TO SOLVE A DIFFERENTIAL EQUATION BY EULER'S METHOD

Theory

The general form of a first order one-variable ordinary differential equation is

$\frac{\mathrm{dx}}{\mathrm{dt}} = f(x,t)$ … (1)

where $f(x,t)$ is some function.

With initial condition $x = 0$ at $t = 0$.

Using Taylor expansion, we can write the value of x a short interval h later , is

$x(t+h) = x(t) + h\frac{dx}{dt} + \frac{1}{2}h^2\frac{d^2x}{dt^2} + \cdots$ (2)

If h is small, then h^2 is very small and can be neglected. Hence:

$x(t+h) = x(t) + hf(x,t)$ … (3) , where $hf(x,t) = h\frac{dx}{dt}$

If we know the value of x at time t we can use the equation (3) to calculate the value a short time later. Then, we can just repeat the exercise to calculate x another interval h after that, and so forth, and thereby calculate x at a succession of evenly spaced points for as long as we want. We don't get $x(t)$ for all values of t from this calculation, only at a finite set of points, but if h is small enough ,we can get a good picture of what the solution to the equation looks like.

Thus, for instance, we might be given a differential equation for x and an initial condition at $t = a$ and asked to make a graph of $x(t)$ for values of t from a to b. To do this, we would divide the interval from a to b into steps of size h and use (3) repeatedly to calculate $x(t)$, then plot the results. This method for solving differential equations is called Euler's method.

Python program to solve given differential equation from Euler's Method

Let us use Euler's method to solve the differential equation

$\frac{\mathrm{dx}}{\mathrm{dt}} = -x^3 + \sin t$ … (4)

with initial condition $x = 0$ at $t = 0$. Here is a program to do the calculation from $t = 0$ to $t = 10$ in 1000 steps and plot the result. The python program to solve equation (4):

```
from math import sin

from numpy import arange

from pylab import plot, xlabel, ylabel, show

def f(x,t):
```

```
    return -x**3 + sin(t)
a = 0.0           #Start of the interval
b = 10.0          # End of the interval
N = 1000          #Number of steps
h = (b-a)/N       #Size of the single step
x = 0.0           #Initial condition
tpoints = arange(a,b,h)
xpoints = []
x = 0.0
for t in tpoints:
    xpoints.append(x)
      x+=h*f(x,t)
plot(tpoints, xpoints)
xlabel("t")
ylabel("x(t)")
show()
```

The program gives the following output plot: