Numerical integration - I

M. Peressi - UniTS - Laurea Magistrale in Physics
Laboratory of Computational Physics - Unit V

- deterministic methods in 1D
  equispaced points (trapezoidal, Simpson...), others...
- Monte Carlo methods
  (acceptance-rejection, sample mean, importance sampling...)

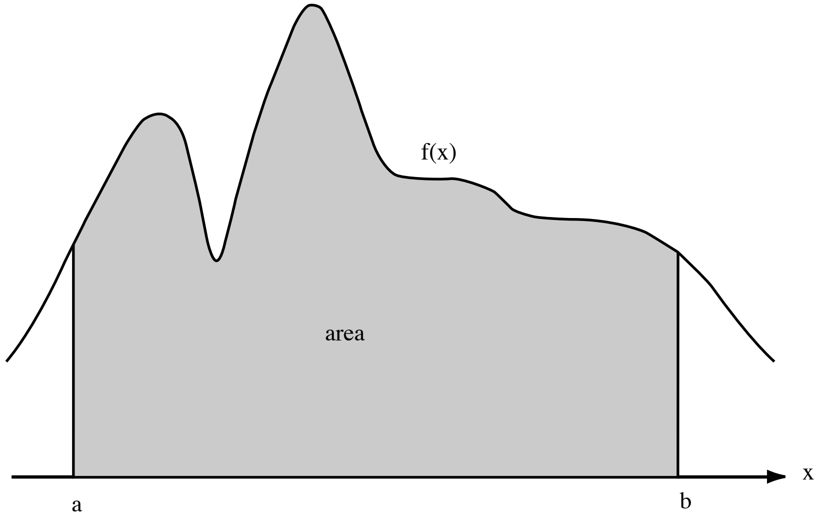
  Error handling:
  sample mean
  block average
  reduction of the variance

Deterministic methods

Deterministic methods

Start from the geometrical interpretation of a definite integral:

$$F = \int_a^b f(x)dx$$



Divide the integration interval into "small" intervals:

$$\Delta x = \frac{b-a}{n},$$

$$x_n = x_0 + n\,\Delta x.$$

(Note: n intervals ⇔ n +1 points)

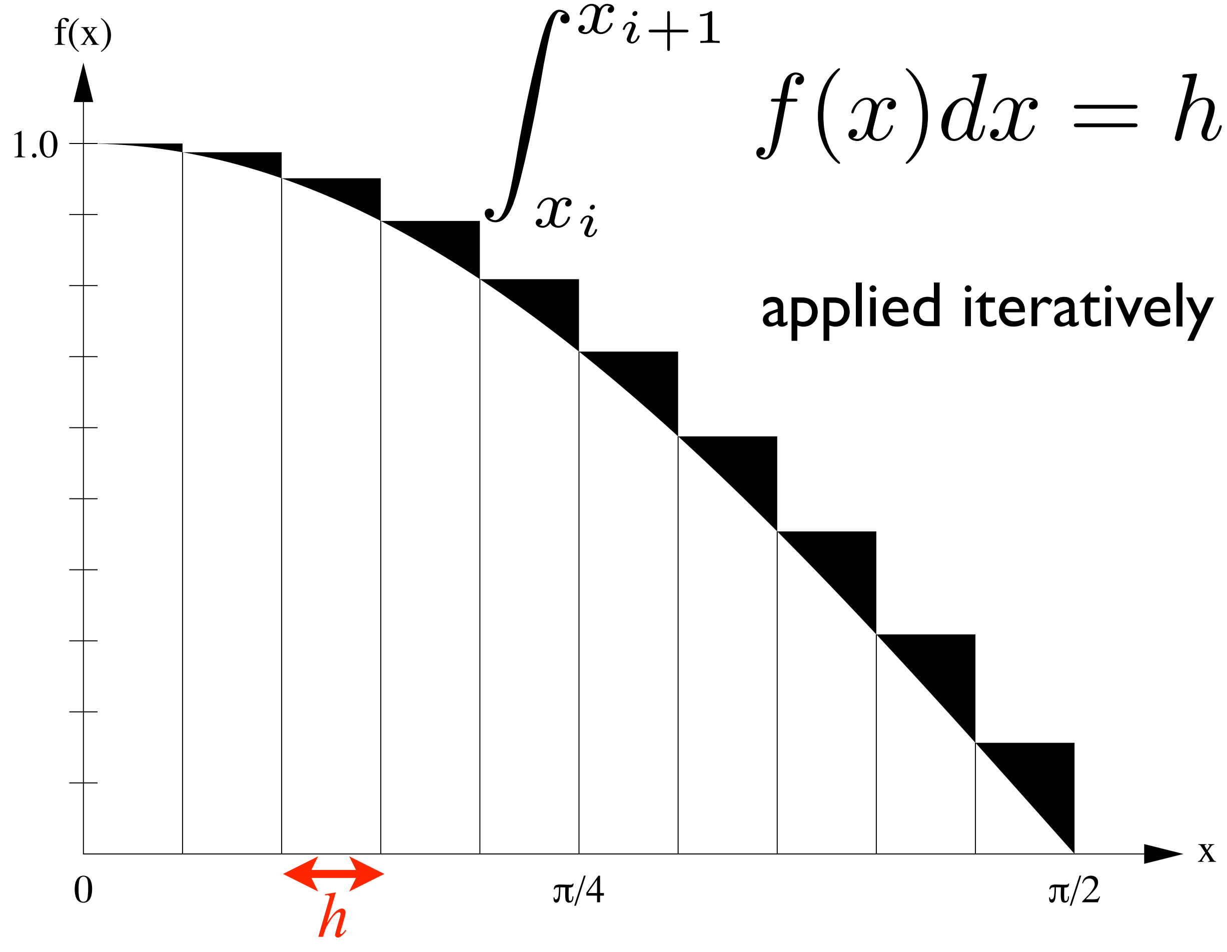
Deterministic methods:
rectangular rule

In one interval:

$$\int_{x_i}^{x_{i+1}} f(x)dx = hf_i$$

with error:

$$\mathcal{O}(h^2 f'), \propto 1/n^2$$



: The rectangular approximation for $f(x) = \cos x$ for $0 \leq x \leq \pi/2$.

applied iteratively over consecutive intervals:

$$F_n = \sum_{i=0}^{n-1} f(x_i)\Delta x.$$

with a total error:

$$\mathcal{O}(hf'), \propto 1/n$$

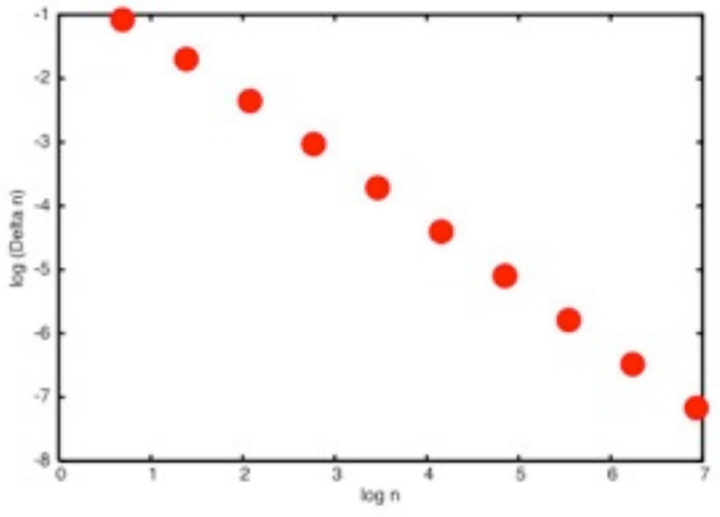
Deterministic methods: rectangular rule - error

$$I = \int_0^{\pi/2} \cos(x)dx = 1$$

$$F_n = \frac{\pi}{2n}\sum_0^{n-1} \cos x_i; \quad x_i = i\frac{\pi}{2n}$$

$$\Delta_n = F_n - I$$

n	F_n	Δ_n
2	1.34076	0.34076
4	1.18347	0.18347
8	1.09496	0.09496
16	1.04828	0.04828
32	1.02434	0.02434
64	1.01222	0.01222
128	1.00612	0.00612
256	1.00306	0.00306
512	1.00153	0.00153
1024	1.00077	0.00077



Rectangular approximations of the integral of $\cos x$ from $x = 0$ to $x = \pi/2$ as a function of n, the number of intervals. The error Δ_n is the difference between the rectangular approximation and the exact result of unity. Note that the error Δ_n decreases approximately as n^{-1}, that is, if n is increased by a factor of 2, Δ_n decreases by a factor 2.

Deterministic methods: generalities

- sum values of $f(x_i)$ with $x_i \in [a, b]$
- we want to have $F = \int_a^b f(x)dx$ as accurate as possible but with the minimum number of calculations of $f(x_i)$

OK simple algorithms, but if the number of calculations is too high, improve the algorithm!

Deterministic methods:

trapezoidal rule

In one interval:

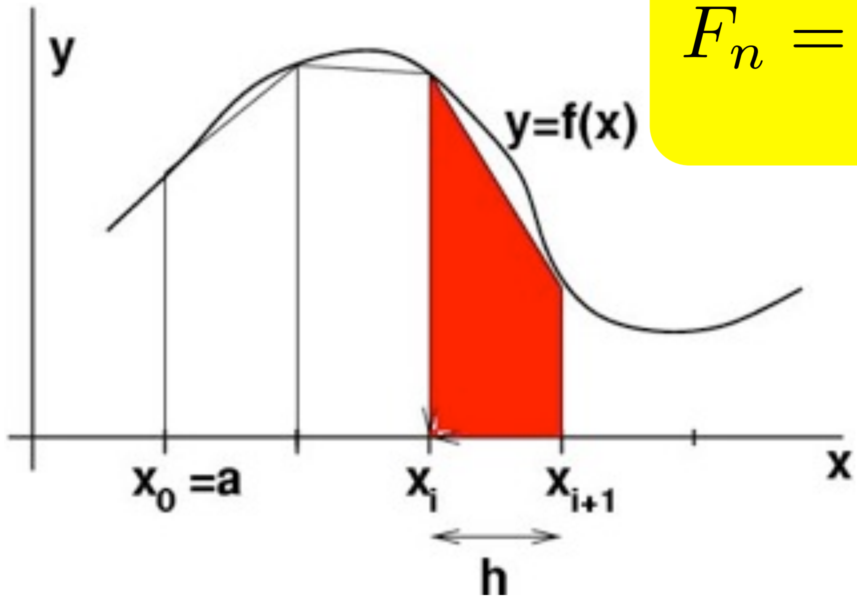
$$\int_{x_i}^{x_{i+1}} f(x)dx = h\left[\frac{1}{2}f_i + \frac{1}{2}f_{i+1}\right]$$

with error:

$$\mathcal{O}(h^3 f'''), \propto 1/n^3$$

Applied iteratively over consecutive intervals:

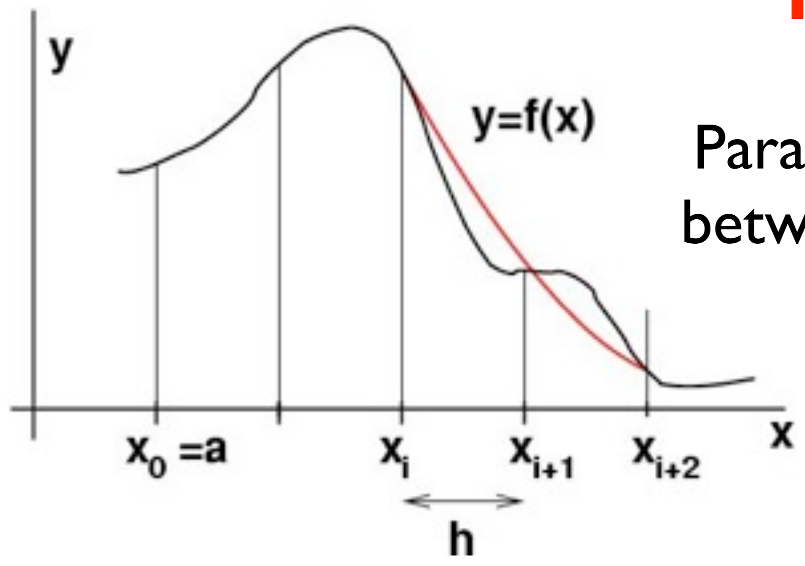
$$F_n = \left[\frac{1}{2}f(x_0) + \sum_{i=1}^{n-1} f(x_i) + \frac{1}{2}f(x_n)\right]\Delta x.$$



with a total error:

$$\mathcal{O}(h^2 f'''), \propto 1/n^2$$

Deterministic methods:
Simpson's rule



Parabolic interpolation procedure between triplets of adjacent points

In one interval:

$$\int_{x_i}^{x_{i+2}} f(x)dx = h\left[\frac{1}{3}f_i + \frac{4}{3}f_{i+1} + \frac{1}{3}f_{i+2}\right] + \mathcal{O}(h^5 f^{IV})\ (error\ \propto 1/n^5)$$

Iteratively applied to the whole interval of integration (odd number of points!):

$$\int_{x_0}^{x_n} f(x)dx = h\left[\frac{1}{3}f_0 + \frac{4}{3}f_1 + \frac{2}{3}f_2 + \frac{4}{3}f_3 + \ldots + \frac{2}{3}f_{n-2} + \frac{4}{3}f_{n-1} + \frac{1}{3}f_n\right] + \mathcal{O}(h^4 f^{IV})\ (error\ \propto 1/n^4)$$

Errors in deterministic methods

Error estimate for numerical integration with deterministic methods

$$\int f(x)dx = F_n + error$$

How to evaluate the error? Consider the Taylor expansion of the integrand function and then integrate:

$$f(x) = f(x_i) + f'(x_i)(x - x_i) + \frac{1}{2}f''(x_i)(x - x_i)^2 + \ldots, \quad \textbf{(*)}$$

$$\int_{x_i}^{x_{i+1}} f(x)\,dx = f(x_i)\Delta x + \frac{1}{2}f'(x_i)(\Delta x)^2 + \frac{1}{6}f''(x_i)(\Delta x)^3 + \ldots \textbf{(**)}$$

$$\Delta x \equiv x_{i+1} - x_i$$

Error estimate for numerical integration: Rectangular approximation

$$\int_{x_i}^{x_{i+1}} f(x)dx \approx f(x_i)\Delta x$$

Compare with (**):

$$\int_{x_i}^{x_{i+1}} f(x)\,dx = f(x_i)\Delta x + \frac{1}{2}\boxed{f'(x_i)(\Delta x)^2} + \frac{1}{6}f''(x_i)(\Delta x)^3 + \ldots$$

error

(leading order in Δx)

For n intervals $(\Delta x = (b-a)/n)$: error is $n(\Delta x)^2 \sim 1/n$

Error estimate for numerical integration: Trapezoidal approximation

$$\int_{x_i}^{x_{i+1}} f(x)dx \approx \frac{1}{2}\left[f(x_{i+1}) + f(x_i)\right]\Delta x$$

$$f(x_{i+1}) \approx f(x_i) + f'(x_i)\Delta x + \frac{1}{2}f''(x_i)\Delta x^2 + \ldots$$

$$\approx \frac{1}{2}\left[2f(x_i) + f'(x_i)\Delta x + \frac{1}{2}f''(x_i)\Delta x^2 + \ldots\right]\Delta x$$

Compare with (**):

$$\int_{x_i}^{x_{i+1}} f(x)\,dx = f(x_i)\Delta x + \frac{1}{2}f'(x_i)(\Delta x)^2 + \frac{1}{6}\underbrace{f''(x_i)(\Delta x)^3}_{\text{error}} + \ldots$$

(leading order in Δx)

For n intervals: error is $n(\Delta x)^3 \sim 1/n^2$

Error estimate for numerical integration: Simpson approximation

$$\int_{x_i}^{x_{i+2}} f(x)dx \approx \left[\frac{1}{3}f(x_i) + \frac{4}{3}f(x_{i+1}) + \frac{1}{3}f(x_{i+2})\right]\Delta x$$

.... (homework!)

$$f(x_{i+1}) \approx f(x_i) + f'(x_i)\Delta x + \frac{1}{2}f''(x_i)\Delta x + \ldots$$

....

Compare with (**):

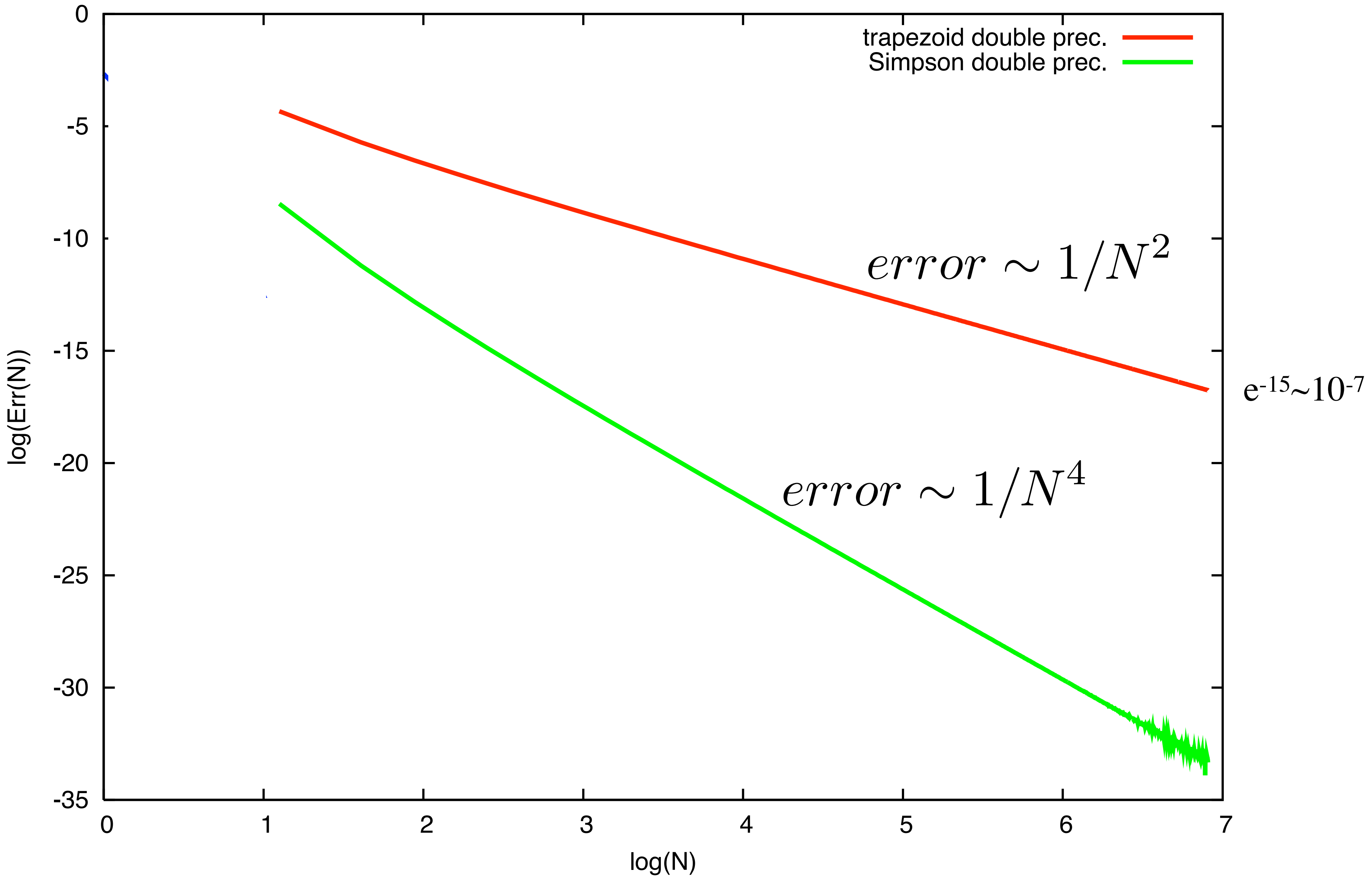
$$\int_{x_i}^{x_{i+2}} f(x)dx = f(x_i)\Delta x + \frac{1}{2!}f'(x_i)(\Delta x)^2 + \frac{1}{3!}f''(x_i)(\Delta x)^3 + \frac{1}{4!}f'''(x_i)(\Delta x)^4 + \frac{1}{5!}f''''(x_i)(\Delta x)^5 + \ldots$$

error

(leading order in Δx)

For n intervals: error is $n(\Delta x)^5 \sim 1/n^4$

Numerical integration - deterministic methods: comparison of errors in 1D



Deterministic methods - I

We use a piecewise polynomial interpolation:

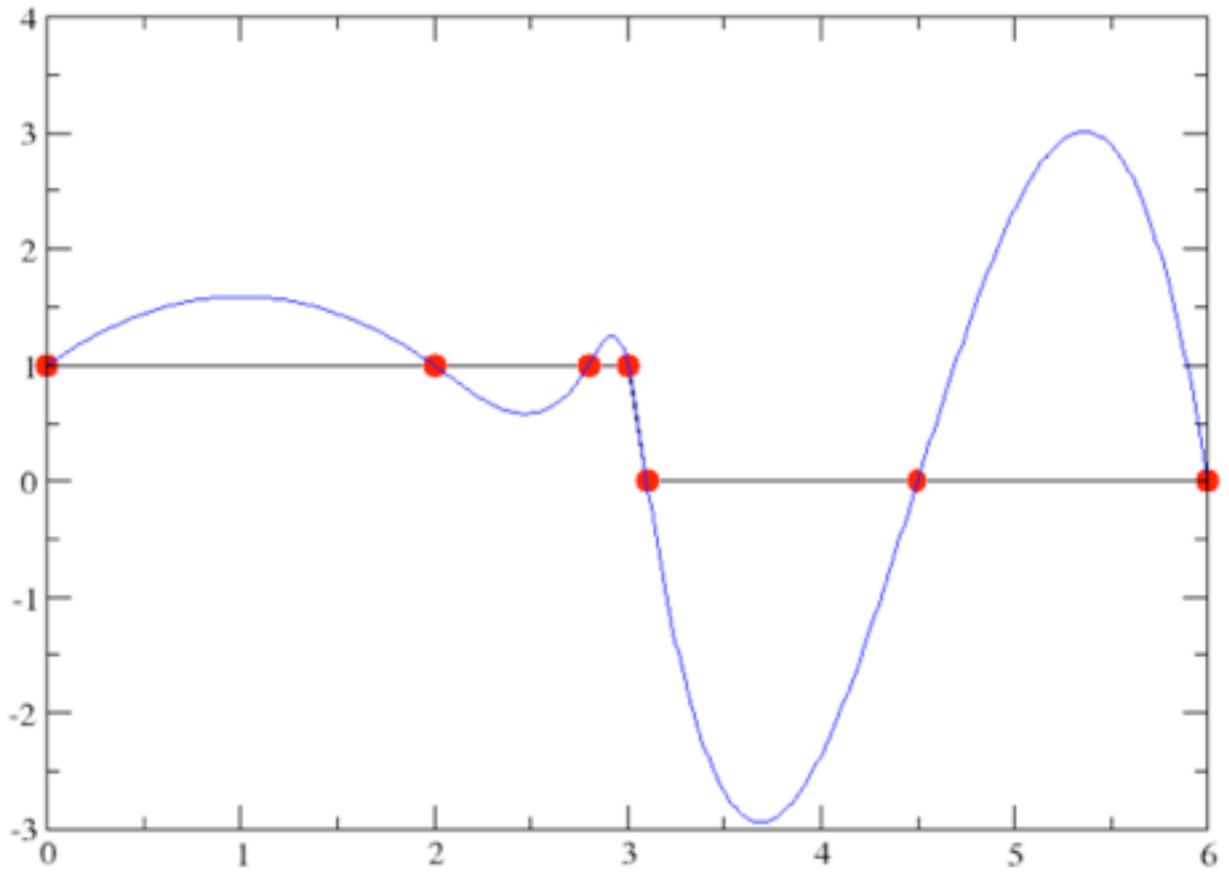
- constant interpolation $\Rightarrow$ 1 point $\Rightarrow$ rectangular rule
- linear interpolation $\Rightarrow$ 2 points $\Rightarrow$ trapezoidal rule
- parabolic interpolation $\Rightarrow$ 3 points $\Rightarrow$ Simpson's rule
- ... .
- higher-order polynomial $\Rightarrow$ many points $\Rightarrow$

NOT CONVENIENT!

Warning: using higher degrees does not always improve accuracy!

(see also: Runge phenomenon (polynomial interpolation, oscillation at the edges of an interval), Gibbs phenomenon ...)

Deterministic methods - II



●$(x_i,f(x_i))$

Warning:
using high-order piecewise polynomial interpolation: possible strong oscillations between consecutive $(x_i,f(x_i))$, giving a bad interpolation of f(x).
Here: f(x) step function; - linear interp.; - cubic spline

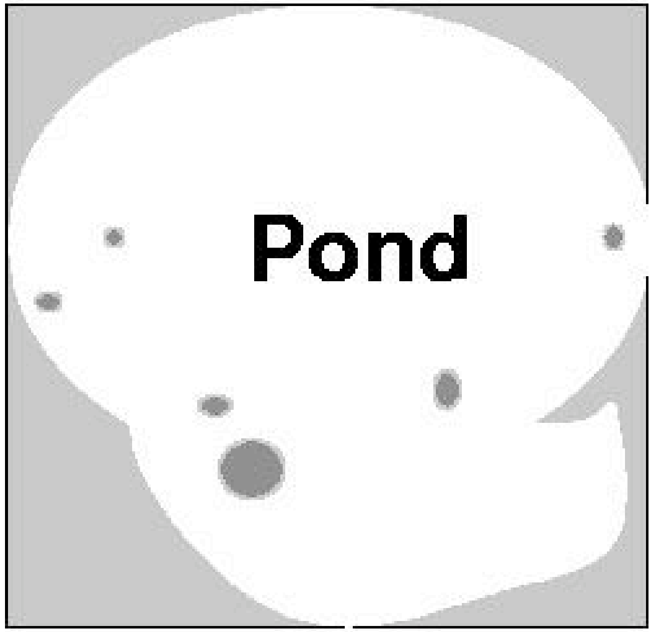
Monte Carlo methods

Monte Carlo methods:

"acceptance-rejection" or "hit or miss"

(to calculate areas)

which is A_{pond} ?



- enclose the pond in a box of Area A_{box}
- throw pebbles uniformly and randomly in the box
- count the number of pebbles felt in the pond with respect to the number felt in the box
- Assuming a uniform distribution, the number of pebbles falling into the ponds is proportional to the area of the pond:

$$\frac{N_{pond}}{N_{pond}+N_{box}} = \frac{A_{pond}}{A_{box}}$$

$$\Rightarrow \quad A_{pond} = \frac{N_{pond}}{N_{pond}+N_{box}} A_{box}$$

Monte Carlo methods:

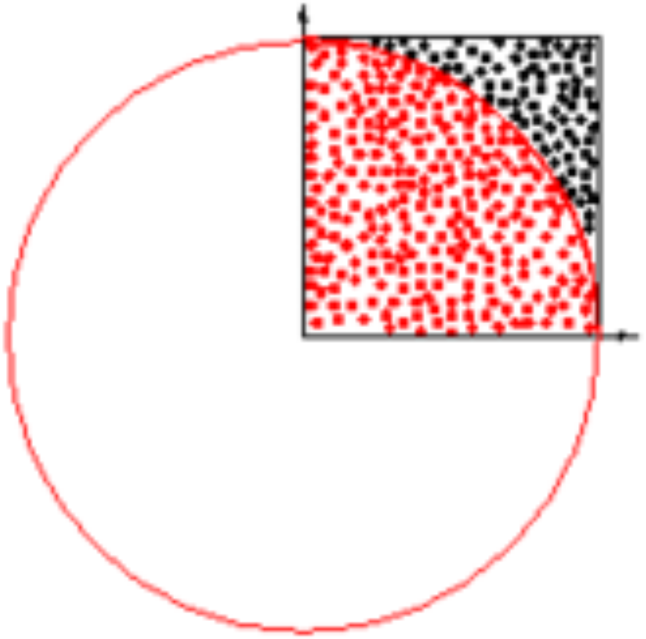
“acceptance-rejection” or “hit or miss”

(to calculate areas)

$$\boxed{\pi = ???}$$

N random points in the unit square

coordinates x_i, y_i

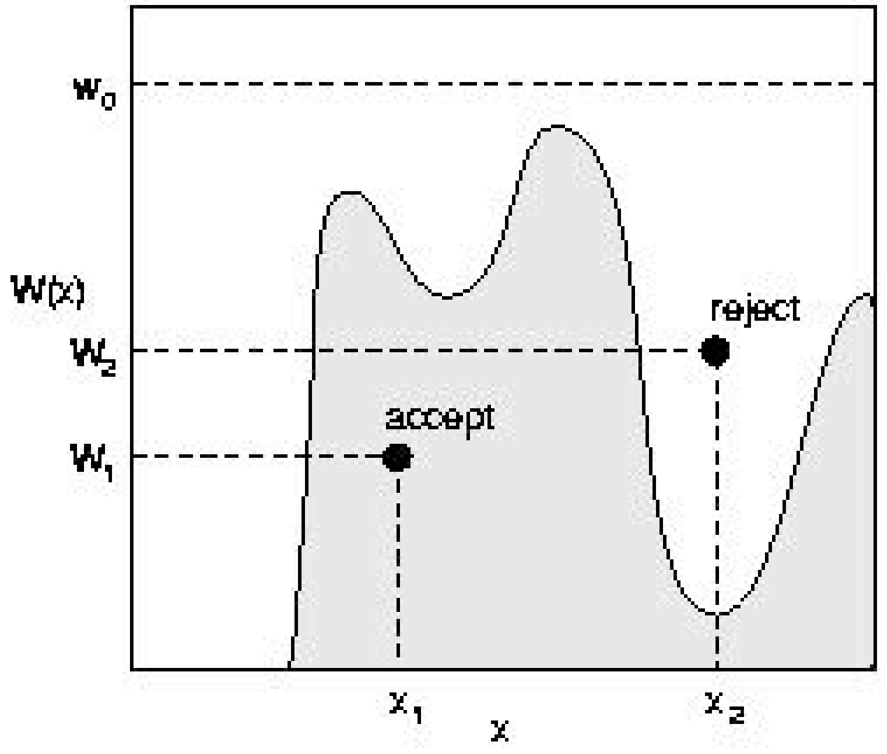


Then, the number of points N_c lying within the quarter circle (i.e. fulfilling the relation $x^2 + y^2 \leq 1$) is compared to the total number N of points and the fraction will give us an approximate value of π:

$$\pi(N) = 4\frac{N_c(N)}{N}$$

Monte Carlo methods:
“acceptance-rejection” or “hit or miss”
(to calculate definite integrals)

$$\int W(x)dx = \ ?$$



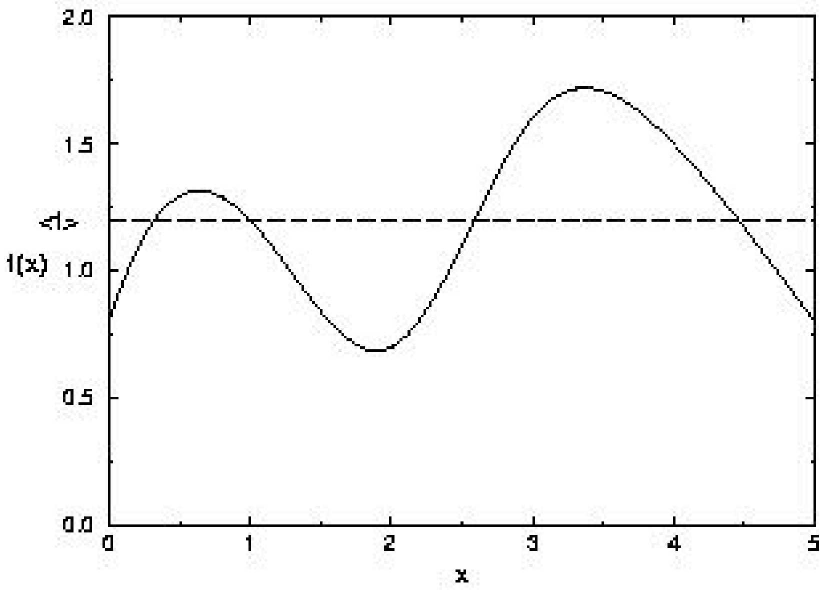
For W(x) positive in the integration interval, the value of the area under W(x) can be obtained by producing random points (i.e. (x,y) random pairs) uniformly distributed in a rectangle containing W(x).

For each point (x,y) compare y with W(x): if y<W(x), the point is accepted. The area under W(x) is the number of points accepted divided by the total number of points generated and multiplied by the area of the rectangle.

(remember: also used to generate random numbers x_i distributed according W(x))

Other simple
Monte Carlo methods

We can always write: $$I = \int_a^b f(x)dx = (b-a)\langle f \rangle$$

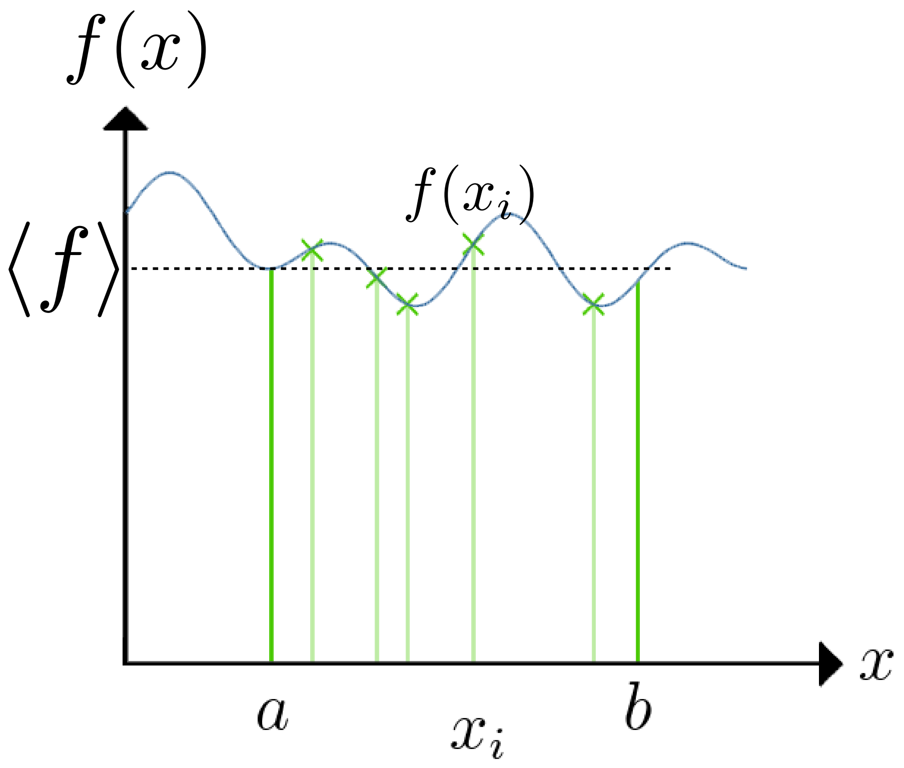


i.e., the value of the integral of f(x) between a and b equals the length of the interval (b-a) times the average value of the function <f> over the same interval.

(If f:[a,b] → R is a continuous function, then there exists a number c in [a,b] such that f(c)=<f> **(mean value theorem for integration)**)

how to estimate <f> efficiently and accurately?

A simple Monte Carlo method: "sample mean"



$$I = \int_a^b f(x)dx = (b-a)\langle f \rangle$$

The sample mean can be calculated by sampling the function *(if smooth enough...)* with a sequence of N uniform random numbers in [a,b]:

$$\langle f \rangle \approx \frac{1}{N}\sum_{i=1}^{N} f(x_i)$$

$$\int_a^b f(x)dx \approx (b-a)\frac{1}{N}\sum_{i=1}^{N} f(x_i) = (b-a)\langle f \rangle$$

Monte Carlo methods:
error estimate

Example: MC estimate of π (exact value known)

We can use either acceptance-rejection or sample mean method: $I = 4\int_0^1 \sqrt{1-x^2} = \pi = 3.1416\ldots$

Since we know the "exact" result I, we can calculate the **error** in two ways:

1) the **actual error** from the difference with respect to the exact value:

$$\Delta_n = |F_n - I|$$ with $F_n = (b-a)\frac{1}{n}\sum_{i=1}^{n} f(x_i), \quad x_i \text{ random}$

2) the numerical error from the **variance of the data** $\{f(x_i)\}$:

$$\sigma^2 = \langle f^2 \rangle - \langle f \rangle^2,$$

where

$$\langle f \rangle = \frac{1}{n}\sum_{i=1}^{n} f(x_i) \quad \text{and} \quad \langle f^2 \rangle = \frac{1}{n}\sum_{i=1}^{n} f(x_i)^2$$

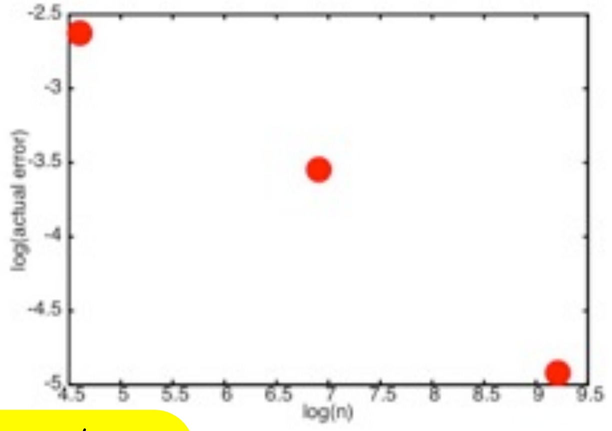
Monte Carlo methods:

error estimate

Results:

$$I = 4\int_0^1 \sqrt{1-x^2} = \pi = 3.1416\ldots$$

n	F_n	actual error	σ_n
10^2	3.0692	0.0724	0.8550
10^3	3.1704	0.0288	0.8790
10^4	3.1489	0.0073	0.8850



1) the actual error Δ_n decreases as $1/n^{1/2}$

2) the numerical error from the variance of the data, σ_n, is roughly constant and is much larger than the actual error

what is the correct error estimate?

Monte Carlo methods:
error estimate

...typically you do not know which is the "actual error" (you do not know the "true" value and you cannot compare your result with that!)....
but we would like to give an error to our numerical estimate...
(to which extent is our numerical estimate reliable?)

Two methods to estimate the error numerically from the variance of the data (**"reduction of variance"**):

I) average of the averages

II) block average

MC error handling: method I "average of the averages"

make additional runs of n trials each.
Let M_α be the average of each run :

run α	M_α	actual error
1	3.1489	0.0073
2	3.1326	0.0090
3	3.1404	0.0012
4	3.1460	0.0044
5	3.1526	0.0110
6	3.1397	0.0019
7	3.1311	0.0105
8	3.1358	0.0058
9	3.1344	0.0072
10	3.1405	0.0011

one run $\equiv n = 10^4$ *trials each*

Examples of Monte Carlo measurements of the mean value of $f(x) = 4\sqrt{1-x^2}$ in the interval $[0,1]$. A total of 10 measurements of $n = 10^4$ trials each were made. The mean value M_α and the actual error $|M_\alpha - \pi|$ for each measurement are shown.

Calculate: $\sigma_m{}^2 = \langle M^2 \rangle - \langle M \rangle^2$ with $\langle M \rangle = \frac{1}{m}\sum_{\alpha=1}^{m} M_\alpha$, $\langle M^2 \rangle = \frac{1}{m}\sum_{\alpha=1}^{m} M_\alpha{}^2$

$\Longrightarrow \sigma_m = 0.0068$

σ_m is consistent with the results for the actual errors

MC error handling: method II "block averages"

Instead of doing additional measurements, divide them into "s SUBSETS" and let S_k be the average within each subset :

subset k	S_k
1	3.14326
2	3.15633
3	3.10940
4	3.15337
5	3.15352
6	3.11506
7	3.17989
8	3.12398
9	3.17565
10	3.17878

The variance associated to the average of the subsets $\sigma_s{}^2 = \langle S^2 \rangle - \langle S \rangle^2$

gives $\sigma_s = 0.025$, but

$\sigma_s/\sqrt{s}$, which for our example is approximately $0.025/\sqrt(10) \approx 0.008$.

is consistent with the actual error

Monte Carlo methods:
error estimate - variance reduction
summary

$$\sigma_n/\sqrt{n} \approx \sigma_m \approx \sigma_s/\sqrt{s}$$

from the variance of the whole set of data

the variance of the **average of the averages**

from the variance of the **block averages**

Note: for uncorrelated data !

the most convenient!
but: change block size
and check that
it does not change

(proof)

Monte Carlo methods:
summary

We have introduced :

* "acceptance-rejection"

* "sample mean" to estimate $\langle f \rangle \approx \frac{1}{N} \sum_{i=1}^{N} f(x_i)$

both OK for smoothly varying functions, but
not very efficient for rapidly varying functions

How to improve the efficiency of MC integration?

A trick for numerical integration: “reduction of variance”

(Note: same word, but different meaning w.r.t. previous slides on error handling)

Given a function $f(x)$ to integrate, suppose that $g(x)$ exists, whose integral is known and such that:

$$|f(x) - g(x)| << \varepsilon$$

Therefore:

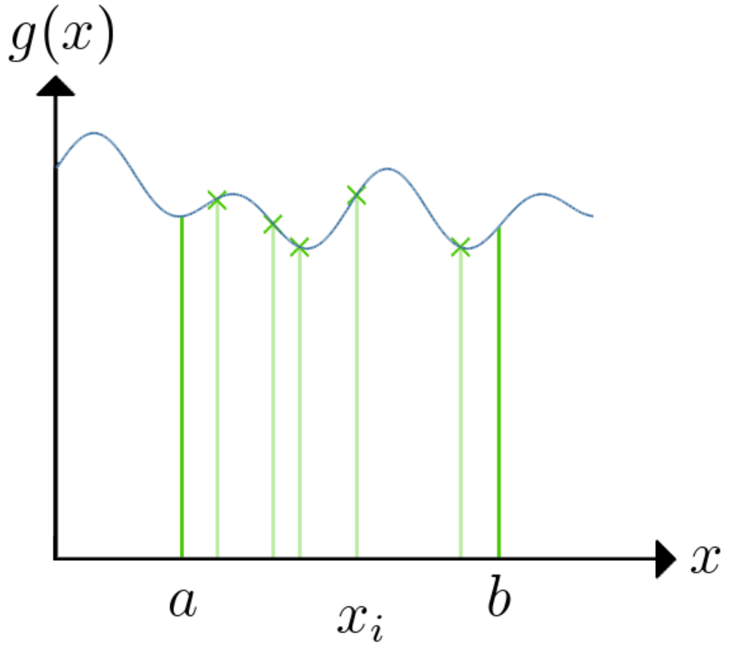
$$F = \int_a^b f(x)dx = \int_a^b \big((f(x) - g(x)) + g(x)\big)dx = \int (f(x) - g(x))\, dx + \int g(x)dx$$

$\uparrow$

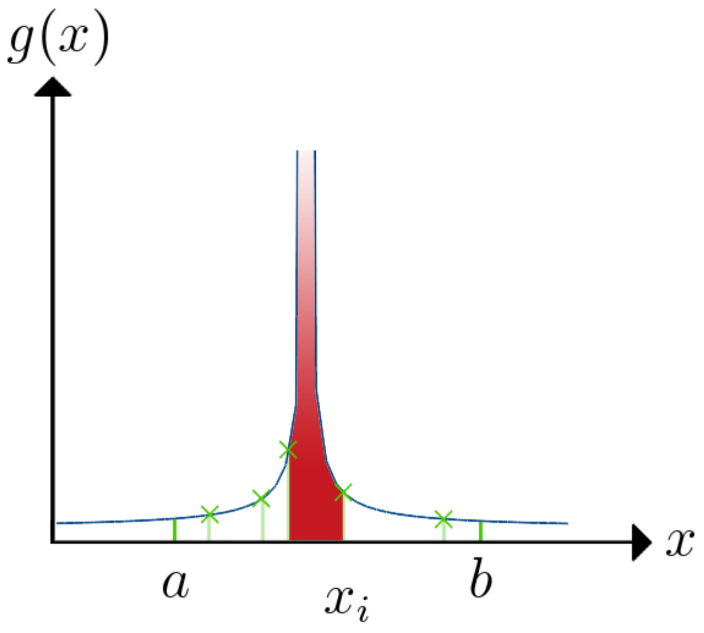
easy to calculate

Another simple Monte Carlo method: "importance sampling"

Mean value: easy to calculate for smoothly varying functions.
But not for functions rapidly varying.



smooth function



function with singularity

How to manage such cases?

Another simple Monte Carlo method: "importance sampling"

Mean value: easy to calculate for smoothly varying functions.

Idea: in order to calculate:

$$\langle f \rangle \approx \frac{1}{N}\sum_{i=1}^{N} f(x_i)$$

consider a distribution function $p(x)$ easy to integrate analytically and close to $f(x)$:

$$F = \int_a^b f(x)dx = \int_a^b \left[\frac{f(x)}{p(x)}\right] p(x)dx = \left\langle \frac{f(x)}{p(x)} \right\rangle \int_a^b p(x)dx$$

where $\left\langle \frac{f(x)}{p(x)} \right\rangle \approx \frac{1}{N}\sum_{i=1}^{N}\left[\frac{f(x_i)}{p(x_i)}\right]$

(particular case: uniform distrib. p(x)=1/(b-a) ...)

with $\{x_i\}$ distributed according to $p(x)$

Monte Carlo methods: "importance sampling"

Calculate:

$$F = \int_0^1 e^{-x^2}\, dx.$$

with "sample mean" with random numbers with uniform distribution or using the "importance sampling" with $p(x) = e^{-x}$

	$p(x) = 1$	$p(x) = Ae^{-x}$
n (trials)	4×10^5	8×10^3
F_n	0.7471	0.7469
σ	0.2010	0.0550
$\sigma/\sqrt{n}$	3×10^{-4}	6×10^{-4}
Total CPU time (s)	35	1.35
CPU time per trial (s)	10^{-4}	2×10^{-4}

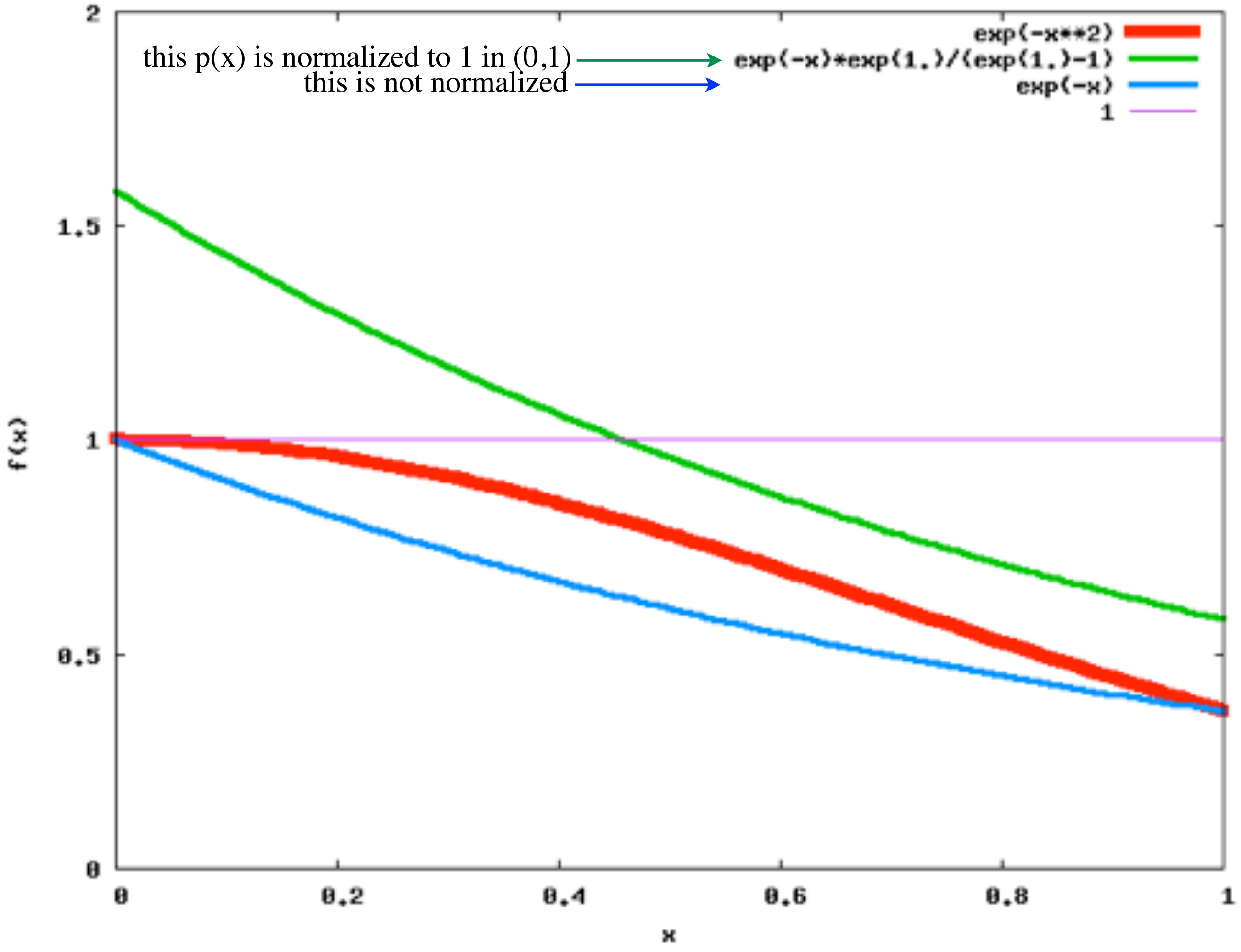
$\leftarrow$ efficient !

(pay attention to the normalization of p(x)...)

Choice of the importance sampling function
Ex. 2

$$\boxed{F = \int_0^1 e^{-x^2}\, dx.}$$

$$F = \int_a^b f(x)dx = \int_a^b \left[\frac{f(x)}{p(x)}\right] p(x)dx = \left\langle \frac{f(x)}{p(x)} \right\rangle \int_a^b p(x)dx$$



(pay attention to the normalization of p(x)...)

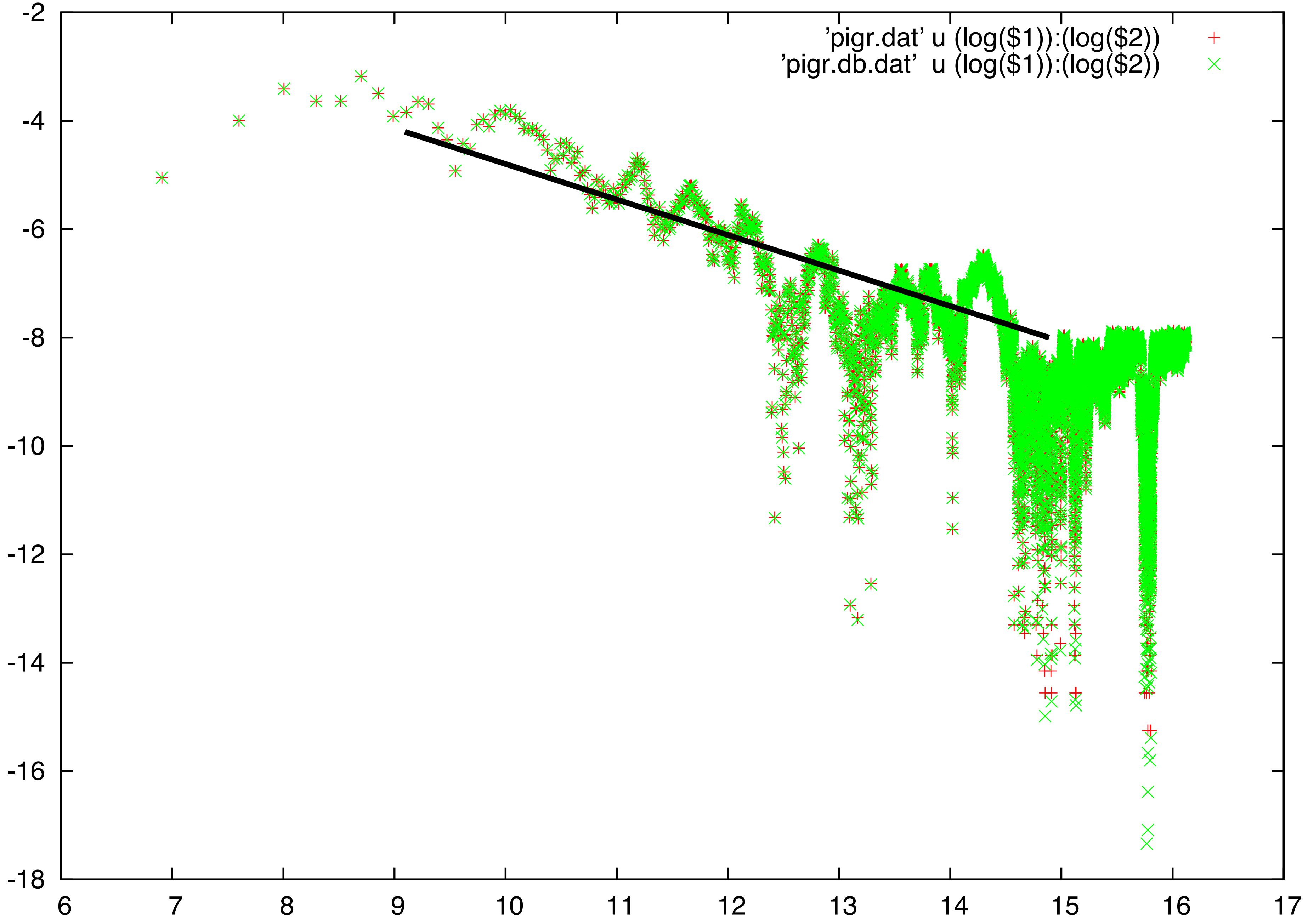
Some programs:

on
$/home/peressi/comp-phys/V-integr/f90
[do: $cp /home/peressi/.../f90/* .]
and on https://moodle2.units.it/

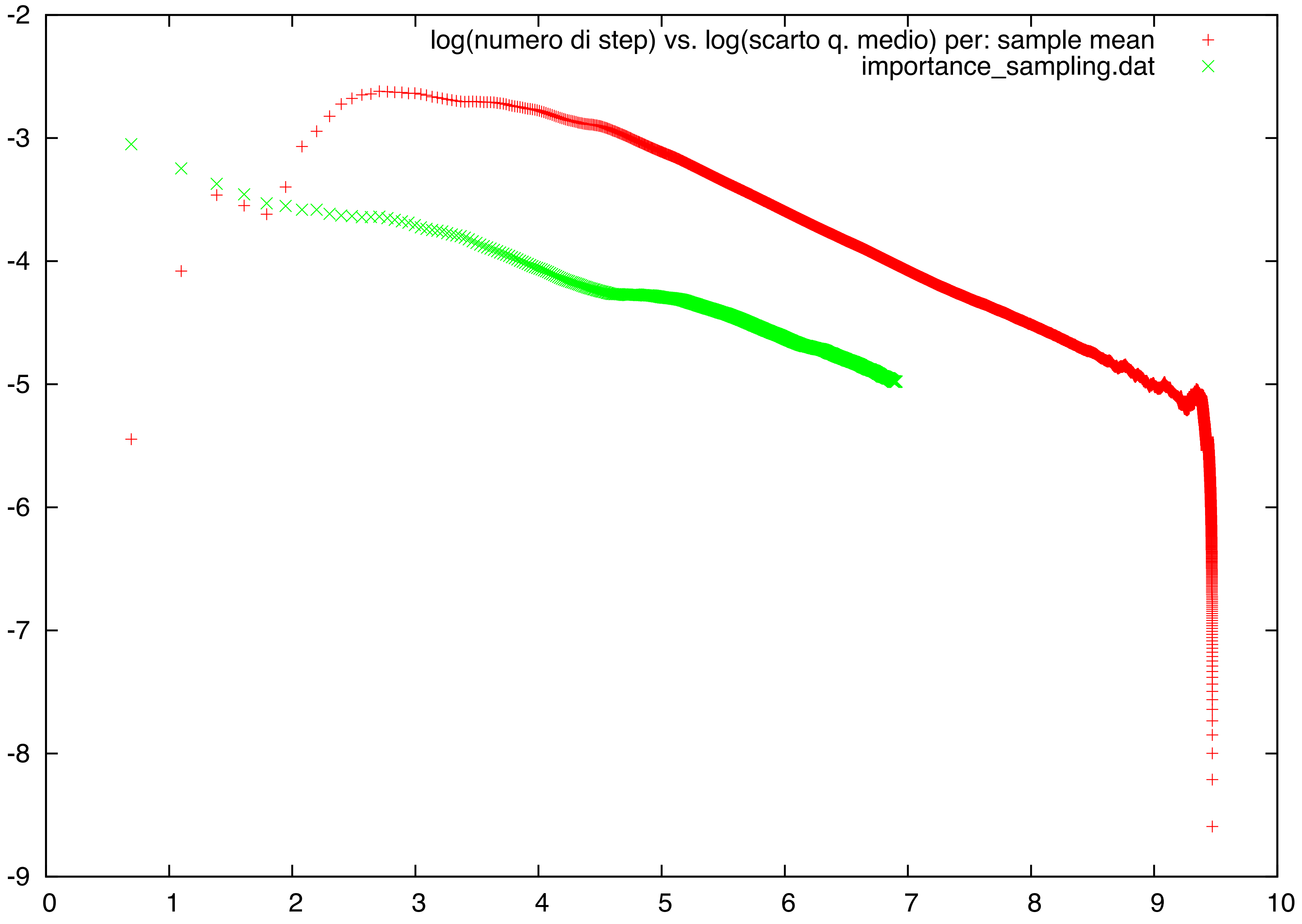
int.f90 (trapezoidal and Simpson integration) for Ex. 1
pi.f90 (Montecarlo integration for calculation of π) for Ex. 3

for the other exercises: write yourself the code!

error(MC)~1/$\sqrt{N}$ => see log(error) vs. log(N)



error(MC)~1/√N => see log(error) vs. log(N) but with different prefactors for sample means vs importance sampling


log(numero di step) vs. log(scarto q. medio) per: sample mean +
importance_sampling.dat ×