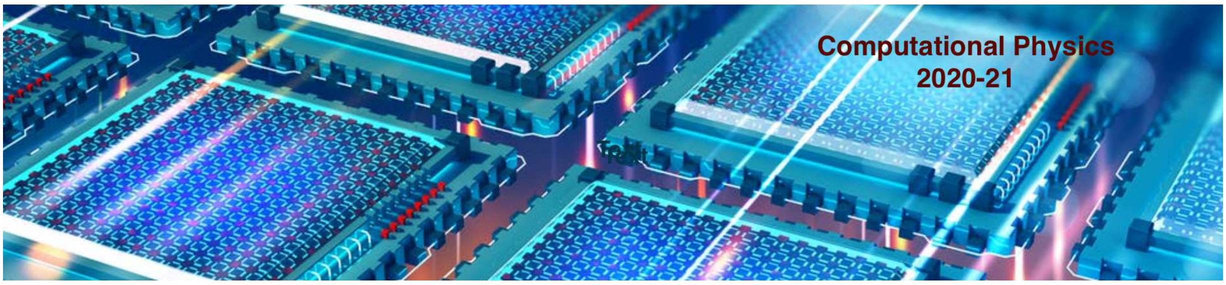
Computational Physics

numerical methods with C++ (and UNIX)

2020-21



Fernando Barao

Instituto Superior Técnico, Dep. Fisica

email: fernando.barao@tecnico.ulisboa.pt

Computational Physics

Numerical integration

Fernando Barao, Phys Department IST (Lisbon)

Romberg integration

It improves the results of numerical integration using error-correction techniques

uses two estimates of the integral with different precisions, to compute a more accurate approximation

trapezoidal rule

$I = I(h) + E(h)$

I: exact value of integral

$I(h)$: integral evaluation using trapezoidal rule with step size $\frac{b-a}{n}$

$E(h)$: truncature error
$E(h) \simeq -\frac{b-a}{12}\ h^2\ \bar{f}''$

How to combine different precision estimations?

$I = I(h_1) + E(h_1) = I(h_2) + E(h_2)$

with $E(h_i) = \mathcal{O}(h_i^2)$

Assuming $\bar{f}''$ constant regardless of step size (h),

$\frac{E(h_1)}{E(h_2)} \simeq \left(\frac{h_1}{h_2}\right)^2 \ \rightarrow\ E(h_2) = \frac{I(h_2)-I(h_1)}{\left(\frac{h_1}{h_2}\right)^2-1}$

It can be shown that the error is $\mathcal{O}(h^4)$

$$\boxed{I = I(h_2) + E(h_2) = I(h_2) + \frac{I(h_2)-I(h_1)}{\left(\frac{h_1}{h_2}\right)^2-1}}$$

Romberg integration (cont.)

Defining following indices:

k, level of integration

$k = 1$ trapezoidal rule, $\mathcal{O}(h^2)$

$k = 2$ differences, $\mathcal{O}(h^4)$

$k = 3$ differences, $\mathcal{O}(h^6)$

$\ldots$

j, integral accuracy level related to number of slices (h size)

$j = 1, 2, 3, \cdots$

$$\boxed{I_{j+1,k+1} = I_{j+1,k} + \frac{1}{4^k-1}\left(I_{j+1,k} - I_{j,k}\right)}$$

For example,

$I_{2,2} = I_{2,1} + \frac{1}{4-1}\left(I_{2,1} - I_{1,1}\right)$

$I_{3,2} = I_{3,1} + \frac{1}{4-1}\left(I_{3,1} - I_{2,1}\right)$

$$\begin{array}{cc}
\begin{array}{cccc} k=1 & k=2 & k=3 & k=n \end{array} & \\
\begin{pmatrix}
I_{1,1} & & & \\
I_{2,1} & I_{2,2} & & \\
I_{3,1} & I_{3,2} & I_{3,3} & \\
\vdots & \vdots & \vdots & \vdots \\
I_{n,1} & I_{n,2} & I_{n,3} & I_{n,n}
\end{pmatrix} &
\begin{array}{c} (j=1) \\ (j=2) \\ (j=3) \\ \\ (j=n) \end{array}
\end{array}$$

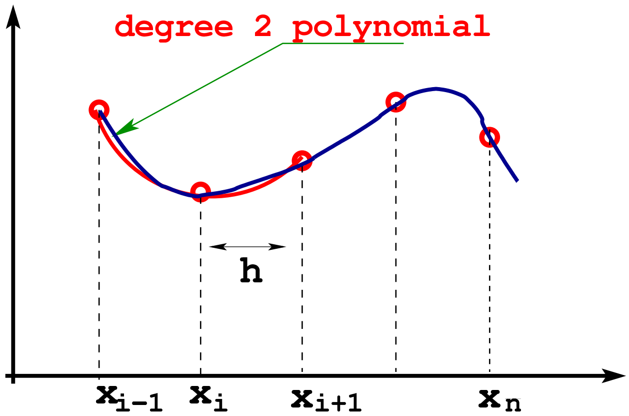
Within same level of integration, integral estimation is improving (check variation)

Next level of integration is better (check)

Optimal value will be $I_{n,n}$

Simpson rule

- Making $n = 2$ in Newton-Cotes formula is equivalent to use a degree 2 polynomial approximation for describing the function $f(x)$
- This method requires segments defined by **pairs of slices** in order to have the polynomial defined (adjacent slices)



- The result is that the **number of slices has to be even**. The integral for a pair of slices made with the three points $[x_{i-1}, x_i, x_{i+1}]$

$$F_i = \int_{x_{i-1}}^{x_{i+1}} f(x)\,dx \simeq \frac{h}{3}\left[f(x_{i-1}) + 4f(x_i) + f(x_{i+1})\right]$$

Simpson rule (cont.)

- For an integration range $[a, b]$, we divide it in n intervals (even) of width $h = \frac{b-a}{n}$,

$$\begin{aligned} F &= \int_a^b f(x)\,dx \simeq \sum_{i=1,3,5,\cdots}^{n} \left[\int_{x_{i-1}}^{x_{i+1}} f(x)dx\right] \\ &= \frac{h}{3}\left[f(x_0) + 4f(x_1) + 2f(x_2) + 4f(x_3) + \cdots + 2f(x_{n-2}) + 4f(x_{n-1}) + f(x_n)\right] \end{aligned}$$

- Error:

$$\Delta F = \frac{(b-a)h^4}{180} f^{(4)}(\chi)$$

Simpson rule requires that the number of slices n shall be even. If this is not the case, we can integrate over the $n-1$ slices with Simpson method and integrate the last slice using a degree 2 polynomial built from $[x_{n-2}, x_{n-1}, x_n]$

$$\int_{x_n-h}^{x_n} f(x)dx = \frac{h}{12}\left(-f_{n-2} + 8f_{n-1} + 5f_n\right)$$

Integration errors: step size

Aiming at obtaining an accuracy ε

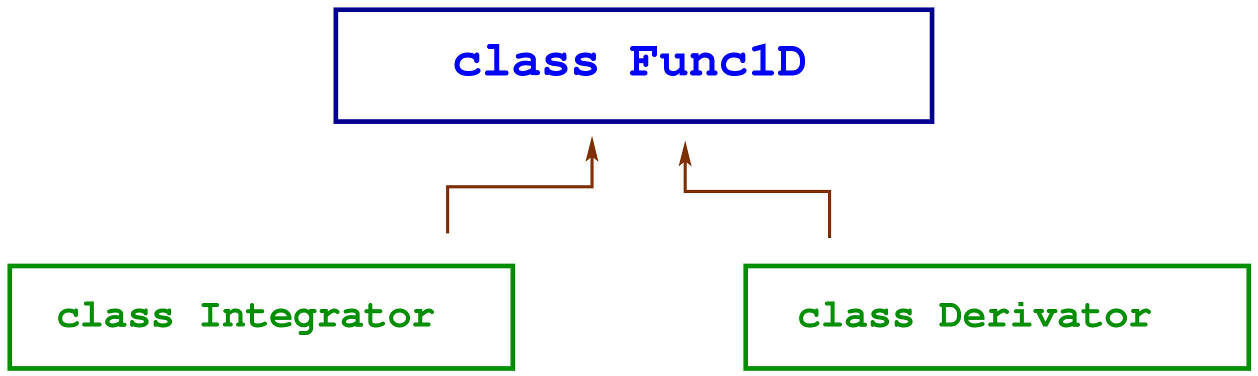
trapezoidal rule

$$\Delta F = \frac{h^2}{12}(b-a)M_{(2)} = \frac{(b-a)^3}{12}\frac{M_{(2)}}{n^2} < \varepsilon \Rightarrow \boxed{n^2 > \frac{1}{\varepsilon}\frac{M_{(2)}}{12}(b-a)^3}$$

simpson rule

$$\Delta F = \frac{h^4}{180}(b-a)M_{(4)} = \frac{(b-a)^5}{180}\frac{M_{(4)}}{n^4} < \varepsilon \Rightarrow \boxed{n^4 > \frac{1}{\varepsilon}\frac{M_{(4)}}{180}(b-a)^5}$$

C++ classes




```
class Func1D {
public:
 Func1D(TF1 *fp=NULL);
 // other constructors?
 ~Func1D();
 void Draw();
 double Evaluate();
protected:
 TF1 *p;
};
```

```
class Integrator: public Func1D {
public:
 Integrator(double fx0, double fx1, TF1 *fp=NULL) :
           x0(fx0), x1(fx1), Func1D(fp) {;}
 ~Integrator();
 void TrapezoidalRule(int n, double& Integral, double& Error);
 void SimpsonRule(int n, double& Integral, double& Error);
protected:
 double x0;
 double x1;
};
```

Computational Physics
Monte-Carlo methods

Fernando Barao, Phys Department IST (Lisbon)

Monte-Carlo methods

- any method using random variables for a numerical calculation
  - ☞ we ask for a statistical answer!
- founding article:
  "The monte carlo method", N. Metropolis, S. Ulam (1949)
- applications: physics, engineering, finance, ...
- aims of the method:
  - generate samples of random variables ($\vec{X}$) according to a density probability distribution $p(\vec{X})$
  - estimate expectation values ($<>$) of variables or functions

Monte-Carlo methods

JOURNAL OF THE AMERICAN STATISTICAL ASSOCIATION

Number 247 SEPTEMBER 1949 *Volume 44*

THE MONTE CARLO METHOD

NICHOLAS METROPOLIS AND S. ULAM
Los Alamos Laboratory

We shall present here the motivation and a general description of a method dealing with a class of problems in mathematical physics. The method is, essentially, a statistical approach to the study of differential equations, or more generally, of integro-differential equations that occur in various branches of the natural sciences.

ALREADY in the nineteenth century a sharp distinction began to appear between two different mathematical methods of treating

Statistical concepts

✔ the expected value of a variable X sampled N times $(X_1, X_2, \cdots, X_N)$

$$E(X) = \langle X \rangle = \frac{1}{N}\sum_{i=1}^{N} X_i$$

✔ the variance of the sample:

$$Var(X) \equiv \sigma_X^2 \simeq \frac{1}{N}\sum_{i=1}^{N}(X_i - \langle X \rangle)^2 = \left\langle (X - \langle X \rangle)^2 \right\rangle = \left\langle X^2 \right\rangle - \langle X \rangle^2$$

✔ the standard deviation of the sample:

$$\sigma_X \simeq \sqrt{\frac{1}{N}\sum_{i=1}^{N}(X_i - \langle X \rangle)^2}$$

PDFs - prob density distributions

- PDFs: the probability density function $p(X)$ give us the probability of an event (a value X_i in this case) to occur

$$\int_{-\infty}^{+\infty} p(X)\, dX = 1$$

- for a discrete variable X, its expectation value is given by:

$$\langle X \rangle = \frac{1}{N} \sum_{i=1}^{N} p(X_i)\, X_i$$

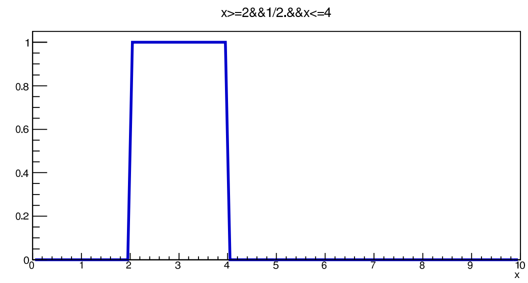
- for a continuous variable X or function $f(X)$, the expectation value is given by:

$$\langle X \rangle = \int_{-\infty}^{+\infty} p(X)\, X\, dX$$
$$\langle f \rangle = \int_{-\infty}^{+\infty} p(X)\, f(X)\, dX$$

Important PDFs

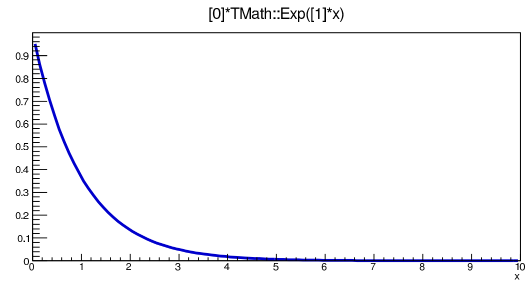
- uniform distribution: $X[a, b]$

$$p(X) = \frac{1}{b-a} H(X-a)H(b-X)$$



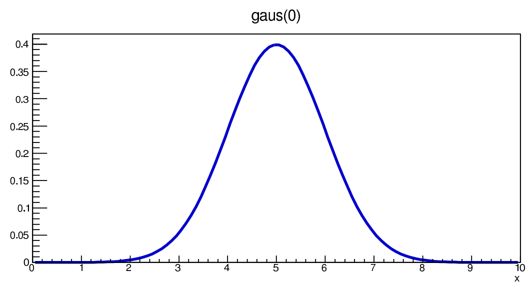
- exponential distribution: $X[0, \infty]$

$$p(X) = \alpha e^{-\alpha X}$$

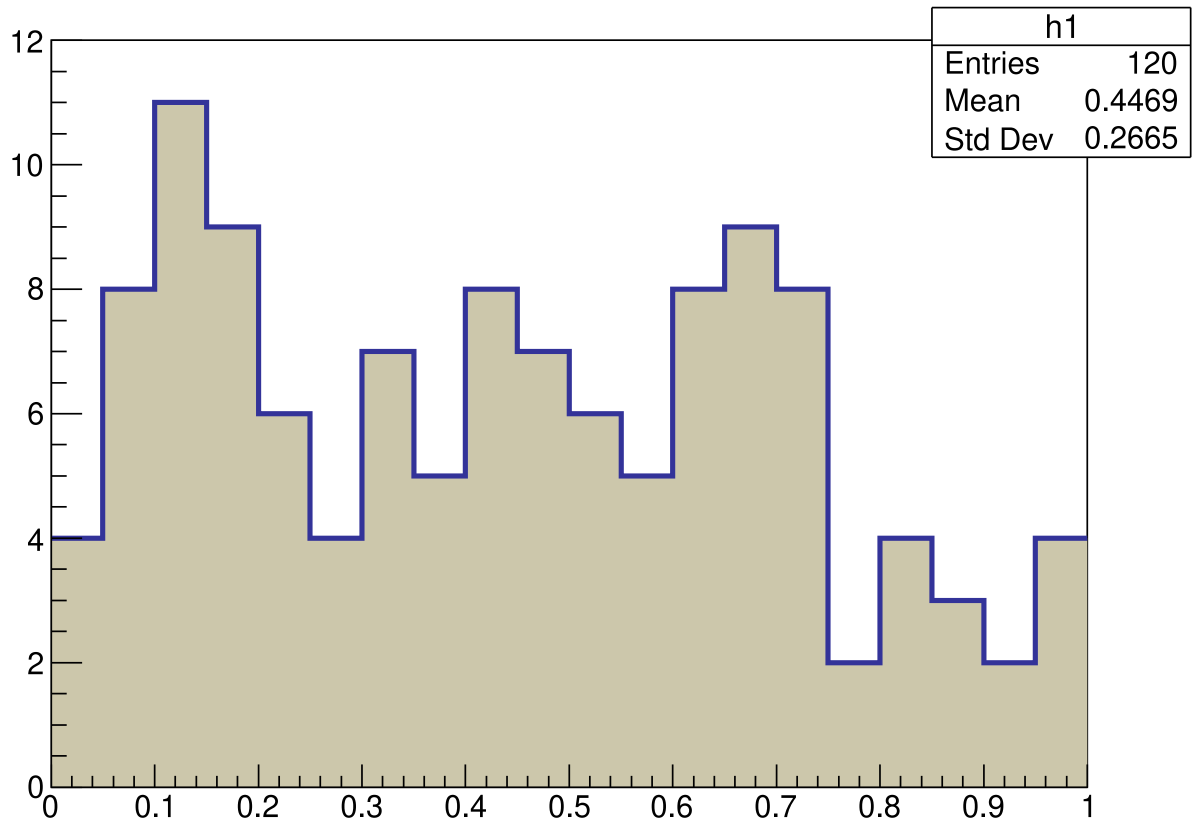


- normal distribution: $X[-\infty, +\infty]$

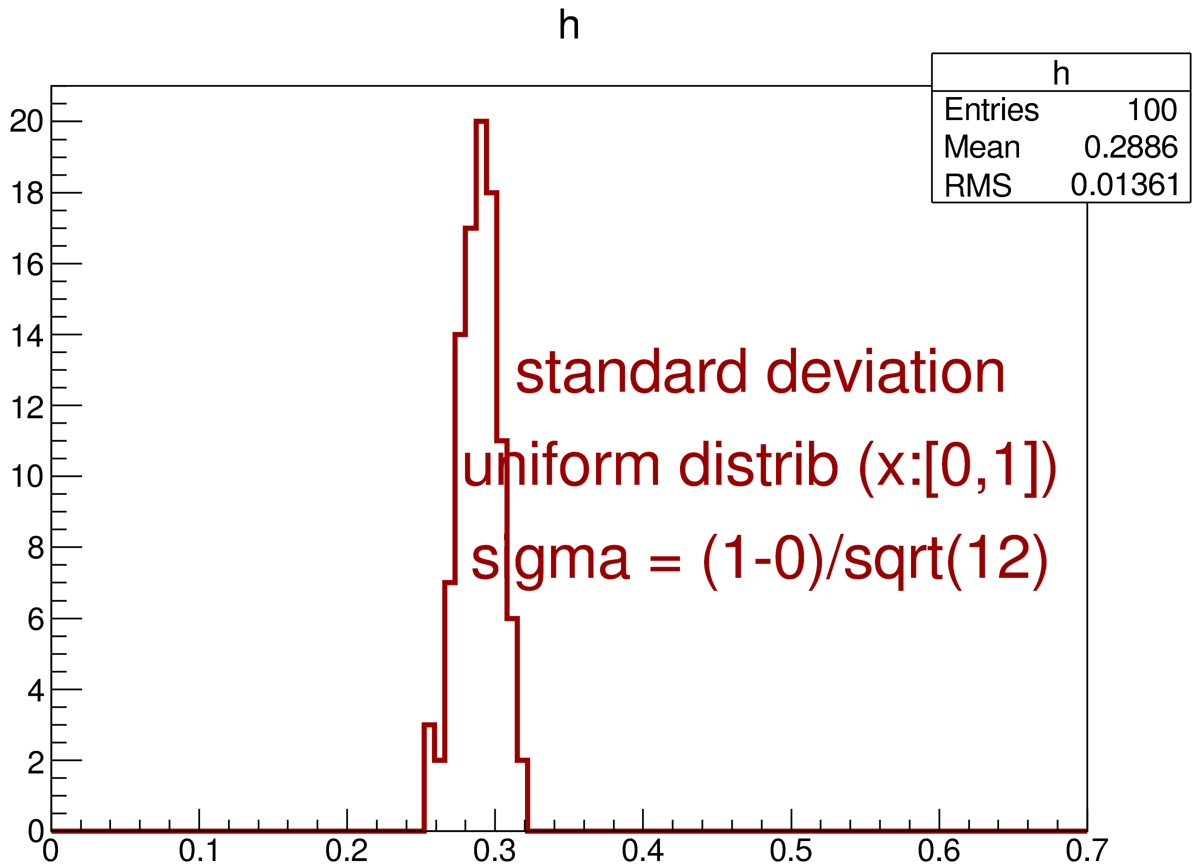
$$p(X) = \frac{1}{\sqrt{2\pi}\sigma}\, e^{-\frac{1}{2}\left(\frac{X-\mu}{\sigma}\right)^2}$$



uniform distribution



uniform dist: sigma



60-1

60-2