Tutorial 7: Coupled numerical differential equations in *Mathematica*

```
Off[General::spell];
<< Graphics`
<< Graphics`Animation`
```

Version 1, BRW, 8/1/07

The NDSolve function can be used to numercially solve coupled differential equations in *Mathematica* . For lack of a better example, I will solve a set of four coupled 1st order differential equation. These differential equations describe the motion of a double pendululm (see http://scienceworld.wolfram.com/physics/DoublePendulum.html)

Lets define the length a, mass m and gravitational constant g.

```
a = 1.5;
g = 9.81;
m = 0.1;
ω0 = Sqrt[g/a];
```

We need to solve for the angle of pendulum 1 (φ_1) and the angle of pendulum 2 (φ_2) as a function of time. Let us solve for the motion for some specific cases

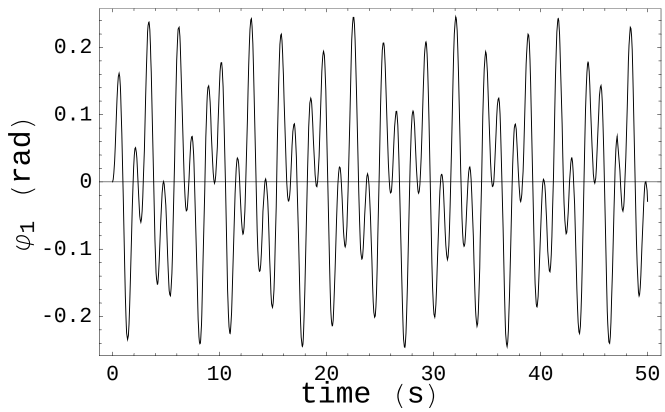
■ Example of small displacement of Pendulum 2

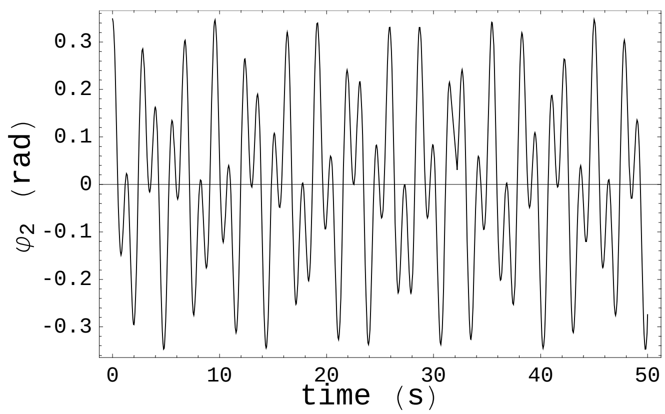
I used the Lagrange equation to give two coupled 2nd order differential equations, these equations can be found from the above webpage. To make things less complicated I then write these as four coupled 1st order differential equations and solve numerically. Here I define x1=φ_1, x3=φ_2, x2=dφ_1/dt and x4=dφ_2/dt. Note that since I have four coupled 1st order differential equations, I need four initial conditions. These initial conditions are φ_2(0)=π/9, φ_1(0)=0 , dφ_1/dt=0, and dφ_2/dt=0. Again, I use NDSolve which will give me four interpolation function as the solutions.

```
sol1 = NDSolve[
   {x1'[t] == x2[t],
    x2'[t] ==
     -1/2 x4'[t] Cos[x1[t] - x3[t]] - ω0^2 Sin[x1[t]] - 1/2 x4[t]^2 Sin[x1[t] - x3[t]],
    x3'[t] == x4[t],
    x4'[t] == -x2'[t] Cos[x1[t] - x3[t]] - ω0^2 Sin[x3[t]] + x2[t]^2 Sin[x1[t] - x3[t]],
    x1[0] == 0, x2[0] == 0, x3[0] == π / 9, x4[0] == 0},
   {x1, x2, x3, x4}, {t, 0, 50}, SolveDelayed -> True];
x1sol[t_] := x1[t] /. sol1[[1]]
x2sol[t_] := x2[t] /. sol1[[1]]
x3sol[t_] := x3[t] /. sol1[[1]]
x4sol[t_] := x4[t] /. sol1[[1]]
```

Below I plot φ_2(t) and φ_1(t)=0. Notice that even for a small displacment, the motion is quite complicated.

```
Plot[x1sol[t], {t, 0, 50}, Frame → True, PlotRange → {All, All}, FrameLabel ->
   {StyleForm["time (s) ", FontSize → 14], StyleForm["φ1 (rad)", FontSize → 14] }];
Plot[x3sol[t], {t, 0, 50}, Frame → True, PlotRange → {All, All}, FrameLabel ->
   {StyleForm["time (s) ", FontSize → 14], StyleForm["φ2 (rad)", FontSize → 14] }];
```





Just for giggles, let us plot the trajectory in a verticle plane y z as a movie. Wiggin cool!

```
z1[t_] = a Sin[x1sol[t]];
y1[t_] = -a Cos[x1sol[t]];
z2[t_] = a (Sin[x1sol[t]] + Sin[x3sol[t]]) ;
y2[t_] = -a (Cos[x1sol[t]] + Cos[x3sol[t]]) ;

Animate[ListPlot[{{0, 0}, {z1[n], y1[n]}, {z2[n], y2[n]}}, Frame → True,
  AspectRatio → Automatic, PlotStyle → {PointSize[0.05], RGBColor[1, 0, 1]},
  PlotRange -> {{-3, 3}, {-3.5, 2}}, GridLines → Automatic,
  PlotLabel → StyleForm["Trajectory", FontSize → 14, FontWeight → "Bold"],
    FrameLabel -> {StyleForm[" x (m) ", FontSize → 14],
    StyleForm["y (m)", FontSize → 14] }], {n, 0.01, 10, 0.1}]
```

The command below plots the move and saves it as an avi file on your computer.

```
p1 = Table[ListPlot[{{0, 0}, {z1[n], y1[n]}, {z2[n], y2[n]}}, Frame → True,
   AspectRatio → Automatic, PlotStyle → {PointSize[0.05], RGBColor[1, 0, 1]},
   PlotRange -> {{-3, 3}, {-3.5, 2}}, GridLines → Automatic,
   PlotLabel → StyleForm["Trajectory", FontSize → 14, FontWeight → "Bold"],
    FrameLabel -> {StyleForm[" x (m) ", FontSize → 14],
     StyleForm["y (m)", FontSize → 14] }], {n, 0.01, 10, 0.1}]
Export["c:\\Temp\\double_case1.avi", p1, "AVI"]
```

```
c:\Temp\double_case1.avi
```