Mathematica demo

http://www.wolfram.com/

This notebook "MathematicaDemo.nb" can be downloaded from the PMMI course web page (see http://physics.unibe.ch).

Solving equations

Evaluate cells by pressing "shift-enter"

In[23]= `Solve[x^2 + 2*x - 4 == 0, x]

Out[23]= \{ {x -> -1 - \sqrt{5} }, {x -> -1 + \sqrt{5} } \}

The solution is given as a list.

In[24]= `Solve[x^2 + 2*x + 4 == 0, x]

Out[24]= \{ {x -> -1 - i \sqrt{3} }, {x -> -1 + i \sqrt{3} } \}

Differentiation and integration

Define a function f

In[25]= f = Sin[x]^2 / Cos[x];

Note the square brackets for function arguments!

2nd derivative of f

In[26]= D[f, \{x, 2\}]


Taylor series around x=0 to 10th order:

In[27]= `Series[f, \{x, 0, 4\}]

Out[27]= x^2 + x^4 / 6 + O[x]^5

Integrals

In[28]= intf = Integrate[f, x]

Out[28]= -Log[Cos[x/2] - Sin[x/2]] + Log[Cos[x/2] + Sin[x/2]] - Sin[x]

In[29]= intf2 = Integrate[f, \{x, 0, Pi/4\}]

Out[29]= 1 / \sqrt{2} + 2 ArcTanh[Tan[\pi / 8]]
Numerical value of the integral, to 30 digits accuracy

In[30]= N[intf2, 30]
Out[30]= 0.174266805832995500831764962875

Differential equations

- Analytical solution

In[31]= DE = D[x[t], {t, 2}] - D[x[t], t] - Sin[t]
Out[31]= -Sin[t] - x'[t] + x''[t]

This second order equation involves two integration constants C[1] and C[2]:

Note: A=B means A is defined to be B, while A == B represents the equation A=B:

In[32]= DSolve[0 == DE, x[t], t]

The boundary conditions x(0)=1 and dx/dt(0) = 2 are specified as follows:

In[33]= soln = DSolve[(0 == DE, x[0] == 1, x'[0] == 2), x[t], t]
Out[33]= {{x[t] -> 1/2 (-4 + 5 e^t + Cos[t] - Sin[t])}}

Mathematica gives the solutions as a list, even if it is unique. To obtain the solution itself, use soln[[1, 1]]

In[34]= solutionWithBoundary = x[t] /. soln[[1, 1]]
Out[34]= 1/2 (-4 + 5 e^t + Cos[t] - Sin[t])

In[35]= Plot[solutionWithBoundary, {t, 0, 4}]
 Numerical solution

 First order

In[36]:= 
firstOrder = NDSolve[
{y'[x] == x*y[x]*(2-y[x]), y[0] == 1, y[0] == 1, y[0] == 1}, {x, 0, 1}]

Out[36]= 
{{y[x] \[Rule] InterpolatingFunction[[0., 1.], <>][x]}}

The solution is computed for a set of points, and Mathematica provides an interpolation for arbitrary values in the specified interval, in our case x\in[0,1].

In[37]:= yfun = y[x] /. firstOrder[1, 1]

Out[37]= InterpolatingFunction[{{0., 1.}, <>}, <>][x]

In[38]:= Plot[yfun, {x, 0, 1}]

Out[38]=

 Second order

In[39]:= secondOrder = NDSolve[
{y''[x] == -y[x]*3 + Cos[x], y[0] == 1, y'[0] == 0}, y[x], {x, 0, 30}]

Out[39]= 
{{y[x] \[Rule] InterpolatingFunction[[0., 30.], <>][x]}}

In[40]:= yfun2 = y[x] /. secondOrder[1, 1]

Out[40]= InterpolatingFunction[{{0., 30.}, <>}[x]

In[41]:= Plot[yfun2, {x, 0, 30}]

Out[41]=
In[42]:= \text{vel} = D[yfun2, x]  
Out[42]= InterpolatingFunction[{{0., 30.}}, <>][x]  

In[43]:= ParametricPlot[{yfun2, vel}, {x, 0, 30}, PlotRange -> All]