Lesson 09
Methods of integrating ordinary differential equations

Suppose we do not have the commands from *Mathematica,* what can we do?

http://numericalmethods.eng.usf.edu/

Euler's Method

Euler's method solves a first order differential equation of the type:

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

We need to have the initial condition xo and yo.
From there, we choose a step h and for a list of values of x:
xo+h, xo+2h, xo+3h, ...
we try to approximate the values of y(xo+h), y(xo+2h), y(xo+3h), ...
Since we know the slope at xo, which is just just f(xo,yo), we use the expression for the tangent to approximate y(xo+h)

tangent = $f(xo,yo) = \frac{y(xo+h)-y(xo)}{(xo+h-xo)} \Longrightarrow y(xo+h) = y(xo) + f(xo,yo)\, h$

which is equivalent to Taylor's expansion to first order:

$y(xo+h) \simeq y(xo) + y'(xo)\, h$

We continue with this procedure for successive points.

```
SetDirectory["YESHIVA"];
SetDirectory["ANGEL"];
SetDirectory["Presentations_Computational"];
```

```
Import["Euler_Figure.jpg"]
```

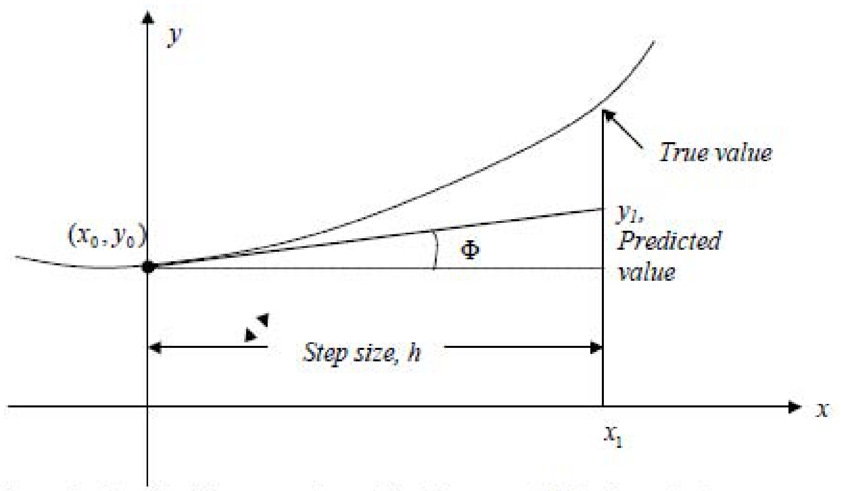


Figure 1 Graphical interpretation of the first step of Euler's method.

Watch the movies below and then try to find
y(0), y(1), y(2), y(3), y(4), y(5) with high accuracy, given that
y'(x) + 0.4 y = 3 exp(-x)   with   y(0)=5

http://numericalmethods.eng.usf.edu/videos/youtube/08   ode/euler/euler_08ode_derivation_youtube.html

Example:
http://numericalmethods.eng.usf.edu/videos/youtube/08ode/euler/euler_08ode_example_youtube.html

When writing your little codes, check each line before writing the next one.

```
Clear[x, y, dy, xo, yo, h];
dy = -0.4 y + 3. Exp[-x];
xo = 0.;
yo = 5.;
h = 1.;

Do[
 tanY = dy /. {x → xo, y → yo};
 xo = xo + h;
 yo = yo + tanY h;
 Print[{xo, yo}];
 , {k, 1, 5}]
```

*) Let us make the little code above more general, so that we change only the value of h and get all the results up to y(5).
Let us also store all the pair results we get, from (xo,y(xo)) to (5, y(5)).

```
Clear[dy, xo, yo, h];
dy = -0.4 y + 3. Exp[-x];
xo = 0.;
yo = 5.;
h = 1.;

Clear[XX, YY];
XX[0] = xo;
YY[0] = yo;

Clear[final, howmany];
final = 5.;
howmany = (final - xo) / h;

Do[
 tanY = dy /. {x → xo, y → yo};
 xo = xo + h;
 yo = yo + tanY h;
 Print[{xo, yo}];
 XX[k] = xo;
 YY[k] = yo;
 , {k, 1, howmany}]
```

*) Now get the exact solutions using DSolve and compare them with the results above. Put them all in the form of a Table, where the first column gives the values of x, the second the values of your numerical approximations for y, and the third the exact results. Write the heading of this table.

```
Clear[solu];
solu = DSolve[{y'[x] == -0.4 y[x] + 3. Exp[-x], y[0] == 5.}, y[x], x];
exactSol = solu[[1, 1, 2]]
Do[
  exactY[k] = exactSol /. x → h k;
  Print[{h k, exactY[k]}];
  , {k, 0, howmany}];

Clear[form];
form = Table[{XX[k], YY[k], exactY[k]}, {k, 0, howmany}];

TableForm[ form, TableHeadings -> {None,  { "x", "Euler", "Exact"} } ]
(* None above is used because there is no heading associated with the lines,
only with the columns *)
```

*) Now put both codes for approximate and exact solutions together in a single code. Use a smaller value of h, but print only the values for x=0, 1, 2, 3, 4, 5, and do the same when getting the table form. Reduce the value of h until both results look very similar.

- **h = 0.1**

```
Clear[dy, xo, yo, h];
dy = -0.4 y + 3. Exp[-x];
xo = 0.;
yo = 5.;
h = 0.1;

Clear[XX, YY];
XX[0] = xo;
YY[0] = yo;

Clear[final, howmany];
final = 5.;
howmany = (final - xo) / h;

Do[
  tanY = dy /. {x → xo, y → yo};
  xo = xo + h;
  yo = yo + tanY h;
  If[Mod[k, Floor[1 / h]] == 0, Print[{xo, yo}]];

  XX[k] = xo;
  YY[k] = yo;
  , {k, 1, howmany}];

Clear[solu];
solu = DSolve[{y'[x] == -0.4 y[x] + 3. Exp[-x], y[0] == 5.}, y[x], x];
exactSol = solu[[1, 1, 2]];
Do[
  exactY[k] = exactSol /. x → h k;
  , {k, 0, howmany}];

Clear[form];
form = Table[{XX[k], YY[k], exactY[k]}, {k, 0, howmany, Floor[1 / h]}];

TableForm[ form, TableHeadings -> {None, { "x", "Euler", "Exact"} } ]
```

```
{1., 4.93913}

{2., 3.87781}

{3., 2.79666}

{4., 1.93971}

{5., 1.31916}
```

x	Euler	Exact
0.	5.	5.
1.	4.93913	4.8638
2.	3.87781	3.81661
3.	2.79666	2.76301
4.	1.93971	1.92739
5.	1.31916	1.31966

- **h = 0.0001**

```
Clear[dy, xo, yo, h];
dy = -0.4 y + 3. Exp[-x];
xo = 0.;
yo = 5.;
h = 0.0001;

Clear[XX, YY];
XX[0] = xo;
YY[0] = yo;

Clear[final, howmany];
final = 5.;
howmany = (final - xo) / h;

Do[
  tanY = dy /. {x → xo, y → yo};
  xo = xo + h;
  yo = yo + tanY h;
  If[Mod[k, Floor[1 / h]] == 0, Print[{xo, yo}]];

  XX[k] = xo;
  YY[k] = yo;
  , {k, 1, howmany}];

Clear[solu];
solu = DSolve[{y'[x] == -0.4 y[x] + 3. Exp[-x], y[0] == 5.}, y[x], x];
exactSol = solu[[1, 1, 2]];
Do[
  exactY[k] = exactSol /. x → h k;
  , {k, 0, howmany}];

Clear[form];
form = Table[{XX[k], YY[k], exactY[k]}, {k, 0, howmany, Floor[1 / h]}];

TableForm[ form, TableHeadings -> {None, { "x", "Euler", "Exact"} } ]
```

```
{1., 4.86388}

{2., 3.81667}

{3., 2.76304}

{4., 1.9274}

{5., 1.31966}
```

```
x     Euler      Exact
0.    5.         5.
1.    4.86388    4.8638
2.    3.81667    3.81661
3.    2.76304    2.76301
4.    1.9274     1.92739
5.    1.31966    1.31966
```

Taylor Method

We need one point (an initial condition) to start the method:

yo=y(xo)

The idea now is to find other points close to it, by using an increment h, so that the next x and the next y are

xo+h and y(xo+h).

To find an approximation to y(xo+h), we use Taylor expansion:

$$y(xo+h) \simeq y(xo) + y'(xo)\,h + \frac{y''(xo)}{2!}\,h^2 + \ldots + \frac{y^{(n)}(xo)}{n!}\,h^n$$

$$y(xo+h) = y(xo) + f(xo,yo).h + \frac{f'(xo,yo)}{2!}\,h^2 + \ldots + \frac{f^{(n-1)}(xo,yo)}{n!}\,h^n$$

<u>Example:</u>

Apply the local Taylor series method to obtain a solution of

$y' = x\, y^{1/3}$,

given the initial condition y(1) =1. Use the expansion to 4th order and find y(x) for x=1 to x=5. Use step size h=0.1.

```
Clear[y1, y2, y3, y4];
y1 = x y[x]^(1 / 3);
y2 = D[y1, x] /. y'[x] → y1;
y3 = D[y2, x] /. y'[x] → y1;
y4 = D[y3, x] /. y'[x] → y1;

Clear[xo, yo, h];
xo = 1.;
yo = 1.;
h = 0.1;
Print["Initial condition: ", {xo, yo}];

Clear[XX, YY];
XX[0] = xo;
YY[0] = yo;

Clear[final, howmany];
final = 5.;
howmany = (final - xo) / h;

Clear[TaylorY, Ry1, Ry2, Ry3, Ry4];
Do[

 Ry1 = y1 /. {x → xo, y[x] → yo};
 Ry2 = y2 /. {x → xo, y[x] → yo};
 Ry3 = y3 /. {x → xo, y[x] → yo};
 Ry4 = y4 /. {x → xo, y[x] → yo};

 TaylorY = yo + h Ry1 + (1 / 2) h^2 Ry2 + (1 / 6) h^3 Ry3 + (1 / 24) h^4 Ry4;

 xo = xo + h;
 yo = TaylorY;

 If[ Mod[k, Floor[1 / h]] == 0, Print[{xo, yo}]];
 XX[k] = xo;
 YY[k] = yo;

 , {k, 1, howmany}]
```

```
Initial condition: {1., 1.}

{2., 2.82843}

{3., 7.02113}

{4., 14.6969}

{5., 27.}
```

*) Compare your results with the result from *Mathematica*.

```
Clear[solu, y, x, exactSol, exactTab];
solu = DSolve[{y'[x] == x y[x]^(1/3), y[1] == 1}, y[x], x];
exactSol = Simplify[solu[[1, 1, 2]]]
Do[
  exactTab[k] = exactSol /. x → (1 + k h);
  , {k, 0, howmany}];

Clear[form];
form = Table[{XX[k], YY[k], exactTab[k]}, {k, 0, howmany}];

TableForm[ form, TableHeadings -> {None,  { "x", "Taylor", "Exact"} } ]
(* None above is used because there is no heading associated with the lines,
only with the columns *)
```

$$\frac{\left(2+x^2\right)^{3/2}}{3\sqrt{3}}$$

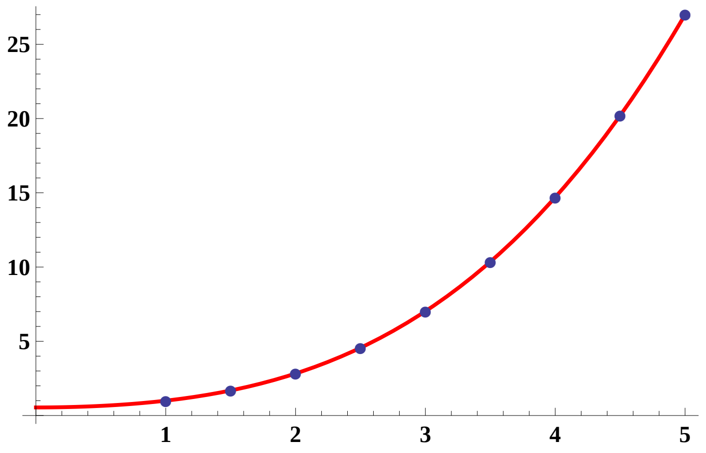
```
x      Taylor     Exact
1.     1.         1.
1.1    1.10682    1.10682
1.2    1.22788    1.22788
1.3    1.36414    1.36414
1.4    1.51656    1.51656
1.5    1.68617    1.68617
1.6    1.87398    1.87398
1.7    2.08105    2.08104
1.8    2.30842    2.30842
1.9    2.55719    2.55719
2.     2.82843    2.82843
2.1    3.12324    3.12324
2.2    3.44272    3.44272
2.3    3.788      3.788
2.4    4.16017    4.16017
2.5    4.56036    4.56036
2.6    4.9897     4.9897
2.7    5.44931    5.44931
2.8    5.94033    5.94033
2.9    6.46389    6.46389
3.     7.02113    7.02113
3.1    7.61319    7.61319
3.2    8.2412     8.2412
3.3    8.9063     8.9063
3.4    9.60965    9.60965
3.5    10.3524    10.3524
3.6    11.1356    11.1356
3.7    11.9606    11.9606
3.8    12.8284    12.8284
3.9    13.7401    13.7401
4.     14.6969    14.6969
4.1    15.7001    15.7001
4.2    16.7506    16.7506
4.3    17.8497    17.8497
4.4    18.9985    18.9985
4.5    20.1982    20.1982
4.6    21.4499    21.4499
4.7    22.7548    22.7548
4.8    24.114     24.114
4.9    25.5287    25.5287
5.     27.        27.
```

*) Use a larger value of h, for example h=0.5, and compare your results by writing a table and also a plot. Include both approximate and exact method in a single little code.

```
Clear[TaylorY, Ry1, Ry2, Ry3, Ry4];

y1 = x y[x]^(1/3);
y2 = D[y1, x] /. y'[x] → y1;
y3 = D[y2, x] /. y'[x] → y1;
y4 = D[y3, x] /. y'[x] → y1;

Clear[XX, YY, xo, yo, h, TaylorY];
xo = 1.;
```

```
yo = 1.;
h = 0.5;
(* Print["Initial condition: ",{xo,yo}]; *)

XX[0] = xo;
YY[0] = yo;

Clear[howmany];
howmany = (5 - xo) / h;

Clear[TaylorY, Ry1, Ry2, Ry3, Ry4];
Do[

  Ry1 = y1 /. {x → xo, y[x] → yo};
  Ry2 = y2 /. {x → xo, y[x] → yo};
  Ry3 = y3 /. {x → xo, y[x] → yo};
  Ry4 = y4 /. {x → xo, y[x] → yo};

  TaylorY = yo + h Ry1 + (1 / 2) h^2 Ry2 + (1 / 6) h^3 Ry3 + (1 / 24) h^4 Ry4;

  xo = xo + h;
  yo = TaylorY;

  XX[k] = xo;
  YY[k] = yo;

  , {k, 1, howmany}];


Clear[solu, y, x, exactSol, exactTab];
solu = DSolve[{y'[x] == x y[x]^(1 / 3), y[1] == 1}, y[x], x];
exactSol = Simplify[solu[[1, 1, 2]]];
Do[
  exactTab[k] = exactSol /. x → (1 + k h);
  , {k, 0, howmany}];

Clear[form];
form = Table[{XX[k], YY[k], exactTab[k]}, {k, 0, howmany}];

TableForm[ form, TableHeadings -> {None,  { "x", "Taylor", "Exact"} } ]

Clear[lTaylor, pTaylor, pExact];
lTaylor = Table[{XX[k], YY[k]}, {k, 0, howmany}];
pTaylor = ListPlot[lTaylor,
   PlotMarkers → Automatic, LabelStyle → Directive[Black, Bold, Medium]];
pExact = Plot[exactSol, {x, 0, 5}, PlotStyle → {Red, Thick},
   LabelStyle → Directive[Black, Bold, Medium]];
Show[pExact, pTaylor]
```

x	Taylor	Exact
1.	1.	1.
1.5	1.68634	1.68617
2.	2.82871	2.82843
2.5	4.56072	4.56036
3.	7.02156	7.02113
3.5	10.3529	10.3524
4.	14.6975	14.6969
4.5	20.1988	20.1982
5.	27.0007	27.



Runge-Kutta 4th order Method

http://numericalmethods.eng.usf.edu/videos/youtube/08ode/rungekutta4th/rungekutta4th_08ode_formula.html

http://www.ee.nthu.edu.tw/bschen/files/c16-1.pdf
http://www.youtube.com/watch?v=AT7Olelic8U
http://www.youtube.com/watch?v=gzgghqto1Ws&feature=relmfu

Write the Taylor expansion to 4th order:

$$y(xo+h) \simeq y(xo) + y'(xo)\,h + \frac{y''(xo)}{2!}\,h^2 + \frac{y'''(xo)}{3!}\,h^3 + \frac{y''''(xo)}{4!}\,h^4$$

$$y(xo+h) = y(xo) + f(xo,yo).h + \frac{f'(xo,yo)}{2!}\,h^2 + \frac{f''(xo,yo)}{3!}\,h^3 + \frac{f'''(xo,yo)}{4!}\,h^4$$

Runge and Kutta equate the above equation to the equation below

y(xo+h) = y(xo) +(a1 k1 + a2 k2 + a3 k3 + a4 k4). h

and found expressions for the constants, so that we would not need to compute the higher order derivatives of y to find the approximate solutions. The expressions they found were

$$y_{i+1} = y_i + \frac{1}{6}\,(k1 + 2\,k2 + 2\,k3 + k4).\,h$$

$k1 = f(x_i, y_i)$

$k2 = f(x_i + h/2,\ y_i + k1\ h/2)$

$$k3 = f(x_i + h/2, y_i + k2\, h/2)$$
$$k4 = f(x_i + h, y_i + k3\, h)$$

Example:
A ball at 1200 K is allowed to cool down in air at an ambient temperature of 300 K. Assuming heat is lost only due to radiation, the differential equation for the temperature of the ball is given by

$$\frac{d\theta}{dt} = -2.2067\times10^{-12}(\theta^4 - 81\times10^8) \qquad \theta(0)=1200\text{K}$$

where θ is in K and t in seconds. Find the temperature at t = 480 s using Runge-Kutta 4th order method. Assume a step size of h = 240 s.

Compare the result with the exact solution

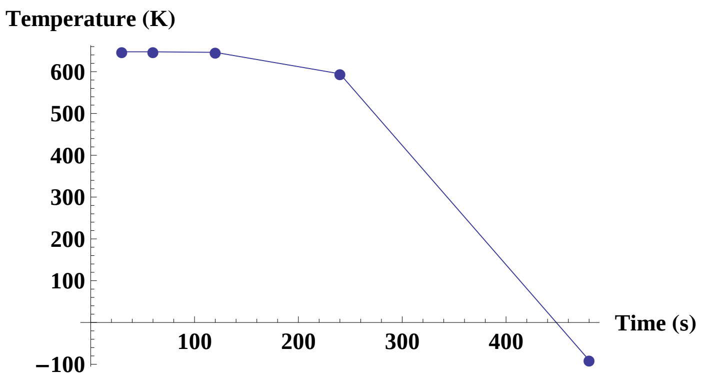
```
Clear[f, h, to, thetao, theta, t, k1, k2, k3, k4];
f = -2.2067 10^(-12) (theta^4 - 81 × 10^8);
h = 240.;

Clear[to, theta, howmany];
to = 0;
thetao = 1200;
howmany = (480 - to) / h;

Print["Approximation"];
Clear[ k1, k2, k3, k4];
Do[
 k1 = f /. {t → to , theta → thetao};
 k2 = f /. {t → to + h / 2 , theta → thetao + k1 h / 2};
 k3 = f /. {t → to + h / 2 , theta → thetao + k2 h / 2};
 k4 = f /. {t → to + h , theta → thetao + k3 h};

 to = to + h;
 thetao = thetao + (k1 + 2 k2 + 2 k3 + k4) h / 6;
 Print[{to, thetao}];
 , {k, 1, howmany}]

Print[];
Print["Exact"];
Clear[theta, t, sol, solut];
sol = NDSolve[{theta'[t] == -2.2067 10^(-12) (theta[t]^4 - 81 × 10^8),
    theta[0] == 1200}, theta, {t, 0, 500}];
solut = sol[[1, 1, 2]];

{240, solut[240]}
{480, solut[480]}
```

```
Approximation

{240., 675.651}

{480., 594.913}


Exact

{240, 775.091}

{480, 647.573}
```

Example:
Make a plot of the solution for theta at t=480 vs the step sizes 30, 60,120, 240, and 480.

```
Clear[f, h, to, theta, t, k1, k2, k3, k4];
f = -2.2067 10^(-12) (theta^4 - 81 × 10^8);

Clear[tot, hh];
tot = 5;
hh = {30., 60., 120., 240., 480.};

Clear[temp];
Do[
  h[kk] = hh[[kk]];

  Clear[to, thetao, howmany];
  to = 0;
  thetao = 1200;
  howmany = (480 - to) / h[kk];

  Clear[ k1, k2, k3, k4];
  Do[
   k1 = f /. {t → to , theta → thetao};
   k2 = f /. {t → to + h / 2 , theta → thetao + k1 h[kk] / 2};
   k3 = f /. {t → to + h / 2 , theta → thetao + k2 h[kk] / 2};
   k4 = f /. {t → to + h , theta → thetao + k3 h[kk]};

   to = to + h[kk];
   thetao = thetao + (k1 + 2 k2 + 2 k3 + k4) h[kk] / 6;
   , {k, 1, howmany}];

  temp[kk] = thetao;
  Print[{h[kk], temp[kk]}];
  , {kk, 1, tot}];

Clear[lis];
lis = Table[{h[kk], temp[kk]}, {kk, 1, tot}];

ListPlot[lis, Joined → True, PlotMarkers → Automatic,
 LabelStyle → Directive[Black, Bold, Medium],
 AxesLabel → {"Time (s)", "Temperature (K)"}]
```

```
{30., 647.572}

{60., 647.539}

{120., 646.161}

{240., 594.913}

{480., -90.2779}
```



Higher order differential equations

We need to rewrite the equation as a set of first order differential equations.
Example:
3 y''[x] + 2 y'[x] + 5 y[x] = exp(-x), y(0)=5, y'(0)=7

Assume
y'[x] = z[x]

Then one equation is
y'[x] = f1(x, y, z) with y(0)=5 where f(x, y, z)= z.

The other equation comes from
3 z'[x] + 2 z [x]+ 5 y [x] = exp(-x)
which is
z'[x] = f2(x, y, z) with z(0)=7 where f2(x, y, z) = [exp(-x) - 2 z - 5 y]/3

Example:
Given
y''[t] + 2 y'[t] + y[t] = exp(-t), y(0)=1, y'(0) =2

Find
a) y(0.75) directly with *Mathematica*
b) y(0.75) by Euler's method [use a step h=0.25]
c) Compute the absolute relative errors between the solution from Euler and the "exact" solution from *Mathematica*:

$$\left|\frac{\text{exact} - \text{Euler}}{\text{exact}}\right| \times 100$$

We need the first order differential equations
y'[t] = z with y[0]= 1

and
z'[t] = exp(-t) - 2 z[t] - y   with   z[0]=2

■ Mathematica

```
Clear[sol, y, t, solut];
sol = DSolve[{y''[t] + 2 y'[t] + y[t] == Exp[-t], y[0] == 1, y'[0] == 2}, y[t], t];
solut = sol[[1, 1, 2]]
solut /. t → 0.75
```

$$\frac{1}{2} e^{-t} \left(2 + 6\,t + t^2\right)$$

```
1.66804
```

■ Euler

```
Clear[to, yo, zo, h];
to = 0.;
yo = 1.;
zo = 2.;
h = 0.25;

Clear[howmany, final];
final = 0.75;
howmany = Floor[(final - to) / h];

Clear[dz, dy];
dy = z;
dz = Exp[-t] - 2 z - y;

Clear[tanY, tanZ];
Do[
 tanY = dy /. z → zo;
 tanZ = dz /. {t → to, y → yo, z → zo};
 to = to + h;
 yo = yo + tanY h;
 zo = zo + tanZ h;
 Print["to=", to, "   yo=", yo, "   zo=", zo];
 , {k, 1, howmany}]
```

```
to=0.25   yo=1.5   zo=1.

to=0.5   yo=1.75   zo=0.3197

to=0.75   yo=1.82993   zo=-0.126017
```

■ Error

```
Abs[((solut /. t → 0.75) - yo) / (solut /. t → 0.75)] 100
```

```
9.70482
```