Monte Carlo - Basics

Prof. Dr. Narayan Prasad Adhikari
Central Department of Physics
Tribhuvan University Kirtipur, Kathmandu, Nepal

March 15, 2024

Markov Chain and Master Equation

- Random variables
- Concept of errors
- Estimation of errors
- Markov Chain
- Master Equation

Random variables

- Consider an elementary event with a countable set of random outcomes, $A_1, A_2, ...A_k$ (e.g. you can consider a rolling dice OR a set of "Khodkhode".)
- You are data scientist so you need to consider this event occurring repeatedly say N times such that $N >>> 1$ and we count how often the outcome A_k is observed (N_k).
- The probabilities p_k for outcome A_k is

$$p_k = \lim_{N \to \infty} \left(\frac{N_k}{N} \right) \tag{1}$$

with $\sum_k p_k = 1$.
Obviously $0 \leq p_k \leq 1$
You are familiar with conditional probability $P(j/i)$, average of any outcomes of such random events x_i, its variances and so on.

Statistical errors

- Suppose the quantity A is distributed according to a Gaussian with mean value $\langle A \rangle$ and width σ. We consider n statistically independent observations $\{A_i\}$ of this quantity A.

- An unbiased estimator of the mean $\langle A \rangle$ of this distribution is

$$\overline{A} = \frac{1}{n} \sum_{i=1}^{n} A_i \tag{2}$$

and the standard error of this estimate is

$$\text{error} = \frac{\sigma}{\sqrt{n}} \tag{3}$$

Statistical errors

- The variance is obtained from mean square deviation

$$\delta\bar{A}^2 = \frac{1}{n}\sum_{i=1}^{n}\left(\delta A_i\right)^2 = \bar{A^2} - \left(\bar{A}\right)^2 \tag{4}$$

The expectation value of this quantity is easily related to $\sigma^2 = \langle A^2\rangle - \langle A\rangle^2$ as

$$\langle\delta\bar{A}^2\rangle = \sigma^2(1 - 1/n) \tag{5}$$

$$\therefore \text{error} = \sqrt{\frac{\delta\bar{A}^2}{(n-1)}} = \sqrt{\frac{\sum_{i=1}^{n}\left(\delta A_i\right)^2}{(n(n-1))}} \tag{6}$$

Ingredients of MC

- As mentioned before there are at least four ingredients which are crucial in order to understand the basic MC strategy. (i) Random variables
  (ii) Probability distribution functions (PDF),
  (iii) Moments of a PDF (iv) and pertinent variance σ

Random Variables

- Let us first demistify the somewhat obscure concept of a random variable. The example we choose is the classic one, the tossing of two dice, its outcome and the corresponding probability. In principle, we could imagine being able to determine exactly the motion of the two dice, and with given initial conditions determine the outcome of the tossing. Alas, we are not capable of pursuing this ideal scheme. However, it does not mean that we do not have a certain knowledge of the outcome. This partial knowledge is given by the probablity of obtaining a certain number when tossing the dice. To be more precise, the tossing of the dice yields the following possible values

$$[\,2,3,4,5,6,7,8,9,10,11,12.] \tag{7}$$

Random Variables

- These values are called the domain. To this domain we have the corresponding probabilities

$$[\ 1/36, 2/36, 3/36, 4/36, 5/36, 6/36, 5/36, 4/36, 3/36, 2/36, 1/36] \tag{8}$$

- These values are called the domain. To this domain we have the corresponding probabilities

- The numbers in the domain are the outcomes of the physical process tossing the dice. We cannot tell beforehand whether the outcome is 3 or 5 or any other number in this domain. This defines the randomness of the outcome, or unexpectedness or any other synonimous word which encompasses the uncertitude of the final outcome.

Random Variables

- The only thing we can tell beforehand is that say the outcome 2 has a certain probability. If our favorite hobby is to spend an hour every evening throwing dice and registering the sequence of outcomes, we will note that the numbers in the above domain

$$[\,2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12.] \tag{9}$$

appear in a random order.
after (say) 11 throws the results may look like

$$[\,10, 8, 6, 3, 6, 9, 11, 8, 12, 4, 5] \tag{10}$$

- Eleven new attempts may results in a totally different sequence of numbers and so forth. Repeating this exercise the next evening, will most likely never give you the same sequences. Thus, we say that the outcome of this hobby of ours is truely random.

Random Variables

- *Random variables are hence characterized by a domain which contains all possible values that the random value may take. This domain has a corresponding PDF.*

MC Illustration - Integration

- Consider an integration

$$I = \int_0^1 f(x)dx \simeq \sum_{i=1}^{N} w_i f(x_i) \tag{11}$$

where w_i are the weights determined by specific integration methods like Trapeziod, Simpson etc. In the crudest approach here in MC integration we set up $w_i = 1$ then above eq becomes

$$I = \int_0^1 f(x)dx \simeq \frac{1}{N}\sum_{i=1}^{N} f(x_i) \tag{12}$$

Now introduce the concept of the average of the function f for a given PDF $p(x)$ as

$$\langle f \rangle = \frac{1}{N}\sum_{i=1}^{N} f(x_i)p(x_i) \tag{13}$$

MC Illustration - Integration

Now identify $p(x_i) = 1$ with the uniform distribution when $x \in [0, 1)$ and zero for all other values of x. Then

$$I = \int_0^1 f(x)dx \simeq \frac{1}{N}\sum_{i=1}^{N} f(x_i) \simeq \langle f \rangle \tag{14}$$

Similarly the variance (which is also important in MC methods) is

$$\sigma_f^2 = \frac{1}{N}\sum_{i=1}^{N} \left(f(x_i) - \langle f \rangle\right)^2 p(x_i) \tag{15}$$

After inserting value of $p(x_i)$ we get

$$\sigma_f^2 = \frac{1}{N}\sum_{i=1}^{N} f^2(x_i) - \left(\langle f \rangle\right)^2 \tag{16}$$

MC Illustration - Integration:Algorithm

- Choose the number of Monte Carlo samples N.
- Perform a loop over N and for each step generate a a random number x_i in the interval [0, 1] through a call to a random number generator. Translate the random numbers to other required interval if it needs.
- Use this number to evaluate $f(x_i)$.
- Evaluate the contributions to the mean value and the standard deviation for each loop.
- After N samples calculate the final mean value and the standard deviation.

MC Illustration - Integration

- As an example evaluate following by MC method:

$$I = \int_0^1 \exp(x) dx \tag{17}$$

and

$$I = \int_1^3 \exp(x) dx \tag{18}$$

Also Compare the final results with the correct and hence estimate the errors.

MC Illustration - Integration

```
import random
import numpy as np
import math
a = 0.
b = 1.
integral = 0.0
i=0
while i<1000:
x=random.random()
integral += math.exp(x)
i=i+1
ans=integral*(b-a)/float(N)
print ("The value calculated by monte carlo integration is {
}".format(ans))
```

HW: You also estimate it following above algorithm. Further estimate integral for different set of random numbers and hence estimate σ_N.

MC Illustration - Estimate π

1. Initialize circle points, square points and interval to 0.
2. Generate random point x.
3. Generate random point y.
4. Calculate d = x*x + y*y.
5. If d <= 1, increment circle points.
6. Increment square points.
7. Increment interval.
8. If increment < NOOFITERATIONS, repeat from 2.
9. Calculate pi = 4*(circle points/square points).
10. Terminate.

Also follow the discussion in my lecture

HW: Estimate π

1. Estimate value of π also from

$$\pi = \int_0^1 4 \frac{dx}{1+x^2} \tag{19}$$

2. What we did in previous slide was to estimate value of π from area of a circle in 2 dimensions. Can you think of similar methods for higher dimensions? You may need volume of a hypersphere of radius R in n dimensions:

$$V_n(R) = \frac{\pi^{n/2}}{\left(\frac{n}{2}\right)!} R^n \tag{20}$$

what you did in 2D is ust a special case of above equation in n=2.

Concept of Importance Sampling

- Till now we discussed about 'simple sampling' of MC
- In principle, MC integrations and other simulations can be performed using the simple sampling techniques we discussed till now. Unfortunately most of the saples produced in this fashion will contribute relatively little to the equilibrium (time independent) averages and more sophisticated methods are required if we are to obtain results of sufficient accuracy to be useful.

- One of such a methods is "Importance Sampling". For this we need to discuss change of variables.

Concept of Importance Sampling

- With improvements we think of a smaller variance and the need for fewer Monte Carlo samples, although each new Monte Carlo sample will most likely be more times consuming than corresponding ones of the brute force method (Simple sampling). For this we consider two topics.

- The first topic deals with change of variables, and is linked to the cumulative function $P(x)$ of a PDF $p(x)$. Obviously, not all integration limits go from x $= 0$ to x $= 1$, rather, in DATA Science we are often confronted with integration domains like $x \in [0, \infty]$ or $x \in [-\infty, \infty]$ etc. Since all random number generators give numbers in the interval $x \in [0, 1]$, we need a mapping from this integration interval to the explicit one under consideration.

Concept of Importance Sampling

- The next topic deals with the shape of the integrand itself. Let us for the sake of simplicity just assume that the integration domain is again from x $= 0$ to x $= 1$. If the function to be integrated $f(x)$ has sharp peaks and is zero or small for many values of $x \in [0, 1]$, most samples of $f(x)$ give contributions to the integral I which are negligible. As a consequence we need many N samples to have a sufficient accuracy in the region where $f(x)$ is peaked. What do we do then? We try to find a new PDF $p(x)$ chosen so as to match $f(x)$ in order to render the integrand smooth. The new PDF $p(x)$ has in turn an x domain which most likely has to be mapped from the domain of the uniform distribution.

Importance Sampling -Change of variables

- Consider uniform distribution

$$\begin{aligned} p(x)dx =& dx \text{ (for 0} \leq \text{x} \leq \text{1)} \\ =& 0 \text{ else} \end{aligned} \tag{21}$$

with $p(x) = 1$ and satisfying

$$\int_{-infty}^{\infty} p(x)dx = 1 \tag{22}$$

All random number generators provided in the program library generate numbers in this domain. When we attempt a transformation to a new variable $x \rightarrow y$ we have to conserve the probability

$$p(y)dy = p(x)dx \tag{23}$$

which for the uniform distribution implies

$$p(y)dy = dx \tag{24}$$

Importance Sampling -Change of variables

Let us assume that $p(y)$ is a PDF different from the uniform PDF $p(x) = 1$ with $x \in [0, 1]$. If we integrate the last expression we arrive at

$$x(y) = \int_0^y p(y')dy' \tag{25}$$

which is nothing but the cumulative distribution of $p(y)$, i.e.

$$x(y) = P(y) = \int_0^y p(y')dy' \tag{26}$$

This is an important result which has consequences for eventual improvements over the brute force Monte Carlo.

Change of variables- an example

Suppose we have the general uniform distribution

$$\begin{aligned} p(y)dy =& \frac{dy}{b-a} \text{ (for a} \leq \text{ y} \leq \text{ b)} \\ =& 0 \text{ else} \end{aligned} \tag{27}$$

If we wish to relate this distribution to the one in the interval $x \in [0, 1]$ we have

$$p(y)dy = \frac{dy}{b-a} \text{ (for a} \leq \text{ y} \leq \text{ b)} = dx \tag{28}$$

and integrating we obtain the cumulative function

$$x(y) = \int_a^y \frac{dy'}{b-a} \tag{29}$$

yielding

$$y = a + (b-a)x \tag{30}$$

Importance Sampling

- With the aid of the above variable transformations we address now one of the most widely used approaches to Monte Carlo integration, namely importance sampling. It will be helpful to sample a function which has peak as we need to consider many more sampling points near the peak.

- Let us assume that $p(y)$ is a PDF whose behavior resembles that of a function F defined in a certain interval $[a, b]$. The normalization condition is

$$\int_a^b p(y)dy = 1 \tag{31}$$

We can rewrite our integral as

Importance Sampling

$$I = \int_a^b F(y)dy = \int_a^b p(y)\frac{F(y)}{p(y)}dy \tag{32}$$

Since random numbers are generated for the uniform distribution $p(x)$ with $x \in [0,1]$, we need to perform a change of variables $x \to y$ through

$$x(y) = \int_a^y p(y')dy' \tag{33}$$

where we used

$$p(x)dx = dx = p(y)dy \tag{34}$$

If we can invert $x(y)$, we find $y(x)$ as well. With this change of variables we can express the integral of Eq. 32 as

$$I = \int_a^b p(y)\frac{F(y)}{p(y)}dy = \int_a^b \frac{F(y(x))}{p(y(x))}dx \tag{35}$$

Importance Sampling

meaning that a Monte Carlo evalutaion of the above integral gives

$$\int_a^b \frac{F(y(x))}{p(y(x))}dx = \sum_{i=1}^{N} \frac{F(y(x_i))}{p(y(x_i))} \tag{36}$$

The advantage of such a change of variables in case $p(y)$ follows closely F is that the integrand becomes smooth and we can sample over relevant values for the integrand. It is however not trivial to find such a function p.
The conditions on p which allow us to perform these transformations are

1. p is normalizable and positive definite,
2. it is analytically integrable and
3. the integral is invertible, allowing us thereby to express a new variable in terms of the old one.

Important Note

- The average is over y(x) distribution.

Therefore above equation 35 can be rewritten as

$$I = \int_a^b p(y)\frac{F(y)}{p(y)}dy = \int_a^b p(y)\left\{\frac{F(y)}{p(y)}\right\}dy = E_{p(y)}\left\{\frac{F(y)}{p(y)}\right\} \quad (37)$$

is actually expectation value of

$$\left\{\frac{F(y)}{p(y)}\right\}$$

with distribution $p(y)$.

Please note that the average is over the distribution $p(y)$ not over $p(x)$.

Therefore in the importance sampling integration you first find the p(y) corresponding to $p(x) \in [0, 1]$. Then find average 36 with distribution $p(y)$. See Example below.

Importance Sampling - Examples

(1) Consider the integral

$$I = \int_0^1 \exp(-x^2) dx \tag{38}$$

Evaluate I using (i) brute force (simple sampling) MC with $p(x) = 1$ and (ii) importance sampling with $p(x) = a \exp(-x)$.

Improtant Note: You first write Algorithm in each case then write code in python language following the Hints

(a) Obtain average of $\exp(-x^2)$ for $x \in [0, 1]$

(b) Find its variance too.

These are results of Simple sampling.

for importance sampling:

(c) Find $p(y)$ corresponding to $p(x) = a \exp(-x)$ from equation 33 where a is normalization constant.

(d) Then find the expectation value of $\left[\frac{\exp(-x^2)}{a \exp(-x)}\right]$ with distribution $p(y)$. (e) Find variance also. (f)Compare both errors or variances and comments on your results.

Monte Carlo - More on above examples

- **Solution of above example (Importance sampling):**

$$I = \int_0^1 \exp\left(-x^2\right) dx \tag{39}$$

with chosen pdf (probability distribution function) $p(x) = a\exp(-x)$ such that $x \in (0,1)$ and

$$\int_0^1 p(x)dx = \int_0^1 a\exp\left(-x\right)dx = 1.$$

resulting $a = \frac{e}{e-1}$.

$$\implies p(x) = \frac{\exp\left(-x\right)}{1-\frac{1}{e}} \tag{40}$$

Monte Carlo - More on above examples

Also check whether p(x) fulfills the criteria for pdf. for this we find $\frac{F(0)}{p(0)}$ and $\frac{F(1)}{p(1)}$. They have to be equal.
Now

$$\frac{F(0)}{p(0)} = \frac{e}{e-1} = \frac{F(1)}{p(1)} \tag{41}$$

Since our pdf fulfills the criteria lets find $y(x)$. For this we perform then the change of variables (via the Cumulative distribution function)

$$y(x) = \int_0^x p(x')dx' = \int_0^x dx' \exp(-x') \times \frac{e}{e-1} \tag{42}$$

Monte Carlo - More on above examples

$$\Longrightarrow \; y(x) = \left(\frac{e}{e-1}\right)\left(1 - e^{-x}\right) \tag{43}$$

after solving for x we get;

$$\Longrightarrow \; x = -\ln\left(1 - y(1 - e^{-1})\right) \tag{44}$$

which gives $y = 0$ for $x = 0$ and $y = 1$ for $x = 1$ as required for the property of pdf.

Now we need to find expectation value of

$$\left[\frac{\exp\left(-x^2\right)}{\exp\left(-x\right)}\right]$$

with distribution $y(x)$. That is we need to evaluate

$$\int_0^1 \exp\left(-x^2\right)dx = \left\langle \frac{\exp\left(-y^2(x)\right)}{\exp\left(-y(x)\right)} \right\rangle$$

Monte Carlo - More on above examples

Algorithm:

1. Start
2. import required libraries like random, numpy ..
3. Define n, functions, initialize summ etc
4. start loop over n
5. generate random numbers $x \in (0, 1)$
6. define function $y(x)$ using above formula from x as $y(x) = \left(\frac{e}{e-1}\right)(1 - e^{-x})$
7. Get sum of the function $\frac{\exp(-y^2(x))}{\exp(-y(x))}$
8. close the loop
9. Find the integration value i.e. $\left\langle \frac{\exp(-y^2(x))}{\exp(-y(x))} \right\rangle$
10. You also find variance and hence the error.

The python code is in next page.

PI note that the code contains the simple sampling also. Compare both results.

Monte Carlo - More on above examples

In [3]:
```
import numpy as np
import random
from scipy.stats import norm
#Define the number of MC steps
n=10000

# Standard (simple sampling Monte Carlo
sum=0.0
i=0
summ=0.
while i<n:
    x = random.random()
    g = np.exp(-x**2)
    sum=sum+g
    summ=summ+g*g
    i=i+1
MC=sum/n
std_MC = summ/n-MC**2
print('Standard Monte-Carlo estimate of given function: ' + str(MC))
print('Standard deviation of simple sampling Monte Carlo: ' + str(std_MC))
print(' ')
#Importance sampling
i=0
sum=0.0
summ=0.0
while i<n:
    x = random.random()
    y=(np.exp(1.)/(np.exp(1.0)-1.)*(1-np.exp(-x)))
    f = np.exp(-y**2)/np.exp(-y)
    sum=sum+f
    summ=summ+f*f
    i=i+1
meanf=sum/n
MCI=meanf*(1.0-np.exp(-1.0))
std_MCI=summ/n-meanf**2
print('Importance Sampling Monte-Carlo estimate of given function: ' + str(MCI))
print('Standard deviation of Impostance sampling  Monte Carlo: ' + str(std_MCI))
print(' ')
```

```
Standard Monte-Carlo estimate of given function: 0.7469875583049324
Standard deviation of simple sampling Monte Carlo: 0.04042059388000929

Importance Sampling Monte-Carlo estimate of given function: 0.7442631266953907
Standard deviation of Impostance sampling  Monte Carlo: 0.00778395938152987
```

Importance Sampling - Examples

Now compare the variances OR errors due to simple sampling and importance sampling both.
(2) Consider the integral

$$I = \int_0^{\pi} \frac{1}{x^2 + \cos^2 x} dx \tag{45}$$

Evaluate I using importance sampling with $p(x) = a \exp(-x)$ where a is a constant.
Improtant Note: You first write Algorithm then write code in python language following the Algorithm. Can you find the value of a which minimizes the variance.

Importance Sampling - Examples

Evaluate

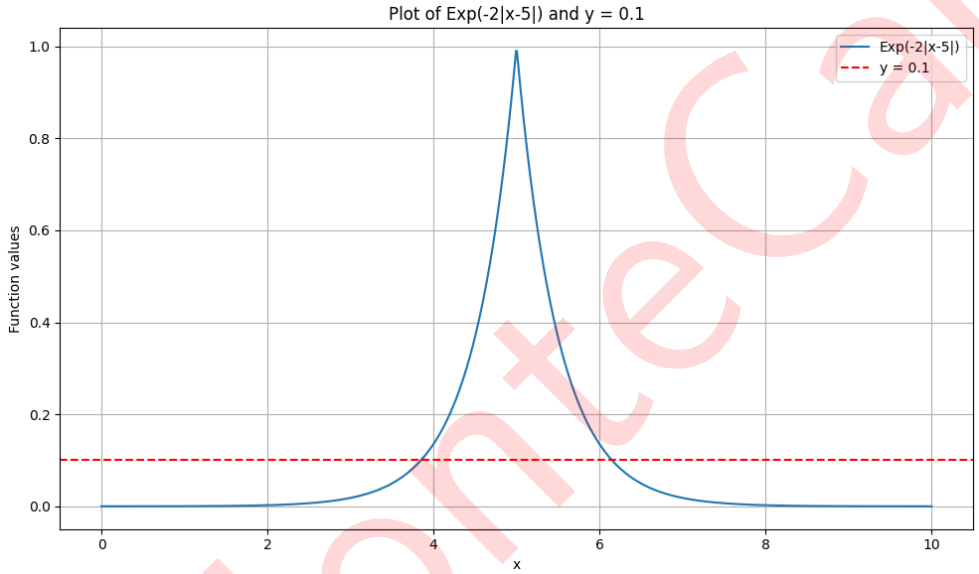
$$\int_0^{10} \exp(-2|x-5|)dx \tag{46}$$

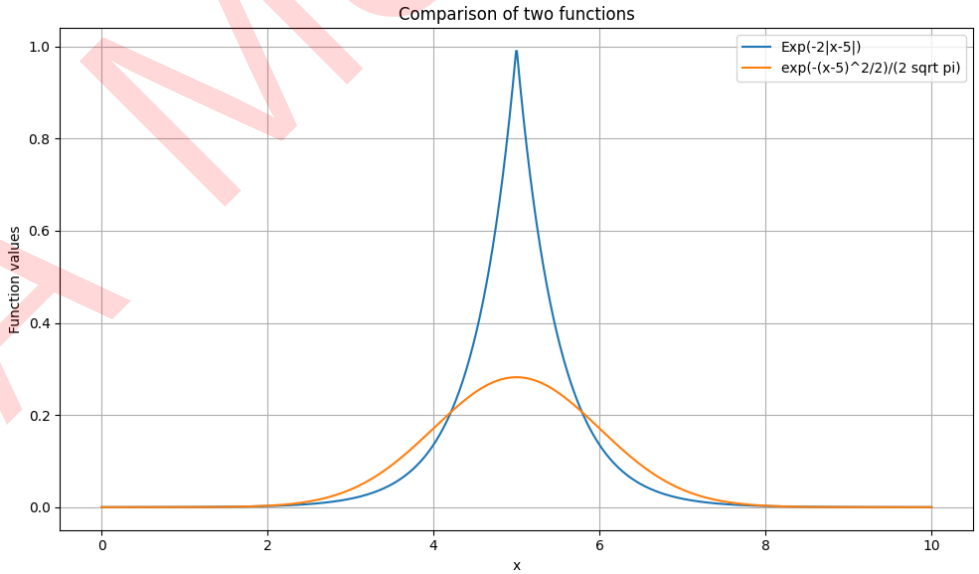
The true value of this integral is about 1. Simple way is Simple Sampling and integrating it.
However the error decreases if we take

$$p(x) = \frac{1}{\sqrt{2\pi}} \exp\left(-(x-5)^2/2\right) \tag{47}$$

You plot the integrand and the density function $p(x)$ defined above in the same plot. Explain why error decreases by choosing $p(x)$.

Importance Sampling - Examples





Monte Carlo Integration - Homework

Consider

$$I = \int_0^1 dx_1 \int_0^1 dx_2 ... \int_0^1 dx_n g(x_1, x_2, ..., x_n) \tag{48}$$

with x_i defined in the interval $[a_i, b_i]$ we would typically need a transformation of variables of the form

$$x_i = a_i + (b_i - a_i) * t_i$$

if we were to use the uniform distribution on the interval $[0, 1]$.
As an example, evaluate

$$I = \int_{-5}^{5} d\mathbf{x}\ d\mathbf{y}\ g(\mathbf{x},\ \mathbf{y}) \tag{49}$$

with

$$g(\mathbf{x},\ \mathbf{y}) = \exp\left(-\mathbf{x}^2 - \mathbf{y}^2 - (\mathbf{x} - \mathbf{y})^2/2\right)$$

Again you write Algorithm and code.

Monte Carlo Acceptance-Rejection method

It is simple and and appealing method after von Neumann. Assume that we are looking at an interval $x \in [a, b]$, this being the domain of the PDF $p(x)$. Suppose also that the largest value our distribution function takes in this interval is M , that is

$$p(x) \leq M \qquad x \in [a, b] \tag{50}$$

Then we generate a random number x from the uniform distribution for $x \in [a, b]$ and a corresponding number s for the uniform distribution between $[0, M]$. If

$$p(x) \geq s \tag{51}$$

we accept the new value of x, else we generate again two new random numbers x and s and perform the test in the latter equation again.

Acceptance-Rejection method: an example

- Actually Acceptance-Rejection sampling is the conceptually simplest way to generate samples of some arbitrary probability function without having to do any transformations.
- No integration, no trickery, you simply trade computational efficiency away to keep everything as simple as possible.
- Consider an example:

$$f(x) = 1.2 - x^4$$

You want to sample points in the given function for $x \in (0, 1)$. Well, if you integrate $f(x)$ between 0 and 1 you get 1.

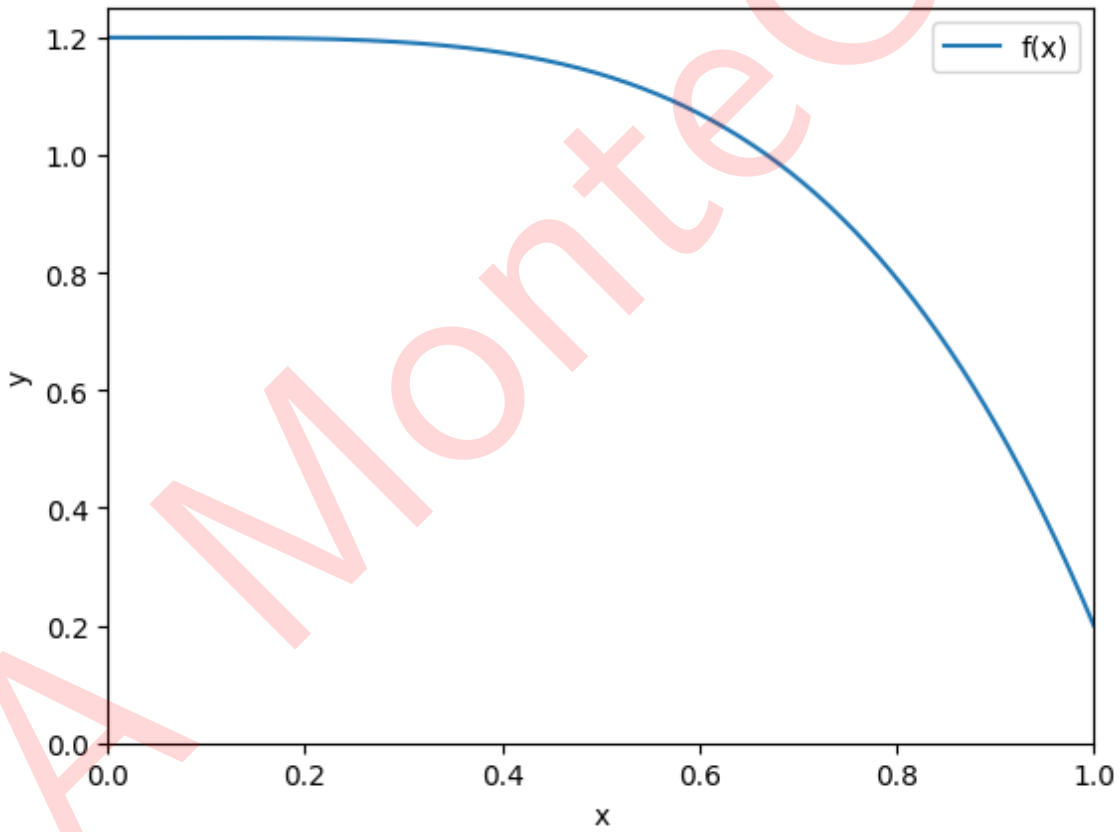
Acceptance-Rejection method: an example

Let's first plot the function just to see how does it look like

```
import numpy as np
import matplotlib.pyplot as plt
def f(x):
  return 1.2 - x**4
xs = np.linspace(0, 1, 1000)
ys = f(xs)

plt.plot(xs, ys, label="f(x)") plt.xlim(0, 1), plt.ylim(0, 1.25),
plt.xlabel("x"), plt.ylabel("y"), plt.legend();
```

Acceptance-Rejection method: an example



Acceptance-Rejection method: an example

- **Algorithm**
- Pick two random numbers. One for x (between 0 and 1), one for y (between 0 and 1.2).
- If the y value we randomly picked is less than f(x), keep it, otherwise go back to above step

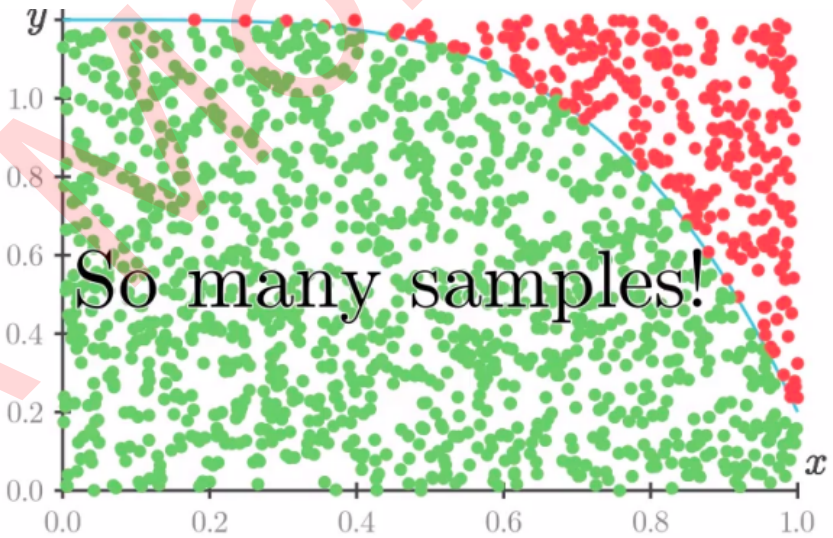


Figure: Green points accepted and red one rejected

Acceptance-Rejection method: an example

- So you can see that the reason this is so straightforward is that we get samples according to the function by simply throwing away the right number of samples when the function has a smaller value.
- In our function, this means if we get a small x value, we'd normally keep the sample (and indeed the distribution is pretty flat for $x < 0.5$), but for values close to $x = 1$, we'd throw them out most of the time

Acceptance-Rejection method: an example

```
def sample(function, xmin=0, xmax=1, ymax=1.2):
  while True:
    x = np.random.uniform(low=xmin, high=xmax)
    y = np.random.uniform(low=0, high=ymax)
    if y < function(x):
      return x

samps = [sample(f) for i in range(10000)]
plt.plot(xs, ys, label="f(x)")
plt.hist(samps, density=True, alpha=0.2, label="Sample
distribution")
plt.xlim(0, 1), plt.ylim(0, 1.4), plt.xlabel("x"), plt.ylabel("y"),
plt.legend();
```

■ Acceptance-Rejection method: an example