Lecture 3

More Stats:

Expectations, LLN, Monte Carlo, and the CLT

Collaboration Policy

"You can collaborate with up-to two additional people on any homework, as long as you write your own homework and mention your collaborators on it."

What does this mean?

- You can talk to anyone as long as you write your own work for anything you plan on submitting
- You can turn in 3 identical problem sets as long as each of you name both of your collaborators on your homework. Each person MUST turn in their own problem set.
- Attribution is key!! Anything that you “borrow” from someone/ someplace else please make sure to attribute!!!

So far:

- Probability and Bayes Theorem
- Distributions
- Frequentist Statistics
- Maximum Likelihood Estimation
- Sampling Distribution

Today

- Expectations and some notation
- The Law of large numbers
- Simulation and Monte Carlo for Integration
- Sampling and the CLT
- Errors in Monte Carlo

Expectation $E_f[X]$

Why calculate it?

- we'll see it corresponds to the frequentist notion of probability
- we often want point estimates

Expectations are always with respect to a pmf or density. Often just called the **mean** of the mass function or density. More weight to more probable values.

For the discrete random variable X:

$$E_f[X] = \sum_x x\, f(x).$$

Continuous case:

$$E_f[X] = \int x\, f(x) dx = \int x dF(x),$$

Notation

The expected value, or mean, or first moment, of X is defined to be

$$E_f X = \int x dF(x) = \begin{cases} \sum_x x f(x) & \text{if X is discrete} \\ \int x f(x) dx & \text{if X is continuous} \end{cases}$$

assuming that the sum (or integral) is well defined.

The discrete sum can be said to be an integral with respect to a counting measure.

LOTUS: Law of the unconscious statistician

Also known as **The rule of the lazy statistician**.

Theorem:

if $Y = r(X)$,

$$E[Y] = \int r(x) dF(x)$$

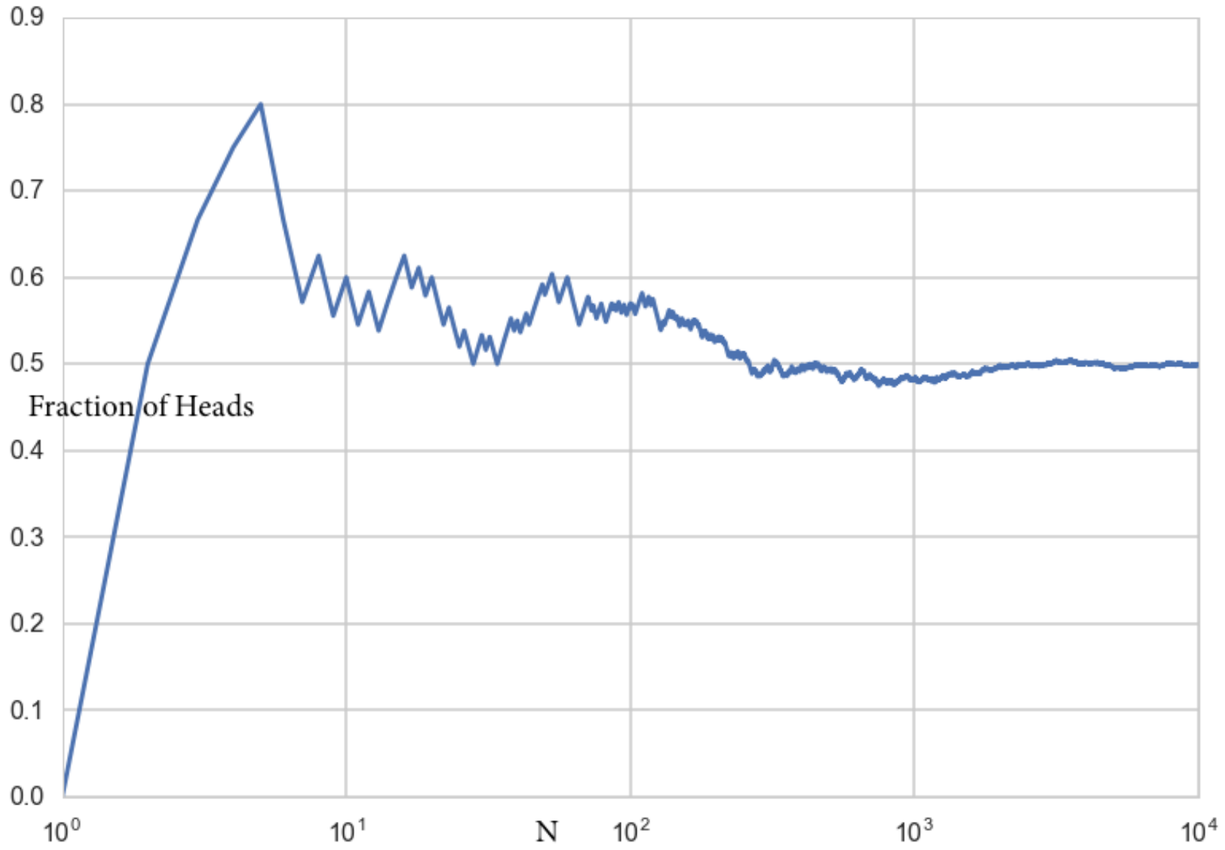
Application: Probability as Expectation

Let A be an event and let $r(x) = I_A(x)$ (Indicator for event A)

Then:

$$E_f[I_A(X)] = \int I_A(x) dF(x) = \int_A f_X(x) dx = p(X \in A)$$

Ever longer sequences for means



Law of Large numbers

Let $x_1, x_2, \ldots, x_n$ be a sequence of IID values from random variable X, which has finite mean μ. Let:

$$S_n = \frac{1}{n} \sum_{i=1}^{n} x_i,$$

Then:

$$S_n \to \mu \, as \, n \to \infty.$$

Frequentist Interpretation of probability

$$E_F[I_A(X)] = p(X \in A)$$

Suppose $Z = I_A(X) \sim Bernoulli(p = P(A))$.

Now if we take a long sequence `seq=10010011100....` from Z, then

$$P(A) = \texttt{mean(seq)} \text{ as } \texttt{length(seq)} \to \infty$$

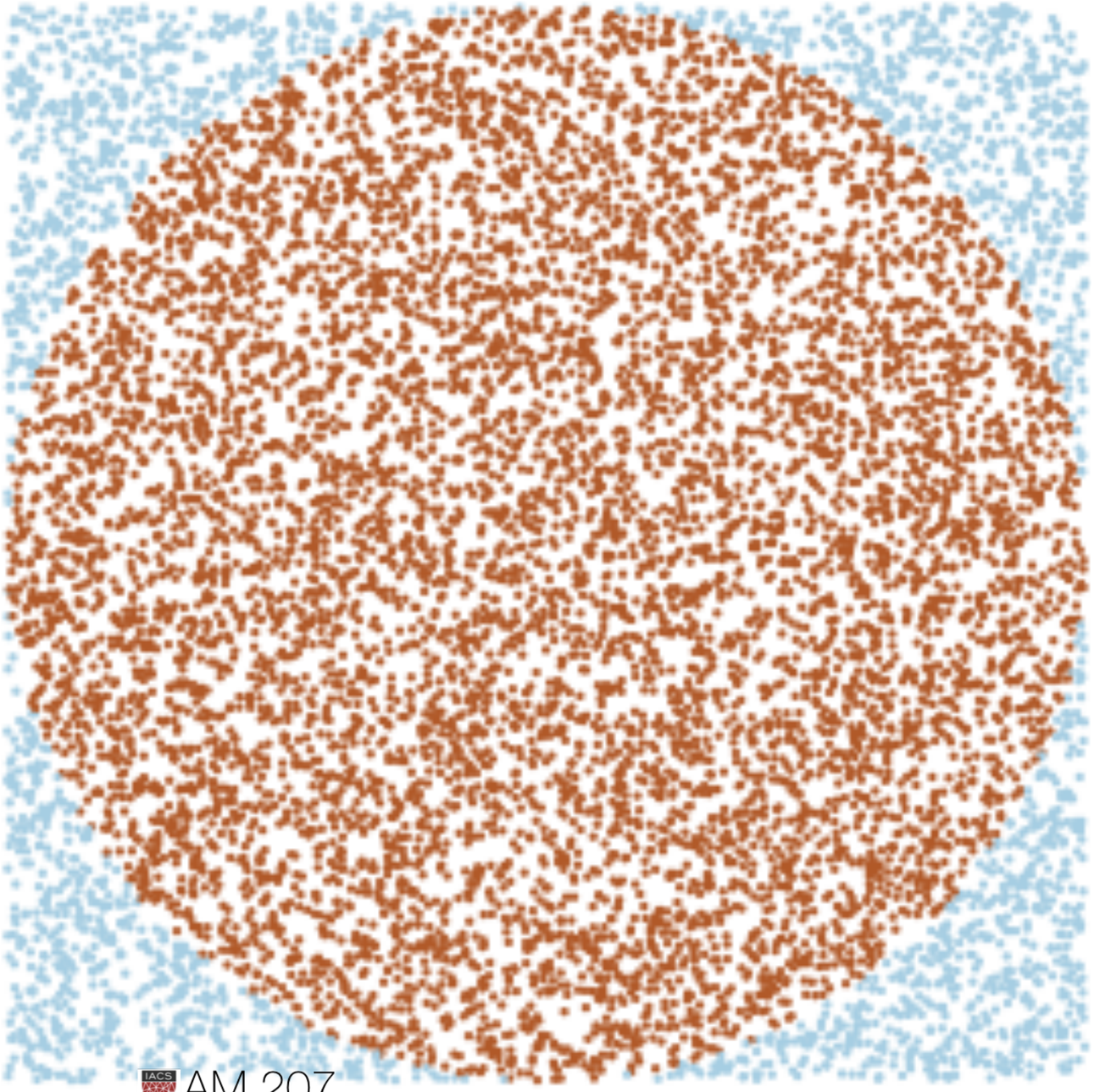
Monte Carlo Algorithm

- use randomness to solve what is often a deterministic problem
- application of the law of large numbers
- integrals, expectations, marginalization
- we'll study optimization, integration, and obtaining draws from a probability distribution

...I wondered whether a more practical method than "abstract thinking" might not be to lay it out say one hundred times and simply observe and count the number of successful plays

...and more generally how to change processes described by certain differential equations into an equivalent form interpretable as a succession of random operations

— *Stanislaw Ulam*



estimating π

$$A = \int_x \int_y I_{\in C}(x,y)dxdy = \int\int_{\in C} dxdy$$

$$E_f[I_{\in C}(X,Y)] = \int I_{\in C}(X,Y)dF(X,Y)$$

$$= \int\int_{\in C} f_{X,Y}(x,y)dxdy = p(X,Y \in C)$$

If $f_{X,Y}(x,y) \sim Uniform(V)$:

$$= \frac{1}{V}\int\int_{\in C} dxdy = \frac{A}{V}$$

Formalize Monte Carlo Integration idea

For Uniform pdf: $U_{ab}(x) = 1/V = 1/(b-a)$

$$J = \int_a^b f(x) U_{ab}(x)\, dx = \int_a^b f(x)\, dx / V = I/V$$

From LOTUS and the law of large numbers:

$$I = V \times J = V \times E_U[f] = V \times \lim_{n\to\infty} \frac{1}{N} \sum_{x_i \sim U} f(x_i)$$

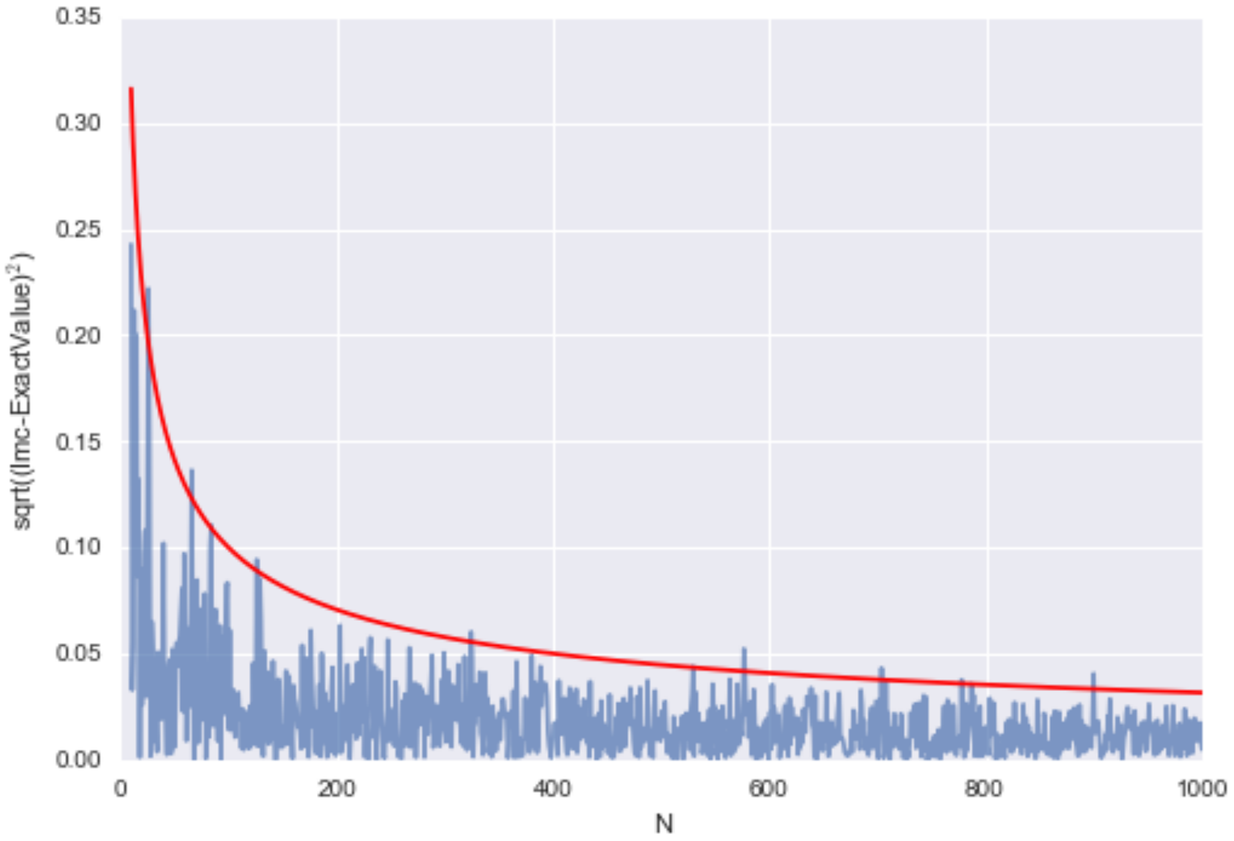
Example

$$I = \int_2^3 [x^2 + 4\,x\,\sin(x)]\,dx.$$

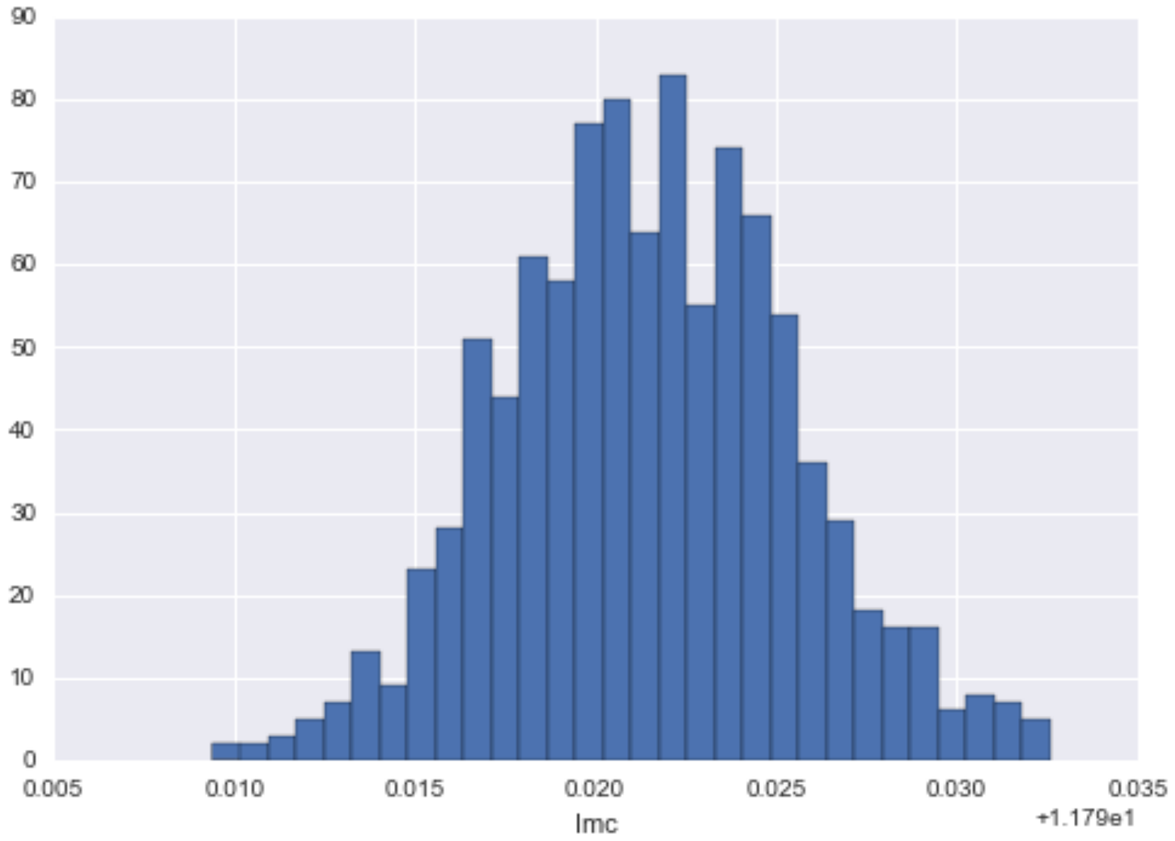
```
def f(x):
    return x**2 + 4*x*np.sin(x)
def intf(x):
    return x**3/3.0+4.0*np.sin(x) - 4.0*x*np.cos(x)
a = 2;
b = 3;
N= 10000
X = np.random.uniform(low=a, high=b, size=N)
Y =f(X)
V = b-a
Imc= V * np.sum(Y)/ N;
exactval=intf(b)-intf(a)
print("Monte Carlo estimation=",Imc, "Exact number=", intf(b)-intf(a))
```

```
Monte Carlo estimation= 11.8120823531 Exact number= 11.8113589251
```

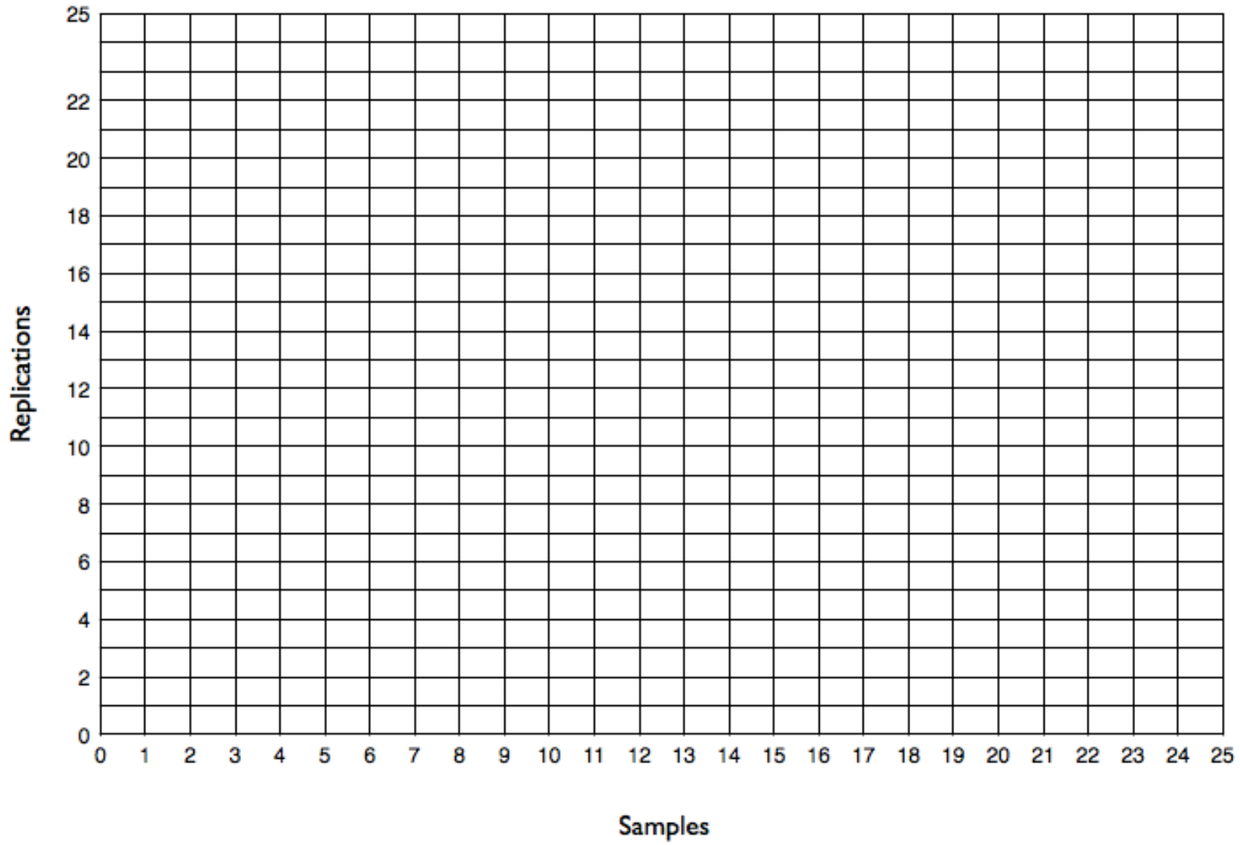
Accuracy as a function of the number of samples



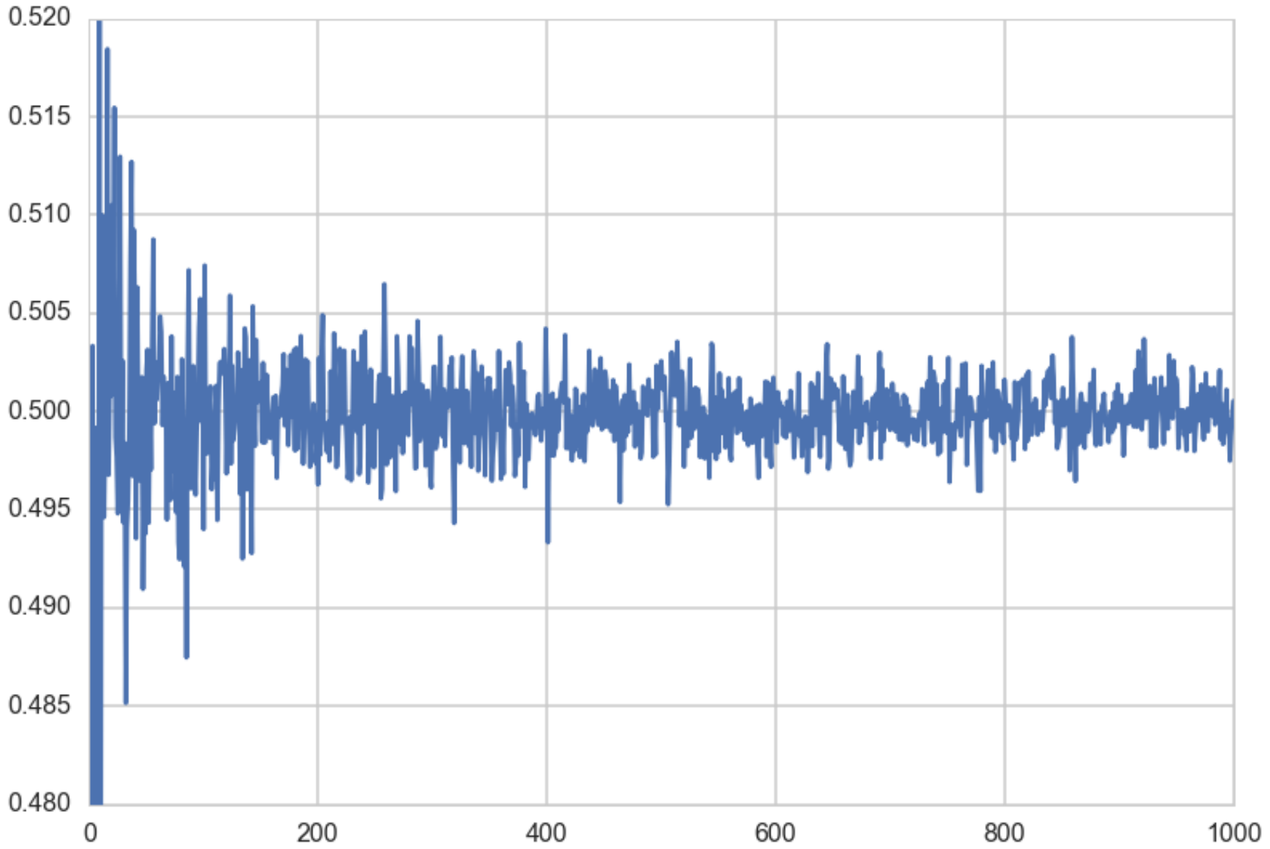
Variance of the estimate



M replications of N coin tosses



sample means: 200 replications of N coin tosses



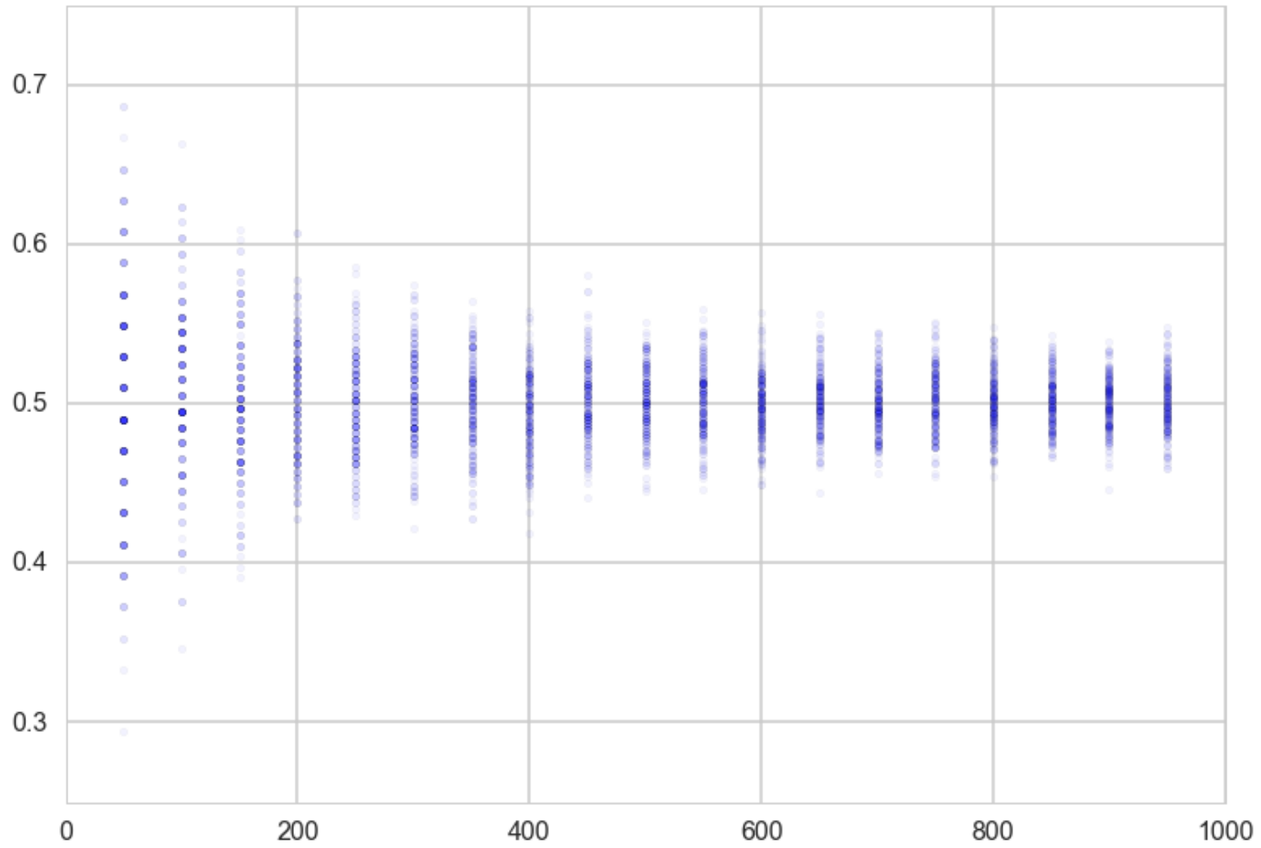
$$E_{\{R\}}(N\,\bar{x}) = E_{\{R\}}(x_1 + x_2 + \ldots + x_N) = E_{\{R\}}(x_1) + E_{\{R\}}(x_2) + \ldots + E_{\{R\}}(x_N)$$

In limit $M \to \infty$ of replications, each of the expectations in RHS can be replaced by the population mean μ using the law of large numbers! Thus:

$$E_{\{R\}}(N\,\bar{x}) = N\,\mu$$
$$E_{\{R\}}(\bar{x}) = \mu$$

In limit $M \to \infty$ of replications the expectation value of the sample means converges to the population mean.

Distribution of Sample Means



Now let underlying distribution have well defined mean μ AND a well defined variance σ^2.

$$V_{\{R\}}(N\,\bar{x}) = V_{\{R\}}(x_1 + x_2 + \ldots + x_N) = V_{\{R\}}(x_1) + V_{\{R\}}(x_2) + \ldots + V_{\{R\}}(x_N)$$

Now in limit $M \to \infty$, each of the variances in the RHS can be replaced by the population variance using the law of large numbers! Thus:

$$V_{\{R\}}(N\,\bar{x}) = N\,\sigma^2$$

$$V(\bar{x}) = \frac{\sigma^2}{N}$$

The Central Limit Theorem (CLT)

Let $x_1, x_2, \ldots, x_n$ be a sequence of IID values from a random variable X. Suppose that X has the finite mean μ AND finite variance σ^2. Then:

$$S_n = \frac{1}{n}\sum_{i=1}^{n} x_i, \text{ converges to}$$

$$S_n \sim N(\mu, \frac{\sigma^2}{n}) \, as \, n \to \infty.$$

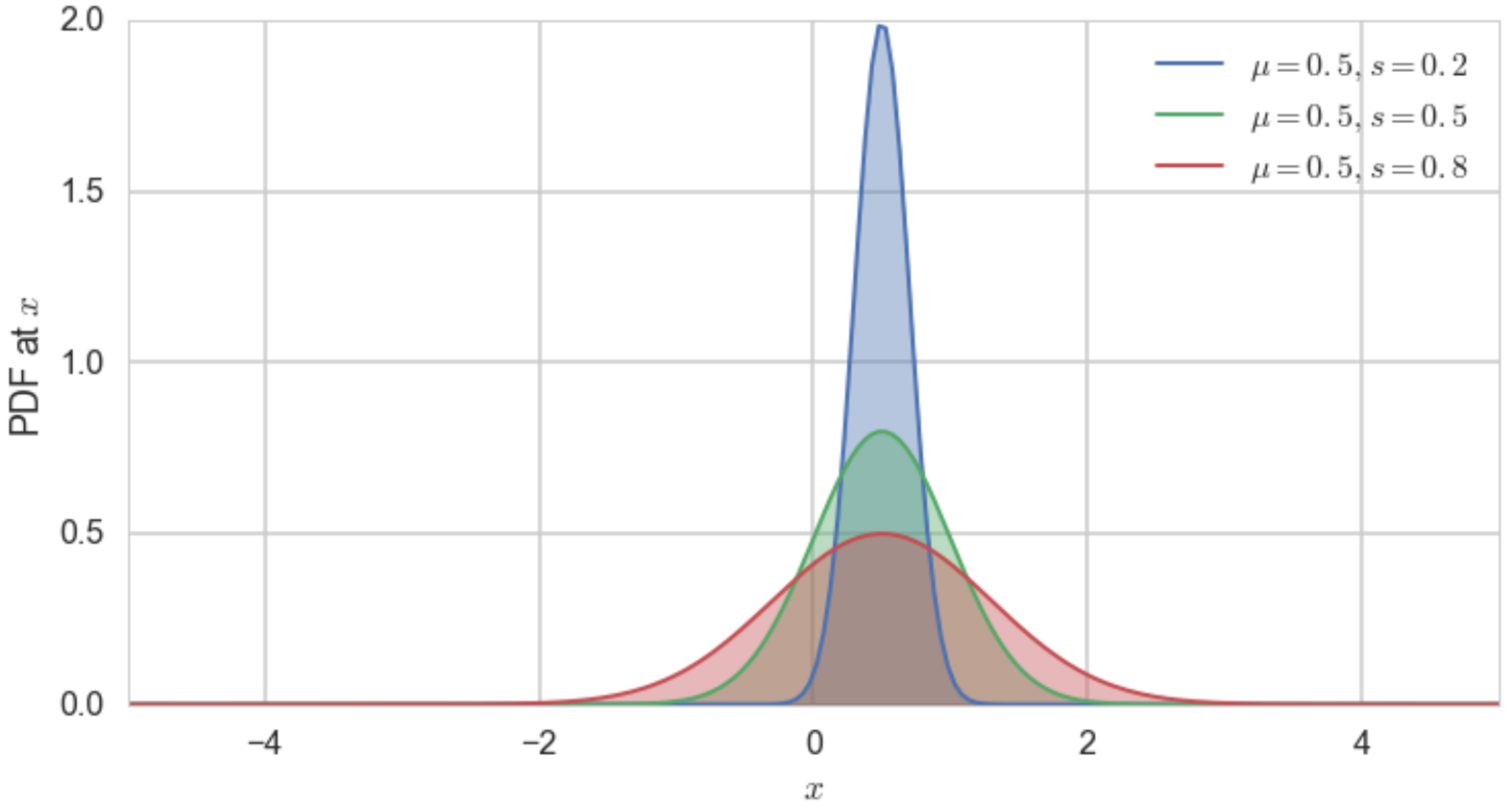
Origin story of the Gaussian

Under Lyapunov conditions, the x_i dont have to be identically distributed, as long as μ is the mean of the means and σ^2 is the sum of the individual variances. This gives us the standard origin story of the gaussian.

$\sqrt{V}$ is called the **standard error** s.

$$s = \frac{\sigma}{\sqrt{n}}$$

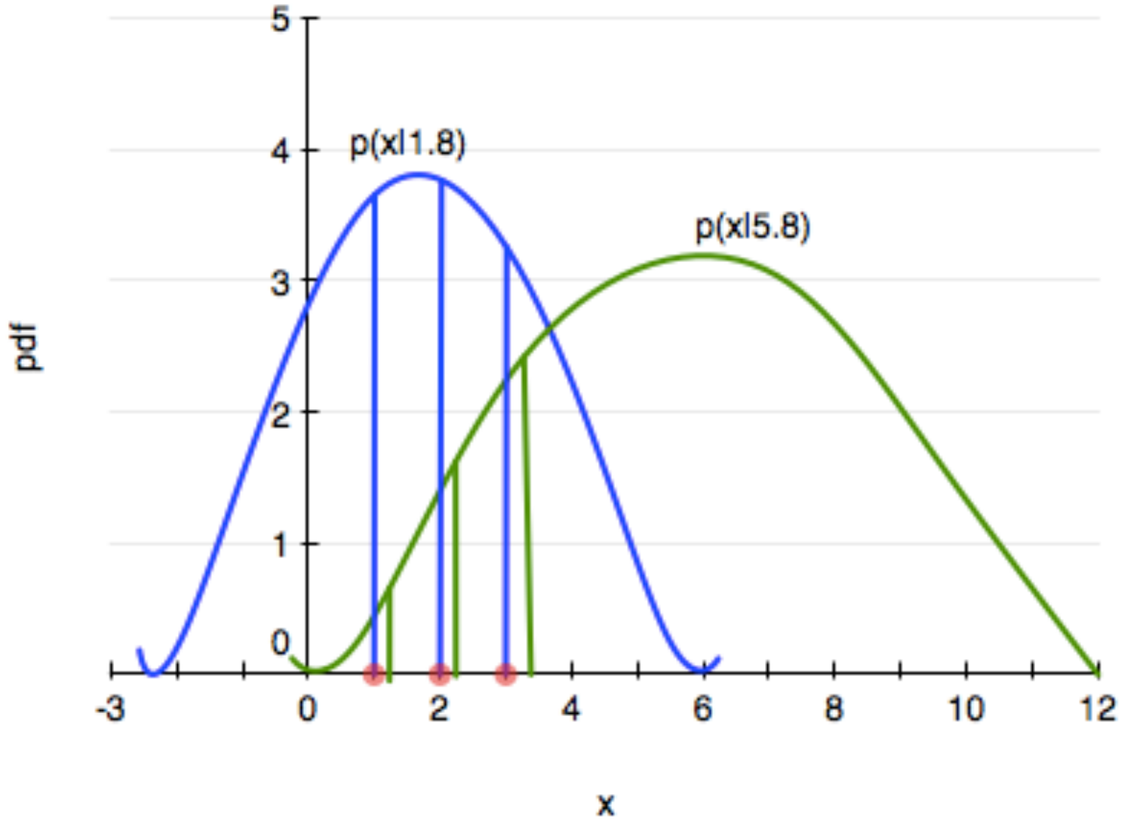
Gaussians



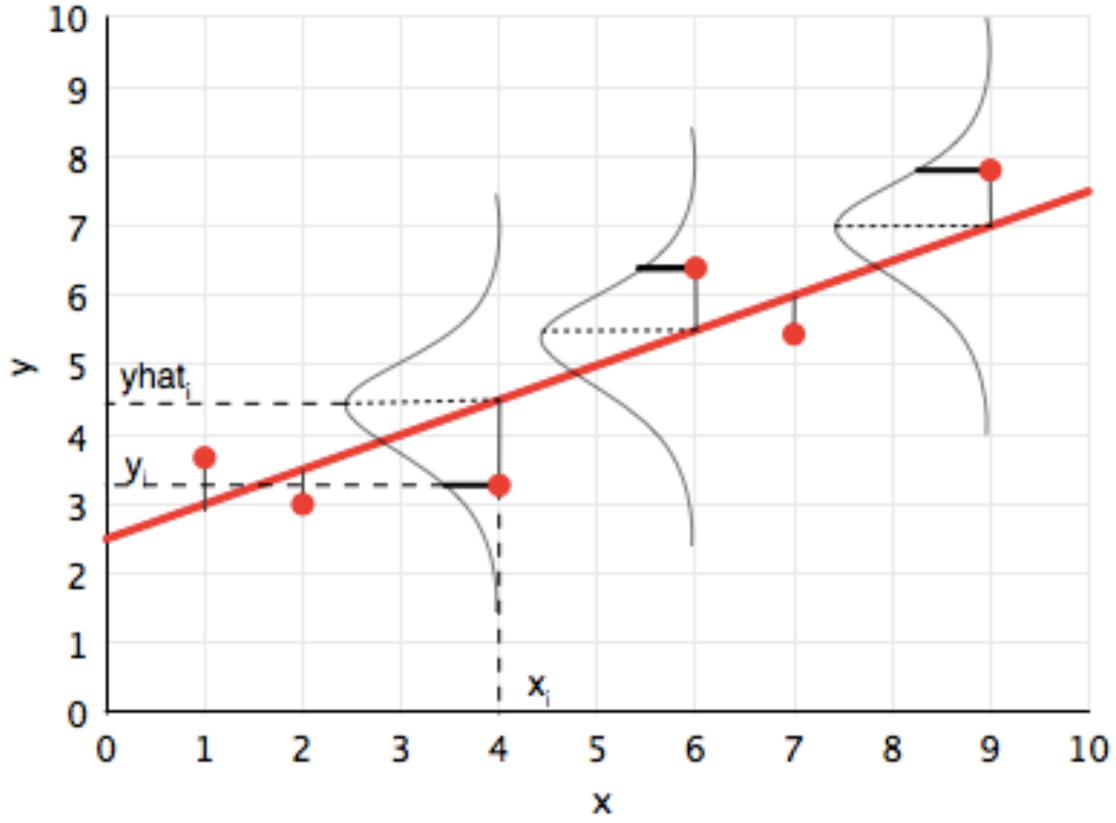
Meaning

- weight-watchers' study of 1000 people, average weight is 150 lbs with σ of 30lbs.
- Randomly choose many samples of 100 people each, the mean weights of those samples would cluster around 150lbs with a standard error of 3lbs.
- a different sample of 100 people with an average weight of 170lbs would be more than 6 standard errors beyond the population mean.

Maximum Likelihood estimation



Linear Regression MLE



Gaussian Distribution assumption

Each y_i is gaussian distributed with mean $\mathbf{w} \cdot \mathbf{x}_i$ (the y predicted by the regression line) and variance σ^2:

$$y_i \sim N(\mathbf{w} \cdot \mathbf{x}_i, \sigma^2).$$

$$N(\mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(y-\mu)^2/2\sigma^2},$$

We can then write the likelihood:

$$\mathcal{L} = p(\mathbf{y}|\mathbf{x}, \mathbf{w}, \sigma) = \prod_i p(\mathbf{y}_i|\mathbf{x}_i, \mathbf{w}, \sigma)$$

$$\mathcal{L} = (2\pi\sigma^2)^{(-n/2)} e^{\frac{-1}{2\sigma^2}\sum_i (y_i - \mathbf{w}\cdot\mathbf{x}_i)^2}.$$

The log likelihood ℓ then is given by:

$$\ell = \frac{-n}{2} log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_i (y_i - \mathbf{w}\cdot\mathbf{x}_i)^2.$$

Maximizing gives:

$$\mathbf{w}_{MLE} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y},$$

where we stack rows to get:

$$\mathbf{X} = stack(\{\mathbf{x}_i\})$$

$$\sigma^2_{MLE} = \frac{1}{n}\sum_i (y_i - \mathbf{w}\cdot\mathbf{x}_i)^2.$$

Back to Monte Carlo

We want to calculate:

$$S_n(f) = \frac{1}{n}\sum_{i=1}^{n} f(x_i)$$

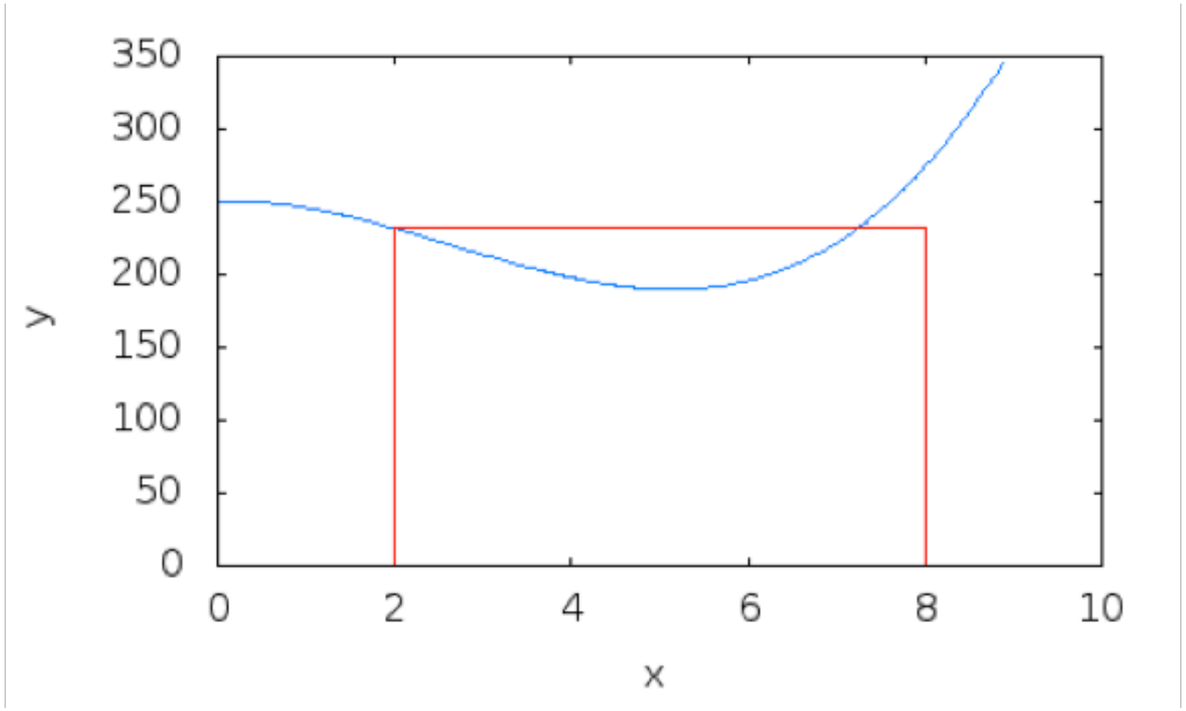
- Whatever $V[f(X)]$ is, the variance of the sampling distribution of the mean goes down as $1/n$
- Thus s goes down as $1/\sqrt{n}$

Why is this important?

- In higher dimensions d, the CLT still holds and the error still scales as $\frac{1}{\sqrt{n}}$.
- How does this compete with numerical integration? For $n = N^{1/d}$:
  - left or right rule: $\propto 1/n$, Midpoint rule: $\propto 1/n^2$
  - Trapezoid: $\propto 1/n^2$, Simpson: $\propto 1/n^4$

Basic Numerical Integration idea

(from wikipedia)

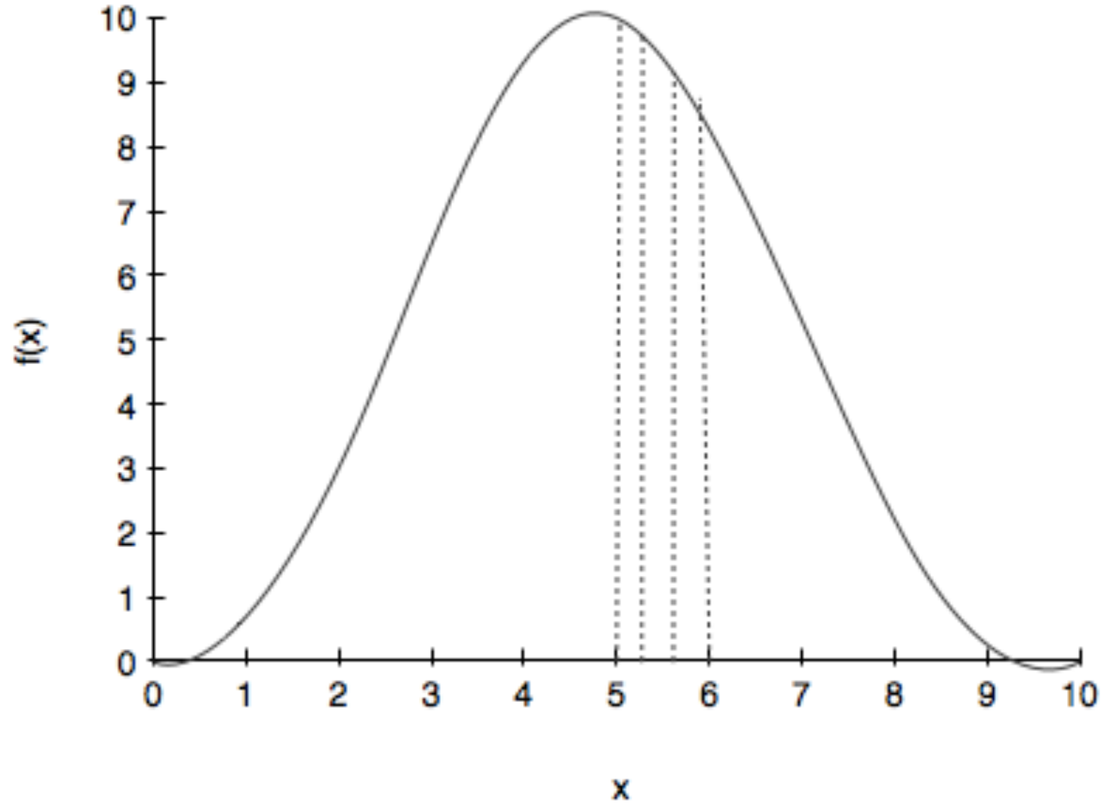


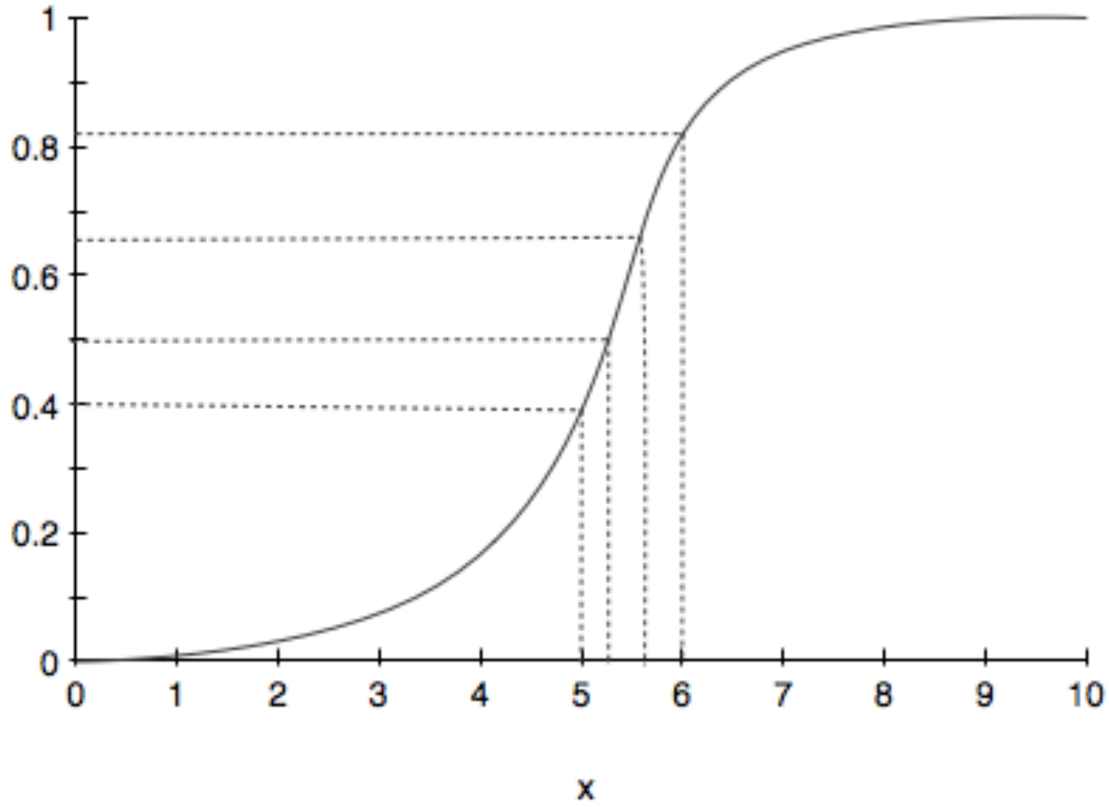
Next time

In order to calculate expectations, do integrals, and do statistics, we must learn how to do

SAMPLING

A taste: Inverse transform





algorithm

The CDF F must be invertible!

1. get a uniform sample u from $Unif(0, 1)$
2. solve for x yielding a new equation $x = F^{-1}(u)$ where F is the CDF of the distribution we desire.
3. repeat.

Example: exponential

pdf: $f(x) = \frac{1}{\lambda} e^{-x/\lambda}$ for $x \geq 0$ and $f(x) = 0$ otherwise.

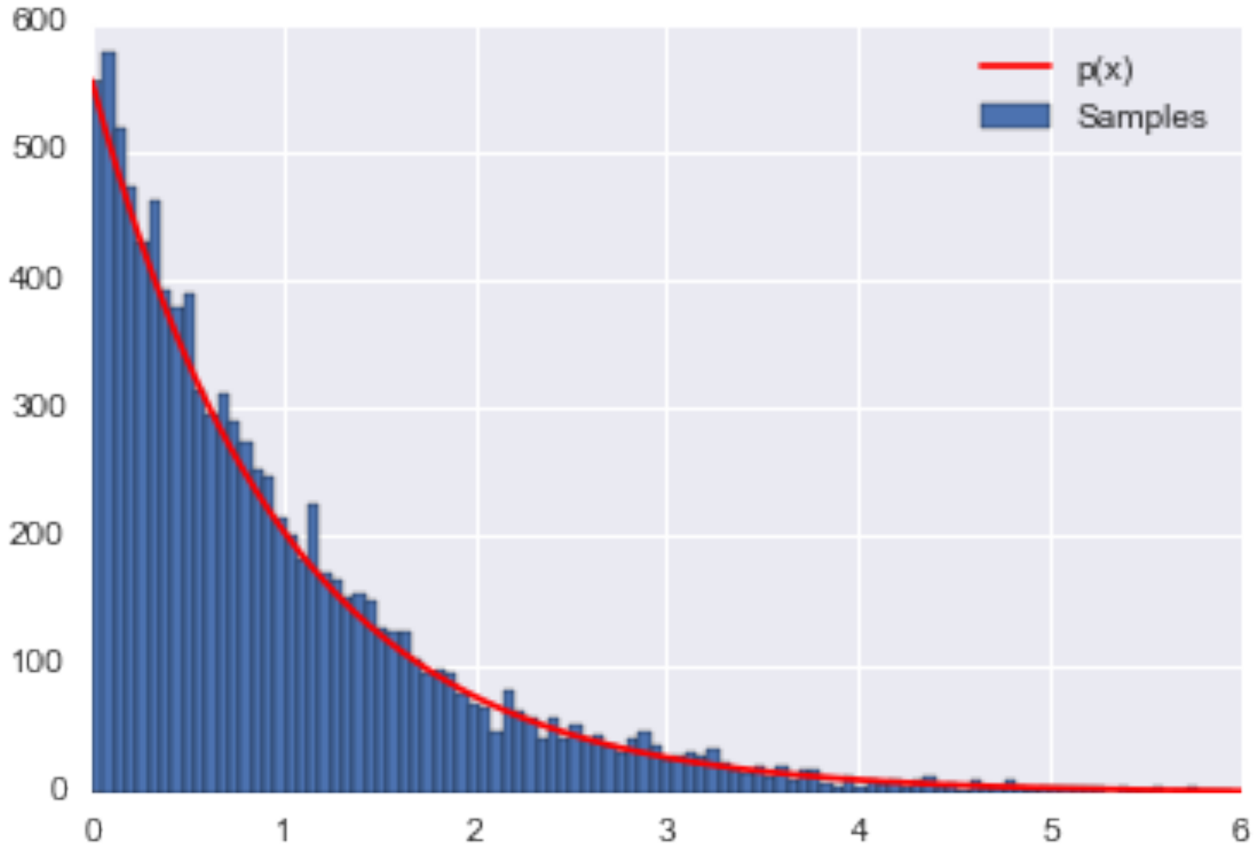
$$u = \int_0^x \frac{1}{\lambda} e^{-x'/\lambda} dx' = 1 - e^{-x/\lambda}$$

Solving for x

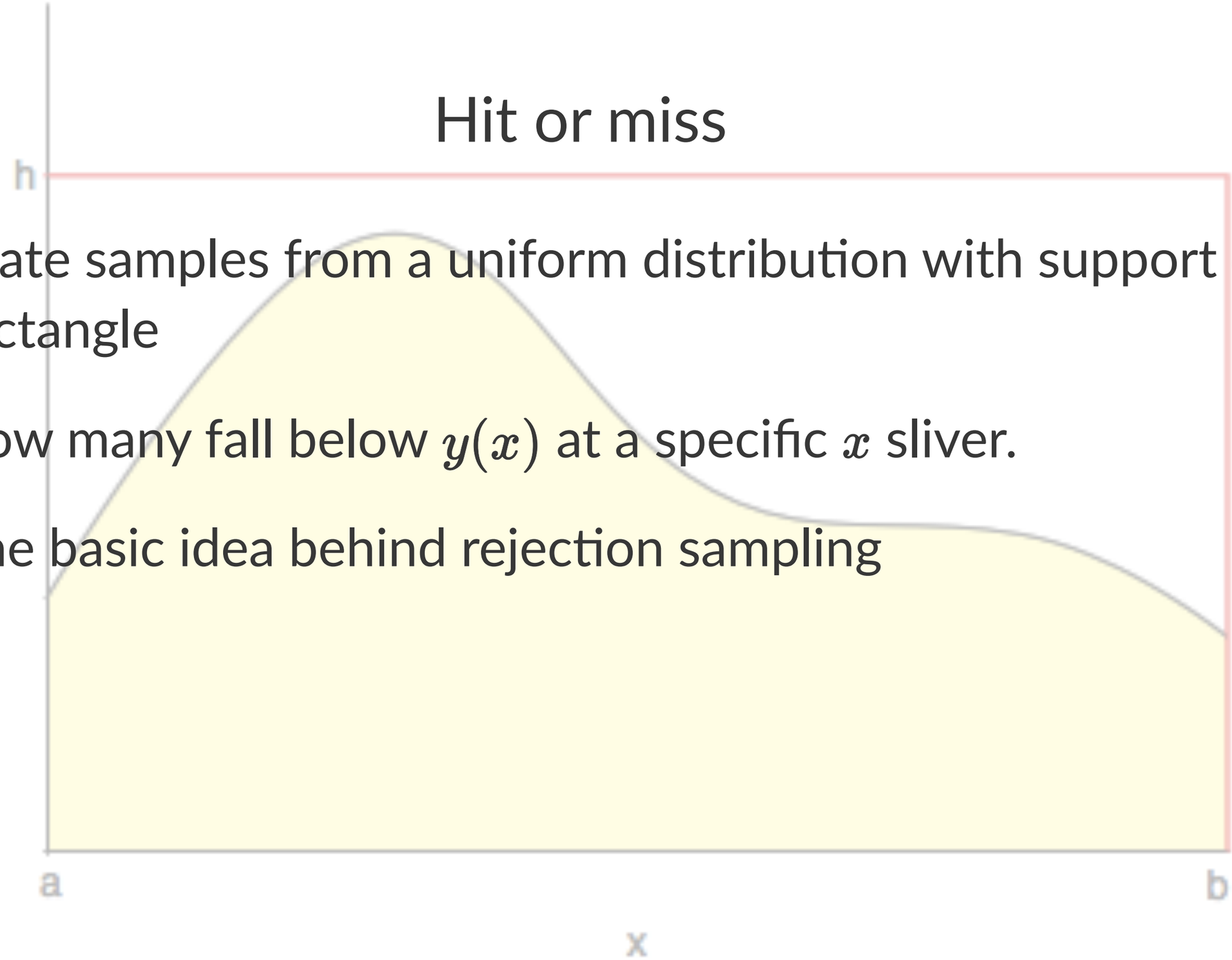
$$x = -\lambda \ln(1 - u)$$

code

```
p = lambda x: np.exp(-x)
CDF = lambda x: 1-np.exp(-x)
invCDF = lambda r: -np.log(1-r) # invert the CDF
xmin = 0 # the lower limit of our domain
xmax = 6 # the upper limit of our domain
rmin = CDF(xmin)
rmax = CDF(xmax)
N = 10000
# generate uniform samples in our range then invert the CDF
# to get samples of our target distribution
R = np.random.uniform(rmin, rmax, N)
X = invCDF(R)
hinfo = np.histogram(X,100)
plt.hist(X,bins=100, label=u'Samples');
# plot our (normalized) function
xvals=np.linspace(xmin, xmax, 1000)
plt.plot(xvals, hinfo[0][0]*p(xvals), 'r', label=u'p(x)')
plt.legend()
```



Hit or miss



- Generate samples from a uniform distribution with support on the rectangle
- See how many fall below $y(x)$ at a specific x sliver.

This is the basic idea behind rejection sampling