dsolve - Solve ordinary differential equations (ODEs)

Calling Sequences:

dsolve(ODE)

dsolve(ODE, y(x), extra_args)

dsolve({ODE, ICs}, y(x), extra_args)

dsolve({sysODE, ICs}, {funcs}, extra_args)

Parameters:

ODE       - an ordinary differential equation

y(x)      - any indeterminate function of one variable

ICs       - initial conditions

{sysODE}   - a set with a system of ODEs

{funcs}    - a set with indeterminate functions

extra_args - optional, depends on the type of problem being solved (see below)

* Herein \( \{x, y(x)\} \) represent any pair of independent and dependent variables.

Description:

- As a general ODE solver, dsolve is able to handle different types of ODE problems. These include:
  * looking for closed form solutions for a single ODE (see below) or a system of ODEs (see ?dsolve, system);
  * solving ODEs or a system of them with given initial conditions (boundary value problems, see ?dsolve, ICs);
  * looking for formal power series solutions (see ?dsolve, formal_series) to a linear ODE with polynomial coefficients;
  * looking for solutions using integral transforms (Laplace, Fourier, etc.; see ?dsolve, integral_transform);
  * looking for numerical (see ?dsolve, numeric) or series solutions (see ?dsolve, series) to ODEs or systems of them.

In the case of a single ODE, dsolve will try to solve it using either classification methods or symmetry methods; in the latter case, dsolve first looks for the generators of symmetry groups of the given ODE, and then uses this information to integrate it, or at least reduce its order. Classification methods are
used when the ODE matches a recognizable pattern (that is, for which a solving method is already implemented), and symmetry methods are reserved for the non-classifiable cases.

To see what method is being used to solve a given ODE, you can assign the following (see \texttt{infolevel}): \[
> \texttt{infolevel[dsolve]} := 3;
\]

\textbf{INPUT AND OUTPUT}

- Given an ODE, an extra argument indicating the dependent variable is required only when the given ODE involves more than one function being differentiated. When extra arguments are given, they can be placed in any order after the first one.

- Closed form solutions are returned by \texttt{dsolve} as a sequence of explicit (\(y(x)=F(x,\_C_n)\)) or implicit (\(F(y(x),x,\_C_n)=0\)) equations, where the \(\_C_n\) (n=integer) are arbitrary constants. By default, \texttt{dsolve} will return the answer in explicit form, unless \texttt{solve} is not able to isolate the dependent variable, or its isolation requires the inversion of fractional powers, or the isolation can only be obtained by means of \texttt{RootOf} (as say \(y(x)=\text{RootOf}(...)\)). Note that \texttt{dsolve} is sensitive to the \texttt{\_EnvExplicit} variable used by \texttt{solve}.

- For first order ODEs, especially when they are of high degree in \(dy/dx\), answers may also appear in parametric form as \([x(_T)=f(_T), y(_T)=g(_T)]\), where \(T\) is the parameter and the right hand sides are explicit expressions of \(T\).

- For high order ODEs, \texttt{dsolve} might succeed in reducing the order of the ODE but not in solving the problem to the end. The answer then will be expressed using a scheme for conveying reductions of order (see \texttt{dsolve,ODESolStruc} and the example below). The user may then be able to obtain a solution for the reduced ODE by manipulating it using the tools available in \texttt{DEtools}, or as a \texttt{series} expansion, or by other means. When a solution to the reduced ODE is obtained, a solution to the original problem can be built using \texttt{DEtools[buildsol]}

- For linear ODEs, when \texttt{dsolve} is not able to find a solution or a reduction of order, an answer using \texttt{DESol} is returned. DESol structures can also appear in answers involving reductions of orders of linear ODEs. Although DESol structures don't contain more information than the ODE itself, they can be useful in further computations since Maple can manipulate them; for example, by expanding in series, or by simplifying, etc. (see \texttt{DESol}).

- Integrals appearing in answers returned by \texttt{dsolve} are expressed using the inert \texttt{Int} and \texttt{Intat} (not \texttt{int} or \texttt{intat}). These integrals will appear when \texttt{int} was not able to calculate them or when it appeared to be convenient not to evaluate them during the solving process. One can request the evaluation of these integrals using the \texttt{value} command.

- Floating point numbers appearing in the given ODE will be converted to rational exact numbers before attempting to solve the problem (see optional argument \texttt{convert\_to\_exact=false} below).

- The symbol variables \(_\text{yn} (n=\text{integer})\) are reserved for use by internal routines and should not be
THE OPTIONAL ARGUMENTS

- For the optional arguments in the context of IC problems, systems of ODEs, series or numeric solutions, or the use of integral transforms, see the links to the respective help pages above.

- In the case of single ODEs, optional arguments can be given in any order after the first one. A summary of the optional arguments most frequently used is given by:

  'implicit' - to avoid dsolve trying to make an answer explicit;

  'explicit' - to request answers in explicit form in all cases (provided that solve succeeds in isolating the dependent variable);

  'parametric' - only for first order ODEs, to request the use of only the parametric solving scheme. Note however that by default dsolve will try to remove the parameter used during the solving process. To keep the parameter, one must give the optional argument 'implicit' together with 'parametric';

  'useInt' - to request the use of Int (the inert integral) instead of int during the solving process. This option is useful to speed up the solving process in many cases, and to see the form of the answer before the integrals are performed (pedagogical purposes). To perform the integrals afterwards one can apply the value command to dsolve's answer.

  'class' - to request the use of only classification methods (see ?odeadvisor) and avoid using symmetry methods;

  'Lie' - to request the use of Lie's symmetry methods before trying the classification methods. Additional information about the symmetry scheme and various other related optional arguments is found in ?dsolve, Lie;

  '[method1, method2, ...]' - to request the use of only [method1, method2, ...], and in that order, when solving a given ODE. Each method is represented by a related keyword in a list of methods. For first order ODEs, for instance, the methods available are:

> `dsolve/methods`[1];

[ quadrature, linear, Bernoulli, separable, inverse_linear, homogeneous, Chini, lin_sym, exact, Abel, pot_sym ]

> `dsolve/methods`[1,'semiclass'];

[ Riccati ]

> `dsolve/methods`[1,'high_degree'];

[ missing, dAlembert, homogeneous_B, sym_implicit ]
THE SOLVING METHODS

• The present implementation of `dsolve` makes use of a combination of symmetry methods and classification methods, including various decision algorithms for linear ODEs.

• Most of the classification methods implemented have a related help page (see ?odeadvisor). These help pages can be displayed by either following the hyperlinks found in that page, or by using the odeadvisor command itself. The decision algorithms for linear ODEs are explained in ?dsolve.linear. The implementation of Lie's symmetry method in `dsolve` is explained in ?dsolve.Lie. For references about the solving methods and its computational implementations see ?dsolve, references. For details about the new `dsolve`'s structure and solving strategy see E.S. Cheb-Terrab, “The new dsolve of MapleV R5”, MapleTech (in preparation).

• Other tools for working with ODEs are available in the DEtools package. These include, for instance, routines for plotting ODEs; manipulating linear ODEs and linear differential operators; and performing most of the steps of the symmetry method. For all this, see ?DEtools. For changing variables see ?dchange.

Examples:

• The examples below illustrate the use of `dsolve` in solving a single ODE. For examples related to more specific problems see: `dsolve.initial_conditions`, `dsolve.system`, `dsolve.numeric`, `dsolve.integral_transform`, `dsolve.series`, and `dsolve.formal_series`. 
First order ODEs (see \texttt{odeadvisor}).

\begin{verbatim}
> ode1 := diff(y(x),x)-y(x)^2+y(x)*sin(x)-cos(x);

ode1 := \left( \frac{\partial}{\partial x} y(x) \right) - y(x)^2 + y(x) \sin(x) - \cos(x)

> ans1 := dsolve(ode1);

ans1 := y(x) = \sin(x) - \frac{-C1 \ e^{-\cos(x)}}{-C1 \int e^{-\cos(x)} \ dx + 1}

> with(DEtools): odeadvisor(ode1);

[\texttt{\_Riccati}]
\end{verbatim}

ODE of high degree in dy/dx.

\begin{verbatim}
> ode2 := x^(n-1)*diff(y(x),x)^n-n*x*diff(y(x),x)+y(x);

ode2 := x^{(n-1)} \left( \frac{\partial}{\partial x} y(x) \right)^n - n \ x \ \left( \frac{\partial}{\partial x} y(x) \right) + y(x)

In these cases, dsolve first tries a set of methods for high degree ODEs, including a parametric solving scheme (see \texttt{odeadvisor,parametric}).

> dsolve(ode2, parametric);

y(x) = -_C1^{(n-1)} + _C1 \ n \ \left( \frac{x}{_C1} \right)^\frac{1}{n}

To see the answer "in parametric form" one can use the extra argument 'implicit'. For some high degree ODEs, dsolve may return possible singular solutions together with the general solution. Also, depending on the case dsolve may not be able to remove the parameter introduced in the solving process. For example,

\begin{verbatim}
> ode3 := y(x)/x = F(diff(y(x),x));

ode3 := \frac{y(x)}{x} = F \left( \frac{\partial}{\partial x} y(x) \right)

> odeadvisor(ode3); \texttt{\# try also: odeadvisor(ode3, help)};

[[\texttt{\_homogeneous, class B}], \texttt{\_dAlembert}]
\end{verbatim}
A linear ODE

\[ \sin(x)\frac{\partial}{\partial x}(y(x)) - \cos(x)\cdot y(x) = 0 \]

One can indicate to `dsolve` to use a specific sequence of 'methods' to tackle the ODE. In this example the method for 'linear' ODEs is indicated; in addition the optional argument 'useInt' makes `dsolve` not evaluate the integral appearing in the answer.

\[ y(x) = C1 \cdot e^{\int \frac{\cos(x)}{\sin(x)} \, dx} \]

To see the answer before evaluating integrals is useful to understand how the answer was obtained; integrals can be evaluated afterwards using `value`.

\[ y(x) = C1 \cdot \sin(x) \]

This particular linear ODE is also separable,

\[ \int \frac{\cos(x)}{\sin(x)} \, dx - \frac{1}{\sin(x)} \, y(x) + C1 = 0 \]

As a general alternative, one can look for an integrating factor (see `DEtools, intfactor`).
> mu := intfactor(ode_L);

\[
\mu := \frac{1}{\sin(x)^2}
\]

Integrating factors turn ODEs exact; to indicate to dsolve to use the scheme for exact ODEs:

> dsolve( mu*ode_L, [exact], useInt);

\[
y(x) = -\frac{-C1}{\frac{1}{2} \tan \left(\frac{1}{2} x\right) + \frac{1}{2} \frac{1}{\tan \left(\frac{1}{2} x\right)} + \frac{x}{2} - 1 + x^2 - 2 x^2 y(x) + 2 x^4}{(x^2 - y(x))(x + 1)}
\]

The possibility of indicating the solving method, plus the 'useInt' and 'implicit' optional arguments provide the control of the relevant steps of the solving process.

An ODE of Abel type.

> ode_A := diff(y(x),x) =

\[
x*(-x+1+x^2-2*x^2*y(x)+2*x^4)/((x^2-y(x))*(x+1));
\]

A related integrating factor

> mu := intfactor(ode_A);

\[
\mu := \frac{x^2 - y(x)}{2 x^2 + 1 - 2 y(x)}
\]

The solution for the related exact ODE

> ans_A1 := dsolve( mu*ode_A, [exact]);

\[
ans_A1 := y(x) = \frac{1}{2} \text{LambertW} \left( -\frac{e^{\left(\frac{4}{3} x^3\right)} (e^x)^4 \left( e^{\left(\frac{4}{3} x^3\right)} - 1 \right)}{(e^{x^2})^4 (x + 1)^4} \right) + x^2 + \frac{1}{2} + C1
\]

Explicit or implicit answers can be tested, in principle, using odetest.

> odetest(ans_A1, ode_A);
A second order linear homogeneous ODE

\[ \text{ode4} := \frac{\partial^2}{\partial x^2} y(x) = x^n (n-1 + x^n - \cos(x) \cdot x) / x^2 + \cos(x) \frac{\partial}{\partial x} y(x); \]

\[ \text{ode4} := \frac{\partial^2}{\partial x^2} y(x) = x^n \left( n - 1 + x^n - \cos(x) \cdot x \right) / x^2 \cdot y(x) + \cos(x) \left( \frac{\partial}{\partial x} y(x) \right) \]

\[ \text{dsolve(ode4);} \]

\[ y(x) = \left\{ \int \frac{e^{\sin(x)}}{2} \, dx \_C1 + \_C2 \right\} \left( e^{x^n} \right) \]

\[ \text{ode41 := diff(y(x),x,x) = } \]

\[ \left( y(x) \cdot x + y(x) \cdot \ln(x) \cdot x^2 + \text{diff}(y(x),x) \cdot \ln(x) \cdot x^2 \right) / \ln(x) / x^2 \cdot \exp(x) - y(x) / \ln(x) / x^2; \]

\[ \text{ode41 := diff(y(x),x,x) = } \]

\[ \frac{\partial^2}{\partial x^2} y(x) = \left( y(x) \cdot x + y(x) \cdot \ln(x) \cdot x^2 + \left( \frac{\partial}{\partial x} y(x) \right) \ln(x) \cdot x^2 \right) e^x - \frac{y(x)}{\ln(x) \cdot x^2} \]

\[ \text{dsolve(ode41);} \]

\[ y(x) = -\left\{ \int \frac{e^{-x}}{\ln(x)^2} \, dx \_C1 - \_C2 \right\} \ln(x) \]

For this linear ODE it is possible to find two integrating factors at once

\[ \text{intfactor(ode41);} \]

\[ \ln(x), \int \frac{e^{-x}}{\ln(x)^2} \, dx \ln(x) \]

These integrating factors lead to

\[ \text{dsolve(ode41);} \]

\[ y(x) = -\left\{ \int \frac{e^{-x}}{\ln(x)^2} \, dx \_C1 - \_C2 \right\} \ln(x) \]

\[ e^{-x} \]
A non-linear second order example with an arbitrary function (F) of its arguments

\[ ode5 := \frac{\partial^2}{\partial x^2} y(x) = \frac{F\left(\frac{\partial}{\partial x} y(x) \right) \left(r + s \cdot x\right)}{u + w \cdot y(x)} \cdot \frac{u + w \cdot y(x)}{\left(r + s \cdot x\right)^2}; \]

\[ ode5 := \frac{\partial^2}{\partial x^2} y(x) = \frac{\left(\frac{\partial}{\partial x} y(x) \right) \left(r + s \cdot x\right)}{u + w \cdot y(x)} \cdot \frac{u + w \cdot y(x)}{\left(r + s \cdot x\right)^2}; \]

\[ odeadvisor(ode5) ; \]

\[ [[\_2nd\_order, \_with\_linear\_symmetries]] \]

\[ ans5 := dsolve(ode5) ; \]

\[ ans5 := y(x) = e^{\frac{-\ln(-r - s \cdot x)}{s} w \cdot \left[\frac{\omega}{-a s + a^2} \left(1 \cdot \frac{1}{d_a - b + _C1} d_a - d_b + _C2 \right) - u\right]} \]

\[ odetest. \]

\[ odetest(ans5, ode5) ; \]

\[ 0 \]

A non-linear high order example solved using symmetry methods:

\[ ode6 := \frac{\partial^2}{\partial x^2} y(x) \left(\frac{\partial}{\partial x} y(x) \right) y(x) x^6 - 2 \cdot \frac{\partial}{\partial x} y(x) \left(3 \cdot x^6 + 2 \cdot \frac{\partial}{\partial x} y(x) \right) 2 \cdot y(x) \left(x^5 + y(x)^5\right); \]

\[ ode6 := \left(\frac{\partial}{\partial x} y(x) \right)^3 y(x) x^6 - 2 \left(\frac{\partial}{\partial x} y(x) \right) x^6 + 2 \left(\frac{\partial}{\partial x} y(x) \right) x^5 + y(x)^5; \]

\[ dsolve(ode6) ; \]

\[ y(x) = 0, y(x) = 3 \cdot \frac{x^3 \left(3/2\right)}{(2 x - 2 \cdot _C1 x^2)^{1/2}} + 3 \cdot _C2 x^3; \]
A reduction of order from 4 to 1 (\texttt{dsolve,ODESolStruc}), using Lie's symmetry methods (\texttt{dsolve,Lie}), for example 14 (non-linear high order) from Kamke's book:

\begin{align*}
  y(x) &= 3 \frac{x^3}{-(2x - 2C1x^2)^{(3/2)}} + 3C2x^3 \\
\end{align*}

\begin{align*}
  \text{ode14} := & \frac{\partial}{\partial x} y(x) \left( \frac{\partial^4}{\partial x^4} y(x) \right) - \left( \frac{\partial^2}{\partial x^2} y(x) \right) \left( \frac{\partial^3}{\partial x^3} y(x) \right) + \left( \frac{\partial}{\partial x} y(x) \right)^3 \left( \frac{\partial^3}{\partial x^3} y(x) \right) \\
  \text{odeadvisor}(\text{ode14}); \\
  \text{ans14} := & \text{dsolve}(\text{ode14}, \text{Lie}); \\
  \text{ans14} := & y(x) = C1, \quad y(x) = \left\{ \left( \frac{h(-g)}{\int_{h(-g)} d_g + C2} \right)^{-1} \right\} e^{\int_{h(-g)} d_g + C3} & \text{&where} \\
  \left\{ \frac{\partial}{\partial g} h(-g) = (12_g + 3) h(-g)^3 + \left( \frac{10_g + 1}{g} \right) h(-g)^2 + \frac{h(-g)}{g} \right\}, \\
  \left\{ h(-g) = \left( \frac{\partial}{\partial x} y(x) \right)^3, \quad g = \frac{\partial^2}{\partial x^2} y(x) \right\}, \\
  \left\{ \frac{\partial^3}{\partial x^3} y(x) \left( \frac{\partial}{\partial x} y(x) \right) - 3 \left( \frac{\partial^2}{\partial x^2} y(x) \right)^3 \right\}, \\
  \left\{ \frac{\partial^2}{\partial x^2} y(x) \right\}, \\
\end{align*}
\[
\begin{align*}
x &= \int \frac{\_h(\_g)}{\_g \left( \int \_h(\_g) \_g \, d\_g + \_C2 \right)^2} d\_g + \_C1, \\
y(x) &= \int \frac{\_h(\_g)}{\_g \left( \int \_h(\_g) \_g \, d\_g + \_C2 \right)^2} d\_g + \_C3
\end{align*}
\]

In the above, the right hand side is a structure of two operands, where the first one is the answer in terms of new variables, and the second one is a list with three sets, respectively containing the reduced ODE, the transformation of variables used to solve the problem, and the inverse transformation (see ?dsolve,ODESolStruc).

These reductions of order can also be tested using odetest

\[
> \text{map(odetest, [ans14], ode14);}
\]

\[
[0, 0]
\]

See Also:

odeadvisor, dsolve,algorithms, dsolve,education, dsolve,ICs, dsolve,inttrans, dsolve,formal_series, dsolve,Lie, dsolve,linear, dsolve,numeric; dsolve,piecewise, dsolve,series, dsolve,system, odeadvisor,types, and the Maple packages for differential equations DEtools, PDEtools