`DSolve[eq,y,x]`	Give the solution of the differential equation in the form of a pure function

In this form of **`DSolve`** we ask for the solution for **`y`**, not for **`y[x]`**. This form of **`DSolve`** can also be used for initial and boundary value problems. The solution is now given as a pure function:

```
sol2 = DSolve[eq, y, x]              {{y -> (-2 + 3 E^#1 - #1&)}}
```

(For more about pure functions, see Section 14.2.1). The argument of the solution is **`#1`**, and a pure function is identified by the ampersand **`&`**. This form of the solution is handy in calculations:

```
y[0] /. sol2[[1]]                    1
```

■ Example 1

Before we use the solution **`sol2`** to perform some calculations, it may be useful to pick up the single solution we got above:

```
sol2 = sol2[[1]]                     {y -> (-2 + 3 E^#1 - #1&)}
```

We can now calculate, for example, the value of the solution at specific points:

```
y[1]/.sol2                           -3 + 3 E

y[x]/.sol2                           -2 + 3 E^x - x
```

We can derive and integrate the solution:

```
y'[x]/.sol2                          -1 + 3 E^x

Integrate[y[x]/.sol2, {x,0,1}]       {-11/2 + 3 E}
```

We can check that the solution **`sol2`** is correct by substituting it into the equation:

```
eq /. sol2                           {True, True}
```

Both the equation and the initial condition are satisfied, and the solution is therefore correct.

■ Example 2

Let us check that the solution of the following equation is correct:

```
eq = y'[x] == a y[x] + b x + c;

sol = DSolve[eq, y, x][[1]]
```

$$\left\{y \to \left(\frac{-b-a\,c}{a^2} + E^{a\,\#1}\,C[1] - \frac{b\,\#1}{a}\,\&\right)\right\}$$

We substitute the solution into the equation:

```
eq /. sol
```

$$-\frac{b}{a} + a\,E^{a\,x}\,C[1] == c + b\,x + a\left(\frac{-b-a\,c}{a^2} - \frac{b\,x}{a} + E^{a\,x}\,C[1]\right)$$

When simplified, the right-hand side reduces to an expression exactly the same as the left-hand side, and the solution is therefore correct:

```
% //Simplify                              True
```

What should the constant **C[1]** be to give **y[0] == d**?

```
Solve[(y[0]/.sol) == d, C[1]]
```

$$\left\{\left\{C[1] \to -\frac{-b-a\,c-a^2\,d}{a^2}\right\}\right\}$$

We can substitute this into the solution:

```
sol /. %[[1]]
```

$$\left\{y \to \left(\frac{-b-a\,c}{a^2} + \frac{E^{a\,\#1}\,(-(-b-a\,c-a^2\,d))}{a^2} - \frac{b\,\#1}{a}\,\&\right)\right\}$$

This solution really has the value **d** at 0:

```
y[0] /. % //Simplify                      d
```

■ Solution as an Expression

At first, you may feel uncomfortable with the rule **y[x]->...** or **y->...** in the solution given by **DSolve**. In time, however, you may find that the rule has advantages. Indeed, in the two preceding examples we could nicely do some computations with the solution. However, if you want, you can get rid of the rule.

sol2 = y[x] /. sol	Form an expression **sol2** from the solution **sol**

As an example, let us consider the equation of Example 1:

```
eq = {y'[x] == y[x] + x + 1, y[0] == 1};

sol = DSolve[eq, y[x], x]           {{y[x] -> -2 + 3 E^x - x}}
```

Here is the solution as a simple expression, without the rule:

```
sol2 = y[x] /. sol[[1]]             -2 + 3 E^x - x
```

(We can proceed in exactly the same way with the solution given as a pure function.) Now we can use **sol2**. We can, for example, calculate values of the solution at specific points:

```
sol2 /. x->1                        -3 + 3 E^x
```

We can derive and integrate:

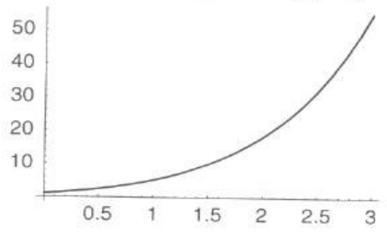
```
D[sol2,x]                           -1 + 3 E^x

Integrate[sol2, {x,0,1}]            -11/2 + 3 E
```

We can check that the solution **sol2** is correct by substituting the solution into the equation, but now we have to calculate the needed derivatives and values ourselves and write rules for **y[x]** and **y'[x]**:

```
eq /. {y[x]->sol2, y'[x]->D[sol2,x], y[0]->sol2/.x->0}
{True, True}
```

(Compare this command with the simple command **eq/.sol2** we used in Example 1.) We can plot the solution:

```
Plot[Evaluate[sol2], {x,0,3}];
```



■ Solution as a Function

We can also form a function from the solution.

`yy[x_] = y[x] /. sol`	Form a function **yy** from the solution **sol**

Note that functions are usually defined with **:=**, but here we must use only **=**, to get the value of the expression immediately assigned for **yy**. We again consider the equation of Example 1:

```
eq = {y'[x] == y[x] + x + 1, y[0] == 1};
```

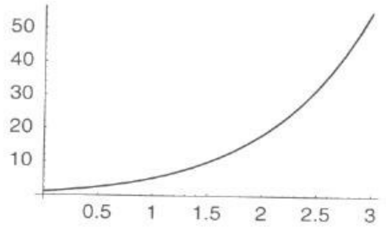
```
sol = DSolve[eq, y[x], x]          {{y[x] -> -2 + 3 E^x - x}}
```

We now define a function from the solution:

```
yy[x_] = y[x] /. sol[[1]]          -2 + 3 E^x - x
```

We can then calculate as with any function:

```
yy[1]                              -3 + 3 E

yy'[x]                             -1 + 3 E^x

Integrate[yy[x], {x,0,1}]          -11/2 + 3 E

eq /. {y -> yy}                    {True, True}

Plot[Evaluate[yy[x]], {x,0,3}];
```



22.2.2 *Using the Laplace Transform*

■ One Equation

We try to solve the equation

```
eq = y''[x] + y[x] == x;
```

by using the Laplace transform (see Section 17.4.1). The initial conditions are $y(0) = a$ and $y'(0) = b$. First we take the Laplace transform of the equation:

```
Needs["Calculus`LaplaceTransform`"]

LaplaceTransform[eq,x,s]
```

$$\text{LaplaceTransform[y[x], x, s]} + s^2\,\text{LaplaceTransform[y[x], x, s]} - s\,y[0] - y'[0] == \frac{1}{s^2}$$

and substitute the initial values:

```
% /. {y[0]->a, y'[0]->b}
```

$$-b - a\,s + \text{LaplaceTransform[y[x], x, s]} + s^2\,\text{LaplaceTransform[y[x], x, s]} == \frac{1}{s^2}$$

From this equation we solve the transform and lastly take the inverse transform, which is then the solution to the initial value problem:

```
Solve[%, LaplaceTransform[y[x],x,s]] //Simplify
```

$$\left\{\left\{\text{LaplaceTransform[y[x], x, s]} \to \frac{1 + b\,s^2 + a\,s^3}{s^2 + s^4}\right\}\right\}$$

```
InverseLaplaceTransform[%,s,x]

{{y[x] → x + a Cos[x] + (-1 + b) Sin[x]}}
```

The same solution is obtained with **DSolve**:

```
DSolve[{eq, y[0]==a, y'[0]==b}, y[x], x]

{{y[x] → x + a Cos[x] - (1 - b) Sin[x]}}
```

■ Simultaneous Equations

We solve the same initial value problem as in Example 2 of Section 22.1.3:

```
eq = {y'[x] == y[x] - z[x], z'[x] == 5y[x] - 3z[x]};

LaplaceTransform[eq,x,s] /. {y[0]->10000, z[0]->0}

{-10000 + s LaplaceTransform[y[x], x, s] ==
   LaplaceTransform[y[x], x, s] - LaplaceTransform[z[x], x, s],
  s LaplaceTransform[z[x], x, s] ==
   5 LaplaceTransform[y[x], x, s] - 3 LaplaceTransform[z[x], x, s]}

Solve[%, {LaplaceTransform[y[x],x,s], LaplaceTransform[z[x],x,s]}]
```

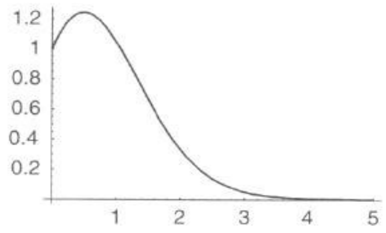
$$\left\{\left\{\text{LaplaceTransform[y[x], x, s]} \to \frac{10000\,(3 + s)}{2 + 2\,s + s^2},\ \text{LaplaceTransform[z[x], x, s]} \to \frac{50000}{2 + 2\,s + s^2}\right\}\right\}$$

```
InverseLaplaceTransform[%,s,x] //Simplify
```

$$\{\{y[x] \to 10000\,E^{-x}\,(\text{Cos}[x] + 2\,\text{Sin}[x]),\ z[x] \to 50000\,E^{-x}\,\text{Sin}[x]\}\}$$

The solution is the same as the one obtained in Section 22.1.3.

```
Plot[Evaluate[xx[t]], {t,0,5}];
```



```
{xx[0], xx[1], xx[2]}                          {1., 1.04612, 0.331509}

xx'[1]                                         -0.678228

NIntegrate[xx[t],{t,0,5}]                      2.02246
```

22.3.2 Two Equations

`sol = NDSolve[{eq1,eq2,conds}, {x[t],y[t]}, {t,a,b}]`	Solve the problem
`Plot[Evaluate[{x[t],y[t]}/.sol], {t,a,b}]`	Plot **`x[t]`** and **`y[t]`**
`ParametricPlot[Evaluate[{x[t],y[t]}/.sol], {t,a,b}]`	Plot a phase trajectory

Next we present three examples of systems of two nonlinear differential equations. For more about such systems, see Murrell (1994).

■ Example 1: Competing Species

Let us first define

```
f = x[t] (p - r x[t] - q y[t]);
g = y[t] (P - Q x[t] - R y[t]);
```

and then the model

```
eq = {x'[t] == f, y'[t] == g, x[0] == x0, y[0] == y0};
```

This model describes competing species; see Mesterton-Gibbons (1989, p. 151). The variables **`x[t]`** and **`y[t]`** are the population magnitudes at time **`t`**. First we calculate the curves (so-called isoclines) where the derivatives **`x'[t]`** and **`y'[t]`** are zero:

```
h1 = y[t] /. Solve[f==0, y[t]][[1]] /. x[t]->x
```

$$-\frac{-p+r\,x}{q}$$

```
h2 = y[t] /. Solve[g==0, y[t]][[2]] /. x[t]->x
```

$$\frac{P-Q\,x}{R}$$

Then we calculate the equilibrium points. At these points the derivatives are simultaneously zero:

```
equi = Solve[{f==0, g==0}, {x[t],y[t]}] //Simplify
```

$$\left\{\{x[t] \to 0,\ y[t] \to 0\},\ \left\{x[t] \to \frac{p}{r},\ y[t] \to 0\right\},\ \left\{x[t] \to \frac{P q - p R}{q Q - r R},\ y[t] \to \frac{p Q - P r}{q Q - r R}\right\},\ \left\{y[t] \to \frac{P}{R},\ x[t] \to 0\right\}\right\}$$

To check the nature of these points, we define a funtion to calculate a Jacobian:

```
jacobi[f_List,x_List] := Outer[D,f,x]
```

The Jacobian of **f** and **g** is

```
jac = jacobi[{f,g}, {x[t],y[t]}]
{{p - 2 r x[t] - q y[t], -q x[t]}, {-Q y[t], P - Q x[t] - 2 R y[t]}}
```

The eigenvalues of the Jacobian at the equilibrium points are

```
eig = Map[Eigenvalues[jac/.#]&, equi] //Simplify
```

$$\left\{\{p,\ P\},\ \left\{-p,\ P - \frac{p Q}{r}\right\},\ \left\{\frac{1}{2 q Q - 2 r R}\left(p\,(-Q + r)\,R + P r\,(-q + R) - \sqrt{(P r\,(q - R) + p\,(Q - r)\,R)^2 - 4\,(p Q - P r)\,(-P q + p R)\,(q Q - r R)}\right),\ \frac{1}{2 q Q - 2 r R}\left(p\,(-Q + r)\,R + P r\,(-q + R) + \sqrt{(P r\,(q - R) + p\,(Q - r)\,R)^2 - 4\,(p Q - P r)\,(-P q + p R)\,(q Q - r R)}\right)\right\},\ \left\{-P,\ p - \frac{P q}{R}\right\}\right\}$$

We define the following numerical values for the constants:

```
p=2; q=2;   r=2/3;
P=2; Q=4/3; R=1;
```

The equilibrium points are now

```
equi
```

$$\left\{\{x[t] \to 0,\ y[t] \to 0\},\ \{x[t] \to 3,\ y[t] \to 0\},\ \left\{x[t] \to 1,\ y[t] \to \frac{2}{3}\right\},\ \{y[t] \to 2,\ x[t] \to 0\}\right\}$$

and the eigenvalues are

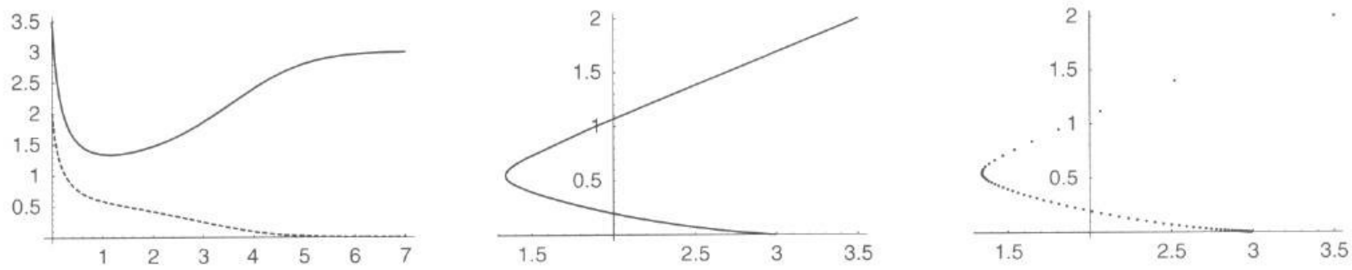
```
eig
```

$$\left\{\{2,\ 2\},\ \{-2,\ -2\},\ \left\{-2,\ \frac{2}{3}\right\},\ \{-2,\ -2\}\right\}$$

Thus, the equilibrium points are an unstable node, a stable node, a saddle point, and a stable node, respectively. We plot one solution to the system in three ways:

```
sol = NDSolve[eq /. {x0->3.5, y0->2}, {x[t],y[t]}, {t,0,7}];

Block[{$DisplayFunction = Identity},
 g1 = Plot[Evaluate[{x[t],y[t]}/.sol], {t,0,7},
   PlotStyle->{{},Dashing[{0.01}]}];
 g2 = ParametricPlot[Evaluate[{x[t],y[t]}/.sol], {t,0,7},
   PlotRange->All];
 g3 = ListPlot[Table[{x[t],y[t]}/.sol[[1]], {t,0,7,0.1}],
   PlotRange->All]];
```

```
Show[GraphicsArray[{g1,g2,g3}]];
```



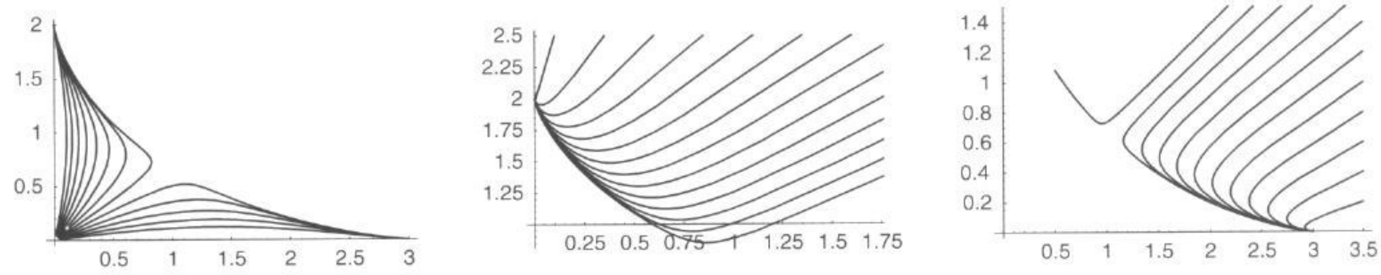
The first figure shows the development of the populations as functions of time. We see that both populations first become smaller, but then, at about time one population **x** begins to win the race. Population **y** dies out in about six time units, and population **x** approaches the level three. The second figure shows the trajectory in the (**x**, **y**) plane. The third figure shows the trajectory sampled at equally spaced time instants. This figure gives an impression of the speed of motion in the plane (the speed can be seen from the first figure, too, but not from the second figure; see also Example 3).

A collection of phase trajectories is very illustrative. We calculate solutions starting from the lines $y = 0.06 - x$, $y = 2.5$, and $x = 3.5$:

```
s1 = Table[NDSolve[eq /. y0->0.06-x0, {x[t],y[t]}, {t,0,7}],
  {x0,0.01,0.052,0.0035}];
s2 = Table[NDSolve[eq /. y0->2.5,     {x[t],y[t]}, {t,0,5}],
  {x0,0.1,3.35,0.25}];
s3 = Table[NDSolve[eq /. x0->3.5,     {x[t],y[t]}, {t,0,5}],
  {y0,0.2,2.4,0.2}];
```

Here are the corresponding trajectories:

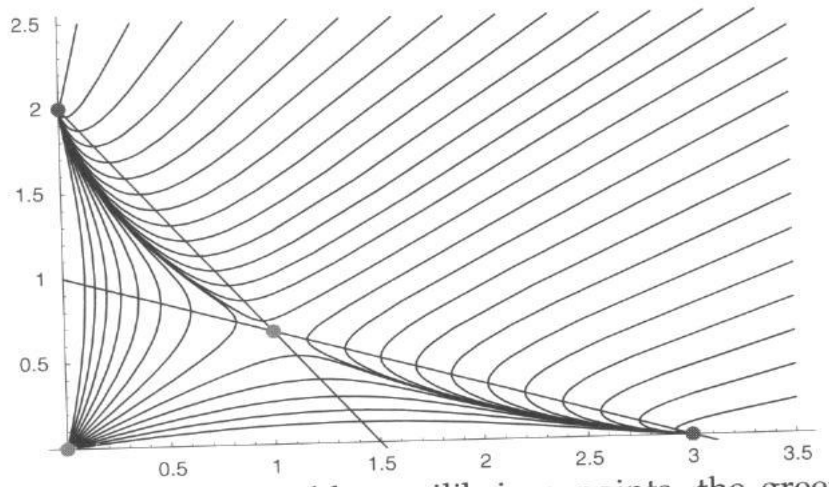
```
Block[{$DisplayFunction = Identity},
  g1 = ParametricPlot[Evaluate[{x[t],y[t]}/.s1], {t,0,7},
    Compiled->False];
  g2 = ParametricPlot[Evaluate[{x[t],y[t]}/.s2], {t,0,5},
    Compiled->False];
  g3 = ParametricPlot[Evaluate[{x[t],y[t]}/.s3], {t,0,5},
    Compiled->False]];

Show[GraphicsArray[{g1,g2,g3}]];
```



We plot the curves where the derivatives are zero and then show all the plots in one figure:

```
g4 = Plot[{h1,h2}, {x,0,3.5}, PlotStyle->RGBColor[0,0,1],
  DisplayFunction->Identity];
```

```
Show[g1,g2,g3,g4, PlotRange->{-0.04,2.55}, Epilog->
  {PointSize[0.02], RGBColor[1,0,0], Point[{3,0}], Point[{0,2}],
  RGBColor[0,1,0], Point[{0,0}], Point[{1,2/3}]}];
```



The red points are the stable equilibrium points, the green points are unstable equilibrium points, and the blue lines are the lines where **x'[t]** or **y'[t]** is zero. The direction of motion is toward the equilibrium points (3, 0) and (0, 2). (Note that a considerable amount of experimenting was needed to prepare this example. To get a figure where the curves are somewhat "uniformly distributed" in the plotting area, experimentation is needed to obtain good starting points **{x0,y0}** for the curves.)

```
Remove["Global`*"]
```

■ Example 2: A Predator–Prey Model

Let us first define

```
f = x[t] (p (1 - x[t]) - r y[t]/(x[t] + q))
```

$$x[t]\left(p\,(1-x[t])-\frac{r\,y[t]}{q+x[t]}\right)$$

```
g = y[t] (1 - y[t]/x[t])
```

$$y[t]\left(1-\frac{y[t]}{x[t]}\right)$$

and then the model

```
eq = {x'[t] == f, y'[t] == g, x[0] == x0, y[0] == y0};
```

This model describes a predator–prey model; see Mesterton-Gibbons (1989, p. 389). The variables **x[t]** and **y[t]** are the magnitudes of the prey and predator populations at time **t**. First we calculate the curves where the derivatives **x'[t]** and **y'[t]** are zero:

```
h1 = y[t] /. Solve[f==0, y[t]][[1]] /. x[t]->x
```

$$-\frac{p\,(-1+x)\,(q+x)}{r}$$

```
h2 = y[t] /. Solve[g==0, y[t]][[2]] /. x[t]->x
```

```
x
```

Then we calculate the equilibrium points:

```
equi = Solve[{f==0, g==0}, {x[t],y[t]}] //Simplify
```

$$\left\{\{y[t]\to 0,\ x[t]\to 1\},\ \left\{y[t]\to -\frac{p\,(-1+q)+r+\sqrt{4\,p^2\,q+(p\,(-1+q)+r)^2}}{2\,p},\right.\right.$$
$$\left. x[t]\to -\frac{p\,(-1+q)+r+\sqrt{4\,p^2\,q+(p\,(-1+q)+r)^2}}{2\,p}\right\},$$
$$\left\{y[t]\to \frac{p-p\,q-r+\sqrt{4\,p^2\,q+(p\,(-1+q)+r)^2}}{2\,p},\right.$$
$$\left.\left. x[t]\to \frac{p-p\,q-r+\sqrt{4\,p^2\,q+(p\,(-1+q)+r)^2}}{2\,p}\right\}\right\}$$

The Jacobian of **f** and **g** is

```
jacobi[f_List,x_List] := Outer[D,f,x]
jac = jacobi[{f,g}, {x[t],y[t]}] //Simplify
```

$$\left\{\left\{p-\frac{r\,y[t]}{q+x[t]}+x[t]\left(-2\,p+\frac{r\,y[t]}{(q+x[t])^2}\right),\ -\frac{r\,x[t]}{q+x[t]}\right\},\ \left\{\frac{y[t]^2}{x[t]^2},\ 1-\frac{2\,y[t]}{x[t]}\right\}\right\}$$

The eigenvalues of the Jacobian at the equilibrium points are

```
eig = Map[Eigenvalues[jac/.#]&, equi];
```

We define the following numerical values for the constants:

```
p=5; q=0.15; r=10;
```

The equilibrium points are now

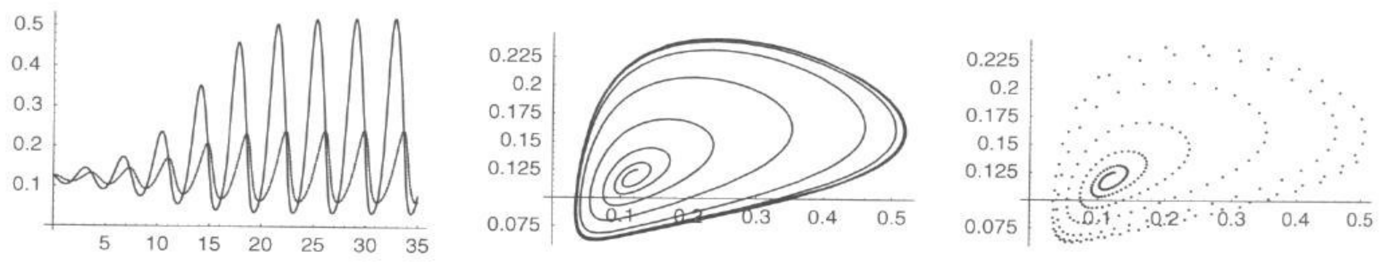
```
equi
{{y[t] → 0, x[t] → 1}, {y[t] → -1.26827, x[t] → -1.26827},
 {y[t] → 0.118271, x[t] → 0.118271}}
```

and the eigenvalues are

```
eig
{{1, -5}, {-0.42213, 18.6261}, {0.176129 - 1.73936 I, 0.176129 + 1.73936 I}}
```

Thus, the equilibrium points are a saddle point, a saddle point, and an unstable focus, respectively. Here is one solution to the system, illustrated in three ways:

```
sol1 = NDSolve[eq /. {x0->0.125, y0->0.125}, {x[t],y[t]}, {t,0,35}];
Block[{$DisplayFunction = Identity},
 g1 = Plot[Evaluate[{x[t],y[t]} /. sol1],{t,0,35},
   PlotStyle->{{}, Dashing[{0.005}]}];
 g2 = ParametricPlot[Evaluate[{x[t],y[t]} /. sol1], {t,0,30}];
 g3 = ListPlot[Table[{x[t],y[t]} /. sol1[[1]], {t,0,25,0.1}]]];
Show[GraphicsArray[{g1,g2,g3}]];
```



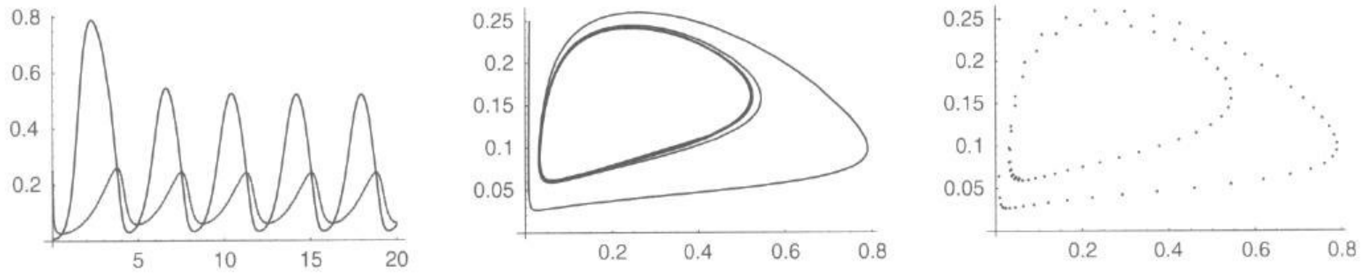
After an initial period of about 20 time units, the populations oscillate regularly. The trajectory in the second figure seems to approach a cycle.

Next we start outside the cycle:

```
sol2 = NDSolve[eq /. {x0->0.01, y0->0.25}, {x[t],y[t]}, {t,0,25}];
```

```
Block[{$DisplayFunction = Identity},
 g4 = Plot[Evaluate[{x[t],y[t]} /. sol2],{t,0,20},
   PlotStyle->{{},Dashing[{0.005}]}];
 g5 = ParametricPlot[Evaluate[{x[t],y[t]} /. sol2], {t,0,25}];
 g6 = ListPlot[Table[{x[t],y[t]} /. sol2[[1]], {t,0,8.7,0.1}]]];
```

```
Show[GraphicsArray[{g4,g5,g6}]];
```

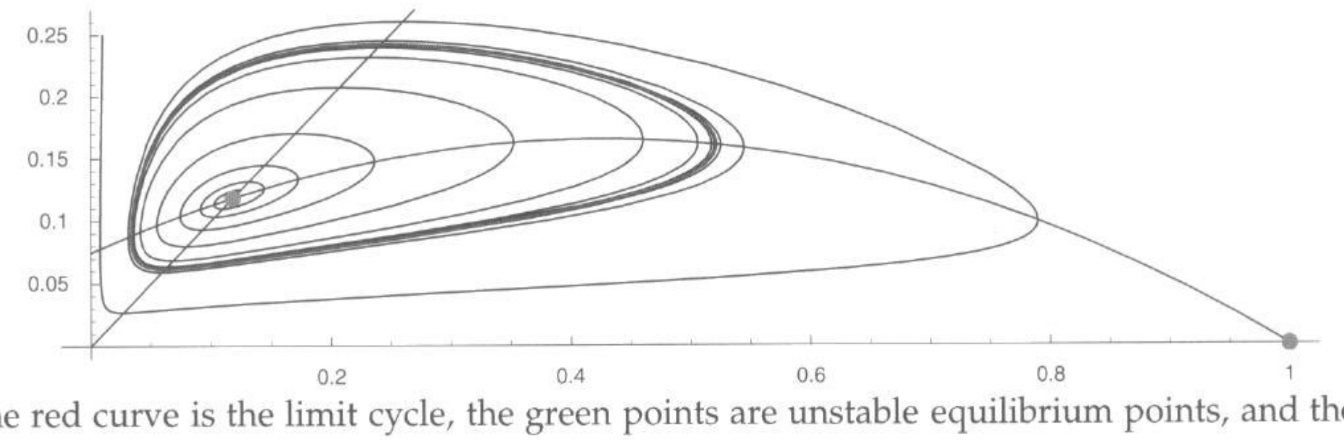


After an initial period of about five time units, the populations oscillate regularly. Again, the trajectory seems to approach a cycle (apparently the same as before).

We make separate plots for the limit cycle and the curves where the derivatives are zero. Then we combine the trajectories:

```
Block[{$DisplayFunction = Identity},
  g7 = ParametricPlot[Evaluate[{x[t],y[t]}/.sol2], {t,21,25},
    PlotStyle->RGBColor[1,0,0]];
  g8 = Plot[Evaluate[{h1,h2}], {x,0,1}, PlotStyle->RGBColor[0,0,1]]];
```

```
Show[g2,g5,g8,g7, PlotRange->{-0.01,0.27}, AspectRatio->Automatic,
  Epilog->{RGBColor[0,1,0], PointSize[0.013],
    Map[Point,{x[t],y[t]}/.equi]}];
```



The red curve is the limit cycle, the green points are unstable equilibrium points, and the blue curves are the lines where **x'[t]** or **y'[t]** is zero. The direction of motion is toward the limit cycle.

```
Remove["Global`*"]
```

■ Example 3: A Model for an Epidemic

The following model describes the development of an epidemic; see Burghes & Borrie (1981, p. 150):

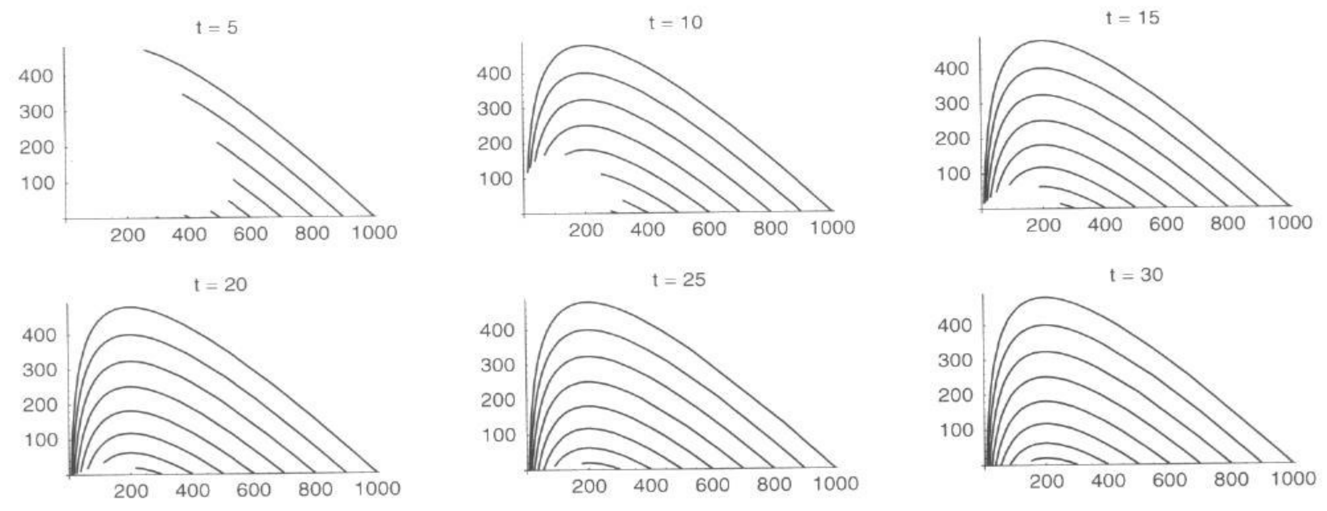
```
eq = {s'[t] == -p s[t] i[t],          s[0] == s0,
      i'[t] == p s[t] i[t] - g i[t], i[0] == i0};
```

The variable **s** is the number of susceptibles (those who are capable of catching the disease) and **i** the number of infectives (those who can transmit the disease). We choose some values for the parameters **g** and **p**, suppose that there is initially only one infective, and calculate a series of solutions where the number of susceptibles is initially 300, 400, ..., 1000:

```
g=0.4; p=0.002; i0=1;
sol = Table[NDSolve[eq/.s0->ss, {s[t],i[t]}, {t,0,30}],
  {ss,300,1000,100}];
```

We plot the solution for time intervals (0, 5), (0, 10), ..., (0, 30):

```
Table[ParametricPlot[Evaluate[{s[t],i[t]}/.sol], {t,0,tt}, Compiled->
  False, PlotLabel->SequenceForm["t = ",tt], PlotRange->
    {{-10,1010},All},
  DisplayFunction->Identity], {tt,5,30,5}];

Show[GraphicsArray[Partition[%,3]]];
```



In this way we can also indicate the speed of the process for trajectories. We can observe that the epidemic develops faster the more susceptibles there are initially. If we have 1000 susceptibles initially, the epidemic is over in about 20 time units, while with 300 initial susceptibles the peak of the epidemic has not even been achieved by that time.

22.3.3 Three Equations

sol = NDSolve[{eq1,eq2,eq3,conds}, {x[t],y[t],z[t]}, {t,a,b}]
Solve the problem

Plot[Evaluate[{x[t],y[t],z[t]}/.sol], {t,a,b}]	Plot one figure
Map[Plot[Evaluate[#[t]/.sol], {t,a,b}]&, {x,y,z}]	Plot three figures

(Continues)