), and it decreases monotonically when damping exists ($b > 0$). Let's use the window t - x to plot E against t . For this you must first rename the window's ordinate. Activate the t - x window (press **F5**) key until the frame of this window becomes dotted) and then press **Ctrl-F7** to rename the ordinate. Type the expression $(y*y)/2-p1*cos(x)$ and press **Enter**; this is the new ordinate label, as well as the rule for evaluating the value of the ordinate. Notice that the new entries OMIN and OMAX appear in the Parameter window. These are the limits of the new ordinate in the active window. Set their values to OMIN=-2 and OMAX=2 by highlighting them and typing in the new values.

Now we'll simulate the model and plot the evolution of the energy function over time. Set the parameter values $P1=1$ and $P2=0$, and the initial conditions $X0=2.925$, and $Y0=0$. Draw axes (**F8**) and then begin the simulation (**F10**). The resulting plot (see Figure 13a) illustrates that in the conservative case the energy function is really constant along the trajectories. Try some additional initial points to observe the relationship between the phase trajectory, the time evolution of x and y , and the energy function. After that you can study the effect of damping by, for example, setting $P2=0.2$ (see Figure 13b). When damping is added the phase trajectory will spiral toward the origin, and $y(t)$, $E(t)$, and $-a$ will decay to zero (oscillating in the first case and decaying monotonically in the second).

Now we'll discuss how the initial point is set. You'll recall that in Lesson 2 we mentioned that the initial point can be set by either entering initial conditions or by moving the IPI directly on the phase plane. If we have only one graphic window there is no problem with moving the IPI. But now we have three graphic windows, and therefore to move the IPI you must activate the window first. There are two occasions, however, when you won't be able to move the IPI along the abscissa or the ordinate in an active window: (1) if the corresponding axis name is not time (t) or a phase variable (x, y, \dots), and (2) if you use the directional arrow keys when the Parameter window is active, this will change the x and y variables, an indirect way of moving the IPI which we used in Lesson 2. For the problem in this lesson it is convenient to activate the window x - y and then set the initial point by moving the IPI.

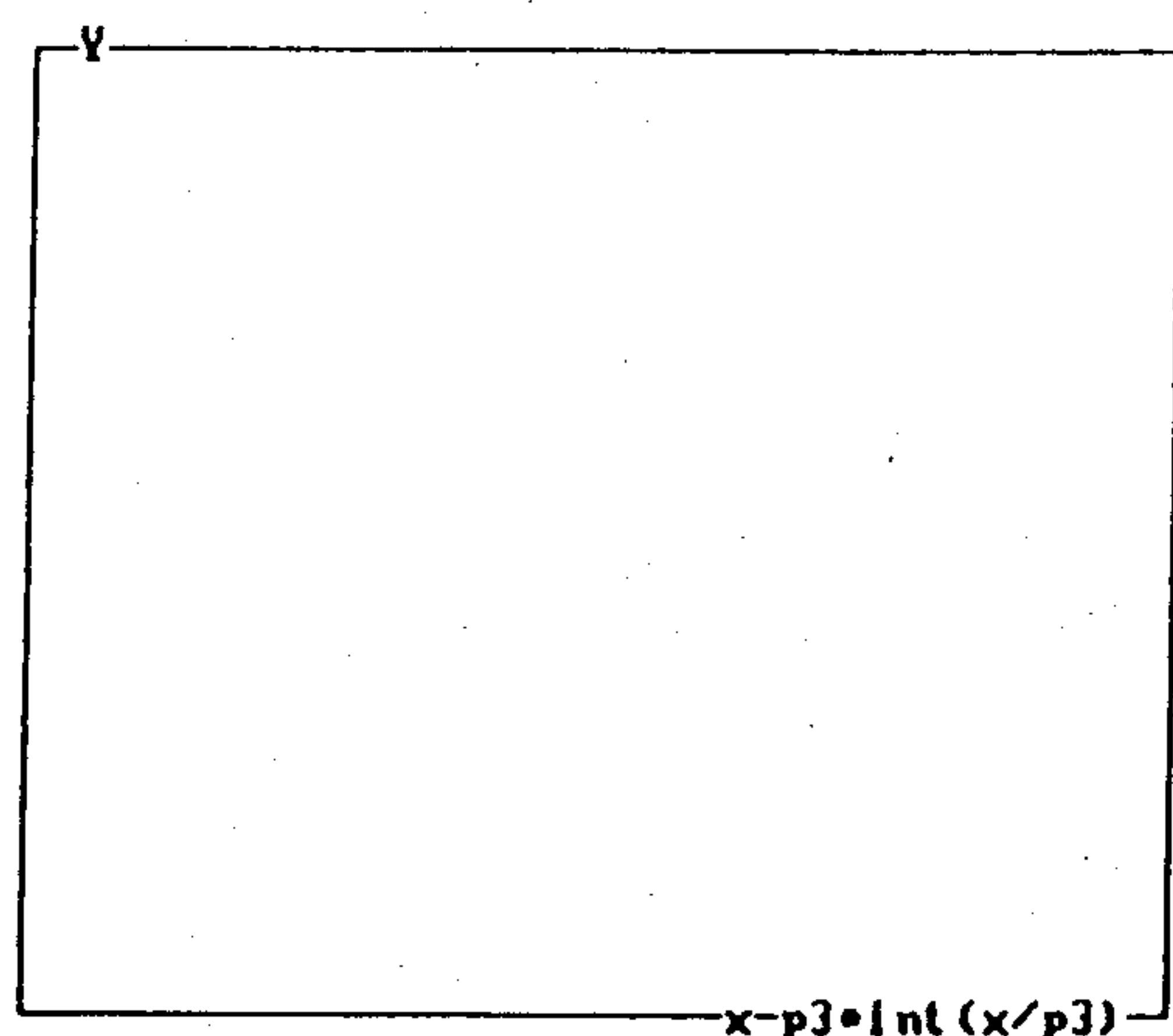
6.3. Cylindric phase space

Now we want to draw your attention to the fact that a natural phase space for equation (2) is a cylinder, not a plane. Although there is no special option of this kind in

TraX, we can plot the phase portrait on a cylindrical surface by redefining an axis, *i.e.*, use a circle variable to label one axis. We will use a 2π -periodic function of x instead of x in the x - y window. Let's try to implement this idea.

Make the window x - y active (**F5**). Press **Ctrl-F8** to rename the abscissa and then enter the expression $x-p3*\text{int}(x/p3)$; this is the label of the new abscissa (here **int** denotes the a function which return the integer part of a real number) and press **↵**. This expression defines a circle variable on $[0, p3)$, where $p3$ is a period to be specified by you. In this case $p3$ should be equal to 2π . Since the new parameter $p3$ has been used in the expression, the additional entry $P3=...$ will appear in the Parameter window (you can reorganize the position of this entry by highlighting it and using **Ctrl-PgUp** or **Ctrl-PgDn** to move (*i.e.*, drag) the entry up or down, respectively). Set $P3=6.28$, but if you want to set the value with higher precision, first highlight the entry $P3=...$ and press **=**. The prompt **Compute:** will appear on the top of the screen in the Information Line. This means that you have to either specify an expression or want to compute the value of an expression and assign this value to the highlighted entry, *e.g.*, $P3$. Type the expression $8*\text{atn}(1)$ (see Figure 14) and press **↵** (function **atn** stands for arctangent). The desired value will now be assigned to $P3$ (note that $8 \cdot \arctan(1)$ gives a convenient numerical approximation of 2π). Before beginning the simulation, set the limits for the modified window. The appropriate values are: $AMIN=0$, $AMAX=6.28$, $Ymin=-2.5$ and $Ymax=2.5$. Here $AMIN$ and $AMAX$ are the abscissa's limits. Since $AMAX$ is set to 2π , you can use the compute feature (*i.e.*, **=**) to enter the value for $AMAX$.

Compute:
8*atn(1)



Ymax	2.5
T0=	0
X0=	2.925
Y0=	0
P1=	1
P2=	0
P3=	0
points	
Merson	
d=0.0000100	
h=	.05
Windows	2
x indicator	

Figure 14. The **Compute:** option is used to assign the value of 2π to parameter $P3$ with higher precision.

It will be convenient for the following experiments to have only one graphic window on the screen to represent the surface of a cylinder. To delete each of other two graphic windows containing the t-axis, make each window active in turn, press **[Del]** and answer Y to the prompt to delete the active window. After deleting both of the windows, increase the size of the remaining window (use **[Shift-F7]**).

Experiment with changing the x-coordinate of the initial point (you'll have to type these new coordinates in). Note that after the IPI crosses the window's right border it appears on the left, and vice versa. This is because of the circle variable used on the abscissa. Simulate the model with $P1=1$ and $P2=0$. Use the **points** option instead of **curves**. Try the **curves** option too, and you will see some peculiar pictures: do you know why they appear this way? After some simulations you should obtain the phase portrait on a cylinder like that shown in Figure 15a.

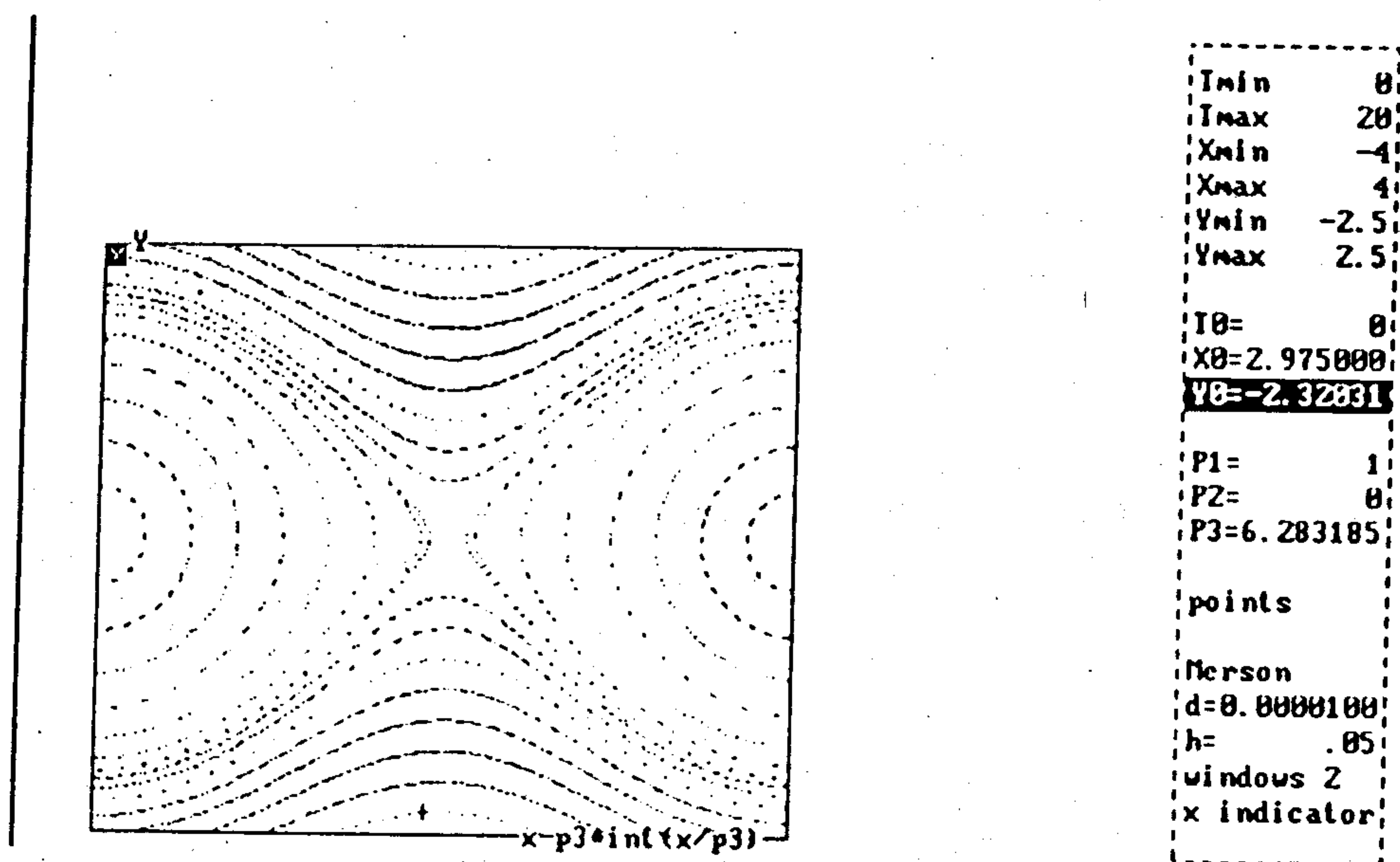
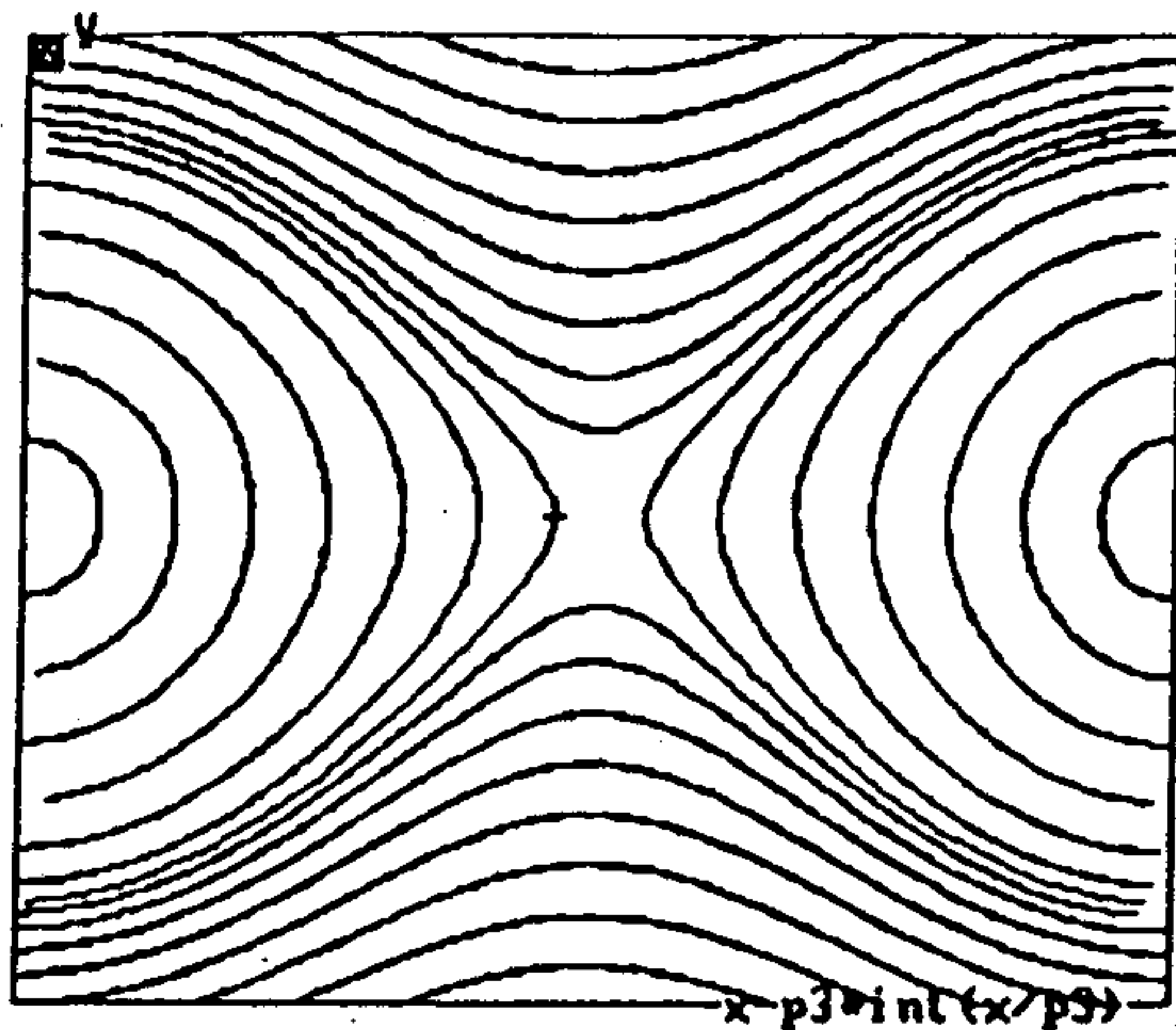


Figure 15a. Phase portrait of (2) on a cylinder plotted by points.

This discussion illustrates how the **points** option is useful for ODEs. Actually, the trajectories on the cylinder may be plotted by curves too, as shown in Figure 15b. You can derive this phase portrait yourself by using the state **Phase portrait on a cylinder plotted by curves**. There are additional options concerning the processing of the trajectory prior to plotting which are also useful and these will be discussed in Lesson 11. Before leaving this lesson, save the last state (**[F3]**).



```

Tmin      0
Tmax      20
Xmin      -4
Xmax       4
Ymin     -2.5
Ymax       2.5

T0=        0
X0=       2.9
Y0=        0

P1=        1
P2=        0
P3=6.283185

curves
Nelson+F0
F0=(glo 10,
windows 2

```

Figure 15b. Phase portrait of (2) on a cylinder plotted by curves.

Lesson 7. The use of colors in TraX

If your computer is equipped with an EGA or VGA adapter and a color monitor, you can use color graphics with TraX. In this short lesson you will learn to manipulate the colors of windows and trajectories.

Let's start with the state 3 windows of the equations Predator-prey model. There are now three graphic windows on the screen and you can change the background color of each window, as well as the color for plotting the trajectories. First, compute several trajectories with different initial conditions and second, activate a graphic window. To change the background color in the active window, press **Ctrl-F6**. Repeat this several times to see all the 16 colors available. Make sure that after a change in the background color, the active window is cleared automatically. To change the trajectory's color in the active window, press **Ctrl-F5**. The window's frame and the IPI will change their colors. The new color will be used for all future simulations (*i.e.*, for the next trajectories) until you change it.

Changing the color of a trajectory, however, does not erase the trajectory nor does it affect the colors of any existing trajectories. This provides a convenient way to outline special or unique trajectories. For example, in the phase portrait of Figure 6, the trajectories corresponding to surviving and extinct ecosystems (*i.e.*, belonging to the basins of two stable steady-states) can be plotted using different colors.

Sometimes you may need mark a trajectory by a specific color, for instance a separatrix, in all the windows at one time. To do this, set the color desired in the active window and begin the simulation by pressing **Alt-F10**. The same color will be used for plotting the trajectory in all the windows.

You can also change the foreground and background and colors in the Parameter window. The same keys as above should be used for this but remember, colors can only be changed in an active window.

Lesson 8. How to simulate difference equations

So far we have only studied ODEs. This lesson is intended to show you how to deal with difference equations, sometimes called iterated maps. Select the Archives entry **Difference equations** and then select the equations **Logistic map** $x' = p1 \cdot x \cdot (1 - x)$. This refers the map

$$x' = a \cdot x \cdot (1 - x) \quad (4)$$

which arises, in particular, in modeling population dynamics (see May, 1976). For the sake of convenience, the name of the entry is the related formula. The real formula for the map is specified in the TraX Editor in the usual manner. You may view the formula pressing **[Shift-F1]**. Select the entry **Initial state** for Lesson 8 in the list of stored states for the chosen map. After entering the Investigation mode you will see one graphic window **n-x**; this will serve as the window to plot the time evolution of x (n is discrete time). The corresponding limits **Nmin**, **Nmax**, **Xmin**, and **Xmax** can be observed in the Parameter window.

Press **[F10]** to begin the simulation with $P1=2.8$. You will see a trajectory which approaches an equilibrium state (Figure 16a). It is plotted by lines since the option **curves** is used (see the Parameter window) but you can plot it using the **points** option as well. To continue simulating the trajectory, press **[F10]** again.

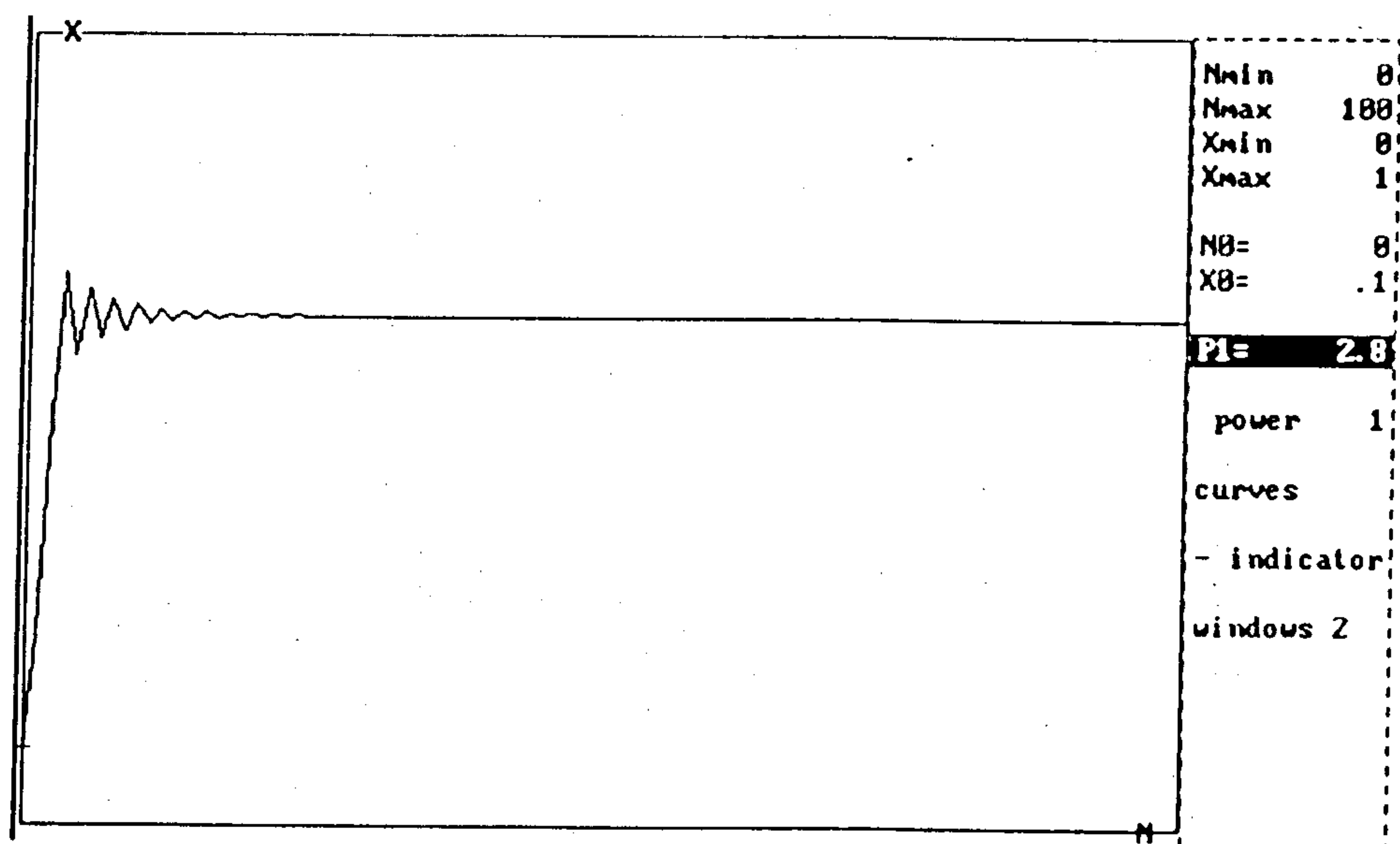


Figure 16a. Simulation of the logistic map (4) for $\alpha = 2.8$ ($P1$); the trajectory stabilizes to an equilibrium point,

You'll recall that **F9** plots a simulation in reverse time for ODEs; this option, however, has no effect for difference equations. The reason for this is that an inverse map should be iterated for reverse time, but usually it is defined implicitly or undefined at all, as in (4).

It is known that if the value of the parameter α in (4) is gradually increased more complicated dynamics appear: these include periodic and chaotic orbits. Let's illustrate this phenomena using TraX and in doing this you'll learn some more useful TraX options.

If the parameter α crosses the critical value $\alpha = 3$, a periodic orbit of period 2 appears. To observe this effect, set $P1=3.2$ and plot a trajectory. The result (see Figure 16b) indicates that after some transients the trajectory stabilizes to a cycle of period 2. Next, plot the trajectory for $P1=3.5$ (Figure 16c). You can see that it is also asymptotically periodic, but it has a period of 4. To see the chaotic behavior, set $P1=4$ and begin the simulation. You will get an irregular or aperiodic trajectory similar to that shown in Figure 16d. If you continue the simulation it will remain irregular.

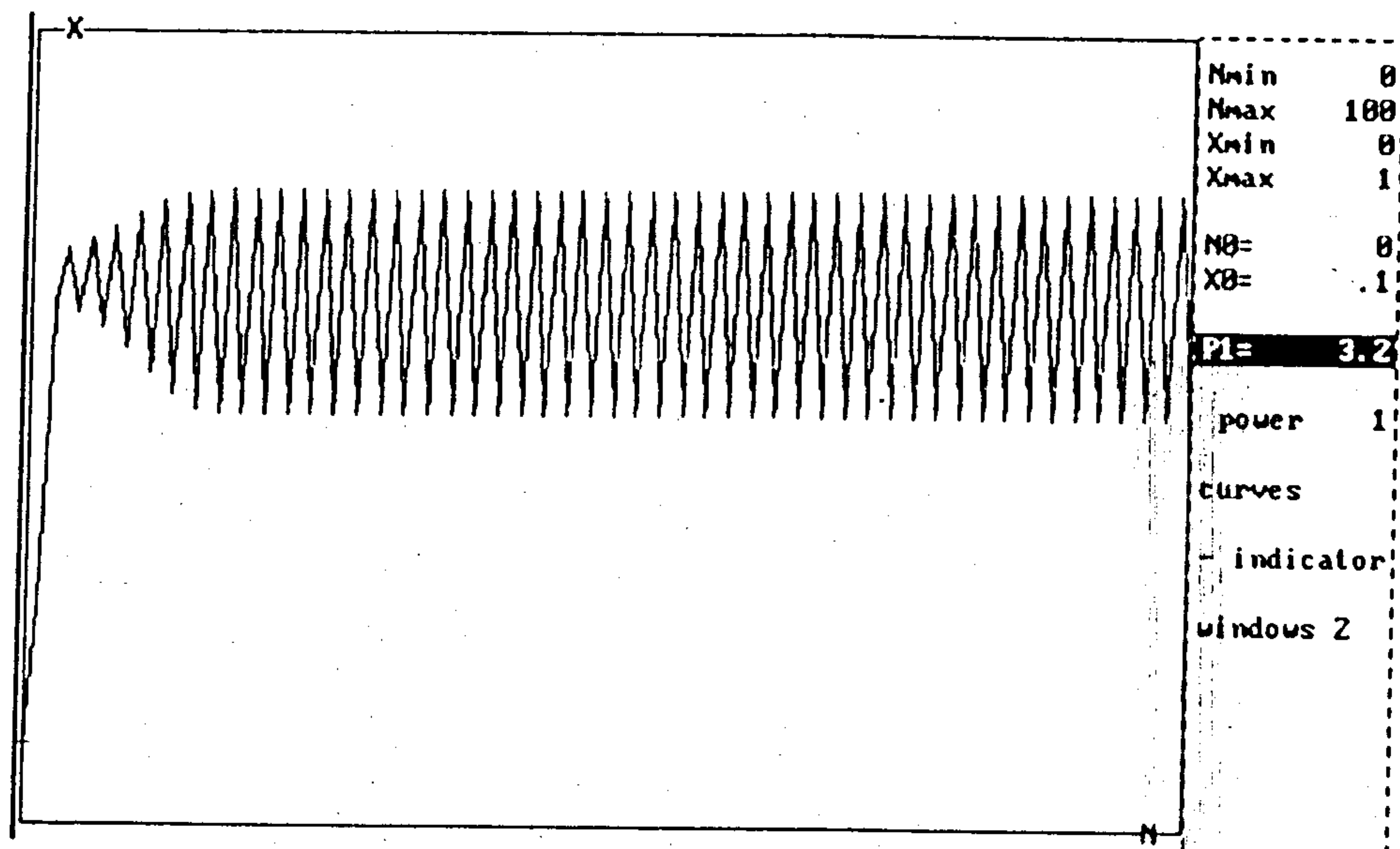


Figure 16b. Simulation of the logistic map (4) for $\alpha = 3.2$ ($P1$); the trajectory illustrates period-2 oscillations.

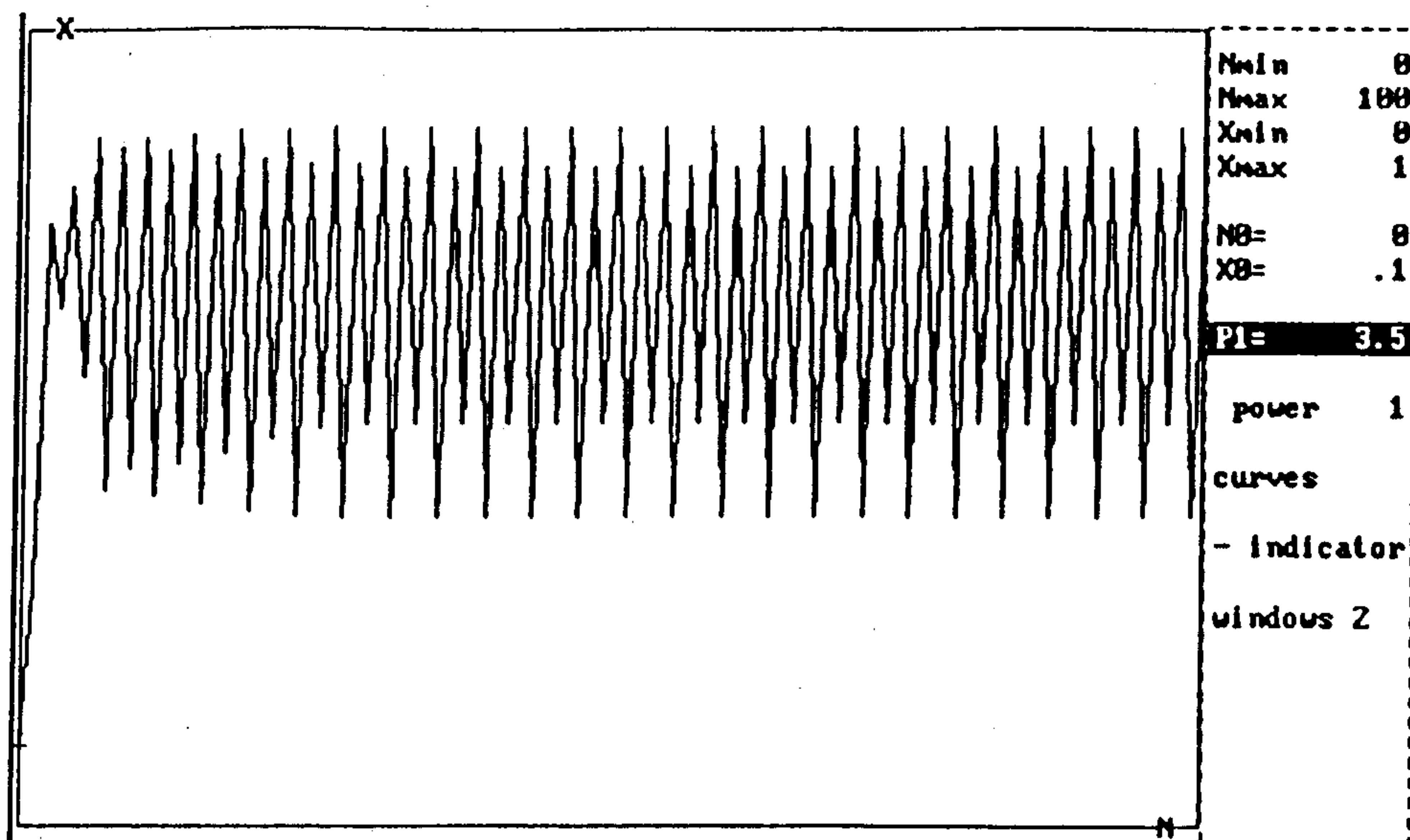


Figure 16c. Simulation of the logistic map (4) for $\alpha = 3.5$ (P1); after the second period-doubling bifurcation, an orbit of period 4 arises;

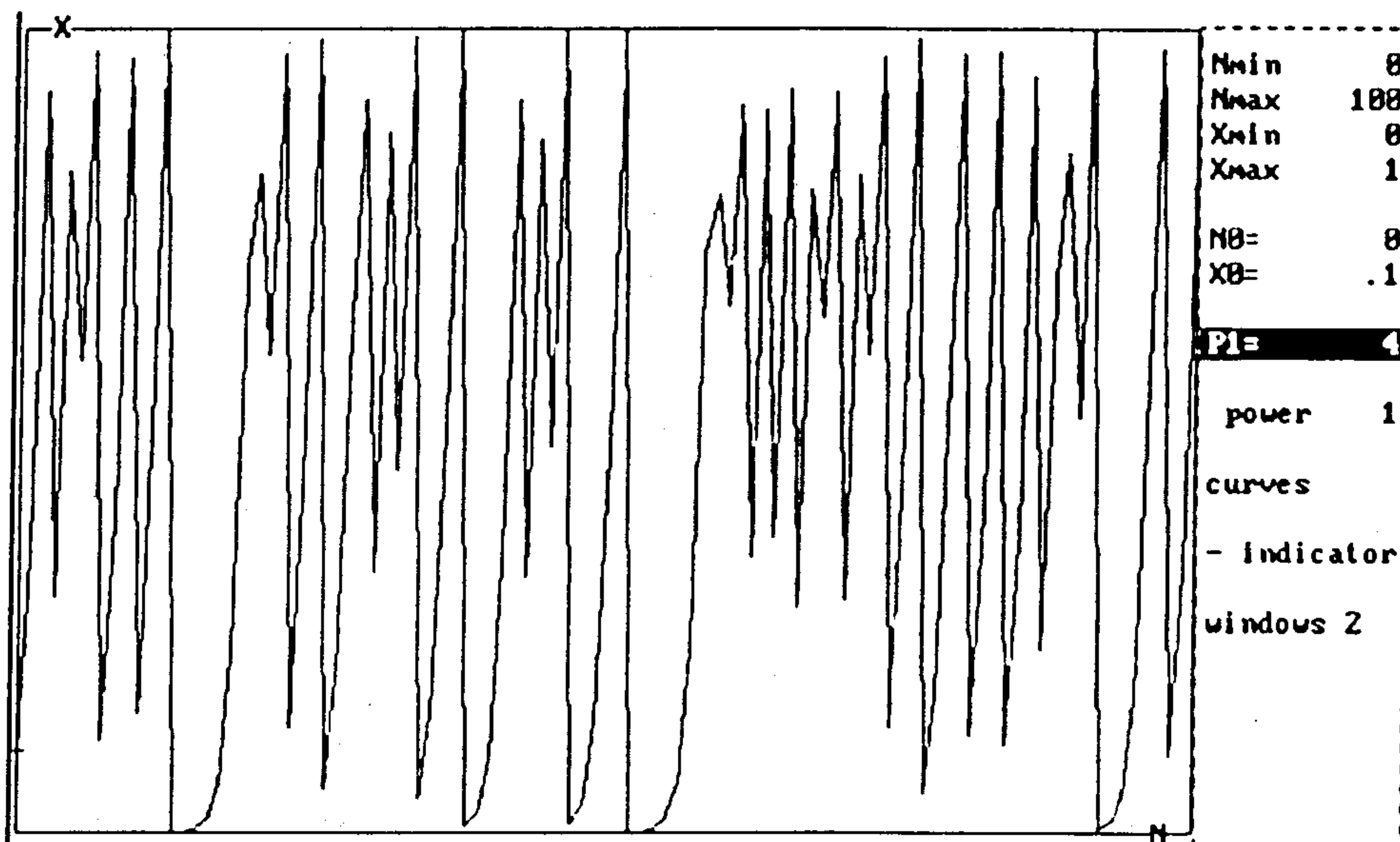


Figure 16d. Simulations of the logistic map (4) for $\alpha = 4.0$ (P1); irregular behavior (chaos) now appears.

Now we'll learn a new option which is provided only for difference equations. Highlight the Parameter window entry $power = 1$, which indicates the power of the map to be iterated. Set the value 2 for the power and repeat the above experiments. You should see plots like that shown in Figure 17. Thus, for the second power of the map (4), the equilibrium state remains the same, the period-2 oscillations turn into an

equilibrium state, the period-4 oscillations become period-2 oscillations, and the chaotic trajectory preserves its irregular character. You may also try a power 4 map to transform the 4-cycle into fixed point. This is a simple way to search for a stable periodic orbit with an unknown period. Now, let's change the window's group (highlight the entry windows 2 and press \leftarrow). The graphic window x-x will appear. This window's abscissa and ordinate have the same label(x) and the same limits. This is a special window for plotting the "1-dimensional staircase" which is a useful tool in studying first-order difference equations or 1-dimensional maps. Plotting a line in this window is similar to plotting a trajectory in a phase plane.

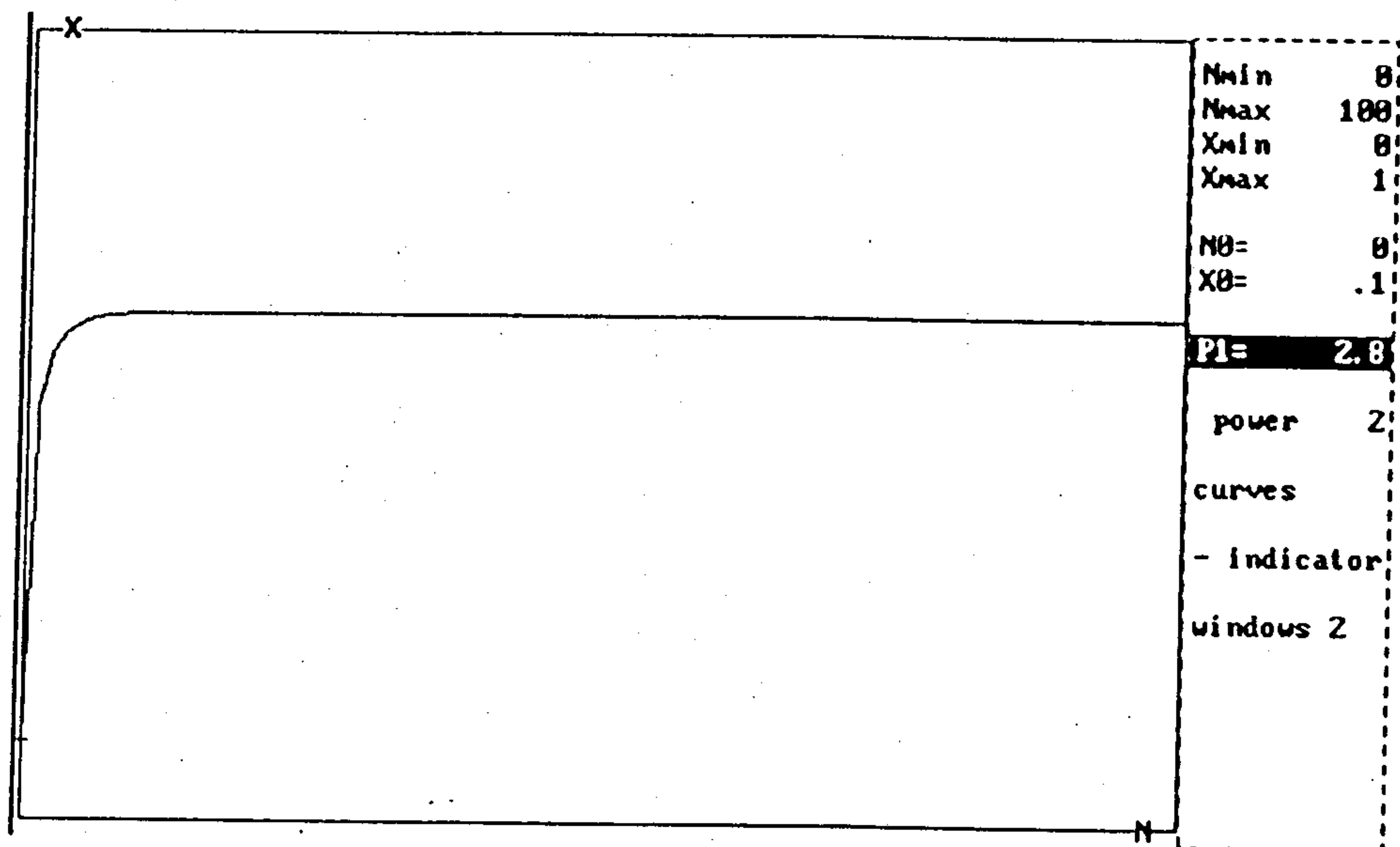


Figure 17a. The same simulation as in Figure 16a but using degree 2 (power 2) of the map.

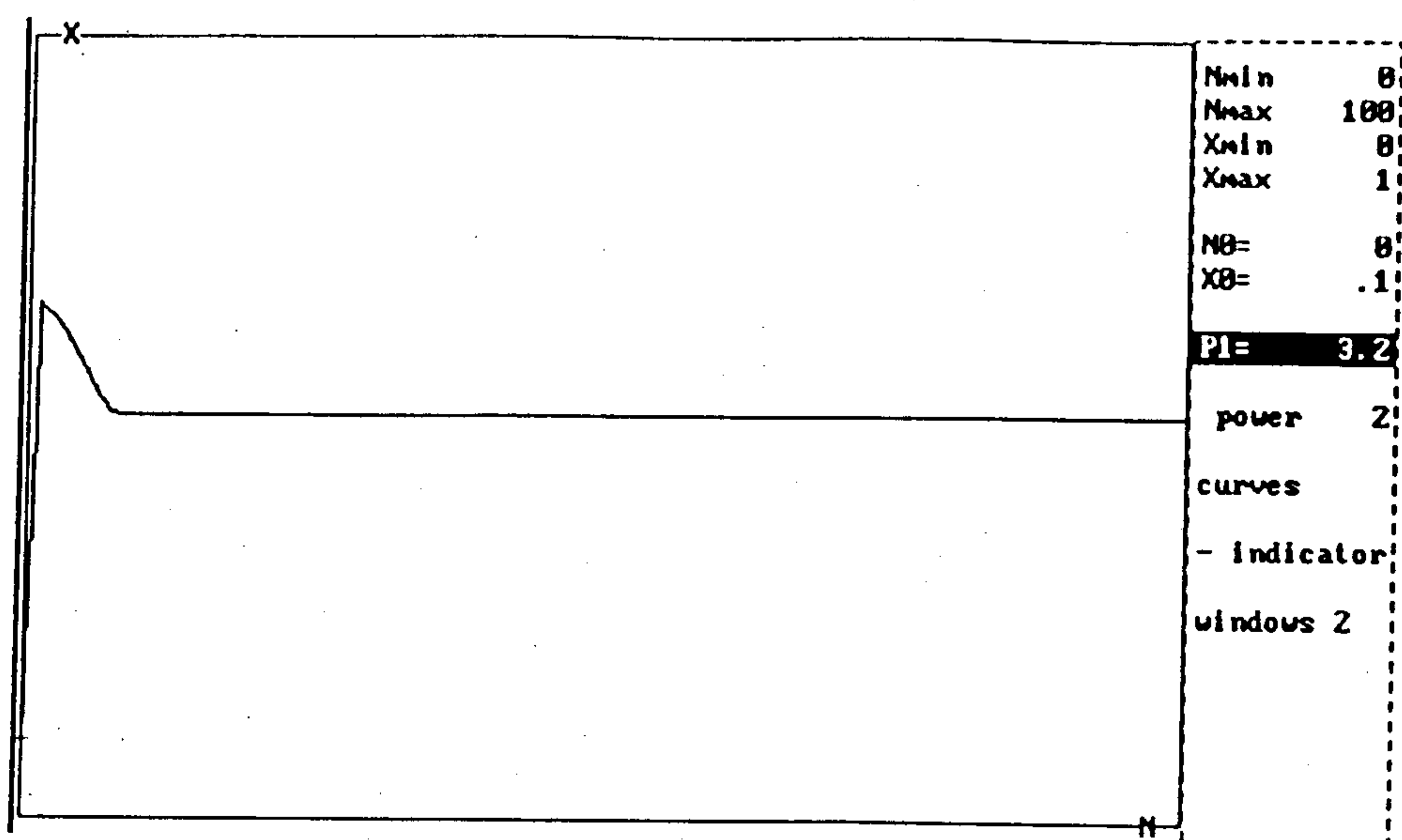


Figure 17b. The same simulation as in Figure 16b but using degree 2 (power 2) of the map.

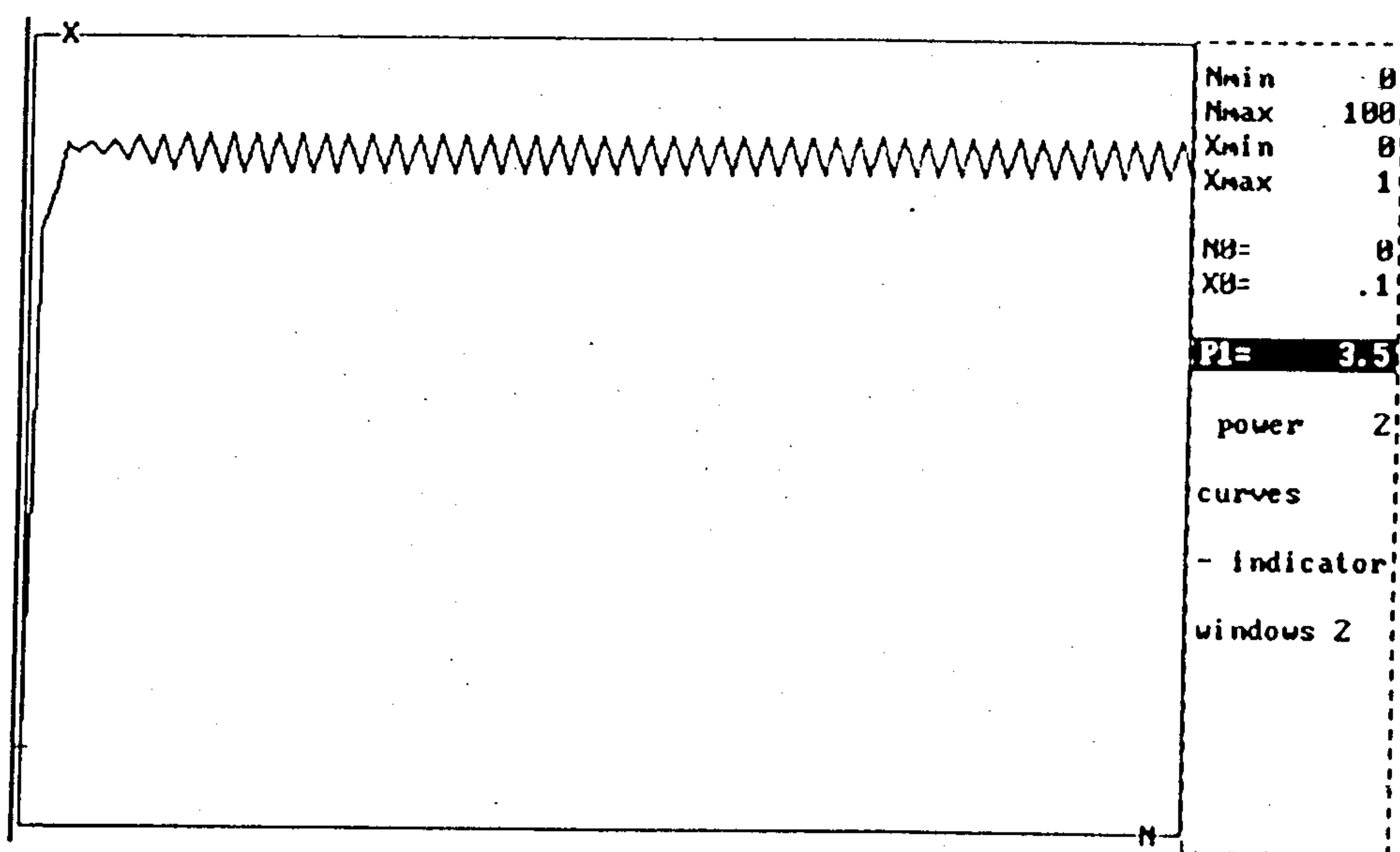


Figure 17c. The same simulation as in Figure 16c but using degree 2 (power 2) of the map.

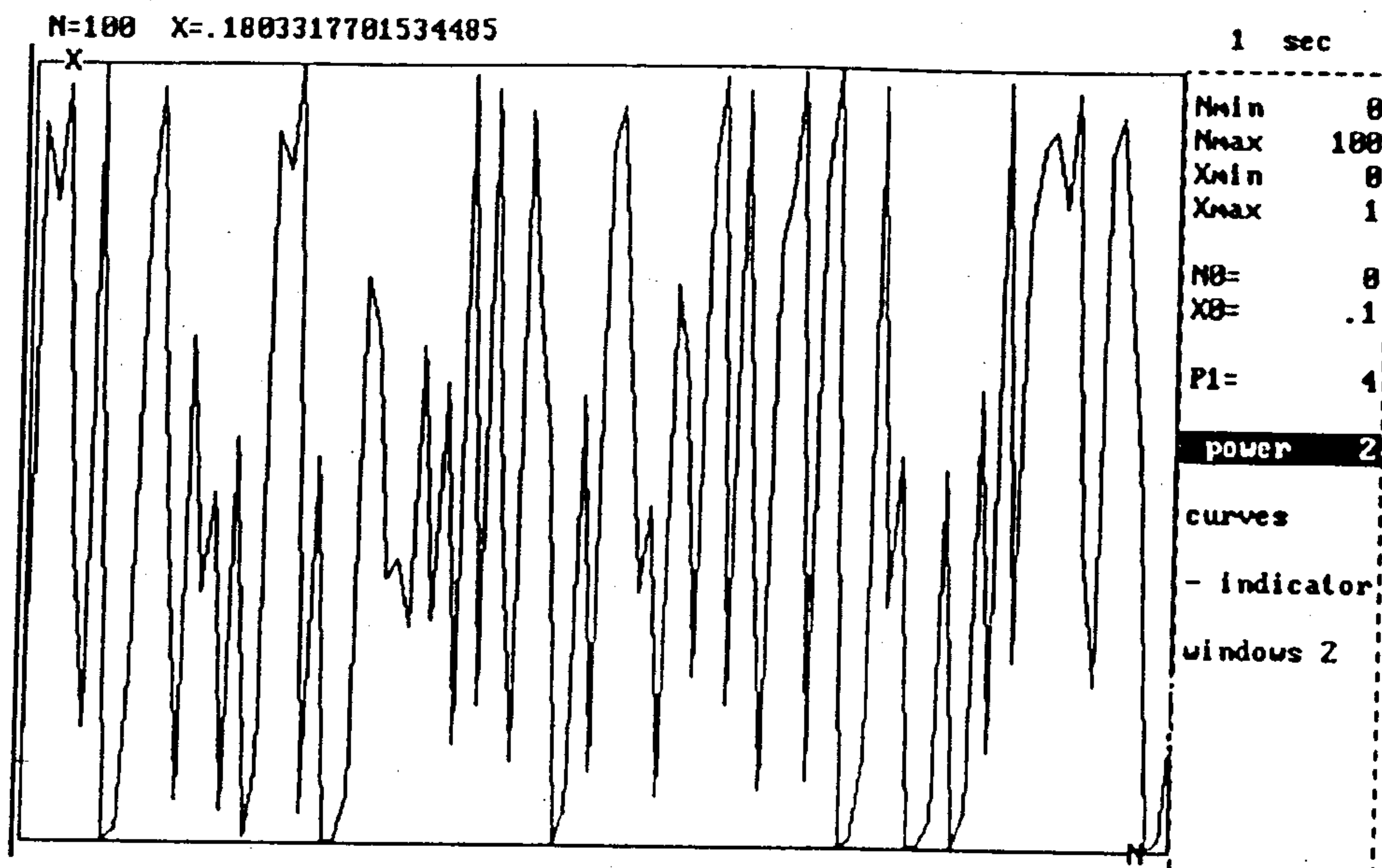


Figure 17d. The same simulation as in Figure 16d but using degree 2 (power 2) of the map.

Repeat the previous experiments using this special window. First, set $P1=2.8$ and press **F8** (the curves option must be toggled on prior to this!). Note that the axes, the bisector line, and the graph of the function defining the right-hand-side of (4) will be plotted. The points where the function graph intersects the bisector line correspond to the fixed points of the map (*i.e.*, the equilibrium states). Begin the simulation and you will see a trajectory which tends to the fixed point (*see* Figure 18a). Repeat this simulation with $P1=3.2$, 3.5 , and 4.0 . The corresponding plots are shown in Figure 18b-d, respectively. In Figure 18b and 18c the asymptotic modes are the 2-cycle and 4-cycle closed curves. If you want to extract these cycles from the transients, press **Shift-F10** after the trajectory approaches the cycle. The initial point will then automatically be reset to the trajectory's most recent point. Following this, clear the window and start the simulation from the new initial point.

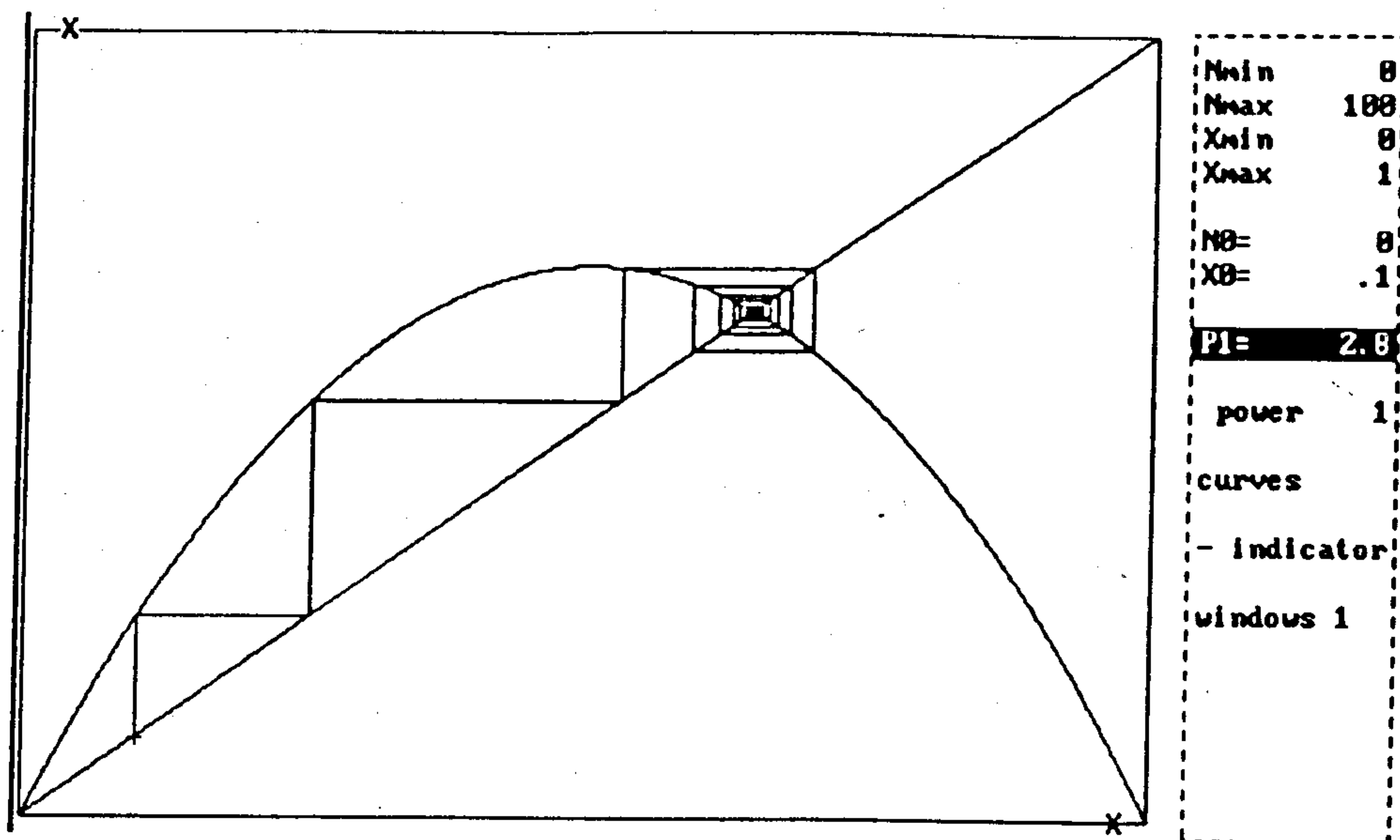


Figure 18a. A 1-D staircase plot of the logistic map (4): $\alpha = 2.8$ (P1).

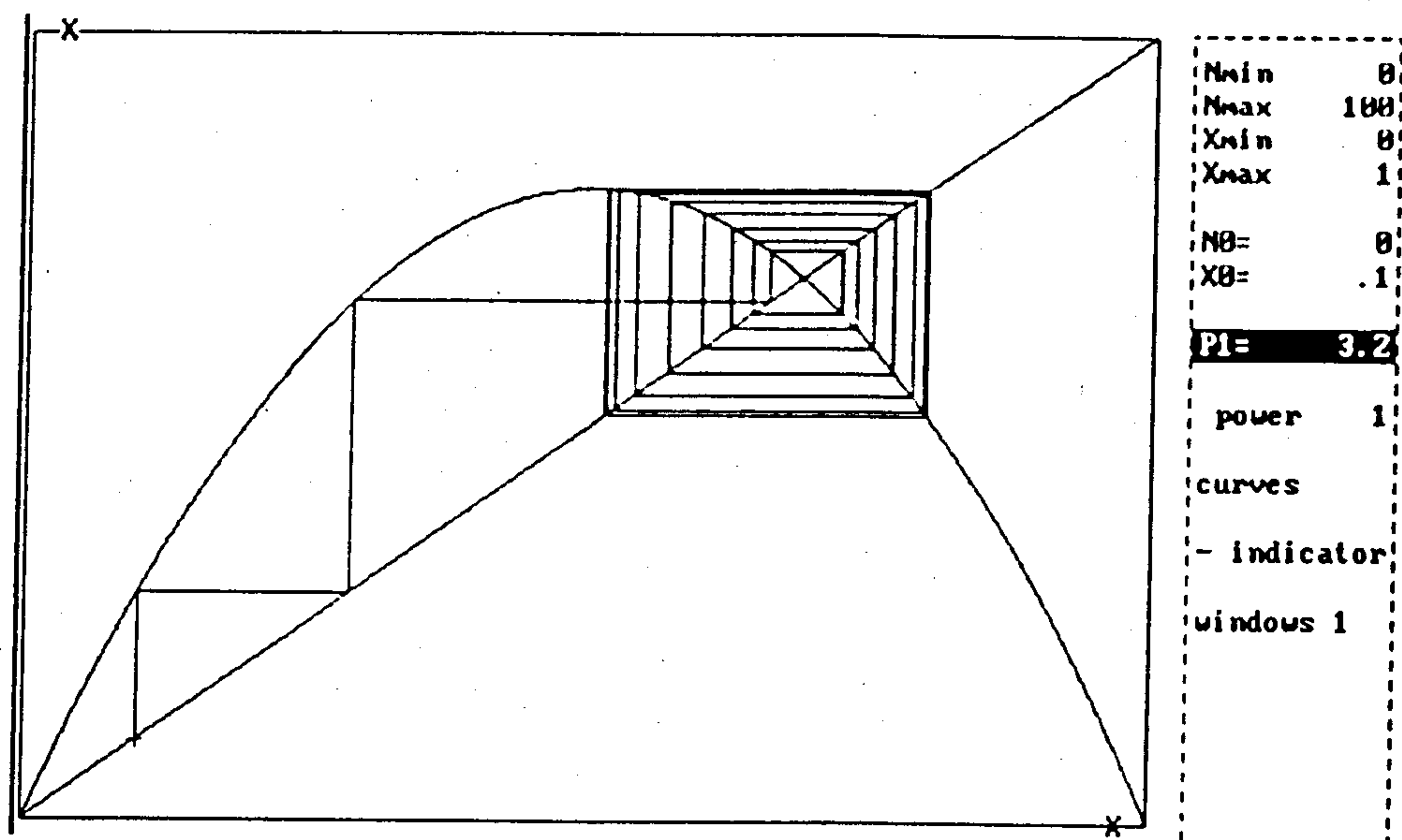


Figure 18b. The 1-D staircase plot of the logistic map (4): $\alpha = 3.2$ (P1).

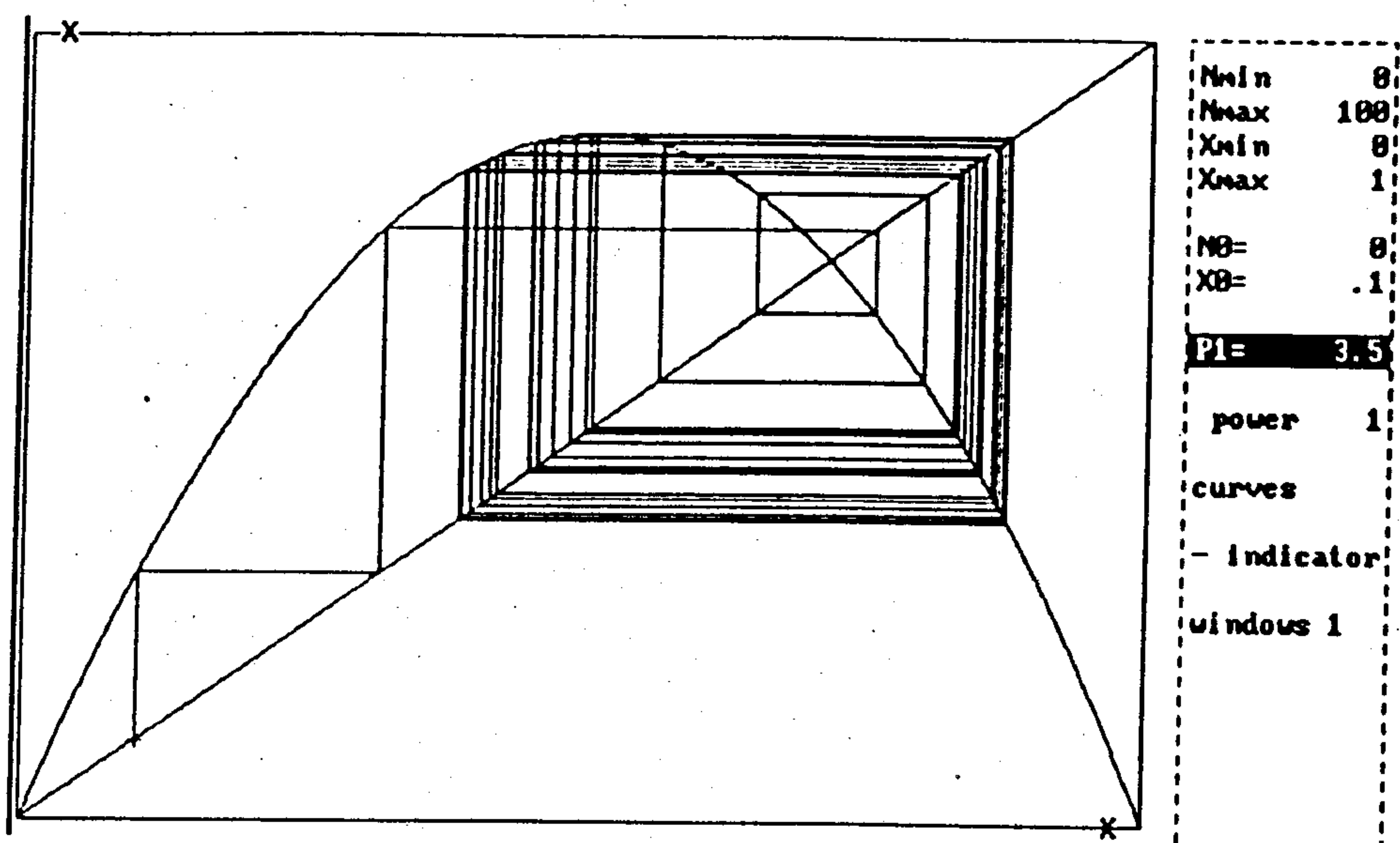


Figure 18c. The 1-D staircase plot of the logistic map (4): $\alpha = 3.5$ (P1).

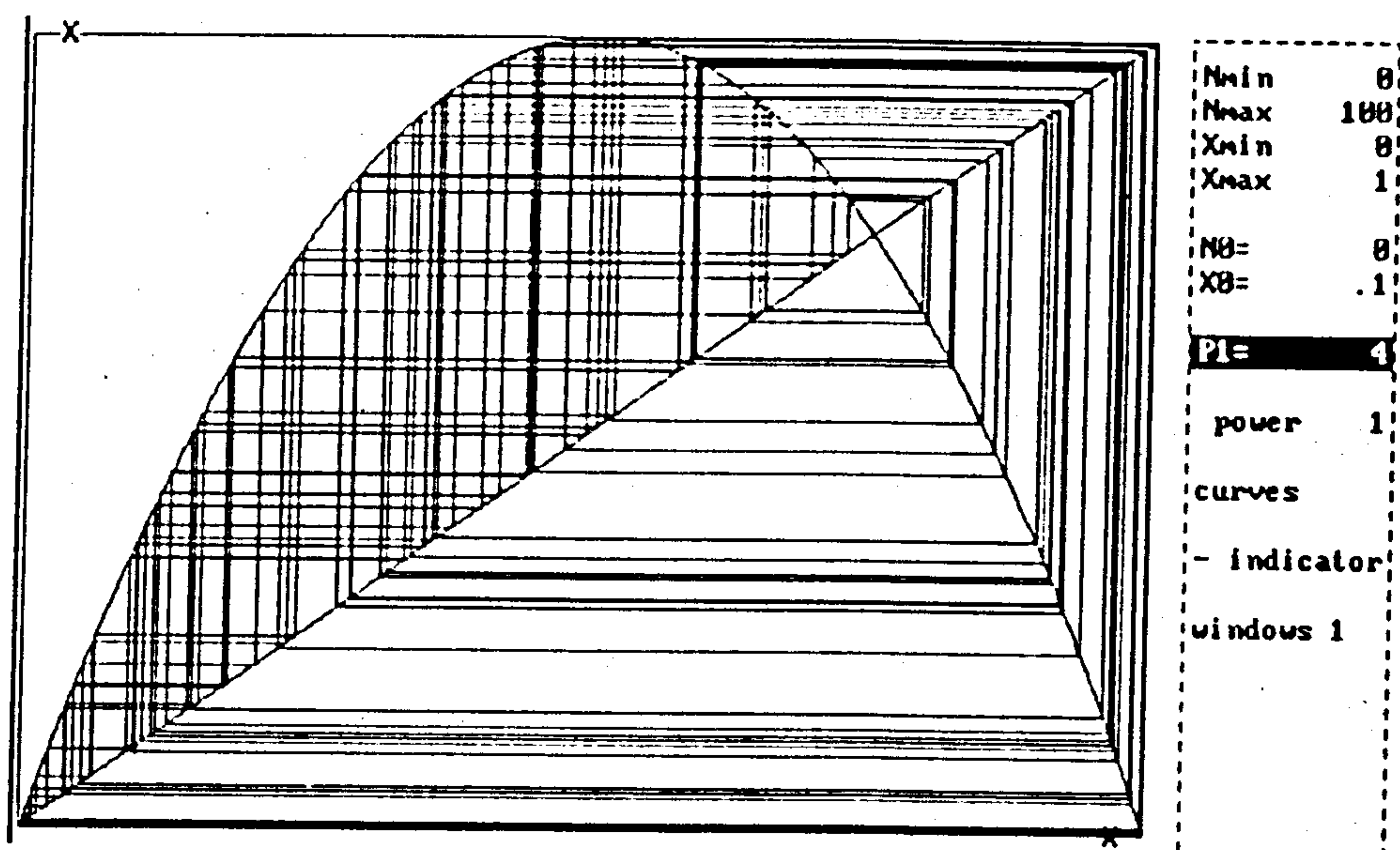


Figure 18d. The 1-D staircase plot of the logistic map (4): $\alpha = 4.0$ (P1).

Repeat these experiments with a power map (see Figure 19). Pay attention that when $P1=4$, the continuation of a trajectory for very large n leads to a solution which fills the entire part of the plane. This is typical of chaotic behavior.

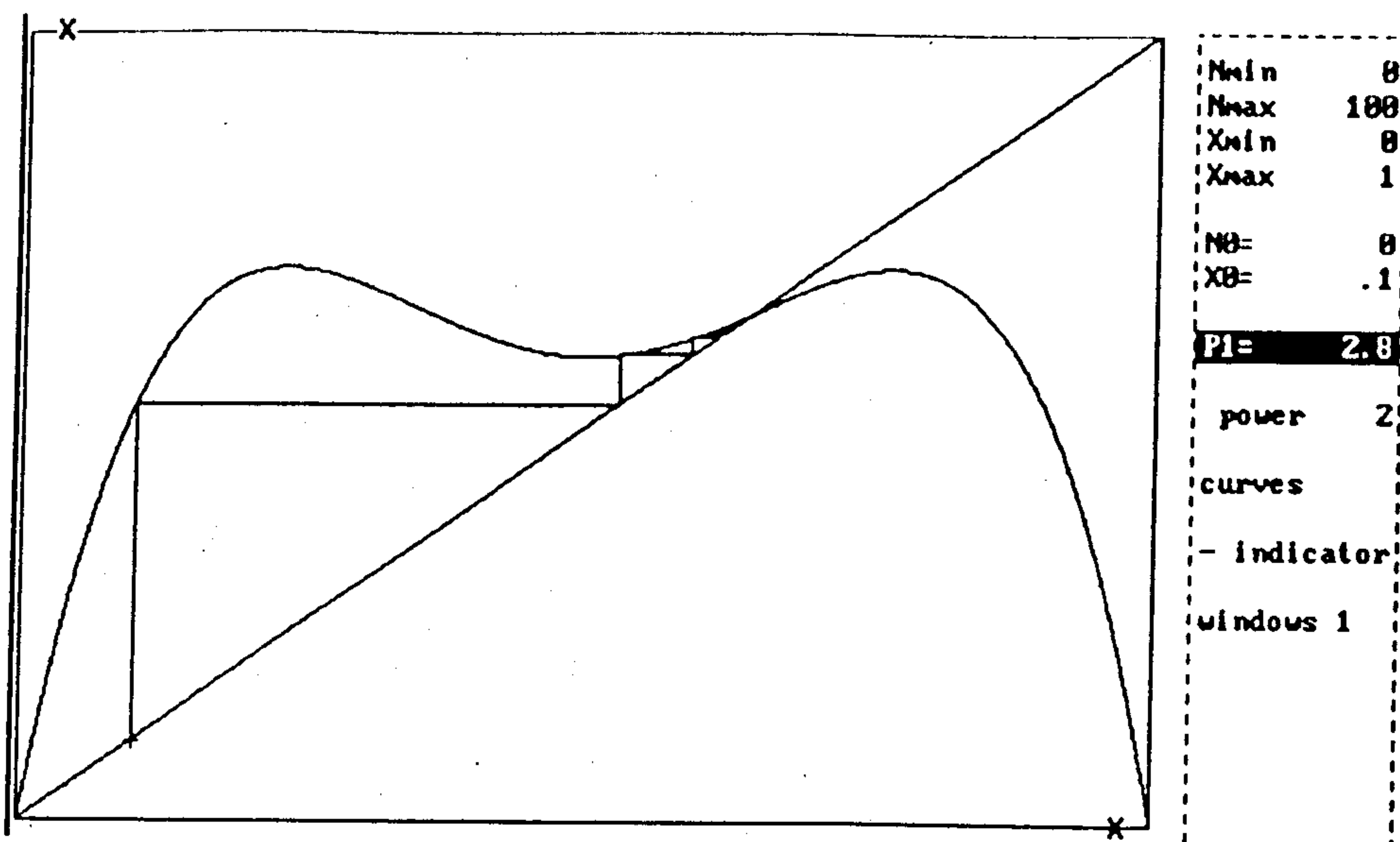


Figure 19a. The same simulation as in Figure 18a but using degree 2 (power 2) of the map.

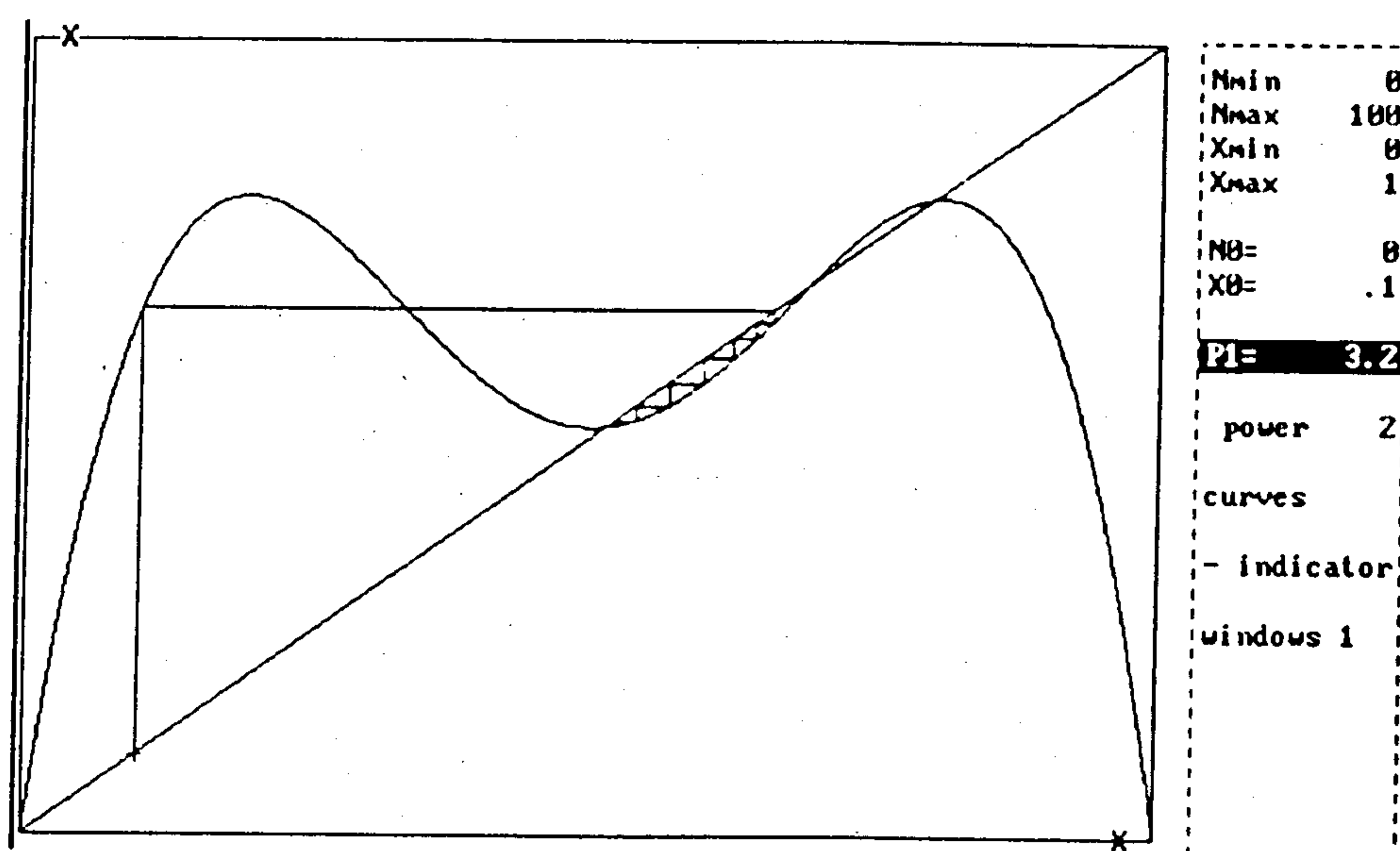


Figure 19b. The same simulation as in Figure 18a but using degree 2 (power 2) of the map.

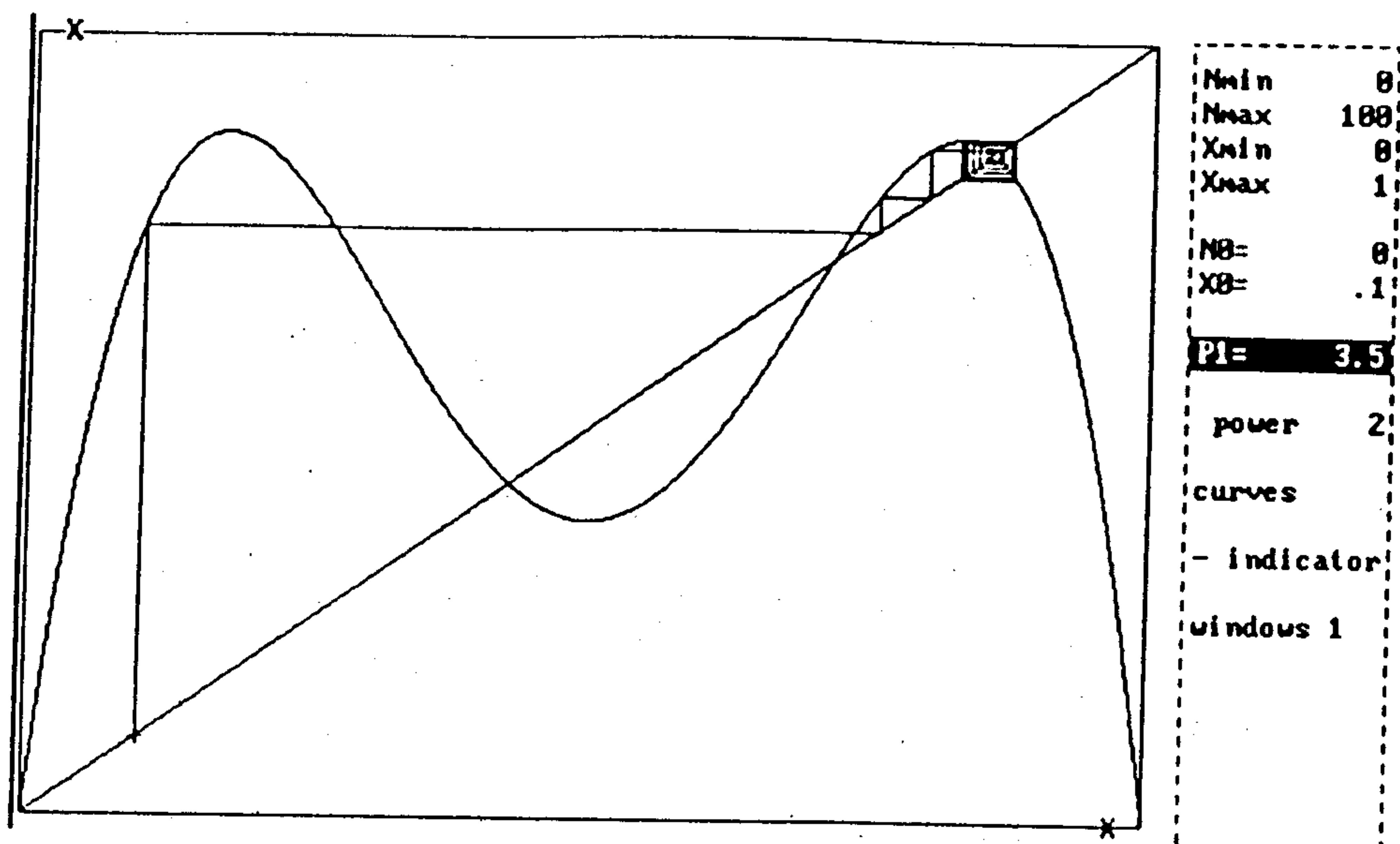


Figure 19c. The same simulation as in Figure 18a but using degree 2 (power 2) of the map.

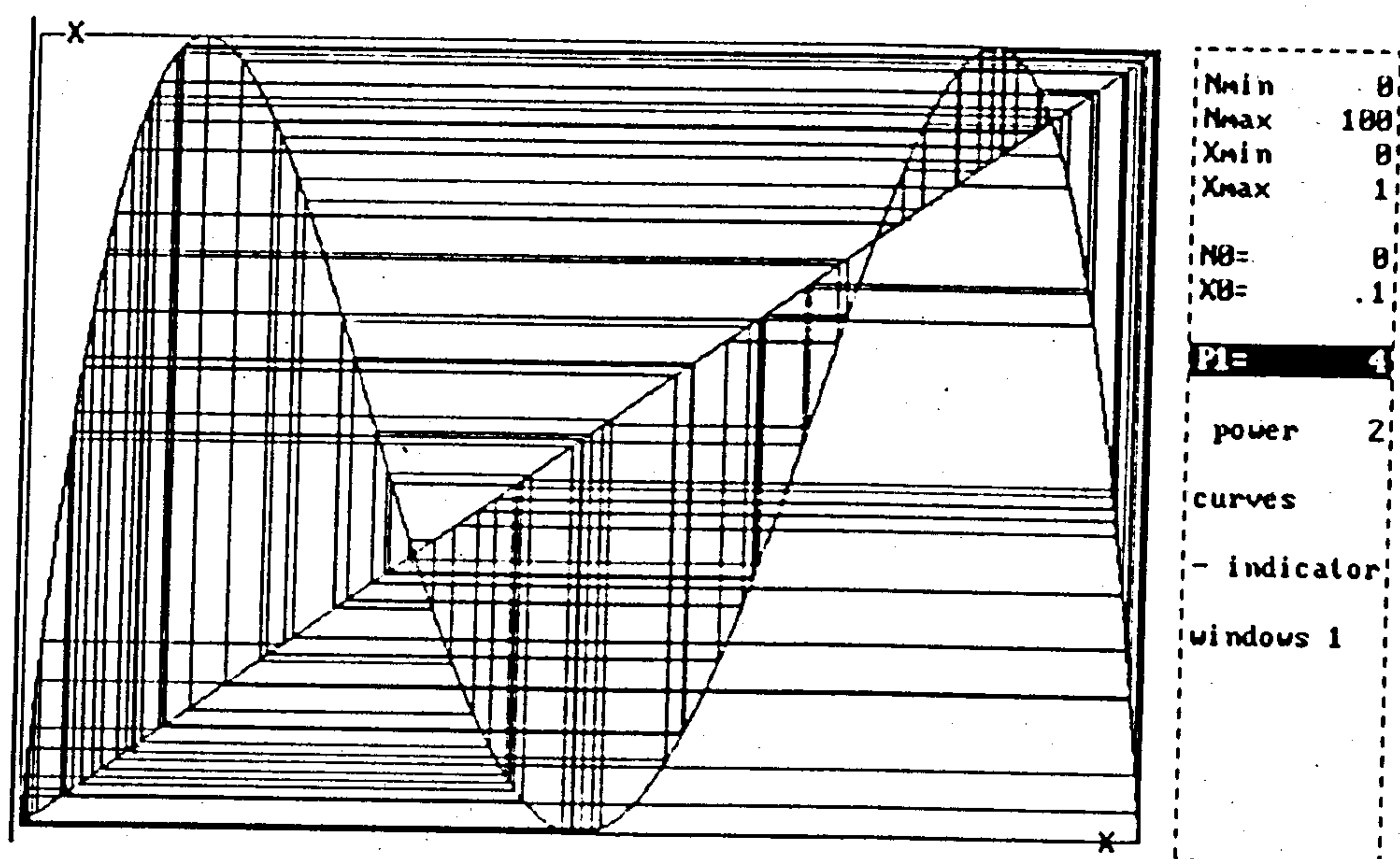


Figure 19d. The same simulation as in Figure 18a but using degree 2 (power 2) of the map.

Lesson 9. The microscope facility

Sometimes you may need to plot a phase portrait or some other picture in a large region and then magnify some portion of it. We'll learn how this can be done in TraX. Let's consider the Henon map (Henon, 1976):

$$\begin{aligned}x' &= 1 - a \cdot x^2 + y \\ y' &= b \cdot x\end{aligned}\tag{5}$$

with $a = 1.4$ and $b = 0.3$. This map produces trajectories on the plane $x - y$ which tend to a limit set called a *strange attractor*. The attractor has a complex structure and through magnification you'll be able to get a good idea of what this complexity means.

Select the entry **Difference equations** in the Archives, then select **Henon map**, and then the **Initial state** for Lesson 9. After entering the Investigation mode, plot the axes (**F8**) and begin the simulation (**F10**). The attractor you will see is shown in Figure 20a. Let's magnify one part of this attractor. First, you should activate the window $x-y$ (**F5**). Next, press **Shift-F4** and a small dotted frame will appear in the upper left corner of the window. You can re-size and move this window as you have already learned (i.e., you can move the window with the directional arrow keys and you can change the window size with **Shift-Right**, **Shift-Up**, etc.; see Lesson 5 for a review). Place this small frame, which is now active, over the part of the attractor which you want to magnify and press **Enter** to store the mathematical limits of the frame. To set these limits on the $x-y$ window, press **Shift-F6**. The limits X_{min} , X_{max} , Y_{min} , and Y_{max} will be updated and displayed in the Parameter Window. Clear the window (there is no automatic clearing!) and simulate again. The selected part of the attractor will now be plotted according to the new scale, and therefore, you have magnified a portion of the original plot (see Figure 20b). You may need more iterations of the map to get an improved picture because the density of points may decrease upon magnification. It is interesting to repeat the magnification of this already magnified portion of the attractor to appreciate the real complexity of its geometrical structure. In fact, the local structure of attractor is similar to the product of a line and an uncountable Cantor set!

Sometimes you may need to plot the entire attractor and the magnified portion on the same screen but in different windows. To do this you'll need to specify a new window with the same axes but with different limits. The limits can be set automatically by pressing **Shift-F6** in accordance with the stored frame limits. Let's see how this is done.

Activate the Parameter window and set the initial limits to $X_{\min}=-1.5$, $X_{\max}=1.5$, $Y_{\min}=-0.5$, and $Y_{\max}=0.5$. Next, activate the x-y window and re-size it to free the lower right corner of the screen for the second window. Activate the window (**F5**), press **Shift-F7** and resize the window by using **Shift-Right**, **Shift-Up**, etc. Now press **Ins** to create a new window and press **Shift-F7** to be able to move it with the directional arrow keys. Move it to the empty part of the screen, resize it and rename the axes to x-y (use **Ctrl-F7** and **Ctrl-F8**). Now there are two x-y windows on the screen with the same limits; one in the upper part and another in the lower part. If the limits of each screen are not the same, active each window in turn with **F5** and observe what the limits are in the Parameter window. Set them equal if they are not equal already. Compute the trajectory (**F10**) and the plots will be the same in both windows because they have the same limits. Activate the first (upper left) window and then press **Shift-F4** and set the frame over the portion of the map which you want to magnify and then press **↵**. Next, activate the second window and press **Shift-F6**. The second window will now receive the new limits corresponding to those of the small frame overlying the full-size map. Clear this window (**Alt-F7**) and start the computation again. The screen should look similar to Figure 21. Now you have learned a valuable and important feature of TraX.

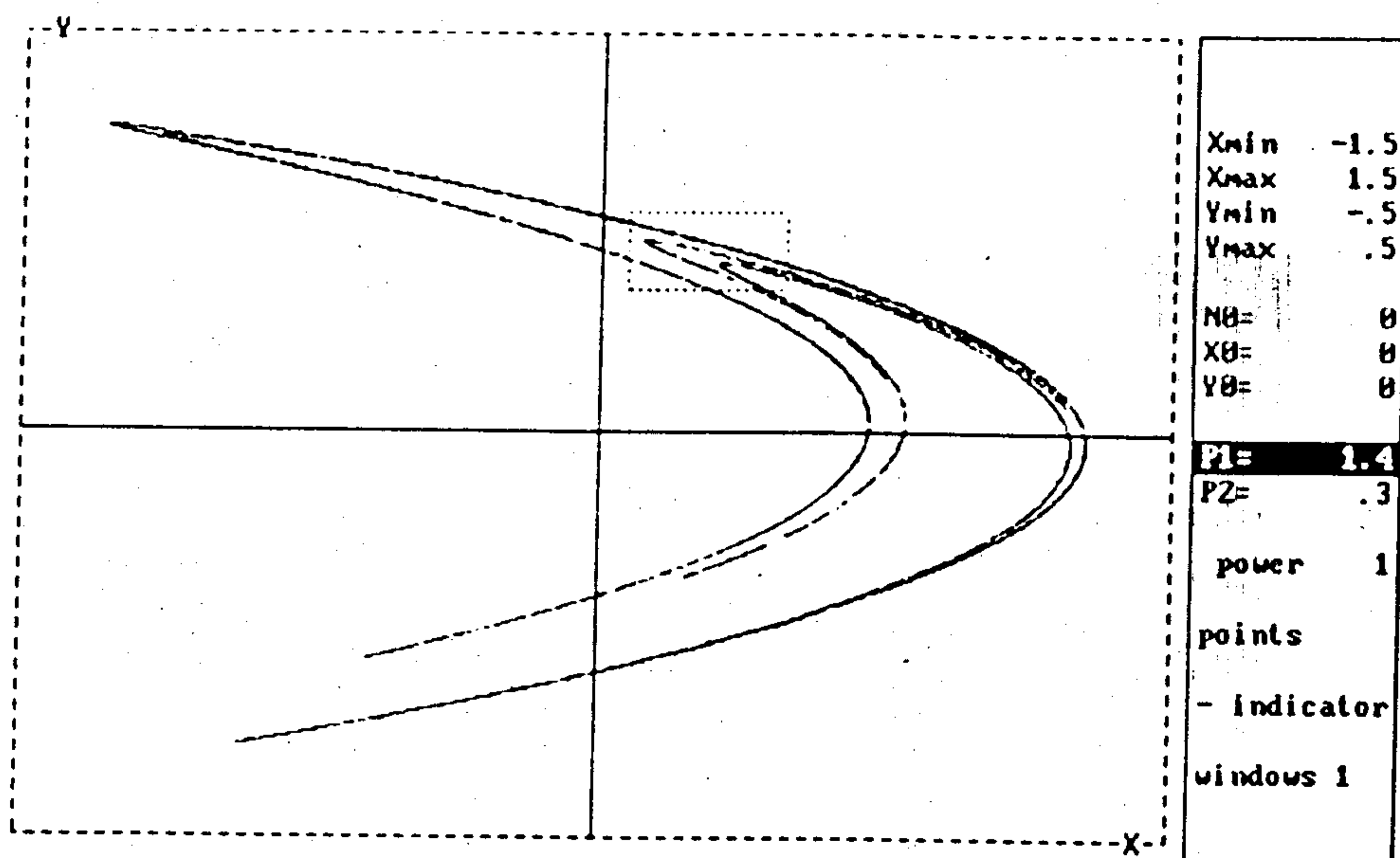


Figure 20a. The strange attractor in the Henon map (5). A window for magnification appears in the upper right quadrant.

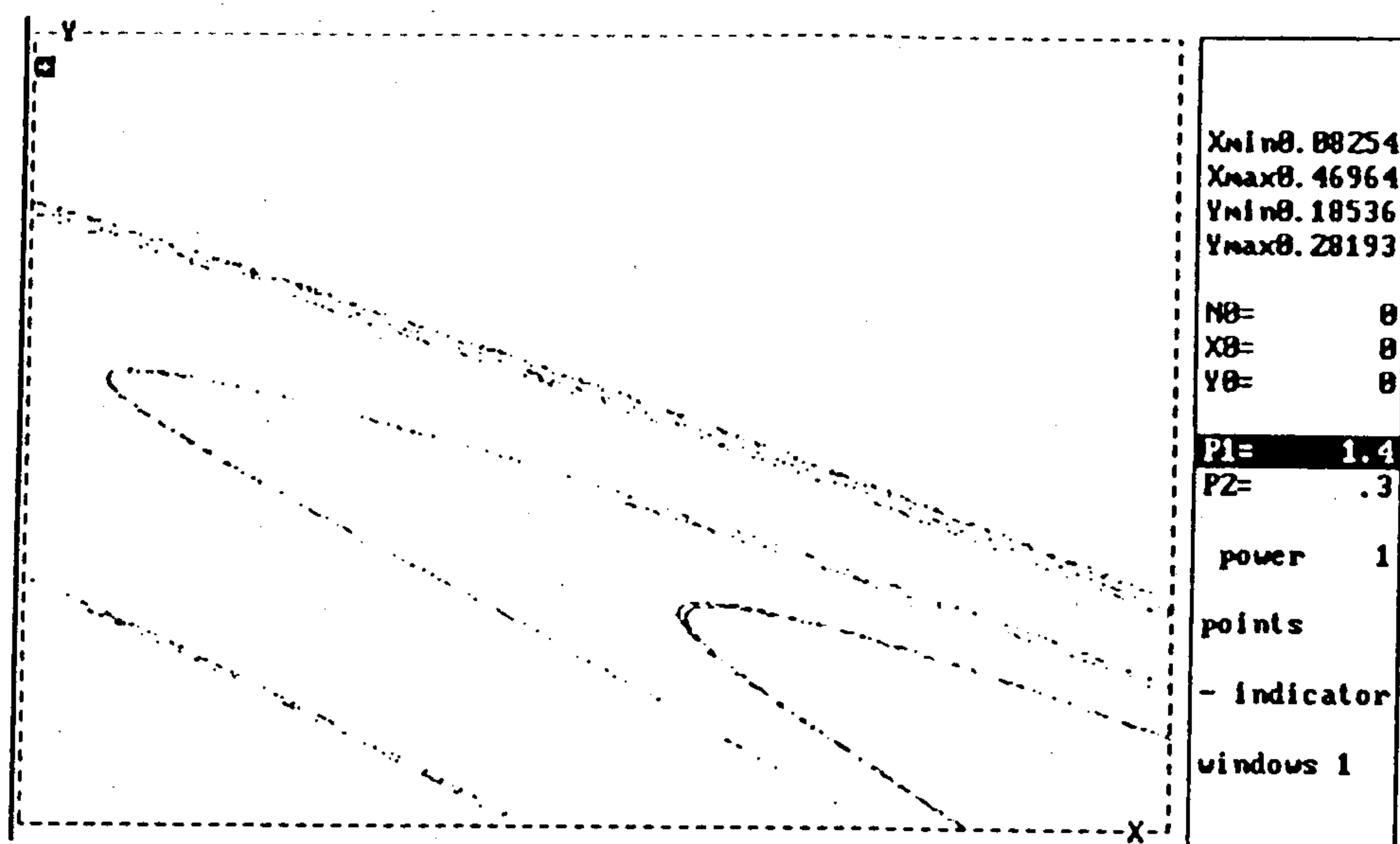


Figure 20b. The strange attractor in the Henon map (5) after magnification of the small window (marked box) in Figure 20a.

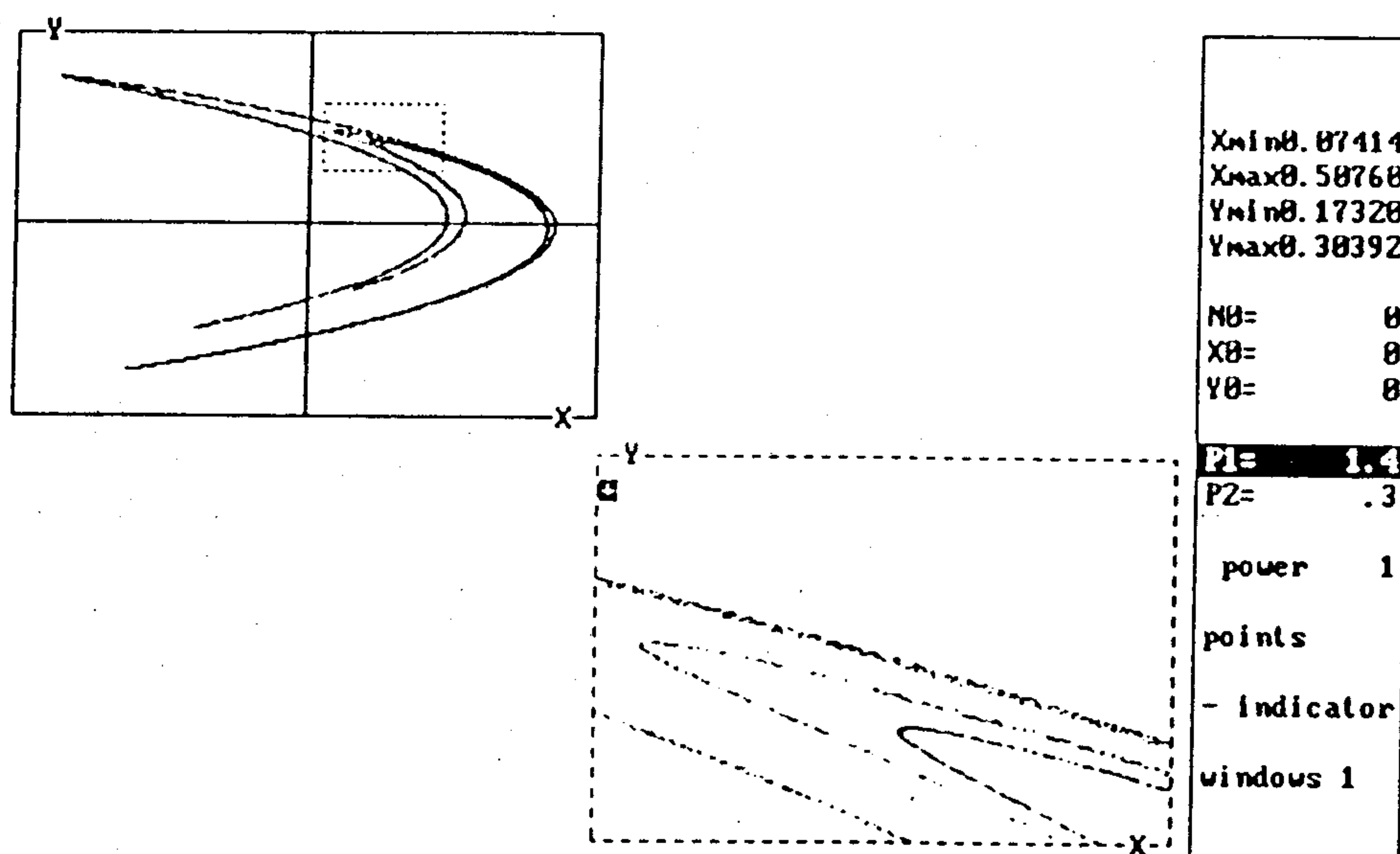


Figure 21. Using two graphic windows you can plot both the entire Henon attractor and a magnified area simultaneously.

You can repeat this procedure by, for example, highlighting the lower window containing the magnified section, press **Shift-F4**, select a section of this magnification, press **↩**, activate the other window (**F5**), press **Shift-F6** to reset its limits, clear the window with **Alt-F7** and you now have magnified an already magnified section but in an alternate window. Save the result of this lesson by storing the current state.

You should be aware, however, that saving pictures requires much more memory than saving only the equations and the state. For many systems which have an easily reproducible trajectory, such as simple or non-stochastic systems, it is generally not necessary to save the picture.

Advanced TraX

Lesson 10. Using functions in equations

In this lesson you will learn to apply user-defined functions when specifying right-hand sides (RHS) of model equations. Such functions may arise as a separately defined part of a model or as common expression in a RHS. The important thing is to have an opportunity to modify these functions independent of any particular model. TraX allows you to do this within the Investigation mode where you are likely to be changing parameter values. In addition, TraX provides a special option for plotting the graph of a user-defined or a standard function.

Consider the third-order ecological model (Bazykin *et al.*, 1983; Bazykin, 1985):

$$\begin{aligned}\frac{dx}{dt} &= F1(x) - z \cdot F2(x) - d \cdot (x - y) \\ \frac{dy}{dt} &= F1(y) - z \cdot F2(y) - d \cdot (y - x) \\ \frac{dz}{dt} &= g \cdot z \cdot \frac{F2(x) + F2(y)}{2} - c \cdot z\end{aligned}\tag{6}$$

This model describes the time evolution of prey and predator populations assuming that the prey population occupies two identical areas with migration occurring between them, and the predators consume prey from both areas as if no border between them exists.

Variables x and y in (6) represent the prey densities within each of the areas, and z represents the density of predators. The function $F1$ describes the growth rate of the prey and $F2$ represents a sort of fitness function. We'll let the function $F2$ have the simple linear form:

$$F2(x) = b \cdot x,\tag{7}$$

and the function $F1$ has one of two forms:

$$F1(x) = a \cdot x \cdot (1 - e \cdot x)\tag{8}$$

or

$$F1(x) = a \cdot x^2 \cdot (1 - e \cdot x)\tag{9}$$

The third-order model (6) has been derived for studying how a spatial (or dissipative) structure arises in an ecosystem. In this case that means there exists a stable steady-state for the prey densities in each of the two distinct areas. We'll try both forms of the function $F1$ in this lesson.

Select in the Archives the entry Differential equations and then press **[→]**. Next, press **[Shift-Ins]** and after entering the name of the equation into the TraX Editor, specify the equations in the form shown in Figure 22. Pay particular attention to how we denote the parameters of model (6): $\alpha = p1$, $b = p2$, $c = p3$, $d = p4$, $e = p5$, $g = p6$. The symbols $a0, a1, \dots, a9$ are the standard names of a function's formal arguments, and $f0, f1, \dots, f9$ are the standard names for user-defined functions.

```

dX/dt=
F1(x)-z*f2(x)-p4*(x-y)

dY/dt=
F1(y)-z*f2(y)-p4*(y-x)

dZ/dt=
p6*z*(F2(x)+F2(y))/2-p3*z

F1(a0)=
p1*a0*(1-p5*a0)

F2(a0)=
p2*a0

```

Figure 22. Specification of the model (6),(7), and (8) in the TraX Editor.

After the equations have been entered (don't forget about **[Shift-Esc]** if you need to correct the formulas before finishing), you will enter the Investigation mode. Examine the context of the Parameter window (scroll to see any hidden parameters) and pay attention to the entries $F1 = p1 \cdot a0 \cdot (1 - p5 \cdot a0)$ and $F2 = p2 \cdot a0$ which contain the specified functions: these can be easily modified during the investigation. You should be aware that these equations are already in the Archives and while you'll learn TraX better by doing everything yourself, as in this tutorial, often, having something to fall back on is equally useful. Look into the Differential equations archives called Bilocal predator-prey model.

It may be easier to follow this tutorial if you rearranged the order of the items in the parameter window to that shown in Figure 23a. To rearrange the items in the Parameter window first highlight the item which you desire to move and then press **[Ctrl-PgUp]** to move the item up or **[Ctrl-PgDn]** to move the item down.

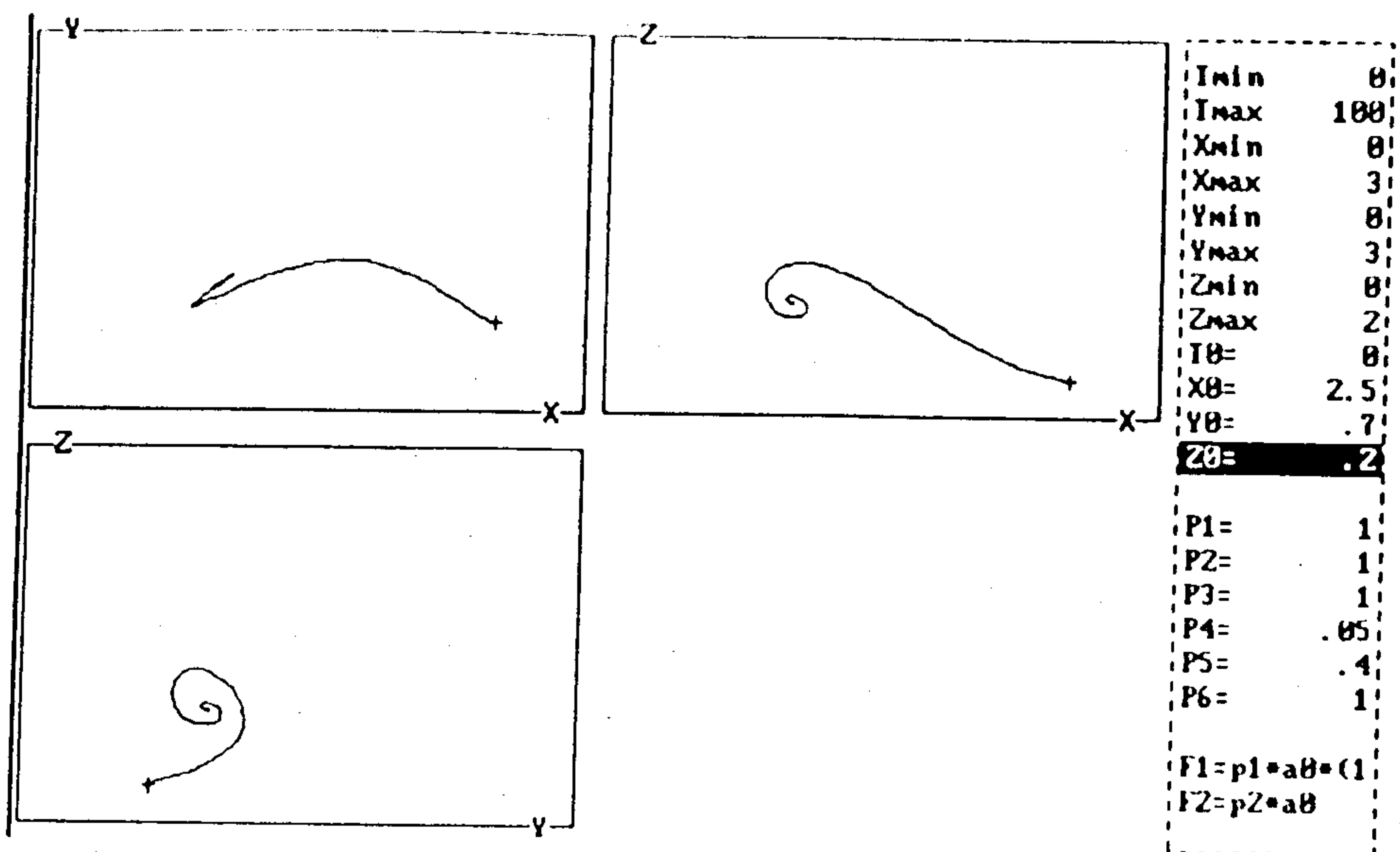


Figure 23a. Simulation of the model (6),(7), and (8): phase projections.

Next, make the following settings: $T_{min}=0$, $T_{max}=100$, $X_{min}=0$, $X_{max}=3$, $Y_{min}=0$, $Y_{max}=3$, $Z_{min}=0$, $Z_{max}=2$, $T_0=0$, $X_0=2.5$, $Y_0=0.7$, $Z_0=0.2$, $P_1=1$, $P_2=1$, $P_3=1$, $P_4=0.05$, $P_5=0.4$, and $P_6=1$. Now, begin the simulation. The result is shown in Figure 23a; it indicates that the trajectory tends to an equilibrium point for which $x = y$. Thus, the distribution of the prey becomes spatially uniform asymptotically, although the initial distribution was nonuniform. Try some other initial values but the result should be the same. After some numerical experiments it can be concluded that model (6), using the functions (7) and (8), does not have a stable spatial structure.

Change to the group windows 2 and repeat the simulations (F10) by plotting the trajectories in the three windows, $t-x$, $t-y$, and $t-z$ (see Figure 23b); when you're finished return to the group windows 1.

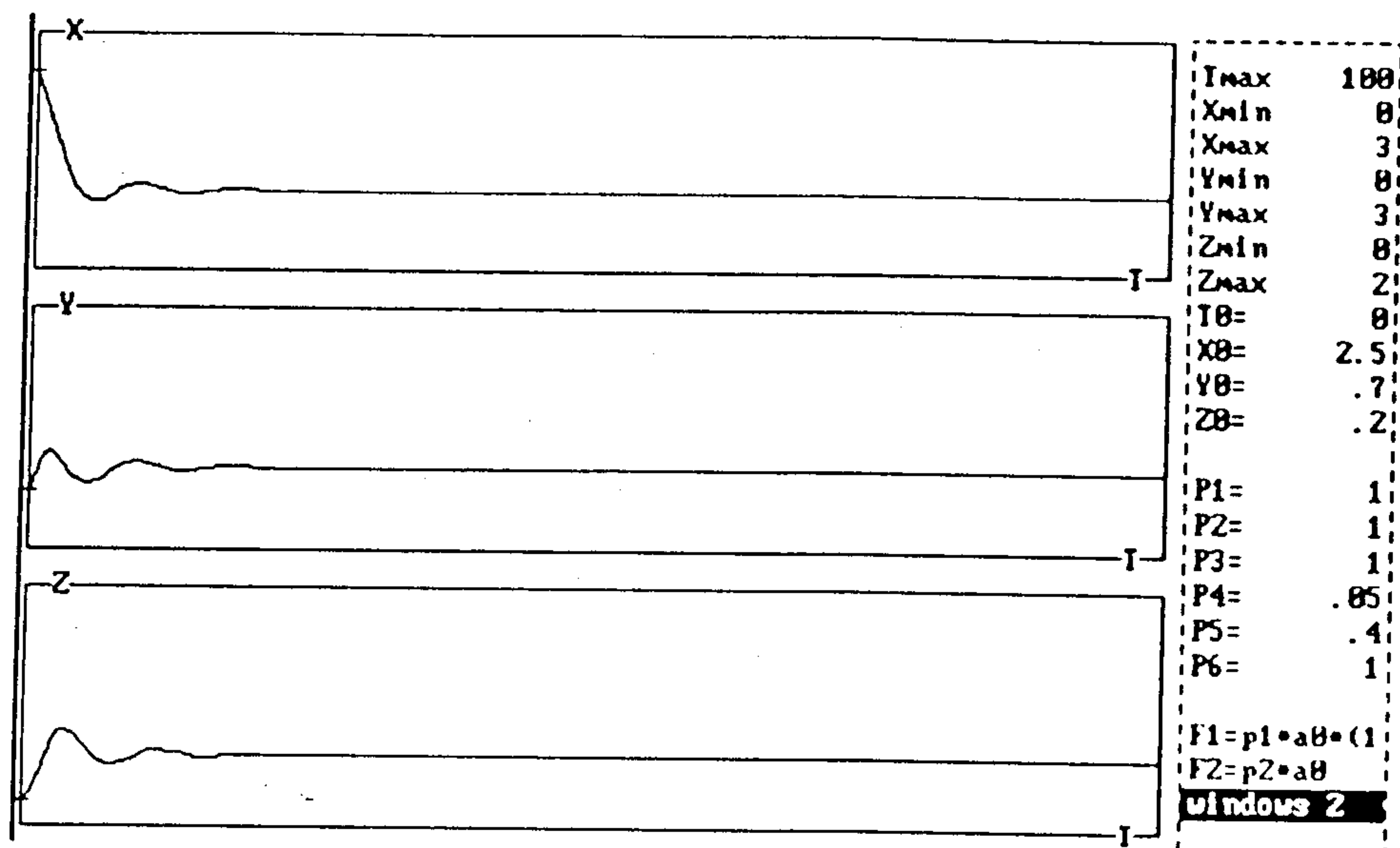


Figure 23b. Simulation of the model (6),(7), and (8): time series of x , y , and z .

Now let's try the second form of function $F1$ (9). To transform (8) to (9), place the highlight on entry $F1=p1*a0*(1-p5*a0)$ and press \leftarrow . Then the screen should look like Figure 24. Edit the expression to obtain the formula $F1=p1*a0*a0*(1-p5*a0)$ and then press \leftarrow . The appearance of the highlight in the Parameter window means that the modified function has been accepted, otherwise an error message will appear. Alternatively, you may select this new modification (9) from the Archives; press F2 twice and then select the state Function $F1$ (as in (9)).

$F1(a0) =$
 $p1*a0*(1-p5*a0)$

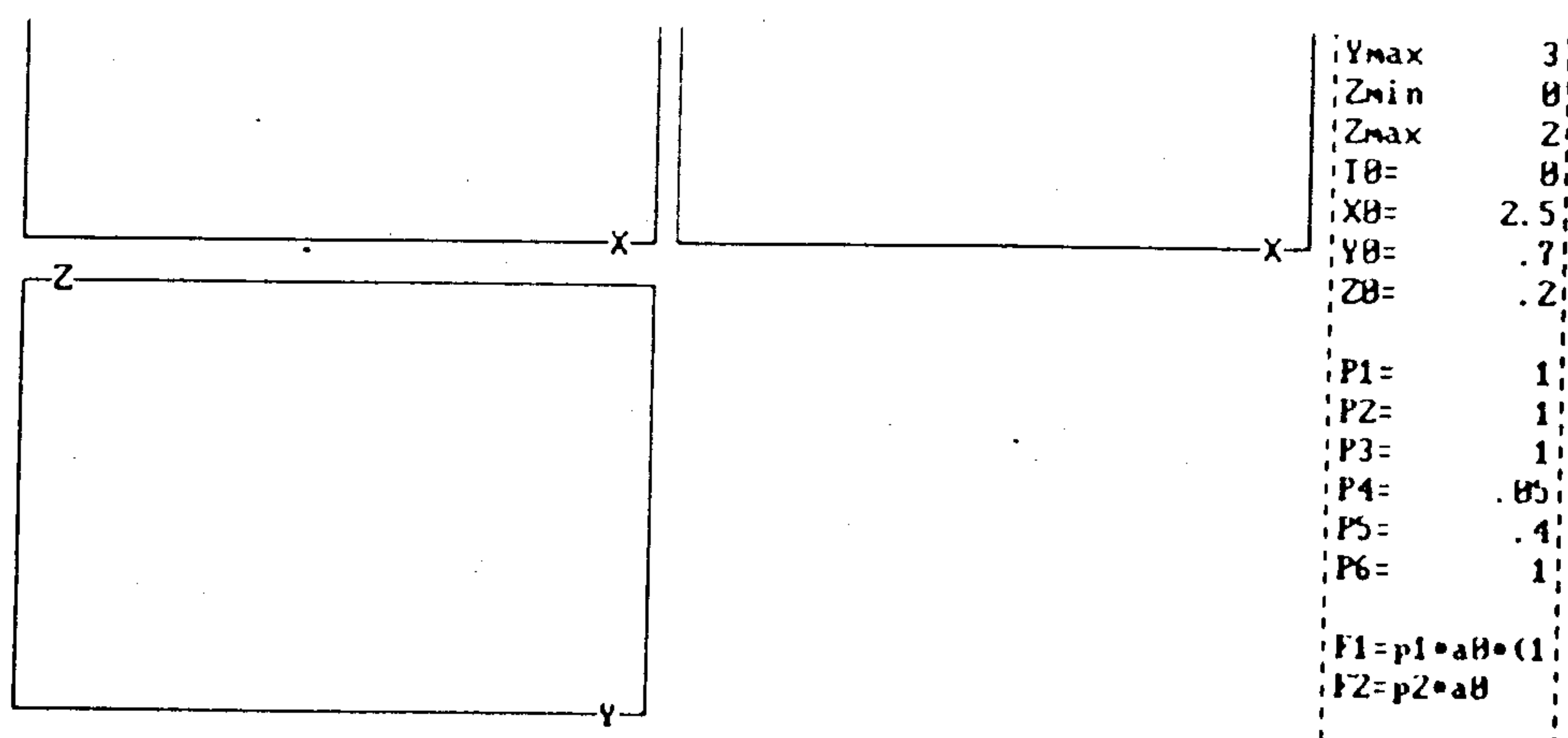


Figure 24. Editing the function $F1$. Form (8) should be replaced by form (9).

Now let's simulate the modified system with the same parameters and initial conditions as above (see Figure 23). Figures 25a and 25b show that the distribution of prey between the areas approaches uniform, but periodic oscillations have now appeared! Try some other initial conditions. If you set $Z_0=0.3$ while retaining the other coordinates the simulations produce the results shown in Figure 26a and 26b. Here, the trajectory approaches a steady-state but with different values of x and y . This steady-state represents the nonuniform spatial structure of the prey population which we were looking for.

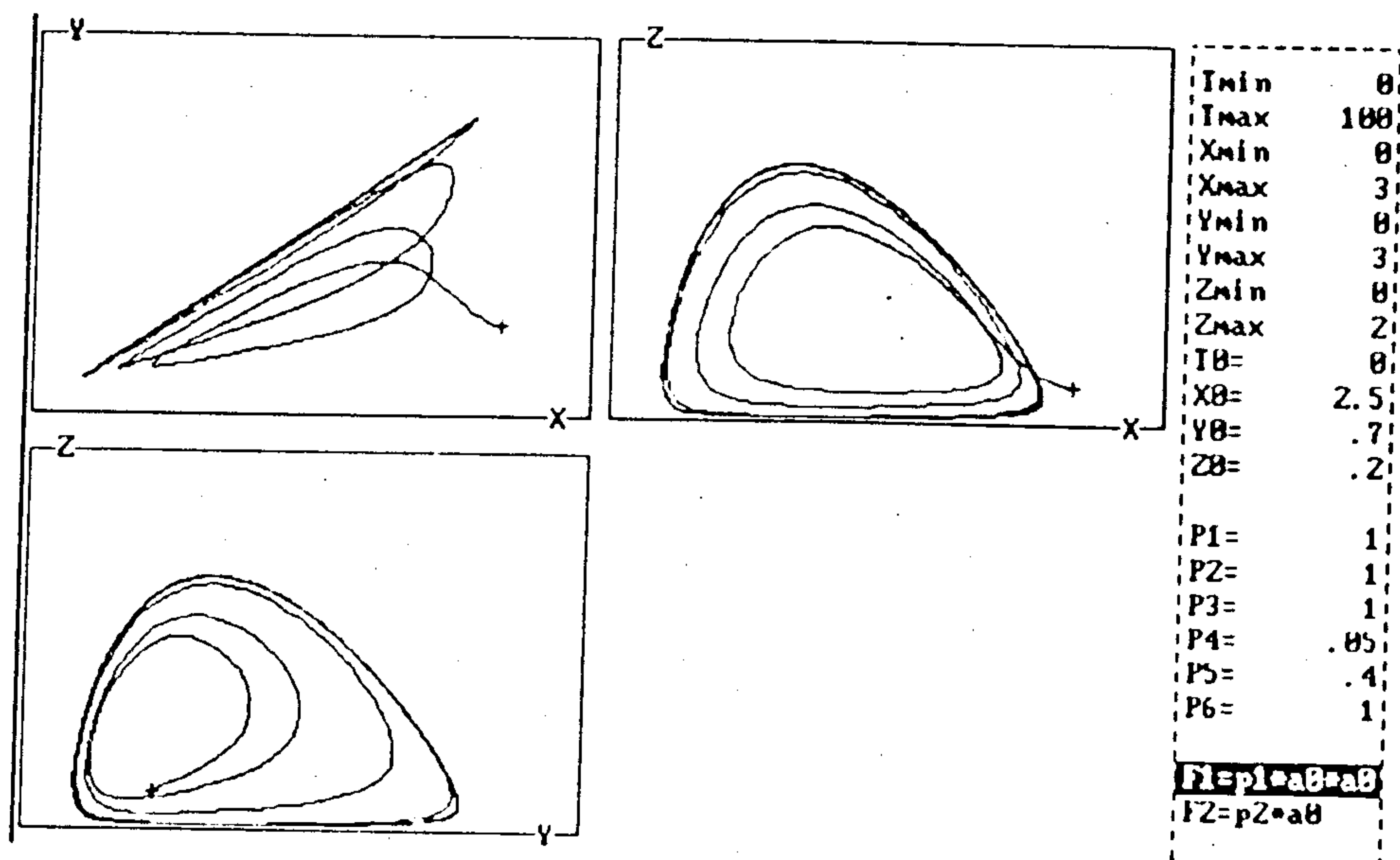


Figure 25a. After replacing (8) with (9) in the model (6), periodic oscillations appear: phase projections (compare with Figure 23a).

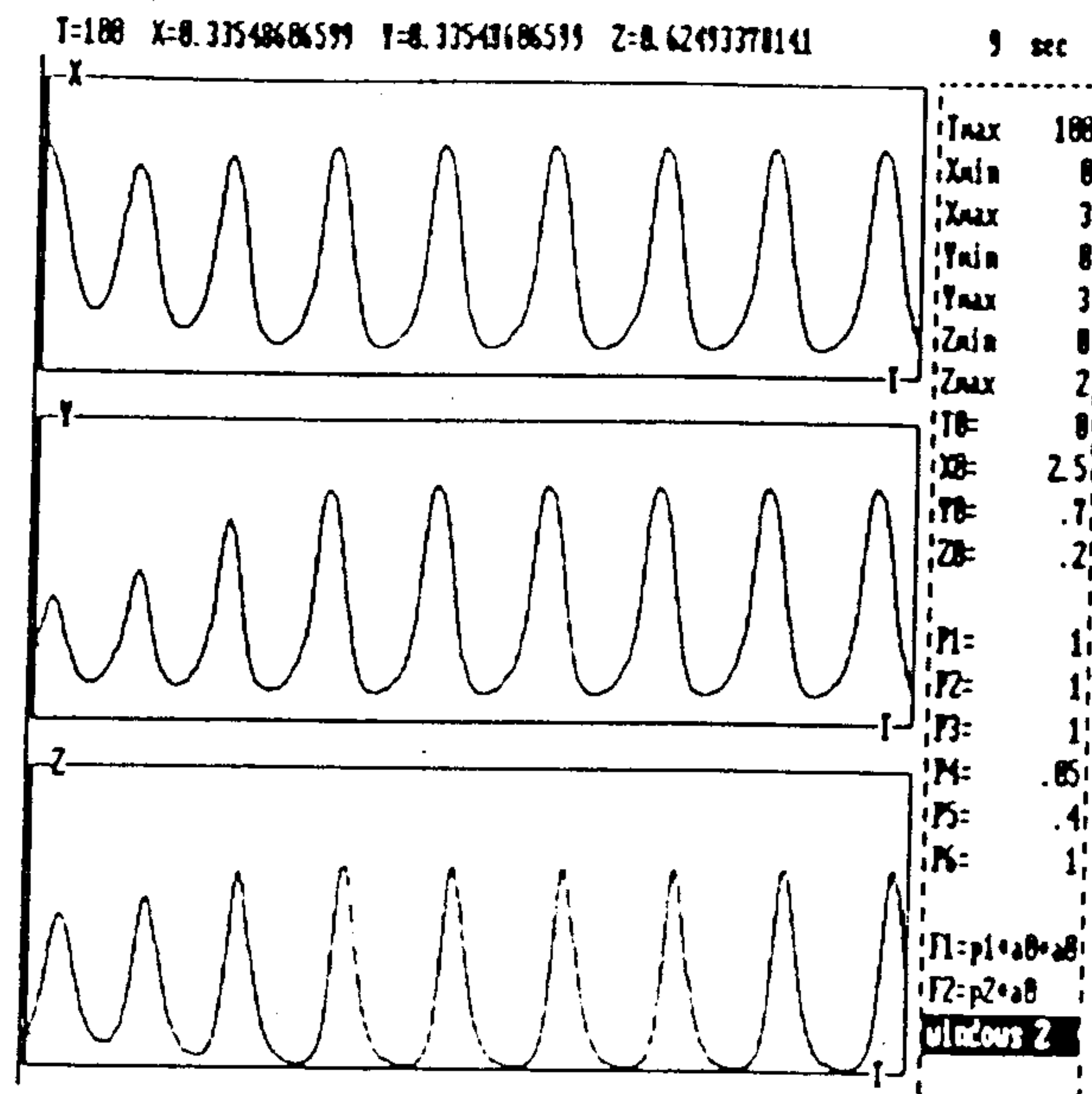


Figure 25b. After replacing (8) with (9) in the model (6), periodic oscillations appear: time series (compare with Figure 23b).

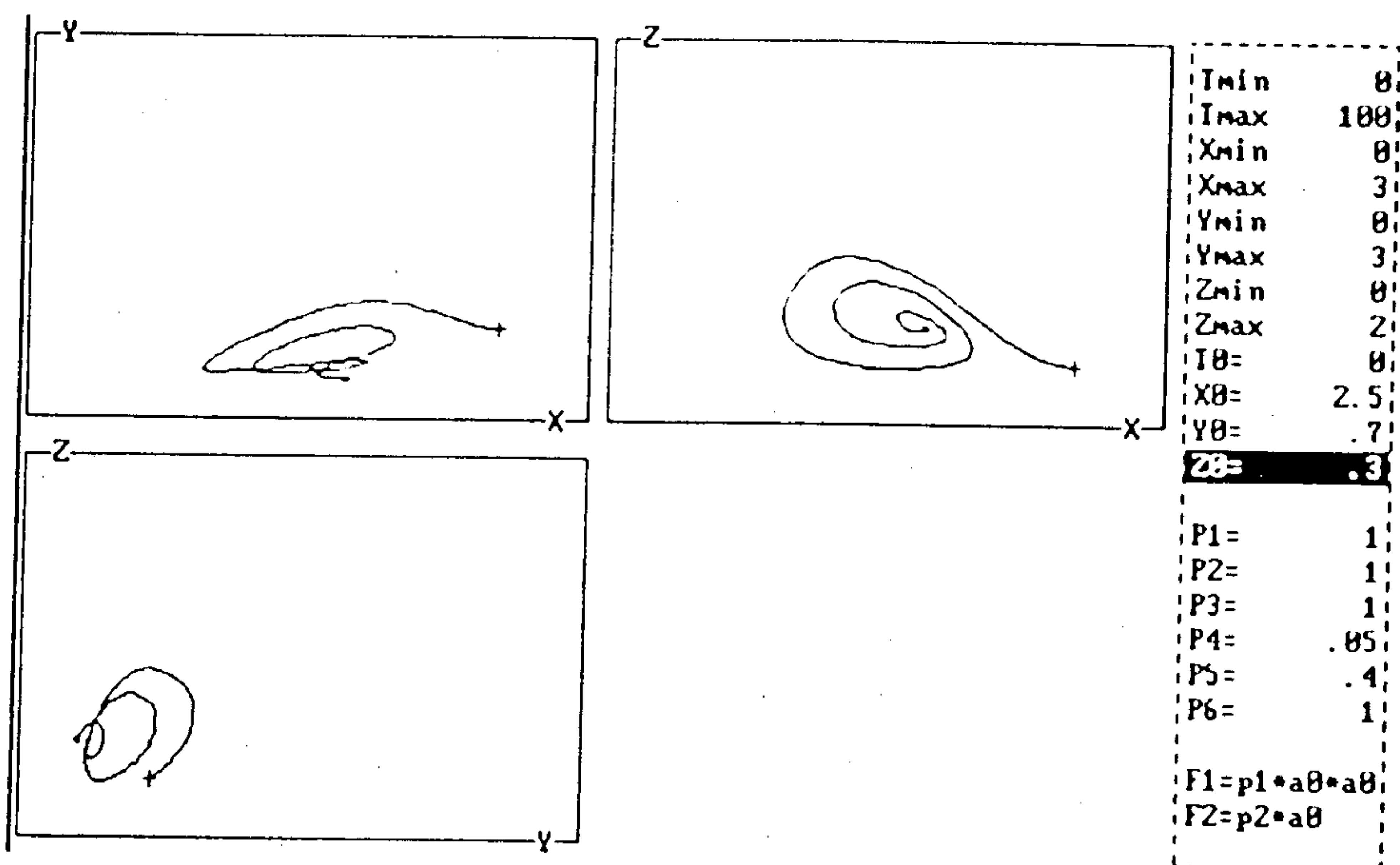


Figure 26a. By varying the initial conditions, the nonuniform distribution of the prey over two areas (dissipative structure) is discovered: phase projections (compare with Figure 25a).

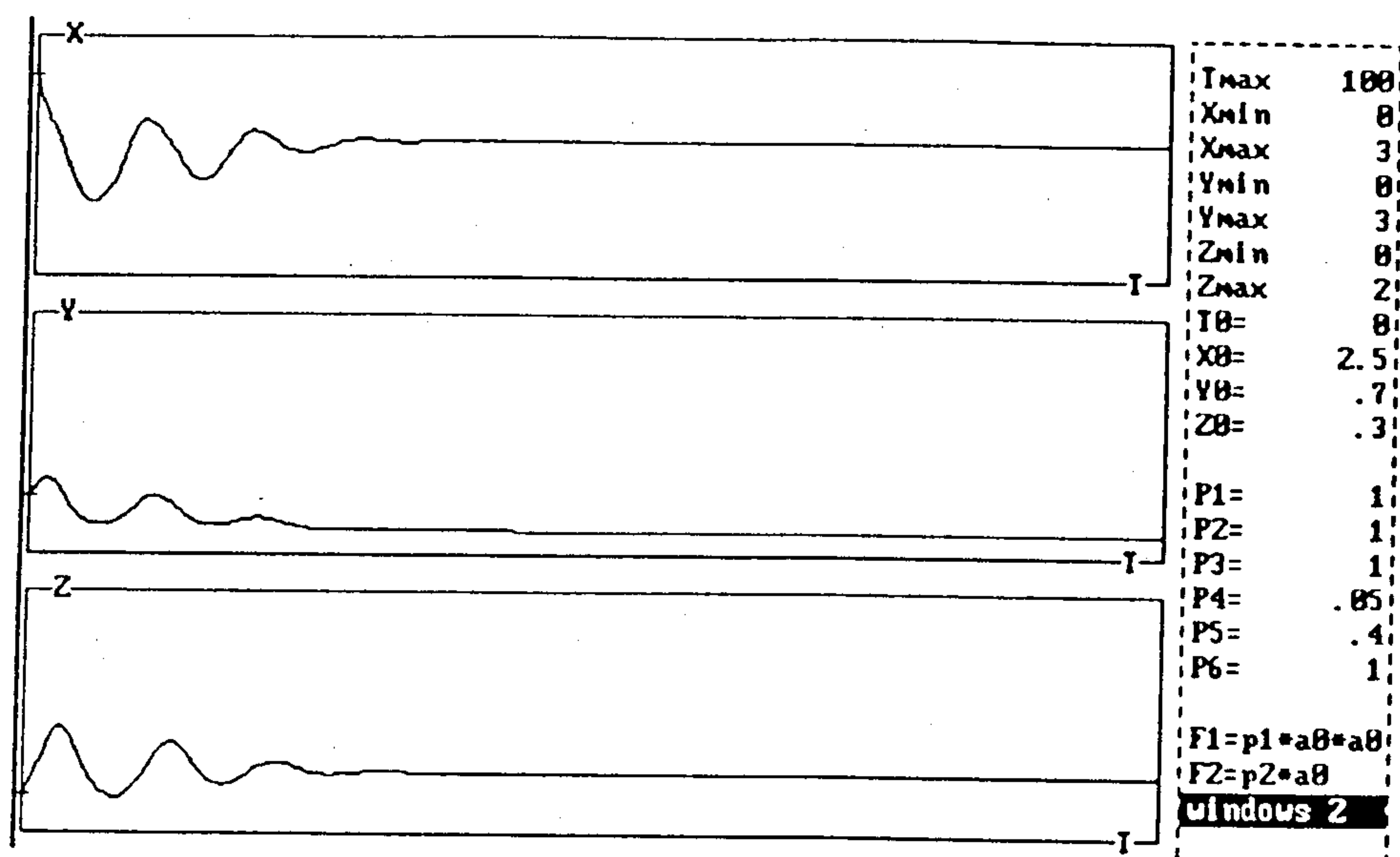


Figure 26b. By varying the initial conditions, the nonuniform distribution of the prey over two areas (dissipative structure) is discovered: time series (compare with Figure 25b).

It is of interest to plot both the solutions $x(t)$ and $y(t)$ using the same axes in order to underscore the nonuniform character of the prey distribution. In TraX this can be accomplished by overlapping the graphic windows. In this case, place the two windows $t-x$ and $t-y$ on the same place of the screen (*i.e.*, on top or over one another) and use the same limits for both abscissas and ordinates. Each of the overlapping windows however, still retains its "individuality" and is assessable by making it active. Let's try this option. Set windows 2 and then activate the $t-y$ window. Before moving this window it will be helpful to change the background color (color monitors only). Use **Ctrl-F5** to select a different background color. Now, press **Shift-F7** and move the window up and exactly over the $t-x$ window and press **Enter**. Of course, you can resize both of these windows to make them larger if you desire (see Figure 27a: activate the windows one at a time with **F5** then press **Shift-F7** and use **Shift** and the directional arrow keys). When this is done, press **F10** to simulate the model. The lower trajectory in the $t-x, y$ window is the y trajectory. Now you'll see the convenience of using a different color to represent this trajectory of this state variable. In addition, you may desire to label these trajectories by writing a comment on the screen. To do this press **F4**, move the text cursor with the directional arrow keys and then write your comment followed by **Enter**. Writing comments on the screen is particularly useful just prior to printing an image of the screen. Note, however, that comments within a window may be erased if the window is cleared or the window group is changed.

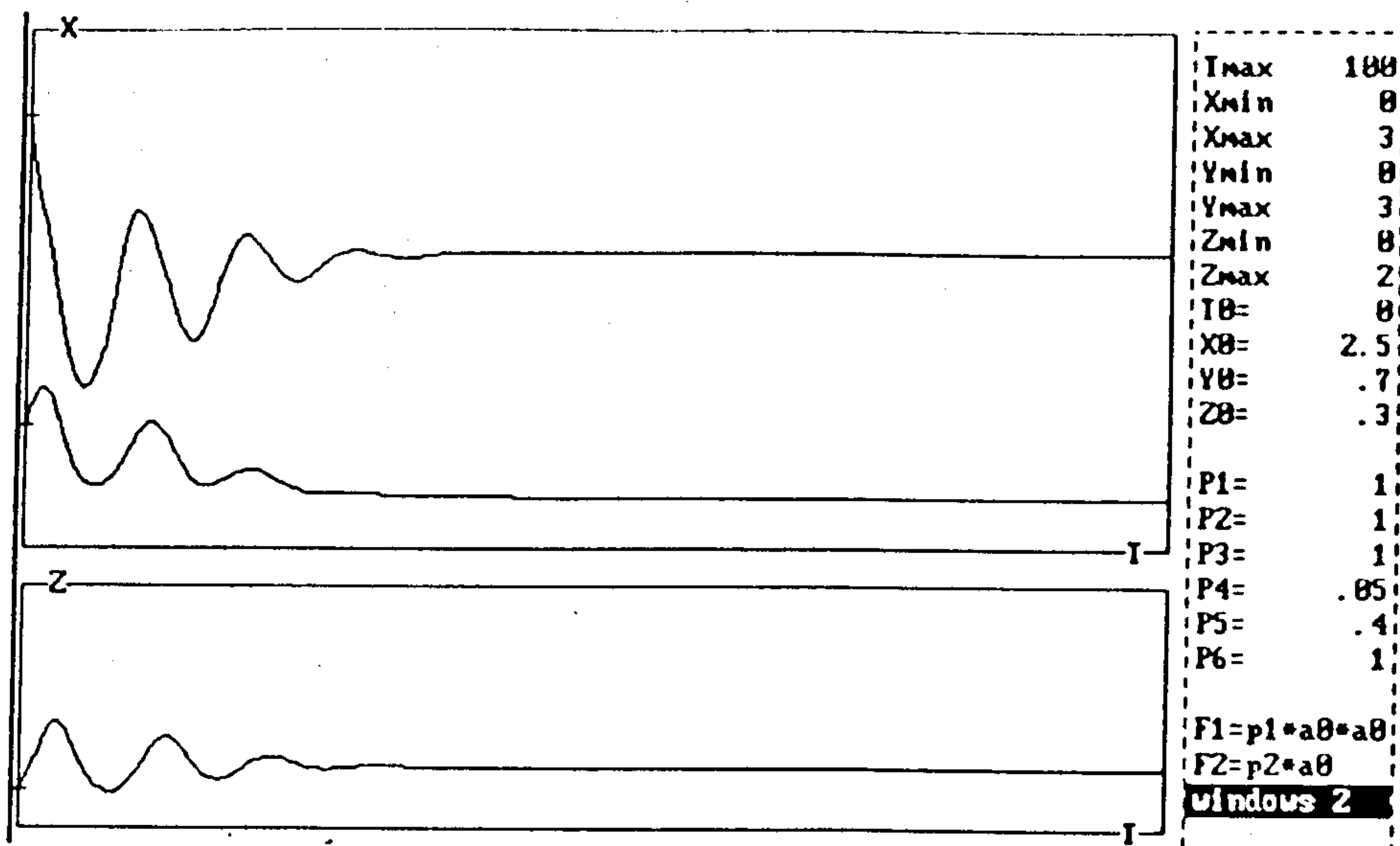


Figure 27a. Using overlapping windows t - x and t - y , the difference between the behavior of the prey in two different areas becomes more obvious.

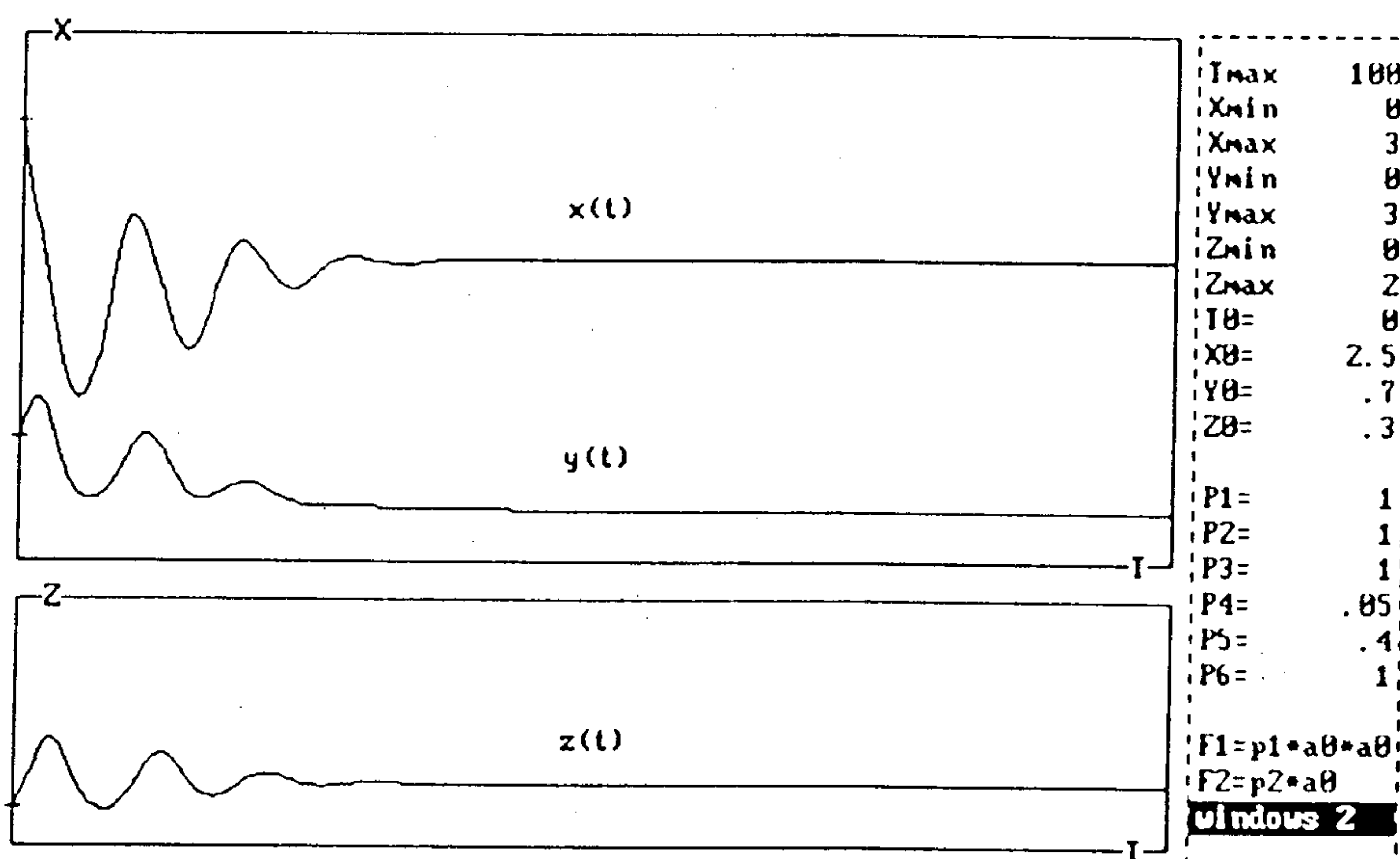


Figure 27b. Figure 27a illustrating the TraX commenting function for labeling the trajectories.

We'll finish this lesson by plotting function graphs. Activate the first group of windows and delete all the windows (press **Del** when a window is active). Next, specify a new window which has an abscissa X (use **Ctrl-F8**) and an ordinate $F1(x)$ (use **Ctrl-F7**). Set the limits of the window to: $X_{\min}=-1$, $X_{\max}=3$, $O_{\min}=-0.5$, $O_{\max}=1.5$ and then press **F6** (the new window must be active!). Then the graph of

function F1 specified by (9) will be plotted as shown in Figure 28a. To compare the graphs of F1 given by (8) and (9), modify F1 once more to obtain the initial form (8) (i.e., highlight F1, press \leftarrow , edit the function, press \leftarrow , make sure the graphic window is active and press F6). This graph will be plotted on the same screen, and the resulting picture with two graphs is shown in Figure 28b.

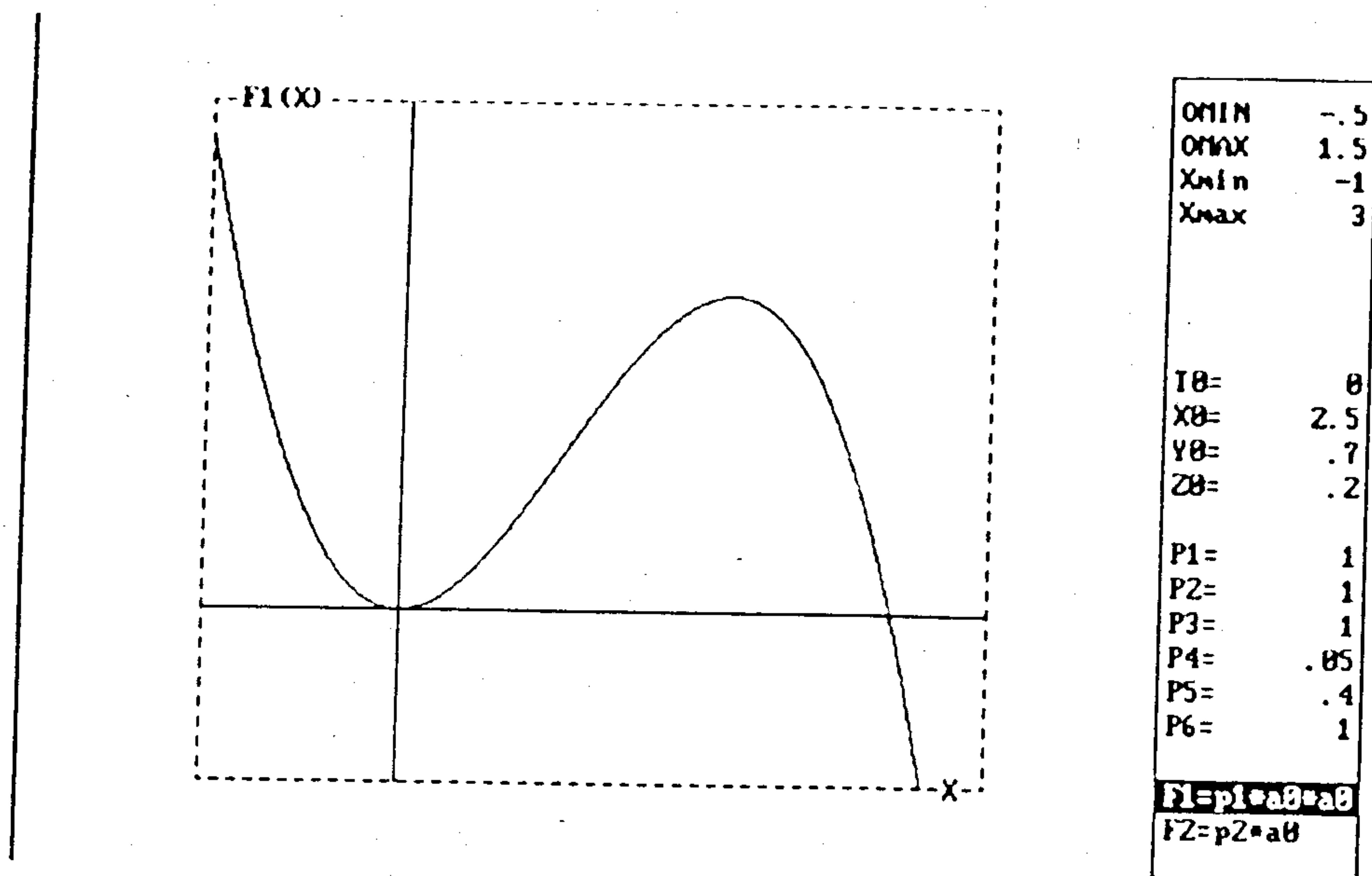


Figure 28a. Plotting the function graph for the function (9).

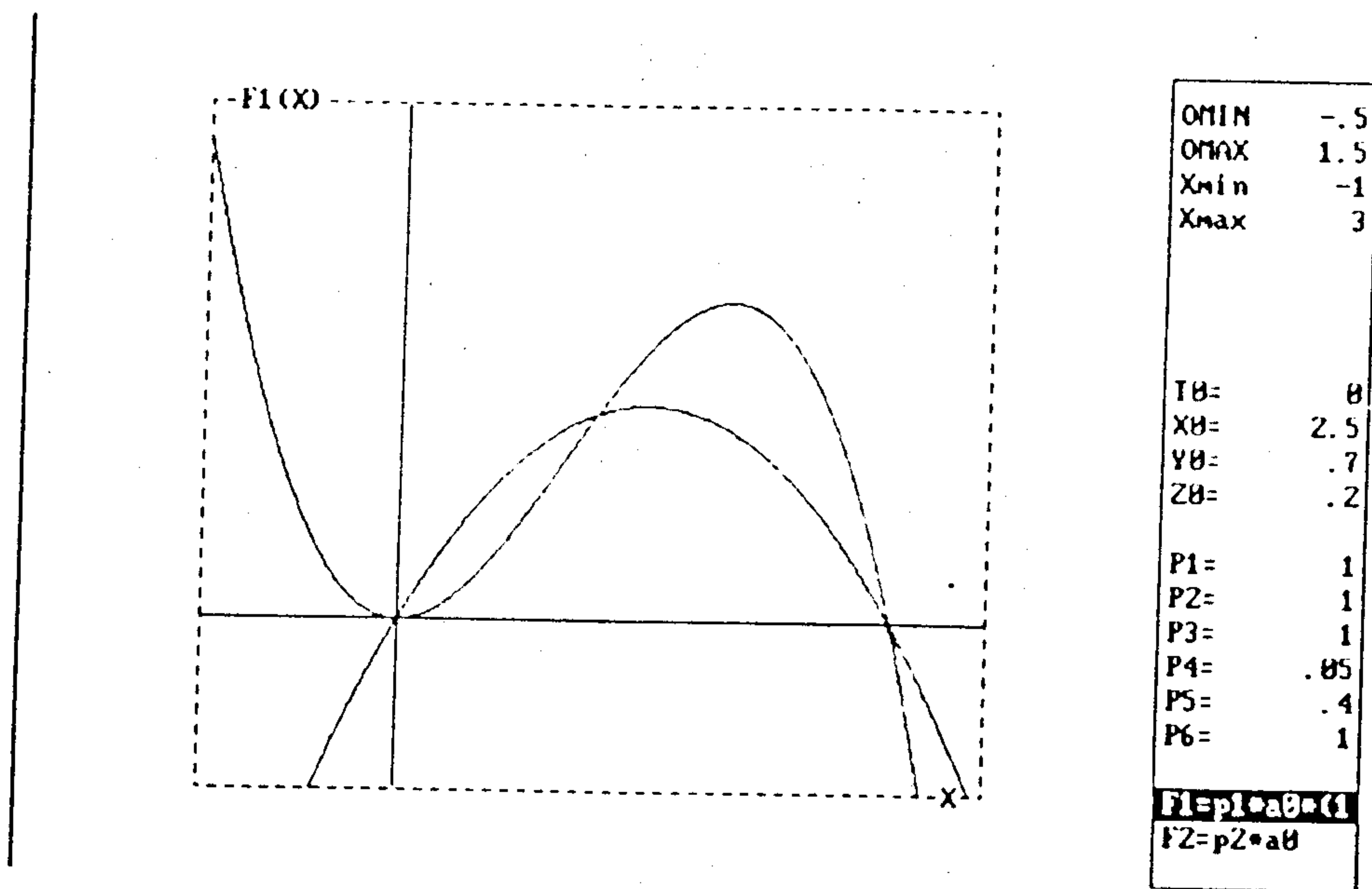


Figure 28b. Plotting the function graph for both functions (8) and (9) together.

Just in case you had some difficulty, all of the previously mentioned experiments with model(6) can be found in the Archives under the level called Bilocal predator-prey model.

Lesson 11. The chaotic dynamics of nonlinear systems

In this lesson, we'll learn how TraX can be used to investigate chaotic dynamics in continuous-time systems. As our first example, we'll use the well-known Lorenz system (Lorenz, 1963; but see also Sparrow, 1982):

$$\begin{aligned}\frac{dx}{dt} &= -\sigma \cdot x + \sigma \cdot y \\ \frac{dy}{dt} &= r \cdot x - y - x \cdot \frac{\bar{z}}{\bar{x}} \\ \frac{dz}{dt} &= x \cdot y - b \cdot z\end{aligned}\tag{10}$$

We'll analyze this system (10) using the parameter values $\sigma = 10$, $r = 28$, and $b = 8/3$, for which a strange attractor exists.

The aim of this lesson is not only illustrate some typical features of chaotic behavior using TraX, but also to present some special features incorporated into TraX such as constructing the next-maximum map and the Poincaré map, calculating the maximum Lyapunov exponent, producing a three-dimensional plot of the attractor and studying homoclinic and heteroclinic trajectories. All of these procedures are based on the computation of one trajectory and therefore, they can be implemented in a special trajectory processing form which operates during trajectory computation. You'll also learn how to write programs in the TraX programming language to process trajectories during their computation.

11.1. Chaotic behavior

Select the differential equations named Lorenz system in the Archives and then select the state Initial state for this lesson. After entering the Investigation mode, display the equations by pressing **Shift-F1**. Note here that the parameters σ , r , b are denoted as $p1$, $p2$, and $p3$, respectively.

Initiate computation and the computed chaotic trajectory will be plotted simultaneously in four windows: $x-y$, $x-z$, $y-z$ and $t-x$ (see Figure 29). You can speed up execution by turning off the indicator of the current point (toggle the Parameter window entry x indicator). Also, the more graphic windows which are in use, the slower the program will perform.

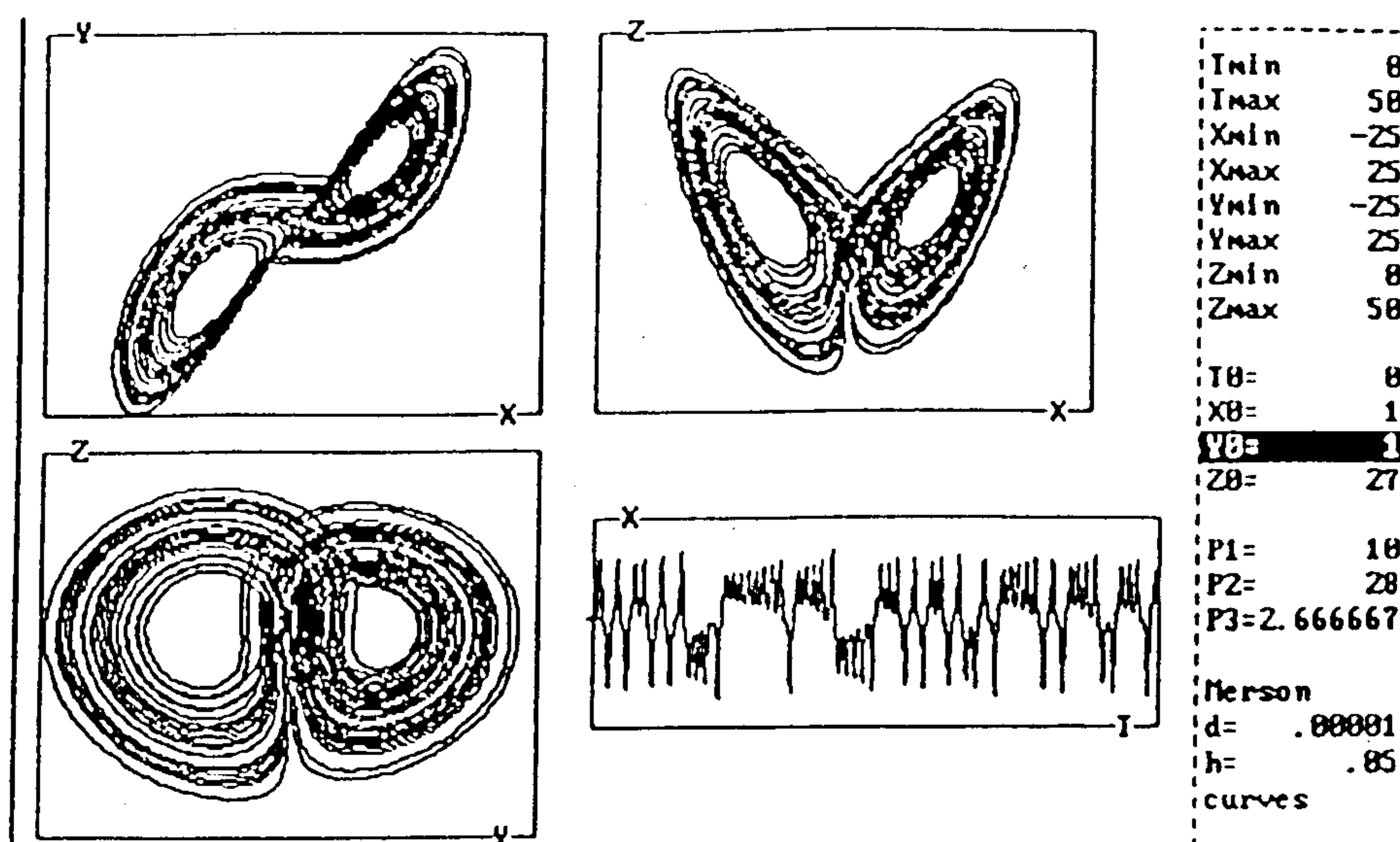


Figure 29. Chaotic behavior of the Lorenz system (10) displayed in three phase projections and one time series plot.

11.2. The sensitive dependence phenomena.

The sensitive dependence of solutions to the initial conditions is the most typical and remarkable feature of chaotic systems. To study this phenomena in the Lorenz system, perturb the initial conditions a little and recompute the trajectory. To distinguish the trajectories with close initial conditions, change the color of the new trajectory before you recompute it (**Alt-F10** allows you to use the color chosen in one window in all of the graphic windows at the same time).

A similar phenomenon may be observed during a study of the influence of the ODE solver type and the tolerance on the chaotic trajectories. Some related results are presented in Figure 30. They correspond to the computation of the same trajectory (*i.e.*, using the same initial conditions) but with different tolerances: $d=.0001$ and $d=.00001$, respectively. You should experiment and perform simulations by varying d from 10^{-3} to 10^{-6} . Also, examine this behavior with the Euler solver. Because the Euler solver has no accuracy control (*i.e.*, the step size remains constant), the only control parameter for this method is the initial step size for integration, h which remains constant throughout integration. The values of h in the interval $[0.005, 0.1]$ are of interest in this case.

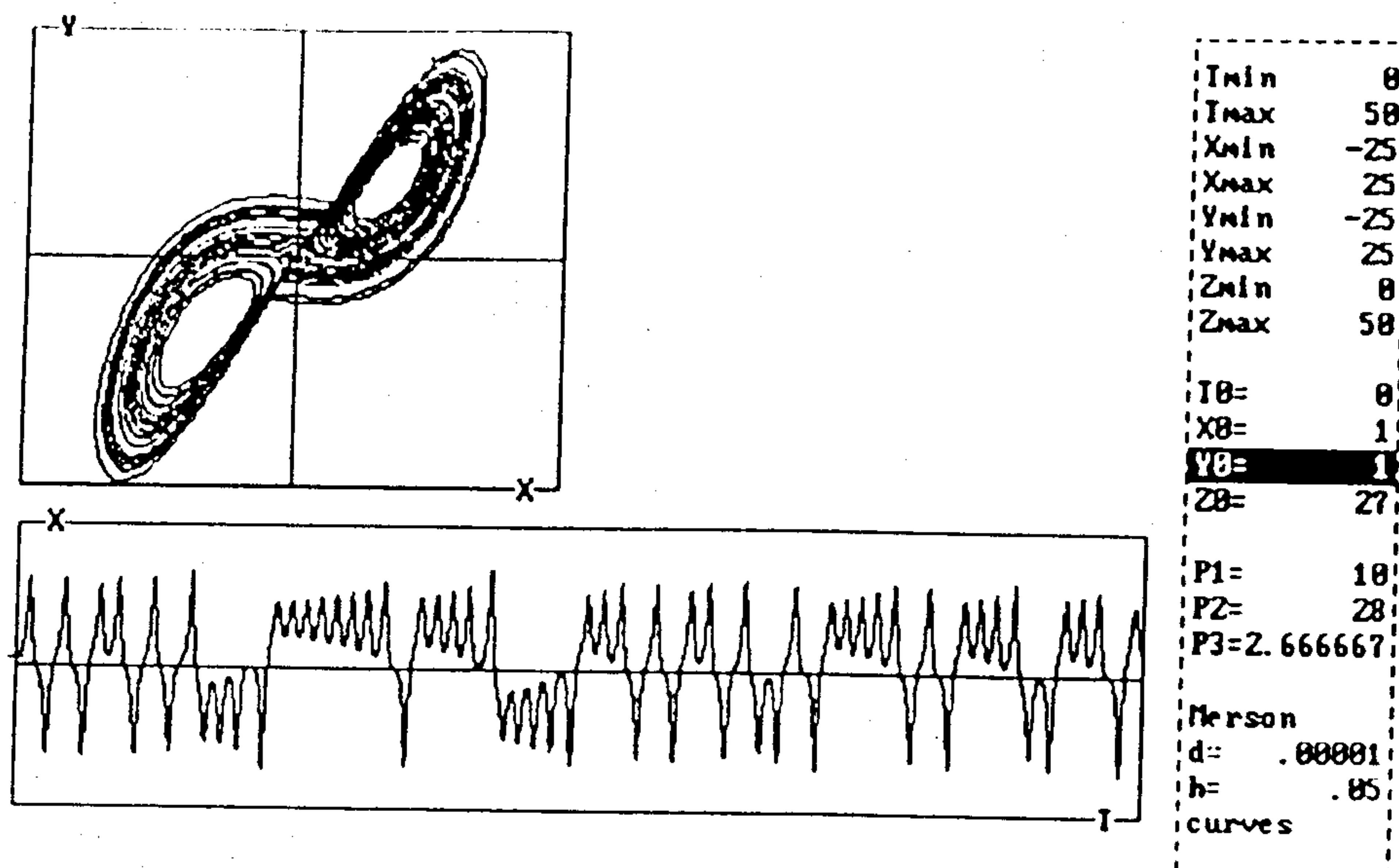


Figure 30. The phenomena of sensitive dependence on initial conditions illustrated by using two different tolerances of the Merson ODE solver: $d=0.00001$ and $d=0.0001$. The two different trajectories which result are both plotted together on each of the two plots.

Notice that the sensitive dependence phenomena is more visible on the graph of x vs. t (Figure 30). It is easy to see on this graph that the numerical solutions corresponding to different tolerances diverge sharply after some transients. This divergence can also be found in phase orbits if you observe the plotting over time. It should be stressed, however, that the resulting attractors for both experiments turn out to be quite similar.

11.3. Homoclinic trajectory.

It is known that some features of chaotic behavior may be better understood if the stable and unstable manifolds of saddle equilibria and limit cycles are studied. Here, we will study (numerically) an unstable separatrix of the saddle point lying in the origin.

For $r > 1$, the equilibrium point $E(0, 0, 0)$ is a saddle equilibrium with one positive and two negative eigenvalues. The positive eigenvalue is

$$\lambda = (-\sigma - 1 + \sqrt{(\sigma - 1)^2 + 4\sigma r}) / 2$$

and the corresponding eigenvector is

$$v = (1, (\sigma + \lambda) / \sigma, 0).$$

By an unstable separatrix we mean a forward time trajectory with initial conditions taken close to the saddle point E on the vector v or $-v$. Unstable separatrices yield an approximation of the unstable manifold of the saddle E if the distance from the initial point to the saddle point is small enough.

The only specific feature of computing an unstable separatrix is that the initial point should be appropriately chosen. We'll choose them as follows:

$$(x_0, y_0, z_0) = (\epsilon, \epsilon \cdot (\sigma + \lambda) / \sigma, 0),$$

where $\epsilon \ll 1$ is a parameter. To provide automatic evaluation of the initial conditions according to above formula, do the following:

(1) Highlight the entry $F_0 = \dots$ and press $\boxed{\leftarrow}$.

(2) Specify the function F_0 as follows:

$F_0 = \{ \text{if } a_0=0 \{ x_0=p_4 \ y_0=p_4 \cdot (p_1-1+\text{sqr}((p_1-1)^2+4 \cdot p_1 \cdot p_2))/2/p_1$
 $z_0=0 \} \} \ 0 \ \boxed{\leftarrow}.$

(3) Set $P_4=0.001$ (P_4 stands for parameter ϵ).

(4) Highlight the entry M_erson and toggle it to $M_\text{erson}+F_0$.

Now you can begin the simulation with arbitrary initial conditions. Immediately after starting, the initial conditions should be changed because of the calling function F_0 . The updated initial conditions will appear in the Parameter window.

Actually, if you're using the solver $M_\text{erson}+F_0$ or $E_\text{uler}+F_0$, the function F_0 is called for processing at the initial point of the trajectory and after each integration step. In this case it is used for calculating the special initial conditions which lie on the unstable eigenvector of the saddle point. Note that the variable a_0 is a formal argument of the function F_0 , and by definition it equals to zero only at the initial point.

Compute the unstable separatrix for various small values of P_4 , either positive or negative, and for various values of the system parameters P_1 , P_2 , and P_3 . Make sure that the function F_0 performs automatic recomputation of the initial conditions for each set of parameter values.

It is interesting to simulate the unstable separatrix to find the homoclinic trajectory when the unstable separatrix approaches the saddle point E as t increases. Since the homoclinic trajectory arises for only certain critical parameter values, you need only to vary the system parameters in order to find this trajectory. Perform several experiments by varying the parameter P_2 ; this parameter corresponds to r in (10).

Figure 31 shows that between the parameter values $P2=13.9265$ and $P2=13.9266$ a homoclinic trajectory exists. Also, note that the state Homoclinic trajectory may be used for these experiments with the unstable separatrix.

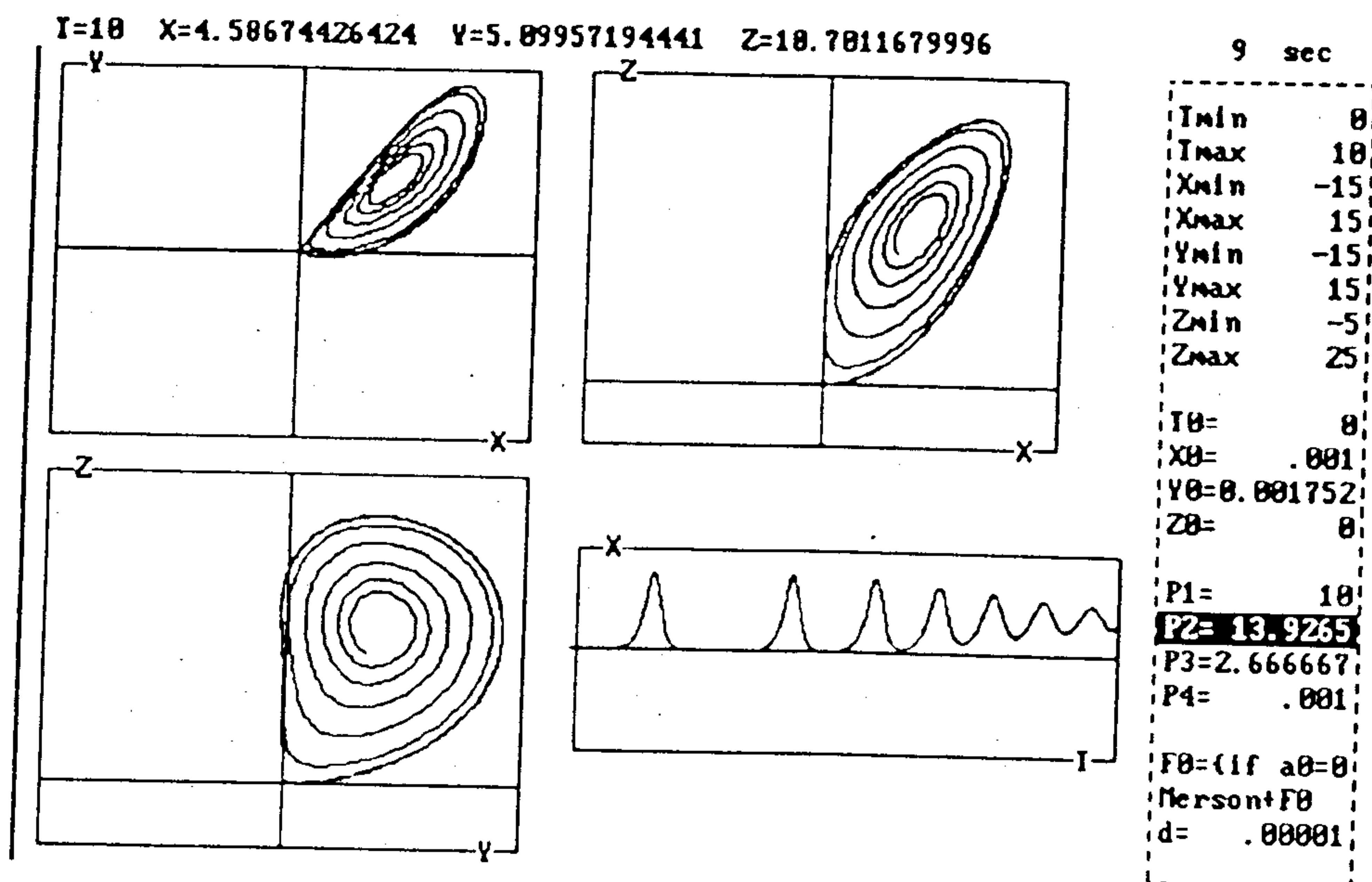


Figure 31a. Simulation of the unstable separatrix in the Lorenz system (10) using $r = 13.9265(P3)$.

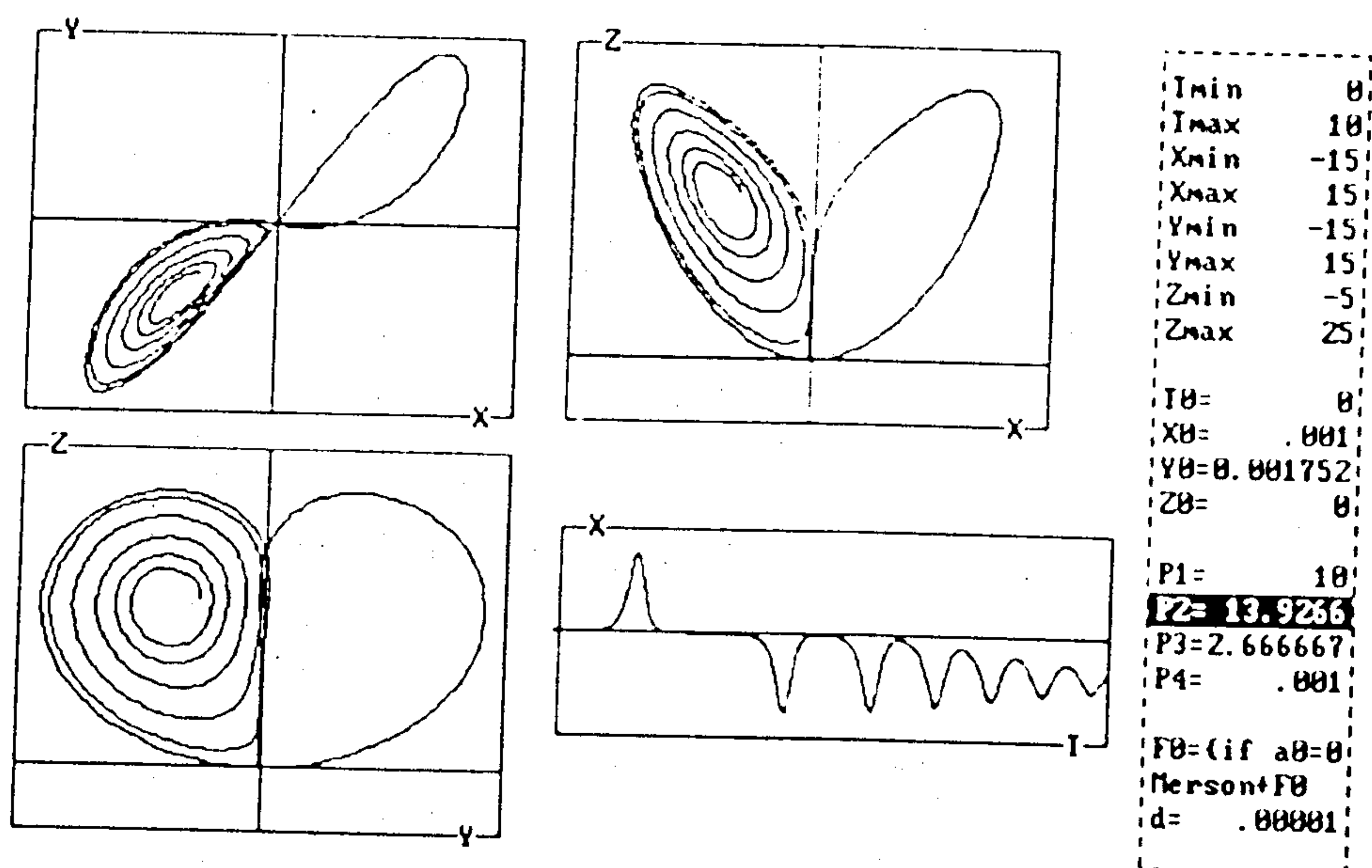


Figure 31b. Simulation of the unstable separatrix in the Lorenz system (10) for $r = 13.9266 (P3)$. This plot shows that between $r = 13.9265$ (Fig. 31a) and $r = 13.9266$ a homoclinic trajectory arises.

11.4. Next-maximum map.

The next-maximum map is a one-dimensional map defined as follows: if $z(n)$ and $z(n+1)$ are two consecutive local maxima of the variable z on a solution of system (10), then $z(n+1)$ is regarded as an image of $z(n)$ by the next-maximum map. To implement this map in TraX, do the following or use the state Next-maximum map located in the Differential equation archive named Lorenz system. Specify the program for function F0 as follows:

```
F0={glo 11,12 loc 10,13,14 if a0=0 {11=0 12=0 13=0 14=0}
10=rhs if 1z<0 {if 14>0 {11=12 12=13-(z-13)/(1z-14)*14}}
13=z 14=1z} 0
```

Next, set the ODE solver to Merson+F0 and the mode to points. Specify a new window with the abscissa named {glo 11}11 and the ordinate named {glo 12}12. Set the limits for this window as AMIN=30, AMAX=45, OMIN=30, and OMAX=50.

Now start the computations and when finished, the plot should appear as shown in Figure 32. You can see that the points in the new window lie approximately on a curve having a cusp. This curve may be regarded as a graph of one-dimensional map modeling some essential features of the Lorenz attractor.

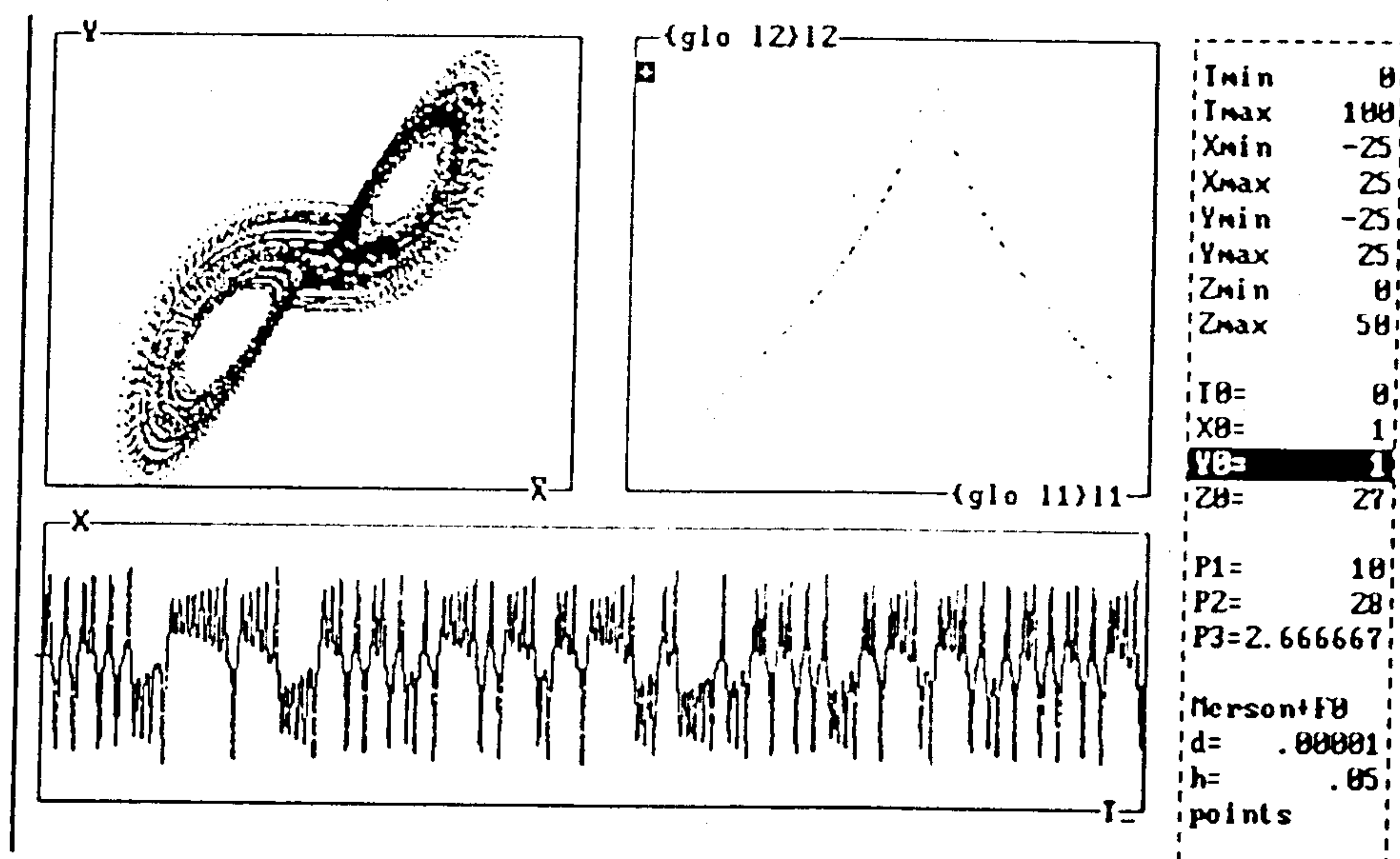


Figure 32. Computation of the next-maximum map for the Lorenz system (10).

There are some particular features about the current program which deserve mention. The variables 11 and 12 are reserved for storing the two consecutive maxima of $z(t)$. These variables are declared as global and their values are computed in F0 and then used as coordinates in the new window, where they also are declared as global.

Variables 10, 13, and 14 in F0 are local and they are used only for evaluating the RHS and storing the values of z and dz/dt at the current point. After the execution of the function RHS, the variable 1z contains a value of dz/dt at the current point. The same goes for 1x and 1y (1x, 1y, ... are the standard global variables, as well as t0, x0, y0, ..., and p0, p1, ...). The condition a0=0 is used to assign the zero values to 11, ..., 14. The rest of the program detects the local maximum of z and locates it by linear interpolation.

In order to get a graph of the next-maximum map for the attractor you'll need to pass the transients before plotting this graph. There are two possible ways to do this: the first is to pass transients, then set a new initial point by pressing **Shift-F10**, clear the windows and begin recomputation, and the second is to use the limits of visibility. These limits are in the Parameter window and their default values are -1D50 and 1D50. The lower limit can reliably be set to 100 but if you do this then increase the upper integration limit so that it is much greater than 100.

11.5. Poincaré map.

Constructing a Poincaré map is performed similar to constructing the next-maximum map. Traditionally, the Poincaré map for the Lorenz system is defined on the cut plane $z = r - 1$, where it maps the intersection of a trajectory with the cut plane to the next intersection in the same direction. The relevant program for F0 has the following form:

```
F0 = {glo 11,12 loc 13,14,15,16 if a0=0 {11=x0 12=y0 13=0
14=0 15=0 16=0} 13=z-p2+1 if 13<0 {if 14>0 {11=15-(x-
15)/(13-14)*14 12=16-(y-16)/(13-14)*14}} 14=13 15=x 16=y} 0
```

In this program the downward intersection points are detected and located. The global variables 11 and 12 represent the x and y-coordinates of the intersection point.

To compute an orbit of the Poincaré map of the underlying chaotic trajectory of Figure 29, select the state Poincaré map (downward intersections with the plane $z=r-1$) and begin the simulation. The plot obtained using this state is shown in Figure 33. Here, the intersections of the trajectory with the cut plane is shown in the top window, and the projections of this trajectory on the planes (x, y) and (x, z) are shown in the bottom two windows. It should be emphasized that computing an orbit of the Poincaré map for the Lorenz system requires a sufficiently large integration interval, otherwise some essential details of the map may be omitted.

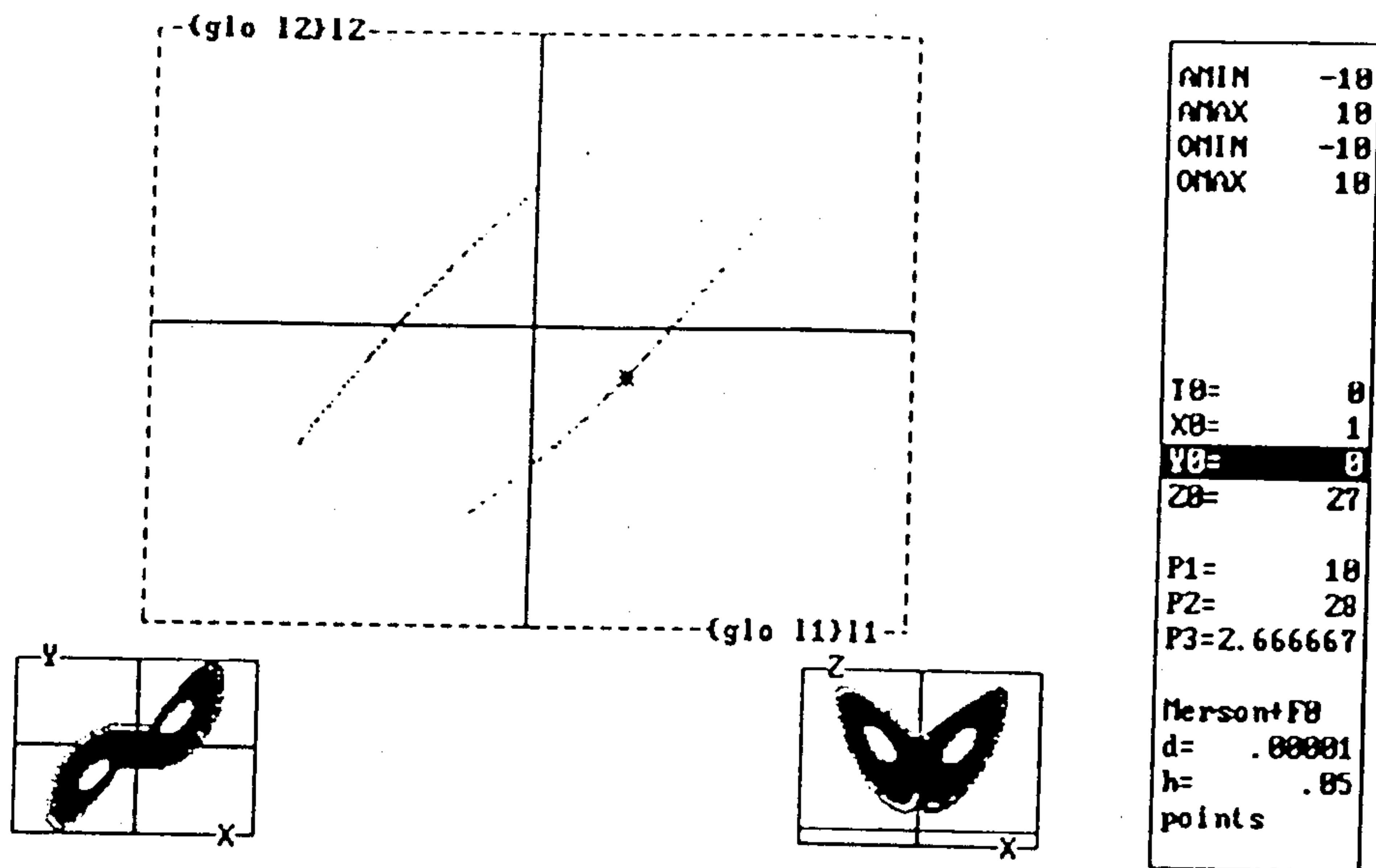


Figure 33. Computation of the Poincaré map for the Lorenz system (10) with two different phase projections of a trajectory shown as well.

For some problems it is necessary to interrupt the simulation after returning to the cut plane. To do this, declare one more local variables, such as 17 in the program above, and insert the operator 17=brk after the operator 12=16-.... The function brk has the same effect as pressing **[Esc]**. The archived state Poincaré map (with an interruption after each downward intersection) refers to this example.

11.6. Three-dimensional plot.

We'll now learn about a method for constructing a three-dimensional plot of an attractor. Consider an orthogonal projection from three-dimensional space with Cartesian coordinates x, y, z and spherical coordinates u, v defined as follows:

$$\begin{aligned} u &= x \cdot \cos(\phi) + y \cdot \sin(\phi) \\ v &= [-x \cdot \sin(\phi) + y \cdot \cos(\phi)] \cdot \cos(\psi) + z \cdot \sin(\psi) \end{aligned} \quad (11)$$

Here ϕ and ψ are the angle coordinates of the point of view of the sphere with the center at the origin, and the plane of representation is orthogonal to a vector connecting the origin and the point of view.

The simplest way to implement formula (11) in TraX is to use the functions F1 and F2 as coordinates of a window, specifying them as follows:

$$\begin{aligned} F1 &= x \cdot \cos(p4) + y \cdot \sin(p4) \\ F2 &= (-x \cdot \sin(p4) + y \cdot \cos(p4)) \cdot \cos(p5) + z \cdot \sin(p5) \end{aligned}$$

where p_4 and p_5 will be the new parameters. After these functions are specified, create a window with abscissa F_1 and ordinate F_2 , choose the appropriate values of parameters P_4 and P_5 , set the corresponding limits for the window, and simulate the system. This approach is supplied by a special procedure for drawing 3-D axes and is implemented in the state 3-D plot. Select this state and begin computation. The resulting graph is shown in Figure 34a. Here the parameters P_4 and P_5 correspond to the angles ϕ and ψ which are measured in degrees, with $P_4=110$ and $P_5=60$. Another projection with $P_4=130$, and $P_5=60$ is shown in Figure 34b. This corresponds to a rotation of the attractor shown in Figure 34a around the z -axis.

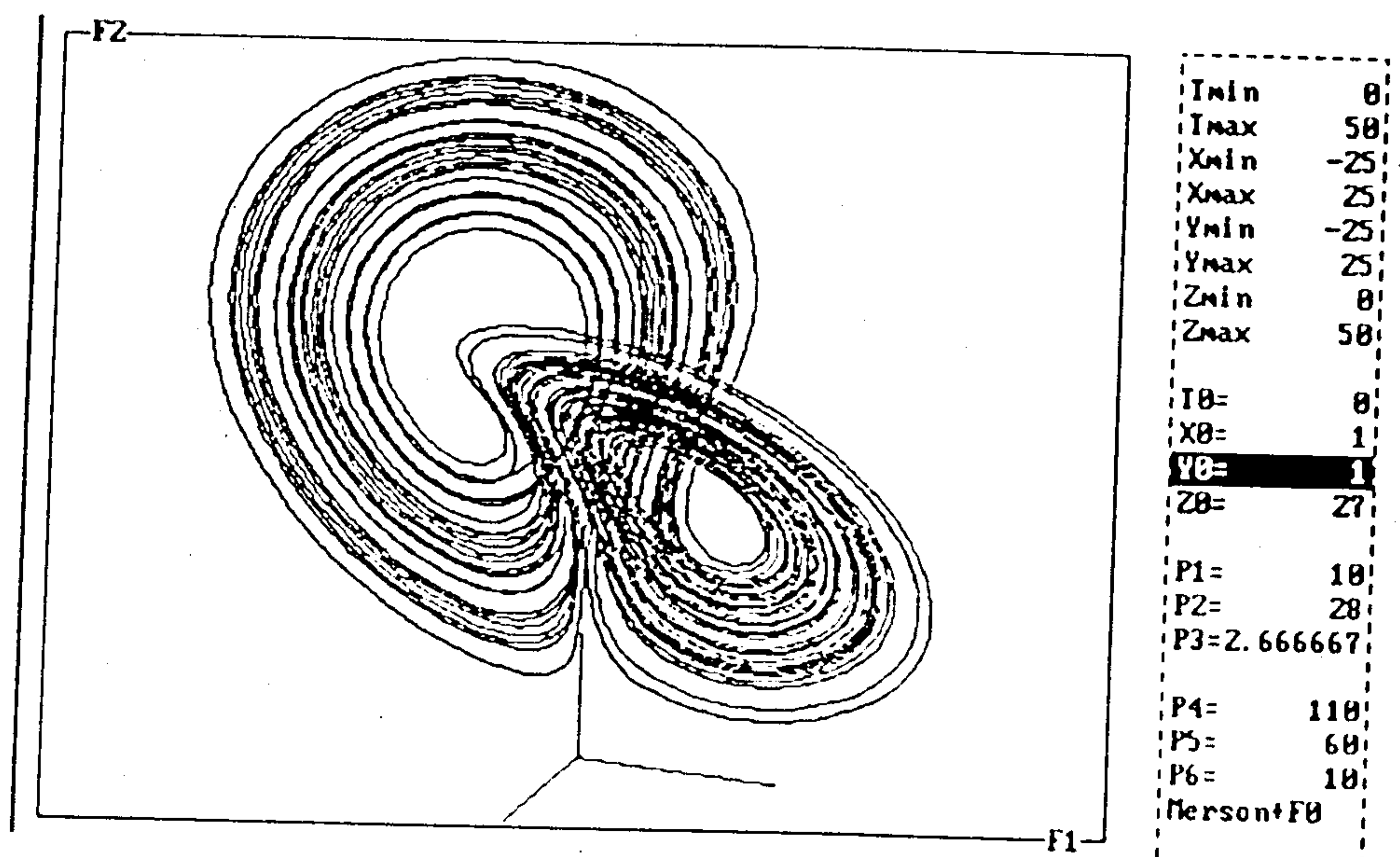


Figure 34a. 3-D plot of the Lorenz attractor.

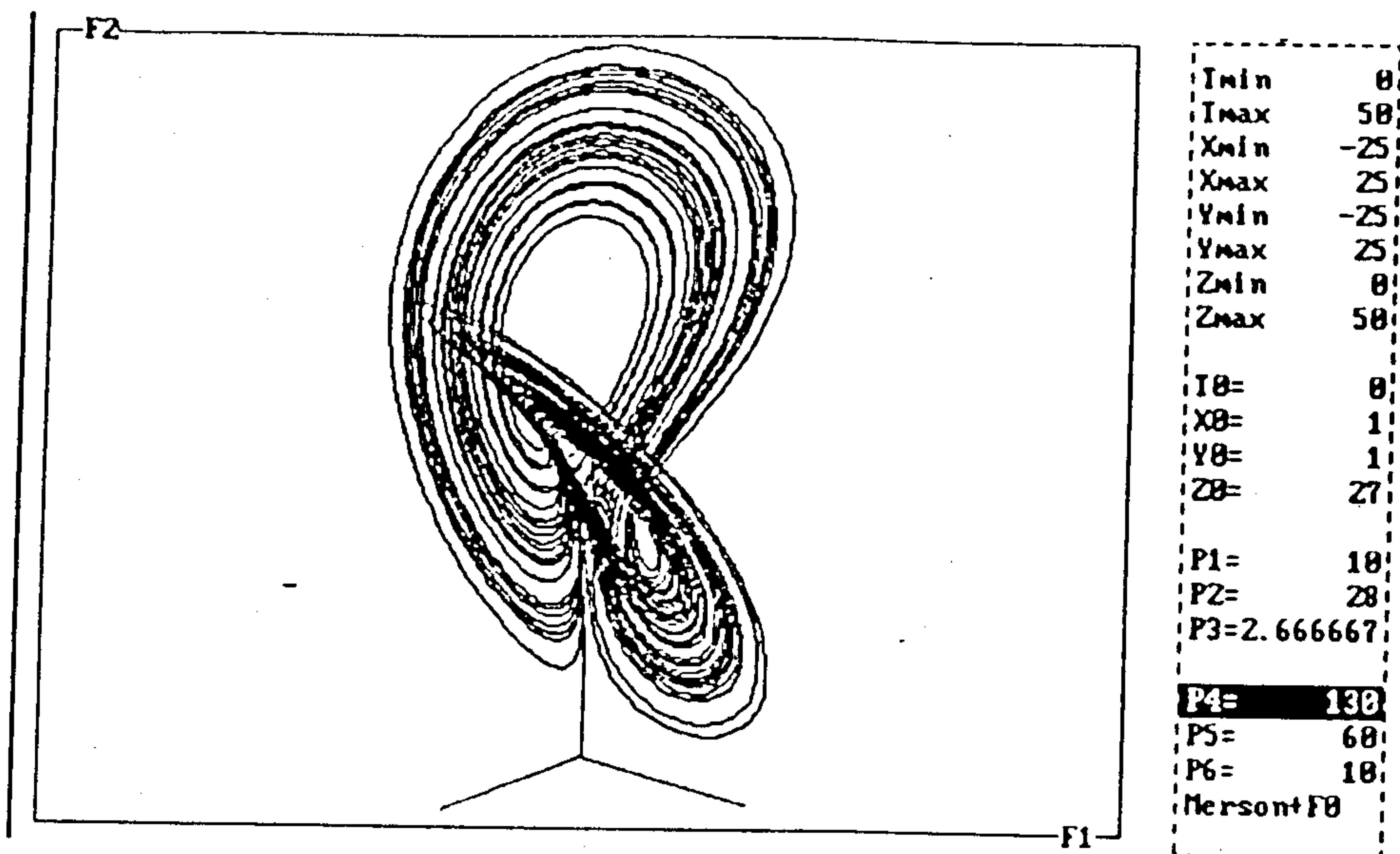


Figure 34b. Same as Figure 34a but the viewing angle has been rotated clockwise around the z-axis by 20 degrees (compare P4 in both Figure 34a and b).

To display the functions used in implementing the 3-D plot, press **Shift-F1** (see Figure 35). In this model, functions F1 and F2 specify the linear transformation of (11) with the coefficients denoted as 11, 12, 13, and 14. Function F3 gives the values of these coefficients and is called from F0 when an initialization is performed. Function F4 furnishes a way to draw axes and it is also called from F0 in the operator 11=F4.. The parameter P6 sets the length of the axes; if you delete this operator no axes will be drawn. The procedure of drawing axes is based upon the capability of being able to call F0 multiple times and change the plotting attributes, which is coded by a value of F0, each time. In general, with this approach you'll often have to experiment to choose the limits of the F1-F2 window appropriately.


```

Press any key
dX/dt=-p1*(x-y)
dY/dt=p2*x-y-x*z
dZ/dt=x*y-p3*z
F0(a0)= {glo 10 loc 11 if a0=0 (10=0 11=f3) else (11=f4 10=10+1)
        } 11
F1= {glo 12,13} 12*x+13*y
F2= {glo 12,13,14,15} 14*(-13*x+12*y)+15*z
F3= {glo 12,13,14,15 loc 11 11=atan(1)/45 12=cos(11*p4) 13=sin(11*p4) 14=cos(11*p5) 15=sin(11*p5)} 11
F4= {glo 10 loc 11 11=0 if 10<=6 {x=0 y=0 z=0 11=1 if 10=0 11=3 if 10=1 x=p6 if 10=3 y=p6 if 10=5 z=p6 if 10=6 {x=x0 y=y0 z=z0 11=3}} 11

```

Figure 35. The functions used to draw the 3-D axes and plot the trajectory in Figure 34.

11.7. Lyapunov exponent.

The last problem to be discussed in this lesson is the calculation of the maximum Lyapunov characteristic exponent (LCE) for the Lorenz system.

Select the state Lyapunov exponent of the equations named Lorenz system with variational equation. Begin computation and you will see (Figure 36) a time evolution of three variables: x (top window), the norm of the variational equation solution $\sqrt{(u^2 + v^2 + w^2)}$ (middle window), and the variable 13 (bottom window). The limit value of this variable as time tends toward infinity gives the maximum LCE, therefore, we'll call 13 the maximum LCE.

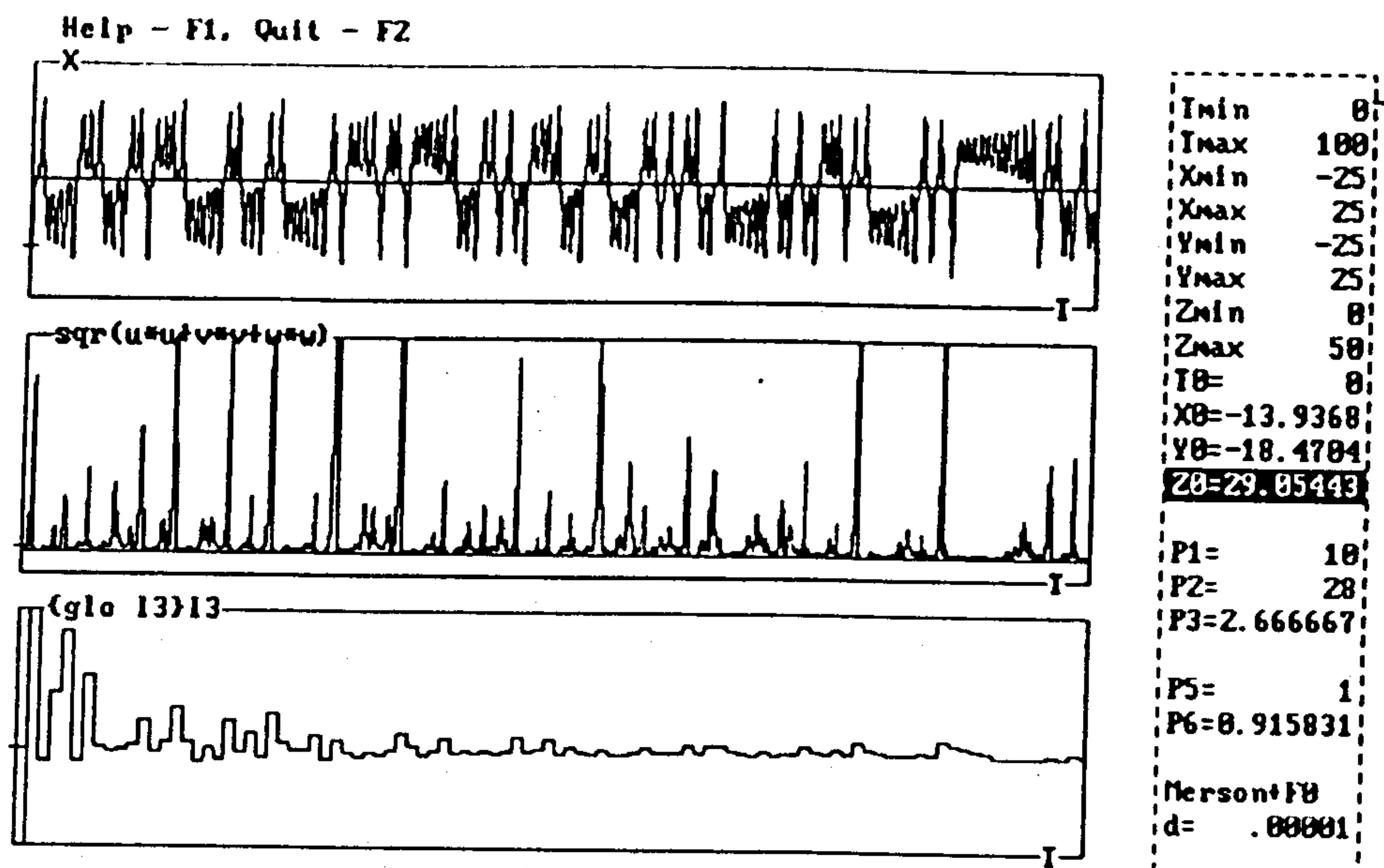


Figure 36. The computation of the maximum Lyapunov characteristic exponent. *Top window* - the time dependence of x ; *middle window* - a norm of the solution of the variational equation versus t ; *bottom window* - the evolution in time of variable 13 which gives an approximation of the Lyapunov exponent as t tends toward infinity. The scale for the norm is $[-5, 50]$ and the scale for 13 is $[0.5, 1.5]$.

Look at the parameter P6 in the Parameter window. It displays the current value of 13. The use of a parameter for output is a useful way to display the values of some variables during integration.

Figure 36 shows the small-amplitude oscillations of 13 near a value 0.9, and this may be regarded as the approximation of the maximum Lyapunov exponent for the given system's parameters. The algorithm used for computing the maximum LCE (see Parker and Chew^{ua}, 1989) is based on integration of system (10) with variational equations. Since the solution of the variational equations displays unlimited growth, it is renormalized after each time interval τ . We consider it a parameter of the algorithm and denote it by P5. This parameter needs an appropriate choice; its value shouldn't be too small or too large. Notice that it is the renormalization which causes the "jumps" on the plots of the norm and 13 versus t in Figure 36. In Figure 37 the integrated 6th-order system and the function F0 which provides the calculation of the maximum LCE are shown. Note that the computation of the maximum LCE is a time-consuming procedure because a 6-dimensional system is being integrated over a large time interval.

```

Press any key

dX/dt=-p1*(x-y)
dY/dt=p2*x-y-x*z
dZ/dt=x*y-p3*z
dU/dt=-p1*(u-v)
dV/dt=(p2-z)*u-v-x*u
dW/dt=y*u+x*v-p3*u
F0(a0)= {glo 10,11,12,13,14 if a0=0 {10=0 11=t 13=0 u=1 v=0 w=0}
        else {if t-11>p5 {12=sqr(u^2+v^2+w^2) 10=10+log(12) 13=
        10/t u=u/12 v=v/12 w=w/12 11=t p6=13)))0

```

Figure 37. The specification of the Lorenz system with the variational equation and the function F0. This function is for calculating the value 13 which is regarded as approximation of the Lyapunov exponent.

Lesson 12: Phase portraits, bifurcation diagrams and keystroke macros

In this lesson you'll learn some special TraX procedures for conducting iterated processes; we'll call this procedure the *keystroke macro*. We'll demonstrate the use of the keystroke macro through two examples. First we'll construct a phase portrait of a two-dimensional differential equation system and second, we'll compute the bifurcation diagram for a one-dimensional map. In both problems the use of a macro will substitute for tediously repeating certain actions. In this manner the keystroke macro becomes a tool for storing a sequence of keystrokes and then replaying these keystrokes by pressing a single **[Alt]**-digit key combination.

12.1. Phase portrait

We'll begin with a predator-prey model (1). In the Archives find the differential equations Predator-prey model. Next, select the state Phase portrait made by a keystroke macro. This state is similar to that previously used in Lesson 2. Using the keystroke macro, we will compute the series of trajectories which have the initial points lying on the line $X = 1.1$. Draw the axes and then press **[Ctrl-F4]** to initiate the macro creation. This macro will contain three operations, these are:

- (1) press **[F10]** to begin computation;
- (2) press **[↓]** to move the initial point down;
- (3) press **[→]** to move the initial point in the same direction using "large" step.

To complete the macro definition, press **[Alt-1]**. Following this, the three actions (1-3) now become a new compound operation which is available by pressing **[Alt-1]**. Press **[Alt-1]** several times and the new trajectory should appear after you execute this macro. The resulting phase portrait is shown in Figure 6. Note, however, that you cannot embed macros within each other; the only way to repeat a macro is to invoke it again by pressing the key combination again.

Next, you can playback the macro with another value of X_0 or (which may be more interesting) with another value of the parameter P_2 . For example, set $P_2 = 0.57$ and $Y_0 = 0.7$, clear the windows, re-draw the axes and press the **[Alt-1]** key several times. You will see a number of trajectories appear which each correspond to a new parameter value. Make sure, however, that a stable periodic orbit (*i.e.*, limit cycle) appears. You must be careful when using a keystroke macro; the starting position when initiating the macro should be the same or very similar to that which was used when the macro was defined. That's why we set the value of Y_0 prior to starting the last experiment.

There is another way to accomplish the same results. Place the IPI in the right upper corner, clear the window and re-draw axes. Now, highlight the entry $Y0=...$ and press **Ctrl-F4** to begin the following macro-definition:

- (1) press **F10** to begin computation;
- (2) type - - 0.05.

Complete this macro definition by pressing **Alt-2**, which is the key combination for invoking this macro. You might be wondering just what - - 0.05 means in this macro. This instruction tells TraX to decrease the value of $Y0$ by 0.5 each time the macro is run (check the values in the Parameter window to be convinced of this). In general, if you highlight an entry containing a variable and then enter - - a or + + a, the highlighted variable will be decreased or increased, respectively, by the value a. Now, if you invoke the **Alt-2** macro several times, the y-coordinate of the initial point will be decreased by 0.05, and for each initial point the trajectory will be recomputed. The result will be similar to that shown in Figure 6 (with $P2=0.65$). Note that if you highlight the entry $X0=...$ and run the **Alt-2** macro, the value of $X0$ will change by the specified increment.

12.2. Bifurcation diagram

Next let's consider the map (4). We want to construct the so-called bifurcation diagram for this map which will illustrate the dependence of an asymptotic mode upon a control parameter. Find the difference equation Logistic map $x'=p1*x*(1-x)$ in the Archives and select the state Bifurcation diagram. After entering the Investigation mode you will see graphic window with the ordinate x and the abscissa $p1$ (note that here, $p1$ is equivalent to parameter a in (4)). You should be aware that a system parameter may be used as the label of a window's axis, as a phase variable, time, or as a user-defined function.

Now, activate the $p1-x$ window and look at the limits. They should be: $AMIN=2.7$, $AMAX=4.02$, $Xmin=0$, $Xmax=1$. We'll define a new macro and name it **Alt-3**. The reason we're using a new name (*i.e.*, key combination) is so that the previously defined macro will not be redefined or overwritten². Highlight the entry $P1=2.7$ and enter **Ctrl-F4** to initiate the new macro-definition as follows:

- 1) enter ++0.01 (increments the parameter value by 0.01);
- 2) enter **F10** (computes the trajectory);
- 3) enter **Shift-F10** (to set a new initial point by placing it on the trajectory's most recent point).

²This suggests how to erase a macro definition, *i.e.*, write a null macro over a previously defined macro.

Complete the macro-definition by pressing **[Alt-3]** and then run this macro several times. The resulting bifurcation diagram is shown in Figure 38. Note, however, that the transients are not visualized on the bifurcation diagram because by setting 100 as the point at which plotting begins (in the Parameter window), TraX begins plotting from $n = 100$. Also, the maximum value of n is $Nmax = 1000$. You can create a more detailed bifurcation diagram if you desire by incrementing the parameter by a smaller value (this will, of course, require more computational time to complete the diagram over the same limits). Unfortunately, a macro-definition can't be edited so you'll have to redefine the entire sequence of keystrokes.

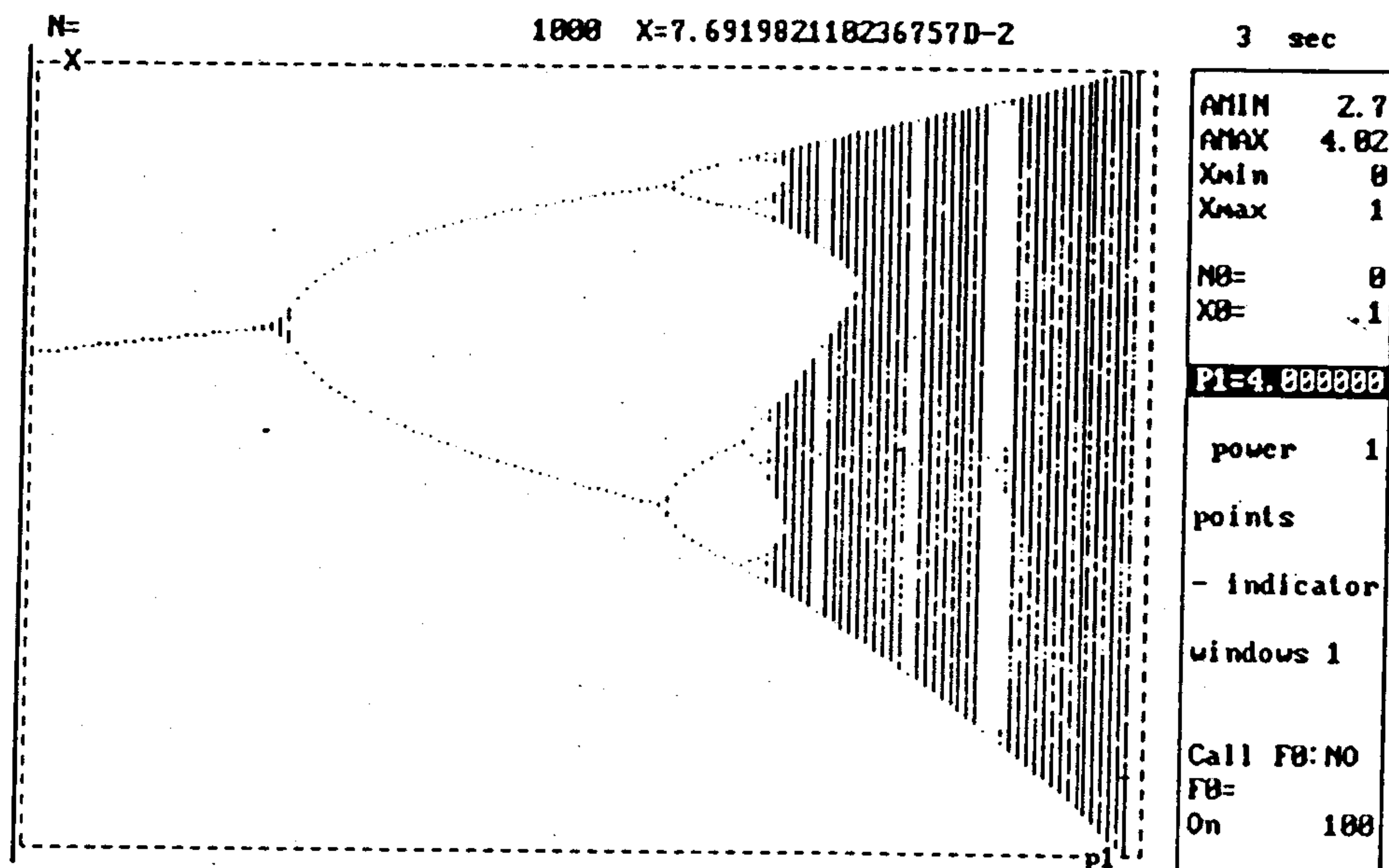


Figure 38. The bifurcation diagram of the logistic map (4) computed by a keystroke macro. The parameter $P1$ is varied from 2.7 to 4.0 by increments of 0.01.

Now, let's consider the same map but with additive noise:

$$x' = \alpha x(1 - x) + e\zeta \quad (12)$$

where ζ is gaussian random noise and e is a parameter. It is interesting to construct a bifurcation diagram for this case and compare it with unperturbed one. To implement this, specify a function $F0$ as follows:

$$F0 = \{x = x + p2 * gau\} 0$$

Here gau is a built-in TraX function which computes a normally distributed pseudo-random value with a mean of zero and a standard deviation of one. The function $F0$ specifies gaussian noise which acts additively on each point of a trajectory during its computation. The parameter $p2$ can be regarded as the amplitude of the random perturbation of a trajectory. Usually, however, the function $F0$ is not called

during computation. To activate the processing of a trajectory by the program specified in the function F0, find the Parameter window entry CALL F0:NO and toggle it to CALL F0:YES. This means the function F0 will be called after each iteration of a map (this option is similar to that of the ODE solvers called Merson+F0 and Euler+F0). Next, set P1=2.7 and P2=0.002, clear the window and playback the macro by pressing **Alt-F3** several times to develop a bifurcation diagram of a randomly perturbed map (12) (don't forget about highlighting the entry P1=... before initiating the macro). Figure 39 shows a bifurcation diagram of randomly perturbed map. Notice that the influence of noise is more significant near the bifurcation values of the parameter. In addition, the periodic orbits which have large periods appears to be more sensitive to the addition of noise. Thus chaotic behavior can arise earlier (*i.e.* for smaller parameter values) than for an unperturbed map.

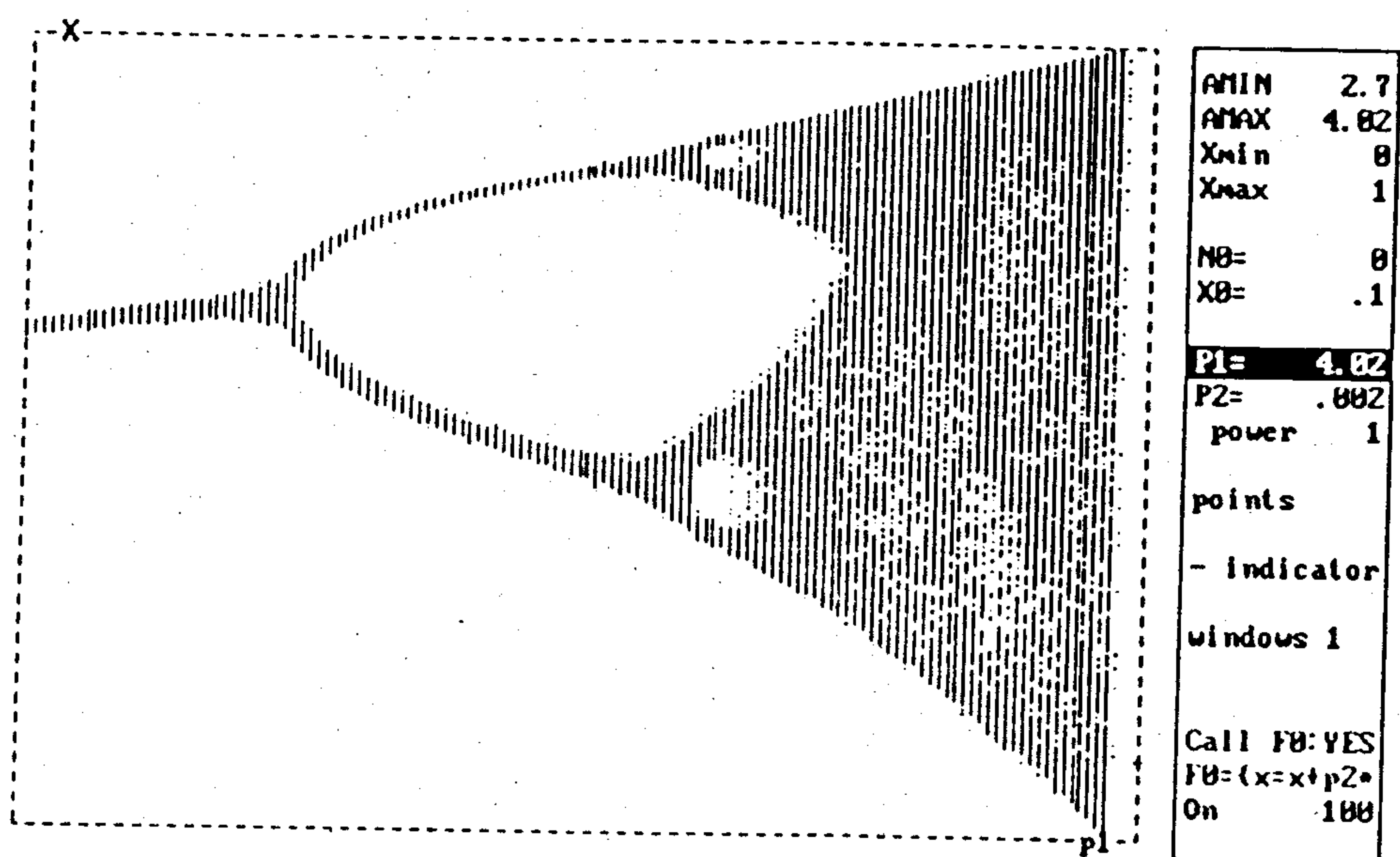


Figure 39. The bifurcation diagram of the logistic map perturbed by gaussian noise (compare with Figure 38).

Lesson 13. Stochastic modeling in TraX

In this final lesson we will study the influence of additive random noise on a dynamical system's behavior and demonstrate how TraX can be used to investigate this kind of problem. We will use as an example a purely stochastic model of neuronal activity.

Consider the model of the so-called *nonformal neuron* (Kryukov 1978). Suppose there are two processes in time which govern the generation of nerve impulses or spikes. The first process is a membrane action potential function and the second one is a threshold of excitation. When the membrane action potential reaches the threshold, the spike is generated. Following the spike, the membrane potential and the threshold both return to their initial values, and both processes start again. It is assumed that the membrane potential and the threshold decay exponentially at each step of the process. And in addition, at each step, *noise* (a normally distributed random number) is added to the membrane potential.

Let us denote the membrane potential by x , and the threshold value by y . Let z be a (boolean) variable which indicates if there is a spike ($z = 1$) or not ($z = 0$). The discrete-time neuron model, therefore, may be described as follows:

If $x(n) \leq y(n)$ then:

$$\begin{aligned} x(n+1) &= F1(x(n)), \\ y(n+1) &= F2(y(n)), \\ z(n+1) &= 0, \\ u(n+1) &= u(n), \\ v(n+1) &= 0, \\ w(n+1) &= w(n). \end{aligned} \tag{15a}$$

Otherwise, if $x(n) > y(n)$, then:

$$\begin{aligned} x(n+1) &= x(0), \\ y(n+1) &= y(0), \\ z(n+1) &= 1, \\ u(n+1) &= n+1, \\ v(n+1) &= u(n+1) - u(n), \\ w(n+1) &= w(n) + 1. \end{aligned} \tag{15b}$$

As you can see, we have introduced three additional variables u, v, w : u is the moment of the last nerve excitation, v is the interspike interval, and w is the ordering number of an action potential or spike. The initial conditions are assumed to be $x(0) = 0$, $y(0) = 1$, $z(0) = 0$, $u(0) = 0$, $v(0) = 0$, $w(0) = 0$. Functions $F1$ and $F2$ are defined as follows:

$$\begin{aligned} F1(x) &= \alpha x + b + c\zeta, \\ F2(y) &= d(y - e) + e, \end{aligned} \tag{16}$$

where ζ is a normally distributed variable with a mean equal to zero and a standard deviation equal to one. Here, α, b, c, d and e are the model parameters, under the restrictions $0 < (\alpha, d, e) < 1$, and $(b, c) > 0$. The first term in $F1$ corresponds to the decay of the potential ($\alpha < 1$), and the others describe the noise component. The function $F2$ implies y decays to e with rate d .

The formulas (15a) describe the process of neuronal activity until the next spike will be generated. As soon as the spike is generated, then, according to formulas (15b), the membrane potential x and the threshold y are reset to their initial values, and the new values of variables u, v and w which completely describe the spike activity are calculated.

The random nature of the membrane potential in the model implies that the time interval v between the spikes is a random value. The most important problem with the model concerns the stochastic properties of the time series of interspike intervals. One of the hypotheses is that the process of the spike generating is close to Poisson with an exponentially decaying probability density distribution. We will analyze this property here by calculating a histogram of interspike intervals. Using the model and the TraX facilities one can study other questions on impulse generation such as the influence of periodic forcing on the process, etc.

Now let's experiment with the model. Find this model in the TraX Archives among difference equations. It is named *Stochastic model of a neuron*. The model was specified in TraX in the form shown in Figure 40. The parameters $P0, P1, P2, P3$, and $P4$ correspond to α, b, c, d and e in (16), respectively. Function $F0$ is considered here as a part of the model specification. It is called after each evaluation of the RHS to correct values of x, y, u, v and w in the case of spike generation, as described in (15b). Besides, the function $F0$ stores a sequence of interspike intervals in a global array $g0$ for statistical analysis which will be done after simulation.


```

X'=f1
Y'=f2
Z'=(loc 10 if x>y 10-1 else 10-0)10
U'=u
V'=0
W'=w
F0(a0)- (gle 11,g0[250] loc 10 if a0=0 (x0=0 y0=1 z0=0 u0=0 v0=0
w0=0 11-w0) if z=1 (x=x0 y=y0 v=a-a u=a w=w+1 11-w g0[1
1]=v) if 11=250 10-brk)0
F1= p0*x+p1+p2*gau
F2= p3*(y-p4)*p4
F3= (gle 11,12,g0[250] loc 10,16 10-0 12-0 while 10<11 {12=1
2*g0[10] 10=10+1} 12=12/11)12
F4= (gle 11,12,13,g0[250] loc 10 10-0 13-0 while 10<11 {13=1
3*(g0[10]-12)^2 10=10+1} 13=sqr(13/(11-1)) } 13
F5= (gle 11,14,g0[250],gl[20] loc 10,16 10-0 while 10<p7 {gl
[10]=0 10=10+1} 14=p7/(p6-p5) 10-0 while 10<11 {16=int((
g0[10]-p5))*14 if 16*(16-p7+1)<=0 gl[16]=gl[16]+1 10=10+
1} 10-0 while 10<p7 {gl[10]=gl[10]/11 10=10+1})0
F6(a0)= (gle 11,14,gl[20] loc 10,16 16=int((a0-p5)*14) if 16*(16
-p7+1)<=0 10=gl[16] else 10-0)10

```

Press any key

Figure 40. The specification of the stochastic model of the nonformal neuron (15) and (16), using procedure-functions for statistical analysis.

The functions F3, F4, F5, and F6 are not directly related to the model description. Functions F3 and F4 are for calculating the mean and standard deviation, respectively, for series of interspike intervals. Function F5 counts the number of interspike intervals. The parameters P5 and P6 give the boundaries of the interval on which the histogram will be built, and parameter P7 gives a number of bins in this interval. The last function, F6, presents the histogram as a function of the interspike interval. All the functions described in this paragraph will be run after the model is simulated using the options Compute and **F6** (plot a function graph).

Select the state Spike generation and histogram computation. After entering the Investigation mode you can immediately start simulation. Figure 41 shows the time evolution of x, y, and z. In the top window there are two curves: the upper curve corresponds to the threshold of excitation, and the lower one shows the membrane potential. On the bottom of the screen you can see the time series of generated spikes.

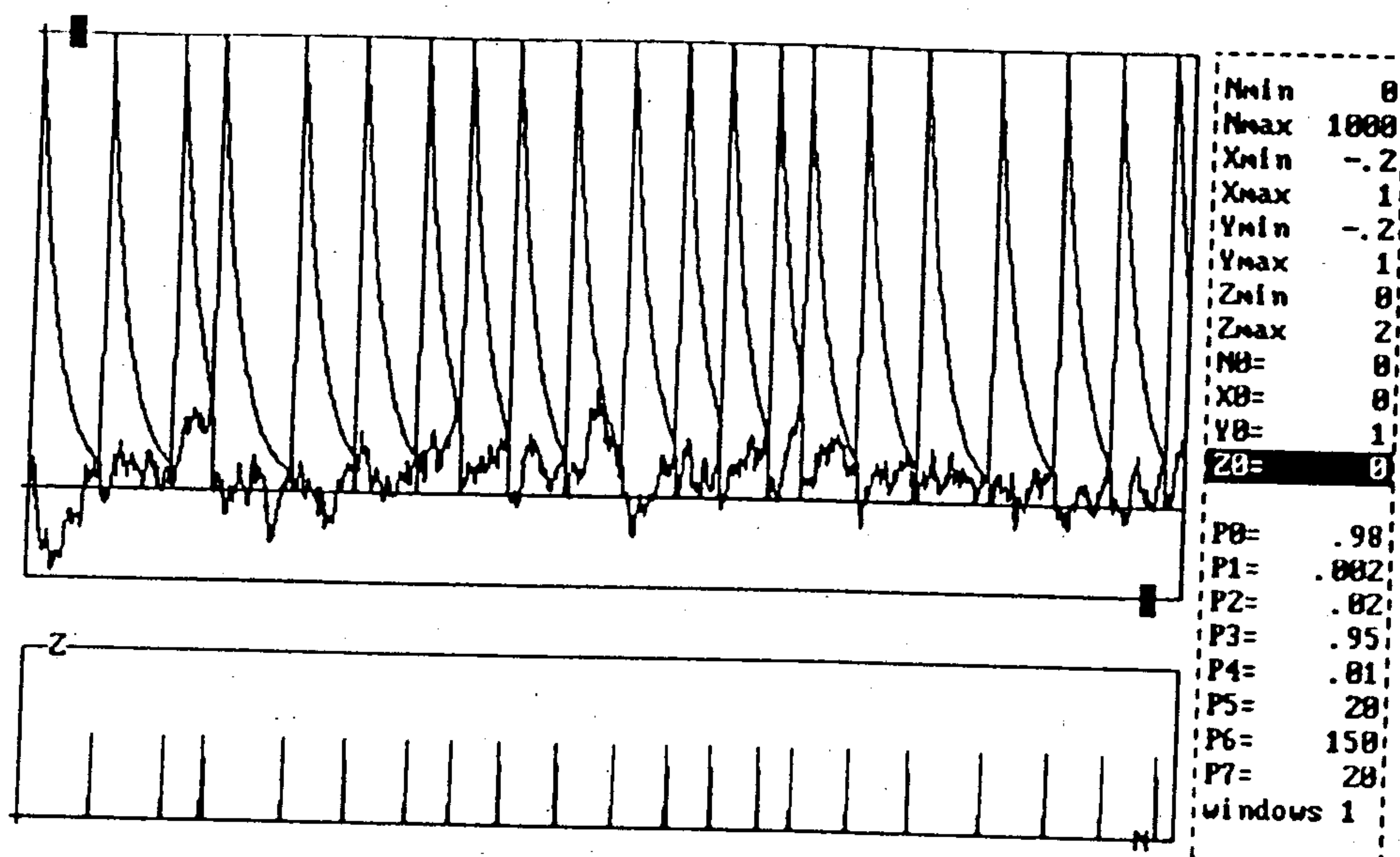


Figure 41. Simulation of the neuronal model (15) and (16). In the top window the threshold and membrane potential functions are plotted. In the bottom window, the series of generated spikes or action potentials are shown.

Notice that there are two overlapping windows in the top of the screen: $t-x$ and $t-y$ (you can check this by activating each window). Both of these windows have the same limits which makes the observation of these processes more convenient. It should be emphasized that your results of the model simulation will not be exactly the same as those shown here because of the stochastic nature of this model. Next we will analyze the statistical properties of the spike generating process.

In order to compute and plot the histogram of interspike intervals, we need to simulate the model on a time interval which is long enough. For the given parameter values, $P_0=0.98$, $P_1=0.002$, $P_2=0.02$, $P_3=0.95$ and $P_4=0.1$, 15000 iterations should be appropriate because this gives approximately 250 spikes. You can see in the description of the function F0 that due to the operator, if $11=250$ $10=brk$, the simulation will be interrupted after reaching 250 spikes (11 is a number of spikes and brk is a standard function with an action equivalent to **Esc**). The limiting number of 250 is exactly the number of elements of the g_0 array in which the interspike intervals are stored.

Now, make the Parameter window active and set $N_{max}=15000$. Start the simulation and after it's finished, initiate the option Compute (press **?**) and calculate the value of the function F3 (type F3), which is a mean value. Then calculate the value of F4, the standard deviation in a similar fashion. In our experiments we found the mean close to 60, and the standard deviation close to 25. Then the function F5 should be computed in the same way (i.e., press **?** and after the prompt Compute:, type F5).

The specific feature of the function F5 is that after it runs the result which appears on the screen (that is zero value) has no special meaning. The main result of function F5 is the computation of a histogram.

To plot the histogram, change the window grouping (*i.e.*, toggle to windows 2). You will see one graphic window with abscissa v and ordinate $F6(v)$. Activate this window (**F5**) and then press **F6** to plot the graph of the function denoted on the ordinate; a histogram similar to that in Figure 42 should be plotted. This confirms the previously mentioned hypothesis that the probability density of interspike intervals is exponentially decaying, but certainly some additional experiments should be done.

For developing a histogram we have chosen an interval ranging from 20 through 150, and the number of bins equals 20. You can set some other values of corresponding parameters P5, P6, and P7, and then compute (using function F5) and re-plot the histogram as before.

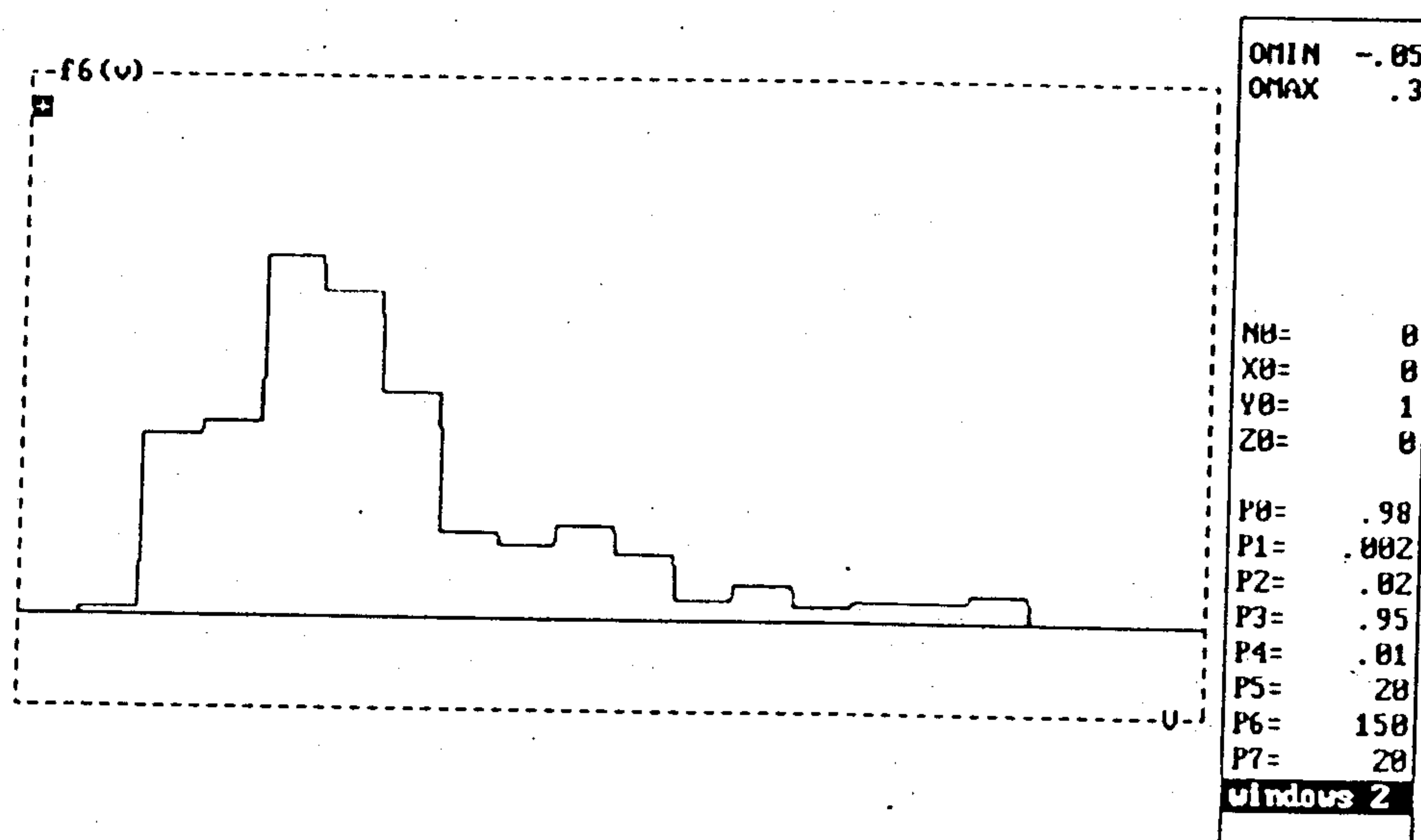


Figure 42. Histogram of interspike time intervals for the neuronal model (15) and (16).

References Cited

- Bazykin, A. D. 1985. Mathematical Biophysics of Interacting Populations. Nauka, Moscow (*in Russian*; English translation will appear in Springer-Verlag).
- Bazykin, A. D., A. I. Khibnik, and E. A. Aponina. 1983. A model of evolutionary appearance of dissipative structure in ecosystems. *J. Math. Biol.* 18: 13-23.
- Henon, M. 1976. A two-dimensional mapping with a strange attractor. *Comm. Math. Phys.* 50: 69-77.
- Kryukov, V. I. 1978. Markov interacting processes and neuronal activity. *Lect. Notes Math.* 653: 122-139.
- Lorenz, E. N. 1963. Deterministic non-periodic flows. *J. Atmos. Sci.* 20: 130-141.
- May, R. 1976. Simple mathematical models with very complicated dynamics. *Nature* 261: 459-467.
- Parker, T. S., and L. O. Chua. 1989. Practical Numerical Algorithms for Chaotic Systems. Springer-Verlag, New York.
- Sparrow, C. 1982. The Lorenz Equations: Bifurcations, Chaos, and Strange Attractors. Applied Mathematical Sciences, vol. 41. Springer-Verlag, New York.

Appendix A





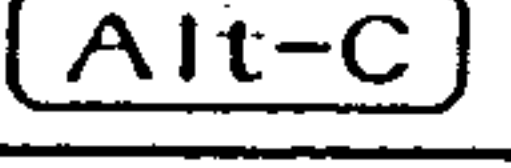

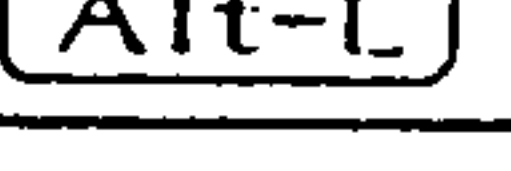

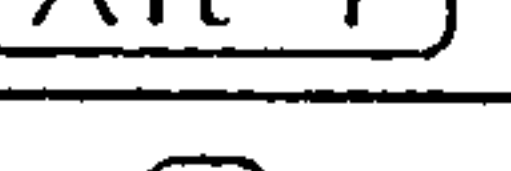

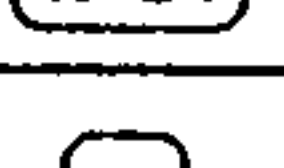
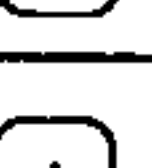


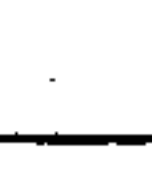
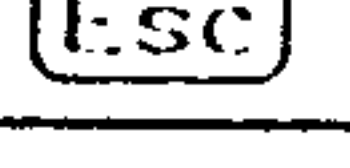
List of TraX commands

This appendix lists the three groups of TraX commands: archive management commands (A.1), input and editing commands (A.2) and system analysis commands (A.3).

A.1 Archive management commands

KEYCAP	FUNCTION
[F1]	Get help.
[F2]	Quit TraX.
[F3]	Print an archive.
↑	Go to the previous line.
↓	Go to the next line.
←	Go to the previous level.
→	Go to the next level.
↩	Select a dynamical system (with the default state) or a state.
Home	Create a new level.
Ins	Create a new dynamical system using an example.
Shift-Ins	Create a new dynamical system without using an example.
Del	Delete an empty level, a dynamical system without stored states, or a state.
	Rename a level, a dynamical system, or a state.
Shift-Ins	Display the RHS of a system.

A.2 Input and editing commands

KEYCAP	FUNCTION
	Input a pair of parentheses ().
	Input a pair of square brackets [].
	Input a pair of braces { }.
	Input ABS().
	Input COS().
	Input EXP().
	Input LOG().
	Input SQR().
	Input TAN().
	Delete a character to the left of the cursor.
	Delete a character at the cursor.
	Move the cursor to the left (to the end of the text).
	Move the cursor to the right (to the start of the text).
	Get help.
	Notify TraX that you have edited a function or have specified a dynamical system (if nothing was typed).
	Re-input the RHS.

A.3 System analysis commands

KEYCAP	FUNCTION
(F1)	Get help.
(F2)	Quit the current dynamical system and return to the archive.
(F3)	Store the current state.
(F4)	Write a comment on the screen.
(F5)	Activate the next window.
(F6)	Plot a function graph in the active window.
(F7)	Clear all graphic windows.
(F8)	Draw axes in all graphic windows.
(F9)	Plot a trajectory in the backwards direction.
(F10)	Plot a trajectory in the forward direction.
(Shift-F1)	Print the RHS of a current dynamical system on the screen.
(Shift-F2)	Compute the values of the RHS at the current point.
(Shift-F3)	Store the current state as the default state.
(Shift-F4)	Store the coordinates of a frame (see (Shift-F6)).
(Shift-F5)	Activate the next window.
(Shift-F6)	Set the limits of visibility in the active window using a frame (see (Shift-F4)).
(Shift-F7)	Move or resize an active window.
(Shift-F9)	Make the current point the new initial point in the backward direction.
(Shift-F10)	Make the current point the new initial point in the forward direction.
(Ctrl-F4)	Start a keystroke macro definition. Press (Alt) + digit key to complete and store the macro definition.
(Ctrl-F5)	Change the foreground color in the active window.
(Ctrl-F6)	Change the background color in the active window.
(Ctrl-F7)	Edit a function located along the ordinate in active window.
(Ctrl-F8)	Edit a function located along the abscissa in active window.
(Ctrl-PgUp)	Exchange a highlighted line with the previous one in the parameter window.
(Ctrl-PgDn)	Exchange a highlighted line with the next one in the parameter window.
(Alt-F7)	Clear the active graphic window.
(Alt-F9)	Plot a trajectory in the backward direction using the foreground color of active window.
(Alt-F10)	Plot a trajectory in the forward direction using the foreground color of active window.

[Alt-1]...	Playback the previously defined keystroke macro number digit or complete macro definition.
[←]	Enter the function editing mode.
[?]	Evaluate a function.
[=]	Evaluate a function and assign the value to the highlighted parameter.
[Del]	Delete an active graphic window.
[Ins]	Create a new graphic window.
[PgUp]	Move the parameter highlight one line up.
[PgDn]	Move the parameter highlight one line down.
[++1]...	Increment the highlighted parameter.
--1]...	Decrement the highlighted parameter.
[X]...	Increment the corresponding coordinate (x, y, u, v, w, o, q, r, or s) of an initial point.
[X]...	Decrement the corresponding coordinate (X, Y, U, V, W, O, Q, R, or S) of an initial point.
[→]	Increment an X-coordinate of an initial point if the parameter window is active, otherwise increase the coordinate corresponding to abscissa.
[←]	Decrement an X-coordinate of an initial point if the parameter window is active, otherwise decrease the coordinate corresponding to abscissa.
[↑]	Increment a Y-coordinate of an initial point if the parameter window is active, otherwise increase the coordinate corresponding to ordinate.
[↓]	Decrement a Y-coordinate of an initial point if the parameter window is active, otherwise decrease the coordinate corresponding to ordinate.
[⇐]	Repeat the previously pressed directional arrow key 32 times.