Demonstration of Maple for ODEs

DEMONSTRATION OF MAPLE FOR ODES (`intro-odes.ms`)

It is a good habit to remember to use `restart` when starting a new problem.

```
> restart;
```

Another good habit is to load the `plots` and `DEtools` packages at the top of all worksheets, just in case they will be needed at some point in the analysis.

```
> with( plots );
```

[*animate*, *animate3d*, *conformal*, *contourplot*, *cylinderplot*, *densityplot*, *display*, *display3d*, *fieldplot*, *fieldplot3d*, *gradplot*, *gradplot3d*, *implicitplot*, *implicitplot3d*, *loglogplot*, *logplot*, *matrixplot*, *odeplot*, *pointplot*, *polarplot*, *polygonplot*, *polygonplot3d*, *polyhedraplot*, *replot*, *setoptions*, *setoptions3d*, *spacecurve*, *sparsematrixplot*, *sphereplot*, *surfdata*, *textplot*, *textplot3d*, *tubeplot*]

```
> with( DEtools );
```

[*DEplot*, *DEplot1*, *DEplot2*, *Dchangevar*, *PDEplot*, *dfieldplot*, *phaseportrait*]

Direction Fields and Graphical Solutions

Many introductory courses begin by trying to develop the student's understanding about what a differential equation is, what it means for a function to solve an ODE, and how to perform some analysis directly from the differential equation. Graphical methods are commonly employed in these discussions. The Maple command `DEplot`, from the `DEtools` package, provides a comprehensive interface for most graphical needs.

To begin, consider the (linear) differential equation

```
> ODE := diff( y(x), x ) = x^2 - y(x);
```

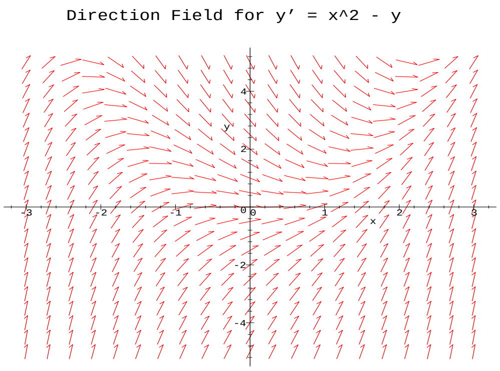
$$ODE := \frac{\partial}{\partial x}\,\mathrm{y}(\,x\,) = x^2 - \mathrm{y}(\,x\,)$$

Note that this is the first example used in Section 1.3 of Nagle and Saff. The next few commands reproduce a few of the figures displayed on p. 17.

The first three arguments to `DEplot` must provide, in order, a single n-th order ODE or a system of n first-order ODEs, a list indicating the names of the independent variable (first) and all dependent variables, and a range for the independent variable. The specific content of the plot is determined by all subsequent arguments to `DEplot`.

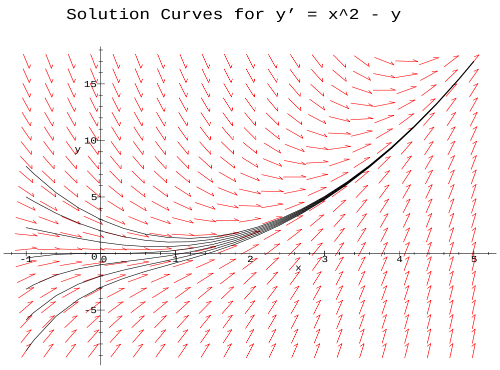
The most basic use of `DEplot` is to display a direction field for a differential equation. A request for a direction field is made by specifying both a range for the independent variable and the `arrows=` option.

```
> DEplot( ODE, [ x, y ], x=-3..3, y=-5..5, arrows=THIN,
>         title=`Direction Field for y' = x^2 - y` );
```



The solution curves through specified points are produced when the fourth argument is a set of initial conditions. If the `arrows=` argument is included, the curves are superimposed over the arrows; if this argument is omitted, only the solution curves are displayed.

```
> DEplot( ODE, [ x, y ], x=-1..5,
>         { [0, -3], [0, -2], [0, -1], [0, 0], [0, 1], [0, 2], [0, 3] },
>         arrows=THIN, title=`Solution Curves for y' = x^2 - y` );
```



Only slight alterations are needed for higher-order equations, or for systems. To illustrate, consider the predator-prey system

```
> SYS := [ diff( P(t), t ) = p(t)*P(t) - P(t),
>          diff( p(t), t ) = 2*p(t) - 2*P(t)*p(t) ];
```

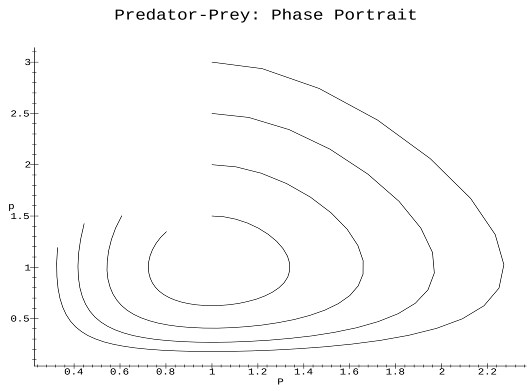
$$SYS := \left[\frac{\partial}{\partial t} \mathrm{P}(t) = \mathrm{p}(t)\,\mathrm{P}(t) - \mathrm{P}(t), \frac{\partial}{\partial t} \mathrm{p}(t) = 2\,\mathrm{p}(t) - 2\,\mathrm{p}(t)\,\mathrm{P}(t) \right]$$

where P denotes the size of the predator population and p is the prey.

A sampling of solution curves in the phase-space can be created by specifying a set of initial conditions[1] and the desired scene; `stepsize=0.1` is included because the default stepsize produces a rather rough graph.

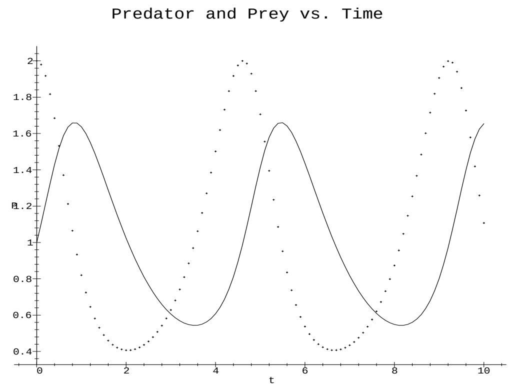
[1] The use of **$** to generate the initial data can be a real time-saver. If this is new to you, please see **?$**

```
> DEplot( SYS, [t,P,p], 0..4, { [0,1,k/2] $ k=2..6 }, scene=[P,p],
>          stepsize=0.1, title=`Predator-Prey: Phase Portrait` );
```



Plots of the individual solutions can be obtained by changing the scene. Here, individual plots of the predator and prey are created, then displayed in a single plot using the `display` command from the `plots` package. Note the use of different styles to identify the different curves in the plot.

```
> plotp := DEplot( SYS, [t,P,p], 0..10, { [0,1,2] }, scene=[t,p],
>                  stepsize=0.1, style=POINT ):
> plotP := DEplot( SYS, [t,P,p], 0..10, { [0,1,2] }, scene=[t,P],
>                  stepsize=0.1, style=LINE ):
> display( { plotp, plotP }, title=`Predator and Prey vs. Time` );
```



This is only the beginning of what can be done with `DEplot` and the `DEtools` package. Please consult the on-line help for additional information for a complete description of the available options and examples.

```
> ?DEtools
> ?DEtools[DEplot]
```

for an explanation and additional examples.

```
> ?DEtools[options]
```

Symbolic Solutions to ODEs and IVPs

While Maple is quite happy to accept the inputs to its commands in almost any form, it is practically, esthetically, and pedagogically preferable to use descriptive names for the individual parts of a problem (*e.g.*, the differential equation and boundary condition). To illustrate, let's find the general solution to $x''+4x'+4x = 2te^{-2t}$. The differential equation can be specified as

```
> ODE := diff( x(t), t$2 ) + 4*diff( x(t), t ) + 4*x(t) = 2*t*exp(-2*t);
```

$$ODE := \left(\frac{\partial^2}{\partial t^2}\mathrm{x}(t)\right) + 4\left(\frac{\partial}{\partial t}\mathrm{x}(t)\right) + 4\,\mathrm{x}(t) = 2\,t\,\mathrm{e}^{(-2\,t)}$$

and the general solution is found using `dsolve`:

```
> GSOLN := dsolve( ODE, x(t) );
```

$$GSOLN := \mathrm{x}(t) = \frac{1}{3}t^3\,\mathrm{e}^{(-2\,t)} + \_C1\,\mathrm{e}^{(-2\,t)} + \_C2\,t\,\mathrm{e}^{(-2\,t)}$$

Note that Maple introduces the constants `_C1` and `_C2` as arbitrary constants.

The solution to an initial value problem can be obtained by including initial conditions in a set containing the differential equation. For example, the initial conditions $x(1) = 0$, $x'(1) = 1$ can be implemented as

```
> IC := x(1)=0, D(x)(1)=1;
```

$$IC := \mathrm{x}(1) = 0, \mathrm{D}(x)(1) = 1$$

Note the use of `D` in the specification of the derivative condition (higher order derivative conditions can be specified using `@@`, Maple's composition operator, *e.g.*, `(D@@2)(x)(0)=3`). Then, the initial value problem is

```
> IVP := { ODE, IC };
```

$$IVP := \left\{\left(\frac{\partial^2}{\partial t^2}\mathrm{x}(t)\right) + 4\left(\frac{\partial}{\partial t}\mathrm{x}(t)\right) + 4\,\mathrm{x}(t) = 2\,t\,\mathrm{e}^{(-2\,t)}, \mathrm{D}(x)(1) = 1, \mathrm{x}(1) = 0\right\}$$

and its solution is found using `dsolve` as follows:

```
> SOLN := dsolve( IVP, x(t) );
```

$$SOLN := \mathrm{x}(t) = \frac{1}{3}t^3\,\mathrm{e}^{(-2\,t)} + \frac{1}{3}\,\frac{(2\,\mathrm{e}^{(-2)} - 3)\,\mathrm{e}^{(-2\,t)}}{\mathrm{e}^{(-2)}} - \frac{(\mathrm{e}^{(-2)} - 1)\,t\,\mathrm{e}^{(-2\,t)}}{\mathrm{e}^{(-2)}}$$

Expressions and Functions

Notice how the output from `dsolve` is a Maple equation (the reason for this choice will be apparent when considering the solution to a system of differential equations). While there are specific cases where these equations are useful, most circumstances call for either an expression or a function. The key is to understand the structure of the output from `dsolve`.

Let's illustrate with the same example as above, with initial conditions specified so that the constants refer to the values of the solution and its first derivative at $t = 1$.

```
> IC := { x(1)=c1, D(x)(1)=c2 };
```

$$IC := \{\mathrm{D}(x)(1) = c2, \mathrm{x}(1) = c1\}$$

Note that these initial conditions are specified as a set, not an expression sequence. Thus, different manipulations are required to express the final IVP as a set.

```
> IVP := { ODE } union IC;
```

$$IVP := \left\{ \left(\frac{\partial^2}{\partial t^2}\,\mathrm{x}(t)\right) + 4\left(\frac{\partial}{\partial t}\,\mathrm{x}(t)\right) + 4\,\mathrm{x}(t) = 2\,t\,\mathrm{e}^{(-2\,t)}, \mathrm{D}(x)(1) = c2, \mathrm{x}(1) = c1 \right\}$$

It is easiest to obtain the solution as an expression; it's just the right-hand side of the object MAPLE returns from `dsolve`.

```
> X := rhs( dsolve( IVP, x(t) ) );
```

$$X := \frac{1}{3}\,t^3\,\mathrm{e}^{(-2\,t)} + \frac{1}{3}\,\frac{(2\,\mathrm{e}^{(-2)} - 3\,c1 - 3\,c2)\,\mathrm{e}^{(-2\,t)}}{\mathrm{e}^{(-2)}} - \frac{(\mathrm{e}^{(-2)} - 2\,c1 - c2)\,t\,\mathrm{e}^{(-2\,t)}}{\mathrm{e}^{(-2)}}$$

When a Maple *function* is desired, it is most expedient to use `unapply`[2]

```
> XX := unapply( rhs( dsolve( IVP, x(t) ) ), t );
```

$$XX := t \to \frac{1}{3}\,t^3\,\mathrm{e}^{(-2\,t)} + \frac{1}{3}\,\frac{(2\,\mathrm{e}^{(-2)} - 3\,c1 - 3\,c2)\,\mathrm{e}^{(-2\,t)}}{\mathrm{e}^{(-2)}} - \frac{(\mathrm{e}^{(-2)} - 2\,c1 - c2)\,t\,\mathrm{e}^{(-2\,t)}}{\mathrm{e}^{(-2)}}$$

To illustrate a potential advantage of the use of `unapply`, consider a situation in which you wish to include the constants as arguments to the function, *e.g.*, $x(t; c_1, c_2)$, where c_1 and c_2 are the values of the function and its first derivative at $t = 1$. Then you would use

```
> XXX := unapply( rhs( dsolve( IVP, x(t) ) ), t, c1, c2 );
```

$$XXX := (t, c1, c2) \to$$
$$\frac{1}{3}\,t^3\,\mathrm{e}^{(-2\,t)} + \frac{1}{3}\,\frac{(2\,\mathrm{e}^{(-2)} - 3\,c1 - 3\,c2)\,\mathrm{e}^{(-2\,t)}}{\mathrm{e}^{(-2)}} - \frac{(\mathrm{e}^{(-2)} - 2\,c1 - c2)\,t\,\mathrm{e}^{(-2\,t)}}{\mathrm{e}^{(-2)}}$$

and the resulting function, `XXX`, can be used to find the solution passing through the point $(1, 0)$ with unit slope as follows

```
> XXX(t,0,1);
```

$$\frac{1}{3}\,t^3\,\mathrm{e}^{(-2\,t)} + \frac{1}{3}\,\frac{(2\,\mathrm{e}^{(-2)} - 3)\,\mathrm{e}^{(-2\,t)}}{\mathrm{e}^{(-2)}} - \frac{(\mathrm{e}^{(-2)} - 1)\,t\,\mathrm{e}^{(-2\,t)}}{\mathrm{e}^{(-2)}}$$

The same computation using `X` and `XX` would appear as:

```
> subs( c1=0, c2=1, X );
```

$$\frac{1}{3}\,t^3\,\mathrm{e}^{(-2\,t)} + \frac{1}{3}\,\frac{(2\,\mathrm{e}^{(-2)} - 3)\,\mathrm{e}^{(-2\,t)}}{\mathrm{e}^{(-2)}} - \frac{(\mathrm{e}^{(-2)} - 1)\,t\,\mathrm{e}^{(-2\,t)}}{\mathrm{e}^{(-2)}}$$

```
> subs( c1=0, c2=1, XX(t) );
```

$$\frac{1}{3}\,t^3\,\mathrm{e}^{(-2\,t)} + \frac{1}{3}\,\frac{(2\,\mathrm{e}^{(-2)} - 3)\,\mathrm{e}^{(-2\,t)}}{\mathrm{e}^{(-2)}} - \frac{(\mathrm{e}^{(-2)} - 1)\,t\,\mathrm{e}^{(-2\,t)}}{\mathrm{e}^{(-2)}}$$

Next, consider the problem of verifying that the solution is correct.

```
> simplify( subs( x(t)=X, ODE ) );
```

$$2\,t\,\mathrm{e}^{(-2\,t)} = 2\,t\,\mathrm{e}^{(-2\,t)}$$

```
> simplify( subs( x(t)=XX(t), ODE ) );
```

$$2\,t\,\mathrm{e}^{(-2\,t)} = 2\,t\,\mathrm{e}^{(-2\,t)}$$

[2] The use of `unapply` can be difficult for students to understand; I prefer to avoid this whenever possible.

```
> simplify( subs( x(t)=XXX(t,c1,c2), ODE ) );
```

$$2\,t\,e^{(-2\,t)} = 2\,t\,e^{(-2\,t)}$$

Note that, because both x and XX are functions of a single variable, it is also possible to use `simplify( subs( x=XX, ODE ) );`. The same is not true for XXX, as this function requires three arguments.

To conclude this discussion, let's plot the solutions that have critical points (*i.e.*, $c2 = 0$) at $(1, -1)$, $(1, 1)$, and $(1, 3)$ using each of X, XX, and XXX:

```
> S1 := { seq( subs( c1=C, c2=0, X ), C={-1,1,3} ) };
```

$$S1 := \left\{\frac{1}{3}\,t^3\,e^{(-2\,t)} + \frac{1}{3}\,\frac{(\,2\,e^{(-2)} - 3\,)\,e^{(-2\,t)}}{e^{(-2)}} - \frac{(\,e^{(-2)} - 2\,)\,t\,e^{(-2\,t)}}{e^{(-2)}},\right.$$
$$\frac{1}{3}\,t^3\,e^{(-2\,t)} + \frac{1}{3}\,\frac{(\,2\,e^{(-2)} + 3\,)\,e^{(-2\,t)}}{e^{(-2)}} - \frac{(\,e^{(-2)} + 2\,)\,t\,e^{(-2\,t)}}{e^{(-2)}},$$
$$\left.\frac{1}{3}\,t^3\,e^{(-2\,t)} + \frac{1}{3}\,\frac{(\,2\,e^{(-2)} - 9\,)\,e^{(-2\,t)}}{e^{(-2)}} - \frac{(\,e^{(-2)} - 6\,)\,t\,e^{(-2\,t)}}{e^{(-2)}}\right\}$$

```
> S2 := { seq( subs( c1=C, c2=0, XX('t') ), C={-1,1,3} ) };
```

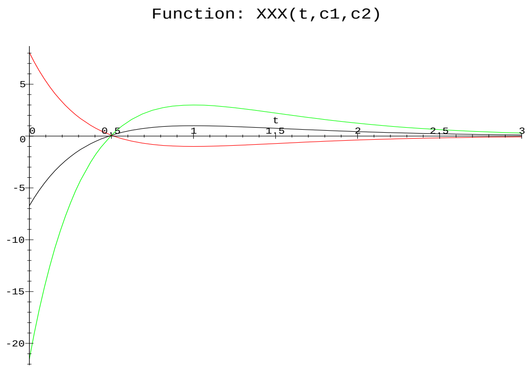
$$S2 := \left\{\frac{1}{3}\,t^3\,e^{(-2\,t)} + \frac{1}{3}\,\frac{(\,2\,e^{(-2)} - 3\,)\,e^{(-2\,t)}}{e^{(-2)}} - \frac{(\,e^{(-2)} - 2\,)\,t\,e^{(-2\,t)}}{e^{(-2)}},\right.$$
$$\frac{1}{3}\,t^3\,e^{(-2\,t)} + \frac{1}{3}\,\frac{(\,2\,e^{(-2)} + 3\,)\,e^{(-2\,t)}}{e^{(-2)}} - \frac{(\,e^{(-2)} + 2\,)\,t\,e^{(-2\,t)}}{e^{(-2)}},$$
$$\left.\frac{1}{3}\,t^3\,e^{(-2\,t)} + \frac{1}{3}\,\frac{(\,2\,e^{(-2)} - 9\,)\,e^{(-2\,t)}}{e^{(-2)}} - \frac{(\,e^{(-2)} - 6\,)\,t\,e^{(-2\,t)}}{e^{(-2)}}\right\}$$

```
> S3 := { seq( XXX('t',C,0), C={-1,1,3} ) };
```

$$S3 := \left\{\frac{1}{3}\,t^3\,e^{(-2\,t)} + \frac{1}{3}\,\frac{(\,2\,e^{(-2)} - 3\,)\,e^{(-2\,t)}}{e^{(-2)}} - \frac{(\,e^{(-2)} - 2\,)\,t\,e^{(-2\,t)}}{e^{(-2)}},\right.$$
$$\frac{1}{3}\,t^3\,e^{(-2\,t)} + \frac{1}{3}\,\frac{(\,2\,e^{(-2)} + 3\,)\,e^{(-2\,t)}}{e^{(-2)}} - \frac{(\,e^{(-2)} + 2\,)\,t\,e^{(-2\,t)}}{e^{(-2)}},$$
$$\left.\frac{1}{3}\,t^3\,e^{(-2\,t)} + \frac{1}{3}\,\frac{(\,2\,e^{(-2)} - 9\,)\,e^{(-2\,t)}}{e^{(-2)}} - \frac{(\,e^{(-2)} - 6\,)\,t\,e^{(-2\,t)}}{e^{(-2)}}\right\}$$

Note that the use of single quotes (') is needed so that the evaluation of t is delayed until after the `subs` command is completed. The same approach is needed to produce a plot of these solutions.

```
> plot( S3, 't'=0..3, title=`Function: XXX(t,c1,c2)` );
```



The structure of this solution (linear combination of a basis of solutions to the homogeneous equation plus a particular solution) is evident the solutions we have found. The individual components can be easily extracted and manipulated. Here we demonstrate how a general solution is used to determine the solution of a specific initial value problem. Once the general solution is found

```
> GSOLN := rhs( dsolve( ODE, x(t) ) );
```

$$GSOLN := \frac{1}{3}\,t^3\,e^{(-2\,t)} + \_C1\,e^{(-2\,t)} + \_C2\,t\,e^{(-2\,t)}$$

the two equations that must be satisfied by `_C1` and `_C2` can be constructed:

```
> eq1 := subs( t=1, GSOLN=3 );
> eq2 := subs( t=1, diff(GSOLN,t)=0 );
```

$$eq1 := \frac{1}{3}\,e^{(-2)} + \_C1\,e^{(-2)} + \_C2\,e^{(-2)} = 3$$

$$eq2 := \frac{1}{3}\,e^{(-2)} - 2\,\_C1\,e^{(-2)} - \_C2\,e^{(-2)} = 0$$

and solved

```
> solC := solve( { eq1, eq2 }, { _C1, _C2 } );
```

$$solC := \left\{\_C1 = \frac{1}{3}\,\frac{2\,e^{(-2)} - 9}{e^{(-2)}}, \_C2 = -\frac{e^{(-2)} - 6}{e^{(-2)}}\right\}$$

These solutions obviously simplify (but require an application of **expand** to force all simplifications)

```
> solC := simplify(expand( solC ));
```

$$solC := \left\{\_C1 = \frac{2}{3} - 3\,e^2, \_C2 = -1 + 6\,e^2\right\}$$

Floating point approximations to these constants are obtained using **fnormal** or **evalf**:

```
> evalf( solC );
```

$$\{\_C1 = -21.50050163, \_C2 = 43.33433659\}$$

The solution to the IVP is found to be

```
> IVPsoln := factor( subs( solC, GSOLN ) );
```

$$IVPsoln := \frac{1}{3}\, e^{(-2\,t)}\,(\,t^3+2-9\,e^2-3\,t+18\,t\,e^2\,)$$

```
> simplify( IVPsoln - subs( c1=3, c2=0, X ) );
```

$$0$$

```
> simplify( IVPsoln - subs( c1=3, c2=0, XX(t) ) );
```

$$0$$

```
> simplify( IVPsoln - XXX(t,3,0) );
```

$$0$$

To conclude, let's see three different ways in which the terms contributing to the homogeneous and particular solutions can be obtained. Working directly from the general solution:

```
> Xp := subs( _C1=0, _C2=0, GSOLN );
```

$$Xp := \frac{1}{3}\,t^3\,e^{(-2\,t)}$$

```
> Xh := GSOLN - Xp;
```

$$Xh := \_C1\,e^{(-2\,t)} + \_C2\,t\,e^{(-2\,t)}$$

```
> X1 := subs( _C1=1, _C2=0, Xh );
```

$$X1 := e^{(-2\,t)}$$

```
> X2 := subs( _C1=0, _C2=1, Xh );
```

$$X2 := t\,e^{(-2\,t)}$$

Alternatively, the same information can be obtained in a directly from `dsolve`:

```
> ODEh := lhs(ODE)=0;
```

$$ODEh := \left(\frac{\partial^2}{\partial t^2}\,\mathrm{x}(\,t\,)\right) + 4\,\left(\frac{\partial}{\partial t}\,\mathrm{x}(\,t\,)\right) + 4\,\mathrm{x}(\,t\,) = 0$$

```
> IC1 := x(0)=0, D(x)(0)=1;
> X1 := rhs( dsolve( { ODEh, IC1 }, x(t) ) );
```

$$IC1 := \mathrm{x}(\,0\,) = 0, \mathrm{D}(\,x\,)(\,0\,) = 1$$

$$X1 := t\,e^{(-2\,t)}$$

```
> IC2 := x(0)=1, D(x)(0)=0;
> X2 := rhs( dsolve( { ODEh, IC2 }, x(t) ) );
```

$$IC2 := \mathrm{x}(\,0\,) = 1, \mathrm{D}(\,x\,)(\,0\,) = 0$$

$$X2 := e^{(-2\,t)} + 2\,t\,e^{(-2\,t)}$$

```
> ICp := x(0)=0, D(x)(0)=0;
> Xp := rhs( dsolve( { ODE, ICp }, x(t) ) );
```

$$ICp := \mathrm{x}(\,0\,) = 0, \mathrm{D}(\,x\,)(\,0\,) = 0$$

$$Xp := \frac{1}{3}\, t^3\, \mathrm{e}^{(-2\,t)}$$

Although, to be honest, in this case it is probably simplest to explicitly identify the appropriate terms in the general solution and explicitly define the components of the solution.

```
> x1 := exp(-2*t);
> x2 := t*exp(-2*t);
> xp := t^3/3 * exp(-2*t);
```

This example hopefully illustrates that almost all manipulations can be done in Maple, some steps should still be done by hand.

Additional information about `dsolve` is available from the on-line help (`?dsolve`). Note, in particular, that `dsolve` may return an implicit solution or a solution in parametric form. Explicit solutions can, sometimes, be coerced by using the optional argument `explicit=true`. Other optional arguments are discussed later in this supplement.

Systems of ODEs

Systems of ODEs are analyzed in a completely parallel manner. The system of equations is specified as a set (with or without initial conditions); the dependent variables are also specified as a set. To illustrate, consider the (linearized) model of a pendulum: $\theta''(t) + 3\theta(t) = 0$. This second-order equation is equivalent to the system of first-order equations

```
> SYS := { diff( theta(t), t ) = v(t), diff( v(t), t ) = -3*theta(t) };
```

$$SYS := \left\{ \frac{\partial}{\partial t}\,\theta(\,t\,) = \mathrm{v}(\,t\,), \frac{\partial}{\partial t}\,\mathrm{v}(\,t\,) = -3\,\theta(\,t\,) \right\}$$

in terms of the dependent variables

```
> FNS := { theta(t), v(t) };
```

$$FNS := \{\,\theta(\,t\,), \mathrm{v}(\,t\,)\,\}$$

The general solution is found to be

```
> SOL := dsolve( SYS, FNS );
```

$$SOL := \left\{ \theta(\,t\,) = \frac{1}{3}\,\_C2\,\sin\left(\sqrt{3}\,t\right)\,\sqrt{3} + \_C1\,\cos\left(\sqrt{3}\,t\right), \right.$$
$$\left. \mathrm{v}(\,t\,) = -\_C1\,\sin\left(\sqrt{3}\,t\right)\,\sqrt{3} + \_C2\,\cos\left(\sqrt{3}\,t\right) \right\}$$

We are now in a position to understand why it is important that `dsolve` returns (a set of) equations. Because a set is unordered, there is no reason to expect the elements of the solution set to appear in the same order each time Maple is executed. That is, there is no "first" term in the solution vector. There is, nonetheless, a very simple means to ensure that the appropriate term from the solution is extracted. It is still possible to obtain the individual component as either an expression

```
> SOLV := subs( SOL, v(t) );
```

$$SOLV := -\_C1\,\sin\left(\sqrt{3}\,t\right)\,\sqrt{3} + \_C2\,\cos\left(\sqrt{3}\,t\right)$$

or as a function

```
> SOLT := unapply( subs( SOL, theta(t) ), t );
```

$$SOLT := t \to \frac{1}{3}\_C2 \sin\left(\sqrt{3}\,t\right)\sqrt{3} + \_C1 \cos\left(\sqrt{3}\,t\right)$$

Most systems of ODEs do not have explicit solutions. Maple has no difficulties producing series solutions for systems. We illustrate using the corresponding nonlinear pendulum equation

```
> SYSnl := { diff( theta(t), t ) = v(t),
>            diff( v(t), t ) = -3*sin( theta(t) ) };
```

$$SYSnl := \left\{\frac{\partial}{\partial t}\theta(t) = \mathrm{v}(t), \frac{\partial}{\partial t}\mathrm{v}(t) = -3\sin(\theta(t))\right\}$$

Only a small number of terms will be needed to compare with the linearized solution.

```
> Order := 3:
> SOLnl := dsolve( SYSnl, FNS, type=series );
```

$$SOLnl := \left\{\mathrm{v}(t) = \mathrm{v}(0) - 3\sin(\theta(0))\,t - \frac{3}{2}\cos(\theta(0))\,\mathrm{v}(0)\,t^2 + \mathrm{O}(t^3),\right.$$
$$\left.\theta(t) = \theta(0) + \mathrm{v}(0)\,t - \frac{3}{2}\sin(\theta(0))\,t^2 + \mathrm{O}(t^3)\right\}$$

```
> SOLnlV := subs( SOLnl, v(t) );
```

$$SOLnlV := \mathrm{v}(0) - 3\sin(\theta(0))\,t - \frac{3}{2}\cos(\theta(0))\,\mathrm{v}(0)\,t^2 + \mathrm{O}(t^3)$$

Comparisons with the solution to the linearized equation can be performed using the `series` command, as follows:

```
> series( SOLV, t=0 );
```

$$\_C2 - 3\,\_C1\,t - \frac{3}{2}\_C2\,t^2 + \mathrm{O}(t^3)$$

The real power of Maple for systems of ODEs is seen in its handling of numerical solutions. This will be demonstrated in the next section.

Numeric Solutions

Numeric solutions to initial value problems can be used in a variety of ways in an introductory course, including: direction fields, qualitative behavior of solutions, existence theory, regularity theory. The `dsolve` command can be used for each of these topics. The default method used is `rkf45`, a Maple implementation of the Fehlberg fourth-fifth order Runge-Kutta method (see subroutine `RKF45` appearing in COMPUTER METHODS FOR MATHEMATICAL COMPUTATIONS by G.E. Forsythe, M.A. Malcolm, and C.B. Moler); a second algorithm is a seventh-eighth order continuous Runge-Kutta method, `dverk78`.

Notice that there are no built-in Maple procedures for Euler's method, improved Euler's method and other typical numerical methods discussed in an introductory course. These methods are easily implemented in Maple, as will demonstrated in a later section.

To begin our discussion, consider the initial value problem

```
> ODE := diff( x(t), t ) = x(t)^2 + t:
> IC := x(0)=0:
> IVP := { ODE, IC };
```

$$IVP := \left\{ \frac{\partial}{\partial t}\,\mathrm{x}(\,t\,) = \mathrm{x}(\,t\,)^2 + t, \mathrm{x}(\,0\,) = 0 \right\}$$

While this problem does have an explicit solution, it cannot be found using methods typically found in an introductory course. But, it is reasonable to ask, *e.g.*, if a solution does exist for all $t > 0$. Numerical solutions can be used, with caution, to discuss questions of this type.

The inclusion of the optional argument `type=numeric` is the basic interface to the numerical version of `dsolve`.

```
> numSOLN := dsolve( IVP, x(t), type=numeric ):
```

The output from `dsolve` is quite different. What this is saying is that `numSOLN` is a Maple procedure that can be used to compute the solution at a given (numeric) value of t. For example:

```
> numSOLN( 0 );
```

$$[\,t = 0, \mathrm{x}(\,t\,) = 0\,]$$

```
> numSOLN( 1/10 );
```

$$\left[t = \frac{1}{10}, \mathrm{x}(\,t\,) = .005000500074847118\right]$$

Notice that the output from the procedure created by `dsolve` is a list of equations; one equation for the independent variable and one equation for each dependent variable. The structure of this solution is ideal for use with the `subs` command to extract specific components of a solution, or even to use the results in subsequent calculations. But first, let's evaluate the solution at more points

```
> numSOLN( 0.25 );
```

$$[\,t = .25, \mathrm{x}(\,t\,) = .03129892370880233\,]$$

```
> numSOLN( 1 );
```

$$[\,t = 1, \mathrm{x}(\,t\,) = .5571617678870113\,]$$

```
> numSOLN( 0 );
```

$$[\,t = 0, \mathrm{x}(\,t\,) = .1498570504840369\,10^{-7}\,]$$

```
> numSOLN( 1 );
```

$$[\,t = 1, \mathrm{x}(\,t\,) = .5571617926830499\,]$$

These last two results are a little distressing – why have the values of the solution changed from the first time the solution was evaluated? To explain this, it is necessary to realize that each call to `numSOLN` uses the results of the most recent computations as the initial conditions for the current call. Thus, the first evaluation of the solution at $t = 0$ required no computations as this is the initial condition specified in the original problem. The second evaluation at $t = 0$ integrates the equation from $t = 1$ back to $t = 0$; the difference between the two values gives an indication of the accumulation error for this solution. To reset the solution to the original initial condition, simply re-execute the `dsolve` command.

This design characteristic can take some time to adjust to, but is quite reasonable for most uses of a numerical solution.

The plot of a numerical solution can be created in a number of different ways. One of the simplest is to use the `odeplot` command from the `plots` package, which must be loaded prior to use. The basic syntax for `odeplot` expects the first argument to be the output from numeric `dsolve`, the second argument should be a list of two or three expressions involving the dependent and independent variables. For example,

```
> numSOLN := dsolve( IVP, x(t), type=numeric ):
> odeplot( numSOLN, [ t, x(t) ], 0..1,
>          title=`Approx. solution to x' = x^2+1, x(0)=0` );
```

Both numeric `dsolve` and `odeplot` work equally well with higher-order equations and with systems. The basic operations for higher-order problem will be illustrated using the damped nonlinear pendulum.

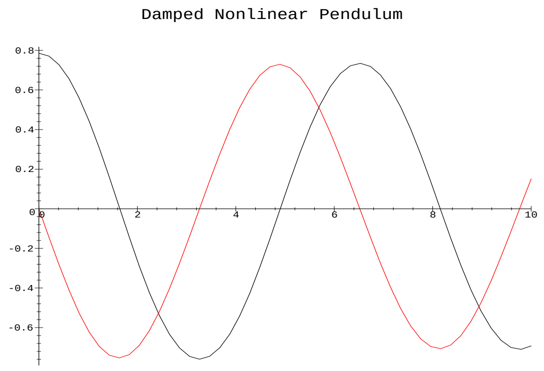
```
> ODE := diff( theta(t), t$2 ) + 0.2*diff( theta(t), t ) + sin(theta(t)) = 0:
> IC := theta(0)=Pi/4, D(theta)(0)=0:
> IVP := { ODE, IC };
```

$$IVP := \left\{ \theta(\,0\,) = \frac{1}{4}\,\pi, \left(\frac{\partial^2}{\partial t^2}\,\theta(\,t\,)\right) + .2\,\left(\frac{\partial}{\partial t}\,\theta(\,t\,)\right) + \sin(\,\theta(\,t\,)\,) = 0, \mathrm{D}(\,\theta\,)(\,0\,) = 0 \right\}$$

```
> PEND := dsolve( IVP, theta(t), type=numeric ):
```

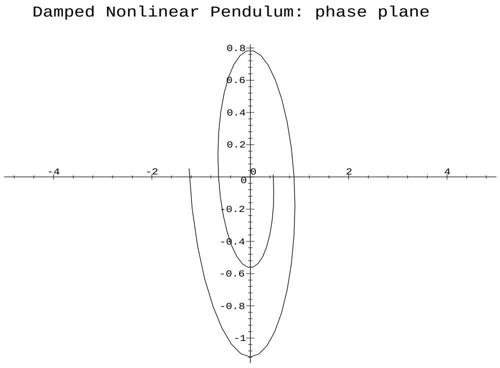
Multiple graphs are included in one plot when the second argument to `odeplot` is a list of lists. (Note how the derivative of the solution is selected using `diff`.)

```
> odeplot( PEND, [ [ t, theta(t) ], [ t, diff(theta(t),t) ] ], 0..10,
>          title=`Damped Nonlinear Pendulum` );
```



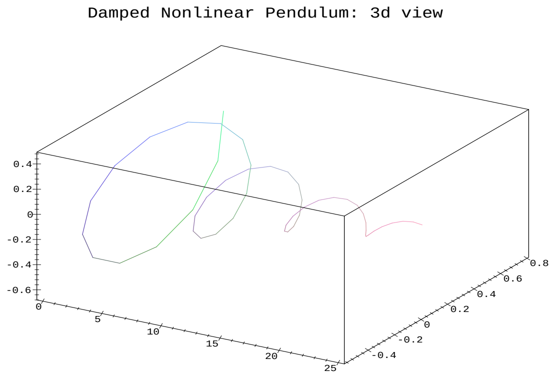
The phase portrait is also easy to create

```
> odeplot( PEND, [ diff(theta(t),t), theta(t) ], -5..5,
>          title=`Damped Nonlinear Pendulum: phase plane` );
```



A three-dimensional view can sometimes be quite illustrative.

```
> odeplot( PEND, [ t, theta(t), diff(theta(t),t) ], 0..25,
>          title=`Damped Nonlinear Pendulum: 3d view`, axes=BOXED );
```



There are times when the standard output from numeric `dsolve` is not suitable for further processing. The `output=` and `value=` optional arguments provide some flexibility. Specifying `output=listprocedure` will cause `dsolve` to return a Maple list of procedures. (As has been used previously, the default, which can be specified using `output=listprocedure`, returns a single procedure whose values are lists of equations.)

```
> PEND2 := dsolve( IVP, theta(t), type=numeric, output=listprocedure );

        PEND2 := [t = proc(t) ... end, theta(t) = proc(t) ... end,

                      d
                     ---- theta(t) = proc(t) ... end]
                      dt
```

```
> THETA := subs( PEND2, theta(t) ):
> dTHETA := subs( PEND2, diff(theta(t),t) ):
```

```
> plot( { THETA, dTHETA }, 0..10,
>        title=`Damped Nonlinear Pendulum: another view` );
```

The `value=` argument is used when the value of the solution is needed only for specified values of the the independent variable. This can be particularly useful when creating a table of data

```
> PEND3 := dsolve( IVP, theta(t), type=numeric,
>                  value=array([ i/100 $ i=320..330 ]) );
```

$$PEND3 := \left[\begin{array}{c} \left[\begin{array}{ccc} t & \theta(t) & \frac{\partial}{\partial t}\theta(t) \end{array}\right] \\ \left[\begin{array}{ccc} 3.200000000000000 & -.5620403787464741 & -.02994513274318901 \\ 3.210000000000000 & -.5623128988554575 & -.02456029970549107 \\ 3.220000000000000 & -.5625316136845078 & -.01918414978400469 \\ 3.230000000000000 & -.5626966122525803 & -.01381712023480108 \\ 3.240000000000000 & -.5628079879421350 & -.008459646504914679 \\ 3.250000000000000 & -.5628658384811120 & -.003112162245458325 \\ 3.260000000000000 & -.5628702659250371 & .002224900675346114 \\ 3.270000000000000 & -.5628213766392575 & .007551112159254484 \\ 3.280000000000000 & -.5627192812813063 & .01286604386524520 \\ 3.290000000000000 & -.5625640947833933 & .01816926919696234 \\ 3.300000000000000 & -.5623559363350199 & .02346036329038838 \end{array}\right] \end{array}\right]$$

Extracting information from this table can be frustrating. Note that the structure of PEND3 is a two-dimensional (column) array. The first element of PEND3 is the three-dimensional (row) vector which describes the contents of the matrix contained in the second element of PEND3.

```
> VARS := evalm( PEND3[1,1] );
```

$$VARS := \left[t \;\; \theta(t) \;\; \frac{\partial}{\partial t}\theta(t) \right]$$

```
> VARS[2];
```

$$\theta(t)$$

It is also possible to nest the indices to directly access entries in this data structure. For example, the pendulum is at rest sometime between $t = 3.25$ and $t = 3.26$. At this time the angle describing the pendulum's position is approximately

```
> rest := ( PEND3[2,1][6,2] + PEND3[2,1][7,2] )/2;
```

$$rest := -.5628680523$$

which, converted to degrees, is

```
> evalf( rest * 180/Pi );
```

$$-32.24996381$$

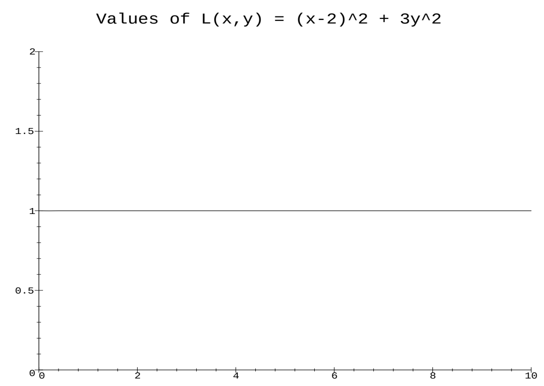
To illustrate the use of numeric `dsolve` for a system, consider the second-order system

```
> ODE := diff( x(t), t ) = 3*x(t)^2*y(t) - 6*y(t)^2,
>        diff( y(t), t ) = - x(t)^3 + 2*x(t)^2 + 2*x(t)*y(t) - 4*y(t):
> IC := x(0) = 1, y(0)=0:
> IVP := { ODE, IC };
```

$$IVP := \left\{ \frac{\partial}{\partial t}\, \mathrm{x}(t) = 3\,\mathrm{x}(t)^2\,\mathrm{y}(t) - 6\,\mathrm{y}(t)^2, \mathrm{x}(0) = 1, \mathrm{y}(0) = 0, \right.$$
$$\left. \frac{\partial}{\partial t}\, \mathrm{y}(t) = -\mathrm{x}(t)^3 + 2\,\mathrm{x}(t)^2 + 2\,\mathrm{x}(t)\,\mathrm{y}(t) - 4\,\mathrm{y}(t) \right\}$$

The level curves of the (Liapunov) function $L(x, y) = (x-2)^2 + 3y^2$ are trajectories for this system. That is, $L(x(t), y(t)) = L(x_0, y_0)$ for all $t > 0$. This can be illustrated using numeric `dsolve` and `odeplot` as follows:

```
> numSOLN := dsolve( IVP, [ x(t), y(t) ], type=numeric ):
> odeplot( numSOLN, [ t, (x(t)-2)^2 + 3*y(t)^2 ], 0..10, 0..2,
>          title=`Values of L(x,y) = (x-2)^2 + 3y^2` );
```



Note that if the vertical range `0..2` is omitted, then the plot looks a little surprising at first glance. A close examination of the default vertical range selected by Maple explains this apparent anomaly.

The preceding examples illustrate a few of the options that are available in numeric `dsolve`. Full details are available from the on-line help pages for

```
> ?dsolve
> ?dsolve[numeric]
> ?dsolve[dverk78]
```

and

```
> ?plots[odeplot]
```

Series Solutions

Some differential equations, and systems, are most approachable using power series techniques. Maple provides access to this method by way of the `type=series` optional argument to `dsolve`.

Consider the differential equation, in normal form,

```
> ODE := diff( y(x), x$2 ) + 1/x * diff( y(x), x ) - 1/(x^2+1) * y(x) = 0;
```

$$ODE := \left(\frac{\partial^2}{\partial x^2}\mathrm{y}(x)\right) + \frac{\frac{\partial}{\partial x}\mathrm{y}(x)}{x} - \frac{\mathrm{y}(x)}{x^2+1} = 0$$

The first few terms of the power series, centered at $x = 0$, can be obtained with the command

```
> dsolve( ODE, y(x), type=series );
```

$$\mathrm{y}(x) = \_C1\left(1 + \frac{1}{4}x^2 - \frac{3}{64}x^4 + \mathrm{O}(x^6)\right) + \_C2\left(\ln(x)\left(1 + \frac{1}{4}x^2 - \frac{3}{64}x^4 + \mathrm{O}(x^6)\right) + \left(-\frac{1}{4}x^2 + \frac{1}{128}x^4 + \mathrm{O}(x^6)\right)\right)$$

To obtain additional terms in the sequence, modify the system constant `Order`

```
> Order := 10:
> Y9 := dsolve( ODE, y(x), type=series );
```

$$Y9 := \mathrm{y}(x) = \_C1\left(1 + \frac{1}{4}x^2 - \frac{3}{64}x^4 + \frac{5}{256}x^6 - \frac{175}{16384}x^8 + \mathrm{O}(x^{10})\right) + \_C2\Bigg(\ln(x)\left(1 + \frac{1}{4}x^2 - \frac{3}{64}x^4 + \frac{5}{256}x^6 - \frac{175}{16384}x^8 + \mathrm{O}(x^{10})\right) + \left(-\frac{1}{4}x^2 + \frac{1}{128}x^4 + \frac{1}{1536}x^6 - \frac{265}{196608}x^8 + \mathrm{O}(x^{10})\right)\Bigg)$$

Note that $x = 0$ is a regular singular point for this equation. Note also the structure of this solution. The presence of the logarithmic factor is enlightening; that same term also complicates the conversion of this solution from an equation whose RHS is of type `series` into an expression that can be plotted, *etc.*.

Series solutions centered at other points can be requested by specifying a full set of initial conditions at the desired value of the independent variable. Here, the series expansion of the solution centered at $x = -1$ will be produced

```
> IVP := { ODE, y(-1)=c1, D(y)(-1)=c2 };
```

$$IVP := \left\{\mathrm{D}(y)(-1) = c2, \mathrm{y}(-1) = c1, \left(\frac{\partial^2}{\partial x^2}\mathrm{y}(x)\right) + \frac{\frac{\partial}{\partial x}\mathrm{y}(x)}{x} - \frac{\mathrm{y}(x)}{x^2+1} = 0\right\}$$

```
> Order:=6:
> Y5 := dsolve( IVP, y(x), type=series );
```

$$Y5 := \mathrm{y}(x) = c1 + c2\,(x+1) + \left(\frac{1}{4}c1 + \frac{1}{2}c2\right)(x+1)^2 + \left(\frac{5}{12}c2 + \frac{1}{6}c1\right)(x+1)^3 + \left(\frac{1}{3}c2 + \frac{11}{96}c1\right)(x+1)^4 + \left(\frac{127}{480}c2 + \frac{1}{12}c1\right)(x+1)^5 + \mathrm{O}\left((x+1)^6\right)$$

Here are two ways of obtaining the series solutions for two linearly independent solutions to this ODE. First, it is possible to simply specify two different sets of values for the constants `c_1` and `c_2`:

```
> Y5a := subs( c1=1, c2=0, rhs(Y5 ) );
> Y5b := subs( c1=0, c2=1, rhs(Y5 ) );
```

$$Y5a := 1+\frac{1}{4}(x+1)^2+\frac{1}{6}(x+1)^3+\frac{11}{96}(x+1)^4+\frac{1}{12}(x+1)^5+\mathrm{O}\left((x+1)^6\right)$$

$$Y5b := x+1+\frac{1}{2}(x+1)^2+\frac{5}{12}(x+1)^3+\frac{1}{3}(x+1)^4+\frac{127}{480}(x+1)^5+\mathrm{O}\left((x+1)^6\right)$$

Alternatively, Maple can `collect` the terms involving `c1` and `c2` separately. But, before this `collect` can be used, the solution must be converted to a polynomial. (So long as the `O` term is present, the solution is an equation whose RHS is of type `series`.)

```
> Y5 := convert( rhs(Y5), polynom );
```

$$Y5 := c1 + c2\,(x+1) + \left(\frac{1}{4}\,c1+\frac{1}{2}\,c2\right)(x+1)^2 + \left(\frac{5}{12}\,c2+\frac{1}{6}\,c1\right)(x+1)^3 + \left(\frac{1}{3}\,c2+\frac{11}{96}\,c1\right)(x+1)^4 + \left(\frac{127}{480}\,c2+\frac{1}{12}\,c1\right)(x+1)^5$$

```
> Y5 := collect( Y5, [ c1, c2 ] );
```

$$Y5 := \left(\frac{1}{4}(x+1)^2+\frac{1}{6}(x+1)^3+\frac{11}{96}(x+1)^4+1+\frac{1}{12}(x+1)^5\right)c1 + \left(\frac{5}{12}(x+1)^3+x+1+\frac{1}{2}(x+1)^2+\frac{127}{480}(x+1)^5+\frac{1}{3}(x+1)^4\right)c2$$

Note that the ordering of the terms in this last expression is less than optimal. (Controlling the order of terms in a polynomial is not a simple task in Maple.)

```
> plot( map( convert, { Y5a, Y5b }, polynom ), x=-2..1,
>       title=`Linearly Independent Solution` );
```

For additional assistance, please consult the on-line help

```
> ?dsolve
```

Laplace Transforms

All uses of Laplace transforms should begin by reading the Laplace transform library into the Maple session.

```
> readlib( laplace ):
```

The two commands that form the Laplace transform pair are `laplace` and `invlaplace`. These commands can be used explicitly or implicitly to solve a differential equation. To illustrate both approaches, consider the the problem of determining the general solution to

```
> ODE := diff( y(t), t$2 ) + 3*diff( y(t), t ) - 4 * y(t) = 0;
```

$$ODE := \left(\frac{\partial^2}{\partial t^2}\mathrm{y}(t)\right)+3\left(\frac{\partial}{\partial t}\mathrm{y}(t)\right)-4\,\mathrm{y}(t)=0$$

One approach is to begin by applying the Laplace transform to this equation and solving for the Laplace transform, $Y(s)$, of the solution $y(t)$:

```
> lapODE := laplace( ODE, t, s );
```

$$lapODE := (\,\mathrm{laplace}(\,\mathrm{y}(\,t\,), t, s\,)\,s - \mathrm{y}(\,0\,)\,)\,s - \mathrm{D}(\,y\,)(\,0\,) + 3\,\mathrm{laplace}(\,\mathrm{y}(\,t\,), t, s\,)\,s - 3\,\mathrm{y}(\,0\,) - 4\,\mathrm{laplace}(\,\mathrm{y}(\,t\,), t, s\,) = 0$$

```
> SOLY := solve( lapODE, laplace( y(t), t, s ) );
```

$$SOLY := -\frac{-s\,\mathrm{y}(\,0\,) - \mathrm{D}(\,y\,)(\,0\,) - 3\,\mathrm{y}(\,0\,)}{s^2 + 3\,s - 4}$$

Next, the inverse Laplace transform is applied to find the solution $y(t)$:

```
> SOLy := invlaplace( SOLY, s, t );
```

$$SOLy := \frac{1}{5}\,\mathrm{y}(\,0\,)\,\mathrm{e}^{(-4\,t)} - \frac{1}{5}\,\mathrm{D}(\,y\,)(\,0\,)\,\mathrm{e}^{(-4\,t)} + \frac{4}{5}\,\mathrm{y}(\,0\,)\,\mathrm{e}^{t} + \frac{1}{5}\,\mathrm{D}(\,y\,)(\,0\,)\,\mathrm{e}^{t}$$

or, in a form that is sometimes more useful,

```
> collect( SOLy, [ y(0), D(y)(0) ] );
```

$$\left(\frac{1}{5}\,\mathrm{e}^{(-4\,t)} + \frac{4}{5}\,\mathrm{e}^{t}\right)\,\mathrm{y}(\,0\,) + \left(-\frac{1}{5}\,\mathrm{e}^{(-4\,t)} + \frac{1}{5}\,\mathrm{e}^{t}\right)\,\mathrm{D}(\,y\,)(\,0\,)$$

Alternatively, a one-step solution is possible with the use of the `method=laplace` optional argument to `dsolve`.

```
> dsolve( ODE, y(t), method=laplace );
```

$$\mathrm{y}(\,t\,) = \frac{1}{5}\,\mathrm{y}(\,0\,)\,\mathrm{e}^{(-4\,t)} - \frac{1}{5}\,\mathrm{D}(\,y\,)(\,0\,)\,\mathrm{e}^{(-4\,t)} + \frac{4}{5}\,\mathrm{y}(\,0\,)\,\mathrm{e}^{t} + \frac{1}{5}\,\mathrm{D}(\,y\,)(\,0\,)\,\mathrm{e}^{t}$$

For additional assistance, please consult the on-line help for

```
> ?dsolve
```

as well as the help pages for the individual commands

```
> ?laplace
> ?invlaplace
```

DESol

Maple provides a data structure for the explicit manipulation of a differential equation *without* attempting to solve the equation. This data structure, `DESol`, is not a prime candidate for use in an introductory course in differential equations, and is not discussed in any detail in this supplement. Full information is available from the on-line help: `?DESol`