Using Matlab to get Phase Portraits

Once upon a time if you wanted to use the computer to study continuous dynamical systems you had to learn a lot about numerical methods. Now we have Matlab that does a lot of this work for us. On this page I explain how to use Matlab to draw phase portraits for the the two linear systems

```
(1) dx/dt = x + 3 y , dy/dt = -5 x + 2 y

(2) dx/dt = 4 x - 2 y , dy/dt = x - 3 y
```

(1) has an unstable focus at (0,0).
(2) has a saddle point at (0,0)

The first thing you need when you want to solve a system of differential equations in Matlab is a function that computes the derivative (i.e. the right hand sides of the differential equations). So for system (1) make a file called **dxdt1.m** with the following content:

```
function d=dxdt1(t,x)

d=[ x(1)+3*x(2); -5*x(1)+2*x(2) ];
```

Note two things:
First, Matlab uses x(1),x(2) instead of x and y.
Second, note the appearance of "t" in the definition of the function (first line), even though it is not used. Never forget this!

For the system (2) the file, which we will call **dxdt2.m**, should look like this:

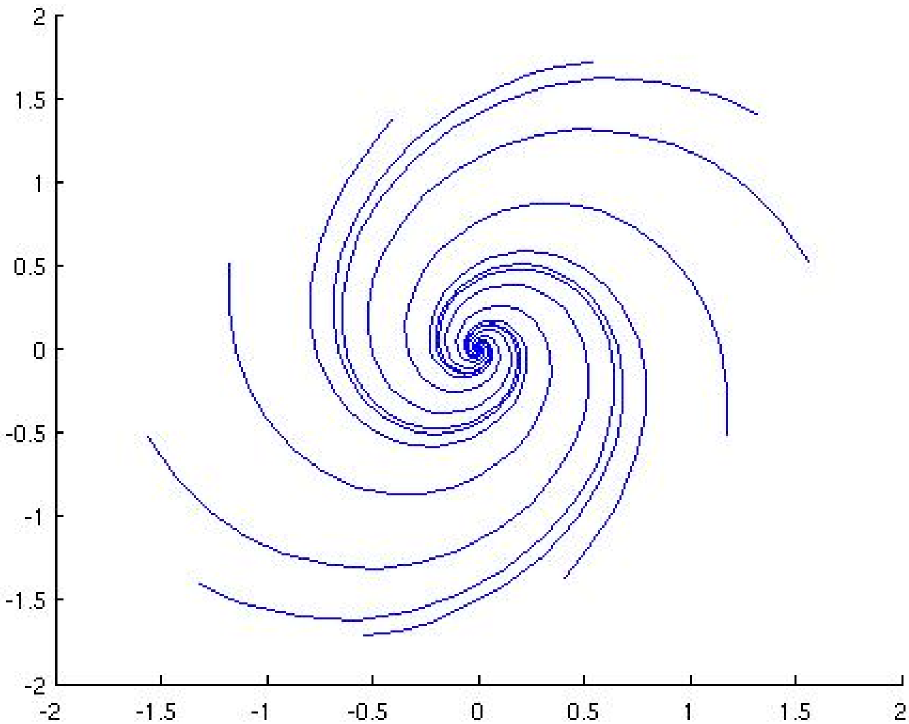
```
function d=dxdt2(t,x)

d=[ 4*x(1)-2*x(2); x(1)-3*x(2) ];
```

Here is some Matlab code that makes 11 different orbits for system (1) and plots them:

```
figure(1)
hold on
for theta=[0:10]*pi/5
    x0=1e-5*[cos(theta);sin(theta)];
    [t,x]=ode45(@dxdt1,[0 8],x0);
    plot(x(:,1),x(:,2))
end
```

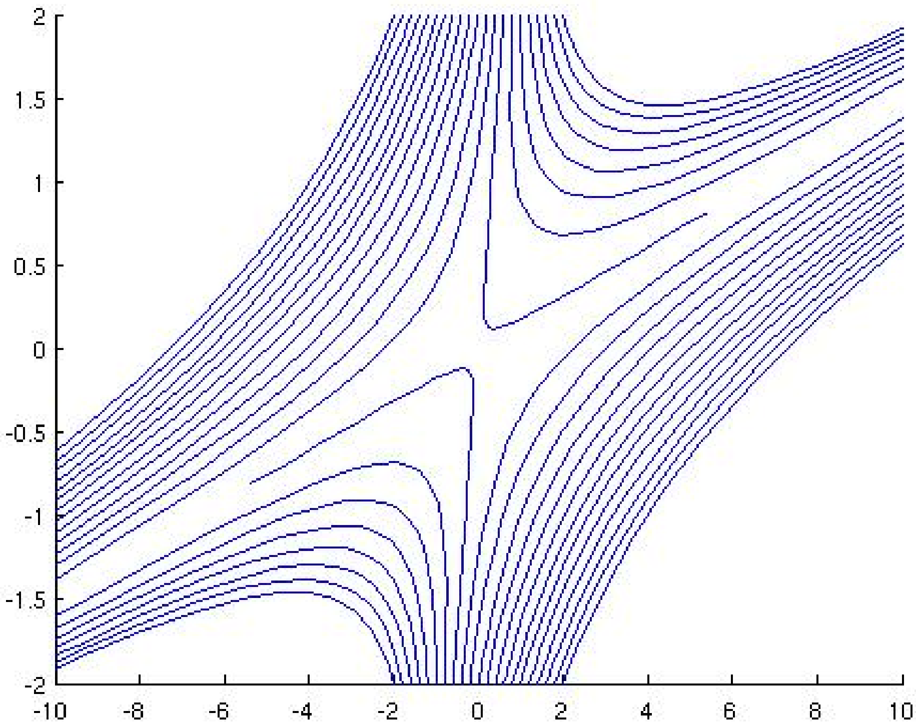
For each orbit, the point x0 specifies the initial condition. We take these points around a circle of radius 1e-5 centered at the origin. The "ode45" command generates the solution of the system, integrating from t=0 to t=8. Finally the "plot" command plots x(2) against x(1) (i.e. y against x). Here's the result:



For the system (2) things are a little harder. Here is the code I used:

```
figure(2)
hold on
for theta=[-10:10]/5
    x0=[theta 2];
    [t,x]=ode45(@dxdt2,[0 2],x0);
    plot(x(:,1),x(:,2))
    x0=[theta -2];
    [t,x]=ode45(@dxdt2,[0 2],x0);
    plot(x(:,1),x(:,2))
end
axis([-10 10 -2 2])
```

Here I started with initial conditions on the lines y=2 and y=-2. I only integrated up to t=2. And the last command "axis([-10 10 -2 2])" limits the plot to the critical region with x between -10 and 10 and y between -2 and 2. Here's what you get:



The difficulty here lies in the fact that we want to draw orbits both close and far from the fixed point. But the orbits that are far diverge much faster. Still, by integrating only till t=2 and keeping a fixed range for the plot, we have achieved a satisfactory result.

Back to main course page
Back to my main teaching page
Back to my home page