Statistical Methods in Particle Physics

4. Monte Carlo Methods

Prof. Dr. Klaus Reygers (lectures)
Dr. Sebastian Neubert (tutorials)

Heidelberg University
WS 2017/18

Monte Carlo Method

- Any method which solves a problem by generating suitable random numbers
- Useful for obtaining numerical solutions to problems which are too complicated to solve analytically
- The most common application of the Monte Carlo method is Monte Carlo integration
- Pioneers
  - Enrico Fermi
  - Stanislaw Ulam
  - John von Neumann
  - Nicholas Metropolis

https://en.wikipedia.org

Enrico Fermi

Stanislaw Ulam

J. von Neumann

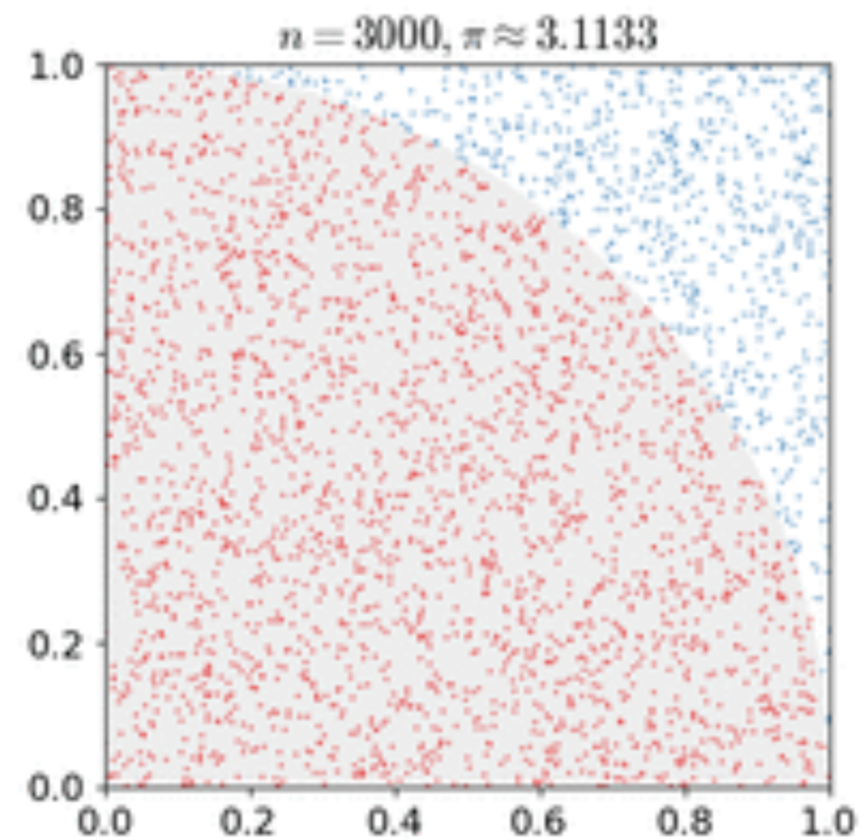
N. Metropolis

http://mathworld.wolfram.com/MonteCarloMethod.html

Monte Carlo Method: Examples

[from Bohm, Zech: Introduction to Statistics and Data Analysis for Physicists]

- Area of a circle
- Volume of the intersection of a cone and a torus
  - Hard to solve analytically
  - Easy to solve by scattering points homogeneously inside a cuboid containing the intersect
- Efficiency of particle detection with a scintillator
  - Produced photons are reflected at the surfaces and sometime absorbed
  - Almost impossible to calculate analytically for different parameters like incident angle, particle energy, …
  - Monte Carlo simulation is the only sensible approach

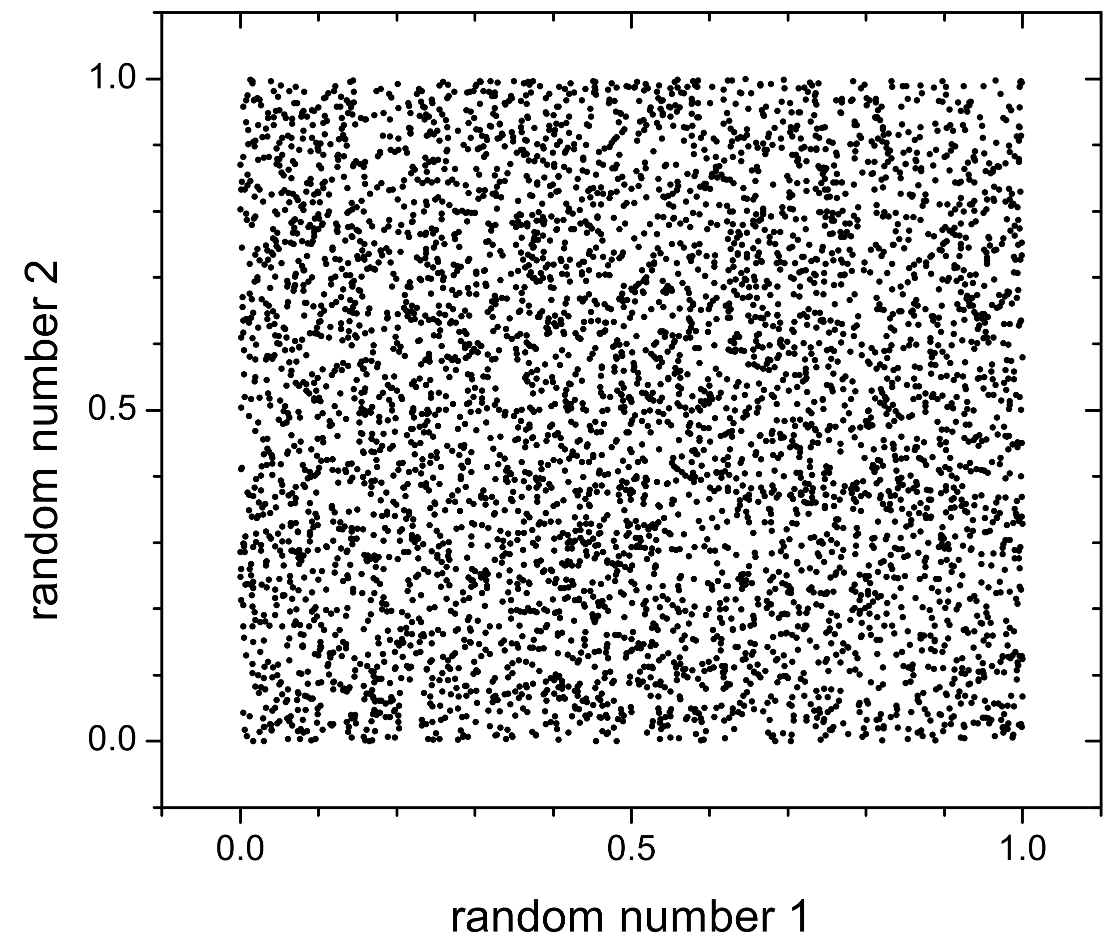


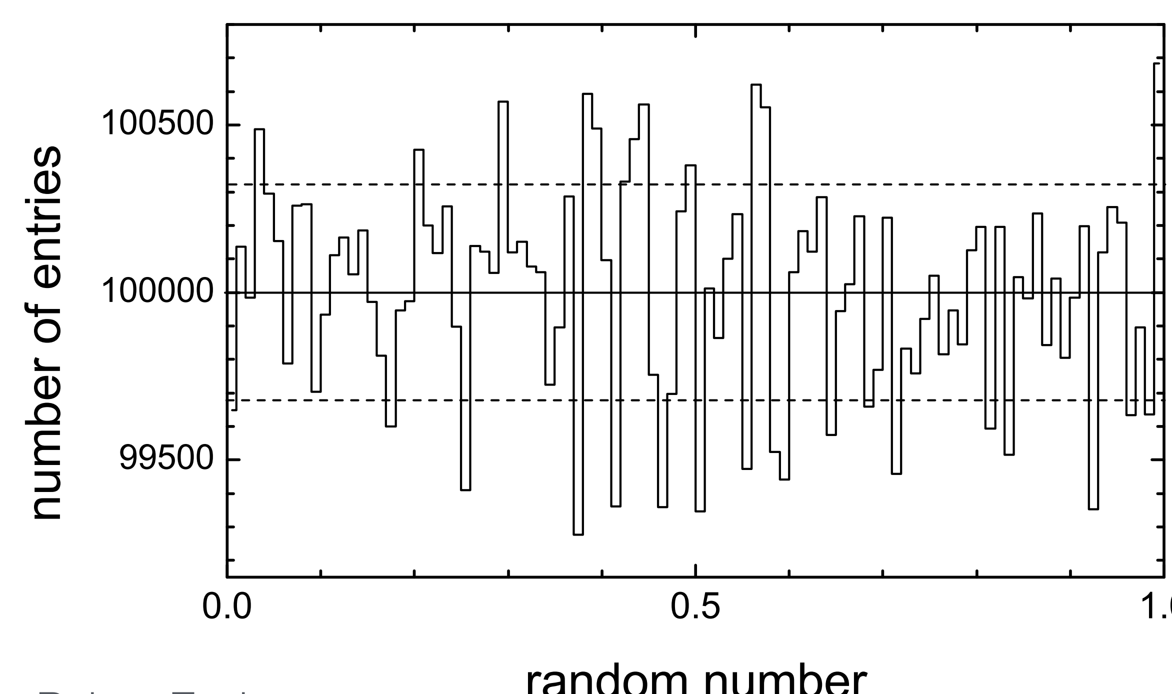
Pseudo-Random Numbers

- Generated uniformly in [0,1]
- Principle: Use insignificant digits of an operation to generate next number:

$$x_{i+1} = n^{-1}\mathrm{mod}(\lambda x_i; n)$$

- User can provide a *seed*
  - Same seed gives same sequence of random numbers
- Example: *Mersenne twister*
  - Invented 1997 by M. Matsomoto and T. Nishimura
  - Sequence repeats after 2^{19937} calls, i.e., never …
- Quality checks
  - Frequency of occurrence
  - Plot correlations between consecutive random numbers





Bohm, Zech:
http://www-library.desy.de/preparch/books/vstatmp_engl.pdf

Random Numbers from Distributions: Inverse Transform Method

Consider a distribution f from which we want to draw random numbers. Let $u(r)$ be the uniform distribution in [0, 1]:

$$\int_{-\infty}^{x} f(x')\,\mathrm{d}x' = \int_{0}^{r(x)} u(r')\,\mathrm{d}r' = r(x)$$

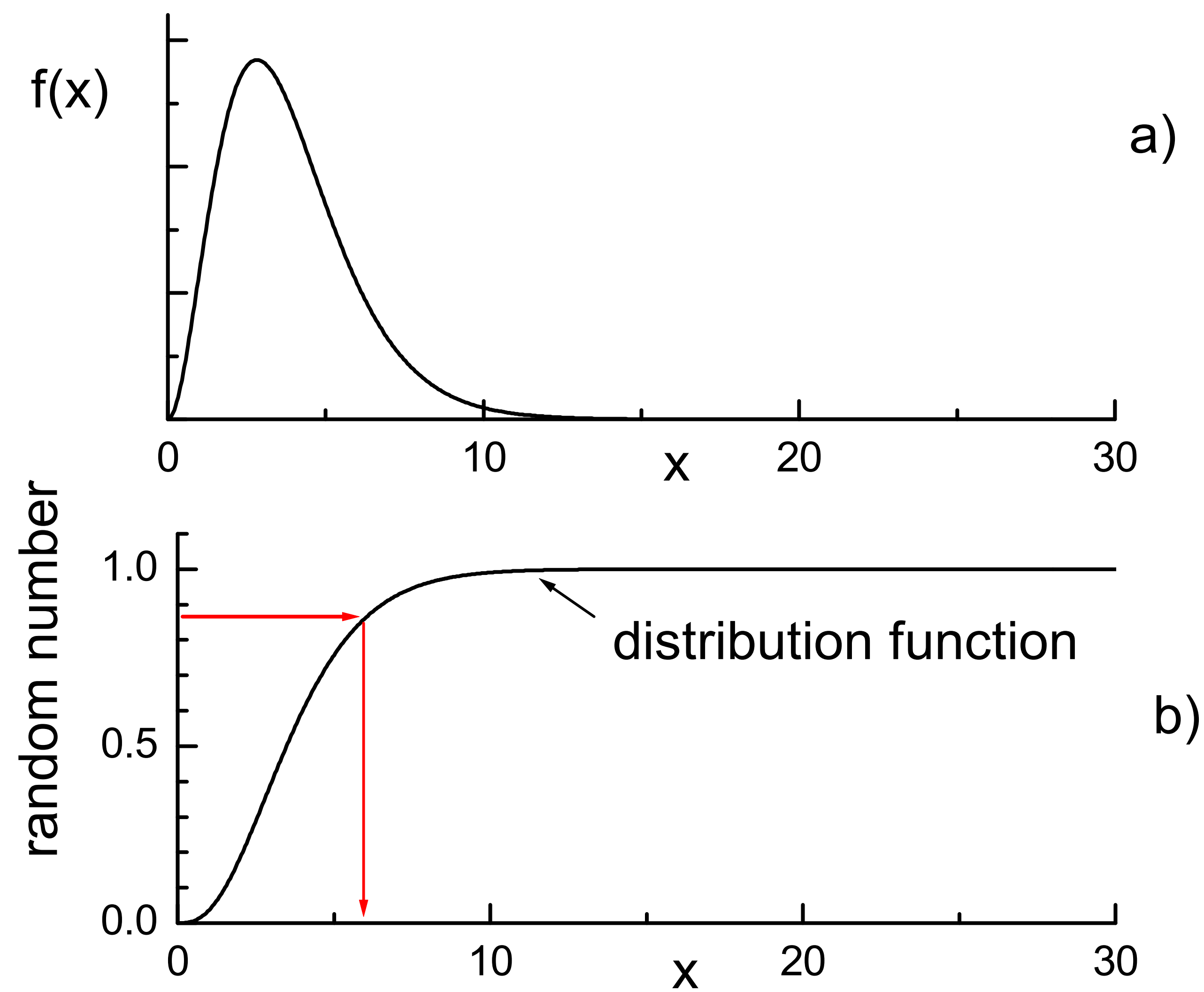
With $F(x)$ = cumulative distr.:

$$F(x) = r$$

We get the random number x from the inverse of the cumulative distribution:

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

Bohm, Zech:
http://www-library.desy.de/preparch/books/vstatmp_engl.pdf



Cross check:

$$\frac{\mathrm{d}p}{\mathrm{d}x} = \underbrace{\frac{\mathrm{d}p}{\mathrm{d}r}}_{=1}\frac{\mathrm{d}r}{\mathrm{d}x} = \frac{\mathrm{d}F(x)}{\mathrm{d}x} = f(x) \qquad \square$$

Examples I

Linear function:

$$f(x) = 2x, \qquad 0 \le x \le 1$$

$$F(x) = x^2 \quad \rightarrow \quad x = \sqrt{r}$$

Exponential:

$$f(x) = \gamma e^{-\gamma x}, \qquad x \ge 0$$

$$F(x) = 1 - e^{-\gamma x} \quad \rightarrow \quad x = -\frac{\ln(1-r)}{\gamma}$$

One can store *F*(x) as a histogram if there is no analytical solution, cf. root's **`GetRandom()`** function:

```
root [0] TF1 f("f", "x^3/(exp(x)-1)", 0., 15.);
root [1] cout << f.GetRandom() << endl;
13.9571
```

Example II: Uniform Points on a Sphere

$$\frac{\mathrm{d}p}{\mathrm{d}\Omega} = \frac{\mathrm{d}p}{\sin\theta \, \mathrm{d}\theta \, \mathrm{d}\phi} = \mathrm{const} \equiv k \qquad\qquad \frac{\mathrm{d}p}{\mathrm{d}\theta \, \mathrm{d}\phi} = k \sin\theta \equiv f(\phi) g(\theta)$$

Distributions for θ and ϕ:

$$f(\phi) \equiv \frac{\mathrm{d}p}{\mathrm{d}\phi} = \mathrm{const} = \frac{1}{2\pi}, \qquad 0 \leq \phi \leq 2\pi$$

$$g(\theta) \equiv \frac{\mathrm{d}p}{\mathrm{d}\theta} = \frac{1}{2} \sin\theta, \qquad 0 \leq \theta \leq \pi$$

Calculating the inverse of the cumulative distribution we obtain:

$$\phi = 2\pi r_1$$

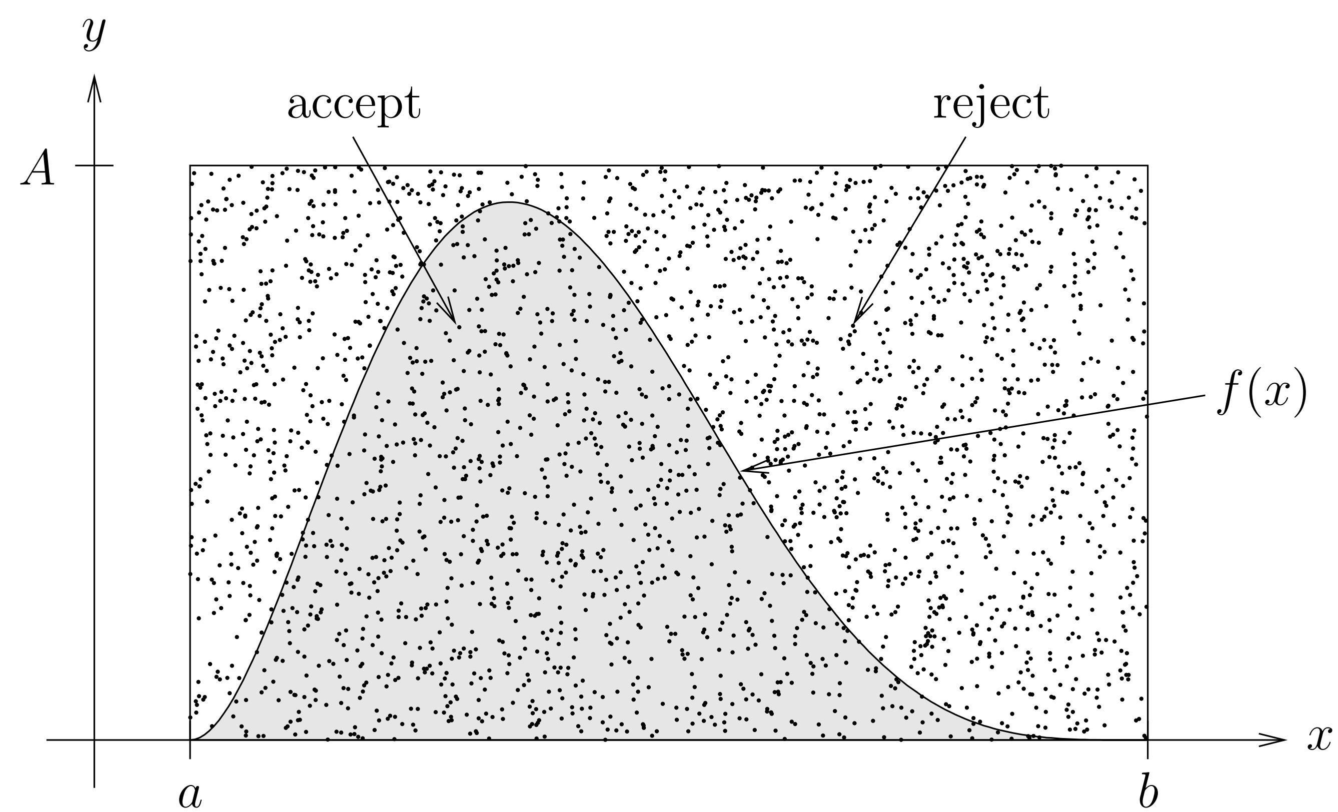
$$\theta = \arccos(1 - 2r_2) \qquad [\text{as } G(\theta) = \frac{1}{2}(1 - \cos\theta)]$$

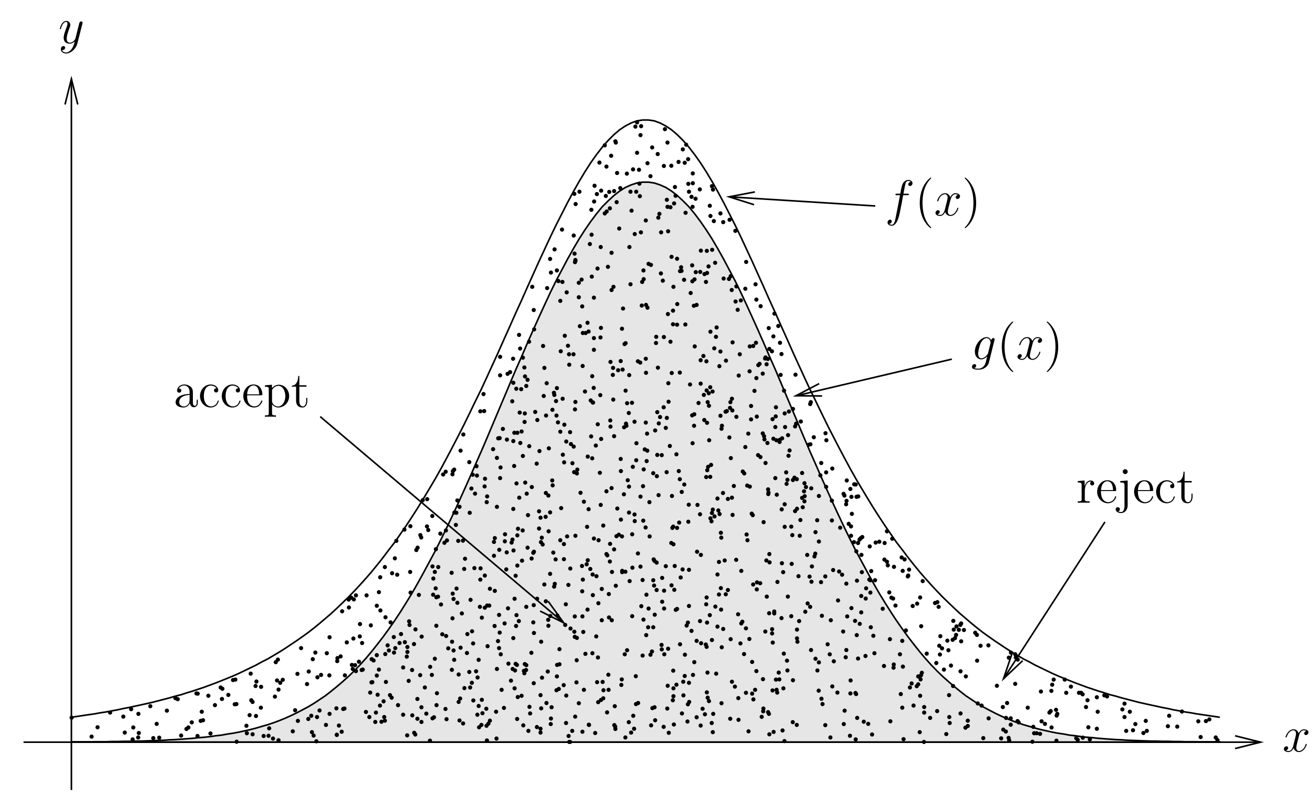
Upshot: ϕ and cos θ need to be distributed uniformly

Monte Carlo Integration: Acceptance-Rejection Method

Lecture M. Neumann, www.exp.univie.ac.at/cp1/cp6.pdf

- Algorithm
  - Generate random number *x* uniformly between *a* and *b*
  - Generate second random *y* number uniformly between 0 and *A*
  - Accept *x* if *y* < f(*x*)
  - Repeat many times
- The efficiency of this algorithm can be quite small
- Improvement possible by choosing a majorant, i.e., a function which encloses *f*(x) and whose integral is known ("importance sampling")





Monte Carlo Integration

Naïve Monte Carlo integration:

x_i: uniformly distributed random numbers

uniform distribution in $[a, b]$

$$\underbrace{\int_a^b f(x)\,\mathrm{d}x}_{=:\,I} = (b-a)\int_a^b f(x)u(x)\,\mathrm{d}x = (b-a)\langle f(x)\rangle \approx \underbrace{(b-a)\cdot\frac{1}{n}\sum_{i=1}^{n} f(x_i)}_{=:\,\hat{I}}$$

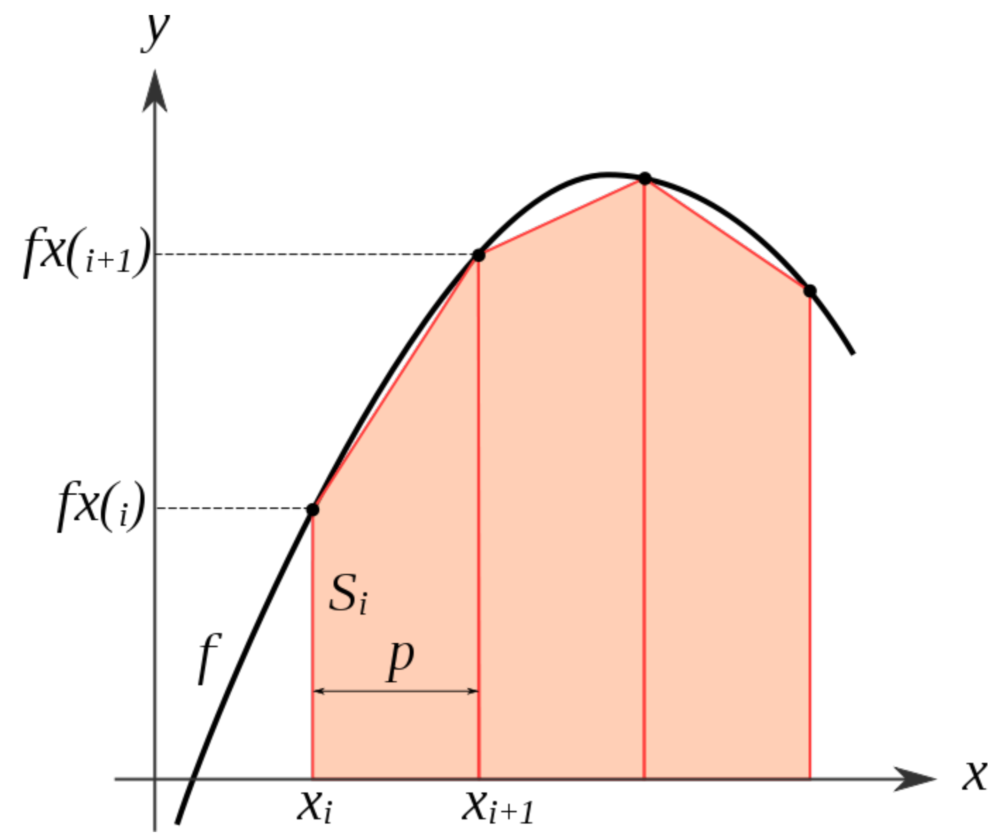
Typical deviation from the true value of the integral (standard deviation)

$$V[\hat{I}] = \frac{(b-a)^2}{n^2} V\Big[\sum_{i=1}^{n} f(x_i)\Big] = \frac{(b-a)^2}{n^2}\cdot n\cdot\sigma^2[f] \quad\rightarrow\quad \sigma[\hat{I}] = \frac{b-a}{\sqrt{n}}\sigma[f]$$

Monte Carlo Integration: Multidimensional Integrals

Trapezoidal rule in one dimension

- accuracy improves as $1/n^2$ with the number of points
- Much better than $1/\sqrt{n}$ scaling of the MC methods



https://en.wikipedia.org/wiki/Trapezoidal_rule

Monte Carlo integration in d dimensions:

$$I = \int_\Omega f(\vec{x})\,\mathrm{d}\vec{x}, \quad \Omega \subset \mathbb{R}^d, \qquad V = \int_\Omega \mathrm{d}\vec{x}$$

$$I \approx \hat{I} = V \frac{1}{n} \sum_{i=1}^{n} f(\vec{x}_i), \qquad \sigma[\hat{I}] \approx V \frac{\sigma[f]}{\sqrt{n}}$$

same as in 1d case

Trapezoidal rule in d dimension:

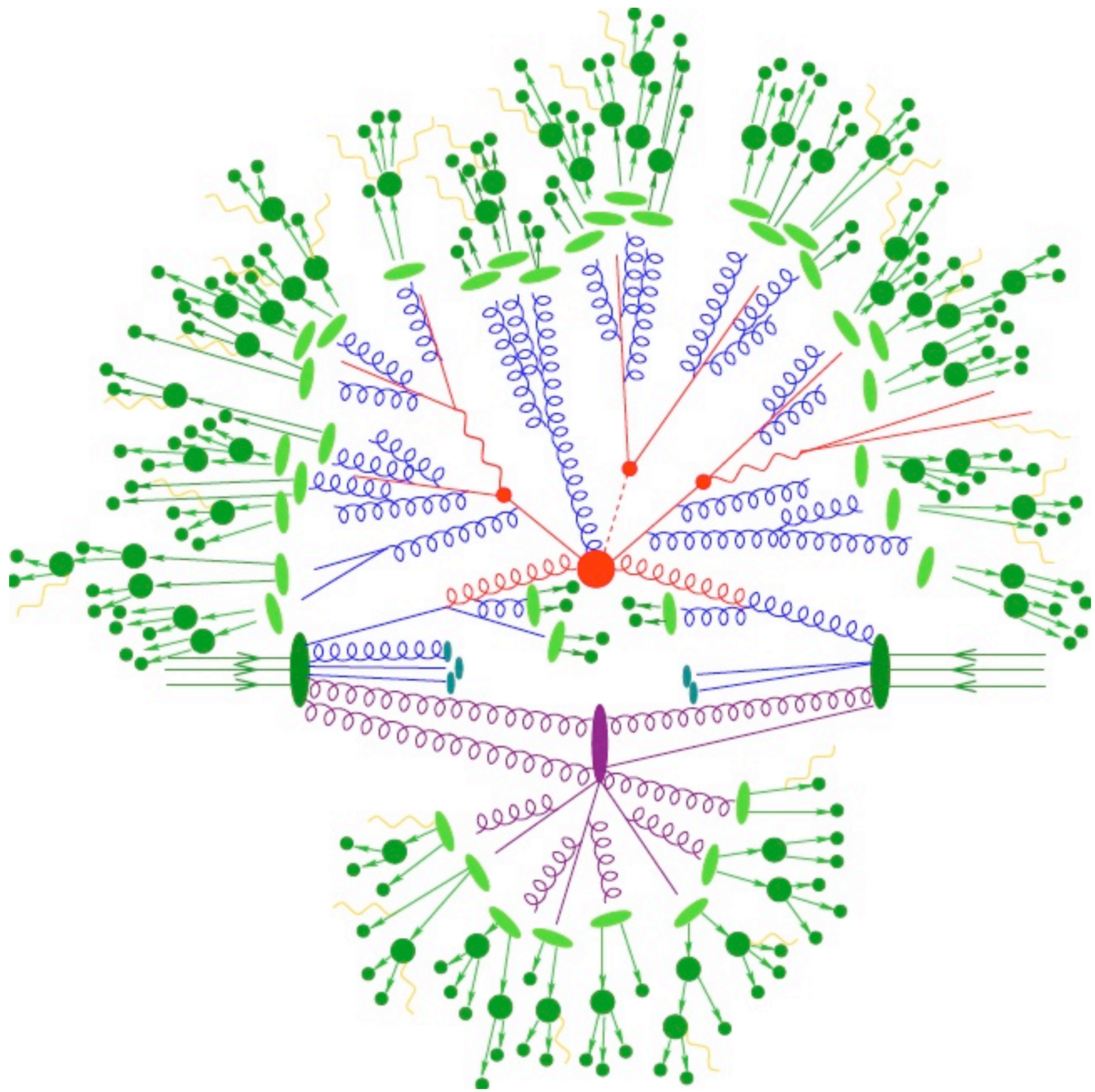
- accuracy improves as $1/n^{2/d}$ with the number of points
- for $d > 4$ the dependence on n is better for MC integration

For multidimensional integrals MC integration outperforms other numerical integration methods

Monte Carlo Simulation I: Event Generators (Pythia, Sherpa, …)

Examples: Pythia

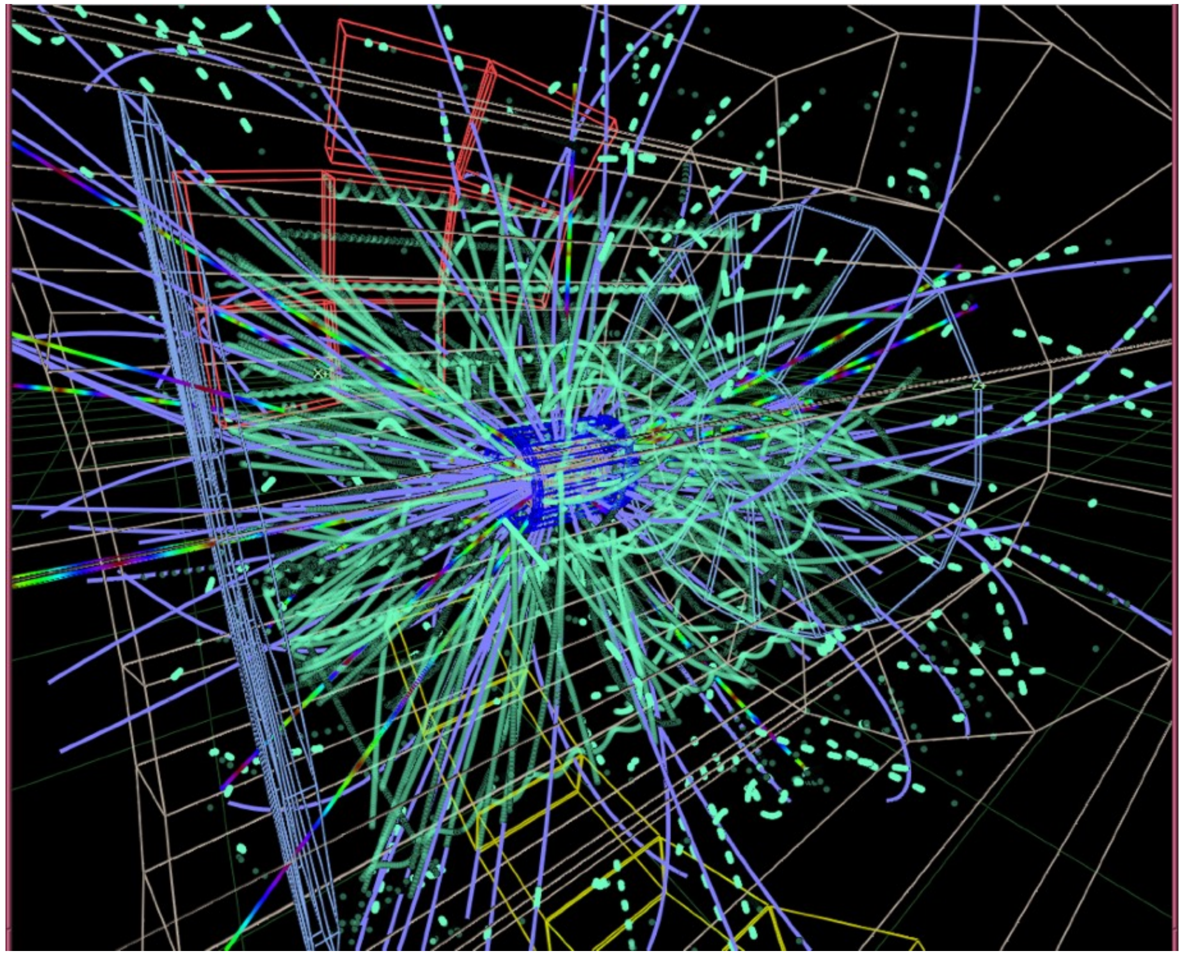
- Simulation of pp and e^+e^- collision on quark and gluon level
- Hard and soft interactions, parton showers, fragmentation and particle decay
- Many applications
  - Test underlying physics, e.g., perturbative QCD
  - Calculate QCD background processes, e.g., in Higgs searches
  - Calculation of detector efficiencies



Pythia

Output:

Four-vectors of of produced particles



Event listing (summary)

I	particle/jet		KS	KF	orig	p_x	p_y	p_z	E	m
1	(u)	A	12	2	0	0.000	0.000	10.000	10.000	0.006
2	(ubar)	V	11	-2	0	0.000	0.000	-10.000	10.000	0.006
3	(string)		11	92	1	0.000	0.000	0.000	20.000	20.000
4	(rho+)		11	213	3	0.098	-0.154	2.710	2.856	0.885
5	(rho-)		11	-213	3	-0.227	0.145	6.538	6.590	0.781
6	pi+		1	211	3	0.125	-0.266	0.097	0.339	0.140
7	(Sigma0)		11	3212	3	-0.254	0.034	-1.397	1.855	1.193
8	(K*+)		11	323	3	-0.124	0.709	-2.753	2.968	0.846
9	p~-		1	-2212	3	0.395	-0.614	-3.806	3.988	0.938
10	pi-		1	-211	3	-0.013	0.146	-1.389	1.403	0.140
11	pi+		1	211	4	0.109	-0.456	2.164	2.218	0.140

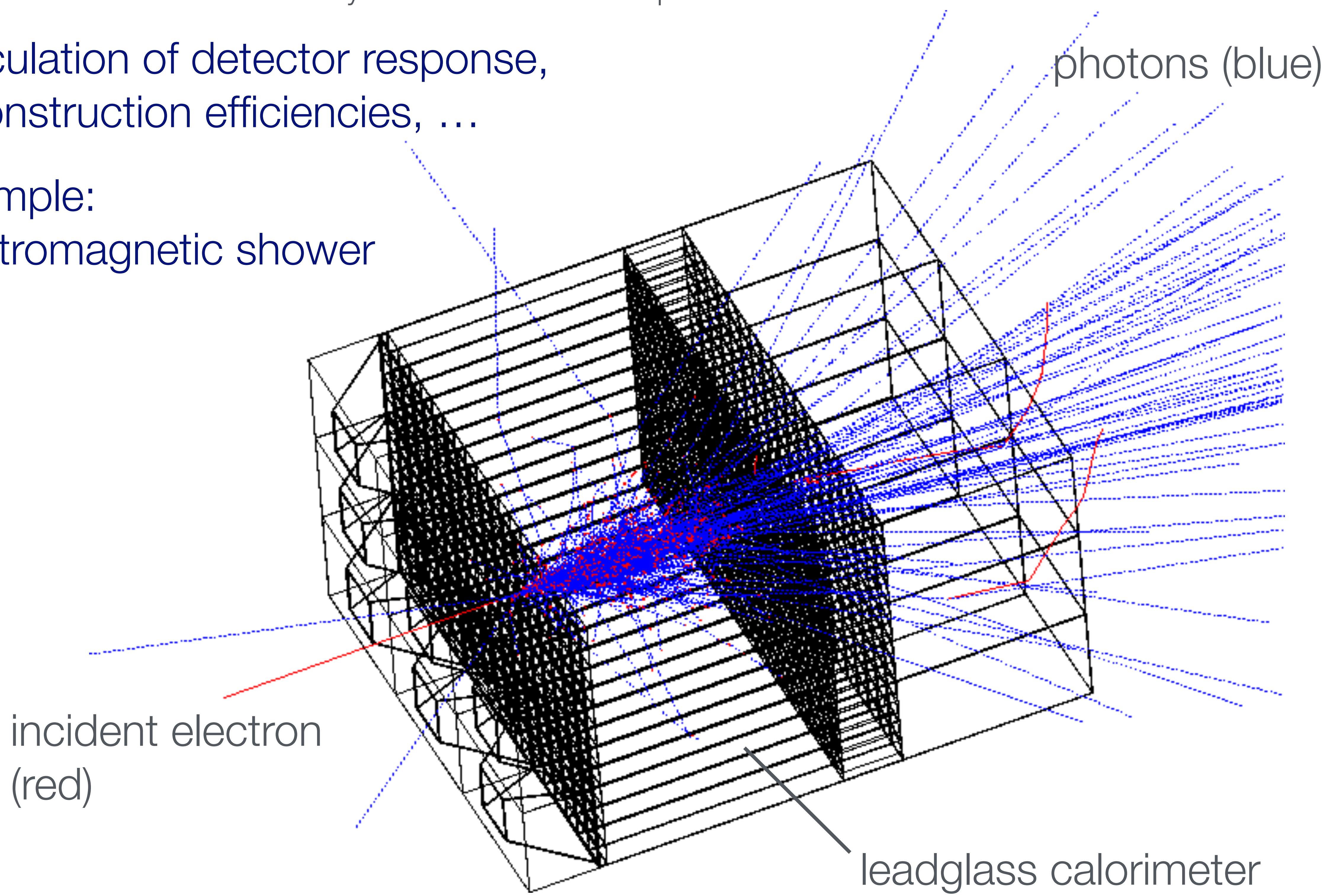
Monte Carlo Simulation II: Detector Simulation with GEANT

http://geant4.cern.ch/

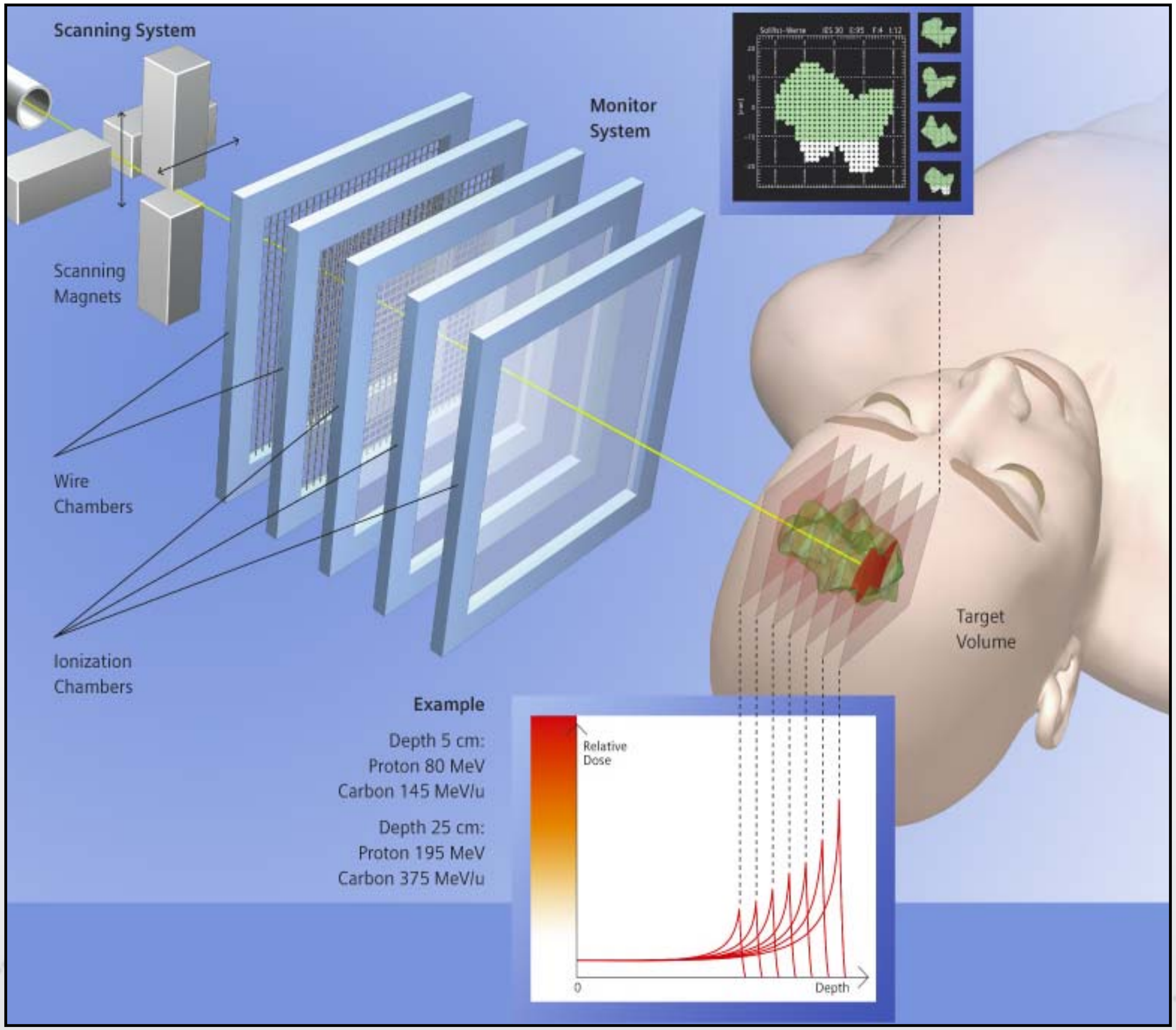
http://www.uni-muenster.de/Physik.KP/santo/thesis/diplom/kees

Calculation of detector response, reconstruction efficiencies, …

Example:
electromagnetic shower

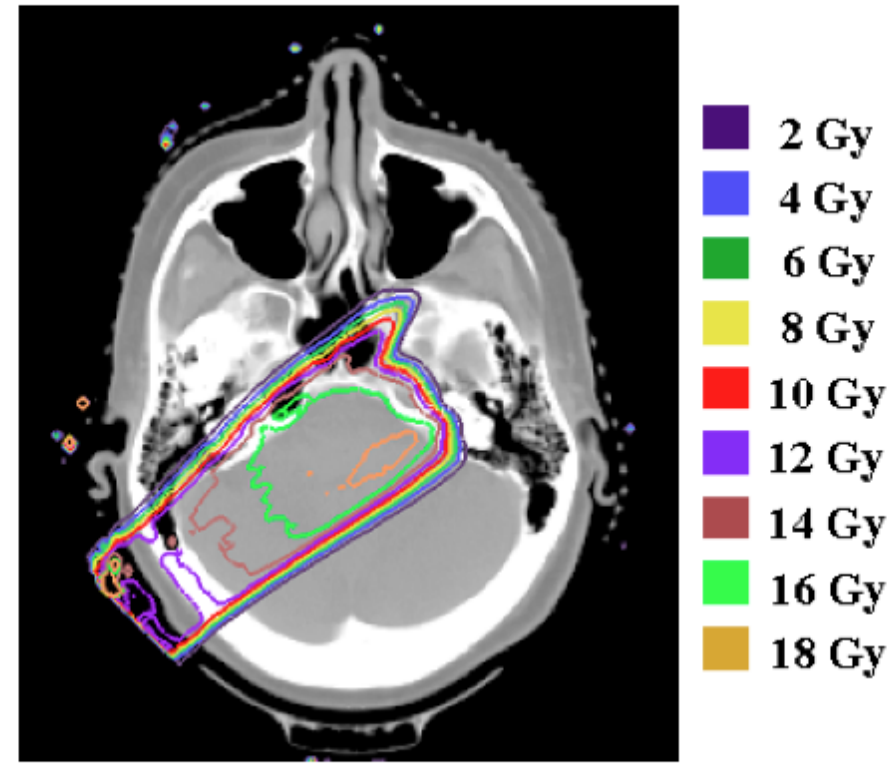


Monte Carlo Simulation III: Treatment Planning in Radiation Therapy



Intensity-Controlled Rasterscan Technique, Haberer et al., GSI, NIM A,1993

Source: GSI



Codes

- GEANT 4
- FLUKA
- …