Exercícios de

Física Computacional

(Parte 6)

Mestrado em Engenharia Física-Tecnológica (MEFT)

Rui Coelho

Departamento de Física do Instituto Superior Técnico

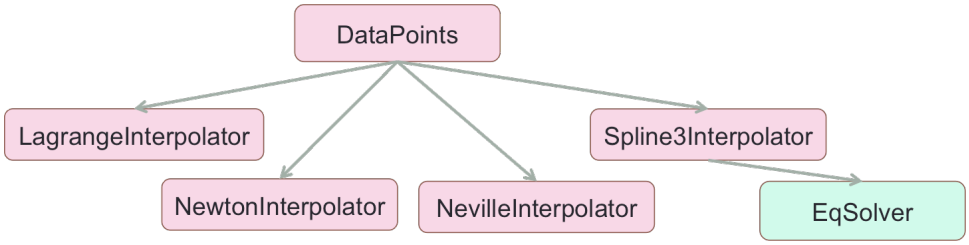
Ano Lectivo: 2019-20

rui.alves.coelho@tecnico.ulisboa.pt

versão: 1 de Outubro de 2019

6. Interpolação

Exercício 45. (*adaptado de Barão 2016*) Para a realização de interpolações, pode-se definir uma classe ***DataPoints***, que conterá os dados respeitantes aos pontos e classes que herdem desta onde se implementarão os diferentes métodos de interpolação. Teremos assim, para este exercício, as classes de interpolação ***LagrangeInterpolator, NevilleInterpolator*** e ***Spline3Interpolator***



A classe ***DataPoints*** deve ser usada para o armazenamento dos pontos, podendo também ser feita a sua representação gráfica. A descrição da classe encontra-se abaixo indicada

```
#include <string>
using namespace std;

class DataPoints {
 public:
  DataPoints();
  DataPoints(int, double*, double*);
  virtual ~DataPoints();

  virtual double Interpolate(double x) {return 0.;}
  virtual void Draw();
  virtual void Print(string FILE="");
 protected:
  int N; // number of data points
  double *x, *y; // arrays
  static int Nplots; //how many plots already done
};
```

Para a implementação dos interpoladores, devemos ter em conta a descrição seguinte para LagrangeInterpolator (para Spline3Interpolator, são necessários métodos adicionais – ver teórica).

```
class LagrangeInterpolator : public DataPoints {

 public:
  LagrangeInterpolator(int N=0, double *x=NULL, double *y=NULL, TF1* fF0=NULL);
  ~LagrangeInterpolator() {;}

  double Interpolate(double x);
  void Draw(); //draw everything (points and interpolation function)
  TF1* GetInterpolationFunction() {return FInterpolator;}

  void SetFunction(TF1*);
  void Print(string FILE=""); // print results to file (optional)

 private:
  double fInterpolator(double *fx, double *par) {
    return Interpolate(fx[0]);
  }
  TF1* FInterpolator; //interpolation function
  TF1* F0;  //eventual underlying function
};
```

Proceda à implementação das classes.

Exercício 46 : (adaptado de *Barão 2016*) Usando as classes construídas anteriormente e dados os seguintes pontos:

x	-1.2	0.3	1.1
y	-5.76	-5.61	-3.69

realize um programa ***main*** que determine o valor ***y(0)*** e desenhe a função interpoladora dos pontos usando os diferentes métodos:

a) Lagrange
b) Neville
c) Spline cúbico
d) Newton (*opcional*)

Exercício 47 : (adaptado de *Barão 2016*) Usando a classe Spline3Interpolator e dados os seguintes pontos:

x	1	2	3	4	5
y	0	1	0	1	0

realize um programa ***main*** que desenhe a função interpoladora dos pontos.

Exercício 48 : Usando a classe Spline3Interpolator e dados 10,000 pontos (x=[0,maxgr]) da função f(x) definida por:

```
maxrg=80.0;
f = new TF1("f","TMath::Cos(3.*x)", 0., maxrg);
```

realize um programa ***main*** que desenhe os pontos e a função interpoladora dos pontos. Use uma canvas suficientemente larga para poder ver a sinusoide bem definida assim como a curva interpoladora.

Será que a curva interpoladora está "*errada*" ou será este um efeito "*de sampling*" ?! Para verificar, investigue se pode melhorar o *sampling* da função interpoladora que é desenhada.