Solve a System of Differential Equations

Solve a system of several ordinary differential equations in several variables by using the `dsolve` function, with or without initial conditions. To solve a single differential equation, see Solve Differential Equation.

- Solve System of Differential Equations
- Solve Differential Equations in Matrix Form

Solve System of Differential Equations

Solve this system of linear first-order differential equations.

$$\frac{du}{dt} = 3u + 4v,$$

$$\frac{dv}{dt} = -4u + 3v.$$

First, represent u and v by using `syms` to create the symbolic functions `u(t)` and `v(t)`.

```
syms u(t) v(t)
```

Define the equations using `==` and represent differentiation using the `diff` function.

```
ode1 = diff(u) == 3*u + 4*v;
ode2 = diff(v) == -4*u + 3*v;
odes = [ode1; ode2]
```

```
odes(t) =
 diff(u(t), t) == 3*u(t) + 4*v(t)
 diff(v(t), t) == 3*v(t) - 4*u(t)
```

Solve the system using the `dsolve` function which returns the solutions as elements of a structure.

```
S = dsolve(odes)
```

```
S =
  struct with fields:

    v: [1×1 sym]
    u: [1×1 sym]
```

If `dsolve` cannot solve your equation, then try solving the equation numerically. See Solve Differential Equation Numerically.

To access `u(t)` and `v(t)`, index into the structure `S`.

```
uSol(t) = S.u
vSol(t) = S.v
```

```
uSol(t) =
C2*cos(4*t)*exp(3*t) + C1*sin(4*t)*exp(3*t)
vSol(t) =
C1*cos(4*t)*exp(3*t) - C2*sin(4*t)*exp(3*t)
```

Alternatively, store `u(t)` and `v(t)` directly by providing multiple output arguments.

```
[uSol(t), vSol(t)] = dsolve(odes)
```

```
uSol(t) =
C2*cos(4*t)*exp(3*t) + C1*sin(4*t)*exp(3*t)
vSol(t) =
C1*cos(4*t)*exp(3*t) - C2*sin(4*t)*exp(3*t)
```

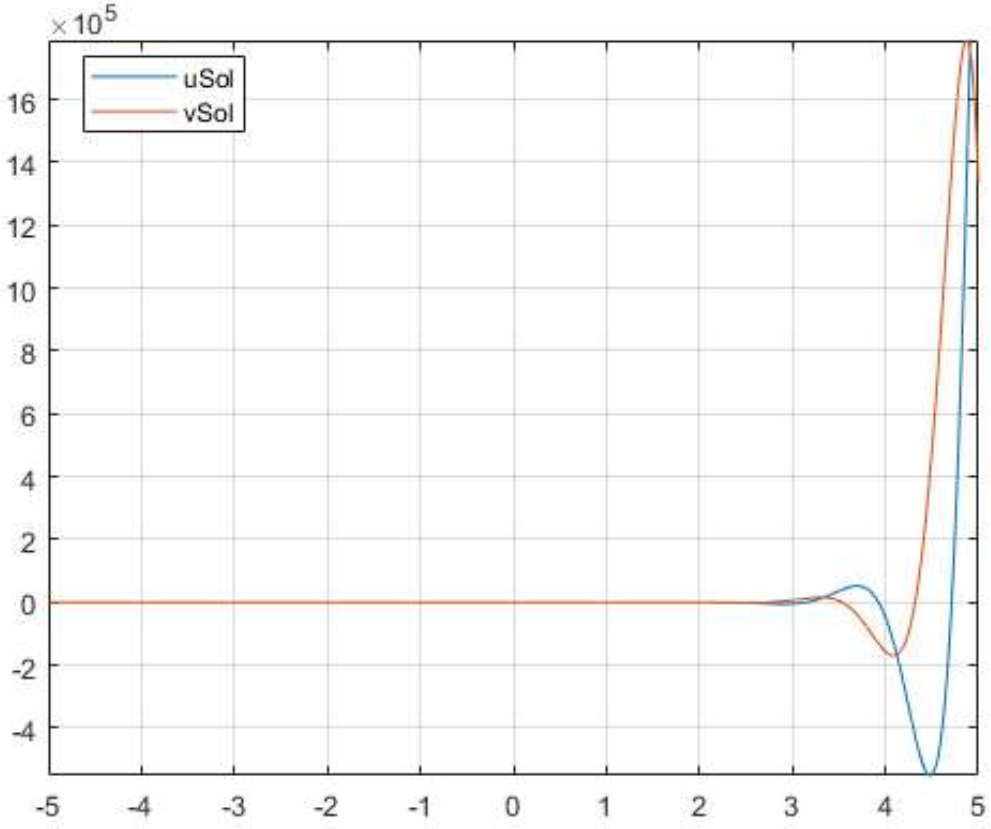
The constants `C1` and `C2` appear because no conditions are specified. Solve the system with the initial conditions `u(0) == 0` and `v(0) == 0`. The `dsolve` function finds values for the constants that satisfy these conditions.

```
cond1 = u(0) == 0;
cond2 = v(0) == 1;
conds = [cond1; cond2];
[uSol(t), vSol(t)] = dsolve(odes,conds)
```

```
uSol(t) =
sin(4*t)*exp(3*t)
vSol(t) =
cos(4*t)*exp(3*t)
```

Visualize the solution using `fplot`. Before R2016a, use `ezplot` instead.

```
fplot(uSol)
hold on
fplot(vSol)
grid on
legend('uSol','vSol','Location','best')
```



Solve Differential Equations in Matrix Form

Solve differential equations in matrix form by using `dsolve`.

Consider this system of differential equations.

$$\frac{dx}{dt} = x + 2y + 1,$$
$$\frac{dy}{dt} = -x + y + t.$$

The matrix form of the system is

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 2 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 1 \\ t \end{bmatrix}.$$

Let

$$Y = \begin{bmatrix} x \\ y \end{bmatrix}, A = \begin{bmatrix} 1 & 2 \\ -1 & 1 \end{bmatrix}, B = \begin{bmatrix} 1 \\ t \end{bmatrix}.$$

The system is now $Y' = AY + B$.

Define these matrices and the matrix equation.

```
syms x(t) y(t)
A = [1 2; -1 1];
B = [1; t];
Y = [x; y];
odes = diff(Y) == A*Y + B
```

```
odes(t) =
  diff(x(t), t) == x(t) + 2*y(t) + 1
   diff(y(t), t) == t - x(t) + y(t)
```

Solve the matrix equation using `dsolve`. Simplify the solution by using the `simplify` function.

```
[xSol(t), ySol(t)] = dsolve(odes);
xSol(t) = simplify(xSol(t))
ySol(t) = simplify(ySol(t))
```

```
xSol(t) =
(2*t)/3 + 2^(1/2)*C2*exp(t)*cos(2^(1/2)*t) + 2^(1/2)*C1*exp(t)*sin(2^(1/2)*t) + 1/9
ySol(t) =
C1*exp(t)*cos(2^(1/2)*t) - t/3 - C2*exp(t)*sin(2^(1/2)*t) - 2/9
```

The constants `C1` and `C2` appear because no conditions are specified.

Solve the system with the initial conditions $u(0) = 2$ and $v(0) = -1$. When specifying equations in matrix form, you must specify initial conditions in matrix form too. `dsolve` finds values for the constants that satisfy these conditions.

```
C = Y(0) == [2; -1];
[xSol(t), ySol(t)] = dsolve(odes,C)
```

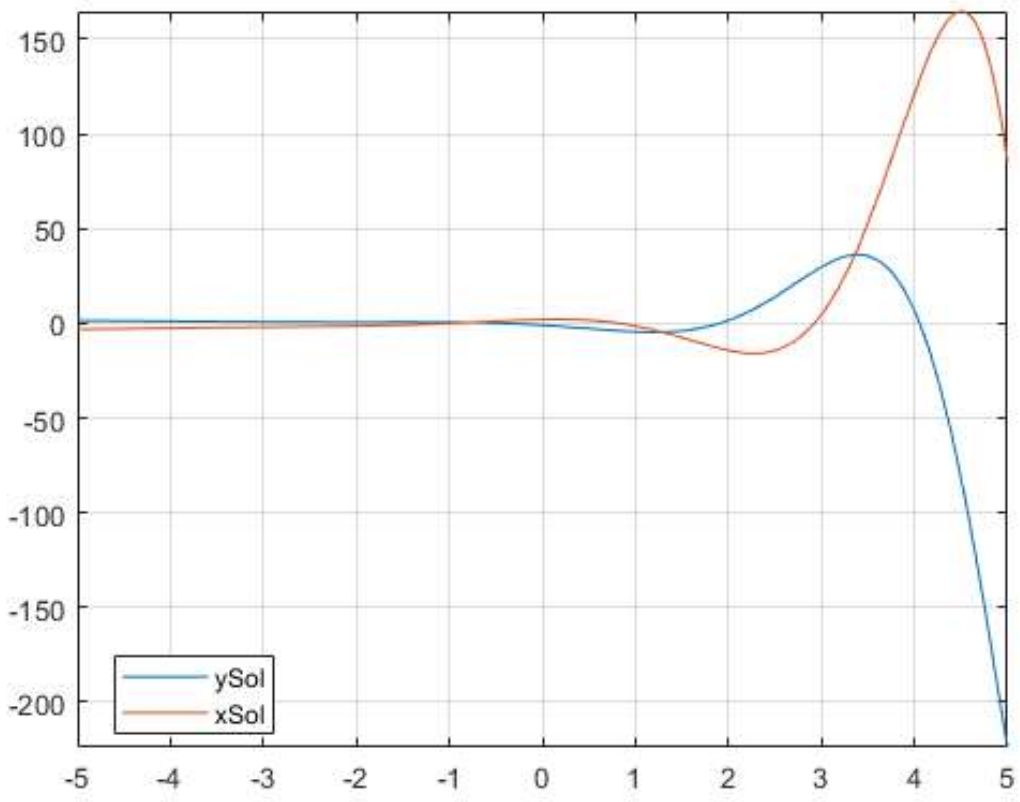
```
xSol(t) =
(2*t)/3 + (17*exp(t)*cos(2^(1/2)*t))/9 - (7*2^(1/2)*exp(t)*sin(2^(1/2)*t))/9 + 1/9
ySol(t) =
- t/3 - (7*exp(t)*cos(2^(1/2)*t))/9 - (17*2^(1/2)*exp(t)*sin(2^(1/2)*t))/18 - 2/9
```

Visualize the solution using `fplot`. Before R2016a, use `ezplot` instead.

```

```

```
clf
fplot(ySol)
hold on
fplot(xSol)
grid on
legend('ySol','xSol','Location','best')
```



See Also

Solve Differential Equation