

The Monte Carlo Method

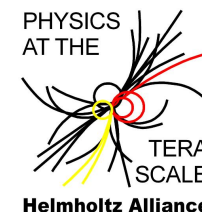
Basics and (simple) applications

Thanks to H.
Jung for a lot
of material !!!

Thomas Schörner-Sadenius



GEANT4 Training Event
DESY Hamburg, 45 March 2010



OUTLINE

- > The Monte Carlo method
 - Monte Carlo: What, why, and where?
- > Random numbers
 - True random versus pseudo-random ...
- > Some mathematical basics
- > Generating arbitrary distributions
 - Hit&miss, inversion of cumulative distribution, ...
- > MC integration
 - Hit&Miss, importance sampling, ...
- > Applications in HEP

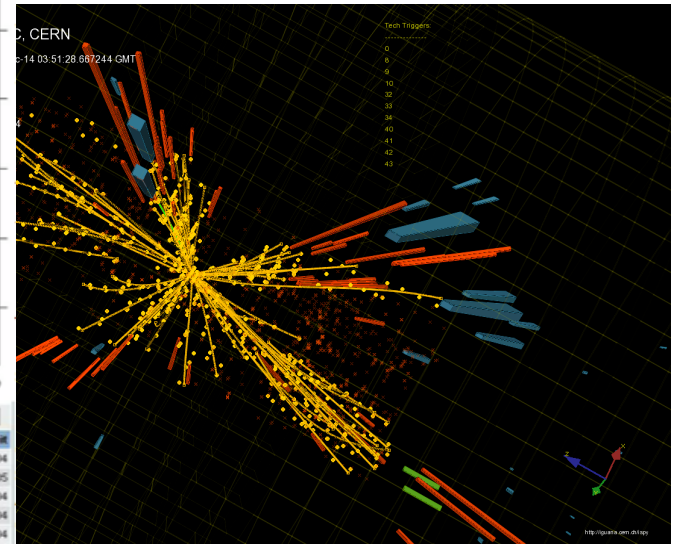
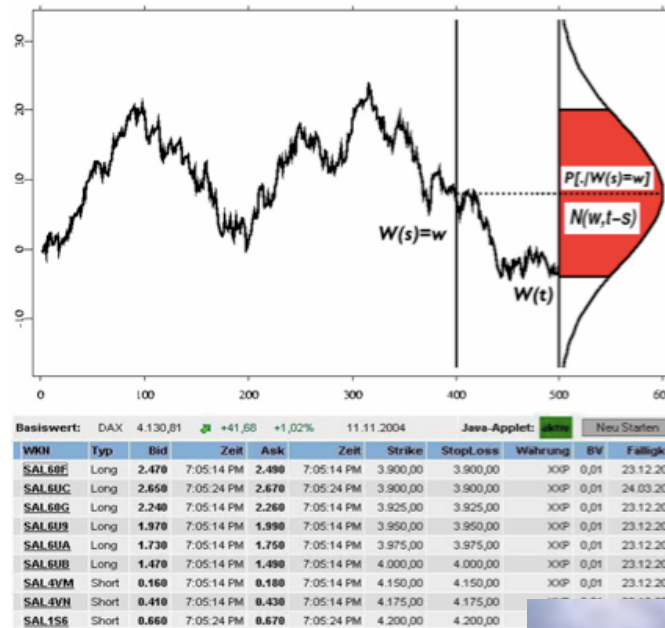
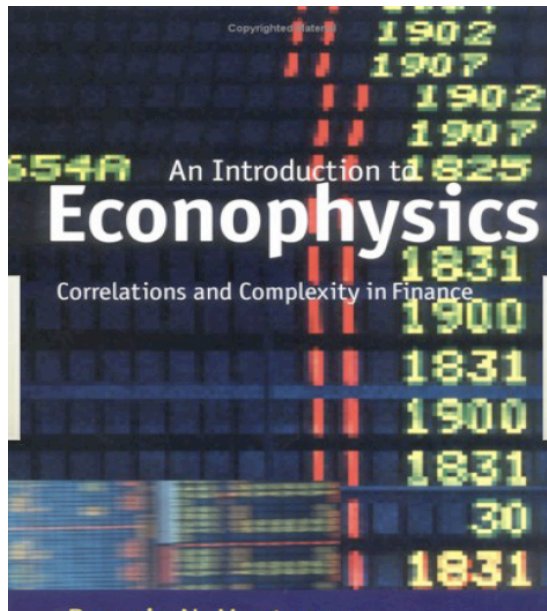
- > Monte Carlo methods ... a class of [computational algorithms](#) that rely on repeated [random](#) sampling to compute their results. Monte Carlo methods are often used when [simulating physical](#) and [mathematical](#) systems. Because of their reliance on repeated computation of [random](#) or [pseudo-random](#) numbers, these methods are most suited to calculation by a [computer](#) and tend to be used when it is infeasible or impossible to compute an exact result with a [deterministic algorithm](#).
- > So use MC when there is **no (known) analytical solution** to a (mathematical or physical) problem:
 - Difficult integrals, numerical analysis
 - Complex systems with many degrees of freedom, natural phenomena
 - Social or economical systems
 - Higher-order processes in particle physics
- > In the following, “MC method” denotes any algorithm that arrives, by the use of “random” numbers, at the solution of a problem.

> Monte Carlo methods:

- “ ... their methods (v. Neumann et al.) ... were aptly named after the international gaming destination ...”
- “ ... after the War a wide range of sticky problems yielded to the new techniques ...”
- “ ... virtually impossible to find a succinct definition of ‘Monte Carlo’ method ...”
- “ ... some authors prefer the term ‘stochastic simulation’ ...”
- “Monte Carlo is the art (sic!) of approximating an expectation by the **sample mean of a function of simulated random variables**.”
- “Monte Carlo is about **invoking laws of large numbers** to approximate expectations.”

MONTE CARLO METHOD: APPLICATIONS

Monte Carlo:
what, why, where?

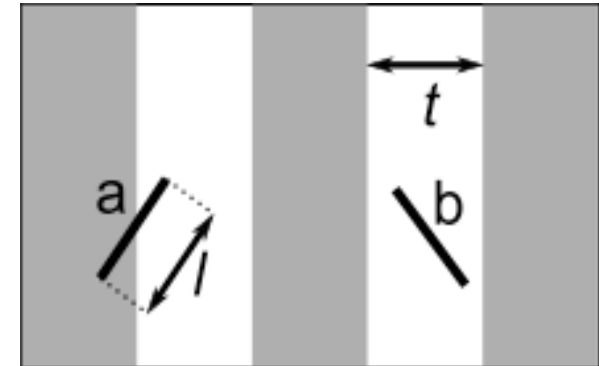



PARADIGMATIC EXAMPLES

Monte Carlo:
what, why, where?

> Buffon's needle: a question first posed in the 18th century by Georges-Louis Leclerc, Comte de Buffon:

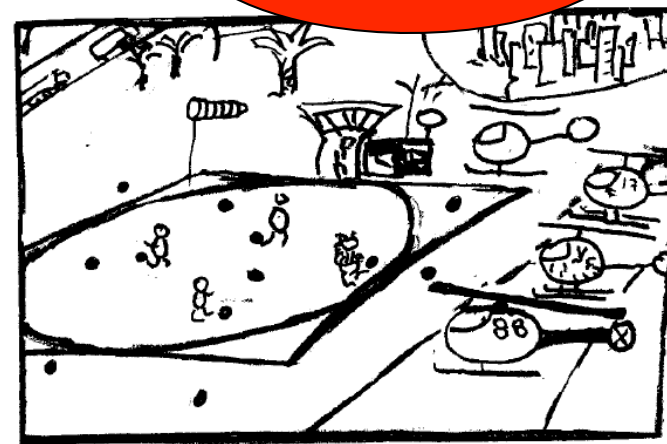
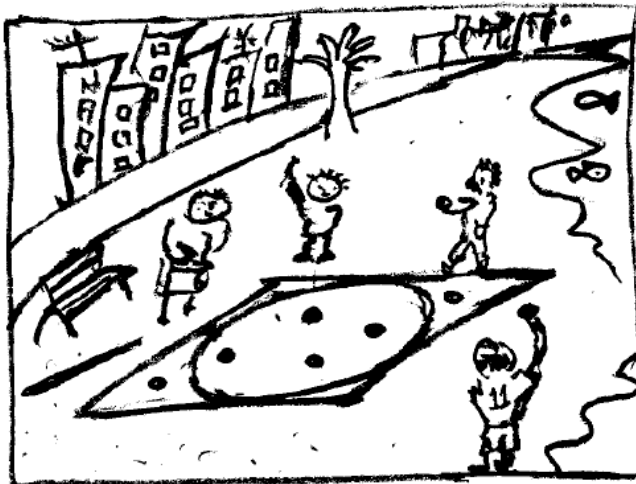
- Suppose we have a floor made of parallel strips of wood, each of the same width t , and we drop a needle of length l onto the floor. What is the probability that the needle will lie across a line between two strips?



> Calculating pi

- Very amusing description in W. Krauth, "Introduction To Monte Carlo Algorithms" (arXiv:cond-mat/9612186v2)

See example
later today



(PSEUDO)RANDOM NUMBERS

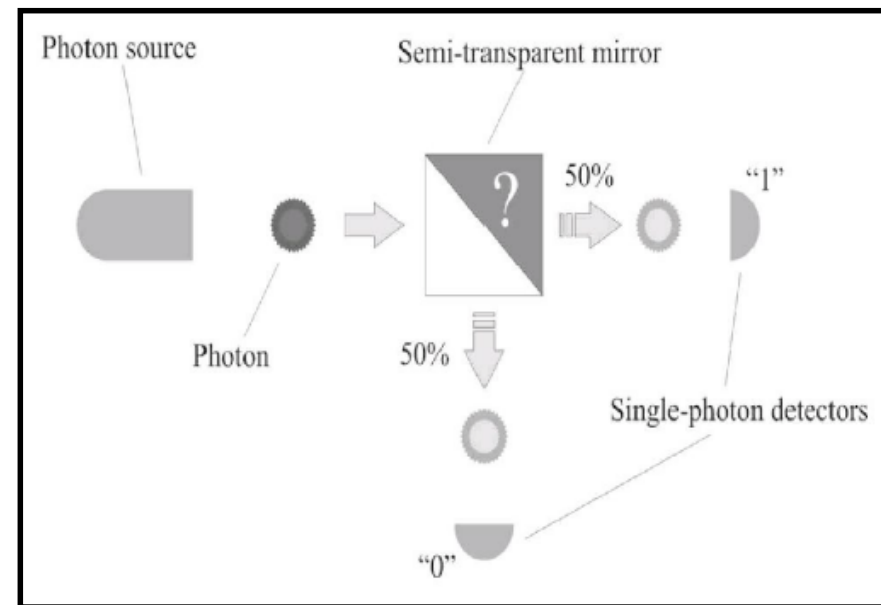
Random
numbers

- “Random” numbers are at the heart of the MC method.
- A **sequence of random numbers** is a set of numbers that have nothing to do with the other numbers in the sequence.
- There is no unique random number (sequence)!
- So ... how to get random numbers usable for our purposes?
- One (very attractive) option: Gambling in Monaco!



Tiresome and
potentially
expensive!

- > To obtain true random numbers ...
- > ... use some **classical chaotic system** like roulette, lotto, dices, coin tossing...
 - In principle, knowing all initial conditions, such a system is predictable – however, it is extremely sensitive → true random ...
- > ... or use **“modern physics” random processes** like
 - radioactive decay, ...
 - ... or other quantum mechanical process. Nice example: Photons on semi-transparent mirror!
→ available and tested by DESY summer student!
- > However, for typical nowadays applications (implementation in computer programs) all these true random number principles are too, slow, to cumbersome, not easily automatised, etc.
→ go for “second-best”: **pseudo-random numbers**!



(TRUE) RANDOM NUMBERS

Random
numbers

- Another nice example:
random.org
 - Making use of atmospheric noise which can be picked up with a normal radio!



- > ... are a sequence of numbers that ... well, APPEAR to be random,
 - but where in fact each number is derived from the previous N numbers by a well-defined algorithm.
- > More precisely, you want to generate integers I_n in the interval $[0;M]$ and from that derive $R_i = I_n/M$.
- > Numerous algorithms developed, for example “Middle Square Algorithm” (J. v. Neumann, 1946):
 - Start with a number of 10 digits, square it, take the middle 10 digits as the next number etc.
 - More complex algos don't necessarily lead to better results. Best to use algorithms that are well understood in their degree of “randomness”.
- > Nice example: “Linear congruential generator”:

$$I_{n+1} = \text{mod}(a \cdot I_n + c, m)$$

Seed I_0 modulus m
Multiplicative constant a
Additive constant c

$$R_{n+1} = I_{n+1} / m$$

Example for linear congruential generator: $l_0 = 10$, $a = 2$, $c = 5$, $m = 20$:

$$\begin{aligned}l_1 &= \text{mod}(2 \cdot 10 + 5, 19) = 6 \\l_2 &= \text{mod}(2 \cdot 6 + 5, 19) = 17 \\l_3 &= \text{mod}(2 \cdot 17 + 5, 19) = 1 \\l_4 &= \text{mod}(2 \cdot 1 + 5, 19) = 7 \\l_5 &= \text{mod}(2 \cdot 7 + 5, 19) = 0 \\l_6 &= \dots \quad 5, 15, 16, 18, 3, 11, 8, 2, 9, 4, 13, 12, 10, \mathbf{6}, \dots\end{aligned}$$

➔ After m steps, the sequence repeats!

Note: **Criteria** for randomness:

- uniformity
- **correlation tests**
- sequence-up / sequence-down tests
- gap tests
- random walk tests
- ...

- > By definition, the LCG generator has a maximum “random” sequence of length m .

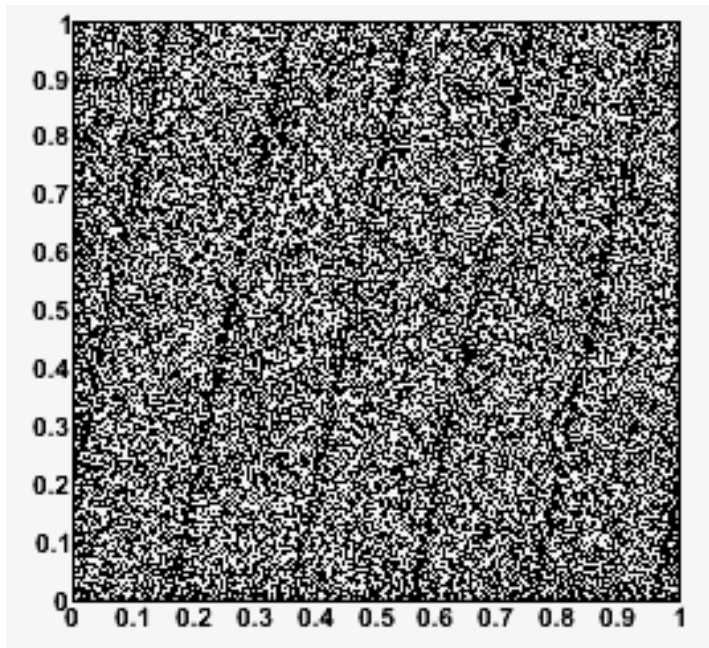
$$I_{n+1} = \text{mod}(a \cdot I_n + c, m)$$
$$R_{n+1} = I_{n+1} / m$$

- After that, repetition \rightarrow strong correlation of generated numbers:
- > Example for demonstration ($I_0 = 4711$, $a = 205$, $c = 29573$, $m = 139968$).
 - Shown is the correlation between pairs of numbers ($R_n; R_{n+1}$):

- > By definition, the LCG generator has a maximum “random” sequence of length m .

$$I_{n+1} = \text{mod}(a \cdot I_n + c, m)$$
$$R_{n+1} = I_{n+1} / m$$

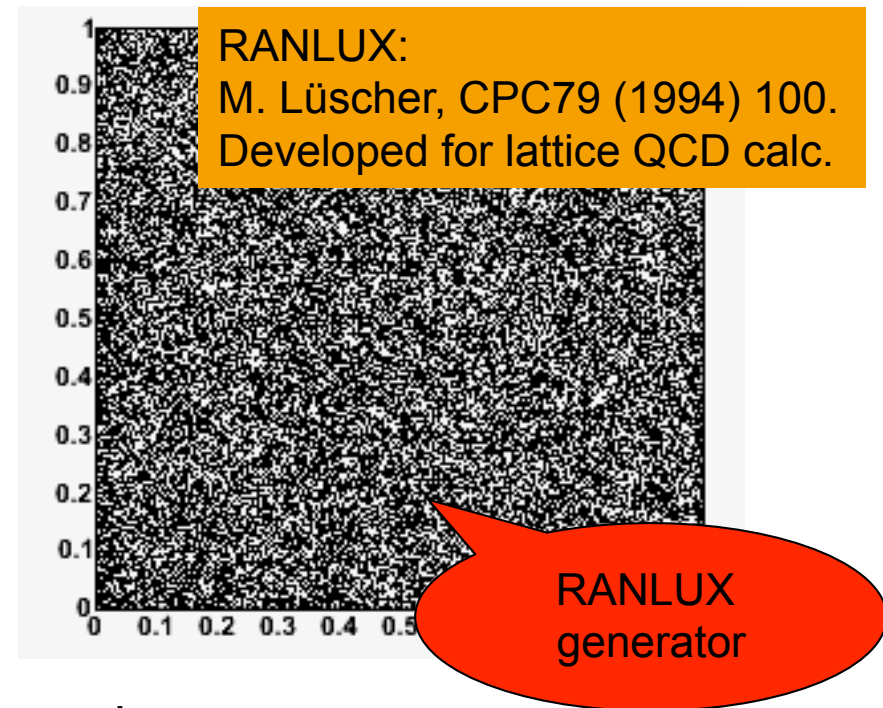
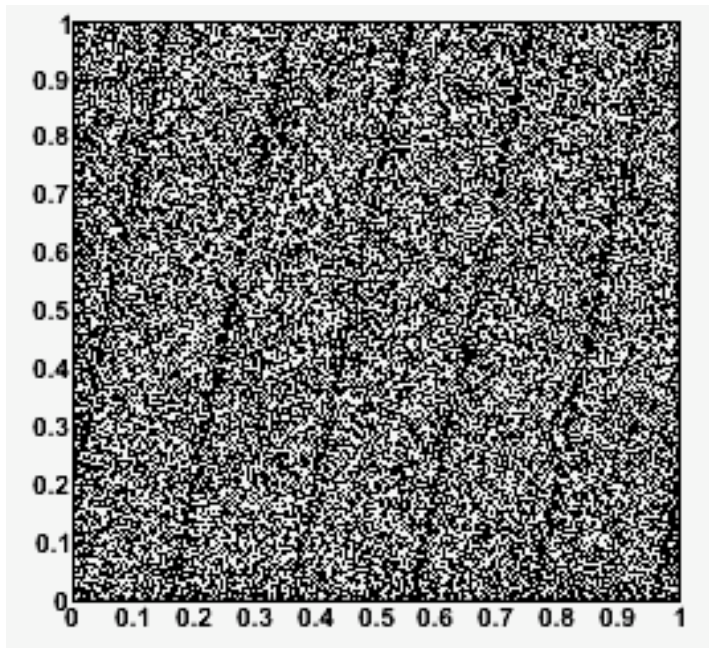
- After that, repetition \rightarrow strong correlation of generated numbers:
- > Example for demonstration ($I_0 = 4711$, $a = 205$, $c = 29573$, $m = 139968$).
 - Shown is the correlation between pairs of numbers ($R_n; R_{n+1}$):



- > By definition, the LCG generator has a maximum “random” sequence of length m .

$$I_{n+1} = \text{mod}(a \cdot I_n + c, m)$$
$$R_{n+1} = I_{n+1} / m$$

- After that, repetition \rightarrow strong correlation of generated numbers:
- > Example for demonstration ($I_0 = 4711$, $a = 205$, $c = 29573$, $m = 139968$).
 - Shown is the correlation between pairs of numbers ($R_n; R_{n+1}$):



- > ... let's now assume we have random numbers ...

- > Expectation value $E(f)$ or μ : defined as the average/mean value of function f :
("mean value theorem")

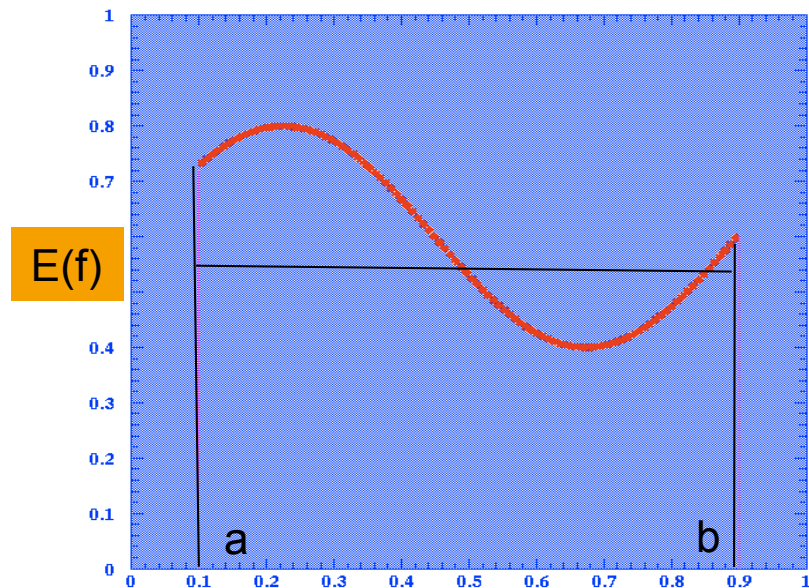
$$E(f) = \int f(u) dG(u) \underset{\text{uniformPDF}}{=} \left(\frac{1}{b-a} \int_a^b f(u) du \right)$$
$$E(cx + y) = cE(x) + E(y)$$

→ close connection to integration (later) etc.!

- > Variance $V(f)$ (\sim standard deviation σ^2):

$$V(f) = \int (f - E(f))^2 dG = \left(\frac{1}{b-a} \int_a^b (f - E(f))^2 du \right)$$

- relevance: want small uncertainties on MC predictions / results
- aim to reduce variance (not really covered here).



> **Law of large numbers**: For large enough statistics, relative frequency of an outcome approaches probability.

- For MC relevant (integration!): Choose N numbers u_i randomly with uniform probability density in interval $[a;b]$, evaluate $f(u_i)$ for each u_i :

$$\frac{1}{N} \sum_{i=1}^N f(u_i) \rightarrow \frac{1}{b-a} \int_a^b f(u) du$$

For large enough N , the Monte Carlo estimate of the integral converges to the correct answer.

> **Central limit theorem**: For large N , the sum of N independent random variables is **ALWAYS** normally (Gaussian) distributed!

$$Z_n \equiv \sum_{i=1}^N x_i \text{ is Gaussian (for } n \rightarrow \infty \text{)!!! :}$$

$$f(Z_n) \stackrel{n \rightarrow \infty}{=} \frac{1}{\sigma \sqrt{2\pi}} \exp \left[-\frac{(Z_n - \mu)^2}{2\sigma^2} \right]$$

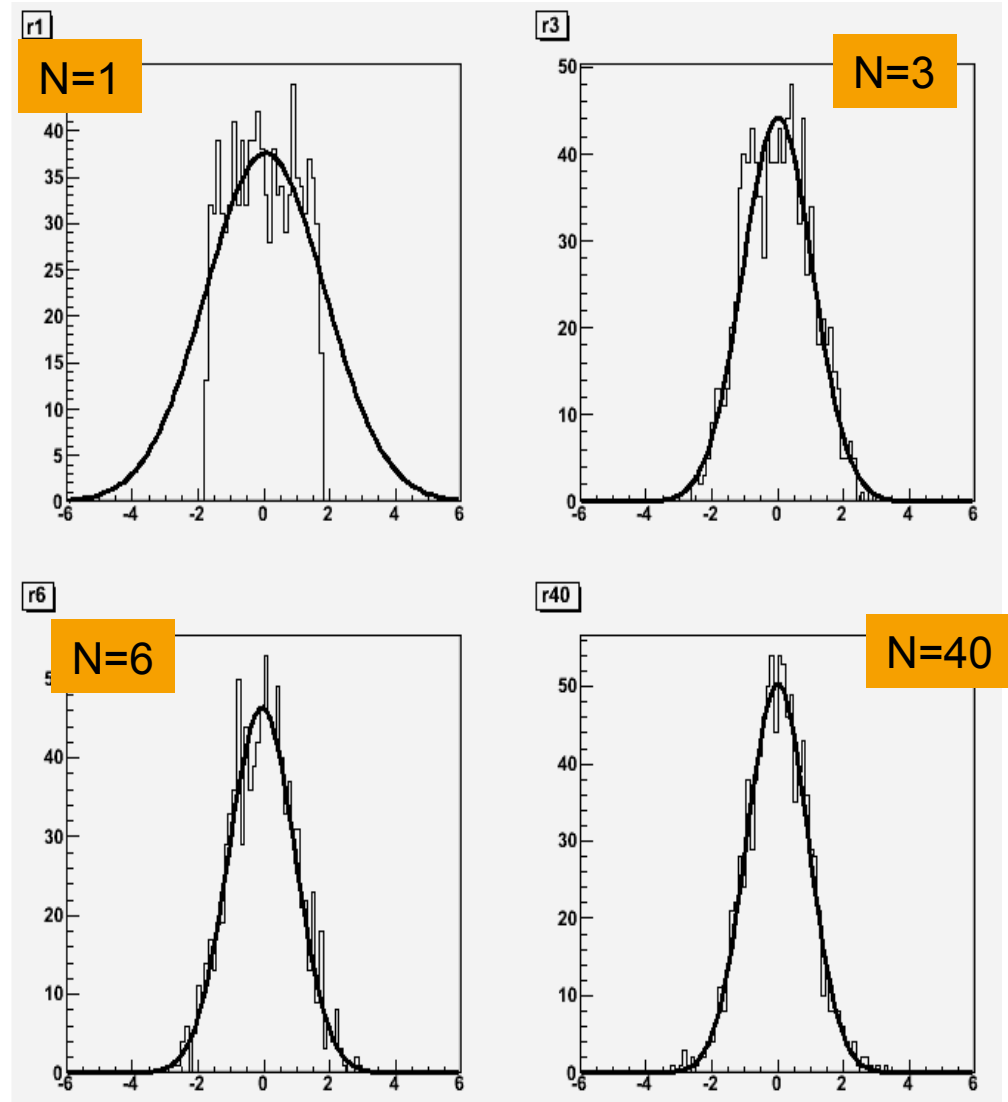
→ This outcome is **independent of the original distributions** of the x_i !

- Simple example: sum of n random numbers x_i from $[0;1]$:

$$\begin{aligned}R_n &= \sum_{i=1}^n R_i \\E(R_1) &= \int_0^1 u du = 1/2 \\V(R_1) &= \int_0^1 (u - 1/2)^2 du = 1/12 \\E(R_n) &= n/2 \\V(R_n) &= n/12\end{aligned}$$

> Simple example: sum of n random numbers x_i from $[0;1]$:

$$R_n = \sum_{i=1}^n R_i$$
$$E(R_1) = \int_0^1 u du = 1/2$$
$$V(R_1) = \int_0^1 (u - 1/2)^2 du = 1/12$$
$$E(R_n) = n/2$$
$$V(R_n) = n/12$$



- > Simple example: sum of n random numbers x_i from $[0;1]$:

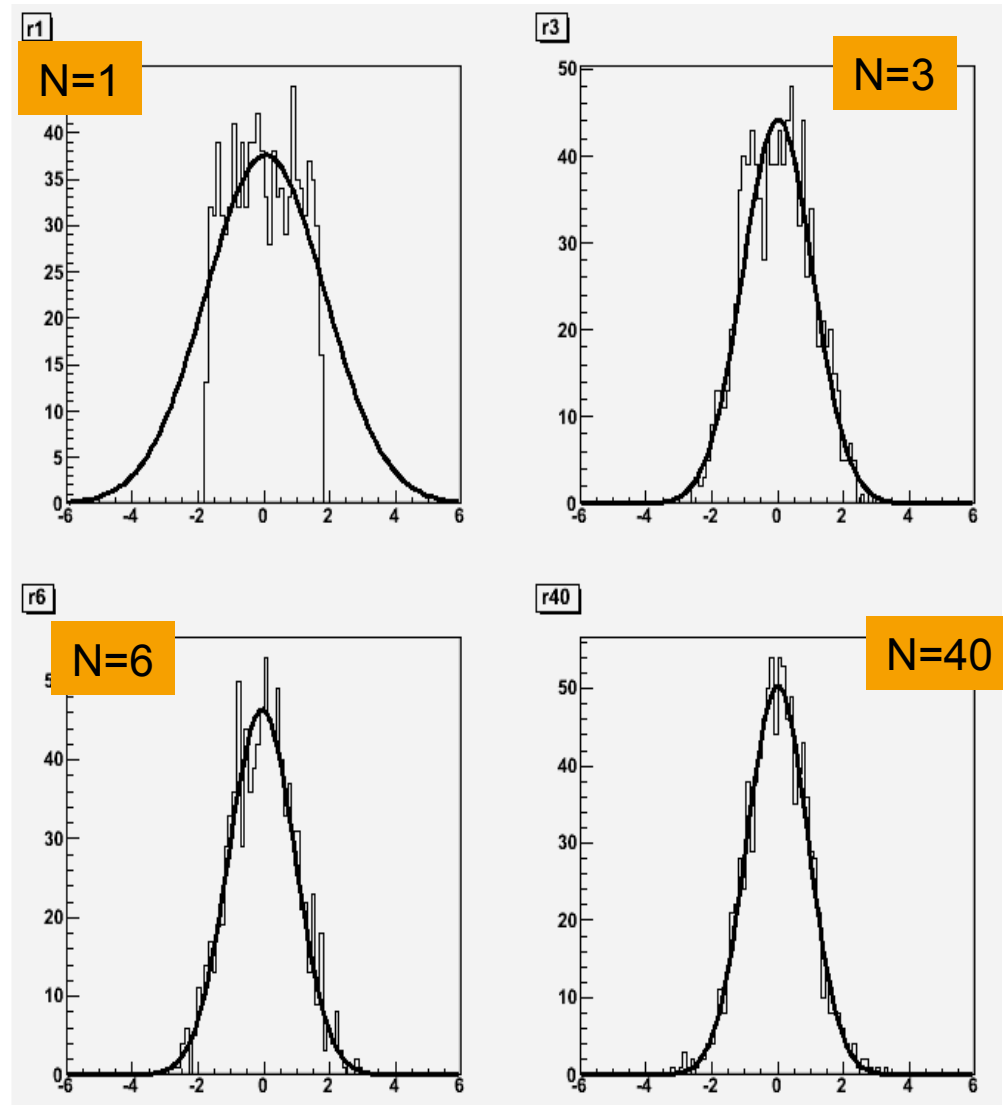
$$\begin{aligned} R_n &= \sum_{i=1}^n R_i \\ E(R_1) &= \int_0^1 u du = 1/2 \\ V(R_1) &= \int_0^1 (u - 1/2)^2 du = 1/12 \\ E(R_n) &= n/2 \\ V(R_n) &= n/12 \end{aligned}$$

- > To generate normal distribution centered at 0, variance 1, use:

$$\frac{R_n - n/2}{\sqrt{n/12}} = \frac{\sum_{i=1}^n R_i - \sum_{i=1}^n \mu_i}{\sqrt{\sum_{i=1}^n \sigma_i^2}}$$

more generally:

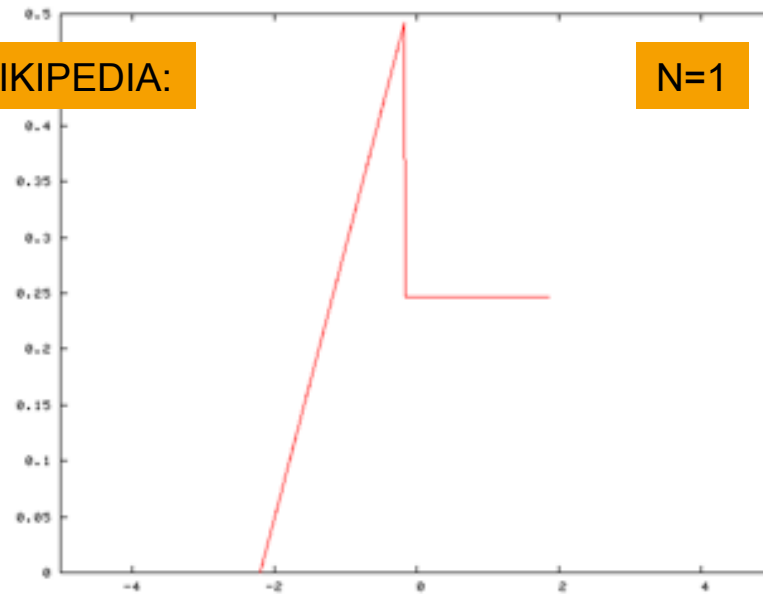
$$Z_n \equiv \frac{\sum_i x_i - n \cdot \mu}{\sqrt{n} \sigma} \rightarrow N(0,1)$$



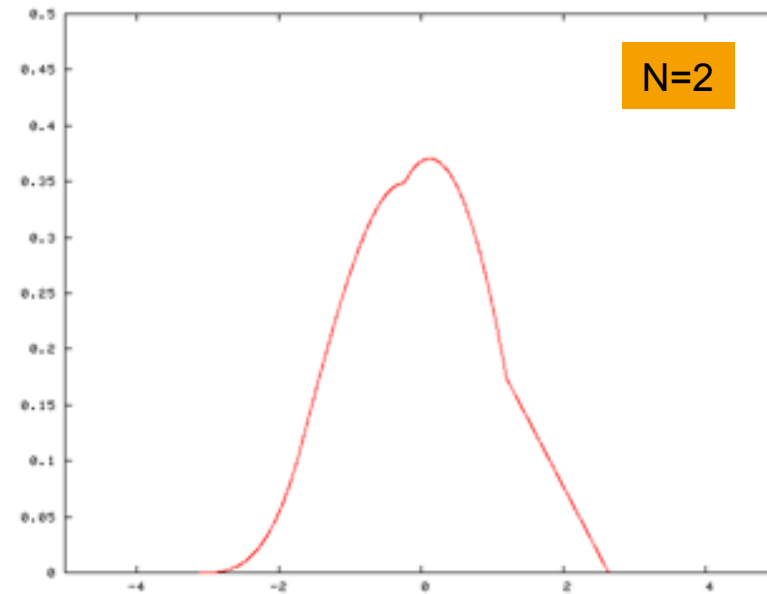
(MATHEMATICAL) BASICS

Mathematical
basics

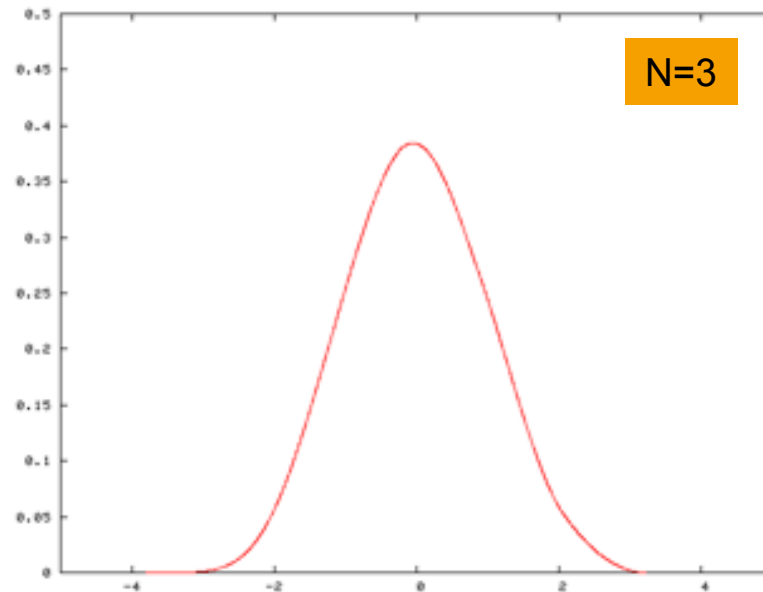
From WIKIPEDIA:



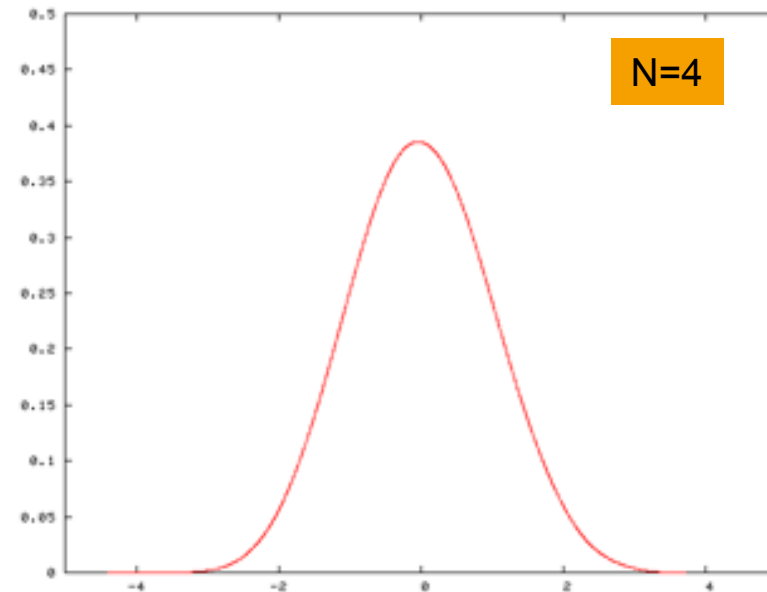
$N=1$



$N=2$

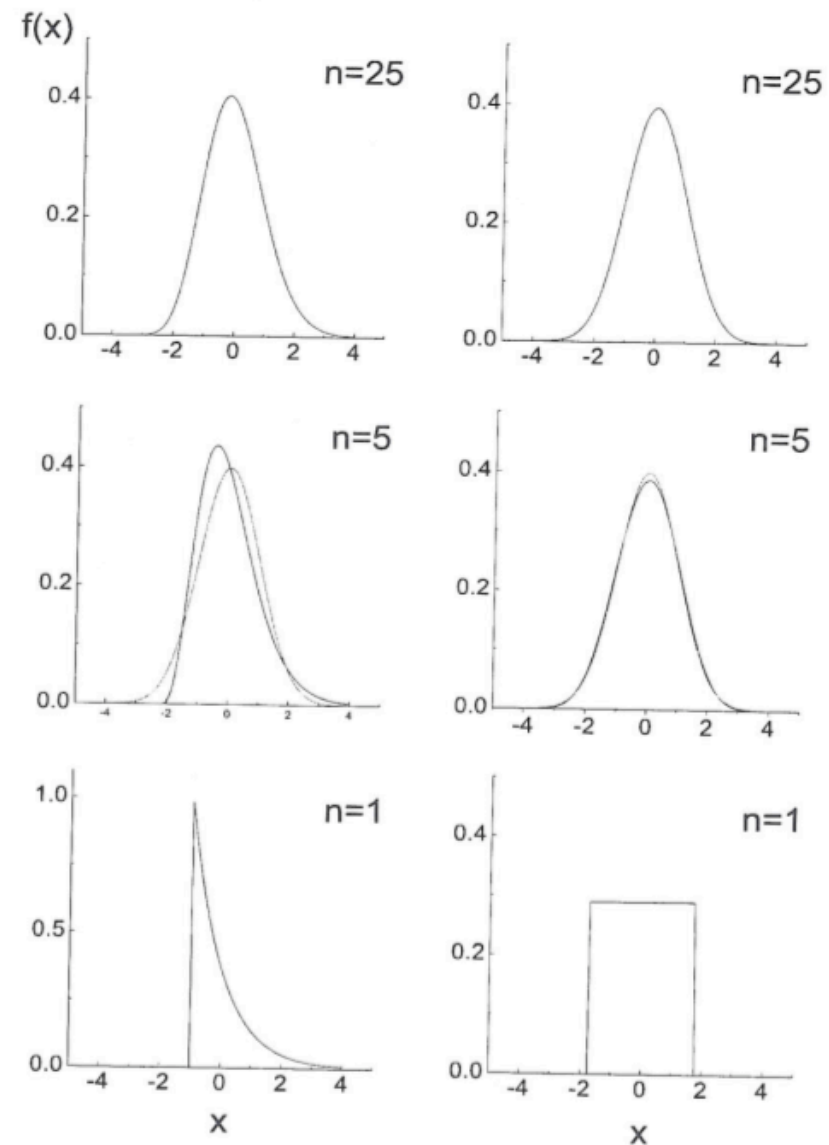


$N=3$

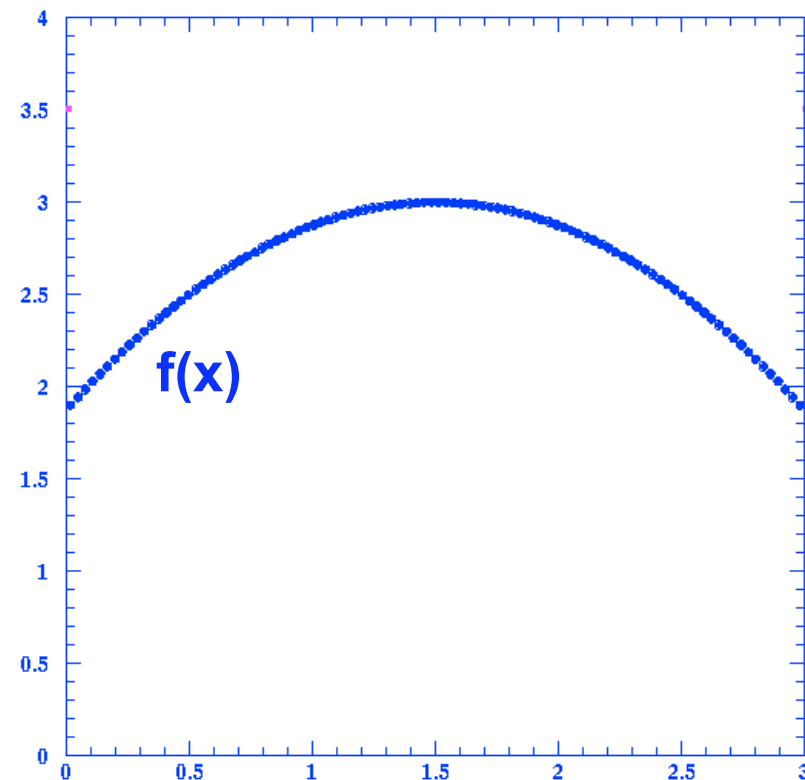


$N=4$

- Bohm&Zech: Einführung in die Statistik und Messwertanalyse:
 - Works for ANY starting distribution!!!!

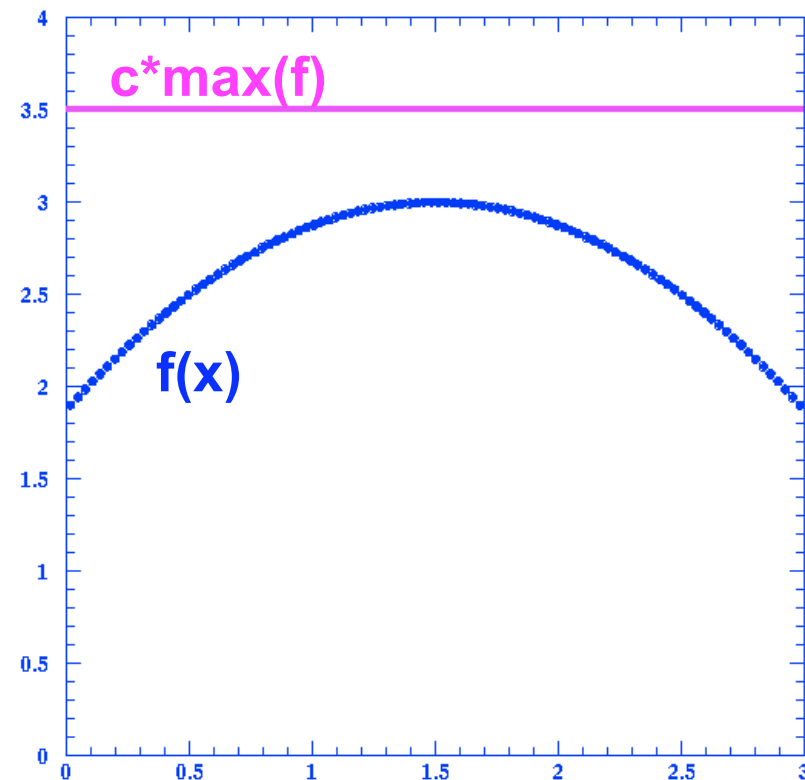


- > Assume you want to create events according to some distribution $f(x)$!
- > Brute force or Hit&Miss method:
 - Works always – but not very elegant and not always efficient!



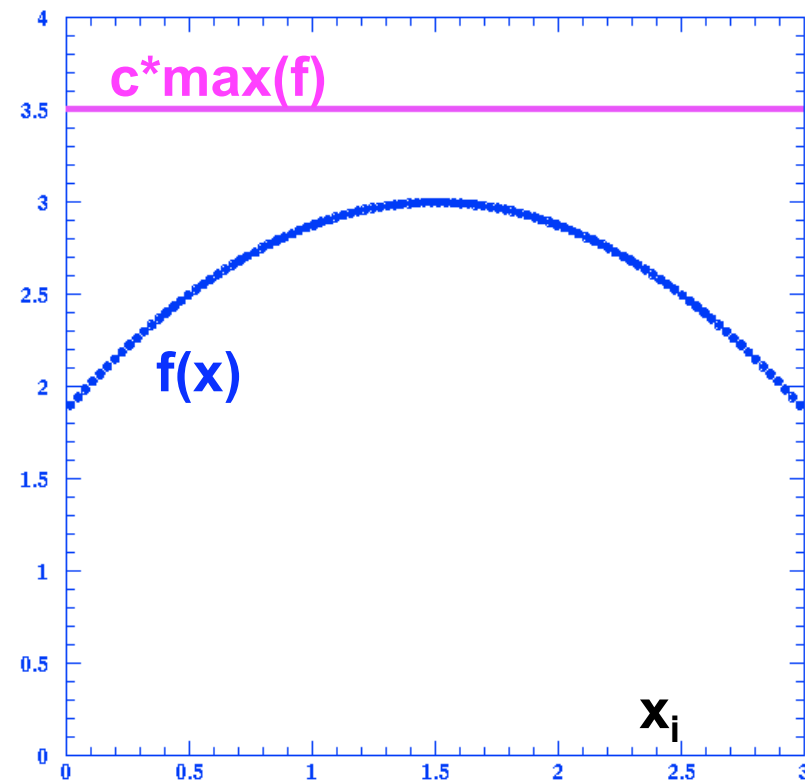
- > Assume you want to create events according to some distribution $f(x)$!
- > Brute force or Hit&Miss method:
 - Works always – but not very elegant and not always efficient!

- Find maximum $c \cdot \max(f)$.



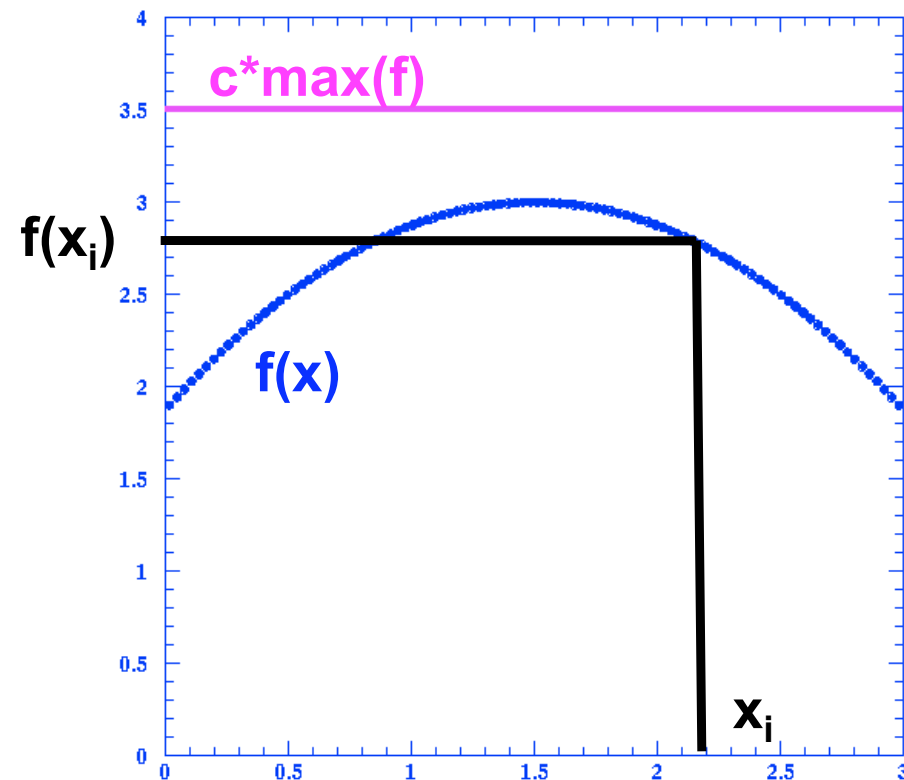
- > Assume you want to create events according to some distribution $f(x)$!
- > Brute force or Hit&Miss method:
 - Works always – but not very elegant and not always efficient!

- Find maximum $c \cdot \max(f)$.
- Choose random number x_i .



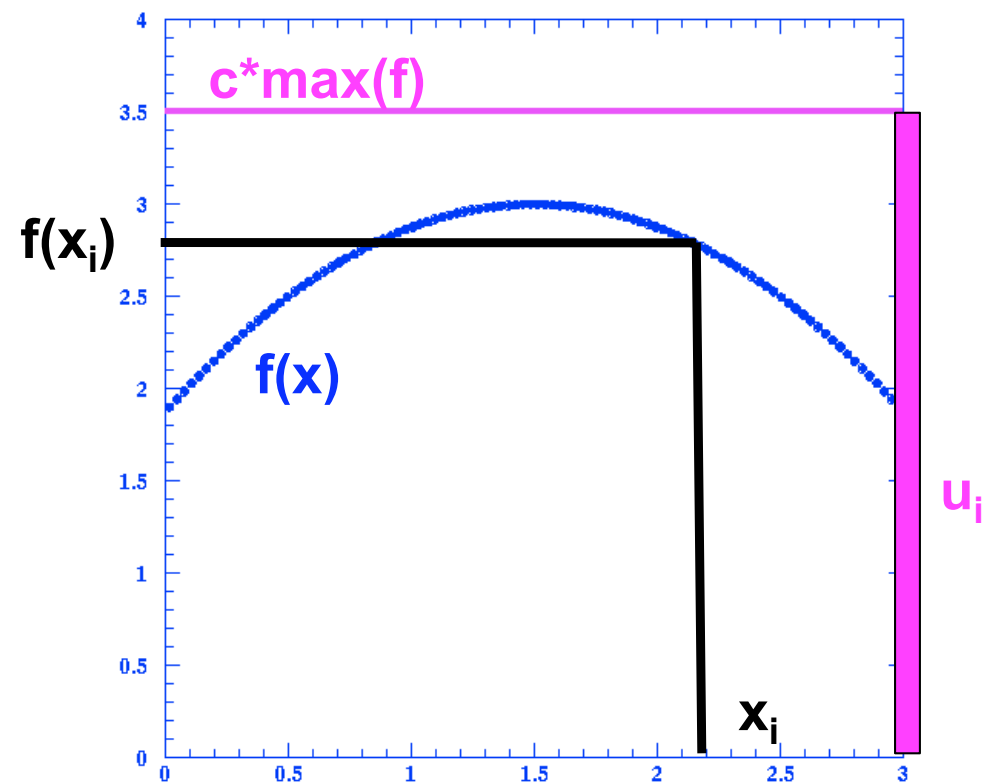
- > Assume you want to create events according to some distribution $f(x)$!
- > Brute force or Hit&Miss method:
 - Works always – but not very elegant and not always efficient!

- Find maximum $c \cdot \max(f)$.
- Choose random number x_i .
- Calculate $f(x_i)$.



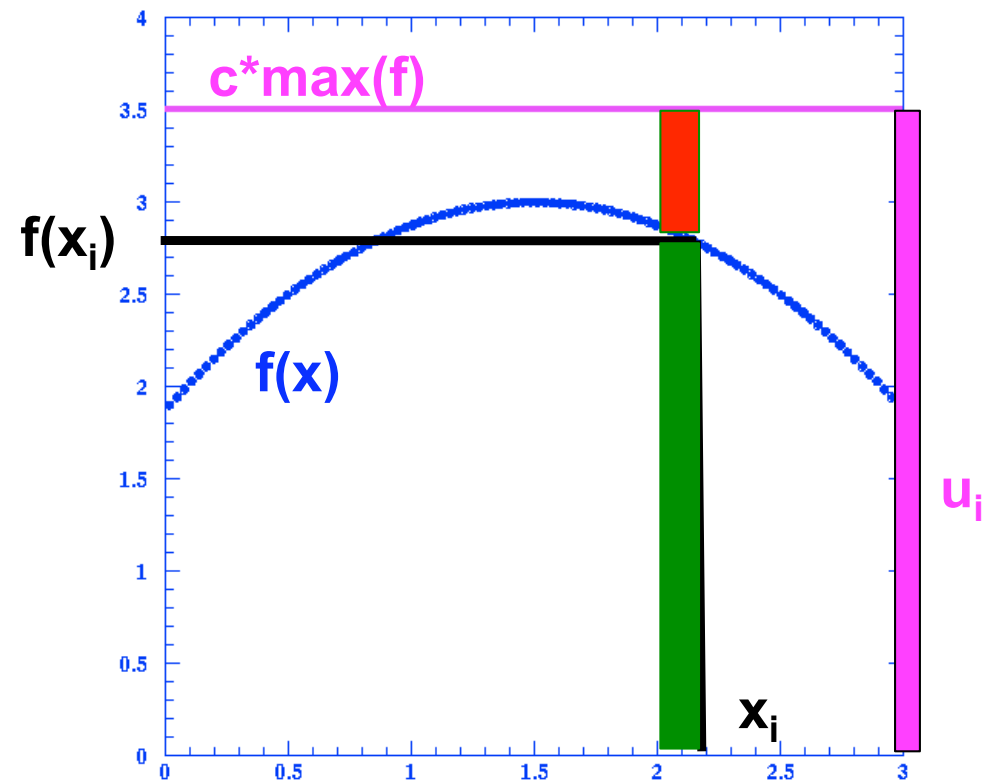
- Assume you want to create events according to some distribution $f(x)$!
- Brute force or Hit&Miss method:
 - Works always – but not very elegant and not always efficient!

- Find maximum $c \cdot \max(f)$.
- Choose random number x_i .
- Calculate $f(x_i)$.
- Choose random number u_i from interval $[0; c \cdot \max(f)]$.

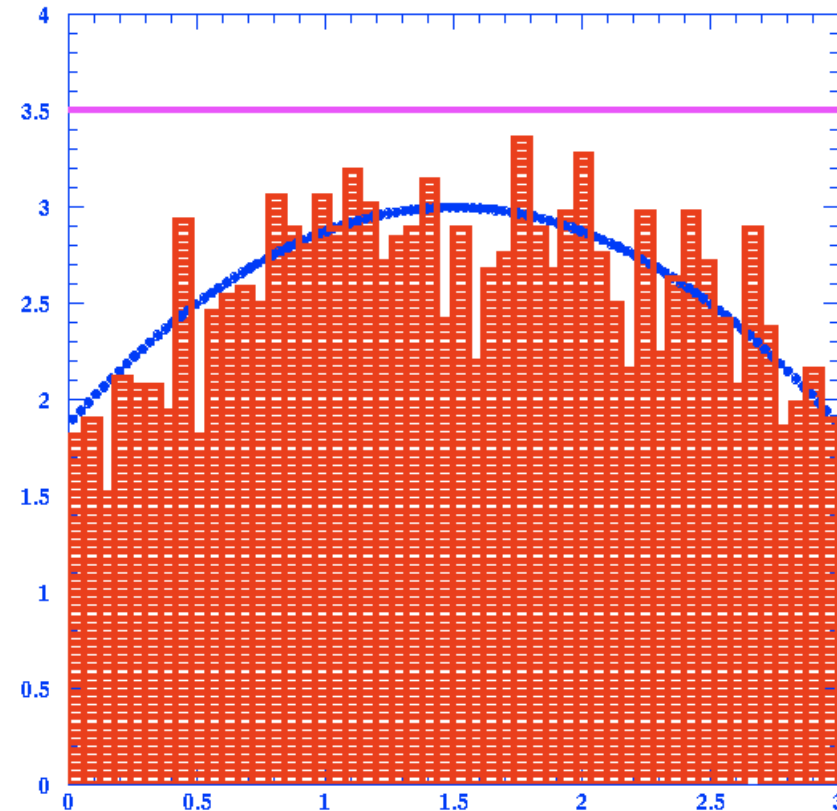
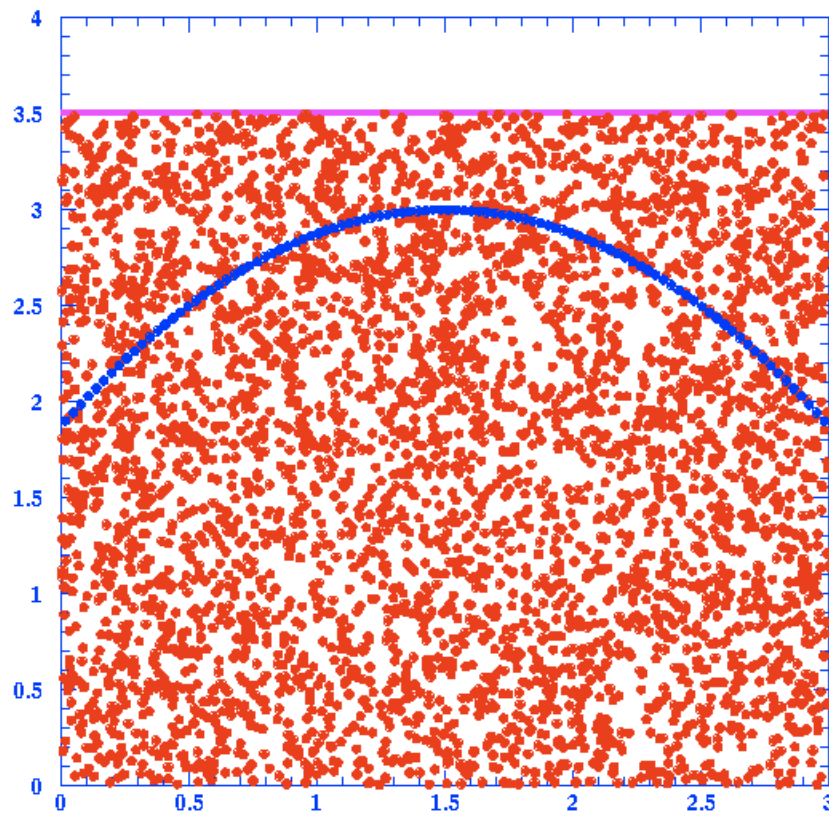


- > Assume you want to create events according to some distribution $f(x)$!
- > Brute force or Hit&Miss method:
 - Works always – but not very elegant and not always efficient!

- Find maximum $c \cdot \max(f)$.
- Choose random number x_i .
- Calculate $f(x_i)$.
- Choose random number u_i from interval $[0; c \cdot \max(f)]$.
- If $u_i > f(x_i)$ **reject event**.
- Else **accept**
(and make entry in histogram)!

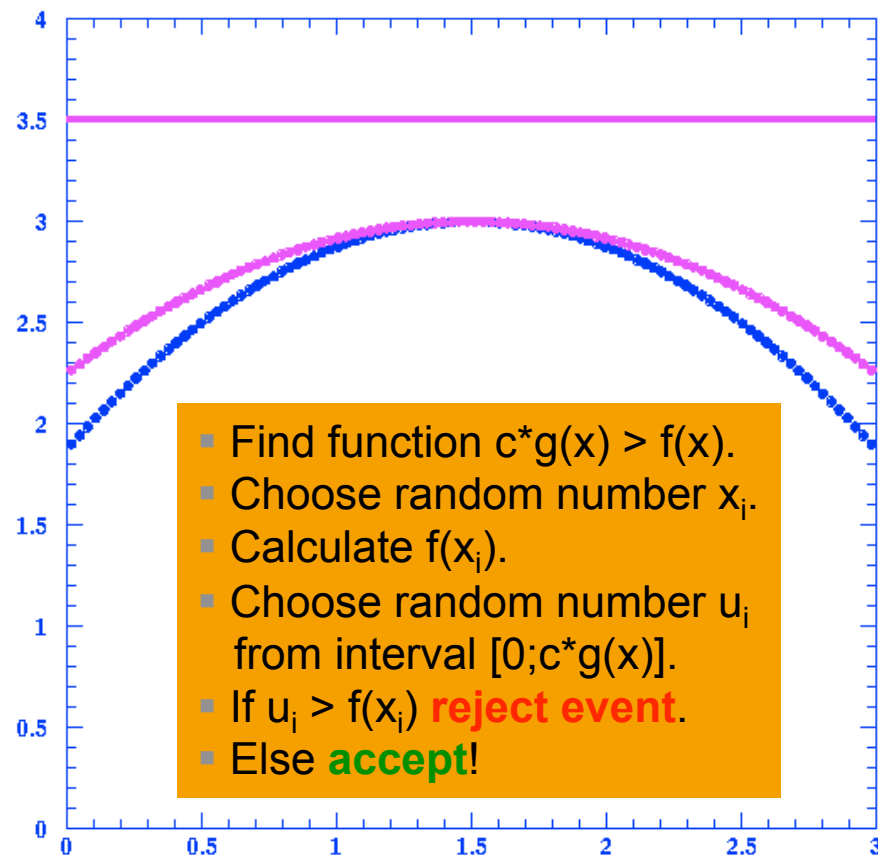


- Brute force or Hit&Miss method:
 - Preferred if no simple analytical solution exists!



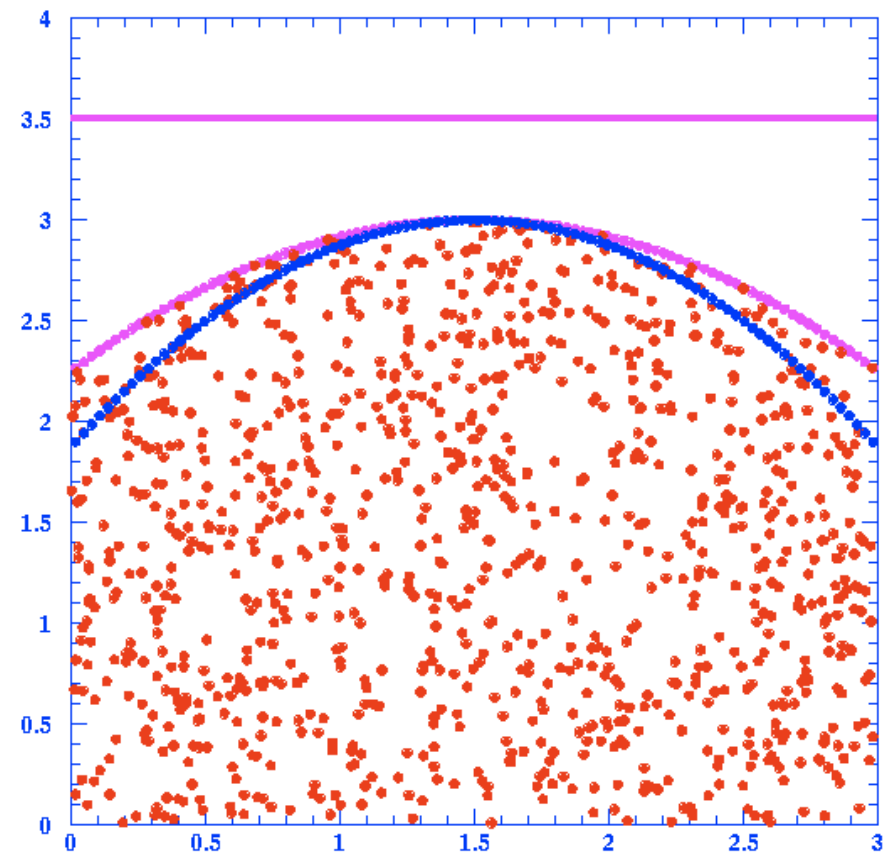
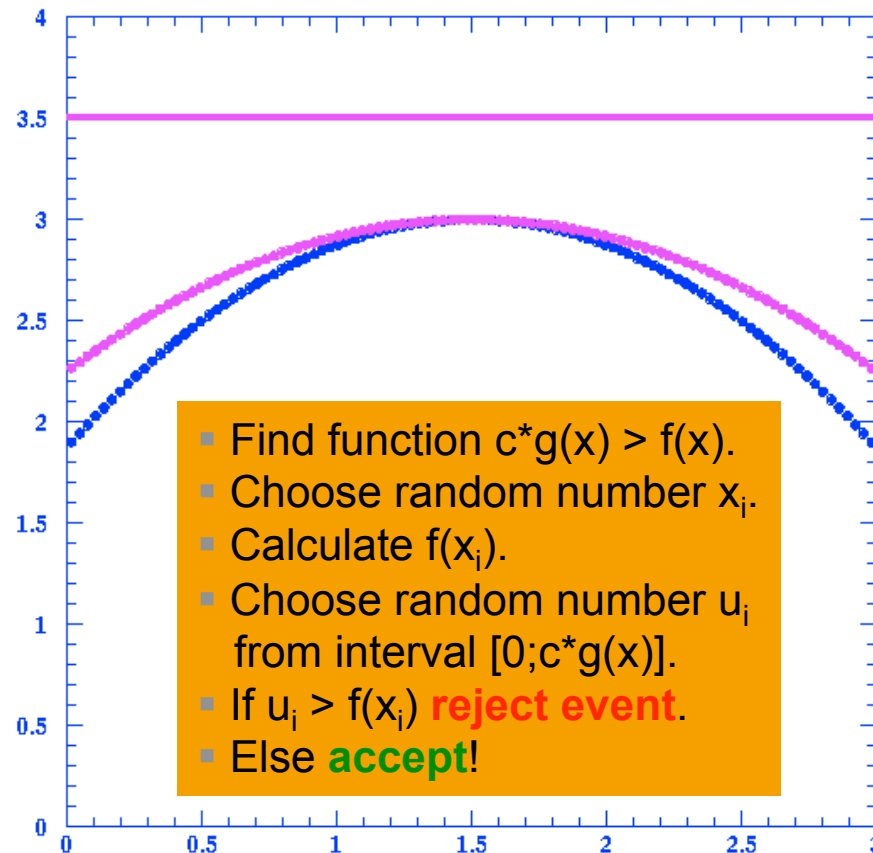
> Modified brute force or Hit&Miss method:

- Improve efficiency by variable transformation or better adjusted estimate of maximum!



> Modified brute force or Hit&Miss method:

- Improve efficiency by variable transformation or better adjusted estimate of maximum!



> More elegantly – if analytically possible:

- Generation via “inversion of cumulative distribution function”

- Let $f(t)$ be the function to simulate:
- First build the cumulative distribution (“Stammfunktion”)
- Build the inverse of $F(x)$, $F^{-1}(Z)$.
- Proposition: With Z from $[0.;1.]$ $F^{-1}(Z)$ is distributed as $f(t)$.
- Proof in many books and lectures ;-).

$$f(t)$$

$$F(x) = \int_0^x f(t) dt$$

$$x_i = F^{-1}(Z_i)$$

> Works nicely and elegantly -

- ... see examples on next pages.

> More elegantly – if analytically possible:

- Generation via “inversion of cumulative distribution function”

- Let $f(t)$ be the function to simulate:
- First build the cumulative distribution (“Stammfunktion”)
- Build the inverse of $F(x)$, $F^{-1}(Z)$.
- Proposition: With Z from $[0.;1.]$ $F^{-1}(Z)$ is distributed as $f(t)$.
- Proof in many books and lectures ;-).

$$f(t)$$

$$F(x) = \int_0^x f(t) dt$$

$$x_i = F^{-1}(Z_i)$$

$$f(t) = 1/t$$

$$F(x) \propto \int 1/t dt \propto \ln x$$

$$x_i = F^{-1}(Z_i) = \exp(Z_i)$$

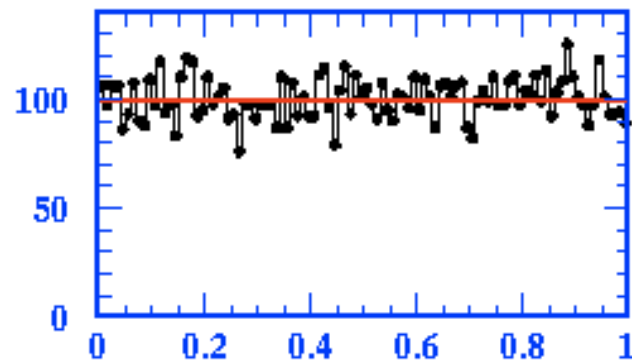
Take uniform numbers Z_i and use as “random” number x_i .

> Works nicely and elegantly -

- ... see examples on next pages.

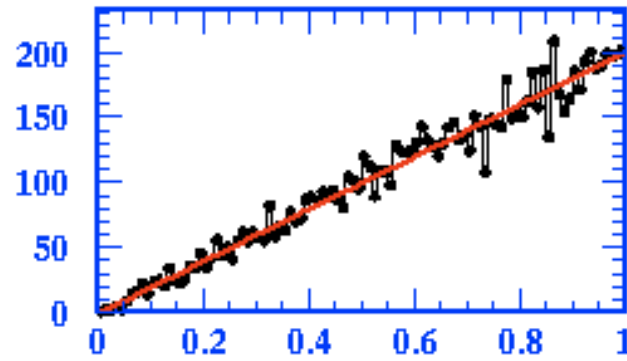
GENERATING DISTRIBUTIONS (3)

Generating
distributions



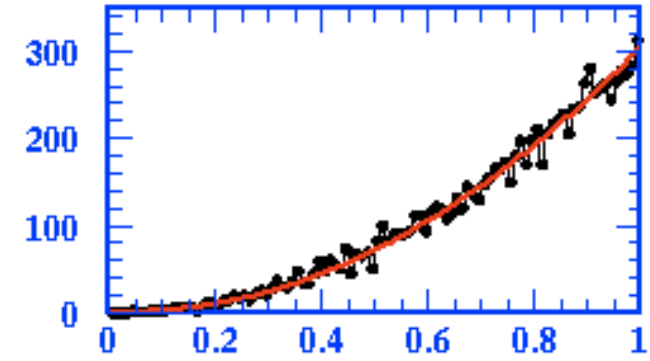
$$x_i = F^{-1}(Z_i) \quad Z \in [0;1.]$$

$$F^{-1}(Z_i) = Z_i$$



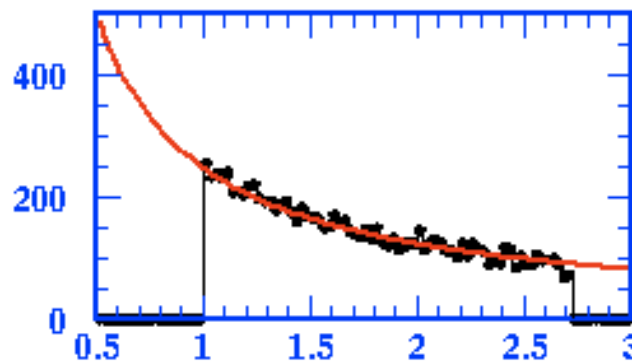
linear function

$$F^{-1}(Z_i) = \sqrt{Z_i}$$



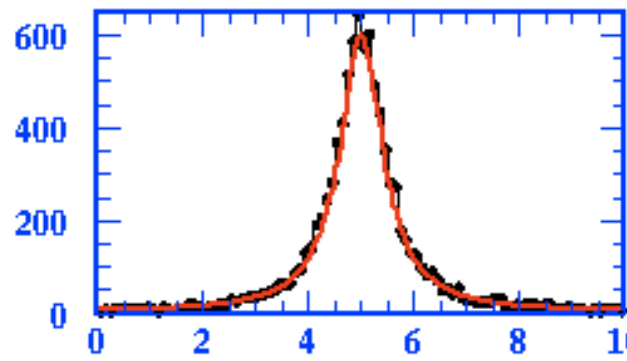
quadratic function

$$F^{-1}(Z_i) = \sqrt[3]{Z_i}$$



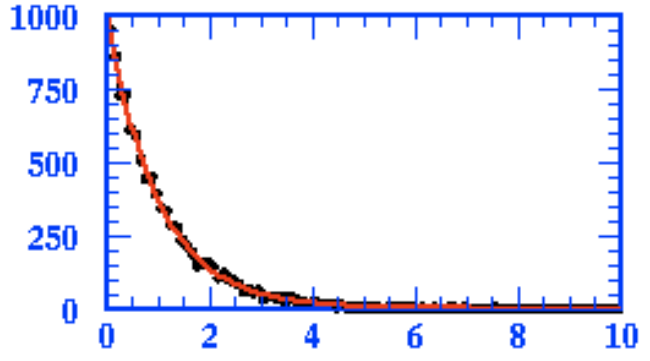
1/x function

$$F^{-1}(Z_i) = \exp(Z_i)$$



Lorentz curve

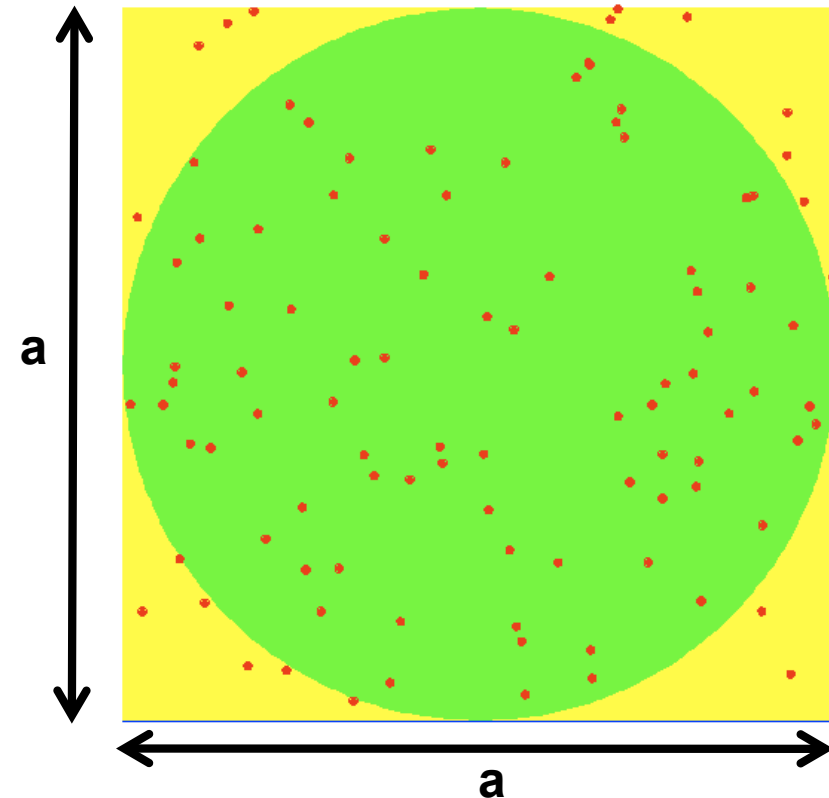
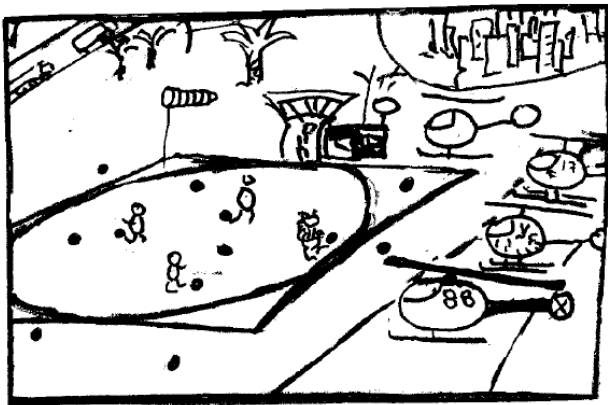
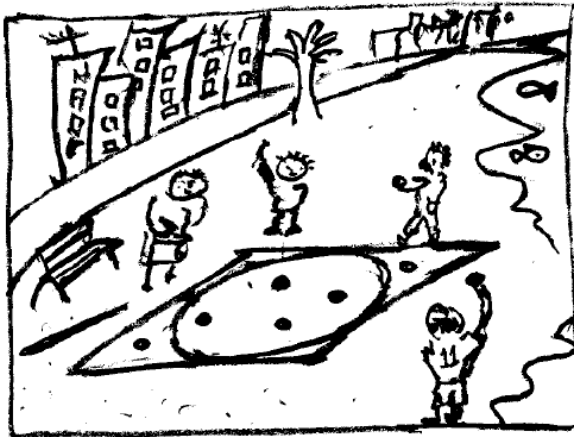
$$F^{-1}(Z_i) = a + b \cdot \tan(\pi \cdot Z_i - 1/2)$$







exponential

$$F^{-1}(Z_i) = -\ln Z_i$$

- Another application of hit&miss: Calculating π using pebbles on the beach or helicopters in MC:

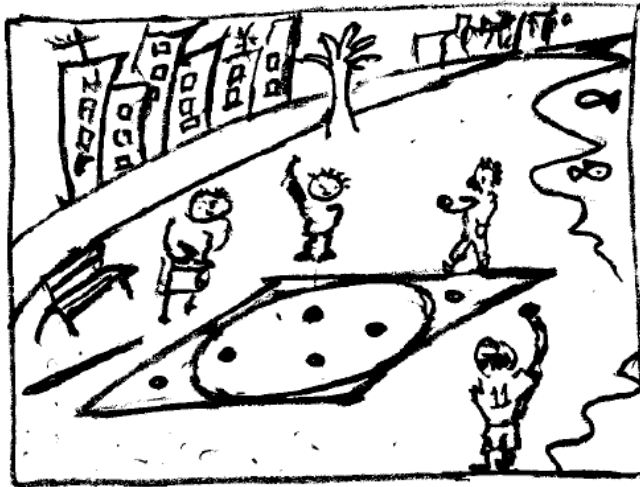


Area  = a^2  /  = $4/\pi \rightarrow N_{\text{yellow}} / N_{\text{green}}$

Area  = $\pi \cdot (a/2)^2$ $\pi = 4 \cdot N_{\text{green}} / N_{\text{yellow}}$

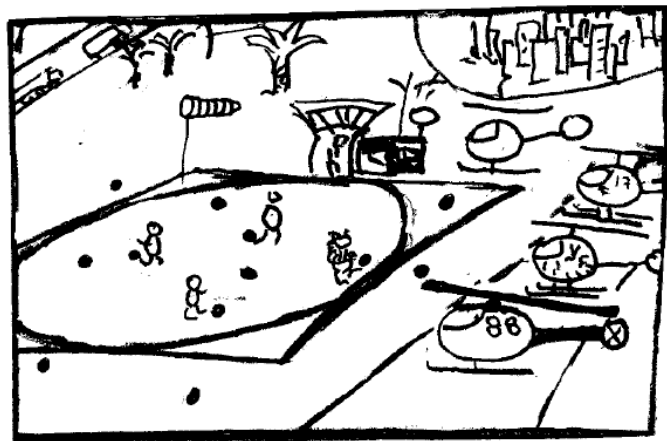
... invoking again the law of large numbers ...

- Calculating π using pebbles on the beach or helicopters in Monte Carlo:



10^3 :	3.056000
10^4 :	3.132800
10^5 :	3.145680
10^6 :	3.141996
10^7 :	3.141707
10^8 :	...

$\pi = 3.141592653...$



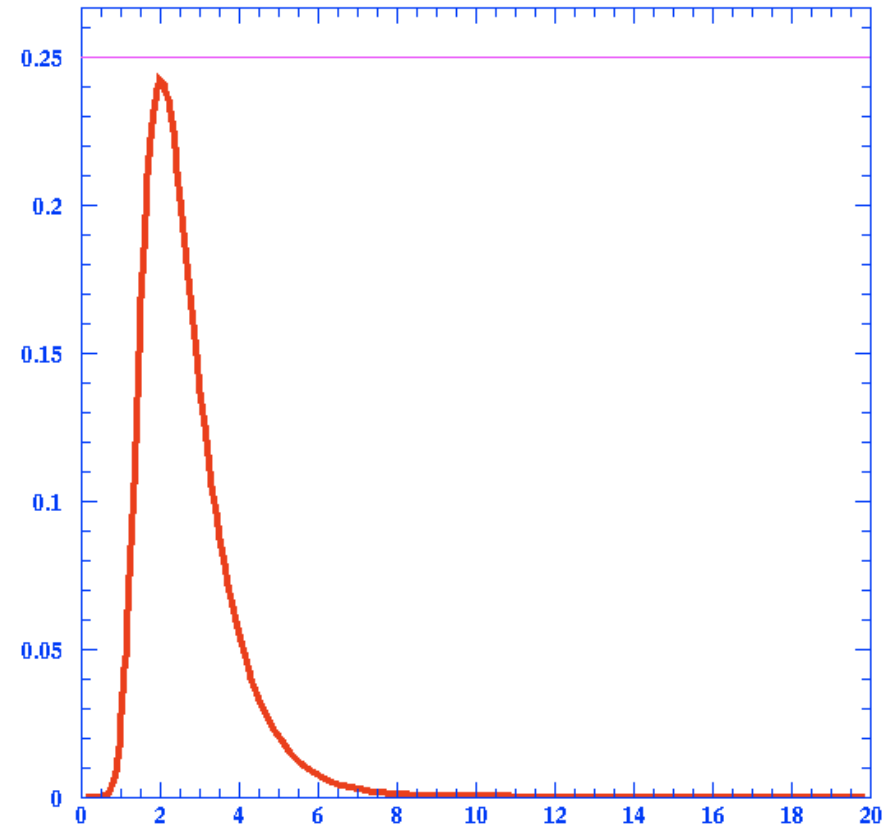
> Integration with Hit&Miss method:

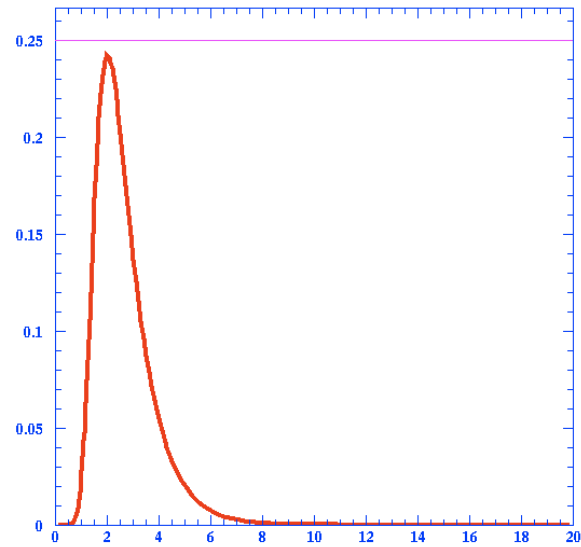
- Get random number R_1 for x axis in interval $[0.;20.]$ (here).
- Get random number R_2 for y axis between 0. and $c \cdot \max(f)$.
- Reject if $R_2 > f(R_1)$ (point “above” function).
- Else accept.

→ In principle comparison with area of known size!

> Example: Landau distribution (energy loss of particles passing a thin layer of matter)

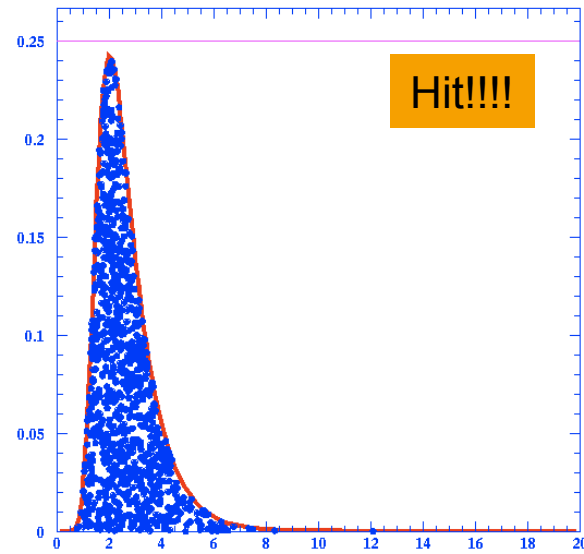
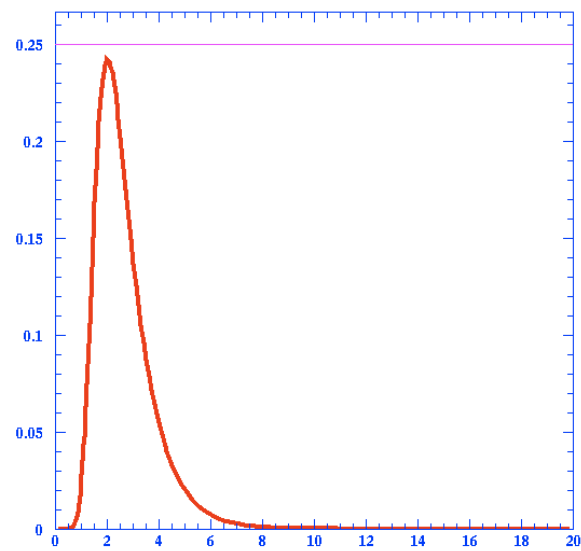
- Strongly peaked signal
- ... now do hit and miss ...





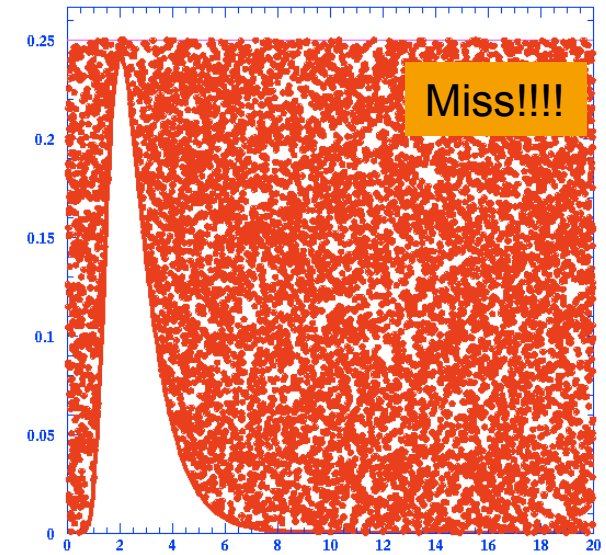
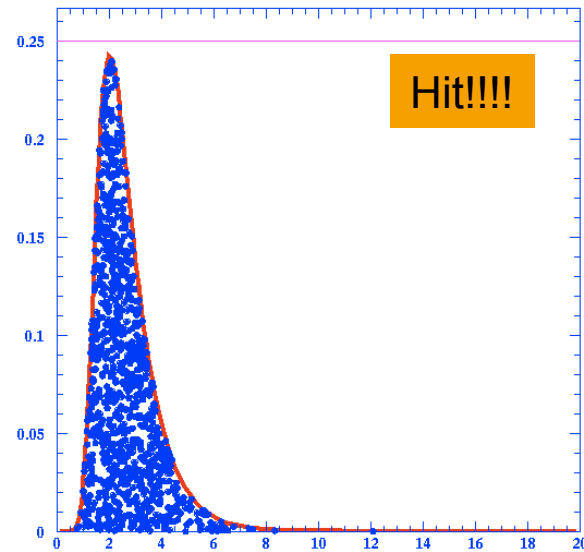
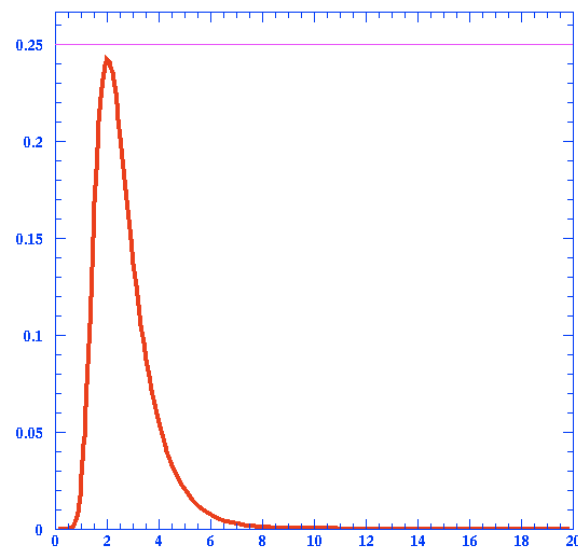
MC INTEGRATION (2)

MC
Integration



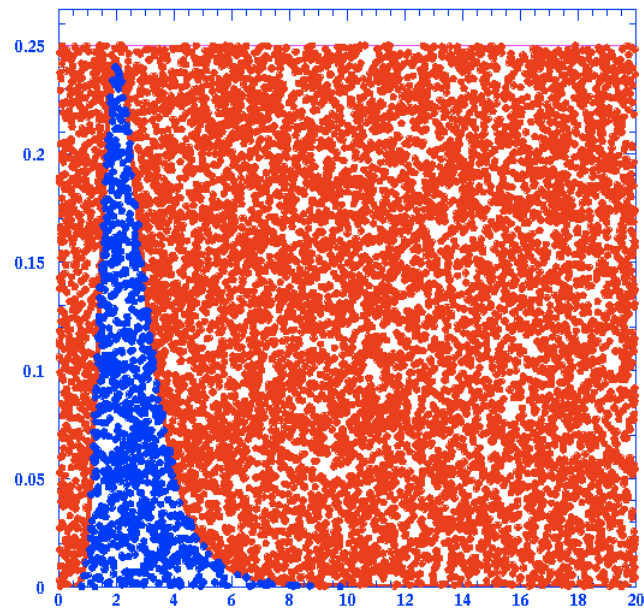
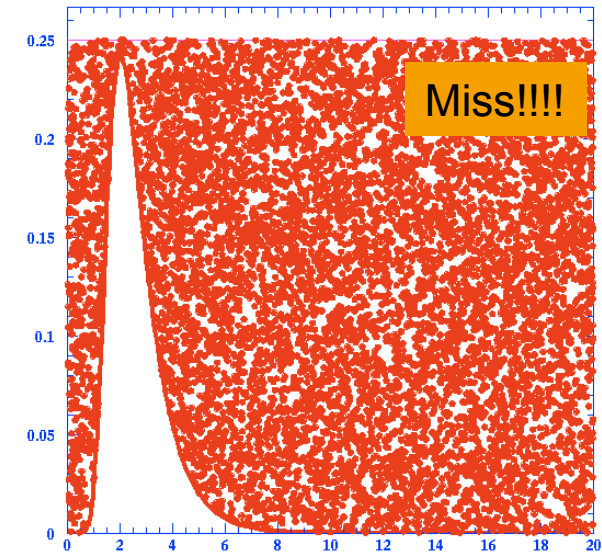
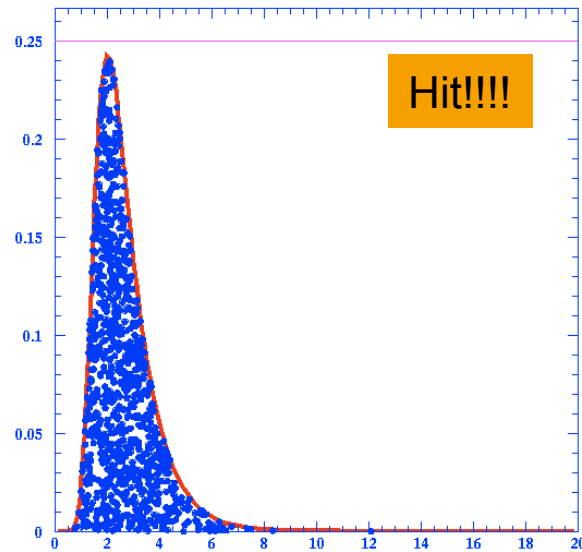
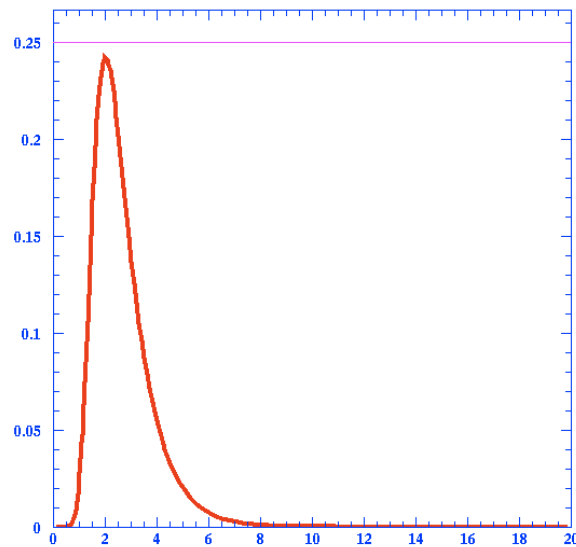
MC INTEGRATION (2)

MC Integration



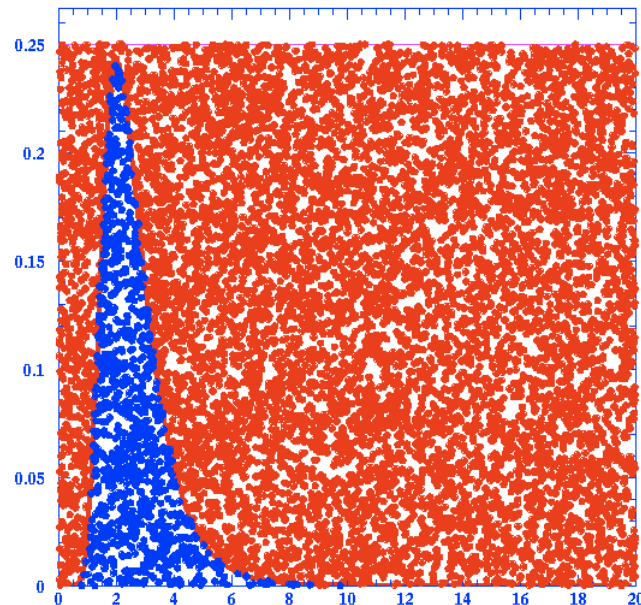
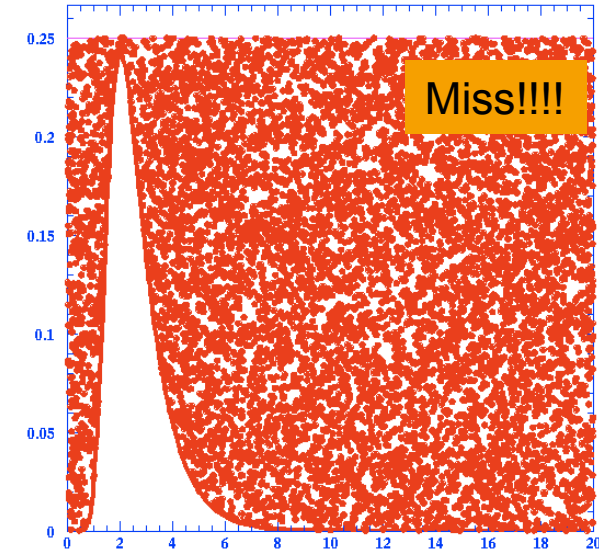
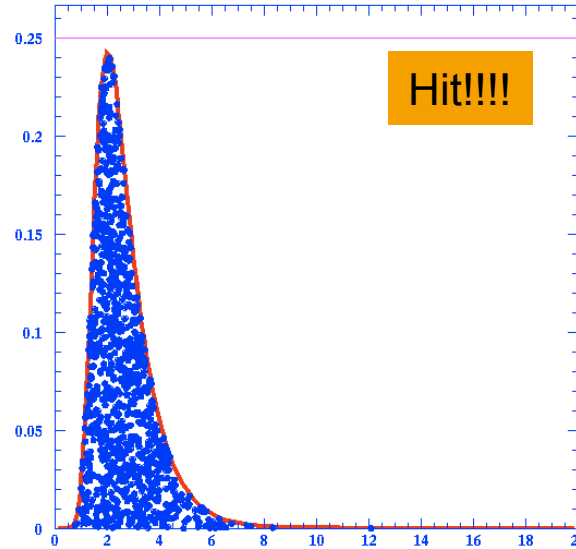
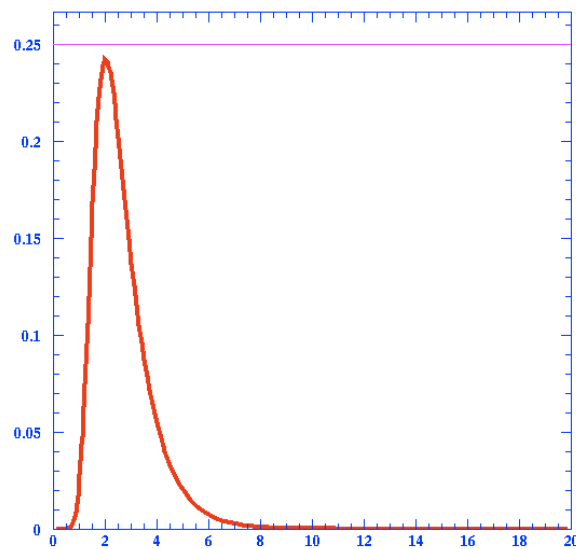
MC INTEGRATION (2)

MC Integration



MC INTEGRATION (2)

MC Integration

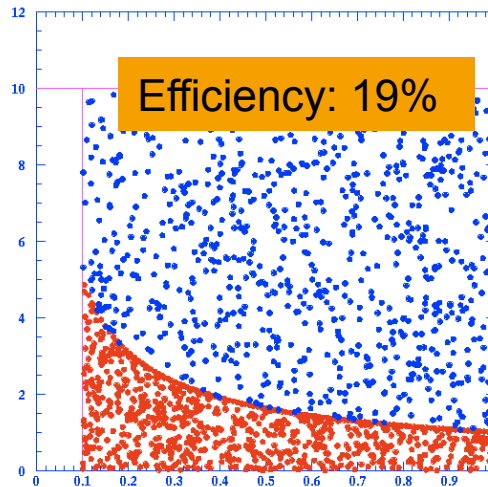
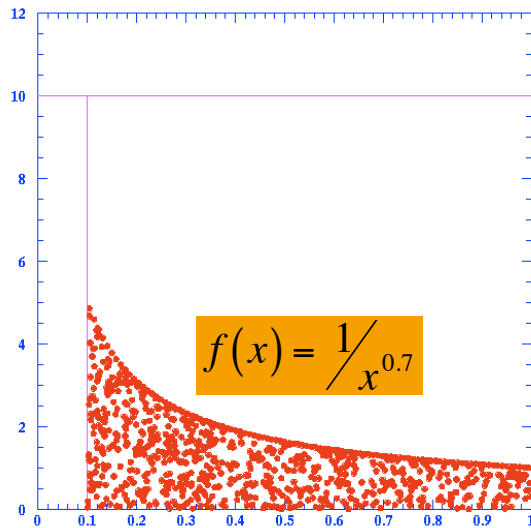


- Efficiency (ratio of hit/miss): 9% for hits example!
- 91% of “events” generated for the garbage
- Highly inefficient CPU usage!

➔ Can we do better?

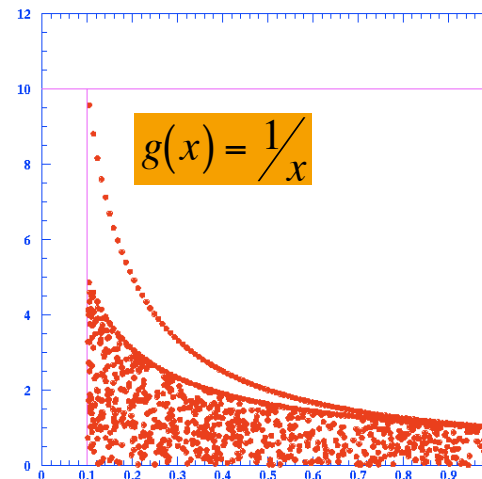
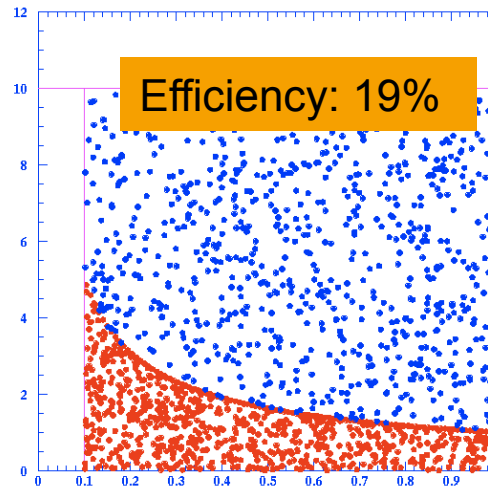
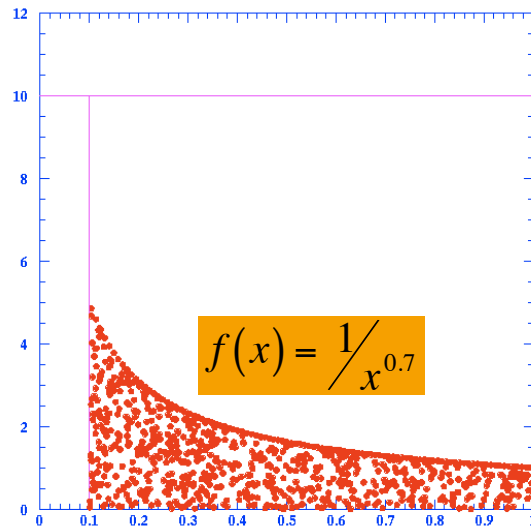


- Remember function generation by hit&miss: Increase efficiency by choosing better function maximum!



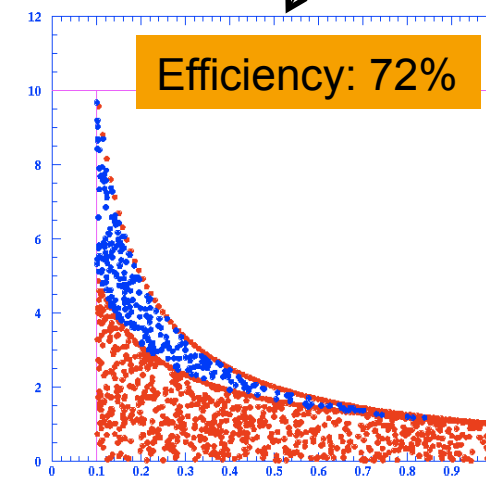
... now try something similar to
distribution generation ...
Choose $g(x) \sim 1/x$...

- Remember function generation by hit&miss: Increase efficiency by choosing better function maximum!



Side effect: potential reduction of variance:

$$\sigma = \frac{\sqrt{V(f)/V(g)}}{\sqrt{N}}$$



> Mathematically: Solve

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

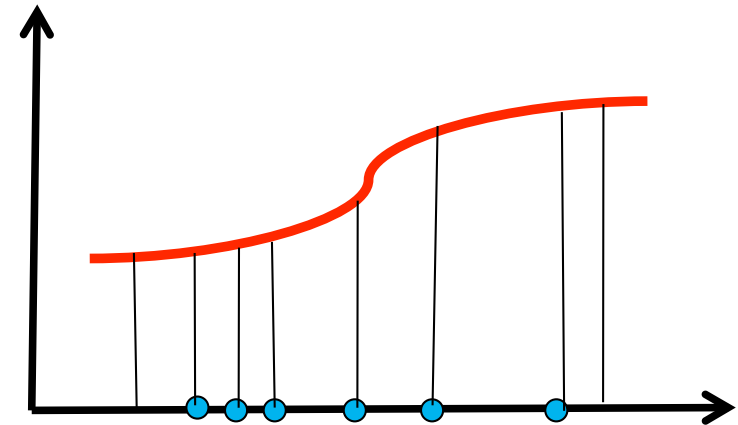
(Mean value theorem)

> Remember **Law of Large Numbers**:

$$\tilde{E}(f(x)) = \frac{1}{N} \sum_{i=1}^N f(u_i) \rightarrow \frac{1}{b-a} \int_a^b f(u) du$$

MC estimate converges to true integral:

$$I \approx I_{MC} = \frac{b-a}{N} \sum_{i=1}^N f(x_i)$$



> Remember **Central Limit Theorem**:

- MC estimate is asymptotically normally distributed, approaching Gaussian density with

$$\sigma = \frac{\sqrt{V(f)}}{\sqrt{N}} \propto \frac{1}{\sqrt{N}}$$

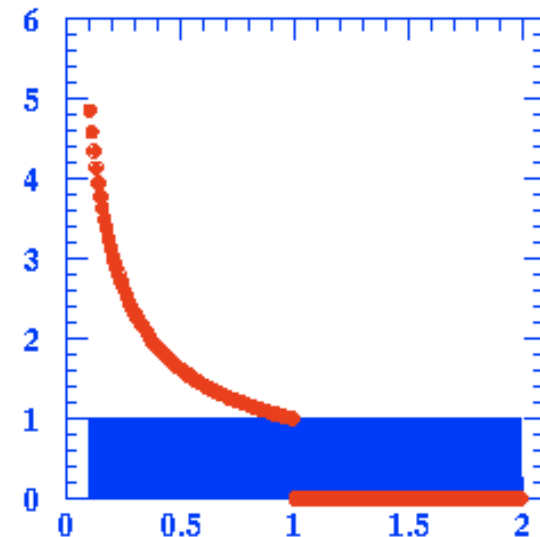
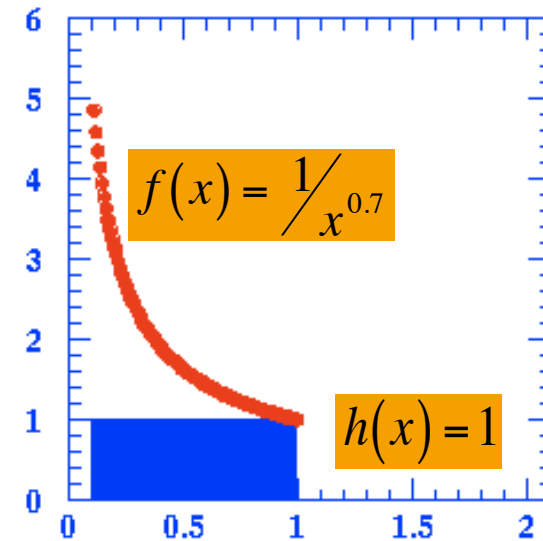
→ to decrease estimate uncertainty, increase N!
(or try to reduce the variance of the relevant function)

> **Importance sampling:** increase efficiency;
start with `barely relevant' sampling

→ no sense to sample where
function $f(x)$ is 0 (or small).

→ extending the integration (and the underlying probability density) to regions which don't contribute is ... a waste of resources.

→ concentrate on the relevant regions, and invest more CPU time in them!



➤ **Importance sampling:** increase efficiency;
start with 'barely relevant' sampling

➔ no sense to sample where
function $f(x)$ is 0 (or small).

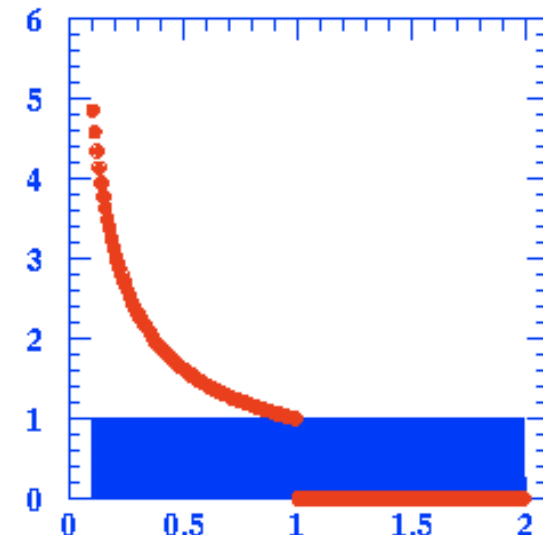
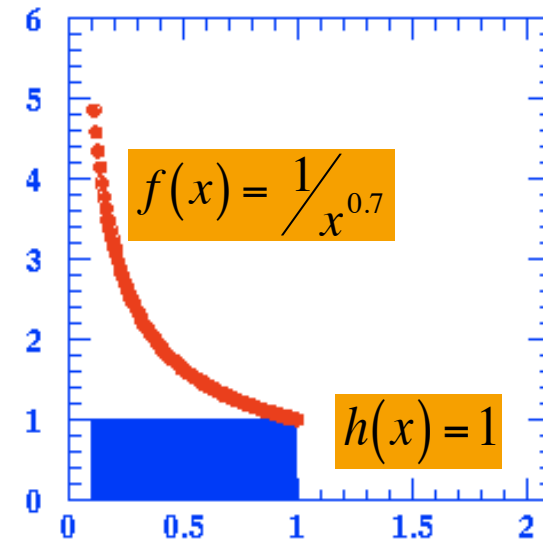
➔ extending the integration (and the underlying probability density) to regions which don't contribute is ... a waste of resources.

➔ concentrate on the relevant regions, and invest more CPU time in them!

➔ Some mathematics:
[mean value theorem] $I[f(x)] = (b-a)E[f(x)]$

$$E[f(x)] = E_h[f(x)] = \int f(x)dx = \int f(x)h(x)dx$$

Mostly assuming constant PDF ($h(x)=1$ here)

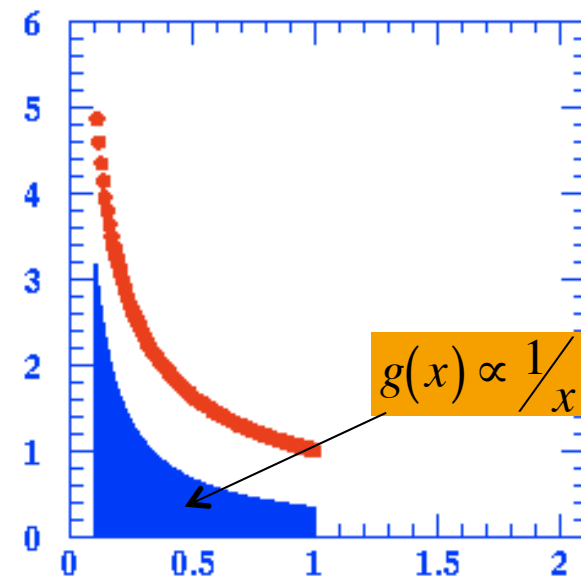
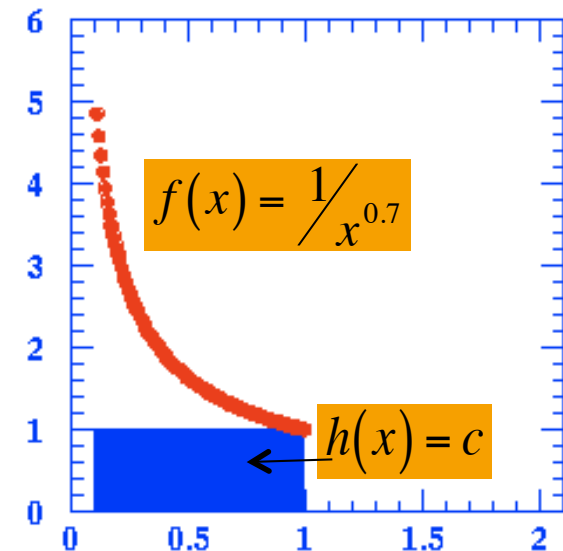


- > Importance sampling more rigorously: $I[f(x)] = (b-a)E[f(x)]$
 → Introduce a **better suited PDF** $g(x)$:

$$\begin{aligned} E_h[f(x)] &= \int f(x)h(x)dx = \int f(x)h(x)\frac{g(x)}{g(x)}dx \\ &= \int f(x)\frac{h(x)}{g(x)}g(x)dx \equiv \int f(x)w(x)g(x)dx \\ &= E_g[f(x)w(x)] \end{aligned}$$

- Thus we can calculate the integral I by
 - Generating a sample x_i according to $g(x)$
 - Using $\frac{1}{n} \sum_{i=1}^n f(x_i) \xrightarrow{n \rightarrow \infty} E_h[f(x)]$ get I as:

$$\begin{aligned} I[f(x)] &= (b-a)E_h[f(x)] = E_g[f(x)w(x)] \\ &= (b-a)\frac{1}{n} \sum_{i=1}^n f(x_i)w(x_i) \end{aligned}$$



- > Importance sampling more rigorously: $I[f(x)] = (b-a)E[f(x)]$
 → Introduce a **better suited PDF** $g(x)$:

$$\begin{aligned} E_h[f(x)] &= \int f(x)h(x)dx = \int f(x)h(x)\frac{g(x)}{g(x)}dx \\ &= \int f(x)\frac{h(x)}{g(x)}g(x)dx \equiv \int f(x)w(x)g(x)dx \\ &= E_g[f(x)w(x)] \end{aligned}$$

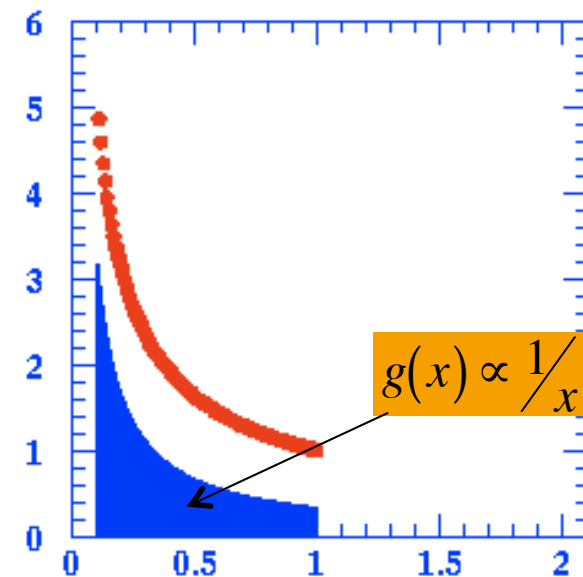
- Thus we can calculate the integral I by
 - Generating a sample x_i according to $g(x)$
 - Using $\frac{1}{n} \sum_{i=1}^n f(x_i) \xrightarrow{n \rightarrow \infty} E_h[f(x)]$ get I as:

$$\begin{aligned} I[f(x)] &= (b-a)E_h[f(x)] = E_g[f(x)w(x)] \\ &= (b-a)\frac{1}{n} \sum_{i=1}^n f(x_i)w(x_i) \end{aligned}$$

Requirements on $g(x)$:

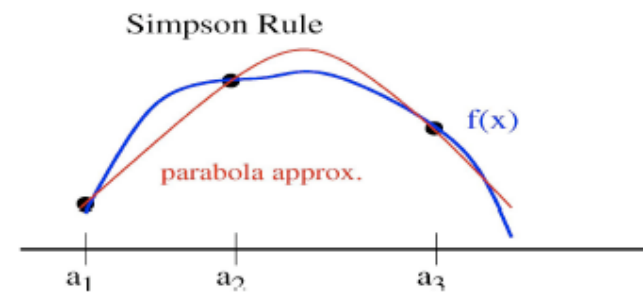
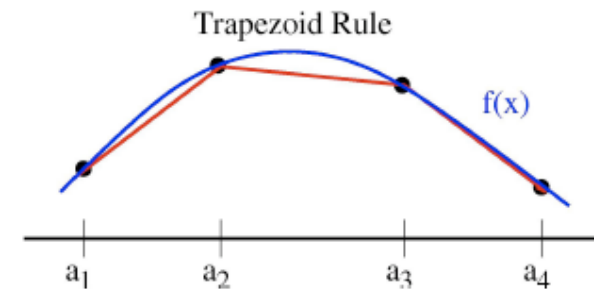
- Integrable
- simple
- close in shape to $f(x)$
- Efficient to generate
- ...

→ Not always easy to find!



➤ Comparison of MC integration with other numerical methods (from Hannes Jung):

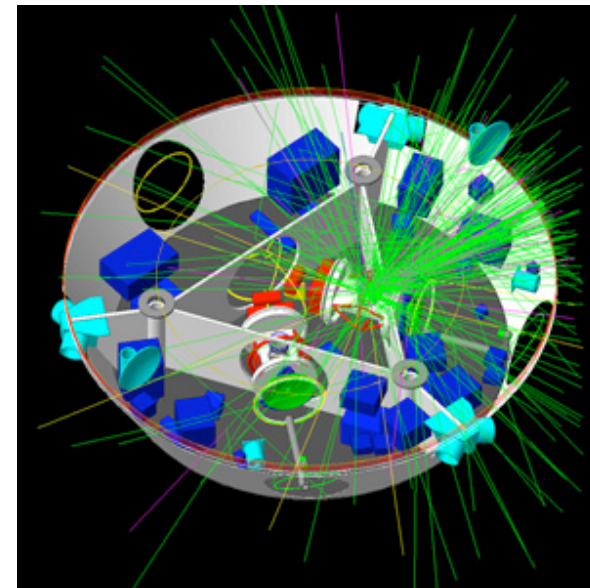
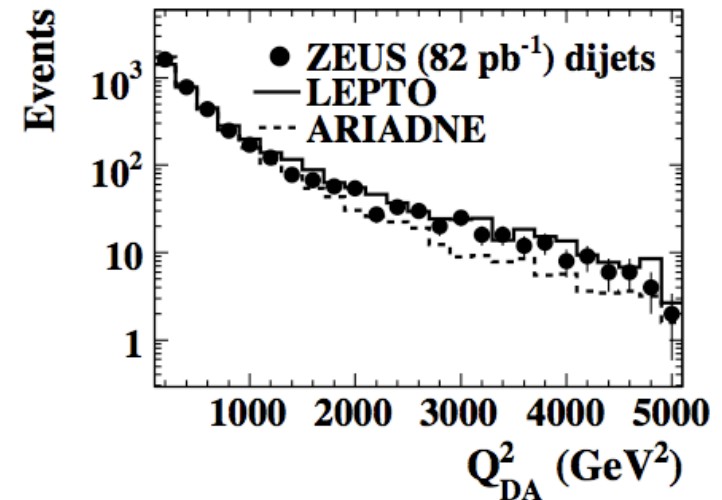
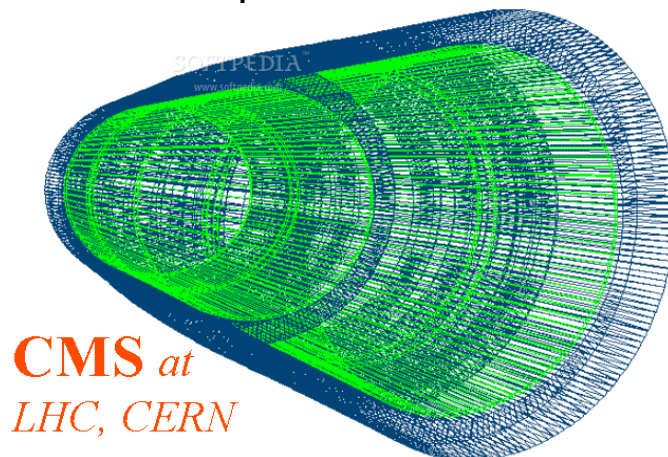
- Monte Carlo: Hit & Miss
- Trapezoidal Rule: approximate integral in subinterval by area of trapezoid below (above) curve
- Simpson quadrature: approximate by parabola
- Gauss quadrature: approximate by higher order polynomial



➔ Especially for higher dimensions, MC integration wins very often!

method	err (1d)	error
MC	$n^{-1/2}$	$n^{-1/2}$
Trapez	n^{-2}	$n^{-2/d}$
Simpson	n^{-4}	$n^{-4/d}$
Gauss	n^{-2m+1}	$n^{-(2m-1)/d}$

- > “MC generators”: Computer programs to simulate “arbitrarily” complex physics following some distribution.
 - Higher-order calculations, parton shower ...
 - HERWIG, PYTHIA, etc.
- > Statistics: Markov chains etc.
- > Detector simulation programs
 - Simulate interactions of particles with the matter.
 - ... for example for detector corrections.



SUMMARY

- > “MC methods”: no unique definition, but ...
 - ... invoking law of large numbers to approximate expectations ...
 - ... making use of “random” numbers for sampling purposes.
- > (Pseudo)Random numbers:
 - Necessary ingredient, but not too easy to obtain. Different methods with more or less good properties.
- > Generating distributions ...
 - ... by means of hit&miss or, for example, inversion of the cumulative function, ...
- > Integration as one of the main applications.
 - ... hit and miss, ...
 - ... importance sampling ...
- > Numerous applications
 - ... in HEP
 - ... and elsewhere.