

Differential Equations & Mathematica

©1999 Bill Davis and Jerry Uhl

Produced by Bruce Carpenter

Published by Math Everywhere, Inc.

www.matheverywhere.com

DE.05 First Order Differential Equations Basics

B.1) Reading an autonomous diffeq through phase lines

□B.1.a.i)

Here's an old friend, the logistic differential equation:

```
Clear[diffeq, y, t, f, starter];
r = 0.45;
b = 6;
f[t_, y_] = r y (1 - y/b);

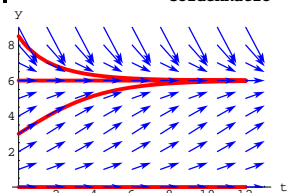
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == 0.45 (1 -  $\frac{y[t]}{6}$ ) y[t]
y[0] == starter
```

Here is a flow plot shown with plots of four solutions of this diffeq.

```
{ylow, yhigh} = {0, 9};
{tlow, thigh} = {0, 12};
flowplot = Table[Arrow[{1, f[t, y]},
  Tail -> {t, y}, VectorColor -> Blue, HeadSize -> 0.6],
  {t, tlow, thigh,  $\frac{thigh - tlow}{8}$ }, {y, ylow, yhigh,  $\frac{yhigh - ylow}{9}$ };

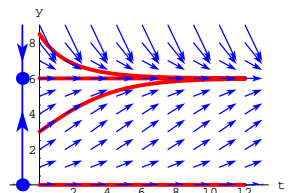
Clear[y1, y2, y3, y, t];
{starter1, starter2, starter3, starter4} = {0, 3.0, 6.0, 8.5};
endtime = thigh;
y1[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter1, y[t], {t, 0, endtime}][[1]];
y2[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter2, y[t], {t, 0, endtime}][[1]];
y3[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter3, y[t], {t, 0, endtime}][[1]];
y4[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter4, y[t], {t, 0, endtime}][[1]];

solutionplots = Plot[{y1[t], y2[t], y3[t], y4[t]}, {t, 0, endtime},
  PlotStyle -> {{Thickness[0.015], Red}}, DisplayFunction -> Identity];
setup = Show[solutionplots, flowplot,
  Axes -> True, AxesLabel -> {"t", "y"}, PlotRange -> {ylow, yhigh},
  AspectRatio ->  $\frac{1}{\text{GoldenRatio}}$ , DisplayFunction -> $DisplayFunction];
```



Look at this embellishment of the graphic immediately above.

```
phaseline = PhaseLine[f[t, y], {y, ylow, yhigh}, Blue, -1];
flowphaseplot = Show[setup, phaseline];
```



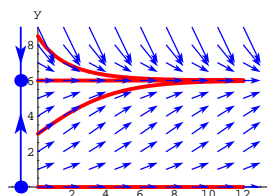
Folks like to call that gadget on the far left by the name "phase line" for the diffeq.

Explain what those arrowheads on the far left mean.

□Answer:

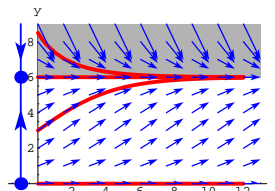
Take another look:

```
Show[flowphaseplot];
```



Shade in to the right of the down arrowhead:

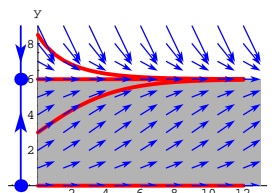
```
downshadow = Graphics[{GrayLevel[0.7], Polygon[
  {{tlow, 6}, {tlow, yhigh}, {thigh + 1, yhigh}, {thigh + 1, 6}}];];
Show[downshadow, flowphaseplot, Axes -> True, AxesLabel -> {"t", "y"},
  PlotRange -> {ylow, yhigh}, AspectRatio ->  $\frac{1}{\text{GoldenRatio}}$ ];
```



The down arrowhead indicates that any solution passing through the shaded part above is going down and must level out at the cutoff point at $y = 6$.

Now shade in to the right of the up arrowhead:

```
upshadow = Graphics[{GrayLevel[0.7], Polygon[
  {{tlow, ylow}, {tlow, 6}, {thigh + 1, 6}, {thigh + 1, ylow}}];];
Show[upshadow, flowphaseplot, Axes -> True, AxesLabel -> {"t", "y"},
  PlotRange -> {ylow, yhigh}, AspectRatio ->  $\frac{1}{\text{GoldenRatio}}$ ];
```



The up arrowhead indicates that any solution passing through the shaded part above is going up and must level out at the cutoff point at $y = 6$.

The solution corresponding to $y[0] = 6$ stays flat.

All other solutions plot out entirely above or entirely below the flat solution.

□B.1.a.ii)

What's the idea behind the phase line?

□Answer:

See how the diffeq was entered:

```
Clear[diffeq, y, t, f, starter];
b = 6;
f[t_, y_] = 0.45 y (1 -  $\frac{y}{b}$ );

(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == 0.45 (1 -  $\frac{y[t]}{6}$ ) y[t]
y[0] == starter
```

This allows you to read off y' as a function of y :

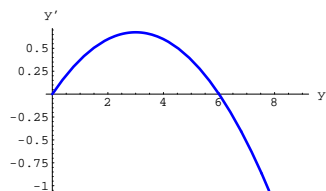
```
yprime = f[t, y]
0.45 (1 -  $\frac{y}{6}$ ) y
```

Plot y' as a function of y :

```

phaseplot =
Plot[f[t, y], {y, ylow, yhigh}, PlotStyle -> {{Thickness[0.01], Blue}},
AxesLabel -> {"y", "y'"}, AspectRatio ->  $\frac{1}{\text{GoldenRatio}}$ ];

```



This plot tells you that

$$y' = f[t, y] > 0$$

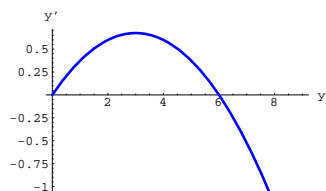
for y between 0 and 6.0.

The upshot:

If a solution is ever between 0 and 6.0, then that solution is going up and will level off as it gets near 6.0.

Take another look:

```
Show[phaseplot];
```



This plot also tells you that

$$y' = f[t, y] < 0$$

for y bigger than 6.0.

The upshot:

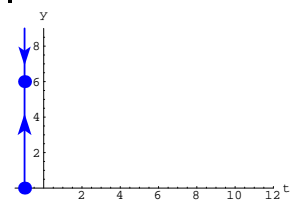
If a solution is ever bigger than 6.0, then that solution is going down and will level off as it gets near 6.0.

This is exactly the same thing that the phase line plot is telling you:

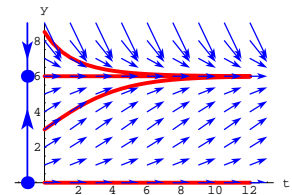
```

Show[phaseline, Axes -> True, AxesLabel -> {"t", "y"}, PlotRange ->
{{(tlow - 1.5, thigh), (ylow, yhigh)}, AspectRatio ->  $\frac{1}{\text{GoldenRatio}}$ ];

```



```
Show[flowphaseplot];
```



□B.1.b.i)

Here's a new diffeq:

```

Clear[diffeq, y, t, f, starter];
b = 6;
f[t_, y_] = 0.08 (y - 2.0) (4.0 - y) (y - 8.0);
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm

```

```

y'[t] == 0.08 (4. - y[t]) (-8. + y[t]) (-2. + y[t])
y[0] == starter

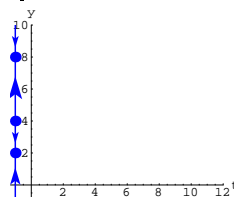
```

Here is the phase line:

```

{ylow, yhigh} = {-1, 10};
{tlow, thigh} = {0, 12};
phaseline = PhaseLine[f[t, y], {y, ylow, yhigh}, Blue, -1];
phaselineplot =
Show[phaseline, PlotRange -> {{tlow - 1.3, thigh}, {ylow, yhigh}},
Axes -> True, AxesLabel -> {"t", "y"}];

```



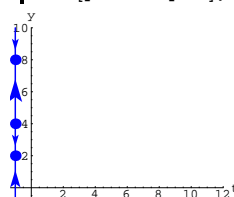
Milk that plot for all the information you can get about how solutions behave.

□ Answer:

This gives you just about as much information as the flow field.

Take another look:

```
Show[phaselineplot];
```



Read off:

Solutions starting with $y[0] > 8$ go down and level off at $y = 8$.

Solutions starting with $4 < y[0] < 8$ go up and level off at $y = 8$.

Solutions starting with $2 < y[0] < 4$ go down and level off at $y = 2$.

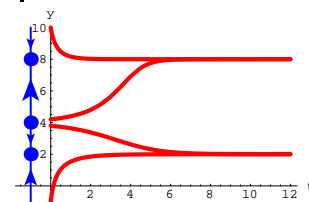
Solutions starting with $y[0] < 2$ go up and level off at $y = 2$.

Check it out with some sample solution plots:

```

Clear[y1, y2, y3, y4, y, t];
{starter1, starter2, starter3, starter4} = {-1.0, 3.8, 4.2, 10};
endtime = thigh;
y1[t_] = y[t] /.
NDSolve[diffeq /. starter -> starter1, y[t], {t, 0, endtime}][[1]];
y2[t_] = y[t] /.
NDSolve[diffeq /. starter -> starter2, y[t], {t, 0, endtime}][[1]];
y3[t_] = y[t] /.
NDSolve[diffeq /. starter -> starter3, y[t], {t, 0, endtime}][[1]];
y4[t_] = y[t] /.
NDSolve[diffeq /. starter -> starter4, y[t], {t, 0, endtime}][[1]];
solutionplots = Plot[{y1[t], y2[t], y3[t], y4[t]}, {t, 0, endtime},
PlotStyle -> {{Thickness[0.015], Red}}, DisplayFunction -> Identity];
Show[solutionplots, phaselineplot, Axes -> True, AxesLabel -> {"t", "y"},
PlotRange -> {ylow, yhigh}, AspectRatio ->  $\frac{1}{\text{GoldenRatio}}$ ,
DisplayFunction -> $DisplayFunction];

```



Sure enough.

Solutions starting with $y[0] > 8$ go down and level off near $y = 8$.

Solutions starting with $4 < y[0] < 8$ go up and level off near $y = 8$.

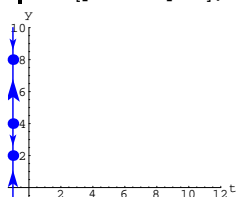
Solutions starting with $2 < y[0] < 4$ go down and level off at $y = 2$.

Solutions starting with $y[0] < 2$ go up and level off at $y = 2$.

□B.1.b.ii)

Stay with the same diffeq and look at the phase line plot again:

```
Show[phaselineplot];
```

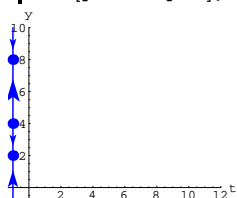


How does the phase plot signal you to beware of extreme sensitivity to errors in starter data for starting values $y[0]$ near $y[0] = 4.0$?

□Answer:

Take another look:

```
Show[phaselineplot];
```



Just above $y = 4$, you see an up arrowhead and just below $y = 4$, you see a down arrowhead.

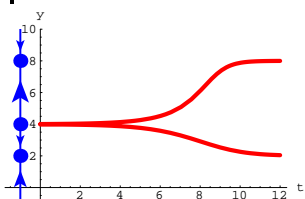
This tells you that that for $y[0]$ near 4.0, you can expect sensitive dependence on starter data:

In fact, solutions starting at 4.01 go up, but solutions starting at 3.99 go down.

See it happen for two solutions $y1[t]$ and $y2[t]$ with $y1[0] = 3.99$ and $y2[0] = 4.01$:

```
Clear[y1, y2, y3, y4, y, t];
{starter1, starter2} = {3.99, 4.01};
endtime = thigh;
y1[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter1, y[t], {t, 0, endtime}][[1]];
y2[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter2, y[t], {t, 0, endtime}][[1]];
solutionplots = Plot[{y1[t], y2[t]}, {t, 0, endtime},
  PlotStyle -> {{Thickness[0.015], Red}}, DisplayFunction -> Identity];

Show[solutionplots, phaselineplot, Axes -> True, AxesLabel -> {"t", "y"},
  PlotRange -> {ylow, yhigh}, AspectRatio -> 1/GoldenRatio,
  DisplayFunction -> $DisplayFunction];
```



That's extreme sensitivity to errors in the starting value of $y[0]$ for $y[0]$ near 4.0.

B.2) For non-autonomous diffeqs, a single phase line is not all that useful;

multiple phase lines are sometimes useful

□B.2.a.i) Sometimes a single phaseline won't cut it

Here's a diffeq related to the logistic diffeq.

```
Clear[diffeq1, y, t, f1, starter];
b = 7.0;
```

$$f1[t_, y_] = 0.2 y^2 \left(1 - \frac{y}{b}\right);$$

```
(diffeq1 = {y'[t] == f1[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == 0.2 (1 - 0.142857 y[t]) y[t]^2
y[0] == starter
```

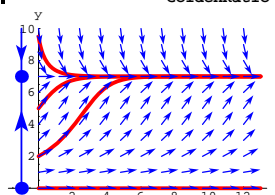
Here's a flow plot shown with plots of four solutions of this diffeq.

```
{ylow, yhigh} = {0, 10};
{tlow, thigh} = {0, 12};
flowplot1 = Table[Arrow[{1, f1[t, y]}, Tail -> {t, y},
  VectorColor -> Blue, ScaleFactor -> Normalize, HeadSize -> 0.6],
  {t, tlow, thigh, (thigh - tlow)/10}, {y, ylow, yhigh, (yhigh - ylow)/10}];

Clear[y1, y2, y3, y, t];
{starter1, starter2, starter3, starter4} = {0, 2.0, 5.0, 9.5};
endtime = thigh + 1;
y1[t_] = y[t] /.
  NDSolve[diffeq1 /. starter -> starter1, y[t], {t, 0, endtime}][[1]];
y2[t_] = y[t] /.
  NDSolve[diffeq1 /. starter -> starter2, y[t], {t, 0, endtime}][[1]];
y3[t_] = y[t] /.
  NDSolve[diffeq1 /. starter -> starter3, y[t], {t, 0, endtime}][[1]];
y4[t_] = y[t] /.
  NDSolve[diffeq1 /. starter -> starter4, y[t], {t, 0, endtime}][[1]];

solutionplots1 = Plot[{y1[t], y2[t], y3[t], y4[t]}, {t, 0, endtime},
  PlotStyle -> {{Thickness[0.015], Red}}, DisplayFunction -> Identity];
phaseline1 = PhaseLine[f1[t, y], {y, ylow, yhigh}, Blue, -1];

setup1 = Show[solutionplots1, flowplot1, phaseline1,
  Axes -> True, AxesLabel -> {"t", "y"}, PlotRange -> {ylow, yhigh},
  AspectRatio -> 1/GoldenRatio, DisplayFunction -> $DisplayFunction];
```



The single phase line works beautifully and reveals just about all you need to know about this diffeq.

Now look at this new diffeq:

```
Clear[newdiffeq, y, t, f2, starter];
f2[t_, y_] = y - t;
(diffeq2 = {y'[t] == f2[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == -t + y[t]
y[0] == starter
```

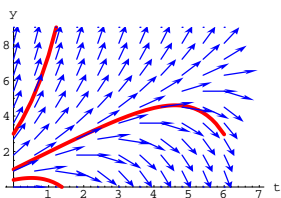
And look at this flow plot shown with plots of three solutions of this diffeq:

```
{ylow, yhigh} = {0, 9};
{tlow, thigh} = {0, 6};
flowplot2 = Table[Arrow[{1, f2[t, y]}, Tail -> {t, y},
  VectorColor -> Blue, ScaleFactor -> Normalize, HeadSize -> 0.4],
  {t, tlow, thigh, (thigh - tlow)/10}, {y, ylow, yhigh, (yhigh - ylow)/10}];

Clear[y1, y2, y3, y, t];
{starter1, starter2, starter3} = {0.4, 0.99, 3.0};
endtime = thigh;
y1[t_] = y[t] /.
  NDSolve[diffeq2 /. starter -> starter1, y[t], {t, 0, endtime}][[1]];
y2[t_] = y[t] /.
  NDSolve[diffeq2 /. starter -> starter2, y[t], {t, 0, endtime}][[1]];
y3[t_] = y[t] /.
  NDSolve[diffeq2 /. starter -> starter3, y[t], {t, 0, endtime}][[1]];

solutionplots2 = Plot[{y1[t], y2[t], y3[t]}, {t, 0, endtime},
  PlotStyle -> {{Thickness[0.015], Red}}, DisplayFunction -> Identity];

setup2 = Show[solutionplots2, flowplot2,
  Axes -> True, AxesLabel -> {"t", "y"}, PlotRange -> {ylow, yhigh},
  AspectRatio -> 1/GoldenRatio, DisplayFunction -> $DisplayFunction];
```

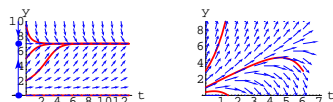


Explain why a single phase line wouldn't have been useful for the second diffeq.

□Answer:

Look at both plots side by side:

```
both = Show[GraphicsArray[{setup1, setup2}]];
```



If the plots are too small, grab the plot and drag one of the handles to make it bigger.

On the left, the single phase line does a great job.

But on the right a single phase line won't work.

Reason: On the right, some of the solutions go up for a while and then turn around and go down. This makes an overall single phase line out of the question.

□B.2.a.ii)

Why did that happen?

□Answer:

Look at the way the two diffeqs were entered:

```
Clear[diffeq1, y, t, f1, starter];
b = 7.0;
```

```
f1[t_, y_] = 0.2 y^2 (1 - y/b);
(diffeq1 = {y'[t] == f1[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == 0.2 (1 - 0.142857 y[t]) y[t]^2
y[0] == starter
Clear[diffeq2, y, t, f2, starter];
f2[t_, y_] = y - t;
(diffeq2 = {y'[t] == f2[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == -t + y[t]
y[0] == starter
```

Now look at the vectors that define each flow:

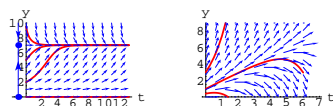
```
flow1 = {1, f1[t, y]}
{1, 0.2 (1 - 0.142857 y) y^2}
flow2 = {1, f2[t, y]}
{1, -t + y}
```

The slope of the flow vectors for diffeq1 depends only on y.

The slope of the flow vectors for diffeq2 depends on BOTH t and y.

See it:

```
Show[both];
```



That's why a single phase line is not all that useful for diffeq2.

□B.2.a.iii) Recognizing autonomous diffeqs

Folks say that a diffeq is autonomous if a single phase line tells all. How do you recognize in advance whether a given diffeq is autonomous?

□Answer:

Look at the function f[t, y] used to enter the diffeq.

If f[t, y] has no dependence on t, then you have an autonomous diffeq.

Otherwise you don't.

Samples:

```
Clear[diffeq, b, a, y, t, f, starter];
f[t_, y_] = a y^3 (1 - y^2/b);
(diffeq1 = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == a y[t]^3 (1 - y[t]^2/b)
y[0] == starter
```

Look at:

```
f[t, y]
a y^3 (1 - y^2/b)
```

No dependence on t.

The upshot: This diffeq is autonomous and is ripe for analysis through a phase line.

Another:

```
Clear[diffeq, b, a, y, t, f, starter];
f[t_, y_] = a y^2 (Sin[t] - y/b);
(diffeq1 = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == a y[t]^2 (Sin[t] - y[t]/b)
y[0] == starter
```

Look at:

```
f[t, y]
a y^2 (-y/b + Sin[t])
```

Vivid dependence on t.

The upshot: This diffeq is not autonomous.

□B.2.a.iv) Multiple phaselines for non-autonomous diffeqs

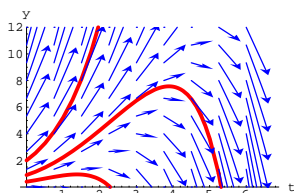
Go with this new non-autonomous diffeq:

```
Clear[newdiffeq, y, t, f, starter];
f[t_, y_] = y - 0.5 t^2;
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == -0.5 t^2 + y[t]
y[0] == starter
```

You can see the t^2 term all by itself. This signals that this diffeq is not autonomous.

Look at this flow plot shown with plots of three solutions of this diffeq:

```
{ylow, yhigh} = {0, 12};
{tlow, thigh} = {0, 6};
scalefactor = 0.6;
flowplot = Table[Arrow[{1, f[t, y]}, Tail -> {t, y},
  VectorColor -> Blue, ScaleFactor -> scalefactor, HeadSize -> 0.5],
  {t, tlow, thigh, (thigh - tlow)/8}, {y, ylow, yhigh, (yhigh - ylow)/9}];
Clear[y1, y2, y3, y, t];
{starter1, starter2, starter3} = {0.4, 0.9, 2};
endtime = thigh;
y1[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter1, y[t], {t, 0, endtime}][[1]];
y2[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter2, y[t], {t, 0, endtime}][[1]];
y3[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter3, y[t], {t, 0, endtime}][[1]];
solutionplots = Plot[{y1[t], y2[t], y3[t]}, {t, 0, endtime},
  PlotStyle -> {{Thickness[0.015], Red}}, DisplayFunction -> Identity];
setup = Show[solutionplots, flowplot,
  Axes -> True, AxesLabel -> {"t", "y"}, PlotRange -> {ylow, yhigh},
  AspectRatio -> 1/GoldenRatio, DisplayFunction -> $DisplayFunction];
```



This diffeq is not autonomous. Thus, it is not possible to use a single phase line to analyze it.

Can you use multiple phase lines to any profit?

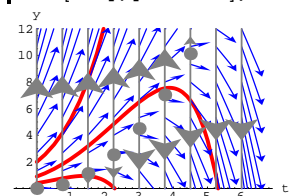
□ Answer:

Ya sure! You bettcha!

Make a different phase line for each t and then plot it at the corresponding t like this:

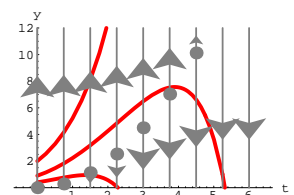
```
Clear[phaseline];
phaseline[t_] :=
  PhaseLine[f[t, y], {y, ylow, yhigh}, GrayLevel[0.5], t];
phaselines = Table[phaseline[t], {t, tlow, thigh,  $\frac{\text{thigh} - \text{tlow}}{8}$ }]];
```

```
Show[setup, phaselines];
```



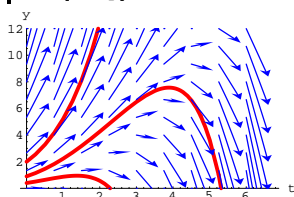
See the phase lines without the flow plot:

```
Show[solutionplots, phaselines, Axes → True, AxesLabel → {"t", "y"},
  PlotRange → {y, ylow, yhigh}, AspectRatio →  $\frac{1}{\text{GoldenRatio}}$ ,
  DisplayFunction → $DisplayFunction];
```



This gives some (but not all) of the info in the flowplot:

```
Show[setup];
```



What a great tool this computer is.

B.3) Autonomous diffeqs with parameters. Bifurcations and bifurcation points

□ B.3.a.i)

You are the manager of the Red Oak Catfish Farm in Hartland, Wisconsin. Today your attention is focused on a lake where the logistic model

$$y'[t] = a y[t] \left(1 - \frac{y[t]}{b}\right) \text{ with } 0 < a \text{ and } 0 < b$$

is used to estimate the fish population t weeks after the lake is stocked.

Incorporate a constant weekly harvest rate r . The model

$$y'[t] = a y[t] \left(1 - \frac{y[t]}{b}\right) - r,$$

is reasonable.

The specifics for this lake are in the code below:

In the code below:

→ t is measured in weeks

→ $y[t]$ and b are measured in thousands of fish.

→ The harvest rate r is measured in thousands of fish per week

```
Clear[diffeq, y, t, f, r, starter];
a = 0.34;
b = 7.1;
f[t_, y_] = a y (1 -  $\frac{y}{b}$ ) - r;

setup =
  (diffeq == (y'[t] == f[t, y[t]], y[0] == starter)) // ColumnForm
y'[t] == -r + 0.34 (1 - 0.140845 y[t]) y[t]
y[0] == starter
```

The secrets of this model are contained in the function $f[t, y]$:

```
f[t, y]
-r + 0.34 (1 - 0.140845 y)
```

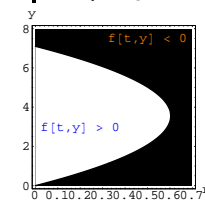
For each constant harvest rate r , this diffeq is autonomous.

Reason: $f[t, y]$ has no dependence on t .

Take a look at this labeled contour plot of $f[t, y]$ as a function of y and the harvest rate r :

```
{y, ylow, yhigh} = {0, 8};
{r, rlow, rhigh} = {0, 0.7};

fcontourplot = ContourPlot[f[t, y], {r, rlow, rhigh},
  {y, ylow, yhigh}, ContourSmoothing → Automatic, PlotPoints → 50,
  Contours → {0}, Axes → True, AxesLabel → {"r", "y"},
  Epilog → {{Blue, Text["f[t,y] > 0", {0.2, 3}]},
  {Orange, Text["f[t,y] < 0", {0.5, 7.5}]}}];
```



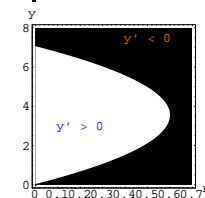
The light region indicates the $\{r, y\}$'s for which $f[t, y, r] > 0$ and the dark region indicates the $\{r, y\}$'s for which $f[t, y, r] < 0$.

Because

$$y'[t] = f[t, y[t]],$$

you can relabel this plot:

```
phaseplot =
  Show[fcontourplot, Epilog → {{Blue, Text["y' > 0", {0.2, 3}]},
  {Orange, Text["y' < 0", {0.5, 7.5}]}}];
```

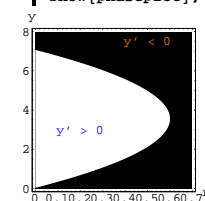


What does this plot tell you ?

□ Answer:

Take another look:

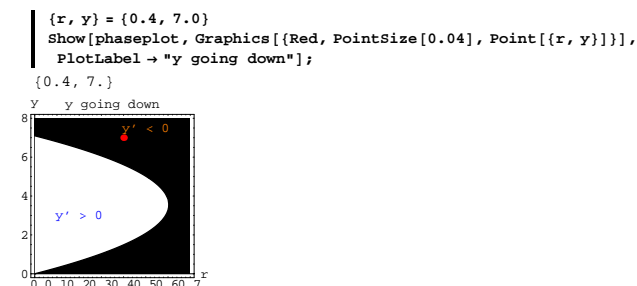
```
Show[phaseplot];
```



At the surface level, it tells you:

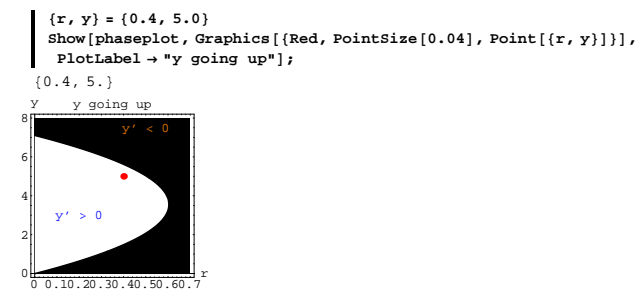
- 1) If you have a fish population y and go with harvest rate r so that $\{r, y\}$ plots out in the black region, then the fish population y is going down.
- 2) If you have a fish population y and go with harvest rate r so that $\{r, y\}$ plots out in the white region, then the fish population y is going up.

For example, if the harvest rate is $r = 0.4$ and the fish population is $y = 7.0$, then the point $\{r, y\} = \{0.4, 7.0\}$ plots out as follows:



This signals when you go with harvest rate $r = 0.4$ and the fish population is $y = 7.0$, then the fish population is going down.

See what happens when you go with harvest rate $r = 0.4$ and the fish population is $y = 5.0$:



This signals when you go with harvest rate $r = 0.4$ and the fish population is $y = 5.0$, then the fish population is going up.

population will decrease and level off at about 6 (thousand).

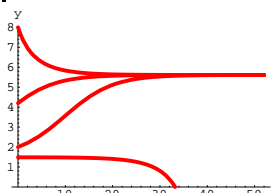
→ If you start with $y[0]$ below about 6 (thousand), but above about 1.5, then the fish population will increase and level off at about 6 (thousand).

→ If you start with $y[0]$ below about 1.5 (thousand) then the fish population will decrease to extinction.

See all this happen:

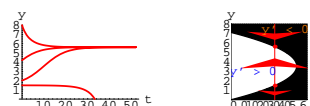
```
r = 0.4;
endtime = 52;
Clear[y1, y2, y3, y4, y, t];
{starter1, starter2, starter3, starter4} = {1.487, 2.0, 4.2, 8.0};
y1[t_] = y[t] /.
NDSolve[diffeq /. starter -> starter1, y[t], {t, 0, endtime}][[1]];
y2[t_] = y[t] /.
NDSolve[diffeq /. starter -> starter2, y[t], {t, 0, endtime}][[1]];
y3[t_] = y[t] /.
NDSolve[diffeq /. starter -> starter3, y[t], {t, 0, endtime}][[1]];
y4[t_] = y[t] /.
NDSolve[diffeq /. starter -> starter4, y[t], {t, 0, endtime}][[1]];
solutionplots = Plot[{y1[t], y2[t], y3[t], y4[t]}, {t, 0, endtime},
PlotStyle -> {{Thickness[0.015], Red}}, DisplayFunction -> Identity];

outcomes = Show[solutionplots, Axes -> True, AxesLabel -> {"t", "y"},
PlotRange -> {ylow, yhigh}, AspectRatio -> 1/GoldenRatio,
DisplayFunction -> $DisplayFunction];
```



Compare:

```
Show[GraphicsArray[{outcomes, newphaseplot}]];
```

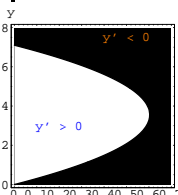


The new phase plot tells you all you need to know.

□B.3.a.iii)

Take another look at the phase plot:

```
Show[phaseplot];
```

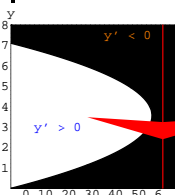


Use this plot to say everything you can about how the population y behaves when you go with a constant harvest rate $r = 0.65$.

□Answer:

Throw in a phase line corresponding to $r = 0.65$:

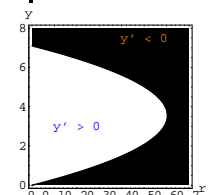
```
Clear[y];
r = 0.65;
phaseline = PhaseLine[f[t, y], {y, ylow, yhigh}, Red, r];
newphaseplot = Show[phaseplot, phaseline,
PlotRange -> {{rlow, rhigh}, {ylow, yhigh}}];
```



□B.3.a.ii)

Take another look at the phase plot:

```
Show[phaseplot];
```

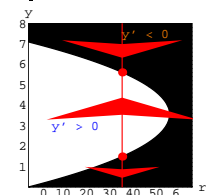


Use this plot to say everything you can about how the population y behaves when you go with a constant harvest rate $r = 0.4$.

□Answer:

Throw in a phase line corresponding to $r = 0.4$:

```
Clear[phaseline, y];
r = 0.4;
phaseline = PhaseLine[f[t, y], {y, ylow, yhigh}, Red, r];
newphaseplot = Show[phaseplot, phaseline,
PlotRange -> {{rlow, rhigh}, {ylow, yhigh}}];
```



Try not to be frightened by the big arrowheads.

This tells you that when you maintain a constant harvest rate

$r = 0.4$,

then:

→ If you start with $y[0]$ above about 6 (thousand), then the fish

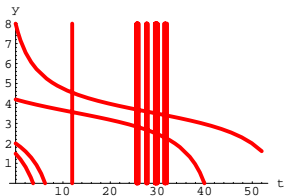
This tells you that when you maintain a constant harvest rate of $r = 0.65$,

then the fish population eventually dies out.

See it happen:

```
r = 0.65;
endtime = 52;
Clear[y1, y2, y3, y4, y, t];
{starter1, starter2, starter3, starter4} = {1.487, 2.0, 4.2, 8.0};
y1[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter1, y[t], {t, 0, endtime}][[1]];
y2[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter2, y[t], {t, 0, endtime}][[1]];
y3[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter3, y[t], {t, 0, endtime}][[1]];
y4[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter4, y[t], {t, 0, endtime}][[1]];

solutionplots = Plot[{y1[t], y2[t], y3[t], y4[t]}, {t, 0, endtime},
  PlotStyle -> {{Thickness[0.015], Red}}, DisplayFunction -> Identity];
outcomes = Show[solutionplots, Axes -> True, AxesLabel -> {"t", "y"},
  PlotRange -> {ylow, yhigh}, AspectRatio -> 1/GoldenRatio,
  DisplayFunction -> $DisplayFunction];
```



Disaster.

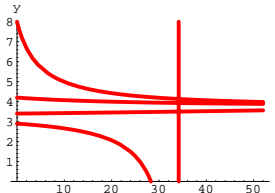
Compare:

```
Show[GraphicsArray[{outcomes, newphaseplot}]];
```

And to sustain this harvest rate you'll have to start with $y[0]$ more than about 3.3 (thousand fish). Check it out:

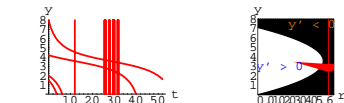
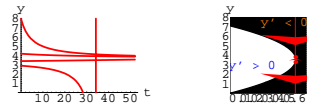
```
r = 0.6;
endtime = 52;
Clear[y1, y2, y3, y4, y, t];
{starter1, starter2, starter3, starter4} = {2.9, 3.4, 4.2, 8.0};
y1[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter1, y[t], {t, 0, endtime}][[1]];
y2[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter2, y[t], {t, 0, endtime}][[1]];
y3[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter3, y[t], {t, 0, endtime}][[1]];
y4[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter4, y[t], {t, 0, endtime}][[1]];

solutionplots = Plot[{y1[t], y2[t], y3[t], y4[t]}, {t, 0, endtime},
  PlotStyle -> {{Thickness[0.015], Red}}, DisplayFunction -> Identity];
outcomes = Show[solutionplots, Axes -> True, AxesLabel -> {"t", "y"},
  PlotRange -> {ylow, yhigh}, AspectRatio -> 1/GoldenRatio,
  DisplayFunction -> $DisplayFunction];
```



Compare:

```
Show[GraphicsArray[{outcomes, newphaseplot}]];
```



Again, the new phase plot tells you all you need to know.

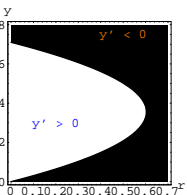
□B.3.a.iv)

Use the phase plot to estimate the largest sustainable harvest rate r . Estimate how large the starting value $y[0]$ will have to be to be able to maintain this largest sustainable harvest rate.

□ Answer:

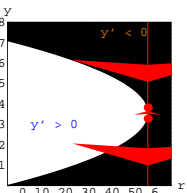
Take a peek:

```
Show[phaseplot];
```



The largest sustainable harvest rate r is about $r \approx 0.6$:

```
r = 0.6;
phaseline = PhaseLine[f[t, y], {y, ylow, yhigh}, Red, r];
newphaseplot = Show[phaseplot, phaseline,
  PlotRange -> {{rlow, rhigh}, {ylow, yhigh}}];
```

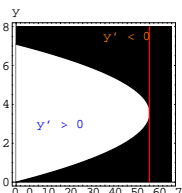


Grab the last two plot and animate.

□B.3.a.v) Estimating the bifurcation point

Take yet another look at the phase plot this time shown with a line indicating the largest sustainable harvest rate estimate:

```
r = 0.6;
bifurcationplot =
  Show[phaseplot, Graphics[{Red, Line[{r, ylow}, {r, yhigh}]}]]];
```



Lots of folks call this a bifurcation plot. Neat word.

Why do they call it by that funny name?

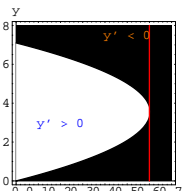
Estimate the bifurcation point.

□ Answer:

A bifurcation is any situation in which you see something forking into two branches.

And you are seeing it here:

```
Show[bifurcationplot];
```



Now look at the diffeq again;

```
Clear[r];
diffeq
{y'[t] == -r + 0.34 (1 - 0.140845 y[t]) y[t], y[0] == starter}
```

If you make r bigger than the r indicated in the plot, then ALL solutions of the diffeq decay to 0.

If you make r lower than the r indicated in the plot, then some of the solutions decay to 0, but others do not.

The cutoff point between the differences in solution behavior is the r indicated in the plot.

That's why this plot is called a bifurcation plot. And that's why the r indicated in the plot is called the bifurcation point.

B.4) Sensitive dependence on starter data

□B.4.a.i) Flow plots

Here's a simple differential equation:

```
Clear[diffeq, y, t, f, starter];
b = 10;
f[t_, y_] = y - 0.8 t;

setup =
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == -0.8 t + y[t]
y[0] == starter
```

And a flow plot together with the plots of two solutions::

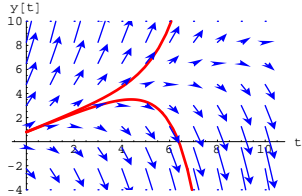
```
endtime = 6;
scalefactor = 0.4;
{ylow, yhigh} = {-4, 10};
{tlow, thigh} = {0, 10};

flowplot = Table[Arrow[{1, f[t, y]}, Tail -> {t, y},
VectorColor -> Blue, ScaleFactor -> scalefactor, HeadSize -> 0.7],
{t, tlow, thigh,  $\frac{thigh - tlow}{10}$ }, {y, ylow, yhigh,  $\frac{yhigh - ylow}{8}$ }}];
```

```
Clear[y1, y2, y3, t];
{starter1, starter2} = {0.79, 0.81};
endtime = 7;
y1[t_] = y[t] /.
NDSolve[diffeq /. starter -> starter1, y[t], {t, 0, endtime}][[1]];
y2[t_] = y[t] /.
NDSolve[diffeq /. starter -> starter2, y[t], {t, 0, endtime}][[1]];

solutionplots = Plot[{y1[t], y2[t]},
{t, 0, endtime}, PlotStyle -> {{Thickness[0.01], Red}},
AspectRatio ->  $\frac{1}{GoldenRatio}$ , PlotRange -> {ylow, yhigh},
AxesLabel -> {"t", "y[t]"}, DisplayFunction -> Identity];

Show[solutionplots, flowplot, DisplayFunction -> $DisplayFunction];
```



What do folks mean when they talk about "sensitive dependence on starter data?"

□Answer:

You're looking at it.

Take another look:

```
Show[solutionplots, flowplot, DisplayFunction -> $DisplayFunction];
```

What you see are two solutions of:

```
diffeq
{y'[t] == -0.8 t + y[t], y[0] == starter}
```

The starting points of the two solutions are almost the same; yet the two solutions are totally different in character.

You gotta agree that small changes in the starting conditions can result in big changes in solutions. This is exactly what folks mean when they talk about sensitive dependence on starting data.

□B.4.a.ii)

Do all differential equations exhibit sensitive dependence on starter data?

□Answer:

Not all of them.

Here's one that doesn't exhibit sensitive dependence on starter data:

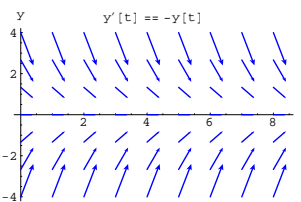
```
Clear[f, t, y];
b = 10;
f[t_, y_] = -y;

setup =
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == -y[t]
y[0] == starter

endtime = 6;
scalefactor = 0.4;
{ylow, yhigh} = {-4, 4};
{tlow, thigh} = {0, 8};

flowplot = Table[
Arrow[scalefactor {1, f[t, y]}, Tail -> {t, y}, VectorColor -> Blue],
{t, tlow, thigh,  $\frac{thigh - tlow}{8}$ }, {y, ylow, yhigh,  $\frac{yhigh - ylow}{6}$ }}];

Show[flowplot, Axes -> True, AxesLabel -> {"t", "y"},
AspectRatio ->  $\frac{1}{GoldenRatio}$ , PlotLabel -> setup[[1, 1]]];
```



The flow plot tells you to expect all solutions to be sucked onto the t -axis.

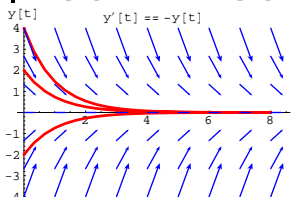
Here it is for three solutions:

```
Clear[y1, y2, y3, t];
{starter1, starter2, starter3} = {-2.0, 2.0, 4.0};
endtime = thigh;

y1[t_] = y[t] /.
NDSolve[diffeq /. starter -> starter1, y[t], {t, 0, endtime}][[1]];
y2[t_] = y[t] /.
NDSolve[diffeq /. starter -> starter2, y[t], {t, 0, endtime}][[1]];
y3[t_] = y[t] /.
NDSolve[diffeq /. starter -> starter3, y[t], {t, 0, endtime}][[1]];

solutionplots = Plot[{y1[t], y2[t], y3[t]},
{t, 0, endtime}, PlotStyle -> {{Thickness[0.01], Red}},
AspectRatio ->  $\frac{1}{GoldenRatio}$ , PlotRange -> {ylow, yhigh},
AxesLabel -> {"t", "y[t]"}, DisplayFunction -> Identity];

Show[solutionplots, flowplot, PlotLabel -> setup[[1, 1]],
DisplayFunction -> $DisplayFunction];
```



If two solutions of this differential equation have nearby starting values on $y[0]$, then the corresponding solutions will remain close to each other forever.

This differential equation is nearly insensitive to starter data.

□B.4.b) Using the phase line to detect extreme sensitivity to errors in starter data

Here's an autonomous diffeq:

```
Clear[diffeq, y, t, f, starter];
b = 6;
c = 2;
f[t_, y_] = 0.31 y (1 - y/b) (1 - y/c);

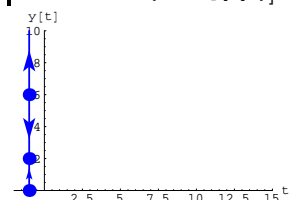
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == 0.31 (1 - y[t]/b) (1 - y[t]/c) y[t]
y[0] == starter
```

Here is a phase line for this diffeq:

```
{y_low, y_high} = {0, 10};
{t_low, t_high} = {0, 15};

phaseline = PhaseLine[f[t, y], {y, y_low, y_high}, Blue, -1];

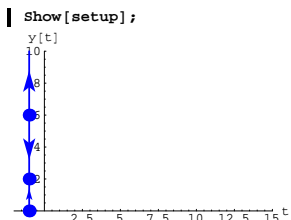
setup = Show[phaseline, AspectRatio -> 1/GoldenRatio,
  Axes -> True, PlotRange -> {{t_low - 2, t_high}, {y_low, y_high}},
  AxesLabel -> {"t", "y[t]"}];
```



Use the phase line to determine starting values on $y[0]$ at which this diffeq is extremely sensitive to error in $y[0]$.

□Answer:

Take another look:



If $y[0]$ is just below 6, then solutions go down and level off at $y = 2$.

If $y[0]$ is just above 6, then solutions go up.

So you have sensitive dependence on starter data for $y[0]$ near 6.

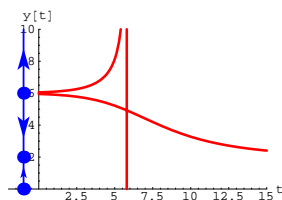
Check it out:

```
endtime = thigh;
{starter1, starter2} = {5.95, 6.05};

y1[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter1, y[t], {t, 0, endtime}][[1]];
y2[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter2, y[t], {t, 0, endtime}][[1]];

solutionplots = Plot[{y1[t], y2[t]},
  {t, 0, endtime}, PlotStyle -> {{Thickness[0.01], Red}},
  AspectRatio -> 1/GoldenRatio, PlotRange -> {y_low, y_high},
  AxesLabel -> {"t", "y[t]"}, DisplayFunction -> Identity];

Show[setup, solutionplots, DisplayFunction -> $DisplayFunction];
```



Pay no attention to the vertical line; it is a bug in *Mathematica*.

Now that's sensitive!

□B.4.c) Using bifurcation plots to detect extreme sensitivity to errors in starter data

Experience with B.3) will be helpful here.

You are the manager of the Red Oak Catfish Farm in Hartland, Wisconsin. Today your attention is focused on a lake where the logistic model

$$y'[t] = a y[t] \left(1 - \frac{y[t]}{b}\right) \text{ with } 0 < a \text{ and } 0 < b$$

is used to estimate the fish population t weeks after the lake is stocked.

Incorporate a constant weekly harvest rate r . The model

$$y'[t] = a y[t] \left(1 - \frac{y[t]}{b}\right) - r,$$

is reasonable.

The specifics for this lake are in the code below:

```
In the code below:
  -> t is measured in weeks
  -> y[t] and b are measured in thousands of fish.
  -> The harvest rate r is measured in thousands of fish per week

Clear[diffeq, y, t, f, r, starter];
a = 0.34;
b = 7.1;
f[t_, y_] = a y (1 - y/b) - r;

setup =
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
```

```
y'[t] == -r + 0.34 (1 - 0.140845 y[t]) y[t]
y[0] == starter
```

The secrets of this model are contained in the function $f[t, y]$:

```
f[t, y]
-r + 0.34 (1 - 0.140845 y)
```

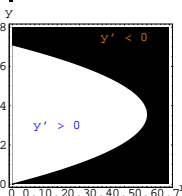
For each constant harvest rate r , this diffeq is autonomous.

Reason: $f[t, y]$ has no dependence on t .

Take a look at this labeled contour plot of $y' = f[t, y]$ as a function of y and the harvest rate r :

```
{y_low, y_high} = {0, 8};
{r_low, r_high} = {0, 0.7};

phaseplot = ContourPlot[f[t, y], {r, r_low, r_high},
  {y, y_low, y_high}, ContourSmoothing -> Automatic,
  PlotPoints -> 50, Contours -> {0}, Axes -> True,
  AxesLabel -> {"r", "y"}, Epilog -> {{Blue, Text["y' > 0", {0.2, 3}]},
  {Orange, Text["y' < 0", {0.5, 7.5}]}];
```



The light region indicates the $\{r, y\}$'s for which $y' = f[t, y] > 0$ and the dark region indicates the $\{r, y\}$'s for which $y' = f[t, y] < 0$.

What does this plot tell you about sensitive dependence on starter conditions for this diffeq?

□Answer:

Take another look:

```
Show[phaseplot];
```



```
y'[t] == a y[t] (-bt + y[t])
```

Ask *Mathematica* for a solution this way:

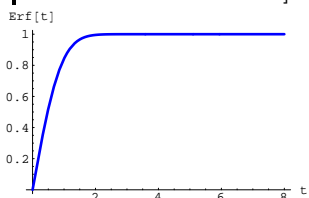
```
thisa = 0.3; thisb = 0.4;
formulasol = DSolve[{diffeq, y[0] == starter}, y[t], t] /.
{a -> thisa, b -> thisb, starter -> thisstarter}
{{y[t] ->  $\frac{1.13842 E^{-0.06 t^2}}{1.26491 - 1.23564 \operatorname{Erf}[0.244949 t]}$ }}
```

Fish out the formula:

```
Clear[yformula];
yformula[t_] = Chop[y[t] /. formulasol[[1]]]
 $\frac{1.13842 E^{-0.06 t^2}}{1.26491 - 1.23564 \operatorname{Erf}[0.244949 t]}$ 
```

As $t \rightarrow \infty$, $\operatorname{Erf}[t] \rightarrow 1$:

```
Plot[Erf[t], {t, 0, 8}, PlotStyle -> {{Thickness[0.01], Blue}},
PlotRange -> All, AspectRatio ->  $\frac{1}{\text{GoldenRatio}}$ ,
AxesLabel -> {"t", "Erf[t]"}];
```

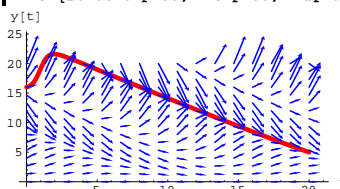


So as $t \rightarrow \infty$, the dominant term is the $e^{-0.06 t^2}$ term which goes to 0 as $t \rightarrow \infty$. This forces the solution to go to 0 as $t \rightarrow \infty$:

```
Limit[yformula[t], t -> \infty]
0
```

Now you know for sure that the solution plotted above is 0 in the global scale.

```
PlotStyle -> {{Thickness[0.015], Red}}, AxesLabel -> {"t", "y[t]"},
AspectRatio ->  $\frac{1}{2}$ , DisplayFunction -> Identity];
Show[solutionplot, flowplot, DisplayFunction -> $DisplayFunction];
```



It looks like this particular solution gets sucked onto a straight line. How can you tell for sure?

□ Answer:

You could ask *Mathematica* for a formula for the solution, but *Mathematica* will fail. This is not because of any deficiency of *Mathematica*.

No machine, no person, no anything has ever come up with a formula for the solution of this differential equation.

In this case, you'll have to depend on the visual evidence because the confirming formula is simply not available.

□ B.5.a.iii) Formulas are luxuries

What's the moral?

□ Answer:

Professor Peter Lax, former President of the American Mathematical Society, once said that the class of differential equations whose solutions can be described by exact formulas is "pitifully small."

□ B.5.a.ii)

If you have a formula for $f[t, y]$, then you can always plot the flow of solutions of

$$y'[t] = f[t, y[t]].$$

For example:

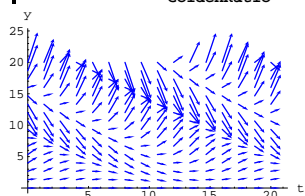
```
Clear[f, t, y];
f[t_, y_] = 0.3 y Sin[0.4 (y + t)];
diffeq = (y'[t] == f[t, y[t]])
y'[t] == 0.3 Sin[0.4 (t + y[t])] y[t]
```

And a flow plot:

```
scalefactor = 0.8;
{y_low, y_high} = {0, 20};
{t_low, t_high} = {0, 20};

flowplot = Table[Arrow[scalefactor {1, f[t, y]}, Tail -> {t, y},
VectorColor -> Blue, ScaleFactor -> 1, HeadSize -> 0.6],
{t, t_low, t_high,  $\frac{t_{high} - t_{low}}{15}$ }, {y, y_low, y_high,  $\frac{y_{high} - y_{low}}{15}$ }}];

Show[flowplot, Axes -> True, AxesLabel -> {"t", "y"},
AspectRatio ->  $\frac{1}{\text{GoldenRatio}}$ ];
```



And you can plot a sample solution:

```
Clear[y, ysol, t];
endtime = t_high;
starter = 16;

sol = NDSolve[{diffeq, y[0] == starter}, y[t], {t, 0, endtime}];
ysol[t_] = y[t] /. sol[[1]];

solutionplot = Plot[ysol[t], {t, 0, endtime}, PlotRange -> All,
```

Students in differential equations courses of the past have been forced to concentrate on this small class to the exclusion of others. You have bigger fish to fry.

Think of it this way:

If you can get a neat simple formula for a solution of a given differential equation, then you are very lucky.

According to Lax, the main thing to remember is "the general idea that every differential equation has a solution and that this solution is uniquely determined by initial data. . . That today we can use computers to explore the solutions of [differential] equations is truly revolutionary; we are only beginning to glimpse the consequences."

You are in on the ground floor of this revolution.

DE.05 First Order Differential Equations Tutorials

T.1) Setting harvest rates to control the catfish population

This is a problem from an area of mathematical engineering called control theory. If your university has an electrical engineering department, you're likely to find control theory folks. Some math departments also have control theory folks as well.

□T.1.a.i)

You are the manager of the Black Oak Catfish Farm in Moberly, Missouri. One day a team from the University of Missouri College of Agriculture stops by with some advice based on new research. First, they tell you that pushing the population too high invites disease and death of fish. They go on to say that you should try to set your harvest rate so that the long term fish population is 0.7 times the carrying capacity of the lake.

You decide to apply this to a certain lake modeled by

$$y'[t] = a y[t] \left(1 - \frac{y[t]}{b}\right) - r$$

where $y[t]$ measures the population of the lake (in thousands), $a = 0.12$, $b = 9$ (thousand) and r is your weekly harvest rate (in thousands).

Come up with an estimate of the r that does this.

□Answer:

Knowing that a bifurcation diagram reveals a heckuva lot, you whip out your PowerPC laptop, ignite *Mathematica* and set up your model:

```
In the code below:
→ t is measured in weeks
→ y[t] and b are measured in thousands of fish.
→ The harvest rate r is measured in thousands of fish per week

Clear[diffeq, y, t, f, r, starter];
a = 0.12;
b = 9;
f[t_, y_] = a y (1 - y/b) - r;

model =
(diffeq == {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == -r + 0.12 (1 - y[t]/9) y[t]
y[0] == starter
```

You know that the secrets of the model are contained in:

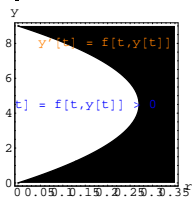
```
f[t, y]
```

$$-r + 0.12 \left(1 - \frac{y}{9}\right) y$$

You make this labeled contour plot of $f[t, y]$ as a function of y and the weekly harvest rate r :

```
{ylow, yhigh} = {0, b};
{rlow, rhigh} = {0, 0.35};

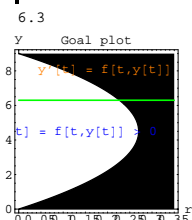
phaseplot = ContourPlot[f[t, y], {r, rlow, rhigh},
{y, ylow, yhigh}, ContourSmoothing → Automatic, PlotPoints → 50,
Contours → {0}, Axes → True, AxesLabel → {"r", "y"},
Epilog → {{Orange, Text["y'[t] = f[t, y[t]] < 0", {0.23, 0.9 b}]},
{Blue, Text["y'[t] = f[t, y[t]] > 0", {0.125, 0.5 b}]}];
```



The light region indicates the $\{r, y\}$'s for which $y' = f[t, y] > 0$ and the dark region indicates the $\{r, y\}$'s for which $y' = f[t, y] < 0$.

The carrying capacity of the lake is b and you want to control the long term population to be $0.7b$, so you plot:

```
ygoal = 0.7 b
goalplot = Show[phaseplot, Graphics[
{Green, Thickness[0.01], Line[{{rlow, ygoal}, {rhigh, ygoal}}]},
PlotLabel → "Goal plot"];
6.3
```



You want

$$y'[t] = 0 \text{ when } y[t] = ygoal$$

(so that the population doesn't change when it reaches $ygoal$), so you want to set r so that $\{r, ygoal\}$ is the point at which the goal line goes from the light region (where $y'[t] > 0$) to the black region (where $y'[t] < 0$).

Because

$$y'[t] = f[t, y[t]],$$

the r you want is the r that makes

$$f[t, ygoal] = 0:$$

```
Clear[r];
controleqn = 0 == f[t, y[t]] /. y[t] → ygoal;
controlsol = Solve[controleqn, r];

rcontroller = r /. controlsol[[1]]
0.2268
```

Or:

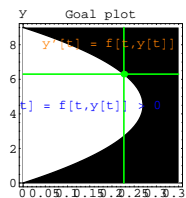
```
FindRoot[controleqn, {r, 0.2}]
{r → 0.2268}
```

This is the harvest rate the model tells you to use.

Take a look:

```
controlline = Graphics[{Green, Thickness[0.01],
Line[{{rcontroller, ylow}, {rcontroller, yhigh}}]}];

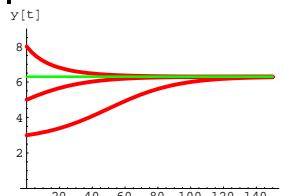
Show[goalplot, controlline,
Graphics[{Green, PointSize[0.04], Point[{{rcontroller, ygoal}}]}];
```



Try out this harvest rate r for several starting values on $y[0]$:

```
r = rcontroller;
Clear[y1, y2, y3, t];
{starter1, starter2, starter3} = {3, 5, 8};
endtime = 150;
y1[t_] = y[t] /.
NDSolve[diffeq /. starter → starter1, y[t], {t, 0, endtime}][[1]];
y2[t_] = y[t] /.
NDSolve[diffeq /. starter → starter2, y[t], {t, 0, endtime}][[1]];
y3[t_] = y[t] /.
NDSolve[diffeq /. starter → starter3, y[t], {t, 0, endtime}][[1]];

Plot[{y1[t], y2[t], y3[t]}, {t, 0, endtime},
PlotStyle → {{Thickness[0.015], Red}}, AspectRatio → GoldenRatio,
PlotRange → {ylow, yhigh}, AxesLabel → {"t", "y[t]"}, Epilog →
{Green, Thickness[0.01], Line[{{0, ygoal}, {endtime, ygoal}}]}];
```



Yes!

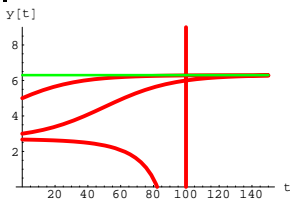
The long term population seems to be controlled just as it was supposed to be.

□T.1.a.ii)

Stay with the model in part i) above, and make sure all parts of part i) are live on your machine and look at this:

```
r = rcontroller;
Clear[y1, y2, y3, t];
{starter1, starter2, starter3} = {2.67, 3.0, 5.0};
endtime = 150;
y1[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter1, y[t], {t, 0, endtime}][[1]];
y2[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter2, y[t], {t, 0, endtime}][[1]];
y3[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter3, y[t], {t, 0, endtime}][[1]];

Plot[{y1[t], y2[t], y3[t]}, {t, 0, endtime},
  PlotStyle -> {{Thickness[0.015], Red}}, AspectRatio ->  $\frac{1}{\text{GoldenRatio}}$ ,
  PlotRange -> {ylow, yhigh}, AxesLabel -> {"t", "y[t]"}, Epilog ->
  {Green, Thickness[0.01], Line[{0, ygoal], {endtime, ygoal}}]}];
```



Disregard the vertical line.

Apparently setting the harvest rate

$r = rcontroller$
works for some starting values on $y[0]$ and not for others.

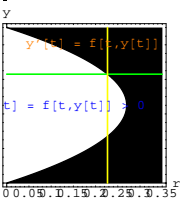
How big does the starting value on $y[0]$ have to be to guarantee that setting

$r = rcontroller$
does the job it was selected to do?

□Answer:

Look at:

```
startplot =
  Show[phaseplot, goalplot, Graphics[{Yellow, Thickness[0.01],
    Line[{rcontroller, ylow}, {rcontroller, yhigh}]}]];
```



If you go with a starter value on $y[0]$ so that $rcontroller, y[0]$ lands on the plotted line in the lower part of the black region, then you are in trouble because you are starting with $y'[0] < 0$ and things will only get worse as time goes on.

You can estimate from the plot that

$r = rcontroller$

will not work for starting values on

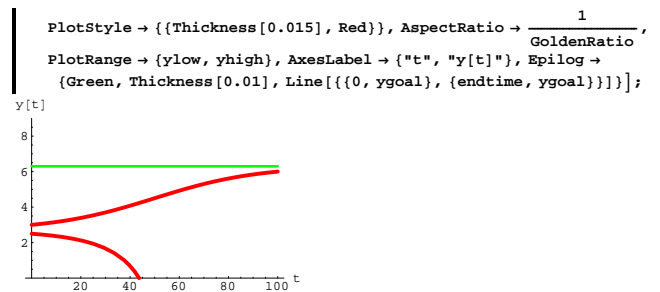
$y[0] < 2.5$

and that it will work fine for starting values on

$y[0] > 3.0$:

```
r = rcontroller;
Clear[y1, y2, t];
{starter1, starter2} = {2.5, 3.0};
endtime = 100;
y1[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter1, y[t], {t, 0, endtime}][[1]];
y2[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter2, y[t], {t, 0, endtime}][[1]];

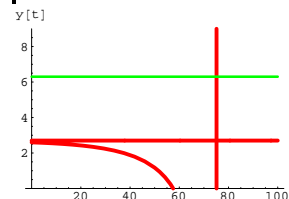
twosolutions = Plot[{y1[t], y2[t]}, {t, 0, endtime},
```



You can refine your estimate of the critical starting value by plotting:

```
Clear[y1, y2, t];
{starter1, starter2} = {2.6, 2.7};
endtime = 100;
y1[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter1, y[t], {t, 0, endtime}][[1]];
y2[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter2, y[t], {t, 0, endtime}][[1]];

twosolutions = Plot[{y1[t], y2[t]}, {t, 0, endtime},
  PlotStyle -> {{Thickness[0.015], Red}}, AspectRatio ->  $\frac{1}{\text{GoldenRatio}}$ ,
  PlotRange -> {ylow, yhigh}, AxesLabel -> {"t", "y[t]"}, Epilog ->
  {Green, Thickness[0.01], Line[{0, ygoal], {endtime, ygoal}}]}];
```



The flow plot confirms that when you go with $y[0] > 2.7$, then the control will work, but when you go with $y[0] < 2.6$, then the control

will not work. With further experimentation, you can refine this estimate.

□T.1.a.iii)

The University of Missouri team comes back to your fish farm and you excitedly tell them what you've learned. They are happy to see your analysis. They go on to say, "The model

$$y'[t] = a y[t] \left(1 - \frac{y[t]}{b}\right) - r$$

treats every day the same way. Bright summer days make for faster growth and higher lake capacity than the dark days of winter."

In fact, the proportion of the day getting daylight is given by this function (with t measuring weeks since January 1):"

```
Clear[daylightproportion, t];
daylightproportion[t_] =  $\frac{1}{24} \left(12.00 + 2.4 \sin\left[0.0172 \left(\frac{365 t}{52} - 80\right)\right]\right)$ 
 $\frac{1}{24} \left(12. + 2.4 \sin\left[0.0172 \left(-80 + \frac{365 t}{52}\right)\right]\right)$ 
```

You plot this function:

```
Plot[daylightproportion[t], {t, 0, 52},
  PlotStyle -> {{Thickness[0.01], Red}}, AspectRatio ->  $\frac{1}{4}$ ];
```

You say: "This looks good because it's high in the summer and low in the winter. Now let me check it by calculating

$$\frac{1}{52} \int_0^{52} \text{daylightproportion}[t] dt:"$$

```
 $\frac{1}{52}$  NIntegrate[daylightproportion[t], {t, 0, 52}]
0.500081
```

And you react: "I like it because now I know for sure that the average value of daylightproportion[t] is very close to 0.5, as it should be."

The University of Missouri teams says that the growth rate varies with sunlight as does the lake capacity b . And they say that you should refine the model by replacing the constant growth factor a by a

function whose average value is a , but is proportional to the hours of daylight on day t and whose average value is 1 like this:

```
Clear[bettera];
bettera[t_] = a 2 daylightproportion[t]
0.01 (12. + 2.4 Sin[0.0172 (-80 +  $\frac{365 t}{52}$ )])
```

And you should replace the constant lake capacity b by b times the same function:

```
Clear[betterb];
betterb[t_] = b 2 daylightproportion[t]
 $\frac{3}{4}$  (12. + 2.4 Sin[0.0172 (-80 +  $\frac{365 t}{52}$ )])
```

You say: "Let's see what this refined model predicts for two years with a weekly harvest rate $r = 0.2$ (thousand) fish but starting out with 4000 fish."

```
r = 0.2;
Clear[diffeq, y, t, f, starter];

f[t_, y_] = bettera[t] y (1 -  $\frac{y}{\text{betterb}[t]}$ ) - r;
model = (diffeq = {y'[t] == f[t, y[t]], y[0] == starter});
diffeq // ColumnForm

Clear[y1, y2, t];
{starter1} = {4};
endtime = 2 52;
ygoal = 6.3;
y1[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter1, y[t], {t, 0, endtime}][[1]];

solutionplot = Plot[{y1[t]}, {t, 0, endtime},
  PlotStyle -> {{Thickness[0.015], Red}}, AspectRatio ->  $\frac{1}{\text{GoldenRatio}}$ ,
  AxesLabel -> {"t", "y[t]"}, DisplayFunction -> Identity];

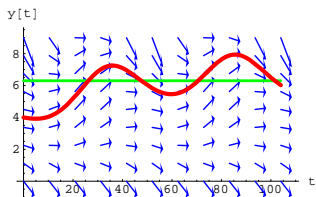
goalplot = Graphics[
  {Green, Thickness[0.01], Line[{0, ygoal], {endtime, ygoal}}]};
scalefactor = 4.5;
{y1low, y1high} = {0, 9};

flowplot = Table[Arrow[{1, f[t, y]}, Tail -> {t, y},
  VectorColor -> Blue, ScaleFactor -> scalefactor, HeadSize -> 1.3],
  {t, 0, endtime,  $\frac{\text{endtime}}{10}$ }, {y, y1low, y1high,  $\frac{y1high - y1low}{8}$ }]];
```

```
{t, 0, endtime,  $\frac{\text{endtime}}{10}$ }, {y, y1low, y1high,  $\frac{y1high - y1low}{8}$ }]];
```

```
Show[flowplot, goalplot, solutionplot, AspectRatio ->  $\frac{1}{\text{GoldenRatio}}$ ,
  Axes -> True, AxesLabel -> {"t", "y[t]"}, PlotRange -> All,
  DisplayFunction -> $DisplayFunction];
```

```
y'[t] == -0.2 + 0.01 (12. + 2.4 Sin[0.0172 (-80 +  $\frac{365 t}{52}$ )]) y[t] (1 -  $\frac{y[t]}{3(12. + 2.4 Sin[0.0172 (-80 + \frac{365 t}{52})])}$ )
y[0] == starter
```



You say: "I like this. Now let me try to use a variable harvest rate $r[t]$ to control the fish population so that, in the long term, $y[t]$ settles in on the goal of 6.3 (thousand) fish."

Do it. Estimate the size of the first year's harvest.

□ Answer:

You do the same thing you did above but incorporate a variable weekly harvest rate $r[t]$:

```
Clear[diffeq, y, t, r, f, starter];

f[t_, y_] = bettera[t] y (1 -  $\frac{y}{\text{betterb}[t]}$ ) - r[t];
model = (diffeq = {y'[t] == f[t, y[t]], y[0] == starter});
{y'[t] == -r[t] + 0.01 (12. + 2.4 Sin[0.0172 (-80 +  $\frac{365 t}{52}$ )])}
y[t] (1 -  $\frac{4 y[t]}{3 (12. + 2.4 Sin[0.0172 (-80 + \frac{365 t}{52})])}$ ),
y[0] == starter}
```

You want to maintain the fish population at 6.3 (thousand); so

$$y'[t] = f[t, y[t]]$$

should be 0 when $y[t] = 6.3$:

```
controlq = 0 == f[t, y[t]] /. y[t] -> 6.3
0 == -r[t] + 0.063 (12. + 2.4 Sin[0.0172 (-80 +  $\frac{365 t}{52}$ )])
(1 -  $\frac{8.4}{12. + 2.4 Sin[0.0172 (-80 + \frac{365 t}{52})]}$ )
```

Your variable weekly harvest rate $r[t]$ is:

```
sol = Solve[controlq, r[t]];
r[t_] = r[t] /. sol[[1]]
0.063 (12. + 2.4 Sin[0.0172 (-80 +  $\frac{365 t}{52}$ )])
(1 -  $\frac{8.4}{12. + 2.4 Sin[0.0172 (-80 + \frac{365 t}{52})]}$ )
```

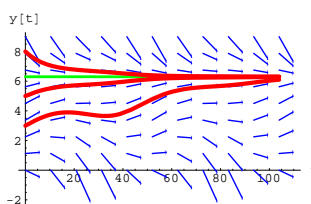
Check it out:

```
Clear[y1, y2, y3, t];
{starter1, starter2, starter3} = {3.0, 5.0, 8.0};
y1[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter1, y[t], {t, 0, endtime}][[1]];
y2[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter2, y[t], {t, 0, endtime}][[1]];
y3[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter3, y[t], {t, 0, endtime}][[1]];

solutionplot = Plot[{y1[t], y2[t], y3[t]}, {t, 0, endtime},
  PlotStyle -> {{Thickness[0.015], Red}}, DisplayFunction -> Identity];
scalefactor = 3.5;
{y1low, y1high} = {0, 9};

flowplot = Table[Arrow[{1, f[t, y]}, Tail -> {t, y},
  VectorColor -> Blue, ScaleFactor -> 6, HeadSize -> 0.6],
  {t, 0, endtime,  $\frac{\text{endtime}}{10}$ }, {y, y1low, y1high,  $\frac{y1high - y1low}{8}$ }]];

Show[flowplot, goalplot, solutionplot, AspectRatio ->  $\frac{1}{\text{GoldenRatio}}$ ,
  Axes -> True, AxesLabel -> {"t", "y[t]"}, PlotRange -> All,
  DisplayFunction -> $DisplayFunction];
```



Looks good and feels good.

In all these cases, the first year's harvest brings in this many fish:

```
1000 NIntegrate[r[t], {t, 0, 52}]
11800.
```

Not bad.

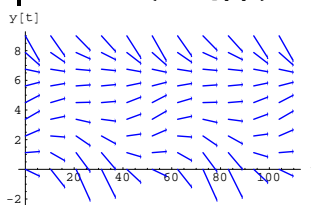
□ T.1.a.iv)

Keep everything (except the starting values on $y[0]$) the same as in the previous part and estimate the minimum starter value on $y[0]$ that will guarantee that the fish population will not die out.

□ Answer:

Take a look at the flow plot:

```
Show[flowplot, AspectRatio ->  $\frac{1}{\text{GoldenRatio}}$ , Axes -> True,
  AxesLabel -> {"t", "y[t]"}, PlotRange -> All];
```



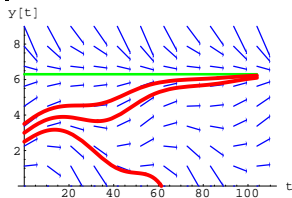
It looks like the minimum starter value of $y[0]$ that will guarantee that the fish population will not die out is around $y[0] \approx 3.0$.

Check it out:

```
Clear[y1, y2, y3, t];
{starter1, starter2, starter3} = {2.5, 3.0, 3.5};
y1[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter1, y[t], {t, 0, endtime}][[1];
y2[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter2, y[t], {t, 0, endtime}][[1];
y3[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter3, y[t], {t, 0, endtime}][[1];

solutionplot = Plot[{y1[t], y2[t], y3[t]}, {t, 0, endtime},
  PlotStyle -> {{Thickness[0.015], Red}}, DisplayFunction -> Identity];

Show[flowplot, goalplot, solutionplot, AspectRatio ->  $\frac{1}{\text{GoldenRatio}}$ ,
  Axes -> True, AxesLabel -> {"t", "y[t]"}, PlotRange -> {ylow, yhigh},
  DisplayFunction -> $DisplayFunction];
```



The three starter values in the plot above were {2.5, 3.0, 3.5}, and the starter value of 2.5 did not lead to a sustainable harvest. This leads to the estimate $y[0] \approx 3.0$ for the smallest starter value for a sustainable harvest.

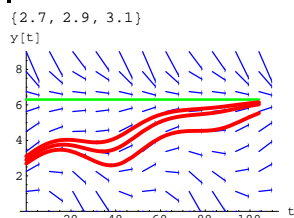
You can refine this estimate :

```
Clear[y1, y2, y3, t];
{starter1, starter2, starter3} = {2.7, 2.9, 3.1}
y1[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter1, y[t], {t, 0, endtime}][[1];
```

```
y2[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter2, y[t], {t, 0, endtime}][[1];
y3[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter3, y[t], {t, 0, endtime}][[1];

solutionplot = Plot[{y1[t], y2[t], y3[t]}, {t, 0, endtime},
  PlotStyle -> {{Thickness[0.015], Red}}, DisplayFunction -> Identity];

Show[flowplot, goalplot, solutionplot, AspectRatio ->  $\frac{1}{\text{GoldenRatio}}$ ,
  Axes -> True, AxesLabel -> {"t", "y[t]"}, PlotRange -> {ylow, yhigh},
  DisplayFunction -> $DisplayFunction];
```



The three starter values for this plot were: {2.7, 2.9, 3.1}.

A good estimate of the minimum starter value on $y[0]$ that will guarantee that the fish population will not die out is

$$y[0] \approx 2.7$$

but when you start with this value, you are playing with fire.

T.2) Hand symbol manipulation: Separating the variables

□T.2.a)

All oldtime DiffEq folks have heard of the method of separation of variables.

Just what is this famous procedure?

□Answer:

It's a way of using symbolic manipulation to go after an exact formula for the solution of certain differential equations. It is normally used only in simple, special cases.

Here it is in action on the following diffeq:

```
Clear[diffeq, y, t, f, r, starter];
f[t_, y_] = -2.1 y^2;
(diffeq = {y'[t] == f[t, y[t]], y[0] == 3.1}) // ColumnForm
y'[t] == -2.1 y[t]^2
y[0] == 3.1
```

Before *Mathematica* will help you, you have to re-enter the diffeq with cleared constants this way:

```
thisa = -2.1;
thisstarter = 3.1;
Clear[a, y, t, diffeq, y, t, f, r, starter];
f[t_, y_] = a y^2 t;
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == a t y[t]^2
y[0] == starter
```

Separating the variables involves putting all the $y[t]$ and $y'[t]$ terms on the left and leaving the t terms on the right like this:

```
newleft =  $\frac{y'[t]}{y[t]^2}$ ;
newright = a t;
samediffeq = newleft == newright
 $\frac{y'[t]}{y[t]^2} == a t$ 
```

Now integrate both sides from 0 to t :

```
integratedeqn =  $\int_0^t$  newleft dt ==  $\int_0^t$  newright dt
 $\frac{1}{y[0]} - \frac{1}{y[t]} == \frac{a t^2}{2}$ 
```

Replace $y[0]$ by starter:

```
neweqn = integratedeqn /. y[0] -> starter
 $\frac{1}{\text{starter}} - \frac{1}{y[t]} == \frac{a t^2}{2}$ 
```

Now solve for $y[t]$

```
Clear[symbolformulay];
ysol = Solve[neweqn, y[t]];
symbolformulay[t_] = y[t] /. ysol[[1]]
 $-\frac{2 \text{starter}}{-2 + a \text{starter } t^2}$ 
```

Now replace a with the given a and starter with the given starter:

```
Clear[formulay];
formulay[t_] =
  symbolformulay[t] /. {a -> thisa, starter -> thisstarter}
 $-\frac{6.2}{-2 - 6.51 t^2}$ 
```

Done.

□T.2.b)

Here's another diffeq:

```
Clear[diffeq, y, t, f, r, b, starter];
f[t_, y_] = t (1.2 + 2.4 y^2);
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == t (1.2 + 2.4 y[t]^2)
y[0] == starter
```

Use separation of variables to try to come up with a formula for the solution.

□Answer:

Take another look:

```
Clear[diffeq, y, t, f, r, b, starter];
f[t_, y_] = t (1.2 + 2.4 y^2);
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == t (1.2 + 2.4 y[t]^2)
y[0] == starter
```

Before *Mathematica* will help you, you have to re-enter the diffeq with cleared constants this way:

```

thisa = 1.2;
thisb = 2.4;
Clear[a, y, t, diffeq, y, t, f, r, starter];
f[t_, y_] = t (a + b y^2);

diffeq = {y'[t] == f[t, y[t]], y[0] == starter}
{y'[t] == t (a + b y[t]^2), y[0] == starter}

```

Separating the variables involves putting all the $y[t]$ and $y'[t]$ terms on the left and leaving the t terms on the right like this:

```

newleft =  $\frac{y'[t]}{a + b y[t]^2}$ ;
newright = t;
samediffeq = newleft == newright
 $\frac{y'[t]}{a + b y[t]^2} == t$ 

```

Now integrate both sides from 0 to t :

```

integratedeqn =  $\int_0^t$  newleft dt ==  $\int_0^t$  newright dt
 $-\frac{\text{ArcTan}\left[\frac{\sqrt{b} y[0]}{\sqrt{a}}\right]}{\sqrt{a} \sqrt{b}} + \frac{\text{ArcTan}\left[\frac{\sqrt{b} y[t]}{\sqrt{a}}\right]}{\sqrt{a} \sqrt{b}} == \frac{t^2}{2}$ 

```

Replace $y[0]$ by *starter*:

```

neweqn = integratedeqn /. y[0] -> starter
 $-\frac{\text{ArcTan}\left[\frac{\sqrt{b} \text{starter}}{\sqrt{a}}\right]}{\sqrt{a} \sqrt{b}} + \frac{\text{ArcTan}\left[\frac{\sqrt{b} y[t]}{\sqrt{a}}\right]}{\sqrt{a} \sqrt{b}} == \frac{t^2}{2}$ 

```

Now solve for $y[t]$

```

Clear[symbolformulay];
ysol = Solve[neweqn, y[t]];
symbolformulay[t_] = y[t] /. ysol[[1]]
 $\frac{\sqrt{a} \text{Tan}\left[\frac{1}{2} \left(\sqrt{a} \sqrt{b} t^2 + 2 \text{ArcTan}\left[\frac{\sqrt{b} \text{starter}}{\sqrt{a}}\right]\right)\right]}{\sqrt{b}}$ 

```

Now replace a with the given a and b with the given b :

```

Clear[formulay]
formulay[t_] = symbolformulay[t] /. {a -> thisa, b -> thisb}
0.707107 Tan[ $\frac{1}{2} (1.69706 t^2 + 2 \text{ArcTan}[1.41421 \text{starter}])$ ]

```

Done.

□T.2.c) The logistic diffeq

Here's the logistic diffeq:

$$y'[t] = r y[t] \left(1 - \frac{y[t]}{b}\right)$$

with $y[0] = \text{starter}$.

```

Clear[diffeq, y, t, f, r, b, starter];
f[t_, y_] = r y (1 -  $\frac{y}{b}$ );

(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == r y[t] (1 -  $\frac{y[t]}{b}$ )
y[0] == starter

```

Here's *Mathematica* cranking out a formula for the solution:

```

DSolve[diffeq, y[t], t]
{{y[t] ->  $\frac{b E^{r t} \text{starter}}{b - \text{starter} + E^{r t} \text{starter}}$ }}

```

Use separation of variables to explain where this formula comes from.

□Answer:

Look at the diffeq again:

```

diffeq
{y'[t] == r y[t] (1 -  $\frac{y[t]}{b}$ ), y[0] == starter}

```

Separate the variables:

```

newleft =  $\frac{y'[t]}{y[t] (1 - \frac{y[t]}{b})}$ ;
newright = r;
samediffeq = newleft == newright
 $\frac{y'[t]}{y[t] (1 - \frac{y[t]}{b})} == r$ 

```

Integrate both sides from 0 to t :

```

integratedeqn =  $\int_0^t$  newleft dt ==  $\int_0^t$  newright dt
-Log[y[0]] + Log[-b + y[0]] + Log[y[t]] - Log[-b + y[t]] == r t

```

Replace $y[0]$ with the (cleared) starter value on $y[0]$:

```

neweqn = integratedeqn /. y[0] -> starter
-Log[starter] + Log[-b + starter] + Log[y[t]] - Log[-b + y[t]] == r t

```

And solve for $y[t]$:

```

Clear[formulay];
ysol = Solve[neweqn, y[t]]
{{y[t] ->  $\frac{b E^{r t} \text{starter}}{b - \text{starter} + E^{r t} \text{starter}}$ }}

```

There it is.

□T.2.d) Sometimes separation of variables is useless

Here are two new differential equations:

```

Clear[diffeq, y, t, f, r, starter];
f[t_, y_] = -2.1 Sin[y + t] + y;

(diffeq1 = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == -2.1 Sin[t + y[t]] + y[t]
y[0] == starter

Clear[diffeq, y, t, f, r, starter];
f[t_, y_] = y^2 + t;

(diffeq2 = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == t + y[t]^2
y[0] == starter

```

Why is separation of variables useless for these two differential equations?

□Answer:

Look at the first one:

```

diffeq1
{y'[t] == -2.1 Sin[t + y[t]] + y[t], y[0] == starter}

```

Try as you may, you can't separate the $y[t]$'s from the t 's.

Look at the second:

```

diffeq2
{y'[t] == t + y[t]^2, y[0] == starter}

```

This time you can separate the variables:

```

newleft = y'[t] - y[t]^2;
newright = t;
samediffeq = newleft == newright
-y[t]^2 + y'[t] == t

```

But when you integrate both sides from 0 to t , you get:

```

integratedeqn =  $\int_0^t$  newleft dt ==  $\int_0^t$  newright dt
 $\int_0^t (-y[t]^2 + y'[t]) dt == \frac{t^2}{2}$ 

```

The integral on the left could not be done.

Mathematica has its own obscure ways of solving this:

```

DSolve[diffeq2, y[t], t]
{{y[t] -> -((-1)^{1/3} (AiryAiPrime[(-1)^{1/3} t] + (AiryBiPrime[(-1)^{1/3} t]
(-3^{1/6} starter Gamma[ $\frac{1}{3}$ ] + (-1)^{1/3} \sqrt{3} Gamma[ $\frac{2}{3}$ ])) /
(3^{2/3} starter Gamma[ $\frac{1}{3}$ ] + 3 (-1)^{1/3} Gamma[ $\frac{2}{3}$ ])) /
(AiryAi[(-1)^{1/3} t] + (AiryBi[(-1)^{1/3} t]
(-3^{1/6} starter Gamma[ $\frac{1}{3}$ ] + (-1)^{1/3} \sqrt{3} Gamma[ $\frac{2}{3}$ ])) /
(3^{2/3} starter Gamma[ $\frac{1}{3}$ ] + 3 (-1)^{1/3} Gamma[ $\frac{2}{3}$ ]))}}

```

But the formula is overwhelming.

To get control of this one, go to flow plots and `ndsolve`.

T.3) The danger zone: Where the unexpected sometimes happens and separation of variables produces bogus formulas

□T.3.a.i)

Here is an apparently innocent-looking differential equation:

```
Clear[diffeq, y, t, f, r, starter];
f[t_, y_] = 0.26 t (sqrt[1 - y^2]);
(diffeq = {y'[t] == f[t, y[t]], y[0] == 0.2}) // ColumnForm
y'[t] == 0.26 t sqrt[1 - y[t]^2]
y[0] == 0.2
```

You can try to generate your own formula for the solution of this differential equation by separating the variables:

Separate and integrate:

```
starter = 0.2;
a = 0.26;
Clear[symbolyformula];
newleft = y'[t] / sqrt[1 - y[t]^2];
newright = a t;
neweqn = (integrate newleft dt /. y[0] -> starter) == integrate newright dt;
ysol = Solve[neweqn, y[t]];
symbolyformula[t_] = N[ExpandAll[y[t] /. ysol[[1]], 6]
1. Sin[0.201358 + 0.13 t^2]
```

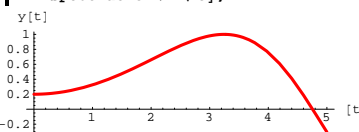
Don't worry about the error messages.

This is the same as the formula delivered by DSolve:

```
dsol = DSolve[{y'[t] == f[t, y[t]], y[0] == 0.2}, y[t], t];
N[ExpandAll[y[t] /. dsol[[1]], 6]
Sin[0.201358 + 0.13 t^2]
```

Here is the way this formula plots out:

```
endtime = 5;
formulaplot = Plot[symbolyformula[t], {t, 0, endtime},
PlotStyle -> {{Thickness[0.01], Red}}, AxesLabel -> {"t", "y[t]"},
AspectRatio -> 1/3];
```

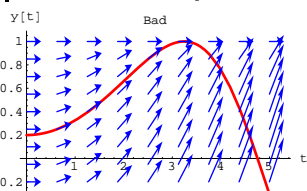


See this plot along with the flow plot for the given diffeq:

```
{y[low, y[high]} = {-0.2, 1.0};
{t[low, t[high]} = {0, endtime};

solutionflowplot = Table[Arrow[{1, f[t, y]}, Tail -> {t, y},
VectorColor -> Blue, ScaleFactor -> 0.3, HeadSize -> 0.15],
{t, t[low, t[high], (t[high] - t[low]) / 8}, {y, y[low, y[high], (y[high] - y[low]) / 8}];

badplot = Show[formulaplot, solutionflowplot, Axes -> True,
AxesLabel -> {"t", "y[t]"}, AspectRatio -> 1/GoldenRatio,
PlotLabel -> "Bad"];
```



The solution plot goes with the flow on the left but not on the right.

On the right, this plot is dead wrong.

Fix it.

□Answer:

This plot is also out of sync with the differential equation:

```
Clear[diffeq, y, t, f, r, starter];
f[t_, y_] = 0.26 t (sqrt[1 - y^2]);
(diffeq = {y'[t] == f[t, y[t]], y[0] == 0.2}) // ColumnForm
y'[t] == 0.26 t sqrt[1 - y[t]^2]
y[0] == 0.2
```

Because

$$0.26 t \sqrt{1 - y[t]^2} \geq 0 \text{ for } t \geq 0$$

and because

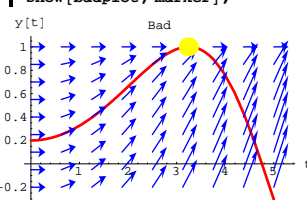
$$y'[t] = 0.26 t \sqrt{1 - y[t]^2} \text{ for } t \geq 0,$$

no solution of this differential equation can ever go down for $t \geq 0$.

This fact is key to understanding what the correct formula must be.

Take another look:

```
hitsol = FindRoot[symbolyformula[t] == 1, {t, 3}]
{t -> 3.24475}
marker = Graphics[{PointSize[0.07], Yellow, Point[{3.24, 1}]}];
Show[badplot, marker];
```



And look at the differential equation again:

```
diffeq
{y'[t] == 0.26 t sqrt[1 - y[t]^2], y[0] == 0.2}
```

When $y[t] < 1$, it's automatic that $y[t]$ goes up because

$$y'[t] = \sqrt{1 - y[t]^2} > 0.$$

But it's impossible for $y[t]$ to be > 1 .

Reason: If $y[t] > 1$, then $\sqrt{1 - y[t]^2}$ is an imaginary number.

The upshot:

The true solution $y[t]$ steadily increases until $y[t]$ gets to 1 at $t = 3.24519$.

For $t > 3.24$,

the true solution $y[t]$ stays equal to 1 because $y[t]$ cannot be bigger than 1 and $y[t]$ cannot go down.

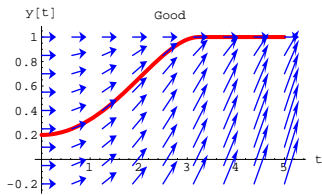
Thus, for $t > 3.24$, $y[t]$ is trapped equal to 1.

Watch the true solution go with the flow:

```
symbolyformula[t]
1. Sin[0.201358 + 0.13 t^2]
b = 3.24;
Clear[truesolution];
truesolution[t_] := 1 /; t > b;
truesolution[t_] := symbolyformula[t] /; t <= b

truesolutionplot = Plot[truesolution[t], {t, 0, endtime},
PlotStyle -> {{Thickness[0.015], Red}},
AxesLabel -> {"t", "y[t]"}, DisplayFunction -> Identity];

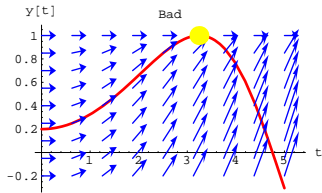
goodplot = Show[truesolutionplot, solutionflowplot, Axes -> True,
AxesLabel -> {"t", "y[t]"}, AspectRatio -> 1/GoldenRatio,
PlotLabel -> "Good", DisplayFunction -> $DisplayFunction];
```



Some folks call this a spline knotted at b.

Compare:

```
Show[badplot, marker];
```



Grab both plots, align and animate slowly.

□T.3.a.ii) The danger zone and the gradient test for the danger zone

Is there a way of getting an advance alert to sticky situations like this one?

□Answer:

You bet your sweet bippy.

There is a way and it's very simple:

```
Clear[diffeq, y, t, f, r, starter];
f[t_, y_] = 0.26 t (√(1 - y^2));
(diffeq = {y'[t] == f[t, y[t]], y[0] == 0.2}) // ColumnForm
y'[t] == 0.26 t √(1 - y[t]^2)
y[0] == 0.2
```

You just look at the gradient of $f[t, y]$:

```
Clear[gradf];
gradf[t_, y_] = {∂t f[t, y], ∂y f[t, y]}
{0.26 √(1 - y^2), -0.26 t y / √(1 - y^2)}
```

Notice $\text{gradf}[t, y]$ has a singularity (blow up) at $y = 1$ and $y = -1$.

```
gradf[t, 1]
{0, ComplexInfinity}
gradf[t, -1]
{0, ComplexInfinity}
```

Yepper.

This tells you that the possible danger zone for this differential equation consists of all points on the line

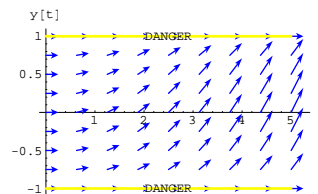
$$y[t] = 1$$

together with all points on the lines

$$y[t] = -1.$$

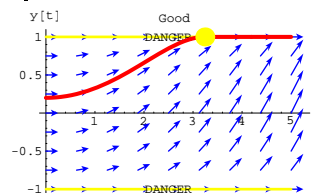
Take a gander:

```
{ydanger1, ydanger2} = {-1.0, 1.0};
{tlow, thigh} = {0, endtime};
dangerzone = {Graphics[{Yellow, Thickness[0.01],
  Line[{tlow, ydanger1}, {thigh, ydanger1}]}],
  Graphics[{Yellow, Thickness[0.01],
  Line[{tlow, ydanger2}, {thigh, ydanger2}]}]};
dangerlabel = {Graphics[Text["DANGER", {endtime/2, ydanger1}]}];
Graphics[Text["DANGER", {endtime/2, ydanger2}]}];
scalefactor = 0.25;
solutionflowplot = Table[Arrow[{1, f[t, y]}, Tail -> {t, y},
  VectorColor -> Blue, ScaleFactor -> scalefactor, HeadSize -> 0.15],
  {t, tlow, thigh, (thigh - tlow)/8},
  {y, ydanger1, ydanger2, (ydanger2 - ydanger1)/8}];
dangerflow = Show[solutionflowplot, dangerzone, dangerlabel,
  Axes -> True, AxesLabel -> {"t", "y[t]"}, AspectRatio -> GoldenRatio];
```

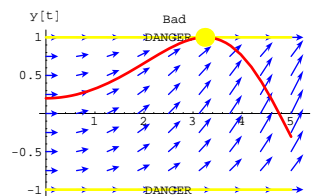


Compare:

```
Show[dangerflow, truesolutionplot, marker, PlotLabel -> "Good"];
```



```
Show[dangerflow, formulaplot, marker, PlotLabel -> "Bad"];
```



Anytime you see a hand or computer-generated formula plot out and actually touch a danger zone, you should become very alert to the possibility of having to do a careful analysis.

□T.3.a.iii)

Here's a diffeq:

```
Clear[diffeq, y, f, t, starter];
f[t_, y_] = -0.4 (√y);
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == -0.4 √y[t]
y[0] == starter
```

Plot the danger zone for this diffeq.

□Answer:

Calculate the gradient of $f[t, y]$:

```
Clear[gradf];
gradf[t_, y_] = {∂t f[t, y], ∂y f[t, y]}
{0, -0.2 / √y}
```

Note that $\text{gradf}[t, y]$ blows up for $y = 0$.

```
gradf[t, 0]
{0, ComplexInfinity}
```

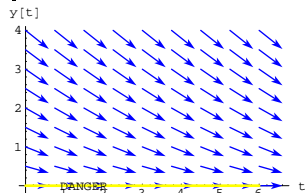
The singularities of $\text{gradf}[t, y]$ are all on the line

$$y = 0.$$

This curve is the danger zone for this diffeq.

Here comes a plot:

```
endtime = 6;
{ylow, yhigh} = {0, 4};
{tlow, thigh} = {0, endtime};
dangerzone = Plot[0, {t, 0, endtime}, PlotStyle ->
  {{Thickness[0.01], Yellow}}, DisplayFunction -> Identity];
dangerlabel = Graphics[Text["DANGER", {-endtime/4, 0}]];
scalefactor = 0.6;
flowplot = Table[Arrow[{1, f[t, y]}, Tail -> {t, y},
  VectorColor -> Blue, ScaleFactor -> scalefactor, HeadSize -> 0.3],
  {t, tlow, thigh, thigh - tlow/8}, {y, ylow, yhigh, yhigh - ylow/8}];
Show[flowplot, dangerzone, dangerlabel, Axes -> True,
  AxesLabel -> {"t", "y[t]"}, AspectRatio -> 1/GoldenRatio];
```



Anytime you see a solution hit this danger zone, you have to be on guard.

□T.3.a.iv)

Here's another diffeq

```
Clear[diffeq, y, f, t, starter];
f[t_, y_] = 0.4 Sin[t] + y^2;
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == 0.4 Sin[t] + y[t]^2
y[0] == starter
```

Plot the danger zone for this diffeq.

□Answer:

Calculate the gradient of $f[t, y]$:

```
Clear[gradf];
gradf[t_, y_] = {D[f[t, y], t], D[f[t, y], y]}
{0.4 Cos[t], 2 y}
```

Note that $\text{gradf}[t, y]$ never blows up for finite t or y .

This tells you that this diffeq has no danger zone. So there is nothing to plot.

This differential equation can't reach out and surprise you.

□T.3.a.v)

Try to explain why the gradient test for the danger zone works

□Answer:

Go with a cleared diffeq:

```
Clear[diffeq, y, t, f, r, starter];
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == f[t, y[t]]
y[0] == starter
```

Because

$$y'[t] = f[t, y[t]],$$

the chain rule from vector calc guarantees that

$$y''[t] = \text{gradf}[t, y[t]] \cdot \{1, y'[t]\}.$$

That little dot is a dot product.

Now here's the key point:

If $\{t, y[t]\}$ hits the danger zone (i.e. a singularity of $\text{gradf}[t, y[t]]$) at a

time $t = \text{tdanger}$,

then the fact that

$$y''[\text{tdanger}] = \text{gradf}[\text{tdanger}, y[\text{tdanger}]] \cdot \{1, y'[\text{tdanger}]\}$$

opens up the possibility of a singularity of

$$y''[t] \text{ at } t = \text{tdanger}.$$

When this happens things can get out of hand!

And they did in part i) above. In part i), the singularity in $y''[t]$ at $\text{tdanger} = 3.24$

made the formula for the solution change as t advanced though $\text{tdanger} = 3.24$

□T.3.b.i) When you're in the danger zone, your intuition may be thrown off

Here's another innocent-looking diffeq:

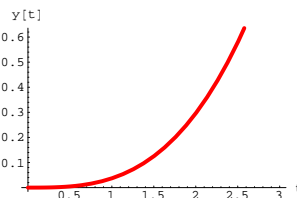
```
Clear[diffeq, y, t, f, starter];
f[t_, y_] = y^{2/3};
(diffeq = {y'[t] == f[t, y[t]], y[0] == 0}) // ColumnForm
y'[t] == y[t]^{2/3}
y[0] == 0
```

Go after a formula for the solution by separating variables:

```
Clear[yformula];
newleft = y'[t] / y[t]^{2/3};
newright = 1;
neweqn = (integrate newleft dt /. y[0] == 0) == integrate newright dt;
ysol = Solve[neweqn, y[t]];
formulay[t_] = y[t] /. ysol[[1]]
t^{3/27}
```

Now plot:

```
endtime = 3;
solutionplot = Plot[formulay[t], {t, 0, endtime},
  PlotStyle -> {{Thickness[0.015], Red}}, AxesLabel -> {"t", "y[t]"},
  AspectRatio -> 1/GoldenRatio];
```

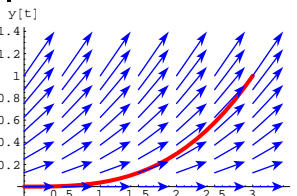


And you check by showing this solution plot with a flow plot:

```
scalefactor = 0.4;
{ylow, yhigh} = {0, 1};
{tlow, thigh} = {0, endtime};

solutionflowplot = Table[Arrow[{1, f[t, y]}, Tail -> {t, y},
  VectorColor -> Blue, ScaleFactor -> scalefactor, HeadSize -> 0.15],
  {t, tlow, thigh, thigh - tlow/6}, {y, ylow, yhigh, yhigh - ylow/8}];

Show[solutionplot, solutionflowplot, Axes -> True,
  AxesLabel -> {"t", "y[t]"}, AspectRatio -> 1/GoldenRatio];
```



The solution is going with the flow.

Is there anything wrong here?

□Answer:

Only one thing:

There is another solution with the same starter data:

This is the silly solution

$$y[t] = 0 \text{ for all } t \geq 0.$$

This $y[t]$ certainly solves the differential equation:

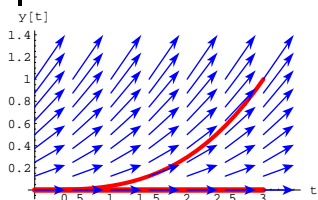
```
diffEq /. {y[t] -> 0, y'[t] -> 0, y[0] -> 0}
{True, True}
```

So the solution $y[t] = 0$ deserves to be plotted too:

```
Clear[altsol];
altsol[t_] = 0;

altsolplot = Plot[altsol[t], {t, 0, endtime},
  PlotStyle -> {{Thickness[0.02], Red}}, DisplayFunction -> Identity];

bummer = Show[altsolplot, solutionplot, solutionflowplot, Axes -> True,
  AxesLabel -> {"t", "y[t]"}, AspectRatio -> 1/GoldenRatio,
  DisplayFunction -> $DisplayFunction];
```

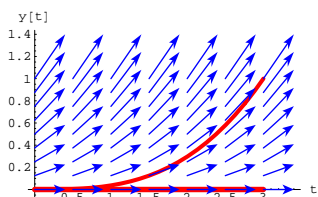


You can see that the alternate silly solution is also going with the horizontal flow along the x -axis.

□T.b.ii) The danger zone strikes again

Take another look at the plot of the two solutions in part i) above:

```
Show[bummer];
```



When you throw your cork into this flow at $\{0, 0\}$, the cork cannot float away in two directions simultaneously. Doesn't this blow the hell out of thinking of solution plots as paths of floating corks?

□Answer:

Yes it does. But before you give up on the idea of thinking of solution plots as paths of floating corks, check the danger zone for this differential equation:

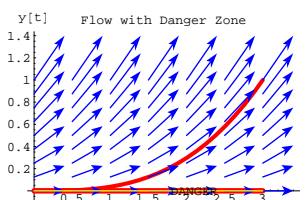
```
Clear[diffeq, y, t, f, r, starter];
f[t_, y_] = y2/3;

(diffeq = {y'[t] == f[t, y[t]], y[0] == 0}) // ColumnForm
y'[t] == y[t]2/3
y[0] == 0
gradf[t_, y_] = {∂tf[t, y], ∂yf[t, y]}
{0, 2/3 y-1/3}
```

There is a big fat singularity in $\text{gradf}[x, y]$ at $y = 0$,

So the danger zone is the line $y[t] = 0$:

```
ydangerous = 0;
dangerzoneplot = Graphics[{Thickness[0.005],
  Yellow, Line[{0, ydangerous}, {endtime, ydangerous}]}];
dangerlabel = Graphics[Text["DANGER", {0.7 endtime, ydangerous}]];
Show[bummer, dangerzoneplot, dangerlabel,
  PlotLabel -> "Flow with Danger Zone"];
```



The reason that your flow intuition doesn't quite work for this diffeq is that the cork was thrown into the flow in the danger zone. In fact, the alternate solution stays in the danger zone all the time. The other one hustles its buns away from the danger zone.

□T.3.b.iii)

Does this mean that when you are going with a differential equation with no danger zone, then you are free to use your intuition thinking of solution plots as the paths of floating corks?

□Answer:

Yes it does.

Fancy diffeq folks say that when there is no danger zone, solutions exist and are unique.

DE.05 First Order Differential Equations Give It a Try!

G.1) Quickies

□G.1.a)

Here's an autonomous diffeq:

```
Clear[diffeq, y, t, f, starter];
a = Random[Real, {2, 3}];
```

```
b = Random[Real, {5, 7}];
c = Random[Real, {9, 11}];
f[t_, y_] = 0.07 (a - y) (y - b) (y - c);
```

```
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == 0.07 (2.28941 - y[t]) (-9.92898 + y[t]) (-6.15197 + y[t])
y[0] == starter
```

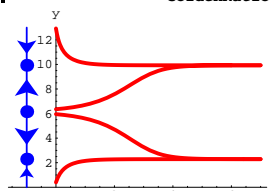
Here is the phase line for this diffeq shown with plots of four solutions of this diffeq.

```
{ylow, yhigh} = {0, 13};
{tlow, thigh} = {0, 7};
Clear[y1, y2, y3, y, t];

{starter1, starter2, starter3, starter4} = {0.4, b - 0.2, b + 0.2, c + 3};
endtime = thigh;
y1[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter1, y[t], {t, 0, endtime}][[1]];
y2[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter2, y[t], {t, 0, endtime}][[1]];
y3[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter3, y[t], {t, 0, endtime}][[1]];
y4[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter4, y[t], {t, 0, endtime}][[1]];

solutionplots = Plot[{y1[t], y2[t], y3[t], y4[t]}, {t, 0, endtime},
  PlotStyle -> {{Thickness[0.015], Red}}, DisplayFunction -> Identity];
phaseline = PhaseLine[f[t, y], {y, ylow, yhigh}, Blue, -1];

setup = Show[solutionplots, phaseline,
  Axes -> True, AxesLabel -> {"t", "y"}, PlotRange -> {ylow, yhigh},
  AspectRatio -> 1/GoldenRatio, DisplayFunction -> $DisplayFunction];
```



That's the phase line for this diffeq on the left.

What's the relationship between the arrowheads on the phase line and

the behavior of the solutions?

Which dot on the phase line signals the place where you expect extreme sensitivity to errors in starter data on $y[0]$?

□G.1.b)

Here's an autonomous diffeq:

```
Clear[diffeq, y, t, f, starter];
f[t_, y_] = 0.9 (1 -  $\frac{y}{2}$ ) (1 -  $\frac{y}{4}$ ) (1 -  $\frac{y}{8}$ );

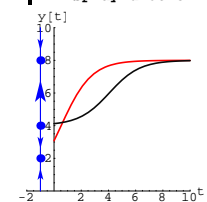
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == 0.9 (1 -  $\frac{y[t]}{2}$ ) (1 -  $\frac{y[t]}{4}$ ) (1 -  $\frac{y[t]}{8}$ )
y[0] == starter
```

Here is the phaseline for this diffeq shown with the plot of two curves:

```
{tlow, thigh} = {0, 10};
{ylo, yhi} = {0, 10};

phaseline = PhaseLine[f[t, y], {y, ylo, yhi}, Blue, -1];
curveplots = Plot[ $\left\{\frac{\text{Exp}[0.9 t]}{0.2 + 0.125 \text{Exp}[0.9 t]}, \frac{157 + 8 \text{Exp}[0.9 t]}{39 + \text{Exp}[0.9 t]}\right\}$ ,
  {t, tlow, thigh}, PlotStyle ->
  {{Thickness[0.01], Red}, {Thickness[0.01], GrayLevel[0.04]}},
  DisplayFunction -> Identity];

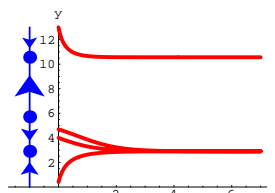
Show[curveplots, phaseline, AxesLabel -> {"t", "y[t]"},
  AspectRatio -> 1, PlotRange -> {{tlow - 2, thigh}, {ylo, yhi}},
  DisplayFunction -> $DisplayFunction];
```



One of these curves looks like it might be a reasonable approximation of a solution of the diffeq.

The other does not.

Which is which?



That's the phase line for this diffeq on the left. Unfortunately, the solution plots do not mirror all the information given by the phase line. Fill in the question marks in the code below and then run so that the resulting plot does a pretty good job of mirroring all the information given by the phase line. Throw in more than four solutions if you like.

```
xx
Clear[y1, y2, y3, y, t];
{starter1, starter2, starter3, starter4} = {?, ?, ?, ?};
endtime = thigh;

y1[t_] = y[t] /. NDSolve[
  (diffeq /. starter -> starter1), y[t], {t, 0, endtime}][[1]];
y2[t_] = y[t] /. NDSolve[
  (diffeq /. starter -> starter2), y[t], {t, 0, endtime}][[1]];
y3[t_] = y[t] /. NDSolve[
  (diffeq /. starter -> starter3), y[t], {t, 0, endtime}][[1]];
y4[t_] = y[t] /. NDSolve[
  (diffeq /. starter -> starter4), y[t], {t, 0, endtime}][[1]];

solutionplots =
  Plot[{y1[t], y2[t], y3[t], y4[t]}, {t, 0, endtime}, PlotStyle ->
  {{Thickness[0.015], Red}}, DisplayFunction -> Identity];

phaseline = PhaseLine[f[t, y], {y, ylo, yhi}, Blue, -1];

setup = Show[solutionplots,
  phaseline, Axes -> True, AxesLabel -> {"t", "y"},
  PlotRange -> {ylo, yhi}, AspectRatio ->  $\frac{1}{\text{GoldenRatio}}$ ,
  DisplayFunction -> $DisplayFunction];
```

□G.1.c)

Here's another autonomous diffeq:

```
Clear[diffeq, y, t, f, starter];
a = Random[Real, {2, 3}];
b = Random[Real, {5, 7}];
c = Random[Real, {9, 11}];
f[t_, y_] = 0.07 (a - y) (y - b) (y - c);

(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == 0.07 (2.90777 - y[t]) (-10.5619 + y[t]) (-5.71538 + y[t])
y[0] == starter
```

Here is the phase line for this diffeq shown with plots of four solutions of this diffeq.

```
{ylo, yhi} = {0, 13};
{tlow, thigh} = {0, 7};
Clear[y1, y2, y3, y, t];
{starter1, starter2, starter3, starter4} = {0.4, 4.0, 4.7, 13.0};
endtime = thigh;
y1[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter1, y[t], {t, 0, endtime}][[1]];
y2[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter2, y[t], {t, 0, endtime}][[1]];
y3[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter3, y[t], {t, 0, endtime}][[1]];
y4[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter4, y[t], {t, 0, endtime}][[1]];

solutionplots = Plot[{y1[t], y2[t], y3[t], y4[t]}, {t, 0, endtime},
  PlotStyle -> {{Thickness[0.015], Red}}, DisplayFunction -> Identity];
phaseline = PhaseLine[f[t, y], {y, ylo, yhi}, Blue, -1];

setup = Show[solutionplots, phaseline,
  Axes -> True, AxesLabel -> {"t", "y"}, PlotRange -> {ylo, yhi},
  AspectRatio ->  $\frac{1}{\text{GoldenRatio}}$ , DisplayFunction -> $DisplayFunction];
```

□G.1.d.i)

Here's a new autonomous diffeq containing some partially random coefficients:

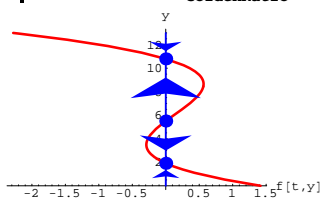
```
Clear[diffeq, y, t, f, starter];
a = Random[Real, {1, 2}];
b = Random[Real, {1, 3}];
c = Random[Real, {5, 7}];
d = Random[Real, {9, 11}];
f[t_, y_] = a (1 -  $\frac{y}{b}$ ) (1 -  $\frac{y}{c}$ ) (1 -  $\frac{y}{d}$ );

(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == 1.41603 (1 - 0.522989 y) (1 - 0.18149 y) (1 - 0.0926319 y)
y[0] == starter
```

Look at this plot:

```
{ylo, yhi} = {0, 13};
fcurve = ParametricPlot[{f[t, y], y}, {y, ylo, yhi},
  PlotStyle -> {{Thickness[0.01], Red}}, DisplayFunction -> Identity];
phaseline = PhaseLine[f[t, y], {y, ylo, yhi}, Blue, 0];

setup = Show[fcurve, phaseline, AxesLabel -> {"f[t,y]", "y"},
  AspectRatio ->  $\frac{1}{\text{GoldenRatio}}$ , DisplayFunction -> $DisplayFunction];
```



Rerun both cells a couple of times.

Points on the curve are of the form:

```
{f[t, y], y}
{1.41603 (1 - 0.522989 y) (1 - 0.18149 y) (1 - 0.0926319 y), y}
```

But the real issues here are:

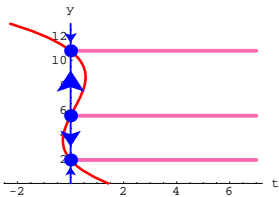
→ How are the arrowheads on the phase line related to the plot of the curve?

→ How are the dots on the phase line related to the plot of the curve?
 → Remembering that $y' = f[t, y]$, explain why it had to turn out this way.

□G.1.d.ii)

Keep everything the same as in part i) above, but throw in plots of the solutions that start on the dots on the phase line:

```
Clear[y1, y2, y3, y, t];
{starter1, starter2, starter3} = {b, c, d};
endtime = thigh;
y1[t_] = y[t] /.
NDSolve[diffeq /. starter → starter1, y[t], {t, 0, endtime}][[1]];
y2[t_] = y[t] /.
NDSolve[diffeq /. starter → starter2, y[t], {t, 0, endtime}][[1]];
y3[t_] = y[t] /.
NDSolve[diffeq /. starter → starter3, y[t], {t, 0, endtime}][[1]];
solutionplots =
Plot[{y1[t], y2[t], y3[t]}, {t, 0, endtime}, PlotStyle →
{{Thickness[0.015], HotPink}}, DisplayFunction → Identity];
Show[solutionplots, setup, Axes → True, AxesLabel → {"t", "y"},
PlotRange → {ylow, yhigh}, AspectRatio →  $\frac{1}{\text{GoldenRatio}}$ ,
DisplayFunction → $DisplayFunction];
```



Got any idea why that happened?
 If so, then give your idea.

□G.1.e)

Here are four diffeqs:

```
Clear[diffeq1, y, t, f1, starter, r];
f1[t_, y_] = 3.8 (y - 1) (y - 2) - t;
(diffeq1 = {y'[t] == f1[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == -t + 3.8 (-2 + y[t]) (-1 + y[t])
y[0] == starter
Clear[diffeq2, y, t, f2, starter, r];
f2[t_, y_] = 3.8 E-0.9 y;
(diffeq2 = {y'[t] == f2[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == 3.8 E-0.9 y[t]
y[0] == starter
Clear[diffeq3, y, t, f3, starter, r];
f3[t_, y_] = 3.8 (y - 1) (y - 2) - r;
(diffeq3 = {y'[t] == f3[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == -r + 3.8 (-2 + y[t]) (-1 + y[t])
y[0] == starter
Clear[diffeq4, y, t, f4, starter, r];
f4[t_, y_] = 3.8 Sin[y] - t + r;
(diffeq4 = {y'[t] == f4[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == r - t + 3.8 Sin[y[t]]
y[0] == starter
```

Which of these diffeqs are autonomous?
 Which of the autonomous diffeqs contains an extra parameter?
 Which of the non-autonomous diffeqs contains an extra parameter?

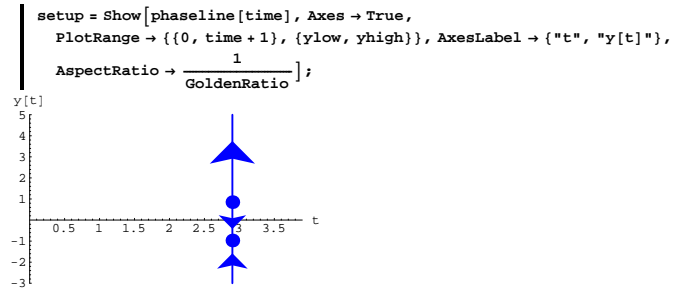
□G.1.f.i)

Here's a new diffeq:

```
Clear[diffeq1, y, t, f, starter, r];
f[t_, y_] = Sin[t] (y - Cos[t]) (y - 0.85);
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == Sin[t] (-0.85 + y[t]) (-Cos[t] + y[t])
y[0] == starter
```

Look at the plot of the phase line that corresponds to $t = 2.9$:

```
time = 2.9;
{ylow, yhigh} = {-3, 5};
Clear[phaseline];
phaseline[t_] := PhaseLine[f[t, y], {y, ylow, yhigh}, Blue, t];
```



Imagine that plots of solutions $y[t]$ of this diffeq come in from the left and pass through the plotted phase line.

Some of them are going up as they cross over this line. Others are going down.

Read the plot to get rough estimates of the cutoff points between those that go up and those that go down as they pass through the plotted phase line.

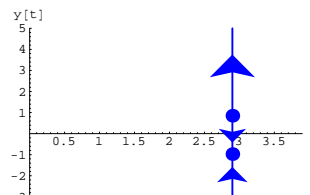
□G.1.f.ii)

Keep everything the same as in part i) above:

```
Clear[diffeq, y, t, f, starter, r];
f[t_, y_] = Sin[t] (y - Cos[t]) (y - 0.85);
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == Sin[t] (-0.85 + y[t]) (-Cos[t] + y[t])
y[0] == starter
```

Look again at this plot of the phase line that corresponds to $t = 2.9$:

```
time = 2.9;
{ylow, yhigh} = {-3, 5};
Clear[phaseline];
phaseline[t_] := PhaseLine[f[t, y], {y, ylow, yhigh}, Blue, t];
setup = Show[phaseline[time], Axes → True,
PlotRange → {{0, time + 1}, {ylow, yhigh}}, AxesLabel → {"t", "y[t]"},
AspectRatio →  $\frac{1}{\text{GoldenRatio}}$ ];
```



Now look at this:

```
f[time, y]
0.239249 (-0.85 + y) (0.970958 + y)
```

Use what you see to improve your estimates from part i) immediately above.

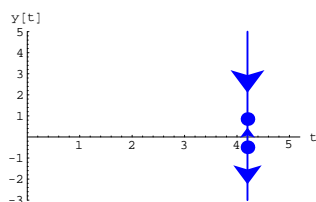
□G.1.f.iii)

Stay with the same diffeq:

```
Clear[diffeq, y, t, f, starter, r];
f[t_, y_] = Sin[t] (y - Cos[t]) (y - 0.85);
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == Sin[t] (-0.85 + y[t]) (-Cos[t] + y[t])
y[0] == starter
```

Look at this plot of the phase line that corresponds to $t = 4.2$:

```
time = 4.2;
{ylow, yhigh} = {-3, 5};
Clear[phaseline];
phaseline[t_] := PhaseLine[f[t, y], {y, ylow, yhigh}, Blue, t];
setup = Show[phaseline[time], Axes → True,
PlotRange → {{0, time + 1}, {ylow, yhigh}}, AxesLabel → {"t", "y[t]"},
AspectRatio →  $\frac{1}{\text{GoldenRatio}}$ ];
```



Imagine that plots of solutions $y[t]$ of this diffeq come in from the left and pass through the plotted phase line.

Some of them are going up as they cross over this line. Others are going down.

Read the plot to get rough estimates of the cutoff points between those that go up and those that go down as they pass through the plotted phase line.

□G.f.iv)

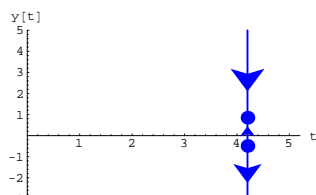
Keep everything the same as in part iii) above:

```
Clear[diffeq1, y, t, f, starter, r];
f[t_, y_] = Sin[t] (y - Cos[t]) (y - 0.85);
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == Sin[t] (-0.85 + y[t]) (-Cos[t] + y[t])
y[0] == starter
```

Look again at this plot of the phase line that corresponds to $t = 4.2$:

```
time = 4.2;
{y_low, y_high} = {-3, 5};
Clear[phaseline];
phaseline[t_] := PhaseLine[f[t, y], {y, y_low, y_high}, Blue, t];

setup = Show[phaseline[time],
  Axes -> True, PlotRange -> {{0, time + 1}, {y_low, y_high}},
  AxesLabel -> {"t", "y[t]"}, AspectRatio ->  $\frac{1}{\text{GoldenRatio}}$ ];
```



Now look at this:

```
f[time, y]
-0.871576 (-0.85 + y) (0.490261 + y)
```

Use what you see to improve your estimates from part iii) immediately above.

□G.1.g.i)

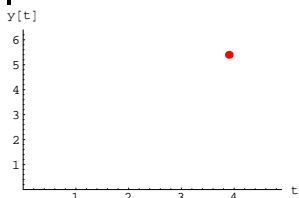
Here's a new diffeq:

```
Clear[diffeq1, y, t, f, starter, r];
f[t_, y_] = 3.8 E^{0.01 t} (y - t);
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == 3.8 E^{0.01 t} (-t + y[t])
y[0] == starter
```

Look at this plot:

```
time = 3.9;
ytime = 5.4;
pointplot = Graphics[{Red, PointSize[0.03], Point[{time, ytime}]}];

setup = Show[pointplot, Axes -> True, PlotRange -> {{0, time + 1}, {0, ytime + 1}},
  AxesLabel -> {"t", "y[t]"}, AspectRatio ->  $\frac{1}{\text{GoldenRatio}}$ ];
```



The plotted point has coordinates

{time, ytime}

as defined in the code above.

Imagine that the plot of a solution $y[t]$ of this diffeq comes in from the left and passes through the plotted point.

Look at this quick calculation of $f[\text{time}, \text{ytime}]$:

```
f[time, ytime]
5.92669
```

Remembering that

$$y'[t] = f[t, y[t]],$$

use this quick calculation to determine whether:

a) The plot of $y[t]$ comes in from the left and down to this point.

Or:

b) The plot of $y[t]$ comes in from the left and up to this point.

□G.1.g.ii)

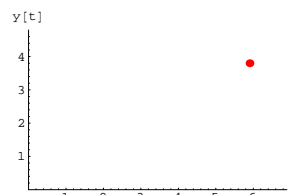
Stay with the same diffeq:

```
Clear[diffeq1, y, t, f, starter, r];
f[t_, y_] = 3.8 E^{0.01 t} (y - t);
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == 3.8 E^{0.01 t} (-t + y[t])
y[0] == starter
```

Look at this new plot:

```
time = 5.9;
ytime = 3.8;
pointplot = Graphics[{Red, PointSize[0.03], Point[{time, ytime}]}];

setup = Show[pointplot, Axes -> True, PlotRange -> {{0, time + 1}, {0, ytime + 1}},
  AxesLabel -> {"t", "y[t]"}, AspectRatio ->  $\frac{1}{\text{GoldenRatio}}$ ];
```



The plotted point has coordinates

{time, ytime}

as defined in the code above.

Imagine that the plot of a solution $y[t]$ of this diffeq comes in from the left and passes through the plotted point.

Look at this quick calculation:

```
f[time, ytime]
-8.46499
```

Remembering that

$$y'[t] = f[t, y[t]],$$

use this quick calculation to determine whether:

a) The plot of $y[t]$ comes in from the left and down to this point.

Or:

b) The plot of $y[t]$ comes in from the left and up to this point.

G.2) Reading diffeq's using flow plots and phase lines

□G.2.a.i)

Here's a differential equation.

```
Clear[diffeq, f, t, y];
f[t_, y_] = 0.3 y (y - 1) (y - 3);
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == 0.3 (-3 + y[t]) (-1 + y[t]) y[t]
y[0] == starter
```

Notice right away that the right hand side has no explicit dependence on t .

Here's the formula for the function $f[t, y]$ that was used to define the diffeq:

```
f[t, y]
0.3 (-3 + y) (-1 + y) y
```

Use what you see to determine whether or not this diffeq is autonomous.

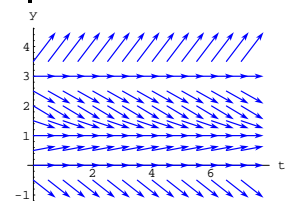
□G.2.a.ii)

Stay with the same diffeq as in part i):

```
Clear[diffeq, f, t, y];
f[t_, y_] = 0.3 y (y - 1) (y - 3);
(diffeq = {y'[t] == f[y[t]], y[0] == starter}) // ColumnForm
y'[t] == f[y[t]]
y[0] == starter
```

And look at the corresponding flow plot:

```
{tlow, thigh} = {0, 7};
{ylo, yhi} = {-0.5, 3.5};
jump = 0.5;
flowplot = Show[Table[Arrow[{1, f[t, y]}, Tail -> {t, y},
VectorColor -> Blue, ScaleFactor -> 0.75, HeadSize -> 0.3],
{t, tlow, thigh, jump}, {y, ylo, yhi, jump}], Axes -> True,
AxesLabel -> {"t", "y"}];
```



How does the flow plot help to confirm that $f[t, y]$ has no dependence on t ?

□G.2.a.iii)

Stay with the same diffeq as in part i):

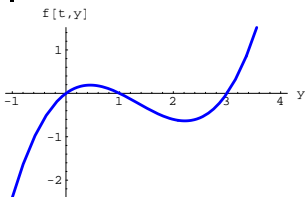
```
Clear[diffeq, f, t, y];
f[t_, y_] = 0.3 y (y - 1) (y - 3);
(diffeq = {y'[t] == f[y[t]], y[0] == starter}) // ColumnForm
y'[t] == 0.3 (-3 + y[t]) (-1 + y[t]) y[t]
y[0] == starter
```

And look at the formula for the function $f[t, y]$ that was used to define the diffeq:

```
f[t, y]
0.3 (-3 + y) (-1 + y) y
```

And a plot of $f[t, y]$

```
Plot[f[t, y], {y, -1, 4}, PlotStyle -> {{Thickness[0.01], Blue}},
AxesLabel -> {"y", "f[t,y]"}, AspectRatio ->  $\frac{1}{\text{GoldenRatio}}$ ];
```



Use the what you see to predict what solution curves are doing for starting values $y[0]$ with:

- $y[0] < 0$
- $y[0] = 0$
- $0 < y[0] < 1$
- $y[0] = 1$
- $1 < y[0] < 3$
- $y[0] = 3$
- $3 < y[0]$

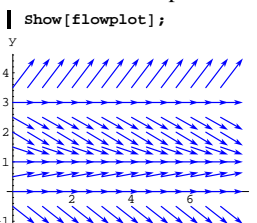
□G.2.a.iv)

Stay with the same diffeq as in part i):

```
Clear[diffeq, f, t, y];
f[t_, y_] = 0.3 y (y - 1) (y - 3);
(diffeq = {y'[t] == f[y[t]], y[0] == starter}) // ColumnForm
```

```
y'[t] == 0.3 (-3 + y[t]) (-1 + y[t]) y[t]
y[0] == starter
```

And look at a flow plot:



Say how the flow plot helps to confirm your answers to the part iii) immediately above.

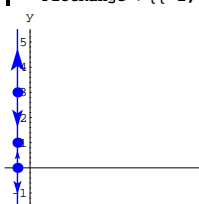
□G.2.a.v)

Stay with the same diffeq as in part i):

```
Clear[diffeq, f, t, y];
f[t_, y_] = 0.3 y (y - 1) (y - 3);
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == 0.3 (-3 + y[t]) (-1 + y[t]) y[t]
y[0] == starter
```

Here's the phase line for this diffeq.

```
Clear[pcline];
phaseline =
Show[PhaseLine[f[t, y], {y, ylo - 1, yhi + 2}, Blue, -0.5],
Axes -> True, AxesLabel -> {"", "y"}, Ticks -> {None, Automatic},
PlotRange -> {{-1, thigh}, {ylo - 1, yhi + 2}}];
```



Say how the phase line helps to confirm your answers to the part iii) above.

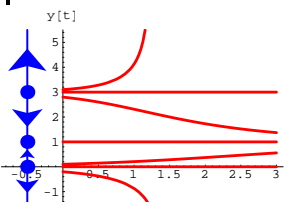
□G.2.a.vi)

Stay with the same diffeq as in part i):

```
Clear[diffeq, f, t, y];
f[t_, y_] = 0.3 y (y - 1) (y - 3);
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == 0.3 (-3 + y[t]) (-1 + y[t]) y[t]
y[0] == starter
```

Here are some samples of solution curves shown with the phase line:

```
endtime = 3;
Clear[sol, starter];
sol[t_, starter_] := y[t] /. NDSolve[
{y'[t] == f[t, y[t]], y[0] == starter}, y[t], {t, tlow, thigh}][[1]]
{starter1,
starter2, starter3, starter4, starter5, starter6, starter7} =
{-0.2, 0.0, 0.1, 1.0, 2.8, 3.0, 3.1};
solutioncurves = Plot[Evaluate[{sol[t, starter1],
sol[t, starter2], sol[t, starter3], sol[t, starter4],
sol[t, starter5], sol[t, starter6], sol[t, starter7]}],
{t, 0, endtime}, PlotStyle -> {{Thickness[0.01], Red}},
PlotRange -> {ylo - 1, yhi + 2},
AxesLabel -> {"t", "y[t]"}, DisplayFunction -> Identity];
Show[solutioncurves, phaseline, AspectRatio ->  $\frac{1}{\text{GoldenRatio}}$ ,
DisplayFunction -> $DisplayFunction];
```



Does this plot agree or disagree with what you decided above?
Write a summary of what the advantages and disadvantages of each of the ways of looking at the behavior of solutions are.

□G.2.b.i)

Here's a new differential equation.

```
Clear[diffeq, f, t, y];
f[t_, y_] = 1.3 y (y - 1.5) (y - 3.0) + 2.3 Sin[t];
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == 2.3 Sin[t] + 1.3 (-3. + y[t]) (-1.5 + y[t]) y[t]
y[0] == starter
```

Is this diffeq autonomous?

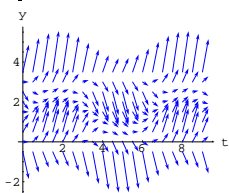
□G.2.b.ii)

Stay with the same diffeq:

```
Clear[diffeq, f, t, y];
f[t_, y_] = 1.3 y (y - 1.5) (y - 3.0) + 2.3 Sin[t];
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == 2.3 Sin[t] + 1.3 (-3. + y[t]) (-1.5 + y[t]) y[t]
y[0] == starter
```

And look at a flow plot for this diffeq:

```
{tlow, thigh} = {0, 9};
{ylo, yhi} = {-0.5, 3.5};
jump = 0.5;
flowplot = Show[Table[Arrow[{1, f[t, y]}, Tail -> {t, y},
VectorColor -> Blue, ScaleFactor -> 0.3, HeadSize -> 0.3],
{t, tlow, thigh, jump}, {y, ylo, yhi, jump}], Axes -> True,
AxesLabel -> {"t", "y"}];
```



How does the flow plot show that $f[t, y]$ depends on t ?
How does this flow plot help to confirm your response to part i) immediately above?

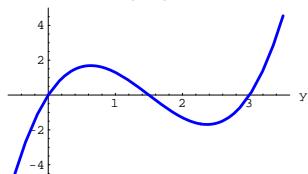
□G.2.b.iii)

Stay with the same diffeq:

```
Clear[diffeq, f, t, y];
f[t_, y_] = 1.3 y (y - 1.5) (y - 3.0) + 2.3 Sin[t];
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == 2.3 Sin[t] + 1.3 (-3. + y[t]) (-1.5 + y[t]) y[t]
y[0] == starter
```

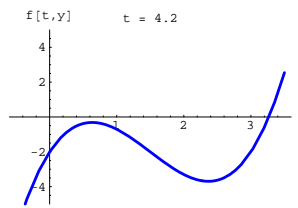
Look at this plot of $f[0, y]$:

```
t0plot =
Plot[f[0, y], {y, ylow, yhigh}, PlotStyle -> {{Thickness[0.01], Blue}},
AxesLabel -> {"y", "f[t,y]"}, AspectRatio ->  $\frac{1}{\text{GoldenRatio}}$ ,
PlotRange -> {-5, 5}, PlotLabel -> "t = 0";
```



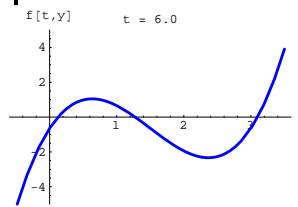
And look at this plot of $f[3.0, y]$:

```
t4plot = Plot[f[4.2, y],
{y, ylow, yhigh}, PlotStyle -> {{Thickness[0.01], Blue}},
AxesLabel -> {"y", "f[t,y]"}, AspectRatio ->  $\frac{1}{\text{GoldenRatio}}$ ,
PlotRange -> {-5, 5}, PlotLabel -> "t = 4.2";
```



And look at this plot of $f[6.0, y]$:

```
t6plot = Plot[f[6.0, y],
{y, ylow, yhigh}, PlotStyle -> {{Thickness[0.01], Blue}},
AxesLabel -> {"y", "f[t,y]"}, AspectRatio ->  $\frac{1}{\text{GoldenRatio}}$ ,
PlotRange -> {-5, 5}, PlotLabel -> "t = 6.0";
```



What does the first plot tell you about the initial behavior of solutions as t advances away from 0?

What does the second plot tell you about how solutions behave as t passes through 4.2?

What does the third plot tell you about how solutions behave as t passes through 6.0?

G.3) Reacting to plots

□G.3.a.i)

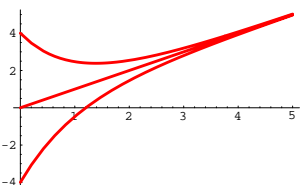
Here's a simple linear exponential differential equation.

```
Clear[diffeq, y, t, f, starter];
f[t_, y_] = 1 + t - y;
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
```

```
y'[t] == 1 + t - y[t]
y[0] == starter
```

Here's a plot showing three solutions starting at $y[0] = 4$, $y[0] = 0$, and $y[0] = -4$:

```
Clear[y1, y2, y3, t];
endtime = 5;
{starter1, starter2, starter3} = {-4, 0, 4};
y1[t_] = y[t] /.
NDSolve[diffeq /. starter -> starter1, y[t], {t, 0, endtime}][[1]];
y2[t_] = y[t] /.
NDSolve[diffeq /. starter -> starter2, y[t], {t, 0, endtime}][[1]];
y3[t_] = y[t] /.
NDSolve[diffeq /. starter -> starter3, y[t], {t, 0, endtime}][[1]];
solutionplot = Plot[{y1[t], y2[t], y3[t]}, {t, 0, endtime},
PlotStyle -> {{Thickness[0.01], Red}}, AspectRatio ->  $\frac{1}{\text{GoldenRatio}}$ ,
PlotRange -> All, AxesLabel -> {"t", ""}];
```



That middle line is just $y[t] = t$.

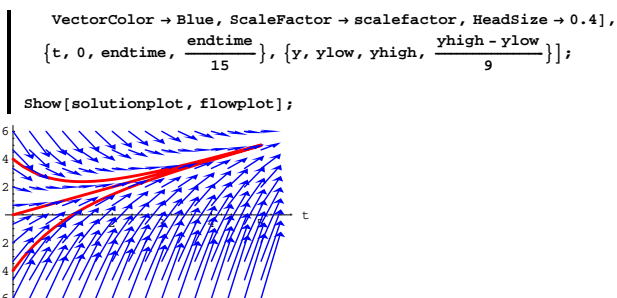
When you go with any starting value $y[0]$ between -4 and 4 , how do you expect the corresponding solution to plot out? Test your ideas by playing with the plot above.

Describe in some detail what you expect to happen to it as t gets larger and larger.

□G.3.a.ii)

Stay with the same differential equation and look at the same plot shown with the flow plot for the differential equation:

```
scalefactor = 0.4;
{ylo, yhi} = {-6, 6};
flowplot = Table[Arrow[{1, f[t, y]}, Tail -> {t, y},
```



Say how this flow plot confirms your answer to part i)

□G.3.a.iii)

Stay with the same differential equation:

```

Clear[diffeq, y, t, starter];
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == 1 + t - y[t]
y[0] == starter

```

Ask *Mathematica* for a formula for solutions:

```

DSolve[diffeq, y[t], t]
{{y[t] -> E^-t (starter + E^t t)}}

```

Analyze this formula to come up with a confirmation of what you said in part i).

□G.3.b.i)

Here's another forced exponential differential equation.

```

Clear[diffeq, y, t, f, starter];
f[t_, y_] = -t^2 + y;
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == -t^2 + y[t]
y[0] == starter

```

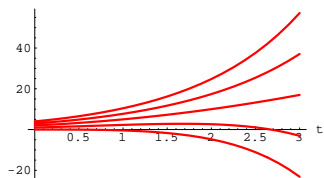
Here's a plot showing five solutions starting at $y[0] = 0$, $y[0] = 1$, $y[0] = 2$, $y[0] = 3$ and $y[0] = 4$:

```

Clear[starter];
Clear[y1, y2, y3, y4, y5, t];
endtime = 3;
{starter1, starter2, starter3, starter4, starter5} = {0, 1, 2, 3, 4};
y1[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter1, y[t], {t, 0, endtime}][[1];
y2[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter2, y[t], {t, 0, endtime}][[1];
y3[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter3, y[t], {t, 0, endtime}][[1];
y4[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter4, y[t], {t, 0, endtime}][[1];
y5[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter5, y[t], {t, 0, endtime}][[1];

solutionplot = Plot[{y1[t], y2[t], y3[t], y4[t], y5[t]},
  {t, 0, endtime}, PlotStyle -> {{Thickness[0.008], Red}},
  AspectRatio ->  $\frac{1}{\text{GoldenRatio}}$ , PlotRange -> All, AxesLabel -> {"t", ""}];

```



What you are seeing here is what some fancy dudes call a "bifurcation."

The solutions fall into two families: Those that eventually go up and those that eventually go down.

Experiment with your own plots to try to come up with an estimate of initial value (starter) on $y[0]$ that gives the break between the two families. Show off your answer with a convincing plot.

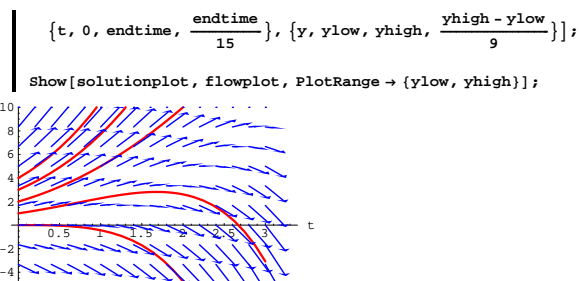
□G.3.b.ii)

Stay with the same differential equation and look at the plot in the above part shown with the flow plot for the differential equation:

```

scalefactor = 0.25;
{ylow, yhigh} = {-5, 10};
flowplot = Table[Arrow[{1, f[t, y]}, Tail -> {t, y},
  VectorColor -> Blue, ScaleFactor -> scalefactor, HeadSize -> 0.25],

```



Does this plot help you to confirm your answer in part i)?

If so, why?

□G.3.b.iii)

Stay with the same differential equation:

```

Clear[diffeq, y, t, starter];
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == -t^2 + y[t]
y[0] == starter

```

Ask *Mathematica* for a formula for solutions:

```

DSolve[diffeq, y[t], t]
{{y[t] -> 2 + E^-t (-2 + starter) + 2 t + t^2}}

```

Use this formula to come up with the exact initial value (starter) on $y[0]$ that gives the break between the two families

□G.3.b.iv)

Stay with the same differential equation:

```

Clear[diffeq, y, t, f, starter];
f[t_, y_] = -t^2 + y;
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == -t^2 + y[t]
y[0] == starter

```

Run the following cells:

```

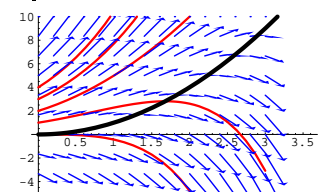
Solve[f[t, y] == 0, y]
{{y -> t^2}}

```

```

auxillaryplot = Plot[t^2, {t, 0, 1.2 endtime},
  PlotStyle -> Thickness[0.015], DisplayFunction -> Identity];
Show[solutionplot, flowplot, auxillaryplot,
  PlotRange -> {ylow, yhigh}];

```



Describe what you see and explain why you see it.

□Tip:

Don't take the bait too fast. The "plotted" function t^2 is not a solution of the differential equation:

```

diffeq = y'[t] == f[t, y[t]]
y'[t] == -t^2 + y[t]
diffeq /. {y[t] -> t^2, y'[t] -> 2 t}
2 t == 0

```

□G.3.c.i)

Here's another innocent-looking differential equation.

```

Clear[diffeq, y, t, f, starter];
f[t_, y_] = y^2 - 0.5 t - 1;
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == -1 - 0.5 t + y[t]^2
y[0] == starter

```

This is a nasty one because clean formulas for solutions are simply not available:

```

thisa = 0.5; Clear[a];
DSolve[{y'[t] == -1 - a t + y[t]^2, y[0] == starter}, y[t], t]
{{y[t] -> -(a^(1/3) (AiryAiPrime[ $\frac{1 + a t}{a^{2/3}}$ ] + ((-starter) AiryAi[ $\frac{1}{a^{2/3}}$ ] -
  a^(1/3) AiryAiPrime[ $\frac{1}{a^{2/3}}$ ]) AiryBiPrime[ $\frac{1 + a t}{a^{2/3}}$ ])) /

```

$$\frac{\left(\text{starter AiryBi}\left[\frac{1}{a^{2/3}}\right] + a^{1/3} \text{AiryBiPrime}\left[\frac{1}{a^{2/3}}\right]\right)}{\left(\text{AiryAi}\left[\frac{1+a t}{a^{2/3}}\right] + \left(-\text{starter AiryAi}\left[\frac{1}{a^{2/3}}\right] - a^{1/3} \text{AiryAiPrime}\left[\frac{1}{a^{2/3}}\right]\right) \text{AiryBi}\left[\frac{1+a t}{a^{2/3}}\right]\right)} \frac{\left(\text{starter AiryBi}\left[\frac{1}{a^{2/3}}\right] + a^{1/3} \text{AiryBiPrime}\left[\frac{1}{a^{2/3}}\right]\right)}{\left(\text{AiryAi}\left[\frac{1+a t}{a^{2/3}}\right] + \left(-\text{starter AiryAi}\left[\frac{1}{a^{2/3}}\right] - a^{1/3} \text{AiryAiPrime}\left[\frac{1}{a^{2/3}}\right]\right) \text{AiryBi}\left[\frac{1+a t}{a^{2/3}}\right]\right)}$$

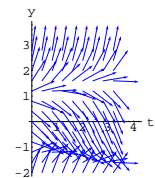
This formula is not illuminating.

Giving up on formulas, you go to flow plots and numerical solutions

(NDSolve):

```
{tlow, thigh} = {0, 3.5};
{ylo, yhi} = {-2, 3};
jump = 0.4;

flowplot = Show[Table[Arrow[{1, f[t, y]},
  Tail -> {t, y}, VectorColor -> Blue, ScaleFactor -> Normalize],
  {t, tlow, thigh, jump}, {y, ylo, yhi, jump}], Axes -> True,
  AxesLabel -> {"t", "y"}];
```

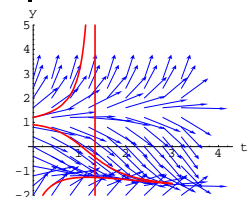


Here are a couple of solutions:

```
Clear[starter];
Clear[y1, y2, y3, t];
endtime = 3;
{starter1, starter2, starter3} = {-3, 0.9, 1.2};
y1[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter1, y[t], {t, 0, endtime}][[1]];
y2[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter2, y[t], {t, 0, endtime}][[1]];
y3[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter3, y[t], {t, 0, endtime}][[1]];

solutionplot = Plot[{y1[t], y2[t], y3[t]}, {t, 0, endtime},
  PlotStyle -> {{Thickness[0.008], Red}}, DisplayFunction -> Identity];
```

```
Show[flowplot, solutionplot, AspectRatio -> 1/1.2,
  PlotRange -> {ylo, yhi + 2}];
```



What you see are samples of each of three types of solutions passing through the plotted part of the ty-plane.

Experiment with your own plots to try to come up with an estimate of initial values (starters) on y[0] that correspond to breaks between the families. Show off your answer with a convincing plot.

□G.3.c.ii)

Look at the differential equation again:

```
Clear[diffeq, y, t, f, starter];
f[t_, y_] = y^2 - 0.5 t - 1;

(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == -1 - 0.5 t + y[t]^2
y[0] == starter
```

Explain the statements:

- When you go with a starting value on y[0] between -1 and 1, then the corresponding solution will initially go down as t advances from 0.
- When you go with a starting value on y[0] with either y[0] < -1 or y[0] > 1, then the corresponding solution will initially go up as t advances from 0.

G.4) Using bifurcation plots to study E. Coli growing in a chemostat

This problem was adapted from an article by Professor S.F. Ellermeyer of Kennesaw State College in the Winter 1995 Newsletter of the Consortium for Ordinary Differential Equations Experiments

□G.4.a.i)

A chemostat is a well-stirred vessel that contains microorganisms into which fresh medium is pumped at a constant rate F. The contents are pumped out at the same rate so that the volume V of the fluid in the chemostat remains constant.

Chemostat experiments performed by Hansen and Hubbell (Science, 20 (1980), pp 1491-1493) in which the amino acid tryptophan was pumped at a constant rate into a chemostat containing a certain strain of E. coli bacteria indicated a growth model:

```
Clear[diffeq, y, t, f, s, d, starter];
f[t_, y_] = (0.81 (s - y) - d) y;

(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == (-d + 0.81 (s - y[t]) / (3 - s - y[t])) y[t]
y[0] == starter
```

Here:

→ y[t] measures the population of the E. coli at time t after the pumping of the amino acid started.

→ s is the concentration (in millionths of grams per liter) in of the amino acid in the fluid pumped into the chemostat.

→ $d = \frac{F}{V}$, so larger values of d correspond to a fast running pump, and low values of d correspond to a slow running pump.

In the experiment under study here, $s = 9.8$; so the model becomes:

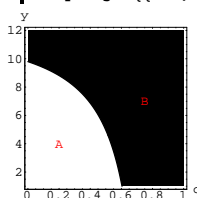
```
s = 9.8;
f[t_, y_] = (0.81 (s - y) - d) y;

(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == (-d + 0.81 (9.8 - y[t]) / (12.8 - y[t])) y[t]
y[0] == starter
```

Take a look at this contour plot of f[t, y] as a function of y and the flushing parameter d:

```
{ylo, yhi} = {1, 12};
{dlo, dhi} = {0, 1};

bifurcationplot = ContourPlot[f[t, y], {d, dlo, dhi},
  {y, ylo, yhi}, ContourSmoothing -> Automatic, PlotPoints -> 50,
  Contours -> {0}, Axes -> True, AxesLabel -> {"d", "y"},
  Epilog -> {{Red, Text["A", {0.2, 4}], {Red, Text["B", {0.75, 7}]}}];
```



The light region indicates the {d, y}'s for which $y' = f[t, y] > 0$ and the dark region indicates the {d, y}'s for which $y' = f[t, y] < 0$.

Explain why this plot signals that when you go with $d = 0.8$, then no matter what the starting population y[0] between 0 and 12 you go with, then the E.coli population is guaranteed to decrease as t increases.

Illustrate with at least one plot corresponding to $d = 0.8$ and a starter value on y[0] between 2 and 10.

□G.4.a.ii)

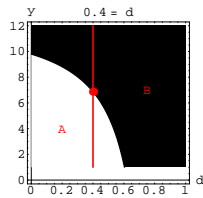
Take a look at this:

```
Clear[d];
dd = 0.4;
ycrossover = FindRoot[{f[t, y] /. d -> dd} == 0, {y, 6}][[1, 2]]
```

6.87317

```
Show[bifurcationplot,
Graphics[{Red, PointSize[0.05], Point[{dd, ycrossover]}]},
Graphics[{Red, Thickness[0.01], Line[{dd, ylow}, {dd, yhigh}]}],
PlotLabel -> dd "=" d"];

```



Use this plot to help you describe the long term behavior of the E. coli population when you go with $d = dd = 0.4$ with a starting value on $y[0]$ between 1 and $y_{\text{crossover}}$.

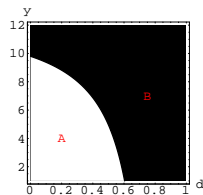
Also use this plot to help you describe the long term behavior of the E. coli population when you go with $d = dd = 0.4$ with a starting value on $y[0]$ between $y_{\text{crossover}}$ and 12.

Illustrate each of your responses with a couple of solution plots.

□G.4.a.iii)

Take another look at the bifurcation plot:

```
Show[bifurcationplot];
```

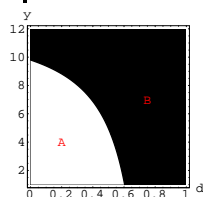


How does the long term E. coli population change as you raise the flushing parameter d from 0 to 1?

□G.4.a.iv) Sensitive dependence on starter values?

Take another look at the bifurcation plot:

```
Show[bifurcationplot];
```



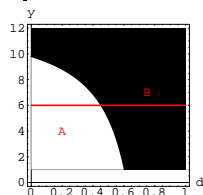
Does this model show any signs of sensitive dependence on starter data on $y[0]$?

□G.4.v) Setting d to control the long term bacteria population

Take another look at this embellished bifurcation plot:

```
ygoal = 6.0;
Show[bifurcationplot, Graphics[
{Red, Thickness[0.01], Line[{dlow, ygoal}, {dhigh, ygoal}]}]];

```



Estimate (by reading the plot) a value of the flushing parameter d that you think will lead to an approximate long-term bacteria population of $y_{\text{goal}} = 6.0$ as long as $1 < y[0] < 12$.

G.5) Reading a diffeq through bifurcation plots:

Parameter bifurcations and sensitive dependence on starter conditions

□G.5.a.i)

Here's an autonomous diffeq containing a parameter r .

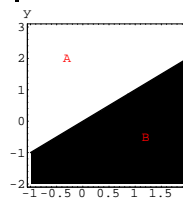
```
Clear[diffeq, y, t, f, r, starter];
f[t_, y_] := y - r;
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == -r + y[t]
y[0] == starter

```

Here comes the bifurcation plot:

```
{ylow, yhigh} = {-2, 3};
{rlow, rhigh} = {-1, 2};
bifurcationplot = ContourPlot[f[t, y], {r, rlow, rhigh},
{y, ylow, yhigh}, ContourSmoothing -> Automatic, PlotPoints -> 50,
Contours -> {0}, Axes -> True, AxesLabel -> {"", "y"}, Epilog ->
{{Red, Text["A", {-0.3, 2}], {Red, Text["B", {1.2, -0.5}]}];

```

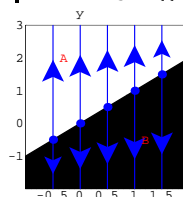


The black region signals $\{r, y\}$'s for which $y' = f[t, y] < 0$.
The white region signals $\{r, y\}$'s for which $y' = f[t, y] > 0$.

Throw in phase lines for various r 's:

```
Clear[phaseline, r, y];
phaseline[r_] := PhaseLine[f[t, y], {y, ylow, yhigh}, Blue, r]
jump = 0.5;
phaselines = Table[phaseline[r], {r, rlow + jump, rhigh - jump, jump}];
newphaseplot = Show[bifurcationplot, phaselines,
PlotRange -> {{rlow, rhigh}, {ylow, yhigh}}];

```



If you go with an r and a starting value on $y[0]$ so that $\{r, y[0]\}$ plots out inside the region labeled A, then what do you expect to happen to the corresponding solution of $y'[t] = f[t, y[t]]$?

If you go with an r and a starting value on $y[0]$ so that $\{r, y[0]\}$ plots out inside the region labeled B, then what do you expect to happen to the corresponding solution of $y'[t] = f[t, y[t]]$?

What choices of $\{r, y[0]\}$ within the plotted region flirt with the dreaded issue of sensitive dependence on starter conditions on $y[0]$?

□G.5.b.i)

Here's another single diffeq containing a parameter r :

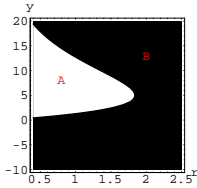
```
Clear[diffeq, y, t, f, s, d, r, starter];
f[t_, y_] := y E^{-y/d} - r;
setup =
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == -r + E^{-y[t]/d} y[t]
y[0] == starter

```

And a bifurcation plot:

```
{ylow, yhigh} = {-10, 20};
{rlow, rhigh} = {0.4, 2.5};

bifurcationplot = ContourPlot[f[t, y], {r, rlow, rhigh},
  {y, ylow, yhigh}, ContourSmoothing -> Automatic, PlotPoints -> 50,
  Contours -> {0}, Axes -> True, AxesLabel -> {"r", "y"},
  Epilog -> {{Red, Text["A", {0.8, 8}]}, {Red, Text["B", {2, 13}]}];
```



The black region signals $\{r, y\}$'s for which $y' = f[t, y] < 0$.
The white region signals $\{r, y\}$'s for which $y' = f[t, y] > 0$.

Owing to the transcendental nature of $f[t, y]$:

```
| f[t, y]
-r + E^{-y/5} y
```

The phase line instruction will not run for this setup. But nevertheless, you can use the plot you see above to answer the following questions:

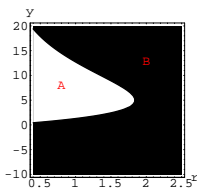
If you go with an r and a starting value on $y[0]$ so that $\{r, y[0]\}$ plots out inside the region labeled A, then what do you expect to happen to the corresponding solution of $y'[t] = f[t, y[t]]$?

Go with $r = 1.0$ and a starting value on $y[0]$ so that $\{1, y[0]\}$ plots out inside the region labeled A and estimate (by reading the plot) the limiting value of $y[t]$ as t goes to ∞ of the corresponding solution of $y'[t] = f[t, y[t]]$.

□G.5.b.ii)

Stay with the same set up as in part i) immediately above and take another look at the bifurcation plot:

```
| Show[bifurcationplot];
```



The black region signals $\{r, y\}$'s for which $y' = f[t, y] < 0$.
The white region signals $\{r, y\}$'s for which $y' = f[t, y] > 0$.

If you go with an r and a starting value of $y[0]$ so that $\{r, y[0]\}$ plots out inside the part of the region labeled B that sits directly ABOVE the region labeled A, then what do you expect to happen to the corresponding solution of $y'[t] = f[t, y[t]]$?

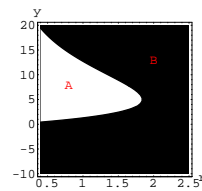
Go with $r = 1.0$ and a starting value of $y[0]$ so that $\{1, y[0]\}$ plots out inside the part of the region labeled B that sits directly above the region labeled A and estimate (by reading the plot) the limiting value of $y[t]$ as t goes to infinity of the corresponding solution of $y'[t] = f[t, y[t]]$.

Go with $r = 1.0$ and a starting value of $y[0]$ so that $\{1, y[0]\}$ plots out inside the part of the region labeled B that sits directly BELOW the region labeled A and estimate (by reading the plot) the limiting value of $y[t]$ as t goes to infinity of the corresponding solution of $y'[t] = f[t, y[t]]$.

□G.5.b.iii)

Stay with the same set up as in part i) immediately above and take another look at the bifurcation plot:

```
| Show[bifurcationplot];
```



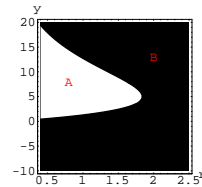
The black region signals $\{r, y\}$'s for which $y' = f[t, y] < 0$.
The white region signals $\{r, y\}$'s for which $y' = f[t, y] > 0$.

Diffeq folks like to say that a parameter point $r = \text{rbifurc}$ is a bifurcation value if the family of solutions corresponding to an r slightly larger than rbifurc look qualitatively different from the family of solutions corresponding to an r slightly less than rbifurc . Use the plot above to estimate the bifurcation point rbifurc for this diffeq.

□G.5.b.iv)

Stay with the same set up as in part i) immediately above and take another look at the bifurcation plot:

```
| Show[bifurcationplot];
```



For what choices of $\{r, y[0]\}$ do you expect to run into the problem of sensitive dependence on starter conditions?

Is there any worry about sensitive dependence on starter conditions when r is safely larger than the bifurcation point rbifurc ?

□G.5.c) Hysteresis

Here's another single diffeq containing a parameter r :

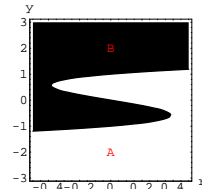
```
Clear[diffeq, y, t, f, r, d, starter];
f[t_, y_] = r + y - y^3;

setup =
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == r + y[t] - y[t]^3
y[0] == starter
```

And a bifurcation plot:

```
{ylow, yhigh} = {-3, 3};
{rlow, rhigh} = {-0.5, 0.5};

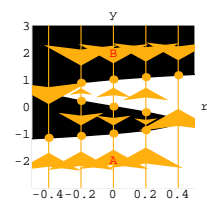
bifurcationplot = ContourPlot[f[t, y], {r, rlow, rhigh},
  {y, ylow, yhigh}, ContourSmoothing -> Automatic, PlotPoints -> 50,
  Contours -> {0}, Axes -> True, AxesLabel -> {"r", "y"},
  Epilog -> {{Red, Text["A", {0, -2}]}, {Red, Text["B", {0, 2}]}];
```



The black region signals $\{r, y\}$'s for which $y' = f[t, y] < 0$.
The white region signals $\{r, y\}$'s for which $y' = f[t, y] > 0$.

```
Clear[phaseline, r, y];
phaseline[r_] :=
PhaseLine[f[t, y], {y, ylow, yhigh}, LightCadmiumYellow, r]
h = (rhigh - rlow) / 10;
jump = (rhigh - rlow) / 5;
phaselines = Table[phaseline[r], {r, rlow + h, rhigh - h, jump}];

newphaseplot = Show[bifurcationplot, phaselines,
  PlotRange -> {{rlow, rhigh}, {ylow, yhigh}}];
0.2
```



Diffeq folks like to say that a parameter point $r = \text{rbifurc}$ is a bifurcation value if the family of solutions corresponding to an r slightly larger than $r = \text{rbifurc}$ look qualitatively different from the family of solutions corresponding to an r slightly less than rbifurc . Use the plot above to estimate the bifurcation points rbifurc for this diffeq.

For what choices of $\{r, y[0]\}$ do you expect to flirt with the problem of sensitive dependence on starter conditions?

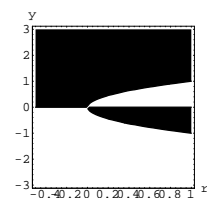
□G.5.d)

Here's another single diffeq containing a parameter r :

```
Clear[diffeq, y, t, f, r, d, starter];
f[t_, y_] = r y - y^3;
setup =
(diffeq = {y'[t] == f[y[t], r], y[0] == starter}) // ColumnForm
y'[t] == r y[t] - y[t]^3
y[0] == starter
```

And a bifurcation plot:

```
{ylow, yhigh} = {-3, 3};
{rlow, rhigh} = {-0.5, 1};
bifurcationplot = ContourPlot[f[y, r], {r, rlow, rhigh},
{y, ylow, yhigh}, ContourSmoothing -> Automatic, PlotPoints -> 50,
Contours -> {0}, Axes -> True, AxesLabel -> {"r", "y"},
Epilog -> {{Red, Text["A", {0.8, 8}]}, {Red, Text["B", {2, 13}]}];
```



The black region signals $\{r, y\}$'s for which $y' = f[t, y] < 0$.
The white region signals $\{r, y\}$'s for which $y' = f[t, y] > 0$.

Use the plot above to estimate the bifurcation point rbifurc for this diffeq.

For what choices of $\{r, y[0]\}$ do you expect to flirt with the problem of sensitive dependence on starter conditions?

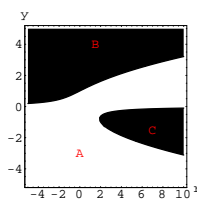
□G.5.e)

Here's yet another single diffeq containing a parameter r :

```
Clear[diffeq, y, t, f, r, d, starter];
f[t_, y_] = 1 + r y - y^3;
setup =
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == 1 + r y[t] - y[t]^3
y[0] == starter
```

And a phase plot:

```
{ylow, yhigh} = {-5, 5};
{rlow, rhigh} = {-5, 10};
bifurcationplot = ContourPlot[f[t, y], {r, rlow, rhigh},
{y, ylow, yhigh}, ContourSmoothing -> Automatic, PlotPoints -> 50,
Contours -> {0}, Axes -> True, AxesLabel -> {"r", "y"},
Epilog -> {{Red, Text["A", {0, -3}]}, {Red, Text["B", {1.5, 4}]},
{Red, Text["C", {7, -1.5}]}];
```



The black region signals $\{r, y\}$'s for which $y' = f[t, y] < 0$.
The white region signals $\{r, y\}$'s for which $y' = f[t, y] > 0$.

Use the plot above to estimate the bifurcation point rbifurc for this diffeq.

For what choices of $\{r, y[0]\}$ do you expect to flirt with the problem of sensitive dependence on starter conditions?

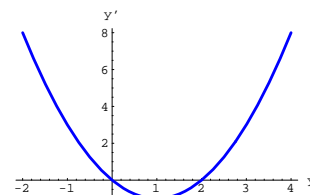
□G.5.f)

Here's another diffeq, but this one contains no parameter:

```
Clear[diffeq, y, t, f, r, d, starter];
f[t_, y_] = (y - 1)^2 - 1;
setup =
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == -1 + (-1 + y[t])^2
y[0] == starter
```

Because there is no parameter, you can't go for the contour plots as above; so you do something similar:

```
phaseplot =
Plot[f[t, y], {y, -2, 4}, PlotStyle -> {{Blue, Thickness[0.01]}},
AxesLabel -> {"y", "y'"}, AspectRatio ->  $\frac{1}{\text{GoldenRatio}}$ ];
```



Indicate how this plot signals that when you go with $0 < y[0] < 2$, then the corresponding solution of

$$y'[t] = f[t, y[t]]$$

tends to 0 as $t \rightarrow \infty$.

For what choices of $y[0]$ are you flirting with sensitive dependence on starter conditions?

G.6) Analysis of $f[t, y]$

□G.6.a.i)

You are given this diffeq to analyze:

```
Clear[diffeq, y, t, f, starter];
f[t_, y_] = y - 2.2 E^-0.3 t;
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == -2.2 E^-0.3 t + y[t]
y[0] == starter
```

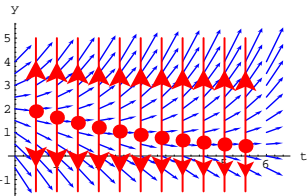
The first thing you do is look at:

```
f[t, y]
-2.2 E^-0.3 t + y
```

And you say to yourself: "One single phase line is not enough. If I want to use phase lines, I'll want several."

```
Clear[phaseline, t, y];
phaseline[t_] := PhaseLine[f[t, y], {y, ylow - 1, yhigh + 1}, Red, t]
{ylow, yhigh} = {-0.5, 4.0};
{tlow, thigh} = {0, 6};
jump = 0.5;
phaselines = Table[phaseline[t], {t, tlow + jump, thigh - jump, jump}];
flowplot = Table[Arrow[{1, f[t, y]}, Tail -> {t, y},
VectorColor -> Blue, ScaleFactor -> 0.4, HeadSize -> 0.2],
```

```
{t, tlow, thigh, jump}, {y, ylow, yhigh, jump}];
Show[flowplot, phaselines, Axes → True, AxesLabel → {"t", "y"},
  AspectRatio →  $\frac{1}{\text{GoldenRatio}}$ ];
```



Look again at the formula for $f[t, y]$:

```
| f[t, y]
-2.2 E^-0.3 t + y
```

What is it about the formula for $f[t, y]$ that made you want to use more than one phase line?

□G.6.a.ii)

You are given a new diffeq to analyze:

```
Clear[diffeq, y, t, f, starter];
f[t_, y_] = 0.9 (y - 1) (y - 3);
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == 0.9 (-3 + y[t]) (-1 + y[t])
y[0] == starter
```

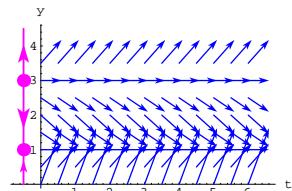
Then you look at:

```
| f[t, y]
0.9 (-3 + y) (-1 + y)
```

And you say to yourself: "One single phase line is enough:"

```
{ylow, yhigh} = {0, 3.5};
phaseline = PhaseLine[f[t, y], {y, ylow, yhigh + 1}, Magenta, -0.5];
{tlow, thigh} = {0, 6};
jump = 0.5;
flowplot = Table[Arrow[{1, f[t, y]}, Tail → {t, y},
  VectorColor → Blue, ScaleFactor → 0.6, HeadSize → 0.3],
  {t, tlow, thigh, jump}, {y, ylow, yhigh, jump}];
```

```
Show[flowplot, phaseline, Axes → True, AxesLabel → {"t", "y"},
  AspectRatio →  $\frac{1}{\text{GoldenRatio}}$ ];
```



Look again at the formula for $f[t, y]$:

```
| f[t, y]
0.9 (-3 + y) (-1 + y)
```

What is it about the formula for $f[t, y]$ that made you know you could get by with just one phase line?

□G.6.a.iii)

You are given a new diffeq to analyze:

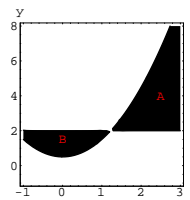
```
Clear[diffeq, y, t, f, r, starter];
f[t_, y_] = 3.9  $\left(\frac{y}{x^2 + 0.5} - 1\right)$  (y - 2);
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == 3.9 (-2 + y[t])  $\left(-1 + \frac{y[t]}{0.5 + x^2}\right)$ 
y[0] == starter
```

Then you look at:

```
| f[t, y]
3.9 (-2 + y)  $\left(-1 + \frac{y}{0.5 + x^2}\right)$ 
```

And you say to yourself: "A bifurcation plot is just the ticket."

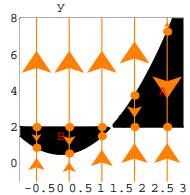
```
{ylow, yhigh} = {-1, 8};
{rlow, rhigh} = {-1, 3};
bifurcationplot = ContourPlot[f[t, y], {x, rlow, rhigh},
  {y, ylow, yhigh}, ContourSmoothing → Automatic, PlotPoints → 50,
  Contours → {0}, Axes → True, AxesLabel → {"x", "y"},
  Epilog → {{Red, Text["A", {2.5, 4}], {Red, Text["B", {0, 1.5}]}];
```



The black region signals $\{r, y\}$'s for which $y' = f[t, y] < 0$.
The white region signals $\{r, y\}$'s for which $y' = f[t, y] > 0$.

And you throw in some phase lines corresponding to different r 's:

```
Clear[phaseline, r, y];
phaseline[r_] := PhaseLine[f[t, y], {y, ylow, yhigh}, Orange, r]
h =  $\frac{rhigh - rlow}{10}$ ;
jump =  $\frac{rhigh - rlow}{5}$ ;
phaselines = Table[phaseline[r], {r, rlow + h, rhigh - h, jump}];
newphaseplot = Show[bifurcationplot, phaselines,
  PlotRange → {{rlow, rhigh}, {y, yhigh}}];
```



Look again at the formula for $f[t, y]$:

```
| f[t, y]
3.9 (-2 + y)  $\left(-1 + \frac{y}{0.5 + x^2}\right)$ 
```

What is it about the formula for $f[t, y]$ that made you want to go with a bifurcation plot and a sampling of phase lines for various r 's?

□G.6.a.iv)

You are given yet another new diffeq to analyze:

```
Clear[diffeq, y, t, f, r, starter];
f[t_, y_] = 3.9  $\left(\frac{y}{x^2 + 0.5} - 1\right)$  (t y - 2);
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == 3.9  $\left(-1 + \frac{y[t]}{0.5 + x^2}\right)$  (-2 + t y[t])
y[0] == starter
```

Then you look at:

```
| f[t, y]
3.9  $\left(-1 + \frac{y}{0.5 + x^2}\right)$  (-2 + t y)
```

And you say: "None of the graphics I used above gives me a clean analysis of the behavior of the solutions of this diffeq." You are right. Why are you right?

□G.6.b)

As entered above, each diffeq was specified by a function $f[t, y]$ (which may or may not contain an extra parameter or two). The key to understanding what the diffeq is all about is a cursory look at the given formula for $f[t, y]$. Write a paragraph or two about what you look for before you go on.

□G.6.c.i)

Now you have the opportunity to practice what you preach.

Here is a new diffeq:

```
Clear[diffeq, y, t, f, starter];
f[t_, y_] = 0.2 (y - 2) (y - 4);
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == 0.2 (-4 + y[t]) (-2 + y[t])
y[0] == starter
```

To analyze this diffeq, would you choose

- single phase line,
- multiple phase lines
- bifurcation plots
- none of the above?

□G.6.c.ii)

Here is a new diffeq:

```
Clear[diffeq, y, t, f, r, starter];
f[t_, y_] = 0.2 (y - 1) (  $\frac{y}{x^2 + 0.5} - 4$  );
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == 0.2 (-1 + y[t]) (-4 +  $\frac{y[t]}{0.5 + x^2}$ )
y[0] == starter
```

To analyze this diffeq, would you choose

- single phase line,
- multiple phase lines
- bifurcation plots
- none of the above?

□G.6.c.iii)

Here is a new diffeq:

```
Clear[diffeq, y, t, f, starter];
f[t_, y_] = (  $\frac{1}{2}$  y Sin[t] - 1 ) (y - Cos[t]);
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == (-Cos[t] + y[t]) (-1 +  $\frac{1}{2}$  Sin[t] y[t])
y[0] == starter
```

To analyze this diffeq, would you choose

- single phase line,
- multiple phase lines
- bifurcation plots
- none of the above?

□G.6.c.iv)

Here is a new diffeq:

```
Clear[diffeq, y, t, f, starter, r];
f[t_, y_] = (  $\frac{y r}{2} - 1$  ) (y - r);
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
```

```
y'[t] == (-r + y[t]) (-1 +  $\frac{1}{2}$  r y[t])
y[0] == starter
```

To analyze this diffeq, would you choose

- single phase line,
- multiple phase lines
- bifurcation plots
- none of the above?

□G.6.c.v)

Here is a new diffeq:

```
Clear[diffeq, y, t, f, starter, r];
f[t_, y_] = (y r / 2 - 1) (y - t);
(diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == (-t + y[t]) (-1 +  $\frac{1}{2}$  r y[t])
y[0] == starter
```

To analyze this diffeq, would you choose

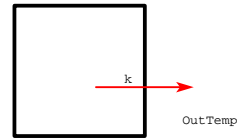
- single phase line,
- multiple phase lines
- bifurcation plots
- none of the above?

G.7) Controlling the temperature

This is a problem from an area of mathematical engineering called control theory. If your university has an electrical engineering department, you're likely to find control theory folks. Some math departments also have control theory folks as well.

Here's a house whose internal temperature at time t is measured by the function $y[t]$:

```
house = Graphics[
  {Thickness[0.01], Line[{{0, 0}, {0, 1}, {1, 1}, {1, 0}, {0, 0}}]};
heatarrow = Arrow[{0.75, 0}, Tail → {0.625, 0.375}, VectorColor → Red];
heatlegends = Graphics[
  {Text["k", {0.875, 0.4375}],
   Text["y[t] = internal temp", {0.5, 9.75}],
   Text["OutTemp", {1.5, 0.125}]}];
Show[house, heatarrow, heatlegends,
  PlotRange → {{-1, 1.9}, {-0.25, 1.25}}];
```



Newton's law of cooling says that the rate at which heat leaves the house to the outside is directly proportional to the difference between the outside temperature and the temperature inside the room. In math talk, this says

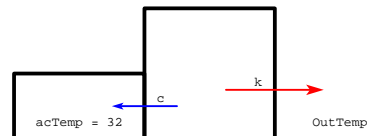
$$y'[t] = k(\text{OutTemp} - y[t])$$

for a positive proportionality constant k which is determined by the insulation of the house.

Glancing at the differential equation, you can tell if the outside temperature is higher than the temperature inside, then $y'[t]$ is positive, and so the room is heating up.

But nobody wants to swelter in heat and soak in sweat; that's why air conditioning is so popular. To air condition this simple house, attach to the house an extra air conditioning room full of ice, 32 degrees (F) cold:

```
ACroom = Graphics[
  {Thickness[0.01],
   Line[{{0, 0}, {0, 0.5}, {-1, 0.5}, {-1, 0}, {0, 0}}]};
ACarrows = Arrow[{-0.5, 0}, Tail → {0.25, 0.25}, VectorColor → Blue];
AClegends = Graphics[
  {Text["c", {0.125, 0.3125}], Text["acTemp = 32", {-0.5, 0.125}]}];
Show[house, heatarrow, heatlegends, ACroom, ACarrows, AClegends,
  PlotRange → {{-1, 1.9}, {-0.25, 1.25}}];
```



The same Newton's law of cooling says that the combined effects of the heating from the outside and the cooling from the ice room results in this crisp differential equation:

$$y'[t] = c(\text{acTemp} - y[t]) + k(\text{OutTemp} - y[t])$$

```
Clear[OutTemp, t, y, c, k];
acTemp = 32;
Clear[f, x, y, a, b, t];
f[t_, y_] = k (OutTemp - y) + c (acTemp - y);
model = diffeq
  {y'[t] == f[t, y[t]], y[0] == starter}
y'[t] == c (32 - y[t]) + k (OutTemp - y[t])
y[0] == starter
```

Here and throughout the rest of this problem,

$y[t]$ is measured in Fahrenheit degrees and t is measured in hours.

In this model, the value of k is determined by the insulation in the walls and roof of the house. You control the value of c by varying the speed of the blower that pushes cold air from the ice room into the house. The bigger you make c , the more cooling the house gets.

□G.7.a.i) Controlling the air conditioner when it's hot outside

Go with the sample case in which $k = 0.4$ and the outdoor temperature is 95° :


```

k = 0.4;
OutTemp = 95;
acTemp = 32;
Clear[f, x, y, a, b, t];
f[t_, y_] = k (OutTemp - y) + c (acTemp - y);

model = diffeq
  {y'[t] == f[t, y[t]], y[0] == starter}
y'[t] == c (32 - y[t]) + 0.4 (95 - y[t])
y[0] == starter

```

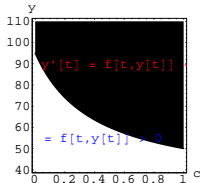
Look at the phase plot:

```

{ylow, yhigh} = {40, 110};
{clow, chigh} = {0, 1};

ContourPlot[f[t, y], {c, clow, chigh},
  {y, ylow, yhigh}, ContourSmoothing -> Automatic, PlotPoints -> 50,
  Contours -> {0}, Axes -> True, AxesLabel -> {"c", "y"},
  Epilog -> {{Red, Text["y'[t] = f[t,y[t]] < 0", {0.6, 90}]},
  {Blue, Text["y'[t] = f[t,y[t]] > 0", {0.3, 55}]}}];

```



Read this phase plot and then come up with the c that will control the long term temperature of the room at approximately 72 degrees (F).

Explain how the plot indicates that your control should work for any starting temperature $y[0]$ between 40 and 110.

Show off your work with a couple of great solution plots. Throw in a flow plot too.

□G.7.a.ii) Variable outdoor temperature.

Up to this point, you went with the assumption that the outdoor temperature remains a constant 95 degrees. This is bogus because the temperature varies as time t advances. Here's how you set up a model in which the outdoor temperature $\text{OutTemp}[t]$ varies as t advances.

```

Clear[f, t, y, c, OutTemp, k];
acTemp = 32;
k = 0.4;
idealtemp = 70;

f[t_, y_] = k (OutTemp[t] - y) + c[t] (acTemp - y);

model =
  (diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == c[t] (32 - y[t]) + 0.4 (OutTemp[t] - y[t])
y[0] == starter

```

Notice that in this model, the air conditioning strength $c[t]$ is also allowed to vary with time t .

Come up with a function $c[t]$ expressed in terms of $\text{OutTemp}[t]$ so that solutions $y[t]$ of:

```

diffeq
  {y'[t] == c[t] (32 - y[t]) + 0.4 (OutTemp[t] - y[t]), y[0] == starter}

```

can be expected to settle in on the ideal temperature of 72°.

The beauty of this is that you can have the outdoor temperature control the air conditioner instantaneously.

□G.7.a.iii) Putting your $c[t]$ control function to work.

Here are some actual outdoor temperatures in the form {hour, temperature} collected over a steamy 24 hours in Columbus, Ohio.

```

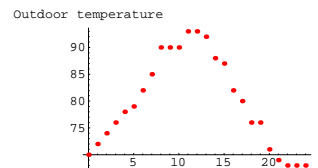
Columbustemps =
  {{0, 70}, {1, 72}, {2, 74}, {3, 76}, {4, 78},
  {5, 79}, {6, 82}, {7, 85}, {8, 90}, {9, 90},
  {10, 90}, {11, 93}, {12, 93}, {13, 92}, {14, 88},
  {15, 87}, {16, 82}, {17, 80}, {18, 76}, {19, 76},
  {20, 71}, {21, 69}, {22, 68}, {23, 68}, {24, 68}};

```

```

heatplot =
  ListPlot[Columbustemps, PlotStyle -> {PointSize[0.02], Red},
  AxesLabel -> {"t", "Outdoor temperature"}];

```



To get a good approximation of $\text{OutTemp}[t]$, fit the temperature data with well-chosen functions:

```

Clear[OutTemp, t];
OutTemp[t_] = Fit[Columbustemps, {1, Sin[2πt/24], Cos[2πt/24]}, t]
80.3314 - 11.2839 Cos[πt/12] + 3.25691 Sin[πt/12]

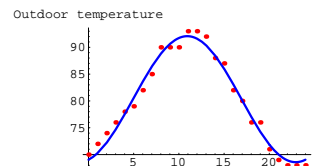
```

Check the fit:

```

fitplot = Plot[OutTemp[t], {t, 0, 24},
  PlotStyle -> {{Blue, Thickness[0.01]}}, DisplayFunction -> Identity];
Show[heatplot, fitplot, DisplayFunction -> $DisplayFunction];

```



Pretty decent fit.

The resulting differential equation model is:

```

Clear[f, t, y, c, k];
acTemp = 32;
k = 0.4;
idealtemp = 72;

f[t_, y_] = k (OutTemp[t] - y) + c[t] (acTemp - y);

```

```

model =
  (diffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == c[t] (32 - y[t]) + 0.4 (80.3314 - 11.2839 Cos[πt/12] + 3.25691 Sin[πt/12] -
y[0] == starter

```

In the last part you came up with a formula for $c[t]$ in terms of $\text{OutTemp}[t]$ that you hoped would make the internal temperature settle in on the ideal temperature of 72°.

Type in your $c[t]$ formula here and run:

```

c[t_] = TypeHere
TypeHere

```

Test out your control function $c[t]$ in the cases that the initial temperature of the house is 85°, 95° and 105°:

```

Clear[y1, y2, y3, t];
{starter1, starter2, starter3} = {85, 95, 105};
endtime = 36;
ygoal = 72;

y1[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter1, y[t], {t, 0, endtime}][[1]];
y2[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter2, y[t], {t, 0, endtime}][[1]];
y3[t_] = y[t] /.
  NDSolve[diffeq /. starter -> starter3, y[t], {t, 0, endtime}][[1]];

Plot[{y1[t], y2[t], y3[t]}, {t, 0, endtime},
  PlotStyle -> {{Thickness[0.015], Red}}, AspectRatio -> 1/GoldenRatio,
  PlotRange -> {ygoal, yhigh}, AxesLabel -> {"t", "y[t]"}, Epilog ->
  {Green, Thickness[0.01], Line[{0, ygoal}, {endtime, ygoal}]}];

```

Test out your control function $c[t]$ on other cases.

Report what you find. Show a flow plot and analyze it to see whether your controlling function fails for extreme starter temperatures.

□G.7.a.iv)

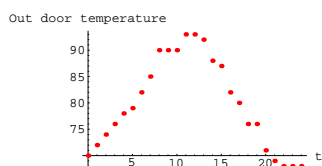
In the last part, you went with Columbus, Ohio temperature data:

```

Columbustemps = {{0, 70}, {1, 72}, {2, 74}, {3, 76}, {4, 78}, {5, 79},
  {6, 82}, {7, 85}, {8, 90}, {9, 90}, {10, 90}, {11, 93}, {12, 93},
  {13, 92}, {14, 88}, {15, 87}, {16, 82}, {17, 80}, {18, 76},
  {19, 76}, {20, 71}, {21, 69}, {22, 68}, {23, 68}, {24, 68}};

```

```
heatplot = ListPlot[Columbustemps, PlotStyle -> {PointSize[0.02], Red},
  AxesLabel -> {"t", "Out door temperature"}];
```

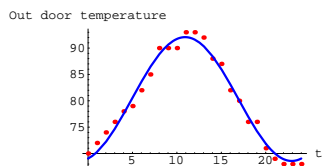


And you fit the temperature data with well-chosen functions:

```
Clear[OutTemp, t];
OutTemp[t_] = Fit[Columbustemps, {1, Sin[2πt/24], Cos[2πt/24]}, t]
80.3314 - 11.2839 Cos[πt/12] + 3.25691 Sin[πt/12]
```

Check the fit:

```
fitplot = Plot[OutTemp[t], {t, 0, 24},
  PlotStyle -> {{Blue, Thickness[0.01]}}, DisplayFunction -> Identity];
Show[heatplot, fitplot, DisplayFunction -> $DisplayFunction];
```



Nice fit.

Why is it usually a good idea to use, as above, combinations of $1, \sin\left[\frac{2\pi t}{24}\right], \cos\left[\frac{2\pi t}{24}\right]$

to go after a decent fit of 24 hour temperature data?

□Tip:

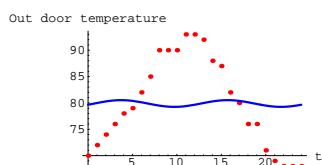
When you change the 24 to, say 12, you get garbage:

```
Clear[OutTemp, t];
OutTemp[t_] = Fit[Columbustemps, {1, Sin[2πt/12], Cos[2πt/12]}, t]
```

```
79.8894 - 0.23533 Cos[πt/6] + 0.591506 Sin[πt/6]
```

Check the fit:

```
fitplot = Plot[OutTemp[t], {t, 0, 24},
  PlotStyle -> {{Blue, Thickness[0.01]}}, DisplayFunction -> Identity];
Show[heatplot, fitplot, PlotRange -> All,
  DisplayFunction -> $DisplayFunction];
```



In addition to the fact that there are 24 hours in a day, there is something about $\sin\left[\frac{2\pi t}{24}\right]$ and $\cos\left[\frac{2\pi t}{24}\right]$ that makes these natural fitters of 24 hour temperature data. What is it?

G.8) The falling body and the leaking bucket:

Setting parameters to fit the situation.

Getting there in infinite time versus getting there in finite time

□G.8.a.i) The leaking bucket

This part was adapted from the Hubbard-West text "Differential Equations: A Dynamical Systems Approach." Springer-Verlag, 1991
DiffEq&Mathematica students will find lots of good stuff in this modern text.

Here is a sample of the differential equation of radioactive decay:

```
Clear[diffeq, y, t, f, r, starter];
f[t_, y_] = -r y;
(radiodiffeq = {y'[t] == f[t, y[t]], y[0] == 100}) // ColumnForm
```

```
y'[t] == -r y[t]
y[0] == 100
```

Here Γ is a positive number.

The negative sign indicates that $y[t]$ decreases as t increases.

Here $y[t]$ measures the percentage of original radioactivity that is left after t time units. Formulas for the solutions, which you already know, are easily available:

```
DSolve[radiodiffeq, y[t], t]
{{y[t] -> 100 E^-r t}}
```

Although $y[t] \rightarrow 0$ as $t \rightarrow \infty$, there is no way to get $y[t] = 0$ for a finite t .

But when you have a leaking bucket full of water, then all the water leaks out in finite time.

To study this from the diffeq point of view, go with differential equation of leaking cylindrical bucket 10 inches high:

A derivation of this differential equation is given in the Hubbard-West text.

```
Clear[diffeq, y, t, f, r, starter];
f[t_, y_] = -r Sqrt[y];
starter = 10;
(bucketdiffeq = {y'[t] == f[t, y[t]], y[0] == starter}) // ColumnForm
y'[t] == -r Sqrt[y[t]]
y[0] == 10
```

Here Γ is a positive number you will determine later.

$y[t]$ measures the depth of the water in the bucket at time t .

The negative sign indicates that $y[t]$ decreases as t increases.

The starting condition $y[0] = 10$ indicates that the bucket is full at time $t = 0$.

You can go after a formula for $y[t]$ by separating and integrating:

```
Clear[r, t, y];
newleft = y'[t] / Sqrt[y[t]];
newright = -r;
samediffeq = newleft == newright;
```

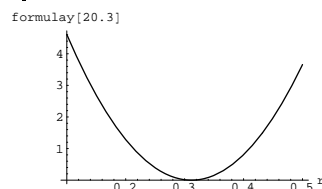
```
integratedeqn = Integrate[newleft dt == Integrate[newright dt];
neweqn = integratedeqn /. y[0] -> starter;
```

```
Clear[formulay];
ysol = Solve[neweqn, y[t]];
formulay[t_] = y[t] /. ysol[[1]]
1/4 (40 - 4 Sqrt[10] r t + r^2 t^2)
```

Disregard the error messages.

Given that the bucket leaks dry in 20.3 seconds, look at this plot of formulay[20.3] as a function of r :

```
Plot[formulay[20.3], {r, 0.1, 0.5}, AspectRatio -> 1/GoldenRatio,
  AxesLabel -> {"r", "formulay[20.3]"}];
```



And use what you see to set r so that formulay[20.3] = 0.

□G.8.a.ii)

Having set r so that formulay[20.3] = 0, plot formulay[t] as a function of t for t running from 0 to 20.3.

Then plot formulay[t] as a function of t for t running from 0 to 40.

Do you trust the second plot as accurate?

Why or why not? Does this the second plot enter the danger zone?

□G.8.b.i) Terminal velocity of a falling body

A body falling in a vacuum is subject only to the force of acceleration. This means that the velocity $v[t]$ of a body falling in a vacuum solves the simple diffeq

$$v'[t] = 9.8 \text{ (meters per second per second).}$$

When air resistance is incorporated, many folks like to assume that the air resistance is proportional to the velocity itself. This leads to the model:

```
Clear[f, v, r, t];
f[t_, v_] = 9.8 - r v;

model = (diffeq = {v'[t] == f[t, v[t]], v[0] == 0}) // ColumnForm

v'[t] == 9.8 - r v[t]
v[0] == 0
```

You are investigating the terminal velocity; so you type and run:

```
Solve[f[t, v] == 0, v]
{{v -> 9.8/r}}
```

If you are given that the terminal velocity is 25 meters per second, then what r do you go with?

□G.8.b.ii) The falling cat

This part was adapted from an early draft of the book by Paul Blanchard, Robert Devany and G. R. Hall, "Differential Equations," PWS Publishers. If you like differential equations, then you will like this book.

According to the New York Times (August 22, 1989), cats sometimes fall from high apartment building windows and survive. The same article reports that experts say that the terminal velocity of a falling cat is about 27 meters per second (60 miles per hour). Set r so that in the model above, the terminal velocity is

27 meters per second

and plot $v[t]$ as a function of t .

About how long does this model predict it will take for $v[t]$ to begin to close in on the terminal velocity?

How far does this model predict that the cat must fall before its velocity is approximately equal to its terminal velocity?

Do you think the falling cat reaches its terminal velocity in
in finite time

or
in infinite time?

□Tip:

If you can get your hands on a formula for $v[t]$, then you can measure how far the cat falls during the first s seconds by calculating

$$\int_0^s v[t] dt.$$

If you use `NDSolve` to get your plot of $v[t]$, then you can measure how far the cat falls during the first s seconds by calculating

$$NIntegrate[v[t], {t, 0, s}].$$

□G.8.b.iii)

Some folks prefer the model that makes make air resistance proportional to the square of the velocity:

Most folks prefer the first model for hard objects such as baseballs. They often prefer the second model for fluffy things such as rag mops.

```
Clear[f, v, r, t];
f[t_, v_] = 9.8 - r v^2;

model = (diffeq = {v'[t] == f[t, v[t]], v[0] == 0}) // ColumnForm

v'[t] == 9.8 - r v[t]^2
v[0] == 0
```

Investigate, as above, the terminal velocity as a function of r , and then set r so that the terminal velocity is 27 meters per second.

Plot $v[t]$ as a function of t .

About how long does this model predict it takes for $v[t]$ to begin to close in on the terminal velocity?

How far does this model predict that the cat must fall before its velocity is approximately equal its terminal velocity?

□G.8.b.iv)

Compare the results you got from both models. Is the information you got from each strikingly different? Is there a danger zone in either model?

G.9) Hand symbolic manipulation: Separating and integrating**□G.9.a.i)**

Here's a little diffeq:

```
Clear[diffeq, y, t, f, r, b, starter];
f[t_, y_] = 1 + (y/2);

(diffeq = {y'[t] == f[t, y[t]], y[0] == 0.5}) // ColumnForm

y'[t] == 1 + y[t]/2
y[0] == 0.5
```

Use separation of variables to try to come up with a formula for the solution.

Analyze the formula and determine whether the solution escapes to infinity in finite time or infinite time.

□G.9.a.ii)

Here's a little diffeq related to the diffeq in part i) above:

```
Clear[diffeq, y, t, f, r, b, starter];
f[t_, y_] = 1 + (y/2)^2;

(diffeq = {y'[t] == f[t, y[t]], y[0] == 0.5}) // ColumnForm

y'[t] == 1 + y[t]^2/4
y[0] == 0.5
```

Use separation of variables to try to come up with a formula for the solution.

Analyze the formula and determine whether the solution escapes to infinity in finite time or infinite time.

□G.9.a.iii)

Here's a little diffeq related to the diffeq in part i) above:

```
Clear[diffeq, y, t, f, r, b, starter];
f[t_, y_] = Sqrt[1 - y^2];

(diffeq = {y'[t] == f[t, y[t]], y[0] == 0}) // ColumnForm

y'[t] == Sqrt[1 - y[t]^2]
y[0] == 0
```

Use separation of variables to try to come up with a formula for the solution.

Analyze the formula and determine the t 's for which the formula is correct.

□Tip:

The danger zone!

□G.9.a.iv)

Here's a new differential equation:

```
Clear[diffeq, y, t, f, r, b, starter];
f[t_, y_] = t - y^2;

(diffeq1 = {y'[t] == f[t, y[t]], y[0] == 4.8}) // ColumnForm

y'[t] == t - y[t]^2
y[0] == 4.8
```

Separation of variables is useless for this one.

Got any idea why?