Wolfram *Mathematica*® Tutorial Collection

DIFFERENTIAL EQUATION SOLVING WITH DSOLVE

For use with Wolfram *Mathematica*® 7.0 and later.

For the latest updates and corrections to this manual:
visit reference.wolfram.com

For information on additional copies of this documentation:
visit the Customer Service website at www.wolfram.com/services/customerservice
or email Customer Service at info@wolfram.com

Comments on this manual are welcomed at:
comments@wolfram.com

Content authored by:
Devendra Kapadia

Printed in the United States of America.

15 14 13 12 11 10 9 8 7 6 5 4 3 2

Contents

Introduction to Differential Equation Solving with DSolve

The *Mathematica* function `DSolve` finds *symbolic solutions* to differential equations. (The *Mathematica* function `NDSolve`, on the other hand, is a general numerical differential equation solver.) `DSolve` can handle the following types of equations:

- *Ordinary Differential Equations* (ODEs), in which there is a single independent variable t and one or more dependent variables $x_i(t)$. `DSolve` is equipped with a wide variety of techniques for solving single ODEs as well as systems of ODEs.
- *Partial Differential Equations* (PDEs), in which there are two or more independent variables and one dependent variable. Finding exact symbolic solutions of PDEs is a difficult problem, but `DSolve` can solve most first-order PDEs and a limited number of the second-order PDEs found in standard reference books.
- *Differential-Algebraic Equations* (DAEs), in which some members of the system are differential equations and the others are purely algebraic, having no derivatives in them. As with PDEs, it is difficult to find exact solutions to DAEs, but `DSolve` can solve many examples of such systems that occur in applications.

`DSolve[`*eqn*`,`*y*`[`*x*`],`*x*`]`	solve a differential equation for $y[x]$
`DSolve[{`*eqn*$_1$`,`*eqn*$_2$`,...},{`*y*$_1$`[`*x*`],`*y*$_2$`[`*x*`],...},`*x*`]`	solve a system of differential equations for $y_i[x]$

Finding symbolic solutions to ordinary differential equations.

`DSolve` returns results as lists of rules. This makes it possible to return multiple solutions to an equation. For a system of equations, possibly multiple solution sets are grouped together. You can use the rules to substitute the solutions into other calculations.

This finds the *general solution* for the given ODE. A rule for the function that satisfies the equation is returned.

```
In[1]:= DSolve[{y'[x] == y[x]}, y[x], x]
Out[1]= {{y[x] → e^x C[1]}}
```

You can pick out a specific solution by using `/.` (`ReplaceAll`).

```
In[2]:= y[x] /. DSolve[{y'[x] == y[x]}, y[x], x]
Out[2]= {{e^x C[1]}}
```

A general solution contains arbitrary parameters `C[i]` that can be varied to produce particular solutions for the equation. When an adequate number of *initial conditions* are specified, `DSolve` returns particular solutions to the given equations.

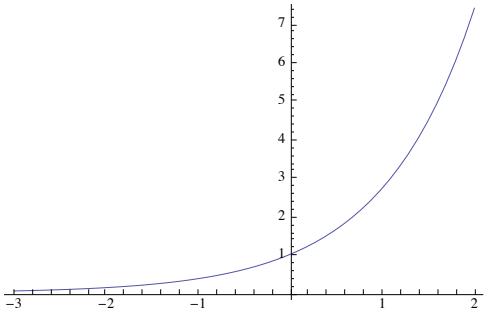
Here, the initial condition `y[0] == 1` is specified, and `DSolve` returns a particular solution for the problem.

```
In[3]:= sol = DSolve[{y'[x] == y[x], y[0] == 1}, y[x], x]
Out[3]= {{y[x] → e^x}}
```

This plots the solution. `ReplaceAll` (`/.`) is used in the `Plot` command to substitute the solution for $y[x]$.

```
In[4]:= Plot[y[x] /. sol, {x, -3, 2}]
```

Out[4]=



`DSolve[`eqn, y, x`]`	solve a differential equation for y as a pure function
`DSolve[{`$eqn_1, eqn_2, \ldots$`}, {`$y_1, y_2, \ldots$`}, `x`]`	solve a system of differential equations for the pure functions y_i

Finding symbolic solutions to ordinary differential equations as pure functions.

When the second argument to `DSolve` is specified as y instead of $y[x]$, the solution is returned as a pure function. This form is useful for verifying the solution of the ODE and for using the solution in further work. More details are given in "Setting Up the Problem".

The solution to this differential equation is given as a pure function.

```
In[5]:= eqn = {y'[x] == y[x]^2, y[0] == 1};
        sol = DSolve[eqn, y, x]
```

$$\text{Out[6]= } \left\{\left\{y \to \text{Function}\left[\{x\}, \frac{1}{1-x}\right]\right\}\right\}$$

This verifies the solution.

```
In[7]:= eqn /. sol
```

```
Out[7]= {{True, True}}
```

This solves a system of ODEs. Each solution is labeled according to the name of the function (here, x and y), making it easier to pick out individual functions.

```
In[8]:= eqns = {(t^2 + 1) * x'[t] == -t * x[t] + y[t] - Sign[t],
            (t^2 + 1) * y'[t] == -x[t] - t * y[t] + t * UnitStep[t], x[0] == -1 / 2, y[0] == 2};
        sol = DSolve[eqns, {x, y}, t]
```

$$\text{Out[9]= } \left\{\left\{x \to \text{Function}\left[\{t\}, \frac{-1 + 4t + 2\begin{pmatrix}\text{ArcTan}[t] & t \le 0 \\ -t & \text{True}\end{pmatrix} + 2t\begin{pmatrix}\frac{1}{2}\text{Log}\left[1+t^2\right] & t \le 0 \\ 0 & \text{True}\end{pmatrix}}{2\left(1+t^2\right)}\right],\right.$$

$$\left. y \to \text{Function}\left[\{t\}, \frac{4 + t - 2t\begin{pmatrix}\text{ArcTan}[t] & t \le 0 \\ -t & \text{True}\end{pmatrix} + 2\begin{pmatrix}\frac{1}{2}\text{Log}\left[1+t^2\right] & t \le 0 \\ 0 & \text{True}\end{pmatrix}}{2\left(1+t^2\right)}\right]\right\}\right\}$$

This substitutes a random value for the independent variable and shows that the solutions are correct at that point.

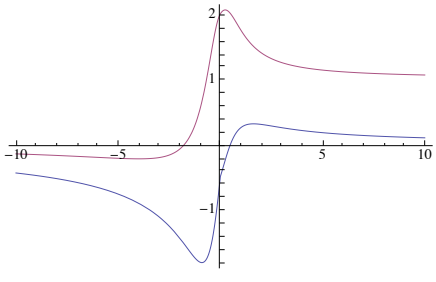
```
In[10]:= eqns /. sol /. {t → RandomReal[]}
```

```
Out[10]= {{True, True, True, True}}
```

This plots the solutions.

```
In[11]:= Plot[{x[t] /. sol, y[t] /. sol}, {t, -10, 10}]
```

```
Out[11]=
```



`DSolve[`*eqn*`,`*u*`[`*x*`,`*y*`],{`*x*`,`*y*`}]`	solve a partial differential equation for $u[x, y]$

Finding symbolic solutions to partial differential equations.

While general solutions to ordinary differential equations involve arbitrary *constants*, general solutions to partial differential equations involve arbitrary *functions*. `DSolve` labels these arbitrary functions as `C[i]`.

Here is the general solution to a linear first-order PDE. In the solution, `C[1]` labels an arbitrary function of $\frac{-x+y}{x\,y}$.

```
In[12]:= eqn = x^2 * D[u[x, y], x] + y^2 * D[u[x, y], y] - (x + y) * u[x, y];
         sol = DSolve[eqn == 0, u, {x, y}]
```

$$\text{Out[13]= } \left\{\left\{u \to \text{Function}\left[\{x, y\}, -x\,y\,C[1]\left[\frac{-x+y}{x\,y}\right]\right]\right\}\right\}$$

This obtains a particular solution to the PDE for a specific choice of `C[1]`.

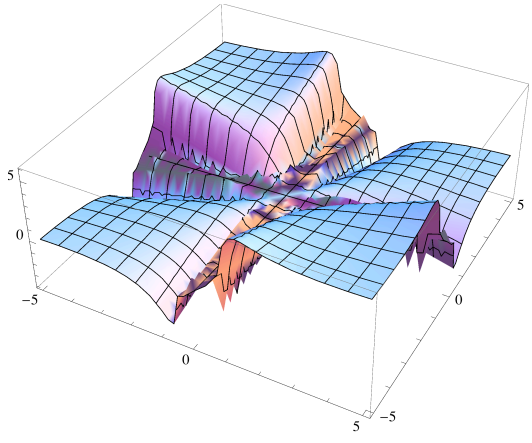
```
In[14]:= fn = u[x, y] /. sol[[1]] /. {C[1][t_] → Sin[t^2] + (t / 10)}
```

$$\text{Out[14]= } -x\,y\left(\frac{-x+y}{10\,x\,y} + \text{Sin}\left[\frac{(-x+y)^2}{x^2\,y^2}\right]\right)$$

Here is a plot of the surface for this solution.

```
In[15]:= Plot3D[fn, {x, -5, 5}, {y, -5, 5}]
```

Out[15]=



`DSolve` can also solve differential-algebraic equations. The syntax is the same as for a system of ordinary differential equations.

This solves a DAE.

```
In[16]:= eqns = {f''[x] == g[x], f[x] + g[x] == 3 Sin[x], f[Pi] == 1, f'[Pi] == 0};
         sol = DSolve[eqns, {f, g}, x]
```

$$\text{Out[17]= } \left\{\left\{f \to \text{Function}\left[\{x\}, \frac{1}{2}\left(-2\,\text{Cos}[x] + 3\,\pi\,\text{Cos}[x] - 3\,x\,\text{Cos}[x] + 3\,\text{Sin}[x]\right)\right],\right.$$
$$\left.g \to \text{Function}\left[\{x\}, \frac{1}{2}\left(2\,\text{Cos}[x] - 3\,\pi\,\text{Cos}[x] + 3\,x\,\text{Cos}[x] + 3\,\text{Sin}[x]\right)\right]\right\}\right\}$$

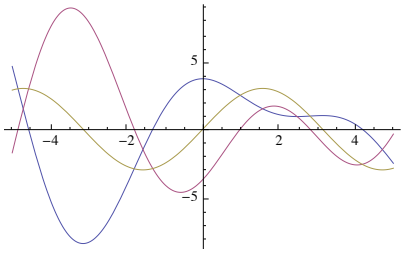
This verifies the solutions.

```
In[18]:= Simplify[eqns /. sol]
Out[18]= {{True, True, True, True}}
```

A plot of the solutions shows that their sum satisfies the algebraic relation $f[x] + g[x] == 3\,\text{Sin}[x]$.

```
In[19]:= Plot[{f[x] /. sol, g[x] /. sol, f[x] + g[x] /. sol}, {x, -5, 5}]
```

Out[19]=



Goals of Differential Equation Solving with DSolve Tutorials

The design of DSolve is modular: the algorithms for different classes of problems work independently of one another. Once a problem has been classified (as described in "Classification of Differential Equations"), the available methods for that class are tried in a specific sequence until a solution is obtained. The code has a hierarchical structure whereby the solution of complex problems is reduced to the solution of relatively simpler problems, for which a greater variety of methods is available. For example, higher-order ODEs are typically solved by reducing their order to 1 or 2.

The process described is done internally and does not require any intervention from the user. For this reason, these tutorials have the following basic goals.

- To provide enough information and tips so that users can pose problems to DSolve in the most appropriate form and apply the solutions in their work. This is accomplished through a substantial number of examples. A summary of this information is given in "Working with DSolve".
- To give a catalog of the kinds of problems that can be handled by DSolve as well as the nature of the solutions for each case. This is provided in the tutorials on ODEs, PDEs, DAEs, and boundary value problems (BVPs).

The author hopes that these Differential Equation Solving with `DSolve` tutorials will be useful in acquiring a basic knowledge of `DSolve` and also serve as a ready reference for information on more advanced topics.

Classification of Differential Equations

While differential equations have three basic types—ordinary (ODEs), partial (PDEs), or differential-algebraic (DAEs), they can be further described by attributes such as order, linearity, and degree. The solution method used by `DSolve` and the nature of the solutions depend heavily on the class of equation being solved.

The *order* of a differential equation is the order of the highest derivative in the equation.

This is a first-order ODE because its highest derivative is of order 1.

```
In[1]:= DSolve[x^2 (1 - x^2) * y'[x] == (x - 3 x^3 - y[x]) y[x], y[x], x]
```

$$\text{Out[1]}= \left\{\left\{y[x] \to \frac{-x + x^3}{C[1] - \text{Log}[x]}\right\}\right\}$$

Here is the general solution to a fourth-order ODE.

```
In[2]:= DSolve[y''''[x] - 16 * y[x] == x^2, y[x], x]
```

$$\text{Out[2]}= \left\{\left\{y[x] \to -\frac{x^2}{16} + e^{2x} C[1] + e^{-2x} C[3] + C[2] \text{Cos}[2x] + C[4] \text{Sin}[2x]\right\}\right\}$$

A differential equation is linear if the equation is of the first degree in y and its derivatives, and if the coefficients are functions of the independent variable.

This is a *nonlinear* second-order ODE that represents the motion of a circular pendulum. It is nonlinear because `Sin[`$y[x]$`]` is not a linear function of $y[x]$. The `Solve::ifun` warning message appears because `Solve` uses `JacobiAmplitude` (the inverse of `EllipticF`) to find an expression for $y[x]$.

```
In[3]:= sol = DSolve[y''[x] + 3 * Sin[y[x]] == 0, y, x]
```

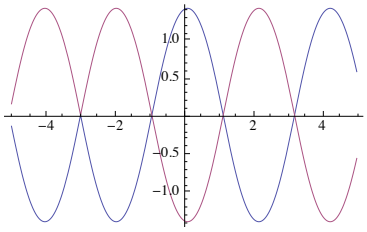
```
Solve::ifun :
  Inverse functions are being used by Solve, so some solutions may not be found; use Reduce for
      complete solution information. >>
```

$$\text{Out[3]}= \left\{\left\{y \to \text{Function}\left[\{x\}, -2\,\text{JacobiAmplitude}\left[\frac{1}{2}\sqrt{(6 + C[1])\,(x + C[2])^2}, \frac{12}{6 + C[1]}\right]\right]\right\},\right.$$
$$\left.\left\{y \to \text{Function}\left[\{x\}, 2\,\text{JacobiAmplitude}\left[\frac{1}{2}\sqrt{(6 + C[1])\,(x + C[2])^2}, \frac{12}{6 + C[1]}\right]\right]\right\}\right\}$$

This plots the solutions. The discontinuity in the graphs at $x = -3$ results from the choice of inverse functions used by `Solve`.

In[4]:=
```
Plot[{y[x] /. sol[[1]] /. {C[1] → -1, C[2] → 3},
  y[x] /. sol[[2]] /. {C[1] → -1, C[2] → 3}}, {x, -5, 5}]
```

Out[4]=



It should be noted that sometimes the solutions to fairly simple nonlinear equations are only available in implicit form. In these cases, `DSolve` returns an unevaluated `Solve` object.

This nonlinear differential equation only has an implicit solution. The `Solve::tdep` messages can be ignored; they appear because `Solve` cannot find an explicit expression for $y[x]$ because non-algebraic functions (`ArcTan` and `Log`) are involved.

In[5]:=
```
DSolve[(y[x] + x - 1) * y'[x] - y[x] + 2 x + 3 == 0, y[x], x]
```

Solve::tdep :
The equations appear to involve the variables to be solved for in an essentially non−algebraic way. ≫

Solve::tdep :
The equations appear to involve the variables to be solved for in an essentially non−algebraic way. ≫

Out[5]=

$$\text{Solve}\left[\frac{2}{3}\left(\sqrt{2}\ \text{ArcTan}\left[\frac{-2+\frac{2\,(2+3\,x)}{-1+x+y[x]}}{2\sqrt{2}}\right]-\text{Log}\left[\frac{(-1+x+y[x])^{2}\left(3+\frac{(2+3\,x)\left(-2+\frac{2+3\,x}{-1+x+y[x]}\right)}{-1+x+y[x]}\right)}{(2+3\,x)^{2}}\right]\right)==C[1]+\frac{4}{3}\text{Log}[2+3\,x],\ y\right]$$

When the coefficients of a linear ODE do not depend on x, the ODE is said to have *constant coefficients.*

This is an ODE with constant coefficients.

In[6]:=
```
eqn = y'''[x] + 3 * y''[x] - 25 * y'[x] + 21 * y[x];
sol = DSolve[eqn == 0, y[x], x]
```

Out[7]=
```
{{y[x] → e^(-7 x) C[1] + e^x C[2] + e^(3 x) C[3]}}
```

The previous equation is also *homogeneous*: all terms contain y or derivatives of y and its right-hand side is zero. Adding a function of the independent variable makes the equation *inhomogeneous*. The general solution to an inhomogeneous equation with constant coefficients is obtained by adding a particular integral to the solution to the corresponding homogeneous equation.

Here, x^2 is added to the right-hand side of the previous equation, making the new equation inhomogeneous. The general solution to this new equation is the sum of the previous solution and a particular integral.

```
In[8]:= sol2 = DSolve[eqn == x^2, y[x], x]
```

$$\text{Out[8]= } \left\{\left\{y[x] \to \frac{1124 + 1050\,x + 441\,x^2}{9261} + e^{-7x}\,C[1] + e^{x}\,C[2] + e^{3x}\,C[3]\right\}\right\}$$

When the coefficients of an ODE depend on x, the ODE is said to have *variable coefficients*. Since equations with variable coefficients that are *rational functions of* x have singularities that are easily classified, there are sophisticated algorithms available for solving them.

The coefficients of this equation are rational functions of x.

```
In[9]:= sol =
         DSolve[{y''[x] - ((1 / x) - (3 / (16 x^2))) * y[x] == 0, y[1] == 1, y'[1] == 4}, y[x], x]
```

$$\text{Out[9]= } \left\{\left\{y[x] \to \frac{1}{8} e^{-2-2\sqrt{x}} \left(-11\,e^{4} + 19\,e^{4\sqrt{x}}\right) x^{1/4}\right\}\right\}$$

There is a close relationship between functions and differential equations. Starting with a function of almost any type, it is possible to construct a differential equation satisfied by that function. Conversely, any differential equation gives rise to one or more functions, in the form of solutions to that equation. In fact, many *special functions* from classical analysis have their origins in the solution of differential equations. *Mathieu functions* are one such class of special functions. Mathieu was interested in studying the vibrations of elliptical membranes. The eigenfunctions for the wave equation that describes this motion are given by products of Mathieu functions.

This linear second-order ODE with rational coefficients has a general solution given by Mathieu functions.

```
In[10]:= DSolve[(t - 1) (t + 1) * y''[t] + t * y'[t] + (-2 - 6 * t^2) * y[t] == 0, y[t], t]
```

$$\text{Out[10]= } \left\{\left\{y[t] \to C[1]\,\text{MathieuC}\left[5, -\frac{3}{2}, \text{ArcCos}[t]\right] + C[2]\,\text{MathieuS}\left[5, -\frac{3}{2}, \text{ArcCos}[t]\right]\right\}\right\}$$

The presence of `ArcCos[t]` in the previous solution suggests that the equation can be given a simpler form using trigonometric functions. This is the form in which these equations were introduced by Mathieu in 1868.

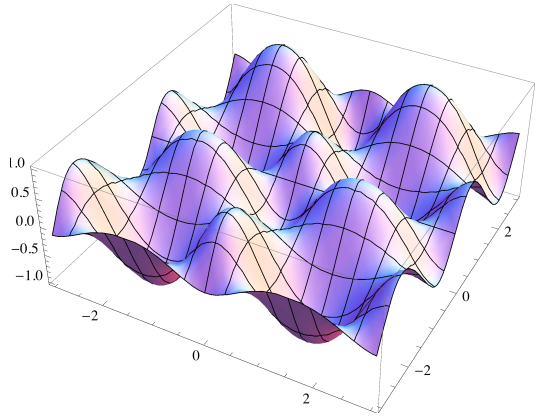
```
In[11]:= DSolve[y''[x] + (3 Cos[2 x] + 5) y[x] == 0, y, x]
```

$$\text{Out[11]= } \left\{\left\{y \to \text{Function}\left[\{x\}, C[1]\,\text{MathieuC}\left[5, -\frac{3}{2}, x\right] + C[2]\,\text{MathieuS}\left[5, -\frac{3}{2}, x\right]\right]\right\}\right\}$$

This plots the surface for a particular product of solutions to this equation.

In[12]:=
```
Plot3D[MathieuC[5, -3/2, x] * MathieuS[5, -3/2, y], {x, -3, 3}, {y, -3, 3}]
```

Out[12]=



The *degree* of a differential equation is the highest power of the highest-order derivative in the equation.

This is a first-order ODE of degree 2.

In[13]:=
```
sol = DSolve[{y'[x]^2 == 1 - x^2, y[0] == 2}, y, x]
```

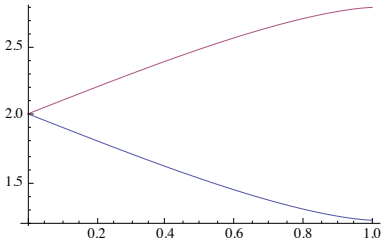
Out[13]=

$$\left\{\left\{y \to \text{Function}\left[\{x\}, \frac{1}{2}\left(4 - x\sqrt{1-x^2} - \text{ArcSin}[x]\right)\right]\right\}, \left\{y \to \text{Function}\left[\{x\}, \frac{1}{2}\left(4 + x\sqrt{1-x^2} + \text{ArcSin}[x]\right)\right]\right\}\right\}$$

The higher degree leads to non-uniqueness of the solution.

In[14]:=
```
Plot[{y[x] /. sol[[1]], y[x] /. sol[[2]]}, {x, 0, 1}]
```

Out[14]=



The examples in this tutorial have focused on the classification of ODEs. The classification of PDEs is similar but more involved. PDEs can also be classified by linearity or nonlinearity, order, degree, and constant or variable coefficients. More important is the classification that identifies a PDE as hyperbolic, parabolic, or elliptic. These classifications are discussed in further detail in "Second-Order PDEs".

Ordinary Differential Equations (ODEs)

Overview of Ordinary Differential Equations (ODEs)

There are four major areas in the study of ordinary differential equations that are of interest in pure and applied science.

- Exact solutions, which are closed-form or implicit analytical expressions that satisfy the given problem.
- Numerical solutions, which are available for a wider class of problems, but are typically only valid over a limited range of the independent variables.
- Qualitative theory, which is concerned with the global properties of solutions and is particularly important in the modern approach to dynamical systems.
- Existence and uniqueness theorems, which guarantee that there are solutions with certain desirable properties provided a set of conditions is fulfilled by the differential equation.

Of these four areas, the study of exact solutions has the longest history, dating back to the period just after the discovery of calculus by Sir Isaac Newton and Gottfried Wilhelm von Leibniz. The following table introduces the types of equations that can be solved by `DSolve`.

name of equation	*general form*	*date of discovery*	*mathematician*
Separable	$y'(x) = f(x)\, g(y)$	1691	G. Leibniz
Homogeneous	$y'(x) = f\left(\frac{x}{y(x)}\right)$	1691	G. Leibniz
Linear first-order ODE	$y'(x) + P(x)\, y(x) = Q(x)$	1694	G. Leibniz
Bernoulli	$y'(x) + P(x)\, y(x) = Q(x)$	1695	James Bernoulli
Riccati	$y'(x) = f(x) + g(x)\, y(x) + h(x)\, y(x)^2$	1724	Count Riccati
Exact first-order ODE	$M\, d\, x + N\, d\, y = 0$ with $\frac{\partial M}{\partial y} = \frac{\partial N}{\partial x}$	1734	L. Euler
Clairaut	$y(x) = x\, y'(x) + f(y'(x))$	1734	A-C. Clairaut
Linear with constant coefficients	$y^{(n)}(x) + a_{n-1}\, y^{(n-1)}(x) + \ldots + a_0\, y(x) = P(x)$ with a_1 constant	1743	L. Euler
Hypergeometric	$x(1-x)\, y''(x) + (c - (a+b+1)\, x)\, y'(x) - a\, b\, y(x) = 0$	1769	L. Euler
Legendre	$\left(1 - x^2\right) y''(x) - 2\, x\, y'(x) + n(n+1)\, y(x) = 0$	1785	M. Legendre
Bessel	$x^2\, y''(x) + x\, y'(x) + \left(x^2 - n^2\right) y(x) = 0$	1824	F. Bessel
Mathieu	$y''(x) + (a - 2\, q \cos(2\, x))\, y(x) = 0$	1868	E. Mathieu
Abel	$y'(x) = f(x) + g(x)\, y(x) + h(x)\, y(x)^2 + k(x)\, y(x)^3$	1834	N. H. Abel
Chini	$y'(x) = f(x) + g(x)\, y(x) + h(x)\, y(x)^n$	1924	M. Chini

Examples of ODEs belonging to each of these types are given in other tutorials (clicking a link in the table will bring up the relevant examples).

First-Order ODEs

Straight Integration

This equation is solved by simply integrating the right-hand side with respect to x.

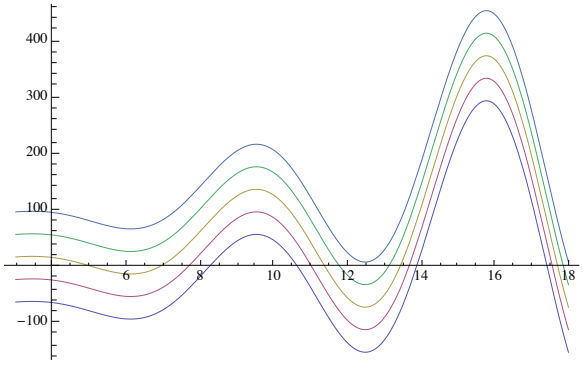
```
In[1]:= sol = DSolve[y'[x] == x^2 * Sin[x] + Sqrt[1 + x^2], y, x]
```

$$\text{Out[1]= } \left\{\left\{\text{y} \to \text{Function}\left[\{\text{x}\}, \frac{1}{2}\, \text{x}\, \sqrt{1+\text{x}^2} + \frac{\text{ArcSinh[x]}}{2} + \text{C[1]} - \left(-2 + \text{x}^2\right) \text{Cos[x]} + 2\, \text{x}\, \text{Sin[x]}\right]\right\}\right\}$$

Here is a plot of the integral curves for different values of the arbitrary constant `C[1]`.

```
In[2]:= tab = Table[y[x] /. sol[[1]] /. {C[1] -> k}, {k, -80, 80, 40}];
In[3]:= Plot[Evaluate[tab], {x, 3, 18}]
```

Out[3]=



Separable Equations

The general solution to this equation is found by separation of variables.

```
In[1]:= DSolve[y'[x] == x^2 y[x]^2 / Sqrt[3 - x^2], y, x]
```

Out[1]= $\left\{\left\{y \to \text{Function}\left[\{x\}, \frac{2}{x\sqrt{3-x^2} - 3\,\text{ArcSin}\left[\frac{x}{\sqrt{3}}\right] - 2\,C[1]}\right]\right\}\right\}$

Even when variables can be separated, the final solution might be accompanied by a warning message from `Solve`, or it might only be given as an `InverseFunction` object.

Solving this ODE generates a warning message because `Solve` obtains an expression for $y[x]$ using `Log`, the inverse of `Exp`. This warning message can be ignored.

```
In[2]:= DSolve[y'[x] == x^2 Exp[y[x]] / Sqrt[3 - x^2], y, x]
```

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

Out[2]= $\left\{\left\{y \to \text{Function}\left[\{x\}, -\text{Log}\left[\frac{1}{2}x\sqrt{3-x^2} - \frac{3}{2}\,\text{ArcSin}\left[\frac{x}{\sqrt{3}}\right] - C[1]\right]\right]\right\}\right\}$

The solution to this equation is given as an `InverseFunction` object, in order to get an explicit expression for $y[x]$.

```
In[3]:= sol = DSolve[y'[x] == x^2 / (Sqrt[9 - x^2] Log[y[x]] * Sin[y[x]]), y[x], x]
```

```
InverseFunction::ifun: Inverse functions are being used. Values may be lost for multivalued inverses. >>

InverseFunction::ifun: Inverse functions are being used. Values may be lost for multivalued inverses. >>

Solve::tdep:
  The equations appear to involve the variables to be solved for in an essentially non-algebraic way. >>
```

```
Out[3]= {{y[x] → InverseFunction[CosIntegral[#1] - Cos[#1] Log[#1] &][
    -i (1/2 Sqrt[-3 + x] x Sqrt[3 + x] + 9 ArcSinh[Sqrt[-3 + x]/Sqrt[6]]) + C[1]]}}
```

Homogeneous Equations

Here is a homogeneous equation in which the total degree of both the numerator and the denominator of the right-hand side is 2. The two parts of the solution list give branches of the integral curves in the form $y = f(x)$.

```
In[1]:= eqn = y'[x] == -(x^2 - 3 y[x]^2) / (x * y[x]);
        sol = DSolve[eqn, y, x]
```

```
Out[2]= {{y → Function[{x}, -Sqrt[x^2/2 + x^6 C[1]]]}, {y → Function[{x}, Sqrt[x^2/2 + x^6 C[1]]]}}
```

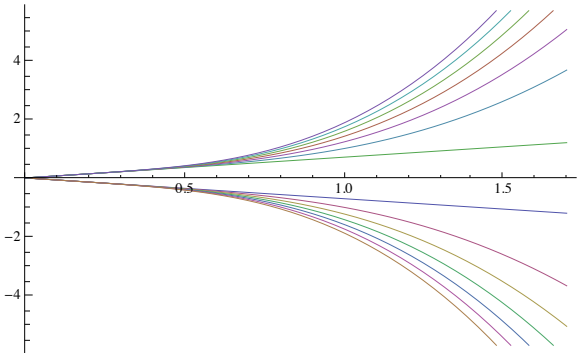
This plots both branches together, showing the complete integral curves $y^2 = C[1]\, x^6 + \frac{x^2}{2}$ for several values of `C[1]`.

```
In[3]:= tab1 = Table[y[x] /. sol[[1]] /. C[1] → k, {k, 0, 3, 0.5}];

In[4]:= tab2 = Table[y[x] /. sol[[2]] /. C[1] → k, {k, 0, 3, 0.5}];

In[5]:= Plot[Evaluate[Join[tab1, tab2]], {x, 0, 1.7}]
```

Out[5]=



If an initial condition is specified, `DSolve` picks the branch that passes through the initial point. The `DSolve::bvnul` message indicates that one branch of the general solution (the lower branch in the previous graph) did not give a solution satisfying the given initial condition $y[1] == 3$.

In[6]:=
```
DSolve[{eqn, y[1] == 3}, y[x], x]
```

DSolve::bvnul :
For some branches of the general solution, the given boundary conditions lead to an empty solution. ≫

Out[6]=
$$\left\{\left\{y[x] \to \frac{\sqrt{x^2\left(1+17\,x^4\right)}}{\sqrt{2}}\right\}\right\}$$

Linear First-Order Equations

The following is a linear first-order ODE because both $y[x]$ and $y'[x]$ occur in it with power 1 and $y'[x]$ is the highest derivative. Note that the solution contains the imaginary error function `Erfi`.

In[1]:=
```
DSolve[y'[x] + x * y[x] == Exp[3 x], y[x], x]
```

Out[1]=
$$\left\{\left\{y[x] \to e^{-\frac{x^2}{2}} C[1] + e^{-\frac{9}{2}-\frac{x^2}{2}} \sqrt{\frac{\pi}{2}}\ \text{Erfi}\left[\frac{3+x}{\sqrt{2}}\right]\right\}\right\}$$

Here is the solution for a more general linear first-order ODE. The `K` variables are used as dummy variables for integration. The `Erfi` term in the previous example comes from the integral in the second term of the general solution as follows.

In[2]:=
```
sol = DSolve[y'[x] + y[x] == Q[x], y[x], x]
```

Out[2]=
$$\left\{\left\{y[x] \to e^{-x} C[1] + e^{-x} \int_1^x e^{K[1]} Q[K[1]]\, dK[1]\right\}\right\}$$

A more traditional form of the solution can be obtained by replacing `K[1]` with a variable such as *t*.

In[3]:=
```
sol /. {K[1] → t}
```

Out[3]=
$$\left\{\left\{y[x] \to e^{-x} C[1] + e^{-x} \int_1^x e^{t} Q[t]\, dt\right\}\right\}$$

Inverse Linear Equations

It may happen that a given ODE is not linear in $y(x)$ but can be viewed as a linear ODE in $x(y)$. In this case, it is said to be an *inverse linear* ODE.

This is a inverse linear ODE. It is constructed by interchanging x and y in an earlier example.

```
In[1]:= DSolve[y'[x] == 1 / (-x * y[x] + Exp[3 * y[x]]), y[x], x]
```

```
Solve::tdep :
  The equations appear to involve the variables to be solved for in an essentially non-algebraic way. >>
```

```
Out[1]= Solve[x == e^(-1/2 y[x]^2) C[1] + e^(-9/2 - y[x]^2/2) Sqrt[π/2] Erfi[(3 + y[x])/Sqrt[2]], y[x]]
```

Bernoulli Equations

A Bernoulli equation is a first-order equation of the form

$$y'(x) + P(x)\, y(x) = Q(x) y(x)^n.$$

The problem of solving equations of this type was posed by James Bernoulli in 1695. A year later, in 1696, G. Leibniz showed that it can be reduced to a linear equation by a change of variable.

Here is an example of a Bernoulli equation.

```
In[1]:= eqn = y'[x] + 11 x * y[x] == x^3 * y[x]^3
```

```
Out[1]= 11 x y[x] + y'[x] == x^3 y[x]^3
```

```
In[2]:= sol = DSolve[eqn, y, x]
```

$$\text{Out[2]= } \left\{\left\{y \to \text{Function}\left[\{x\}, -\frac{11}{\sqrt{1 + 11\, x^2 + 121\, e^{11 x^2} C[1]}}\right]\right\}, \left\{y \to \text{Function}\left[\{x\}, \frac{11}{\sqrt{1 + 11\, x^2 + 121\, e^{11 x^2} C[1]}}\right]\right\}\right\}$$

This verifies that the solution is correct.

```
In[3]:= eqn /. sol // Simplify
```

```
Out[3]= {True, True}
```

In general, the solution to a Bernoulli equation will consist of $n-1$ branches, where n is the degree of $y(x)$ in the equation.

Here is an example of a Bernoulli equation with $n = 5$. The solution has four branches.

```
In[4]:= DSolve[3 x * y'[x] - 7 x * Log[x] * y[x]^5 - y[x] == 0, y[x], x]
```

$$\text{Out[4]}= \left\{\left\{y[x] \to -\frac{7^{1/4}\, x^{1/3}}{\left(12\, x^{7/3} + 7\, C[1] - 28\, x^{7/3}\, \text{Log}[x]\right)^{1/4}}\right\}, \left\{y[x] \to -\frac{i\, 7^{1/4}\, x^{1/3}}{\left(12\, x^{7/3} + 7\, C[1] - 28\, x^{7/3}\, \text{Log}[x]\right)^{1/4}}\right\},\right.$$
$$\left.\left\{y[x] \to \frac{i\, 7^{1/4}\, x^{1/3}}{\left(12\, x^{7/3} + 7\, C[1] - 28\, x^{7/3}\, \text{Log}[x]\right)^{1/4}}\right\}, \left\{y[x] \to \frac{7^{1/4}\, x^{1/3}}{\left(12\, x^{7/3} + 7\, C[1] - 28\, x^{7/3}\, \text{Log}[x]\right)^{1/4}}\right\}\right\}$$

Riccati Equations

A Riccati equation is a first-order equation of the form

$$y'(x) = f(x) + g(x)\, y(x) + h(x)\, y(x)^2.$$

This equation was used by Count Riccati of Venice (1676-1754) to help in solving second-order ordinary differential equations.

Solving Riccati equations is considerably more difficult than solving linear ODEs.

Here is a simple Riccati equation for which the solution is available in closed form.

```
In[1]:= DSolve[y'[x] + (2 / x^2) - 3 y[x]^2 == 0, y[x], x] // Simplify
```

$$\text{Out[1]}= \left\{\left\{y[x] \to -\frac{3\, x^5 - 2\, C[1]}{3\, x^6 + 3\, x\, C[1]}\right\}\right\}$$

Any Riccati equation can be transformed to a second-order linear ODE. If the latter can be solved explicitly, then a solution for the Riccati equation can be derived.

Here is an example of a Riccati equation and the corresponding second-order ODE, which is Legendre's equation.

```
In[2]:= DSolve[u'[x] == ((2 x) / (1 - x^2)) * u[x] - ((15 / 4) / (1 - x^2)) - u[x]^2, u[x], x] //
         Simplify
```

$$\text{Out[2]}= \left\{\left\{u[x] \to \left(5\left(-x\, C[1]\, \text{LegendreP}\left[\frac{3}{2}, x\right] + C[1]\, \text{LegendreP}\left[\frac{5}{2}, x\right] - x\, \text{LegendreQ}\left[\frac{3}{2}, x\right] + \text{LegendreQ}\left[\frac{5}{2}, x\right]\right)\right) \Big/ \right.$$
$$\left.\left(2\left(-1 + x^2\right)\left(C[1]\, \text{LegendreP}\left[\frac{3}{2}, x\right] + \text{LegendreQ}\left[\frac{3}{2}, x\right]\right)\right)\right\}\right\}$$

```
In[3]:= DSolve[(1 - x^2) * y''[x] - 2 x * y'[x] + (15 / 4) * y[x] == 0, y[x], x]
```

$$\text{Out[3]}= \left\{\left\{y[x] \to C[1]\, \text{LegendreP}\left[\frac{3}{2}, x\right] + C[2]\, \text{LegendreQ}\left[\frac{3}{2}, x\right]\right\}\right\}$$

Finally, consider the following Riccati equation. It is integrable because the sum of the coefficients of the terms on the right-hand side is 0.

```
In[4]:= eqn = y'[x] == 3 x + 5 * y[x] - (3 x + 5) * y[x]^2;

In[5]:= RightHandSideCoeffs = {3 x, 5, -(3 x + 5)};

In[6]:= Total[RightHandSideCoeffs]
Out[6]= 0
```

```
In[7]:= sol = DSolve[eqn, y, x]
```

$$\text{Out[7]= } \left\{\left\{y \to \text{Function}\left[\{x\}, 1 + \frac{e^{-5x-3x^2}}{C[1] + \frac{1}{12}\left(-6\, e^{-x(5+3x)} + 5\, e^{25/12}\sqrt{3\pi}\,\text{Erf}\left[\frac{5+6x}{2\sqrt{3}}\right]\right)}\right]\right\}\right\}$$

This verifies the solution.

```
In[8]:= eqn /. sol // Simplify
Out[8]= {True}
```

Exact Equations

Here is an example of an exact ODE.

```
In[1]:= P[x_, y_] := -(5 x^2 - 2 y^2 + 11)

In[2]:= Q[x_, y_] := (Sin[y] + 4 x * y + 3)

In[3]:= Simplify[D[P[x, y], y] - D[Q[x, y], x]]
Out[3]= 0
```

```
In[4]:= eqn = y'[x] == -P[x, y[x]] / Q[x, y[x]]
```

$$\text{Out[4]= } y'[x] == \frac{11 + 5x^2 - 2\,y[x]^2}{3 + \text{Sin}[y[x]] + 4\,x\,y[x]}$$

```
In[5]:= sol = DSolve[eqn, y[x], x]
```

```
Solve::tdep :
  The equations appear to involve the variables to be solved for in an essentially non-algebraic way. >>
```

$$\text{Out[5]= } \text{Solve}\left[-11x - \frac{5x^3}{3} - \text{Cos}[y[x]] + 3\,y[x] + 2\,x\,y[x]^2 == C[1], y[x]\right]$$

This verifies the solution.

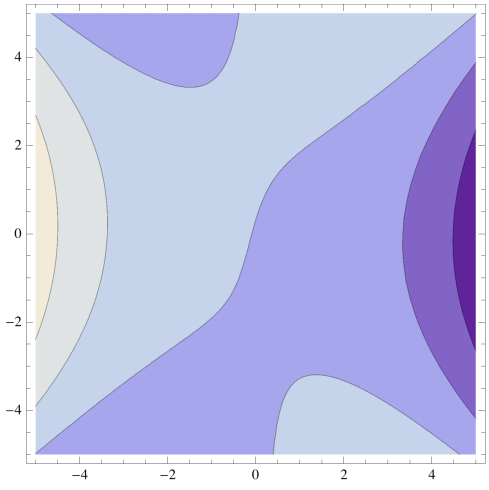
```
In[6]:= Solve[D[sol[[1]], x], y'[x]] // Simplify
```

$$\text{Out[6]}= \left\{\left\{y'[x] \to \frac{11 + 5\,x^2 - 2\,y[x]^2}{3 + \text{Sin}[y[x]] + 4\,x\,y[x]}\right\}\right\}$$

Here is a contour plot of the solution.

```
In[7]:= ContourPlot[Evaluate[sol[[1, 1]] /. {y[x] → y}], {x, -5, 5}, {y, -5, 5}]
```

Out[7]=



If an equation is not exact, it may be possible to find an integrating factor (a multiplier for the functions `P` and `Q`, defined previously) that converts the equation into exact form. `DSolve` tries a variety of techniques to automatically find integrating factors in such situations.

Clairaut Equations

A Clairaut equation is a first-order equation of the form

$$y(x) = x\,y'(x) + f(y'(x)).$$

A remarkable feature of this nonlinear equation is that its general solution has a very simple form.

This is an example of a Clairaut equation. The warning message from `Solve` can be ignored. It is given because `DSolve` first tries to find an expression for `y'[x]` from the given ODE.

```
In[1]:= sol = DSolve[y[x] == x * y'[x] + y'[x]^2 + Exp[y'[x]], y[x], x]
```

```
Solve::tdep :
  The equations appear to involve the variables to be solved for in an essentially non-algebraic way. ≫
```

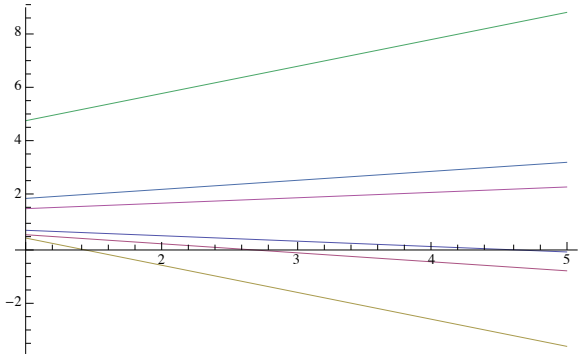
```
Out[1]= {{y[x] → e^C[1] + x C[1] + C[1]^2}}
```

The general solution to Clairaut equations is simply a family of straight lines.

This plots the solution for several values of `C[1]`.

```
In[2]:= Plot[Evaluate[Table[y[x] /. sol /. {C[1] → 1 / k}, {k, -5, 5, 2}]], {x, 1, 5}]
```

Out[2]=



Abel Equations

An Abel ODE is a first-order equation of the form

$$y'(x) = f(x) + g(x)\,y(x) + h(x)\,y(x)^2 + k(x)\,y(x)^3.$$

This equation arose in the context of the studies of Niels Henrik Abel on the theory of elliptic functions, and represents a natural generalization of the Riccati equation.

Associated with any Abel ODE is a sequence of expressions that is built from the coefficients of the equation $\{f_0, f_1, f_2, f_3\}$ and invariant under certain coordinate transformations of the independent variable and the dependent variable. These *invariants* characterize each equation and can be used for identifying integrable classes of Abel ODEs. In particular, Abel ODEs with zero or constant invariants can be integrated easily and constitute an important integrable class of these equations.

Here is the construction of a particular invariant with value 0 and the solution of the corresponding Abel ODE.

```
In[1]:= f0 = 0; f1 = 1/x; f2 = -3; f3 = x;

In[2]:= Invariant = f0 f3^2 + 1/3 (2 f2^3/9 - f1 f3 f2 - ∂x f3 f2 + f3 ∂x f2)

Out[2]= 0

In[3]:= AbelODE = y'[x] == f0 + f1 y[x] + f2 y[x]^2 + f3 y[x]^3

Out[3]= y'[x] == y[x]/x - 3 y[x]^2 + x y[x]^3

In[4]:= sol = DSolve[AbelODE, y, x]

Out[4]= {{y → Function[{x}, 1/x - 1/(x^2 Sqrt[1/x^2 + C[1]])]}, {y → Function[{x}, 1/x + 1/(x^2 Sqrt[1/x^2 + C[1]])]}}

In[5]:= AbelODE /. sol // Simplify

Out[5]= {True, True}

In[6]:= Clear[f0, f1, f2, f3]
```

Another important class of integrable Abel ODEs are those that can be reduced to inverse linear first-order ODEs using a nonlinear coordinate transformation.

This Abel ODE is solved by transforming it to an inverse linear first-order ODE. The `ExpIntegralEi` term in the solution to this equation comes from solving the linear ODE.

```
In[7]:= DSolve[y'[x] == y[x]^3 - x y[x]^2/(x - 1), y[x], x]

InverseFunction::ifun: Inverse functions are being used. Values may be lost for multivalued inverses. >>

InverseFunction::ifun: Inverse functions are being used. Values may be lost for multivalued inverses. >>

Solve::tdep:
  The equations appear to involve the variables to be solved for in an essentially non-algebraic way. >>

Out[7]= Solve[E^(1 - x + 1/y[x])/(-1 + x) + C[1] + ExpIntegralEi[1 - x + 1/y[x]] == 0, y[x]]
```

Another important class of integrable Abel ODEs consists of those that can be transformed to an inverse Riccati equation. Since Riccati equations can be transformed to second-order linear ODEs, the solutions for this class are usually given in terms of special functions such as `AiryAi` and `BesselJ`.

This Abel ODE is solved by reducing it to an inverse Riccati equation.

```
In[8]:= AbelODE = y'[x] == y[x]^3/(8 x^2) - y[x]^2
```

$$\text{Out[8]= } y'[x] == -y[x]^2 + \frac{y[x]^3}{8\,x^2}$$

```
In[9]:= sol = DSolve[AbelODE, y[x], x]
```

```
InverseFunction::ifun: Inverse functions are being used. Values may be lost for multivalued inverses. >>

InverseFunction::ifun: Inverse functions are being used. Values may be lost for multivalued inverses. >>

InverseFunction::ifun: Inverse functions are being used. Values may be lost for multivalued inverses. >>

General::stop: Further output of InverseFunction::ifun will be suppressed during this calculation. >>

Solve::tdep:
  The equations appear to involve the variables to be solved for in an essentially non-algebraic way. >>
```

$$\text{Out[9]= Solve}\left[C[1] + \left(\text{AiryAiPrime}\left[-\frac{(-1)^{2/3}}{2\times 2^{1/3}\,x} + \left(-(-2)^{1/3}\,x + \frac{(-2)^{1/3}}{y[x]}\right)^2\right] + \text{AiryAi}\left[-\frac{(-1)^{2/3}}{2\times 2^{1/3}\,x} + \left(-(-2)^{1/3}\,x + \frac{(-2)^{1/3}}{y[x]}\right)^2\right]\left(-(-2)^{1/3}\,x + \frac{(-2)^{1/3}}{y[x]}\right)\right) \Big/ \left(\text{AiryBiPrime}\left[-\frac{(-1)^{2/3}}{2\times 2^{1/3}\,x} + \left(-(-2)^{1/3}\,x + \frac{(-2)^{1/3}}{y[x]}\right)^2\right] + \text{AiryBi}\left[-\frac{(-1)^{2/3}}{2\times 2^{1/3}\,x} + \left(-(-2)^{1/3}\,x + \frac{(-2)^{1/3}}{y[x]}\right)^2\right]\left(-(-2)^{1/3}\,x + \frac{(-2)^{1/3}}{y[x]}\right)\right) == 0,\ y[x]\right]$$

This verifies the solution.

```
In[10]:= Solve[D[sol[[1]], x], y'[x]] // FullSimplify
```

$$\text{Out[10]= } \left\{\left\{y'[x] \to \frac{1}{8}\,y[x]^2\left(-8 + \frac{y[x]}{x^2}\right)\right\}\right\}$$

The Abel ODEs considered so far are said to be of the *first kind.* Abel ODEs of the *second kind* are given by the following general formula.

$$y'(x) = \frac{f(x) + g(x)\,y(x) + h(x)\,y(x)^2 + k(x)\,y(x)^3}{a(x) + b(x)\,y(x)}$$

An Abel ODE of the second kind can be converted to an equation of the first kind with a coordinate transformation. Thus, the solution methods for this kind of Abel ODE are identical to the methods for equations of the first kind.

Here is the solution for an Abel ODE of the second kind.

```
In[11]:= sol = DSolve[y'[x] == (-2 x/9 + x^3 + y[x])/y[x], y[x], x]
```

```
Solve::tdep :
  The equations appear to involve the variables to be solved for in an essentially non-algebraic way. >>

Solve::tdep :
  The equations appear to involve the variables to be solved for in an essentially non-algebraic way. >>
```

$$\text{Out[11]= } \text{Solve}\left[\text{C}[1] == \left(-1 + \left(\frac{\sqrt{2}}{3\,x} - \frac{\sqrt{2}\ \text{y}[x]}{x^2}\right)^2\right)^{1/4} \left(-\frac{3\,x}{\sqrt{2}} - \frac{\text{Hypergeometric2F1}\left[\frac{1}{2}, \frac{3}{4}, \frac{3}{2}, \left(\frac{\sqrt{2}}{3\,x} - \frac{\sqrt{2}\ \text{y}[x]}{x^2}\right)^2\right]\left(\frac{\sqrt{2}}{3\,x} - \frac{\sqrt{2}\ \text{y}[x]}{x^2}\right)}{2\left(1 - \left(\frac{\sqrt{2}}{3\,x} - \frac{\sqrt{2}\ \text{y}[x]}{x^2}\right)^2\right)^{1/4}}\right), \text{y}[x]\right]$$

This verifies the solution.

```
In[12]:= Solve[D[sol[[1]], x], y'[x]] // FullSimplify
```

$$\text{Out[12]= } \left\{\left\{\text{y}'[x] \to 1 + \frac{-\frac{2\,x}{9} + x^3}{\text{y}[x]}\right\}\right\}$$

Chini Equations

Chini equations are a generalization of Abel and Riccati equations.

This solves a Chini equation.

```
In[1]:= DSolve[y'[x] == 5 * y[x]^4 + 3 * x^(-4 / 3), y[x], x]
```

```
Solve::tdep :
  The equations appear to involve the variables to be solved for in an essentially non-algebraic way. >>

Solve::tdep :
  The equations appear to involve the variables to be solved for in an essentially non-algebraic way. >>
```

$$\text{Out[1]= } \text{Solve}\left[-45\ \text{RootSum}\left[-45 + 3^{1/4}\,5^{3/4}\,\#1 - 45\,\#1^4\ \&,\ \frac{\text{Log}\left[-\#1 + \left(\frac{5}{3}\right)^{1/4}\left(x^{4/3}\right)^{1/4}\text{y}[x]\right]}{3^{1/4}\,5^{3/4} - 180\,\#1^3}\ \&\right] == \text{C}[1] + \frac{3^{3/4}\,5^{1/4}\left(x^{4/3}\right)^{1/4}\text{Log}[x]}{x^{1/3}}, \text{y}[x]\right]$$

Linear Second-Order ODEs

Overview of Linear Second-Order ODEs

Solving linear first-order ODEs is straightforward and only requires the use of a suitable integrating factor. In sharp contrast, there are a large number of methods available for handling linear second-order ODEs, but the solution to the general equation belonging to this class is still not available. Therefore the linear case is discussed in detail before moving on to nonlinear second-order ODEs.

The general *linear second-order ODE* has the form

$$y''(x) + P(x)\,y'(x) + Q(x)\,y(x) = R(x).$$

Here, $P(x)$, $Q(x)$ and $R(x)$ are arbitrary functions of x. The term "linear" refers to the fact that the degree of each term in $y(x)$, $y'(x)$ and $y''(x)$ is 1. (Thus, terms like $y(x)^2$ or $y(x)\,y''(x)$ would make the equation nonlinear.)

Linear Second-Order Equations with Constant Coefficients

The simplest type of linear second-order ODE is one with *constant coefficients*.

This linear second-order ODE has constant coefficients.

```
In[1]:= sol = DSolve[y''[x] + 5 * y'[x] - 6 y[x] == 0, y, x]
Out[1]= {{y → Function[{x}, e^(-6 x) C[1] + e^x C[2]]}}
```

Notice that the general solution is a linear combination of two exponential functions. The arbitrary constants `C[1]` and `C[2]` can be varied to produce particular solutions.

This is one particular solution to the equation.

```
In[2]:= sol1 = y[x] /. sol[[1]] /. {C[1] → 2, C[2] → 3}
Out[2]= 2 e^(-6 x) + 3 e^x
```

The exponents -6 and 1 in the *basis* $\left\{e^{-6x}, e^{x}\right\}$ are obtained by solving the associated quadratic equation. This quadratic equation is called the *auxiliary* or *characteristic* equation.

This solves the auxiliary equation.

```
In[3]:= Solve[m^2 + 5 m - 6 == 0, m]
Out[3]= {{m → -6}, {m → 1}}
```

The roots are real and distinct in this case. There are two other cases of interest: real and equal roots, and imaginary roots.

This example has real and equal roots.

```
In[4]:= sol = DSolve[y''[x] - 6 y'[x] + 9 y[x] == 0, y, x]
Out[4]= {{y → Function[{x}, e^(3 x) C[1] + e^(3 x) x C[2]]}}

In[5]:= sol2 = y[x] /. sol[[1]] /. {C[1] → 2, C[2] → 3}
Out[5]= 2 e^(3 x) + 3 e^(3 x) x
```

This example has roots with nonzero imaginary parts.

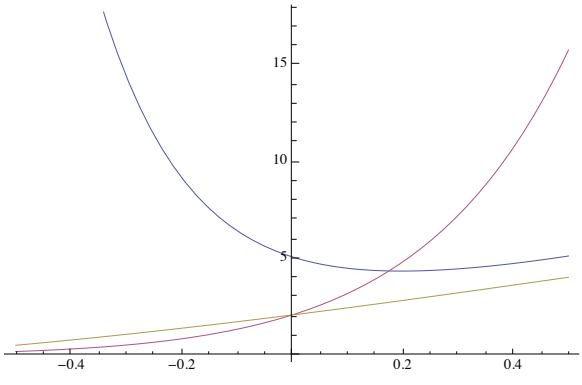
```
In[6]:= sol = DSolve[y''[x] - y'[x] + y[x] == 0, y, x]
Out[6]= {{y → Function[{x}, e^(x/2) C[1] Cos[Sqrt[3] x/2] + e^(x/2) C[2] Sin[Sqrt[3] x/2]]}}

In[7]:= sol3 = y[x] /. sol[[1]] /. {C[1] → 2, C[2] → 3}
Out[7]= 2 e^(x/2) Cos[Sqrt[3] x/2] + 3 e^(x/2) Sin[Sqrt[3] x/2]
```

Here is a plot of the three solutions.

```
In[8]:= Plot[{sol1, sol2, sol3}, {x, -0.5, 0.5}]
```

Out[8]=



Euler and Legendre Equations

An *Euler equation* has the general form

$$x^2\, y''(x) + a\, x\, y'(x) + b\, y(x) = 0.$$

Euler equations can be solved by transforming them to equations with constant coefficients.

This is an example of an Euler equation.

```
In[1]:= DSolve[x^2 * y''[x] + 5 * x * y'[x] + 6 * y[x] == 0, y[x], x]
```

$$\text{Out[1]= } \left\{\left\{y[x] \to \frac{C[2]\,\text{Cos}\left[\sqrt{2}\,\text{Log}[x]\right]}{x^2} + \frac{C[1]\,\text{Sin}\left[\sqrt{2}\,\text{Log}[x]\right]}{x^2}\right\}\right\}$$

The *Legendre linear equation* is a generalization of the Euler equation. It is an ODE of the form

$$(c\, x + d)^2\, y''(x) + a\,(c\, x + d)\, y'(x) + b\, y(x) = 0.$$

Here is an example of a Legendre linear equation.

```
In[2]:= DSolve[(3 x + 1)^2 * y''[x] + 5 * (3 x + 1) * y'[x] + 6 * y[x] == 0, y[x], x]
```

$$\text{Out[2]= } \left\{\left\{y[x] \to \frac{C[2]\,\text{Cos}\left[\frac{1}{3}\sqrt{5}\,\text{Log}[1 + 3\,x]\right]}{(1 + 3\,x)^{1/3}} + \frac{C[1]\,\text{Sin}\left[\frac{1}{3}\sqrt{5}\,\text{Log}[1 + 3\,x]\right]}{(1 + 3\,x)^{1/3}}\right\}\right\}$$

Exact Linear Second-Order Equations

A linear second-order ordinary differential equation

$$a_0(x)\, y''(x) + a_1(x)\, y'(x) + a_2(x)\, y(x) = 0$$

is said to be *exact* if

$$a_0''(x) - a_1'(x) + a_2(x) = 0.$$

An exact linear second-order ODE is solved by reduction to a linear first-order ODE.

Here is an example. The appearance of the unevaluated integral in the solution is explained here.

```
In[1]:= a0 = 1;
```

```
In[2]:= a1 = Log[x];
```

```
In[3]:= a2 = 1 / x;

In[4]:= eqn = a0 * y''[x] + a1 * y'[x] + a2 * y[x] == 0;

In[5]:= conditionforexactness = (D[a0, {x, 2}] - D[a1, x] + a2 == 0)
Out[5]= True

In[6]:= sol = DSolve[eqn, y, x]
```

$$\text{Out[6]= } \left\{\left\{y \to \text{Function}\left[\{x\}, e^{x-x \text{Log}[x]} C[2] + e^{x-x \text{Log}[x]} \int_1^x e^{-K[1]+K[1] \text{Log}[K[1]]} C[1] \, dK[1]\right]\right\}\right\}$$

This verifies the solution.

```
In[7]:= eqn /. sol // FullSimplify
Out[7]= {True}

In[8]:= Clear[a0, a1, a2]
```

Linear Second-Order Equations with Solutions Involving Special Functions

`DSolve` can find solutions for most of the standard linear second-order ODEs that occur in applied mathematics.

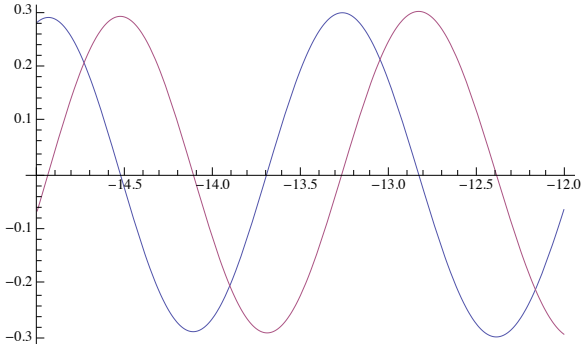
Here is the solution for *Airy's equation*.

```
In[1]:= DSolve[y''[x] - x * y[x] == 0, y[x], x]
Out[1]= {{y[x] → AiryAi[x] C[1] + AiryBi[x] C[2]}}
```

Here is a plot that shows the oscillatory behavior of the Airy functions for large negative values of x.

```
In[2]:= Plot[{AiryAi[x], AiryBi[x]}, {x, -15, -12}]
```

Out[2]=



The solution to this equation is given in terms of the derivatives of the Airy functions, `AiryAiPrime` and `AiryBiPrime`.

```
In[3]:= DSolve[ ((a * x + b )) * y''[x] - a * y'[x] - (a * (a * x + b))^2 * y[x] == 0, y, x]
Out[3]= {{y → Function[{x}, AiryAiPrime[b + a x] C[1] + AiryBiPrime[b + a x] C[2]]}}
```

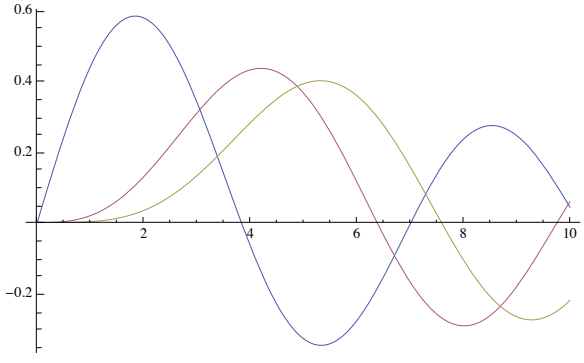
Here is the solution for *Bessel's equation* with $n = 4$. Note that the solution is given in terms of Bessel functions of the first kind, `BesselJ`, as well as those of the second kind, `BesselY`.

```
In[4]:= DSolve[x^2 * y''[x] + x * y'[x] + (x^2 - 16) * y[x] == 0, y[x], x]
Out[4]= {{y[x] → BesselJ[4, x] C[1] + BesselY[4, x] C[2]}}
```

Here is a plot of the `BesselJ` functions for specific values of n.

```
In[5]:= Plot[{BesselJ[1, x], BesselJ[3, x], BesselJ[4, x]}, {x, 0, 10}]
```

Out[5]=



Here is the general solution for *Legendre's equation* with $n = 7$.

```
In[6]:= DSolve[16 (1 - x^2) * y''[x] - 32 x * y'[x] + 21 * y[x] == 0, y[x], x]
Out[6]= {{y[x] → C[1] LegendreP[3/4, x] + C[2] LegendreQ[3/4, x]}}
```

These special functions can be expressed in terms of elementary functions for certain values of their parameters. *Mathematica* performs this conversion automatically wherever it is possible.

These are some of these expressions that are automatically converted.

```
In[7]:= {BesselJ[1 / 2, x], LegendreP[4, x], HermiteH[5, x]}
Out[7]= {Sqrt[2/π] Sin[x] / Sqrt[x], 1/8 (3 - 30 x^2 + 35 x^4), 120 x - 160 x^3 + 32 x^5}
```

As a result of these conversions, the solutions of certain ODEs can be partially expressed in terms of elementary functions. Hermite's equation is one such ODE.

Here is the solution for Hermite's equation with arbitrary n.

```
In[8]:= DSolve[y''[x] - 2 x * y'[x] + 2 n * y[x] == 0, y[x], x]
```

$$\text{Out[8]= } \left\{\left\{y[x] \to C[1]\ \text{HermiteH}[n, x] + C[2]\ \text{Hypergeometric1F1}\left[-\frac{n}{2}, \frac{1}{2}, x^2\right]\right\}\right\}$$

With n set to 5, the solution is given in terms of polynomials, exponentials, and `Erfi`.

```
In[9]:= Collect[Simplify[PowerExpand[
          y[x] /. DSolve[y''[x] - 2 * x * y'[x] + 10 * y[x] == 0, y[x], x][[1]]]], {C[1], C[2]}]
```

$$\text{Out[9]= } \frac{1}{8}\left(960\,x - 1280\,x^3 + 256\,x^5\right) C[1] + \frac{1}{8} C[2]\left(8\,e^{x^2} - 18\,e^{x^2}x^2 + 4\,e^{x^2}x^4 + \sqrt{\pi}\,x\left(-15 + 20\,x^2 - 4\,x^4\right)\text{Erfi}[x]\right)$$

Linear Second-Order ODEs with Rational Coefficients

The hypergeometric functions play a unifying role in mathematical analysis since many important functions, such as the Bessel functions and Legendre functions, are special cases of them. Each hypergeometric function is associated with a linear ODE having rational coefficients.

Here is the ODE for the `Hypergeometric2F1` function.

```
In[1]:= DSolve[(x^2 - x) * y''[x] + ((a + b + 1) * x - c) * y'[x] + b * a * y[x] == 0, y[x], x]
```

$$\text{Out[1]= } \left\{\left\{y[x] \to C[1]\ \text{Hypergeometric2F1}[a, b, c, x] + (-1)^{1-c}\,x^{1-c}\,C[2]\ \text{Hypergeometric2F1}[1 + a - c, 1 + b - c, 2 - c, x]\right\}\right\}$$

`DSolve` can solve a large class of second-order linear ODEs by reducing them to the ODEs for hypergeometric functions. The reduction involves coordinate transformations of both the independent and dependent variables.

This equation is *equivalent* to the ODE for `Hypergeometric2F1`.

```
In[2]:= sol = DSolve[64 x^2 (x - 1)^2 y''[x] + 32 x (x - 1) (3 x - 1) y'[x] + (5 x - 21) y[x] == 0, y, x]
```

$$\text{Out[2]= } \left\{\left\{y \to \text{Function}\left[\{x\}, \frac{e^{\frac{1}{4}(-2\,\text{Log}[-1+x]-\text{Log}[x])}\,C[1]\ \text{Hypergeometric2F1}\left[-\frac{7}{8}, -\frac{3}{8}, -\frac{1}{4}, x\right]}{x^{1/8}} - (-1)^{1/4}\,e^{\frac{1}{4}(-2\,\text{Log}[-1+x]-\text{Log}[x])}\,x^{9/8}\,C[2]\ \text{Hypergeometric2F1}\left[\frac{3}{8}, \frac{7}{8}, \frac{9}{4}, x\right]\right]\right\}\right\}$$

This verifies the solution using numerical values.

```
In[3]:= 64 x^2 (x - 1)^2 y''[x] + 32 x (x - 1) (3 x - 1) y'[x] + (5 x - 21) y[x] /. sol[[1]] /.
          {x → RandomComplex[]} // Simplify // Chop
```

```
Out[3]= 0
```

Solutions to this equation are returned in terms of `HypergeometricU` (the confluent hypergeometric function) and `LaguerreL`. This example appears on (equation 2.16, page 403 of [K59]).

```
In[4]:= sol = Simplify[PowerExpand[y[x] /. DSolve[
          Derivative[2][y][x] + (a * x^(2 * c) + b * x^(c - 1)) * y[x] == 0, y, x][[1]]]]
```

$$\text{Out[4]}= 2^{\frac{c}{2+2c}}\, e^{-\frac{\sqrt{a}\,x^{1+c}}{\sqrt{-(1+c)^2}}} \left(C[1]\,\text{HypergeometricU}\left[-\frac{\frac{i\,b}{\sqrt{a}}-c}{2+2c},\ \frac{c}{1+c},\ \frac{2\sqrt{a}\,x^{1+c}}{\sqrt{-(1+c)^2}}\right] + C[2]\,\text{LaguerreL}\left[\frac{\frac{i\,b}{\sqrt{a}}-c}{2+2c},\ -\frac{1}{1+c},\ \frac{2\sqrt{a}\,x^{1+c}}{\sqrt{-(1+c)^2}}\right]\right)$$

The ODEs for special functions have been studied since the eighteenth century. During the last thirty years, powerful algorithms have been developed for systematically solving ODEs with rational coefficients. An important algorithm of this type is *Kovacic's algorithm*, a decision procedure that either generates a solution for the given ODE in terms of Liouvillian functions or proves that the given ODE does not have a Liouvillian solution.

This equation is solved using Kovacic's algorithm.

```
In[5]:= DSolve[x * y''[x] + (10 x^3 - 1) * y'[x] + 5 x^2 (5 x^3 + 1) * y[x] == 0, y[x], x]
```

$$\text{Out[5]}= \left\{\left\{y[x] \to e^{-\frac{5x^3}{3}}\,C[1] + \frac{1}{2}\,e^{-\frac{5x^3}{3}}\,x^2\,C[2]\right\}\right\}$$

The solution returned from Kovacic's algorithm may occasionally include functions such as `ExpIntegralEi` or an unevaluated integral of elementary functions because, while it is easy to find a second solution for a second-order linear ODE once one solution is known, the integral involved in finding the second solution may be hard to evaluate explicitly.

The solution to this equation is obtained using Kovacic's algorithm. It includes `ExpIntegralEi`.

```
In[6]:= DSolve[4 x * y''[x] + (7 x + 12) * y'[x] + 21 y[x] == 0, y[x], x]
```

$$\text{Out[6]}= \left\{\left\{y[x] \to e^{-7x/4}\,C[1] - \frac{e^{-7x/4}\,C[2]\left(16\,e^{7x/4} + 28\,e^{7x/4}\,x - 49\,x^2\,\text{ExpIntegralEi}\left[\frac{7x}{4}\right]\right)}{32\,x^2}\right\}\right\}$$

In general, the solutions for linear ODEs with rational coefficients and order greater than one can be given in terms of `DifferentialRoot` objects. This is similar to the representation for solutions of polynomial equations in terms of `Root`.

The solution to this equation is given in terms of `DifferentialRoot`.

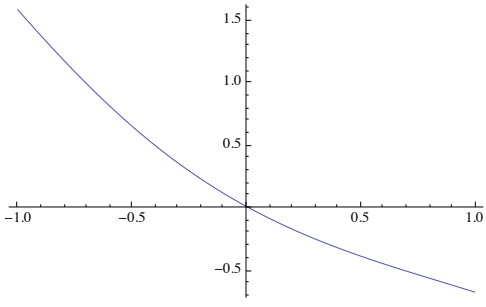
```
In[7]:= sol = DSolve[y''[x] - x^2 y'[x] - y[x] - 1 == 0 && y[0] == 0 && y'[0] == -1, y, x]
Out[7]= {{y → Function[{x},
     DifferentialRoot[Function[{y, x}, {-1 - y[x] - x^2 y'[x] + y''[x] == 0, y[0] == 0, y'[0] == -1}]][x]]}}
```

The solution may be evaluated and plotted in the usual way.

```
In[8]:= Table[y[x] /. sol[[1]], {x, -1, 1, 0.4}]
Out[8]= {1.57269, 0.806671, 0.221252, -0.181388, -0.459045, -0.685918}
```

```
In[9]:= Plot[y[x] /. sol[[1]], {x, -1, 1}]
```

Out[9]=



Equations with Non-Rational Coefficients

The ODEs that arise in practical applications often have non-rational coefficients. In such cases, `DSolve` attempts to convert the equation into one with rational coefficients using a suitable coordinate transformation.

Here is an equation that has `Exp[x]` as a coefficient. It is solved by transforming it to Bessel's equation.

```
In[1]:= DSolve[y''[x] - Exp[5 x] * y[x] == 0, y, x]
Out[1]= {{y → Function[{x}, BesselI[0, 2 Sqrt[e^(5 x)]/5] C[1] + 2 BesselK[0, 2 Sqrt[e^(5 x)]/5] C[2]]}}
```

This equation (equation 2.437, page 507 of [K59]) has trigonometric coefficients. The solution is given in terms of elementary functions.

```
In[2]:= DSolve[y''[x] * Sin[x] * Cos[x]^2 -
    y'[x] * (3 * Sin[x]^2 + 1) * Cos[x] - y[x] * Sin[x]^3 == 0, y, x]
Out[2]= {{y → Function[{x}, C[2] Cos[x]^(-3/2 - Sqrt[13]/2) + C[1] Cos[x]^(-3/2 + Sqrt[13]/2)]}}
```

Here is an equation with a hyperbolic function in the coefficient of $y[x]$. The solution is given in terms of Legendre functions.

```
In[3]:= DSolve[y''[x] + (k^2 + 2 * Sech[x]^2) y[x] == 0, y[x], x]
```

```
Out[3]= {{y[x] → C[1] LegendreP[1, i k, Tanh[x]] + C[2] LegendreQ[1, i k, Tanh[x]]}}
```

The solution to this equation is given in terms of `HypergeometricU` and `LaguerreL`.

```
In[4]:= expr = y''[x] + (-d + d (1 - Exp[-b x])^2) y[x] - l y[x];
```

```
In[5]:= sol = DSolve[expr == 0, y, x]
```

```
Out[5]= {{y → Function[{x},
     e^((-i Sqrt[d] e^(-b x) + Sqrt[l] Log[e^(-b x)])/b) C[1] HypergeometricU[-(-b + 2 i Sqrt[d] - 2 Sqrt[l])/(2 b), 1 + 2 Sqrt[l]/b, 2 i Sqrt[d] e^(-b x)/b] +
     e^((-i Sqrt[d] e^(-b x) + Sqrt[l] Log[e^(-b x)])/b) C[2] LaguerreL[(-b + 2 i Sqrt[d] - 2 Sqrt[l])/(2 b), 2 Sqrt[l]/b, 2 i Sqrt[d] e^(-b x)/b]]}}
```

This verifies the solution using random values of x, b, d, and l.

```
In[6]:= expr /. sol[[1]] /. {x → RandomComplex[], b → RandomComplex[],
      d → RandomComplex[], l → RandomComplex[]} // Simplify // Chop
```

```
Out[6]= 0
```

Inhomogeneous Linear Second-Order Equations

If the given second-order ODE is inhomogeneous, `DSolve` applies the *method of variation of parameters* to return a solution for the problem.

This solves an inhomogeneous linear second-order ODE. The solution is composed of two parts: the first part is the general solution to the homogeneous equation, and the second part is a particular solution to the inhomogeneous equation.

```
In[1]:= sol = DSolve[x^2 y''[x] + y[x] == x^2, y[x], x]
```

```
Out[1]= {{y[x] → Sqrt[x] C[1] Cos[1/2 Sqrt[3] Log[x]] +
     Sqrt[x] C[2] Sin[1/2 Sqrt[3] Log[x]] + 1/3 (x^2 Cos[1/2 Sqrt[3] Log[x]]^2 + x^2 Sin[1/2 Sqrt[3] Log[x]]^2)}}
```

This solves the homogeneous equation, which is an Euler equation.

```
In[2]:= DSolve[x^2 y''[x] + y[x] == 0, y[x], x]
```

```
Out[2]= {{y[x] → Sqrt[x] C[1] Cos[1/2 Sqrt[3] Log[x]] + Sqrt[x] C[2] Sin[1/2 Sqrt[3] Log[x]]}}
```

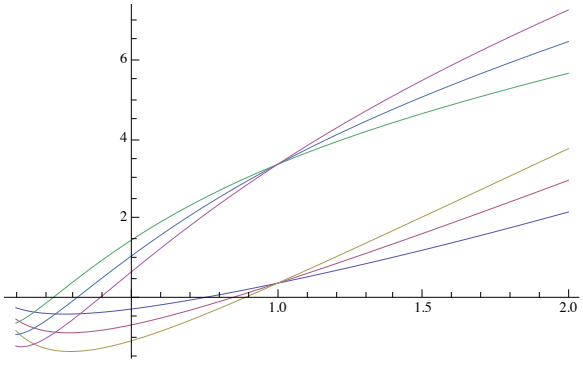
Different particular solutions can be obtained by varying the constants `C[1]` and `C[2]` in the solution.

In[3]:=
```
particularsolutions =
 Flatten[Table[y[x] /. sol /. {C[1] → i, C[2] → j}, {i, 0, 5, 3}, {j, 1, 3}]]
```

Out[3]=
$$\left\{\sqrt{x}\,\text{Sin}\left[\frac{1}{2}\sqrt{3}\,\text{Log}[x]\right]+\frac{1}{3}\left(x^2\,\text{Cos}\left[\frac{1}{2}\sqrt{3}\,\text{Log}[x]\right]^2+x^2\,\text{Sin}\left[\frac{1}{2}\sqrt{3}\,\text{Log}[x]\right]^2\right),\right.$$
$$2\sqrt{x}\,\text{Sin}\left[\frac{1}{2}\sqrt{3}\,\text{Log}[x]\right]+\frac{1}{3}\left(x^2\,\text{Cos}\left[\frac{1}{2}\sqrt{3}\,\text{Log}[x]\right]^2+x^2\,\text{Sin}\left[\frac{1}{2}\sqrt{3}\,\text{Log}[x]\right]^2\right),$$
$$3\sqrt{x}\,\text{Sin}\left[\frac{1}{2}\sqrt{3}\,\text{Log}[x]\right]+\frac{1}{3}\left(x^2\,\text{Cos}\left[\frac{1}{2}\sqrt{3}\,\text{Log}[x]\right]^2+x^2\,\text{Sin}\left[\frac{1}{2}\sqrt{3}\,\text{Log}[x]\right]^2\right),$$
$$3\sqrt{x}\,\text{Cos}\left[\frac{1}{2}\sqrt{3}\,\text{Log}[x]\right]+\sqrt{x}\,\text{Sin}\left[\frac{1}{2}\sqrt{3}\,\text{Log}[x]\right]+\frac{1}{3}\left(x^2\,\text{Cos}\left[\frac{1}{2}\sqrt{3}\,\text{Log}[x]\right]^2+x^2\,\text{Sin}\left[\frac{1}{2}\sqrt{3}\,\text{Log}[x]\right]^2\right),$$
$$3\sqrt{x}\,\text{Cos}\left[\frac{1}{2}\sqrt{3}\,\text{Log}[x]\right]+2\sqrt{x}\,\text{Sin}\left[\frac{1}{2}\sqrt{3}\,\text{Log}[x]\right]+$$
$$\frac{1}{3}\left(x^2\,\text{Cos}\left[\frac{1}{2}\sqrt{3}\,\text{Log}[x]\right]^2+x^2\,\text{Sin}\left[\frac{1}{2}\sqrt{3}\,\text{Log}[x]\right]^2\right),\ 3\sqrt{x}\,\text{Cos}\left[\frac{1}{2}\sqrt{3}\,\text{Log}[x]\right]+$$
$$\left.3\sqrt{x}\,\text{Sin}\left[\frac{1}{2}\sqrt{3}\,\text{Log}[x]\right]+\frac{1}{3}\left(x^2\,\text{Cos}\left[\frac{1}{2}\sqrt{3}\,\text{Log}[x]\right]^2+x^2\,\text{Sin}\left[\frac{1}{2}\sqrt{3}\,\text{Log}[x]\right]^2\right)\right\}$$

In[4]:=
```
Plot[Evaluate[particularsolutions], {x, 0.1, 2}]
```

Out[4]=



Nonlinear Second-Order ODEs

The general form of a nonlinear second-order ODE is

$$F(x, y, y'(x), y''(x)) = 0.$$

For simplicity, assume that the equation can be solved for the highest-order derivative $y''(x)$ to give

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

There are a few classes of nonlinear second-order ODEs for which solutions can be easily found.

The first class consists of equations that do not explicitly depend on $y(x)$; that is, equations of the form $y''(x) = f(x, y'(x))$. Such equations can be regarded as first-order ODEs in $u(x) = y'(x)$.

Here is an example of this type.

```
In[1]:= eqn = y''[x] == 5 x y'[x] + y'[x]^2; sol = DSolve[eqn, y, x]
```

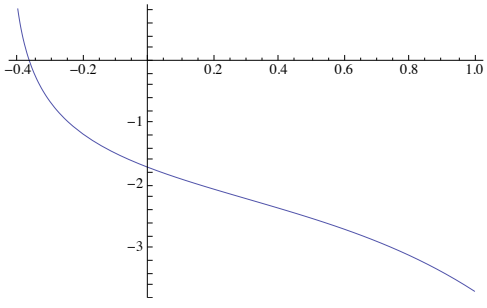
$$\text{Out[1]= } \left\{\left\{y \to \text{Function}\left[\{x\}, C[2] - \text{Log}\left[-10\, C[1] + \sqrt{10\,\pi}\ \text{Erfi}\left[\sqrt{\frac{5}{2}}\ x\right]\right]\right]\right\}\right\}$$

As in the case of linear second-order ODEs, the solution depends on two arbitrary parameters `C[1]` and `C[2]`.

Here is a plot of the solution for a specific choice of parameters.

```
In[2]:= Plot[Evaluate[y[x] /. sol /. {C[1] → -1 / 2, C[2] → -1 / 8}], {x, -0.4, 1}]
```

Out[2]=



This verifies the solution.

```
In[3]:= eqn /. sol // Simplify
Out[3]= {True}
```

The second class of easily solvable nonlinear second-order equations consists of equations that do not depend explicitly on x or $y'(x)$; that is, equations of the form $y''(x) = f(y(x))$. These equations can be reduced to first-order ODEs with independent variable y. Inverse functions are needed to give the final solution for $y(x)$.

Here is an example of this type.

```
In[4]:= DSolve[y''[x] == Exp[3 * y[x]], y[x], x]
```

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

$$\text{Out[4]}= \left\{\left\{y[x] \to \text{Log}\left[-\left(-\frac{3}{2}\right)^{1/3}\left(-C[1]+C[1]\,\text{Tanh}\left[\frac{3}{2}\sqrt{C[1]\,(x+C[2])^2}\right]^2\right)^{1/3}\right]\right\},\right.$$
$$\left\{y[x] \to \text{Log}\left[\left(\frac{3}{2}\right)^{1/3}\left(-C[1]+C[1]\,\text{Tanh}\left[\frac{3}{2}\sqrt{C[1]\,(x+C[2])^2}\right]^2\right)^{1/3}\right]\right\},$$
$$\left.\left\{y[x] \to \text{Log}\left[(-1)^{2/3}\left(\frac{3}{2}\right)^{1/3}\left(-C[1]+C[1]\,\text{Tanh}\left[\frac{3}{2}\sqrt{C[1]\,(x+C[2])^2}\right]^2\right)^{1/3}\right]\right\}\right\}$$

The third class consists of equations that do not depend explicitly on x; that is, equations of the form $y''(x) = f(y(x), y'(x))$. Once again, these equations can be reduced to first-order ODEs with independent variable y.

This example is based on (equation 6.40, page 550 of [K59]). The underlying first-order ODE is an Abel equation. The hyperbolic functions in the solution result from the automatic simplification of Bessel functions.

```
In[5]:= sol =
         y[x] /. DSolve[y''[x] == 3 * y[x] * y'[x] + (3 y[x]^2 + 4 * y[x] + 1), y[x], x][[1]] //
          Simplify
```

$$\text{Out[5]}= \left(\left(i\sqrt{6}\,\sqrt{e^{-2x}}\,\sqrt{C[1]}-6\,C[2]\right)\text{Cosh}\left[\sqrt{\frac{3}{2}}\,\sqrt{e^{-2x}}\,\sqrt{C[1]}\right]+\right.$$
$$\left.\left(-3i+2\sqrt{6}\,\sqrt{e^{-2x}}\,\sqrt{C[1]}\,C[2]\right)\text{Sinh}\left[\sqrt{\frac{3}{2}}\,\sqrt{e^{-2x}}\,\sqrt{C[1]}\right]\right)\Big/$$
$$\left(6\,C[2]\,\text{Cosh}\left[\sqrt{\frac{3}{2}}\,\sqrt{e^{-2x}}\,\sqrt{C[1]}\right]+3i\,\text{Sinh}\left[\sqrt{\frac{3}{2}}\,\sqrt{e^{-2x}}\,\sqrt{C[1]}\right]\right)$$

The fourth class consists of equations that are homogeneous in some or all of the variables x, $y(x)$, and $y''(x)$. There are several possibilities in this case, but here only the following simple example is considered.

In this equation, each term has a total degree of 2 in the variables $y[x]$, $y'[x]$, and $y''[x]$. This equation can be solved by transforming it to a first-order ODE.

```
In[6]:= DSolve[7 * y[x] * y''[x] - 11 * y'[x]^2 == 0, y[x], x]
```

$$\text{Out[6]}= \left\{\left\{y[x] \to \frac{C[2]}{(4x+7\,C[1])^{7/4}}\right\}\right\}$$

The fifth and final class of easily solvable nonlinear second-order ODEs consists of equations that are exact or can be made to be exact using an integrating factor.

Here is an example of this type, based on (equation 6.51, page 554 of [K59]).

```
In[7]:= eqn = y''[x] + y[x] * y'[x]^2 - x^2 * y'[x] == 0;

In[8]:= sol = DSolve[eqn, y[x], x]
```

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

$$\text{Out[8]= } \left\{\left\{y[x] \to -i\sqrt{2}\ \text{InverseErf}\left[\frac{i\left(3\sqrt{2\pi}\ C[2] - \frac{3^{1/3}\sqrt{2\pi}\ (-x^3)^{2/3} C[1]\ \text{Gamma}\left[\frac{1}{3}, -\frac{x^3}{3}\right]}{x^2}\right)}{3\pi}\right]\right\}\right\}$$

It is important to note that the solutions to fairly simple-looking nonlinear ODEs can be complicated. Verifying and applying the solutions in such cases is a difficult problem.

Higher-Order ODEs

Overview of Higher-Order ODEs

The general form of an ODE with order n is

$$F\left(x, y(x), y'(x), y''(x), \ldots, y^{(n)}(x)\right) = 0.$$

As in the case of second-order ODEs, such an ODE can be classified as linear or nonlinear. The general form of a linear ODE of order n is

$$a_0(x)\, y^{(n)}(x) + a_1(x)\, y^{(n-1)}(x) + \ldots + a_n(x)\, y(x) = b(x).$$

If $b(x)$ is the zero function, the equation is said to be *homogeneous*. This discussion is primarily restricted to that case.

Many methods for solving linear second-order ODEs can be generalized to linear ODEs of order n, where n is greater than 2. If the order of the ODE is not important, it is simply called a linear ODE.

Linear Higher-Order Equations with Constant Coefficients

A linear ODE with *constant coefficients* can be easily solved once the roots of the auxiliary equation (or characteristic equation) are known. Some examples of this type follow.

The characteristic equation of this ODE has real and distinct roots: 4, 1, and 7. Hence the solution is composed entirely of exponential functions.

```
In[1]:= DSolve[y'''[x] - 4 * y''[x] - 25 * y'[x] + 28 * y[x] == 0, y[x], x]
Out[1]= {{y[x] → e^(-4 x) C[1] + e^x C[2] + e^(7 x) C[3]}}
```

The characteristic equation of this ODE has two pairs of equal roots: -3 and -5. The repeated roots give rise to the *basis* of the solutions, $\{e^{3x}, x\,e^{3x}, e^{5x}, x\,e^{5x}\}$.

```
In[2]:= DSolve[y''''[x] - 16 * y'''[x] + 94 * y''[x] - 240 * y'[x] + 225 * y[x] == 0, y[x], x]
Out[2]= {{y[x] → e^(3 x) C[1] + e^(3 x) x C[2] + e^(5 x) C[3] + e^(5 x) x C[4]}}
```

The characteristic equation for this ODE has two pairs of roots with nonzero imaginary parts: $3+4i$, $3-4i$, $2+i$, and $2-i$. Hence the solution basis can be expressed with trigonometric and exponential functions.

```
In[3]:= DSolve[y''''[x] - 10 * y'''[x] + 54 * y''[x] - 130 * y'[x] + 125 * y[x] == 0, y[x], x]
Out[3]= {{y[x] → e^(2 x) C[2] Cos[x] + e^(3 x) C[4] Cos[4 x] + e^(2 x) C[1] Sin[x] + e^(3 x) C[3] Sin[4 x]}}
```

Finally, here is an example that combines all the previous kinds of solutions.

```
In[4]:= DSolve[y'''''[x] - 17 * y''''[x] +
    108 * y'''[x] - 330 * y''[x] + 488 * y'[x] - 280 * y[x] == 0, y[x], x]
Out[4]= {{y[x] → e^(2 x) C[3] + e^(2 x) x C[4] + e^(7 x) C[5] + e^(3 x) C[2] Cos[x] + e^(3 x) C[1] Sin[x]}}
```

Higher-Order Euler and Legendre Equations

An *Euler equation* is an ODE of the form

$$x^n\, y^{(n)}(x) + a_1\, x^{n-1}\, y^{(n-1)}(x) + a_2\, x^{n-2}\, y^{(n-2)}(x) \ldots + a_n\, y(x) = 0.$$

The following is an example of an Euler equation.

```
In[1]:= DSolve[
  x^4 * y''''[x] - 2 * x^3 * y'''[x] - x^2 * y''[x] + 5 * x * y'[x] + y[x] == 0, y[x], x]
Out[1]= {{y[x] → x^Root[1-4 #1+16 #1^2-8 #1^3+#1^4&,1] C[1] + x^Root[1-4 #1+16 #1^2-8 #1^3+#1^4&,2] C[2] +
      x^Root[1-4 #1+16 #1^2-8 #1^3+#1^4&,3] C[3] + x^Root[1-4 #1+16 #1^2-8 #1^3+#1^4&,4] C[4]}}
```

The *Legendre linear* equation is a generalization of the Euler equation. It has the form

$$(c\,x+d)^n\,y^{(n)}(x)+a_1\,(c\,x+d)^{n-1}\,y^{(n-1)}(x)+a_2\,(c\,x+d)^{n-2}\,y^{(n-2)}(x)\ldots+a_n\,y(x)=0.$$

This is a Legendre linear equation.

```
In[2]:= DSolve[(3 x + 5)^4 * y''''[x] - 2 * (3 x + 5)^3 * y'''[x] -
          (3 x + 5)^2 * y''[x] + 5 * (3 x + 5) * y'[x] + y[x] == 0, y[x], x]
```

```
Out[2]= {{y[x] → (5 + 3 x)^Root[1-570 #1+1044 #1^2-540 #1^3+81 #1^4&,1] C[1] + (5 + 3 x)^Root[1-570 #1+1044 #1^2-540 #1^3+81 #1^4&,2] C[2] +
          (5 + 3 x)^Root[1-570 #1+1044 #1^2-540 #1^3+81 #1^4&,3] C[3] + (5 + 3 x)^Root[1-570 #1+1044 #1^2-540 #1^3+81 #1^4&,4] C[4]}}
```

Exact Higher-Order Equations

A linear ordinary differential equation of order n

$$a_0(x)\,y^{(n)}(x)+a_1(x)\,y^{(n-1)}(x)+\ldots+a_{n-1}(x)\,y'(x)+a_n(x)\,y(x)=0$$

is said to be *exact* if

$$(-1)^n\,a_0{}^{(n)}(x)+(-1)^{(n-1)}\,a_1{}^{(n-1)}(x)+\ldots-a_{n-1}{}'(x)+a_n(x)=0.$$

The condition of exactness can be used to reduce the problem to that of solving an equation of order $n-1$.

This is an example of an exact ODE.

```
In[1]:= a0 = 1;
```

```
In[2]:= a1 = -1 ;
```

```
In[3]:= a2 = 5 x;
```

```
In[4]:= a3 = 5;
```

```
In[5]:= ExactODE = a0 * y''''[x] + a1 * y'''[x] + a2 * y''[x] + a3 * y[x]
```

```
Out[5]= 5 y[x] + 5 x y'[x] - y''[x] + y^(3)[x]
```

This verifies the condition for exactness.

```
In[6]:= conditionforexactness = -D[a0, {x, 3}] + D[a1, {x, 2}] - D[a2, x] + a3
```

```
Out[6]= 0
```

This solves the equation.

```
In[7]:= sol = DSolve[ExactODE == 0, y, x]
```

$$\text{Out[7]}= \left\{\left\{y \to \text{Function}\left[\{x\}, e^{x/2}\,\text{AiryAi}\left[-\frac{(-1)^{1/3}\left(\frac{1}{4}-5x\right)}{5^{2/3}}\right]C[2] + e^{x/2}\,\text{AiryBi}\left[-\frac{(-1)^{1/3}\left(\frac{1}{4}-5x\right)}{5^{2/3}}\right]C[3] + e^{x/2}\left(\text{AiryBi}\left[-\frac{(-1)^{1/3}\left(\frac{1}{4}-5x\right)}{5^{2/3}}\right]\int_1^x -\frac{(-1)^{2/3}\,e^{-\frac{K[2]}{2}}\,\pi\,\text{AiryAi}\left[-\frac{(-1)^{1/3}\left(\frac{1}{4}-5K[2]\right)}{5^{2/3}}\right]C[1]}{5^{1/3}}\,dK[2] + \text{AiryAi}\left[-\frac{(-1)^{1/3}\left(\frac{1}{4}-5x\right)}{5^{2/3}}\right]\int_1^x \frac{(-1)^{2/3}\,e^{-\frac{K[1]}{2}}\,\pi\,\text{AiryBi}\left[-\frac{(-1)^{1/3}\left(\frac{1}{4}-5K[1]\right)}{5^{2/3}}\right]C[1]}{5^{1/3}}\,dK[1]\right)\right]\right\}\right\}$$

This verifies the solution.

```
In[8]:= ExactODE /. sol[[1]] /. {x → RandomReal[],
          C[1] → RandomReal[], C[2] → RandomReal[]} // N // Simplify // Chop
Out[8]= 0

In[9]:= Clear[a0, a1, a2, a3]
```

Further Examples of Exactly Solvable Higher-Order Equations

The solutions to many second-order ODEs can be expressed in terms of special functions. Solutions to certain higher-order ODEs can also be expressed using `AiryAi`, `BesselJ`, and other special functions.

The solution to this third-order ODE is given by products of Airy functions.

```
In[1]:= sol1 = DSolve[y'''[x] - 4 * (x + 2) * y'[x] - 2 * y[x] == 0, y, x]
Out[1]= {{y → Function[{x}, AiryAi[2 + x]^2 C[1] + AiryAi[2 + x] AiryBi[2 + x] C[2] + AiryBi[2 + x]^2 C[3]]}}
```

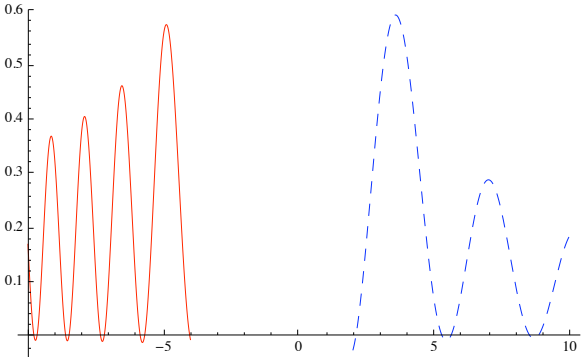
The solution to this third-order ODE is given by Bessel functions.

```
In[2]:= sol2 = DSolve[
          x^3 * y'''[x] + 3 * x^2 * y''[x] + (4 x^3 - 11 x) * y'[x] + 4 x^2 * y[x] == 0, y, x]
Out[2]= {{y → Function[{x},
            BesselJ[Sqrt[3], x]^2 C[1] + BesselJ[Sqrt[3], x] BesselY[Sqrt[3], x] C[2] + BesselY[Sqrt[3], x]^2 C[3]]}}
```

This plot shows the oscillatory behavior of the solutions on different parts of the real line.

```
In[3]:= Show[{Plot[y[x] /. sol1 /. {C[1] → 2, C[2] → 3, C[3] → 1}, {x, -10, -4},
    PlotStyle → {Red}], Plot[y[x] /. sol2 /. {C[1] → 2, C[2] → 3, C[3] → 1}, {x, 2, 10},
    PlotStyle → {Dashing[{0.02}], Blue}]}, PlotRange → {{-10, 10}, Automatic}]
```

Out[3]=



The solution to this fourth-order linear ODE is expressed in terms of HypergeometricPFQ.

```
In[4]:= HypergeometricPFQTypeEquation = 30 * x^3 * y''''[x] +
    193 * x^2 * y'''[x] + (219 * x - 30 x^2) * y''[x] + (21 - 90 x) * y'[x] - 30 y[x];
```

```
In[5]:= sol3 = DSolve[HypergeometricPFQTypeEquation == 0, y, x]
```

Out[5]=

$$\left\{\left\{y \to \text{Function}\left[\{x\}, \frac{(-1)^{2/3}\, C[1]\, \text{HypergeometricPFQ}\left[\left\{-\frac{1}{3}\right\}, \left\{-\frac{5}{6}, -\frac{11}{15}\right\}, x\right]}{x^{4/3}} + (-1)^{2/5}\, x^{2/5}\, C[3]\, \text{HypergeometricPFQ}\left[\left\{\frac{7}{5}\right\}, \left\{\frac{9}{10}, \frac{41}{15}\right\}, x\right] + i\sqrt{x}\, C[4]\, \text{HypergeometricPFQ}\left[\left\{\frac{3}{2}\right\}, \left\{\frac{11}{10}, \frac{17}{6}\right\}, x\right] + C[2]\, \text{HypergeometricPFQ}\left[\{1, 1\}, \left\{\frac{1}{2}, \frac{3}{5}, \frac{7}{3}\right\}, x\right]\right]\right\}\right\}$$

This verifies that the solution is correct using numerical values.

```
In[6]:= HypergeometricPFQTypeEquation /. sol3[[1]] /. {x → RandomComplex[]} // Simplify //
    Chop
```

```
Out[6]= 0
```

As for second-order linear ODEs, there are modern algorithms for solving higher-order ODEs with rational coefficients. These algorithms give "rational-exponential" solutions, which are combinations of rational functions and exponentials of the integrals of rational functions. These algorithms are combined with techniques such as reduction of order to produce a complete solution for the given ODE.

The general solution to this equation has a rational term and terms that depend on Airy functions. The Airy functions come from reducing the order of the equation to 2.

```
In[7]:= DSolve[((6 - 24 * x + 4 * x^2 - 10 * x^3 + 9 * x^4 - 4 * x^5 + x^6) * y[x]) /
          ((-1 + x)^4 * x^2) +
        ((-8 + 30 * x - 10 * x^2 + 10 * x^3 - 3 * x^4 - 2 * x^5 + x^6) *
           Derivative[1][y][x]) / ((-1 + x)^3 * x^2) +
        ((4 - 16 * x + 5 * x^2 - 2 * x^3) * Derivative[2][y][x]) / ((-1 + x)^2 * x^2) +
        ((4 - x^2) * Derivative[3][y][x]) / ((-1 + x) * x) == 0, y, x]
```

$$\text{Out[7]= } \left\{\left\{y \to \text{Function}\left[\{x\}, \frac{(-1+x)\,C[1]}{-4+x^2} + \frac{(-1+x)\,\text{AiryAi}[x]\,C[2]}{-4+x^2} + \frac{(-1+x)\,\text{AiryBi}[x]\,C[3]}{-4+x^2}\right]\right\}\right\}$$

The equations considered so far have been homogeneous; that is, with no term free of $y(x)$ or its derivatives. If the given ODE is inhomogeneous, `DSolve` applies the method of *variation of parameters* to obtain the solution.

Here is an example of this type. The exponential terms in the solution come from the general solution to the homogeneous equation, and the remaining term is a particular solution (or particular integral) to the problem.

```
In[8]:= sol = DSolve[y'''[x] - 13 * y''[x] + 19 * y'[x] + 33 * y[x] == Cos[2 x], y[x], x]
```

$$\text{Out[8]= } \left\{\left\{y[x] \to e^{-x}\,C[1] + e^{3x}\,C[2] + e^{11x}\,C[3] + \frac{17\,\text{Cos}[2x] + 6\,\text{Sin}[2x]}{1625}\right\}\right\}$$

This is the general solution to the homogeneous equation.

```
In[9]:= generalsolution =
         y[x] /. DSolve[y'''[x] - 13 * y''[x] + 19 * y'[x] + 33 * y[x] == 0, y[x], x][[1]]
```

$$\text{Out[9]= } e^{-x}\,C[1] + e^{3x}\,C[2] + e^{11x}\,C[3]$$

This particular solution is part of the general solution to the inhomogeneous equation.

```
In[10]:= particularsolution = (y[x] /. sol[[1]]) - generalsolution
```

$$\text{Out[10]= } \frac{17\,\text{Cos}[2x] + 6\,\text{Sin}[2x]}{1625}$$

Thus, the general solution for the inhomogeneous equation is the sum of the general solution to the homogeneous equation and a particular integral of the ODE.

The solution methods for nonlinear ODEs of higher order rely to a great extent on reducing the problem to one of lower order.

Here is a nonlinear third-order ODE with no explicit dependence on x or $y[x]$. It is solved by reducing the order to 2 using a simple integration.

```
In[11]:= DSolve[7 * y'[x] * y'''[x] - 11 * y''[x]^2 == 0, y[x], x]
```

$$\text{Out[11]= } \left\{\left\{y[x] \to -\frac{C[2]}{3\ (4\ x + 7\ C[1])^{3/4}} + C[3]\right\}\right\}$$

Systems of ODEs

Introduction to Systems of ODEs

Systems of ODEs are important in various fields of science, such as the study of electricity and population biology. Like single ODEs, systems of ODEs can classified as *linear* or *nonlinear*.

A system of linear first-order ODEs can be represented in the form

$$X'(t) = A(t)\, X(t) + B(t).$$

Here $X(t)$ is a vector of unknown functions, $A(t)$ is the matrix of the coefficients of the unknown functions, and $B(t)$ is a vector representing the inhomogeneous part of the system.

In the two-dimensional case, the system can be written more concretely as

$$x'(t) = p(t)\, x(t) + q(t)\, y(t) + u(t)$$

$$y'(t) = r(t)\, x(t) + s(t)\, y(t) + v(t).$$

If all the entries of the matrix $A(t)$ are constants, then the system is said to be *linear with constant coefficients*. If $B(t)$ is the zero vector, then the system is said to be *homogeneous*.

The important global features of the solutions to linear systems can be clarified by considering homogeneous systems of ODEs with constant coefficients.

Linear Systems of ODEs

Here is a system of two ODEs whose coefficient matrix has real and distinct eigenvalues.

```
In[1]:= A = {{4, -6}, {1, -1}};
```

```
In[2]:= Eigenvalues[A]
Out[2]= {2, 1}
```

```
In[3]:= X[t_] = {x[t], y[t]};

In[4]:= system = MapThread[#1 == #2 &, {X'[t], A.X[t]}]
Out[4]= {x'[t] == 4 x[t] - 6 y[t], y'[t] == x[t] - y[t]}
```

This solves the system. Note that the general solution depends on two arbitrary constants `C[1]` and `C[2]`.

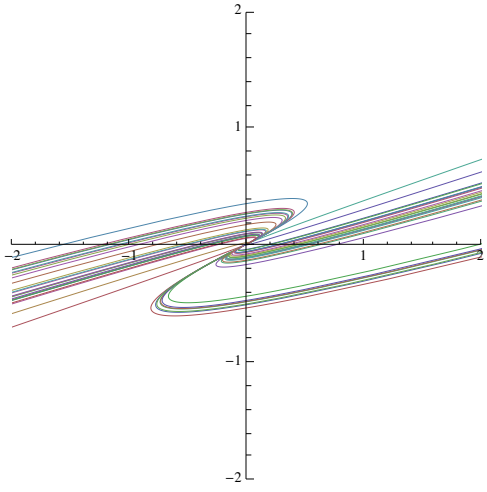
```
In[5]:= sol = DSolve[system, {x, y}, t]
Out[5]= {{x → Function[{t}, ℯ^t (-2 + 3 ℯ^t) C[1] - 6 ℯ^t (-1 + ℯ^t) C[2]],
    y → Function[{t}, ℯ^t (-1 + ℯ^t) C[1] - ℯ^t (-3 + 2 ℯ^t) C[2]]}}
```

Here is a plot of some particular solutions obtained by giving specific values to `C[1]` and `C[2]`. In this case, the origin is a called a *node.*

```
In[6]:= particularsols =
          Partition[Flatten[Table[{x[t], y[t]} /. sol /. {C[1] → 1 / i, C[2] → 1 / j},
              {i, -20, 20, 6}, {j, -20, 20, 6}]], 2];

In[7]:= ParametricPlot[Evaluate[particularsols], {t, -3, 3}, PlotRange -> {-2, 2}]
```

Out[7]=



In this system the eigenvalues of the coefficient matrix are complex conjugates of each other.

```
In[8]:= A = {{7, -8}, {5, -5}};

In[9]:= Eigenvalues[A]
Out[9]= {1 + 2 i, 1 - 2 i}

In[10]:= X[t_] = {x[t], y[t]};

In[11]:= system = MapThread[#1 == #2 &, {X'[t], A.X[t]}]
Out[11]= {x'[t] == 7 x[t] - 8 y[t], y'[t] == 5 x[t] - 5 y[t]}
```

This solves the system.

```
In[12]:= sol = DSolve[system, {x, y}, t]
```

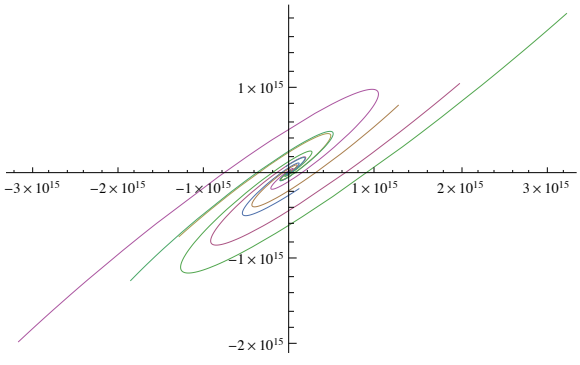
```
Out[12]= {{x → Function[{t}, -4 e^t C[2] Sin[2 t] + e^t C[1] (Cos[2 t] + 3 Sin[2 t])],
    y → Function[{t}, e^t C[2] (Cos[2 t] - 3 Sin[2 t]) + 5/2 e^t C[1] Sin[2 t]]}}
```

This plots the solution for various values of the arbitrary parameters. The *spiraling* behavior is typical for systems with complex eigenvalues.

```
In[13]:= particularsols =
  Partition[Flatten[Table[{x[t], y[t]} /. sol /. {C[1] → 1 / i, C[2] → 1 / j},
      {i, -10, 10, 8}, {j, -10, 10, 8}]], 2];
```

```
In[14]:= ParametricPlot[Evaluate[particularsols], {t, -35, 35},
 PlotRange → All, PlotPoints → 70, Method -> {Compiled → False}]
```

Out[14]=



Solving homogeneous systems of ODEs with constant coefficients and of arbitrary order is a straightforward matter. They are solved by converting them to a system of first-order ODEs.

This solves a homogeneous system of ODEs of order 3, with constant coefficients.

```
In[15]:= system = {x'''[t] + y[t], y'''[t] - 64 x[t]};
```

```
In[16]:= sol = DSolve[{system[[1]] == 0, system[[2]] == 0}, {x, y}, t]
```

```
Out[16]= {{x → Function[{t}, 1/3 E^(-Sqrt[3] t) C[3] (Cos[t] + E^(2 Sqrt[3] t) Cos[t] + E^(Sqrt[3] t) Cos[2 t]) +
      1/96 E^(-Sqrt[3] t) C[6] (Cos[t] + E^(2 Sqrt[3] t) Cos[t] - 2 E^(Sqrt[3] t) Cos[2 t] + Sqrt[3] Sin[t] - Sqrt[3] E^(2 Sqrt[3] t) Sin[t]) +
      1/24 E^(-Sqrt[3] t) C[1] (Cos[t] + E^(2 Sqrt[3] t) Cos[t] - 2 E^(Sqrt[3] t) Cos[2 t] - Sqrt[3] Sin[t] + Sqrt[3] E^(2 Sqrt[3] t) Sin[t]) +
      1/192 E^(-Sqrt[3] t) C[2] (-Sqrt[3] Cos[t] + Sqrt[3] E^(2 Sqrt[3] t) Cos[t] - Sin[t] - E^(2 Sqrt[3] t) Sin[t] - 2 E^(Sqrt[3] t) Sin[2 t]) -
      1/24 E^(-Sqrt[3] t) C[5] (Sin[t] + E^(2 Sqrt[3] t) Sin[t] - E^(Sqrt[3] t) Sin[2 t]) +
      1/12 E^(-Sqrt[3] t) C[4] (-Sqrt[3] Cos[t] + Sqrt[3] E^(2 Sqrt[3] t) Cos[t] + Sin[t] + E^(2 Sqrt[3] t) Sin[t] + 2 E^(Sqrt[3] t) Sin[2 t])],
    y → Function[{t}, 1/3 E^(-Sqrt[3] t) C[5] (Cos[t] + E^(2 Sqrt[3] t) Cos[t] + E^(Sqrt[3] t) Cos[2 t]) -
      2/3 E^(-Sqrt[3] t) C[4] (Cos[t] + E^(2 Sqrt[3] t) Cos[t] - 2 E^(Sqrt[3] t) Cos[2 t] + Sqrt[3] Sin[t] - Sqrt[3] E^(2 Sqrt[3] t) Sin[t]) +
      1/24 E^(-Sqrt[3] t) C[2] (Cos[t] + E^(2 Sqrt[3] t) Cos[t] - 2 E^(Sqrt[3] t) Cos[2 t] - Sqrt[3] Sin[t] + Sqrt[3] E^(2 Sqrt[3] t) Sin[t]) -
      1/3 E^(-Sqrt[3] t) C[1] (-Sqrt[3] Cos[t] + Sqrt[3] E^(2 Sqrt[3] t) Cos[t] - Sin[t] - E^(2 Sqrt[3] t) Sin[t] - 2 E^(Sqrt[3] t) Sin[2 t]) +
      8/3 E^(-Sqrt[3] t) C[3] (Sin[t] + E^(2 Sqrt[3] t) Sin[t] - E^(Sqrt[3] t) Sin[2 t]) +
      1/12 E^(-Sqrt[3] t) C[6] (-Sqrt[3] Cos[t] + Sqrt[3] E^(2 Sqrt[3] t) Cos[t] + Sin[t] + E^(2 Sqrt[3] t) Sin[t] + 2 E^(Sqrt[3] t) Sin[2 t])]}}
```

This verifies the solution.

```
In[17]:= system /. sol[[1]] /. {t → RandomComplex[], C[1] → RandomComplex[],
     C[2] → RandomComplex[], C[3] → RandomComplex[], C[4] → RandomComplex[],
     C[5] → RandomComplex[], C[6] → RandomComplex[]} // Chop
Out[17]= {0, 0}
```

In general, systems of linear ODEs with non-constant coefficients can only be solved in cases where the coefficient matrix has a simple structure, as illustrated in the following examples.

This first-order system has a diagonal coefficient matrix. The system is *uncoupled* because the first equation involves only $x[t]$ and the second equation depends only on $y[t]$. Thus, each equation in the system can be integrated independently of the other.

```
In[18]:= DSolve[{x'[t] == Sin[t] * x[t], y'[t] == t^2 * y[t]}, {x, y}, t]
```

```
Out[18]= {{x → Function[{t}, E^(-Cos[t]) C[1]], y → Function[{t}, E^(t^3/3) C[2]]}}
```

The rows of the coefficient matrix for this system form an orthogonal set of vectors.

```
In[19]:= A = {{E^t, Tan[t]}, {-Tan[t], E^t}};

In[20]:= A.Transpose[A] / Det[A]
Out[20]= {{1, 0}, {0, 1}}
```

```
In[21]:= X[t_] = {x[t], y[t]};

In[22]:= system = MapThread[#1 == #2 &, {X'[t], A.X[t]}]

Out[22]= {x'[t] == E^t x[t] + Tan[t] y[t], y'[t] == -Tan[t] x[t] + E^t y[t]}

In[23]:= sol = DSolve[system, {x, y}, t]

Out[23]= {{x → Function[{t}, E^E^t C[1] Cos[Log[Cos[t]]] - E^E^t C[2] Sin[Log[Cos[t]]]],
    y → Function[{t}, E^E^t C[2] Cos[Log[Cos[t]]] + E^E^t C[1] Sin[Log[Cos[t]]]]}}

In[24]:= system /. sol[[1]] // Simplify

Out[24]= {True, True}
```

Here is a system of three first-order ODEs. The coefficient matrix is upper triangular.

```
In[25]:= A = {{E^t, 2, 3}, {0, 2, -1}, {0, 0, 1}};

In[26]:= MatrixForm[A]

Out[26]//MatrixForm=
⎛ E^t  2   3 ⎞
⎜ 0    2  -1 ⎟
⎝ 0    0   1 ⎠

In[27]:= X[t_] = {x[t], y[t], z[t]};

In[28]:= system = MapThread[#1 == #2 &, {X'[t], A.X[t]}]

Out[28]= {x'[t] == E^t x[t] + 2 y[t] + 3 z[t], y'[t] == 2 y[t] - z[t], z'[t] == z[t]}

In[29]:= sol = DSolve[system, {x, y, z}, t]

Out[29]= {{x → Function[{t}, E^E^t C[1] + 2 (-1 - E^t) C[2] - 5 C[3]],
    y → Function[{t}, E^(2 t) C[2] + E^t C[3]], z → Function[{t}, E^t C[3]]}}
```

As for single ODEs, there are sophisticated modern algorithms for solving systems of ODEs with rational coefficients.

This solves a system of two first-order ODEs with rational coefficients. Note that the solution is composed entirely of rational functions.

```
In[30]:= DSolve[{y'[x] == (5 + x) w[x] / ((-3 - 2 x + x^2) (-1 + x^3)) + (6 + 2 x - 3 x^3 - x^5) y[x] / (x (-3 - 2 x + x^2) (-1 + x^3)),
    w'[x] == (1 + 20 x^2 + 3 x^3) w[x] / ((5 + x) (-1 + x^3)) - 4 x (-3 - 2 x + x^2) y[x] / ((5 + x) (-1 + x^3))}, {y, w}, x]

Out[30]= {{y → Function[{x}, x C[1] / (-3 - 2 x + x^2) + x^2 C[2] / (-3 - 2 x + x^2)], w → Function[{x}, C[1] / (5 + x) + x^4 C[2] / (5 + x)]}}
```

In the following example, the algorithm finds one rational solution for $x[t]$ and $y[t]$. (The equation for $z[t]$ is uncoupled from the rest of the system.) Using the rational solution, `DSolve` is able to find the remaining exponential solution for $x[t]$ and $y[t]$.

```
In[31]:= system = {x'[t] == (2 * ((-9 + t + t^2) * x[t] + (-6 + t^2) * y[t])) / ((-2 + t) * t),
           y'[t] == (-3 * (-12 + 2 * t + t^2) * x[t] + (24 - 2 * t - 3 * t^2) * y[t]) / ((-2 + t) * t),
           z'[t] == t * z[t]};

In[32]:= sol = DSolve[system, {x, y, z}, t]
```

$$\text{Out[32]= } \left\{\left\{x \to \text{Function}\left[\{t\}, \frac{(-2-t)\,C[1]}{t^3} - 2\,e^{-t}\,C[2]\right],\right.\right.$$
$$\left.\left. y \to \text{Function}\left[\{t\}, \frac{(4+t)\,C[1]}{t^3} + 3\,e^{-t}\,C[2]\right],\ z \to \text{Function}\left[\{t\}, e^{\frac{t^2}{2}}\,C[3]\right]\right\}\right\}$$

```
In[33]:= system /. sol // Simplify
Out[33]= {{True, True, True}}
```

The systems considered so far have all been homogeneous. If the system is inhomogeneous (that is, if there are terms free from any dependent variables and their derivatives), `DSolve` applies either the *method of variation of parameters* or the *method of undetermined coefficients* to find the general solution.

This solves an inhomogeneous system.

```
In[34]:= A = {{7, -8}, {5, -5}};

In[35]:= B = {E^(t / 10), t};

In[36]:= X[t_] = {x[t], y[t]};

In[37]:= system = MapThread[#1 == #2 &, {X'[t], A.X[t] + B}]
```

$$\text{Out[37]= } \left\{x'[t] == e^{t/10} + 7\,x[t] - 8\,y[t],\ y'[t] == t + 5\,x[t] - 5\,y[t]\right\}$$

In[38]:=
```
sol = DSolve[system, {x, y}, t]
```

Out[38]=
$$\left\{\left\{x \to \text{Function}\left[\{t\},\right.\right.\right.$$
$$-4\, e^{t} C[2] \sin[2t] + e^{t} C[1] (\cos[2t] + 3\sin[2t]) - 4\, e^{t} \sin[2t] \left(\frac{500}{481} e^{-9t/10} \cos[2t] - \frac{1}{50} e^{-t} (18 + 70t) \cos[2t] + \frac{225}{481} e^{-9t/10} \sin[2t] - \frac{1}{50} e^{-t} (-26 + 10t) \sin[2t]\right) +$$
$$e^{t} (\cos[2t] + 3\sin[2t]) \left(\frac{510}{481} e^{-9t/10} \cos[2t] - \frac{4}{25} e^{-t} (4 + 10t) \cos[2t] + \frac{470}{481} e^{-9t/10} \sin[2t] - \frac{2}{25} e^{-t} (-6 + 10t) \sin[2t]\right)\Big],$$
$$y \to \text{Function}\Big[\{t\}, e^{t} C[2] (\cos[2t] - 3\sin[2t]) + \frac{5}{2} e^{t} C[1] \sin[2t] + e^{t} (\cos[2t] - 3\sin[2t]) \left(\frac{500}{481} e^{-9t/10} \cos[2t] - \frac{1}{50} e^{-t} (18 + 70t) \cos[2t] + \frac{225}{481} e^{-9t/10} \sin[2t] - \frac{1}{50} e^{-t} (-26 + 10t) \sin[2t]\right) + \frac{5}{2} e^{t} \sin[2t] \left(\frac{510}{481} e^{-9t/10} \cos[2t] - \frac{4}{25} e^{-t} (4 + 10t) \cos[2t] + \frac{470}{481} e^{-9t/10} \sin[2t] - \frac{2}{25} e^{-t} (-6 + 10t) \sin[2t]\right)\Big]\Big\}\Big\}$$

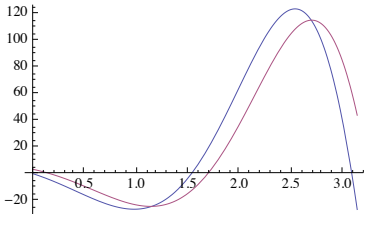
Particular solutions to the system can be obtained by assigning values to the constants `C[1]` and `C[2]`.

Here is a plot of the solution for one choice of parameters.

In[39]:=
```
particularsol = {x[t], y[t]} /. sol[[1]] /. {C[1] → -1, C[2] → 2};
```

In[40]:=
```
Plot[Evaluate[particularsol], {t, 0, Pi}]
```

Out[40]=



Nonlinear Systems of ODEs

Following are two examples of nonlinear systems of ODEs that can be solved symbolically using `DSolve`.

The first three equations in this system of four nonlinear ODEs can be solved independently because none of their right-hand sides depend on p, q, r, or s.

In[1]:=
```
system =
  {p'[x] == 1, q'[x] == x, r'[x] == 0, s'[x] == r[x] / (p[x] + 4 * q[x] * r[x])};
```

```
In[2]:= sol = DSolve[system, {p, q, r, s}, x]
```

$$\text{Out[2]}= \left\{\left\{\text{p}\to \text{Function}[\{\text{x}\},\ \text{x}+\text{C}[1]],\ \text{q}\to \text{Function}\left[\{\text{x}\},\ \frac{\text{x}^2}{2}+\text{C}[2]\right],\right.\right.$$
$$\left.\left.\text{r}\to \text{Function}[\{\text{x}\},\ \text{C}[3]],\ \text{s}\to \text{Function}\left[\{\text{x}\},\ \frac{2\ \text{ArcTan}\left[\frac{1+4\,\text{x}\,\text{C}[3]}{\sqrt{-1+8\,\text{C}[1]\,\text{C}[3]+32\,\text{C}[2]\,\text{C}[3]^2}}\right]\text{C}[3]}{\sqrt{-1+8\,\text{C}[1]\,\text{C}[3]+32\,\text{C}[2]\,\text{C}[3]^2}}+\text{C}[4]\right]\right\}\right\}$$

```
In[3]:= system /. sol // Simplify
Out[3]= {{True, True, True, True}}
```

This system of two nonlinear ODEs is *autonomous*, in the sense that the right-hand sides of the equations do not depend on x.

```
In[4]:= system = {u'[x] == 1 / Sqrt[v[x]], v'[x] == u[x]};

In[5]:= sol = DSolve[system, {u, v}, x][[1]]
```

$$\text{Out[5]}= \left\{\text{v}\to \text{Function}\left[\{\text{x}\},\ \frac{1}{16}\left(4\,\text{C}[1]^2+4\,\text{C}[1]\left(\frac{6\times 2^{1/3}\,\text{C}[1]}{\left(-324\,\text{x}+\sqrt{23\,328\,\text{C}[1]^3+(-324\,\text{x}-81\,\text{C}[2])^2}-81\,\text{C}[2]\right)^{1/3}}-\right.\right.\right.$$
$$\left.\left.\left.\frac{\left(-324\,\text{x}+\sqrt{23\,328\,\text{C}[1]^3+(-324\,\text{x}-81\,\text{C}[2])^2}-81\,\text{C}[2]\right)^{1/3}}{3\times 2^{1/3}}\right)^2+\right.\right.$$
$$\left.\left.\left(\frac{6\times 2^{1/3}\,\text{C}[1]}{\left(-324\,\text{x}+\sqrt{23\,328\,\text{C}[1]^3+(-324\,\text{x}-81\,\text{C}[2])^2}-81\,\text{C}[2]\right)^{1/3}}-\right.\right.\right.$$
$$\left.\left.\left.\frac{\left(-324\,\text{x}+\sqrt{23\,328\,\text{C}[1]^3+(-324\,\text{x}-81\,\text{C}[2])^2}-81\,\text{C}[2]\right)^{1/3}}{3\times 2^{1/3}}\right)^4\right)\right],$$
$$\text{u}\to \text{Function}\left[\{\text{x}\},\ \frac{6\times 2^{1/3}\,\text{C}[1]}{\left(-324\,\text{x}+\sqrt{23\,328\,\text{C}[1]^3+(-324\,\text{x}-81\,\text{C}[2])^2}-81\,\text{C}[2]\right)^{1/3}}-\right.$$
$$\left.\frac{\left(-324\,\text{x}+\sqrt{23\,328\,\text{C}[1]^3+(-324\,\text{x}-81\,\text{C}[2])^2}-81\,\text{C}[2]\right)^{1/3}}{3\times 2^{1/3}}\right]\Big\}$$

```
In[6]:= system /. sol // Simplify // PowerExpand // Simplify
Out[6]= {True, True}
```

The previous two examples demonstrate that the solutions to fairly simple systems are usually complicated expressions of the independent variable. In fact, the solution is often available only in implicit form and may thus contain `InverseFunction` objects or unevaluated `Solve` objects.

Lie Symmetry Methods for Solving Nonlinear ODEs

Around 1870, Marius Sophus Lie realized that many of the methods for solving differential equations could be unified using group theory. Lie symmetry methods are central to the modern approach for studying nonlinear ODEs. They use the notion of symmetry to generate solutions in a systematic manner. Here is a brief introduction to Lie's approach that provides some examples that are solved in this way by `DSolve`.

A key notion in Lie's method is that of an *infinitesimal generator* for a symmetry group. This concept is illustrated in the following example.

Here is the well-known transformation for rotations in the x-y plane. This is a one-parameter group of transformations with parameter t.

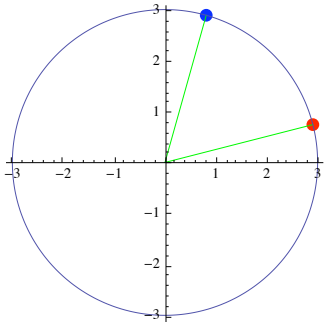
```
In[1]:= m = x * Cos[t] + y * Sin[t];
```

```
In[2]:= n = -x * Sin[t] + y * Cos[t];
```

For a fixed value of t, the point (m, n) (in blue) can be obtained by rotating the line joining (x, y) (in red) to the origin through an angle of t in the counterclockwise direction.

```
In[3]:= Show[
         {Graphics[{{Red, PointSize[0.04], Point[{3 * Cos[1 / 4], 3 * Sin[1 / 4]}]}, {Blue,
              PointSize[0.04], Point[{3 * Cos[(1 / 4) + (Pi / 3)], 3 * Sin[(1 / 4) + (Pi / 3)]}]},
            {Green, Line[{{0, 0}, {3 * Cos[1 / 4], 3 * Sin[1 / 4]}}]},
            {Green, Line[{{0, 0}, {3 * Cos[(1 / 4) + (Pi / 3)], 3 * Sin[(1 / 4) + (Pi / 3)]}}]}]},
          ParametricPlot[{3 * Cos[t], 3 * Sin[t]}, {t, 0, 2 Pi},
           DisplayFunction → Identity]},
         AspectRatio → 1, ImageSize → 200, Axes → True]
```

Out[3]=



A rotation through t can be represented by the matrix

$$M_t = \begin{pmatrix} \cos(t) & \sin(t) \\ -\sin(t) & \cos(t) \end{pmatrix}.$$

This shows that the set of all rotations in the plane satisfies the properties for forming a group.

```
In[3]:= M[t_] := {{Cos[t], Sin[t]}, {-Sin[t], Cos[t]}}

In[4]:= Simplify[M[a + b] == M[a].M[b]]
Out[4]= True

In[5]:= M[a].M[0] == M[0].M[a] == M[a]
Out[5]= True

In[6]:= M[0] == IdentityMatrix[2]
Out[6]= True

In[7]:= M[a].M[-a] == M[-a].M[a] == IdentityMatrix[2] // Simplify
Out[7]= True
```

The Lie symmetry method requires calculating a first-order approximation to the expressions for the group. This approximation is called an *infinitesimal generator*.

This expands the expressions for m and n in a series with respect to t and around the origin 0 to obtain linear approximations.

```
In[8]:= Series[m , {t, 0, 1}] // Normal
Out[8]= x + t y

In[9]:= Series[n , {t, 0, 1}] // Normal
Out[9]= -t x + y
```

The coefficients of the linear terms in t are y and $-x$, respectively. The infinitesimal generator for the rotation group in the plane is defined to be the following differential operator.

```
In[10]:= v = (y * D[#, x] - x * D[#, y]) &;
```

Starting from the infinitesimal generator, the original group can be recovered by integrating the fundamental system of *Lie equations.* For the group of rotations, the Lie equations are given by the first argument to `DSolve` shown here.

```
In[11]:= DSolve[{x'[t] == y[t], y'[t] == -x[t]}, {x[t], y[t]}, t]
Out[11]= {{x[t] → C[1] Cos[t] + C[2] Sin[t], y[t] → C[2] Cos[t] - C[1] Sin[t]}}
```

The rotation group arises in the study of symmetries of geometrical objects; it is an example of a *symmetry group*. The infinitesimal generator, a differential operator, is a convenient local representation for this symmetric group, which is a set of matrices.

An expression that reduces to 0 under the action of the infinitesimal generator is called an *invariant* of the group.

Here is an invariant for this group.

```
In[12]:= invariant = x^2 + y^2;
```

This states that the distance from the origin to (x, y), $\sqrt{x^2 + y^2}$, is preserved under rotation.

```
In[13]:= v[invariant]
Out[13]= 0
```

In the following examples, these ideas are applied to differential equations.

This is an example of a Riccati equation, from page 103 of [I99].

```
In[14]:= Riccatiequation = y'[x] + y[x]^2 - 1 / x^2 == 0;
```

The equation is invariant under the following *scaling* transformation.

```
In[15]:= m = x * E^t;
```

```
In[16]:= n = y * E^(-t);
```

The infinitesimal generator for this one-parameter group of transformations is found as before.

```
In[17]:= Series[m, {t, 0, 1}] // Normal
Out[17]= x + t x
```

```
In[18]:= Series[n, {t, 0, 1}] // Normal
Out[18]= y - t y
```

```
In[19]:= v = (x * D[#, x] - y * D[#, y]) &;
```

Now, the Riccati equation depends on three variables: x, $y = y[x]$, and $p = y'[x]$. Hence, the infinitesimal generator v must be *prolonged* to act on all three variables in this first-order equation.

It turns out that the required *prolongation* is as follows.

```
In[20]:= prolongedv = (x * D[#, x] - y * D[#, y] - 2 p * D[#, p] ) &;
```

This shows that the expression for the Riccati equation in the (x, y, p) coordinates is indeed invariant under `prolongedv`.

```
In[21]:= Riccatiexpression = p + y^2 - (1 / x^2);

In[22]:= prolongedv[Riccatiexpression] /. {p → (1 / x^2) - y^2} // Together
Out[22]= 0
```

Depending on the order of the given equation, the knowledge of a symmetry (in the form of an infinitesimal generator) can be used in three ways.

- If the order of the equation is 1, it gives an integrating factor for the ODE that makes the equation *exact* and hence solvable.
- It gives a set of *canonical coordinates* in which the equation has a simple (integrable) form.
- It *reduces* the problem of solving an ODE of order n to that of solving an ODE of order $n-1$, which is typically a simpler problem.

The `DSolve` function checks for certain standard types of symmetries in the given ODE and uses them to return a solution. Following are three examples of ODEs for which `DSolve` uses such a symmetry method.

Here is a nonlinear first-order ODE (equation 1.120, page 315 of [K59]).

```
In[23]:= FirstOrderODE = x * y'[x] == y[x] * (x * Log[x^2 / y[x]] + 2);
```

This ODE has a symmetry with the following infinitesimal generator.

```
In[24]:= v = (-2 * Exp[-x] * y) * D[#, y] &;
```

The presence of this symmetry allows `DSolve` to calculate an integrating factor and return the solution.

```
In[25]:= sol = DSolve[FirstOrderODE, y, x]
Out[25]= {{y → Function[{x}, e^(-2 e^-x C[1]) x^2]}}
```

This verifies the solution.

```
In[26]:= FirstOrderODE /. sol[[1]] /. {x → RandomReal[], C[1] → RandomReal[]} // Simplify //
          Chop
Out[26]= True
```

Here is a second-order nonlinear ODE, based on equation 6.93 on page 213 of [K59].

```
In[27]:= SecondOrderODE = x^3 * y''[x] == 6 * (x * y'[x] - y[x])^2;
```

This equation is invariant under the following scaling transformation.

```
In[28]:= m = x * E^t;

In[29]:= n = y * E^t;
```

The presence of this *scaling symmetry* allows `DSolve` to find new coordinates in which the independent variable is not explicitly present. Hence the problem is solved easily.

```
In[30]:= sol = DSolve[SecondOrderODE, y, x]
```

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

$$\text{Out[30]= } \left\{\left\{\text{y} \to \text{Function}\left[\{\text{x}\}, -\frac{1}{6}\,\text{x}\,\text{Log}\left[6\left(-\frac{\text{C[1]}}{\text{x}} - \text{C[2]}\right)\right]\right]\right\}\right\}$$

This verifies the solution.

```
In[31]:= SecondOrderODE /. sol // Simplify
Out[31]= {True}
```

Finally, here is a system of two nonlinear first-order ODEs that can be solved by using a shift: $u[x] \to u[x] - x$. After the shift, the system becomes autonomous (it does not depend explicitly on x) and hence it can be solved by reduction to a first-order ODE for v as a function of u. The `Solve::ifun` message can be ignored; it is generated while inverting the expression for `Exp[v]` to give an expression in terms of `Log`.

```
In[32]:= Clear[u, v]

In[33]:= NonlinearSystem = {u'[x] == Exp[v[x]] + 1, v'[x] == u[x] - x};

In[34]:= sol = DSolve[NonlinearSystem, {u, v}, x]
```

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

$$\text{Out[34]= } \left\{\left\{\text{v} \to \text{Function}\left[\{\text{x}\}, \text{Log}\left[\text{C[1]} + \text{C[1]}\,\text{Tan}\left[\frac{1}{2}\left(\sqrt{2}\,\text{x}\sqrt{\text{C[1]}} + 2\sqrt{2}\,\sqrt{\text{C[1]}}\,\text{C[2]}\right)\right]^2\right]\right],\right.$$
$$\left.\text{u} \to \text{Function}\left[\{\text{x}\}, \text{x} + \sqrt{2}\,\sqrt{\text{C[1]}}\,\text{Tan}\left[\frac{1}{2}\left(\sqrt{2}\,\text{x}\sqrt{\text{C[1]}} + 2\sqrt{2}\,\sqrt{\text{C[1]}}\,\text{C[2]}\right)\right]\right]\right\}\right\}$$

```
In[35]:= NonlinearSystem /. sol // Simplify
Out[35]= {{True, True}}

In[36]:= Clear[m, n, u, v]
```

This concludes the discussion of ordinary differential equations.

Partial Differential Equations (PDEs)

Introduction to Partial Differential Equations (PDEs)

A partial differential equation (PDE) is a relationship between an unknown function $u(x_1, x_2, \ldots, x_n)$ and its derivatives with respect to the variables $x_1, x_2, \ldots, x_n$.

Here is an example of a PDE.

```
In[1]:= equation1 = ∂u(x, y)/∂x + x ∂u(x, y)/∂y == sin(x);
```

PDEs occur naturally in applications; they model the rate of change of a physical quantity with respect to both space variables and time variables. At this stage of development, `DSolve` typically only works with PDEs having two independent variables.

The order of a PDE is the order of the highest derivative that occurs in it. The previous equation is a first-order PDE.

A function $u(x, y)$ is a *solution* to a given PDE if u and its derivatives satisfy the equation.

Here is one solution to the previous equation.

```
In[2]:= sol = u /. DSolve[equation1, u, {x, y}][[1]] /. C[1][t_] → t
Out[2]= Function[{x, y}, -Cos[x] + 1/2 (-x^2 + 2 y)]
```

This verifies the solution.

```
In[3]:= equation1 /. {u → sol}
Out[3]= True
```

Here are some well-known examples of PDEs (clicking a link in the table will bring up the relevant examples). `DSolve` gives symbolic solutions to equations of all these types, with certain restrictions, particularly for second-order PDEs.

name of equation	*general form*	*classification*
transport equation	$\frac{\partial u}{\partial x} + c\,\frac{\partial u}{\partial y} = 0$ with c constant	linear first-order PDE
Burgers' equation	$\frac{\partial u}{\partial t} + u\,\frac{\partial u}{\partial x} = 0$	quasilinear first-order PDE
eikonal equation	$\left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial u}{\partial y}\right)^2 = 1$	nonlinear first-order PDE
Laplace's equation	$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0$	elliptic linear second-order PDE
wave equation	$\frac{\partial^2 u}{\partial x^2} = c^2\,\frac{\partial^2 u}{\partial t^2}$ where c is the speed of light	hyperbolic linear second-order PDE
heat equation	$\frac{\partial^2 u}{\partial x^2} = k\,\frac{\partial u}{\partial t}$ where k is the thermal diffusivity	parabolic linear second-order PDE

Recall that the general solutions to PDEs involve arbitrary *functions* rather than arbitrary *constants*. The reason for this can be seen from the following example.

The partial derivative with respect to y does not appear in this example, so an arbitrary function `C[1][y]` can be added to the solution, since the partial derivative of `C[1][y]` with respect to x is 0.

```
In[4]:= DSolve[D[u[x, y], x] == 1, u[x, y], {x, y}]
Out[4]= {{u[x, y] → x + C[1][y]}}
```

If there are several arbitrary functions in the solution, they are labeled as `C[1]`, `C[2]`, and so on.

First-Order PDEs

Linear and Quasi-Linear PDEs

First-order PDEs are usually classified as linear, quasi-linear, or nonlinear. The first two types are discussed in this tutorial.

A first-order PDE for an unknown function $u(x, y)$ is said to be *linear* if it can be expressed in the form

$$a(x, y)\,\frac{\partial u(x, y)}{\partial x} + b(x, y)\,\frac{\partial u(x, y)}{\partial y} + c(x, y)\,u(x, y) = d(x, y).$$

The PDE is said to be *quasilinear* if it can be expressed in the form

$$a(x, y, u(x, y))\,\frac{\partial u(x, y)}{\partial x} + b(x, y, u(x, y))\,\frac{\partial u(x, y)}{\partial y} = c(x, y, u(x, y)).$$

A PDE which is neither linear nor quasi-linear is said to *nonlinear*.

For convenience, the symbols z, p, and q are used throughout this tutorial to denote the unknown function and its partial derivatives.

$$z = u(x, y);\; p = \frac{\partial u(x, y)}{\partial x};\; q = \frac{\partial u(x, y)}{\partial y}$$

Here is a linear homogeneous first-order PDE with constant coefficients.

```
In[1]:= z := u[x, y]
In[2]:= p := D[u[x, y], x]
In[3]:= q := D[u[x, y], y]
In[4]:= eqn = 2 * p + 3 * q + z == 0;
```

The equation is linear because the left-hand side is a linear polynomial in z, p, and q. Since there is no term free of z, p, or q, the PDE is also homogeneous.

As mentioned earlier, the general solution contains an arbitrary function `C[1]` of the argument $\frac{1}{2}(2y - 3x)$.

```
In[5]:= sol = DSolve[eqn, u, {x, y}]
Out[5]= {{u → Function[{x, y}, e^(-x/2) C[1][1/2 (-3 x + 2 y)]]}}
```

This verifies that the solution is correct.

```
In[6]:= eqn /. sol[[1]] // Simplify
Out[6]= True
```

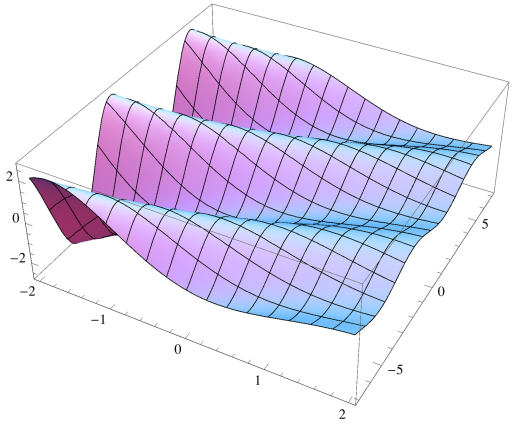
Particular solutions of the homogeneous PDE are obtained by specifying the function `C[1]`.

```
In[7]:= particularsolution = u[x, y] /. sol[[1]] /. C[1][a_] → Sin[a]
Out[7]= e^(-x/2) Sin[1/2 (-3 x + 2 y)]
```

Here is a plot of the surface for this particular solution.

```
In[8]:= Plot3D[particularsolution, {x, -2, 2}, {y, -7, 7}, PlotPoints → 30]
```

Out[8]=



The *transport equation* is a good example of a linear first-order homogeneous PDE with constant coefficients.

In this transport equation, $c = 1$ for convenience.

```
In[9]:= DSolve[D[u[x, y], x] + D[u[x, y], y] == 0, u[x, y], {x, y}]
Out[9]= {{u[x, y] → C[1][-x + y]}}
```

Note that the solution to the transport equation is constant on any straight line of the form $y = x + \alpha$ in the plane. These straight lines are called the *base characteristic curves*. The equation $y = x + \alpha$ defines a plane in three dimensions. The intersections of these planes with the solution surface are called *characteristic curves*. Since the characteristic curves are solutions to a system of ODEs, the problem of solving the PDE is reduced to that of solving a system of ODEs for $x(t)$, $y(t)$, and $u(t)$, where t is a parameter along the characteristic curves. These ODEs are called characteristic ODEs.

The solution to an inhomogeneous PDE has two components: the general solution to the homogeneous PDE and a particular solution to the inhomogeneous PDE.

This is a linear inhomogeneous PDE of the first order.

```
In[10]:= eqn = 7 * p + 3 * q + z == x + y;
```

The first part of the solution, $-10 + x + y$, is the particular solution to the inhomogeneous PDE. The rest of the solution is the general solution to the homogenous equation.

```
In[11]:= sol = u[x, y] /. DSolve[eqn, u[x, y], {x, y}][[1]] // Expand
```

$$\text{Out[11]= } -10 + x + y + e^{-x/7}\, C[1]\left[\frac{1}{7}\,(-3\,x + 7\,y)\right]$$

Here is a linear homogeneous PDE with variable coefficients.

```
In[12]:= eqn = Sin[x] * p + E^x * q == 0;
```

```
In[13]:= sol = DSolve[eqn, u, {x, y}]
```

$$\text{Out[13]= } \left\{\left\{u \to \text{Function}\left[\{x, y\}, C[1]\left[y + (1 + i)\, e^{(1+i)\,x}\, \text{Hypergeometric2F1}\left[\frac{1}{2} - \frac{i}{2}, 1, \frac{3}{2} - \frac{i}{2}, e^{2\,i\,x}\right]\right]\right]\right\}\right\}$$

This verifies the solution.

```
In[14]:= eqn /. sol[[1]] // Simplify
```

```
Out[14]= True
```

Here is a linear inhomogeneous PDE with variable coefficients.

```
In[15]:= eqn = p + x * q == Cos[x];
```

The solution is once again composed of the general solution to the homogeneous PDE and a particular solution, `Sin[x]`, to the inhomogeneous PDE.

```
In[16]:= sol = DSolve[eqn, u, {x, y}]
```

$$\text{Out[16]= } \left\{\left\{u \to \text{Function}\left[\{x, y\}, \text{Sin}[x] + C[1]\left[\frac{1}{2}\,(-x^2 + 2\,y)\right]\right]\right\}\right\}$$

Now consider some examples of *first-order quasi-linear PDEs*.

This PDE is quasi-linear because of the term z^2 on the right-hand side.

```
In[17]:= eqn = p + x * q == z^2 + 5;
```

```
In[18]:= sol = DSolve[eqn, u, {x, y}]
```

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

$$\text{Out[18]= } \left\{\left\{u \to \text{Function}\left[\{x, y\}, \sqrt{5}\ \text{Tan}\left[\sqrt{5}\ x + \sqrt{5}\ C[1]\left[\frac{1}{2}\,(-x^2 + 2\,y)\right]\right]\right]\right\}\right\}$$

This verifies the solution.

```
In[19]:= eqn /. sol[[1]] // Simplify
```

```
Out[19]= True
```

Burgers' equation is an important example of a quasi-linear PDE.

$$\frac{\partial u(x, y)}{\partial x} + u(x, y) \frac{\partial u(x, y)}{\partial y} = 0$$

It can be written using the notation introduced earlier.

```
In[20]:= BurgersEquation = p + z q == 0;
```

The term $z\,q$ makes this equation quasi-linear.

This solves the equation.

```
In[21]:= sol = DSolve[BurgersEquation, u, {x, y}]
Out[21]= Solve[C[1][u[x, y], y - x u[x, y]] == 0, u[x, y]]
```

This verifies the solution to Burgers' equation.

```
In[22]:= p1 = p /. Solve[D[sol[[1]], x], p][[1]];
In[23]:= q1 = q /. Solve[D[sol[[1]], y], q][[1]];
In[24]:= p1 + z * q1
Out[24]= 0
```

A practical consequence of quasi-linearity is the appearance of shocks and steepening and breaking of solutions. Thus, although the procedures for finding general solutions to linear and quasi-linear PDEs are quite similar, there are sharp differences in the nature of the solutions.

Nonlinear PDEs

The general first-order nonlinear PDE for an unknown function $u(x, y)$ is given by

$$F(u, p, q) = 0.$$

Here F is a function of $u = u(x, y)$, $p = \frac{\partial u(x,y)}{\partial x}$, and $q = \frac{\partial u(x,y)}{\partial y}$.

The term "nonlinear" refers to the fact that F is a nonlinear function of p and q. For instance, the eikonal equation involves a quadratic expression in p and q.

The general solution to a first-order linear or quasi-linear PDE involves an arbitrary function. If the PDE is nonlinear, a very useful solution is given by the complete integral. This is a function of $u(x, y, C[1], C[2])$, where `C[1]` and `C[2]` are independent parameters and u satisfies the PDE for all values of `(C[1], C[2])` in an open subset of the plane. The complete integral can be used to find a general solution for the PDE as well as to solve initial value problems for it.

Here is a simple nonlinear PDE.

```
In[1]:= z := u[x, y]
In[2]:= p := D[u[x, y], x]
In[3]:= q := D[u[x, y], y]
In[4]:= eqn = p * q == 1;
```

The complete integral depends on the parameters `C[1]` and `C[2]`. Since `DSolve` returns a general solution for linear and quasi-linear PDEs, a warning message appears before a complete integral is returned.

```
In[5]:= sol = DSolve[eqn, u, {x, y}]
```

```
DSolve::nlpde:
  Solution requested to nonlinear partial differential equation. Trying to build a special solution. >>
```

$$\text{Out[5]= } \left\{\left\{u \to \text{Function}\left[\{x, y\}, C[1] + \frac{x}{C[2]} + y\, C[2]\right]\right\}\right\}$$

This verifies the solution.

```
In[6]:= eqn /. sol
Out[6]= {True}
```

If the values of `C[1]` and `C[2]` are fixed, the previous solution represents a plane in three dimensions. Thus, the complete integral for this PDE is a two-parameter family of planes, each of which is a solution surface for the equation.

Next, the envelope of a one-parameter family of surfaces is a surface that touches each member of the family. If the complete integral is restricted to a one-parameter family of planes, for example by setting `C[2] = 5 C[1]`, the envelope of this family is also a solution to the PDE called a general integral.

This finds the envelope of the one-parameter family given by setting C[2] = 5 C[1] in the complete integral for the preceding PDE p * q == 1.

```
In[7]:= oneparametersol = u[x, y] == (u[x, y] /. sol[[1]] /. C[2] → 5 * C[1])
```

$$\text{Out[7]= } u[x, y] == \frac{x}{5\,C[1]} + C[1] + 5\,y\,C[1]$$

```
In[8]:= oneparameterenvelope =
         Eliminate[{oneparametersol, D[oneparametersol, C[1]]}, {C[1]}]
```

$$\text{Out[8]= } 5\,u[x, y]^2 == x\,(4 + 20\,y)$$

This verifies that the envelope surface is a solution to the PDE.

```
In[9]:= p1 = D[u[x, y], x] /. Solve[D[oneparameterenvelope, x], D[u[x, y], x]][[1]]
```

$$\text{Out[9]= } \frac{2\,(1 + 5\,y)}{5\,u[x, y]}$$

```
In[10]:= q1 = D[u[x, y], y] /. Solve[D[oneparameterenvelope, y], D[u[x, y], y]][[1]]
```

$$\text{Out[10]= } \frac{2\,x}{u[x, y]}$$

```
In[11]:= FullSimplify[p1 * q1, {oneparameterenvelope}]
Out[11]= 1
```

Like nonlinear ODEs, some nonlinear PDEs also have a singular solution (or singular integral) that is obtained by constructing the envelope of the entire two-parameter family of surfaces represented by the complete integral.

Here is an example of such a construction, (equation 6.4.13, page 429 of [K00]).

```
In[12]:= sol = DSolve[4 z + p^2 + q^2 == 4, u, {x, y}]
```

DSolve::nlpde:
Solution requested to nonlinear partial differential equation. Trying to build a special solution. ≫

$$\text{Out[12]= } \left\{\left\{u \to \text{Function}\left[\{x, y\}, \frac{1 - y^2 - 2\,x\,y\,C[1] + C[1]^2 - x^2\,C[1]^2 - 2\,y\,C[2] - 2\,x\,C[1]\,C[2] - C[2]^2}{1 + C[1]^2}\right]\right\}\right\}$$

```
In[13]:= twoparameterfamily = u[x, y] == (u[x, y] /. sol[[1]])
```

$$\text{Out[13]= } u[x, y] == \frac{1 - y^2 - 2\,x\,y\,C[1] + C[1]^2 - x^2\,C[1]^2 - 2\,y\,C[2] - 2\,x\,C[1]\,C[2] - C[2]^2}{1 + C[1]^2}$$

```
In[14]:= envelopeoftwoparameterfamily =
          Eliminate[{twoparameterfamily, D[twoparameterfamily, C[1]],
            D[twoparameterfamily, C[2]]}, {C[1], C[2]}]
Out[14]= u[x, y] == 1
```

Thus, the singular integral for this PDE is a plane parallel to the x-y plane.

To summarize, the complete integral for a nonlinear PDE includes a rich variety of solutions.

- Every member of the two-parameter family gives a particular solution to the PDE.
- The envelope of any one-parameter family is a solution called a general integral of the PDE.
- The envelope of the entire two-parameter family is a solution called the singular integral of the PDE.
- The complete integral is not unique, but any other complete integral for the PDE can be obtained from it by the process of envelope formation.

These remarkable properties account for the usefulness of the complete integral in geometrical optics, dynamics, and other areas of application. Following are various examples of nonlinear PDEs that show different kinds of complete integrals.

Here is the complete integral for the eikonal equation.

```
In[15]:= Eikonal = D[u[x, y], x]^2 + D[u[x, y], y]^2 == 1;

In[16]:= sol = DSolve[Eikonal, u, {x, y}]
```

```
DSolve::nlpde:
  Solution requested to nonlinear partial differential equation. Trying to build a special solution. >>
```

$$\text{Out[16]}= \left\{\left\{u \to \text{Function}\left[\{x, y\}, C[1] + y\, C[2] - x\sqrt{1 - C[2]^2}\right]\right\},\ \left\{u \to \text{Function}\left[\{x, y\}, C[1] + y\, C[2] + x\sqrt{1 - C[2]^2}\right]\right\}\right\}$$

This complete integral is a two-parameter family of planes. This type of solution arises whenever the PDE depends explicitly only on p and q, but not on $u[x, y]$, x, or y. For a fixed value of $u[x, y]$, it is a line in the plane at a distance of `C[1]` units from the origin that makes an angle of `ArcCos[C[2]]` with the x axis. This is the familiar picture of wave-front propagation from geometrical optics.

This verifies the solution for the eikonal equation.

```
In[17]:= Eikonal /. sol
Out[17]= {True, True}
```

This is an example of a Clairaut equation ($z = p\,x + q\,y + f(p, q)$).

```
In[18]:= Clairaut = z == x * p + y * q + 2 p * q * Sqrt[1 - p^2];
```

Once again, the complete integral is a family of planes.

In[19]:=
```
sol = DSolve[Clairaut, u, {x, y}]
```

DSolve::nlpde:
Solution requested to nonlinear partial differential equation. Trying to build a special solution. ≫

Out[19]=
$$\left\{\left\{u \to \text{Function}\left[\{x, y\}, x\,C[1] + y\,C[2] + 2\,C[1]\sqrt{1 - C[1]^2}\;C[2]\right]\right\}\right\}$$

This verifies the solution.

In[20]:=
```
Clairaut /. sol
```

Out[20]=
```
{True}
```

In the following equation, the variables can be separated; that is, the PDE can be written in the form $f(x, p) = g(y, q)$. Hence, the equation can be integrated easily.

In[21]:=
```
Separable = p^2 + a * q == x + 3 y;
```

In[22]:=
```
sol = DSolve[Separable, u, {x, y}]
```

DSolve::nlpde:
Solution requested to nonlinear partial differential equation. Trying to build a special solution. ≫

Out[22]=
$$\left\{\left\{u \to \text{Function}\left[\{x, y\}, \frac{3\,y^2}{2\,a} - \frac{2}{3}(x - C[1])^{3/2} + \frac{y\,C[1]}{a} + C[2]\right]\right\},\right.$$
$$\left.\left\{u \to \text{Function}\left[\{x, y\}, \frac{3\,y^2}{2\,a} + \frac{2}{3}(x - C[1])^{3/2} + \frac{y\,C[1]}{a} + C[2]\right]\right\}\right\}$$

This verifies the solution.

In[23]:=
```
Separable /. sol // Simplify
```

Out[23]=
```
{True, True}
```

In this example (equation 6.49, page 202 of [K74]), the independent variables x and y are not explicitly present.

In[24]:=
```
MissingIndependentVariables = a * p^2 + b * p * q == c * z^2;
```

In[25]:=
```
sol = DSolve[MissingIndependentVariables, u, {x, y}]
```

DSolve::nlpde:
Solution requested to nonlinear partial differential equation. Trying to build a special solution. ≫

Out[25]=
$$\left\{\left\{u \to \text{Function}\left[\{x, y\}, e^{-\frac{i\sqrt{c}\,y}{\sqrt{-b\,C[1]-a\,C[1]^2}}-\frac{i\sqrt{c}\,x\,C[1]}{\sqrt{-b\,C[1]-a\,C[1]^2}}-\frac{i\sqrt{c}\,C[2]}{\sqrt{-b\,C[1]-a\,C[1]^2}}}\right]\right\}\right\}$$

This verifies the solution.

```
In[26]:= MissingIndependentVariables /. sol // Simplify
Out[26]= {True}
```

Often a coordinate transformation can be used to cast a given PDE into one of the previous types. The expression for the complete integral will then have the same form as for the standard types. Here are some examples of nonlinear PDEs for which `DSolve` applies a coordinate transformation to find the complete integral.

This PDE (equation 6.47, page 201 of [K74]) can be reduced to the form $f(p, q) = 0$ using the transformation $X = \log(x)$ and $Y = \log(y)$.

```
In[27]:= DSolve[x * y * p * q == 1, u[x, y], {x, y}]
```

```
DSolve::nlpde:
  Solution requested to nonlinear partial differential equation. Trying to build a special solution. >>
```

$$\text{Out[27]}= \left\{\left\{u[x, y] \to C[2] + C[1]\,\text{Log}[x] + \frac{\text{Log}[y]}{C[1]}\right\}\right\}$$

This PDE (equation 6.93, page 213 of [K74]) can be solved easily in a polar coordinate system, in which the variables are separable.

```
In[28]:= DSolve[(y * p - x * q) ^ 2 + a * (x * p + y * q) == b, z, {x, y}]
```

```
DSolve::nlpde:
  Solution requested to nonlinear partial differential equation. Trying to build a special solution. >>
```

$$\text{Out[28]}= \left\{\left\{u[x, y] \to -\text{ArcTan}\left[\frac{y}{x}\right]\sqrt{C[1]} + C[2] + \frac{b\,\text{Log}\left[\sqrt{x^2 + y^2}\right]}{a} - \frac{C[1]\,\text{Log}\left[\sqrt{x^2 + y^2}\right]}{a}\right\}\right\}$$

This equation (equation 6.36, page 196 of [K74]) can be transformed into a linear PDE using a Legendre transformation.

```
In[29]:= LegendreTransformable = y * p * q - z * p + a * q == 0;

In[30]:= sol = DSolve[LegendreTransformable, u, {x, y}]
```

```
DSolve::nlpde:
  Solution requested to nonlinear partial differential equation. Trying to build a special solution. >>
```

$$\text{Out[30]}= \left\{\left\{u \to \text{Function}\left[\{x, y\},\right.\right.$$
$$\left.\frac{1}{4\,C[1]}\left(-y\,(-2x + 2C[2]) - \sqrt{y^2\,(-2x + 2C[2])^2 + 8\,a\,C[1]\,\left(x^2 - 2x\,C[2] + C[2]^2\right)}\right)\right]\},$$
$$\left\{u \to \text{Function}\left[\{x, y\},\ \frac{1}{4\,C[1]}\left(-y\,(-2x + 2C[2]) + \sqrt{y^2\,(-2x + 2C[2])^2 + 8\,a\,C[1]\,\left(x^2 - 2x\,C[2] + C[2]^2\right)}\right)\right]\right\}\right\}$$

This verifies the solution.

```
In[31]:= LegendreTransformable /. sol // Simplify
Out[31]= {True, True}
```

It should be noted that there is no general practical algorithm for finding complete integrals, and that the answers are often available only in implicit form.

The solution to this example (problem 2, page 66 of [S57]) is in implicit form.

```
In[32]:= sol = DSolve[(1 + q^2) * z == p * x, u, {x, y}]
```

```
DSolve::nlpde:
  Solution requested to nonlinear partial differential equation. Trying to build a special solution. >>

Solve::tdep:
  The equations appear to involve the variables to be solved for in an essentially non-algebraic way. >>

Solve::tdep:
  The equations appear to involve the variables to be solved for in an essentially non-algebraic way. >>
```

$$\text{Out[32]= } \text{Solve}\left[\frac{1}{2}\left(\text{C[1]}\,\text{Log}\left[-\text{C[1]}+\sqrt{\text{C[1]}^2-4\,\text{u[x, y]}^2}\right]+\sqrt{\text{C[1]}^2-4\,\text{u[x, y]}^2}\right)==\text{y}+\text{C[2]}+\text{C[1]}\,\text{Log[x]},\ \text{u[x, y]}\right]$$

The solution can be verified as follows.

```
In[33]:= p = D[u[x, y], x] /. Solve[D[sol[[1]], x], D[u[x, y], x]][[1]]
```

$$\text{Out[33]= } \frac{\text{C[1]}\left(\text{C[1]}-\sqrt{\text{C[1]}^2-4\,\text{u[x, y]}^2}\right)}{2\,\text{x}\,\text{u[x, y]}}$$

```
In[34]:= q = D[u[x, y], y] /. Solve[D[sol[[1]], y], D[u[x, y], y]][[1]]
```

$$\text{Out[34]= } \frac{\text{C[1]}-\sqrt{\text{C[1]}^2-4\,\text{u[x, y]}^2}}{2\,\text{u[x, y]}}$$

```
In[35]:= (1 + q^2) * z - p * x // Simplify
Out[35]= 0
```

Second-Order PDEs

The general form of a linear second-order PDE is

$$a\frac{\partial^2 u}{\partial x^2}+b\frac{\partial^2 u}{\partial x\,\partial y}+c\frac{\partial^2 u}{\partial y^2}+d\frac{\partial u}{\partial x}+e\frac{\partial u}{\partial y}+f\,u=g.$$

Here $u = u(x, y)$, and a, b, c, d, e, f, and g are functions of x and y only—they do not depend on u. If $g = 0$, the equation is said to be homogeneous.

The first three terms containing the second derivatives are called the *principal part* of the PDE. They determine the nature of the general solution to the equation. In fact, the coefficients of the principal part can be used to classify the PDE as follows.

The PDE is said to be *elliptic* if $b^2 - 4\,a\,c < 0$. The Laplace equation has $a = 1$, $b = 0$, and $c = 1$ and is therefore an elliptic PDE.

The PDE is said to be *hyperbolic* if $b^2 - 4\,a\,c > 0$. The wave equation has $a = 1$, $b = 0$, and $c = -1$ and is therefore a hyperbolic PDE.

The PDE is said to be *parabolic* if $b^2 - 4\,a\,c = 0$. The heat equation has $a = 1$, $b = 0$, and $c = 0$ and is therefore a parabolic PDE.

`DSolve` can find the general solution for a restricted type of homogeneous linear second-order PDEs; namely, equations of the form

$$a\,\frac{\partial^2 u}{\partial x^2} + b\,\frac{\partial^2 u}{\partial x\,\partial y} + c\,\frac{\partial^2 u}{\partial y^2} = 0.$$

Here a, b, and c are constants. Thus, `DSolve` assumes that the equation has constant coefficients and a vanishing non-principal part.

Following are some examples of the three basic types (elliptic, hyperbolic, and parabolic) and an explanation of their significance.

Here is the general solution for Laplace's equation, an elliptic PDE.

```
In[1]:= LaplaceEquation = D[u[x, y], {x, 2}] + D[u[x, y], {y, 2}] == 0;

In[2]:= DSolve[LaplaceEquation, u[x, y], {x, y}]
Out[2]= {{u[x, y] → C[1][i x + y] + C[2][-i x + y]}}
```

This general solution contains two arbitrary functions, `C[1]` and `C[2]`. The arguments of these functions, $y + i\,x$ and $y - i\,x$, indicate that the solution is constant along the imaginary straight line $y = -i\,x + \alpha$ when `C[2] == 0` and along $y = i\,x + \alpha$ when `C[1] == 0` . These straight lines are called characteristic curves of the PDE. In general, elliptic PDEs have imaginary characteristic curves.

Here is another elliptic PDE.

```
In[3]:= a = 3; b = 1; c = 5; b^2 - 4 a * c
Out[3]= -59
```

```
In[4]:= eqn = a * D[u[x, y], {x, 2}] + b * D[u[x, y], x, y] + c * D[u[x, y], {y, 2}] == 0;
```

Note the imaginary characteristic curves for the equation.

```
In[5]:= sol = DSolve[eqn, u, {x, y}]
Out[5]= {{u → Function[{x, y}, C[1][1/6 (-1 + i Sqrt[59]) x + y] + C[2][1/6 (-1 - i Sqrt[59]) x + y]]}}
```

The solution is verified as follows.

```
In[6]:= eqn /. sol // Simplify
Out[6]= {True}
```

This finds the general solution of the wave equation, a hyperbolic PDE. The constant c in the wave equation represents the speed of light and is set to 1 here for convenience.

```
In[7]:= WaveEquation = D[u[x, t], {x, 2}] - D[u[x, t], {t, 2}] == 0;
```

```
In[8]:= DSolve[WaveEquation , u[x, t], {t, x}]
Out[8]= {{u[x, t] → C[1][-t + x] + C[2][t + x]}}
```

The characteristic lines for the wave equation are $x = k + t$ and $x = k - t$ where k is an arbitrary constant. Hence the wave equation (or any hyperbolic PDE) has two families of real characteristic curves. If initial conditions are specified for the wave equation, the solution propagates along the characteristic lines. Also, any fixed pair of characteristic lines determine the null cone of an observer sitting at their intersection.

Here is another example of a hyperbolic PDE.

```
In[9]:= a = 2; b = 7; c = -1; b^2 - 4 a * c
Out[9]= 57
```

```
In[10]:= eqn = a * D[u[x, y], {x, 2}] + b * D[u[x, y], x, y] + c * D[u[x, y], {y, 2}] == 0;
```

Notice that the equation has two families of real characteristics.

```
In[11]:= sol = DSolve[eqn, u, {x, y}]
Out[11]= {{u → Function[{x, y}, C[1][-1/4 (7 + Sqrt[57]) x + y] + C[2][-1/4 (7 - Sqrt[57]) x + y]]}}
```

The solution can be verified as follows.

```
In[12]:= eqn /. sol // Simplify
Out[12]= {True}
```

Finally, here is an example of a parabolic PDE.

```
In[13]:= a = 3; b = 30; c = 75; b^2 - 4 a * c
Out[13]= 0

In[14]:= eqn = a * D[u[x, y], {x, 2}] + b * D[u[x, y], x, y] + c * D[u[x, y], {y, 2}] == 0;

In[15]:= sol = DSolve[eqn, u, {x, y}]
Out[15]= {{u → Function[{x, y}, C[1][-5 x + y] + x C[2][-5 x + y]]}}
```

The equation has only one family of real characteristics, the lines $y = 5x + \alpha$. In fact, any parabolic PDE has only a single family of real characteristics.

The solution can be verified as follows.

```
In[16]:= eqn /. sol // Simplify
Out[16]= {True}
```

The heat equation is parabolic, but it is not considered here because it has a nonvanishing non-principal part, and the algorithm used by `DSolve` is not applicable in this case.

Differential-Algebraic Equations (DAEs)

Introduction to Differential-Algebraic Equations (DAEs)

The systems of equations that govern certain phenomena (in electrical circuits, chemical kinetics, etc.) contain a combination of differential equations and algebraic equations. The differential equations are responsible for the dynamical evolution of the system, while the algebraic equations serve to constrain the solutions to certain manifolds. It is therefore of some interest to study the solutions of such *differential-algebraic equations* (DAEs).

Here is a simple example of a DAE. The first equation is an ODE for the function $x[t]$, while the second equation constrains the functions $x[t]$ and $y[t]$ to lie in a submanifold (a straight line) in $\{x, y\}$ space.

```
In[1]:= dae = {x'[t] == y[t], x[t] + y[t] == 1};
```

These tutorials are restricted to *linear DAEs,* which are defined as systems of equations of the following type.

$$A.x'(t) + B.x(t) = F$$

Here A and B are matrix functions of the independent variable t, F is a vector function of t, and $x(t)$ is the vector of unknowns. If the matrix A is nonsingular (that is, invertible) then this is a system of ODEs. Thus, the system is a DAE if the matrix A is *singular*.

If $F = 0$, then the system is said to be *homogeneous*. As for ODEs, the general solution to a DAE is composed of the general solution to the corresponding homogeneous problem and a particular solution to the inhomogeneous system.

`DSolve` can find the solutions to all DAEs in which the entries of the matrices A and B are constants. Such DAEs are said to have constant coefficients. The algorithm used by `DSolve` is based on decomposing both A and B into a nonsingular and nilpotent part. This decomposition is used to calculate a generalized inverse for A and B, which effectively reduces the problem to solving a system of ODEs.

It is important to realize that the initial values for a DAE must be prescribed carefully to guarantee a solution for the problem. This can be seen by considering the following system of equations.

$$\{x_1(t) + x_2'(t) = 0,\ x_2(t) = 0,\ x_1(0) = 1,\ x_2(0) = 0\}$$

This gives

$$x_2(t) = 0 \Longrightarrow x_2'(t) = 0 \Longrightarrow x_1(t) = 0.$$

Hence the only solution is

$$x_1(t) = 0 \text{ and } x_2(t) = 0.$$

But this solution is inconsistent with the initial condition $x_1(0) = 1$.

`DSolve` can solve DAEs with constant coefficients; see "Examples of DAEs".

Examples of DAEs

This is a simple homogeneous DAE with constant coefficients.

```
In[1]:= eqns = {x'[t] - y[t] == 0, x[t] + y[t] == 0};
```

This finds the general solution. It has only one arbitrary constant because the second equation in the system specifies the relationship between $x[t]$ and $y[t]$.

```
In[2]:= sol = DSolve[eqns, {x, y}, t]
```

$$\text{Out[2]= } \left\{\left\{\text{x} \to \text{Function}\left[\{\text{t}\},\ \frac{1}{4}\, e^{-t}\, \text{C}[1]\right],\ \text{y} \to \text{Function}\left[\{\text{t}\},\ -\frac{1}{4}\, e^{-t}\, \text{C}[1]\right]\right\}\right\}$$

This verifies the solution.

```
In[3]:= eqns /. sol // Simplify
Out[3]= {{True, True}}
```

Here is an inhomogeneous system derived from the previous example.

```
In[4]:= eqns = {x'[t] - y[t] == Sin[t], x[t] + y[t] == 1};
```

The general solution is composed of the general solution to the corresponding homogeneous system and a particular solution to the inhomogeneous equation.

```
In[5]:= sol = {x[t], y[t]} /. DSolve[eqns, {x[t], y[t]}, t][[1]] // Expand
```

$$\text{Out[5]}= \left\{1+\frac{1}{4}e^{-t}C[1]-\frac{\text{Cos}[t]}{2}+\frac{\text{Sin}[t]}{2},\ -\frac{1}{4}e^{-t}C[1]+\frac{\text{Cos}[t]}{2}-\frac{\text{Sin}[t]}{2}\right\}$$

This solves an initial value problem for the previous equation.

```
In[6]:= eqns = {x'[t] - y[t] == Sin[t], x[t] + y[t] == 1, x[Pi] == 1 / 2};
```

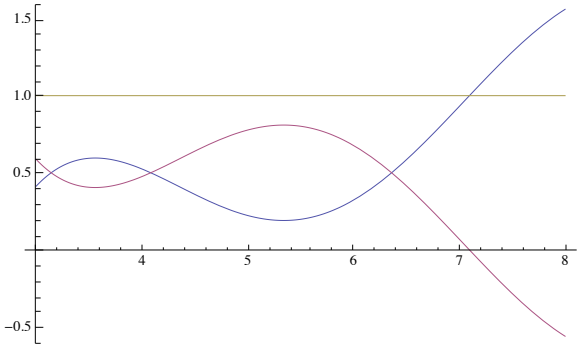
```
In[7]:= sol = DSolve[eqns, {x, y}, t]
```

$$\text{Out[7]}= \left\{\left\{x\to\text{Function}\left[\{t\},\ -\frac{1}{2}e^{-t}\left(2e^{\pi}-2e^{t}+e^{t}\text{Cos}[t]-e^{t}\text{Sin}[t]\right)\right],\right.\right.$$
$$\left.\left.y\to\text{Function}\left[\{t\},\ \frac{1}{2}e^{-t}\left(2e^{\pi}+e^{t}\text{Cos}[t]-e^{t}\text{Sin}[t]\right)\right]\right\}\right\}$$

Here is a plot of the solution and the constraint (algebraic) condition.

```
In[8]:= Plot[{x[t] /. sol, y[t] /. sol, (x[t] + y[t]) /. sol}, {t, 3, 8}]
```

Out[8]=



In this DAE, the inhomogeneous part is quite general.

```
In[9]:= Clear[x, y, z, f, g, h, t]
```

```
In[10]:= eqns = {x[t] + y'[t] == f[t], 2 y[t] + z'[t] == g[t], 5 z[t] == h[t]};
```

Note that there are no degrees of freedom in the solution (that is, there are no arbitrary constants) because $z[t]$ is given algebraically, and thus $x[t]$ and $y[t]$ can be determined uniquely from $z[t]$ using differentiation.

```
In[11]:= sol = DSolve[eqns, {x, y, z}, t]
```

$$\text{Out[11]}= \left\{\left\{x\to\text{Function}\left[\{t\},\ f[t]-\frac{g'[t]}{2}+\frac{h''[t]}{10}\right],\ y\to\text{Function}\left[\{t\},\ \frac{g[t]}{2}-\frac{h'[t]}{10}\right],\ z\to\text{Function}\left[\{t\},\ \frac{h[t]}{5}\right]\right\}\right\}$$

```
In[12]:= eqns /. sol // Simplify
```

```
Out[12]= {{True, True, True}}
```

In this example, the algebraic constraint is present only implicitly: all three equations contain derivatives of the unknown functions.

```
In[13]:= Clear[x1, x2, x3, t, eqns]

In[14]:= eqns = {x2[t] + 2 x3[t] + x1'[t] - 2 x3'[t] == 0,
    -27 x1[t] - 22 x2[t] - 17 x3[t] - x1'[t] + 2 x3'[t] == 0,
    18 x1[t] + 14 x2[t] + 10 x3[t] + 2 x1'[t] + 3 x2'[t] + 2 x3'[t] == 0};
```

The Jacobian with respect to the derivatives of the unknown functions is singular, so that it is not possible to solve for them.

```
In[15]:= A = D[eqns[[All, 1]], {{x1'[t], x2'[t], x3'[t]}}]
Out[15]= {{1, 0, -2}, {-1, 0, 2}, {2, 3, 2}}

In[16]:= Det[A]
Out[16]= 0
```

The differential-algebraic character of this problem is clear from the smaller number of arbitrary constants (two rather than three) in the general solution.

```
In[17]:= sol = DSolve[eqns, {x1, x2, x3}, t]
```

$$\text{Out[17]}= \left\{\left\{\text{x1} \to \text{Function}\left[\{t\}, -C[1] - 3\, e^{2t/3} C[2] + \frac{3}{2}\left(-1 + e^{2t/3}\right) C[2]\right],\right.\right.$$
$$\text{x2} \to \text{Function}\left[\{t\}, -9\, e^{2t/3} C[2] + 2\left(C[1] - \frac{3}{2}\left(-1 + e^{2t/3}\right) C[2]\right)\right],$$
$$\left.\left.\text{x3} \to \text{Function}\left[\{t\}, -C[1] + 18\, e^{2t/3} C[2] + \frac{3}{2}\left(-1 + e^{2t/3}\right) C[2]\right]\right\}\right\}$$

Systems of equations with higher-order derivatives are solved by reducing them to first-order systems.

Here is the general solution to a homogeneous DAE of order two with constant coefficients.

```
In[18]:= eqns = {x''[t] == y[t], x[t] + 4 y[t] == 0};

In[19]:= sol = DSolve[eqns, {x, y}, t]
```

$$\text{Out[19]}= \left\{\left\{\text{x} \to \text{Function}\left[\{t\}, \frac{176}{125}\left(C[2]\,\text{Cos}\left[\frac{t}{2}\right] - \frac{1}{2} C[1]\,\text{Sin}\left[\frac{t}{2}\right]\right) + \frac{16}{125}\left(C[1]\,\text{Cos}\left[\frac{t}{2}\right] + 2\,C[2]\,\text{Sin}\left[\frac{t}{2}\right]\right)\right],\right.$$
$$\left.\left.\text{y} \to \text{Function}\left[\{t\}, -\frac{44}{125}\left(C[2]\,\text{Cos}\left[\frac{t}{2}\right] - \frac{1}{2} C[1]\,\text{Sin}\left[\frac{t}{2}\right]\right) - \frac{4}{125}\left(C[1]\,\text{Cos}\left[\frac{t}{2}\right] + 2\,C[2]\,\text{Sin}\left[\frac{t}{2}\right]\right)\right]\right\}\right\}$$

```
In[20]:= eqns /. sol // Simplify
Out[20]= {{True, True}}
```

This inhomogeneous system of ODEs is based on the previous example.

```
In[21]:= eqns = {x''[t] == y[t], x[t] + 4 y[t] == 6 Sin[t]};
```

```
In[22]:= sol = DSolve[eqns, {x, y}, t]
```

```
Out[22]= {{x → Function[{t}, 176/125 (C[2] Cos[t/2] - 1/2 C[1] Sin[t/2] + 1/16 (-8 Cos[t] - 22 Sin[t])) +
        16/125 (C[1] Cos[t/2] + 2 C[2] Sin[t/2] + 1/8 (44 Cos[t] - 4 Sin[t]))],
    y → Function[{t}, -44/125 (C[2] Cos[t/2] - 1/2 C[1] Sin[t/2] + 1/16 (-8 Cos[t] - 22 Sin[t])) -
        4/125 (C[1] Cos[t/2] + 2 C[2] Sin[t/2] + 1/8 (44 Cos[t] - 4 Sin[t])) + (3 Sin[t])/2]}}
```

```
In[23]:= eqns /. sol // Simplify
```

```
Out[23]= {{True, True}}
```

Here is an initial value problem for the previous system of equations.

```
In[24]:= eqns = {x''[t] == y[t], x[t] + 4 y[t] == Sin[t], x[Pi] == 1, x'[Pi] == 0};
```

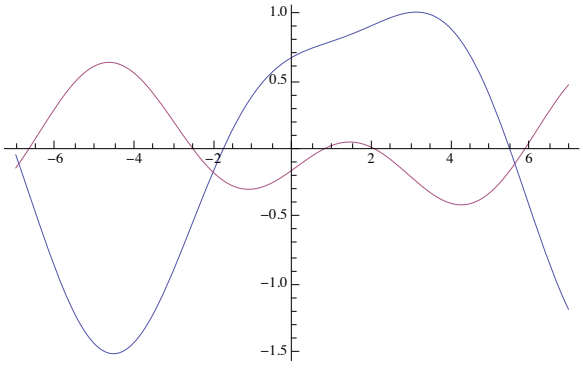
```
In[25]:= sol = DSolve[eqns, {x, y}, t]
```

```
Out[25]= {{x → Function[{t}, 1/3 (2 Cos[t/2] + 3 Sin[t/2] - Sin[t])],
    y → Function[{t}, 1/12 (-2 Cos[t/2] - 3 Sin[t/2] + 4 Sin[t])]}}
```

Here is a plot of the solution.

```
In[26]:= Plot[{x[t] /. sol, y[t] /. sol}, {t, -7, 7}]
```

```
Out[26]=
```



Finally, here is a system with a third-order ODE. Since the coefficients are exact quantities, the computation takes some time.

```
In[27]:= Clear [p, q, r]
```

```
In[28]:= eqns = {p'''[t] - q[t] + r[t] - Sin[t], p''[t] - r[t] - Cos[t],
    p'[t] - q[t] + 4, p[0] - 1, p'[0] - 1, p''[0] - 1};
```

In[29]:=
```
Timing[sol = DSolve[(#1 == 0 &) /@ eqns, {p, q, r}, t]]
```

Out[29]=
$$\left\{9.156,\ \left\{\left\{p \to \text{Function}\left[\{t\},\ -\frac{1}{5\left(-1+\sqrt{5}\right)\left(1+\sqrt{5}\right)}4\left(-410-17\,e^{\frac{1}{2}\left(-1-\sqrt{5}\right)t}+9\sqrt{5}\,e^{\frac{1}{2}\left(-1-\sqrt{5}\right)t}-17\,e^{\frac{1}{2}\left(-1+\sqrt{5}\right)t}-9\sqrt{5}\,e^{\frac{1}{2}\left(-1+\sqrt{5}\right)t}+440\,e^{\frac{t}{2}+\frac{\sqrt{5}\,t}{2}+\frac{1}{2}\left(-1-\sqrt{5}\right)t}+20\,t-34\cos[t]+33\,e^{\frac{t}{2}+\frac{\sqrt{5}\,t}{2}+\frac{1}{2}\left(-1-\sqrt{5}\right)t}\cos[t]-118\sin[t]+121\,e^{\frac{t}{2}+\frac{\sqrt{5}\,t}{2}+\frac{1}{2}\left(-1-\sqrt{5}\right)t}\sin[t]\right)\right],\right.$$
$$q \to \text{Function}\left[\{t\},\ \frac{1}{5}\left(20+14\,e^{\frac{1}{2}\left(-1-\sqrt{5}\right)t}-4\sqrt{5}\,e^{\frac{1}{2}\left(-1-\sqrt{5}\right)t}+14\,e^{\frac{1}{2}\left(-1+\sqrt{5}\right)t}+4\sqrt{5}\,e^{\frac{1}{2}\left(-1+\sqrt{5}\right)t}-20\,e^{\frac{t}{2}+\frac{\sqrt{5}\,t}{2}+\frac{1}{2}\left(-1-\sqrt{5}\right)t}-3\,e^{\frac{t}{2}+\frac{\sqrt{5}\,t}{2}+\frac{1}{2}\left(-1-\sqrt{5}\right)t}\cos[t]-e^{\frac{t}{2}+\frac{\sqrt{5}\,t}{2}+\frac{1}{2}\left(-1-\sqrt{5}\right)t}\sin[t]\right)\right],$$
$$\left.r \to \text{Function}\left[\{t\},\ \frac{1}{5}\left(3\,e^{\frac{1}{2}\left(-1-\sqrt{5}\right)t}-5\sqrt{5}\,e^{\frac{1}{2}\left(-1-\sqrt{5}\right)t}+3\,e^{\frac{1}{2}\left(-1+\sqrt{5}\right)t}+5\sqrt{5}\,e^{\frac{1}{2}\left(-1+\sqrt{5}\right)t}-5\cos[t]-e^{\frac{t}{2}+\frac{\sqrt{5}\,t}{2}+\frac{1}{2}\left(-1-\sqrt{5}\right)t}\cos[t]+3\,e^{\frac{t}{2}+\frac{\sqrt{5}\,t}{2}+\frac{1}{2}\left(-1-\sqrt{5}\right)t}\sin[t]\right)\right]\right\}\right\}\right\}$$

In[30]:=
```
eqns /. sol // Simplify
```

Out[30]=
```
{{0, 0, 0, 0, 0, 0}}
```

The symbolic solution of DAEs that are nonlinear or have non-constant coefficients is a difficult problem. Such systems can often be solved numerically with the *Mathematica* function `NDSolve`.

Initial and Boundary Value Problems

Introduction to Initial and Boundary Value Problems

`DSolve` can be used for finding the general solution to a differential equation or system of differential equations. The general solution gives information about the structure of the complete solution space for the problem. However, in practice, one is often interested only in particular solutions that satisfy some conditions related to the area of application. These conditions are usually of two types.

- The solution $x(t)$ and/or its derivatives are required to have specific values at a single point, for example, $x(0) = 1$ and $x'(0) = 2$. Such problems are traditionally called *initial value problems* (IVPs) because the system is assumed to start evolving from the fixed initial point (in this case, 0).
- The solution $x(t)$ is required to have specific values at a pair of points, for example, $x(0) = 3$ and $x(1) = 5$. These problems are known as *boundary value problems* (BVPs) because the points 0 and 1 are regarded as boundary points (or edges) of the domain of interest in the application.

The symbolic solution of both IVPs and BVPs requires knowledge of the general solution for the problem. The final step, in which the particular solution is obtained using the initial or boundary values, involves mostly algebraic operations, and is similar for IVPs and for BVPs.

IVPs and BVPs for *linear* differential equations are solved rather easily since the final algebraic step involves the solution of linear equations. However, if the underlying equations are *nonlinear*, the solution could have several branches, or the arbitrary constants from the general solution could occur in different arguments of transcendental functions. As a result, it is not always possible to complete the final algebraic step for nonlinear problems. Finally, if the underlying equations have *piecewise* (that is, discontinuous) coefficients, an IVP naturally breaks up into simpler IVPs over the regions in which the coefficients are continuous.

Linear IVPs and BVPs

To begin, consider an initial value problem for a linear first-order ODE.

This is a linear first-order ODE.

In[1]:=
```
linearequation = y'[t] - 3 * t * y[t] == 1;
```

Notice that the general solution is a linear function of the arbitrary constant `C[1]`.

In[2]:=
```
generalsolution = DSolve[linearequation, y[t], t]
```

Out[2]=
$$\left\{\left\{y[t] \to e^{\frac{3 t^2}{2}} C[1] + e^{\frac{3 t^2}{2}} \sqrt{\frac{\pi}{6}}\ \text{Erf}\left[\sqrt{\frac{3}{2}}\ t\right]\right\}\right\}$$

This finds a particular solution for the initial condition $y[0] == 4$.

In[3]:=
```
particularsolution = DSolve[{linearequation, y[0] == 4}, y, t]
```

Out[3]=
$$\left\{\left\{y \to \text{Function}\left[\{t\}, \frac{1}{6} e^{\frac{3 t^2}{2}} \left(24 + \sqrt{6 \pi}\ \text{Erf}\left[\sqrt{\frac{3}{2}}\ t\right]\right)\right]\right\}\right\}$$

This verifies that the solution satisfies both the equation and the initial condition.

In[4]:=
```
linearequation /. particularsolution[[1]]
```

Out[4]=
```
True
```

In[5]:=
```
y[0] /. particularsolution[[1]]
```

Out[5]=
```
4
```

Here is the solution to the same problem with the general initial condition $y[0] == a$.

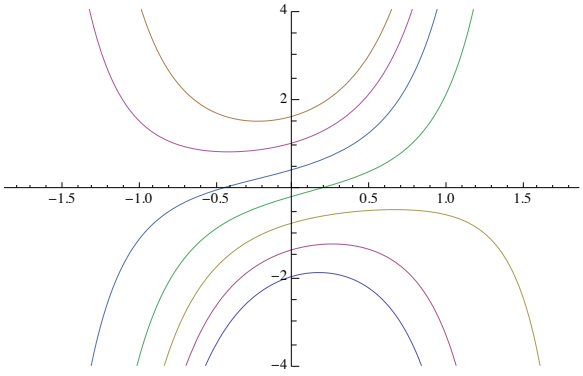
In[6]:=
```
particularsolution = DSolve[{linearequation, y[0] == a}, y, t]
```

Out[6]=
$$\left\{\left\{y \to \text{Function}\left[\{t\}, \frac{1}{6} e^{\frac{3 t^2}{2}} \left(6 a + \sqrt{6 \pi}\ \text{Erf}\left[\sqrt{\frac{3}{2}}\ t\right]\right)\right]\right\}\right\}$$

This plots several integral curves of the equation for different values of a. The plot shows that the solutions have an inflection point if the parameter a lies between -1 and 1, while a global maximum or minimum arises for other values of a.

In[7]:=
```
Plot[Evaluate[Table[y[t] /. particularsolution[[1]] /. a → i, {i, -2, 2, 0.6}]],
 {t, -1.8, 1.8}, PlotRange → {-4, 4}]
```

Out[7]=



Here is the solution to a linear second-order equation with initial values prescribed for $x[t]$ and $x'[t]$ at $t == 0$.

In[8]:=
```
linearsecondorderODE = x''[t] + 5 * x'[t] + 6 * x[t] == 0;
```

In[9]:=
```
generalsolution = DSolve[linearsecondorderODE, x, t]
```

Out[9]=
```
{{x → Function[{t}, e^(-3 t) C[1] + e^(-2 t) C[2]]}}
```

In[10]:=
```
particularsolution = DSolve[{linearsecondorderODE, x[0] == 1, x'[0] == 1}, x, t]
```

Out[10]=
```
{{x → Function[{t}, e^(-3 t) (-3 + 4 e^t)]}}
```

This verifies that the solution satisfies the equation and the initial conditions.

In[11]:=
```
linearsecondorderODE /. particularsolution[[1]] // Simplify
```

Out[11]=
```
True
```

In[12]:=
```
x[0] /. particularsolution[[1]]
```

Out[12]=
```
1
```

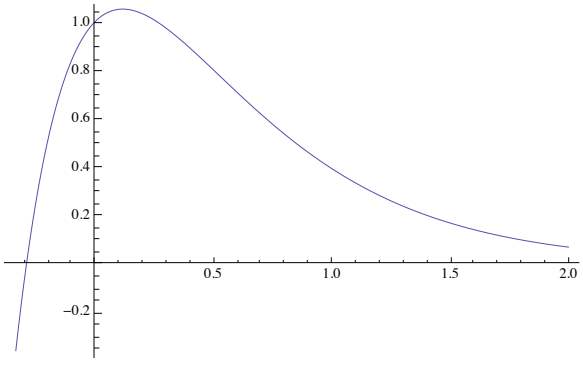
In[13]:=
```
x'[0] /. particularsolution[[1]]
```

Out[13]=
```
1
```

Here is a plot of the solution.

In[14]:=
```
Plot[x[t] /. particularsolution, {t, -1 / 3, 2}]
```

Out[14]=



To get more information about the solutions for the problem, set $x'[0] == x0$.

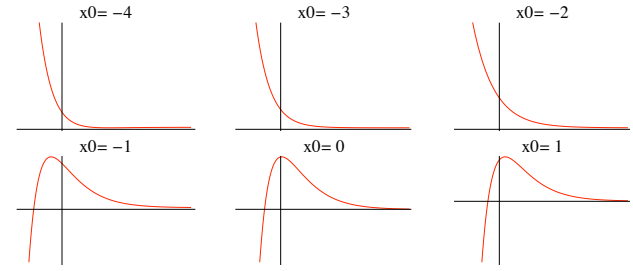
In[15]:=
```
particularsolution = DSolve[{linearsecondorderODE, x[0] == 1, x'[0] == x0}, x, t]
```

Out[15]=
```
{{x → Function[{t}, e^(-3 t) (-2 + 3 e^t - x0 + e^t x0)]}}
```

Here is a plot of the solutions for different initial directions. The solution approaches $-\infty$ or ∞ as $t \to -\infty$ according to whether the value of $x0$ is less than or greater than -2, which is the largest root of the auxiliary equation for the ODE.

In[16]:=
```
Show[GraphicsArray[Partition[
    Table[Plot[Evaluate[x[t] /. particularsolution[[1]] /. x0 → i], {t, -1, 3},
      PlotStyle → {Red}, PlotLabel → StringJoin["x0= ", ToString[i]],
      Ticks → None], {i, -4, 1}], {3}], ImageSize → 400]]
```

Out[16]=



Here is a BVP for an inhomogeneous linear second-order equation.

In[17]:=
```
inhomogeneousequation = y''[x] + y[x] == E^x;
```

In[18]:=
```
generalsolution = DSolve[inhomogeneousequation, y, x]
```

Out[18]=
```
{{y → Function[{x}, C[1] Cos[x] + C[2] Sin[x] + 1/2 e^x (Cos[x]^2 + Sin[x]^2)]}}
```

```
In[19]:= DSolve[{inhomogeneousequation, y[0] == 1, y[1] == 1 / 2}, y, x]

Out[19]= {{y → Function[{x}, 1/2 (Cos[x] + E^x Cos[x]^2 - Cot[1] Sin[x] -
            E Cos[1] Cot[1] Sin[x] + Csc[1] Sin[x] - E Sin[1] Sin[x] + E^x Sin[x]^2)]}}
```

It should be noted that, in contrast to initial value problems, there are no general existence or uniqueness theorems when boundary values are prescribed, and there may be no solution in some cases.

This problem has no solution because the term with C[2] in the general solution vanishes at both $x = 0$ and $x = \pi$. Hence there are two inconsistent conditions for the parameter C[1] and the solution is an empty set.

```
In[20]:= DSolve[{inhomogeneousequation, y[0] == 1, y[Pi] == 6}, y, x]

DSolve::bvnul :
  For some branches of the general solution, the given boundary conditions lead to an empty solution. >>

Out[20]= {}
```

The previous discussion of linear equations generalizes to the case of higher-order linear ODEs and linear systems of ODEs.

Here is the solution to an Initial Value Problem (IVP) for a linear ODE of order four.

```
In[21]:= fourthorderODE = y''''[x] + 2 * y''[x] + y[x] == Cos[x];

In[22]:= sol = DSolve[{fourthorderODE, y[0] == 1, y'[0] == 6, y''[0] == 3, y'''[0] == -1}, y, x]

Out[22]= {{y → Function[{x}, 1/16 (11 Cos[x] - 40 x Cos[x] - 2 x^2 Cos[x] +
            4 Cos[x]^3 + Cos[x] Cos[2 x] + 136 Sin[x] + 34 x Sin[x] + 3 Sin[x] Sin[2 x])]}}
```

This verifies the solution and the initial conditions.

```
In[23]:= {fourthorderODE, y[0], y'[0], y''[0], y'''[0]} /. sol // Simplify

Out[23]= {{True, 1, 6, 3, -1}}
```

Since this is a fourth-order ODE, four independent conditions must be specified to find a particular solution for an IVP. If there is an insufficient number of conditions, the solution returned by DSolve may contain some of the arbitrary parameters, as follows.

```
In[24]:= DSolve[{fourthorderODE, y[0] == 1, y'[0] == 6}, y, x]

Out[24]= {{y → Function[{x}, 1/16 (11 Cos[x] + 96 x Cos[x] - 2 x^2 Cos[x] - 16 x C[3] Cos[x] + 4 Cos[x]^3 +
            Cos[x] Cos[2 x] + 4 x Sin[x] + 16 C[3] Sin[x] + 16 x C[4] Sin[x] + 3 Sin[x] Sin[2 x])]}}
```

Finally, here is the solution of an IVP for a linear system of ODEs.

```
In[25]:= Clear[x, y, z, t]

In[26]:= linearsystem = {x'[t] == x[t] - 4 * y[t] + 1, y'[t] == 4 * x[t] + y[t], z'[t] == z[t]};

In[27]:= initialvalues = {x[0] == 2, y[0] == -1, z[0] == 1};

In[28]:= sol = DSolve[Join[linearsystem, initialvalues], {x, y, z}, t]
```

$$\text{Out[28]=}\ \left\{\left\{\text{x}\to\text{Function}\left[\{\text{t}\},\ \frac{1}{17}\left(35\,e^{\text{t}}\,\text{Cos}[4\,\text{t}]-\text{Cos}[4\,\text{t}]^2+21\,e^{\text{t}}\,\text{Sin}[4\,\text{t}]-\text{Sin}[4\,\text{t}]^2\right)\right],\right.\right.$$
$$\text{y}\to\text{Function}\left[\{\text{t}\},\ \frac{1}{17}\left(-21\,e^{\text{t}}\,\text{Cos}[4\,\text{t}]+4\,\text{Cos}[4\,\text{t}]^2+35\,e^{\text{t}}\,\text{Sin}[4\,\text{t}]+4\,\text{Sin}[4\,\text{t}]^2\right)\right],$$
$$\left.\left.\text{z}\to\text{Function}\left[\{\text{t}\},\ e^{\text{t}}\right]\right\}\right\}$$

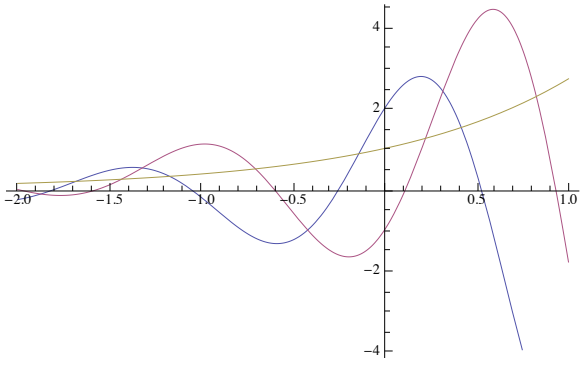
This verifies that the solution satisfies the system and the initial conditions.

```
In[29]:= {linearsystem, initialvalues} /. sol[[1]] // Simplify
Out[29]= {{True, True, True}, {True, True, True}}
```

The solutions $x[t]$, $y[t]$, and $z[t]$ are parametrized by the variable t and can be plotted separately in the plane or as a curve in space.

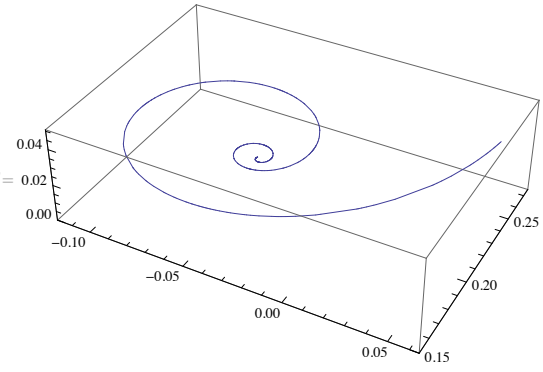
```
In[30]:= Plot[Evaluate[{x[t], y[t], z[t]} /. sol], {t, -2, 1}]
```

Out[30]=



In[31]:=
```
ParametricPlot3D[Evaluate[{x[t], y[t], z[t]} /. sol],
 {t, -7, -3}, PlotRange → All]
```

Out[31]=



Nonlinear IVPs and BVPs

Many real-world applications require the solution of IVPs and BVPs for nonlinear ODEs. For example, consider the *logistic equation,* which occurs in population dynamics.

This is the logistic equation.

In[1]:=
```
LogisticEquation = y'[t] == r (1 - (y[t] / K)) * y[t];
```

The right-hand side of the equation can be expanded to a quadratic polynomial in $y[t]$. Hence, the logistic equation is simply a Riccati equation, and its general solution can be easily found.

In[2]:=
```
DSolve[LogisticEquation, y, t]
```

Out[2]=
$$\left\{\left\{y \to \text{Function}\left[\{t\}, \frac{e^{r\,t+K\,C[1]}\,K}{-1+e^{r\,t+K\,C[1]}}\right]\right\}\right\}$$

This sets the intrinsic growth rate `r` to `1 / 2` and the saturation level `K` to `4` and solves the initial value problem. The warning message from `Solve` is issued while solving for the arbitrary constant `C[1]` from the general solution.

In[3]:=
```
DSolve[{LogisticEquation /. {r → (1 / 2), K → 4}, y[0] == 1}, y, t]
```

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

Out[3]=
$$\left\{\left\{y \to \text{Function}\left[\{t\}, \frac{4\,e^{t/2}}{3+e^{t/2}}\right]\right\}\right\}$$

This solves the initial value problem for the logistic equation with symbolic parameters `r` and `K`.

In[4]:=
```
sol = DSolve[{LogisticEquation, y[0] == a * K}, y, t]
```

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

Out[4]=
$$\left\{\left\{y \to \text{Function}\left[\{t\}, \frac{a\, e^{r\, t}\, K}{1 - a + a\, e^{r\, t}}\right]\right\}\right\}$$

This verifies that the solution satisfies the equation and the initial condition.

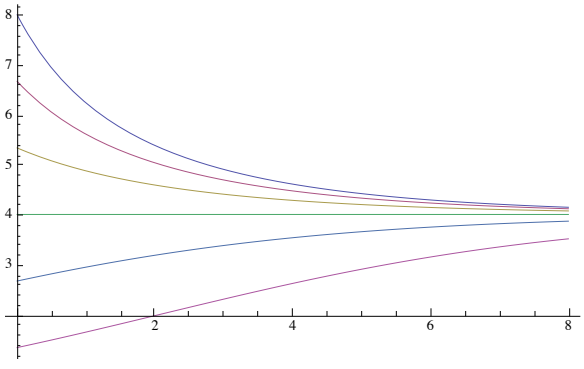
In[5]:=
```
{LogisticEquation, y[0]} /. sol[[1]] // Simplify
```

Out[5]=
```
{True, a K}
```

Here is a plot of the solution for different values of `r` and `K`.

In[6]:=
```
Plot[Evaluate[Table[y[t] /. sol[[1]] /. {K → 4, a → i, r → (1 / 3)},
    {i, 2, 1 / 10, -1 / 3}]], {t, 0, 8}, PlotRange → All]
```

Out[6]=



Here is an example of an IVP for a second-order nonlinear ODE whose general solution can be obtained in explicit form.

In[7]:=
```
eqn = y''[x] - (1 / 2) * (y'[x]^2 / y[x]) + 1 / (2 * y[x]) == 0;
```

In[8]:=
```
sol = DSolve[{eqn, y[0] == 1, y'[0] == 2}, y, x]
```

Out[8]=
$$\left\{\left\{y \to \text{Function}\left[\{x\}, \frac{1}{4}\left(4 + 8\, x + 3\, x^2\right)\right]\right\}\right\}$$

This verifies that the solution satisfies the equation and the initial conditions.

In[9]:=
```
{eqn, y[0], y'[0]} /. sol[[1]] // Simplify
```

Out[9]=
```
{True, 1, 2}
```

Finally, here is a boundary value problem for a nonlinear second-order ODE. The solution is required to satisfy boundary conditions at 0 and infinity. The `Solve::ifun` message is generated while finding the general solution in terms of `JacobiSN`, the inverse of `EllipticF`. The `DSolve::bvlim` messages are given because the limit required for satisfying the condition `y'[Infinity] == 0` cannot be calculated for either branch of the general solution. However, the solution for the boundary value problem is found using an alternative method to determine the values of the constants `C[1]` and `C[2]` in the general solution.

In[10]:=
```
generalsolution = DSolve[{y''[x] / 2 == y[x]^3 - y[x]}, y[x], x]
```

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

Out[10]=
$$\left\{\left\{y[x] \to i \sqrt{-\frac{1}{1-\sqrt{1-C[1]}}}\ \text{JacobiSN}\left[\sqrt{x^2 + x^2\sqrt{1-C[1]} + 2\,x\,C[2] + 2\,x\sqrt{1-C[1]}\,C[2] + C[2]^2 + \sqrt{1-C[1]}\,C[2]^2},\ \frac{1-\sqrt{1-C[1]}}{1+\sqrt{1-C[1]}}\right] - i\sqrt{-\frac{1}{1-\sqrt{1-C[1]}}}\ \sqrt{1-C[1]}\ \text{JacobiSN}\left[\sqrt{x^2 + x^2\sqrt{1-C[1]} + 2\,x\,C[2] + 2\,x\sqrt{1-C[1]}\,C[2] + C[2]^2 + \sqrt{1-C[1]}\,C[2]^2},\ \frac{1-\sqrt{1-C[1]}}{1+\sqrt{1-C[1]}}\right]\right\},\ \left\{y[x] \to -i\sqrt{-\frac{1}{1-\sqrt{1-C[1]}}}\ \text{JacobiSN}\left[\sqrt{x^2 + x^2\sqrt{1-C[1]} + 2\,x\,C[2] + 2\,x\sqrt{1-C[1]}\,C[2] + C[2]^2 + \sqrt{1-C[1]}\,C[2]^2},\ \frac{1-\sqrt{1-C[1]}}{1+\sqrt{1-C[1]}}\right] + i\sqrt{-\frac{1}{1-\sqrt{1-C[1]}}}\ \sqrt{1-C[1]}\ \text{JacobiSN}\left[\sqrt{x^2 + x^2\sqrt{1-C[1]} + 2\,x\,C[2] + 2\,x\sqrt{1-C[1]}\,C[2] + C[2]^2 + \sqrt{1-C[1]}\,C[2]^2},\ \frac{1-\sqrt{1-C[1]}}{1+\sqrt{1-C[1]}}\right]\right\}\right\}$$

In[11]:=
```
sol =
 DSolve[{y''[x] / 2 == y[x]^3 - y[x], y[0] == 0, y'[Infinity] == 0}, y[x], x]
```

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

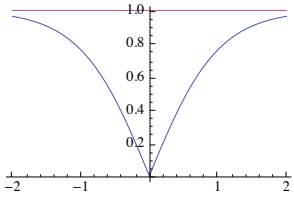
DSolve::bvlim :
For some branches of the general solution, unable to compute the limit at the given points. Some of the solutions may be lost. >>

DSolve::bvlim :
For some branches of the general solution, unable to compute the limit at the given points. Some of the solutions may be lost. >>

Out[11]=
$$\left\{\left\{y[x] \to -\text{Tanh}\left[\sqrt{x^2}\right]\right\},\ \left\{y[x] \to \text{Tanh}\left[\sqrt{x^2}\right]\right\}\right\}$$

```
In[12]:= Plot[{y[x] /. sol[[2]], 1}, {x, -2, 2}, PlotRange → All]
```

Out[12]=



It may not always be possible to obtain a symbolic solution to an IVP or BVP for a nonlinear equation. Numerical methods may be necessary in such cases.

IVPs with Piecewise Coefficients

The differential equations that arise in modern applications often have discontinuous coefficients. `DSolve` can handle a wide variety of such *ODEs with piecewise coefficients.* Some of the functions used in these equations are `UnitStep`, `Max`, `Min`, `Sign`, and `Abs`. These functions and combinations of them can be converted into `Piecewise` objects.

This converts the given expression into a `Piecewise` expression.

```
In[1]:= PiecewiseExpand[UnitStep[x] + Max[x, x^2]]
```

$$\text{Out[1]=} \begin{cases} x^2 & x < 0 \\ 1 + x & 0 \le x \le 1 \\ 1 + x^2 & \text{True} \end{cases}$$

Here is the general solution to a first-order ODE that contains `UnitStep`.

```
In[2]:= DSolve[y'[x] == UnitStep[x], y, x]
Out[2]= {{y → Function[{x}, C[1] + x UnitStep[x]]}}
```

Here is the solution to the same ODE with an initial condition.

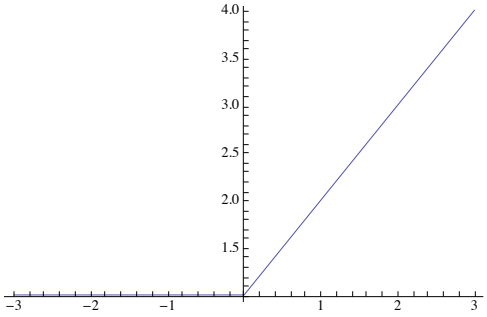
```
In[3]:= eqn = {y'[x] == UnitStep[x], y[0] == 1};

In[4]:= sol = DSolve[eqn, y, x]
Out[4]= {{y → Function[{x}, 1 + x UnitStep[x]]}}
```

The solution can be plotted in the usual way. Note that the solution is continuous but not differentiable at $x = 0$.

```
In[5]:= Plot[y[x] /. sol, {x, -3, 3}]
```

Out[5]=



This verifies the solution.

```
In[6]:= Simplify[eqn /. sol[[1]], x > 0 || x < 0]
Out[6]= {True, True}
```

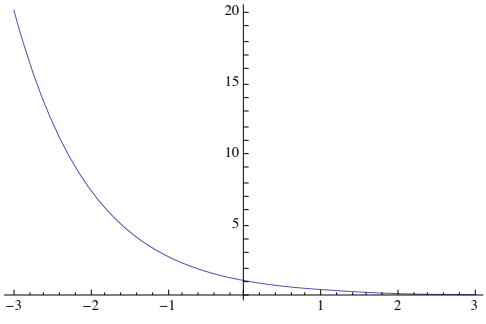
Here is a piecewise ODE that has `Max` in its coefficients.

```
In[7]:= sol = DSolve[ {y'[x] + Max[x, 1] y[x] == 0, y[0] == 1}, y[x], x ]
```

$$\text{Out[7]= } \left\{\left\{y[x] \to \begin{cases} e^{-x} & x \le 1 \\ e^{-\frac{1}{2}-\frac{x^2}{2}} & \text{True} \end{cases}\right\}\right\}$$

```
In[8]:= Plot[y[x] /. sol, {x, -3, 3}]
```

Out[8]=



A piecewise ODE can be thought of as a collection of ODEs over disjoint intervals such that the expressions for the coefficients and the boundary conditions change from one interval to another. Thus, different intervals have different solutions, and the final solution for the ODE is obtained by patching together the solutions over the different intervals.

For this piecewise ODE, the expression for `FinalSol` is obtained by patching together `SolFromMinusInfinityToTwo` and `SolFromTwoToInfinity`. The boundary condition for the interval `(-Infinity, 2]` is simply $y[0] = 1$, while the initial condition for the interval `[2, Infinity)` is $y[2] = e^2$ (given by the final value for the solution over the first interval).

```
In[9]:= FinalSol = DSolve[ {y'[t] == If[ t ≤ 2, y[t], -y[t] / 2] , y[0] == 1}, y, t]
```

$$\text{Out[9]= } \left\{\left\{y \to \text{Function}\left[\{t\}, \begin{cases} e^{t} & t \le 2 \\ e^{3-\frac{t}{2}} & \text{True} \end{cases}\right]\right\}\right\}$$

```
In[10]:= SolFromMinusInfinityToTwo = DSolve[{y'[t] == y[t], y[0] == 1}, y, t]
```

$$\text{Out[10]= } \left\{\left\{y \to \text{Function}\left[\{t\}, e^{t}\right]\right\}\right\}$$

```
In[11]:= SolFromTwoToInfinity = DSolve[{y'[t] == -y[t] / 2, y[2] == E^2}, y, t]
```

$$\text{Out[11]= } \left\{\left\{y \to \text{Function}\left[\{t\}, e^{3-\frac{t}{2}}\right]\right\}\right\}$$

If there are a large number of discontinuities in a problem, it is convenient to use `Piecewise` directly in the formulation of the problem.

This second-order ODE contains a `Piecewise` term.

```
In[12]:= eqn = {y''[t] + y[t] == Piecewise[{{-1, t < 0}, {1, t < 1}, {Sin[t], t < 2}}],
    y[0] == 1, y'[0] == 1};
```

```
In[13]:= sol = DSolve[ eqn, y, t ]
```

$$\text{Out[13]= } \left\{\left\{y \to \text{Function}\left[\{t\}, \begin{cases} -1 + 2\cos[t] + \sin[t] & t \le 0 \\ 1 + \sin[t] & 0 < t \le 1 \\ \frac{1}{4\left(\cos[1]^2+\sin[1]^2\right)} \left(4\cos[1]\cos[t] + 2\cos[1]^2\cos[t] - 2t\cos[1]^2\cos[t] - 2\cos[1]\cos[t]\sin[1] + 2\cos[t]\sin[1]^2 - 2t\cos[t]\sin[1]^2 + 6\cos[1]^2\sin[t] - 2\cos[1]^2\cos[t]^2\sin[t] + 4\sin[1]\sin[t] + 4\sin[1]^2\sin[t] - 2\cos[t]^2\sin[1]^2\sin[t] + \cos[1]^2\cos[t]\sin[2t] + \cos[t]\sin[1]^2\sin[2t]\right) & 1 < t \le 2 \\ \frac{1}{4\left(\cos[2]^2+\sin[2]^2\right)} \left(4\cos[1]\cos[2]\cos[t] - 2\cos[2]^2\cos[t] - \cos[t]\sin[2] + 2\cos[2]\cos[t]\sin[2] + 4\cos[t]\sin[1]\sin[2] - 2\cos[t]\sin[2]^2 + \cos[2]\sin[t] + 3\cos[2]^2\sin[t] - 4\cos[2]\sin[1]\sin[t] + 4\cos[1]\sin[2]\sin[t] + 5\sin[2]^2\sin[t]\right) & \text{True} \end{cases}\right]\right\}\right\}$$

This ODE contains the `Clip` function. The solutions are given in terms of Airy functions.

```
In[14]:= eqn = {y''[x] - Clip[x] * y[x] == 0, y[0] == 0, y'[0] == -1};
```

In[15]:=
```
DSolve[eqn, y, x]
```

Out[15]=
$$\left\{\left\{y \to \text{Function}\left[\{x\},\right.\right.\right.$$

$$\left\{\begin{array}{ll}
\frac{1}{6\left(\text{Cos}[1]^2+\text{Sin}[1]^2\right)} & x \leq \\
\left(3\times 3^{1/3}\,\text{AiryAi}[-1]\,\text{Cos}[1]\,\text{Cos}[x]\,\text{Gamma}\left[\frac{1}{3}\right] - 3^{5/6}\,\text{AiryBi}[-1]\,\text{Cos}[1]\,\text{Cos}[x]\,\text{Gamma}\left[\frac{1}{3}\right] + \right. & \\
\quad 3\times 3^{1/3}\,\text{AiryAiPrime}[-1]\,\text{Cos}[x]\,\text{Gamma}\left[\frac{1}{3}\right]\,\text{Sin}[1] - 3^{5/6}\,\text{AiryBiPrime}[-1] & \\
\quad \text{Cos}[x]\,\text{Gamma}\left[\frac{1}{3}\right]\,\text{Sin}[1] + 3\times 3^{1/3}\,\text{AiryAiPrime}[-1]\,\text{Cos}[1]\,\text{Gamma}\left[\frac{1}{3}\right]\,\text{Sin}[x] - & \\
\quad 3^{5/6}\,\text{AiryBiPrime}[-1]\,\text{Cos}[1]\,\text{Gamma}\left[\frac{1}{3}\right]\,\text{Sin}[x] - 3\times 3^{1/3}\,\text{AiryAi}[-1]\,\text{Gamma}\left[\frac{1}{3}\right]\,\text{Sin}[1]\,\text{Sin}[x] + & \\
\quad \left. 3^{5/6}\,\text{AiryBi}[-1]\,\text{Gamma}\left[\frac{1}{3}\right]\,\text{Sin}[1]\,\text{Sin}[x]\right) & \\
\frac{1}{6}\left(3\times 3^{1/3}\,\text{AiryAi}[x]\,\text{Gamma}\left[\frac{1}{3}\right] - 3^{5/6}\,\text{AiryBi}[x]\,\text{Gamma}\left[\frac{1}{3}\right]\right) & -1 < \\
-\frac{1}{4\times 3^{2/3}}\,e^{-1-x}\left(-3\,e^2\,\text{AiryAi}[1] - 3\,e^{2x}\,\text{AiryAi}[1] + \right. & \text{True} \\
\quad 3\,e^2\,\text{AiryAiPrime}[1] - 3\,e^{2x}\,\text{AiryAiPrime}[1] + \sqrt{3}\,e^2\,\text{AiryBi}[1] + & \\
\quad \left.\sqrt{3}\,e^{2x}\,\text{AiryBi}[1] - \sqrt{3}\,e^2\,\text{AiryBiPrime}[1] + \sqrt{3}\,e^{2x}\,\text{AiryBiPrime}[1]\right)\text{Gamma}\left[\frac{1}{3}\right] &
\end{array}\right.$$

$$\left.\left.\left.\right]\right\}\right\}$$

Working with DSolve—A User's Guide

Introduction to Working with DSolve

The aim of these tutorials is to provide a self-contained working guide for solving different types of problems with `DSolve`.

The first step in using `DSolve` is to set up the problem correctly. The next step is to use `DSolve` to get an expression for the solution. Once the solution has been found, it can be verified using symbolic or numerical techniques, or it can be plotted using a *Mathematica* function such as `Plot`, `Plot3D`, or `ContourPlot`. Plots often reveal information about the solution that might not be evident from its closed-form expression.

If no boundary conditions are specified for a problem, the output from `DSolve` is some form of a general solution containing arbitrary parameters. The `GeneratedParameters` option can be used to label these arbitrary parameters.

In many applications, differential equations contain symbolic parameters, such as the rate of growth in the logistic equation. A differential equation can also contain inexact quantities, such as machine numbers arising from previous calculations. Both symbolic parameters and inexact quantities are allowed by `DSolve`, but it is good to be aware of their presence and interpret the solution correctly.

When `DSolve` makes any assumptions or encounters difficulty during a calculation, it issues a warning message outlining the problem. These messages can usually be ignored, but sometimes they point to serious limitations in the answer given for the problem.

It is helpful to analyze the statement of the problem for possible ambiguities—in other words, to make sure that the problem is well posed—so that meaningful answers can be obtained from `DSolve`.

Setting Up the Problem

The first argument given to DSolve is the differential equation, the second argument is the unknown function, and the last argument identifies the independent variable.

Here is the input for solving for a first-order linear ODE using DSolve. The variable sol identifies the solution for use in further work.

```
In[1]:= sol = DSolve[y'[x] + 5 y[x] == 1, y[x], x]
```

$$\text{Out[1]}= \left\{\left\{\text{y[x]} \to \frac{1}{5} + e^{-5\,x}\,\text{C[1]}\right\}\right\}$$

The output of DSolve is a list of solutions for the differential equation. The extra list is required since some equations have multiple solutions. Here, since the equation is of order 1 and is linear, there is only one solution: $\text{y[x]} \to \frac{1}{5} + e^{-5\,x}\,\text{C[1]}$. The solution has an undetermined constant C[1] because no initial condition was specified. The solution can be extracted from the list of solutions using a part specification.

This extracts the solution.

```
In[2]:= m = sol[[1]]
```

$$\text{Out[2]}= \left\{\text{y[x]} \to \frac{1}{5} + e^{-5\,x}\,\text{C[1]}\right\}$$

This form of the solution is useful for finding $y[x]$ itself, but not for finding derivatives of $y[x]$ or the value of $y[x]$ at a point.

This shows the value of $y[x]$ given by the solution.

```
In[3]:= y[x] /. m
```

$$\text{Out[3]}= \frac{1}{5} + e^{-5\,x}\,\text{C[1]}$$

The solution does not apply to $y'[x]$ or $y[1]$ because the solution is a rule for $y[x]$ only.

```
In[4]:= y'[x] /. m
Out[4]= y'[x]

In[5]:= y[1] /. m
Out[5]= y[1]
```

If the solution will be used in further work, it is best to specify the unknown function using y rather than $y[x]$. This gives the solution using pure functions of the type `Function[x, expr]`.

Here, the unknown function is specified as y. The solution is a pure function.

```
In[6]:= sol = DSolve[y'[x] + 5 y[x] == 1, y, x]
```

$$\text{Out[6]= } \left\{\left\{y \to \text{Function}\left[\{x\}, \frac{1}{5} + e^{-5x} C[1]\right]\right\}\right\}$$

When the solution is in the form of pure functions, expressions can be found for derivatives of y and for the values of y at specific points.

This gives expressions for $y[x]$, $y'[x]$, and $y[1]$.

```
In[7]:= m = sol[[1]]
```

$$\text{Out[7]= } \left\{y \to \text{Function}\left[\{x\}, \frac{1}{5} + e^{-5x} C[1]\right]\right\}$$

```
In[8]:= y[x] /. m
```

$$\text{Out[8]= } \frac{1}{5} + e^{-5x} C[1]$$

```
In[9]:= y'[x] /. m
```

$$\text{Out[9]= } -5 e^{-5x} C[1]$$

```
In[10]:= y[1] /. m
```

$$\text{Out[10]= } \frac{1}{5} + \frac{C[1]}{e^5}$$

When a problem has multiple solutions, you can pick out individual solutions from the solution list or you can work directly with the list.

This solves a nonlinear first-order equation. There are two solutions.

```
In[11]:= sol = DSolve[y'[x]^2 == x + 11, y, x]
```

$$\text{Out[11]= } \left\{\left\{y \to \text{Function}\left[\{x\}, -\frac{2}{3}(11+x)^{3/2} + C[1]\right]\right\}, \left\{y \to \text{Function}\left[\{x\}, \frac{2}{3}(11+x)^{3/2} + C[1]\right]\right\}\right\}$$

The solutions can be extracted using part specifications.

```
In[12]:= y[x] /. sol[[1]]
```

$$\text{Out[12]= } -\frac{2}{3}(11+x)^{3/2} + C[1]$$

```
In[13]:= y[x] /. sol[[2]]
```

$$\text{Out[13]}= \frac{2}{3}\,(11+x)^{3/2}+C[1]$$

This returns a list of both expressions.

```
In[14]:= y[x] /. sol
```

$$\text{Out[14]}= \left\{-\frac{2}{3}\,(11+x)^{3/2}+C[1],\ \frac{2}{3}\,(11+x)^{3/2}+C[1]\right\}$$

To solve a system of equations, the first argument to `DSolve` must be a list of the equations and the second argument must be a list of the unknown functions.

Here is an example of a system of first-order linear equations with three unknowns. Because this system is linear, there is only one solution.

```
In[15]:= sol = DSolve[
           {x'[t] == y[t] + z[t], y'[t] + z[t] - x[t] == 0, z'[t] + y[t] == x[t]}, {x, y, z}, t]
```

$$\text{Out[15]}= \left\{\left\{x\to \text{Function}\left[\{t\},\ \frac{1}{3}e^{-2t}\left(1+2e^{3t}\right)C[1]+\frac{1}{3}e^{-2t}\left(-1+e^{3t}\right)C[2]+\frac{1}{3}e^{-2t}\left(-1+e^{3t}\right)C[3]\right],\right.\right.$$
$$y\to \text{Function}\left[\{t\},\ \frac{1}{3}e^{-2t}\left(-1+e^{3t}\right)C[1]+\frac{1}{3}e^{-2t}\left(1+2e^{3t}\right)C[2]-\frac{1}{3}e^{-2t}\left(-1+e^{3t}\right)C[3]\right],$$
$$\left.\left.z\to \text{Function}\left[\{t\},\ \frac{1}{3}e^{-2t}\left(-1+e^{3t}\right)C[1]-\frac{1}{3}e^{-2t}\left(-1+e^{3t}\right)C[2]+\frac{1}{3}e^{-2t}\left(1+2e^{3t}\right)C[3]\right]\right\}\right\}$$

Each solution to the system is a list of replacement rules for the unknown functions. The expressions for the unknown functions can be extracted as in previous examples.

This gives a list of the expressions for the unknown functions. `Simplify` is used to return the expressions in a compact form.

```
In[16]:= {x[t], y[t], z[t]} /. sol[[1]] // Simplify
```

$$\text{Out[16]}= \left\{\frac{1}{3}e^{-2t}\left(\left(1+2e^{3t}\right)C[1]+\left(-1+e^{3t}\right)(C[2]+C[3])\right),\right.$$
$$\frac{1}{3}e^{-2t}\left(\left(-1+e^{3t}\right)C[1]+C[2]+2e^{3t}C[2]+C[3]-e^{3t}C[3]\right),$$
$$\left.\frac{1}{3}e^{-2t}\left(\left(-1+e^{3t}\right)C[1]+C[2]-e^{3t}C[2]+C[3]+2e^{3t}C[3]\right)\right\}$$

If initial conditions are prescribed for the problem, some or all of the undetermined constants can be eliminated.

Here the value of the unknown function and its derivative are both prescribed at the initial point.

```
In[17]:= DSolve[{y''[x] + y[x] == 5, y[0] == 1, y'[0] == 7}, y[x], x]
```

$$\text{Out[17]}= \{\{y[x]\to 5-4\,\text{Cos}[x]+7\,\text{Sin}[x]\}\}$$

If only the initial value is specified, then the solution still contains an arbitrary constant.

```
In[18]:= DSolve[{y''[x] + y[x] == 5, y[0] == 1}, y[x], x]
Out[18]= {{y[x] → 5 - 4 Cos[x] + C[2] Sin[x]}}
```

For a partial differential equation, the third argument to `DSolve` is a list of the independent variables for the equation.

This solves a PDE with independent variables x and y. `C[1]` represents an arbitrary function of y + `Cos[x]`.

```
In[19]:= DSolve[D[u[x, y], x] + Sin[x] * D[u[x, y], y] == 8, u, {x, y}]
Out[19]= {{u → Function[{x, y}, 8 x + C[1][y + Cos[x]]]}}
```

A differential-algebraic equation is specified in the same way as a system of ordinary differential equations.

Here is an example of a DAE with an initial condition.

```
In[20]:= DSolve[{x'[t] + y[t] == Sin[t], x[t] + y[t] == 1, x[0] == 4}, {x, y}, t]
```

$$\text{Out[20]= } \left\{\left\{x \to \text{Function}\left[\{t\}, \frac{1}{2}\left(2 + 7\, e^{t} - \text{Cos}[t] - \text{Sin}[t]\right)\right], y \to \text{Function}\left[\{t\}, \frac{1}{2}\left(-7\, e^{t} + \text{Cos}[t] + \text{Sin}[t]\right)\right]\right\}\right\}$$

Note that it is not always possible to give the solutions for a problem in explicit form. In this case, the solution is given using an unevaluated `Solve` object or using `InverseFunction`.

The solution to this equation is not available explicitly. The output represents an implicit solution.

```
In[21]:= sol = DSolve[y'[x] + y[x]^3 + y[x]^2 == 1, y[x], x]

Solve::tdep :
  The equations appear to involve the variables to be solved for in an essentially non-algebraic way. >>

Solve::tdep :
  The equations appear to involve the variables to be solved for in an essentially non-algebraic way. >>
```

$$\text{Out[21]= } \text{Solve}\left[\text{RootSum}\left[-1 + \#1^2 + \#1^3 \&, \frac{\text{Log}[-\#1 + y[x]]}{2\,\#1 + 3\,\#1^2} \&\right] == -x + C[1], y[x]\right]$$

The solution can be extracted as usual with a part specification.

```
In[22]:= sol[[1]]
```

$$\text{Out[22]= } \text{RootSum}\left[-1 + \#1^2 + \#1^3 \&, \frac{\text{Log}[-\#1 + y[x]]}{2\,\#1 + 3\,\#1^2} \&\right] == -x + C[1]$$

The solutions to this equation are given as `InverseFunction` objects.

```
In[23]:= sol = DSolve[Derivative[2][y][x] + y[x] * Derivative[1][y][x]^4 == 0, y, x]
```

```
Solve::tdep :
  The equations appear to involve the variables to be solved for in an essentially non-algebraic way. >>

Solve::tdep :
  The equations appear to involve the variables to be solved for in an essentially non-algebraic way. >>
```

$$\text{Out[23]= } \left\{\left\{y \to \text{Function}\left[\{x\}, \text{InverseFunction}\left[C[1]\,\text{Log}\left[\#1 + \sqrt{-2\,C[1] + \#1^2}\,\right] - \frac{1}{2}\#1\sqrt{-2\,C[1] + \#1^2}\ \&\right][x + C[2]]\right]\right\}, \left\{y \to \text{Function}\left[\{x\}, \text{InverseFunction}\left[-C[1]\,\text{Log}\left[\#1 + \sqrt{-2\,C[1] + \#1^2}\,\right] + \frac{1}{2}\#1\sqrt{-2\,C[1] + \#1^2}\ \&\right][x + C[2]]\right]\right\}\right\}$$

Each solution can be rewritten as an implicit equation by eliminating the `InverseFunction` object as follows.

```
In[24]:= soly = y[x] /. sol[[1]]
```

$$\text{Out[24]= } \text{InverseFunction}\left[C[1]\,\text{Log}\left[\#1 + \sqrt{-2\,C[1] + \#1^2}\,\right] - \frac{1}{2}\#1\sqrt{-2\,C[1] + \#1^2}\ \&\right][x + C[2]]$$

```
In[25]:= implicitequation = ( Head[soly][[1]][y[x]] == soly[[1]] )
```

$$\text{Out[25]= } C[1]\,\text{Log}\left[y[x] + \sqrt{-2\,C[1] + y[x]^2}\,\right] - \frac{1}{2}y[x]\sqrt{-2\,C[1] + y[x]^2} == x + C[2]$$

Verification of the Solution

The solution given by `DSolve` can be verified using various methods. The easiest method involves substituting the solution back into the equation. If the result is `True`, the solution is valid.

In this simple example, the solution is verified by substitution. Note that the first argument to `DSolve` is assigned to *eqn* for convenience in later work.

```
In[1]:= eqn = y'[x] == x;
```

```
In[2]:= sol = DSolve[eqn, y, x]
```

$$\text{Out[2]= } \left\{\left\{y \to \text{Function}\left[\{x\}, \frac{x^2}{2} + C[1]\right]\right\}\right\}$$

```
In[3]:= eqn /. sol
Out[3]= {True}
```

In this example, the equation and an initial condition are verified by substitution.

```
In[4]:= eqn = {y'[x] == x, y[0] == 1};

In[5]:= sol = DSolve[eqn, y, x]

Out[5]= {{y → Function[{x}, 1/2 (2 + x^2)]}}

In[6]:= eqn /. sol
Out[6]= {{True, True}}
```

Sometimes the result of the substitution is more complicated than `True` or `False`. Such examples can be verified by using `Simplify` to simplify the result of the substitution. If the simplified result is `True`, the solution is valid.

Here is the general solution for a second-order inhomogeneous equation.

```
In[7]:= eqn = y''[x] + 5 * y'[x] + 6 y[x] == 1;

In[8]:= sol = DSolve[eqn, y, x]

Out[8]= {{y → Function[{x}, 1/6 + ℯ^(-3 x) C[1] + ℯ^(-2 x) C[2]]}}
```

This substitutes the solution back into the equation.

```
In[9]:= eqn /. sol

Out[9]= {9 ℯ^(-3 x) C[1] + 4 ℯ^(-2 x) C[2] + 5 (-3 ℯ^(-3 x) C[1] - 2 ℯ^(-2 x) C[2]) + 6 (1/6 + ℯ^(-3 x) C[1] + ℯ^(-2 x) C[2]) == 1}
```

The solution can be verified using `Simplify`.

```
In[10]:= Simplify[eqn /. sol]
Out[10]= {True}
```

Here is a linear PDE whose solution can be verified using `Simplify`.

```
In[11]:= PDE = D[u[x, y], x] + Sin[x] * D[u[x, y], y] == x^2;

In[12]:= sol = DSolve[PDE, u, {x, y}]

Out[12]= {{u → Function[{x, y}, 1/3 (x^3 + 3 C[1][y + Cos[x]])]}}

In[13]:= Simplify[PDE /. sol]
Out[13]= {True}
```

If the equation involves special functions, it may be necessary to use `FullSimplify` to verify the solution.

Here is an example of this type involving Bessel's functions.

```
In[14]:= eqn = x * y''[x] + y'[x] - y[x] == 1;

In[15]:= sol = DSolve[eqn, y, x]

Out[15]= {{y → Function[{x}, -1 + BesselI[0, 2 √x] C[1] + 2 BesselK[0, 2 √x] C[2]]}}

In[16]:= FullSimplify[eqn /. sol]
Out[16]= {True}
```

If the solution is large or if `Simplify` and `FullSimplify` do not succeed in verifying the solution, a numerical check can be made by using `RandomReal` or `RandomComplex` to generate values for all the variables and parameters in the problem. It is advisable in such cases to repeat the check with several sets of random values.

Here is an example where numerical verification is useful.

```
In[17]:= Clear[a, y, x, r]

In[18]:= eqn = y''[x] - (a * x^6 + x^2) * y[x];

In[19]:= sol = DSolve[eqn == 0, y, x]

Out[19]= {{y → Function[{x}, 2^(3/8) e^(√a x^4/4) (x^4)^(3/8) C[1] HypergeometricU[(-1+3 √a)/(8 √a), 3/4, -1/2 √a x^4] / x^(3/2) +
    2^(3/8) e^(√a x^4/4) (x^4)^(3/8) C[2] LaguerreL[-(-1+3 √a)/(8 √a), -1/4, -1/2 √a x^4] / x^(3/2)]}}

In[20]:= Union[Flatten[Table[Chop[eqn /. sol /. {x → RandomReal[], a → RandomReal[],
        C[1] → RandomReal[], C[2] → RandomReal[]}], {i, 1, 10}]]]
Out[20]= {0}
```

Although numerical checks cannot verify a solution with certainty, more rigorous checks can be made by using higher precision or by allowing the variables to take complex values.

This verifies the previous solution with higher precision.

```
In[21]:= Chop[eqn /. sol /. {x → RandomReal[{1, 2}, WorkingPrecision -> 20],
    a → RandomReal[{1, 2}, WorkingPrecision -> 20],
    C[1] → RandomReal[{1, 2}, WorkingPrecision -> 20],
    C[2] → RandomReal[{1, 2}, WorkingPrecision -> 20]}]
Out[21]= {0}
```

This uses complex random values to verify the previous solution.

```
In[22]:= r := RandomComplex[]

In[23]:= Chop[N[eqn /. sol /. {x → r, a → r , C[1] → r , C[2] → r}]]
Out[23]= {0}
```

The previous methods are of use only when the solution is available in explicit form. The final example shows how to verify the solution of a first-order ODE when it is given in implicit form.

This solves a first-order ODE.

```
In[24]:= eqn = y'[x] + 2 * x y[x]^2 + y[x]^3;

In[25]:= sol = DSolve[eqn == 0, y, x]
```

InverseFunction::ifun: Inverse functions are being used. Values may be lost for multivalued inverses. ≫

InverseFunction::ifun: Inverse functions are being used. Values may be lost for multivalued inverses. ≫

InverseFunction::ifun: Inverse functions are being used. Values may be lost for multivalued inverses. ≫

General::stop: Further output of InverseFunction::ifun will be suppressed during this calculation. ≫

Solve::tdep:
The equations appear to involve the variables to be solved for in an essentially non−algebraic way. ≫

$$\text{Out[25]= } \text{Solve}\left[\frac{-x\,\text{AiryAi}\left[x^2-\frac{1}{y[x]}\right]+\text{AiryAiPrime}\left[x^2-\frac{1}{y[x]}\right]}{-x\,\text{AiryBi}\left[x^2-\frac{1}{y[x]}\right]+\text{AiryBiPrime}\left[x^2-\frac{1}{y[x]}\right]}+C[1]==0,\ y[x]\right]$$

```
In[26]:= sol[[1]]
```

$$\text{Out[26]= } \frac{-x\,\text{AiryAi}\left[x^2-\frac{1}{y[x]}\right]+\text{AiryAiPrime}\left[x^2-\frac{1}{y[x]}\right]}{-x\,\text{AiryBi}\left[x^2-\frac{1}{y[x]}\right]+\text{AiryBiPrime}\left[x^2-\frac{1}{y[x]}\right]}+C[1]==0$$

This verifies the solution by simplifying its derivative.

```
In[27]:= Simplify[Solve[D[sol[[1]], x], y'[x]]]
Out[27]= {{y'[x] → -y[x]^2 (2 x + y[x])}}
```

Plotting the Solution

A plot of the solution given by `DSolve` can give useful information about the nature of the solution, for instance, whether it is oscillatory in nature. It can also serve as a means of solution verification if the shape of the graph is known from theory or from plotting the vector field associated with the differential equation. A few examples that use different *Mathematica* graphics functions follow.

Here is the general solution to a linear first-order equation.

```
In[1]:= sol = DSolve[y'[x] + x * y[x] == Cos[x^2], y, x]
```

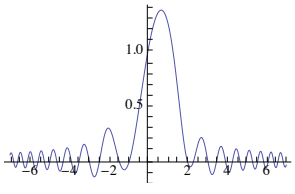
Out[1]=

$$\left\{\left\{y \to \text{Function}\left[\{x\}, e^{-\frac{x^2}{2}} C[1] + \frac{1}{2} e^{-\frac{x^2}{2}} \sqrt{\frac{\pi}{10}} \left(\sqrt{1+2i}\ \text{Erfi}\left[\sqrt{\frac{1}{2}-i}\ x\right] + \sqrt{1-2i}\ \text{Erfi}\left[\sqrt{\frac{1}{2}+i}\ x\right]\right)\right]\right\}\right\}$$

The solution can be plotted for specific values of the constant `C[1]` using `Plot`. The use of `Evaluate` reduces the time taken by `Plot` and can also help in cases where the solution has discontinuities.

```
In[2]:= Plot[Evaluate[y[x] /. sol /. {C[1] → 1}], {x, -7, 7}, PlotRange → All]
```

Out[2]=



Here is the plot for a linear second-order ODE with initial values prescribed at 0.

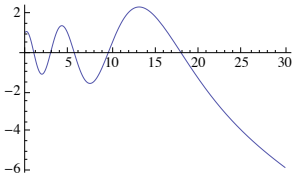
```
In[3]:= sol = DSolve[{y''[x] / y[x] == -4 * Exp[-x / 4], y[0] == 1, y'[0] == 1 / 2}, y, x]
```

Out[3]=

$$\left\{\left\{y \to \text{Function}\left[\{x\}, \left(\text{BesselJ}\left[0, 16\sqrt{e^{-x/4}}\right] \text{BesselY}[0,16] - \text{BesselJ}[0,16]\,\text{BesselY}\left[0, 16\sqrt{e^{-x/4}}\right] + 4\,\text{BesselJ}[1,16]\,\text{BesselY}\left[0, 16\sqrt{e^{-x/4}}\right] - 4\,\text{BesselJ}\left[0, 16\sqrt{e^{-x/4}}\right]\text{BesselY}[1,16]\right) \Big/ \left(4\left(\text{BesselJ}[1,16]\,\text{BesselY}[0,16] - \text{BesselJ}[0,16]\,\text{BesselY}[1,16]\right)\right)\right]\right\}\right\}$$

```
In[4]:= Plot[Evaluate[y[x] /. sol], {x, 0, 30}]
```

Out[4]=



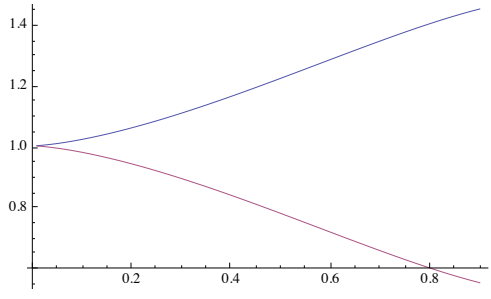
This nonlinear equation has two solutions that can be plotted on the same graph.

In[5]:=
```
sol = DSolve[{y'[x]^2 == x - x^3, y[0] == 1}, y, x]
```

Out[5]=
$$\left\{\left\{y \to \text{Function}\left[\{x\}, \frac{1}{5\sqrt{x}\sqrt{1-x^2}}\left(5\sqrt{x}\sqrt{1-x^2} + 2x^{3/2}\sqrt{1-x^2}\sqrt{-x\left(-1+x^2\right)} + 4\sqrt{-x\left(-1+x^2\right)}\ \text{EllipticE}\left[\text{ArcSin}\left[\sqrt{x}\right], -1\right] - 4\sqrt{-x\left(-1+x^2\right)}\ \text{EllipticF}\left[\text{ArcSin}\left[\sqrt{x}\right], -1\right]\right)\right]\right\},$$
$$\left\{y \to \text{Function}\left[\{x\}, \frac{1}{5\sqrt{x}\sqrt{1-x^2}}\left(5\sqrt{x}\sqrt{1-x^2} - 2x^{3/2}\sqrt{1-x^2}\sqrt{-x\left(-1+x^2\right)} - 4\sqrt{-x\left(-1+x^2\right)}\ \text{EllipticE}\left[\text{ArcSin}\left[\sqrt{x}\right], -1\right] + 4\sqrt{-x\left(-1+x^2\right)}\ \text{EllipticF}\left[\text{ArcSin}\left[\sqrt{x}\right], -1\right]\right)\right]\right\}\right\}$$

In[6]:=
```
Plot[Evaluate[y[x] /. sol], {x, 1 / 100, 9 / 10}]
```

Out[6]=



The solution to this Abel ODE is given in implicit form.

In[7]:=
```
sol = DSolve[y'[x] == -5 y[x]^2 - y[x]^3 / x, y[x], x]
```

Solve::tdep :
The equations appear to involve the variables to be solved for in an essentially non-algebraic way. ≫

Out[7]=
$$\text{Solve}\left[-5x == \frac{2\, e^{\frac{1}{2}\left(-5x+\frac{1}{y[x]}\right)^2}}{2C[1] + \sqrt{2\pi}\ \text{Erfi}\left[\frac{-5x+\frac{1}{y[x]}}{\sqrt{2}}\right]}, y[x]\right]$$

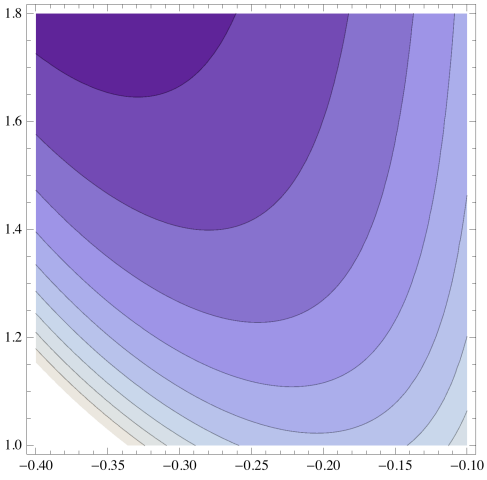
A contour plot can be used to study the nature of the solution. Each contour line corresponds to a solution to the ODE for a fixed value of `C[1]`.

In[8]:=
```
expr = C[1] /. Solve[sol[[1]], C[1]][[1]] /. {y[x] → y}
```

Out[8]=
$$-\frac{2\, e^{\frac{1}{2}\left(-5x+\frac{1}{y}\right)^2} + 5\sqrt{2\pi}\ x\ \text{Erfi}\left[\frac{-5x+\frac{1}{y}}{\sqrt{2}}\right]}{10x}$$

In[9]:=
```
ContourPlot[expr, {x, -0.4, -0.1}, {y, 1., 1.8}]
```

Out[9]=



Here is the plot of the solutions to a system of two linear ODEs. The `WorkingPrecision` option in `Plot` is required because the solution is fairly complicated.

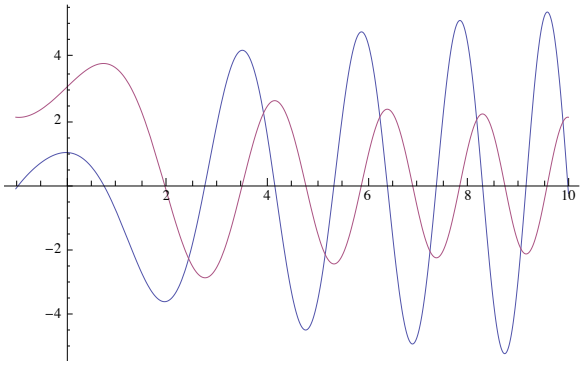
In[10]:=
```
sol =
 DSolve[{x'[t] + t * y[t] == 0, 2 y'[t] - 3 x[t] == 0, x[0] == 1, y[0] == 3}, {x, y}, t]
```

Out[10]=

$$\left\{\left\{x \to \text{Function}\left[\{t\},\right.\right.\right.$$
$$\frac{1}{6}\left(-3\,3^{1/3}\,\text{AiryAiPrime}\left[\left(-\frac{3}{2}\right)^{1/3} t\right]\text{Gamma}\left[\frac{1}{3}\right] + 3^{5/6}\,\text{AiryBiPrime}\left[\left(-\frac{3}{2}\right)^{1/3} t\right]\text{Gamma}\left[\frac{1}{3}\right] + 9\,(-1)^{1/3}\,2^{2/3}\right.$$
$$\left.\text{AiryAiPrime}\left[\left(-\frac{3}{2}\right)^{1/3} t\right]\text{Gamma}\left[\frac{2}{3}\right] + 3\,(-1)^{1/3}\,2^{2/3}\sqrt{3}\,\text{AiryBiPrime}\left[\left(-\frac{3}{2}\right)^{1/3} t\right]\text{Gamma}\left[\frac{2}{3}\right]\right)\Big],$$
$$y \to \text{Function}\left[\{t\}, \frac{1}{4}\left(3\,(-1)^{2/3}\,2^{1/3}\,\text{AiryAi}\left[\left(-\frac{3}{2}\right)^{1/3} t\right]\text{Gamma}\left[\frac{1}{3}\right] - (-1)^{2/3}\,2^{1/3}\sqrt{3}\,\text{AiryBi}\left[\left(-\frac{3}{2}\right)^{1/3} t\right]\right.\right.$$
$$\left.\left.\left.\text{Gamma}\left[\frac{1}{3}\right] + 6\times 3^{2/3}\,\text{AiryAi}\left[\left(-\frac{3}{2}\right)^{1/3} t\right]\text{Gamma}\left[\frac{2}{3}\right] + 6\times 3^{1/6}\,\text{AiryBi}\left[\left(-\frac{3}{2}\right)^{1/3} t\right]\text{Gamma}\left[\frac{2}{3}\right]\right)\right]\right\}\right\}$$

In[11]:=
```
Plot[Evaluate[{x[t], y[t]} /. sol], {t, -1, 10}, WorkingPrecision → 20]
```

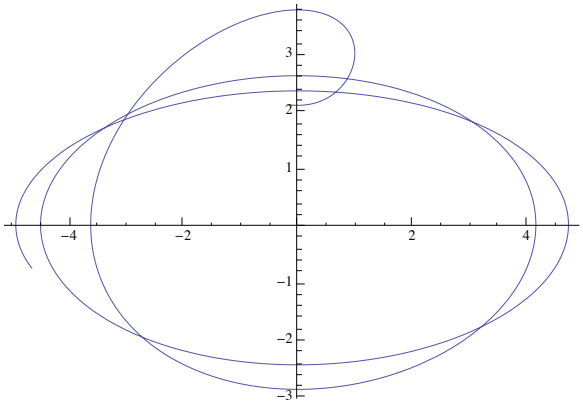
Out[11]=



The `ParametricPlot` function can be used to trace the solution curve $\{x[t], y[t]\}$ in the plane.

In[12]:=
```
ParametricPlot[Evaluate[{x[t], y[t]} /. sol], {t, -1, 7}, WorkingPrecision → 20]
```

Out[12]=



Here is the plot for the solution to a DAE.

In[13]:=
```
sol = DSolve[{x''[t] + 3 * y[t] == UnitStep[t],
    x[t] - 5 * y[t] == t^2, x[0] == 1, x'[0] == 1}, {x, y}, t]
```

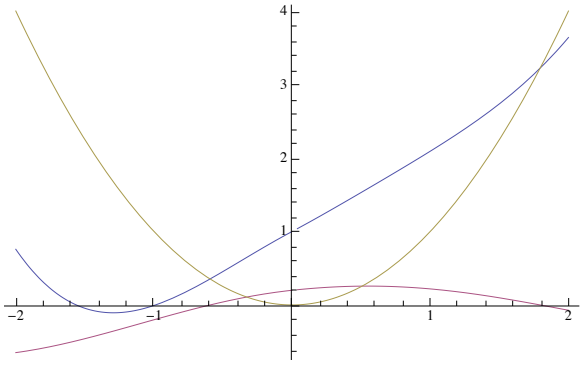
Out[13]=

$$\left\{\left\{x \to \text{Function}\left[\{t\}, \frac{1}{96}\left(-430 - 90\,t + 177\,t^2 + 526\,\text{Cos}\left[\sqrt{\frac{3}{5}}\,t\right] + 62\sqrt{15}\,\text{Sin}\left[\sqrt{\frac{3}{5}}\,t\right] - 160\,\text{UnitStep}[-t] + 160\,\text{Cos}\left[\sqrt{\frac{3}{5}}\,t\right]\text{UnitStep}[-t]\right)\right],\right.$$

$$\left.y \to \text{Function}\left[\{t\}, \frac{1}{480}\left(-430 - 90\,t + 81\,t^2 + 526\,\text{Cos}\left[\sqrt{\frac{3}{5}}\,t\right] + 62\sqrt{15}\,\text{Sin}\left[\sqrt{\frac{3}{5}}\,t\right] - 160\,\text{UnitStep}[-t] + 160\,\text{Cos}\left[\sqrt{\frac{3}{5}}\,t\right]\text{UnitStep}[-t]\right)\right]\right\}\right\}$$

In[14]:=
```
Plot[Evaluate[{x[t], y[t], x[t] - 5 * y[t]} /. sol], {t, -2, 2}]
```

Out[14]=



Here is the general solution to a linear PDE.

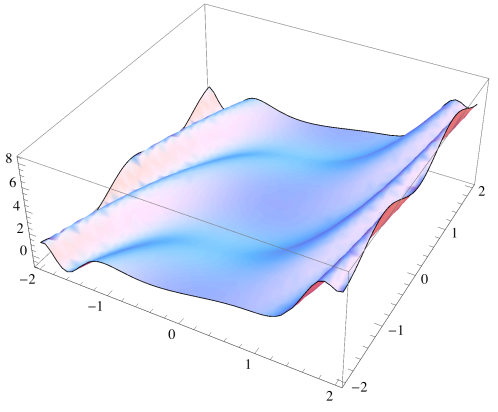
```
In[15]:= sol = DSolve[D[u[x, y], x] + x^2 * D[u[x, y], y] == Exp[x], u, {x, y}]
```

$$\text{Out[15]= } \left\{\left\{u \to \text{Function}\left[\{x, y\}, e^x + C[1]\left[\frac{1}{3}\left(-x^3 + 3y\right)\right]\right]\right\}\right\}$$

Here is a plot of the solution surface for a particular choice of the arbitrary function `C[1]`.

```
In[16]:= Plot3D[Evaluate[u[x, y] /. sol /. {C[1][t_] → Sin[3 * t]}[[1]]],
          {x, -2, 2}, {y, -2, 2}, Mesh → False]
```

Out[16]=



The GeneratedParameters Option

The general solution to a differential equation contains undetermined coefficients that are labeled `C[1]`, `C[2]`, and so on.

This example has one undetermined parameter, `C[1]`.

```
In[1]:= DSolve[y'[x] + y[x] == 1, y[x], x]
```

$$\text{Out[1]= } \left\{\left\{y[x] \to 1 + e^{-x} C[1]\right\}\right\}$$

To change the name of the undetermined parameter, use the `GeneratedParameters` option.

This changes the name of the undetermined coefficient to `P[1]`.

```
In[2]:= DSolve[y'[x] + y[x] == 1, y[x], x, GeneratedParameters → P]
```

$$\text{Out[2]= } \left\{\left\{y[x] \to 1 + e^{-x} P[1]\right\}\right\}$$

The parameter `C` should be thought of as a pure function that acts on a set of indices to generate different constants `C[i]`.

This shows the behavior of C.

```
In[3]:= parameter = C[#] &;

In[4]:= indexset = {1, 2, 3, 4};

In[5]:= parameter[indexset[[1]]]
Out[5]= C[1]

In[6]:= parameter[indexset[[3]]]
Out[6]= C[3]
```

Internally, the use of a pure function allows DSolve to increment the argument i in C[i] correctly for higher-order ODEs and systems of ODEs.

GeneratedParameters can be specified using a pure function.

```
In[7]:= DSolve[y''[x] + y[x] == 1, y[x], x, GeneratedParameters → (const[#] &)]
Out[7]= {{y[x] → 1 + const[1] Cos[x] + const[2] Sin[x]}}
```

Using a pure function is particularly useful if you want to begin indexing the parameters at any value other than 1 (the default).

This uses a pure function to label the parameters in the previous example $const[2]$ and $const[3]$.

```
In[8]:= DSolve[y''[x] + y[x] == 1, y[x], x, GeneratedParameters → (const[1 + #] &)]
Out[8]= {{y[x] → 1 + const[2] Cos[x] + const[3] Sin[x]}}
```

It is sometimes useful to display the solution using subscripts or other styles for the parameter indices.

Here, the parameters are named using subscripted variables.

```
In[9]:= DSolve[y''[x] + y[x] == 1, y[x], x, GeneratedParameters → (c#1 &)]
Out[9]= {{y[x] → 1 + Cos[x] c1 + Sin[x] c2}}
```

Finally, with Module variables, you can get names for the parameters that are unique across different invocations of DSolve.

Here the same DSolve call generates different parameter names.

```
In[10]:= DSolve[y''[x] + y[x] == 1, y[x], x, GeneratedParameters → Module[{C}, C[#] &]]
Out[10]= {{y[x] → 1 + Cos[x] C$102[1] + C$102[2] Sin[x]}}
```

```
In[11]:= DSolve[y''[x] + y[x] == 1, y[x], x, GeneratedParameters → Module[{C}, C[#] &]]
Out[11]= {{y[x] → 1 + Cos[x] C$106[1] + C$106[2] Sin[x]}}
```

Symbolic Parameters and Inexact Quantities

The differential equations that arise in practice are of two types.

- Equations in which the only variables are the independent and dependent variables. Thus, all the variables that appear in the first argument to `DSolve` are also in the second or third arguments.
- Equations in which there are other symbolic quantities, such as mass or the spring constant. The solution in this case depends on the independent variables, the dependent variables, and the additional symbolic parameters.

Here is an example of the first type.

```
In[1]:= DSolve[y''[x] - 8 * x * y[x] == 0, y, x]
Out[1]= {{y → Function[{x}, AiryAi[2 x] C[1] + AiryBi[2 x] C[2]]}}
```

Here is an example of the second type. This equation has a symbolic parameter k.

```
In[2]:= sol = DSolve[{y''[x] + k^2 * y[x] == x^2, y[0] == 3 k + 1, y'[0] == k}, y, x]
```

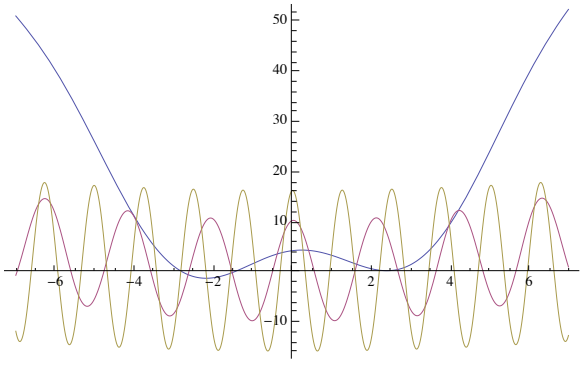
$$\text{Out[2]= } \left\{\left\{y \to \text{Function}\left[\{x\}, \frac{-2 + k^2 x^2 + 2 \cos[k x] + k^4 \cos[k x] + 3 k^5 \cos[k x] + k^4 \sin[k x]}{k^4}\right]\right\}\right\}$$

`DSolve` is equipped to deal with both types of equations. It is extremely useful to have the solution available for all possible values of the parameters in the second type of equation.

Here is a plot of the previous solution for different values of the parameter k.

```
In[3]:= Plot[Evaluate[Table[y[x] /. sol /. {k → i}, {i, 1, 5, 2}]], {x, -7, 7}]
```

Out[3]=



It should be noted that the presence of symbolic parameters can lead to fairly complicated output.

This is seen in the following example (equation 2.14, page 401 of [K59]).

In[4]:=
```
eqn = y''[x] - c * x^a * y[x] ;
```

In[5]:=
```
sol = DSolve[eqn == 0, y, x]
```

Out[5]=
$$\left\{\left\{y \to \text{Function}\left[\{x\},\ (-1)^{\frac{1}{2+a}}\ (2+a)^{-\frac{1}{2+a}}\ c^{\frac{1}{2(2+a)}}\ x^{1-\frac{1+\frac{a}{2}}{2+a}}\ \text{BesselI}\left[\frac{1}{2+a},\ \frac{2\sqrt{c}\ x^{\frac{2+a}{2}}}{2+a}\right]\ \text{C}[2]\ \text{Gamma}\left[1+\frac{1}{2+a}\right]+\right.\right.$$
$$\left.\left.(2+a)^{-\frac{1}{2+a}}\ c^{\frac{1}{2(2+a)}}\ x^{\frac{1+\frac{a}{2}}{2+a}}\ \text{BesselI}\left[\frac{1}{-2-a},\ \frac{2\sqrt{c}\ x^{\frac{2+a}{2}}}{2+a}\right]\ \text{C}[1]\ \text{Gamma}\left[\frac{1}{2+a}+\frac{a}{2+a}\right]\right]\right\}\right\}$$

However, for some special values of the parameters, the solution might be significantly simpler.

For these values of a and c, the solution is much more simple.

In[6]:=
```
sol1 = y[x] /. sol[[1]] /. {a → 0, c → 4}
```

Out[6]=
$$\text{C}[1]\ \text{Cosh}[2\,x] + \frac{1}{2}\,i\,\text{C}[2]\ \text{Sinh}[2\,x]$$

Occasionally, a solution is valid for most, but not all, values of the parameters.

Since the input in this example is not valid at $a = 0$, the solution has the same limitation.

In[7]:=
```
sol = DSolve[{y'[x] == x, y[0] == 1 / a}, y, x]
```

Out[7]=
$$\left\{\left\{y \to \text{Function}\left[\{x\},\ \frac{2 + a\,x^2}{2\,a}\right]\right\}\right\}$$

Of course, there is a simple remedy in this case: setting $k = \frac{1}{a}$.

In[8]:=
```
sol = DSolve[{y'[x] == x, y[0] == k}, y, x]
```

Out[8]=
$$\left\{\left\{y \to \text{Function}\left[\{x\},\ \frac{1}{2}\left(2\,k + x^2\right)\right]\right\}\right\}$$

In summary, the ability to solve differential equations with symbolic parameters is a *powerful and essential feature* of any symbolic solver such as `DSolve`. However, the following points should be noted.

- The solution might be complicated, and such calculations often require significant time and memory.
- The answer might not be valid for certain exceptional values of the parameters.

- The solution might be easy to verify symbolically for some special values of the parameters, but in the general case a numerical verification method is preferable.

Numerical quantities in *Mathematica* can be of three types: infinite precision, machine precision, or arbitrary precision. The first type of number is referred to as "exact", while the remaining two types represent incomplete information and are therefore called "inexact".

Here is a simple example showing all three kinds of numbers.

```
In[9]:= exactpi = Pi;

In[10]:= Precision[exactpi]
Out[10]= ∞

In[11]:= (machinepi = N[Pi]) // InputForm
Out[11]//InputForm= 3.141592653589793

In[12]:= Precision[machinepi]
Out[12]= MachinePrecision

In[13]:= highprecisionpi = N[Pi, 30]
Out[13]= 3.14159265358979323846264338328

In[14]:= Precision[highprecisionpi]
Out[14]= 30.
```

Since `DSolve` is a symbolic solver, the algorithms used by it are primarily based on the assumption of exact input. However, equations that contain inexact quantities are handled by `DSolve` in the usual way.

These equations contain the inexact number 3. and 3.`40, respectively.

```
In[15]:= DSolve[x'[t] == 3. x[t], x, t]
Out[15]= {{x → Function[{t}, e^(3. t) C[1]]}}

In[16]:= DSolve[x'[t] == 3.`40 * x[t], x, t]
Out[16]= {{x → Function[{t}, e^(3.000000000000000000000000000000000000000 t) C[1]]}}
```

Inexact input could arise, for example, when the coefficients in the equations are derived from a previous calculation and are known only approximately. In such cases, it might not be practical to convert the equations to exact form, as this could slow down the calculation significantly.

Here is a system of linear ODEs that all have exact coefficients. Note that even with a fairly small value of n, the calculation takes a long time to finish.

```
In[17]:= n = 8;
         x0[t_] := 0;
         xn[t_] := 1;
         eqns = Table[{xi'[t] - (xi+1[t] - 2 xi[t] + xi-1[t]), xi[0] - (1 / n)}, {i, n - 1}];
         vars = Table[xi, {i, n - 1}];
         sol = DSolve[Map[# == 0 &, Flatten[eqns]], vars, t]; // Timing
Out[17]= {35.703, Null}

In[18]:= LeafCount[sol]
Out[18]= 212 851
```

This verifies the solution. Since the solution is complicated, a numerical verification method is used.

```
In[19]:= eqns /. sol /. {t → RandomReal[{0, 1}, WorkingPrecision → 200]} // N // Chop //
           Flatten // Union
Out[19]= {0}
```

If a single inexact quantity is introduced (in the function $x_0[t]$), the solution is returned more quickly.

```
In[20]:= n = 8;
         x0[t_] := 0.;
         xn[t_] := 1;
         eqns = Table[{xi'[t] - (xi+1[t] - 2 xi[t] + xi-1[t]), xi[0] - (1 / n)}, {i, n - 1}];
         vars = Table[xi, {i, n - 1}];
         sol = DSolve[Map[# == 0 &, Flatten[eqns]], vars, t]; // Timing
Out[20]= {0.75, Null}

In[21]:= LeafCount[sol]
Out[21]= 1563

In[22]:= eqns /. sol /. {t → RandomReal[{0, 1}]} // N[#] & // Chop // Flatten // Union
Out[22]= {0}
```

Thus, it is often desirable to continue working with inexact quantities even within a symbolic function such as `DSolve`. However, it should be noted that the solution obtained in such cases could have a certain amount of numerical error and should be checked carefully. It is therefore recommended that if the problem size is not too large (for instance, if there are fewer than five equations), the input should be converted to exact form using the `Rationalize` function.

This equation contains inexact quantities.

```
In[23]:= DSolve[{x'[t] == 0.0001 * x[t], x[0] == 3.07}, x[t], t]
Out[23]= {{x[t] → 3.07 e^(0.0001 t)}}
```

Here the equation is converted to exact form before being solved.

```
In[24]:= DSolve[Rationalize[{x'[t] == 0.0001 * x[t], x[0] == 3.07}, 0], x[t], t]
```

$$\text{Out[24]}= \left\{\left\{x[t] \to \frac{307\, e^{t/10\,000}}{100}\right\}\right\}$$

Is the Problem Well-Posed?

`DSolve` returns a general solution for a problem if no initial or boundary conditions are specified.

The general solution to this equation is returned.

```
In[1]:= DSolve[y'[x] == 1 - y[x], y, x]
Out[1]= {{y → Function[{x}, 1 + e^-x C[1]]}}
```

However, if initial or boundary conditions are specified, the output from `DSolve` must satisfy both the underlying differential equation as well as the given conditions.

Here is an example with a boundary condition.

```
In[2]:= eqns = {y'[x] == 1 - y[x], y[3] == 5};
In[3]:= sol = DSolve[eqns, y, x]
Out[3]= {{y → Function[{x}, e^-x (4 e^3 + e^x)]}}

In[4]:= eqns /. sol
Out[4]= {{True, True}}
```

In such cases, it is useful to check whether `DSolve` has been asked a reasonable question—in other words, to check whether the problem is well-posed. An initial or boundary value problem is said to be *well-posed* if a solution for it is guaranteed to exist in some well-known class of functions (for example, analytic functions), if the solution is unique, and if the solution depends continuously on the data. Given an ODE of order n (or a system of n first-order equations) and n initial conditions, there are standard existence and uniqueness theorems that show that the problem is well-posed under a specified set of conditions. The right-hand side of the first-order linear ODE in the previous example is a polynomial in $y[x]$ and hence infinitely differentiable. This is sufficient to apply Picard's existence and uniqueness theorem, which only requires that the right-hand side be Lipschitz-continuous.

Most problems that arise in practice are well-posed since they are derived from sound theoretical principles. However, as a note of caution, the following are examples where `DSolve` might have difficulty finding a satisfactory solution to the problem.

Here is the solution to a first-order ODE in which the right-hand side fails to satisfy the Lipschitz condition around 0.

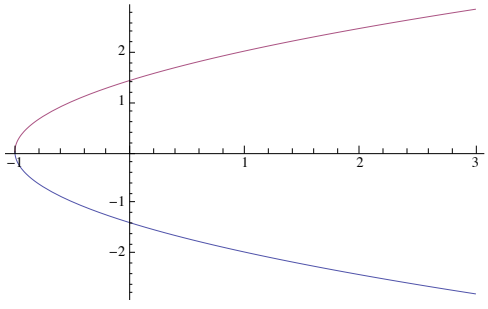
```
In[5]:= generalsol = DSolve[{y'[x] == 1 / y[x]}, y, x]
```

```
Out[5]= {{y → Function[{x}, -√2 √(x + C[1])]}, {y → Function[{x}, √2 √(x + C[1])]}}
```

The general solution has two branches.

```
In[6]:= Plot[Evaluate[y[x] /. generalsol /. {C[1] → 1}], {x, -1, 3}]
```

```
Out[6]=
```



This initial value problem is well-posed in a region around the initial condition and hence `DSolve` succeeds in picking out the correct branch for the given initial condition.

```
In[7]:= DSolve[{y'[x] == 1 / y[x], y[0] == 1}, y, x]
```

```
DSolve::bvnul :
  For some branches of the general solution, the given boundary conditions lead to an empty solution. ≫
```

```
Out[7]= {{y → Function[{x}, √(1 + 2 x)]}}
```

Here is a second-order ODE. The boundary conditions do not allow any solution to this problem.

```
In[8]:= DSolve[{y''[x] + y[x] == 0, y[0] == 1, y[Pi] == 3}, y[x], x]
```

```
DSolve::bvnul :
  For some branches of the general solution, the given boundary conditions lead to an empty solution. ≫
```

```
Out[8]= {}
```

In this example, `DSolve` returns a pair of solutions. As the table shows, the first solution is only valid for values of x greater than or equal to 2.

```
In[9]:= eqn = y'[x] == y[x]^(1 / 2);
```

```
In[10]:= sol = DSolve[{eqn, y[0] == 1}, y, x]
```

$$\text{Out[10]= } \left\{\left\{y \to \text{Function}\left[\{x\}, \frac{1}{4}\left(4 - 4\,x + x^2\right)\right]\right\}, \left\{y \to \text{Function}\left[\{x\}, \frac{1}{4}\left(4 + 4\,x + x^2\right)\right]\right\}\right\}$$

```
In[11]:= Table[eqn /. sol /. {x → i}, {i, 0, 5}]
Out[11]= {{False, True}, {False, True}, {True, True}, {True, True}, {True, True}, {True, True}}
```

Finally, it is possible that a problem has a solution, but that `DSolve` fails to find it because the general solution is in implicit form or involves higher transcendental functions.

In this example, a solution is available only after inverting the roles of the dependent and independent variables.

```
In[12]:= DSolve[y'[x] == 1 / (x - y[x]) && y[0] == 1, y, x]

InverseFunction::ifun: Inverse functions are being used. Values may be lost for multivalued inverses. >>

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

DSolve::bvnul:
  For some branches of the general solution, the given boundary conditions lead to an empty solution. >>

Out[12]= {}
```

```
In[13]:= DSolve[x'[y] == (x[y] - y) && x[1] == 0, x, y]
```

$$\text{Out[13]= } \left\{\left\{x \to \text{Function}\left[\{y\}, \frac{e - 2\,e^y + e\,y}{e}\right]\right\}\right\}$$

This concludes the discussion of the basic principles for effectively working with `DSolve`. See the list of "references" that were found to be useful either during the development of `DSolve` or during the preparation of this documentation.

References

[AB04] Abell, M. L. and J. P. Braselton. *Differential Equations with Mathematica*, 3rd ed. Elsevier Academic Press, 2004.

[A89] Abramov, S. A. "Rational Solutions of Linear Differential and Difference Equations with Polynomial Coefficients." *USSR Comput. Maths. Math. Phys.* 29 (1989): 7-12.

[A96] Abramov, S. A. "Symbolic Search Algorithms for Partial d'Alembertian Solutions of Linear Equations." *Programming and Computer Software* 22, no. 1 (1996): 26.

[AB01] Abramov, S. A. and M. Bronstein. "On Solutions of Linear Functional Systems." In *Proc. ISSAC'01*, 1-6, 2001.

[AK91] Abramov, S. A. and K. Yu. Kvansenko. "Fast Algorithms to Search for the Rational Solutions of Linear Differential Equations with Polynomial Coefficients." In *Proc. ISSAC'91*, 267-270, 1991.

[AP94] Abramov, S. A. and M. Petkovsek. "D'Alembertian Solutions of Linear Differential and Difference Equations." In *Proc. ISSAC'94*, 169-174, 1994.

[ABP95] Abramov, S. A., M. Bronstein, and M. Petkovsek. "On Polynomial Solutions of Linear Operator Equations." In *Proc. ISSAC'95*, 290-296, 1995.

[B93] Bocharov, A. "Symbolic Solvers for Nonlinear Differential Equations." *The Mathematica Journal* 3, no. 2 (1993): 63-69.

[BD97] Boyce, W. F. and R. C. DiPrima, *Elementary Differential Equations*. John Wiley and Sons, 1997.

[BM91] Boyer, C. B. and U. C. Merzbach, *A History of Mathematics*, 2nd ed. John Wiley, 1991.

[B91] Bronstein, M. "The Risch Differential Equation on an Algebraic Curve." In *Proc. ISSAC'91*, 241-246, 1991.

[B92] Bronstein, M. "On Solutions of Linear Ordinary Differential Equations in Their Coefficient Field." *J. Symbolic Computation* 13 (1992): 413-439.

[B92a] Bronstein, M. "Integration and Differential Equations in Computer Algebra." *Programming and Computer Software* 18, no.5 (1992): 201-217.

[B92b] Bronstein, M. "Linear Ordinary Differential Equations: Breaking through the Order 2 Barrier." In *Proc. ISSAC'92*, 42-48, 1992.

[C80] Campbell, S. L. *Singular Systems of Differential Equations I*. Pitman, 1980.

[C82] Campbell, S. L. *Singular Systems of Differential Equations II*. Pitman, 1982.

[CC04] Chan, L. and E. S. Cheb-Terrab. "Non-Liouvillian Solutions for Second Order Linear ODEs." In *Proc. ISSAC'04*, 80-86, 2004.

[CDM97] Cheb-Terrab, E. S., L. G. S. Duarte, and L. A. C. P. da Mota. "Computer Algebra Solving of First Order ODEs Using Symmetry Methods." *Comp. Phys. Comm.* 101 (1997): 254.

[CR99] Cheb-Terrab, E. S. and A. D. Roche. "Integrating Factors for Second Order ODEs." *J. Symbolic Computation* 27 (1999): 501.

[CR00] Cheb-Terrab, E. S. and A. D. Roche. "Abel ODEs: Equivalence and Integrable Classes." *Comp. Phys. Comm.* 130 (2000): 204.

[D58] Drazin, M. P. "Pseudo Inverses in Associative Rays and Semigroups" *American Mathematical Monthly* 65 (1958): 506-514.

[F59] Forsyth, A. R. *Theory of Differential Equations*. Dover, 1959.

[I99] Ibragimov, N. H. *Elementary Lie Group Analysis and Ordinary Differential Equations*. John Wiley & Sons, 1999.

[I44] Ince, E. L. *Ordinary Differential Equations*. Dover, 1944.

[K59] Kamke, E. *Differentialgleichungen: Losungsmethoden und Losungen*. Akademische Verlagsgesellschaft, 1959.

[K74] Kamke, E. *Differentialgleichungen Losungsmethoden und Losungen, Bd. II: Partielle differentialgleichungen*. Chelsea Publishing Co., 1974.

[K00] Kevorkian, J. *Partial Differential Equations: Analytical Solution Techniques*. Springer-Verlag, 2000.

[K72] Kline, M. *Mathematical Thought from Ancient to Modern Times, Vol. 2*. Oxford University Press, 1972.

[K86] Kovacic, J. J. "An Algorithm for Solving Second Order Linear Homogeneous Differential Equations." *J. Symbolic Computation* 2 (1986): 3-43.

[L01] Kovacic, J. J. "An Algorithm for Solving Second Order Linear Homogeneous Differential Equations." Lecture, City College of New York, 2001.

[KPS03] Kythe, P. K., P. Puri, and M. R. Schäferkotter, *Partial Differential Equations and Boundary Value Problems with Mathematica,* 2nd ed. Chapman and Hall/CRC, 2003.

[L65] Lebedev, N. N. *Special Functions and Their Applications*. Prentice-Hall, 1965.

[M00] Meyer, C. D. *Matrix Analysis and Applied Linear Algebra*. SIAM, 2000.

[M60] Murphy, G. M. *Ordinary Differential Equations and Their Solutions*. Van Nostrand, 1960.

[M47] McLachlan, N. W. *Theory and Application of Mathieu Functions*. Oxford University Press, 1947.

[O95] Olver, P. J. *Equivalence, Invariants and Symmetry*. Cambridge University Press, 1995.

[PZ95] Polynanin, A. D. and V. F. Zaitsev, *Handbook of Exact Solutions for Ordinary Differential Equations*. CRC Press, 1995.

[S81] Saunders, B. D. "An Implementation of Kovacic's Algorithm for Solving Second Order Linear Homogeneous Differential Equations." In *Proc. SYMSAC'81* (P. Wang, ed.), 105, 1981.

[S85] Schlesinger, L. *Handbuch der Theorie der Linearen Differentialgleichungen*. Teubner, 1985.

[SS98] Shirvani, M. and J. W.-H. So. "Solutions of Linear Differential Algebraic Equations." *SIAM Review* 40, no. 2 (1998): 344-346.

[S57] Sneddon, I. *Elements of Partial Differential Equations*. McGraw-Hill, 1957.

[T05] Trott, M. *The Mathematica GuideBook for Symbolics*. Springer-Verlag, 2005.

[UW96] Ulmer, F. and J-A. Weil. "Note on Kovacic's Algorithm." *J. Symbolic Computation* 22 (1996): 179-200.

[WW27] Whittaker, E. T. and G. N. Watson. *A Course of Modern Analysis*, 4th ed. Cambridge University Press, 1927.

[W02] Wolfram, S. *A New Kind of Science*. Wolfram Media, Inc., 2002.

[W04] Wolfram, S. *The Mathematica Book*, 5th ed. Wolfram Media, Inc., 2004.

[Z89] Zwillinger, D. *Handbook of Differential Equations*. Academic Press, 1989