Exercícios de

Física Computacional

(Parte 9)

Mestrado em Engenharia Física-Tecnológica (MEFT)

Rui Coelho

Departamento de Física do Instituto Superior Técnico

Ano Lectivo: 2019-20

rui.alves.coelho@tecnico.ulisboa.pt

versão: 1 de Outubro de 2019

9. Equações diferenciais ordinárias

Exercício 60. Neste exercício vamos testar o método de **Euler** num problema muito simples. Considere a equação diferencial

$$dx/dt = t^2 \quad \text{com } x(t=0)=2.0$$

a) Calcule através do método de Euler a solução da equação diferencial entre t=0 e t=4. Use um passo no tempo h=0.1.
b) Produza numa figura a solução numérica, sobreposta à solução analítica. Será o método minimamente preciso ?

Exercício 61. Neste exercício vamos testar o método **trapezoidal** (*aka* Crank-Nicholson ou Heun) no mesmo problema do Exercício 60. A equação diferencial e condição inicial é dada por

$$dx/dt = t^2 \quad \text{com } x(t=0)=2.0$$

a) Queremos calcular a solução da equação diferencial entre t=0 e t=4 usando o mesmo passo no tempo h=0.1.
b) Produza numa figura a solução numérica, sobreposta à solução analítica. Será o método mais preciso que o de Euler ?

Exercício 62. Neste exercício vamos testar o método **Leap-Frog** (*aka* Stormer-Verlet) no mesmo problema do Exercício 60. A equação diferencial e condição inicial é dada por

$$dx/dt = t^2 \quad \text{com } x(t=0)=2.0$$

a) Queremos calcular a solução da equação diferencial entre t=0 e t=4 usando o mesmo passo no tempo h=0.1.
b) Produza numa figura a solução numérica, sobreposta à solução analítica. Será o método mais preciso que o trapezoidal ?

Exercício 63. Neste exercício vamos testar o método **RK2** noutro problema sendo equação diferencial e condição inicial é dada por

$$dy/dt = 0.5Y - t + 1 \quad \text{com } Y(t=0.4)=0.2$$

a) Queremos calcular a solução da equação diferencial entre t=0.4 e t=1.8 usando um passo no tempo h=0.1.
b) Produza uma figura com a solução numérica.

Exercício 64. (adaptado de *Barão 2016*) A resolução de equações diferenciais ordinárias (ODEs) por via numérica, exige a implementação dos diferentes métodos iterativos (Euler, Runge-Kutta, etc.) de forma a obter-se a solução das equações. Dado que o número de variáveis dependentes do sistema, correspondentes ao número de equações diferenciais de primeira ordem a resolver, podem ser diferentes em cada problema, será útil implementar uma classe em C++ suficientemente flexível e capaz de lidar com os diferentes números de variáveis. Além do mais, na resolução do sistema de equações diferenciais, a variável independente (habitualmente o tempo) e as variáveis dependentes (os graus de liberdade necessários à resolução do problema bem como as suas primeiras derivadas) são iteradas e os seus valores registados. Torna-se por isso conveniente, definir uma classe ODEpoint que aramazene em cada iteração os valores das variáveis

a) Implemente a classe ODEpoint de acordo com a declaração incompleta abaixo. A descrição completa encontra-se nos slides da UC e os métodos a implementar deverão seguir ***integralmente*** a descrição dada (eventuais alterações apenas se forem com métodos adicionais).

```
class ODEpoint {
public:
  ODEpoint();//default constructor
  ~ODEpoint();//destructor
…
private:
  double t;
  vector<double> var;
  int Ndim;
};
```

b) Implemente a classe ODEsolver de acordo com a declaração incompleta abaixo. Os métodos a implementar deverão seguir ***integralmente*** a descrição dada (eventuais alterações apenas se forem com métodos adicionais).

```
class ODEsolver{
public:
  ODEsolver(vector<TFormula> Form);
  ~ODEsolver();
  vector <ODEpoint> Eulersolver(const ODEpoint& P0, double xmin, double xmax, double h_step);
  vector<ODEpoint> RK2solver(const ODEpoint& P0, double xmin, double xmax, double h_step);
  vector <ODEpoint> RK4solver(const ODEpoint& P0, double xmin, double xmax, double h_step);
  vector <ODEpoint> RK4_AdapStep(const ODEpoint& P0, double xmin, double xmax, double h_step);
  vector<ODEpoint> Heun(const ODEpoint& P0, double xmin, double xmax, double h_step);
  …
  void SetODEfunc(vector<TFormula> Form);
private:
  ODEpoint Heun_iterator (const ODEpoint&, double step);
  ODEpoint EULER_iterator (const ODEpoint&, double step);
  ODEpoint RK2_iterator (const ODEpoint&, double step);
  ODEpoint RK4_AS_iterator(const ODEpoint&, double step, vector<vector<double> >& K);
  ODEpoint RK4_iterator(const ODEpoint&, double step);
  vector<TFormula> F;
};
```

Exercício 65. Utilize agora a classe ***ODEsolver*** para resolver a seguinte (*a mesma do Exercicio 63*) equação diferencial com condição inicial dada:

dy/dt = 0.5Y - t +1 com Y(t=0.4)=0.2

a) Utilize os métodos de Euler, Runge-Kutta2 e Runge-Kutta4 e obtenha uma figura onde mostre as soluções obtidas pelos 3 métodos no mesmo sistema de eixos. O passo no tempo nos 3 métodos é o mesmo i.e. h=0.1s.

Exercício 66. Utilize agora a classe ***ODEsolver*** para obter a famosa figura do atrator de Lorentz:

$$\frac{dx}{dt} = \sigma(y - x)$$
$$\frac{dy}{dt} = x(\rho - z) - y$$
$$\frac{dz}{dt} = xy - \beta z$$

Parameters

$\sigma = 10$

$\rho = 28$

$\beta = 8/3$

a) Utilize apenas o método de **Runge-Kutta4**, com passo no tempo h=1e-5s e obtenha a trajectória ao longo de ***100s*** no tempo.
b) Obtenha a figura que ilustre a solução de pontos (x,y,z) ao longo do tempo (*há classes do ROOT TGraph2D propositadamente para gráficos 3D…*) tal como indicado nos slides de apoio da Unidade Curricular.

Exercício 67. Finalmente, vamos utilizar uma extensão da classe ***ODEsolver*** com o método de Verlet para a resolução de equações diferenciais de 2ªordem. O objective é resolver a equação do pêndulo

$$\frac{d^2\theta}{dt^2} = -k\sin(\theta) \quad , k = g/L \qquad \theta(0) = \theta_0 \quad , \quad \omega(0) = \omega_0$$

a) Usando k=5, θ_0=2rad e $\omega(0)$=0, obtenha figuras com a trajectória $\theta(t)$ com t=[0,2π] quer com o método de Verlet quer com o método de RK4. Passo no tempo h=0.1s.

Exercício 68. (*Barão 2016*) Uma barra cilíndrica de diâmetro D = 101 cm e comprimento de L = 100 cm está em contacto com uma fonte de calor à temperatura de T_s = 40 graus Celsius.

a) Admitindo que a barra está isolada, conduzindo calor entre as duas extremidades, a temperatura da barra obedece à seguinte equação:

$$\frac{\partial^2 T}{\partial x^2} = 0$$

resolva numericamente esta equação sabendo que as temperaturas nas extremidades da barra são T(x=0)=40 e T(x=L)=10graus Celsius. Utilize como passo **s=10cm** e **s = 2 cm**. Produza um plot para cada um dos casos.