

# Modern Computational Accelerator Physics

**James Amundson**   Alexandru Macridin   *Panagiotis Spentzouris*

Fermilab

USPAS January 2015

# Numerical Solution of ODEs

Start with an  $n^{\text{th}}$ -order set of coupled ODEs involving  $\mathbf{x}$  and  $\frac{d^n \mathbf{x}}{dt^n}$ .  
Reduce to a set of first-order equations by introducing

$$\xi = \frac{d\mathbf{x}}{dt}$$

to obtain an  $n - 1$ -order ODE in  $\mathbf{x}$  coupled to a first-order equation in  $\xi$ .  
Repeat until all equations are first-order.

Then, the general case is

$$\dot{\mathbf{y}} = f(\mathbf{y}, t).$$

We want to integrate this set of equations for a total time  $T = t_1 - t_0$ . To do so, divide  $T$  into steps of size  $h$ .

The simplest method is that of Euler.

$$\mathbf{y}(t+h) = \mathbf{y}(t) + h\dot{\mathbf{y}}(t).$$

In terms of steps,

$$\mathbf{y}^{n+1} = \mathbf{y}^n + h\dot{\mathbf{y}}^n.$$

Taylor expand  $\mathbf{y}(t+h)$ ,

$$\mathbf{y}(t+h) = \mathbf{y}(t) + h\dot{\mathbf{y}}(t) + \frac{h^2}{2}\ddot{\mathbf{y}}(t+h) + \mathcal{O}(h^3),$$

to see that Euler is accurate up to terms of  $\mathcal{O}(h)$ .

We call Euler a first-order scheme.

If we stick with Euler we will need to make  $h$  very small, *i.e.*, we will need to take a large number of steps.

Once we start looking for higher-order methods, we find that there are infinitely many at a given order.

# 4<sup>th</sup>-order Runge-Kutta

4<sup>th</sup>-order Runge-Kutta is a favorite:

$$\mathbf{y}^{n+1} = \mathbf{y}^n + \frac{1}{6}(a + 2b + 2c + d),$$

where

$$a = hf(\mathbf{y}^n, t^n),$$

$$b = hf\left(\mathbf{y}^n + \frac{a}{2}, t^n + \frac{h}{2}\right),$$

$$c = hf\left(\mathbf{y}^n + \frac{b}{2}, t^n + \frac{h}{2}\right),$$

and

$$d = hf(\mathbf{y}^n + c, t^n + h).$$

Note that this method requires four function evaluations.

## 4<sup>th</sup>-order Runge-Kutta, continued

To see that this method is, indeed, accurate to 4<sup>th</sup> order in  $h$ , Taylor expand

$$\mathbf{y}^{n+1} = \mathbf{y}^n + h\dot{\mathbf{y}}^n + \frac{h^2}{2}\ddot{\mathbf{y}}^n + \dots$$

and use

$$\begin{aligned}\frac{d}{dt} &= \frac{\partial}{\partial t} + \frac{d\mathbf{y}}{dt} \frac{\partial}{\partial \mathbf{y}} \\ &= \frac{\partial}{\partial t} + f \frac{\partial}{\partial \mathbf{y}}.\end{aligned}$$

You can find this in “any book.”

There are many algorithms in common use. See the exercise.

Some of the more interesting methods make use of the Jacobian

$$J_{ij} \equiv \frac{\partial f_i}{\partial y_j}.$$

Read Numerical Recipes

## Example

Example: A non-relativistic particle in a uniform gravitational field in one dimension.

$$m \frac{d^2 x}{dt^2} = -g \Rightarrow \frac{d^2 x}{dt^2} = -\frac{g}{m}$$
$$\mathbf{y} = \begin{pmatrix} x \\ \dot{x} \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \end{pmatrix},$$

where

$$\dot{y}_0 = y_1$$
$$\dot{y}_1 = -\frac{g}{m} \equiv -\gamma$$

Finally we have the form we require:

$$\frac{d}{dt} \begin{pmatrix} y_0 \\ y_1 \end{pmatrix} = \begin{pmatrix} y_1 \\ -\gamma \end{pmatrix}$$

and

$$J = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}.$$

Get the example code from the wiki:

```
ode_example.py  
uniform_accel.py
```

# Assignment 1

- (1) Rewrite the example for the simple harmonic oscillator

$$m \frac{d^2 x}{dt^2} = -kx,$$

using

$$\kappa \equiv \frac{k}{m}.$$

Put the new functions in a file `sho.py`. You should include the analytic solution, but you may start with the initial condition  $y_1(0) = 0$ .

- (2) Try all the integrators in GSL. See how many steps and how much time is required for each.
- (3) Calculate the (scaled) energy for the SHO. Redo (2), including the energy in the plots.