First-Order Differential Equations and Slope Fields

Introduction

OBJECTIVE: Visualize the slope fields and solution curves for selected first-order differential equations.

This module contains a special command that plots slope fields and selected solution curves for first-order differential equations. You can use it to obtain solutions for related problems in the text. In addition, you can use the slope field command to study a wide variety of first-order differential equations and to analyze the long-term behavior of solutions.

Technology Guidelines

NOTE: If you have just finished a module, restart *Mathematica* or close the *Kernel* before executing a new module.
TO OPEN CELLS, put your cursor on the right cell bracket and double click.
INITIALIZATION CELLS
When asked if you want to ". . . automatically evaluate all the initialization cells in the notebook . . . ," respond by pressing the "Yes" button.
TO STOP AN EXECUTION
Select the *Kernel* pull-down menu and click on *Abort Evaluation.*
ORDER OF EXECUTION
Execute cells in the order given. Do not skip any Input cells within a given notebook.
SAVING NOTEBOOKS
You can save anytime to any directory you choose, and it is wise to save often.
However, before you do your final save, delete all your output by selecting the *Delete All Output* selection under the *Kernel* pull-down menu.
EXPERIENCING MAJOR PROBLEMS
Save if appropriate, then shut down *Mathematica* and start it up again.

Part I - Drawing Slope Fields and Solution Curves

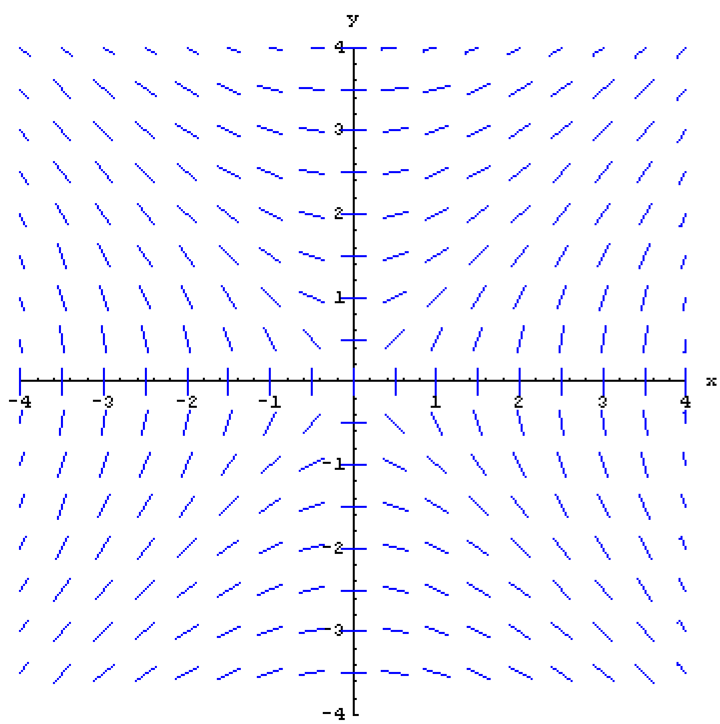
The **slopefield[f_, x_, y_, xmin_, xmax_, ymin_, ymax_, initconditions_(optional)]** command in the next cell generates a slope field and draws selected solution curves for first-order differential equations of the form $\frac{dy}{dx}=f(x,y)$. The arguments of the command are **f** (the right-hand side function), f(x,y), **x** (the independent variable), **y** (the unknown function), **xmin**, **xmax**, **ymin**, and **ymax** (the bounds on the slope field), and **initconditions** (an optional list of initial conditions). If the list of conditions is not included, only the slope field is drawn. Here's how it works for the differential equation $\frac{dy}{dx}=\frac{x}{y}$. First, we plot only the slope field.

In[7]:=

```
Clear[x, y, f]

f = x/y

slopefield[f, x, y, -4, 4, -4, 4];
```

Out[8]=

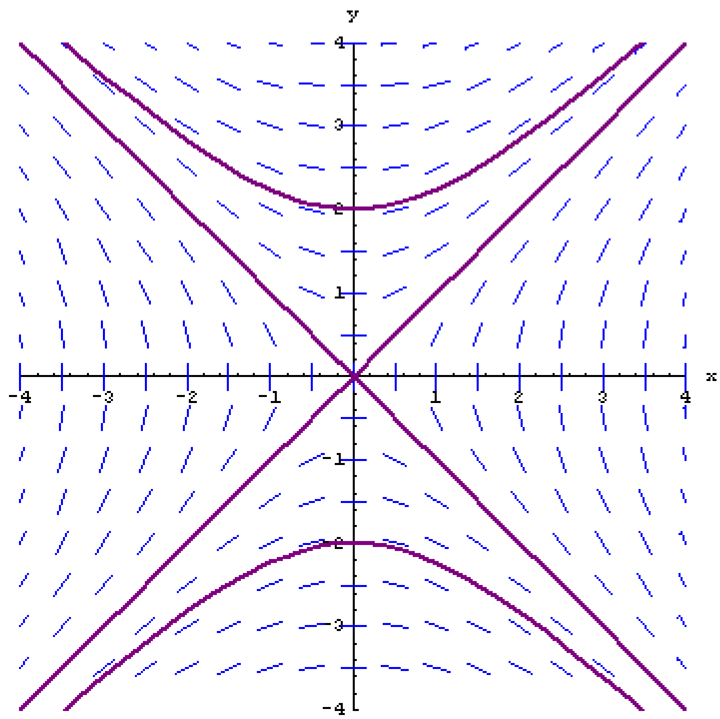
$$\frac{x}{y}$$



Now we include a list of initial conditions to obtain some solution curves. Each of the four initial conditions will usually give a different solution curve.

In[10]:=

```
slopefield[f, x, y, -4, 4, -4, 4,
 {y[0] == 2, y[1] == 1, y[1] == -1, y[0] == -2}];
```



There is a good likelihood that you will get error messages for some initial conditions. For example, the solution curve that passes through the point (1,1/2) cannot be expressed as a function when *y* is a function of *x*. Here's what happens.

In[11]:=

```
slopefield[x/y, x, y, -4, 4, -4, 4, {y[1] == 1/
```

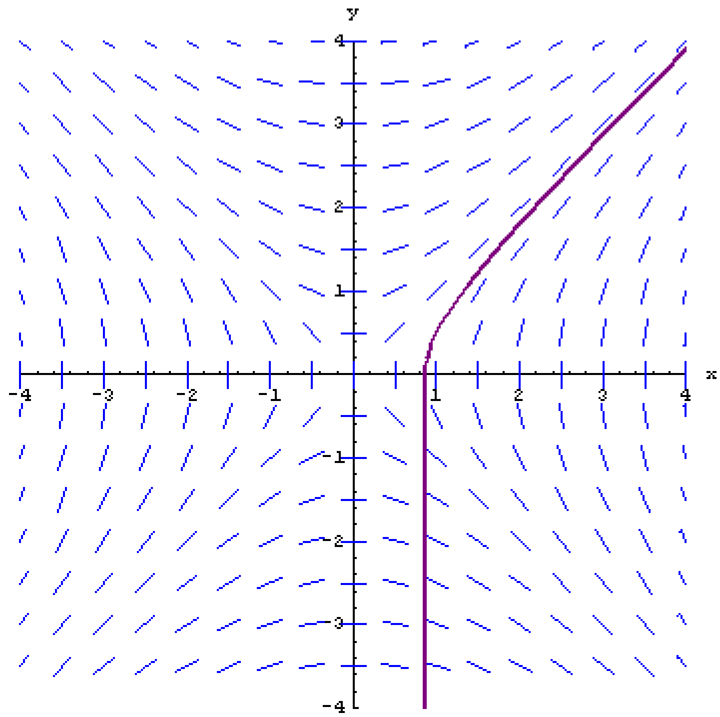
```
NDSolve::ndsz :
 At xloc == 0.8660255267077515`, step
   size is effectively zero; singularity
   or stiff system suspected. More…
```

```
InterpolatingFunction::dmval :
 Input value {-4.} lies outside the range
   of data in the interpolating function.
   Extrapolation will be used. More…
```

```
InterpolatingFunction::dmval :
 Input value {-3.67546} lies outside
   the range of data in the interpolating
   function. Extrapolation will be used. More…
```

```
InterpolatingFunction::dmval :
 Input value {-3.32153} lies outside
   the range of data in the interpolating
   function. Extrapolation will be used. More…
```

```
General::stop : Further output of
   InterpolatingFunction::dmval will be
   suppressed during this calculation. More…
```



When this happens, it is probably better to hand sketch some representative solution curves on the slope field.

You Try It: Part I

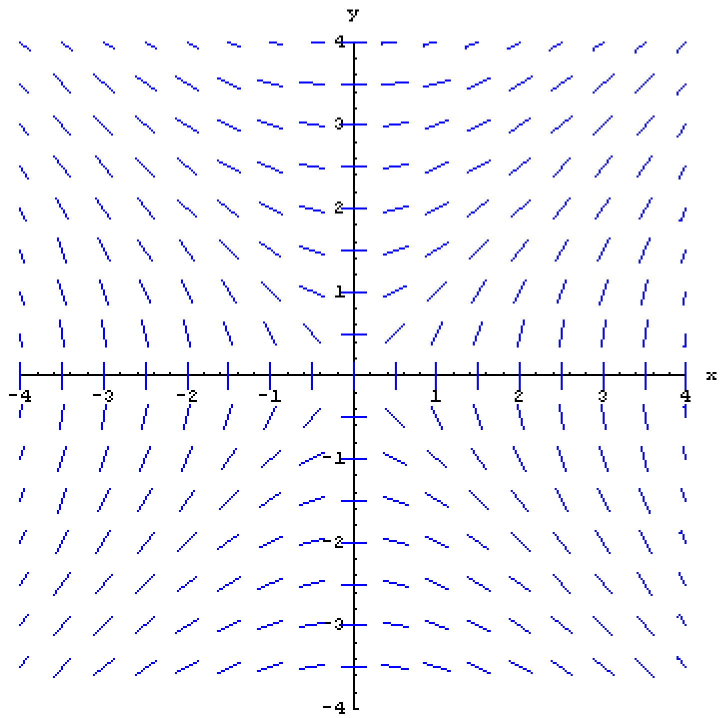
It's fun! Make up some of your own differential equations of the form $\frac{dy}{dx}$ =f(x,y), and see what kind of patterns you can generate. Can you form a differential equation that will give solution curves that are ellipses? hyperbolas? Just try it by changing the function of *x* and *y* in red.

In[12]:=

```
Clear[x, y, f]

f = x/y

slopefield[f, x, y, -4, 4, -4, 4];
```

Out[13]=

$\frac{x}{y}$



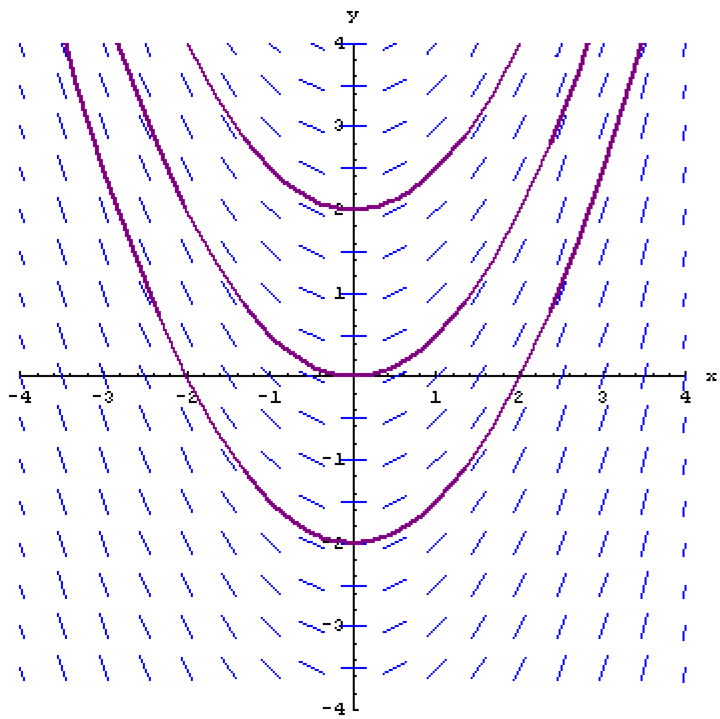
Part II: Antiderivatives

If we consider differential equations of the form $\frac{dy}{dx}$ =f(x), then the solutions for *y* as a function of *x* are simply the antiderivatives of f(x); that is, y=∫f(x)ⅆx. Let's look at some slope fields and solution curves for differential equations of this kind.

First we consider $\frac{dy}{dx} = x$.

In[15]:=

```
Clear[f, x, y]

f = x;

slopefield[f, x, y, -4, 4, -4, 4, {y[0] == -2, y
```



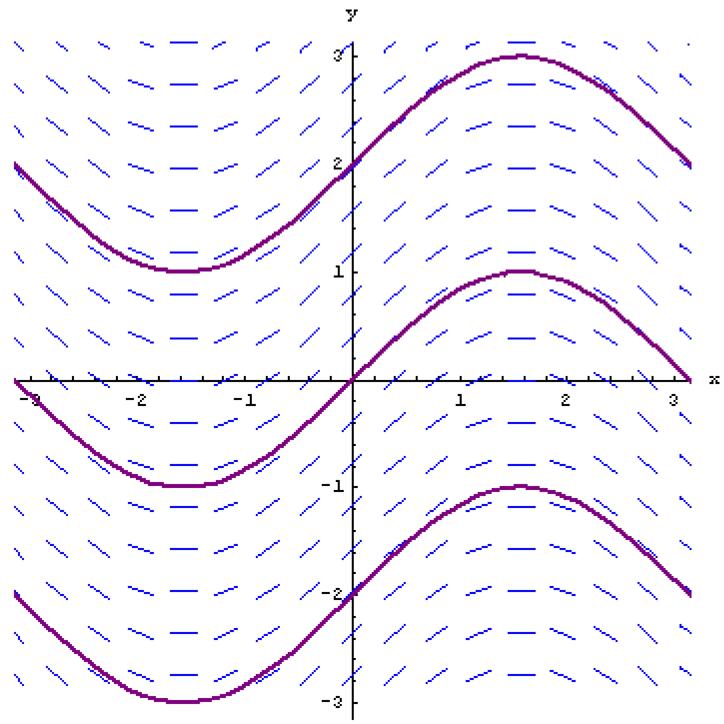
Notice that the slopes in the field above are constant along any vertical line. This shows graphically that the derivative of *y* with respect to *x* is not a function of *y*, as reflected in the differential equation $\frac{dy}{dx}$ =x. Along any horizontal line, however, the slopes vary with *x*, and the pattern shown in the slope field above is consistent with the differential equation $\frac{dy}{dx}$ =x. The slopes are negative when *x* is negative, positive when *x* is positive, and zero when *x* is zero.

Let's try a periodic function, $\frac{dy}{dx}$ =cos x.

In[18]:=

```
f = Cos[x];
```

```
slopefield[f, x, y, -Pi, Pi, -Pi, Pi,
 {y[0] == -2, y[0] == 0, y[0] == 2}];
```

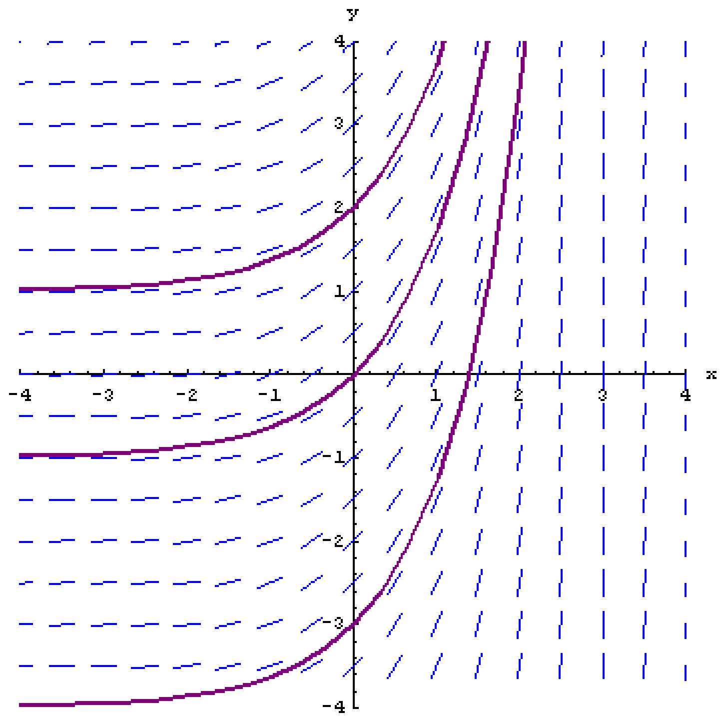


You Try It: Part II

Use the slope field command to study some antiderivative problems of the form $\frac{dy}{dx}=f(x)$ where you pick the function f(x) (in red) and the initial conditions (in red). Don't forget to use the double-equals (==) if you add initial conditions.

In[20]:=

```
Clear[x, y, f]

f = Exp[x];
slopefield[f, x, y, -4, 4, -4, 4,
 {y[0] == -3, y[0] == 0, y[0] == 2}];
```

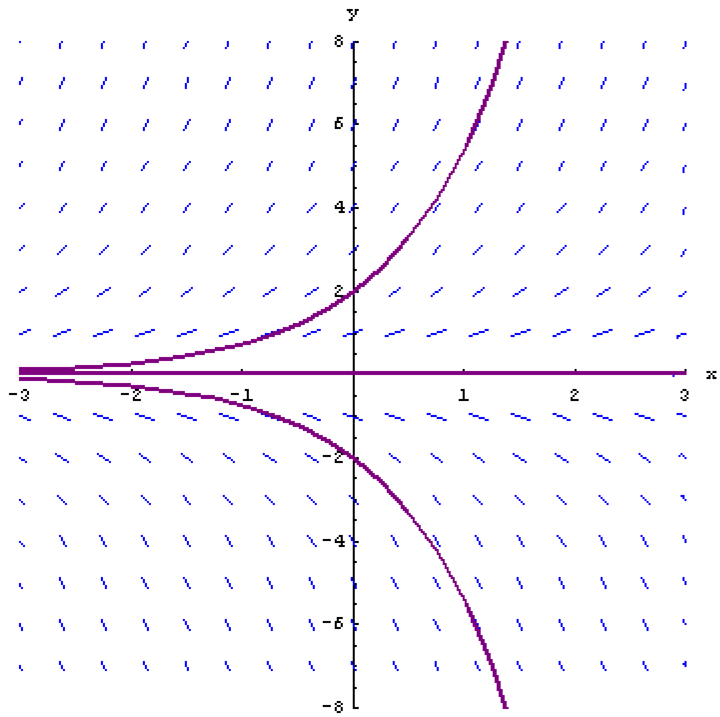


Part III: Autonomous Differential Equations

Differential equations of the form $\frac{dy}{dx}$ = f(y) are called autonomous differential equations. Let's look at the autonomous differential equation $\frac{dy}{dx}$ =*y*. Despite its simplicity, this is surely the most famous of all first-order, autonomous differential equations. Let's draw the slope field and some solution curves.

In[22]:=

```
Clear[f, x, y]

f = y;

slopefield[f, x, y, -3, 3, -8, 8, {y[0] == -2, y
```

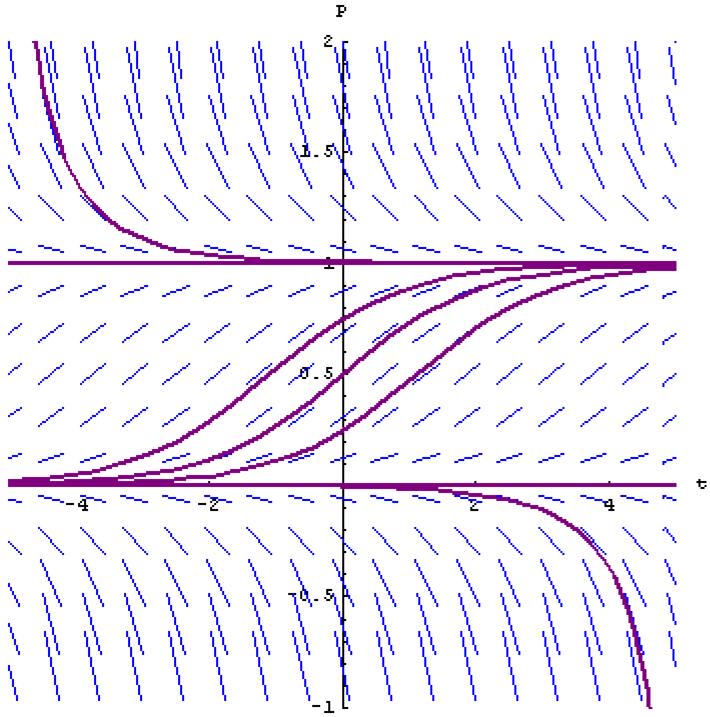


Note that in the slope fields for $\frac{dy}{dx}$ =y, the slopes are constant along any horizontal line and vary linearly along any vertical line. This is the reverse of what we observed for the antiderivative problem of the form $\frac{dy}{dx}$ =x.

The logistic equation is another interesting autonomous first-order differential equation. Let's draw a slope field and some solution curves for a specific one, say $\frac{dP}{dt}$ =P(1-P). The **slopefield** command will generate some error messages because two of the solution curves go off to infinity and have vertical asymptotes. However, the solution curves as drawn are okay.

In[25]:=

```
Clear[f, t, P]

f = P (1 - P);

slopefield[f, t, P, -5, 5, -1, 2,
 {P[0] == -0.005, P[0] == 0., P[0] == 0.25, P[0] ==
  P[0] == 0.75, P[0] == 1.0, P[0] == 1.005}];
```



The long-term behavior of the solution to the logistic equation depends upon the initial condition. The slope field graph shown above suggests the following: $\lim_{t\to\infty}$ P(t)=-∞ when P(0)<0; P(t)=0 for all values of *t* when P(0)=0; and, $\lim_{t\to\infty}$ P(t)=1 when P(0)>0.

You Try It: Part III

The logistic equation has another interesting autonomous form. Let's draw a slope field and some solution curves for

$$\frac{dP}{dt} = -P(1-P)(2-P).$$

The **slopefield** command will generate some error messages. See if you can determine why this is so after you see the plot. The solution curves are drawn okay.

In[28]:=

```
Clear[f, t, P]

f = -0.1*P (1 - P) (2 - P);
```

```
slopefield[f, t, P, -5, 20, -2, 5,
 {P[0] == -0.25, P[0] == -0.005, P[0] == .3, P[0]
  P[0] == 0.5, P[0] == 1.3, P[0] == 1.7, P[0] == 2.
```

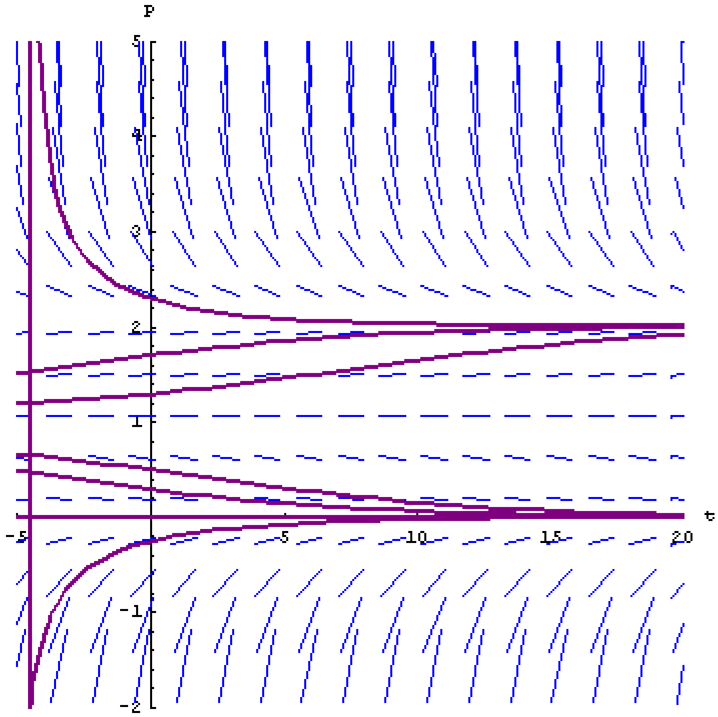
```
NDSolve::ndsz :
 At xloc == -4.47896, step size is effectively
   zero; singularity or stiff
   system suspected. More…
```

```
InterpolatingFunction::dmval :
 Input value {-5.} lies outside the range
   of data in the interpolating function.
   Extrapolation will be used. More…
```

```
InterpolatingFunction::dmval :
 Input value {-4.5063} lies outside
   the range of data in the interpolating
   function. Extrapolation will be used. More…
```

```
InterpolatingFunction::dmval :
 Input value {-4.75438} lies outside
   the range of data in the interpolating
   function. Extrapolation will be used. More…
```

```
General::stop : Further output of
   InterpolatingFunction::dmval will be
   suppressed during this calculation. More…
```



Some solutions approach 2 and some approach 0. What is the dividing line between these sets of solutions? What has caused this to happen?

Look up the concepts of a threshold population size and a carrying capacity. See how those concepts apply here.

You Try It: General

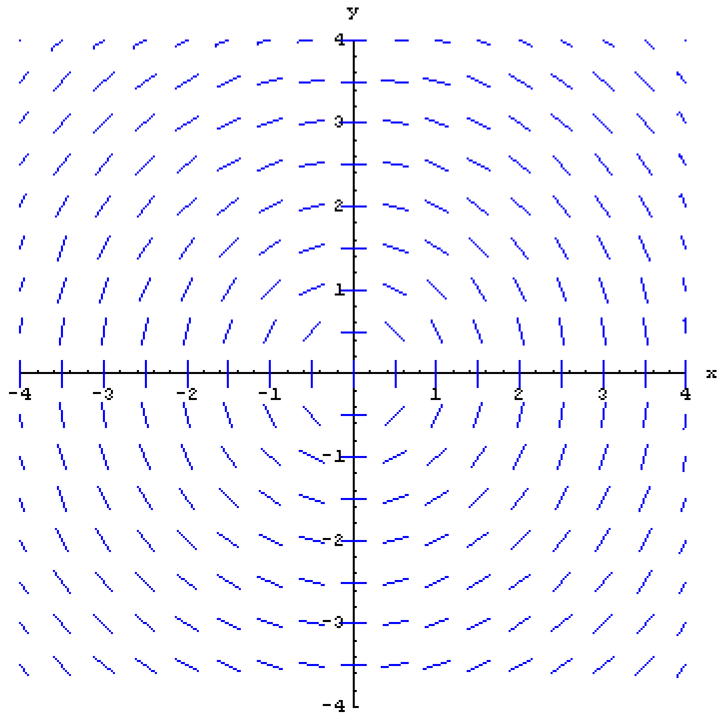
Consider some nonautonomous differential equations of the form $\frac{dy}{dx}$ =f(x,y). In this section we will start with the **slopefield[ ]** command to plot the slope fields, and we will ask you to sketch some representative solution curves. Here are a few examples. See if you can predict what the solution curves will look like by looking at the direction fields. Describe the long-term behavior of the solutions for all possible initial values of y(0).

▪ Solve $\frac{dy}{dx} = -\frac{x}{y}$

In[31]:=

```
f = -x / y;
```

```
slopefield[f, x, y, -4, 4, -4, 4];
```



Change the terms in red to whatever you wish. Remember to use the double-equal sign if you add more initial states. You can expect to get error messages with this. Why might you predict that?

In[33]:=

```
slopefield[f, x, y, -4, 4, -4, 4, {y[0] == -1, }
```

```
NDSolve::ndsz :
 At xloc == -1., step size is effectively zero;
   singularity or stiff system suspected. More…

NDSolve::ndsz :
 At xloc == 0.9999995320107085`, step
   size is effectively zero; singularity
   or stiff system suspected. More…

NDSolve::ndsz :
 At xloc == -3., step size is effectively zero;
   singularity or stiff system suspected. More…
```

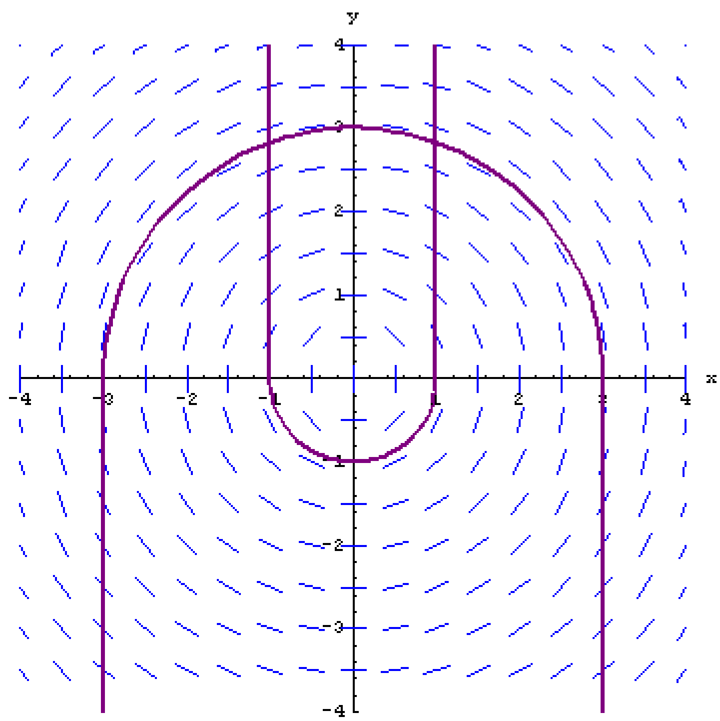
```
General::stop :
 Further output of NDSolve::ndsz will be
   suppressed during this calculation. More…
```

```
InterpolatingFunction::dmval :
 Input value {-4.} lies outside the range
   of data in the interpolating function.
   Extrapolation will be used. More…
```

```
InterpolatingFunction::dmval :
 Input value {-3.67546} lies outside
   the range of data in the interpolating
   function. Extrapolation will be used. More…
```

```
InterpolatingFunction::dmval :
 Input value {-3.32153} lies outside
   the range of data in the interpolating
   function. Extrapolation will be used. More…
```

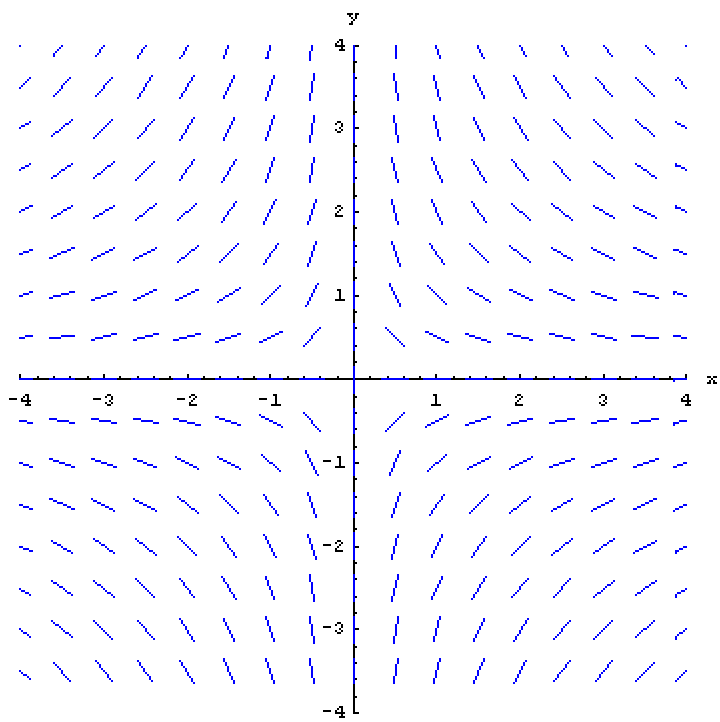
```
General::stop :  Further output of
   InterpolatingFunction::dmval will be
   suppressed during this calculation. More…
```



▪ Solve $\frac{dy}{dx} = -\frac{y}{x}$

In[34]:=

```
f = -y / x;

slopefield[f, x, y, -4, 4, -4, 4];
```



Change the terms in red to whatever you wish. Remember to use the double-equal sign if you add more initial states. You can expect to get error messages with this. Why might you predict that? Why would you avoid initial conditions at x=0?

In[36]:=

```
slopefield[f, x, y, -4, 4, -4, 4, {y[1] == -1, }
```

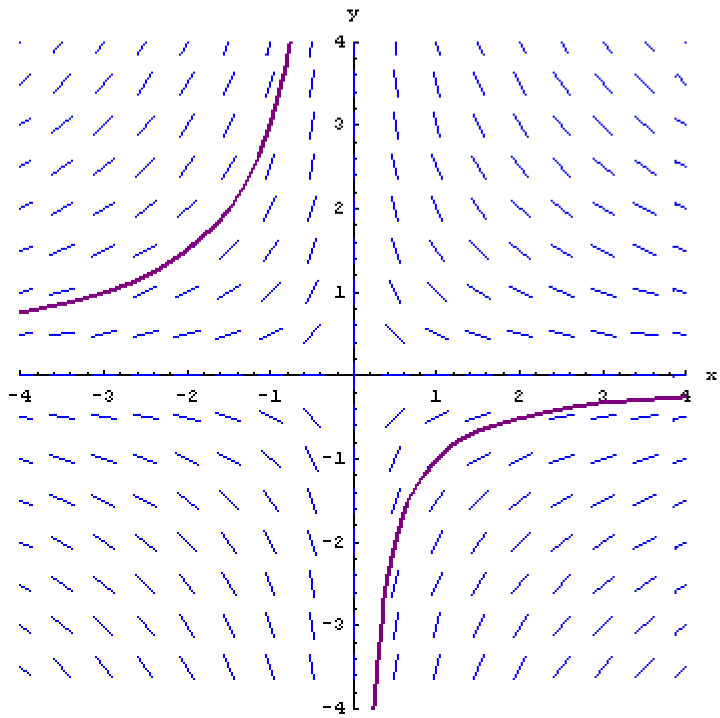
```
NDSolve::mxst :
 Maximum number of 10000 steps reached at the
   point xloc == 1.1245514733181794`*^-172. More…
```

```
NDSolve::mxst :
 Maximum number of 10000 steps reached at
   the point xloc == -3.90892×10^-172. More…

InterpolatingFunction::dmval :
 Input value {-4.} lies outside the range
   of data in the interpolating function.
   Extrapolation will be used. More…

InterpolatingFunction::dprec :
 The precision of input value {-4.}
   and/or the interpolation grid is
   insufficient to compute the value. More…

InterpolatingFunction::dmval :
 Input value {-4.} lies outside the range
   of data in the interpolating function.
   Extrapolation will be used. More…

InterpolatingFunction::dprec :
 The precision of input value {-4.}
   and/or the interpolation grid is
   insufficient to compute the value. More…

InterpolatingFunction::dmval :
 Input value {-4.} lies outside the range
   of data in the interpolating function.
   Extrapolation will be used. More…

General::stop : Further output of
   InterpolatingFunction::dmval will be
   suppressed during this calculation. More…

InterpolatingFunction::dprec :
 The precision of input value {-4.}
   and/or the interpolation grid is
   insufficient to compute the value. More…

General::stop : Further output of
   InterpolatingFunction::dprec will be
   suppressed during this calculation. More…

Plot::plnr :
 InterpolatingFunction[{{1.12455×10^-172, 4.}},
     <<3>>, {Automatic}][xloc] is not a machine-
   size real number at xloc = -4.. More…
```

```
Plot::plnr :
 InterpolatingFunction[{{1.12455×10^-172, 4.}},
    «3», (Automatic)] [xloc]  is not a machine-
  size real number at xloc = -3.67546. More…
```

```
Plot::plnr :
 InterpolatingFunction[{{1.12455×10^-172, 4.}},
    «3», (Automatic)] [xloc]  is not a machine-
  size real number at xloc = -3.32153. More…
```

```
General::stop :
 Further output of Plot::plnr will be
  suppressed during this calculation. More…
```

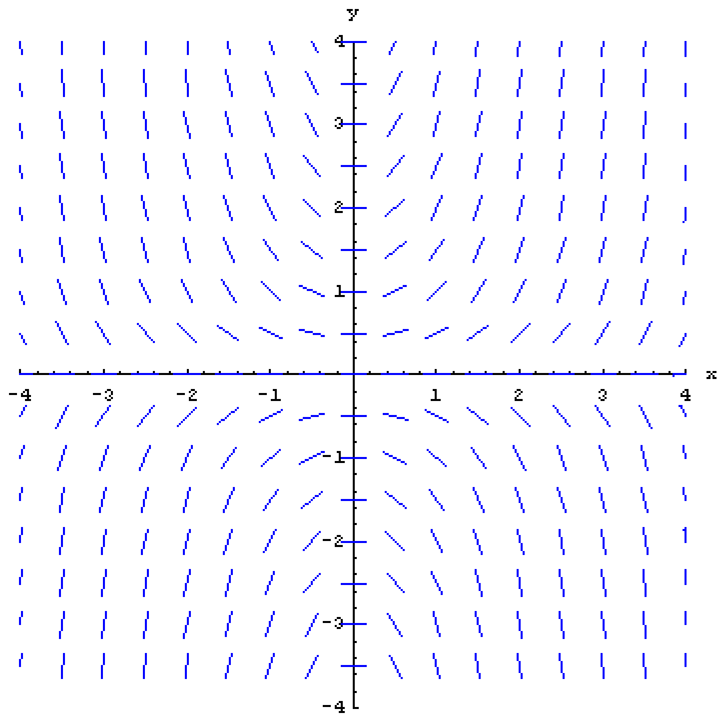


- ## Solve $\frac{dy}{dx}$ =x y

In[37]:=

```
f = x * y;
```

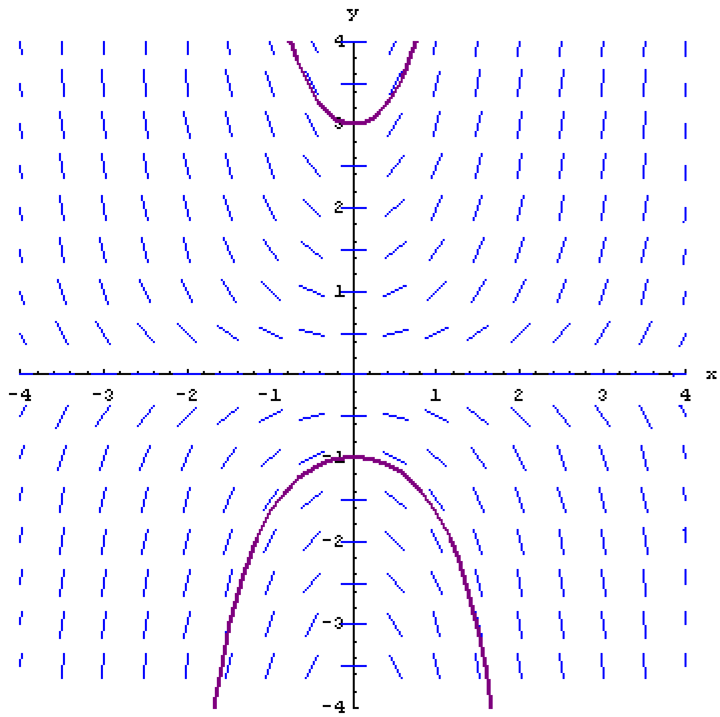
```
slopefield[f, x, y, -4, 4, -4, 4];
```



Change the terms in red to whatever you wish. Remember to use the double-equal sign if you add more initial states.

In[39]:=

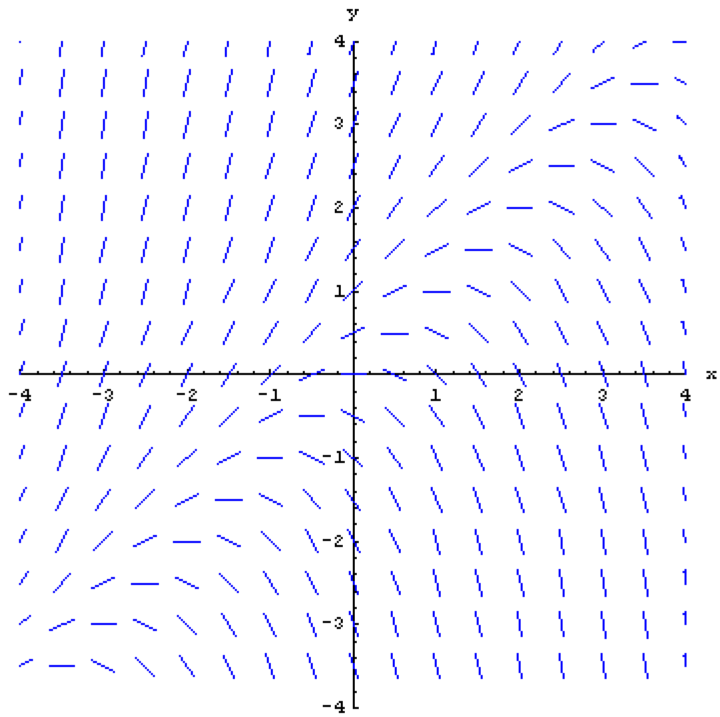
```
slopefield[f, x, y, -4, 4, -4, 4, {y[0] == -1, }
```



- **Solve $\frac{dy}{dx}$ =y-x**

In[40]:=

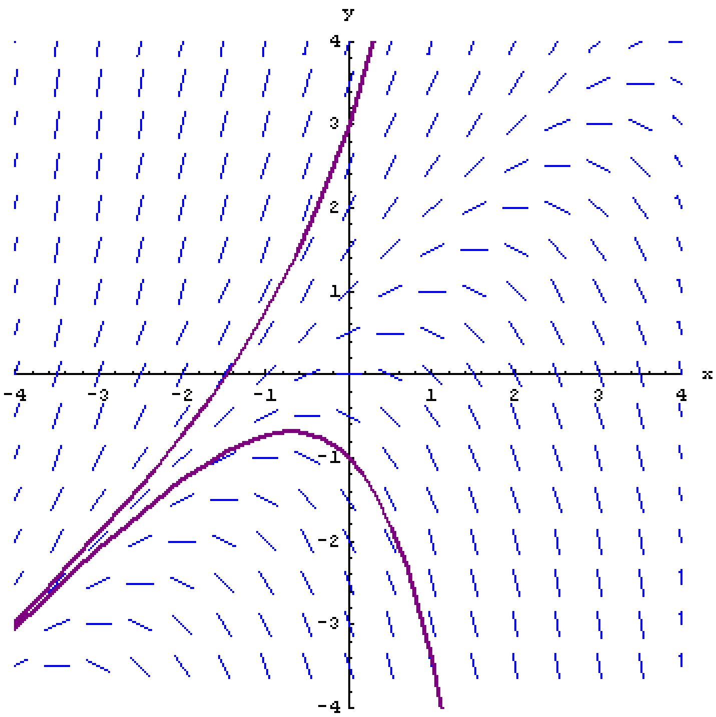
```
f = y - x;

slopefield[f, x, y, -4, 4, -4, 4];
```



Change the terms in red to whatever you wish. Remember to use the double equal sign if you add more initial states. Can you find a solution that is a straight line?

In[42]:=

```
slopefield[f, x, y, -4, 4, -4, 4, {y[0] == -1, }
```



▪ Try More of Your Own

It's fun! Make up some of your own differential equations of the form $\frac{dy}{dx}$ =f(x,y), and see what kind of patterns and solutions you can generate?
