Ordinary Differential Equations

- Euler's method.
- Runge Kutta methods.
- Simultaneous differential equations.

Euler's method

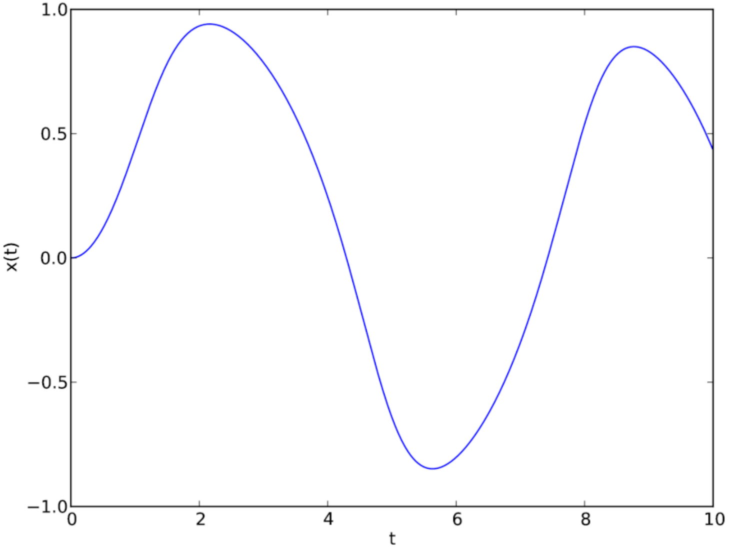
Let us use the Euler's method to solve the following nonlinear inhomogeneous differential equation:

$$\frac{dx}{dt} = -x^3 + \sin t$$

Euler's method – code

```
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 a single step
x = 0.0             # Initial condition

tpoints = arange(a,b,h)
xpoints = []
for t in tpoints:
    xpoints.append(x)
    x += h*f(x,t)

plot(tpoints,xpoints)
```

Euler's method – output



This is a reasonable representation of the actual solution.

Runge-Kutta methods

- Runge-Kutta method is really a set of methods – there are many of them with different orders, which give varying degrees of accuracy.

- Runge-Kutta method is really a set of methods – there are many of them with different orders, which give varying degrees of accuracy.
- Runge-Kutta methods propagate a solution over an interval by combining information from several Euler-style steps, and then using the information obtained to match a Taylor series expansion up to some order.

Runge-Kutta methods

- Runge-Kutta method is really a set of methods – there are many of them with different orders, which give varying degrees of accuracy.
- Runge-Kutta methods propagate a solution over an interval by combining information from several Euler-style steps, and then using the information obtained to match a Taylor series expansion up to some order.
- For many scientific users, fourth-order Runge-Kutta is not just the first word on solving ODE, but the last word as well.

- Runge-Kutta method is really a set of methods – there are many of them with different orders, which give varying degrees of accuracy.
- Runge-Kutta methods propagate a solution over an interval by combining information from several Euler-style steps, and then using the information obtained to match a Taylor series expansion up to some order.
- For many scientific users, fourth-order Runge-Kutta is not just the first word on solving ODE, but the last word as well.
- Technically, Euler's method is the first-order Runge-Kutta method.

Runge-Kutta methods

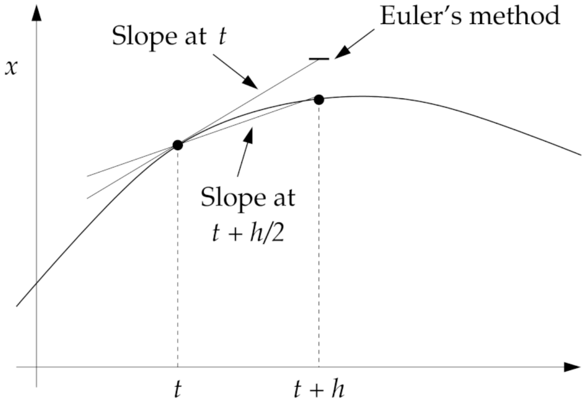
- Runge-Kutta method is really a set of methods – there are many of them with different orders, which give varying degrees of accuracy.
- Runge-Kutta methods propagate a solution over an interval by combining information from several Euler-style steps, and then using the information obtained to match a Taylor series expansion up to some order.
- For many scientific users, fourth-order Runge-Kutta is not just the first word on solving ODE, but the last word as well.
- Technically, Euler's method is the first-order Runge-Kutta method.
- Consider the next method in the series – the second-order Runge-Kutta method.

Runge-Kutta methods

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

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

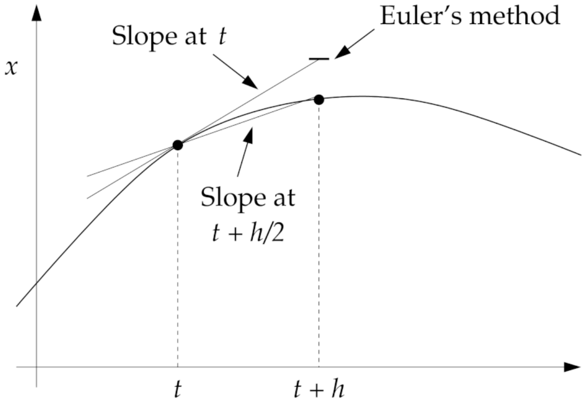
- Euler's method can be represented as:



Runge-Kutta methods

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

- Euler's method can be represented as:



- If we use the slope at $t + \frac{1}{2}h$ to extrapolate, we do better!

- Performing the Taylor expansion arount $t + \frac{1}{2}h$:

$$x(t+h) = x(t+\tfrac{1}{2}h) + \tfrac{1}{2}h\left(\frac{dx}{dt}\right)_{t+\frac{1}{2}h} + \tfrac{1}{8}h^2\left(\frac{d^2x}{dt^2}\right)_{t+\frac{1}{2}h} + \mathcal{O}(h^3)$$

- Performing the Taylor expansion arount $t + \frac{1}{2}h$:

$$x(t+h) = x(t+\tfrac{1}{2}h) + \tfrac{1}{2}h\left(\frac{dx}{dt}\right)_{t+\frac{1}{2}h} + \tfrac{1}{8}h^2\left(\frac{d^2x}{dt^2}\right)_{t+\frac{1}{2}h} + \mathcal{O}(h^3)$$

- Similarly:

$$x(t) = x(t+\tfrac{1}{2}h) - \tfrac{1}{2}h\left(\frac{dx}{dt}\right)_{t+\frac{1}{2}h} + \tfrac{1}{8}h^2\left(\frac{d^2x}{dt^2}\right)_{t+\frac{1}{2}h} + \mathcal{O}(h^3)$$

- Performing the Taylor expansion arount $t + \frac{1}{2}h$:

$$x(t+h) = x(t+\tfrac{1}{2}h)+\tfrac{1}{2}h\left(\frac{dx}{dt}\right)_{t+\frac{1}{2}h}+\tfrac{1}{8}h^2\left(\frac{d^2x}{dt^2}\right)_{t+\frac{1}{2}h}+\mathcal{O}(h^3)$$

- Similarly:

$$x(t) = x(t+\tfrac{1}{2}h)-\tfrac{1}{2}h\left(\frac{dx}{dt}\right)_{t+\frac{1}{2}h}+\tfrac{1}{8}h^2\left(\frac{d^2x}{dt^2}\right)_{t+\frac{1}{2}h}+\mathcal{O}(h^3)$$

$$\begin{aligned} x(t+h) &= x(t) + h\left(\frac{dx}{dt}\right)_{t+\frac{1}{2}h} + \mathcal{O}(h^3) \\ &= x(t) + hf(x(t+\tfrac{1}{2}h), t+\tfrac{1}{2}h) + \mathcal{O}(h^3) \end{aligned}$$

- Performing the Taylor expansion arount $t + \frac{1}{2}h$:

$$x(t+h) = x(t+\tfrac{1}{2}h) + \tfrac{1}{2}h\left(\frac{dx}{dt}\right)_{t+\frac{1}{2}h} + \tfrac{1}{8}h^2\left(\frac{d^2x}{dt^2}\right)_{t+\frac{1}{2}h} + \mathcal{O}(h^3)$$

- Similarly:

$$x(t) = x(t+\tfrac{1}{2}h) - \tfrac{1}{2}h\left(\frac{dx}{dt}\right)_{t+\frac{1}{2}h} + \tfrac{1}{8}h^2\left(\frac{d^2x}{dt^2}\right)_{t+\frac{1}{2}h} + \mathcal{O}(h^3)$$

$$\begin{aligned} x(t+h) &= x(t) + h\left(\frac{dx}{dt}\right)_{t+\frac{1}{2}h} + \mathcal{O}(h^3) \\ &= x(t) + hf(x(t+\tfrac{1}{2}h), t+\tfrac{1}{2}h) + \mathcal{O}(h^3) \end{aligned}$$

Error is now $\mathcal{O}(h^3)$! Better than Euler ($\mathcal{O}(h^2)$).

Runge-Kutta methods

- Problem is that this requires the knowledge of $x(t + \frac{1}{2}h)$, which we dont have!

- Problem is that this requires the knowledge of $x(t + \frac{1}{2}h)$, which we dont have!
- We get around this by approximating $x(t + \frac{1}{2}h)$ using Euler's method:

$$x(t + \tfrac{1}{2}h) = x(t) + \tfrac{1}{2}hf(x, t)$$

- Problem is that this requires the knowledge of $x(t+\frac{1}{2}h)$, which we dont have!
- We get around this by approximating $x(t+\frac{1}{2}h)$ using Euler's method:

$$x(t+\tfrac{1}{2}h) = x(t) + \tfrac{1}{2}hf(x,t)$$

- Then the whole algorithm becomes:

$$\begin{aligned} k_1 &= hf(x,t) \\ k_2 &= hf(x+\tfrac{1}{2}k_1, t+\tfrac{1}{2}h) \\ x(t+h) &= x(t) + k_2 \end{aligned}$$

Runge-Kutta methods

- Problem is that this requires the knowledge of $x(t + \frac{1}{2}h)$, which we dont have!
- We get around this by approximating $x(t + \frac{1}{2}h)$ using Euler's method:

$$x(t + \tfrac{1}{2}h) = x(t) + \tfrac{1}{2}hf(x, t)$$

- Then the whole algorithm becomes:

$$\begin{aligned} k_1 &= hf(x, t) \\ k_2 &= hf(x + \tfrac{1}{2}k_1, t + \tfrac{1}{2}h) \\ x(t + h) &= x(t) + k_2 \end{aligned}$$

- Error in each step is $\mathcal{O}(h^3)$ and global error is $\mathcal{O}(h^2)$.

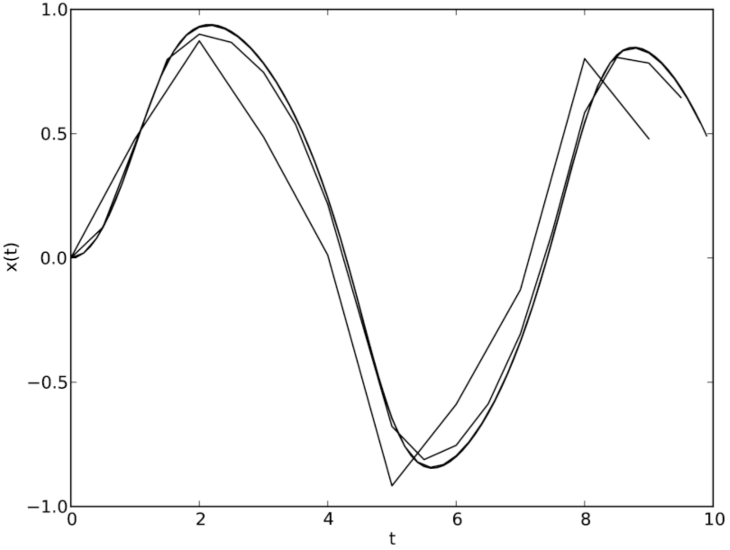
Let us use the second-order Runge-Kutta method to solve the differential equation:

$$\frac{dx}{dt} = -x^3 + \sin t$$

Second order Runge-Kutta method – code

```
from math import sin
from numpy import arange
from pylab import plot
def f(x,t):
    return -x**3 + sin(t)

a = 0.0
b = 10.0
N = 10
h = (b-a)/N
tpoints = arange(a,b,h)
xpoints = []

x = 0.0
for t in tpoints:
    xpoints.append(x)
    k1 = h*f(x,t)
    k2 = h*f(x+0.5*k1,t+0.5*h)
    x += k2
plot(tpoints,xpoints)
```

Second order Runge-Kutta method – output



$N = 10, 20, 50, 100$ Convergence at $N = 50$ vs 1000 for Euler

Fourth order Runge-Kutta method

- Approach can be extended by performing Taylor expansions around various points and taking the right linear combinations to arrange h^3, h^4 terms to cancel!

Fourth order Runge-Kutta method

- Approach can be extended by performing Taylor expansions around various points and taking the right linear combinations to arrange h^3, h^4 terms to cancel!
- Fourth order Runge-Kutta offers a balance between accuracy and ease to program and is considered to be the sweet spot.

$$
\begin{aligned}
k_1 &= hf(x,t) \\
k_2 &= hf(x + \tfrac{1}{2}k_1, t + \tfrac{1}{2}h) \\
k_3 &= hf(x + \tfrac{1}{2}k_2, t + \tfrac{1}{2}h) \\
k_4 &= hf(x + k_3, t + h) \\
x(t+h) &= x(t) + \tfrac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)
\end{aligned}
$$

Fourth order Runge-Kutta method

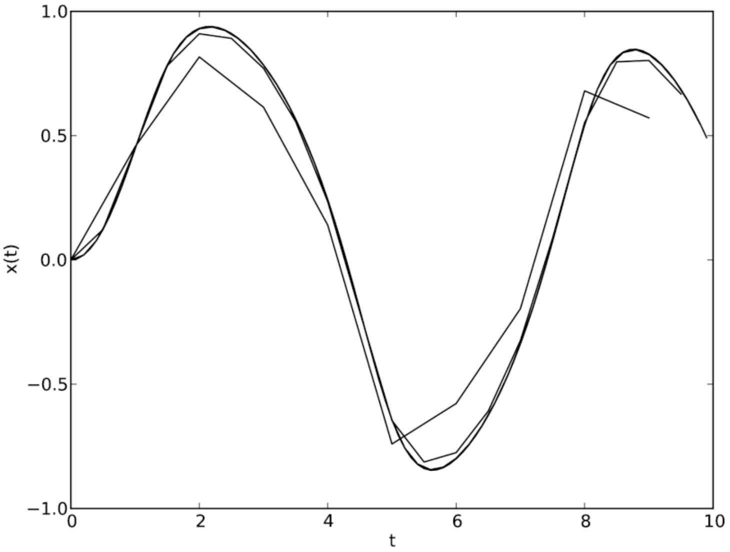
Let us use the fourth-order Runge-Kutta method to solve the differential equation:

$$\frac{dx}{dt} = -x^3 + \sin t$$

Fourth order Runge-Kutta method – code

```
from math import sin
from numpy import arange
from pylab import plot
def f(x,t):
    return -x**3 + sin(t)
a = 0.0
b = 10.0
N = 10
h = (b-a)/N
tpoints = arange(a,b,h)
xpoints = []
x = 0.0
for t in tpoints:
    xpoints.append(x)
    k1 = h*f(x,t)
    k2 = h*f(x+0.5*k1,t+0.5*h)
    k3 = h*f(x+0.5*k2,t+0.5*h)
    k4 = h*f(x+k3,t+h)
    x += (k1+2*k2+2*k3+k4)/6
plot(tpoints,xpoints)
```

Fourth order Runge-Kutta method – output



$N = 10, 20, 50, 100$ Convergence at $N = 20$ vs 1000 for Euler

Solution over infinite ranges

- In some cases, we want to march in time from some initial value to not some finite value in time, but to $t = \infty$.

Solution over infinite ranges

- In some cases, we want to march in time from some initial value to not some finite value in time, but to $t = \infty$.
- Here we can play the same kind of tricks that we had played while doing integrals – change of variables:

$$u = \frac{t}{1+t} \quad \text{or} \quad t = \frac{u}{1-u}$$

Solution over infinite ranges

- In some cases, we want to march in time from some initial value to not some finite value in time, but to $t = \infty$.
- Here we can play the same kind of tricks that we had played while doing integrals – change of variables:

$$u = \frac{t}{1+t} \quad \text{or} \quad t = \frac{u}{1-u}$$

- With this substitution, as $u \to 1$, $t \to \infty$.

Solution over infinite ranges

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

$$\text{Using Chain rule} \quad \frac{dx}{du}\frac{du}{dt} = f(x,t)$$

$$\frac{dx}{du} = \frac{dt}{du} f\left(x, \frac{u}{1-u}\right)$$

$$\text{But} \quad \frac{dt}{du} = \frac{1}{(1-u)^2}$$

$$\frac{dx}{du} = (1-u)^{-2} f\left(x, \frac{u}{1-u}\right)$$

$$\text{define} \quad g(x,u) \equiv (1-u)^{-2} f\left(x, \frac{u}{1-u}\right)$$

$$\frac{dx}{du} = g(x,u)$$

- In principle, not only this transformation, but several others can also be considered – such as those based on trignometric functions, hyperbolic functions etc.

- In principle, not only this transformation, but several others can also be considered – such as those based on trignometric functions, hyperbolic functions etc.
- Choose the transformation which makes the algebra easier!

- In principle, not only this transformation, but several others can also be considered – such as those based on trignometric functions, hyperbolic functions etc.
- Choose the transformation which makes the algebra easier!
- Consider the following equation:

$$\frac{dx}{dt} = \frac{1}{x^2 + t^2}$$

with $x = 1$ at $t = 0$, and we would like to know the solution from $t = 0$ to $t = \infty$.

- In principle, not only this transformation, but several others can also be considered – such as those based on trignometric functions, hyperbolic functions etc.
- Choose the transformation which makes the algebra easier!
- Consider the following equation:

$$\frac{dx}{dt} = \frac{1}{x^2 + t^2}$$

with $x = 1$ at $t = 0$, and we would like to know the solution from $t = 0$ to $t = \infty$.

- Using the substitution:

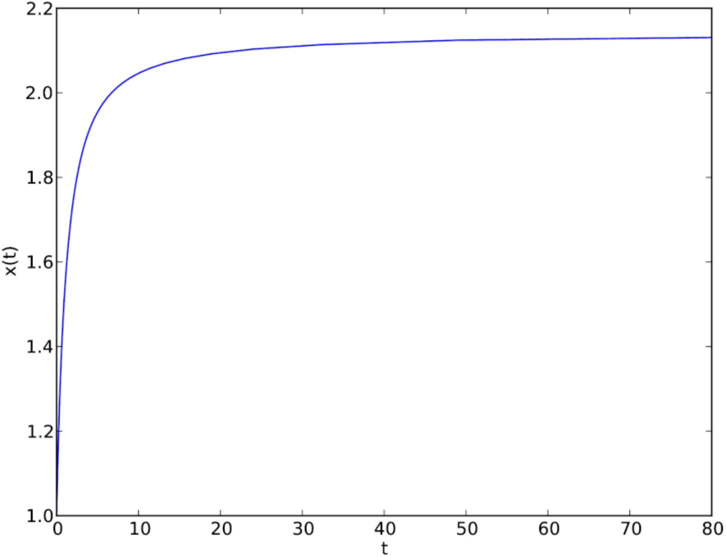
$$\frac{dx}{du} = \frac{1}{x^2(1-u)^2 + u^2}$$

with $x = 1$ at $u = 0$ and range of u goes from $u = 0$ to $u = 1$.

Solution over infinite ranges – code

```
from numpy import arange
from pylab import plot,xlabel,ylabel,xlim,show

def g(x,u):
    return 1/(x**2*(1-u)**2+u**2)

a = 0.0
b = 1.0
N = 100
h = (b-a)/N

upoints = arange(a,b,h)
tpoints = []
xpoints = []

x = 1.0
for u in upoints:
    tpoints.append(u/(1-u))
    xpoints.append(x)
    k1 = h*g(x,u)
    k2 = h*g(x+0.5*k1,u+0.5*h)
    k3 = h*g(x+0.5*k2,u+0.5*h)
    k4 = h*g(x+k3,u+h)
    x += (k1+2*k2+2*k3+k4)/6

plot(tpoints,xpoints)
xlim(0,80)
xlabel("t")
ylabel("x(t)")
show()
```

Solution over infinite ranges – output



Differential equations with more than one variable

- In a lot of physics problems, we have more than one variable – ie we have simultaneous differential equations, where the derivative of each variable can depend on any or all of the variables as well as the independent variable, t:

$$\frac{dx}{dt} = f_x(x, y, t) \quad \frac{dy}{dt} = f_y(x, y, t)$$

where f_x and f_y are possibly, nonlinear functions of x, y and t.

Differential equations with more than one variable

- In a lot of physics problems, we have more than one variable – ie we have simultaneous differential equations, where the derivative of each variable can depend on any or all of the variables as well as the independent variable, t:

$$\frac{dx}{dt} = f_x(x, y, t) \quad \frac{dy}{dt} = f_y(x, y, t)$$

where f_x and f_y are possibly, nonlinear functions of x, y and t.

- These equations can be written in a vector form as:

$$\frac{d\mathbf{r}}{dt} = \mathbf{f}(\mathbf{r}, t)$$

where $\mathbf{r} = (x, y, \ldots)$ and $\mathbf{f}$ is a vector of functions, $\mathbf{f}(\mathbf{r}, t) = (f_x(\mathbf{r}, t), f_y(\mathbf{r}, t), \ldots)$.

- Computationally, these are not much more difficult than one-variable case!

- Computationally, these are not much more difficult than one-variable case!
- We can Taylor expand the vector $\mathbf{r}$ as:

$$\mathbf{r}(t+h) = \mathbf{r}(t) + h\frac{d\mathbf{r}}{dt} + \mathcal{O}(h^2)$$

Differential equations with more than one variable

- Computationally, these are not much more difficult than one-variable case!
- We can Taylor expand the vector $\mathbf{r}$ as:

$$\mathbf{r}(t+h) = \mathbf{r}(t) + h\frac{d\mathbf{r}}{dt} + \mathcal{O}(h^2)$$

- Then Euler's method:

$$\mathbf{r}(t+h) = \mathbf{r}(t) + h\mathbf{f}(\mathbf{r}, t)$$

Differential equations with more than one variable

- Computationally, these are not much more difficult than one-variable case!
- We can Taylor expand the vector $\mathbf{r}$ as:

$$\mathbf{r}(t+h) = \mathbf{r}(t) + h\frac{d\mathbf{r}}{dt} + \mathcal{O}(h^2)$$

- Then Euler's method:

$$\mathbf{r}(t+h) = \mathbf{r}(t) + h\mathbf{f}(\mathbf{r}, t)$$

- Fourth order Runge-Kutta:

$$\begin{aligned}
\mathbf{k}_1 &= h\mathbf{f}(\mathbf{r}, t) \\
\mathbf{k}_2 &= h\mathbf{f}(\mathbf{r} + \tfrac{1}{2}\mathbf{k}_1, t + \tfrac{1}{2}h) \\
\mathbf{k}_3 &= h\mathbf{f}(\mathbf{r} + \tfrac{1}{2}\mathbf{k}_2, t + \tfrac{1}{2}h) \\
\mathbf{k}_4 &= h\mathbf{f}(\mathbf{r} + \mathbf{k}_3, t + h) \\
\mathbf{r}(t+h) &= \mathbf{r}(t) + \tfrac{1}{6}(\mathbf{k}_1 + 2\mathbf{k}_2 + 2\mathbf{k}_3 + \mathbf{k}_4)
\end{aligned}$$

Simultaneous differential equations – example

Consider the following equations:

$$\frac{dx}{dt} = xy - x, \quad \frac{dy}{dt} = y - xy + \sin^2 \omega t$$

with initial conditions:

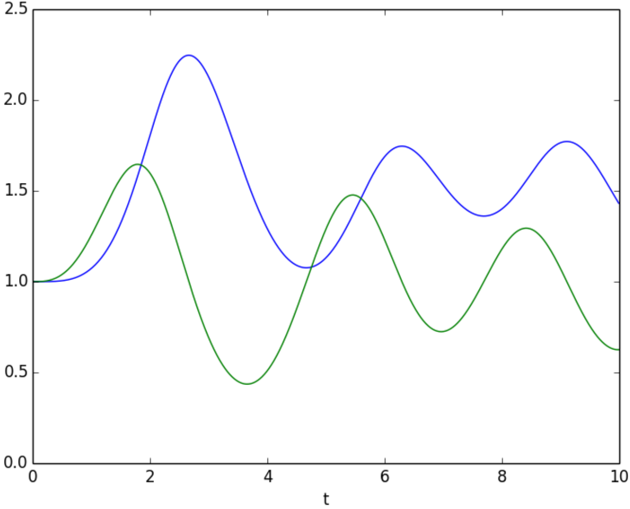
$$x = y = 1 \quad at \quad t = 0$$

and $\omega = 1$.

Simultaneous differential equations – code

```
from math import sin
from numpy import array,arange
from pylab import plot,xlabel,show,savefig

def f(r,t):
    x = r[0]
    y = r[1]
    fx = x*y - x
    fy = y - x*y + sin(t)**2
    return array([fx,fy],float)

a = 0.0
b = 10.0
N = 1000
h = (b-a)/N

tpoints = arange(a,b,h)
xpoints = []
ypoints = []

r = array([1.0,1.0],float)
for t in tpoints:
    xpoints.append(r[0])
    ypoints.append(r[1])
    k1 = h*f(r,t)
    k2 = h*f(r+0.5*k1,t+0.5*h)
    k3 = h*f(r+0.5*k2,t+0.5*h)
    k4 = h*f(r+k3,t+h)
    r += (k1+2*k2+2*k3+k4)/6
plot(tpoints,xpoints)
plot(tpoints,ypoints)
xlabel("t")
savefig('simultaneous.png')
show()
```

Simultaneous differential equations – code



Higher order differential equations

- A general second-order differential equation:

$$\frac{d^2x}{dt^2} = f\left(x, \frac{dx}{dt}, t\right)$$

Higher order differential equations

- A general second-order differential equation:

$$\frac{d^2x}{dt^2} = f\left(x, \frac{dx}{dt}, t\right)$$

- We can reduce it to 2 first-order ODEs:

$$\frac{dx}{dt} = y, \quad \frac{dy}{dt} = f(x, y, t)$$

Higher order differential equations

- A general second-order differential equation:

$$\frac{d^2x}{dt^2} = f\left(x, \frac{dx}{dt}, t\right)$$

- We can reduce it to 2 first-order ODEs:

$$\frac{dx}{dt} = y, \quad \frac{dy}{dt} = f(x, y, t)$$

- Similarly for 3rd order equation:

$$\frac{d^3x}{dt^3} = f\left(x, \frac{dx}{dt}, \frac{d^2x}{dt^2}, t\right)$$

reduces to:

$$\frac{dx}{dt} = y, \quad \frac{dy}{dt} = z, \quad \frac{dz}{dt} = f(x, y, z, t)$$

Higher order differential equations

- A general second-order differential equation:

$$\frac{d^2x}{dt^2} = f\left(x, \frac{dx}{dt}, t\right)$$

- We can reduce it to 2 first-order ODEs:

$$\frac{dx}{dt} = y, \quad \frac{dy}{dt} = f(x, y, t)$$

- Similarly for 3rd order equation:

$$\frac{d^3x}{dt^3} = f\left(x, \frac{dx}{dt}, \frac{d^2x}{dt^2}, t\right)$$

reduces to:

$$\frac{dx}{dt} = y, \quad \frac{dy}{dt} = z, \quad \frac{dz}{dt} = f(x, y, z, t)$$

- We can solve using methods we already know about simultaneous equations.