

Numerical Methods for Systems

David Arnold

Spring 1997

Abstract

This activity introduces students to solution technique available for the numerical solution of systems of first order ordinary differential equations with initial conditions. In particular, euler's method, rk2, and rk4 are compared and contrasted. Because the approach is completely visual, students will need little, if any, prerequisite material on solving systems of ordinary differential equations before attempting this activity.

Prerequisites. Some familiarity with Matlab's `plot` command. Also, you will need to download `eul.m`, `rk2.m`, and `rk4.m`. The activity also uses Version 2 of the Symbolic Toolbox, a Matlab 5 interface to Maple. If you do not have the Symbolic Toolbox, you can use the solutions provided in the activity to continue with the numerical investigation.

1 Systems of First Order Differential Equations

A system of first order ordinary differential equations has the following form:

$$\frac{d\mathbf{Y}}{dt} = \mathbf{F}(t, \mathbf{Y}) \quad (1)$$

Note that equation (1) is similar in form to that of a first order ordinary differential equation:

$$\frac{dy}{dt} = f(t, y) \quad (2)$$

However, equation (1) makes heavy use of vector notation. Vector notation, though elegant, can also be thoroughly confusing. Let's take a moment to level the playing field a bit.

1.1 The Position Vector

You are probably familiar with parametric equations. For example, the parametric equations

$$\begin{aligned} x &= \cos t \\ y &= \sin t \end{aligned} \quad (3)$$

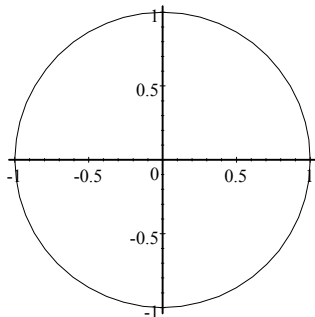


Figure 1: The plot of $x = \cos t$, $y = \sin t$, $0 \leq t \leq 2\pi$.

define the path shown in Figure 1 over the time interval $0 \leq t \leq 2\pi$.

Now, imagine a vector, with tail positioned at the origin, the tip of whose head traces out the path shown in Figure 1. Such a vector is called a *position vector*. There are several equivalent notations for the position vector that traces the path in Figure 1. Multivariable calculus students use the notation $\mathbf{Y}(t) = \cos t \mathbf{i} + \sin t \mathbf{j}$ to define the position vector. Linear algebra students prefer the following notation:

$$\mathbf{Y}(t) = \begin{bmatrix} \cos t \\ \sin t \end{bmatrix} \quad (4)$$

Textbook publishers, in an effort to conserve space, will use $\mathbf{Y}(t) = [\cos t, \sin t]$ or $\mathbf{Y}(t) = (\cos t, \sin t)$ to denote the position vector.

1.2 The Velocity Vector

If your first calculus class, once you had defined the position of a particle on a line as a function of time, then the first derivative (with respect to time) of the position function defined the velocity of the particle as a function of time. It is no different in the plane. If you define the position of a particle as a function of time (by defining the position vector), then the first derivative of the position vector (with respect to time) yields the velocity of the particle as a function of time.

It is a simple matter to differentiate a vector valued function. If $\mathbf{Y}(t) = [x(t), y(t)]$, then

$$\frac{d}{dt} \mathbf{Y}(t) = \begin{bmatrix} \frac{dx}{dt} \\ \frac{dy}{dt} \end{bmatrix}$$

or, in shorthand notation,

$$\frac{d\mathbf{Y}}{dt} = \begin{bmatrix} \frac{dx}{dt} \\ \frac{dy}{dt} \end{bmatrix} \quad (5)$$

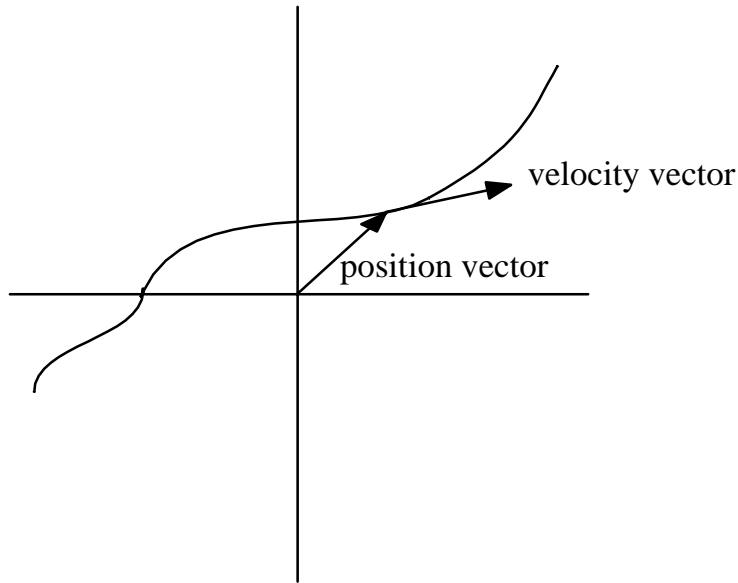


Figure 2: The velocity is the first derivative of the position.

For example, if $\mathbf{Y}(t) = [\cos t, \sin t]$, then $d\mathbf{Y}/dt = [-\sin t, \cos t]$.

1.3 Vector Fields

Consider the function defined by $\mathbf{F}(x, y) = U(x, y)\mathbf{i} + V(x, y)\mathbf{j}$ or

$$\mathbf{F}(x, y) = \begin{bmatrix} U(x, y) \\ V(x, y) \end{bmatrix} \quad (6)$$

Equation (6) associates a vector $[U(x, y), V(x, y)]$ with each point (x, y) in the plane. The set of vectors produced by equation (6) is called a *vector field*.

For example, consider the vector field defined by the equation

$$\mathbf{F}(x, y) = \begin{bmatrix} -y \\ x \end{bmatrix} \quad (7)$$

Equation (7) associates a vector with each point in the plane. For example, $\mathbf{F}(1, 1) = [-1, 1]$. You would plot this result by constructing the vector $[-1, 1]$ at the point $(1, 1)$, as shown in Figure 3. Now, select a grid of points in the plane, calculate $\mathbf{F}(x, y)$ at each point (x, y) in the grid, and attach the resulting vector at the point (x, y) as shown in Figure 3. If you do this at each point of your grid, you get the vector field similar to that shown¹ in Figure 4. If the vectors are left at their exact length, the vector field quickly becomes crowded and

¹The vector field in Figure 4 was produced with the following Matlab commands:

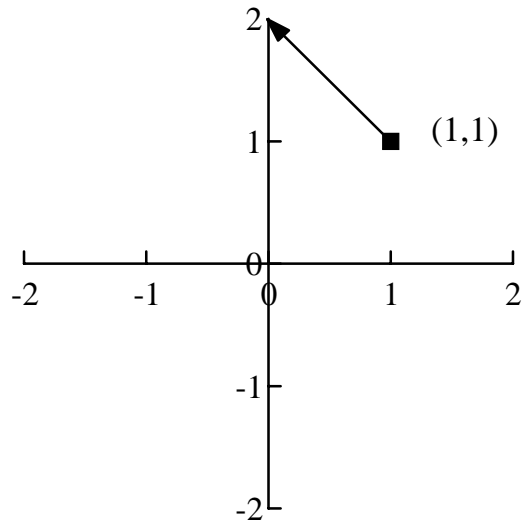


Figure 3: The vector $[-1, 1]$ attached to the point $(1, 1)$.

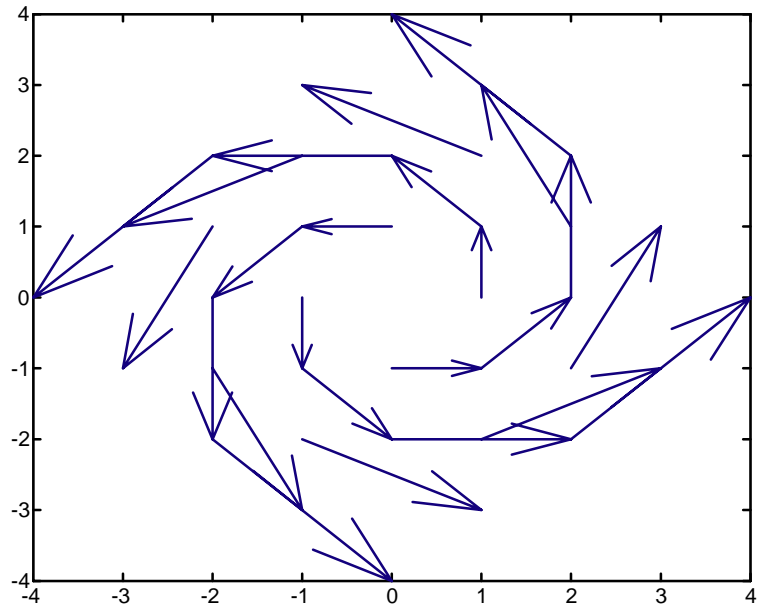


Figure 4: The vector field $\mathbf{F}(x, y) = [-y, x]$.

unreadable, as you can see in Figure 4. Therefore, software programs usually scale the vectors so that their lengths are correct relative to one another. The directions of the vectors remain the same. This gives a good indication of the flow without overly crowding the field. An example of this scaling is shown in Figure 5, where the vector field represented by $\mathbf{F}(x, y) = [-y, x]$ is drawn².

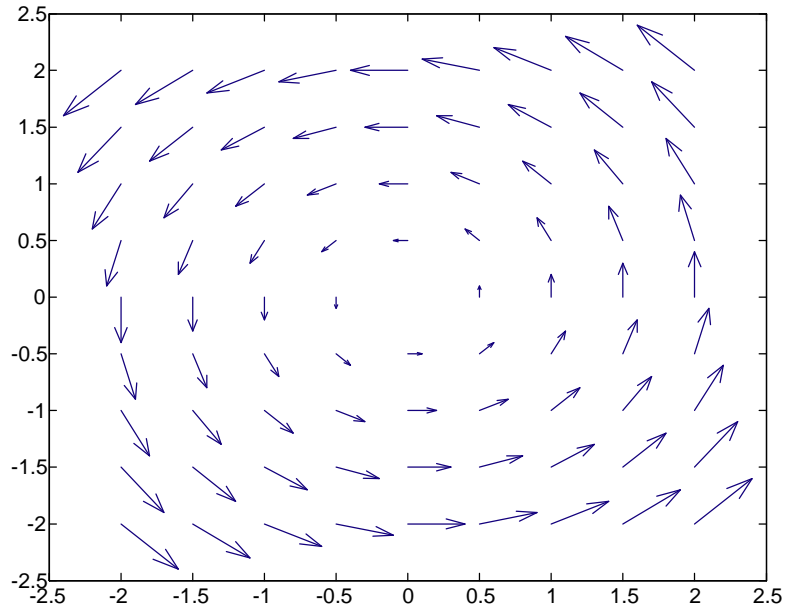


Figure 5: Scaling the vector field $\mathbf{F}(x, y) = [-y, x]$.

1.4 Putting It All Together

Consider the following system of first order ordinary differential equations.

$$\begin{aligned} \frac{dx}{dt} &= -y \\ \frac{dy}{dt} &= x \end{aligned} \tag{8}$$

```
>> [x,y]=meshgrid(-2:1:2);
>> u = -y; v=x;
>> quiver(x,y,u,v,0)
2The vector field in Figure 4 was produced with the following Matlab commands:
>> [x,y]=meshgrid(-2:.5:2);
>> u = -y; v=x;
>> quiver(x,y,u,v)
```

The following system is identical to the system (8), but emphasizes that the solutions x and y are functions of time t .

$$\begin{aligned}\frac{d}{dt}x(t) &= -y(t) \\ \frac{d}{dt}y(t) &= x(t)\end{aligned}\tag{9}$$

The task before us is to find functions $x(t)$ and $y(t)$ that satisfy both differential equations in system (9).

The following system is identical to the systems (8) and (9), but it is written in vector form.

$$\begin{bmatrix} \frac{d}{dt}x(t) \\ \frac{d}{dt}y(t) \end{bmatrix} = \begin{bmatrix} -y(t) \\ x(t) \end{bmatrix}\tag{10}$$

If we write the solution to system (10) as $\mathbf{Y}(t) = [x(t), y(t)]$, then $\frac{d}{dt}\mathbf{Y}(t) = [\frac{d}{dt}x(t), \frac{d}{dt}y(t)]$ and equation (10) takes on the form shown in equations (11) and (12), where $\mathbf{F}(x(t), y(t)) = [-y(t), x(t)]$.

$$\frac{d}{dt}\mathbf{Y}(t) = \mathbf{F}(x(t), y(t))\tag{11}$$

$$\frac{d}{dt}\mathbf{Y}(t) = \mathbf{F}(\mathbf{Y}(t))\tag{12}$$

If you assume that \mathbf{Y} is a function of t , then you can abbreviate further, as shown in equation (13).

$$\frac{d\mathbf{Y}}{dt} = \mathbf{F}(\mathbf{Y})\tag{13}$$

The system represented by equation (13) is *autonomous*. The vector field on the right side of equation (13) does not change with time.

1.5 Non Autonomous

Consider the following system of first order ordinary differential equations.

$$\begin{aligned}\frac{dx}{dt} &= x + y - \cos t \\ \frac{dy}{dt} &= x - y - \sin t\end{aligned}\tag{14}$$

In vector form, system (14) would have the following appearance:

$$\begin{bmatrix} \frac{dx}{dt} \\ \frac{dy}{dt} \end{bmatrix} = \begin{bmatrix} x + y - \cos t \\ x - y - \sin t \end{bmatrix}\tag{15}$$

Once again, emphasize that solutions x and y are functions of time t .

$$\begin{bmatrix} \frac{d}{dt}x(t) \\ \frac{d}{dt}y(t) \end{bmatrix} = \begin{bmatrix} x(t) + y(t) - \cos t \\ x(t) - y(t) - \sin t \end{bmatrix}\tag{16}$$

Again, if the solution is written as $\mathbf{Y}(t) = [x(t), y(t)]$, then equation (16) takes on the form shown in equations (17) and (18), where $\mathbf{F}(t, x(t), y(t)) = [x(t) + y(t) - \cos t, x(t) - y(t) - \sin t]$.

$$\frac{d}{dt}\mathbf{Y}(t) = \mathbf{F}(t, x(t), y(t)) \quad (17)$$

$$\frac{d}{dt}\mathbf{Y}(t) = \mathbf{F}(t, \mathbf{Y}(t)) \quad (18)$$

If you assume that \mathbf{Y} is a function of t , then you can abbreviate further, as shown in equation (19).

$$\frac{d\mathbf{Y}}{dt} = \mathbf{F}(t, \mathbf{Y}) \quad (19)$$

Because the vector field on the right side of equation (19) is a function of both time and position, the system is *non-autonomous*.

2 Exact Solutions

Consider the following initial value problem.

$$\begin{aligned} \frac{dx}{dt} &= -y \\ \frac{dy}{dt} &= x \\ x(0) &= 1 \\ y(0) &= 0 \end{aligned} \quad (20)$$

It is an easy matter to use *Matlab's Symbolic Toolbox* to find a symbolic solution of the IVP (20).

```
>> [x,y]=dsolve('Dx=-y,Dy=x','x(0)=1,y(0)=0')
```

```
x =
```

```
cos(t)
```

```
y =
```

```
sin(t)
```

You can produce x versus t , y versus t , and y versus x plots with the following sequence of commands. Place these commands in a script file called `exact.m` and execute the file by typing `exact` at the *Matlab* prompt.

```

t=linspace(0,2*pi);
x=cos(t);y=sin(t);
subplot(2,2,1)
plot(t,x)
axis tight
title('x versus t plot')
subplot(2,2,2)
plot(t,y)
axis tight
title('y versus t plot')
subplot(2,1,2)
plot(x,y)
axis square
title('y versus x plot')
set(gcf,'NumberTitle','off')
set(gcf,'Name','dx/dt = -y, dy/dt = x, x(0)=1, y(0)=0')

```

The subplot command is used to place several plots on the same page. It is a very flexible command and you can find a nice description by typing `help subplot` at the *Matlab* prompt. Note the fancy use of the `axis` command in the script file `exact.m`. Type `help axis` at the *Matlab* prompt to find number of different uses of the `axis` command. Finally, note how we turned off the figure number and added a name to the figure window. *Matlab's* so-called “handle graphics” allow the user to completely customize the figure window. Type `set(gca)` or `set(gcf)` to get lists of the axis and figure window properties that can be modified with this technique.

3 Euler's Method

You can express the initial value problem posed in equation (20) in vector form as shown in equation (21)

$$\frac{d}{dt}\mathbf{Y}(t) = \mathbf{F}(t, \mathbf{Y}(t)), \quad \mathbf{Y}(0) = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad (21)$$

where $\mathbf{Y}(t) = [x(t), y(t)]$ and $\mathbf{F}(t, \mathbf{Y}(t)) = [-y(t), x(t)]$. Since $x(0) = 1$ and $y(0) = 0$, $\mathbf{Y}(0) = [x(0), y(0)] = [1, 0]$.

Euler's method is a numerical routine for computing solutions of IVPs. Dr. Polking of Rice University has written a routine, `eu1.m`, which will produce a numerical solution of an IVP using Euler's method. You will need to do two things to solve IVP (21):

- Create a function M-file defining the right side of the IVP.
- Use Dr. Polking's `eu1.m` routine to find an approximate solution.

3.1 Creating the M-file

Open a new file in the *Matlab M-File Editor/Debugger* and enter the following lines of code:

```
function Yprime=F(t,Y)
Yprime=zeros(2,1);
Yprime(1)=-Y(2);
Yprime(2)=Y(1);
```

Save the file as `F.m` in a directory on your *Matlab* path. There are several important points to note when you create the M-file for the IVP (21) representing the system of first order ODEs in equation (20).

- Note the use of uppercase to represent vectors, while lower case is used to represent scalars. Although this is not required, it is good practice. Upper case reminds us that the variable represents a vector.
- Since $\mathbf{Y}(t) = [x(t), y(t)]$, the input \mathbf{Y} to the function \mathbf{F} is a *column* vector with two elements. The first element of the vector \mathbf{Y} , which equals $x(t)$, is accessed with the notation $\mathbf{Y}(1)$. The second element of the vector \mathbf{y} , which equals $y(t)$, is accessed with the notation $\mathbf{Y}(2)$.
- Since $\mathbf{Y}(t) = [x(t), y(t)]$ is a vector, so too is $d\mathbf{Y}(t)/dt$. The line `Yprime=zeros(2,1)` creates a two element column vector (2 rows, 1 column) with a zero in each entry. The variable `Yprime` represents $d\mathbf{Y}/dt$. Initializing variables in this manner is recommended by the Mathworks and is a good reminder that `Yprime` is a two-element column vector.
- `Yprime(1)`, which equals dx/dt , is the first element of the vector `Yprime`; `Yprime(2)`, which equals dy/dt , is the second element of the vector `Yprime`.
- Because $dx/dt = -y$, `Yprime(1)=-Y(2)`. Because $dy/dt = x$, `Yprime(2)=Y(1)`.
- The vector field in system (20) is autonomous. Note that the variable `t` is not used in defining `Yprime`. However, you must still include `t` as shown in the first line of the function M-file.

3.2 Calling EUL.M

You will now use `eul.m` to find a numerical solution of (21) on the interval $[0, 2\pi]$. First, type `help eul` and read the resulting helpfile. Note that we need to pass this routine several parameters.

<code>'F'</code>	A string containing the name of the function M-file
<code>tspan</code>	The time interval in the form <code>[t0,tfinal]</code>
<code>Y0</code>	The initial \mathbf{Y} -value. This must be a column vector.
<code>ssize</code>	The step size

The function string name is 'F', the name of the function M-file we created to evaluate the right side of IVP (21). The timespan `tspan` is the interval $[0, 2\pi]$, and the initial value `Y0` is $[1; 0]$ (remember that $\mathbf{Y}(0) = [1, 0]$). Note that the initial value must be a column vector. Let's use a step size of 0.5.

```
>> [teul, Yeul]=eul('F', [0, 2*pi], [1; 0], 0.5)
```

```
teul =
```

```
0
0.5000
1.0000
1.5000
2.0000
2.5000
3.0000
3.5000
4.0000
4.5000
5.0000
5.5000
6.0000
6.2832
```

```
Yeul =
```

```
1.0000    0
1.0000    0.5000
0.7500    1.0000
0.2500    1.3750
-0.4375    1.5000
-1.1875    1.2812
-1.8281    0.6875
-2.1719   -0.2266
-2.0586   -1.3125
-1.4023   -2.3418
-0.2314   -3.0430
1.2900   -3.1587
2.8694   -2.5137
3.5812   -1.7011
```

Let's make several important points about this solution.

- The column vector `teul` contains the time. Note that consecutive elements in this vector are incremented by the step size, in this case 0.5.

- The matrix `Yeul` contains the position of the particle in the xy -plane at the times indicated in `teul`. The first column of `Yeul` contains the x -value of the position, the second column contains the y -value. To find the position of the particle at a particular time, first locate the time in the vector `teul`, then locate the position in the corresponding row of `Yeul`. For example, at time $t = 0.5$ seconds, the particle is at the point (1.0000, 0.5000) in the plane.

3.3 Plotting the Results

Take a moment to create the following plots³.

- You can obtain a plot of x versus t with the command `plot(teul, Yeul(:,1))`.
- You can obtain a plot of y versus t with the command `plot(teul, Yeul(:,2))`.
- You can obtain a plot of y versus x with the command `plot(Yeul(:,1), Yeul(:,2))`.

Let's compare the Euler solution with the exact solution. Open the *Matlab M-File Editor/Debugger* and enter the following lines of code.

```
t=linspace(0,2*pi);
x=cos(t);y=sin(t);
[teul,Yeul]=eul('F',[0,2*pi],[1;0],0.5);
plot(x,y,Yeul(:,1),Yeul(:,2))
axis equal
legend('Exact','Euler')
set(gcf,'NumberTitle','off')
set(gcf,'Name','Comparing the Euler solution and exact solution')
```

Save the file as `exacteul.m`. Enter the command

```
>> exacteul
```

at the *Matlab* prompt to produce a nice comparison of the Euler and exact solutions.

4 RK2

The machinery is in place to easily create a solution using Dr. Polking's `rk2.m` routine, an improved Euler's method. Save the file

³In the examples that follow, the notation `yeul(:,1)` is read "every row, first column," and captures the x -values in the first column of `yeul`. Similarly, the notation `yeul(:,2)` is read "every row, second column," and captures the y -values in the second column of `yeul`.

```

t=linspace(0,2*pi);
x=cos(t);y=sin(t);
[trk2,Yrk2]=rk2('F',[0,2*pi],[1;0],0.5);
plot(x,y,Yrk2(:,1),Yrk2(:,2))
axis equal
legend('Exact','RK2')
set(gcf,'NumberTitle','off')
set(gcf,'Name','Comparing the RK2 solution and exact solution')

```

as `exactrk2.m` and enter the command

```
>> exactrk2
```

at the *Matlab* prompt to produce a nice comparison of the RK2 and exact solutions.

5 RK4

Of course, RK4 should produce the closest numerical approximation of the exact solution. Save the file

```

t=linspace(0,2*pi);
x=cos(t);y=sin(t);
[trk4,Yrk4]=rk4('F',[0,2*pi],[1;0],0.5);
plot(x,y,Yrk4(:,1),Yrk4(:,2))
axis equal
legend('Exact','RK4')
set(gcf,'NumberTitle','off')
set(gcf,'Name','Comparing the RK4 solution and exact solution')

```

as `exactrk4.m` and enter

```
>> exactrk4
```

at the *Matlab* prompt to produce a nice comparison of the RK4 and exact solutions.

6 Comparing the Solutions

Finally, the following code will create a nice visual comparison of the routines `eul.m`, `rk2.m`, and `rk4.m`.

```

>> plot(x,y,Yeul(:,1),Yeul(:,2),Yrk2(:,1),Yrk2(:,2),Yrk4(:,1),Yrk4(:,2))
>> axis equal
>> legend('Exact solution','Euler''s method','RK2 method','RK4 method')

```

Note that you need to use two single apostrophes to produce the apostrophe in the legend for the word `Euler's`.

Experiment by changing the step size by an equal amount in the files `exacteul.m`, `exactrk2.m`, and `exactrk4.m`.

7 Homework

1. Consider the following autonomous system.

$$\begin{aligned}\frac{dx}{dt} &= -3x - 4y \\ \frac{dy}{dt} &= 4x - 3y \\ x(0) &= 1 \\ y(0) &= 0\end{aligned}\tag{22}$$

The goal in this problem is to examine the solution with initial condition $x(0) = 1$ and $y(0) = 0$.

- (a) Use *Matlab's* `dsolve` command to find the exact solution of system (22) with initial condition $x(0) = 1$ and $y(0) = 0$.
- (b) Find numeric solutions using Euler's method, RK2, and RK4. Use a step size of $h = 0.1$ in each case.
- (c) Create the following plots. Print them and hand them in next class.
 - i. Plot the exact solution of system (22) three ways: x versus t , y versus t , and y versus x . Arrange each of the plots in the same figure window with the `subplot` command. Give each subplot an appropriate title. Turn the figure number off and rename the figure window appropriately.
 - ii. On the same axes, plot the graphs of the exact, Euler, RK2, and RK4 solutions. Include a legend with your figure. Turn the figure number off and rename the figure window appropriately.