Part IV. Linear Differential Equations

Section 3. Two Dimensional Systems

A second order differential equation in normal form

$$y'' = f(t, y, y')$$

can always be converted into an equivalent system of two first order equations

$$y' = v$$

$$v' = f(t, y, v).$$

If the function f does not depend upon t, then the second order equation (and the system) is called autonomous.

More generally, a two dimensional system is a pair of first order differential equations of the form

$$x' = f(x, y, t)$$

$$y' = g(x, y, t)$$

The primes denote differentiation with respect to t. If neither f nor g depend upon t the system is autonomous. In this section. Maple is used to analyze the solutions to such systems. By now you are familiar with the tools we need: **DEplot** (**DEtools** package), **dsolve**, and the **odeplot** and **display** procedures in the **plots** package.

```
> with(plots): with(DEtools):
```

```
Warning, the name changecoords has been redefined
```

The solution to a two dimensional system is a pair of functions $x(t)$ and $y(t)$. Solution behavior is analyzed using time series and plots of phase plane trajectories of the form $(x(t), y(t))$ when the system is autonomous. State space trajectories $(x(t), y(t), t)$ can be used when the system is not. We consider the autonomous case first

$$x' = f(x, y)$$

$$y' = g(x, y)$$

See Ledder, Chapter 5, Sections 3 and 4.

Fields and flows

The **DEplot** procedure will output time series and trajectories for a two dimensional system. If the system is autonomous, it will also sketch a direction field. Consider the following example corresponding to the Lotka-Volterra predator prey model (Ledder, page 306, Example 2).

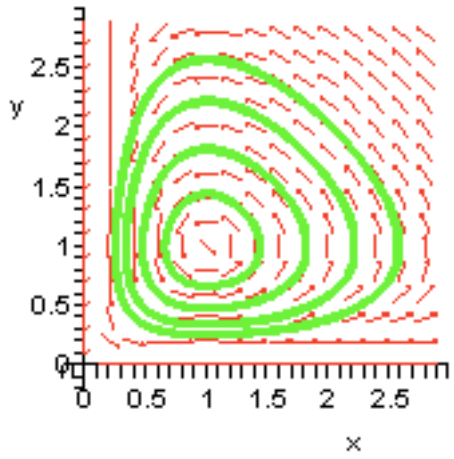
```
> desys := diff(x(t),t) = x(t)*(1 - y(t)),
           diff(y(t),t) = y(t)*(x(t) - 1);
```

$$desys := \frac{d}{dt}x(t) = x(t)\,(1 - y(t)),\ \frac{d}{dt}y(t) = y(t)\,(x(t) - 1)$$

The system is autonomous. **DEplot** displays the direction field and several trajectories. The plot is given a name so it can be displayed again later.

```
> inits := { seq([x(0)=k,y(0)=k], k=[0.4, 0.6, 1.3, 1.8]) }:
  DEplot( {desys}, [x(t),y(t)], t=-40..40, x=0..2.8, y=0..2.8,
          dirgrid=[15,15], inits, stepsize=0.1, linecolor=green);
```

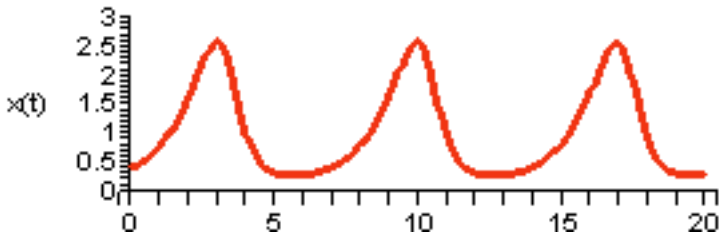
```
flow := %:
```



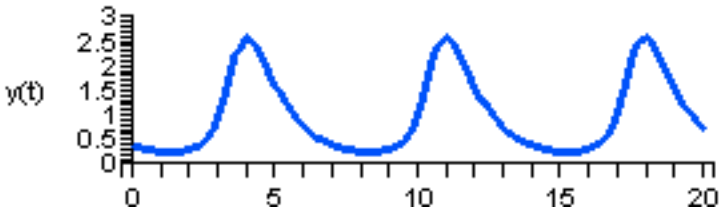
Ledder displays a similar picture on page 306.

The outer trajectory corresponds to the following two time series. The **scene** equation determines which series is plotted.

```
> DEplot( {desys}, [x(t),y(t)], t=0..20, scene=[t,x(t)], view=0..3,
          {[x(0)=0.4,y(0)=0.4]}, stepsize=0.1, linecolor=red );
```

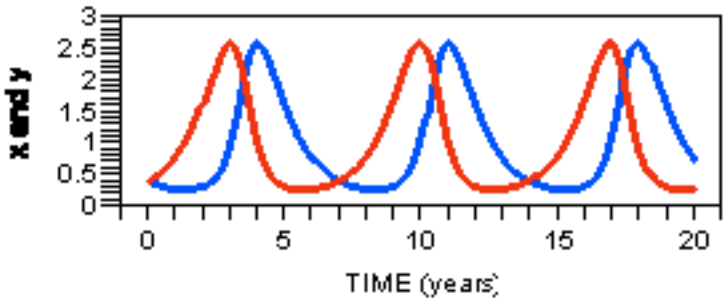



```
> DEplot( {desys}, [x(t),y(t)], t=0..20, scene=[t,y(t)], view=0..3,
          {[x(0)=0.4,y(0)=0.4]}, stepsize=0.1, linecolor=blue );
```



In the next entry the display procedure is used to show the two time series on the same set of axes. We get fancy with the axis labels using designer labels and label directions.

```
> display( %, %% , axes=boxed ,labels=["TIME (years)","x and y"],
                   labeldirections=[horizontal,vertical]);
```



The symmetry in the plots reflect the symmetry in the equations.

Numeric output: dsolve/numeric

If numeric information is required, use **dsolve/numeric**. For example, examination of the time series shows that the oscillations begin to repeat near $t = 7$. A better estimate can be obtained from an array of x, y values for t values near 7. The next entry shows how to make an array that is easy to read.

```
> dsolve( {desys,x(0)=0.4,y(0)=0.4}, [x(t),y(t)], numeric,
          output=array([6.95+0.01*k$k=0..5])): evalf[4](%);
```

$$\begin{bmatrix} [t, x(t), y(t)] \\ \begin{bmatrix} 6.950 & 0.3972 & 0.4028 \\ 6.960 & 0.3996 & 0.4004 \\ 6.970 & 0.4020 & 0.3980 \\ 6.980 & 0.4044 & 0.3957 \\ 6.990 & 0.4069 & 0.3933 \\ 7. & 0.4093 & 0.3910 \end{bmatrix} \end{bmatrix}$$

To the nearest hundredth, the period is 6.96 years.

Nullclines, critical points

The nullclines for an autonomous system are the curves determined by the equations

$$f(x, y) = 0$$

$$g(x, y) = 0$$

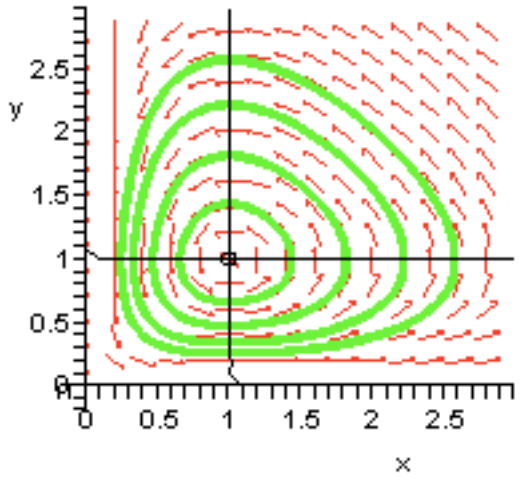
The points where the curves cross are called the critical points or the stationary points of the system.

The **impicitplot** procedure can draw the nullclines. The critical point in the first quadrant is also drawn using the **pointplot** procedure. These plots are given names (output is suppressed).

```
> nullclines := implicitplot( [x*(1-y), y*(x -1)], x=0..3, y=0..3,
                              color=black):
  stationary := pointplot( [1,1], symbol=circle, symbolsize=16,
                           color=black):
```

The display procedure is used to show the flow, the nullclines and the stationary point.

```
> display( flow, nullclines, stationary );
```



Symbolic solutions, integral curves

The Lotka-Volterra system is non-linear. Maple cannot obtain a symbolic solution for the trajectories. However, formulas for curves determined by the trajectories can be obtained by eliminating the t variable to make a first order differential equation for the y variable as a function of the x variable. Solutions to this differential equation are called "integrals" of the system, their graphs are called integral curves. Integral curves for Lotka-Voltera can easily be found "by hand" (Ledder, page 306). Maple's computations are shown in the next few entries.

Create DE:

```
> DE := diff(y(x),x) = rhs(desys[2])/rhs(desys[1]);
```

$$DE := \frac{d}{dx} y(x) = \frac{y(t)\,(x(t) - 1)}{x(t)\,(1 - y(t))}$$

Change $y(t)$ into $y(x)$ and $x(t)$ into x via substitution.

```
> DE := subs({y(t)=y(x),x(t)=x},DE);
```

$$DE := \frac{d}{dx} y(x) = \frac{y(x)\,(x - 1)}{x\,(1 - y(x))}$$

Solve DE implicitly just to make sure dsolve is up to it.

```
> dsolve( DE, implicit );
```

$$x - \ln(x) + y(x) - \ln(y(x)) + \_C1 = 0$$

Now obtain the formula for the implicit solution satisfying y(0.4) = 0.4.

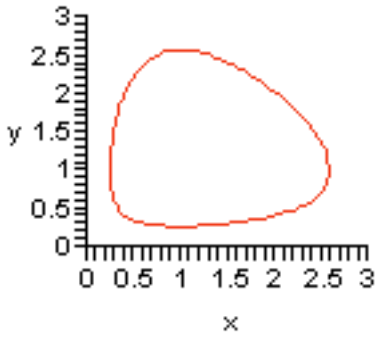
```
> soln := dsolve( {DE,y(0.4)=0.4} , implicit );
```

$$soln := x - \ln(x) + y(x) - \ln(y(x)) - \frac{4}{5} + 2\ln\left(\frac{2}{5}\right) = 0$$

To plot the curve determined by the implicit solution formula we create an x, y equation for the curve and then plot it using impicitplot.

```
> integral_curve := subs(y(x)=y,soln);
  implicitplot( integral_curve, x=0..3, y=0..3, view=[0..3,0..3] );
```

$$integral\_curve := x - \ln(x) + y - \ln(y) - \frac{4}{5} + 2\ln\left(\frac{2}{5}\right) = 0$$



An integral curve displays the path of trajectory, but gives no information about time.

Linear systems, linear flows

Two dimensional linear systems have the form

$$x' = a\,x + b\,y + f$$
$$y' = c\,x + d\,y + g$$

where x, y, a, b, c, d, f, and g are all functions of t. The functions f and g are sometimes referred to as the forcing functions. If they are both zero, the system is called homogeneous (or unforced).

When a, b, c, and d are constants, symbolic solutions can be always be found, provided the functions f and g are elementary (combinations of polynomials, sine, cosine, exp, etc.). Such systems are important not only because they arise in many applications, but also because they are used to approximate non-linear systems in regions near their critical points.

Example Consider the autonomous linear system

$$x' = x - 2\,y + 3$$
$$y' = x + y - 2$$

Sketch the flow, nullclines and stationary point. Find symbolic solutions.

```
> desys := diff(x(t),t) = x(t) - 2*y(t) + 3,
           diff(y(t),t) = x(t) + y(t) - 2;
```

$$desys := \frac{d}{dt}x(t) = x(t) - 2\,y(t) + 3,\ \frac{d}{dt}y(t) = x(t) + y(t) - 2$$

Let's find the stationary point first. The **solve** procedure can be used. Note that the equations and the unknowns must be enclosed in set brackets. The output is a set, or sets, of equations.

```
> stationary := solve( {x-2*y+3=0,x+y-2=0}, {x,y} );
```

$$stationary := \left\{ y = \frac{5}{3}, x = \frac{1}{3} \right\}$$

The solutions can be used to make a "point" as follows.

```
> stat_pt := subs(stationary,[x,y]);
```

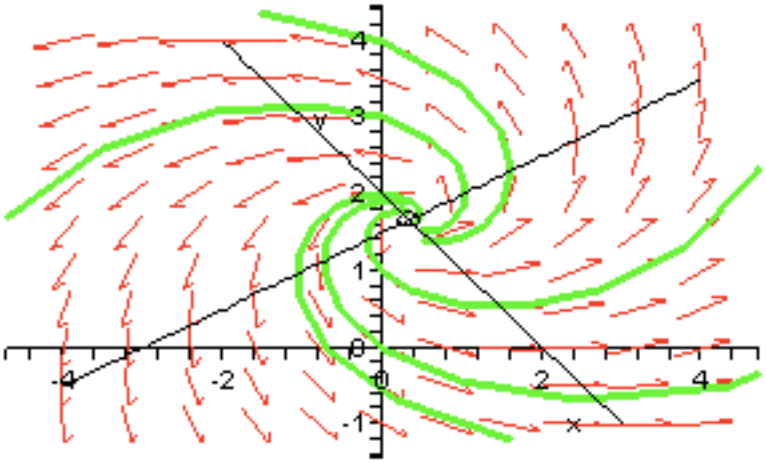
$$stat\_pt := \left[\frac{1}{3}, \frac{5}{3} \right]$$

The next three entries create the plots for flow, nullclines, and the stationary point. Output suppressed.

```
> flow := DEplot( {desys}, [x(t),y(t)], t=-2..3, x=-4..4, y=-1..4,
                  {[x(0)=0,y(0)=k]$k=-0..4}, dirgrid=[11,11],
                  linecolor=green):
  nullclines := implicitplot( [x-2*y+3=0,x+y-2=0], x=-4..4, y=-1..4,
                  color=black):
  stationary_point := pointplot( stat_pt, color=black, symbol=circle,
                                 symbolsize=14):
```

The display procedure displays the three plots together.

```
> display( flow, nullclines, stationary_point );
```



This kind of outward spiraling autonomous flow is called a "source". Phase portraits for constant coefficient autonomous linear systems are often referred to as "linear flows".

Symbolic solution formulas for a source are given in terms of sines, cosines, and exponential functions.

```
> soln_formulas := dsolve( {desys} );
```

$$soln\_formulas := \left\{ y(t) = \frac{5}{3} - \frac{1}{2} \mathbf{e}^{t} \sqrt{2} \, (\_C2 \cos(\sqrt{2}\, t) - \_C1 \sin(\sqrt{2}\, t)), \right.$$
$$\left. x(t) = \frac{1}{3} + \mathbf{e}^{t} (\sin(\sqrt{2}\, t)\, \_C2 + \cos(\sqrt{2}\, t)\, \_C1) \right\}$$

These formulas are easier to understand if displayed in column vector form.

```
> <x(t),y(t)> = subs(soln_formulas,<x(t),y(t)>);
```

$$\begin{bmatrix} x(t) \\ y(t) \end{bmatrix} = \begin{bmatrix} \frac{1}{3} + \mathbf{e}^{t} (\sin(\sqrt{2}\ t)\ \_C2 + \cos(\sqrt{2}\ t)\ \_C1) \\ \frac{5}{3} - \frac{1}{2} \mathbf{e}^{t} \sqrt{2}\ (\_C2 \cos(\sqrt{2}\ t) - \_C1 \sin(\sqrt{2}\ t)) \end{bmatrix}$$

Observe that the "stationary solution" is obtained by setting the constants equal to 0. Linear flows are discussed in more detail in the next section.