DataSci 520

lesson 4

sampling methods

PROFESSIONAL & CONTINUING EDUCATION
UNIVERSITY of WASHINGTON

today's agenda

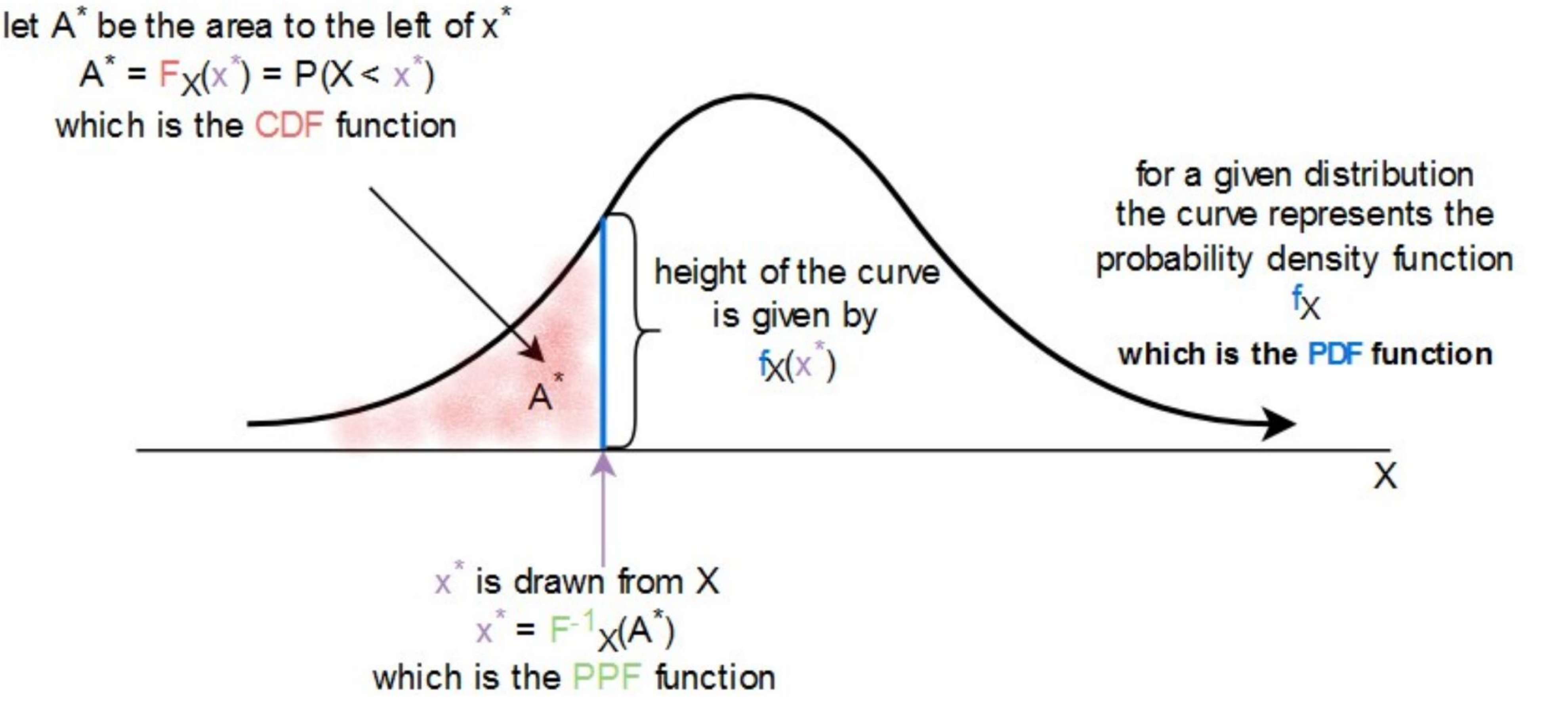
- PDF, CDF and inverse PDF
- common discrete and continuous distributions
- drawing samples from a given distribution
- Monte Carlo methods for estimation

common discrete distributions

- **discrete uniform:** for equally likely outcomes, such as the result of rolling a dice
- **Bernoulli:** for a single binary outcome, such as a single coin flip
- **binomial:** the **number of "successes"** in n independent Bernoulli trials with fixed probability p of success, where "success" is defined by you, such as the number of heads in $n = 20$ coin flips
- **poisson:** used for modeling counts, such as the number of customers visiting a store on any day
- **geometric:** the number of Bernoulli trials (with p fixed) before we see a successful outcome, such as the number of coin flips before we see a tails

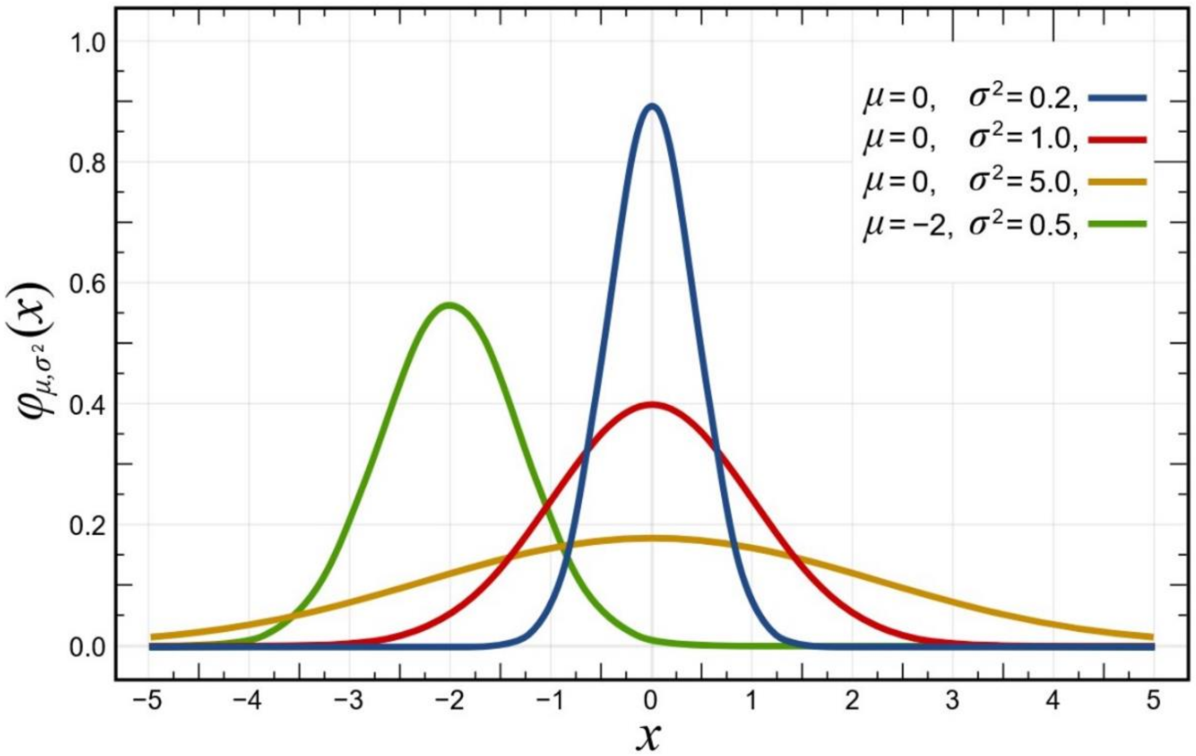
common continuous distributions

- **uniform:** when ranges of data with equal length are equally likely, such as how gas molecules spread in a room
- **normal:** data that is symmetric and bell-shaped, such as people's height, or measurement error if instrument is not **biased**
- **exponential:** heavily right-skewed data, such as lifetime of a light bulb
- **log normal:** skewed data, such as "dwell time" on an online article
- **power law:** for data that appear to follow the **"80-20 rule"**
- **chi-square** is used for a sum of the squares of k independent standard normal random variables, and is used by many **statistical tests**


let A* be the area to the left of x*
A* = FX(x*) = P(X < x*)
which is the CDF function
A*
height of the curve
is given by
fX(x*)
for a given distribution
the curve represents the
probability density function
fX
which is the PDF function
X
x* is drawn from X
x* = F-1X(A*)
which is the PPF function

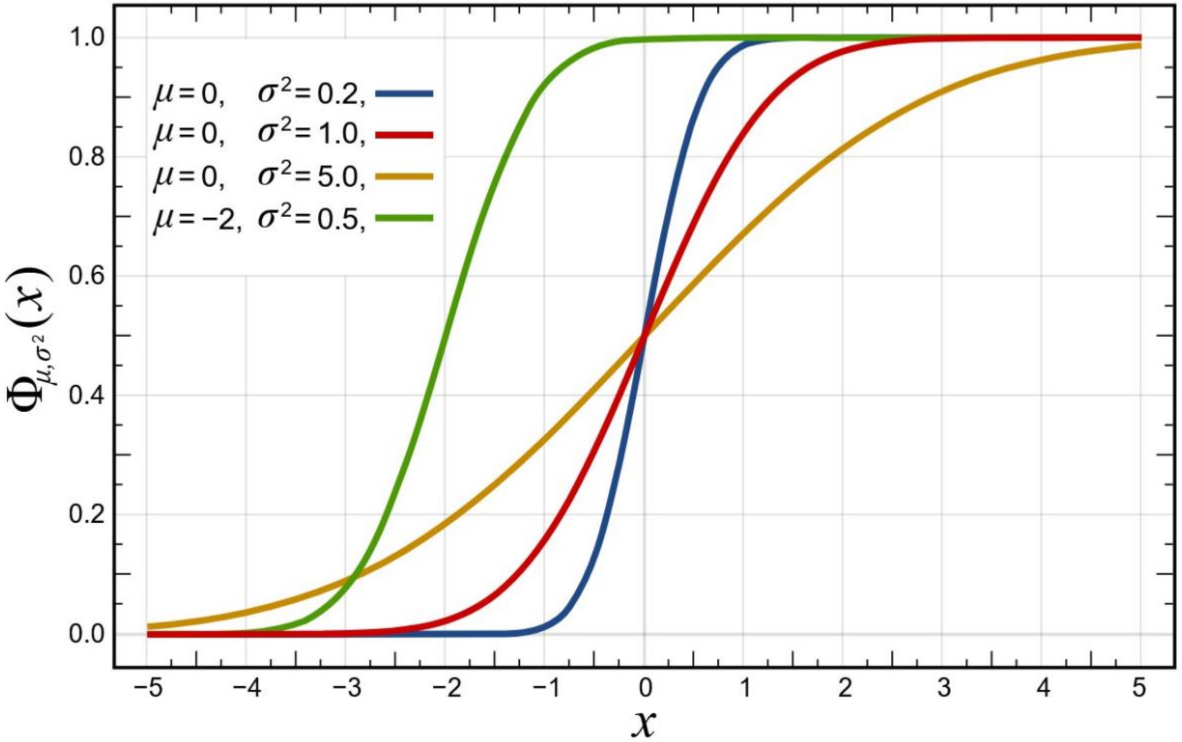
PDF of normal distribution

- the PDF of the **normal distribution** is shown with different means μ and standard deviations σ
- image source: wikipedia.org



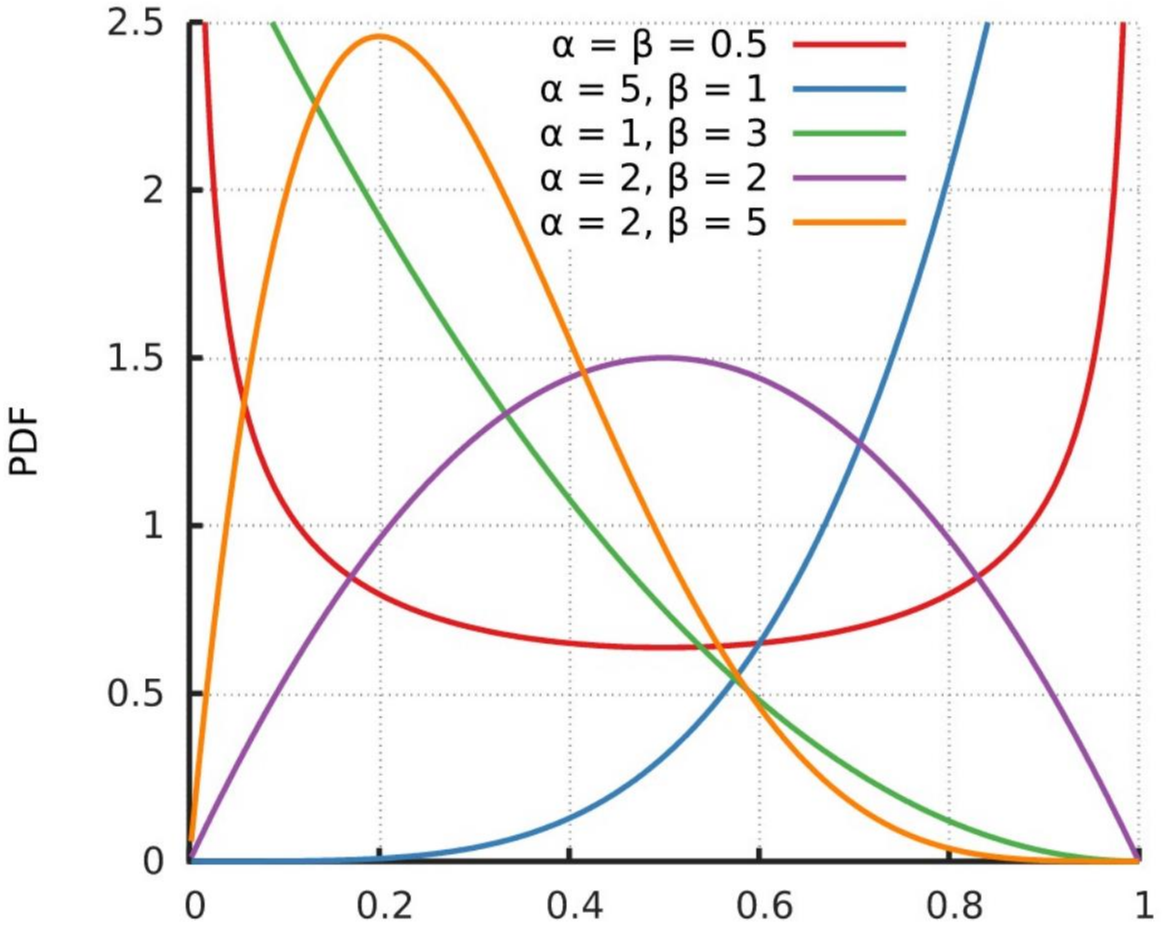
CDF of normal distribution

- the CDF of the **normal distribution** is shown with different means μ and standard deviations σ
- image source: wikipedia.org



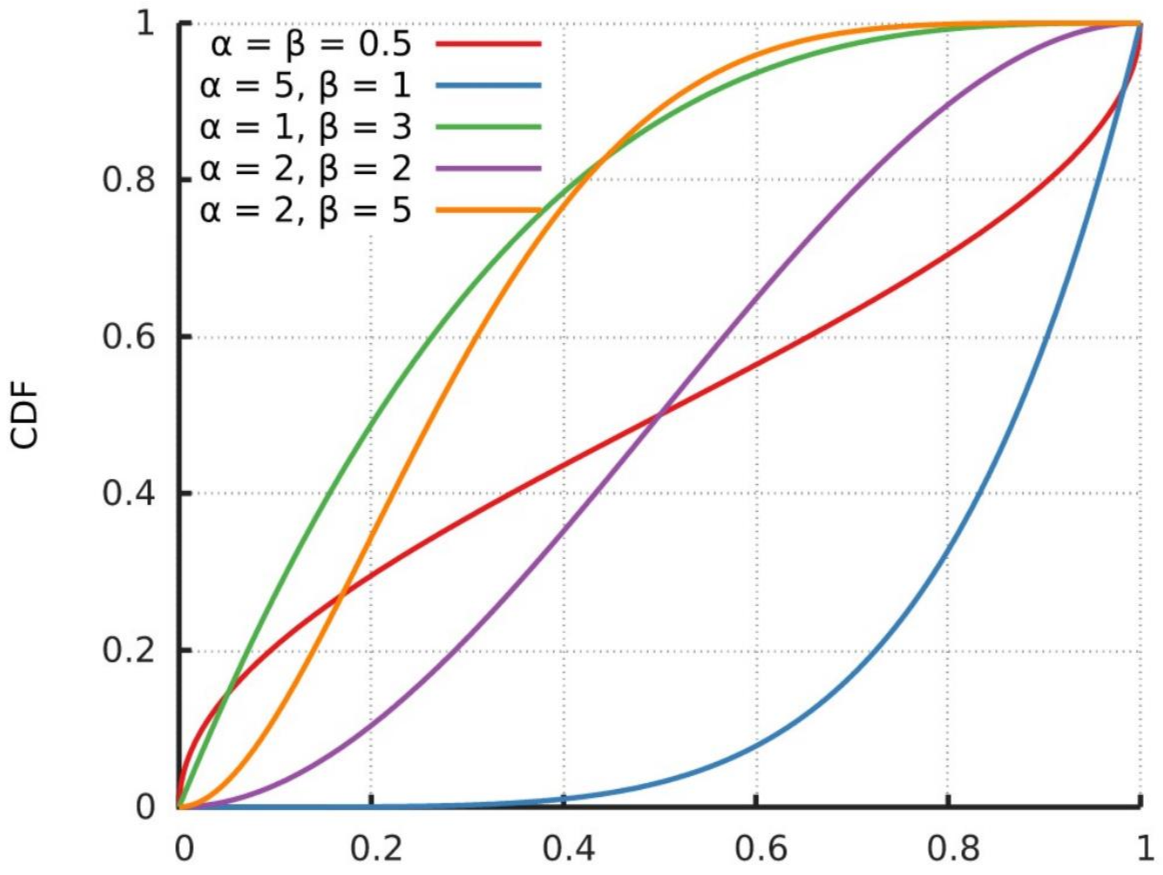
PDF of beta distribution

- the PDF of the **beta distribution** is shown with different parameters α and β
- image source: wikipedia.org



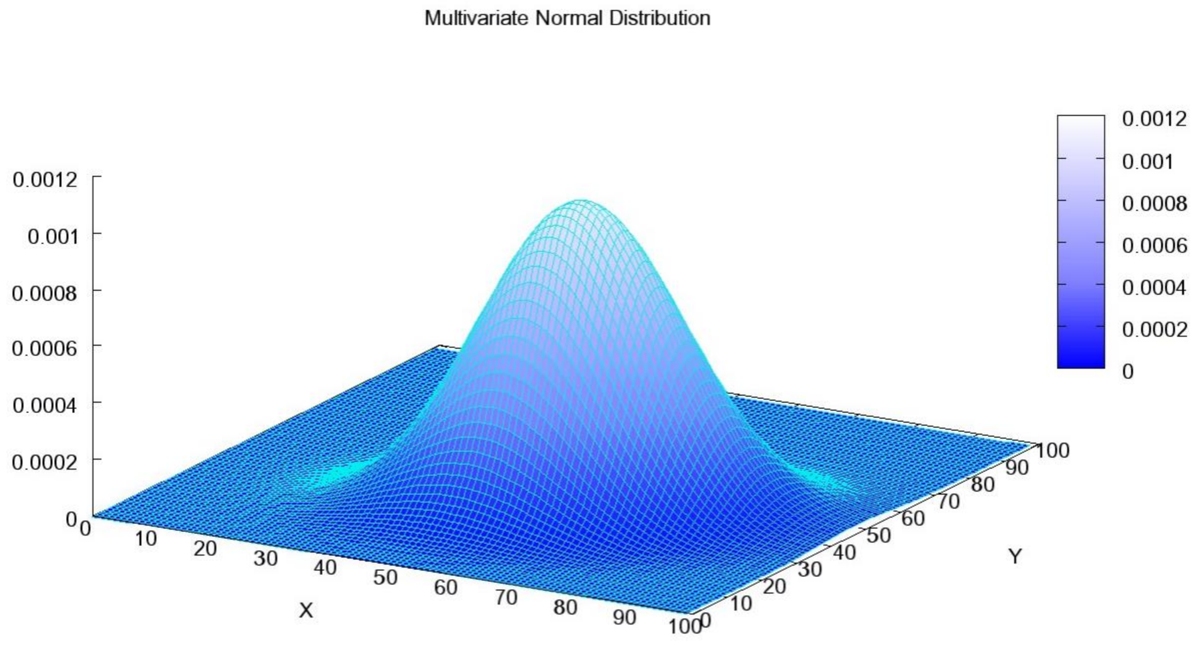
CDF of beta distribution

- the CDF of the **beta distribution** is shown with different parameters α and β
- image source: wikipedia.org



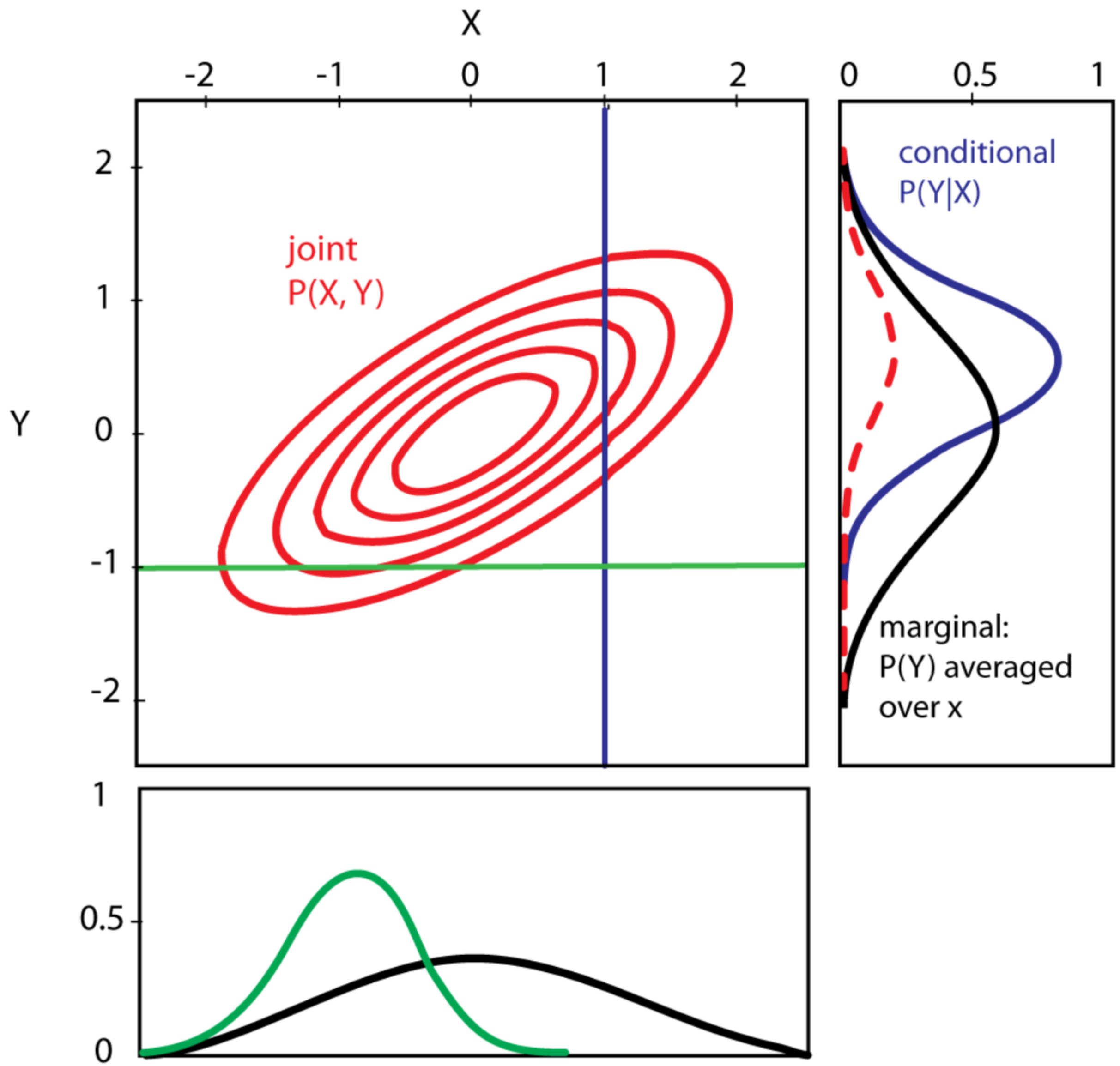
multivariate normal distribution

- the plot shows the probability density function of X and Y together (joint distribution)
- the sample space is 2D: (X, Y)
- the "volume" under the curve adds up to 1
- we can extend this to higher dimensions: (X, Y, Z) etc.
- image source: wikipedia.org



joint density for bivariate normal distribution

- for the multivariate normal distribution, it can be shown that marginal and conditional probabilities also follow a normal distribution
- in general, that it not the case, so the normal distribution is special
- joint and marginals are "fixed", but conditional changes as you slide the blue bar



notebook time

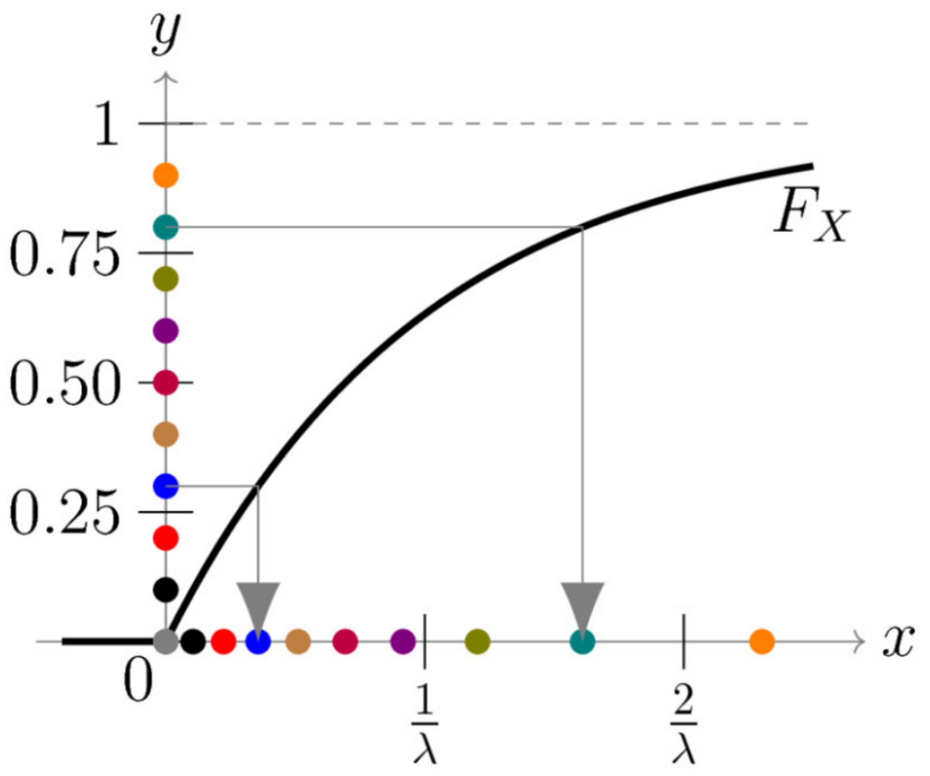
we return to the lecture later

drawing from a given distribution

If we know the functional form F_X of the CDF of X, we can sample from its distribution using **inverse transform sampling**:

- draw a number form the uniform distribution $u \sim U(0,1)$
- we find $F_X^{-1}(u)$ where F_X^{-1} is the PPF (which is the inverse of F_X, the CDF)

This example from wikipedia.org is for the exponential distribution

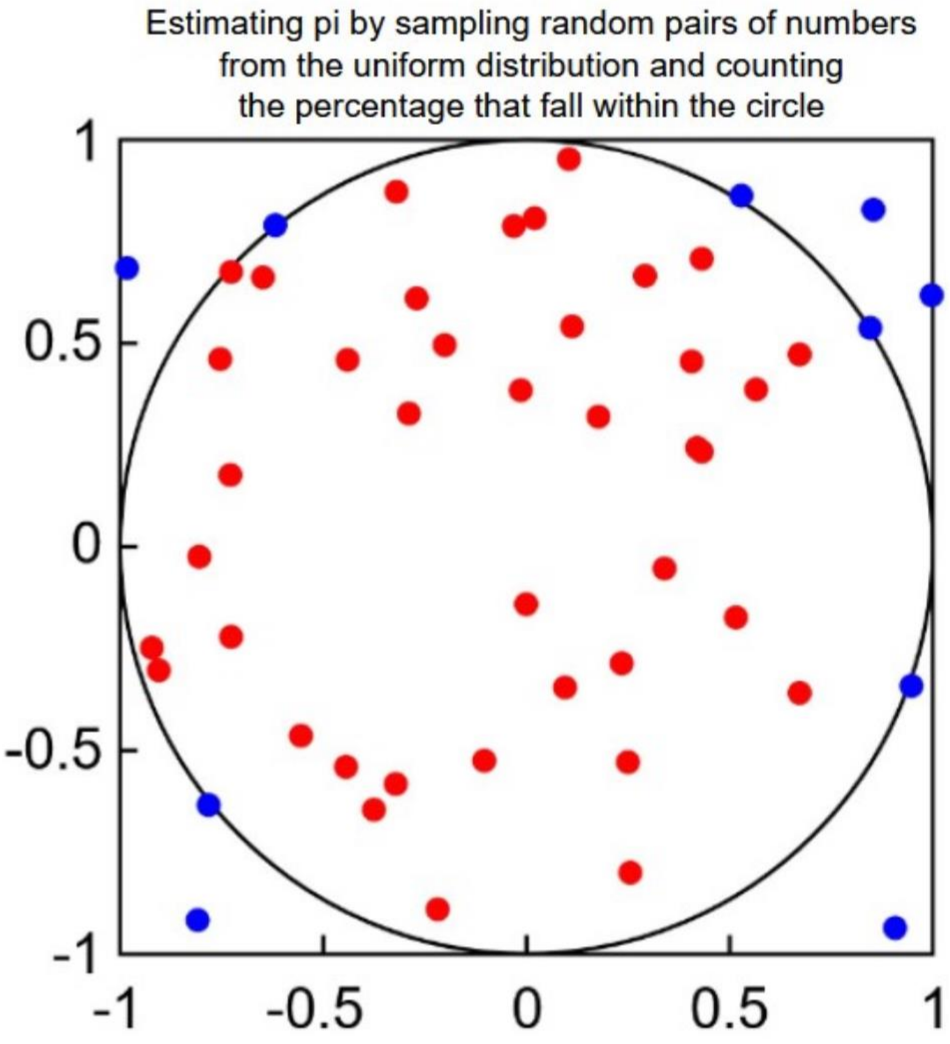


`numpy` and `scipy` functions

- we can **generate** random numbers from a available distributions
  - in `numpy.random`, for example `np.random.binomial` or `np.random.normal`
  - in `scipy.stats`, for example `scipy.stats.norm.rvs` or `scipy.stats.binom.rvs`
- if we want the CDF, PDF and PPF functions, we go to `scipy.stats` for the distribution and call the corresponding method, for example we have
  - for the geometric distribution `scipy.stats.geom.pmf`, `scipy.stats.geom.ppf`, and `scipy.stats.geom.cdf`
  - for the normal distribution `scipy.stats.norm.pdf`, `scipy.stats.norm.ppf`, and `scipy.stats.norm.cdf`
- Monte Carlo methods is all about generating samples!

applications of sampling

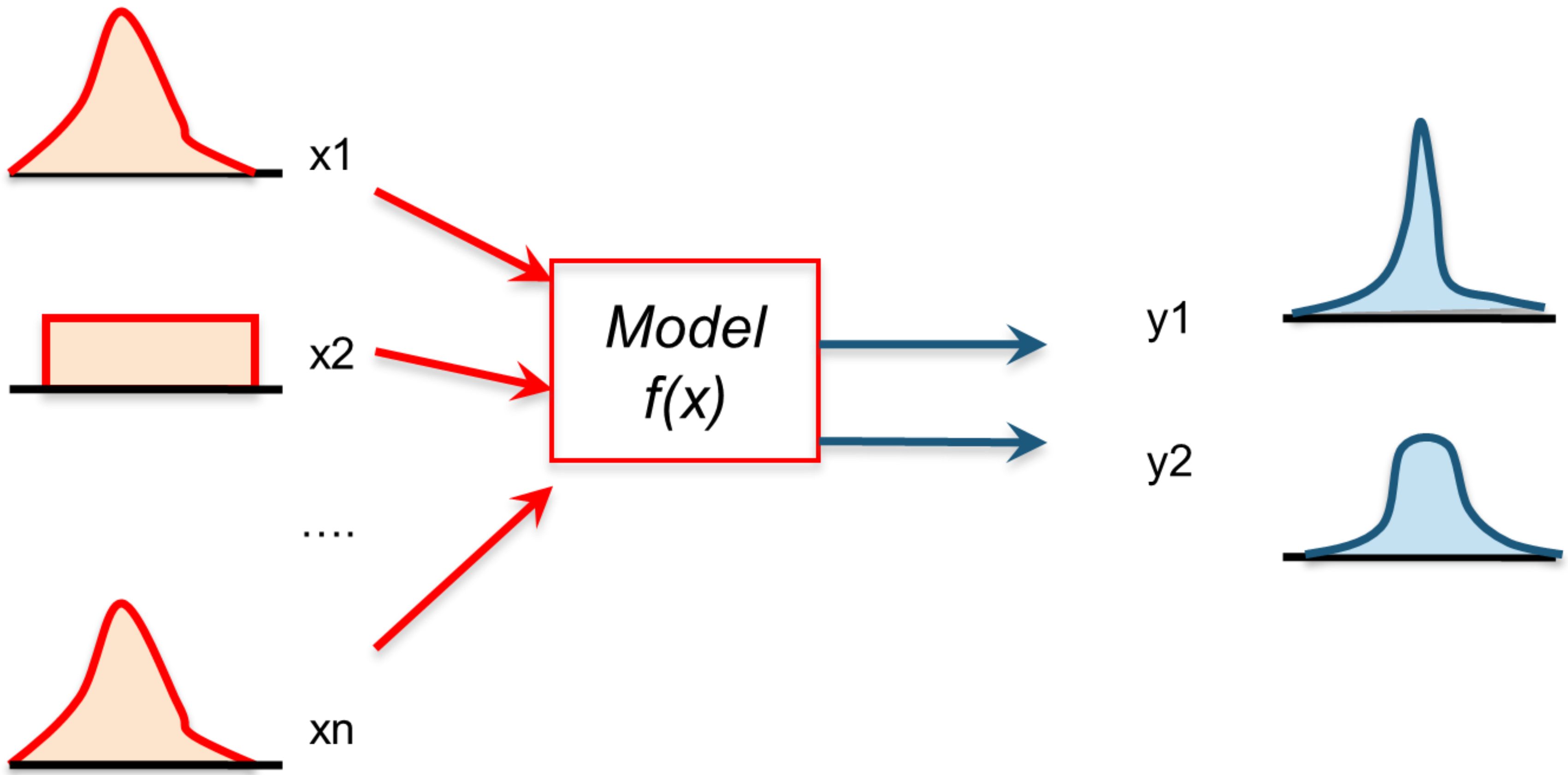
- **Monte Carlo methods** are used in estimation problems that are hard to solve analytically
- sampling can be used to generate fake data that looks similar to real data (e.g. masking data for privacy reasons)
- sampling can be used by machine learning models to generate new examples, such as in **natural language genaration**
- source: wikipedia.org



Monte Carlo Methods

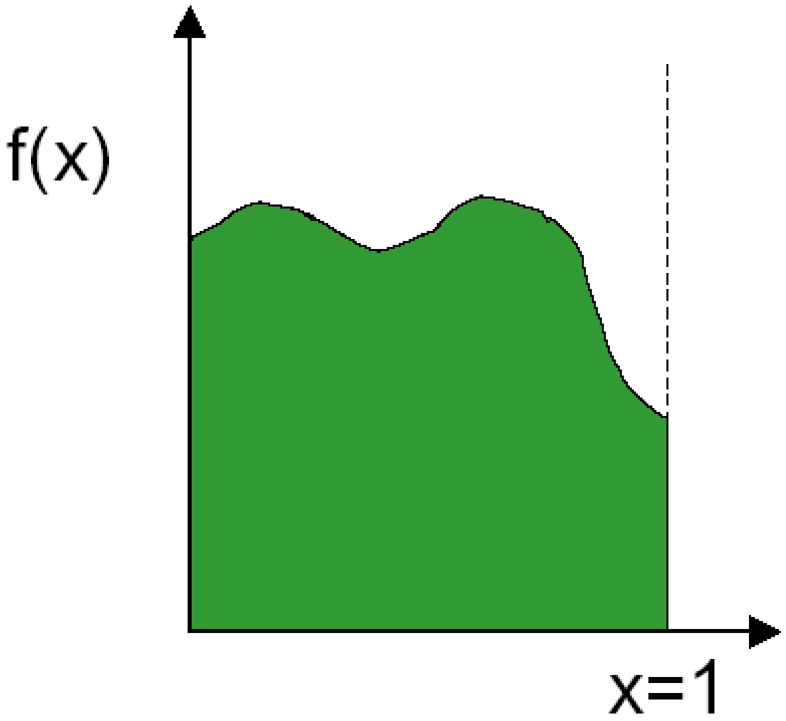
- Monte Carlo is another name for statistical sampling methods of great importance to physics and computer science
- Applications of Monte Carlo Method
  - Evaluating integrals of arbitrary functions of 6+ dimensions
  - Predicting future values of stocks
  - Solving partial differential equations
  - Sharpening satellite images
  - Modeling cell populations
  - Finding approximate solutions to NP-hard problems

Monte Carlo Simulations



Monte Carlo Simulation Example

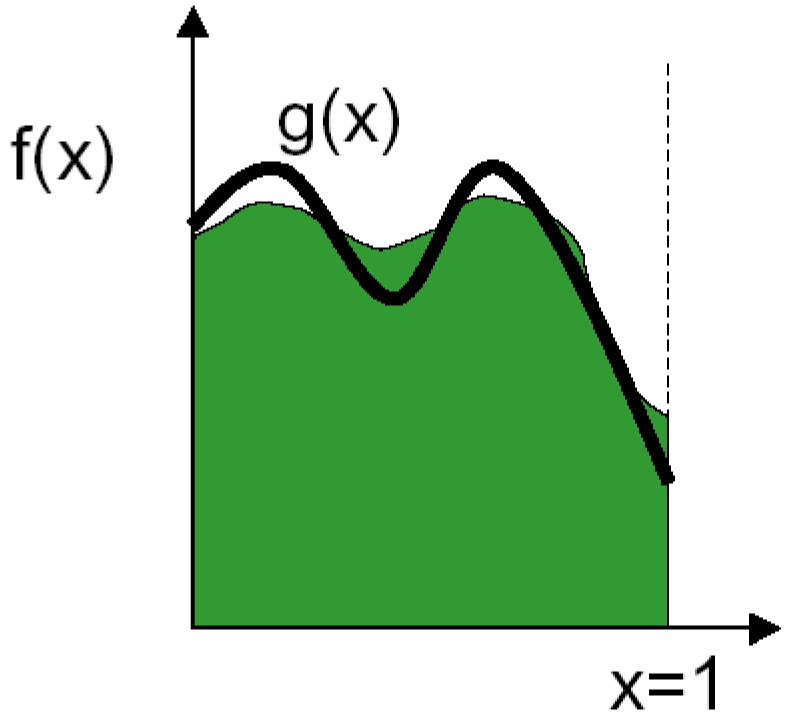
> How to evaluate integral of f(x)?



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

Monte Carlo Simulation Example

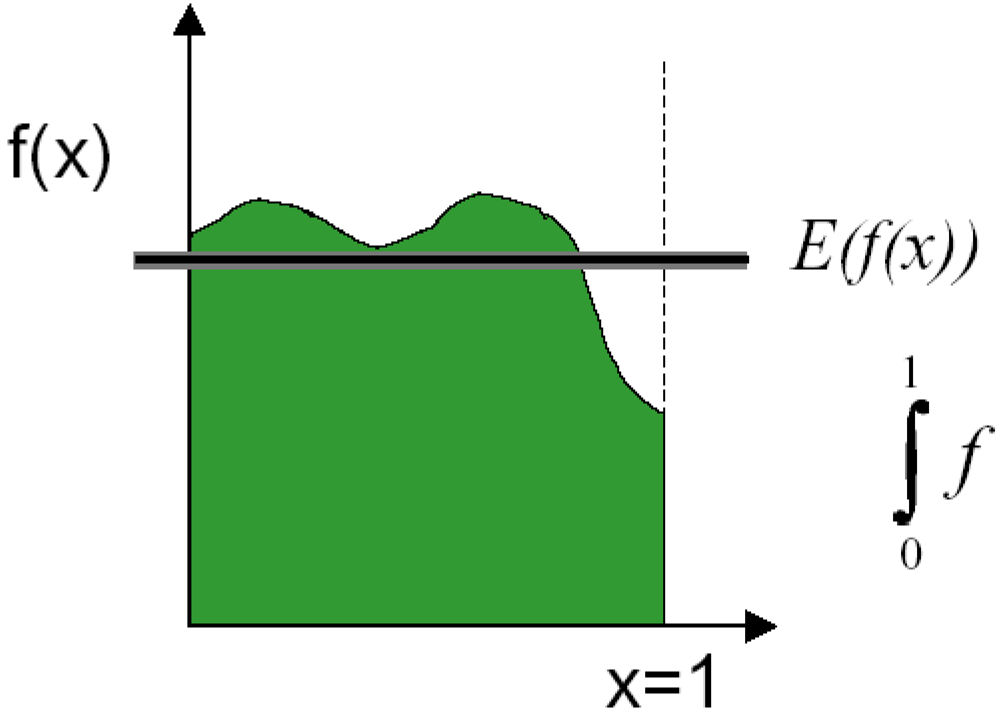
> Can approximate using another function g(x)



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

Monte Carlo Simulation Example

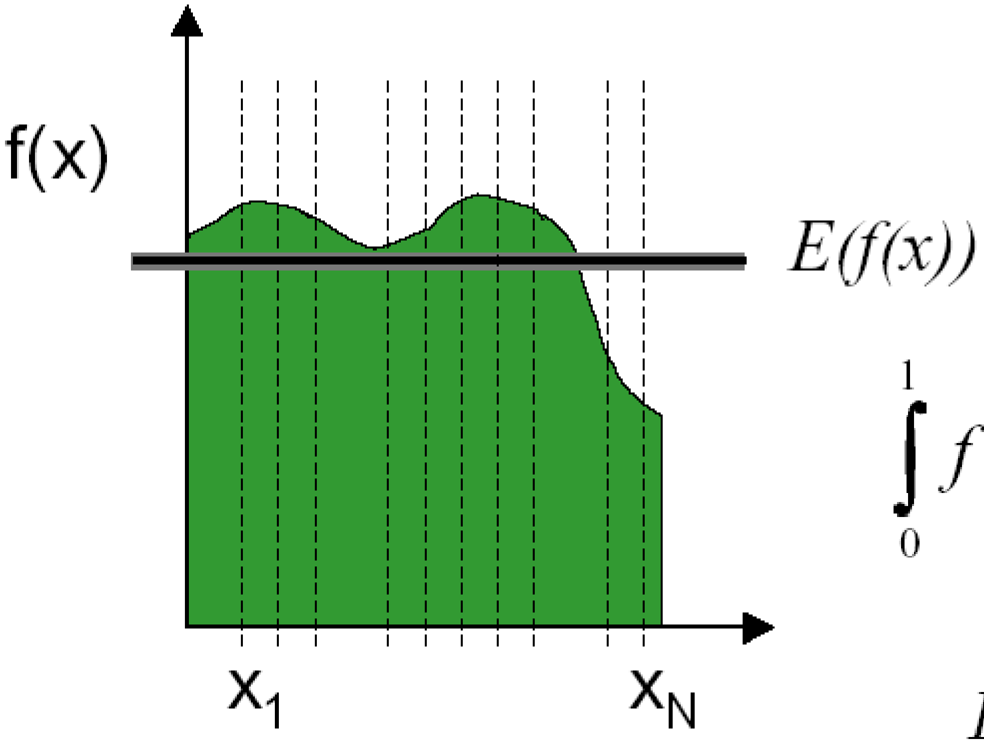
> Can approximate by taking the average or expected value



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

Monte Carlo Simulation Example

> Estimate the average by taking N samples



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

$$E(x) \approx \frac{1}{N}\sum_{i=1}^{N} x_i$$

Monte Carlo Integration

$$\mathbf{I} = \int_{a}^{b} f(x)dx$$

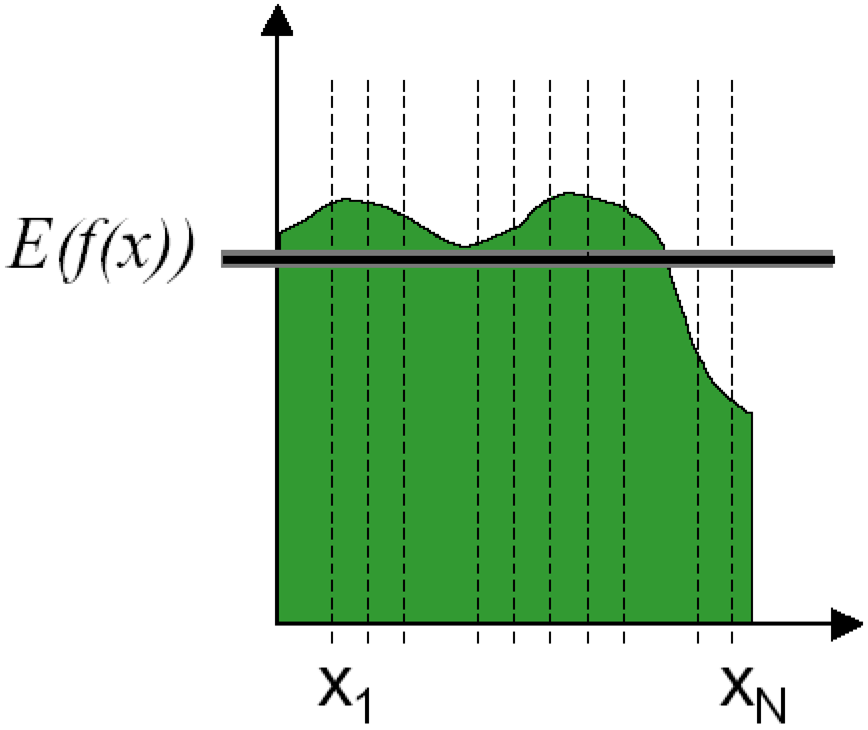
$$\mathbf{I}_m = (b-a)\frac{1}{N}\sum_{i=1}^{N} f(x_i)$$

- I_m = Monte Carlo estimate
- N = number of samples
- $x_1, x_2, \ldots, x_N$ are uniformly distributed random numbers between a and b

$$\lim_{N\to\infty} \mathbf{I}_m = \mathbf{I}$$

Monte Carlo Simulation Example

> The variance describes how much the sampled values *vary* from each other.



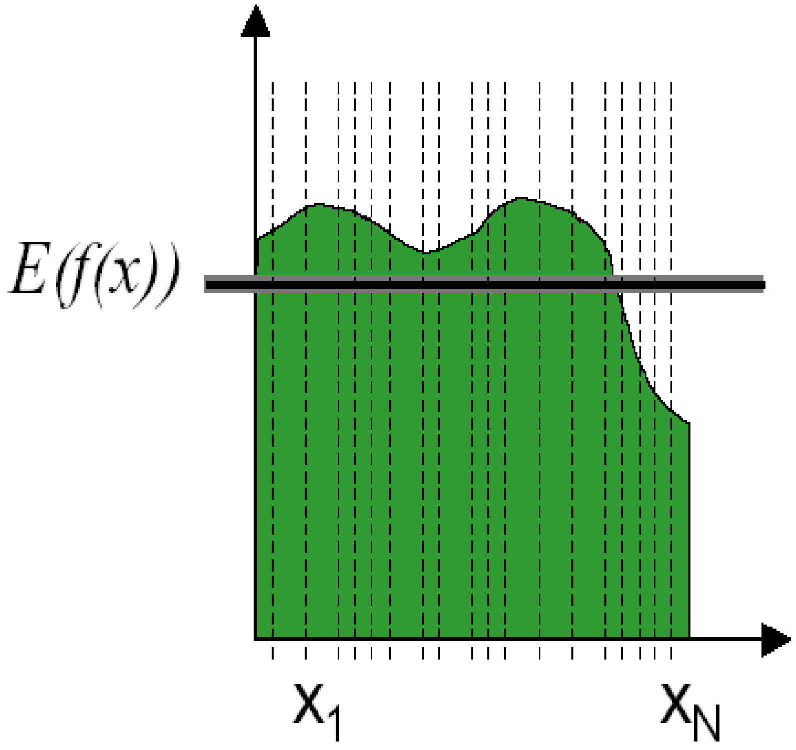
$$Var[E(f(x))] = \sum_{i=1}^{N} [f(x_i) - E(f(x))]^2$$

$$Var[E(f(x))] = \frac{1}{N} Var[f(x)]$$

- Variance proportional to 1/N

Monte Carlo Simulation Example

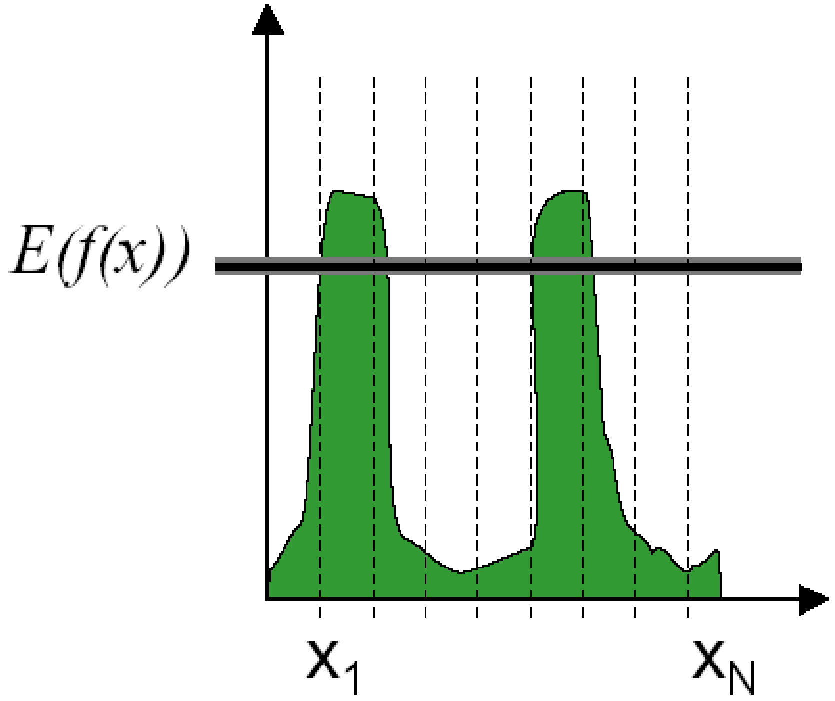
> Standard Deviation is just the square root of the variance
> Standard Deviation proportional to 1 / sqrt(N)



> Need 4X samples to halve the error

Monte Carlo Simulation Example

- Problem:
  - Variance (noise) decreases slowly
  - Using more samples only removes a small amount of noise

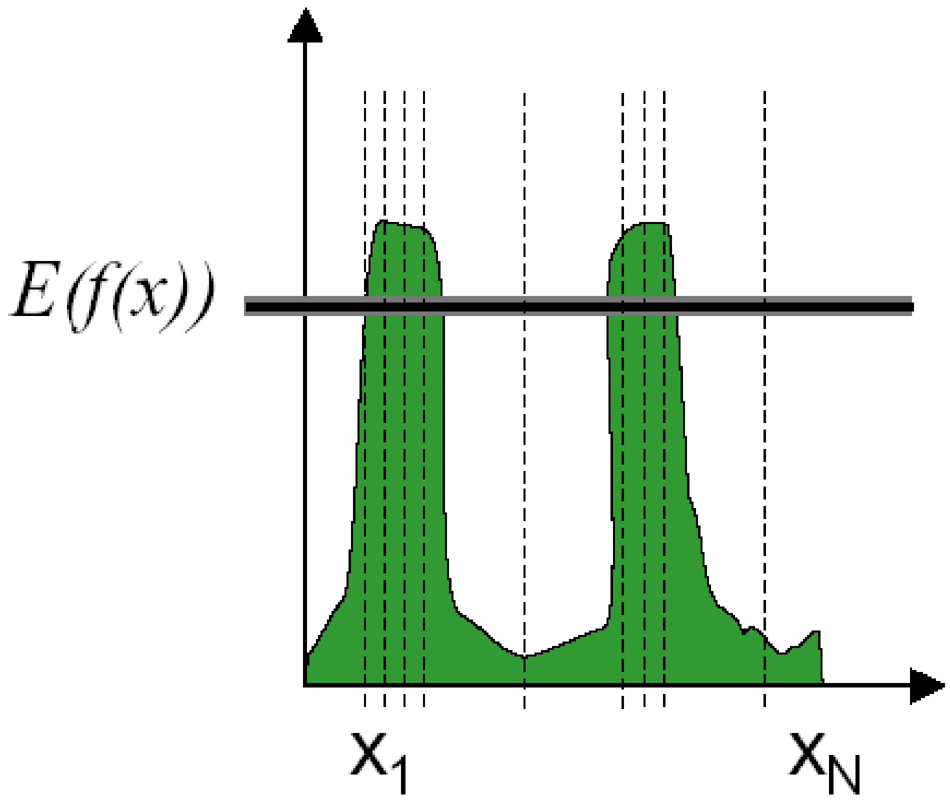


Types of Sampling

- Convenience or Accidental Sampling (This is bad).
  - Grabbing whatever is easier.
  - Grabbing whatever is available/possible.
- Cluster Sampling
  - Divide the data into clusters and sample select few clusters.
- Simple Random Sample (Most common)
  - Every point is subjected to a probability of being sampled.
- Stratified Sampling
  - Sampling subpopulations in a representative fashion.
  - This is sometimes important for class-imbalance problems.
- Systematic Sampling
  - Sampling every k-th element of a population. (Common in SQL servers, "table sampling")
  - Bad for time series data.

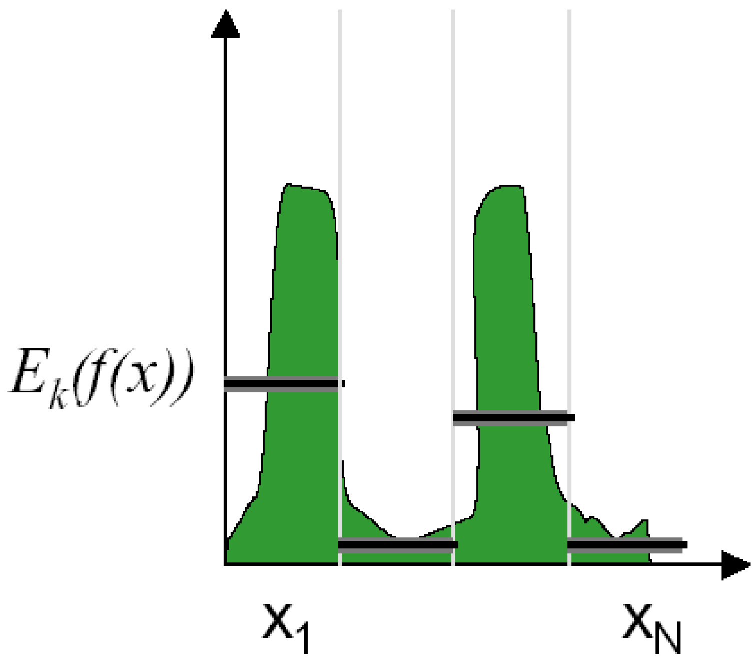
Monte Carlo Simulation Example

- Importance Sampling: use more samples in important regions of the function
- If function is high in small areas, use more samples there



Monte Carlo Simulation Example : Stratified Sampling

- Partition S into smaller domains S_i
- Evaluate integral as sum of integrals over S_i
- Example: jittering for pixel sampling
- Often works much better than importance sampling in practice



Monte Carlo methods example

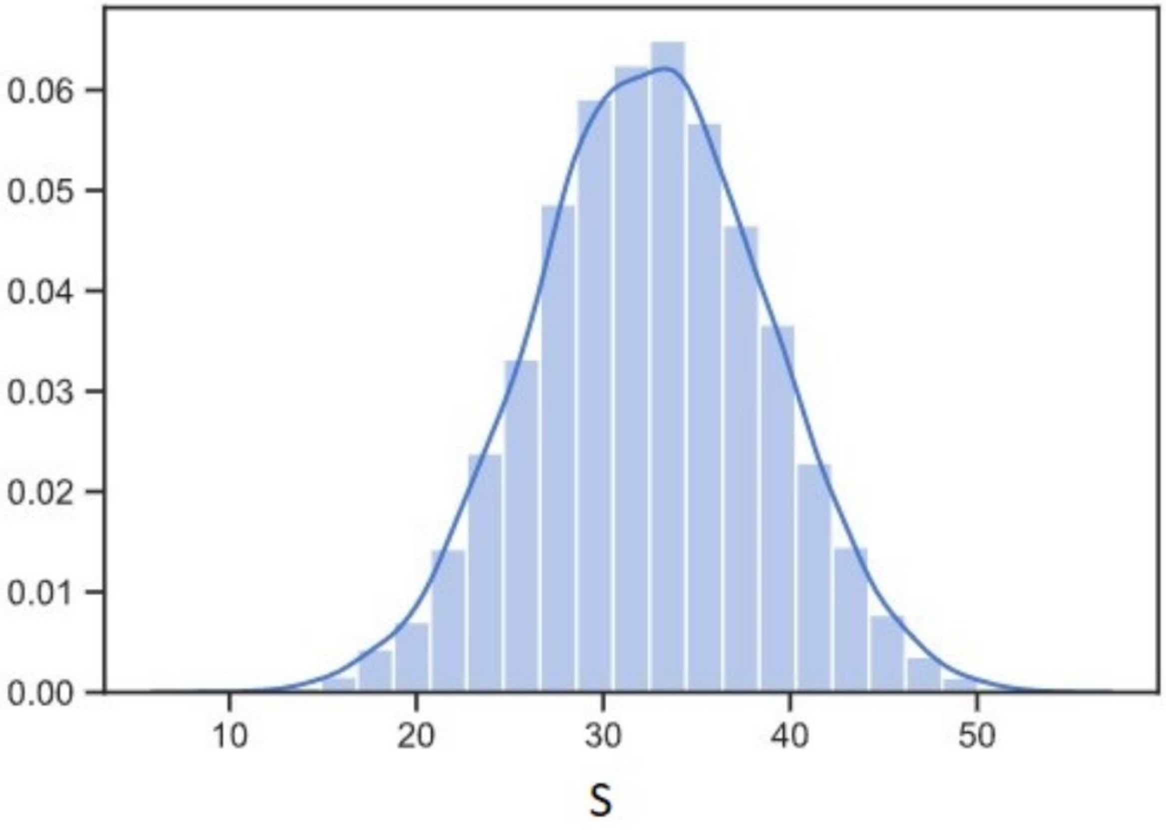
- recall the example of the random variable S representing the sum of two independent dice rolls
- we found the distribution of S by counting all possible outcomes and their relative frequency, for example in 5 out of 36 cases, $S = 6$, so $P(S = 6) = 5/36$
- but what if S represents the sum of 13 independent dice rolls? there are $6^{13} = 13,060,694,016$ possibilities, so forget about counting
- instead we **generate samples** from the sampling space, the more samples the better the estimate
- we can then report estimates from the samples or visualize its distribution

```
import numpy as np
import seaborn as sns

n_iter = 10000
s_samples = np.ones(n_iter)

for i in range(n_iter):
    s = 0
    for j in range(13):
        s += np.random.randint(6)
        s_samples[i] = s


sns.distplot(s_samples, bins = 23)
```



state your assumptions and derivations

- in the previous example, we assume the events (dice rolls) were independent, which simplifies the calculations (and the code) a lot
- in general this is not the case: we can have dependent events so it important to
  - list your constants, random variables, and what you want to estimate
  - for the random variables, state what they represent, what other variables they depend on (are derived from), and what conditional distribution they are assumed to follow given those other variables
  - repeatedly sample from the space in an order that respects the dependencies in the derivations
- for an example, refer to this week's assignment

Parallelism in Monte Carlo Methods

> Monte Carlo methods often amenable to parallelism
> Find an estimate about *p* times faster

OR

> Reduce error of estimate by $p^{1/2}$

Large Samples and the Law of Large Numbers

- If we roll a die 60 times and then 600 times, which of the dice will more likely have exactly 1/6th of the rolls equal to 6 appearing?
  - P(x=10|60 trials)=?
    - 0.13701
  - P(x=100|600trails)=?
    - 0.04366
- Visit: https://stattrek.com/online-calculator/binomial for probability calculations.

Large Samples and the Law of Large Numbers

- If we roll a die 60 times and then 600 times, which of the dice will more likely have exactly 1/6th of the rolls equal to 6 appearing?
  - P(x=10|60 trials)=?
    - BinomPDF(10) = 0.13701
  - P(x=100|600trails)=?
    - BinomPDF(100)=0.04366
- Which die will be more likely to be within 5% of a 1/6?
  - P(7<x<13|60 trails)=?
    - BinomCDF(x=13) – BinomCDF(x=7)
    - BinomPDF(x<=13) – BinomPDF(x<=7)
    - =0.88478 - 0.19580 = 0.68898
  - P(70<x<130|600 trails)=?
    - BinomCDF(x=130) – BinomCDF(x=70)
    - BinomPDF(x<=130) – BinomPDF(x<=70)
    - 0.99939 - 0.00038 = 0.99901

Law of Large Numbers

- Sample statistics converge to the population statistics as more unbiased experiments are performed.
  - Example: the mean of 50 coin flips (0,1)=(T,H) is usually farther away from the true mean of 0.5 than if we did 5,000 coin flips.

Standard Deviation vs. Standard Error

- Standard Deviation: A Computed Measure of variability in a sample or population.
  - "My sample values have a standard deviation of XYZ."
- Standard Error: Measure of variability in the *statistics* of the sample.
  - "I've repeated my experiment many times, and the standard deviation of each of the above standard deviations is small".
- For example:
  - Standard deviation: On a sample.
  - Standard Error: Standard deviation of *a set of means* calculated from multiple samples.
    - You can imagine that the larger my sample, the more confident we can be about the mean.
  - Standard error of a statistic decreases by a rate of 1/sqrt(n) where n is your sample size.

the end