Math 131/135/194, Fall 2004

Applied Calculus

Profs. Kaplan & Flath

Macalester College

Visualizing the Phase Plane using R

Goal of this lab

To understand the relationship between the flow field and the trajectory for systems of two variables, and the nature of single-variable equilibrium and simultaneous equilibrium.

Introduction

We have been studying dynamical systems of the form

$$\frac{dx}{dt} = f(x,y) \quad (1)$$

$$\frac{dy}{dt} = g(x,y) \quad (2)$$

The computer can be used to visualize the flows arising from the functions $f(x,y)$ and $g(x,y)$. This document describes R-stats software for doing this. The software is contained in the file `pplane.r` which must be "sourced" into R-stats using the FILE/SOURCE R CODE menu item.

The pplane software consists of two parts:

1. Definitions of some dynamical systems
2. Software for generating contour maps, flows, and trajectories for these dynamical systems.

Definitions of Dynamical Systems

We have pre-defined several dynamical systems for you, each modeling a different sort of physical or biological situation. In each case, the system is the pair of functions $f(x,y)$ and $g(x,y)$ packaged together; you give a vector (x,y) as an argument and the system function will return two values: the value of f and the value of g at the specified point. For example:

```
> pp(c(3,4))
[1]  2.988 -3.988
```

Here are the predefined systems:

`spring1` The dynamics of a spring-mass system with air friction. In physics, this sort of system is called a "damped harmonic oscillator." The x variable is the spring position, the y variable is the spring velocity.

`spring2` Another damped harmonic oscillor with different parameters. Again, x is the spring position and y is the spring velocity.

`spring3` Still another harmonic oscillor, but where the "air friction" works backwards — it pushes instead of resisting. Again, x is the spring position and y is the spring velocity.

`pp` A predator-prey system.

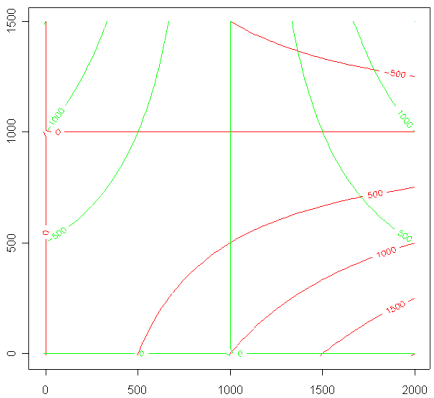
`comp` A competition system in which two populations of animals compete for the same limited resource.

Displaying a Specific Dynamical System

Once you have picked the dynamical system you want to study, you can use several programs to display the functions $f(x,y)$ and $g(x,y)$, the flow field, or specific trajectories. For the following examples, we'll use the `pp` predator-prey system:

`showcontours(system, xlims, ylims, add=F)` This will display the contours of the two functions $f(x,y)$ and $g(x,y)$. Red lines are used for $f(x,y)$ and green lines for $g(x,y)$. In addition to giving the name of the system, you need to define the limits of the x- and y-axes. For example, in the next two lines we define a specific system and then plot out the contours of that system:

```
> showcontours(pp, c(-10,2000), c(-10,1500))
```



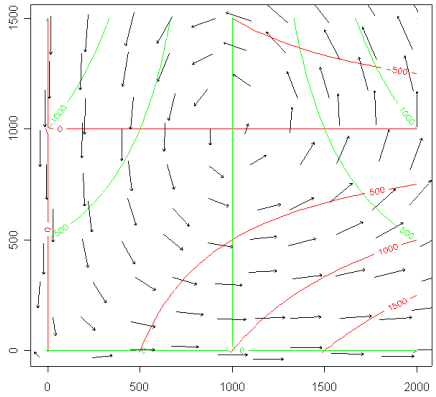
The equilibrium points of the system can be read off the graph as the intersections of the red and green zero contour lines. But note that depending on how you set `xlims` and `ylims` the points may or may not be visible.

The optional argument `add` controls whether the graph display is erased before drawing the contours. Use `add=T` as an argument in order to overlay the contours on the existing graph.

`nullclines(system, xlims, ylims, add=F)` This works in just the same way as `showcontours`, but only the zero-level contour is being displayed.

`phasearrows(system,xlims,ylims,resol=10,col='black',add=F)` This draws in the flow arrows. For example

```
> phasearrows(pp, c(-10,2000), c(-10,1500), add=T)
```

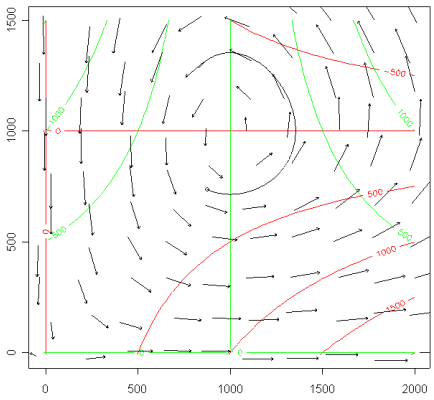


Use `add=T` as an argument in order to overlay the arrows on the existing graph. You can use the `resol` and `col` arguments to control how many arrows are displayed and what color is used.

`phasetraj(system, duration)` draws a trajectory starting at a point you specify with the mouse. When you invoke this function, the display will appear. Click the mouse at the starting point and, after a short delay, the trajectory will appear. For example:

```
> phasetraj(pp, 2)
```

then click at the point you want to start the trajectory:



`eulerxy(system, initial, tstart, tend)` This will integrate the system using Euler's method. You need to specify the name of the system, an initial condition, a starting time, and an ending time. For example:

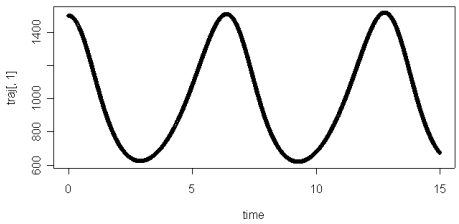
```
> traj = eulerxy(pp, c(1500,1000), 0, 15)
```

The solution can be plotted versus time by using the two components of the returned value. The R-language uses a special notation, involving `$`, when there is more than one component to a variable. The trajectory returned by `eulerxy` has two components: the times at which the solution was calculated and the values of the trajectory. These two components are called `t` and `x` respectively. Since the `x` component involves two variables, it has two columns. This can be confusing, and is best explained by example. To plot the first variable in the solution versus time:

```
> plot(traj$t,traj$x[,1])
```

and for the second variable

```
> plot(traj$t,traj$x[,2])
```



`rk(system, initial, tstart, tend)` Like `eulerxy`, the `rk` program integrates the system. But it uses the more sophisticated Runge-Kutta method that is discussed in courses such as Math 312 and 365. `Rk` gives a more reliable answer than `eulerxy`, so it is to be preferred to `eulerxy`.

`jacobianAtXY(system)` allows the mouse to be used to identify a point and returns the numerical Jacobian matrix at the point the mouse is clicked at. If the mouse is clicked at an equilibrium point, the Jacobian can be used to find the stability of the system. Note that the precision of the mouse click is not necessarily very high. You can also give values to x and y directly, as in

```
> j = jacobianAtXY(pp,1000,1000)
> j
     [,1] [,2]
[1,]    0   -1
[2,]    1    0
```

`eigen` computes the "eigenvalues" and "eigenvectors" of the Jacobian matrix. To assess the stability of an equilibrium, one looks at the eigenvalues of the Jacobian at the equilibrium.

```
> eigen(j)
$values
[1] 0+1i 0-1i
```

A matrix containing "eigenvectors" will also printed; you can ignore this.

Tasks

1. For the `spring1` system, use the `showcontours` program to draw contour lines for system, that is, for the $f(x, y)$ and $g(x, y)$ functions. You should set the `xlim` and `ylim` graphing limits so that an intersection point of the zero contours is near the center of your graph. You will have to experiment to find appropriate graphing limits, but if you read the contour plots you should be able to infer where the intersection is.

2. Once you have a nicely delimited plot, use the `phasetraj` program to draw in a trajectory. The first argument will be the name of the system, `spring1`. The second argument is the duration of the trajectory. You should set this duration argument to give you an informative looking trajectory; experiment with different values (perhaps starting with a duration of 1.) Once the function is invoked, use the mouse to click at the place where you want the trajectory to start. After a few seconds, the trajectory will appear.

   Using a starting point somewhat near the equilibrium of the system, judge from the trajectory whether the equilibrium is stable or unstable.

   Explain in physical terms what is happening to the spring length at various points on the trajectory.

3. Use `phasetraj` to trace a trajectory that starts on one of the zero contours of $f(x, y)$ but is away from the simultaneous equilibrium. That is, the point you choose should be at (x, y) where $f(x, y) = 0$ but $g(x, y) \neq 0$. From this point, what direction does the trajectory go in very close to the point you chose? Next, choose a point (x, y) where $g(x, y) = 0$ but $f(x, y) \neq 0$. What direction does the trajectory go in from this point? Explain why the two directions are so different.

   Explain in physical terms what it means for the system's state to be on each of the two zero contours.

4. Add the phase arrows to your plot. Choose some point in the phase plane that is not so close to the equilibrium and explain how the direction of the arrows relates to the value of the $f(x, y)$ and $g(x, y)$ functions as indicated by the contour plot.

   Then, use `phasetraj` to draw in a trajectory near that point. How does the trajectory relate to the phase arrows.

5. Use the `rk` program to solve/integrate the `spring1` system starting at the point $(x = 100, y = 0)$ and going from time $t = 0$ to $t = 10$. (See the instructions for `rk` and for `euler`.) Make a graph of the solution versus time. Compare this to the trajectory in the phase plane that you find from the same starting point.

6. Use `jacobianAtXY` to find the Jacobian matrix of the system at the equilibrium point. (If you are displaying a contour plot (from `showcontours` or `nullclines`, you can use the mouse to specify to `jacobianAtXY` where the equilibrium is — try to be fairly precise.) Record the eigenvalues of this Jacobian and relate those eigenvalues to the stability of the system. (Note: the eigenvalues may be complex numbers in some cases, so think about what the real and imaginary parts tell you about the stability of the equilibrium.)

7. Repeat steps (1) through (5) for the `spring3` systems. Say how this system is different from a typical damped spring.

8. See if you can to the stability of that system. (Note: the eigenvalues may be complex numbers in some cases.)

9. Play around with the predator prey system. Tell us something interesting about it, such as the stability of the equilibrium.