

ME 406

Assignment #2 Solutions

```
In[299]:=
  sysid
  Mathematica 6.0.3, DynPac 11.01, 1/20/2009
```

```
In[300]:=
  plotreset; intreset;
```

In this notebook, we give for many of the problems both an analytical solution and a solution using either straight *Mathematica* or DynPac. In your solutions, you may use any method you like unless the problem statement specifies a particular method.

All of the calculations in this notebook refer to the system

$$\dot{x} = 3y, \quad \dot{y} = x + 2y - 2.$$

We begin by defining this system for DynPac.

```
In[301]:=
  setstate[{x, y}];

In[302]:=
  setparm[{}];

In[303]:=
  slopevec = {3 y, x + 2 y - 2};
```

■ Problem 1

(a) We find the equilibrium point or points by setting the slope vector to zero, which gives in this case $x = 2, y = 0$. For more complicated nonlinear systems, it is sometimes more difficult to find the equilibria. For polynomial slope vectors, the command `findpolyeq` is helpful. To illustrate its use, we apply it here.

```
In[304]:=
  findpolyeq

Out[304]=
  {{2, 0}}
```

We get the same result. Now we make a translational change of coordinates to put this equilibrium at the origin in the new coordinates. To do this by hand, we would substitute in the slope vector $\tilde{x} = x - 2, \tilde{y} = y$. Then for simplicity we would drop the tildas. The resulting slopevector is then $\{3y, x + 2y\}$. We also can achieve this by the DynPac command `transorg[vec]`, which puts the new origin at `vec`.

```
In[305]:=
  transorg[{2, 0}];
```

We check this by looking at the new slope vector.

```
In[306]:=
  slopevec

Out[306]=
  {3 y, x + 2 y}
```

■ Problem 2

(a) By dividing the second equation by the first, we get

$$\frac{dy}{dx} = \frac{x + 2y}{3y}.$$

We try $y = kx$ in this equation to get

$$k = \frac{1 + 2k}{3k},$$

hence $k = -1/3, 1$. For $k = -1/3$, the first of the original equations gives us $\dot{x} = -x$, with solution $x(t) = \text{constant } e^{-t}$, $y = -(1/3)x$. The solution of the system for this value of k is

$$\begin{pmatrix} x \\ y \end{pmatrix} = C_1 \begin{pmatrix} 3 \\ -1 \end{pmatrix} e^{-t}.$$

In the same way, we find for $k = 1$ a solution

$$\begin{pmatrix} x \\ y \end{pmatrix} = C_2 \begin{pmatrix} 1 \\ 1 \end{pmatrix} e^{3t}.$$

We also can get these results with *Mathematica*. The form $y = kx$ means that the second component of the slope vector is k times the first. Thus

```
In[307]:=
  Solve[k == (slopevec[[2]] / slopevec[[1]]) /. {y -> k * x}, {k}]

Out[307]=
  {{k -> -1/3}, {k -> 1}}
```

These two orbits cross, but because the crossing point is an equilibrium, this does not violate the result obtained in class.

(b) The general solution is the sum of the two solutions we found above.

$$\begin{pmatrix} x \\ y \end{pmatrix} = C_1 \begin{pmatrix} 3 \\ -1 \end{pmatrix} e^{-t} + C_2 \begin{pmatrix} 1 \\ 1 \end{pmatrix} e^{3t}.$$

We also may get this from *Mathematica*, by using DSolve.

```
In[308]:=
```

```
DSolve[{x'[t] == 3 y[t], y'[t] == x[t] + 2 y[t]}, {x[t], y[t]}, t]
```

```
Out[308]=
```

$$\left\{ \left\{ \begin{aligned} x[t] &\rightarrow \frac{1}{4} e^{-t} (3 + e^{4t}) C[1] + \frac{3}{4} e^{-t} (-1 + e^{4t}) C[2], \\ y[t] &\rightarrow \frac{1}{4} e^{-t} (-1 + e^{4t}) C[1] + \frac{1}{4} e^{-t} (1 + 3 e^{4t}) C[2] \end{aligned} \right\} \right\}$$

These two forms are equivalent, although DSolve has given us a different solution basis than the one we found.

(c) Because some of the solutions are growing in time, the equilibrium is unstable. Any solution with a nonzero value of C_2 will head off to infinity. We may also determine the stability using the DynPac function `classify2D`, applied to the equilibrium point.

```
In[309]:=
```

```
classify2D[{0, 0}]
```

```
Abbreviations used in classify2D.
```

```
L = linear, NL = nonlinear, R2 = repeated root.
```

```
Z1 = one zero root, Z2 = two zero roots.
```

```
This message printed once.
```

```
unstable - saddle
```

We are told that the solution is unstable, and that it is a saddle point, a term that we will explain in more detail very soon.

(d) We start with the general solution we found at the beginning of part (b)

$$\begin{pmatrix} x \\ y \end{pmatrix} = C_1 \begin{pmatrix} 3 \\ -1 \end{pmatrix} e^{-t} + C_2 \begin{pmatrix} 1 \\ 1 \end{pmatrix} e^{3t}.$$

We impose the initial condition:

$$\text{at } t = 0, \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix} = C_1 \begin{pmatrix} 3 \\ -1 \end{pmatrix} + C_2 \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

We solve for C_1 and C_2 to get $C_1 = 0$ and $C_2 = 1$, hence

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix} e^{3t}.$$

We also can get this directly from *Mathematica* via DSolve:

```
In[310]:= DSolve[{x'[t] == 3 y[t], y'[t] == x[t] + 2 y[t], x[0] == 1, y[0] == 1}, {x[t], y[t]}, t]
```

```
Out[310]= {{x[t] -> e^{3 t}, y[t] -> e^{3 t}}}
```

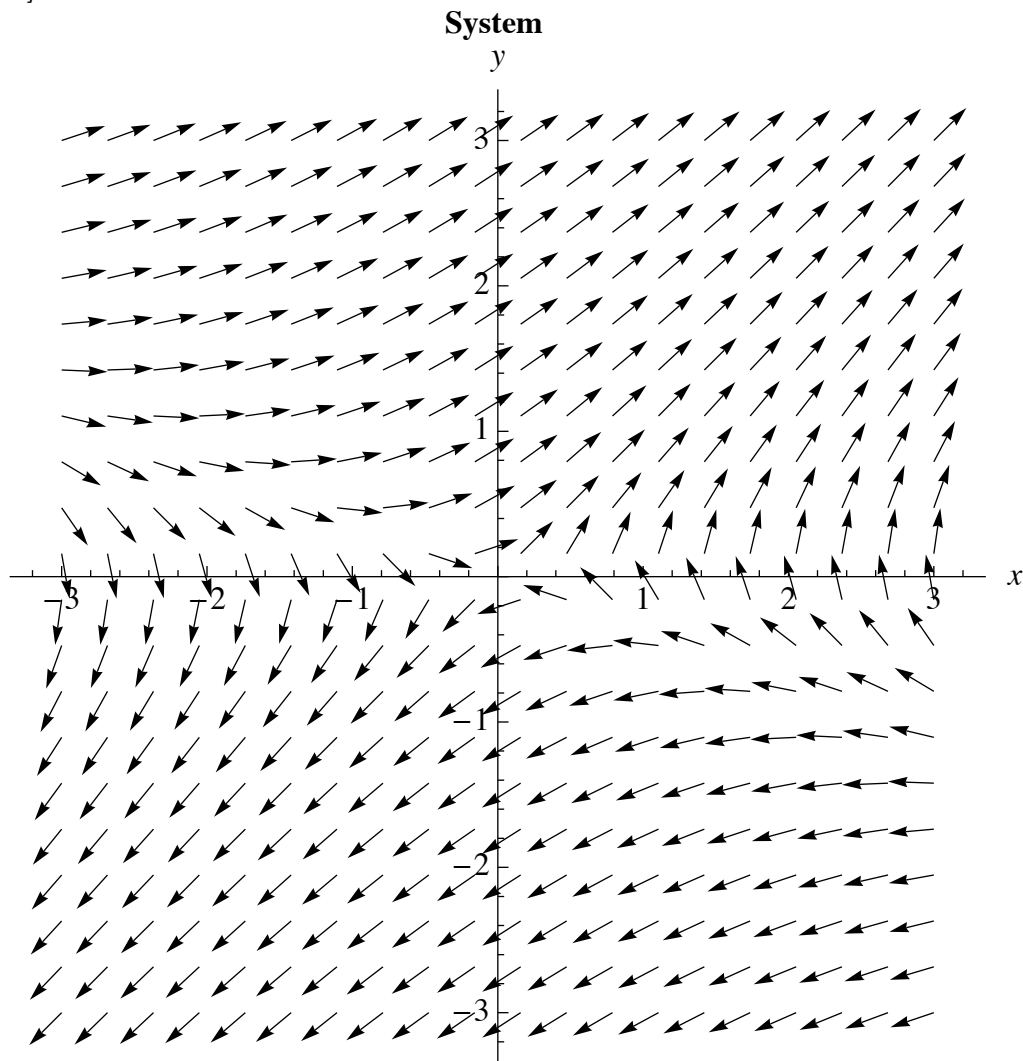
■ Problem 3

(a) We use the DynPac function `dirfield` to plot a direction field. The plotting window is assigned to the variable `plrange`.

```
In[311]:= plrange = {{-3, 3}, {-3, 3}};
```

```
In[312]:= graph1 = dirfield
```

```
Out[312]=
```



We see that near the origin, some of the direction vectors point away, and this suggests instability.

(b) In constructing the integral curves, it is convenient to integrate both directions in time to get a more complete picture of each orbit. We ask for this by setting a flag:

```
In[313]:=  
  bothdirflag = True;
```

Because the solution can grow exponentially in time, we must guard against overflows. We do this by using range checking. To turn on range checking, we set a flag:

```
In[314]:=  
  rangeflag = True;
```

Now integrations will proceed until they hit the edge of a window that we assign to the variable ranger.

```
In[315]:=  
  ranger = plrange;
```

By setting `ranger = plrange`, we have made the range-checking window the same as the plotting window.

Now we choose the integration parameters -- the step size, which we call `h`, the number of steps, which we call `nsteps`, and the initial time, which we call `t0`.

```
In[316]:=  
  h = 0.02; nsteps = 500; t0 = 0;
```

We carry out four integrations, starting at four different initial conditions.

```
In[317]:=  
  sol1 = integrate[{1, 0}, t0, h, nsteps];
```

```
In[318]:=  
  sol2 = integrate[{0, 1}, t0, h, nsteps];
```

```
In[319]:=  
  sol3 = integrate[{-1, 0}, t0, h, nsteps];
```

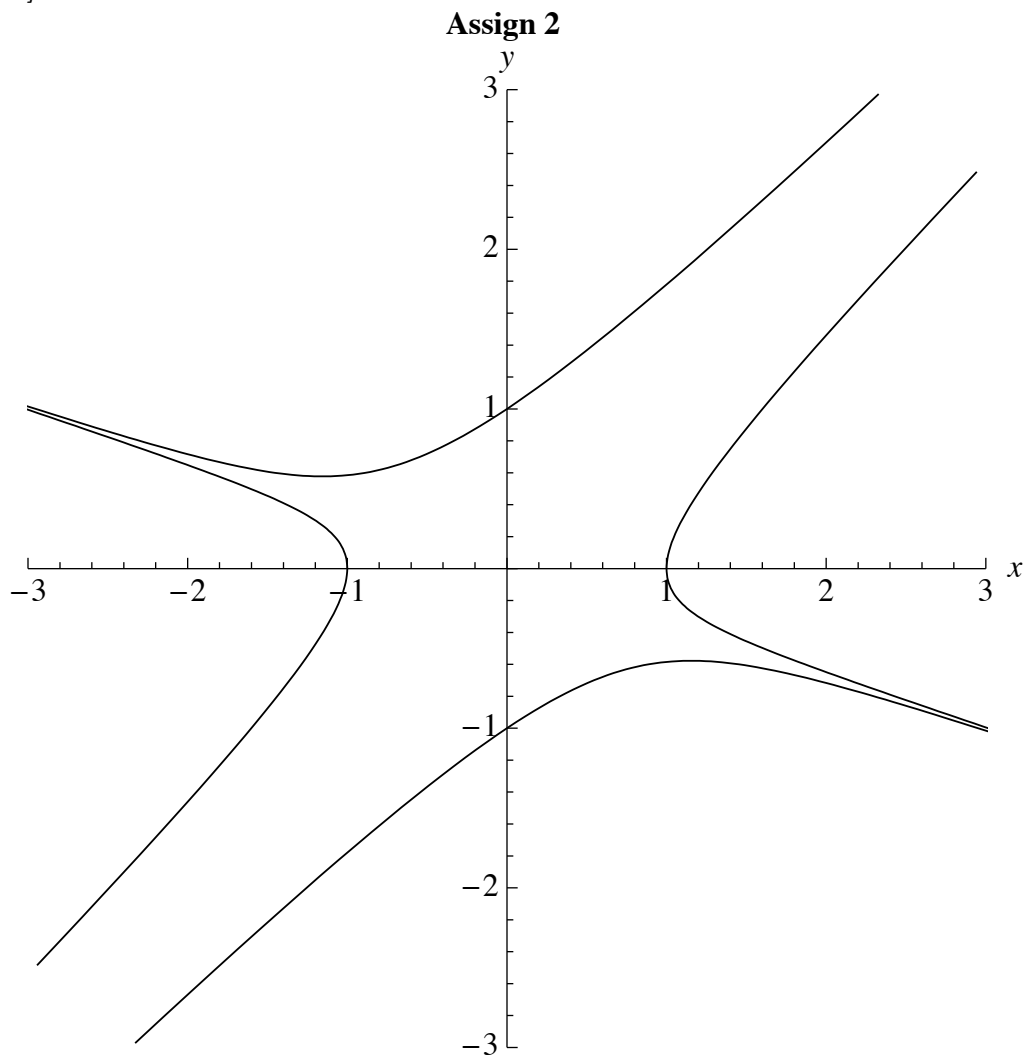
```
In[320]:=  
  sol4 = integrate[{0, -1}, t0, h, nsteps];
```

Now we make a phase plane plot of all of these. First we name the system "Assign 2."

```
In[321]:=  
  sysname = "Assign 2";
```

```
In[322]:=  
graph2 = phaser[{sol1, sol2, sol3, sol4}]
```

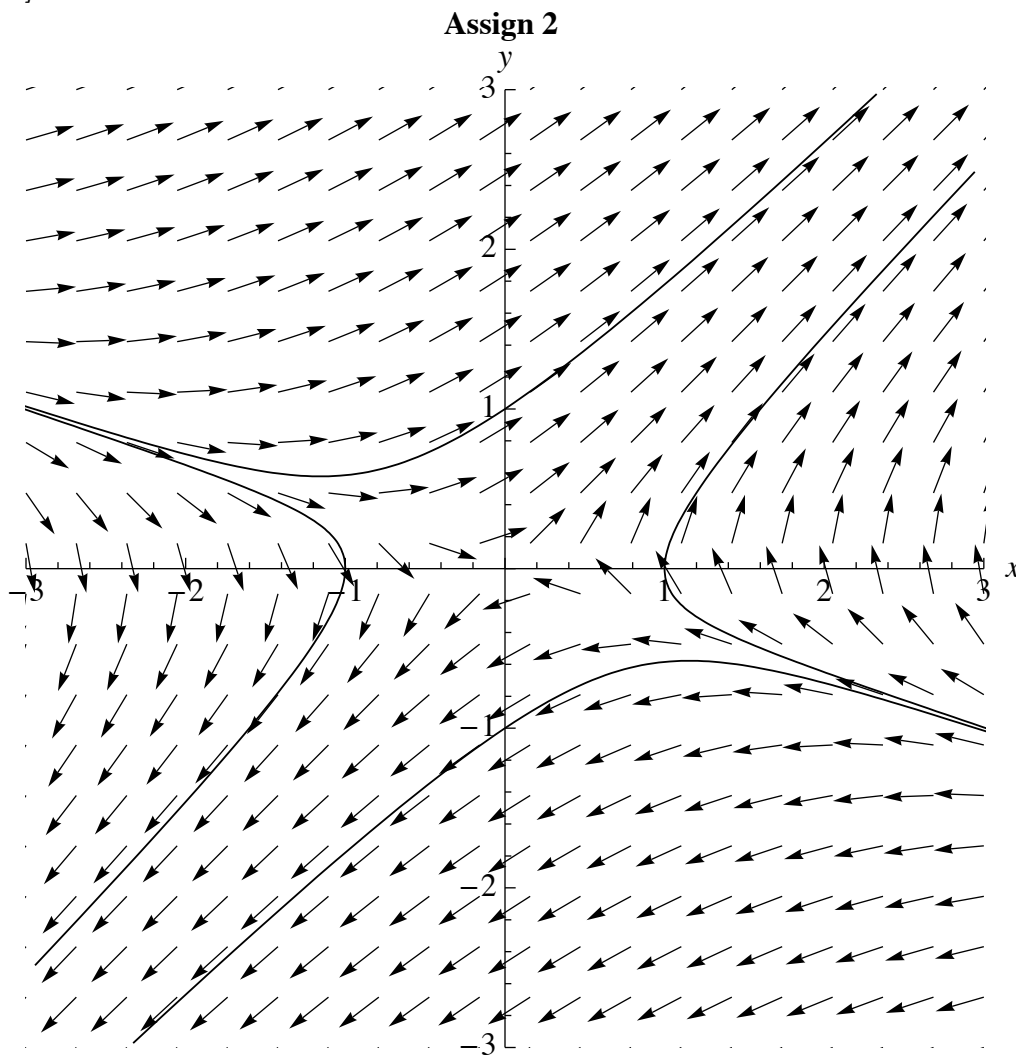
```
Out[322]=
```



Now we combine these plots with the direction field.

```
In[323]:=
  show[graph1, graph2]
```

```
Out[323]=
```



(c) To get a straight line solution, we need to pick an initial point which is on the straight line solution. We start with the solution corresponding to $k = 1$. For that solution, $y = x$. We choose $\{1,1\}$ for an initial point, and continue to integrate both directions in time. This gives us the portion of the orbit in the first quadrant. Then we choose a point on the opposite side of the origin, namely $\{-1,-1\}$, to get the portion of the orbit in the third quadrant.

```
In[324]:=
  sol5 = integrate[{1, 1}, t0, h, nsteps];
```

```
In[325]:=
  sol6 = integrate[{-1, -1}, t0, h, nsteps];
```

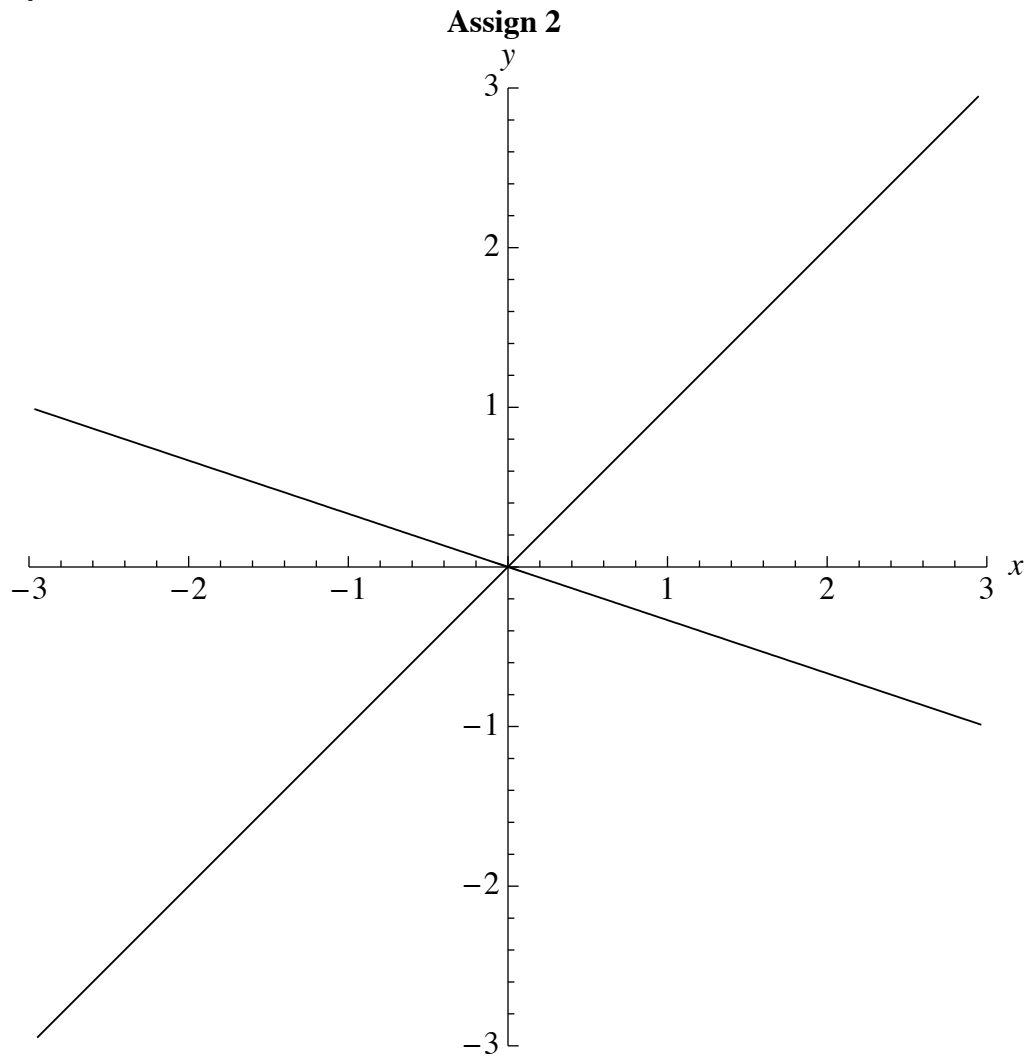
Now the other solution, corresponding to $k = -1/3$, hence $x = -3y$.

```
In[326]:=
  sol7 = integrate[{-1.5, 0.5}, t0, h, nsteps];
```

```
In[327]:=  
sol8 = integrate[{1.5, -0.5}, t0, h, nsteps];
```

```
In[328]:=  
graph3 = phaser[{sol5, sol6, sol7, sol8}]
```

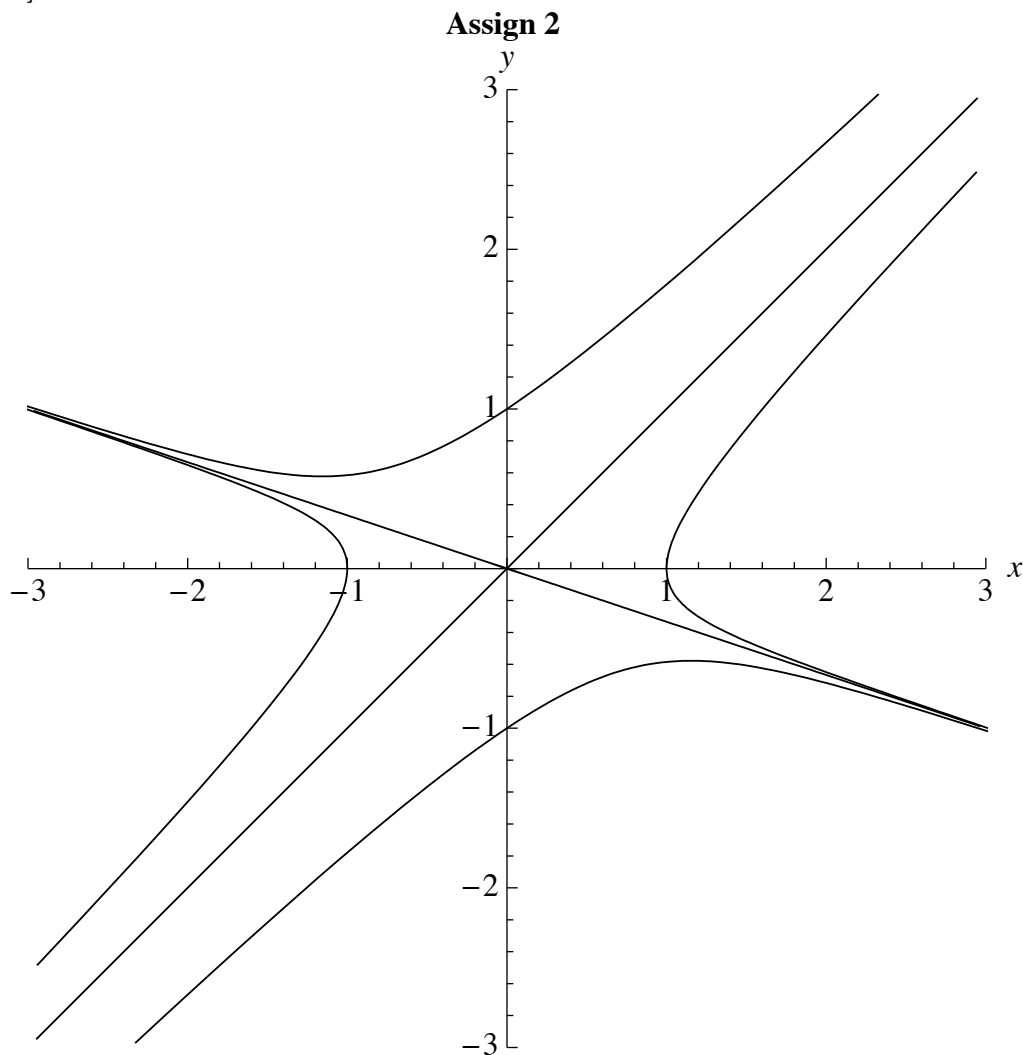
```
Out[328]=
```



(d) Now we combine all of our integral curves into a single phase portrait.

```
In[329]:=
  show[graph2, graph3]
```

```
Out[329]=
```



It would be nice to have arrows on the plots. We can do this by setting arrowflag:

```
In[330]:=
  arrowflag = True;
```

This tells DynPac to put arrows in the direction of increasing time on each integral curve. We may place the arrows at any fraction of the arc length of the curve that we like. Such fractions are specified by assignments to the variable arrowvec. In this case we choose to put the arrows at the mid-point of each curve, by specifying

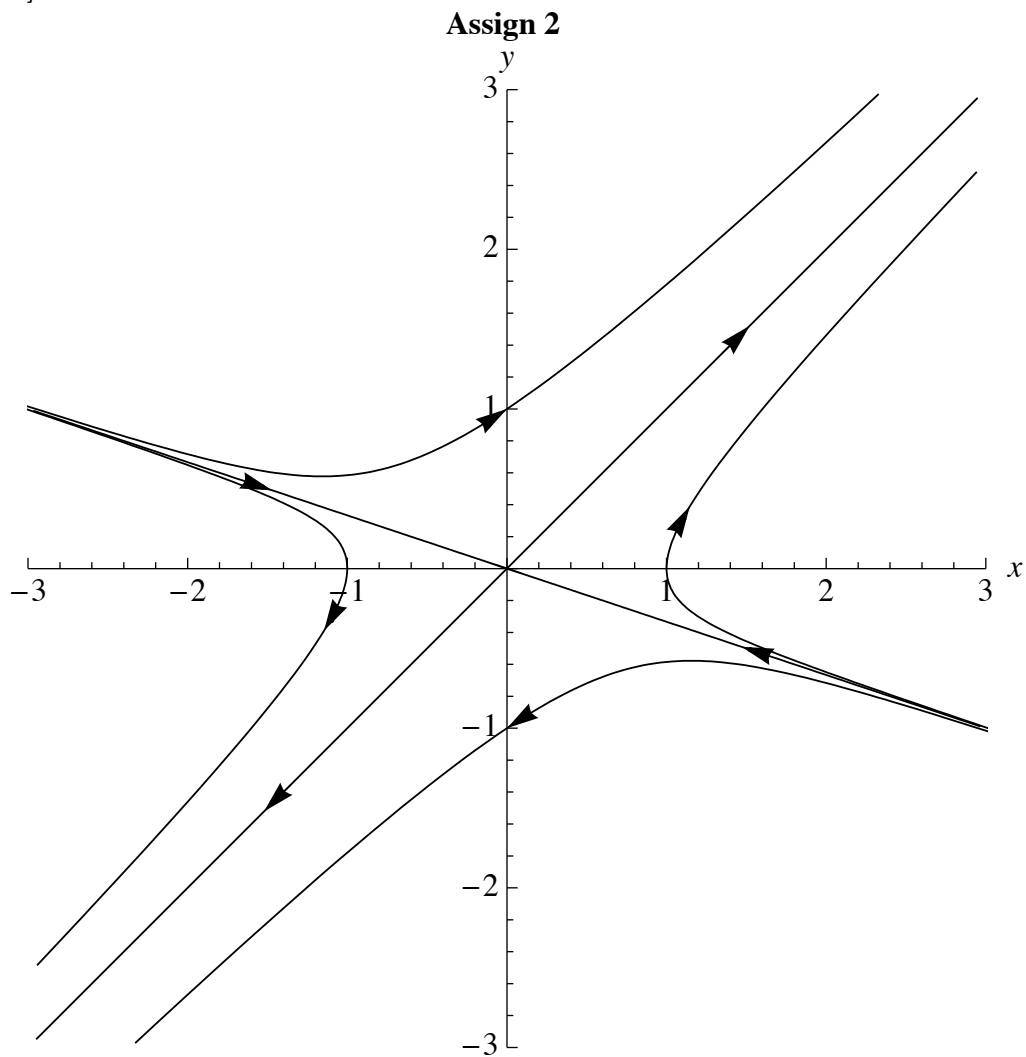
```
In[331]:=
  arrowvec = {1 / 2};
```

Now we re-graph our solutions:

```
In[332]:=
```

```
phaser[{sol1, sol2, sol3, sol4, sol5, sol6, sol7, sol8}]
```

```
Out[332]=
```



(e) Much of what we have just done can be automated. The function `saddleportrait` will do all of this for us. It takes two arguments: the location of the equilibrium point ($\{0,0\}$ in this case) and the plotting window (`prange={{-3,3},{-3,3}}` in this case).

```
In[333]:=  
saddleportrait[{0, 0}, plrange]
```

```
Out[333]=
```

