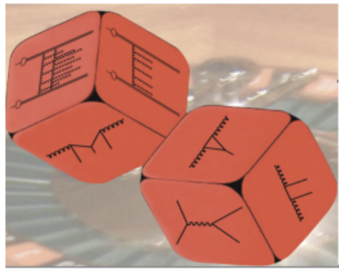
# Summer School 2020

Terascale Summer School: Tutorial/Exercises - QCD and Monte Carlo techniques



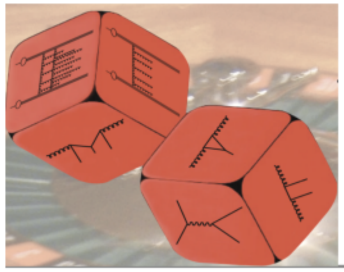
# Summer School 2020

Terascale Summer School: Tutorial/Exercises - QCD and Monte Carlo techniques

- Welcome to the tutorial exercises on Monte Carlo techniques and QCD

- Goal:
  - leave no one behind
  - everyone should be able to do the exercises
    - please ask questions

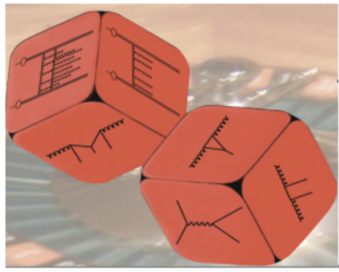
- additional infos under:
  - https://www.desy.de/~jung/Terascale_SummerSchool_2020_Tutorial



# Summer School 2020

Terascale Summer School: Tutorial/Exercises - QCD and Monte Carlo techniques

- Structure:
  - introductory lecture (20-30 min)
  - 5-6 working groups (in breakout-rooms) with tutors:
    - Sara Taheri Monfared
    - Qun Wang
    - Armando Bermudez Martinez
    - Luis Ignacio Estevez Banos
    - Mikel Mendizabal
    - Patrick Connor
  - ask questions
  - share screen to get help

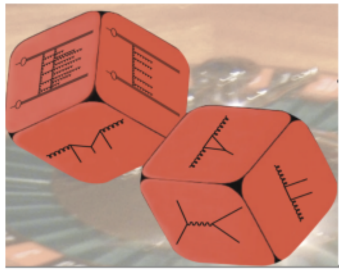




# Environment

- Virtual Machine
  - running full C++ code
  - compilation, execute program

- New:
  - running python on Jupyter Notebook
  - only need web browser
  - Login with School account on Computer-Farm at DESY
    - account will be valid for 2 weeks
    - Accounts on Google Doc
      - please pick an account and mark it (and remember your account and passwd)





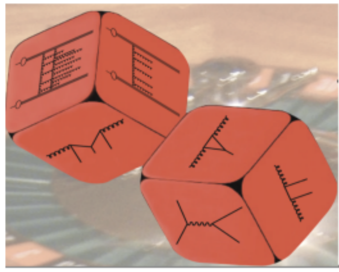
# How to get started

- start VM
  - in case – passwd: terascale1234
  - perhaps change screen resolution

- compiling and running:

```
cd exercise-1
make example-1
./example-1
```

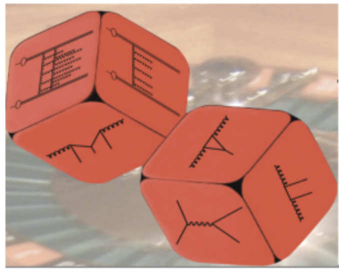
- templates are provided which include the general structure – you only have to fill the interesting – important parts



# How to get started: Jupyter notebook

Terascale Summer School: Tutorial/Exercises - QCD and Monte Carlo techniques

- open browser, and login to: https://naf-jhub.desy.de/
  - we have school accounts, valid for 2 weeks
    - Accounts on Google Doc
      - please pick an account and mark it (and remember your account and passwd)
  - (see https://confluence.desy.de/display/IS/Jupyter+on+NAF)



# How to get started: jupyter notebook



Deutsches Elektronen-Synchrotron DESY
A Research Centre of the Helmholtz Association

**Log in with DESY Account**

**Username:** jung

**Password:** ••••••••

Sign In

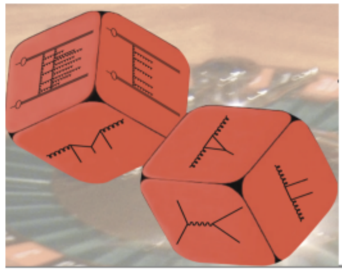
## Welcome to the JupyterHub for NAF Users

In order to login into the JupyterHub you must have your DESY credentials prepared for NAF access. Please follow the documentation of your experiment/group to gain full access to the NAF.

You may also be interested in our other services, like the DESY supercomputer Maxwell.

## News

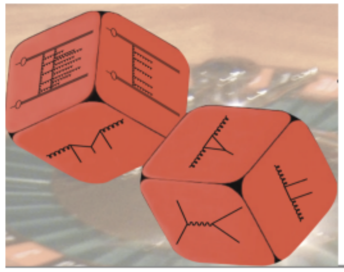
- JupyterLab was updated to version 2.0.1, some extensions might need updating, March 25th
- Update of Jupyterhub, the hub now does not spawn on the last seen machine.
  This should lead to fewer failed spawn attempts. March 10th
- Jupyterhub was updated to version 1.1.0 and JupyterLab to version 1.2.6
  Users can now use their own extensions, see the *Confluence* page for details
- There will be a maintenance of the JupyterHub between 9am and 10am.
  Service might be interrupted. *edit: maintenance finished* January 7th, 2020



# How to get started: jupyter notebook

Terascale Summer School: Tutorial/Exercises - QCD and Monte Carlo techniques

Open a terminal to copy the source



# How to get started: jupyter notebook

Terascale Summer School: Tutorial/Exercises - QCD and Monte Carlo techniques

## Copy the source

Here is the link: https://www.desy.de/~jung/Terascale_SummerSchool_2020_Tutorial/Terascale_SummerSchool_2020_Tutorial_exericises_files/jupyter-python-all.tgz

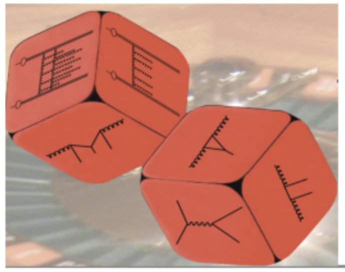
Logout | Control Panel

```
sh-4.2$ wget https://www.desy.de/~jung/Terascale_SummerSchool_2020_Tutorial/Terascale_SummerSchool_20
20_Tutorial_exericises_files/jupyter-python-all.tgz
--2020-08-17 09:40:09--  https://www.desy.de/~jung/Terascale_SummerSchool_2020_Tutorial/Terascale_Sum
merSchool_2020_Tutorial_exericises_files/jupyter-python-all.tgz
Resolving www.desy.de (www.desy.de)... 131.169.180.47
Connecting to www.desy.de (www.desy.de)|131.169.180.47|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 410755 (401K) [application/x-tar]
Saving to: 'jupyter-python-all.tgz'

100%[============================================================>] 410,755     --.-K/s   in 0.006s

2020-08-17 09:40:09 (67.4 MB/s) - 'jupyter-python-all.tgz' saved [410755/410755]

sh-4.2$ tar xvfz jupyter-python-all.tgz
jupyter-python-solutions/README-hannes
jupyter-python-solutions/example-1.ipynb
jupyter-python-solutions/example-1.py
jupyter-python-solutions/example-2.ipynb
jupyter-python-solutions/example-2.py
jupyter-python-solutions/example-3.ipynb
jupyter-python-solutions/example-3.py
jupyter-python-solutions/example-4.ipynb
jupyter-python-solutions/example-4.py
jupyter-python-solutions/example-5.ipynb
jupyter-python-solutions/example-5.py
jupyter-python-solutions/example-6.ipynb
jupyter-python-solutions/example-6.py
jupyter-python-solutions/example-7.ipynb
jupyter-python-solutions/example-7.py
jupyter-python-solutions/example-8.ipynb
jupyter-python-solutions/example-8.py
jupyter-python-solutions/example-9.ipynb
jupyter-python-solutions/example-9.py
jupyter-python-solutions/example-lhapdf.ipynb
```



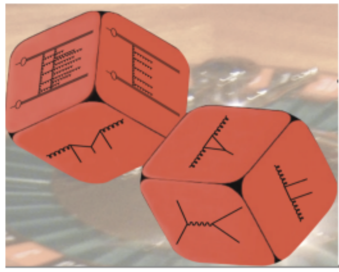
# How to get started: jupyter notebook

Terascale Summer School: Tutorial/Exercises - QCD and Monte Carlo techniques

Logout | Control Pane

```
jupyter-python/example-5.py
jupyter-python/example-5.py.bck
jupyter-python/example-6.ipynb
jupyter-python/example-6.py
jupyter-python/example-6.py.bck
jupyter-python/example-7.ipynb
jupyter-python/example-7.py
jupyter-python/example-7.py.bck
jupyter-python/example-8.ipynb
jupyter-python/example-8.py
jupyter-python/example-8.py.bck
jupyter-python/example-9.ipynb
jupyter-python/example-9.py
jupyter-python/example-9.py.bck
jupyter-python/example-lhapdf.ipynb
jupyter-python/example-lhapdf.py
jupyter-python/example-lhapdf.py.bck
sh-4.2$
sh-4.2$
sh-4.2$ cd jupyter-python
/afs/desy.de/user/j/jung/jupyter-python
sh-4.2$ pws
sh: pws: command not found
sh-4.2$ pwd
/afs/desy.de/user/j/jung/jupyter-python
sh-4.2$ ls
example-1.ipynb    example-3.py       example-5.py.bck   example-8.ipynb        example-lhapdf.py
example-1.py       example-3.py.bck   example-6.ipynb    example-8.py           example-lhapdf.py.bck
example-1.py.bck   example-4.ipynb    example-6.py       example-8.py.bck       README-hannes
example-2.ipynb    example-4.py       example-6.py.bck   example-9.ipynb
example-2.py       example-4.py.bck   example-7.ipynb    example-9.py
example-2.py.bck   example-5.ipynb    example-7.py       example-9.py.bck
example-3.ipynb    example-5.py       example-7.py.bck   example-lhapdf.ipynb
sh-4.2$
sh-4.2$
```

Good luck !



# Exercise 1 - Introduction

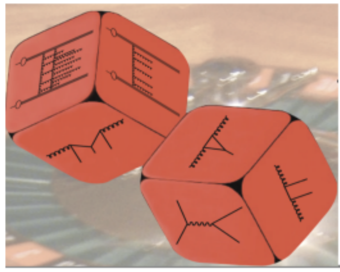


- Schedule:
  - Thursday - Exercise 1:
    - Random numbers
    - MC method
    - MC integration
  - Monday – continue Exercise 1:
    - Random numbers
    - MC method
    - MC integration

- Tuesday - Exercise 2:
  - Sudakov form factor
  - MC solution of evolution equation

- Wednesday-Thursday - Exercise 3
  - Calculation & simulation of Higgs production
  - Using MC solution of evolution equation → calculation of pt spectrum of Higgs at LHC

We can continue, if needed/wanted !



# Example1



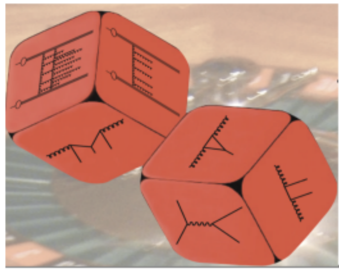
- construct a uniform random number generator from the congruential method:
- Writeup:
  - Page 9, eq (2.1)

2.1. RANDOM NUMBERS 9

A simple random number generator (so called *multiplicative congruential linear random number generator*) can be build as follows [4][p 40ff] and [5][Vol II,p 9]. From an initial number $I_0$ we generate a series of random numbers $R_j$ according to:

$$I_j = mod(aI_{j-1} + c, m)$$
$$R_j = \frac{I_j}{m} \quad (2.1)$$

with $a$ being an multiplicative and $c$ a additive constant and $m$ the modulus[3]. With this procedure one obtains a series of number $R_j$ in the interval $(0, 1)$ (note that the values 0 and 1 are excluded). This random number generator will be tested in the exercise. In fig 2.3 the correlation of 2 random numbers is shown on the left side. The right side shows the same



# Congruential linear generator



- develop our own simple generator

$$I_j = mod(aI_{j-1} + c, m)$$
$$R_j = \frac{I_j}{m}$$

- with $mod(i_1, i_2) = i_1 - INT(i_1/i_2)i_2$

what is $mod$ ?
The modulus operator - or more precisely, the modulo operation - is a way to determine the remainder of a division operation.
Examples:
5 % 1 = 0
// 5 divided by 1 equals 5, with a remainder of 0
5 % 2 = 1
// 5 divided by 2 equals 2, with a remainder of 1
5 % 3 = 2
// 5 divided by 3 equals 1, with a remainder of 2

seed $I_0$, multiplicative constant $a$ and additive constant $c$ modulus $m$

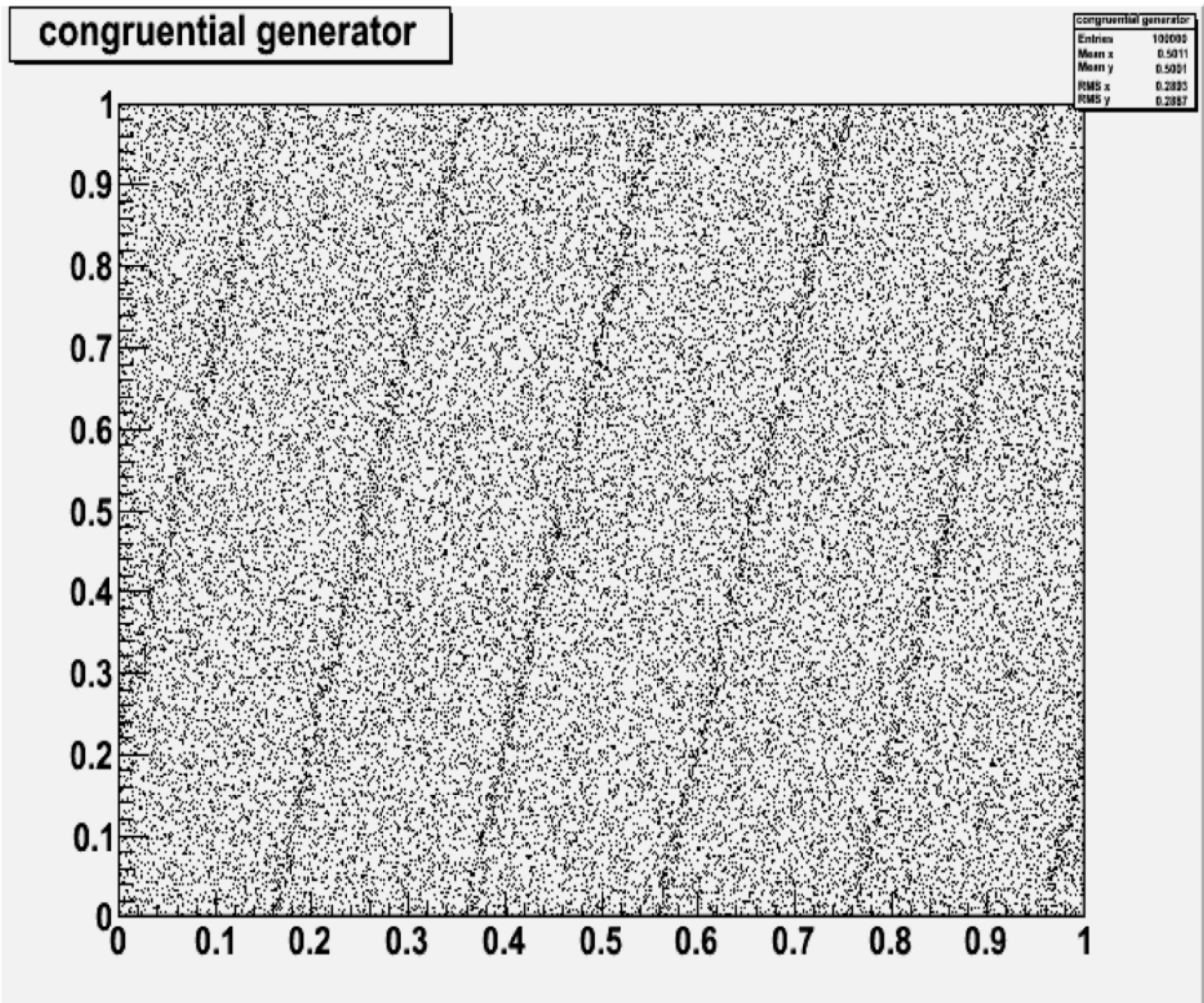
➔ maximal repetition period: $\mathcal{O}(m)$

➔ example:

$$I_0 = 4711$$
$$a = 205$$
$$c = 29573$$
$$m = 139968$$

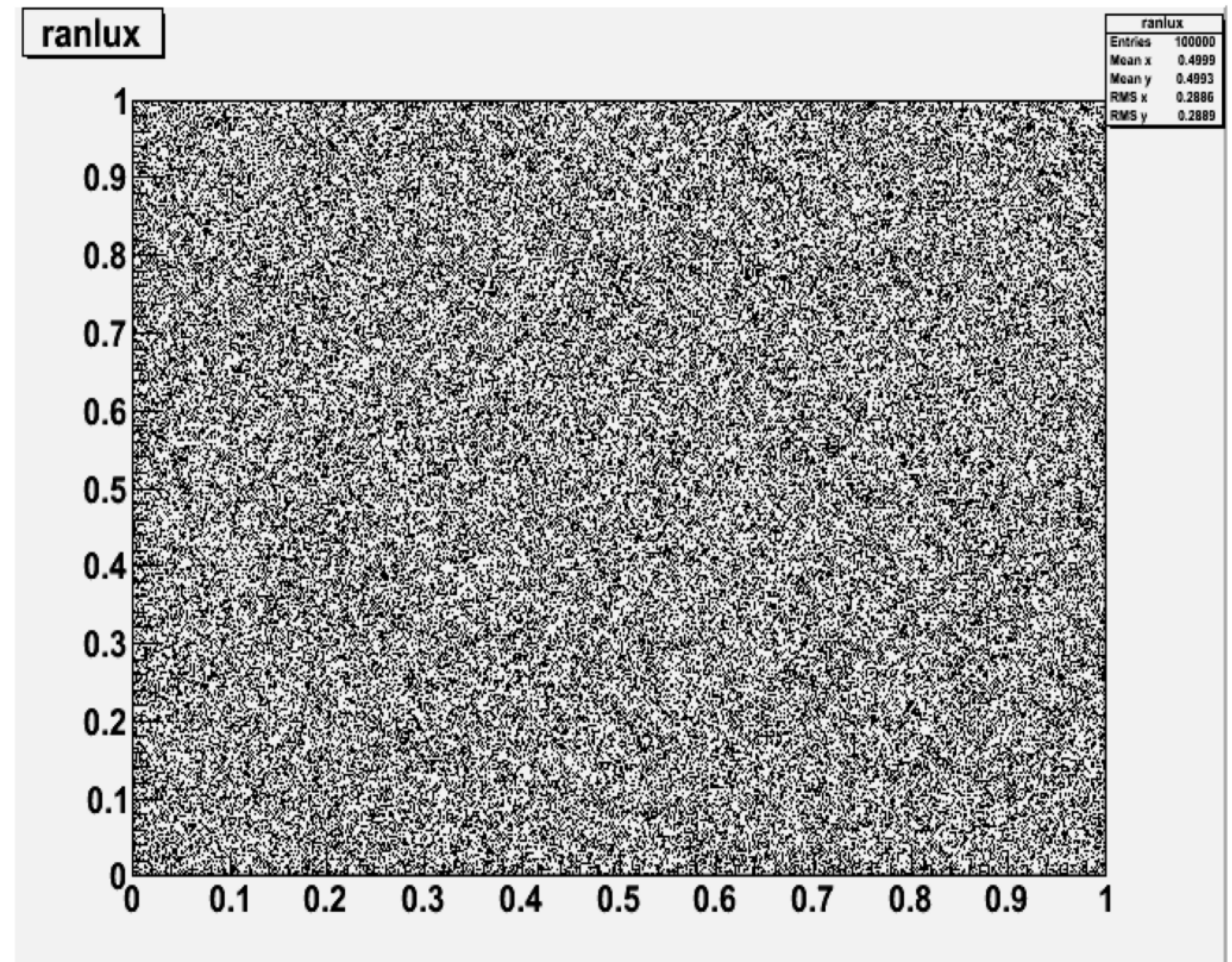
# Randomness tests

- Congruential generator

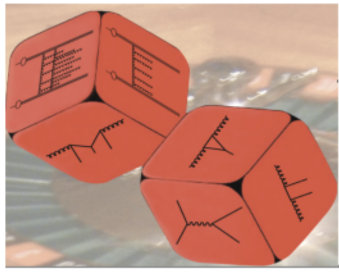



- RANLUX

M. Lüscher, A portable high-quality random number generator for lattice field theory simulations, Computer Physics Communications 79 (1994) 100
http://luscher.web.cern.ch/luscher/ranlux/index.html




➔ RANLUX much more sophisticated. Developed and used for QCD lattice calcs



# Example2

Terascale Summer School: Tutorial/Exercises - QCD and Monte Carlo techniques

- construct a Gaussian random number generator from a uniform random number generator
- Writeup:
  - Page 17, equations after eq(2.25)

- take a sum of uniformly distributed random numbers $R_i$:

$$R_n = \sum_{i=1}^{n} R_i$$

The expectation value and the variance are calculated according to the rules in eq.(2.8,2.11):

$$\begin{aligned} E[R_1] &= \int u du = \frac{1}{2} \\ V[R_1] &= \int \left(u - \frac{1}{2}\right)^2 du = \frac{1}{12} \\ E[R_n] &= \frac{n}{2} \\ V[R_n] &= \frac{n}{12} \end{aligned}$$

18 CHAPTER 2. MONTE CARLO METHODS

According to the Central Limit Theorem the sum of random values is Gauss distributed. To obtain a distribution centered around 0 with $\sigma = 1$ we take:

$$\frac{\sum_i x_i - \sum_i \mu_i}{\sqrt{\sum_i \sigma_i^2}} \to \mathcal{N}(0,1)$$

For example we sum $n = 12$ random numbers (many times $N \to \infty$) and we obtain a "normal" (Gauss) distribution $\mathcal{N}$ [11]:

$$\mathcal{N}(0,1) \to \frac{R_n - n/2}{\sqrt{n/12}} = R_{12} - 6$$

# Central Limit Theorem

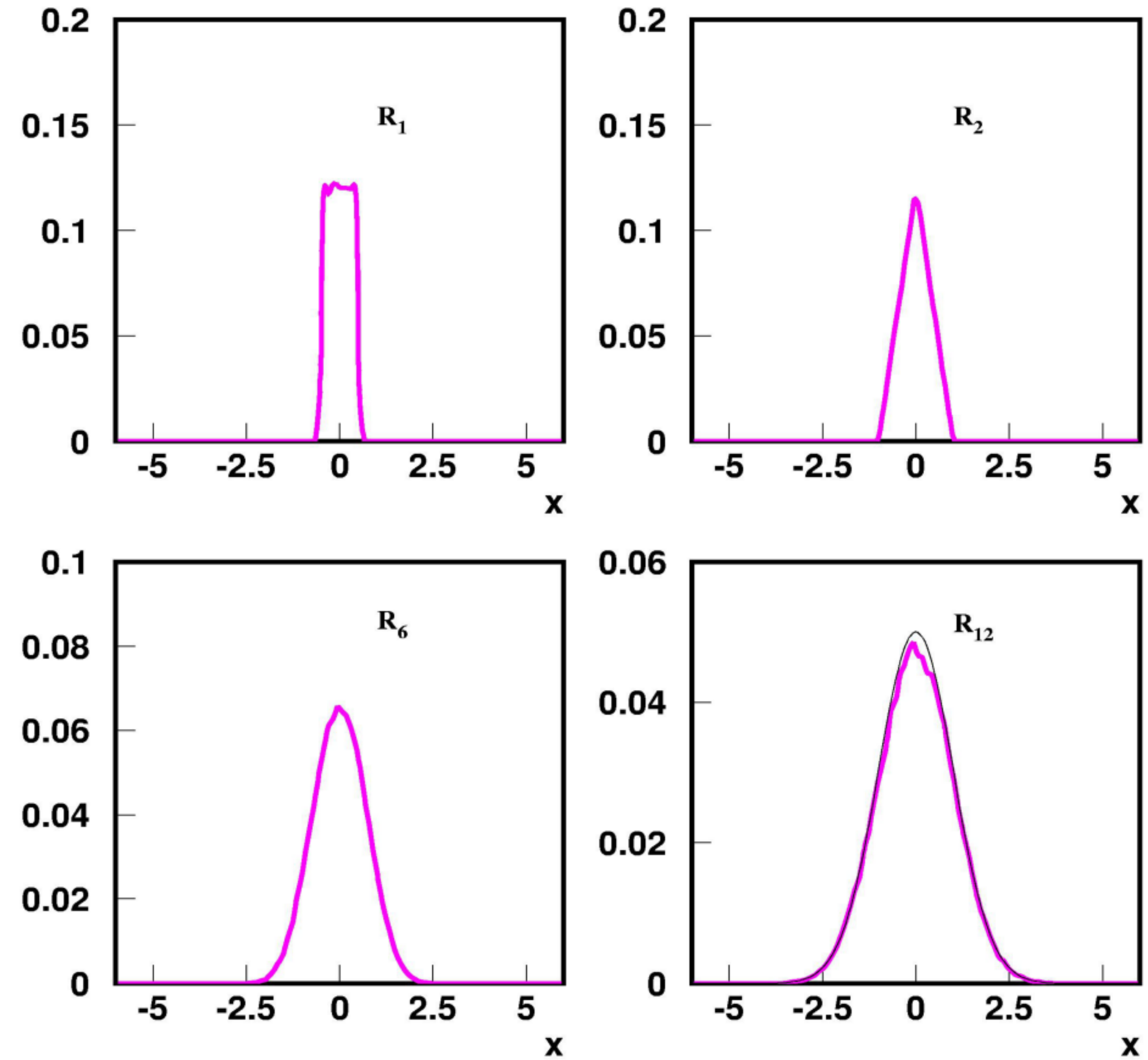
- Central Limit Theorem
  for large N the sum of independent random variables is always normally (Gaussian) distributed:

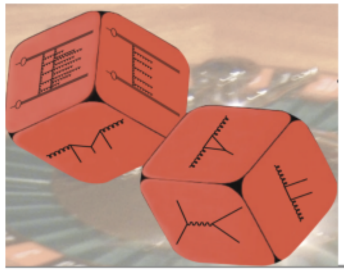
$$f(x) = \frac{1}{s\sqrt{2\pi}} \exp\left[ -\frac{(x-a)^2}{2s^2} \right]$$

- example: take sum of uniformly distributed random numbers:

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

- for Gaussian with mean=0 and variance=1, take for n=12:





# Central Limit Theorem



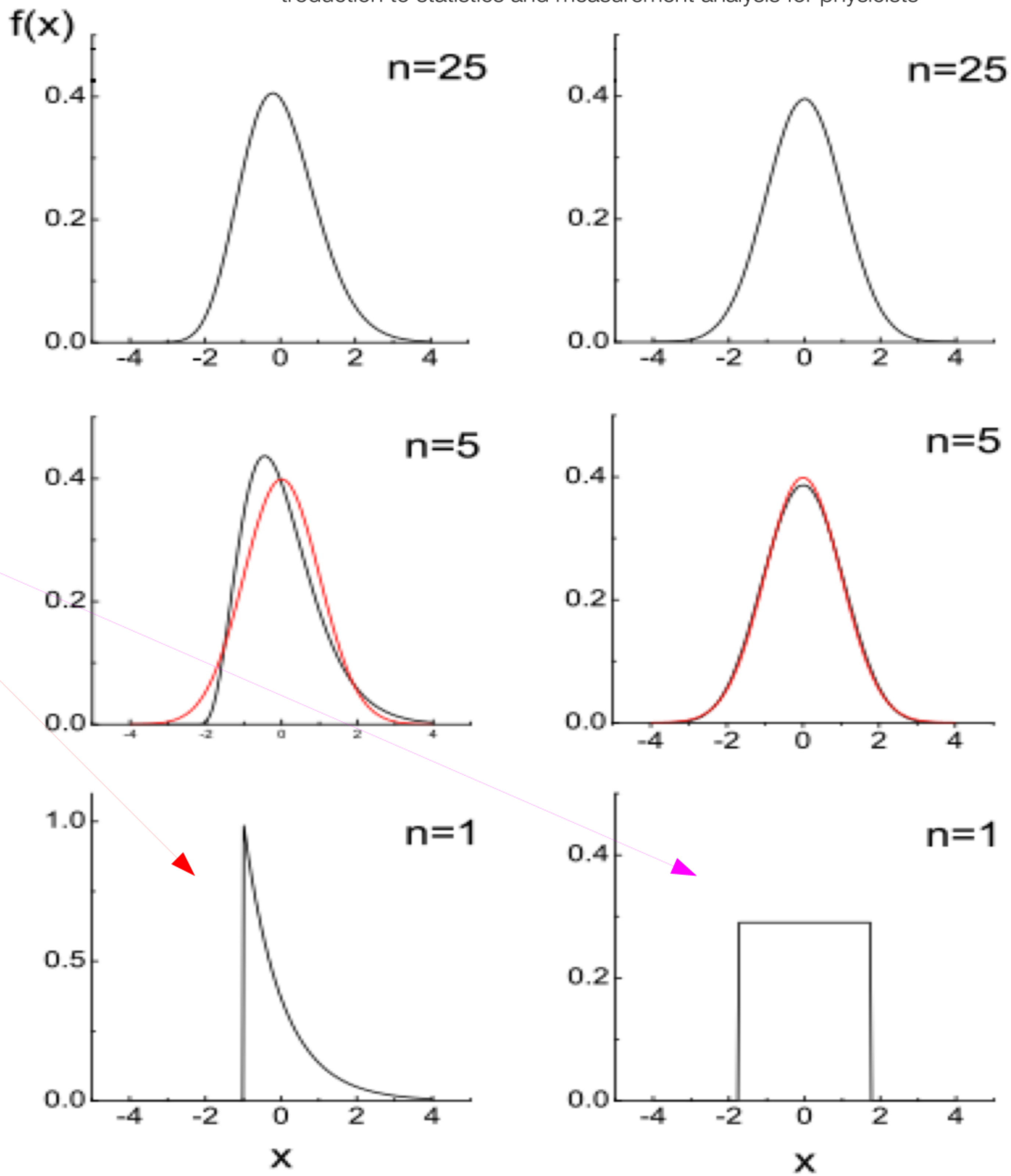
- Central Limit Theorem
  for large N the sum of independent random variables is always normally (Gaussian) distributed

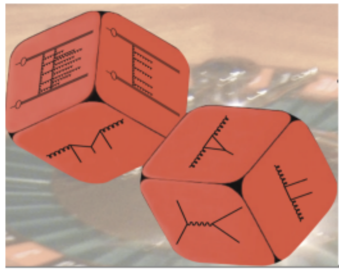
➔ for any starting distribution

➔ for uniform distribution

➔ for exponential distribution

G. Bohm, G. Zechln
troduction to statistics and measurement analysis for physicists





# Example 3

Terascale Summer School: Tutorial/Exercises - QCD and Monte Carlo techniques

- write a small program that integrates (with Monte Carlo method) the function

$$f(x) = 3x^2$$

for

$$\int_0^1 f(x)dx$$

and calculate the uncertainty.

- Writeup:
  - Page 18, eq (2.26 – 2.33)

### 2.5 Monte Carlo Integration

Already in the previous sections we had to deal with the problem to obtain a reliable estimate of the true value of an integral [9]:

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

The integral $I$ is directly connected to the expectation value of the function $f(x)$ with the $x$ values distributed according to a probability density function $g(x)$.

$$E[f] = \int_{-\infty}^{\infty} f(x)g(x)dx$$

where the *p.d.f.* $g(x)$ must be defined such, that it vanishes outside the range of $(a, b)$. In the case of uniformly distributed $x$ this reduces to $g(x) = 1/(b-a)$ for $a < x < b$ (and $g(x) = 0$ otherwise) which gives:

$$E[f] = \int_{-\infty}^{\infty} f(x)g(x)dx = \frac{1}{b-a}\int_a^b f(x)dx$$

The Monte Carlo estimate of the integral is then:

$$I \approx I_{MC} = (b-a)\frac{1}{N}\sum_{i=1}^{N} f(x_i) \tag{2.26}$$

and the variance is:

$$V[I_{MC}] = \sigma_I^2 = V\left[(b-a)\frac{1}{N}\sum_{i=1}^{N} f(x_i)\right] \tag{2.27}$$

$$= \frac{(b-a)^2}{N^2} V\left[\sum_{i=1}^{N} f(x_i)\right] \tag{2.28}$$

$$= \frac{(b-a)^2}{N} V[f] \tag{2.29}$$

*2.5. MONTE CARLO INTEGRATION* 19

The variance depends on the number of times the integrand is evaluated, but also on the variance of $f$: $V[f]$.
Applying the definition of the variance eq.(2.11), the variance $V[f]$ becomes (with $\bar{f} = \int f dx = 1/N \sum f_i$ and assuming $g(x)$ being uniform):

$$V[f] = \int (f-\bar{f})^2 g dx = \int (f^2 - 2f\bar{f} + \bar{f}^2) g dx \tag{2.30}$$

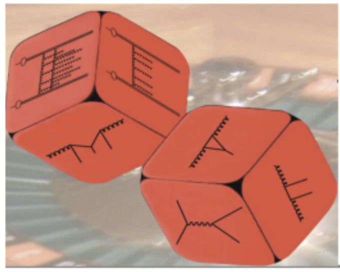
$$= \int f^2 g dx - \bar{f}^2 \tag{2.31}$$

$$= \sum \frac{f_i^2}{N} - \left(\frac{\sum f_i}{N}\right)^2 \tag{2.32}$$

(2.33)

Then the $V[I]$ becomes:

$$V[I] = \frac{1}{N}(b-a)^2\left(\frac{1}{N}\sum f_i^2 - \left(\frac{\sum f_i}{N}\right)^2\right)$$



# Monte Carlo Integration



- solve

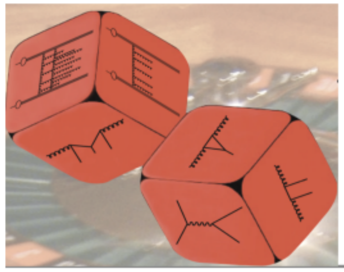
$$I = \int_a^b f(x)dx = (b-a)E[f(x)]$$

- estimate by

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

- with variance

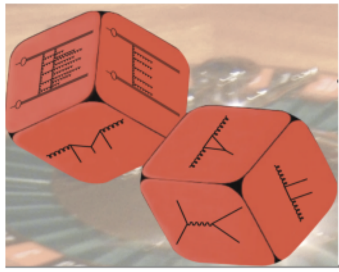
$$\begin{aligned} V[I_{MC}] &= \sigma_I^2 = V\left[\frac{b-a}{n}\sum_i f(x_i)\right] \\ &= \frac{(b-a)^2}{n} V[f] \\ &= \frac{(b-a)^2}{n}\left[\frac{\sum_i f(x_i)^2}{n} - \left(\frac{\sum_i f(x_i)}{n}\right)^2\right] \\ &= \frac{1}{n}\left[(b-a)^2\frac{\sum_i f(x_i)^2}{n} - I_{MC}^2\right] \end{aligned}$$





# Breakout Room

- Now we have the tutorial for the first exercises: 30 min
  - work with your tutors in breakout rooms
    - Sara: Tutorial with C++
    - Armando: Tutorial with Jupyter
    - Mikel: Tutorial with Jupyter





# Example 4

- write a small program that integrates (with Monte Carlo method)

$$\int_0^1 \int_0^x dxdy$$

- Writeup:
  - Page 20ff

- **brute force method**
  The accept-reject method also works for MC integration. Defining $I_0$ as the area in $[a, b]$ and $f_{max}$ as the maximum of the function $f(x)$ in this range. With a random

2.5. *MONTE CARLO INTEGRATION* 21

number $R_i$ we generate $x_i$ and another random number $R_j$ is used to accept or reject the pair of random numbers $i, j$ according to:

$$\begin{aligned} &\text{if } f(x_i) < R_j \cdot f_{max} \rightsquigarrow \text{reject} \\ &\text{if } f(x_i) > R_j \cdot f_{max} \rightsquigarrow \text{accept} \end{aligned}$$

We count the number of trails with $N_0$ and the number of accepted events with $N$. Then we obtain:

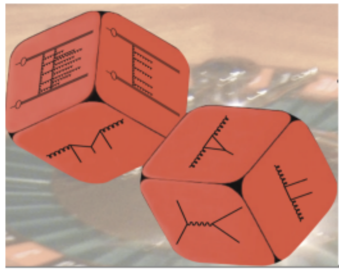
$$\begin{aligned} I &= \int_a^b f(x)dx \\ &= I_0 \frac{N}{N_0} \end{aligned}$$

The variance $V[r] = (\delta(N))^2 = \sigma^2$ is (using binomial statistics with $E[r] = N_0 P$ and $V[r] = N_0 P(1 - P)$ with $P = N/N_0$):

$$V[r] = N(1 - P)$$

With this we can calculate the uncertainty of the integral estimate $\delta(I)$ as:

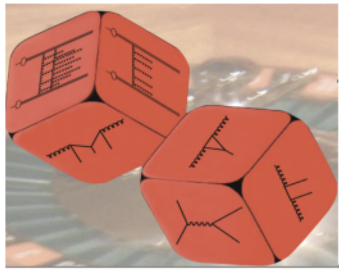
$$\frac{\delta I}{I} = \frac{I_0 \sigma / N_0}{I_0 N / N_0} = \sqrt{\frac{N(1-P)}{N^2}}$$



# Generating distributions



- Brute Force or Hit & Miss method
  - use this if there is no easy way to find a analytic integrable function
  - find $c \leq \max f(x)$
  - reject if $f(x_i) \leq u_j \cdot c$
  - accept if $f(x_i) \geq u_j \cdot c$

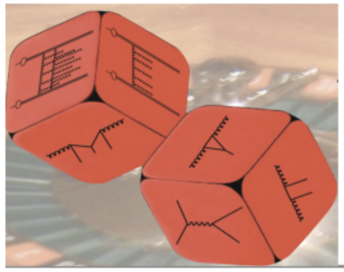


# MC method: hit & miss



- **integration ☛ weighting events**
  - large fluctuations from large weights
  - weights also to errors applied
  - difficult to apply further hadronization
- **real events all have weight = 1 !!!**
- **Hit & Miss method:**

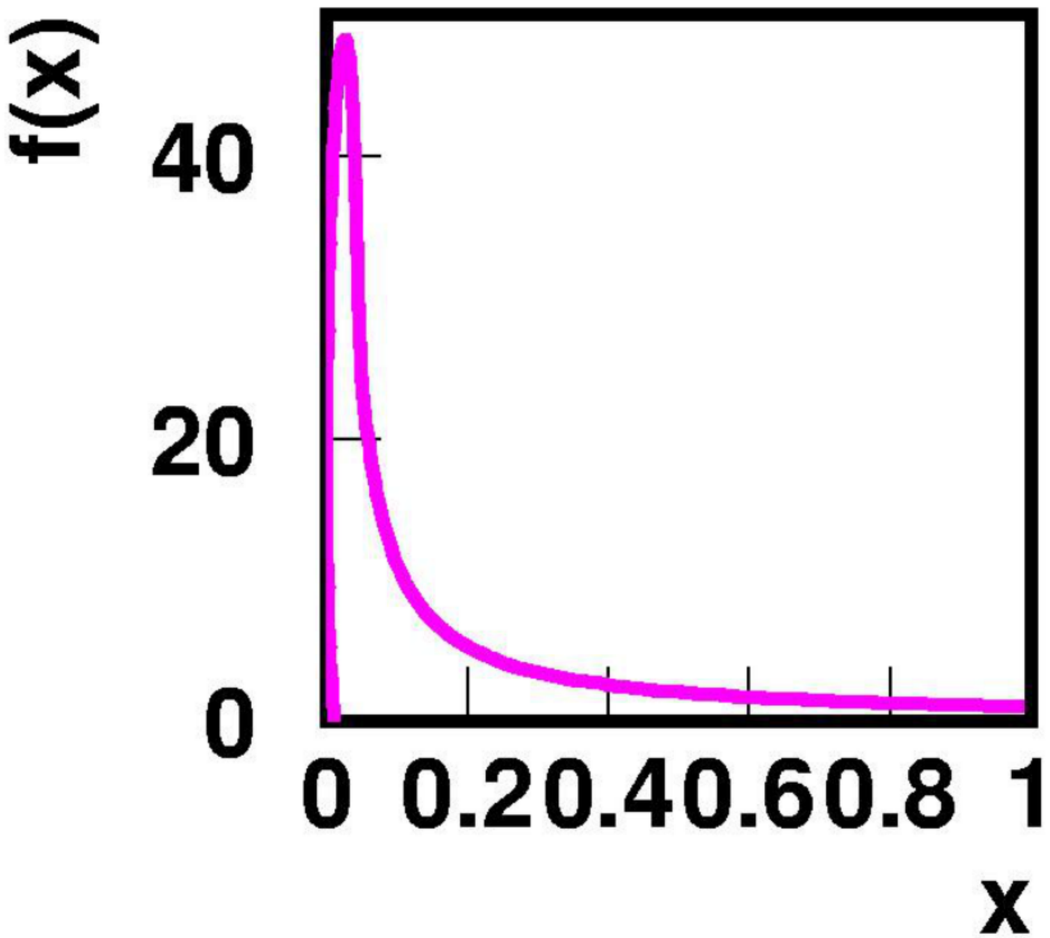
> **MC for function f(x):**
> get random number:
> R1 in (0,1) and R2 in (0,1)
> calculate x = R1
> **reject event if: $f_x < f_{max}$ R2**



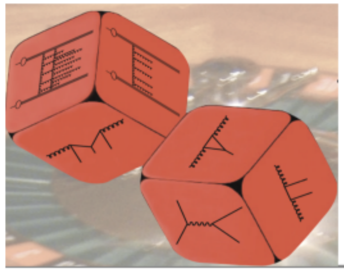
# MC method: hit & miss



- **integration ☛ weighting events**
  - **large fluctuations from large weights**
  - **weights also to errors applie**
  - **difficult to apply further hadronization**
- **real events all have weight = 1 !**
- **Hit & Miss method:**



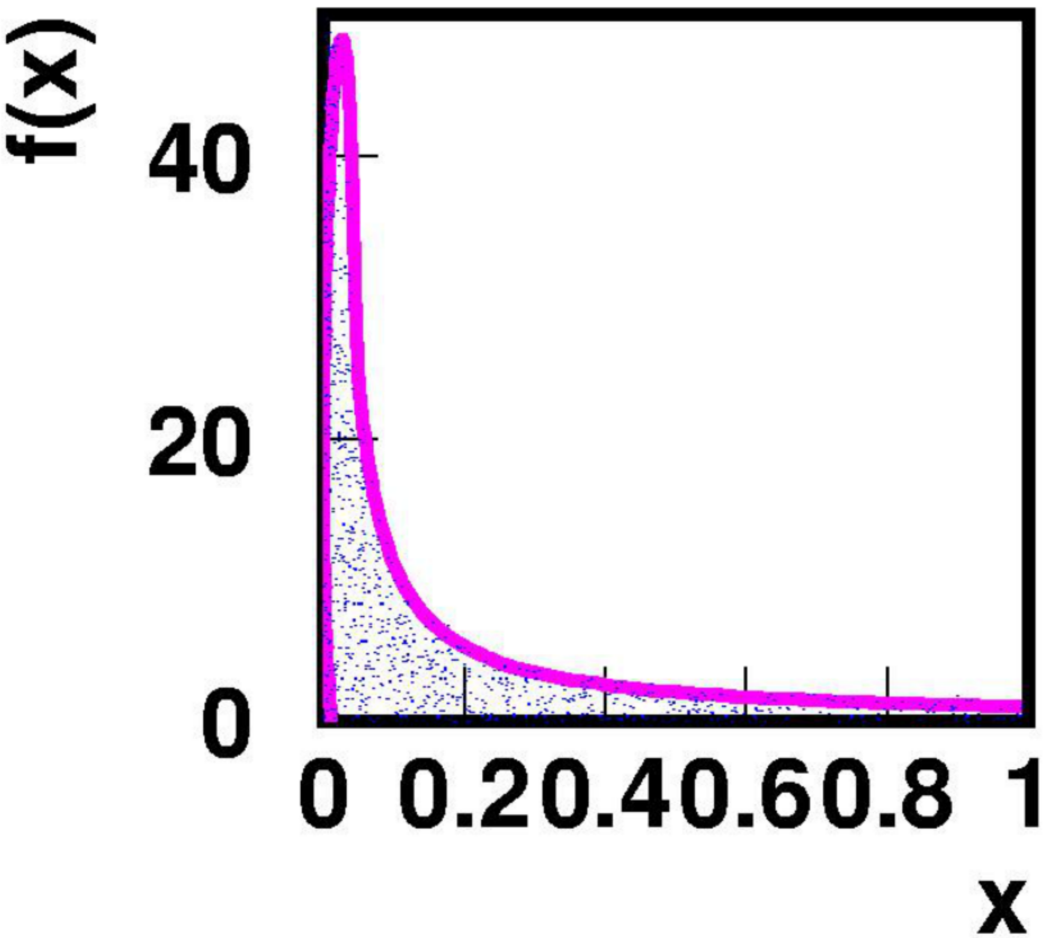

**MC for function f(x):**
**get random number:**
**R1 in (0,1) and R2 in (0,1)**
**calculate x = R1**
**reject event if: $f_x < f_{max}$ R2**



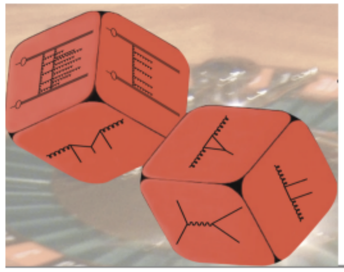
# MC method: hit & miss



- **integration ☛ weighting events**
  - large fluctuations from large weights
  - weights also to errors applie[d]
  - difficult to apply further hadronization
- **real events all have weight = 1 !**
- **Hit & Miss method:**




**MC for function f(x):**
get random number:
R1 in (0,1) and R2 in (0,1)
calculate x = R1
**reject event if: $f_x < f_{max}$ R2**

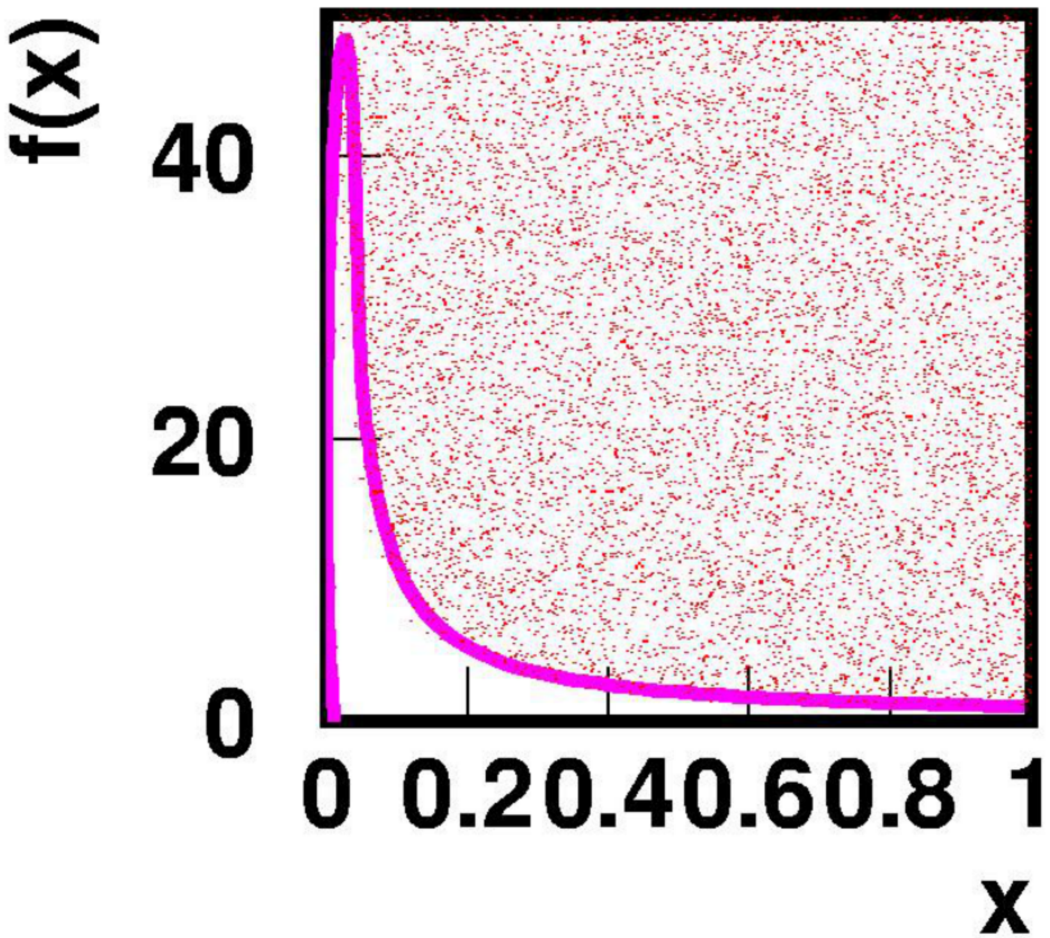


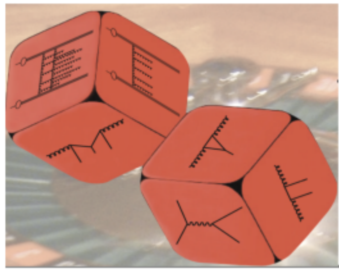
# MC method: hit & miss



- **integration ☛ weighting events**
  - large fluctuations from large weights
  - weights also to errors applie
  - difficult to apply further hadronization
- **real events all have weight = 1 !**
- **Hit & Miss method:**

**MC for function f(x):**
get random number:
R1 in (0,1) and R2 in (0,1)
calculate x = R1
**reject event if: $f_x < f_{max}$ R2**

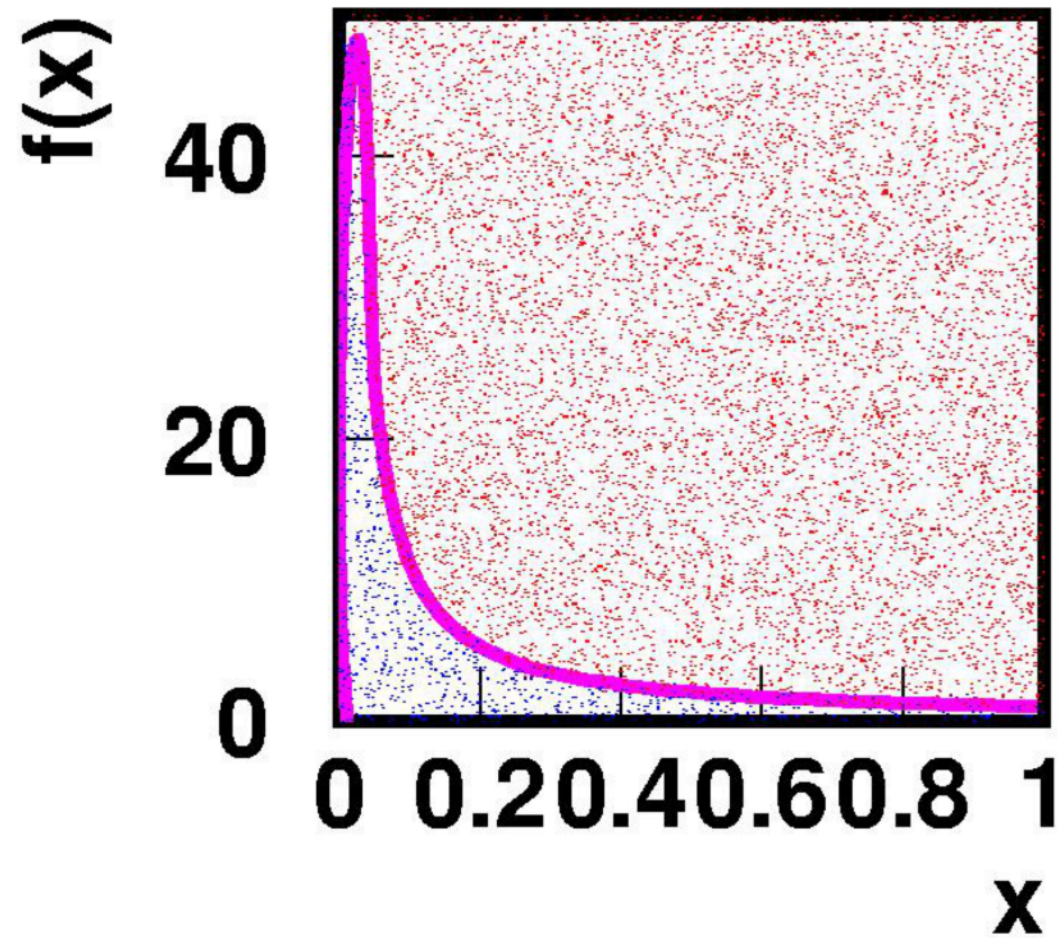




# MC method: hit & miss

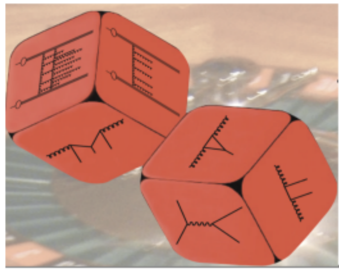
Terascale Summer School: Tutorial/Exercises - QCD and Monte Carlo techniques

- **integration ☛ weighting events**
  - large fluctuations from large weights
  - weights also to errors applie
  - difficult to apply further hadronization
- **real events all have weight = 1 !**
- **Hit & Miss method:**




**MC for function f(x):**
get random number:
R1 in (0,1) and R2 in (0,1)
calculate x = R1
**reject event if: $f_x < f_{max}$ R2**

- **BUT: Hit & Miss method inefficient for peaked *f*(*x*)**



# MC method: hit & miss

Terascale Summer School: Tutorial/Exercises - QCD and Monte Carlo techniques

- Integral in hit & miss

$$
\begin{aligned}
I &= \int_a^b f(x)dx \\
&= I_0 \frac{N}{N_0}
\end{aligned}
$$

- Variance $V[r] = (\delta(N))^2 = \sigma^2$ using binomial statistics with

  $E[r] = N_0 P$ and $V[r] = N_0 P(1-P)$ with $P = N/N_0$

  giving $V[r] = N(1-P)$

- uncertainties in hit & miss method:

$$
\frac{\delta I}{I} = \frac{I_0 \sigma / N_0}{I_0 N / N_0} = \sqrt{\frac{N(1-P)}{N^2}}
$$

# Example 5

- write a small program to integrate a simple function in one dimension:

$$\int_{x_{min}}^{1} g(x)dx = \int_{x_{min}}^{1} (1-x)^5 \frac{dx}{x}$$

- Improve the above integration by using importance sampling.

- Writeup: eq 2.20 ff
  - Page 19ff

- **importance sampling**

  If an approximate function $g(x)$ exists then the integral $I$ can be estimated to:

$$\begin{aligned} I = \int_a^b f(x)dx &= \int_a^b \frac{f(x)}{g(x)} g(x)dx \\ &= \int h(x)g(x)dx \\ &= E\left[\frac{f(x)}{g(x)}\right] \end{aligned}$$

provided $g(x)$ is normalized and integrable in $[a,b]$. Thus the integration reduces to calculating the expectation value of $E[f/g]$, if the values of $x$ are distributed according the *p.d.f* $g(x)$. The values of $x$ can be generated according to the methods discussed in the previous sections and we obtain:

$$I = \frac{1}{N}\sum \frac{f(x_i)}{g(x_i)} \qquad (2.34)$$

20 *CHAPTER 2. MONTE CARLO METHODS*

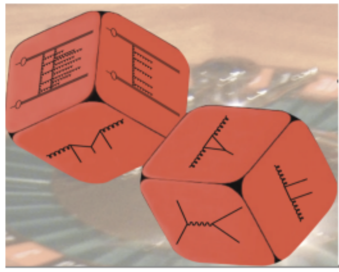
We assume that $g(x)$ is a *p.d.f* normalized to 1 in the integration range. For example using $g(x) = (1/x) 1/\log\left(\frac{x_{max}}{x_{min}}\right)$ (see eq.(2.23)) gives then:

$$I = \frac{\log\left(\frac{x_{max}}{x_{min}}\right)}{N} \sum \frac{f(x_i)}{\frac{1}{x_i}}. \qquad (2.35)$$

The variance is then given by:

$$V\left[\frac{f(x)}{g(x)}\right] = E\left[\left(\frac{f(x)}{g(x)} - E\left[\frac{f(x)}{g(x)}\right]\right)^2\right] \qquad (2.36)$$

A danger in this method is when $g(x)$ becomes zero or approaches zero quickly [3].

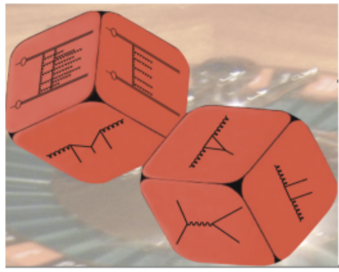


# Generating distributions



- From uniform distributions to distributions for any probability density function
    - use variable transformation
    - 1/x distribution

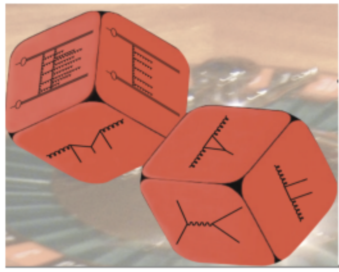
$$
\begin{aligned}
f(x) &= \frac{1}{x} \\
R \int_{x_{min}}^{x_{max}} f(x')dx' &= \int_{x_{min}}^{x} f(x')dx' \\
R \log \frac{x_{max}}{x_{min}} &= \frac{x}{x_{min}} \\
\log \left( \frac{x_{max}}{x_{min}} \right)^R &= \frac{x}{x_{min}} \\
x_j &= x_{min} \left( \frac{x_{max}}{x_{min}} \right)^R
\end{aligned}
$$



# Generating distributions



- Improvements for Hit & Miss method by variable transformation
  - find $c \cdot g(x) \geq f(x)$
  - reject if $f(x) \leq u_j \cdot c \cdot g(x)$
  - accept if $f(x) \geq u_j \cdot c \cdot g(x)$



# MC method: doing better



- Importance sampling

**MC for function f(x)**
**approximate f(x) ~ g(x)**
**with g(x) > f(x) simple and integrable**
**generate x according to g(x):**

$$\int_{x_{min}}^{x} g(x')dx' = R1 \int_{x_{min}}^{x_{max}} g(x')dx'$$

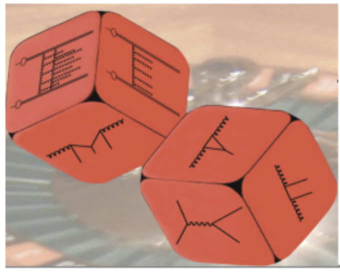
**example:**

$$f(x) = 1/x^{0.7}$$

$$g(x) = 1/x$$

$$x = x_{min} \cdot \left(\frac{x_{max}}{x_{min}}\right)^{R1}$$

**reject event if: f(x) < g(x) R2**



# MC method: doing better



- Importance sampling

**MC for function f(x)**
**approximate f(x) ~ g(x)**
**with g(x) > f(x) simple and integrable**
**generate x according to g(x):**

$$\int_{x_{min}}^{x} g(x')dx' = R1 \int_{x_{min}}^{x_{max}} g(x')dx'$$

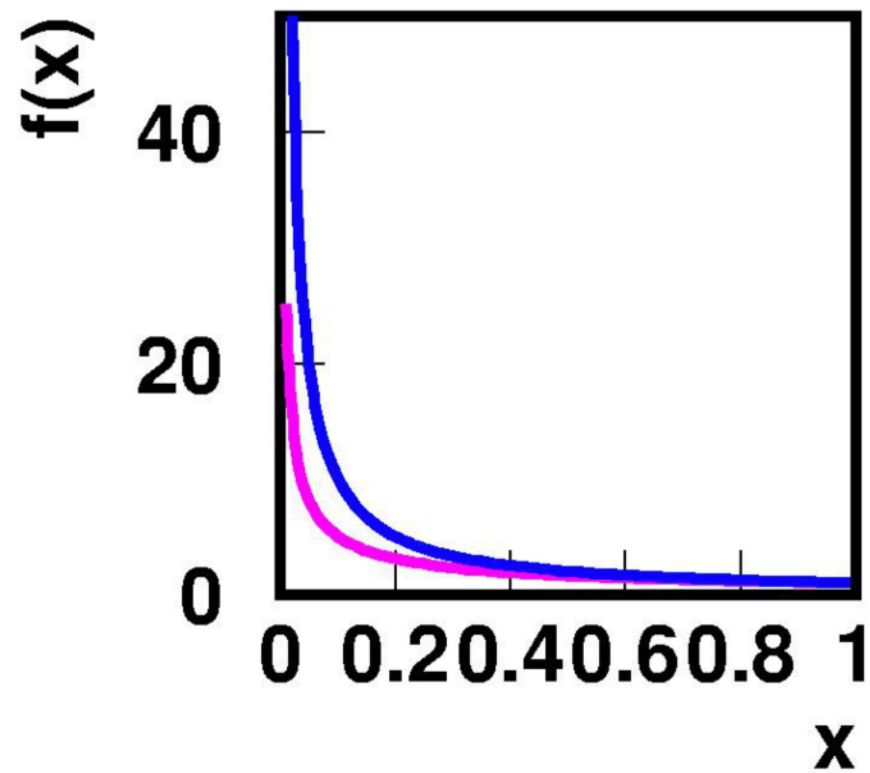
**example:**

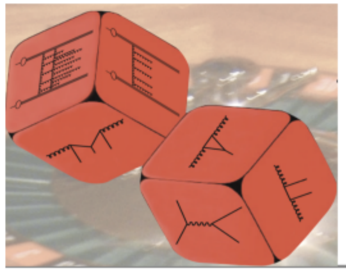
$$f(x) = 1/x^{0.7}$$

$$g(x) = 1/x$$

$$x = x_{min} \cdot \left(\frac{x_{max}}{x_{min}}\right)^{R1}$$

**reject event if: f(x) < g(x) R2**





# MC method: doing better



- Importance sampling

**MC for function f(x)**
**approximate f(x) ~ g(x)**
**with g(x) > f(x) simple and integrable**
**generate x according to g(x):**

$$\int_{x_{min}}^{x} g(x')dx' = R1 \int_{x_{min}}^{x_{max}} g(x')dx'$$

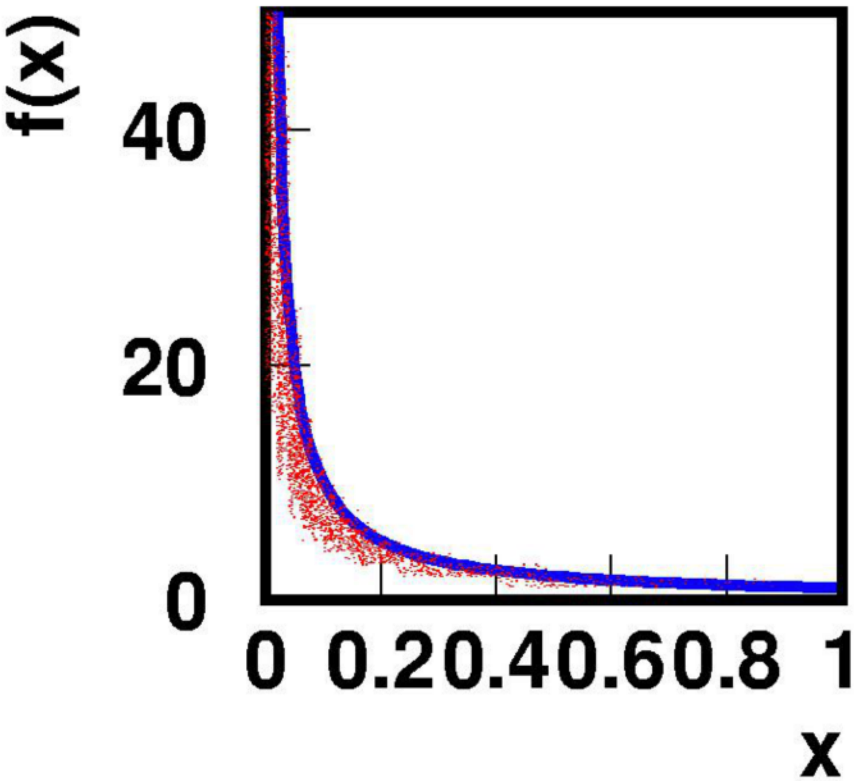
**example:**

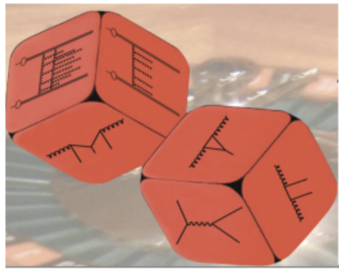
$$f(x) = 1/x^{0.7}$$

$$g(x) = 1/x$$

$$x = x_{min} \cdot \left(\frac{x_{max}}{x_{min}}\right)^{R1}$$

**reject event if: f(x) < g(x) R2**





# MC method: doing better

Terascale Summer School: Tutorial/Exercises - QCD and Monte Carlo techniques

- Importance sampling

**MC for function f(x)**
**approximate f(x) ~ g(x)**
**with g(x) > f(x) simple and integrable**
**generate x according to g(x):**

$$\int_{x_{min}}^{x} g(x')dx' = R1 \int_{x_{min}}^{x_{max}} g(x')dx'$$

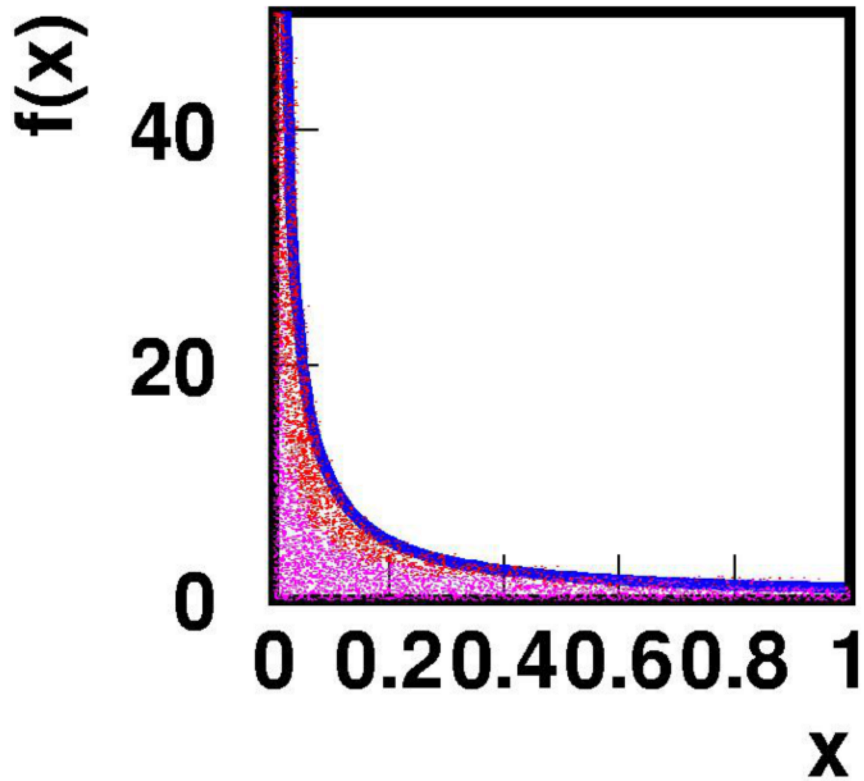
**example:**

$$f(x) = 1/x^{0.7}$$

$$g(x) = 1/x$$

$$x = x_{min} \cdot \left( \frac{x_{max}}{x_{min}} \right)^{R1}$$

**reject event if: f(x) < g(x) R2**



# Example lhapdf

- use the LHAPDF library to calculate the flavor sum rules

$$\int_0^1 dx\, u_V(x, Q^2) = 2$$

$$\int_0^1 dx\, d_V(x, Q^2) = 1$$

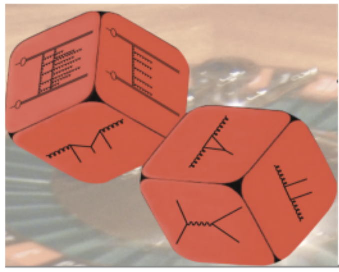
- use the LHAPDF library and calculate the momentum sum rule:

$$\int_0^1 dx \sum_{i=-6}^{6} x p_i(x, Q^2)$$

- use the MRST(MRST2004nlo) set and the LO* (MRST2007lomod) set. How much is the momentum sum rule violated in the LO* set ?

Is the momentum sum rule satisfied (or violated in the same way) for different $Q^2$ values ( $Q^2 = 5, 10, 100, 1000\ GeV^2$ ).

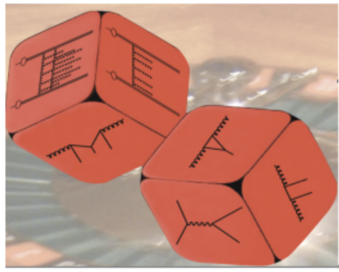
- Improve the above integration by using importance sampling.



# Breakout Room



- Now we have the tutorial for the second exercises: 30 min
  - work with your tutors in breakout rooms
    - Sara: Tutorial with C++
    - Armando: Tutorial with Jupyter
    - Mikel: Tutorial with Jupyter



# Literature for MC Method



- Monte Carlo and QCD lectures
  - https://www.desy.de/~jung/QCD_and_Monte_Carlo_lectures.html

- G. Bohm, G. Zech (2010) Online Book
  Introduction to Statistics and Data Analysis for Physicists
- S. Weinzierl Introduction to Monte Carlo method hep-ph/0006269
- G. Cowan. Statistical data analysis, Oxford, UK: Clarendon (1998)
- J. Vermaseren, Lectures on Monte Carlo, Madrid 2008 ( http://www.nikhef.nl/~form/maindir/courses/course2/course2.html/)
- History of Monte Carlo Method
  (http://www.geocities.com/CollegePark/Quad/2435/history.html)