Statistical Methods in Particle Physics

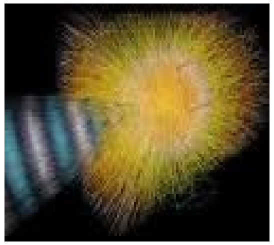
Lecture 5

November 7, 2011

Silvia Masciocchi, GSI Darmstadt
s.masciocchi@gsi.de

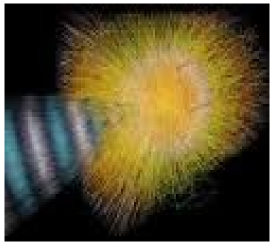
Winter Semester 2011 / 12

Outline



- Monte Carlo methods
  - Transformation method
  - Integration
- Event generators
- Detector simulation

Monte Carlo

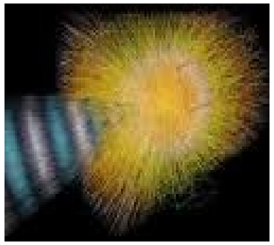


Monte Carlo methods are a class of computational algorithms that rely on repeated random sampling to compute their results.

Monte Carlo methods are often used in computer simulations of physical and mathematical systems.

These methods are most suited to calculation by a computer and tend to be used when it is infeasible to compute an exact result with a deterministic algorithm

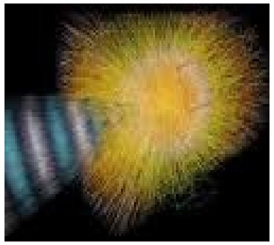
The name



The Monte Carlo method was coined in the **1940s** by John von Neumann, Stanislaw Ulam and Nicholas Metropolis, while they were working on **nuclear weapon projects (Manhattan Project)** in the Los Alamos National Laboratory. It was named in homage to the **Monte Carlo Casino**, a famous casino, where Ulam's uncle would often gamble away his money

Random processes were used extensively for the first time to predict theoretically the interaction of neutrons with matter

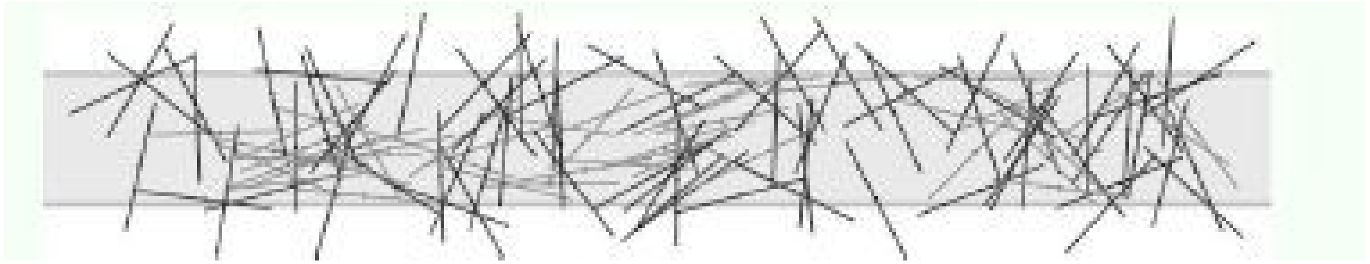
History - 1



An early variant of the Monte Carlo method can be seen in the

Buffon's needle experiment

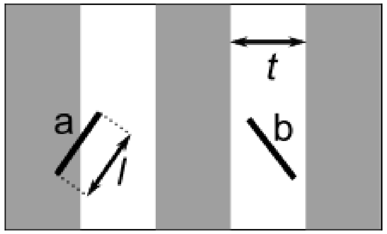
in which π can be estimated by dropping needles on a floor made of parallel strips of wood.



In more mathematical terms:

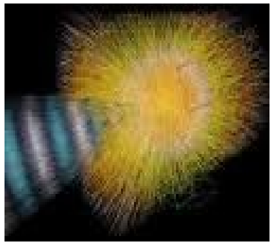
Given a needle of length l dropped on a plane ruled with parallel lines t units apart, what is the probability that the needle will cross a line?

If l ≤ t then: $P = \dfrac{2l}{t\pi}$



Use this to estimate π !

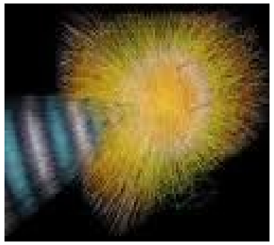
History - 2



- In the 1930s, Enrico Fermi first experimented with the Monte Carlo method while studying neutron diffusion, but did not publish anything on it

- In 1946, physicists at Los Alamos Scientific Laboratory were investigating radiation shielding and the distance that neutrons would likely travel through various materials. Despite having most of the necessary data, such as the average distance a neutron would travel in a substance before it collided with an atomic nucleus or how much energy the neutron was likely to give off following a collision, the problem **could not be solved with analytical calculations**. Stanisław Ulam had the idea of using random experiments.

Monte Carlo and simulation



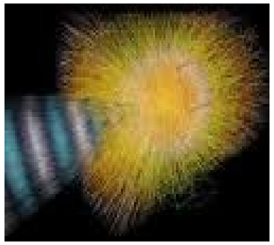
A bit subjective. Sawilowsky:

a simulation is a fictitious representation of reality, a Monte Carlo method is a technique that can be used to solve a mathematical or statistical problem, and a Monte Carlo simulation uses repeated sampling to determine the properties of some phenomenon (or behavior). Examples:

- Simulation: Drawing one pseudo-random uniform variable from the interval [0,1] can be used to simulate the tossing of a coin: If the value is less than or equal to 0.50 designate the outcome as heads, but if the value is greater than 0.50 designate the outcome as tails. This is a simulation, but not a Monte Carlo simulation.
- Monte Carlo method: The area of an irregular figure inscribed in a unit square can be determined by throwing darts at the square and computing the ratio of hits within the irregular figure to the total number of darts thrown. This is a Monte Carlo method of determining area, but not a simulation.
- Monte Carlo simulation: Drawing a large number of pseudo-random uniform variables from the interval [0,1], and assigning values less than or equal to 0.50 as heads and greater than 0.50 as tails, is a Monte Carlo simulation of the behavior of repeatedly tossing a coin.

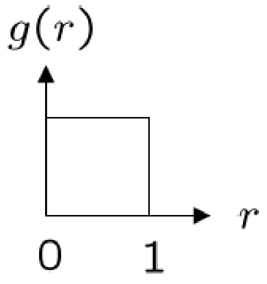
Not always so easy to distinguish

The Monte Carlo method



A numerical technique for calculating probabilities and related quantities using sequences of random numbers

The usual steps:

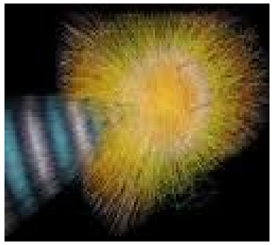


- Generate sequence $r_1, r_2, ..., r_m$ uniform in [0,1]
- Use this to produce another sequence $x_1, x_2, ..., x_m$ distributed according to some pdf f(x) in which we are interested (x can be a vector)
- Use the x values to estimate some property of f(x) e.g., fraction of x values with a < x < b gives $\int_a^b f(x)\,dx$

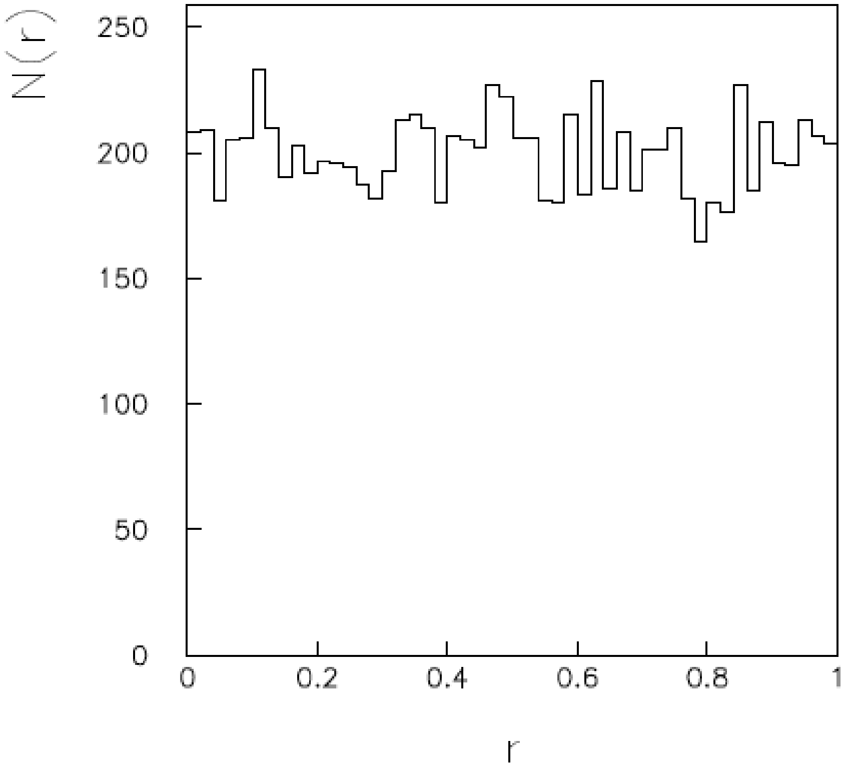
Applications:

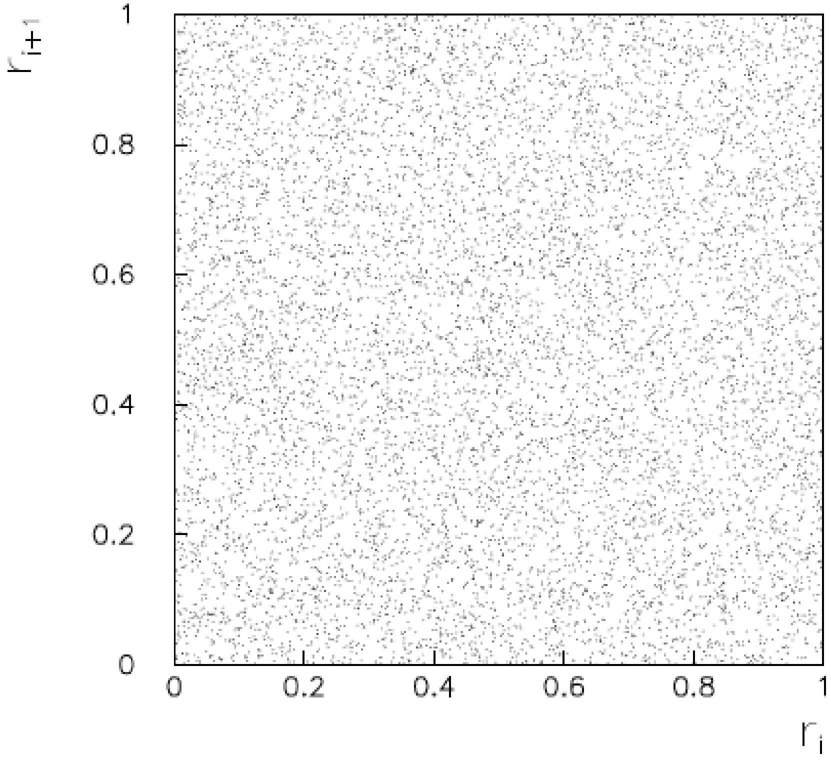
- MC calculation for integration
- Simulation to test statistical procedures

Random numbers

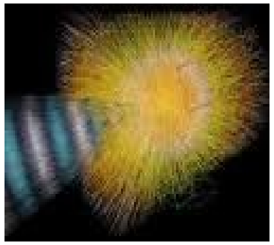


- Extensively appreciated with Nik



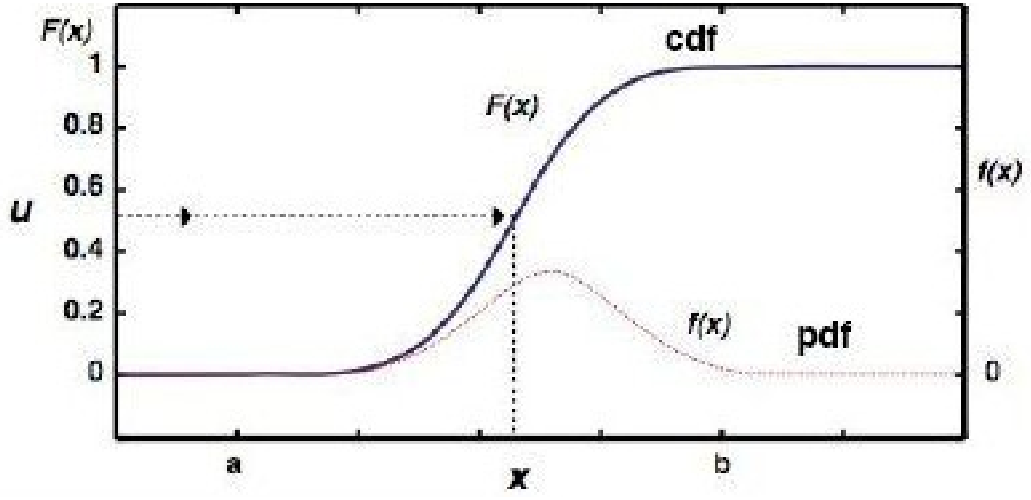


Transformation method



- Random variable x
- Pdf f(x)
- cumulative distribution function

$$F(x) = \int_{-\infty}^{x} f(t)dt$$



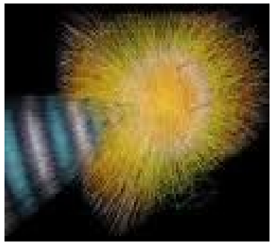
- Case of F(x) analitically invertible:

$$x = F^{-1}(u)$$

- If u_i uniform in [0,1], then $x_i = F^{-1}(u_i)$ follow pdf f(x)

Method is applicable if F(x) and $F^{-1}(u)$ are analytically solvable

Example: exponential distribution



Consider random numbers according to:

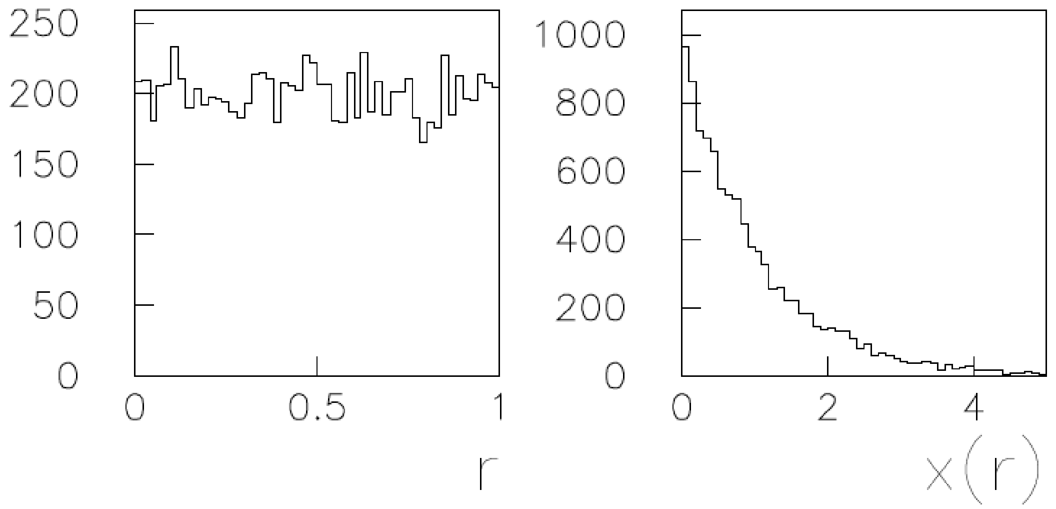
$$f(x,\lambda) = \begin{cases} \lambda e^{-\lambda x}, & x > 0 \\ 0, & x \leq 0 \end{cases}$$

Computation of the inverse pdf:

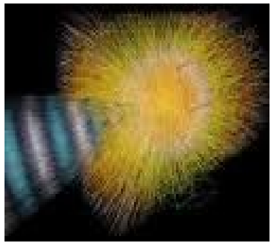
$$\int_{-\infty}^{x} f(t)\,dt = \int_0^x \lambda e^{-\lambda t}\,dt = [e^{-\lambda t}]_0^x = 1 - e^{-\lambda x} = r(x)$$

And solve for x(r).

$$\rightarrow \quad x(r) = -1/\lambda \ln(1-r)$$

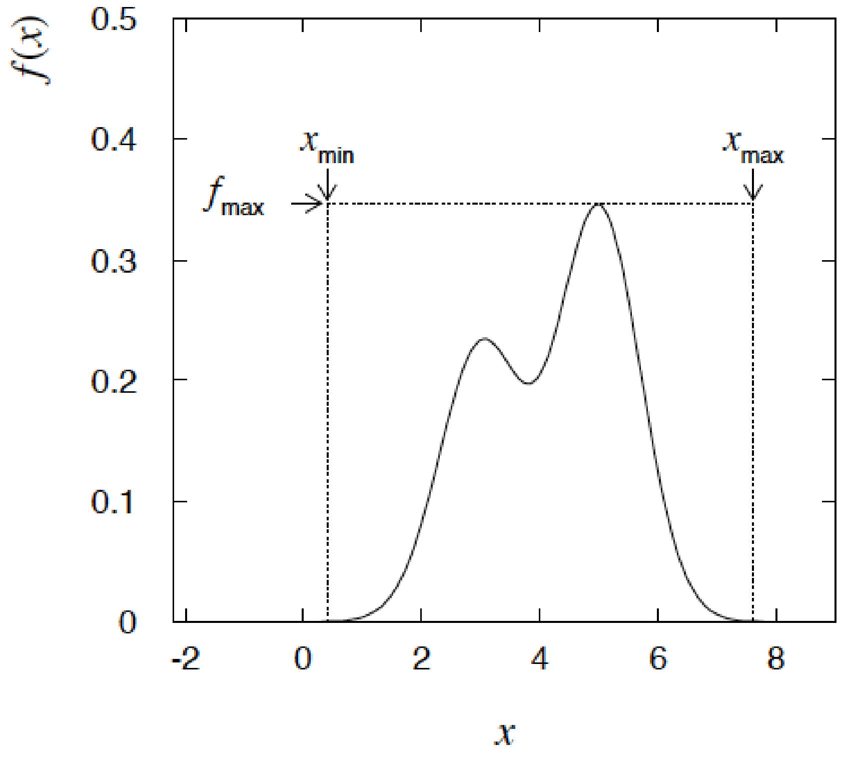


Hit and miss method (von Neumann)



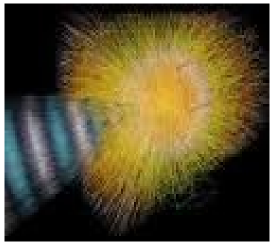
If $F^{-1}(u)$ is not computable, then use “hit and miss” method (also called acceptance-rejection method)

Enclose the pdf in a box:



- Generate a random number x, uniform in $[x_{\text{min}},x_{\text{max}}]$, i.e. $x={}_{\text{xmin}}+r_1(x_{\text{max}} - x_{\text{min}})$ r_1 uniform in [0,1]
- Generate a second indipendent random number uniformly distributed between 0 and fmax, i.e. $u=r_2\, f_{\text{max}}$
- If u < f(x), then accept x if not, reject x and repeat

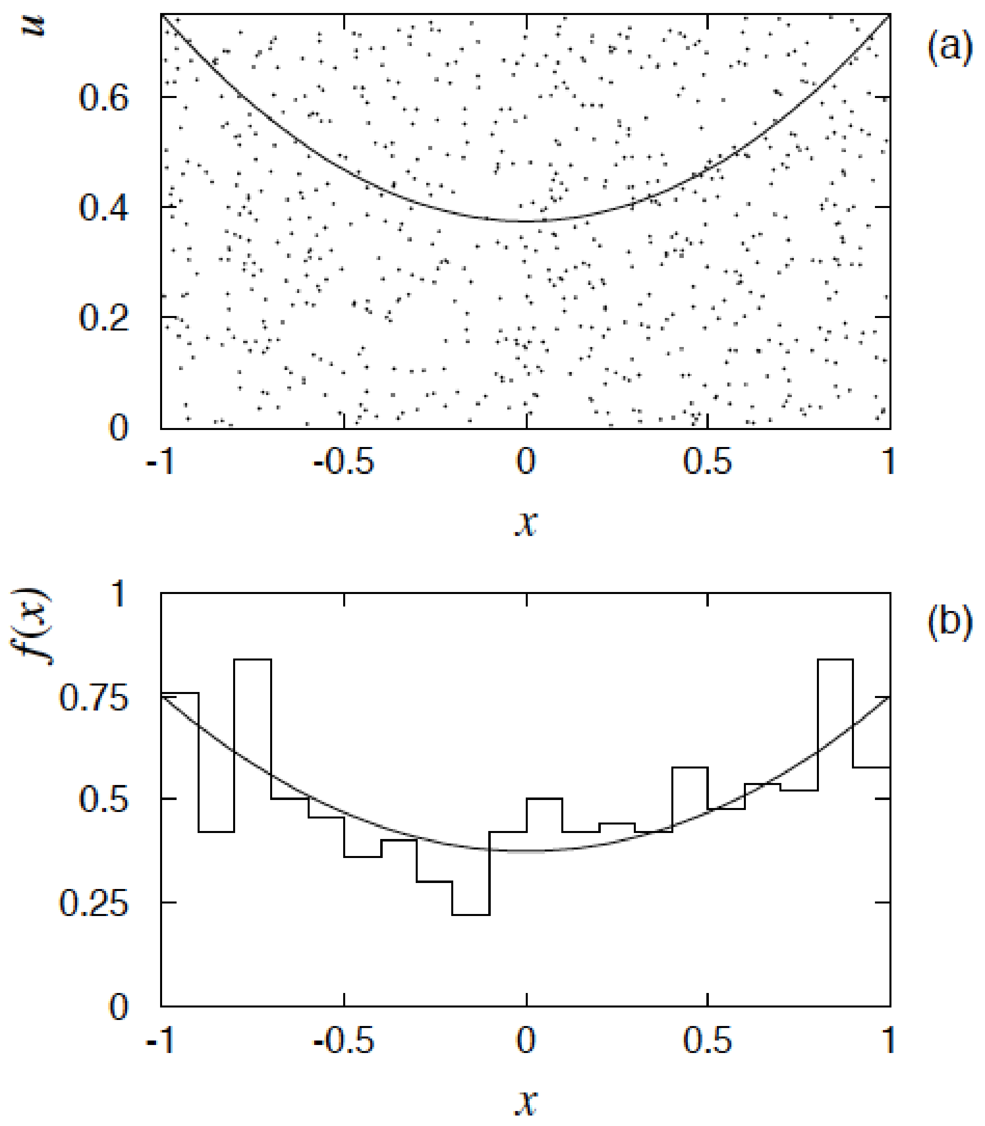
Example



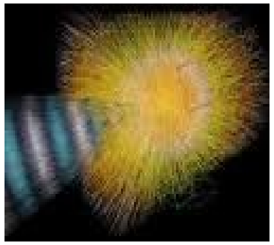
$$f(x) = \frac{3}{8}(1 + x^2)$$

$$(-1 \leq x \leq 1)$$

If dot below curve, use x value in histogram.



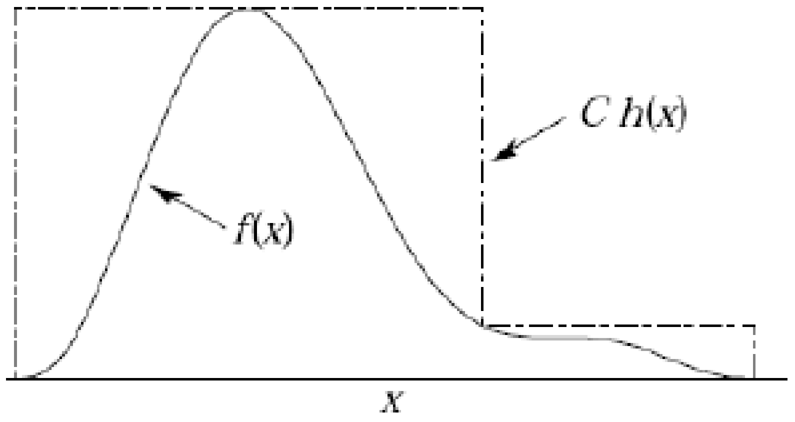
Improve efficiency of hit and miss



The fraction of accepted points is equal to the fraction of the box's area under the curve.

For very peaked distributions, this may be very low and thus the algorithm may be slow.

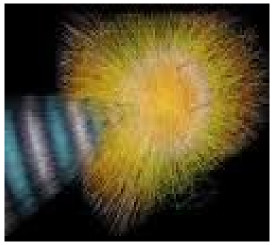
Improve by enclosing the pdf $f(x)$ in a curve $C\,h(x)$ that conforms to $f(x)$ more closely, where $h(x)$ is a pdf from which we can generate random values and C is a constant.



Generate points uniformly over $C\,h(x)$.

If point is below $f(x)$, accept x.

Monte Carlo integration

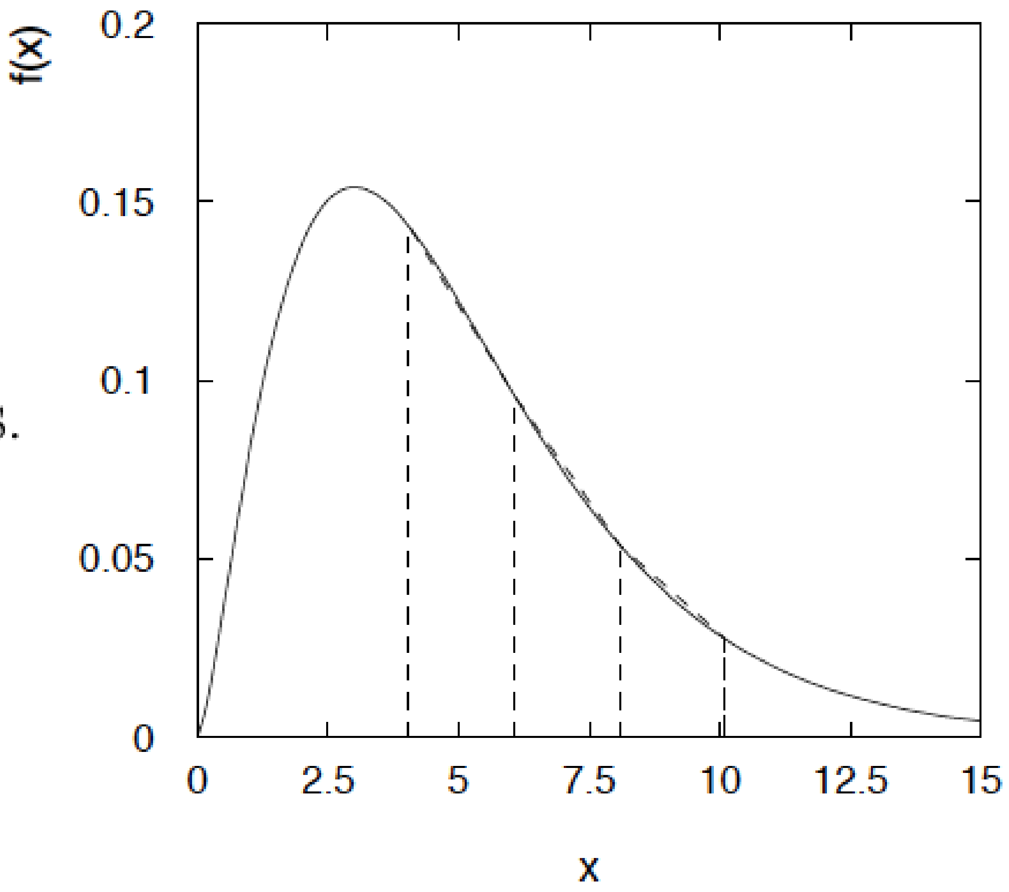


Check accuracy of the method:

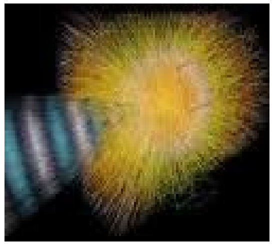
MC calculation = integration.

Compare to trapezoidal rule,

n = number of computing steps.



Accuracy of Monte Carlo integration



For 1-dimensional integral:

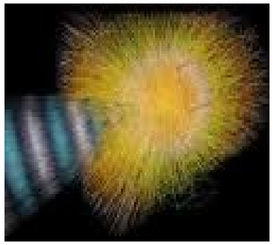
MC:	$n \propto$ number of random values generated
	accuracy $\propto 1/\sqrt{n}$
Trapezoid:	$n \propto$ number of subdivisions
	accuracy $\propto 1/n^2$

Trapezoid wins! But in d dimensions this becomes

MC:	accuracy $\propto 1/\sqrt{n}$	$\leftarrow$ independent of d !
Trapezoid:	accuracy $\propto 1/n^{2/d}$	

For high enough d (d>4), MC always wins

Monte Carlo: statistical experiments



Often an analytical treatment of physical problems is either difficult or impossible. Therefore we do either an approximation or use a statistical description (via Monte Carlo)

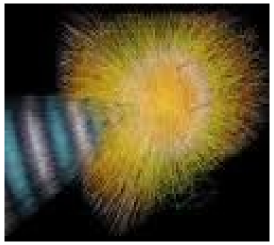
APPLICATIONS:

- High energy and nuclear physics
- Numerical calculations (integration, differentiation)
- Coding/encoding (e.g. secure connections, like ssh)
- Reliability tests
- Investment banking
- Earth sciences

METHODS:

- Find a statistic model
- Produce random numbers properly
- Calculate estimators from random quantities

Monte Carlo: statistical experiments



Qualitatively:

Scattering experiment:

Measure the angular distribution of particles scattered from protons in a fixed target

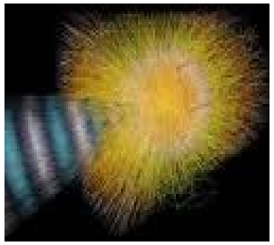
Ingredients:

- Magnitude and direction of the momentum vector of the incident particles
- Probability that a particle will collide with a proton in the target
- Resulting momentum vectors of the scattered particles

All are described in terms of probability distributions !

The final experimental result can be treated in terms of a multiple integration over all these probability distributions !!

A very simple example: K_S^0

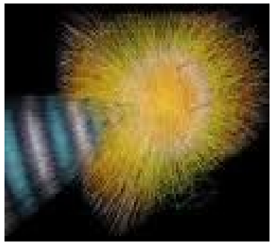


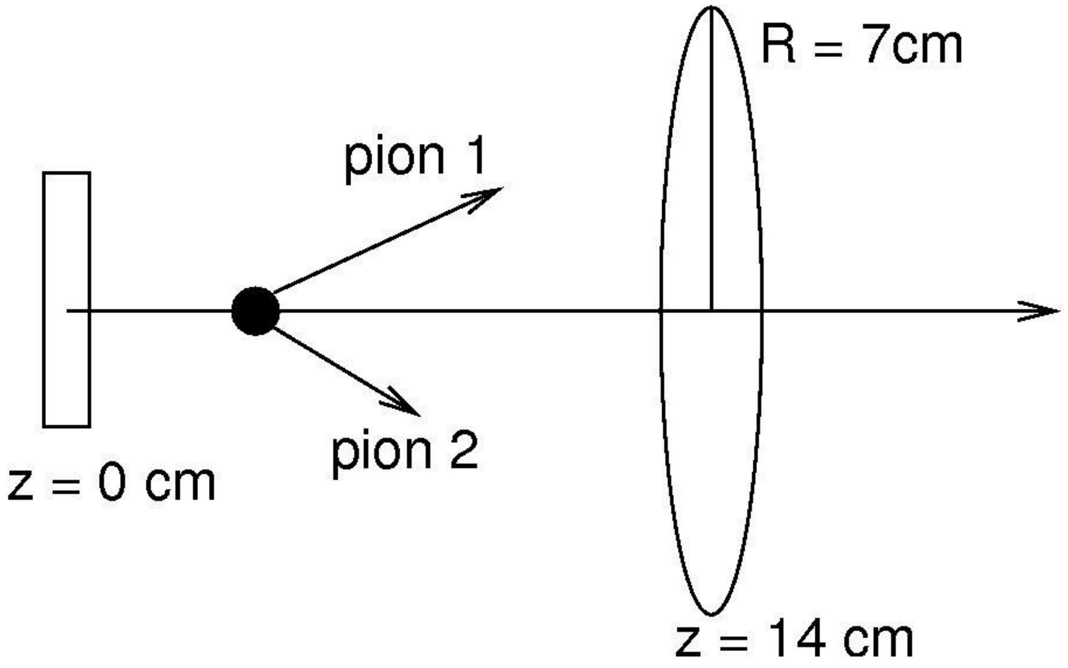
- K_S^0 mesons: mass M=0.489 GeV
- A source produces narrow, mono-energetic beam, with E=0.886 GeV
- K_S^0 have lifetime τ = 8,934 x 10^{-10} sec, $c\tau$ = 2.68 cm. Most K_S^0 decay into $\pi^+\pi^-$ (BR= 69.2%, Data Particle Group)
- In the rest system of the K_S^0, the decay is isotropic

EXPERIMENT

- At a distance of 14 cm behind the source, there is a silicon detector to register the decay pions
- The detector is a circular disk, with radius R=7 cm, centered on the kaon beam
- Determine the probability that both pions from the decay $K_S^0 \rightarrow \pi^+\pi^-$ hit the detector

K_S^0 mesons

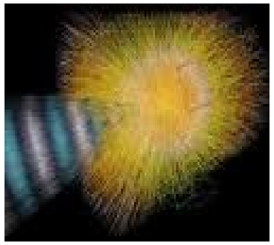




Determine the probability that both pions from the decay $K_S^0 \rightarrow \pi^+\pi^-$ hit the detector

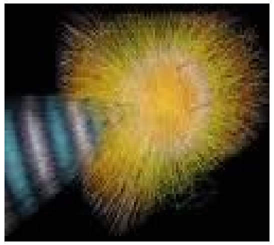
- A. Analytical calculation
- B. Direct simulation

K_s^0 mesons: analytical calculation - 1



- Parametrize the decay in the K_s^0 rest frame:
  - Uniform in φ around the beam direction
  - Cosine of the polar angle to the beam is uniform: $0 < \cos\theta < 1$
- Perform Lorentz boost of the decay pions to the laboratory system
  - Boost parameter is $\gamma = E/M$
- Describe the flight distance z as an exponential distribution:
  - Probability density: $$\rho(z) = \frac{1}{\langle z\rangle} e^{\frac{z}{\langle z\rangle}}$$
  - In the lab system: $$\langle z\rangle = \langle c\tau\rangle \sqrt{\gamma^2 - 1} \approx 4.07\,\mathrm{cm}$$
- Determine the allow z range from which both pions reach the detector as a function of the decay configuration, and from there compute the answer ...

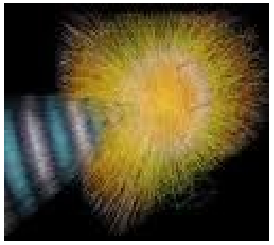
K_s^0 mesons: analytical calculation - 2



$$z_{min} = D - R\gamma\sqrt{\frac{1-\cos\theta}{1+\cos\theta}} \leq z \leq D = z_{max}$$

$$A = \int_0^1 d\cos\theta \int_{z_{min}}^{z^{max}} dz\rho(z) \approx 0.207016$$

K_S^0 mesons: direct simulation - 1

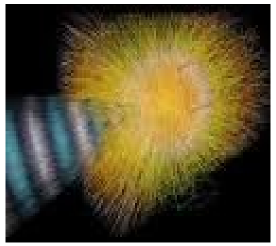


- Generate decay position according to ρ(z)
- Generate isotropic decays in the K_S^0 rest frame
- Perform Lorentz transformation to the lab system
- Analyze the events, check whether the pions hit the detector
  - Count as “success” cases where both pions hit the detector (n)
  - Count as “failure” if at least one particle misses
- Repeat N times. Probability p = n/N

$$A = \frac{n}{N} \pm \sqrt{\frac{p(1-p)}{N}}$$

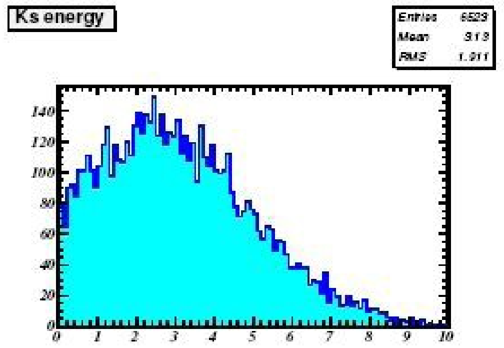
- Example: N = 106, successes n = 207311
  A = 0.207311 ± 0.000405   versus the analytical result A = 0.207016

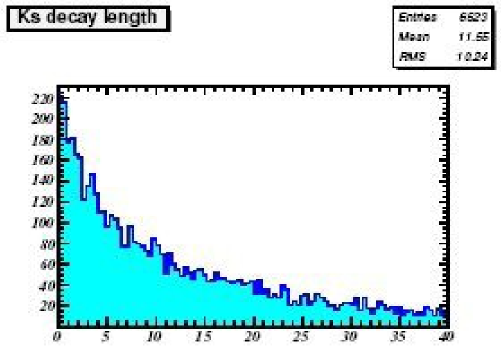
K_S^0 mesons: direct simulation - 2

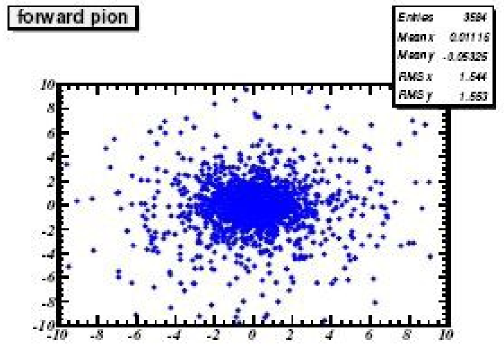


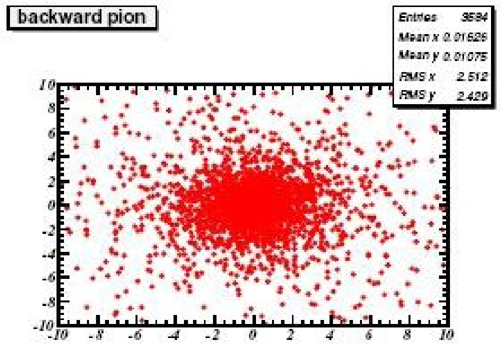
Easy to adapt to more complex cases:

- Other kaon energies
- An energy spectrum
- Rectangular detectors
  - Φ dependence
- Finite efficiency of detector





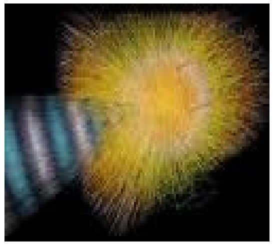




Direct simulation allows detailed studies

Although with finite statistical precision

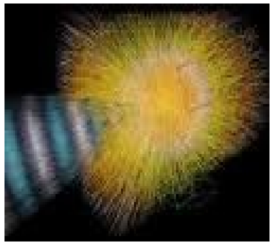
In “our” real life



Several stages:

- **Event generator:**
  - Simulation of the PHYSICS process
  - Colliding particles
  - Cross section, processes involved, fragmentation ...
- **Detector simulation:**
  - Interaction of the produced particles with the material
  - Realistic description of the experimental apparatus
    - efficiencies
    - defects (dead channels, etc)
    - misalignment

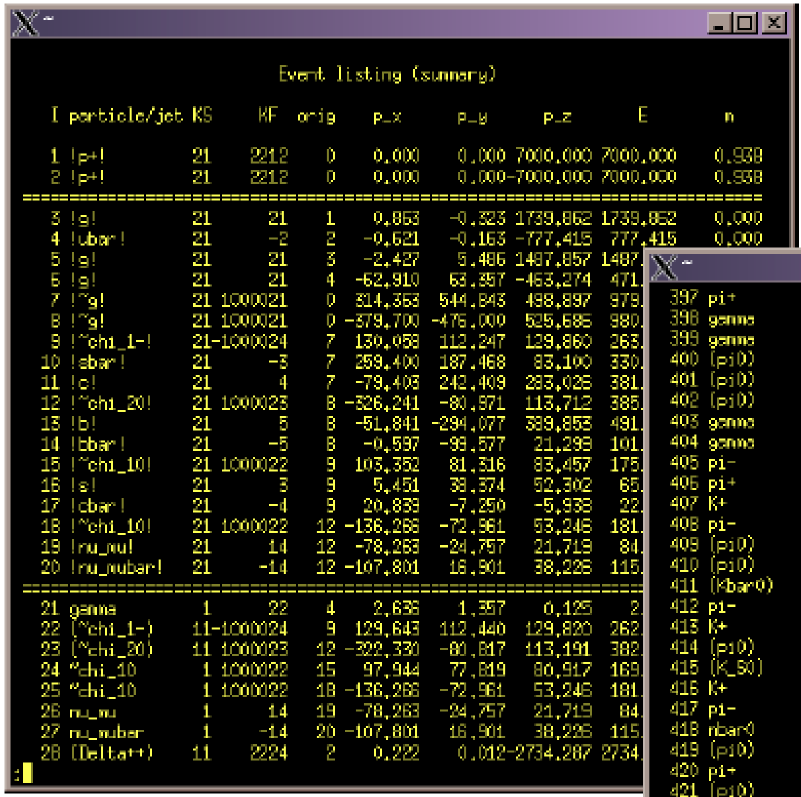
Event generators



$e^+e^- \rightarrow$ hadrons	JETSET (PYTHIA) HERWIG ARIADNE
$e^+e^- \rightarrow$ WW	KORALW EXCALIBUR ERATO
pp $\rightarrow$ hadrons	ISAJET PYTHIA HERWIG
PbPb $\rightarrow$ hadrons	HIJING

Event generators

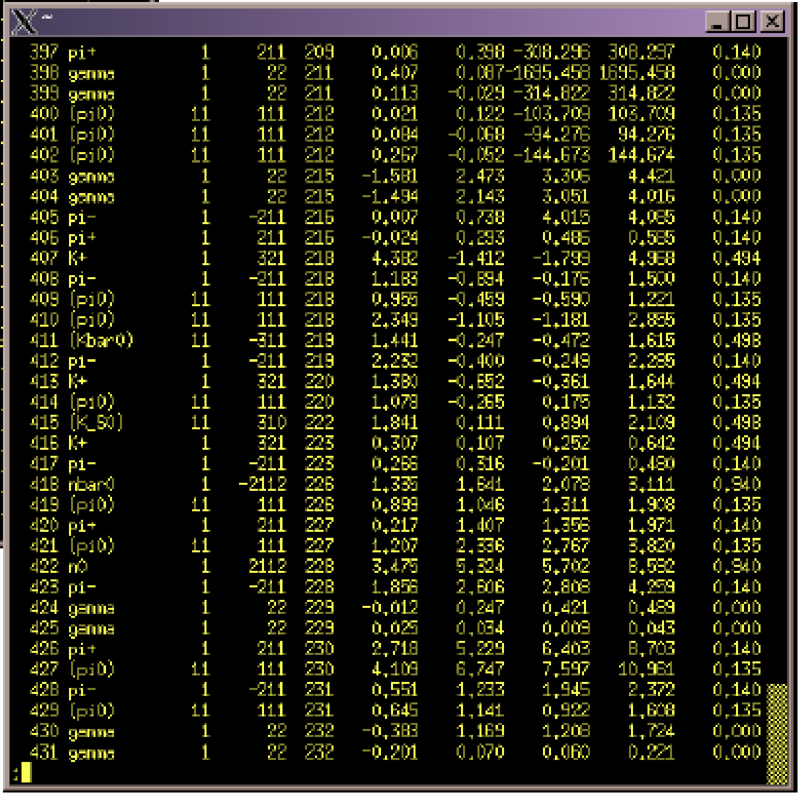
The output are so-called “events”, namely for each collision the programs give a list of final state particles, with their momentum vectors, types, angular distribution, etc



```
                      Event listing (summary)

    I particle/jet KS     KF  orig    p_x      p_y      p_z       E        m

    1 !p+!       21    2212    0     0.000    0.000 7000.000 7000.000    0.938
    2 !p+!       21    2212    0     0.000    0.000-7000.000 7000.000    0.938
 ==============================================================================
    3 !g!        21      21    1     0.853   -0.323 1739.862 1739.862    0.000
    4 !ubar!     21      -2    2    -0.621   -0.163 -777.415  777.415    0.000
    5 !g!        21      21    3    -2.427    5.486 1487.857 1487.
    6 !g!        21      21    4   -62.910   63.357 -463.274  471.
    7 !~g!       21 1000021    0   314.353  544.843  498.897  979.
    8 !~g!       21 1000021    0  -379.700 -476.000  525.685  980.
    9 !~chi_1-!  21-1000024    7   130.058  112.247  129.860  263.
   10 !sbar!     21      -3    7   259.400  187.468   83.100  330.
   11 !c!        21       4    7   -79.403  242.409  283.026  381.
   12 !~chi_20!  21 1000023    8  -326.241  -80.971  113.712  385.
   13 !b!        21       5    8   -51.841 -294.077  389.853  491.
   14 !bbar!     21      -5    8    -0.597  -99.577   21.299  101.
   15 !~chi_10!  21 1000022    9   103.352   81.316   83.457  175.
   16 !s!        21       3    9     5.451   38.374   52.302   65.
   17 !cbar!     21      -4    9    20.839   -7.250   -5.938   22.
   18 !~chi_10!  21 1000022   12  -136.266  -72.961   53.246  181.
   19 !nu_mu!    21      14   12   -78.263  -24.757   21.719   84.
   20 !nu_mubar! 21     -14   12  -107.801   16.901   38.226  115.
 ===========================================================
   21 gamma       1      22    4     2.636    1.357    0.125    2.
   22 (~chi_1-)  11-1000024    9   129.643  112.440  129.820  262.
   23 (~chi_20)  11 1000023   12  -322.330  -80.817  113.191  382.
   24 ~chi_10     1 1000022   15    97.944   77.819   80.917  169.
   25 ~chi_10     1 1000022   18  -136.266  -72.961   53.246  181.
   26 nu_mu       1      14   19   -78.263  -24.757   21.719   84.
   27 nu_mubar    1     -14   20  -107.801   16.901   38.226  115.
   28 (Delta++)  11    2224    2     0.222    0.012-2734.287 2734.
:
```



```
  397 pi+          1     211   209     0.006    0.398 -308.296  308.297    0.140
  398 gamma        1      22   211     0.407    0.087-1695.458 1695.458    0.000
  399 gamma        1      22   211     0.113   -0.029 -314.822  314.822    0.000
  400 (pi0)       11     111   212     0.021    0.122 -103.709  103.709    0.135
  401 (pi0)       11     111   212     0.084   -0.068  -94.276   94.276    0.135
  402 (pi0)       11     111   212     0.267   -0.052 -144.673  144.674    0.135
  403 gamma        1      22   215    -1.581    2.473    3.306    4.421    0.000
  404 gamma        1      22   215    -1.494    2.143    3.051    4.016    0.000
  405 pi-          1    -211   216     0.007    0.738    4.015    4.085    0.140
  406 pi+          1     211   216    -0.024    0.293    0.486    0.585    0.140
  407 K+           1     321   218     4.382   -1.412   -1.799    4.968    0.494
  408 pi-          1    -211   218     1.183   -0.894   -0.176    1.500    0.140
  409 (pi0)       11     111   218     0.955   -0.459   -0.590    1.221    0.135
  410 (pi0)       11     111   218     2.349   -1.105   -1.181    2.855    0.135
  411 (Kbar0)     11    -311   219     1.441   -0.247   -0.472    1.615    0.498
  412 pi-          1    -211   219     2.232   -0.400   -0.249    2.285    0.140
  413 K+           1     321   220     1.380   -0.652   -0.361    1.644    0.494
  414 (pi0)       11     111   220     1.078   -0.265    0.175    1.132    0.135
  415 (K_S0)      11     310   222     1.841    0.111    0.894    2.109    0.498
  416 K+           1     321   223     0.307    0.107    0.252    0.642    0.494
  417 pi-          1    -211   223     0.266    0.316   -0.201    0.480    0.140
  418 nbar0        1   -2112   226     1.335    1.641    2.078    3.111    0.940
  419 (pi0)       11     111   226     0.899    1.046    1.311    1.908    0.135
  420 pi+          1     211   227     0.217    1.407    1.356    1.971    0.140
  421 (pi0)       11     111   227     1.207    2.336    2.767    3.820    0.135
  422 n0           1    2112   228     3.475    5.324    5.702    8.592    0.940
  423 pi-          1    -211   228     1.856    2.606    2.808    4.259    0.140
  424 gamma        1      22   229    -0.012    0.247    0.421    0.489    0.000
  425 gamma        1      22   229     0.025    0.034    0.009    0.043    0.000
  426 pi+          1     211   230     2.718    5.229    6.403    8.703    0.140
  427 (pi0)       11     111   230     4.109    6.747    7.597   10.961    0.135
  428 pi-          1    -211   231     0.551    1.233    1.945    2.372    0.140
  429 (pi0)       11     111   231     0.645    1.141    0.922    1.608    0.135
  430 gamma        1      22   232    -0.383    1.169    1.208    1.724    0.000
  431 gamma        1      22   232    -0.201    0.070    0.060    0.221    0.000
:
```

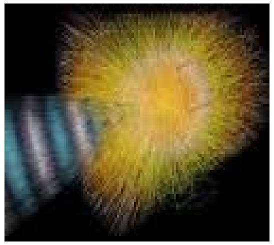
A simulated event

⋮

⋮

PYTHIA Monte Carlo
pp → gluino-gluino

Detector simulation



INPUT: particle list (with their species and momenta) from the event generator

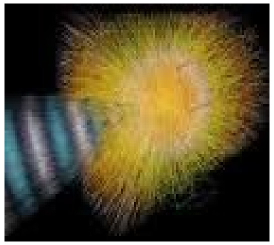
Simulate the detector response taking into account:

- Multiple Coulomb scattering (generate scattering angle)
- Particle decays (generate lifetime), nuclear knock-out
- Ionization energy loss (generate Δ, Landau)
- Bremsstrahlung
- Electromagnetic / hadronic showers

CERN

GEANT

Detector simulation



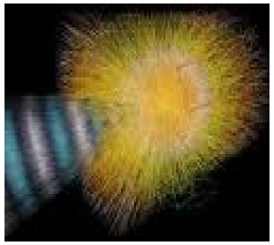
GEANT

- First version ~ 1974
- Till GEANT 3.21, in FORTRAN
- Since ~ 2000, FORTRAN version no longer developed, bug fixes

- Geant4: in C++, with a modern object-oriented design

- Next … Geant 5

Detector simulation



The detector simulation takes into account many more things:

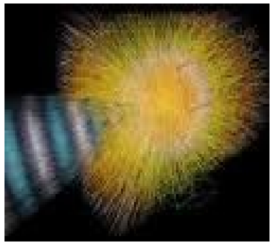
- Production of the signals (electronics response)
- Addition of detector noise

- Description of the detection **efficiency** for each detector component (experiment specific)

- Position and energy **resolution** of each detector component (experiment specific)

The output is a list of digitized signals from all detector components, exactly like real data!! (or data format input for the reconstruction)

Use of simulations



- Develop reconstruction algorithms

  (particle trajectories in the tracking detectors, showers in the calorimeters)

- Optimize trigger selection

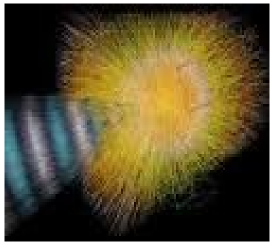
- Identify the best signal signature

- Compute efficiency of selections in real data analysis

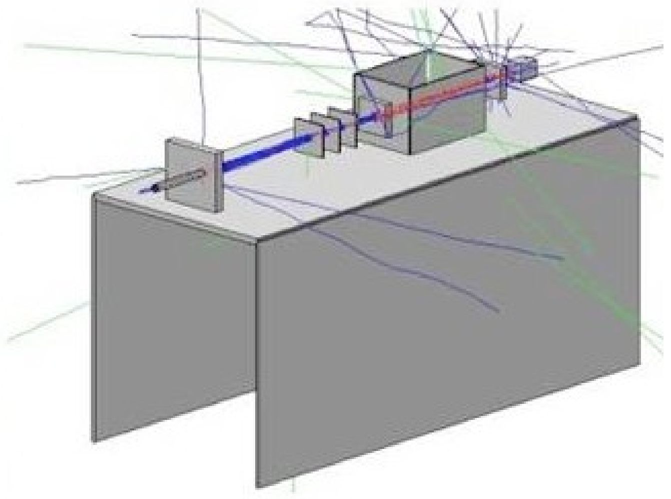
- During design of an experiment: define the detector acceptance, etc

Simulation is absolutely crucial in the planning phase of experiments, for preparation of data taking, to optmize analyses, to evaluate the significance of the results

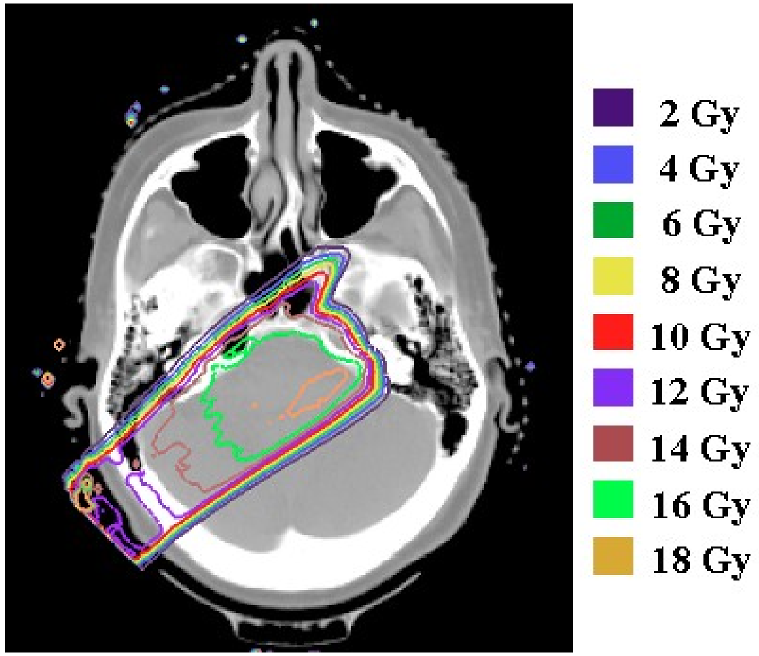
Geant – other applications



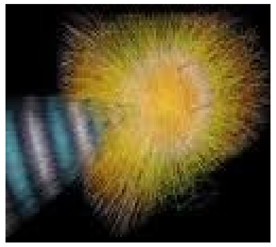
Proton beam eye therapy unit at the Laboratori Nazionali del Sud (INFN) in Catania (left) and a display from the Geant4 advanced example for the simulation of the same beam line (right).

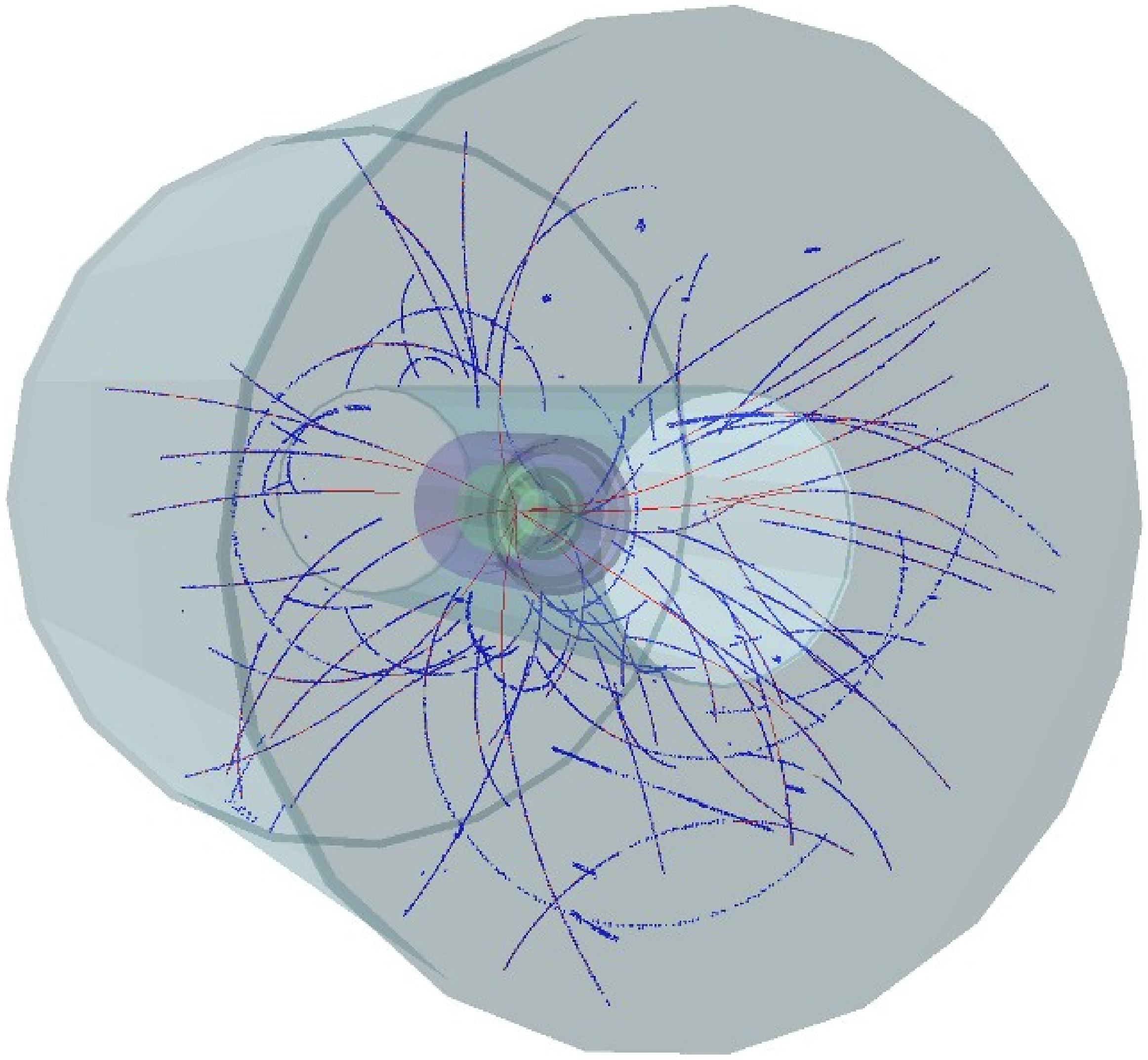


Geant4 simulated dose contours in a human brain with a proton beam.

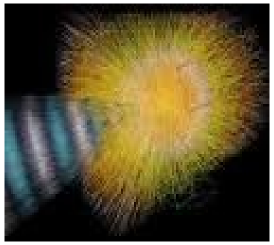


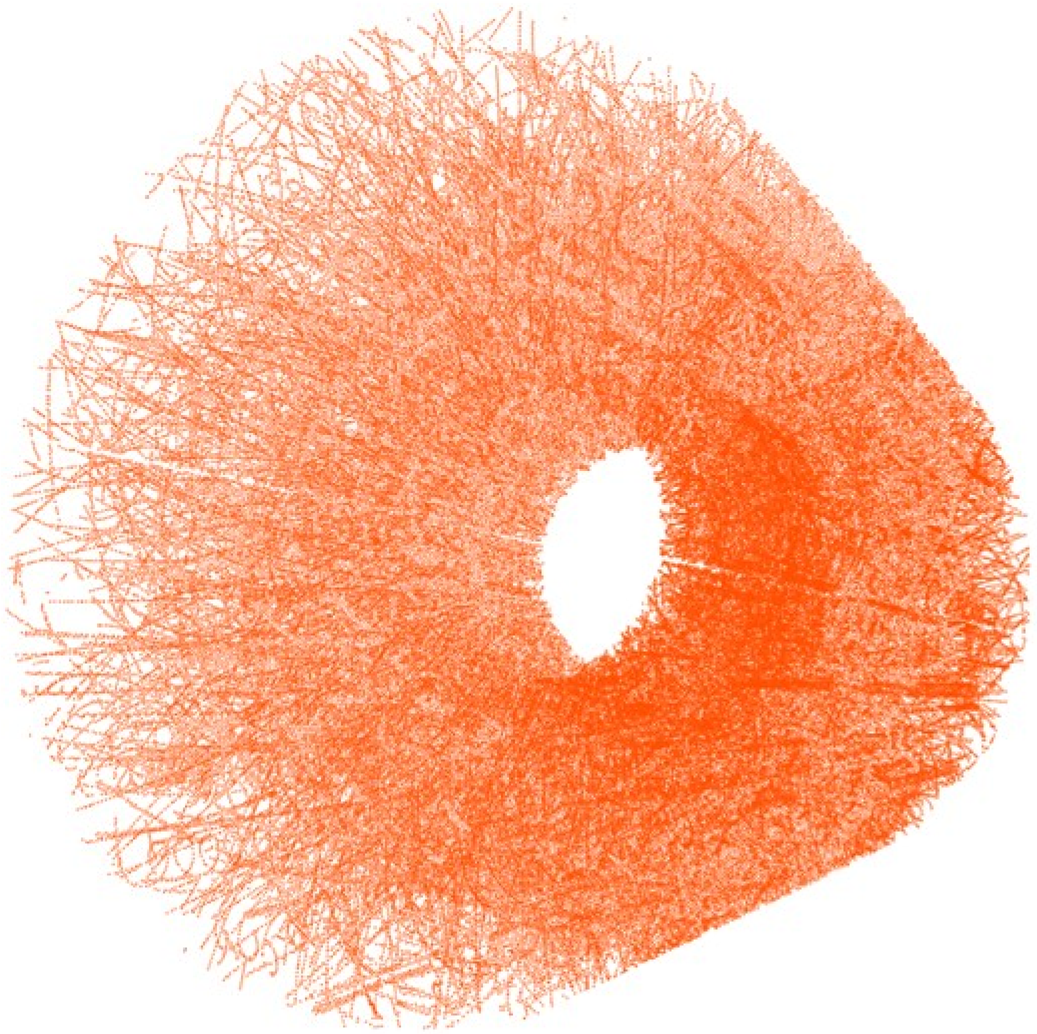
ALICE (TPC) simulation: proton-proton



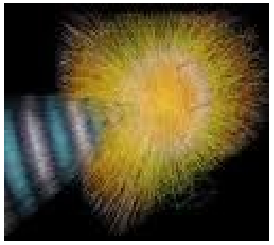


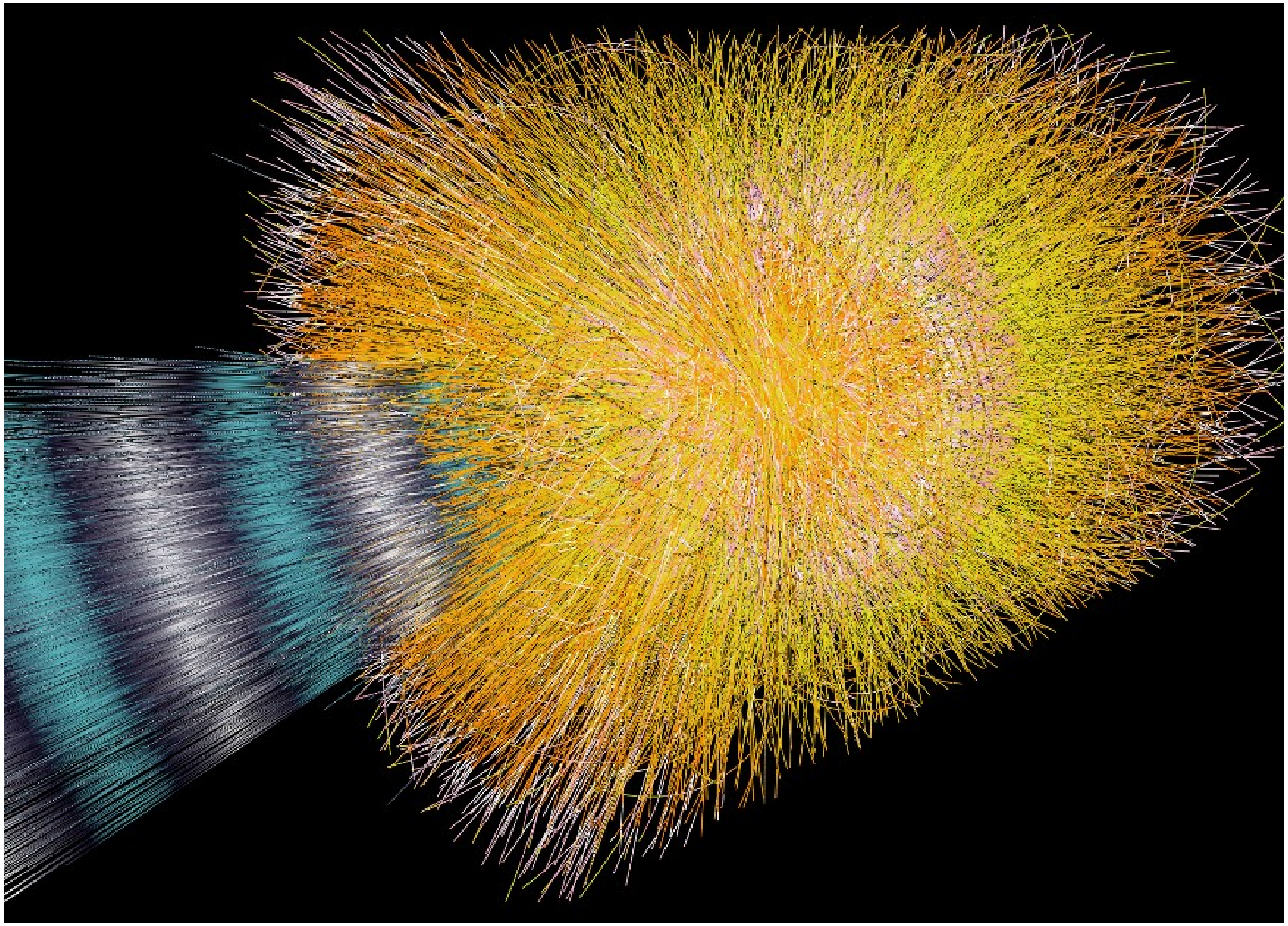
ALICE (TPC) simulation: Pb-Pb





ALICE simulation: Pb-Pb

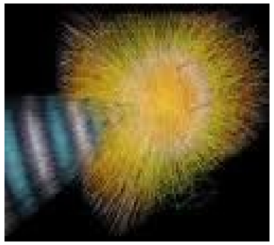


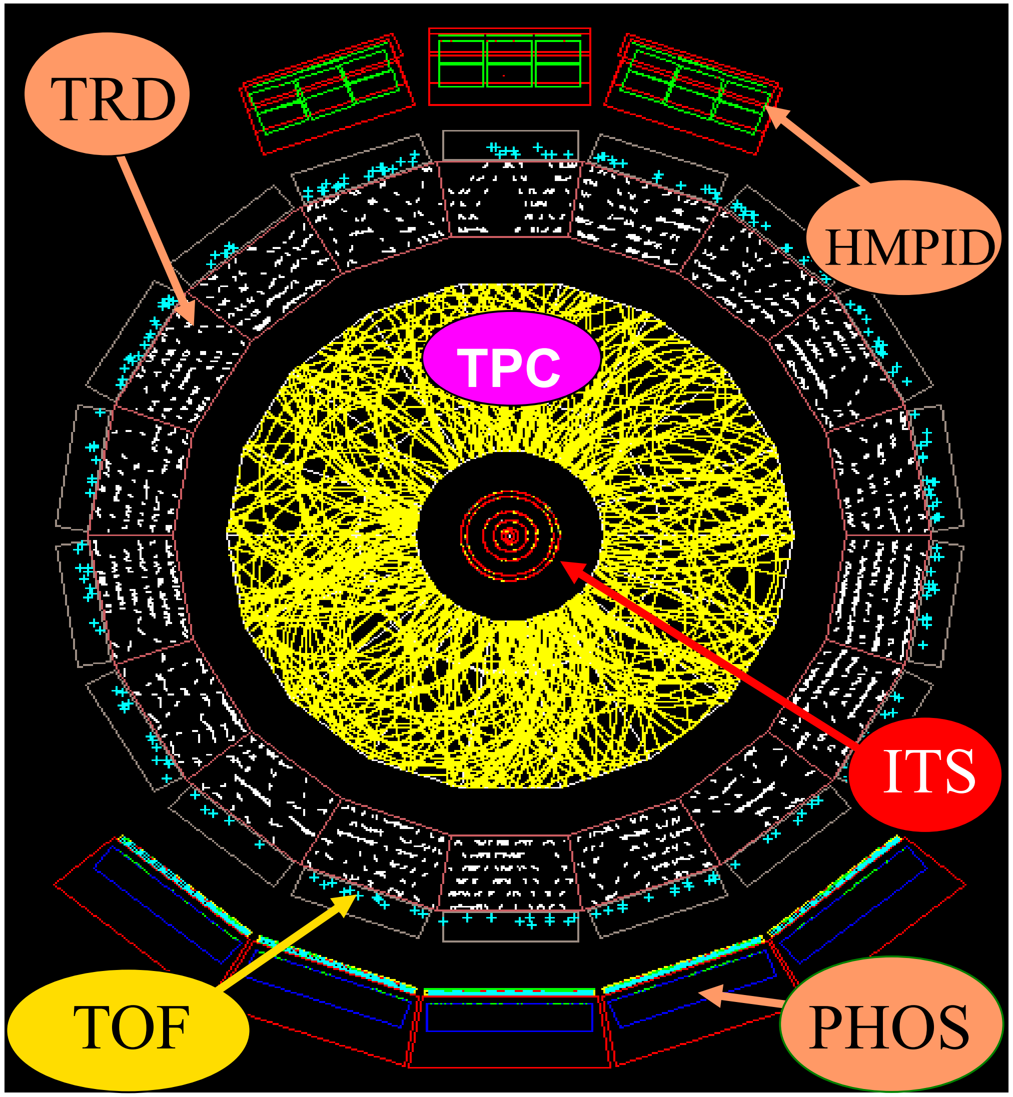


dN/dy ~ 8000

Particles per unit rapidity

ALICE simulation: Pb-Pb





2 degree slice ONLY!!

(~ 500 tracks)

(a bit old ...)