

# Damped Free Linear Oscillator

```
In[419]:=
  sysid
  Mathematica 5.2.0, DynPac 10.71, 9/8/2005
```

```
In[420]:=
  plotreset; intreset;
```

## ■ INTRODUCTION

In this notebook, we use DynPac to look at some solutions of the damped free linear oscillator, including our first look at the phase plane and a direction field in this notebook. The basic equation is

$$m \frac{d^2 x}{dt^2} + c \frac{dx}{dt} + kx = 0.$$

By introducing the velocity  $y$  as a new variable, we can write this as a system of two first-order equations:

$$\begin{aligned} \frac{dx}{dt} &= y, \\ \frac{dy}{dt} &= -\frac{k}{m}x - \frac{c}{m}y. \end{aligned}$$

Now we define this system for DynPac, by specifying the state vector, the slope vector, and the parameter vector.

## ■ DEFINING THE SYSTEM FOR DYNPAC

```
In[421]:=
  setstate[{x, y}];

In[422]:=
  setparm[{m, c, k}];

In[423]:=
  slopevec = {y, -(k/m)*x - (c/m)*y};
```

We name our system Oscillator.

```
In[424]:=
  sysname = "Oscillator";
```

Now we specify the parameter values by setting parmval.

```
In[425]:=
  parmval = {100.0, 200.0, 1600.0};
```

Finally we set the initial time  $t_0$ , the time step  $h$ , and the number of time steps  $nsteps$ .

```
In[426]:=
  t0 = 0.0;
```

```
In[427]:=
  h = 0.025;
```

```
In[428]:=
  nsteps = 200;
```

We are going to solve this equation with three different sets of initial conditions. We start by defining the initial conditions.

```
In[429]:=
  init1 = {1.0, 0.0};
```

```
In[430]:=
  init2 = {0.0, 4.0};
```

```
In[431]:=
  init3 = {1.0, 4.0};
```

We construct the solutions by using the routine `integrate`.

## ■ INTEGRATION

```
In[432]:=
  sol1 = integrate[init1, t0, h, nsteps];
```

```
In[433]:=
  sol2 = integrate[init2, t0, h, nsteps];
```

```
In[434]:=
  sol3 = integrate[init3, t0, h, nsteps];
```

## ■ TIME PLOTS

We will plot  $x$  and  $y$  versus time for each of these solutions. We ask for an aspect ratio of 0.5 by setting the variable `asprat` to 0.5. The function `timeplot` constructs a plot of any component of a solution versus time. The second argument of `timeplot` is the number of the component(s) to be plotted.

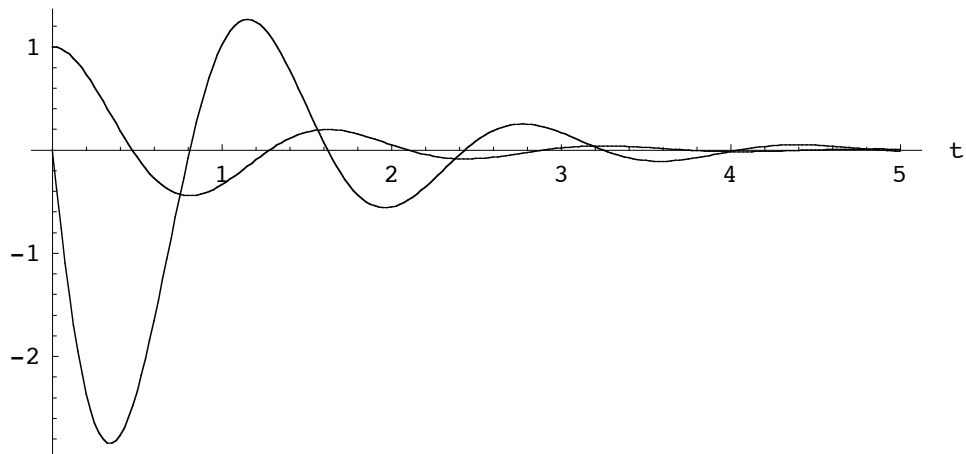
```
In[435]:=
  asprat = 0.5;
```

```
In[436]:=
  imsize = 380;
```

```
In[437]:=
```

```
timegraph1 = timeplot[sol1, {1, 2}];
```

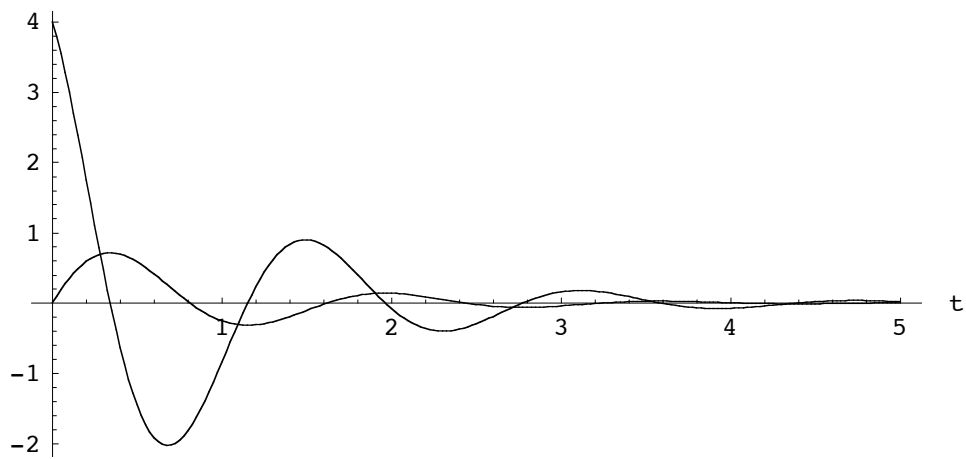
```
{x, y} Oscillator {m, c, k} = { 100.00, 200.00, 1600.00}
```



```
In[438]:=
```

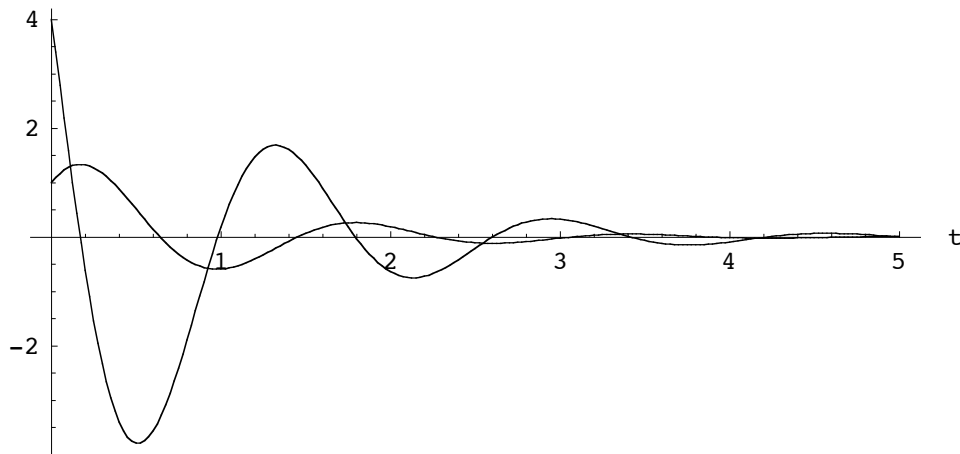
```
timegraph2 = timeplot[sol2, {1, 2}];
```

```
{x, y} Oscillator {m, c, k} = { 100.00, 200.00, 1600.00}
```



```
In[439]:=
timegraph3 = timeplot[sol3, {1, 2}];

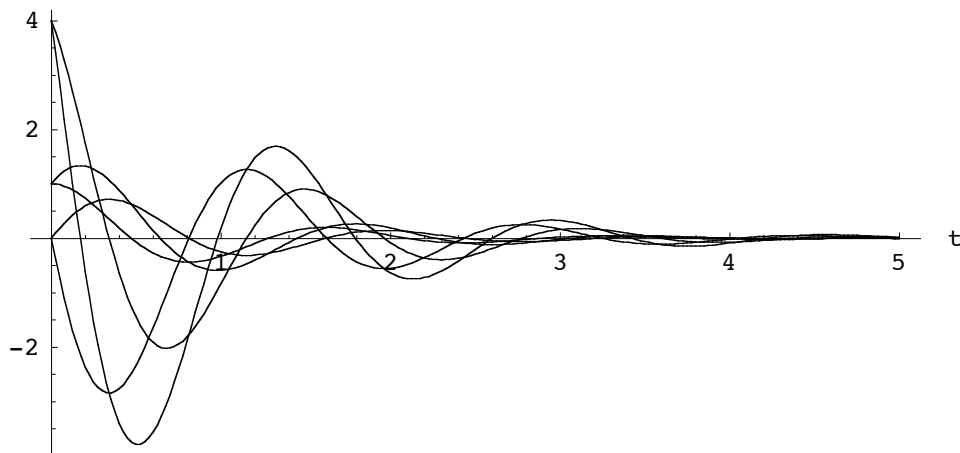
{x, Y} Oscillator {m, c, k}={ 100.00, 200.00, 1600.00}
```



We can view all three of these solutions on the same graph by using the show command:

```
In[440]:=
show[timegraph1, timegraph2, timegraph3];

{x, Y} Oscillator {m, c, k}={ 100.00, 200.00, 1600.00}
```



This last graph is hopelessly busy. Clearly this is not the way to visualize simultaneously three different solutions of this equation. The phase plane, on the other hand, is ideal for this. We now construct the phase plane plots of all three solutions, and then combine them into a single plot. We switch the aspect ratio back to 1.0 first, and we set a plotting window of  $\{-1, 1\}, \{-4, 4\}$ , by assigning this range to `plrange`.

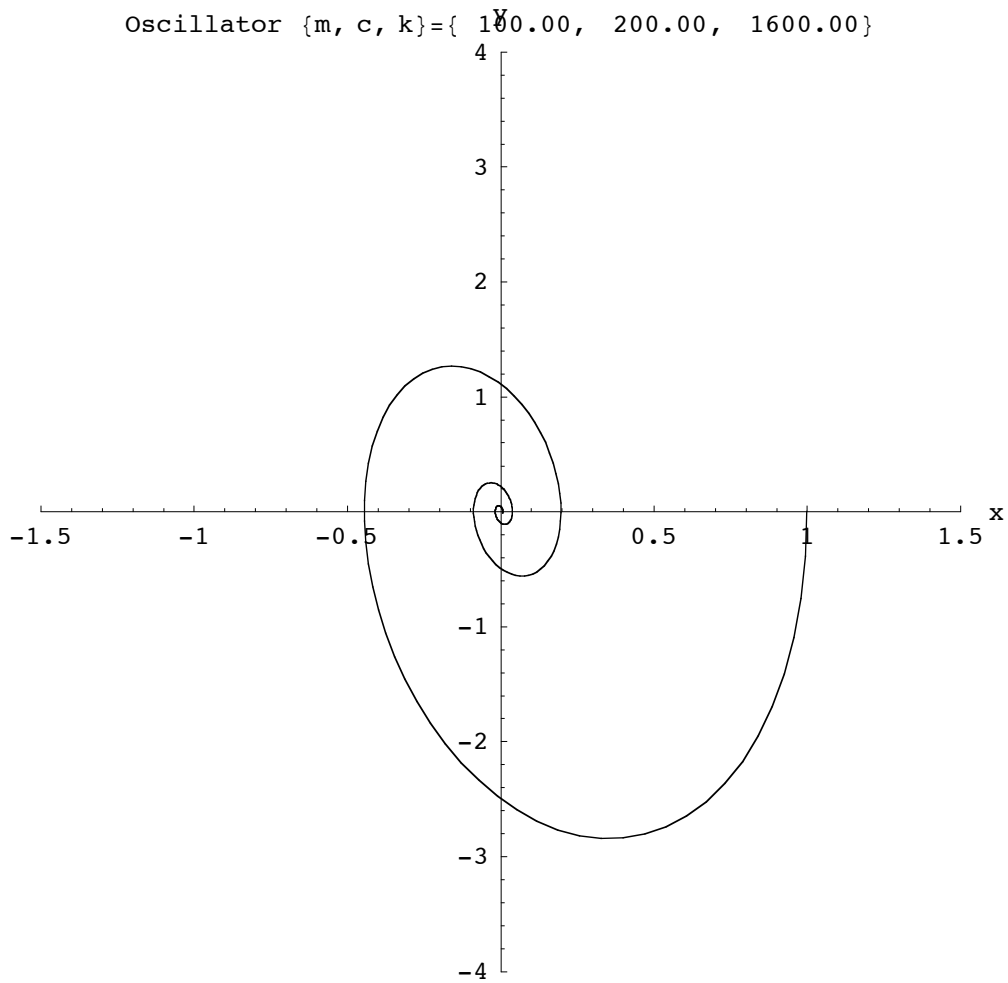
## ■ PHASE PLOTS

```
In[441]:=
plrange = {{-1.5, 1.5}, {-4, 4}};
```

```
In[442]:=
asprat = 1.0;
```

```
In[443]:=
```

```
phasegraph1 = phaseplot[sol1, 1, 2];
```

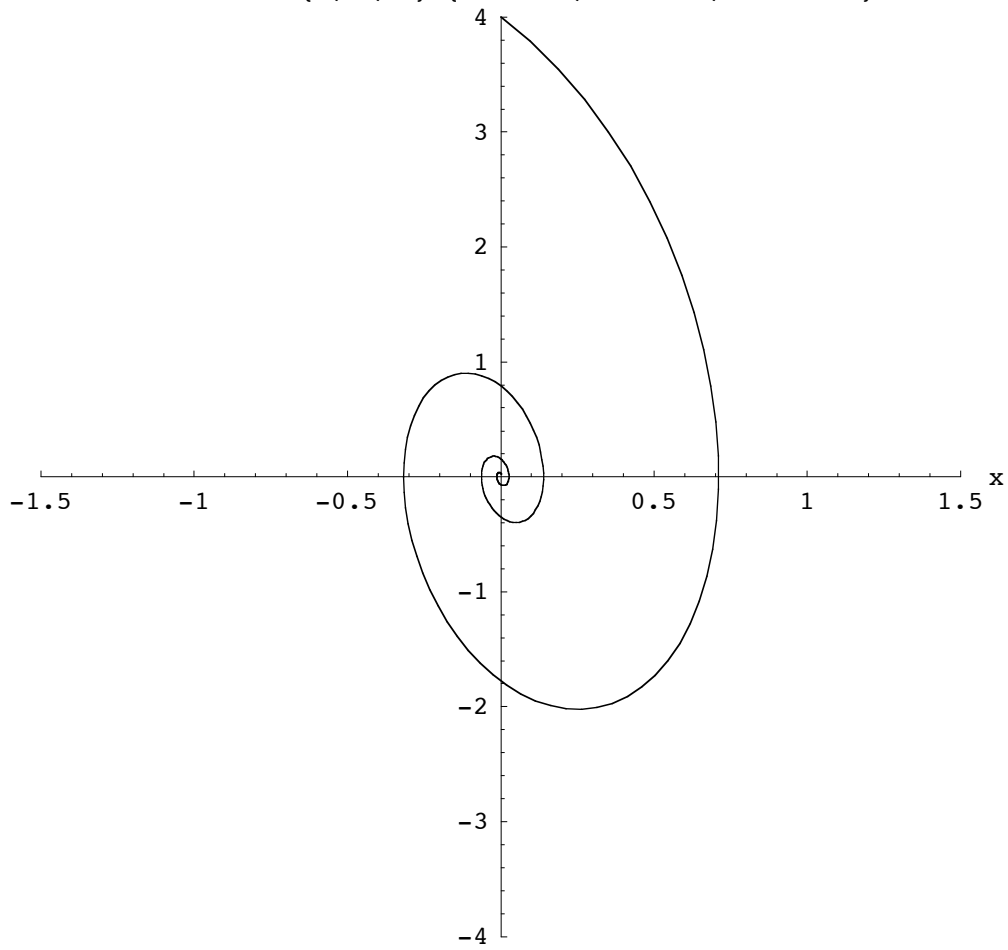


The collision of the axis label and the plot label is an unfortunate bug of long-standing in *Mathematica*.

```
In[444]:=
```

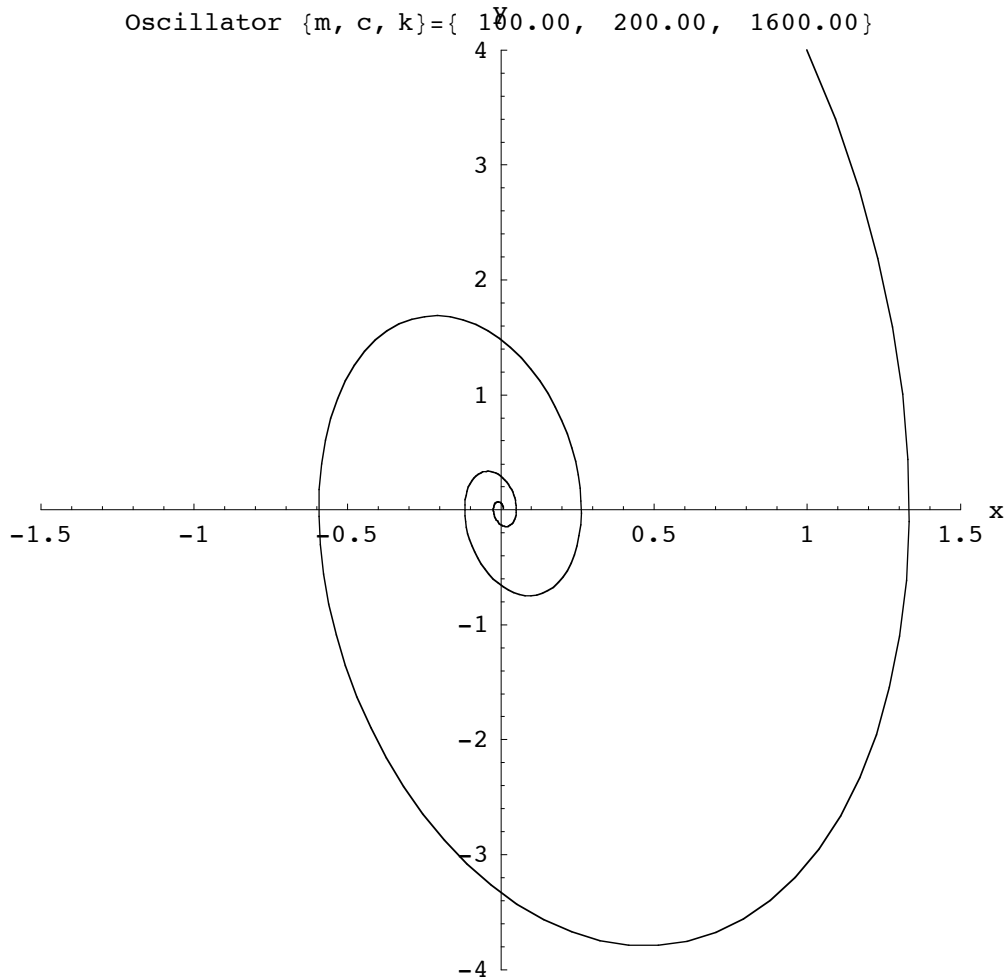
```
phasegraph2 = phaseplot[sol2, 1, 2];
```

```
Oscillator {m, c, k} = { 100.00, 200.00, 1600.00 }
```



```
In[445]:=
```

```
phasegraph3 = phaseplot[sol3, 1, 2];
```

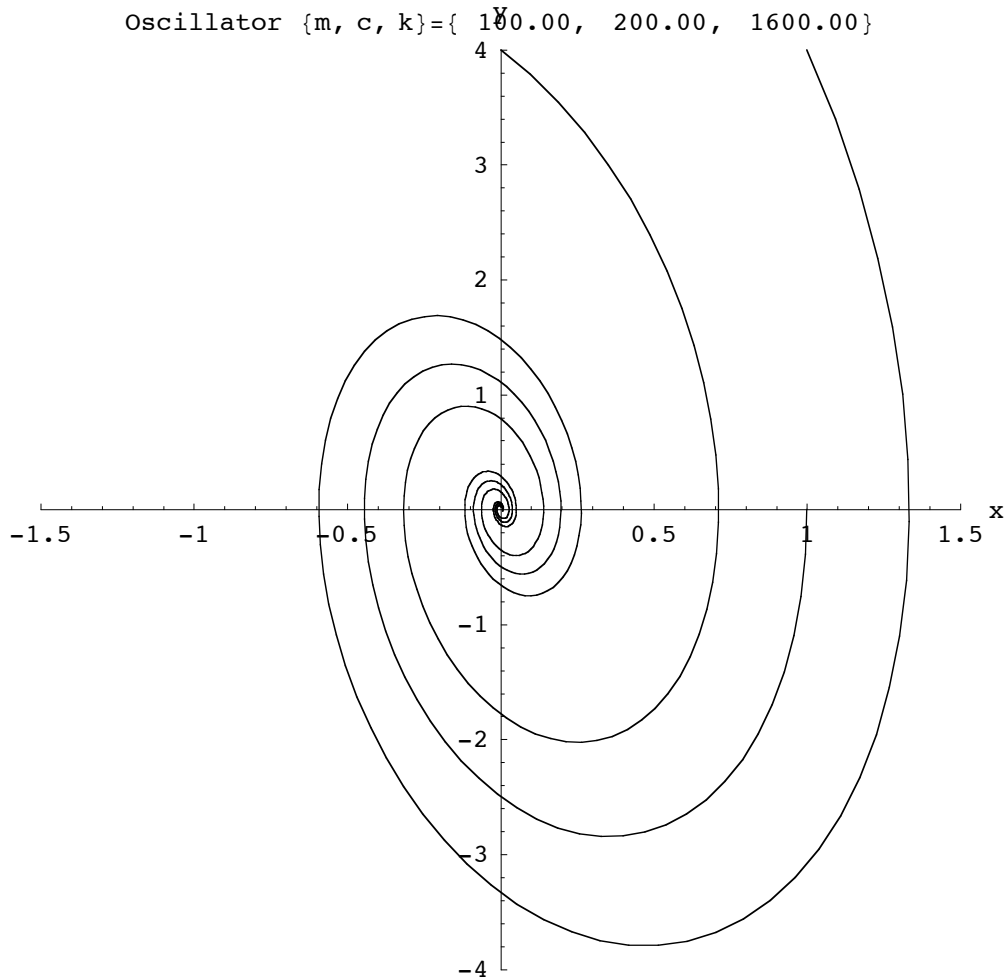


Notice that the solutions are topologically similar. No matter where we start, we spiral into the origin. That is a physically obvious result in this case, because we have an oscillator which is damped and not driven, so no matter where we start, we end up in a state of rest, which is the origin in the phase plot.

Now we combine the three phase plots into a single plot.

```
In[446]:=
```

```
bigphase = show[phasegraph1, phasegraph2, phasegraph3];
```



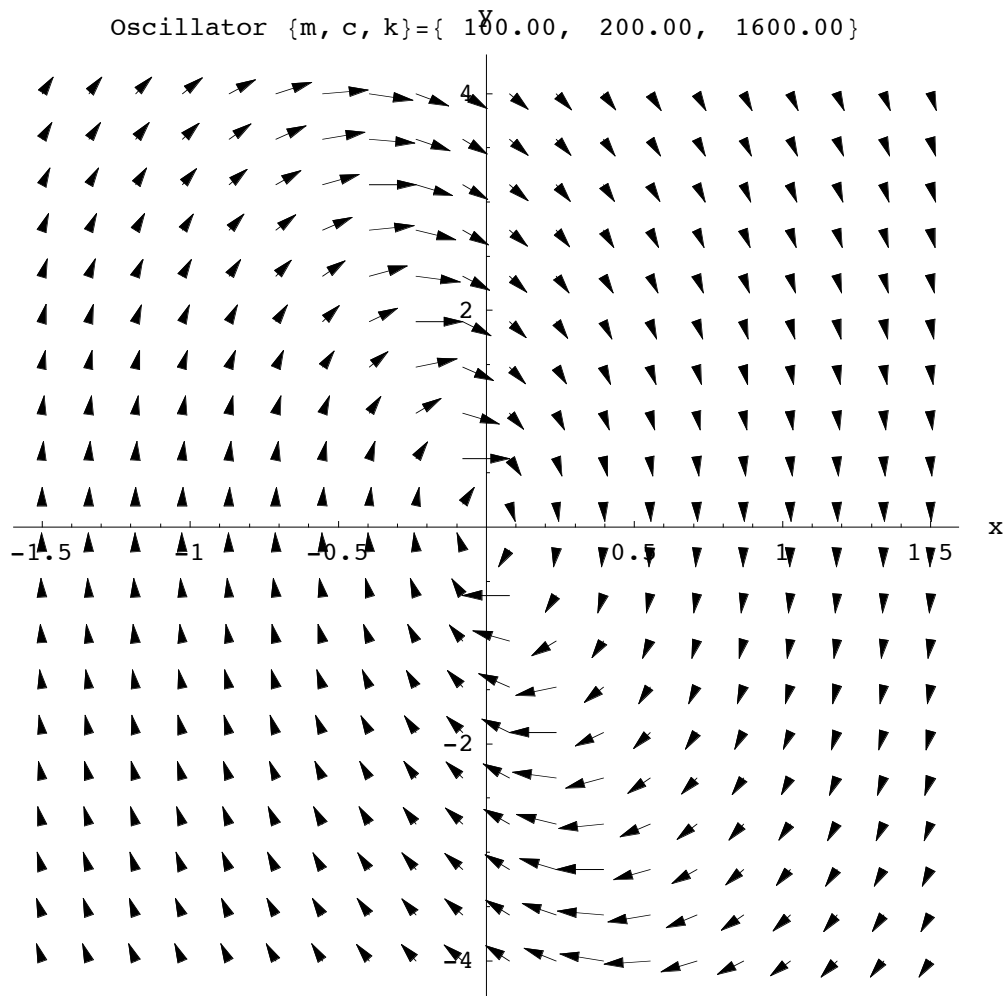
We see that this plot allows easily accomodates multiple solutions without confusion. In fact the plot shows clearly the general feature of this system: every initial condition is eventually attracted to the point at the origin. Such a point is called an attractor for the system.

## ■ DIRECTION FIELD

At each point in the phase plane, the solution has a tangent vector uniquely determined by the right-hand sides of the two equations. From a geometric point of view, the process of solving the equation amounts to connecting these little slope vectors smoothly. By plotting these slope vectors on a grid of points, we can get a good picture of what the solution looks like without actually integrating. Let's try this for our present example. We use the DynPac routine `dirfield` which automatically constructs a direction field for us. The window used by `dirfield` is stored in the variable `plrange`, which we have set earlier.

```
In[447]:=
```

```
dirgraph = dirfield;
```



As a final illustration, we superpose our direction field and our three solutions.

```
In[448]:=  
finale = show[bigphase, dirgraph];
```

