Lesson 08
Methods of integrating ordinary differential equations

Our present views of the fundamental laws of nature are often stated in the form of differential equations. For example, Newton's law is a second-order differential equation.
F = mx''(t)

As another example, consider a colony of bacteria growing in a medium which furnishes an adequate food supply. When the colony is studied, it is natural to suppose that the rate of growth is proportional to the number y(t) present at time t; that is
y'(t) = k y(t)

Nomenclature:
*) Homogeneous first-order linear ordinary differential equation
y'(t) - a y(t) = 0

*) Inhomogeneous second-order linear ordinary differential equation
y''(t) - t y(t) = b t

*) Second-order nonlinear ordinary differential equation
u''(t) + g sin(u) = 0

*) Homogeneous first-order linear partial differential equation
$$\frac{\partial u}{\partial t} + \mathrm{t}\ \frac{\partial u}{\partial x} = 0$$

<u>Using commands from *Mathematica:*</u>
DSolve
or NDSolve

Examples:

*) y'(x) = 3 y(x)
DSolve[y'[x] == 3 y[x], y[x], x]
or simply
DSolve[y'[x] == 3 y[x], y, x]
(first argument = the differential equation, second = the function, third = the independent variable)
Solution:
$\{\{y[x] \to e^{3x} C[1]\}\}$
The answer is given in the form of a replacement rule.
Arbitrary constants C[1], etc indicates that the solution contains infinitely many solutions.

If we want a specific solution, we can <u>specify the constant</u>.

For instance, if we want the solution for which y(2) = Pi, we can write
Clear[y]
DSolve[{ y'[x] == 3 y[x], y[2] == Pi }, y[x], x]

To make a plot:
Clear[solut, y];
solut = DSolve[{ y'[x] == 3 y[x], y[2] == Pi }, y[x], x]
solut[[1]]
solut[[1, 1]]
solut[[1, 1, 2]]
Clear[f]
f = solut[[1, 1, 2]];
Plot[f, {x, 0, 1}]
CAREFUL if you use simply y, instead of y[x] in the second argument:
Clear[solut, y];
solut = DSolve[{ y'[x] == 3 y[x], y[2] == Pi }, y, x]
solut[[1]]
solut[[1, 1]]
solut[[1, 1, 2]]
solut[[1, 1, 2, 2]]

Confirming that y(2) = Pi and that the derivative of y[x] is really 3 y[x]
f /. x -> 2
D[f, x] == 3 f

*) If we want various solutions with different values of C[1].
Let's check it for a different equation y'(x) = (x^2 - y(x))/(x + y(x))
Clear[solut]
solut = DSolve[y'[x] == (x^2 - y[x])/(x + y[x]), y[x], x]
$\left\{\left\{y[x] \to -x - \sqrt{x^2 + \frac{2x^3}{3} + C[1]}\right\}, \left\{y[x] \to -x + \sqrt{x^2 + \frac{2x^3}{3} + C[1]}\right\}\right\}$
There are two solutions. We will plot both for C[1] from 0 to 3.
Clear[tab1, tab2, tab]
tab1 = Table[solut[[1, 1, 2]]/. C[1] -> k, {k,0,3}]
Plot[tab1, {x, -4, 4}]
tab2= Table[solut[[2, 1, 2]]/. C[1] -> k, {k,0,3}]
Plot[tab2, {x, -4, 4}]
tab = Flatten[{tab1, tab2}, 1]
Plot[tab, {x, -4, 4}]

*) Higher order differential equations
y''(x) + y(x) = 0
DSolve[y''[x] + y[x] == 0, y[x], x]
{{y[x] → C[1] Cos[x] + C[2] Sin[x]}}
Specifying the initial conditions determines C[1] and C[2]
DSolve[{ y''[x] + y[x] == 0, y'[0] == 3, y[0] == 2}, y[x], x]
{{y[x] → 2 Cos[x] + 3 Sin[x]}}

*) System of equations
x'(t) = x(t) - 10 y(t)
y'(t) = 15 x(t) + y(t)
Clear[solut, x, y]
solut = DSolve[{ x'[t] == x[t] - 10 y[t], y'[t] == 15 x[t] + y[t] }, {x[t], y[t]}, t]
Clear[f, g]

```
f = solut[ [ 1, 1, 2] ]
g = solut[ [1, 2, 2] ]
```

```
Plot[{f/.{C[1]→3,C[2]→2},g/.{C[1]→3,C[2]→2}}, {t, 0, 4}, PlotRange → All]
```

```
ParametricPlot[{f /.{ C[1] → 3, C[2] → 2}, g/.{ C[1] → 3, C[2] → 2}}, {t, 0, 4}]
```

```
Manipulate[ParametricPlot[{f /.{ C[1] → h1, C[2] → h2}, g/.{ C[1] → h1, C[2] → h2}}, {t, 0, 4}, PlotRange →
{{-200, 200},{-200, 200}}], {h1,1,3}, {h2, 1, 3}]
```

*) Exercise

Suppose an object is falling near the surface of the Earth and that its height off the ground at time t is x(t). If there is no air resistance, and assuming that g=9.8m/s^2, then x satisfies the following differential equation x''(t) = - g (this equation assumes a reference frame which is positive going up and where x=0 at the ground)

Assume that the object is dropped from a height of 1000 m. Use DSolve to find x(t). How long will it take for the object to reach the ground?

```
Clear[solu]
solu=DSolve[ { x''[ t ] == -9.8, x[ 0 ] == 1000, x'[0] == 0 }, x[ t ], t ]
Clear[f]
f = solu[ [ 1,1,2 ] ]
Solve[ f ==0, t]
```

*) <u>Numerical Solutions</u>

Very often we can only find numerical approximations to the solutions

$\frac{dy}{dx}=x^2+\sqrt{y}$ with initial condition y(0) =1

Nothing come out from

```
DSolve[{ y'[ x ] == x^2 + Sqrt[ y[ x ] ], y[ 0 ] ==1 }, y[ x ], x]
```

But we get an answer with NDSolve

NOTICE: we need to give a range of values of x

NOTICE: use y instead of y[x]

```
Clear[ temp]
temp = NDSolve[{ y'[ x ] == x^2 + Sqrt[ y[ x ] ], y[ 0 ] ==1 }, y, {x, 0, 1 }]
```

To get individual values of the function

```
temp[ [ 1, 1, 2] ] [ 0.5 ]
Clear[solution]
solution = temp[ [ 1, 1, 2] ]
```

and with "solution" we can extract specific points

```
solution[ 0.5 ]
```

and even make a plot with a list of values

```
Clear[ tab]
tab = Table[ { 0.1 k, solution[ 0.1 k] }, {k, 0, 10} ]
ListPlot[ tab]
```

or even simpler

```
Plot[ solution[ x ], {x, 0, 1} ]
```

To guarantee that the plot starts at (0,0)

```
Plot[ solution[ x ], {x, 0, 1} , AxesOrigin -> {0,0}]
```