

Math 2250-3

November 12, 2003

First order systems of Differential Equations

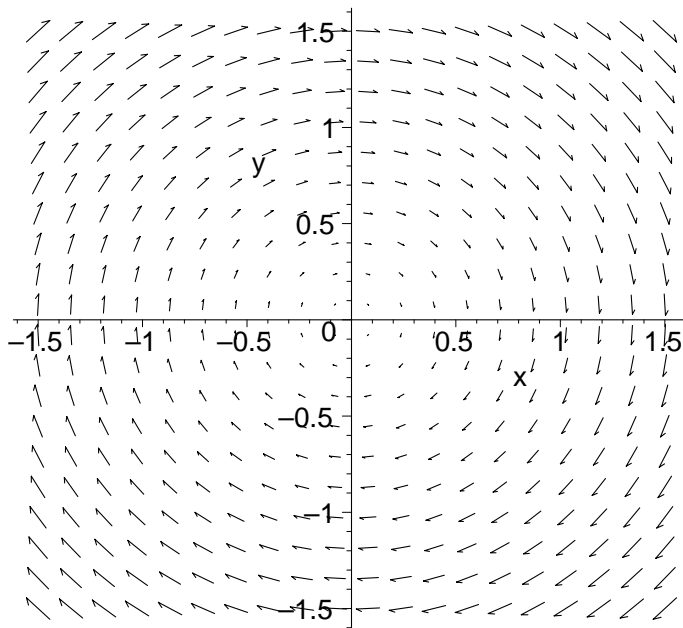
Why you should expect existence and uniqueness for the IVP

Example: Consider the initial value problem

$$\begin{cases} \frac{dx}{dt} \\ \frac{dy}{dt} \end{cases} = \begin{bmatrix} y \\ -x \end{bmatrix}$$
$$\begin{cases} x(0) \\ y(0) \end{cases} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

We'll see later where this system of 2 linear first order differential equations comes from. For now let's see why we expect a solution, and why we expect it to be unique. Remember the geometric interpretation of this IVP: If we plot the solution curve $[(x(t), y(t))]$ in the plane, it passes through $[1, 0]$ at $t=0$, and then moves so that its tangent vector $[x'(t), y'(t)]$ agrees with the DE vector field at the point $[x(t), y(t)]$, namely with $[y, -x]$. So if we create a picture of the vector field we can visualize where the solution trajectory must go:

```
[ > restart:with(DEtools):with(plots):with(linalg):  
[ > fieldplot([y,-x],x=-1.5..1.5,y=-1.5..1.5);
```



Now, pretend (?) you don't know how to find the analytic solution to our page 1 initial value problem. We can use Euler's method to get a numerical solution! In other words, approximate using the tangent line approximation, and replace the tangent vector by what the differential equation says it is:

$$\begin{bmatrix} x(t + \Delta t) \\ y(t + \Delta t) \end{bmatrix} \approx \begin{bmatrix} x(t) \\ y(t) \end{bmatrix} + \Delta t \begin{bmatrix} \frac{d}{dt} x(t) \\ \frac{d}{dt} y(t) \end{bmatrix}$$

$$\begin{bmatrix} x(t + \Delta t) \\ y(t + \Delta t) \end{bmatrix} \approx \begin{bmatrix} x(t) \\ y(t) \end{bmatrix} + \Delta t \begin{bmatrix} y(t) \\ -x(t) \end{bmatrix}$$

With the initial conditions which were specified earlier, we would start at the point [1,0], and then follow the velocity field around curves which look a lot like circles. If we wanted to get a numerical approximation, we could just transpose Euler (or improved Euler or Runge-Kutta) for first order equations to first order systems: (Or we could use one of Maple's many numerical packages.)

```

> f1:=(x,y,t)->y:
  f2:=(x,y,t)->-x:
  #our right hand side F(X,t)=[f1(X,t),f2(X,t)]
  #in our first order system of DE's dX/dt = F(t,X).
> n:=100:
  #number of time steps
  t0:=0: #initial time
  tn:=5.0:
  #final time
  h:=(tn-t0)/n:
  #time step
  xs:=vector(n+1):
  ys:=vector(n+1):
  #vectors to hold Euler approximates
  xs[1]:=1:
  ys[1]:=0:
  #initial conditions

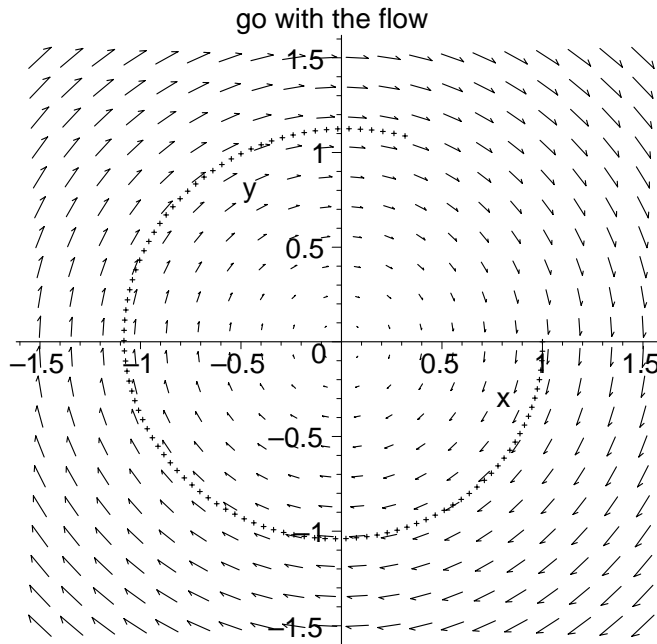
> #Euler loop:
  for i from 1 to n do
    X:=xs[i]:
    Y:=ys[i]:
    t:=t0+(i-1)*h:
    k1:=f1(X,Y,t):
    k2:=f2(X,Y,t):
    #k1 and k2 are velocity components
    xs[i+1]:=X+h*k1:
    ys[i+1]:=Y+h*k2:
    #update xs and ys with Euler!
  od:

```

```

> veloc:=fieldplot([y,-x],x=-1.5..1.5,y=-1.5..1.5):
Eulerapprox:=pointplot({seq([xs[i],ys[i]],i=1..n+1)}):
display({veloc,Eulerapprox},title="go with the flow");

```



Any first order system is analogous to the above example, and that's why we expect existence and uniqueness in general.

Where did this particular IVP come from? Well, this is one which we can deconvert into a single second order DE: Since $x'(t)=y$ and $y'(t)=-x$, we deduce $x''(t)=-x$, i.e.

$$\left(\frac{d^2}{dt^2} x(t)\right) + x(t) = 0$$

And the initial conditions for $x(t)$ become

$$\begin{bmatrix} x(0) \\ D(x)(0) \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

This is the simple harmonic oscillator with $m=k=1$ and $c=0$, so the solution is $x(t)=\cos(t)$. So, the solution to the first order system is $[x(t),x'(t)]=[\cos(t),-\sin(t)]$. This traces out the unit circle in the clockwise direction, exactly as our numerical approximation predicted. Notice the vertical axis is the velocity ($x'(t)$) axis, so this plot is keeping track of position AND velocity of the oscillating mass. Physicists call this picture a phase portrait for the simple harmonic oscillator.

Notice also that this is a conservative system and the total energy (KE+PE) = E satisfies

$$E = \frac{1}{2} \left(\frac{d}{dt} x(t) \right)^2 + \frac{1}{2} x(t)^2$$

$$E = \frac{y^2}{2} + \frac{x^2}{2}$$

And this is constant along any solution trajectory because total energy is conserved! That's why the solution curves to any IVP for this system of DE's are circles.

While we're on the subject of physics here's another beautiful example of a phase plane, this time for the massless, rigid-rod pendulum. Recall, we used conservation of energy to derive the differential equation for the angle theta(t):

$$L \left(\frac{d^2}{dt^2} \theta(t) \right) + g \sin(\theta(t)) = 0$$

Now set L=g=1, and convert this to a first order system:

$$x(t) := \theta(t)$$

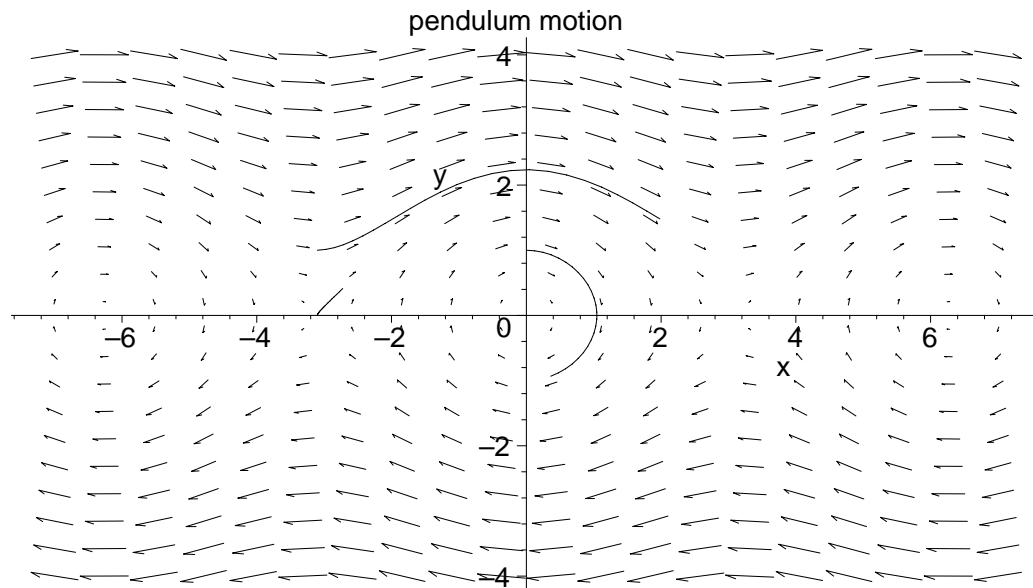
$$y(t) := \frac{d}{dt} \theta(t)$$

so,

$$\begin{bmatrix} \frac{dx}{dt} \\ \frac{dy}{dt} \end{bmatrix} = \begin{bmatrix} y \\ -\sin(x) \end{bmatrix}$$

Here's the phase portrait for the pendulum system, where we called the vertical axis v, for pendulum velocity. Pieces of three solution trajectories are drawn on to the phase portrait. Take a few minutes to try and understand them.

```
[ > restart:with(linalg):with(DEtools):with(plots):
> velocfield:=fieldplot([v,-sin(theta)],theta=-7..7,v=-4..4):
pendeqtn1:={diff(x(t),t)=y(t),diff(y(t),t)=-sin(x(t)),
x(0)=0,y(0)=1}:
traj1:=odeplot(dsolve(pendeqtn1,
[x(t),y(t)], type=numeric),[x(t),y(t)],
0..3,numpoints=25,color=black):
pendeqtn2:={diff(x(t),t)=y(t),diff(y(t),t)=-sin(x(t)),
x(0)=-3.1,y(0)=0}:
traj2:=odeplot(dsolve(pendeqtn2,
[x(t),y(t)], type=numeric),[x(t),y(t)],
0..3,numpoints=25,color=black):
pendeqtn3:={diff(x(t),t)=y(t),diff(y(t),t)=-sin(x(t)),
x(0)=-3.1,y(0)=1}:
traj3:=odeplot(dsolve(pendeqtn3,
[x(t),y(t)], type=numeric),[x(t),y(t)],
0..3,numpoints=25,color=black):
display({velocfield,traj1,traj2,traj3},
title="pendulum motion");
```



Remarks:

(1) Any single n th order differential equation is equivalent to a related first order system of n first order differential equations. You get this system in the same way as we did above. Then the existence and uniqueness theorem for first order systems implies the existence and uniqueness theorem that we quoted a month ago for n th order (linear) differential equations!

(2) If you want to solve an n th order differential equation numerically you first convert it to a first order system, and then use Euler or Runge-Kutta, to solve it! (Or you use a package that does something like that while you aren't looking.)