Section 8.4: Ordinary Differential equations

Created by Tomas de-Camino-Beck
tomasd@math.ualberta.ca

SIR model

Let $i(t)$ be the infected at time t and $s(t)$ suceptibles. Note that we are not really interested in recovery:

```
eq1 = s'[t] == -β s[t] i[t];
eq2 = i'[t] == β s[t] i[t] - α i[t];
```

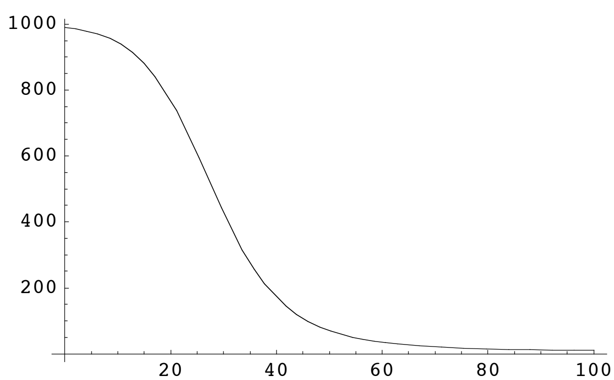
To find a numreical solution define α and β, then we use `NDSolve`:

```
α = 0.04;
β = 0.0002;
solution = NDSolve[{eq1, eq2, s[0] == 990.0, i[0] == 10}, {i, s}, {t, 0, 100}]
```

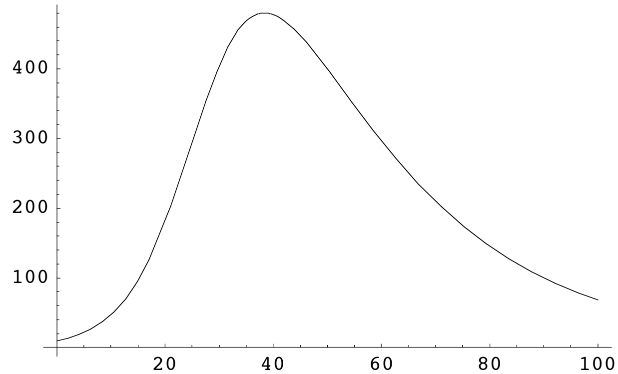
```
{{i → InterpolatingFunction[{{0., 100.}}, <>],
  s → InterpolatingFunction[{{0., 100.}}, <>]}}
```

Plot the numreical solution:

```
ParametricPlot[Evaluate[{t, s[t]} /. solution], {t, 0, 100}];
```

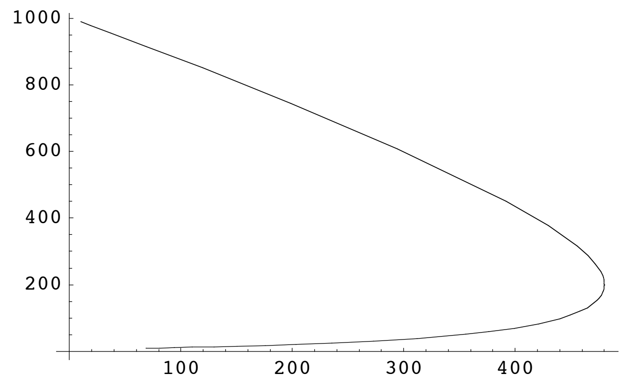


```
ParametricPlot[Evaluate[{t, i[t]} /. solution], {t, 0, 100}];
```



This is a plot of $s(t)$ against $i(t)$:

```
ParametricPlot[Evaluate[{i[t], s[t]} /. solution], {t, 0, 100}];
```



Logistic Equation

Find a solution for the logistic equation

```
sol = DSolve[{x'[t] == r x[t] (1 - x[t]), x[0] == x0}, x, t]
```

```
Solve::ifun : Inverse functions are being used by Solve, so some
   solutions may not be found; use Reduce for complete solution information. More…
```

$$\left\{\left\{x \to \text{Function}\left[\{t\}, \frac{e^{r t}\, x0}{1 - x0 + e^{r t}\, x0}\right]\right\}\right\}$$

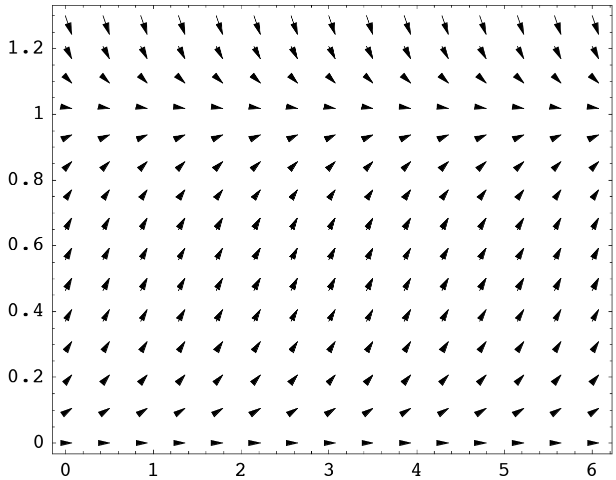
We can plot a vector field to see the trayectory of solutions:

```
<< Graphics`PlotField`
```

```
r = 2;
```

note that we ommit the fact that x is a function of time i.e. we use the non-dimensionalized logistic equation:

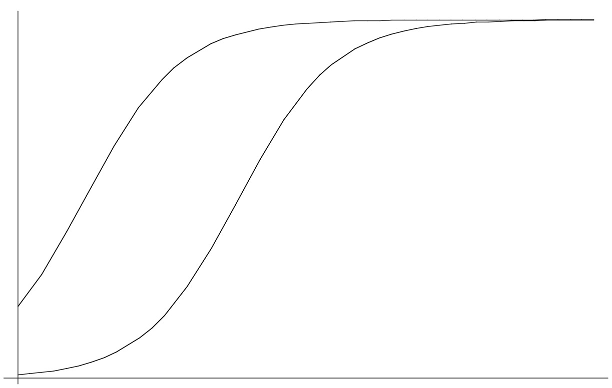
```
vectorplot =
 PlotVectorField[{1, r x (1 - x)}, {t, 0, 6}, {x, 0, 1.3}, AspectRatio → 0.8/1, Frame → True,
  FrameTicks → Automatic]
```




```
- Graphics -
```

Plot some solutions

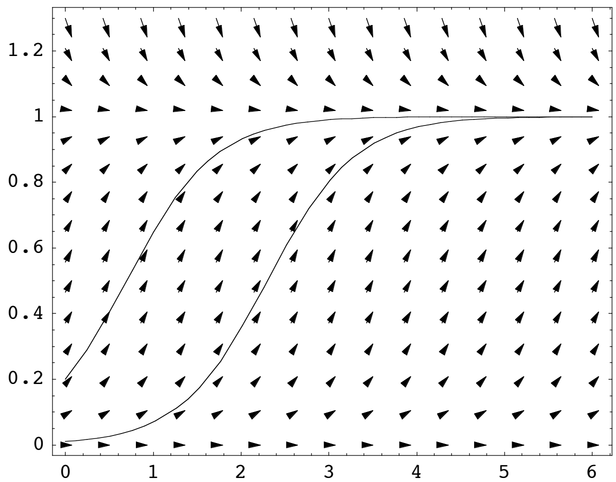
```
solplot = Plot[Evaluate[{
    x[t] /. sol /. x0 → 0.01,
    x[t] /. sol /. x0 → 0.2}], {t, 0, 6}, PlotRange → All, Ticks → None]
```



```
- Graphics -
```

Put them toghether:

```
Show[vectorplot, solplot]
```




```
- Graphics -
```

Predator Prey Model

Consider the following non-dimensional predator *v* and prey *u* model:

```
prey = u'[t] == u[t] (1 - u[t] / k) - u[t] v[t];
pred = v'[t] == g (u[t] - 1) v[t];
```

Solving for steady-states, we use the Solve function, letting $u'(t)$, $v'(t) = 0$:

```
solution = Solve[{prey /. {u'[t] → 0}, pred /. {v'[t] → 0}}, {u[t], v[t]}]
```

$$\left\{\{u[t] \to 0,\ v[t] \to 0\},\ \{u[t] \to k,\ v[t] \to 0\},\ \left\{v[t] \to 1 - \frac{1}{k},\ u[t] \to 1\right\}\right\}$$

The following code, calculates the Jacobian of a system of equations:

```
JacobianMatrix[f_List, var_List] := Outer[D, f, var];
```

Now lets compute the jacobian of the predator prey-system (note that to get the right hanbd side of the equation we use $f[\![2]\!]$:

```
J = JacobianMatrix[{prey[[2]], pred[[2]]}, {u[t], v[t]}]; MatrixForm[J]
```

$$\begin{pmatrix} 1 - \frac{2\,u[t]}{k} - v[t] & -u[t] \\ g\,v[t] & g\,(-1 + u[t]) \end{pmatrix}$$

Evaluated at the steady states:

```
J /. solution
```

$$\left\{\{\{1, 0\}, \{0, -g\}\},\ \{\{-1, -k\}, \{0, g\,(-1 + k)\}\},\ \left\{\left\{-\frac{1}{k}, -1\right\}, \left\{g\left(1 - \frac{1}{k}\right), 0\right\}\right\}\right\}$$

Note that we get our three matrices, that correspond to the steady states. Now, if we choose some parameter values for g and k, we can compute the eigenvalues in order to study the stability of this system:

```
param = {g → 1, k → 2};
```

```
evalJ = J /. solution /. param
```

$$\left\{\{\{1, 0\}, \{0, -1\}\}, \{\{-1, -2\}, \{0, 1\}\}, \left\{\left\{-\frac{1}{2}, -1\right\}, \left\{\frac{1}{2}, 0\right\}\right\}\right\}$$

In matrix from:

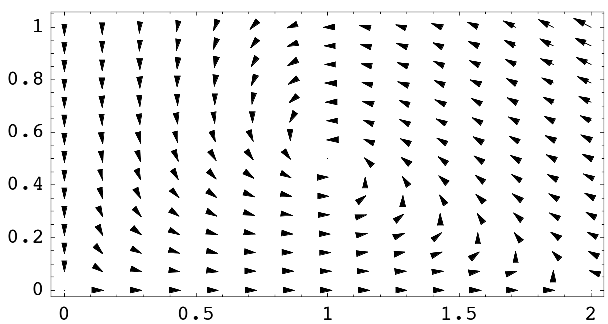
```
TableForm[MatrixForm /@ evalJ]
```

$$\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$
$$\begin{pmatrix} -1 & -2 \\ 0 & 1 \end{pmatrix}$$
$$\begin{pmatrix} -\frac{1}{2} & -1 \\ \frac{1}{2} & 0 \end{pmatrix}$$

We can also study the vector field:

```
<< Graphics`PlotField`
```

```
vp = PlotVectorField[{u (1 - u / k) - u v, g (u - 1) v} /. {g → 1, k → 2},
  {u, 0, 2}, {v, 0, 1}, Frame → True, PlotPoints → 15]
```

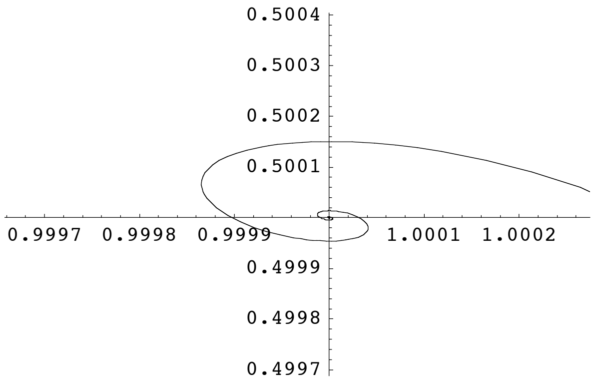


- Graphics -

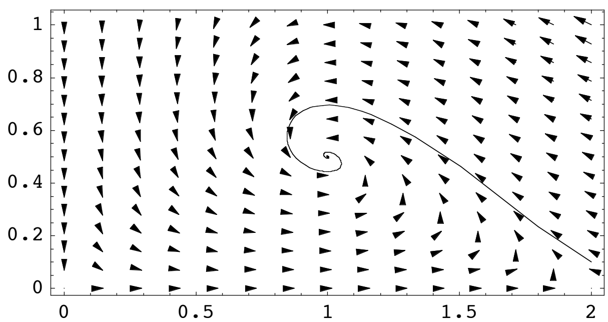
```
solution1 = NDSolve[{prey /. param, pred /. param, u[0] == 2, v[0] == 0.1}, {u, v}, {t, 0, 100}]
```

```
{{u → InterpolatingFunction[{{0., 100.}}, <>],
  v → InterpolatingFunction[{{0., 100.}}, <>]}}
```

```
phase1 = ParametricPlot[Evaluate[{u[t], v[t]} /. solution1], {t, 0, 100}];
```




```
Show[vp, phase1]
```



- Graphics -