How to build a MC generator

Tomasz Golan
University of Wroclaw

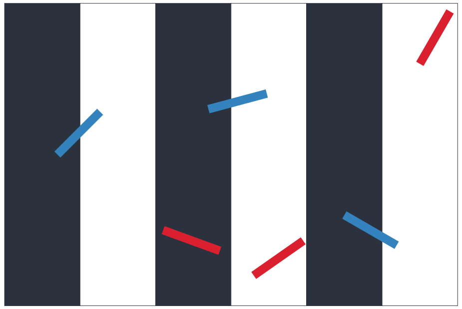
NuSTEC, Fermilab 2017

Monte Carlo method

Buffon's needle problem

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

Georges-Louis Leclerc,
Comte de Buffon
18th century



blue are good

red are bad

Monte Carlo without computers

If needle length (l) $<$ lines width (t):

$$P = \frac{2l}{t\pi}$$

which can be used to estimate π:

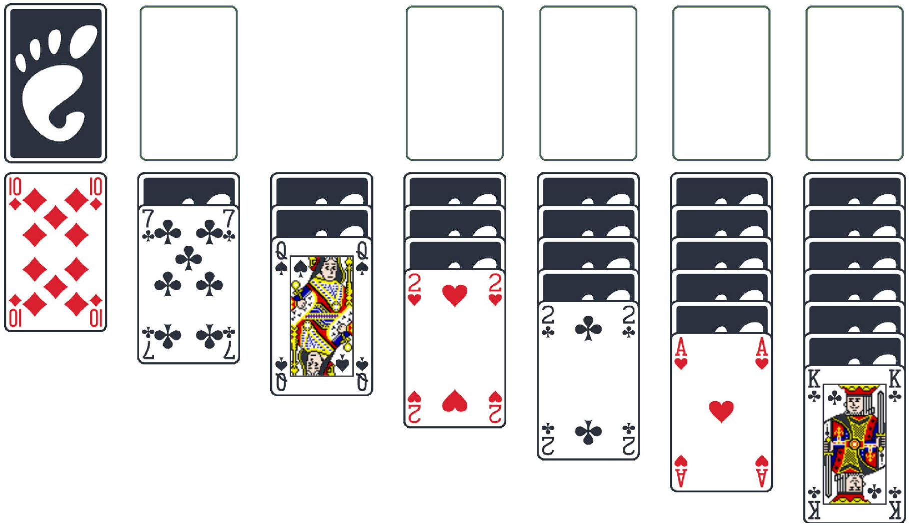
$$\pi = \frac{2l}{tP}$$

MC experiment was performed by Mario Lazzarini in 1901 by throwing 3408 needles:

$$\pi = \frac{2l \cdot 3408}{t \cdot \#red} = \frac{355}{113} = 3.14159292$$

From Solitaire to Monte Carlo method

- Stanisław Ulam was a Polish mathematician
- He invented the Monte Carlo method while playing solitaire
- The method was used in Los Alamos, performed by ENIAC computer



- What is a probability of success in solitaire?
  - Too complex for an analytical calculations
  - Lets try $N = 100$ times and count wins
  - With $N \to \infty$ we are getting closer to correct result

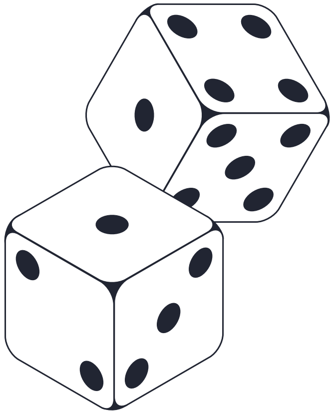
Newton-Pepys problem

Which of the following three propositions has the greatest chance of success?

A *Six fair dice are tossed independently and at least one "6" appears.*

B *Twelve fair dice are tossed independently and at least two "6"s appear.*

C *Eighteen fair dice are tossed independently and at least three "6"s appear.*



Newton-Pepys problem: analytical attempt

- First, lets go back to high school and calculate this analytically
- Let $p = \frac{1}{6}$ be the probability of rolling 6
- The probability of not rolling 6 is $(1 - p)$

A six attempts, at least one six

$$P_A = 1 - (1-p)^6 \approx 0.6651$$

B twelve attempts, at least two sixes

$$P_B = 1 - (1-p)^{12} - \binom{12}{1} p (1-p)^{11} \approx 0.6187$$

C eighteen attempts, at least three sixes

$$P_C = 1 - (1-p)^{18} - \binom{18}{1} p (1-p)^{17} - \binom{18}{2} p^2 (1-p)^{16} \approx 0.5973$$

Newton-Pepys problem: MC attempt

- MC attempt is just "performing the experiment", so we will be rolling dices
- Roll $6n$ times and check if number of sixes is greater or equal n
- Repeat N times and your probability is given by:

$$P = \frac{\text{number of successes}}{N}$$

```
def throw (nSixes):
  n = 0
  for _ in range (6 * nSixes):
    if random.randint (1, 6) == 6: n += 1
  return n >= nSixes

def MC (nSixes, nAttempts):
  n = 0
  for _ in range (nAttempts):
    n += throw (nSixes)
  return float (n) / nAttempts

if __name__ == "__main__":
  for i in range (1, 4):
    print MC (i, 1000)
```

- Your MC result depends on N
- Results for $N = 100$:

$$\begin{aligned} P_A &= 0.71, 0.68, 0.76, 0.65, 0.68 & \quad P_A^{true} = 0.6651 \\ P_B &= 0.70, 0.56, 0.60, 0.63, 0.69 & \quad P_B^{true} = 0.6187 \\ P_C &= 0.62, 0.62, 0.53, 0.57, 0.62 & \quad P_C^{true} = 0.5973 \end{aligned}$$

- Results for $N = 10^6$:

$$\begin{aligned} P_A &= 0.6655, 0.6648, 0.6653, 0.6662, 0.6653 \\ P_B &= 0.6188, 0.6191, 0.6191, 0.6190, 0.6182 \\ P_C &= 0.5975, 0.5979, 0.5972, 0.5978, 0.5973 \end{aligned}$$

- Your MC results also depends on the way how random numbers were generated

Pseudorandom number generator

- PRNG is an algorithm for generating a sequence of "random" numbers
- Example: middle-square method (used in ENIAC)
  - take n-digit number as your seed
  - square it to get $2n$-digit number (add leading zeroes if necessary)
  - n middle digits are the result and the seed for next number
- Middle-square method for $n = 4$ and base seed $= 1111$:

$$\begin{aligned} 1111^2 &= 01234321 \rightarrow 2343 \\ 2343^2 &= 05489649 \rightarrow 4896 \\ &\vdots \\ 1111^2 &= 01234321 \rightarrow 2343 \end{aligned}$$

Pseudorandom number generator

- Nowadays, more sophisticated PRNGs exist, but they also suffer on some common problems:
  - periodicity / different periodicity for different base seed
  - nonuniformity of number distributions
  - correlation of successive numbers

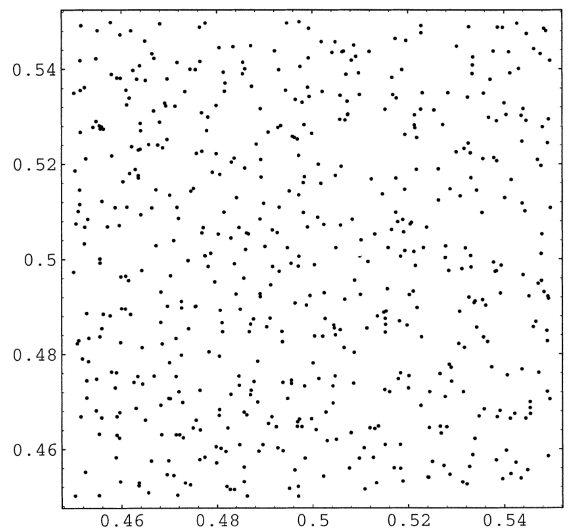


Fig. 1. LCG(2^{31}, 65539, 0, 1) Dimension 2: Zoom into the unit interval.

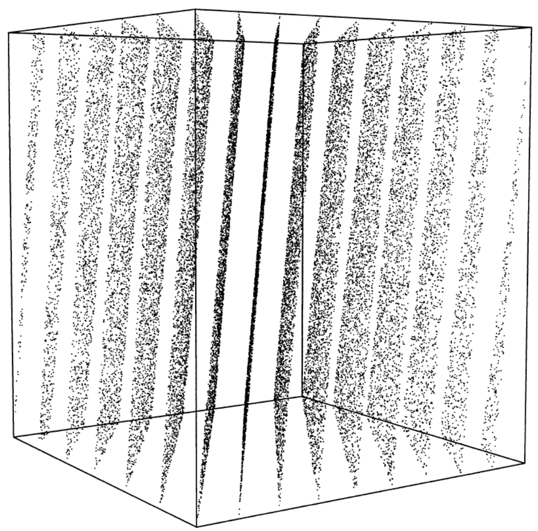


Fig. 2. LCG(2^{31}, 65539, 0, 1) Dimension 3: The 15 planes.

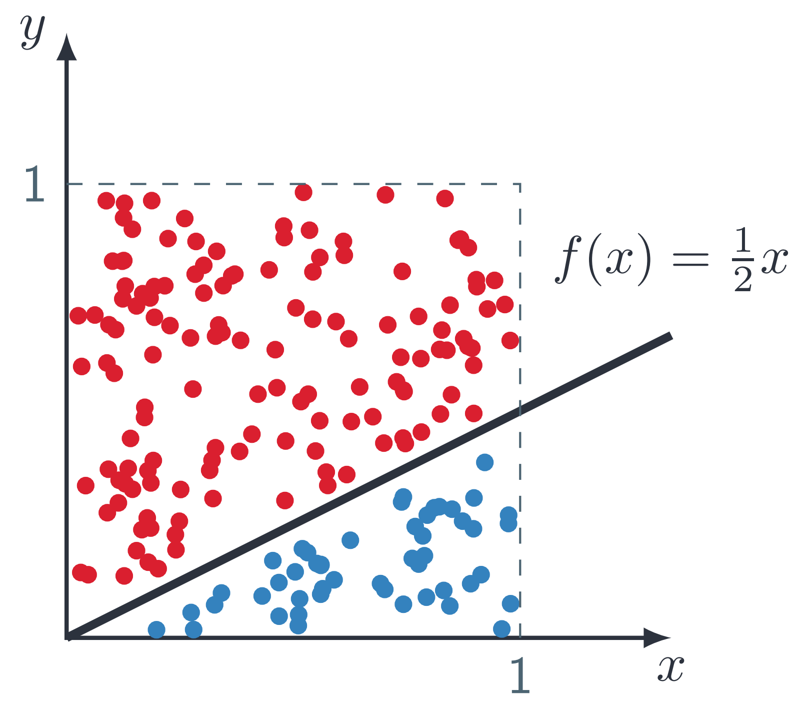
Mathematics and Computers in Simulations 46 (1998) 485-505

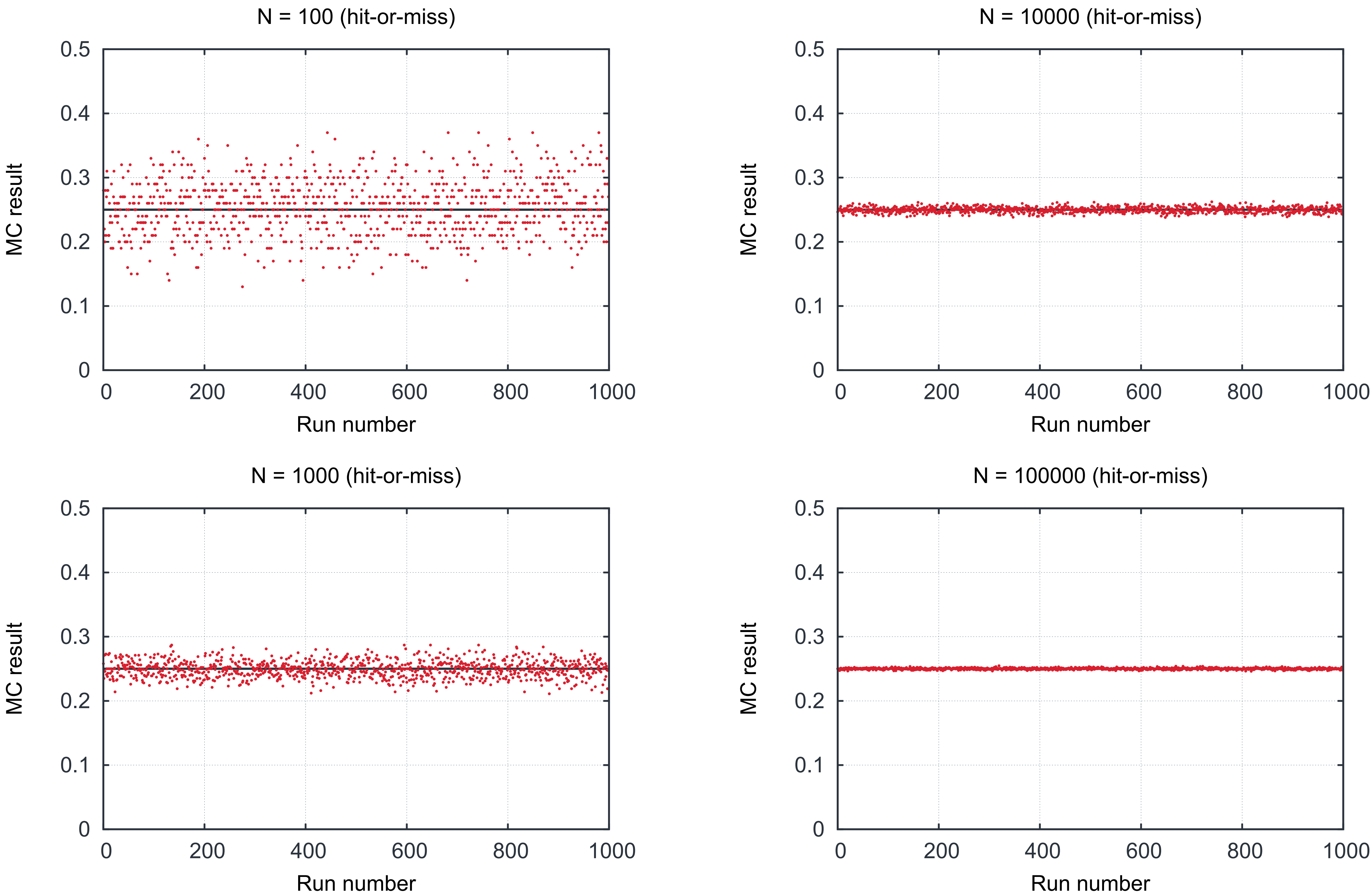
Lets do the following integration using MC method:

$$\int_0^1 f(x)dx = \int_0^1 \left(\frac{1}{2}x\right) dx = \frac{1}{2}\frac{x^2}{2}\Big|_0^1 = \frac{1}{4}$$

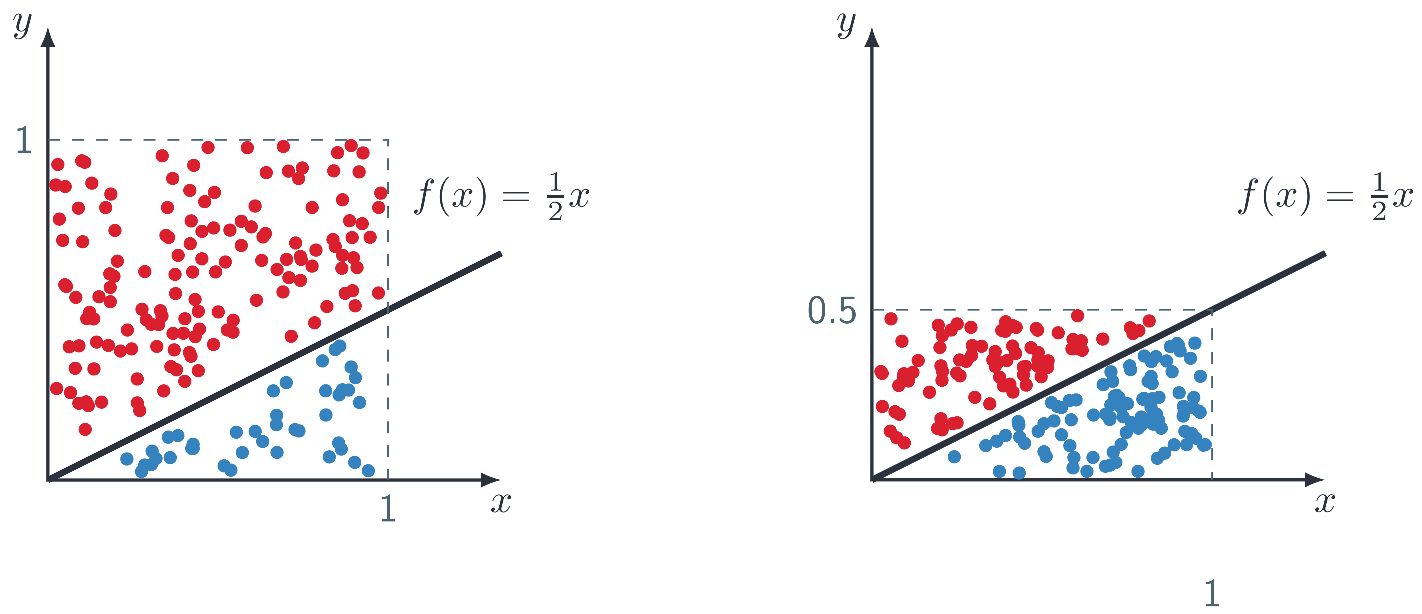
- take a random point from the $[0,1] \times [0,1]$ square
- compare it to your $f(x)$
- repeat N times
- count n points below the function
- you results is given by

$$\int_0^1 f(x)dx = P_\square \cdot \frac{n}{N} = \frac{n}{N}$$




N = 100 (hit-or-miss)
N = 10000 (hit-or-miss)
N = 1000 (hit-or-miss)
N = 100000 (hit-or-miss)
MC result
Run number
0
0.1
0.2
0.3
0.4
0.5
200
400
600
800
1000

Optimization of MC



- You want to avoid generating "red" points as they do not contribute to your integral
- You can choose any rectangle as far as it contains maximum of $f(x)$ in given range

Optimization of MC

- Lets consider the following function:

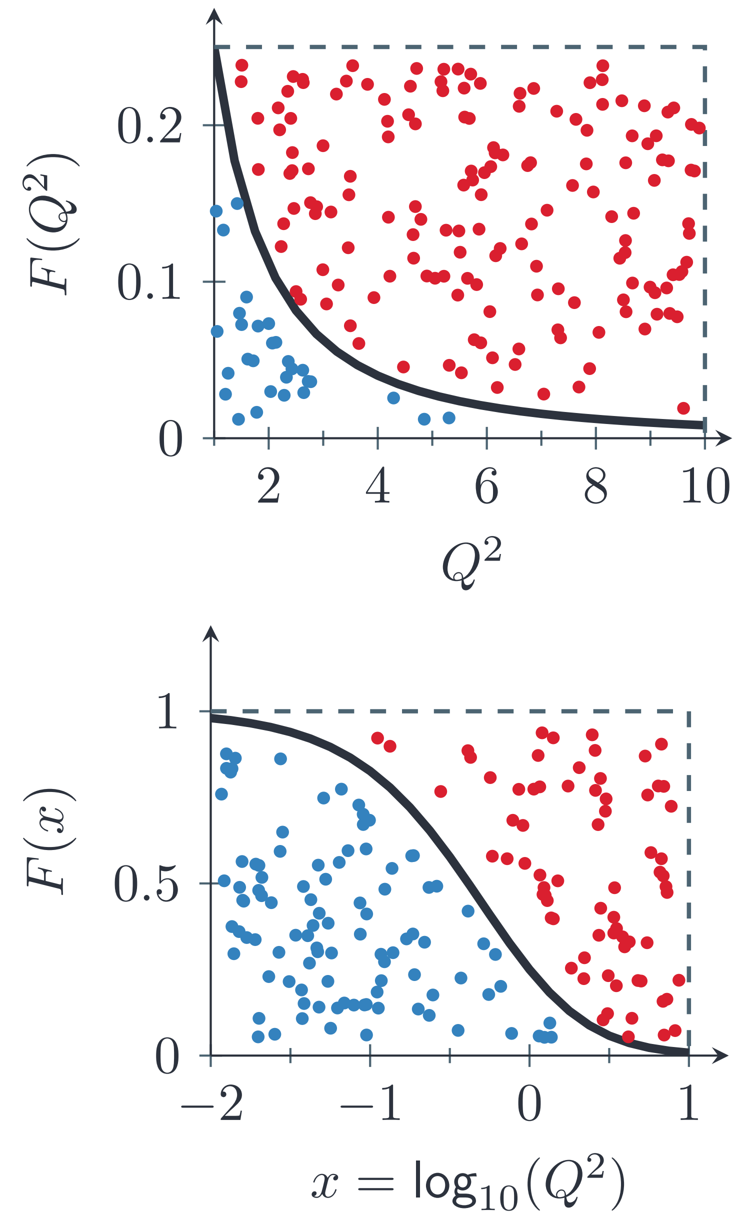
$$F(Q^2) = \frac{1}{(1+Q^2)^2}$$

more or less dipole form factor

- Integrating this function over Q^2 is highly inefficient
- However, one can integrate by substitution to get better performance, e.g.

$$x = \log_{10}(Q^2)$$

don't forget about Jacobian



Lets do the following integration using MC method once again:

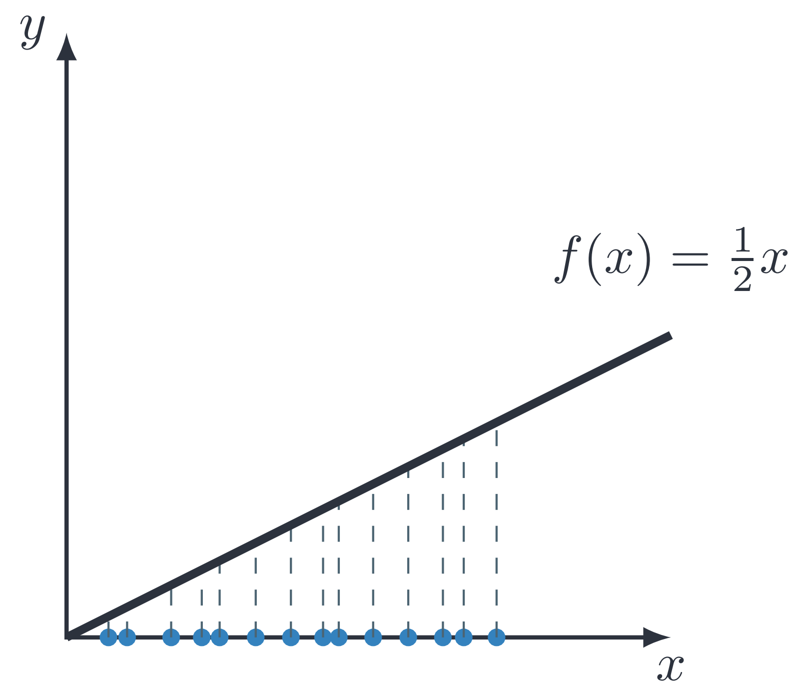
$$\int_0^1 f(x)dx = \int_0^1 \left(\frac{1}{2}x\right) dx = \frac{1}{2}\frac{x^2}{2}\Big|_0^1 = \frac{1}{4}$$

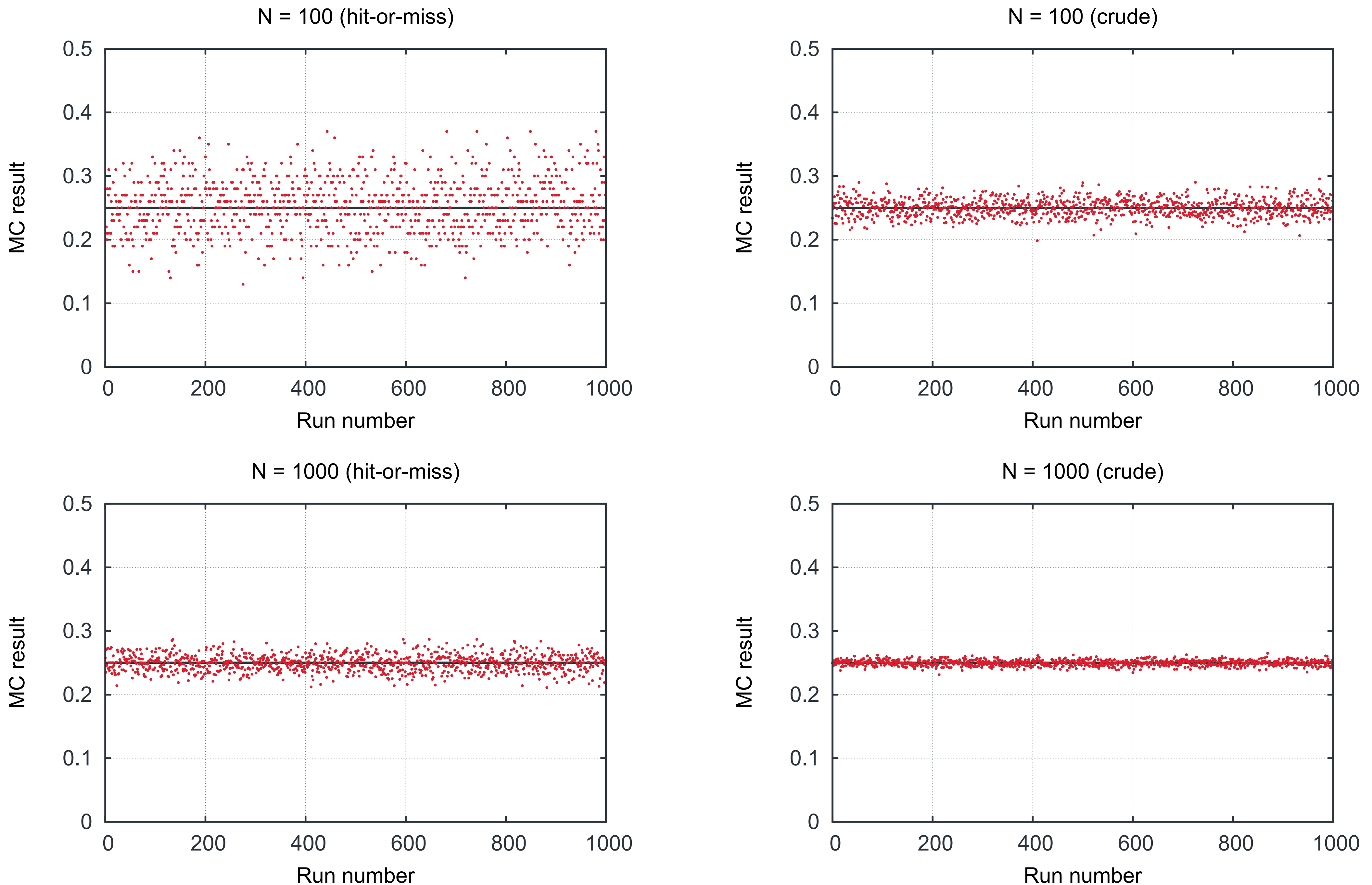
- One can approximate integral

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

where x_i is a random number from $[a, b]$

- It can be shown that crude method is more accurate than hit-or-miss
- We will skip the math and look at some comparisons

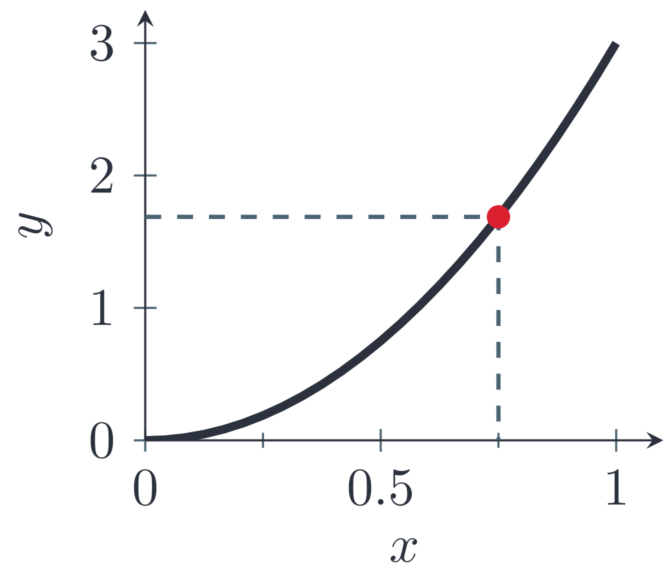




N = 100 (hit-or-miss)
N = 100 (crude)
N = 1000 (hit-or-miss)
N = 1000 (crude)
MC result
Run number
0
0.1
0.2
0.3
0.4
0.5
0
200
400
600
800
1000

Random numbers from probability density function

- How to generate a random number from probability density function?
- Lets consider $f(x) = 3x^2$
- Which means that $x = 1$ should be thrown 2 times more often than $x = \frac{\sqrt{2}}{2}$



Cumulative distribution function

- Cumulative distribution function of a random variable X:

$$F(x) = P(X \le x)$$

Note: $0 \le F(x) \le 1$ *for all* x

- Discrete random variable X:

$$F(x) = \sum_{x_i \le x} f(x_i)$$

where f is probability mass function (PMF)

- Continuous random variable X:

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

where f is probability density function (PDF)

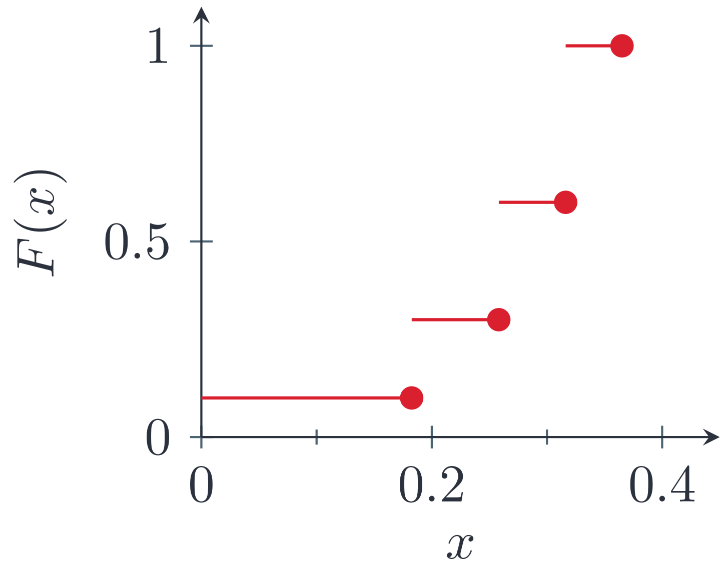
Cumulative distribution function - discrete example

- Probability mass function $f(x) = 3x^2$

  with discrete random variables X is $\{\sqrt{\frac{1}{30}}, \sqrt{\frac{2}{30}}, \sqrt{\frac{3}{30}}, \sqrt{\frac{4}{30}}, \}$

- CDF is given by:

$$F(x) = \begin{cases} \frac{1}{10} & \text{if} \quad x \leq \sqrt{\frac{1}{30}} \\ \frac{3}{10} & \text{if} \quad x \leq \sqrt{\frac{2}{30}} \\ \frac{6}{10} & \text{if} \quad x \leq \sqrt{\frac{3}{30}} \\ \frac{10}{10} & \text{if} \quad x \leq \sqrt{\frac{4}{30}} \end{cases}$$



- With $P = 1$ the random number is less or equal to $\sqrt{\frac{4}{30}}$, with $P = 0.6$ the random number is less or equal $\sqrt{\frac{3}{30}}$ ...

Cumulative distribution function - discrete example

- To generate a random number from X according to $3x^2$:
  - generate a random number u from $[0, 1]$
  - if $u \leq 0.1$: $x = \sqrt{\frac{1}{30}}$
  - else if $u \leq 0.3$: $x = \sqrt{\frac{2}{30}}$ ...
- Results for $N = 10000$:

x	n	n/N	$f(x)$
$\sqrt{\frac{1}{30}}$	989	0.0989	0.1
$\sqrt{\frac{2}{30}}$	1959	0.1959	0.2
$\sqrt{\frac{3}{30}}$	2949	0.2949	0.3
$\sqrt{\frac{4}{30}}$	4103	0.4103	0.4

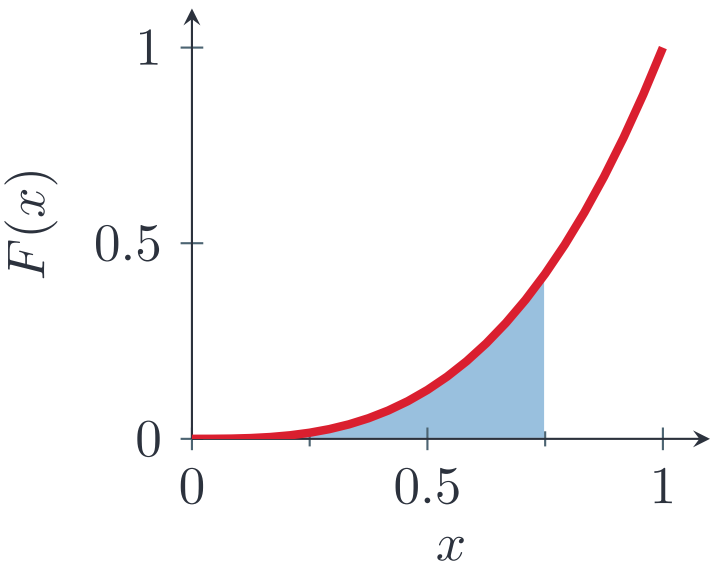
Cumulative distribution function - continuous example

- Probability density function $f(x) = 3x^2$

  with continuous random variables X range $[0, 1]$

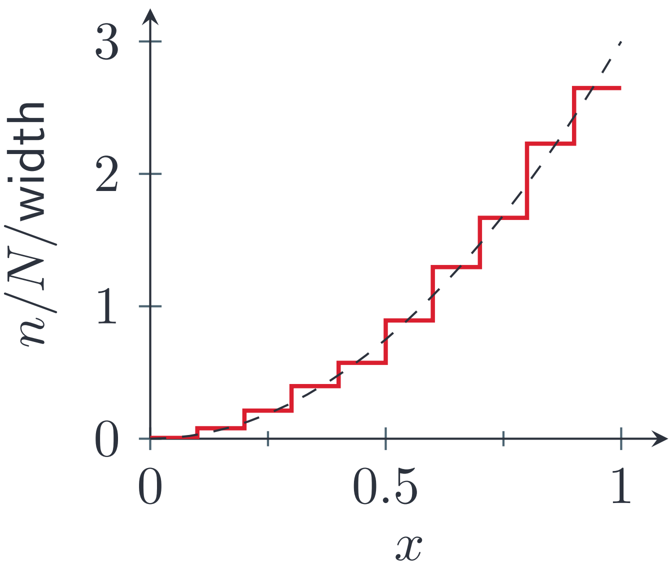
- CDF is given by:

$$\begin{aligned} F(x) &= \int_0^x f(t)dt \\ &= \int_0^x 3t^2 dt \\ &= t^3\Big|_0^x = x^3 \end{aligned}$$



- Blue area gives the probability that $x \leq 0.75$

- To generate a random number from X according to $3x^2$:
  - generate a random number u from $[0, 1]$
  - find x for which $F(x) = u$, i.e. $x = F^{-1}(u)$
  - x is your guy
- Results for $N = 10000$:



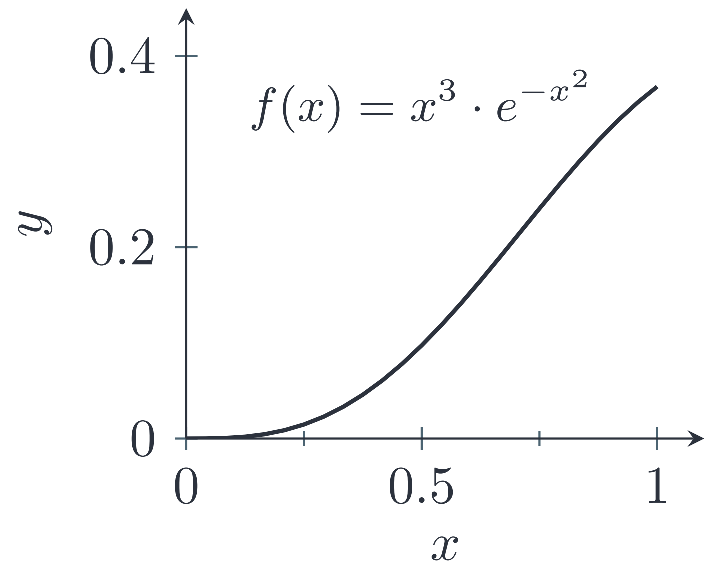
Unfortunately, usually F^{-1} is unknown, which makes this method pretty useless (at least directly).

Acceptance-rejection method

- Lets consider

$$f(x) = A \cdot x^3 \cdot e^{-x^2}$$

with $x \in [0,1]$, $A = \frac{2e}{e-2}$



- CDF is given by

$$F(x) = \frac{N}{2}(x^2 - 1)e^{-x^2}$$

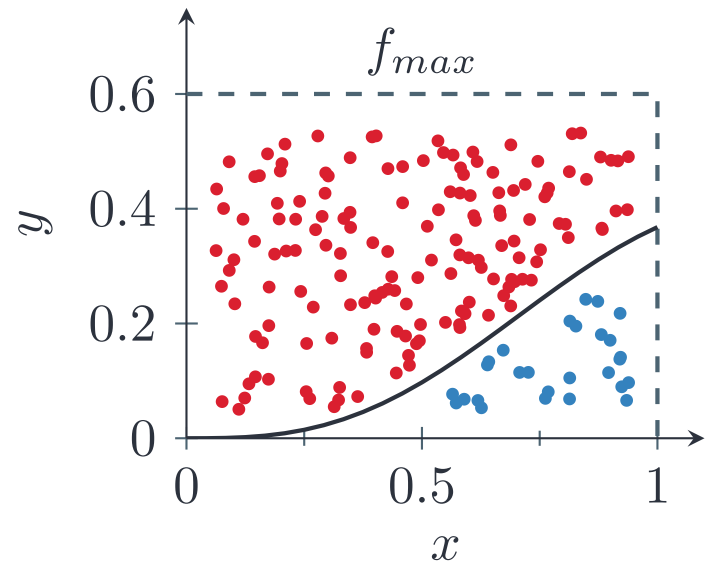
- Since, we do not know F^{-1} we have to find another way to generate x from $f(x)$ distribution

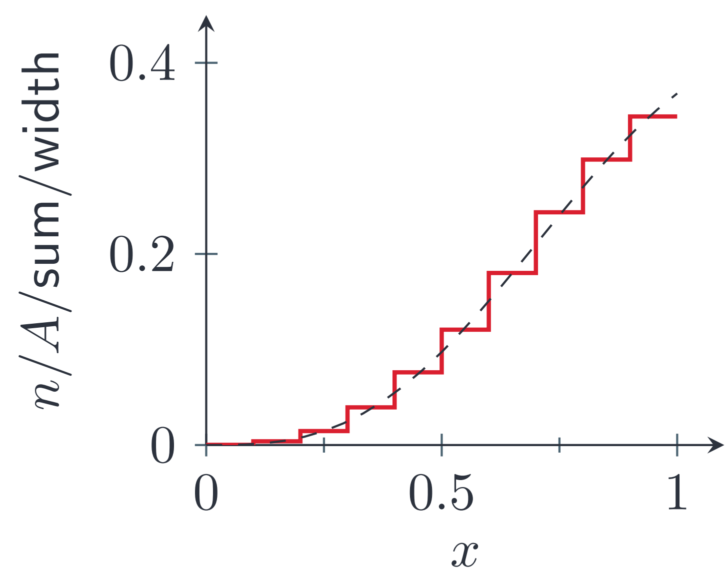
- We will use acceptance-rejection method (do you remember MC integration via hit-or-miss?)

- Evaluate $f_{max} \geq \max(f)$

  Note: $f_{max} > \max(f)$ will affect performance, but the result will be still correct

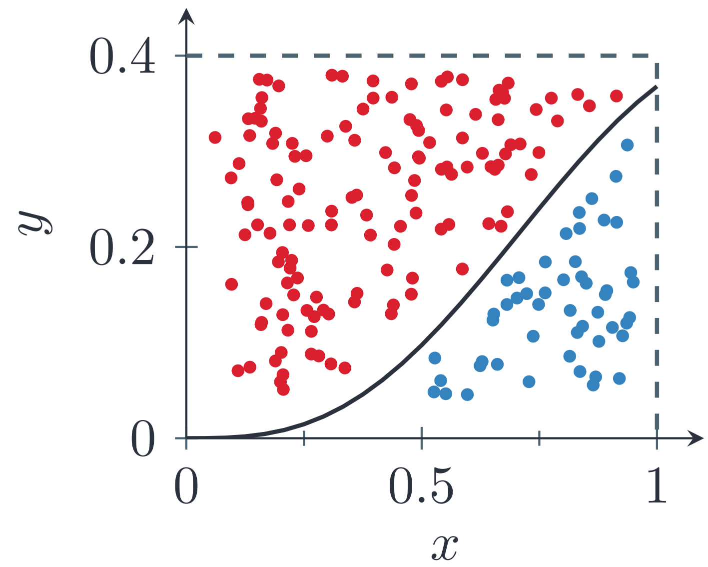
- Generate random x
- Accept x with $P = \frac{f(x)}{f_{max}}$
  - generate a random u from $[0, f_{max}]$
  - accept if $u < f(x)$
- The plot on the right shows the results for $N = 10^5$

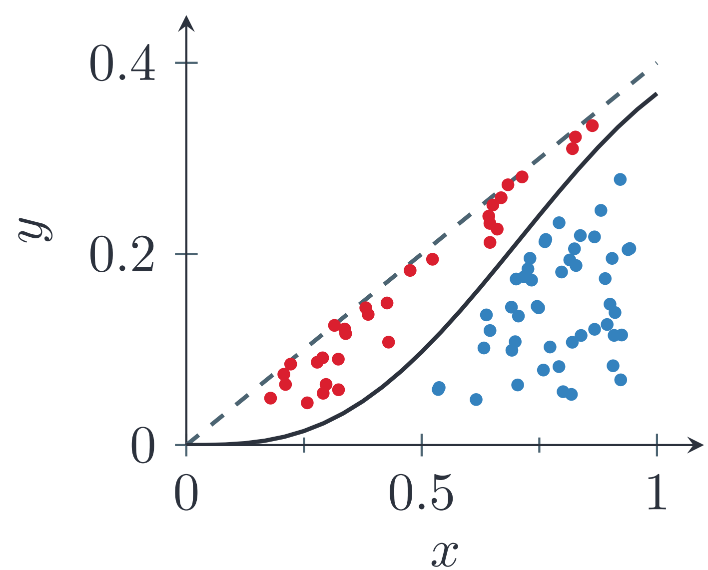




Acceptance-rejection method - optimization

- The area under the plot of $f(x)$ is ~ 0.13
- The total area is 0.4
- Thus, only about 30% of points gives contribution to the final distribution
- One can find $g(x)$ for which CDF method is possible and which encapsulates $f(x)$ in given range and generate x according to $g(x)$
- For $g(x) = 0.4x$ the total area is 0.2, so we speed up twice





- Cumulative distribution function for $g(x) = 2x$

$$G(x) = \int_0^x g(t)dt = x^2 \Rightarrow G^{-1}(x) = \sqrt{x}$$

Note: PDF must be normalized to 1 for CDF

- Generate random number $u \in [0, 1]$
- Calculate your $x = G^{-1}(u)$
- Accept x with probability $P = f(x)/g(x)$

instead of using constant f_{max} *we are using* $f_{max}(x) \equiv g(x)$

Quasi-elastic scattering

Building a generator step by step

Quasi-elastic scattering on a free nucleon

Llewellyn-Smith formula

$$\frac{d\sigma}{d|q^2|}\begin{pmatrix}\nu_l + n \to l^- + p \\ \bar{\nu}_l + p \to l^+ + n\end{pmatrix} = \frac{M^2 G_F^2 \cos\theta_C}{8\pi E_\nu^2}\left[A(q^2) \mp B(q^2)\frac{(s-u)}{M^2} + C(q^2)\frac{(s-u)^2}{M^4}\right]$$

Notation

- Constants: M - nucleon mass, G_F - Fermi constant, θ_C - Cabibbo angle,
- $q^2 = (k-k')^2 = (p'-p)^2$ - four-momentum squared, where k, k', p, p' are four-momenta of initial and final lepton, initial and final nucleon
- E_ν - neutrino energy
- $s = (k+k')^2$ and $u = (k-p')^2$ - Mandelstam variables

Quasi-elastic scattering on a free nucleon

Llewellyn-Smith formula

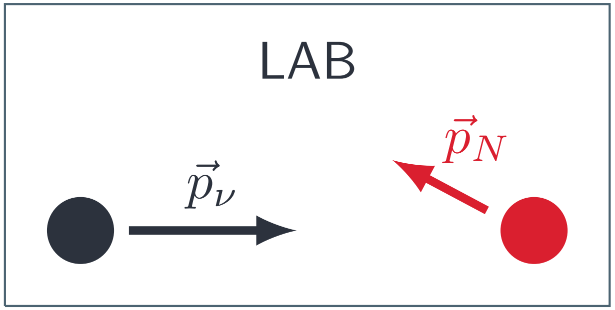
$$\frac{d\sigma}{d|q^2|}\begin{pmatrix}\nu_l + n \to l^- + p \\ \bar{\nu}_l + p \to l^+ + n\end{pmatrix} = \frac{M^2 G_F^2 \cos\theta_C}{8\pi E_\nu^2}\left[A(q^2) \mp B(q^2)\frac{(s-u)}{M^2} + C(q^2)\frac{(s-u)^2}{M^4}\right]$$

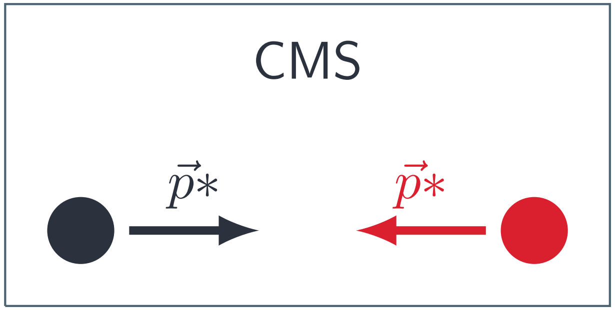
General idea

- Having k and p, generate k' and p'
- Calculate q^2 and $(s-u) = 4ME_\nu + q^2 - m^2$ based on generated kinematics
- Calculate cross section
- Repeat N times and the result is given by:

$$\sigma_{total} \sim \frac{1}{N}\sum_{i=1}^{N}\sigma(q_i^2)$$

Generating kinematics





- Lets consider kinematics in center-of-mass system
- Mandelstam s is invariant under Lorentz transformation

$$s = (k+p)^2 = (E+E_p)^2 - (\vec{k}+\vec{p})^2 = (E^* + E_p^*)^2$$

- $\sqrt{s}$ is the total energy in CMS

$$\sqrt{s} = E^* + E_p^* = \sqrt{p^{*2} + m^2} + \sqrt{p^{*2} + M^2}$$

- We will use it to calculate $p*$

Generating kinematics

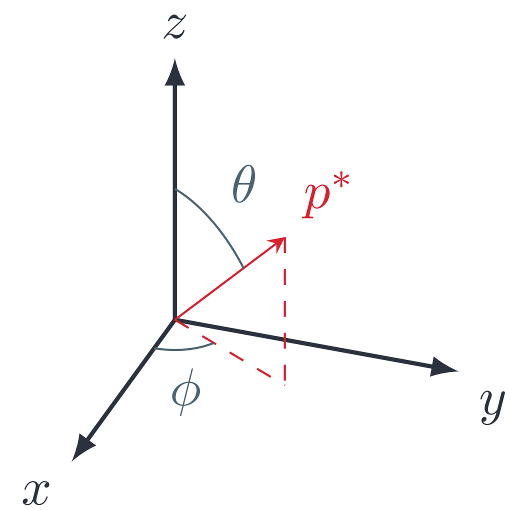
- Lets do some simple algebra:

$$
\begin{aligned}
\sqrt{s} &= E^* + E_p^* = \sqrt{p^{*2} + m^2} + \sqrt{p^{*2} + M^2} \\
\sqrt{s} &= E^* + \sqrt{E^{*2} - m^2 + M^2} \\
s &= E^{*2} + E^{*2} - m^2 + M^2 + 2E^* E_p^* \\
s &= 2E^*(E^* + E_p^*) - m^2 + M^2 \\
s &= 2E^* \sqrt{s} - m^2 + M^2 \\
E^* &= \frac{s + m^2 - M^2}{2\sqrt{s}} \\
E_p^* &= \frac{s + M^2 - m^2}{2\sqrt{s}} \text{ (analogously)}
\end{aligned}
$$

- After more algebra we get:

$$
p^* = \sqrt{E^{*2} - m^2} = \frac{[s - (m - M)^2] \cdot [s - (m + M)^2]}{2\sqrt{s}}
$$

- We use spherical coordinate system to determine momentum direction in CMS:



$$\vec{p^*} = p^* \cdot (\sin\theta\cos\phi, \ \sin\theta\sin\phi, \ \cos\theta)$$

- Generate random angles:

$$\begin{aligned} \phi &= 2\pi \cdot \text{random}[0,1] & \Rightarrow & \quad \sin\phi, \ \cos\phi \\ \cos\theta &= 2 \cdot \text{random}[0,1] - 1 & \Rightarrow & \quad \sin\theta, \ \cos\theta \end{aligned}$$

- All we need to do is to go back to LAB frame

- Lorentz boost in direction $\hat{n} = \frac{\vec{v}}{v}$ of $(t, \vec{r})$:

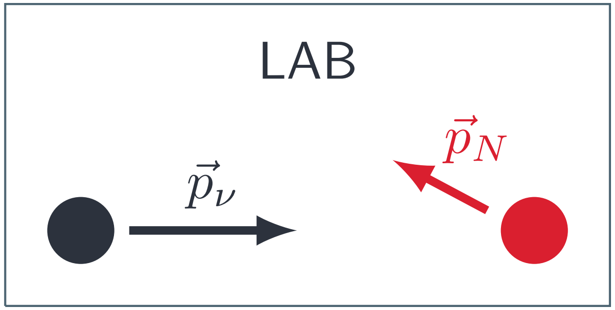
$$\begin{aligned} t' &= \gamma\left(t - v\hat{n}\cdot\vec{r}\right) \\ \vec{r}' &= \vec{r} + (\gamma - 1)(\hat{n}\cdot\vec{r})\hat{n} - \gamma t v\hat{n} \end{aligned}$$

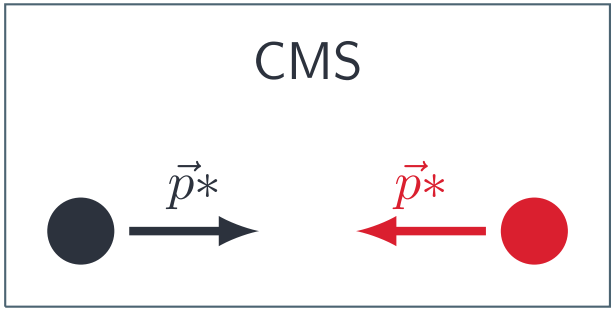
- In our case

$$\vec{v} = \frac{\vec{p}_\nu + \vec{p}_N}{E_\nu + E_N}$$

- Boost from LAB to CMS in $\vec{v}$ direction

- Boost from CMS to LAB in $-\vec{v}$ direction





Calculating cross section

Llewellyn-Smith formula

$$\frac{d\sigma}{d|q^2|}\begin{pmatrix}\nu_l + n \to l^- + p \\ \bar{\nu}_l + p \to l^+ + n\end{pmatrix} = \frac{M^2 G_F^2 \cos\theta_C}{8\pi E_\nu^2}\left[A(q^2) \mp B(q^2)\frac{(s-u)}{M^2} + C(q^2)\frac{(s-u)^2}{M^4}\right]$$

Calculation

- Once we have p' and k' in LAB frame we can calculate q^2 and $(s-u)$
- Once we have q^2 we can calculate $A(q^2)$, $B(q^2)$, $C(q^2)$
- We have everything to calculate cross section
- Do we? Or maybe we are still missing something?

Calculating cross section

Llewellyn-Smith formula

$$\frac{d\sigma}{d|q^2|}\begin{pmatrix}\nu_l + n \to l^- + p \\ \bar{\nu}_l + p \to l^+ + n\end{pmatrix} = \frac{M^2 G_F^2 \cos\theta_C}{8\pi E_\nu^2}\left[A(q^2) \mp B(q^2)\frac{(s-u)}{M^2} + C(q^2)\frac{(s-u)^2}{M^4}\right]$$

Calculation

- Once we have p' and k' in LAB frame we can calculate q^2 and $(s-u)$
- Once we have q^2 we can calculate $A(q^2)$, $B(q^2)$, $C(q^2)$
- We have everything to calculate cross section
- Do we? Or maybe we are still missing something?

We change the variable we integrate over! We need Jacobian!

Calculating cross section

- Express q^2 in terms of angle:

$$q^2 = (k-k')^2 = m^2 - 2kk' = m^2 - 2EE' + 2|\vec{k}||\vec{k}'|\cos\theta$$

- Thus, the Jacobian is given by:

$$dq^2 = 2|\vec{k}||\vec{k}'|d(\cos\theta)$$

Note: must be calculated in CMS

- Total cross section is given by:

$$\sigma = \int_{-1}^{1} \frac{M^2 G_F^2 \cos\theta_C}{8\pi E_\nu^2}\left[A(q^2) \mp B(q^2)\frac{(s-u)}{M^2} + C(q^2)\frac{(s-u)^2}{M^4}\right] 2|\vec{k}||\vec{k}'|d\cos\theta$$

$$\sigma_{MC} = \frac{2}{N}\sum_{i=1}^{N} \frac{M^2 G_F^2 \cos\theta_C}{8\pi E_\nu^2}\left[A(q_i^2) \mp B(q_i^2)\frac{(s_i-u_i)}{M^2} + C(q_i^2)\frac{(s_i-u_i)^2}{M^4}\right] 2|\vec{k}_i||\vec{k}_i'|$$

Calculating cross section

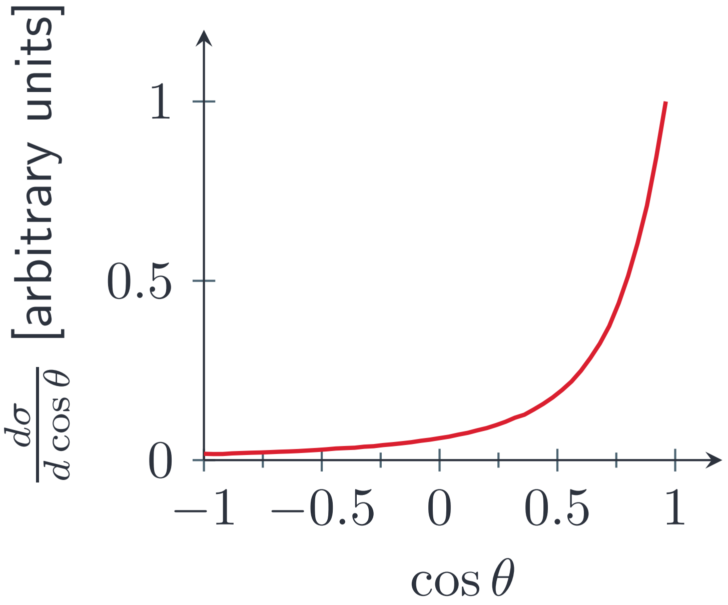
- We want to avoid any sharp peaks
- They affect our efficiency and accuracy
- Lets change variable once again:

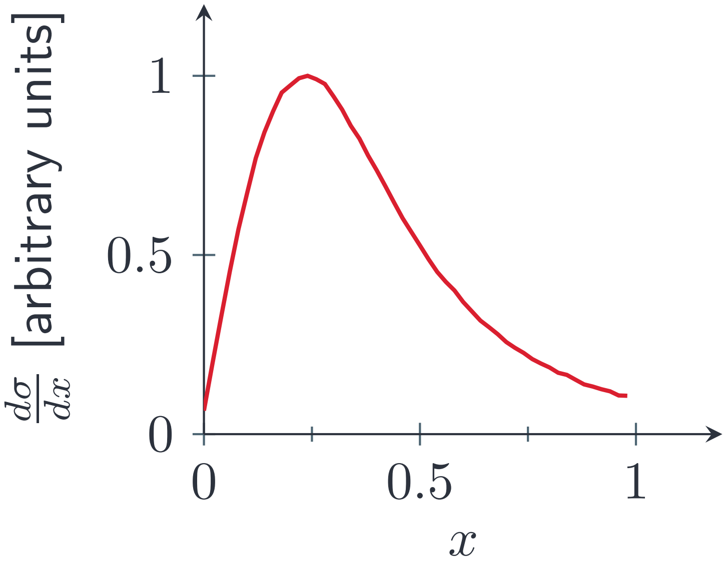
$$\cos\theta = 1 - 2x^2$$

where $x \in [0, 1]$

- Note extra Jacobian and new integration limits

$$2\int_{-1}^{1} d(\cos\theta) \rightarrow \int_{1}^{0} dx(-4x) \rightarrow \int_{0}^{1} 4x dx$$





Calculating cross section

- Finally, the cross section is given by:

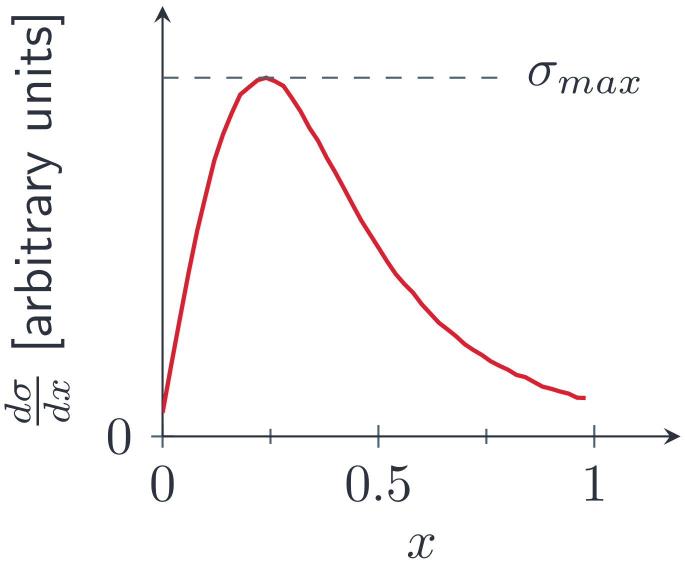
$$
\begin{aligned}
\sigma &= \int_0^1 \frac{M^2 G_F^2 \cos\theta_C}{8\pi E_\nu^2}\left[A(q^2) \mp B(q^2)\frac{(s-u)}{M^2} + C(q^2)\frac{(s-u)^2}{M^4}\right] 2|\vec{k}||\vec{k}'|4x dx \\
\sigma_{MC} &= \frac{1}{N}\sum_{i=1}^{N} \frac{M^2 G_F^2 \cos\theta_C}{8\pi E_\nu^2}\left[A(q_i^2) \mp B(q_i^2)\frac{(s_i-u_i)}{M^2} + C(q_i^2)\frac{(s_i-u_i)^2}{M^4}\right] 2|\vec{k}_i||\vec{k}_i'|4x
\end{aligned}
$$

- In conclusion: do some kinematics and some boosts between CMS and LAB, change integration variable several times... and you are ready to calculate total cross section
- Now we need to generate some events. We want them to be distributed according to our cross section formula.

Generating events

- Generate $x \in [0:1]$
- Do kinematics

$$\begin{aligned} x &\rightarrow \cos\theta \\ \cos\theta &\rightarrow k'^*, p'^* \\ k'^*, p'^* &\rightarrow k', p' \\ &\vdots \end{aligned}$$



- Calculate cross section σ
- Accept an event with the probability given by

$$P = \frac{\sigma}{\sigma_{max}}$$

- And you almost have you MC neutrino-event generator, just a few more steps...

A few more steps

- add other dynamics: resonance pion production, deep inelastic scattering...
- add support for nucleus as a target
- if you have nucleus add some two-body current interactions
- if you have nucleus add some nuclear effects: Pauli blocking, final state interactions, formation zone...
- add support for neutrino beam
- add support for detector geometry
- add some interface to set up simulations parameters and saving the output
- and your MC is done!