An Introduction to Bayesian Statistical Modeling using PyMC

Christopher Fonnesbeck *Vanderbilt University*

Abie Flaxman *University of Washington*

Outline

1. **Introduction to Bayes** *Me*
2. **Model building in PyMC** *Abie*
3. **Model fitting in PyMC** *Abie*
4. **Extending PyMC** *Me*

Rev. Thomas Bayes

Elected fellow of

The Royal Society, 1742

"Doctrine of Fluxions and a Defence of the Mathematicians Against the Objections of the Author of The Analyst"

Died 1761

Moorgate, London

An Essay towards solving a Problem in the Doctrine of Chances (1763)

LII. *An Essay towards solving a Problem in the Doctrine of Chances. By the late Rev. Mr.* Bayes, *communicated by Mr.* Price, *in a letter to* John Canton, *M. A. and F. R. S.*

Dear Sir,

Read Dec. 23, 1763. I now send you an essay which I have found among the papers of our deceased friend Mr. Bayes, and which, in my opinion, has great merit, and well deserves to be preserved. Experimental philosophy, you will find, is nearly interested in the subject of it; and on this account there seems to be particular reason for thinking that a communication of it to the Royal Society cannot be improper.

He had, you know, the honour of being a member of that illustrious Society, and was much esteemed by many as a very able mathematician. In an introduction which he has writ to this Essay, he says, that his design at first in thinking on the subject of it was, to find out a method by which we might judge concerning the probability that an event has to happen, in given circumstances, "*Given* the number of times in which an unknown event has happened and failed:

Required **the chance that the probability of its happening in a single trial lies somewhere between any two degrees of probability can be named”** *"number of times in which an unknown event has happened and failed"*

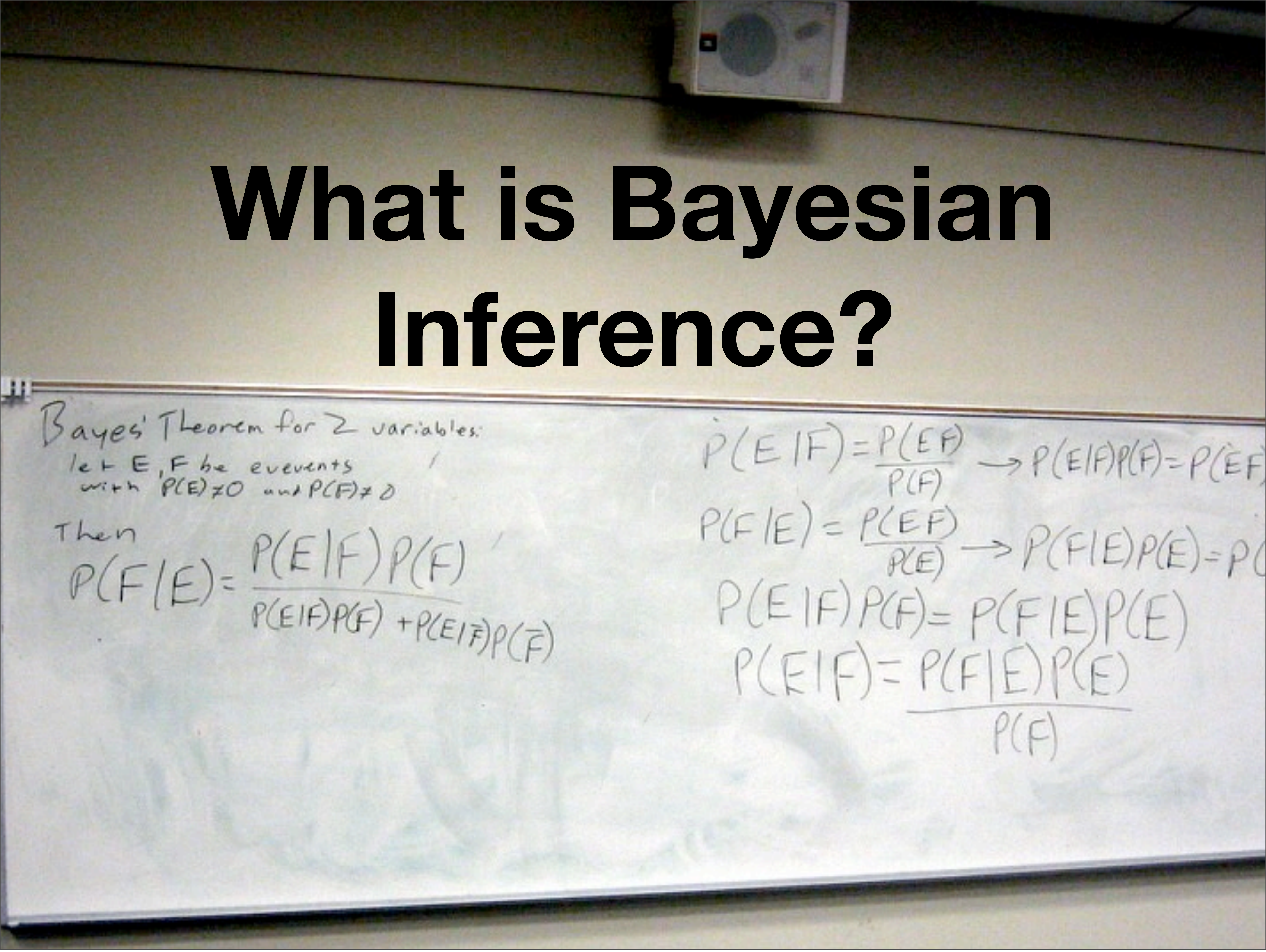
"its happening in a single trial"

$$Pr(a < \theta < b|y) = p$$

"two degrees of probability"

"unknown event" = Bernoulli trial (zero or one)

What is Bayesian Inference?



Practical methods for making inferences from data using probability models for quantities we observe and about which we wish to learn.

Gelman et al., 2004

Conclusions in terms of probability statements

$$p(\theta|y)$$

Classical inference conditions on unknown parameter

$$p(y|\theta)$$

Classical vs Bayesian Statistics

Frequentist

observations

random

model, parameters

fixed

Frequentist Inference

Choose an estimator

$$\hat{\mu} = \frac{\sum x_i}{n}$$

based on frequentist (asymptotic) criteria

Choose a test statistic

$$t = \frac{\bar{x} - \mu}{s/\sqrt{n}}$$

based on frequentist (asymptotic) criteria

Bayesian

observations

fixed

model, parameters

"random"

“inverse probability”

$$p(\theta|y)$$

Components of Bayesian Statistics

1

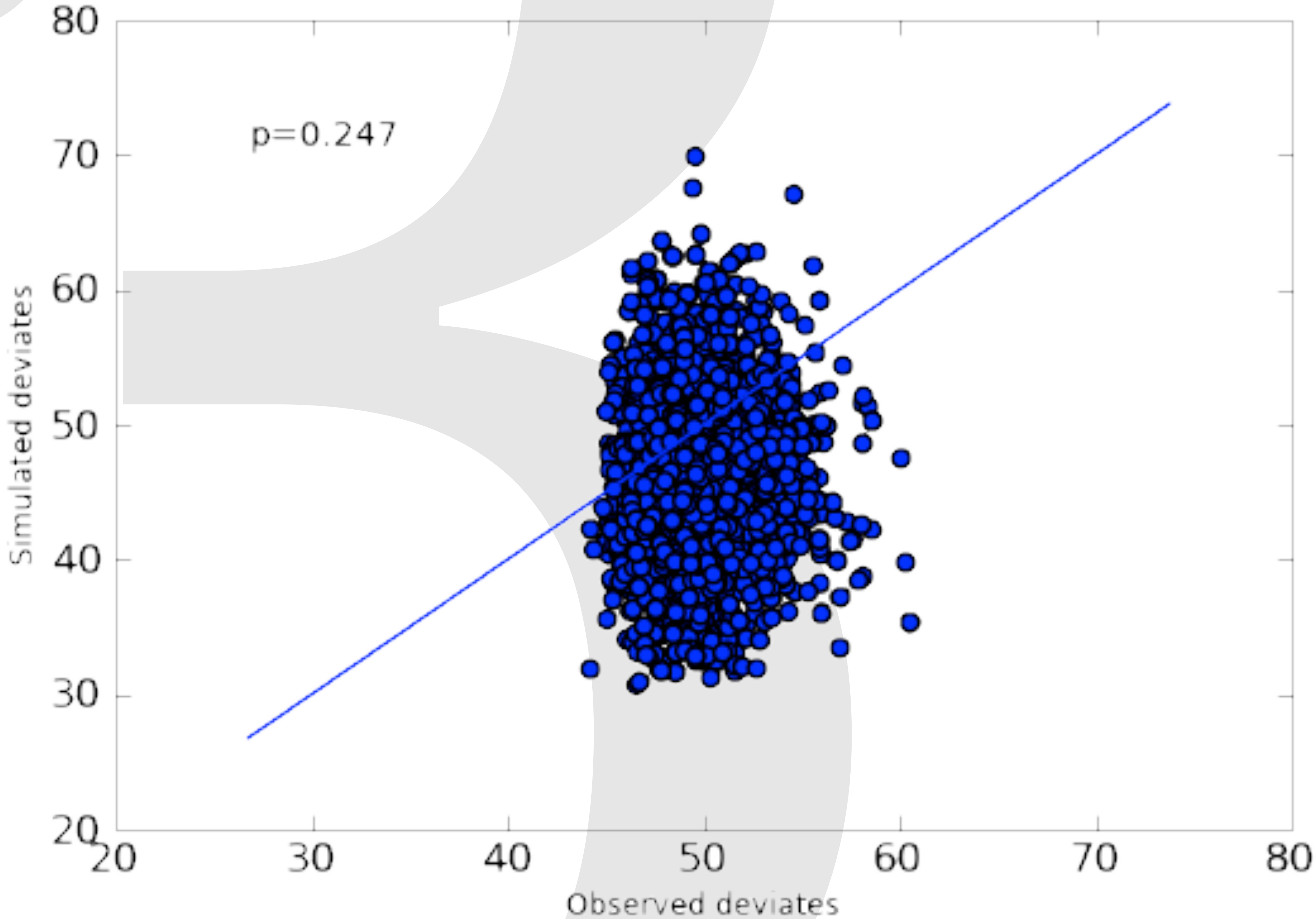
Specify full probability model

$$Pr(y|\theta)Pr(\theta|\phi)Pr(\phi)$$

Calculate
posterior distribution

$$Pr(\theta|y)$$

Check model for lack of fit

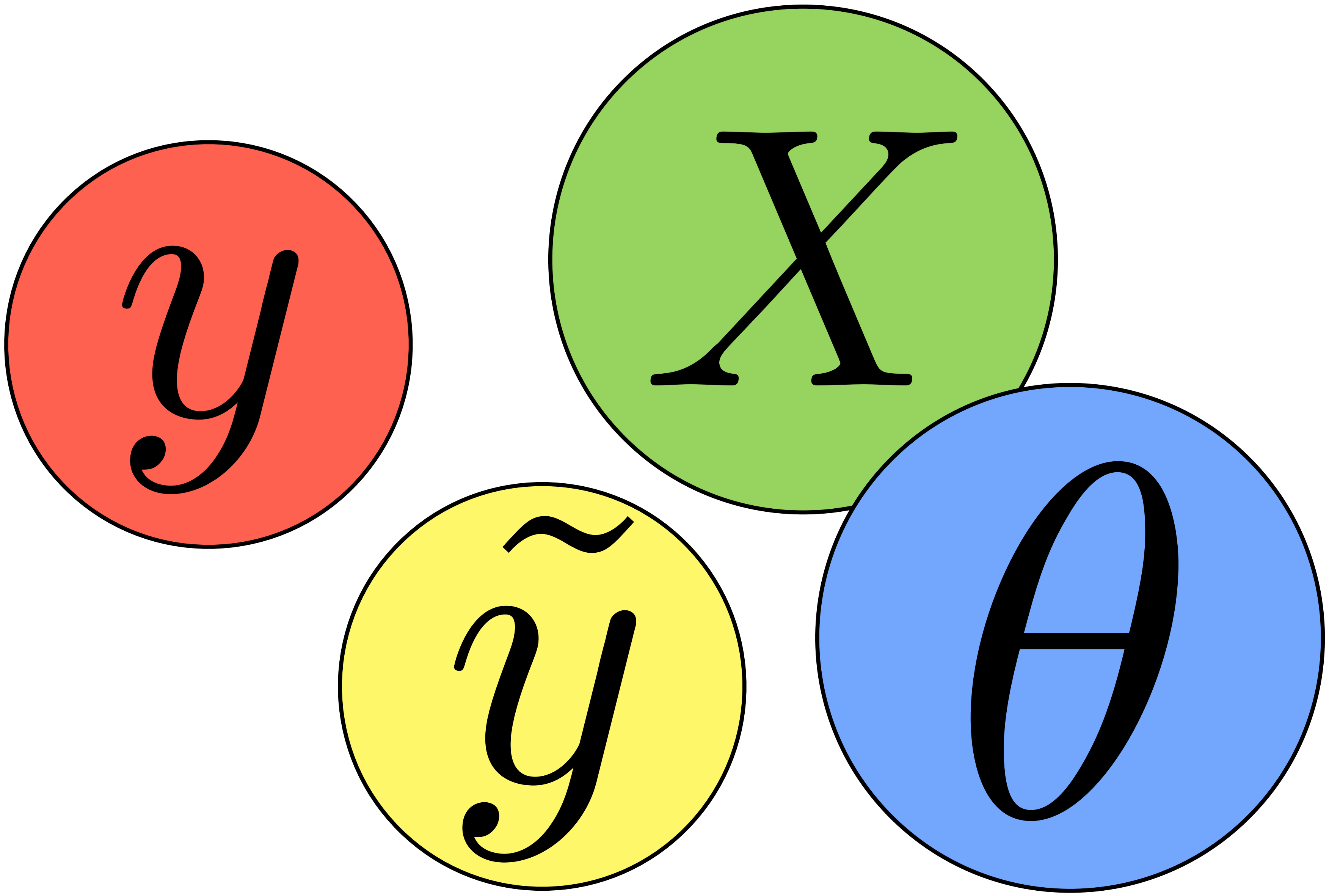


Why Bayes?

“... the Bayesian approach is attractive because it is useful. Its usefulness derives in large measure from its simplicity. Its simplicity allows the investigation of far more complex models than can be handled by the tools in the classical toolbox.”

Link and Barker (2010)

coherency



Interpretation

$$Pr(a(Y) < \theta < b(Y)|\theta) = 0.95$$

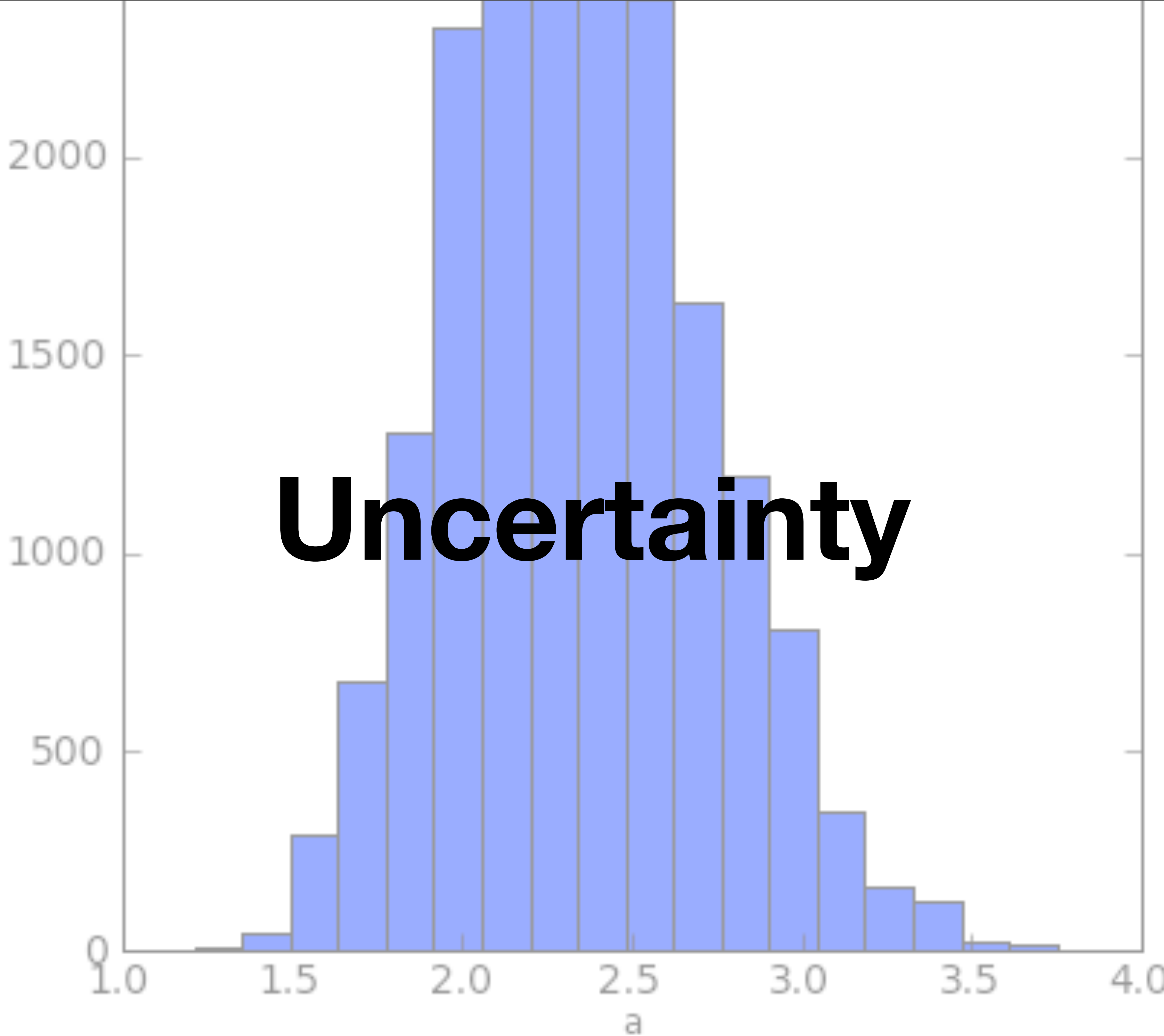
Confidence Interval

$$\Pr(\bar{Y} - 1.96\frac{\sigma}{\sqrt{n}} < \mu < \bar{Y} + 1.96\frac{\sigma}{\sqrt{n}}) = 0.95$$

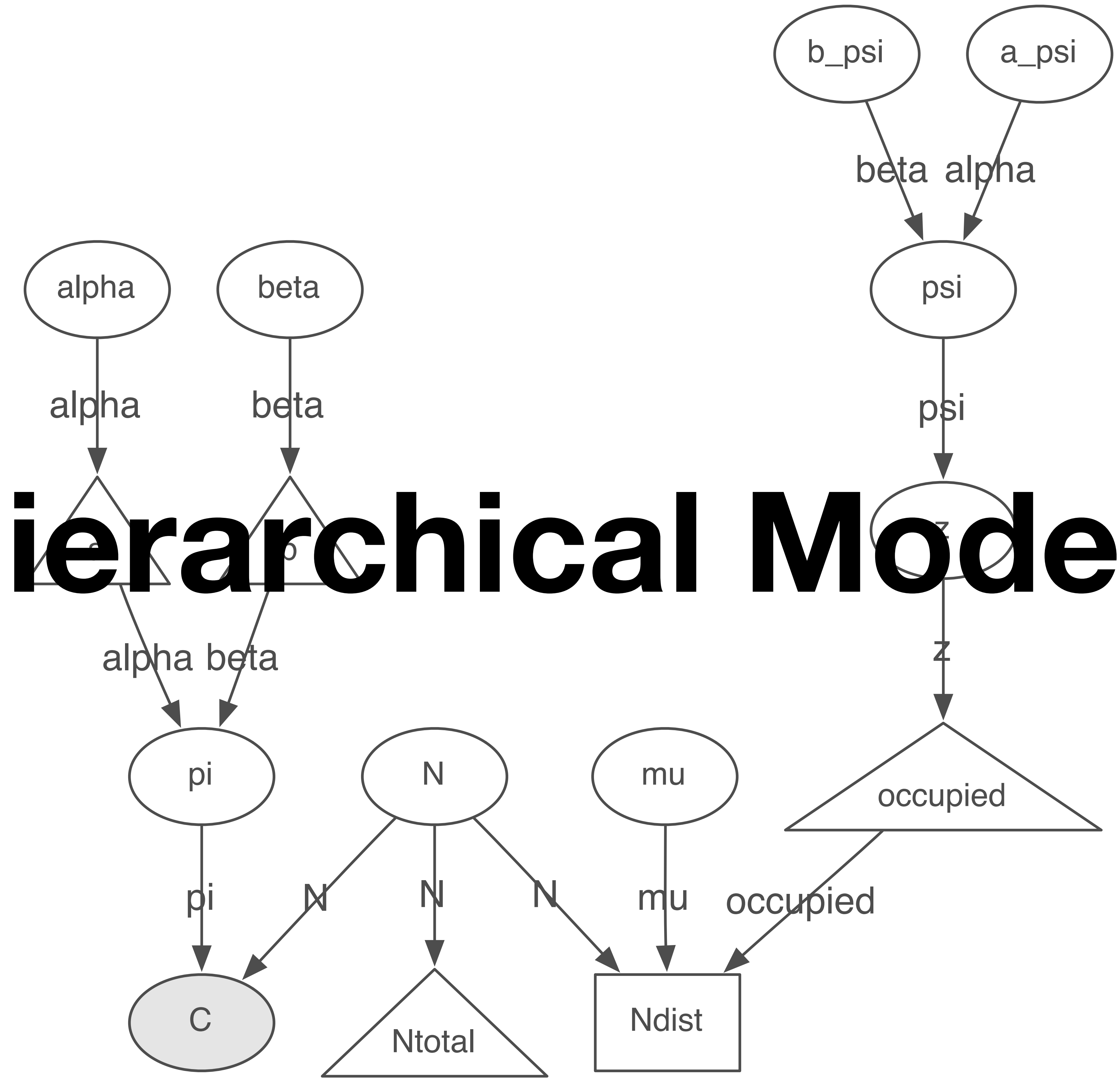
$$Pr(a(y) < \theta < b(y) | Y = y) = 0.95$$

Credible Interval

Uncertainty

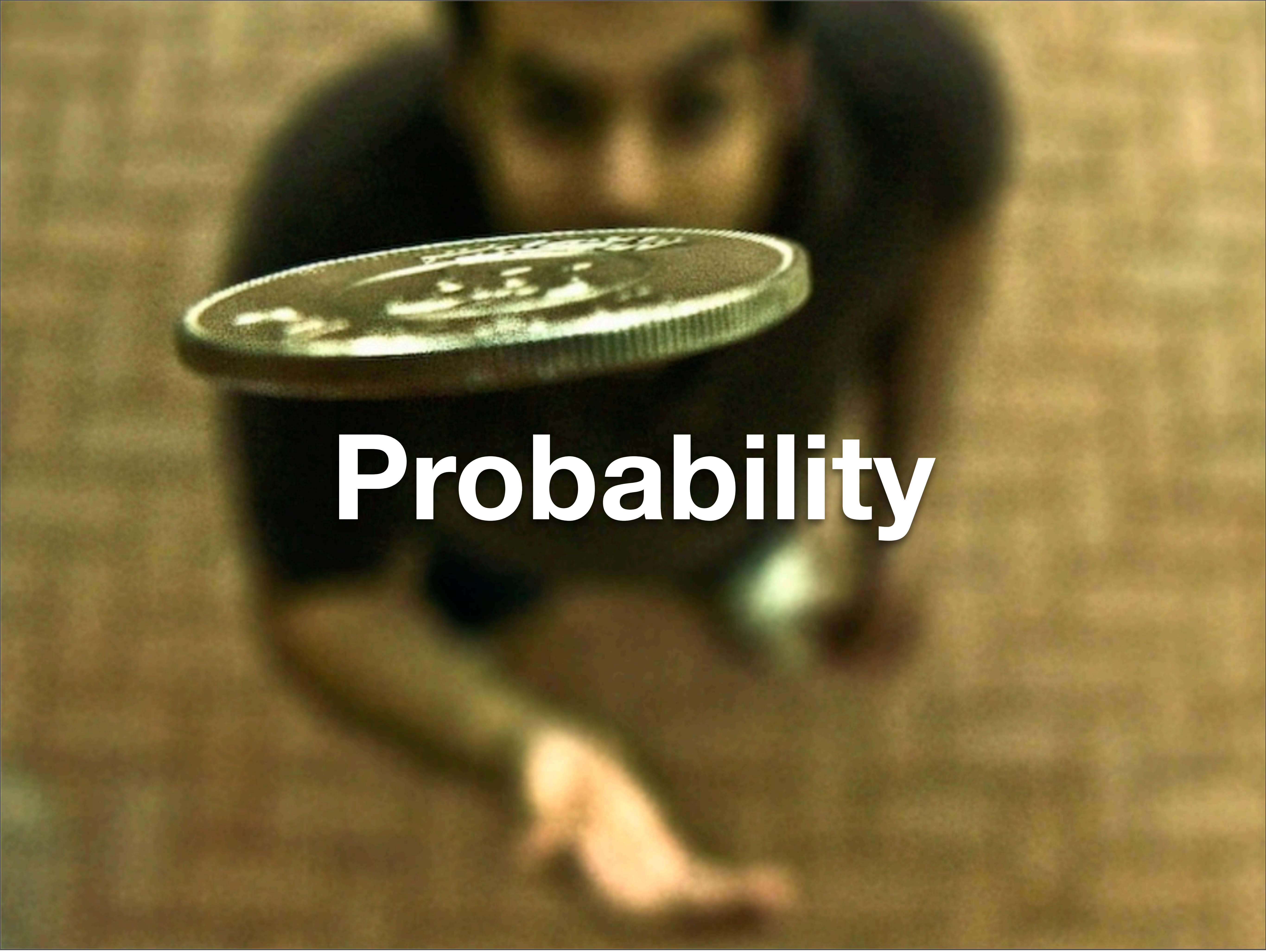


Hierarchical Models



“Bayesian methods constitute a radically different way of doing science ... Bayesians categorically reject various tenets of statistics and the scientific method that are currently widely accepted in ecology and other sciences.”

Dennis (1996)


Probability

(1) classical

$$\Pr(A) = \frac{m}{n}$$

$$\begin{aligned} A &= \text{an event of interest} \\ m &= \text{no. of favourable outcomes} \\ n &= \text{total no. of possible outcomes} \end{aligned}$$

(2) frequentist

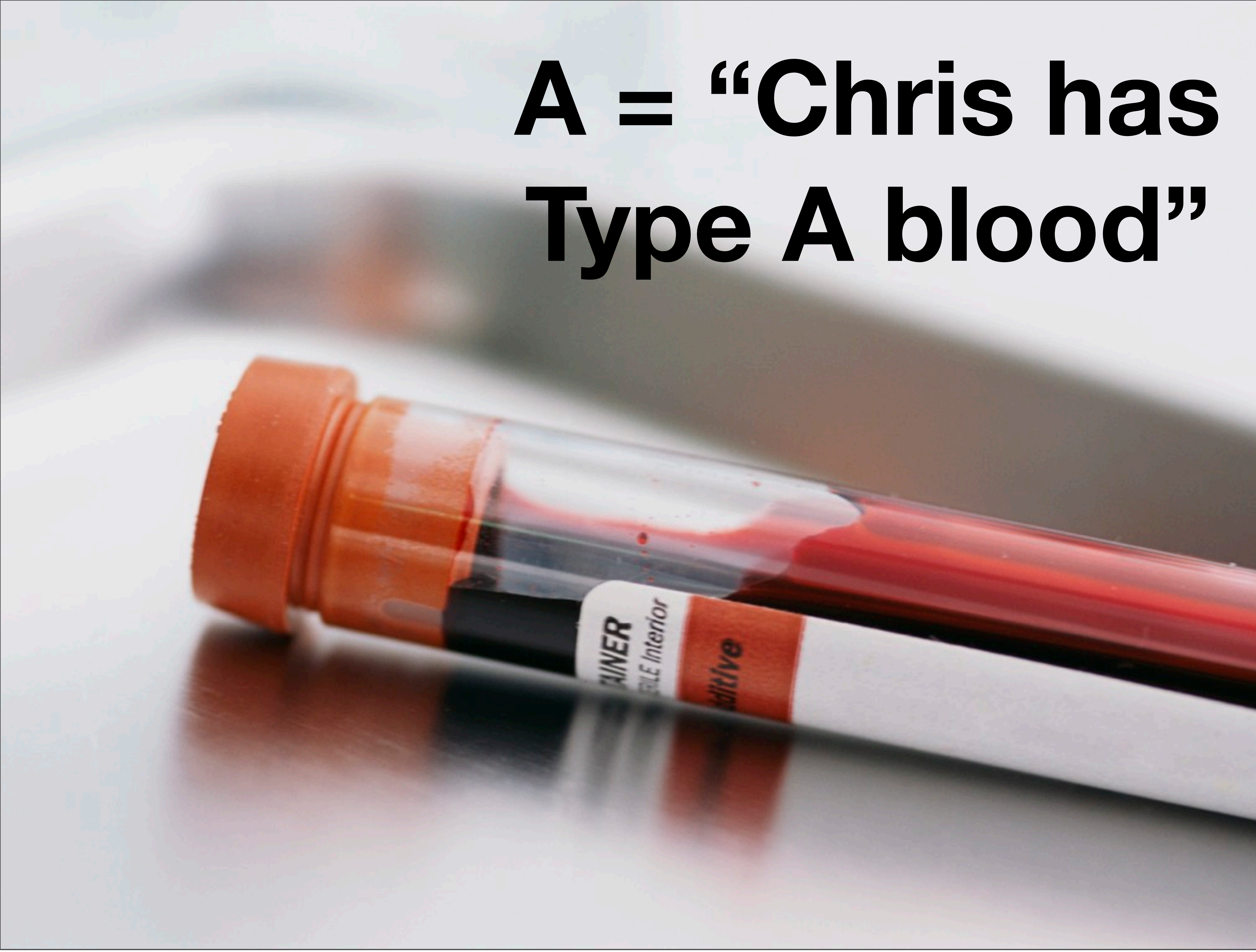
$$\Pr(A) = \lim_{n\to\infty} \frac{m}{n}$$

$$\begin{aligned} n &= \text{no. of identical and independent trials} \\ m &= \text{no. of times A has occurred} \end{aligned}$$

Le Cams Seventh Principle:

"If you need to use asymptotic arguments do not forget to let your number of observations tend to infinity"

A = “Chris has Type A blood”



A = “Vanderbilt will win 2012 national championship”

A = “The prevalence of diabetes in Austin is > 0.15”

(3) subjective

$$\Pr(A)$$

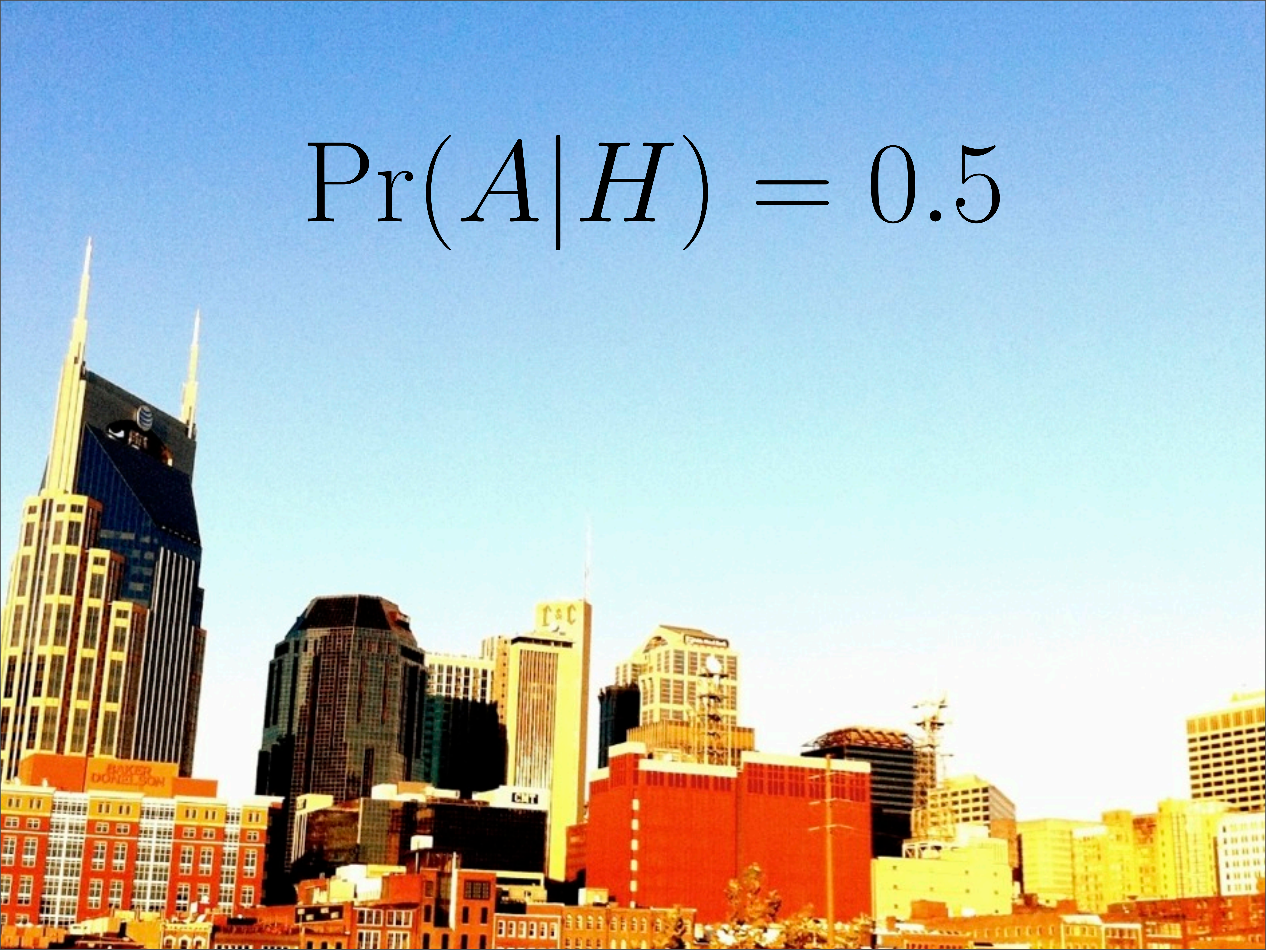
Measure of one's uncertainty regarding the occurrence of A

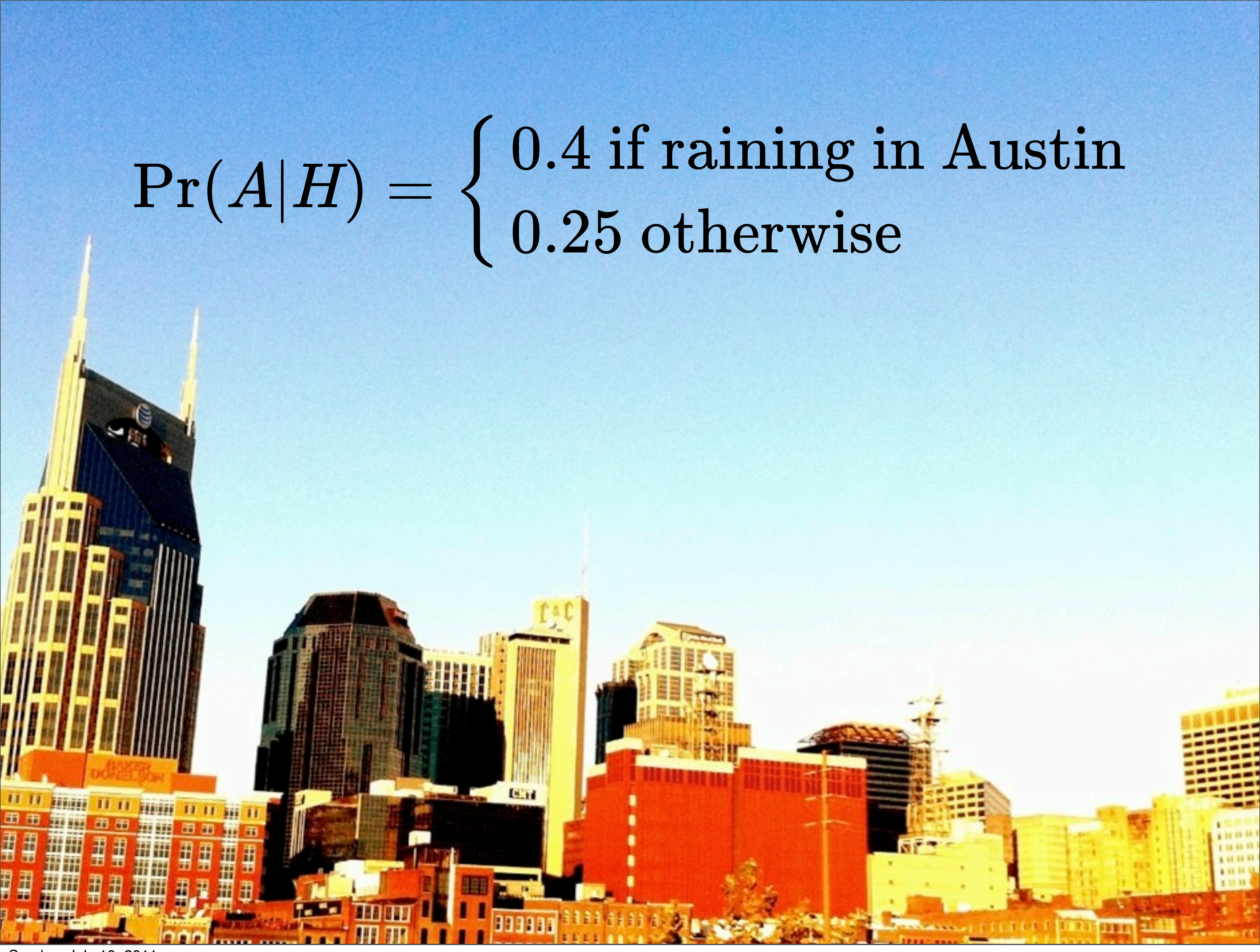
$$\Pr(A)$$

$$\Pr(A|H)$$

A = “It is raining in Nashville”

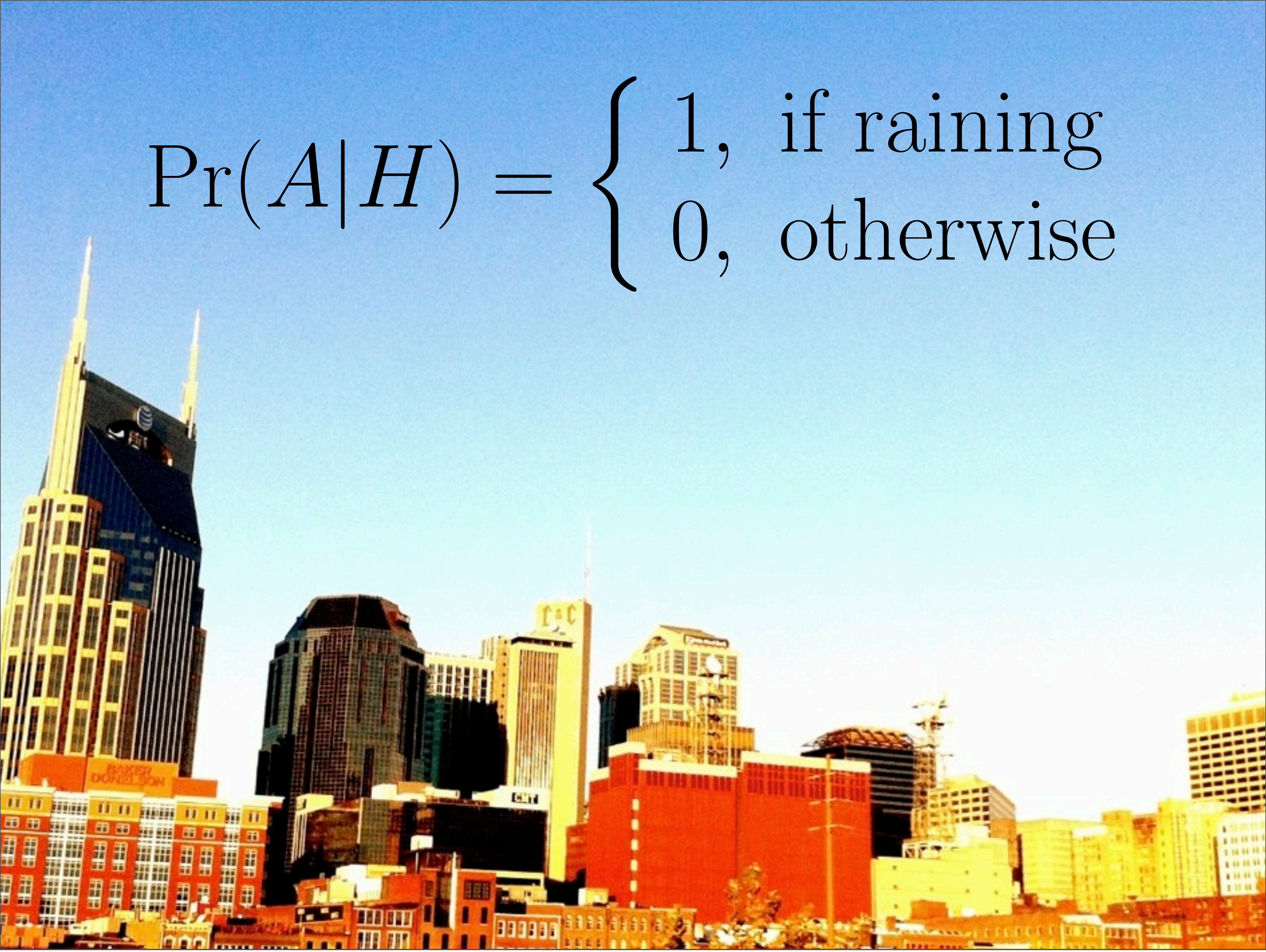
$$\Pr(A|H) = 0.5$$





$$\Pr(A|H) = \begin{cases} 0.4 \text{ if raining in Austin} \\ 0.25 \text{ otherwise} \end{cases}$$

$$\Pr(A|H) = \begin{cases} 1, & \text{if raining} \\ 0, & \text{otherwise} \end{cases}$$



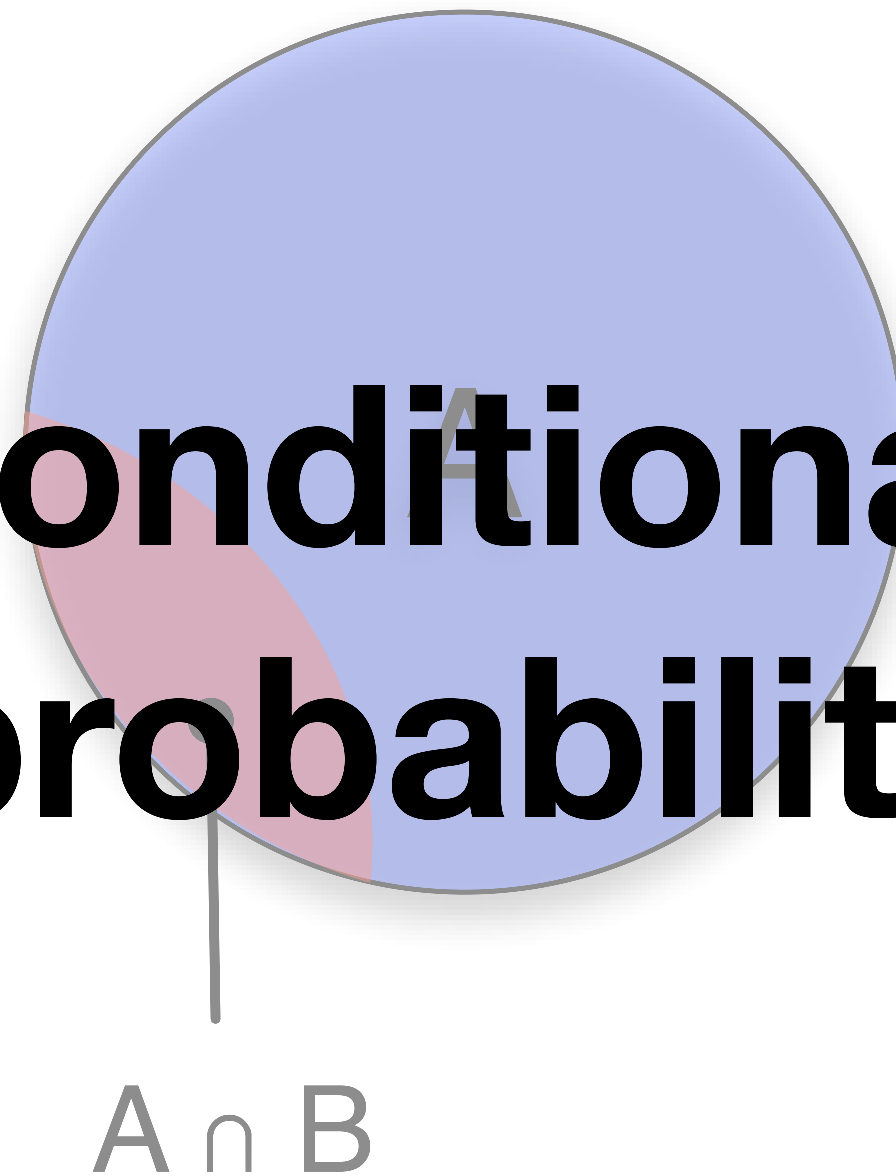
"number of times in which an unknown event has happened and failed"

"its happening in a single trial"

$$Pr(a < \theta < b | y) = p$$

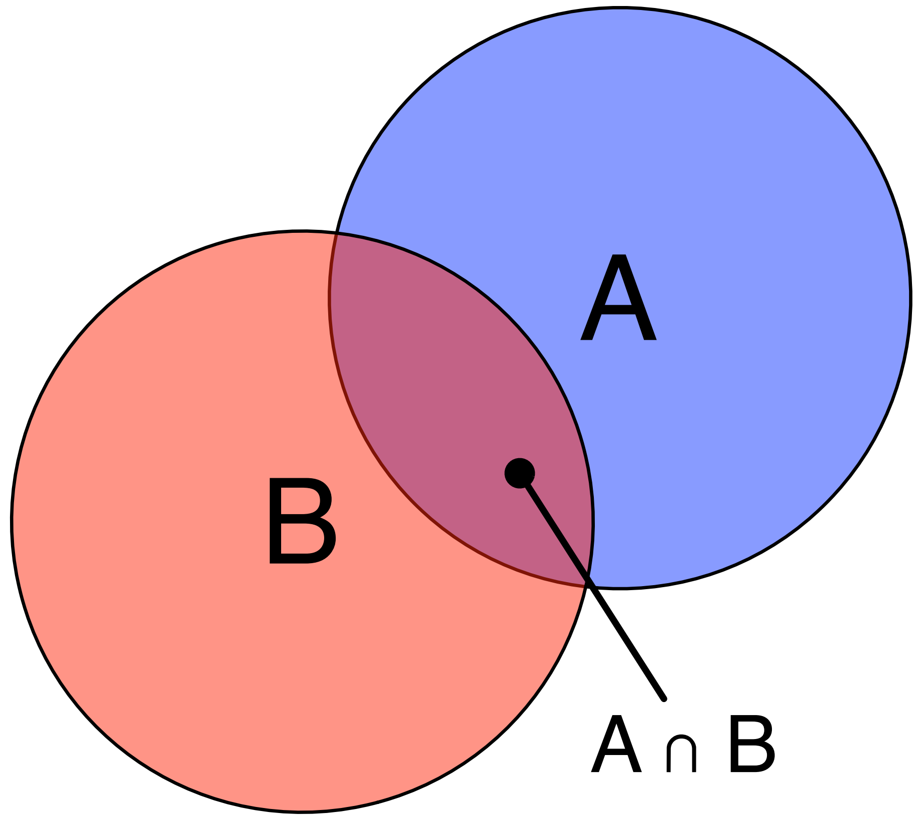
"two degrees of probability"

conditional probability

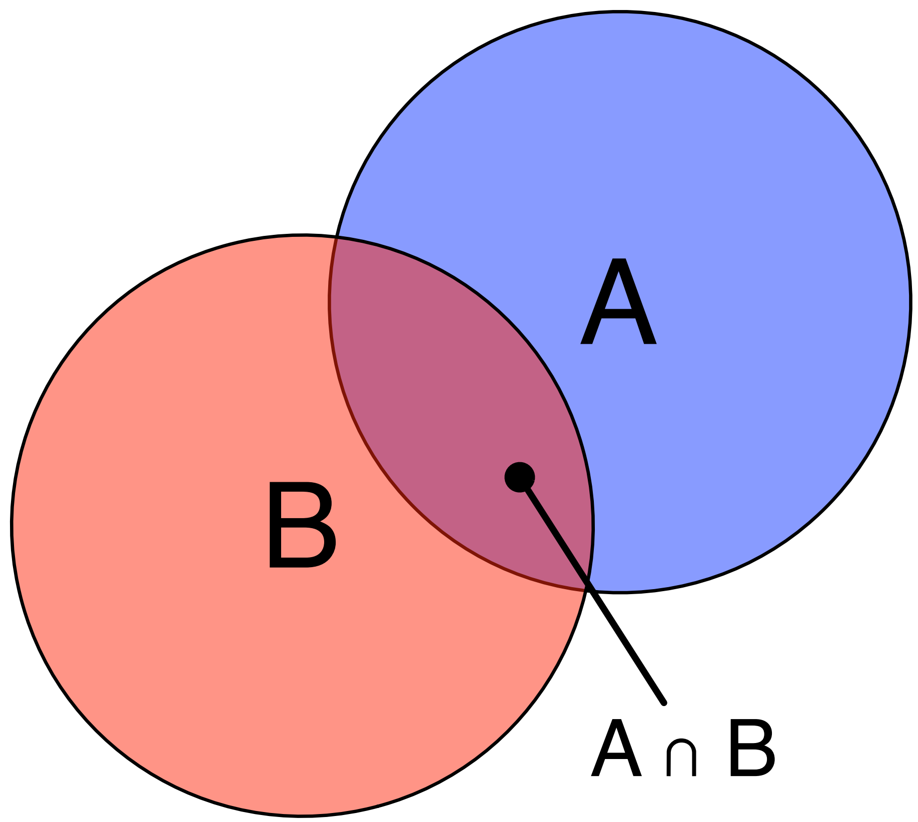


$$Pr(B|A) = \frac{Pr(A \cap B)}{Pr(A)}$$

$$Pr(B|A) = \frac{Pr(A \cap B)}{Pr(A)}$$



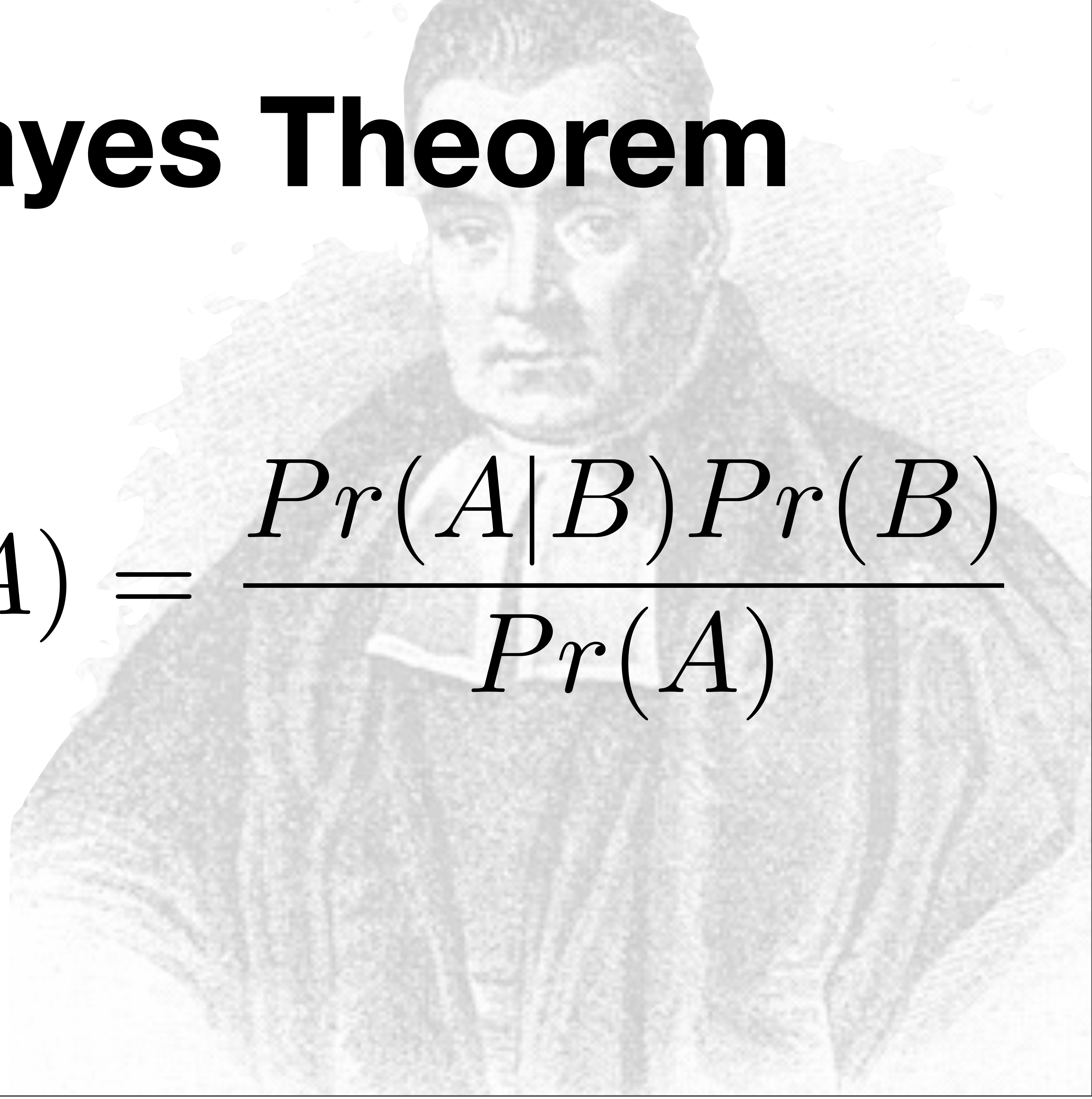
$$Pr(B|A) = \frac{Pr(A \cap B)}{Pr(A)}$$



$$Pr(A|B) = \frac{Pr(A \cap B)}{Pr(B)}$$

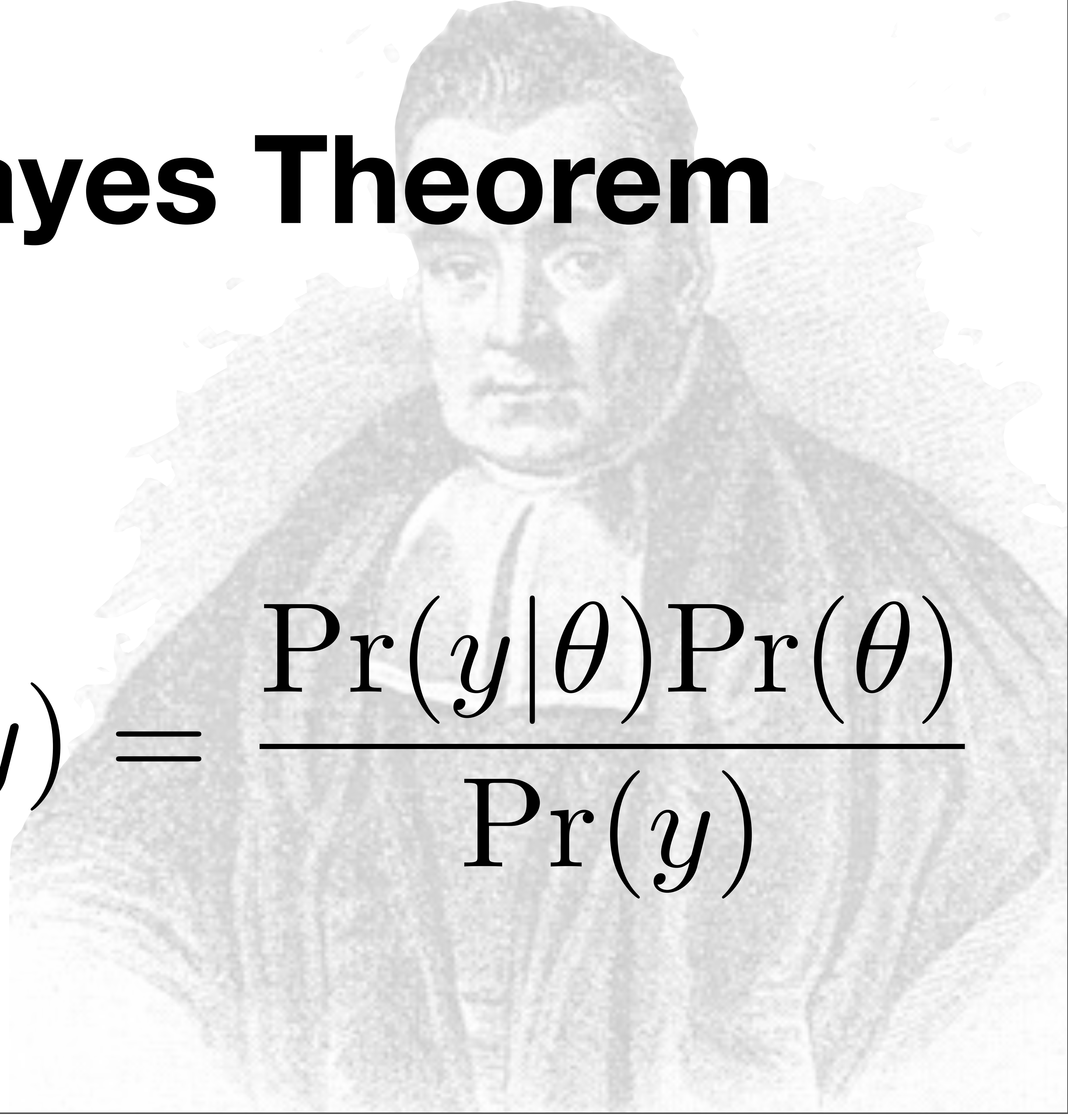
$$
\begin{aligned}
Pr(A \cap B) &= Pr(A|B)Pr(B) \\
&= Pr(B|A)Pr(A)
\end{aligned}
$$

Bayes Theorem



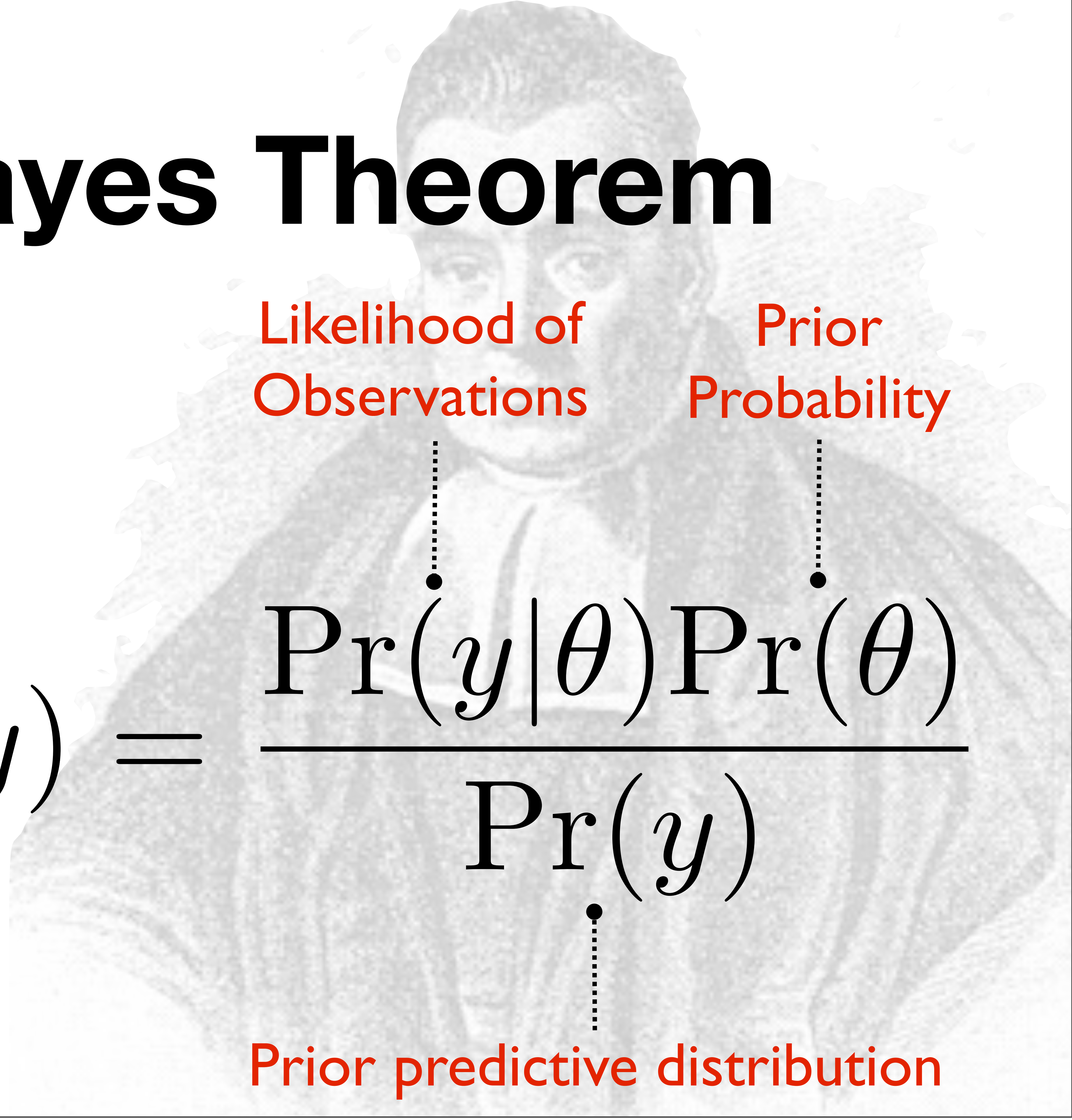
$$Pr(B|A) = \frac{Pr(A|B)Pr(B)}{Pr(A)}$$

Bayes Theorem



$$\Pr(\theta|y) = \frac{\Pr(y|\theta)\Pr(\theta)}{\Pr(y)}$$

Bayes Theorem



Posterior Probability

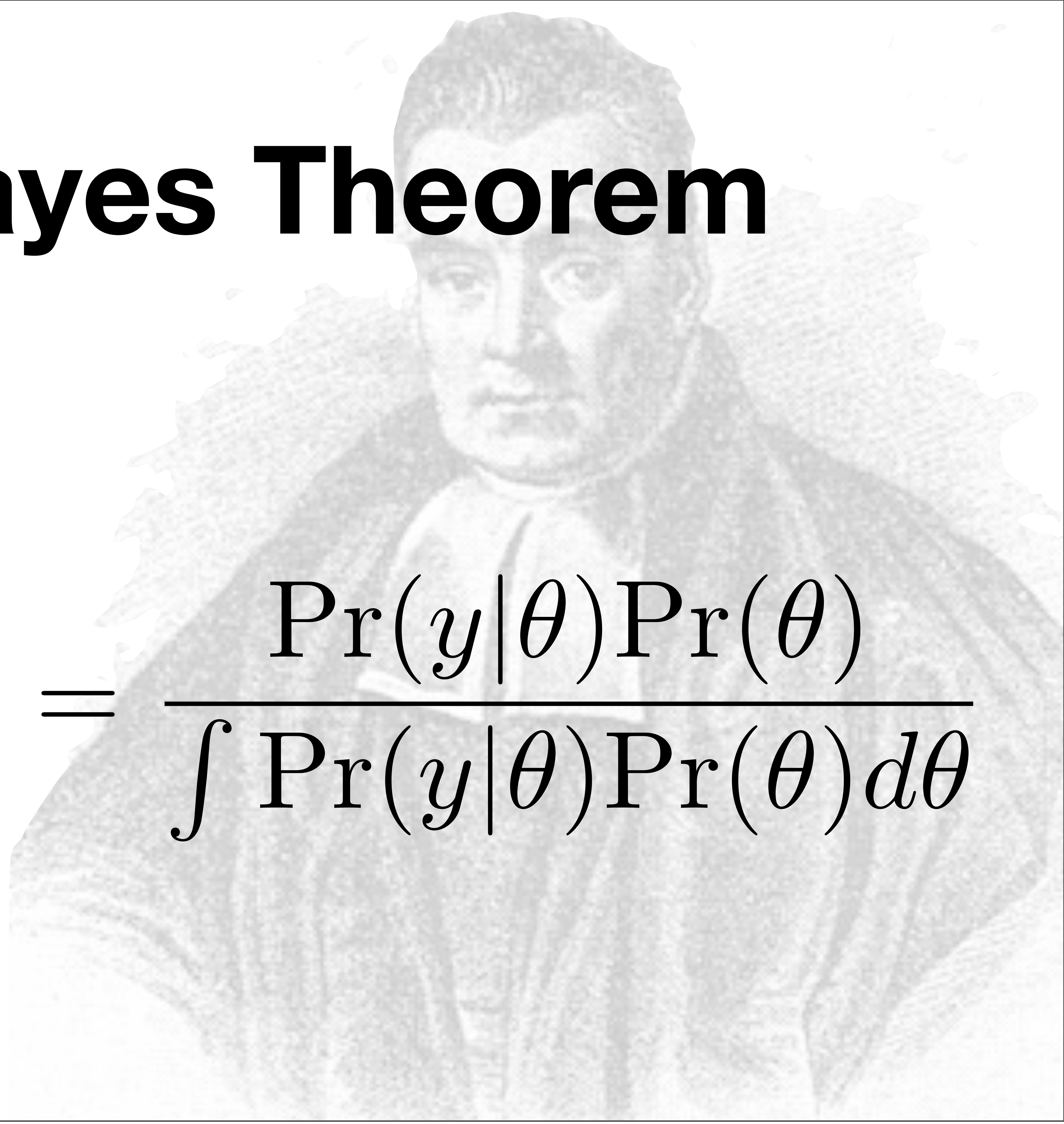
Likelihood of Observations

Prior Probability

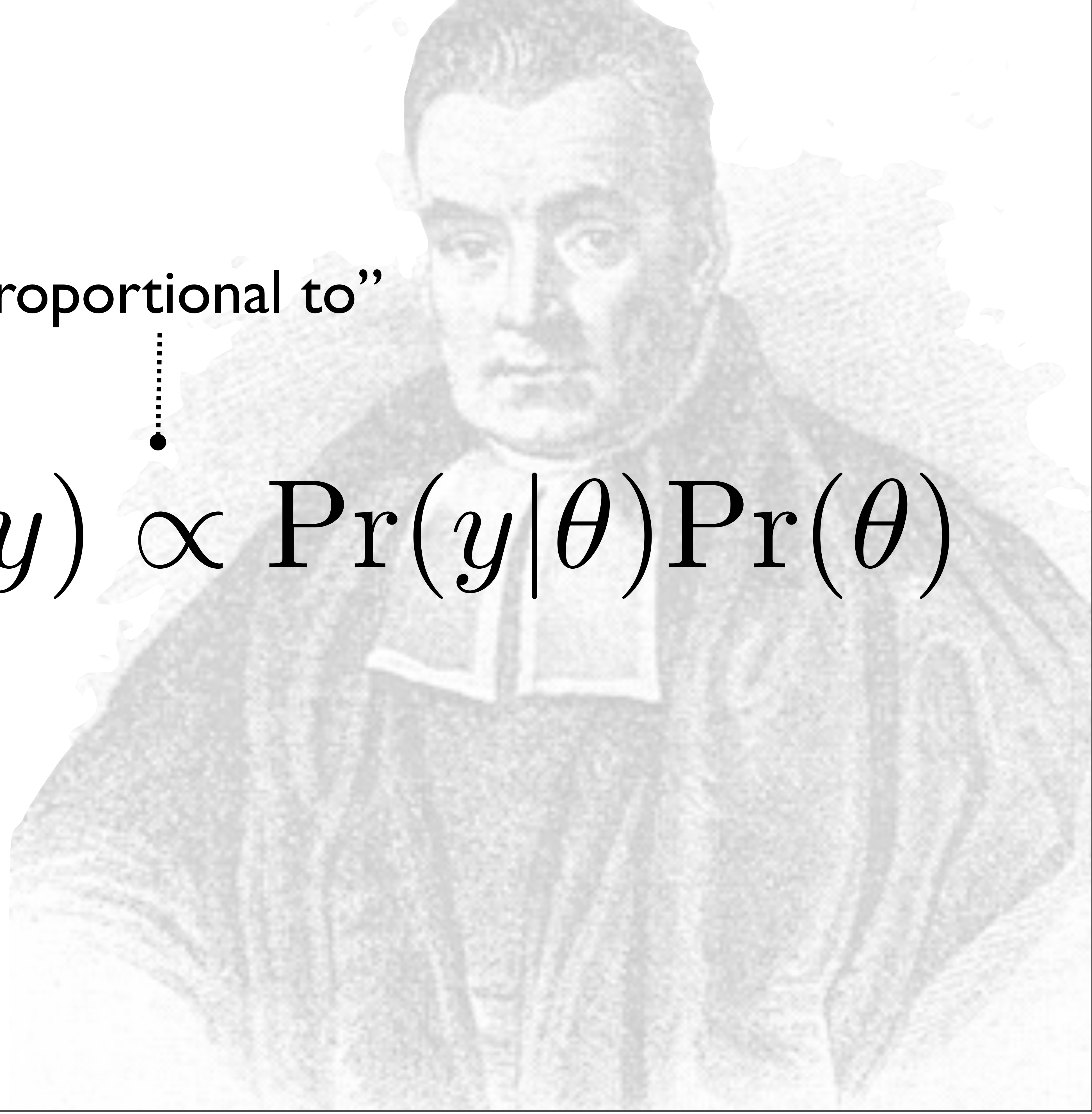
$$\Pr(\theta|y) = \frac{\Pr(y|\theta)\Pr(\theta)}{\Pr(y)}$$

Prior predictive distribution

Bayes Theorem



$$\Pr(\theta|y) = \frac{\Pr(y|\theta)\Pr(\theta)}{\int \Pr(y|\theta)\Pr(\theta)d\theta}$$



“proportional to”

$$\Pr(\theta|y) \propto \Pr(y|\theta)\Pr(\theta)$$

Prediction

$$\Pr(y) = \int \Pr(y|\theta)\Pr(\theta)d\theta$$

Prediction

$$\Pr(y) = \int \Pr(y|\theta)\Pr(\theta)d\theta$$

Prior predictive distribution

Prediction

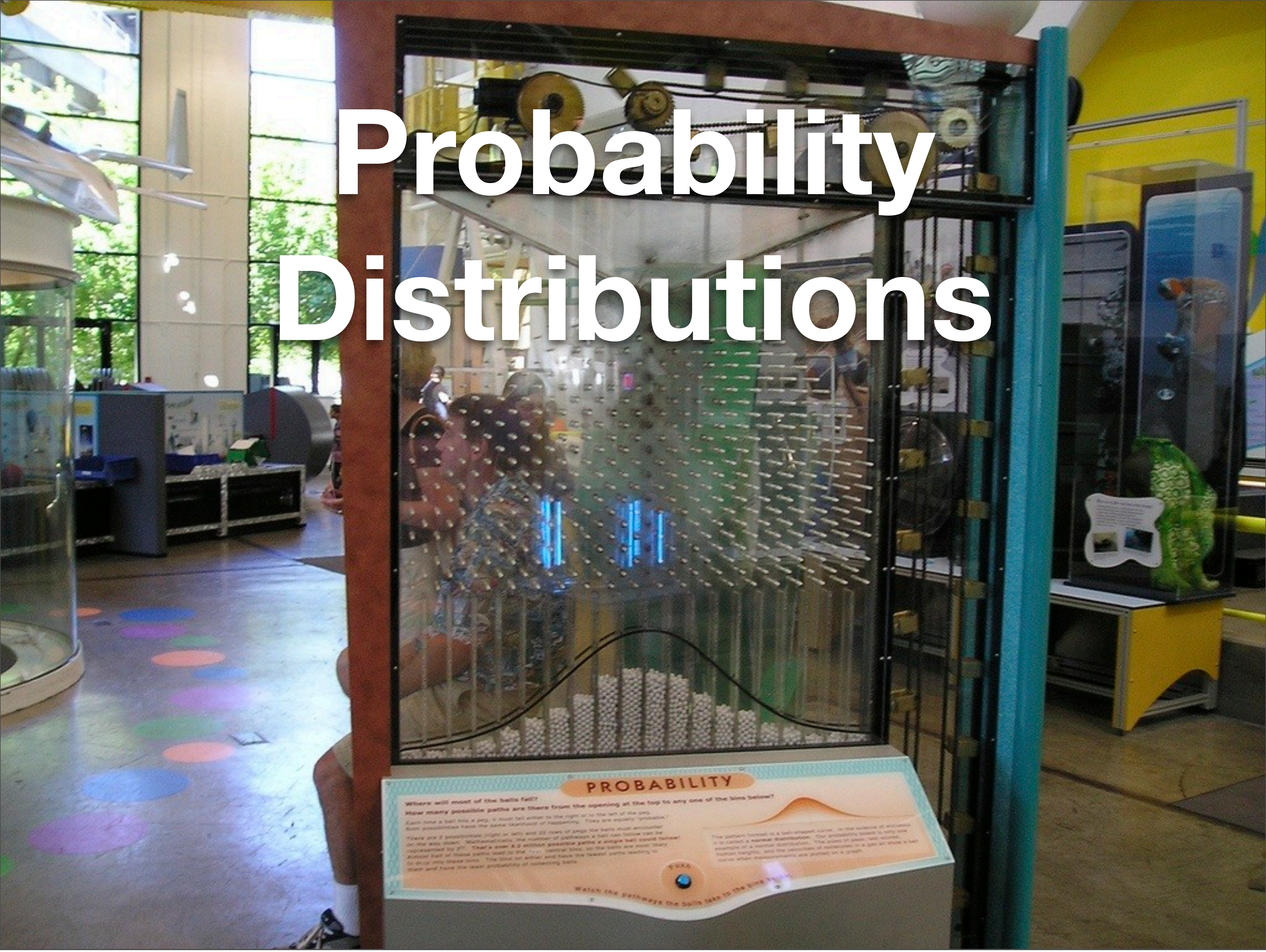
$$
\begin{aligned}
p(\tilde{y}|y) &= \int p(\tilde{y}, \theta|y) d\theta \\
&= \int p(\tilde{y}|\theta, y) p(\theta|y) d\theta \\
&= \int p(\tilde{y}|\theta) p(\theta|y) d\theta
\end{aligned}
$$

Prediction

$$
\begin{aligned}
p(\tilde{y}|y) &= \int p(\tilde{y},\theta|y)d\theta \\
&= \int p(\tilde{y}|\theta,y)p(\theta|y)d\theta \\
&= \int p(\tilde{y}|\theta)p(\theta|y)d\theta
\end{aligned}
$$

Posterior predictive distribution

Probability Distributions



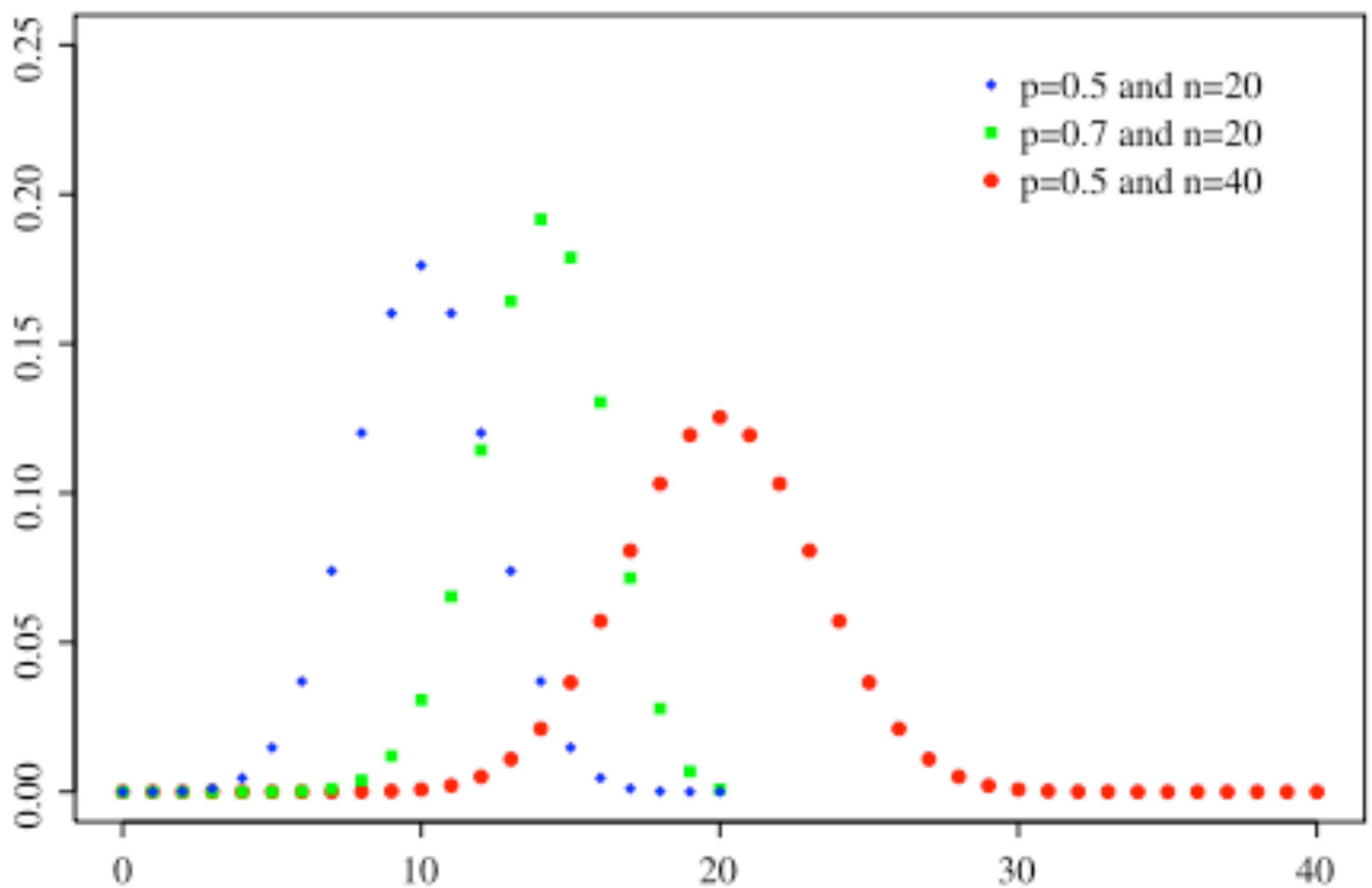
Bernoulli

$$Pr(X = x) = p^x(1 - p)^{1-x}$$

$$X = \{0, 1\}, p \in (0, 1)$$

Binomial

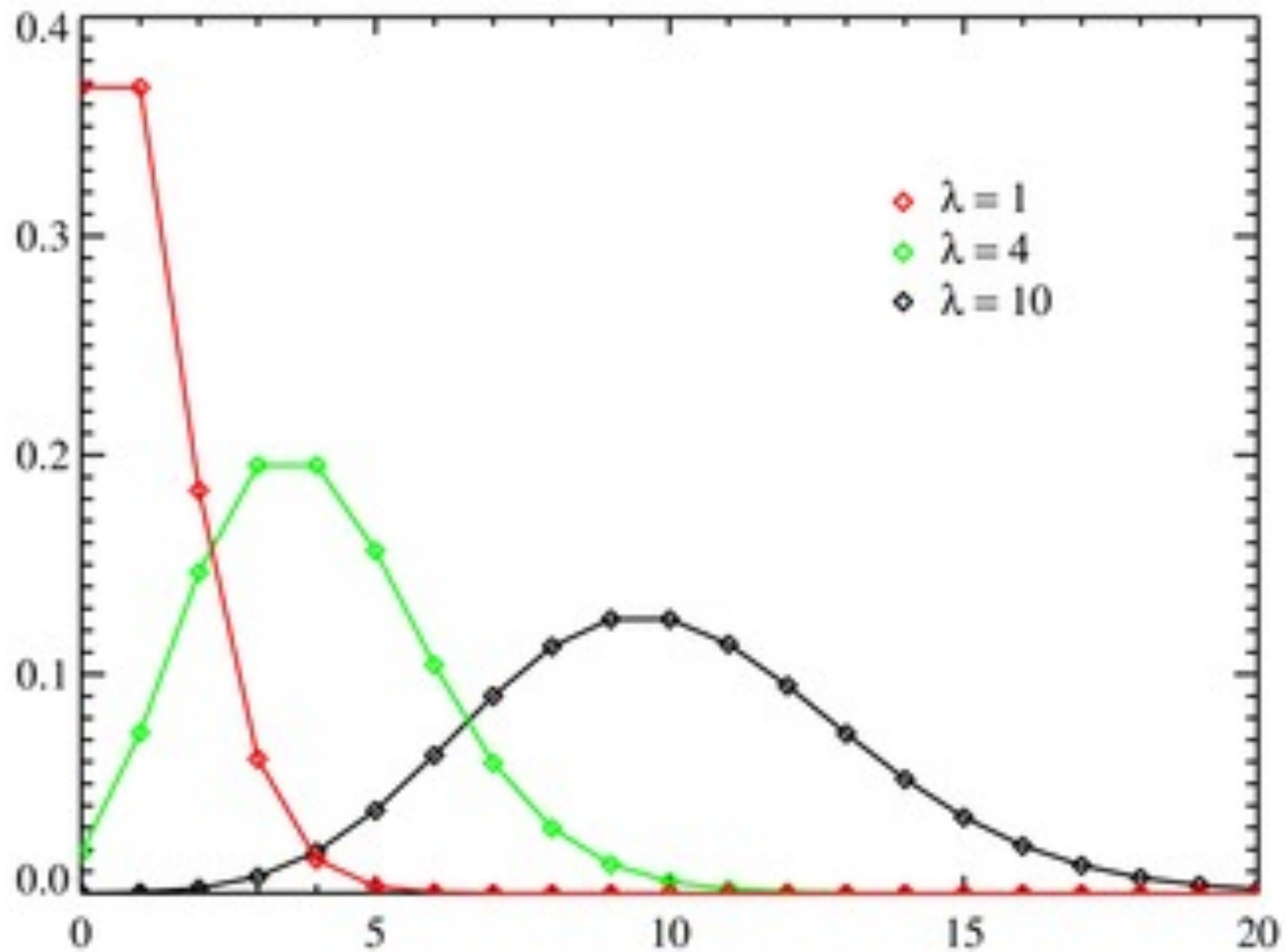
$$Pr(X = x) = \binom{N}{x} p^x (1-p)^{N-x}$$



$$\begin{aligned} X &= \{0, 1, 2, \ldots\} \\ p &\in (0, 1) \end{aligned}$$

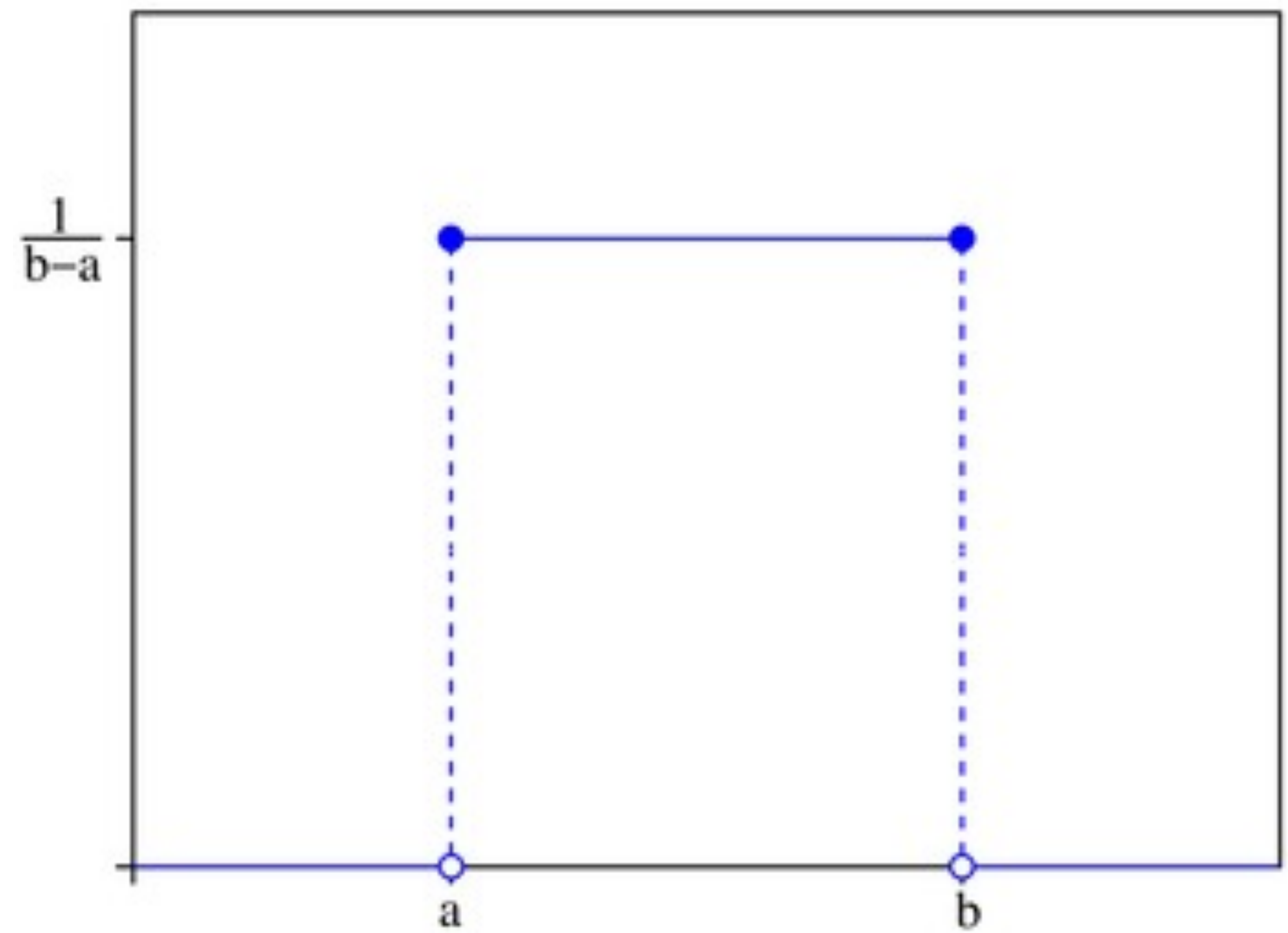
Poisson

$$Pr(X = x) = \frac{e^{-\lambda}\lambda^x}{x!}$$



$$\begin{aligned} X &= \{0, 1, 2, \dots\} \\ \lambda &> 0 \end{aligned}$$

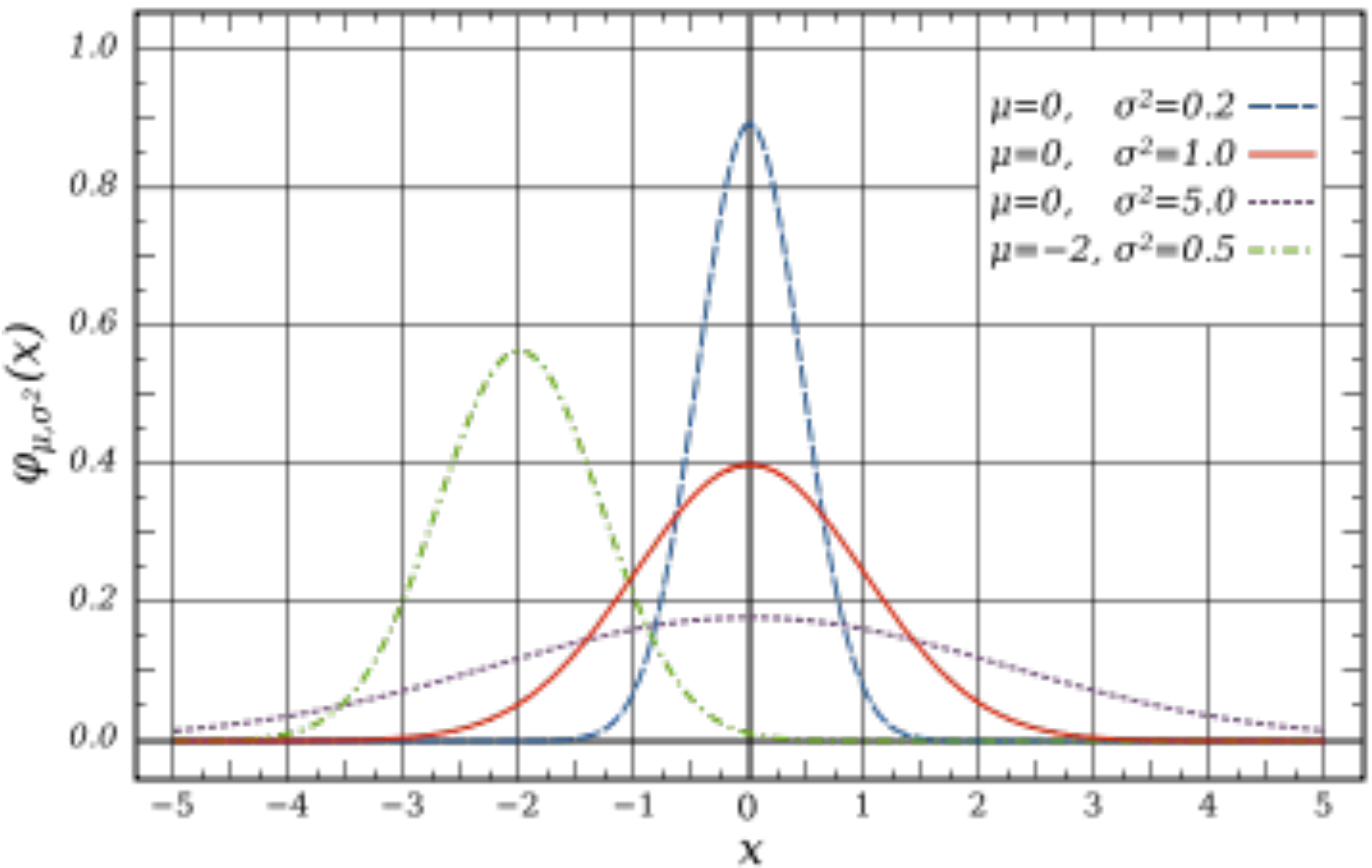
Uniform



$$f(x) = \begin{cases} \frac{1}{b-a} & \text{for } a \le x \le b \\ 0 & \text{otherwise} \end{cases}$$

Normal

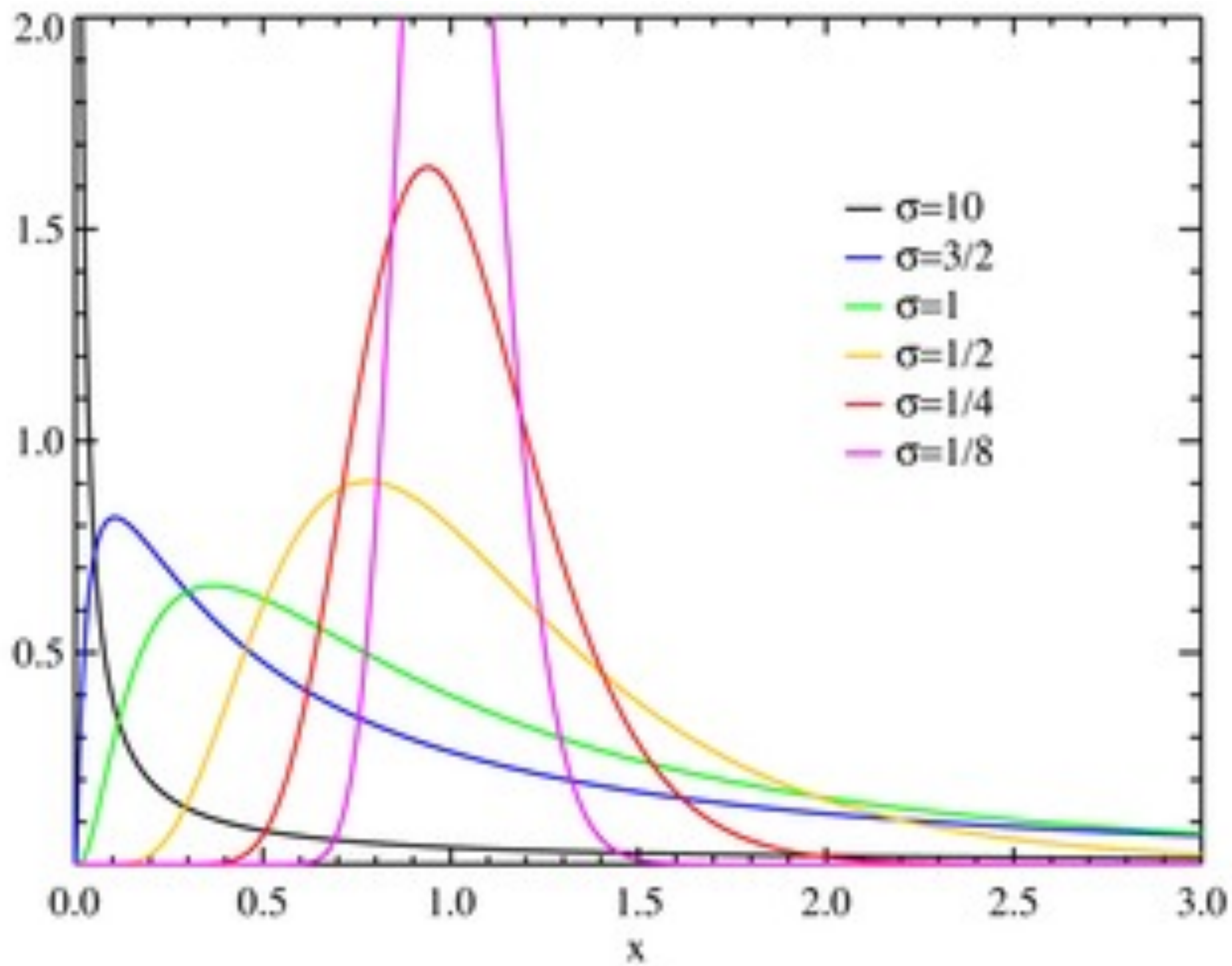
$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right]$$



$$\sigma^2 > 0$$

Log-normal

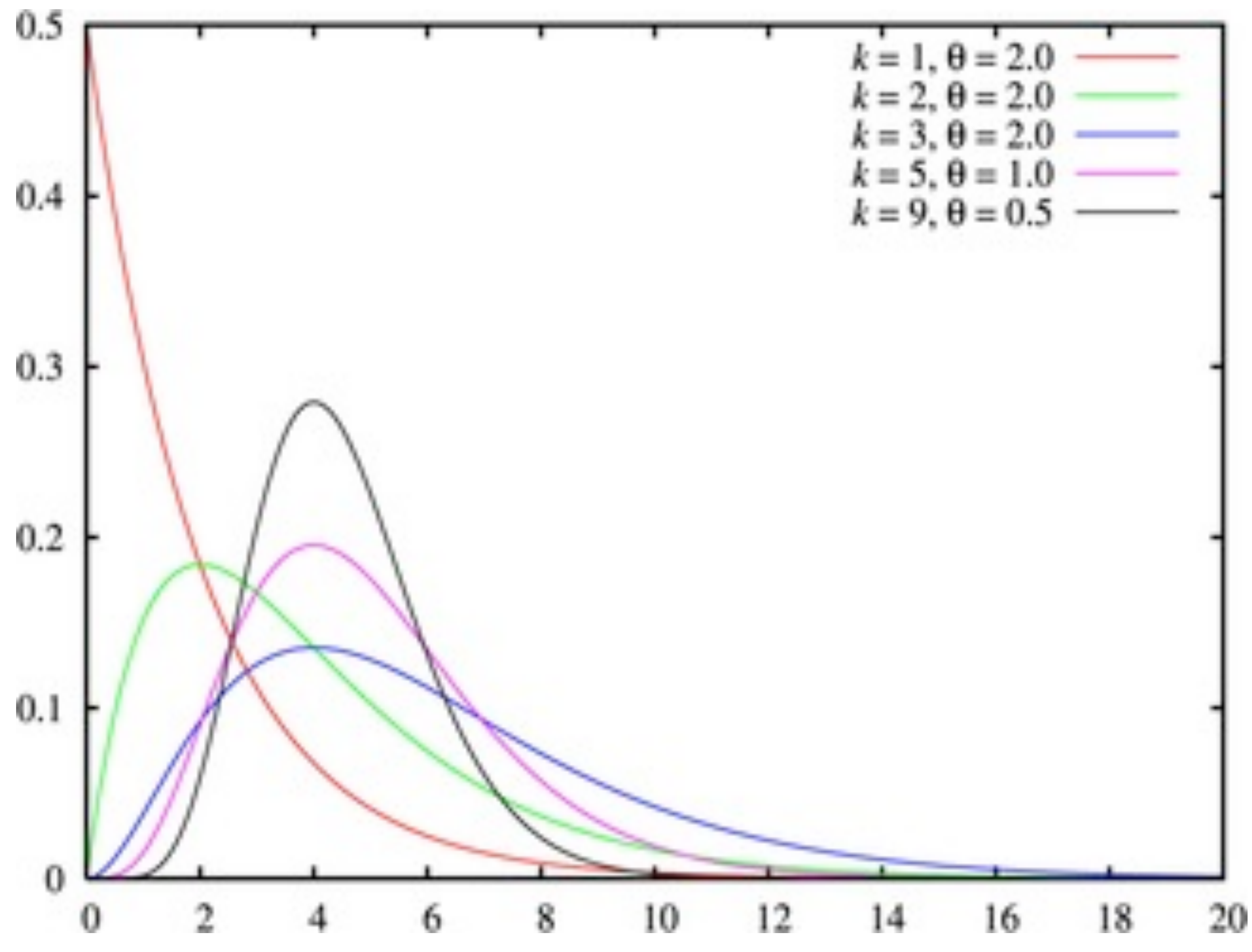
$$f(x) = \frac{1}{x\sqrt{2\pi\sigma^2}} \exp\left[-\frac{(\log(x)-\mu)^2}{2\sigma^2}\right]$$



$$X, \sigma^2 > 0$$

Gamma

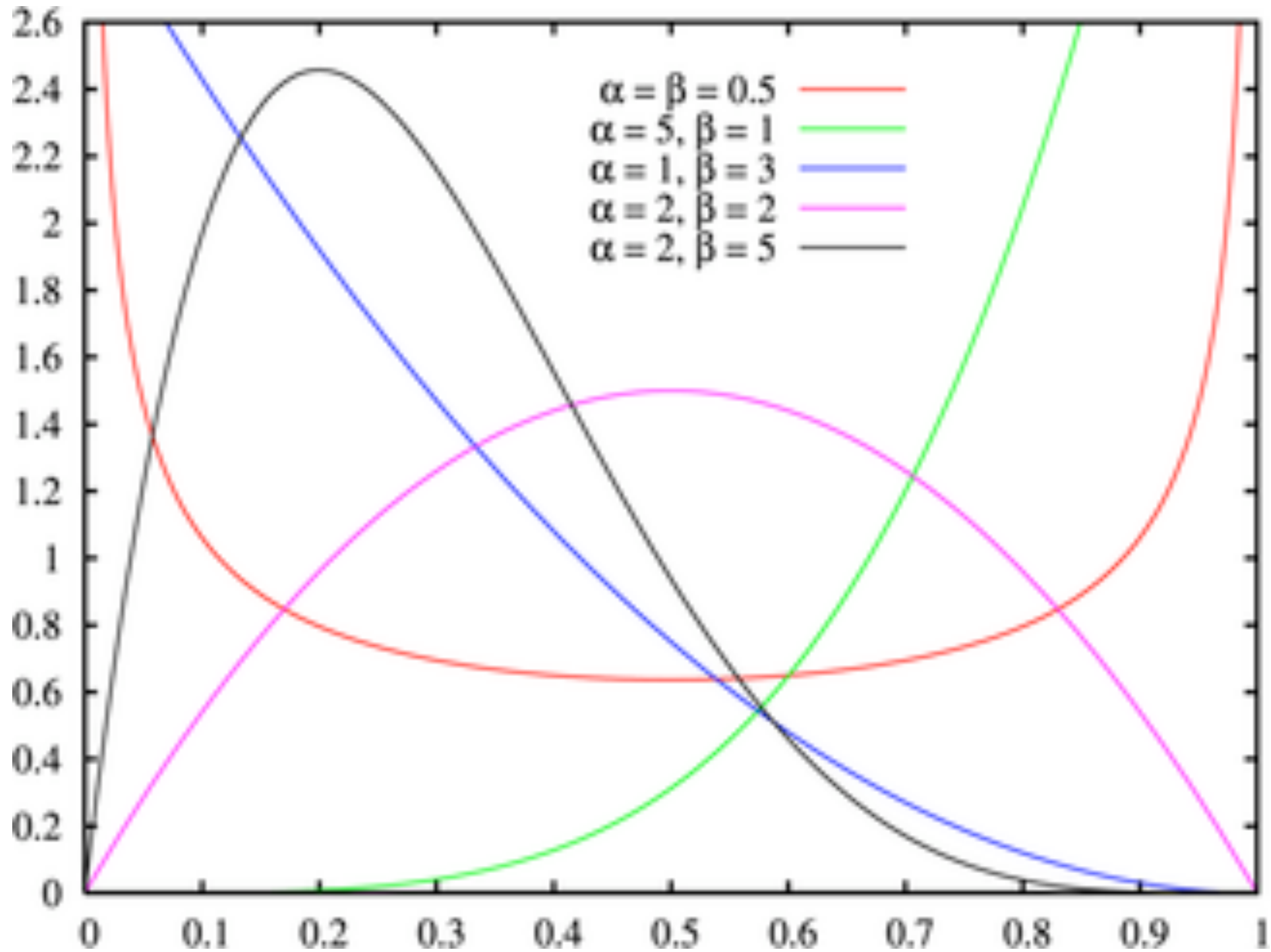
$$f(x) = \frac{e^{-x/\beta} x^{\alpha-1}}{\beta^{\alpha}\Gamma(\alpha)}$$



$$\boxed{X, \alpha, \beta > 0}$$

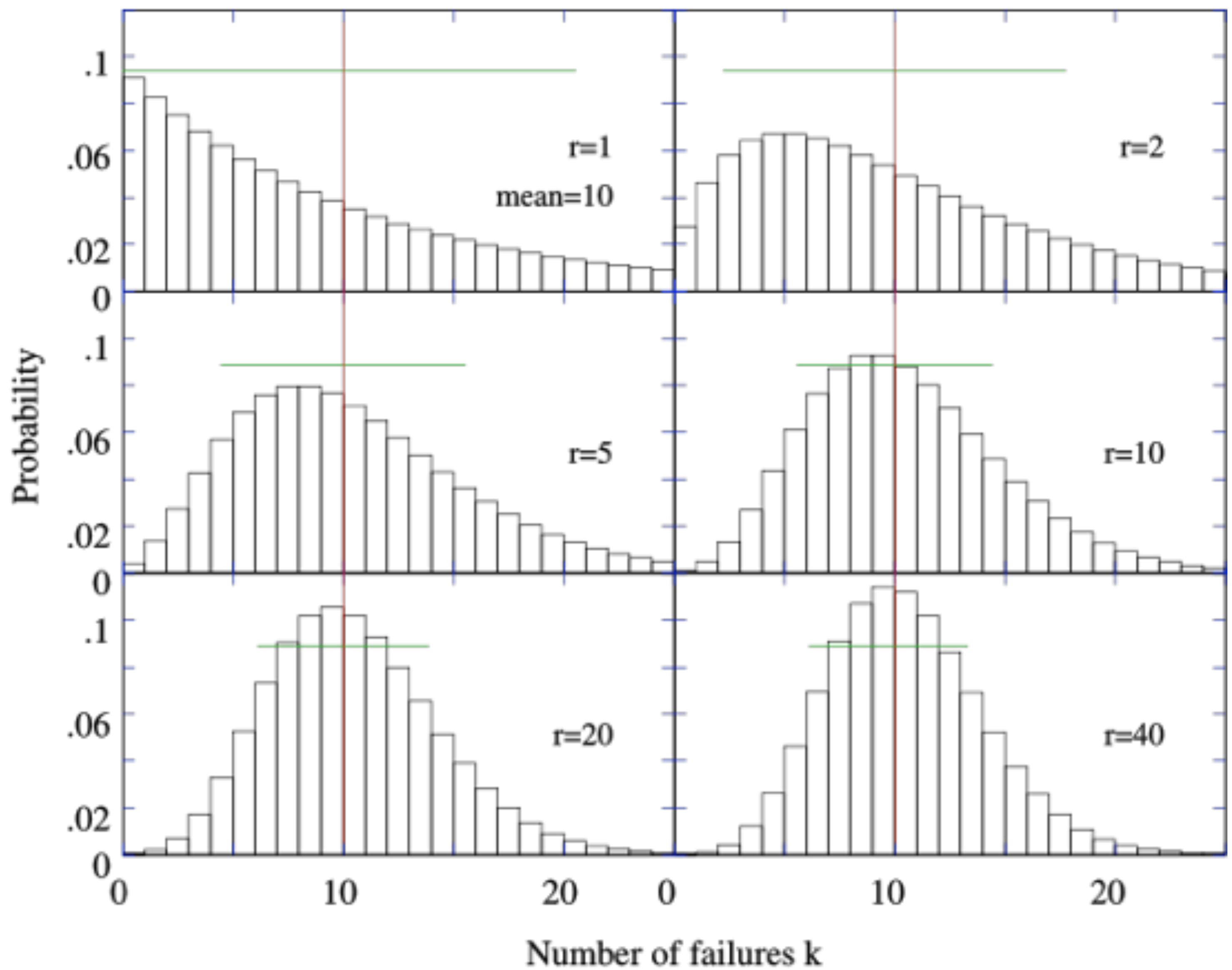
Beta

$$f(x) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1}(1-x)^{\beta-1}$$



$$\begin{aligned} X &\in (0,1) \\ \alpha, \beta &> 0 \end{aligned}$$

Negative Binomial



$$Pr(X = x) = \binom{x + r - 1}{x} p^r (1 - p)^x$$

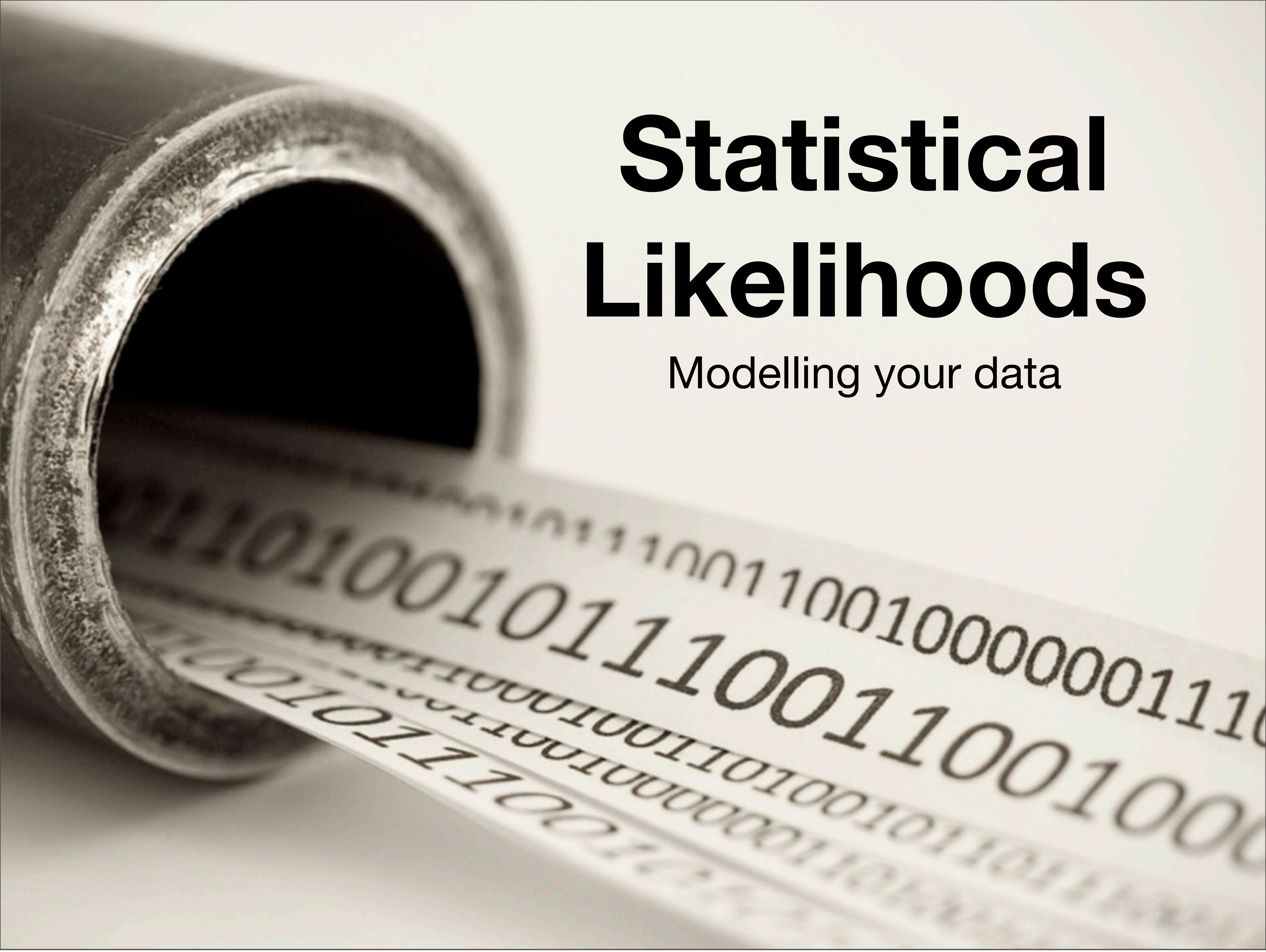
Negative Binomial (2)

$$Pr(X = x) = \frac{\Gamma(k + x)}{\Gamma(k)x!} \left(\frac{k}{k + m} \right)^k \left(\frac{m}{k + m} \right)^x$$

$$\begin{aligned} X &\sim \text{Poisson}(m) \\ m &\sim \text{Gamma}\left(r, \frac{1-p}{p} \right) \end{aligned}$$

Statistical Likelihoods

Modelling your data



binomial model

parameter

$$p(X|\theta) = \binom{N}{n} \theta^x (1-\theta)^{N-x}$$

data

sampling distribution of X

binomial model

$$L(\theta|X) = \binom{N}{n} \theta^x (1-\theta)^{N-x}$$

likelihood function for θ

$$Pr(X_i|p) = \binom{N}{x_i} p^{x_i}(1-p)^{N-x_i}$$

$$L(p_m|x) = \binom{N}{x} p_m^x (1-p_m)^{N-x}$$

$$Pr(X_i|p) = \binom{N}{x_i} p^{x_i}(1-p)^{N-x_i}$$

discrete
sums to one

$$L(p_m|x) = \binom{N}{x} p_m^x (1-p_m)^{N-x}$$

$$Pr(X_i|p) = \binom{N}{x_i} p^{x_i} (1-p)^{N-x_i}$$

discrete
sums to one

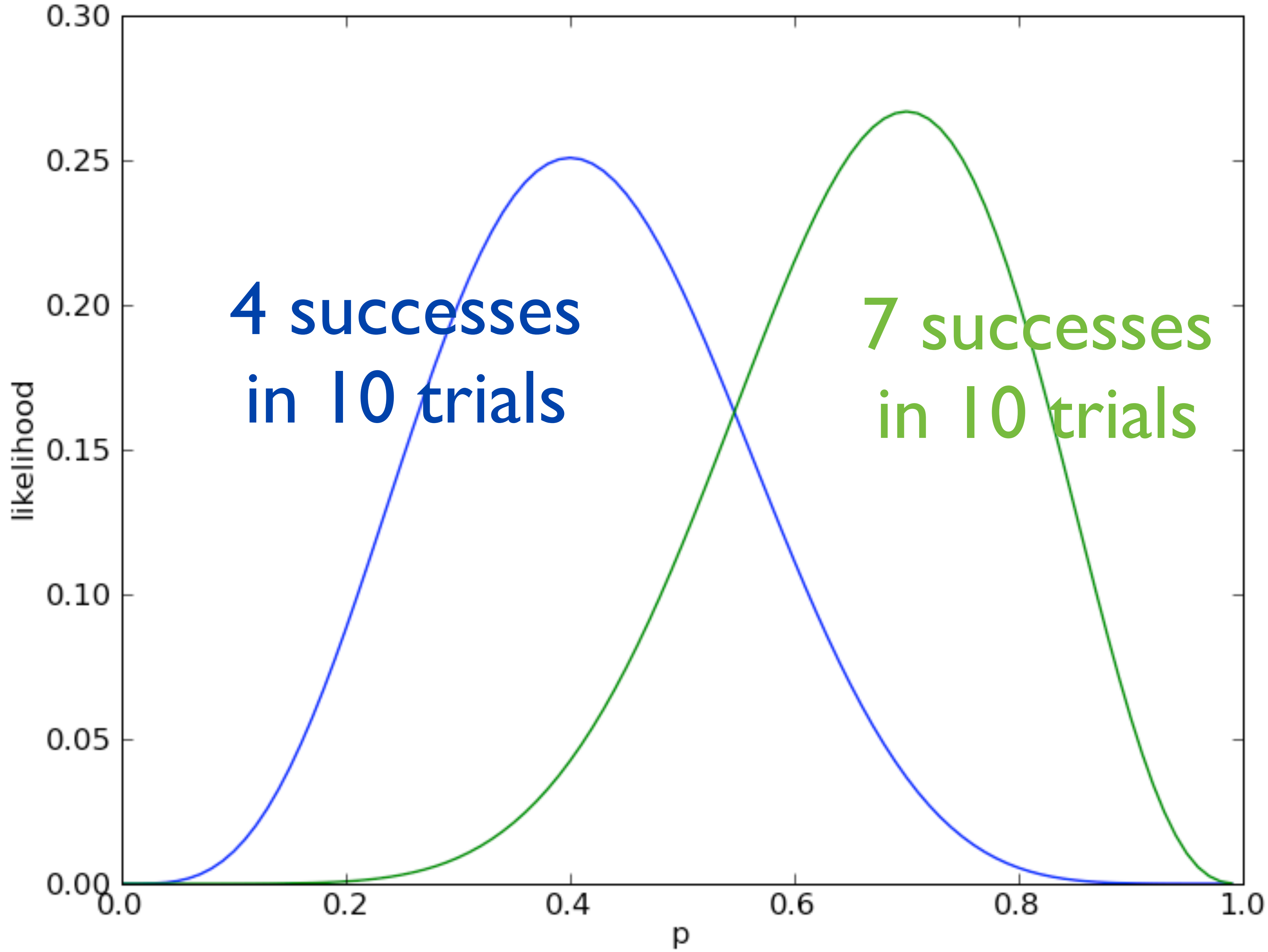
$$L(p_m|x) = \binom{N}{x} p_m^x (1-p_m)^{N-x}$$

continuous
usually doesn't sum to one

maximum likelihood estimator

$$L'(\theta|X) = \frac{dL(\theta|X)}{d\theta} = 0$$

solve for θ


4 successes
in 10 trials
7 successes
in 10 trials
likelihood
p
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.0
0.2
0.4
0.6
0.8
1.0

information

$$p(\theta|y) \propto p(y|\theta)p(\theta)$$

"Following observation of y, the likelihood $L(\theta|y)$ contains all experimental information from y about the unknown θ."

“For a given sample of data, any 2 probability models

$$L(\theta|y)$$

that have the same likelihood function yield the same inference for θ.”

Gelman et al., 2004

Bayesian Computation and Markov chain Monte Carlo

Calculation of Posteriors

$$p(\theta|y) = \frac{p(y|\theta)p(\theta)}{p(y)}$$

$$= \frac{p(y|\theta)p(\theta)}{\int p(y|\theta)p(\theta)d\theta}$$

Marginal Densities

$$\pi(\theta_i) = \int \pi(\theta_i|\theta_{-i})\pi(\theta_{-i})\mathrm{d}\theta_{-i}$$

average over remaining k-1 parameters

Approximate Inference

Normal and Laplace Approximations

Monte Carlo integration

we are attempting to estimate

$$I(a,b) = \int_a^b h(\theta) f(\theta) \mathrm{d}\theta$$

we are attempting to estimate

$$I(a,b) = \int_a^b h(\theta) f(\theta) \mathrm{d}\theta$$

using the sum

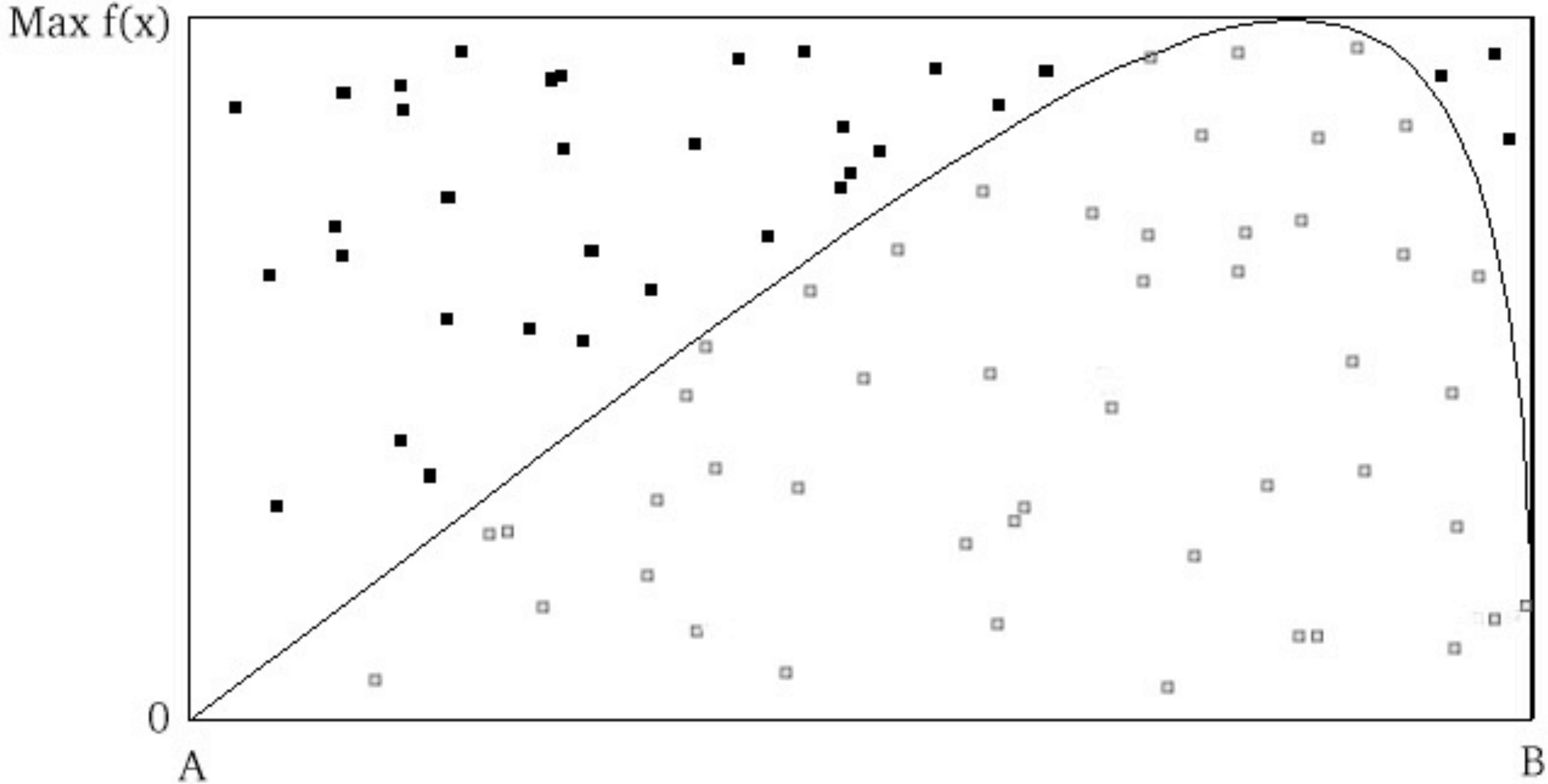
$$\hat{I}(a,b) = \frac{1}{n} \sum_{i=1}^{n} h(\theta_i)$$

$$\hat{I}(a,b) \to I(a,b) \text{ with probability } 1$$

by strong law of large numbers

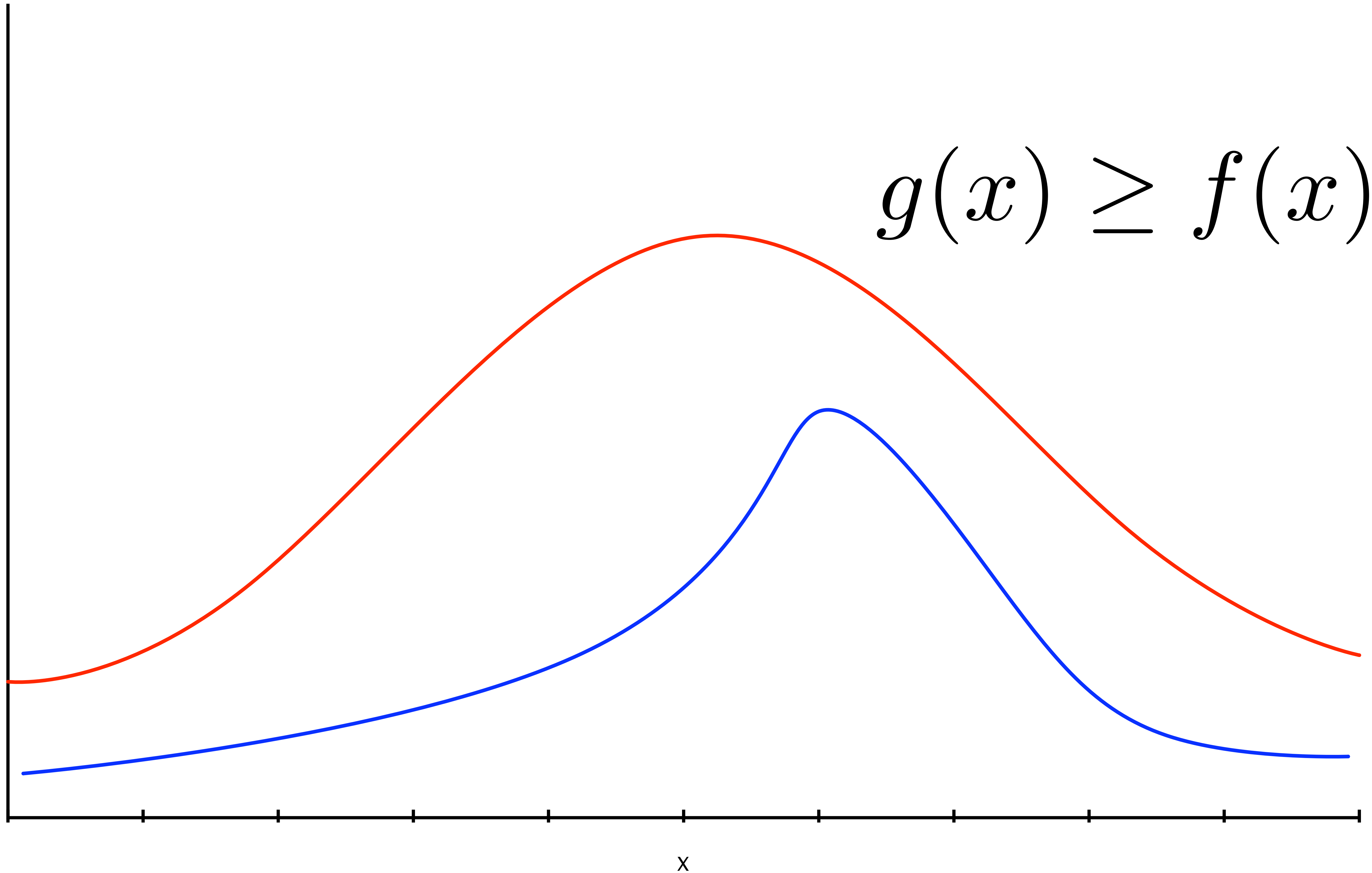
Rejection Sampling

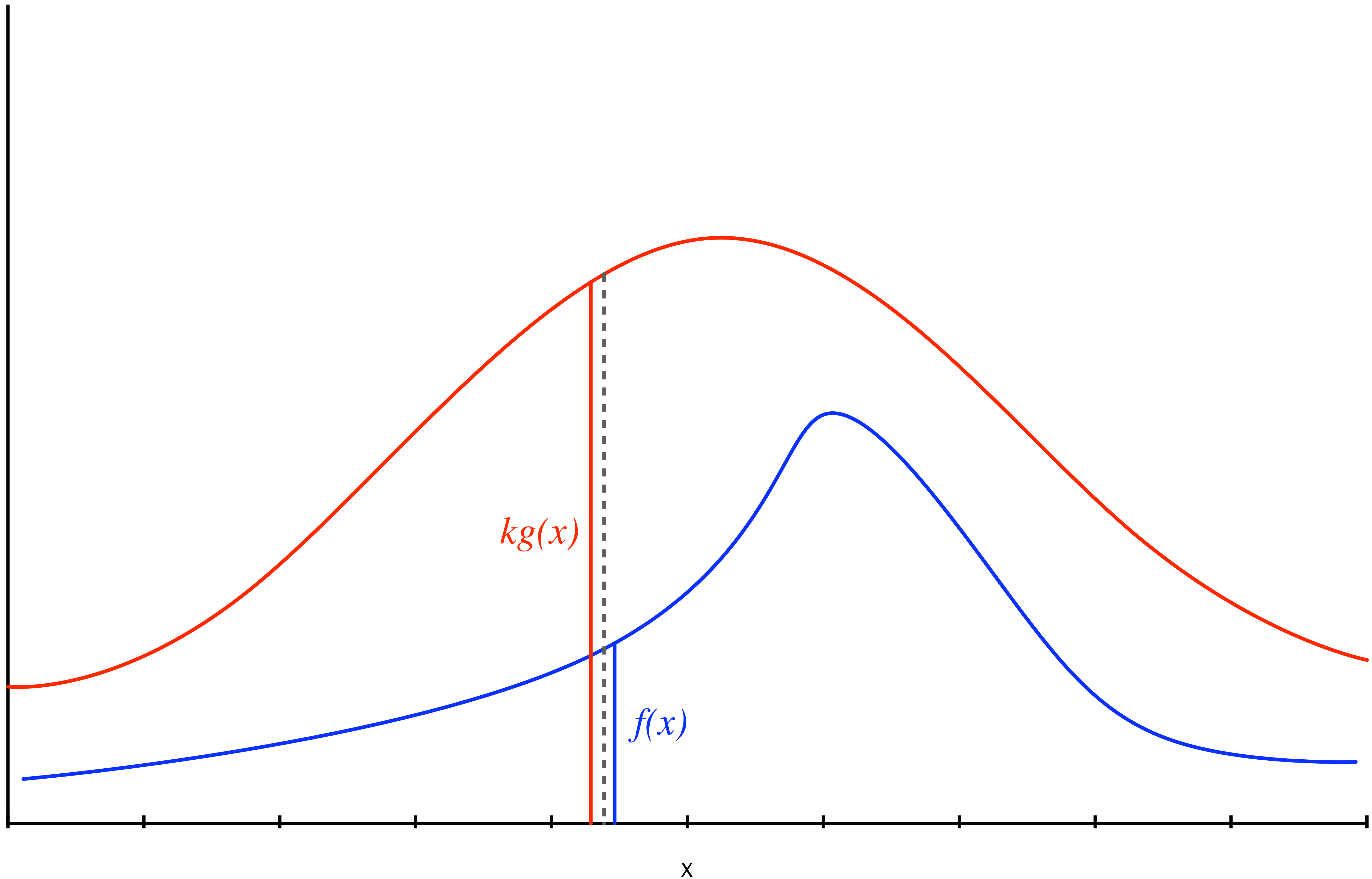
$$\int_A^B f(x)\mathrm{d}x$$



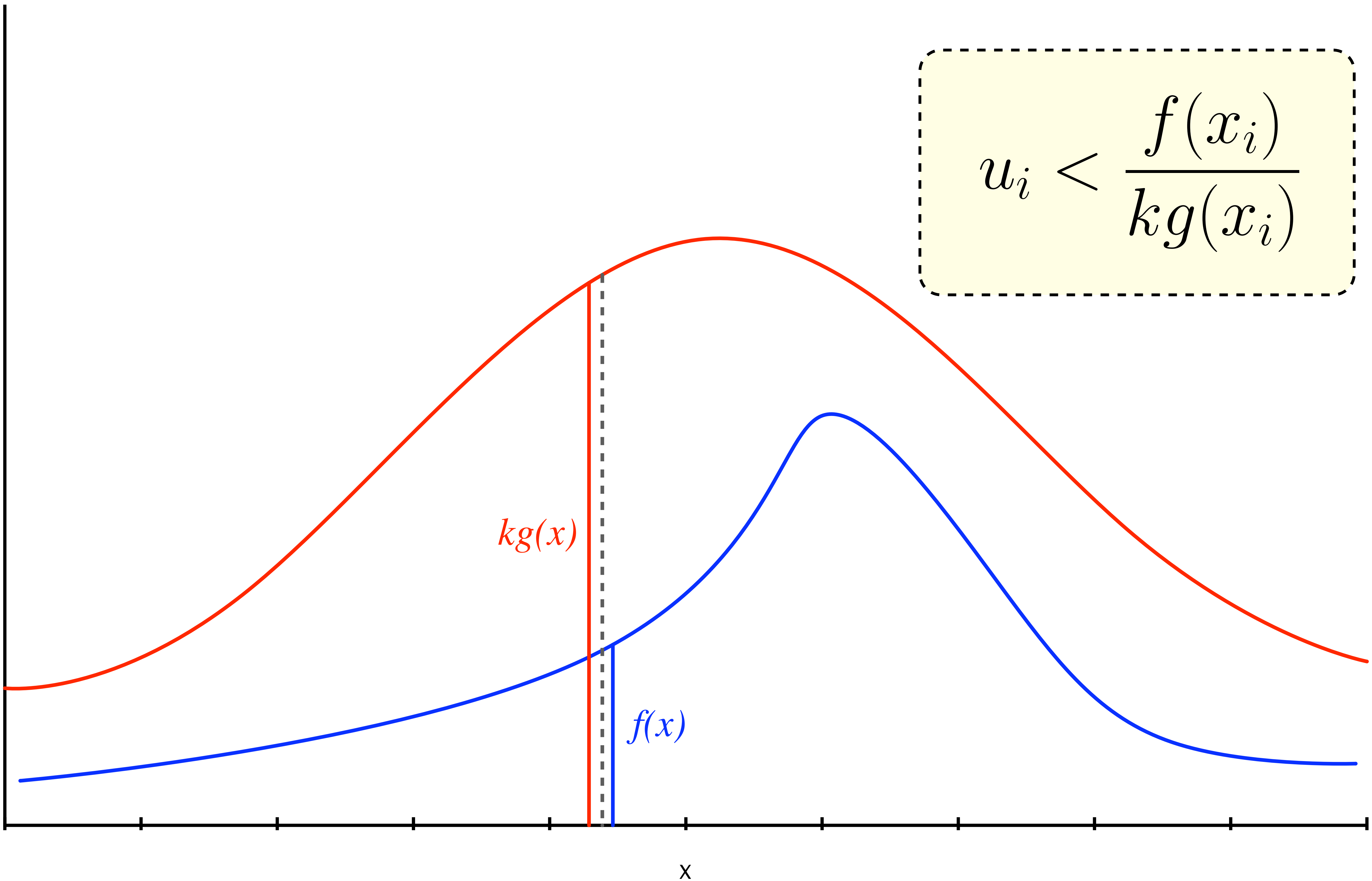
$$\frac{\text{Points under curve}}{\text{Points generated}} \times \text{box area} = \lim_{n\to\infty} \int_A^B f(x)\mathrm{d}x$$

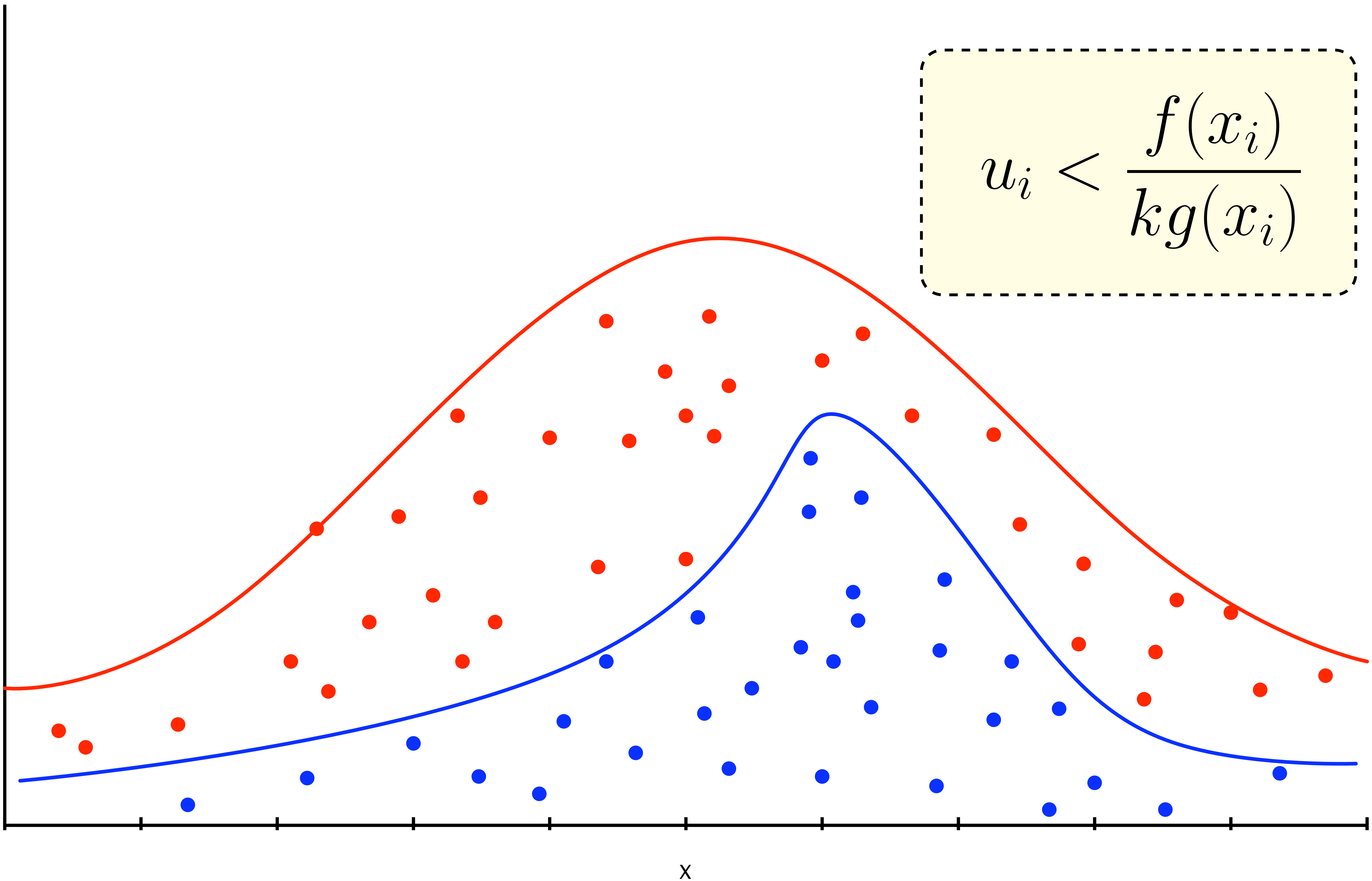
enveloping function




$kg(x)$
$f(x)$
x

$$u_i < \frac{f(x_i)}{kg(x_i)}$$





$$u_i < \frac{f(x_i)}{kg(x_i)}$$

Limitations

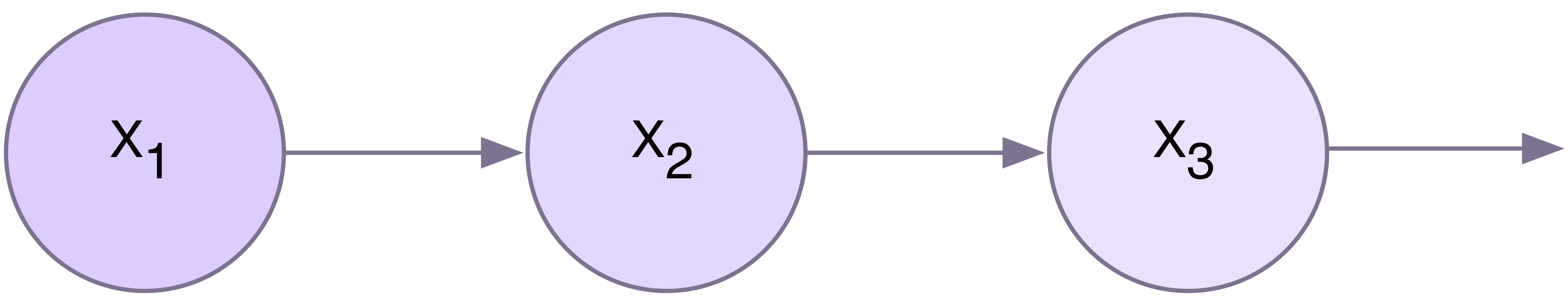
Posterior inference using **Markov chains**

$$\{X_t : t \in T\}$$

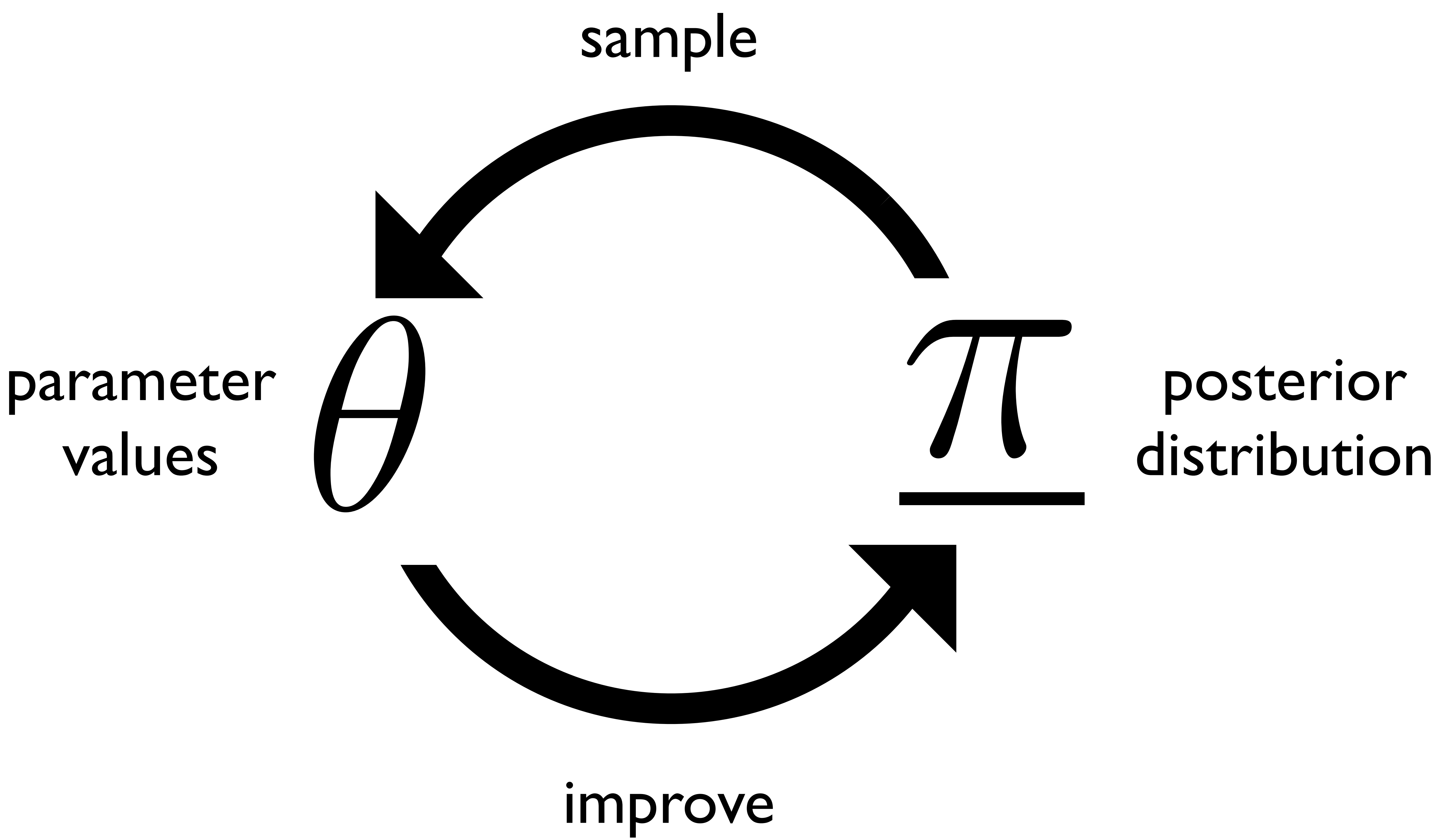
stochastic process

Markovian dependence

$$\begin{aligned} & Pr(X_{t+1} = x_{t+1} | X_t = x_t, X_{t-1} = x_{t-1}, \ldots, X_0 = x_0) \\ = \ & Pr(X_{t+1} = x_{t+1} | X_t = x_t) \end{aligned}$$




10
500



Markov chain simulation

detailed balance equation

$$\pi(x)Pr(y \mid x) = \pi(y)Pr(x \mid y)$$

“reversible”

IF a chain is reversible ...

$$\underline{\pi}$$

is the unique, invariant, stationary distribution of that chain

Markov chain Monte Carlo

(MCMC)

Gibbs Sampling

Geman and Geman (1984)

Gibbs distribution

energy

$$p(y_1, \ldots, y_k) \propto \exp\left[-\frac{1}{cT}E(y_1, \ldots, y_k)\right]$$

temperature

Conditional Sampling

$$\pi_i(\theta_i) = \pi(\theta_i|\theta_{-i})$$

component *i*

everything else

k=2

$$p(\theta_1, \theta_2 | y)$$

$$\downarrow$$

$$p_1(\theta_1 | \theta_2, y),\ p_2(\theta_2 | \theta_1, y)$$

Gibbs sampling algorithm

Choose starting values for parameters

$$\theta = [\theta_1^{(0)}, \theta_2^{(0)}, \ldots, \theta_k^{(0)}]$$

Initialise counter

$$j=1$$

Draw values from each of the *k* conditional distributions

$$
\begin{aligned}
\theta_1^{(j+1)} &\sim \pi(\theta_1 | \theta_2^{(j)}, \theta_3^{(j)}, \ldots, \theta_{k-1}^{(j)}, \theta_k^{(j)}) \\
\theta_2^{(j+1)} &\sim \pi(\theta_2 | \theta_1^{(j+1)}, \theta_3^{(j)}, \ldots, \theta_{k-1}^{(j)}, \theta_k^{(j)}) \\
\theta_3^{(j+1)} &\sim \pi(\theta_3 | \theta_1^{(j+1)}, \theta_2^{(j+1)}, \ldots, \theta_{k-1}^{(j)}, \theta_k^{(j)}) \\
&\vdots \\
\theta_{k-1}^{(j+1)} &\sim \pi(\theta_{k-1} | \theta_1^{(j+1)}, \theta_2^{(j+1)}, \ldots, \theta_{k-2}^{(j+1)}, \theta_k^{(j)}) \\
\theta_k^{(j+1)} &\sim \pi(\theta_k | \theta_1^{(j+1)}, \theta_2^{(j+1)}, \ldots, \theta_{k-2}^{(j+1)}, \theta_{k-1}^{(j+1)})
\end{aligned}
$$

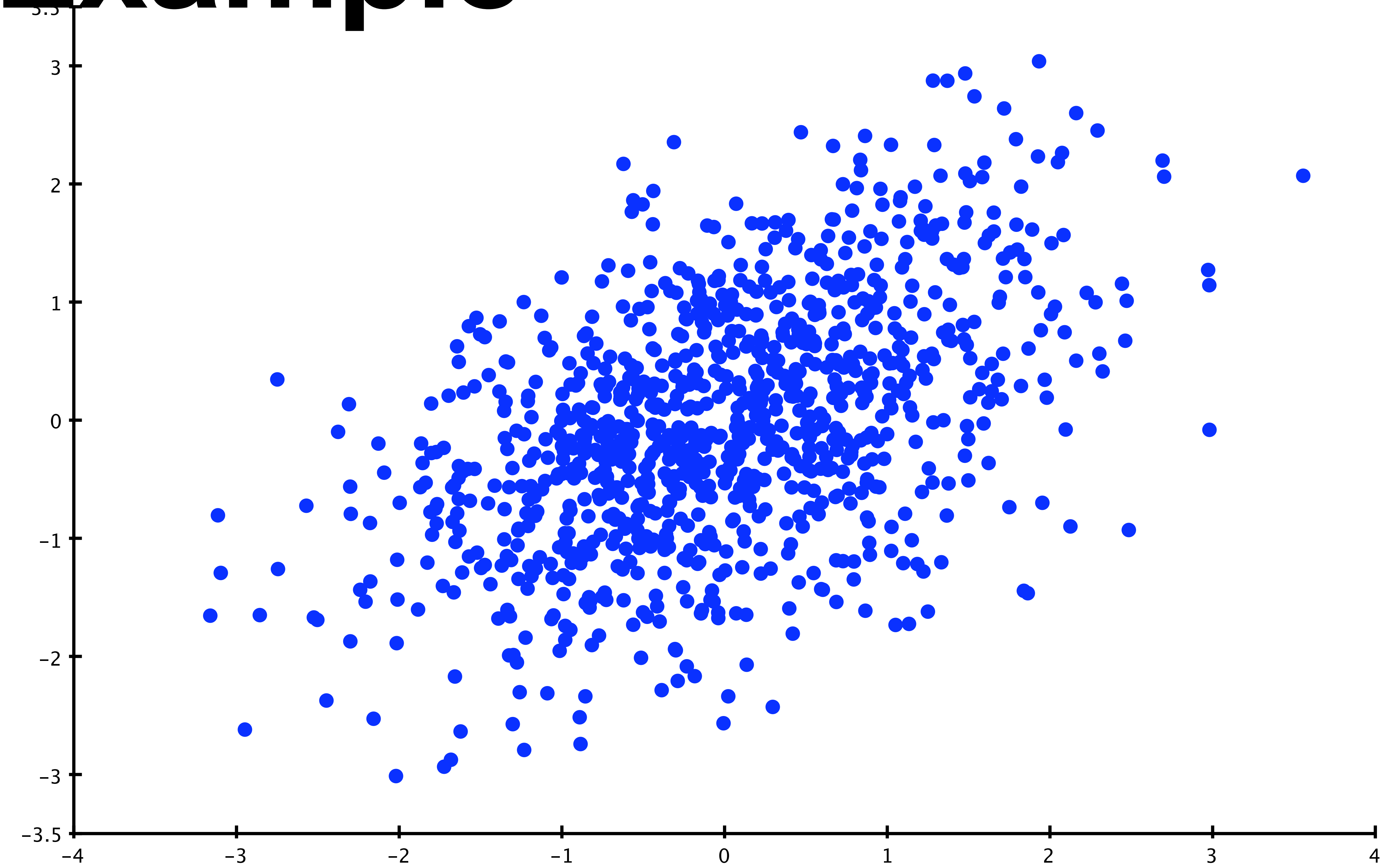
Increment counter and return to step 3 until convergence is reached.

$$j = j + 1$$

Markovian

$$\theta_1^{(j+1)} \sim \pi(\theta_1 | \theta_2^{(j)}, \theta_3^{(j)}, \ldots, \theta_{k-1}^{(j)}, \theta_k^{(j)})$$

Example

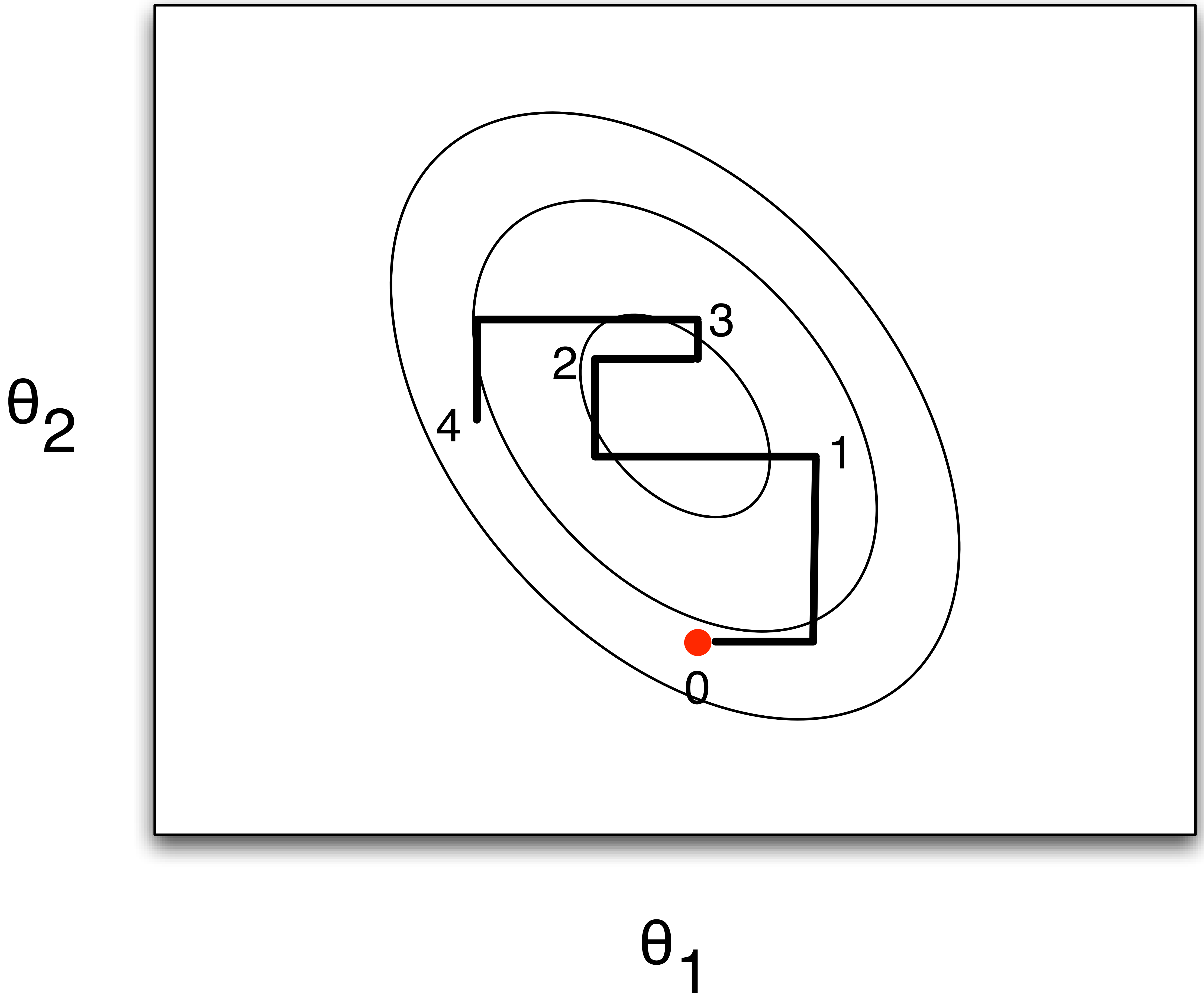


bivariate normal distribution

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \sim N\left(\begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix}, \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}\right)$$

$$\theta_1|\theta_2, y \sim N(y_1 + \rho(\theta_2 - y_2), 1 - \rho^2)$$
$$\theta_2|\theta_1, y \sim N(y_2 + \rho(\theta_1 - y_1), 1 - \rho^2)$$

conditional distributions



θ_2
0
1
2
3
4
θ_1

Obtaining a sample

$$n$$

$$
\begin{array}{ccccccc}
\theta^{(1,0)} & \rightarrow & \theta^{(1,1)} & \rightarrow & \cdots & \rightarrow & \theta^{(1,m)} \\
\theta^{(2,0)} & \rightarrow & \theta^{(2,1)} & \rightarrow & \cdots & \rightarrow & \theta^{(2,m)} \\
 & & & & \vdots & & \\
\theta^{(n,0)} & \rightarrow & \theta^{(n,1)} & \rightarrow & \cdots & \rightarrow & \theta^{(n,m)}
\end{array}
$$

parallel chains

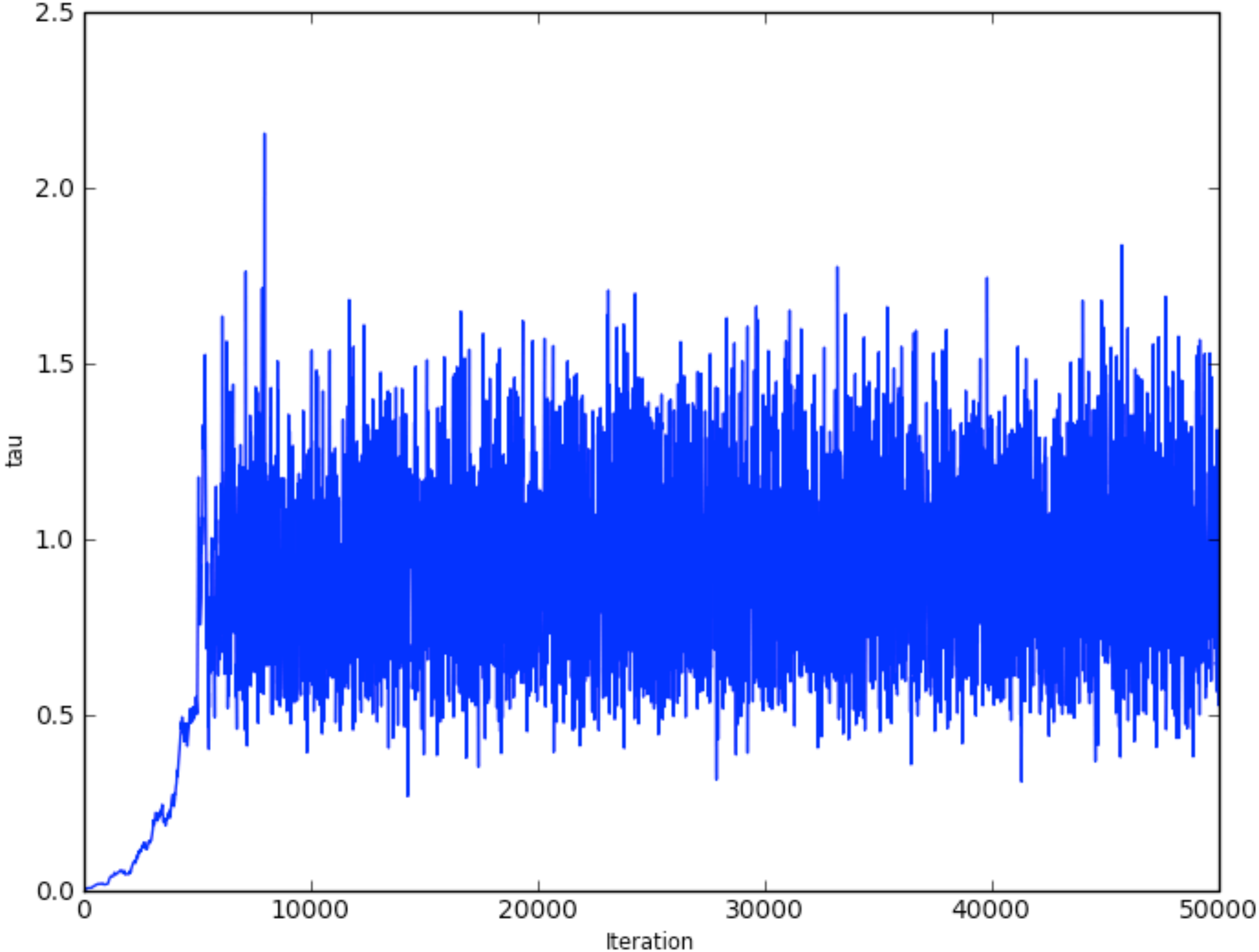
(*mn* iterations)

“burn-in”

$$\theta^{(0)} \longrightarrow \theta^{(1)} \longrightarrow \cdots \longrightarrow \theta^{(m)}$$

$$\theta^{(m+1)} \longrightarrow \theta^{(m+2)} \longrightarrow \cdots \longrightarrow \theta^{(m+n)}$$

inferential sample



tau
Iteration
2.5
2.0
1.5
1.0
0.5
0.0
0
10000
20000
30000
40000
50000

$$\theta_1^{(j+1)} \sim \pi(\theta_1 | \theta_2^{(j)}, \theta_3^{(j)}, \ldots, \theta_{k-1}^{(j)}, \theta_k^{(j)})$$

Markovian

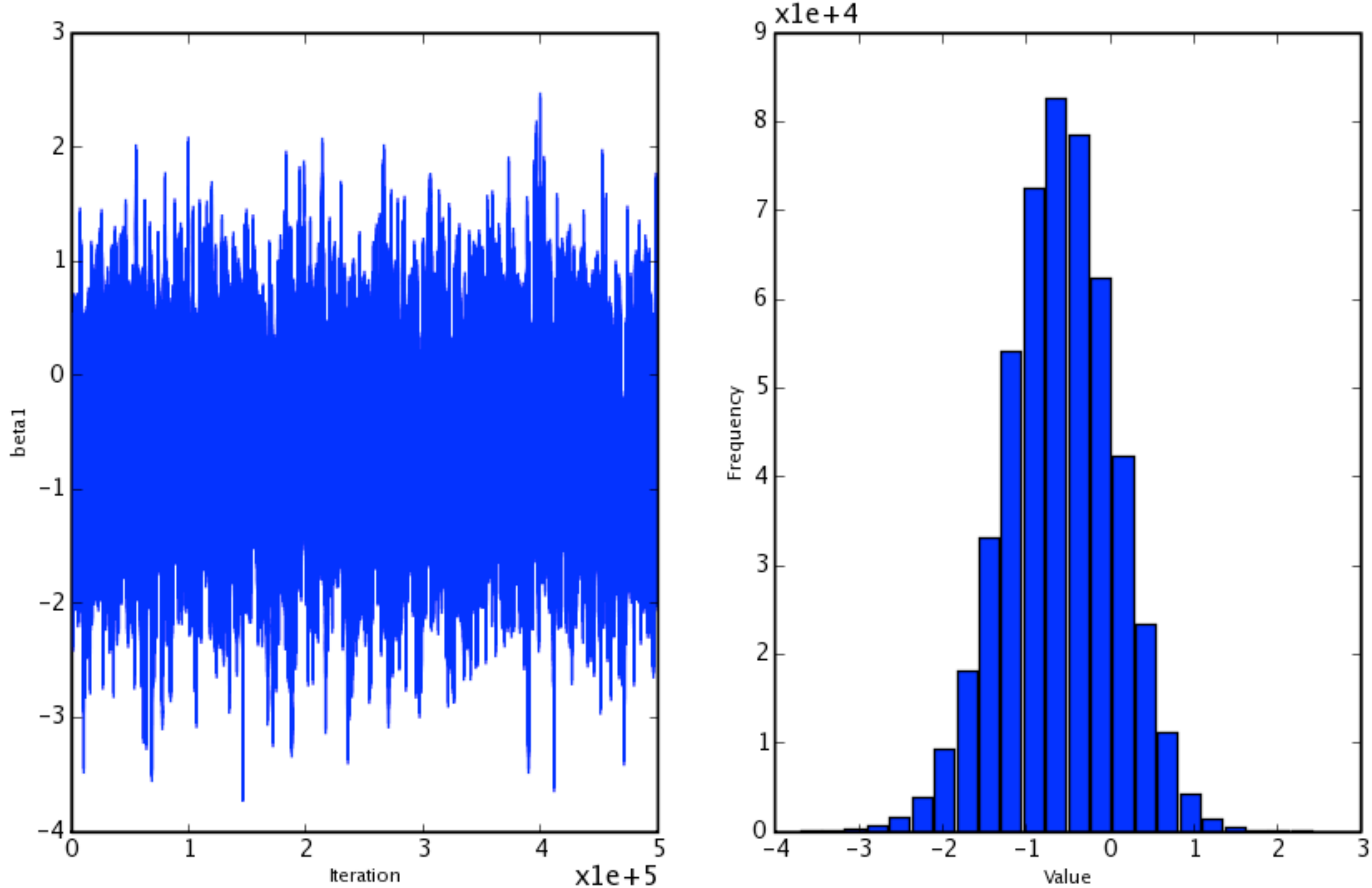
(non-independent)

Convergence

$$\underline{\pi}^{(0)} \xrightarrow{\textbf{??}} \underline{\pi}^{*}$$

Utilising the Sample

$$\{\theta_1, \ldots, \theta_k\}$$



beta1
Iteration
x1e+5
Frequency
Value
x1e+4

Metropolis-Hastings Sampling

Metropolis (1953), Hastings (1970)

$$
\begin{aligned}
\theta_1^{(j+1)} &\sim \pi(\theta_1|\theta_2^{(j)}, \theta_3^{(j)}, \ldots, \theta_{k-1}^{(j)}, \theta_k^{(j)}) \\
\theta_2^{(j+1)} &\sim \pi(\theta_2|\theta_1^{(j+1)}, \theta_3^{(j)}, \ldots, \theta_{k-1}^{(j)}, \theta_k^{(j)}) \\
\theta_3^{(j+1)} &\sim \pi(\theta_3|\theta_1^{(j+1)}, \theta_2^{(j+1)}, \ldots, \theta_{k-1}^{(j)}, \theta_k^{(j)}) \\
&\vdots \\
\theta_{k-1}^{(j+1)} &\sim \pi(\theta_{k-1}|\theta_1^{(j+1)}, \theta_2^{(j+1)}, \ldots, \theta_{k-2}^{(j+1)}, \theta_k^{(j)}) \\
\theta_k^{(j+1)} &\sim \pi(\theta_k|\theta_1^{(j+1)}, \theta_2^{(j+1)}, \ldots, \theta_{k-2}^{(j+1)}, \theta_{k-1}^{(j+1)})
\end{aligned}
$$

full conditionals

transition
probability

$$p(x, y) = q(x, y)\alpha(x, y),\; x \neq y$$

transition
proposal

$$\alpha(x, y) = \min\left\{1, \frac{q(y, x)\pi(y)}{q(x, y)\pi(x)}\right\}$$

can be anything!

$$\alpha(x, y) = \min\left\{1, \frac{q(y, x)\pi(y)}{q(x, y)\pi(x)}\right\}$$

Metropolis-Hastings sampling algorithm

Choose starting values for parameters

$$\theta = \theta^{(0)}$$

Initialise counter

2

$$j=1$$

Propose new parameter value

$$\theta' \sim q(\theta^{(j)}, \cdot)$$

proposal distribution

Evaluate the acceptance probability for the proposed move

$$\alpha(\theta^{(j)}, \theta') = \min\left\{1, \frac{q(\theta', \theta^{(j)})\pi(\theta')}{q(\theta^{(j)}, \theta')\pi(\theta^{(j)})}\right\}$$

ratio of proposals

ratio of posteriors

Generate a uniform random variable

$$u \sim U(0,1)$$

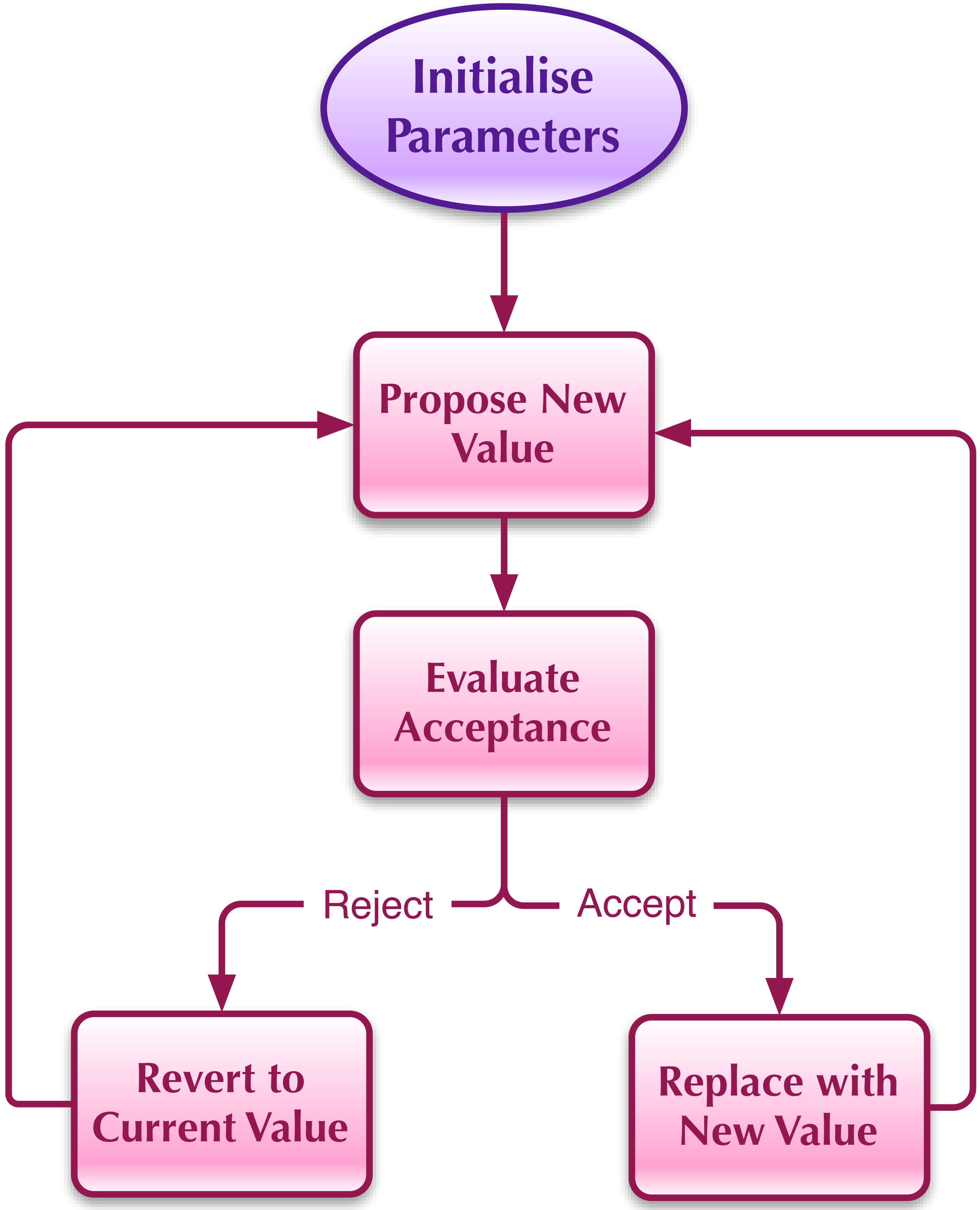
uniform distribution

Evaluate proposed value

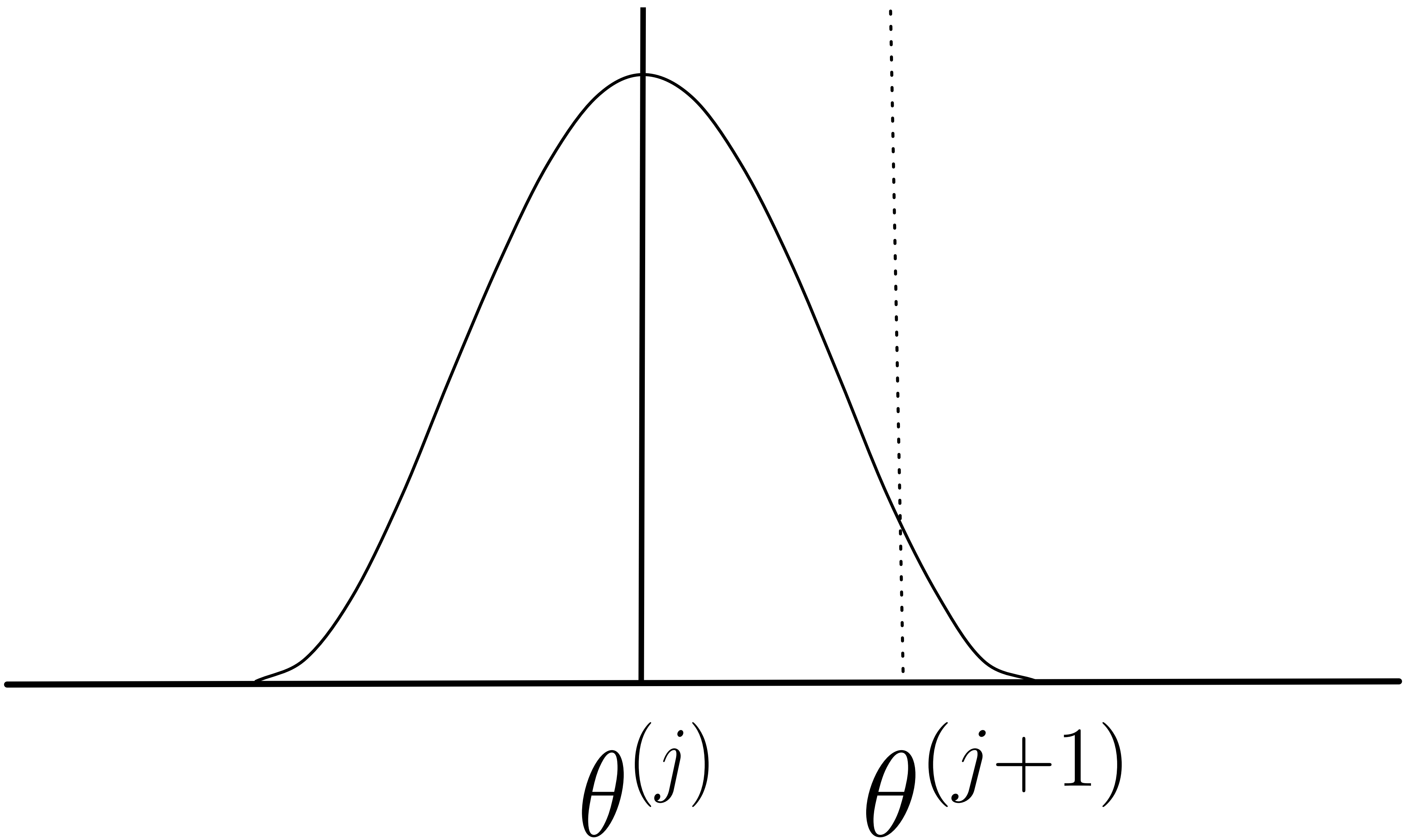
$$\theta^{(j+1)} = \begin{cases} \theta' \text{ if } \alpha(\theta^{(j)}, \theta') \geq u \\ \theta^{(j)} \text{ if } \alpha(\theta^{(j)}, \theta') < u \end{cases}$$

Increment counter and **return to step 3** until convergence is reached.

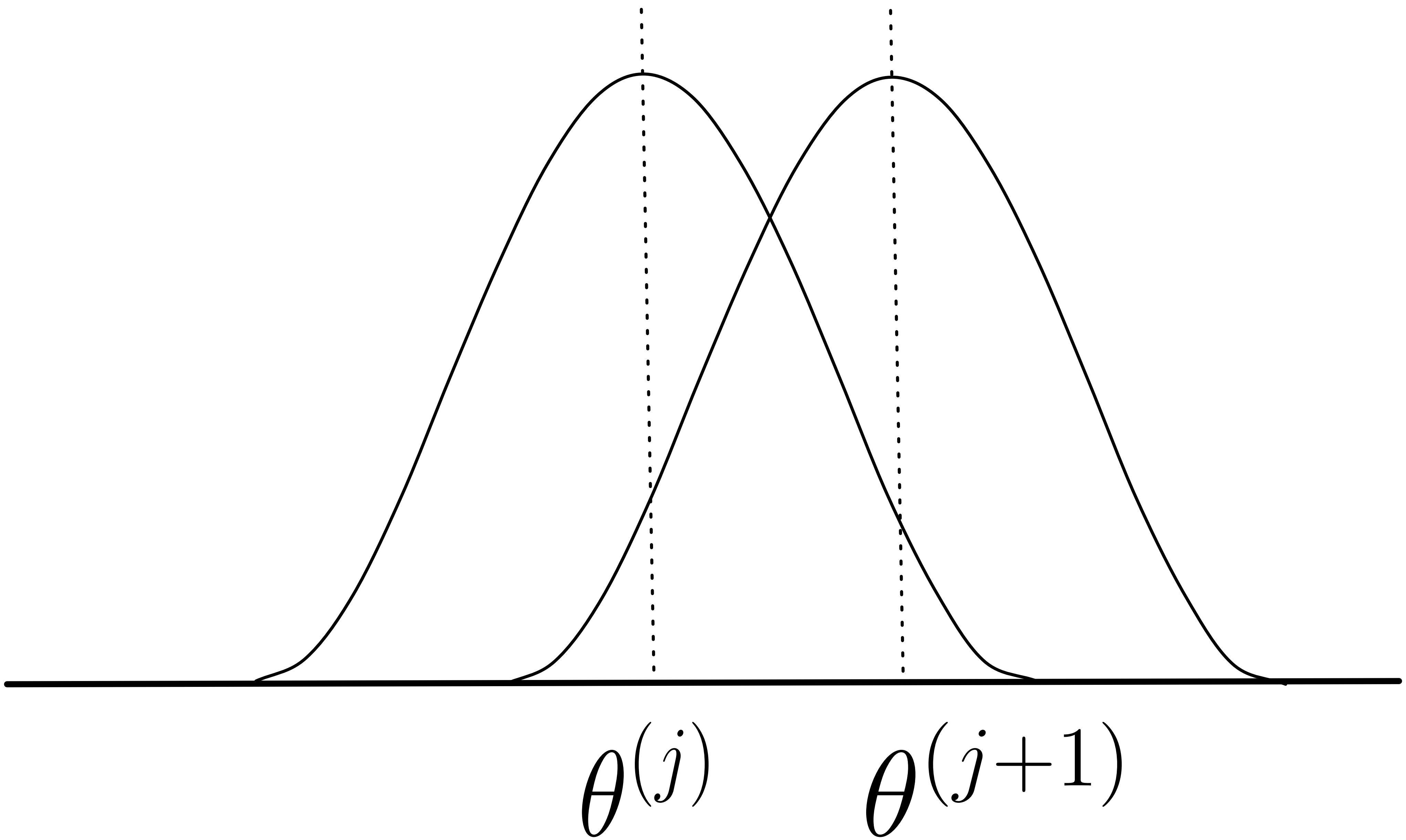
$$j = j + 1$$


Initialise Parameters
Propose New Value
Evaluate Acceptance
Reject
Accept
Revert to Current Value
Replace with New Value

$$q(\theta^{(j)}, \theta') = q(\theta', \theta^{(j)})$$



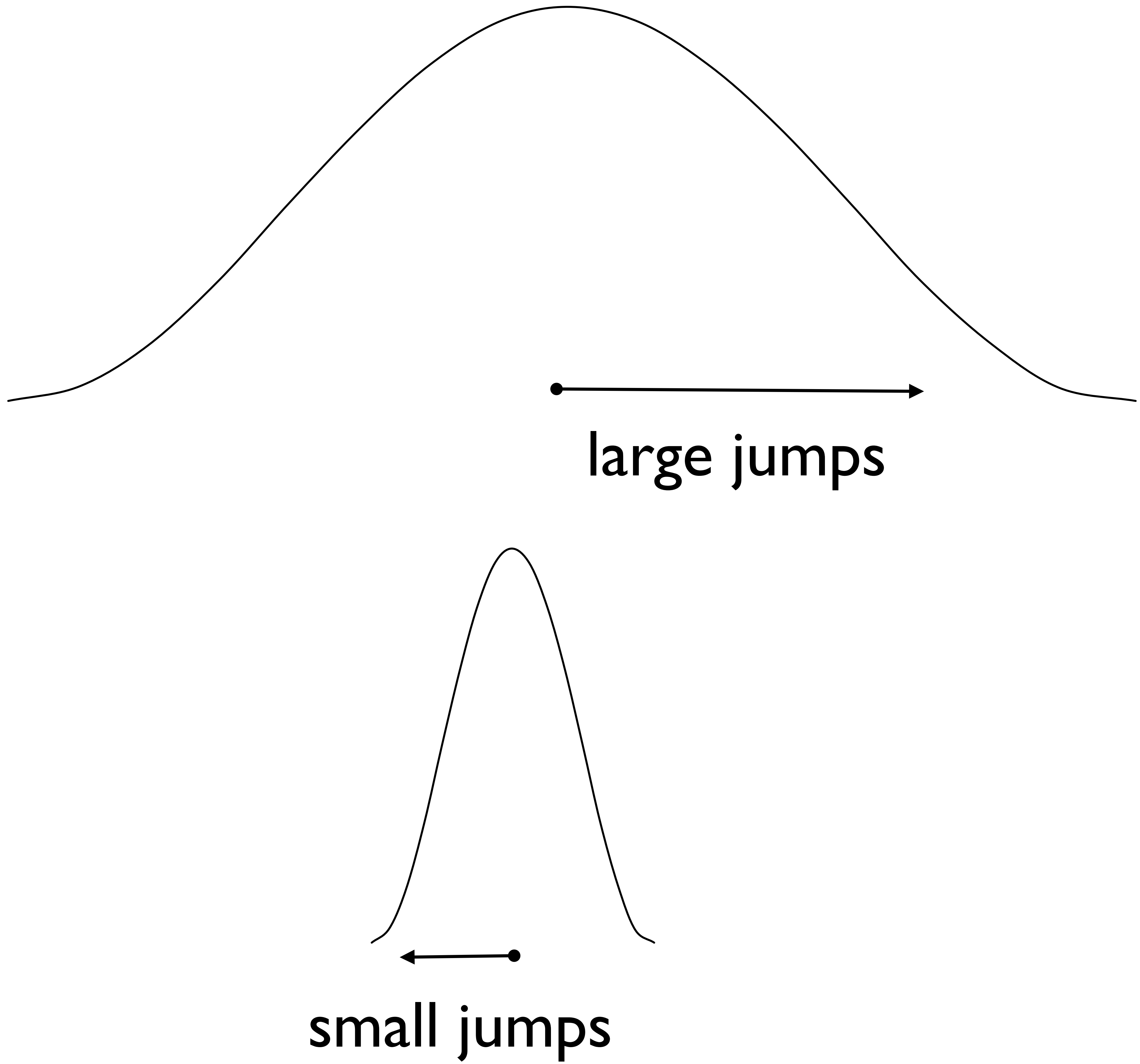
$$\alpha(\theta^{(j)}, \theta') = \min\left\{1, \frac{\pi(\theta')}{\pi(\theta^{(j)})}\right\}$$



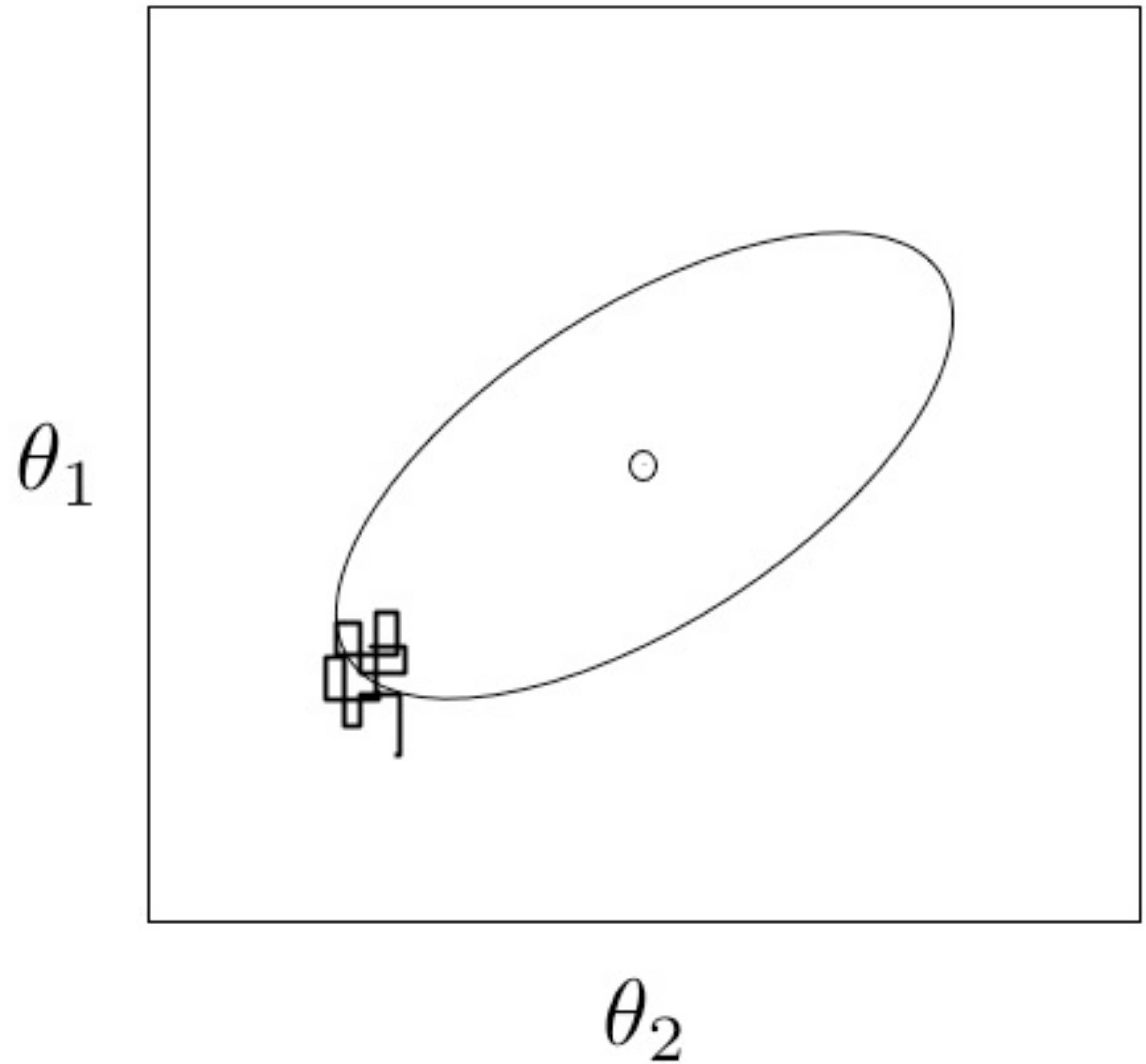
$$\alpha(\theta^{(j)}, \theta') = \min\left\{1, \frac{\pi(\theta')}{\pi(\theta^{(j)})}\right\}$$

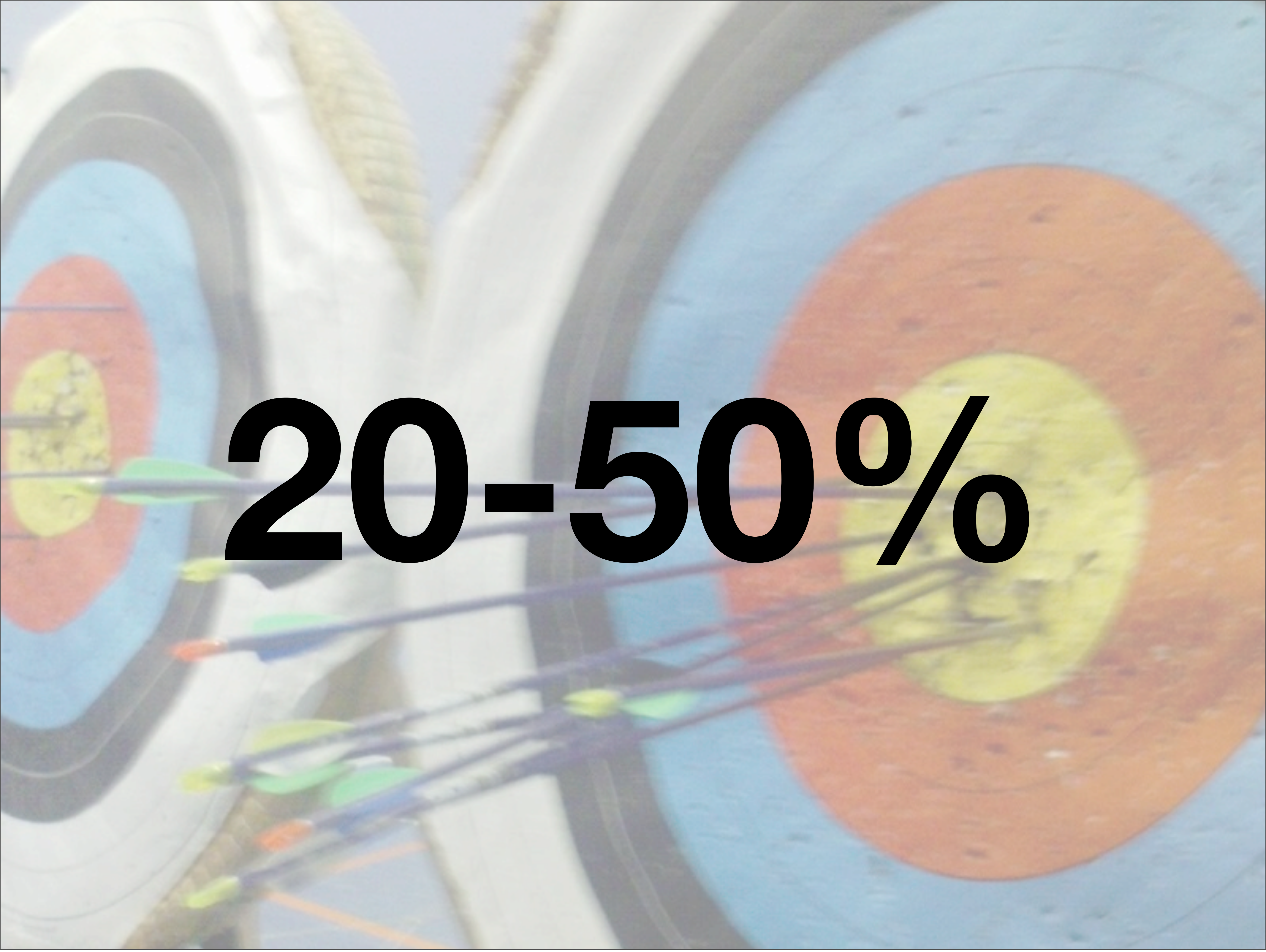
reject if

$$\pi(\theta') \ll \pi(\theta^{(j)})$$


large jumps
small jumps

"poor mixing"




20-50%

$$\frac{\pi(\theta'|y)}{\pi(\theta^{(j)}|y)} = \frac{\frac{L(y|\theta')\pi(\theta')}{\int L(y|\theta)\pi(\theta)d\theta}}{\frac{L(y|\theta^{(j)})\pi(\theta^{(j)})}{\int L(y|\theta)\pi(\theta)d\theta}}$$

$$= \frac{L(y|\theta')\pi(\theta')}{L(y|\theta^{(j)})\pi(\theta^{(j)})}$$

$$\frac{\pi(\theta'|y)}{\pi(\theta^{(j)}|y)} = \frac{\frac{L(y|\theta')\pi(\theta')}{\cancel{\int L(y|\theta)\pi(\theta)d\theta}}}{\frac{L(y|\theta^{(j)})\pi(\theta^{(j)})}{\cancel{\int L(y|\theta)\pi(\theta)d\theta}}}$$

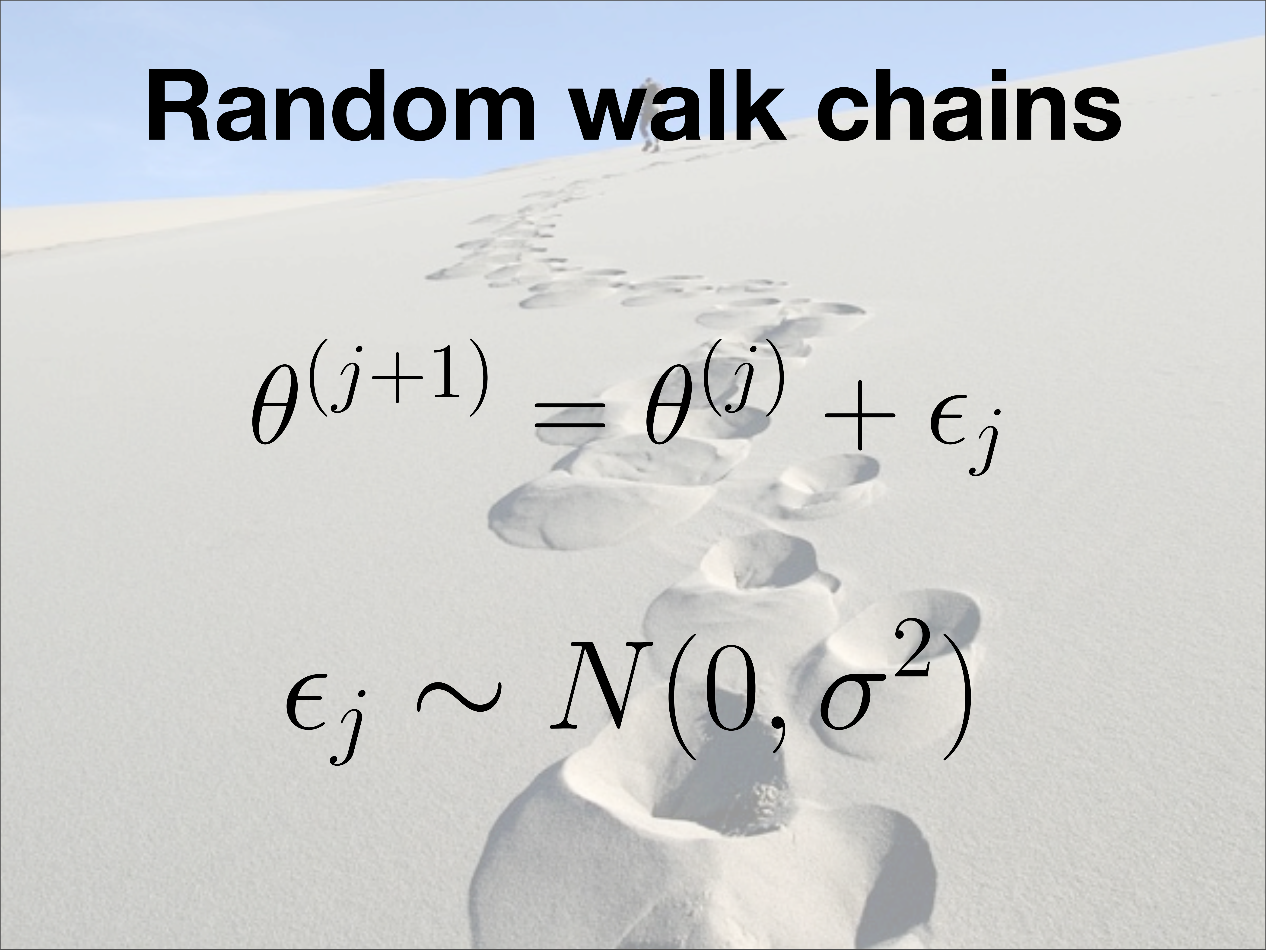
$$= \frac{L(y|\theta')\pi(\theta')}{L(y|\theta^{(j)})\pi(\theta^{(j)})}$$

The Gibbs sampler
is a **special case** of the
Metropolis-Hastings sampler

$$
\begin{aligned}
\frac{q(\theta', \theta^{(j)})\pi(\theta')}{q(\theta^{(j)}, \theta')\pi(\theta^{(j)})} &= \frac{p(\theta_i^{(j)}|\theta'_{-i})}{p(\theta'_i|\theta_{-i}^{(j)})} \frac{p(\theta'_i|\theta'_{-i})p(\theta'_{-i})}{p(\theta_i^{(j)}|\theta_{-i}^{(j)})p(\theta_{-i}^{(j)})} \\
&= \frac{p(\theta_i^{(j)}|\theta_{-i})}{p(\theta'_i|\theta_{-i})} \frac{p(\theta'_i|\theta_{-i})p(\theta_{-i})}{p(\theta_i^{(j)}|\theta_{-i})p(\theta_{-i})} \\
&= 1
\end{aligned}
$$

proposals always accepted

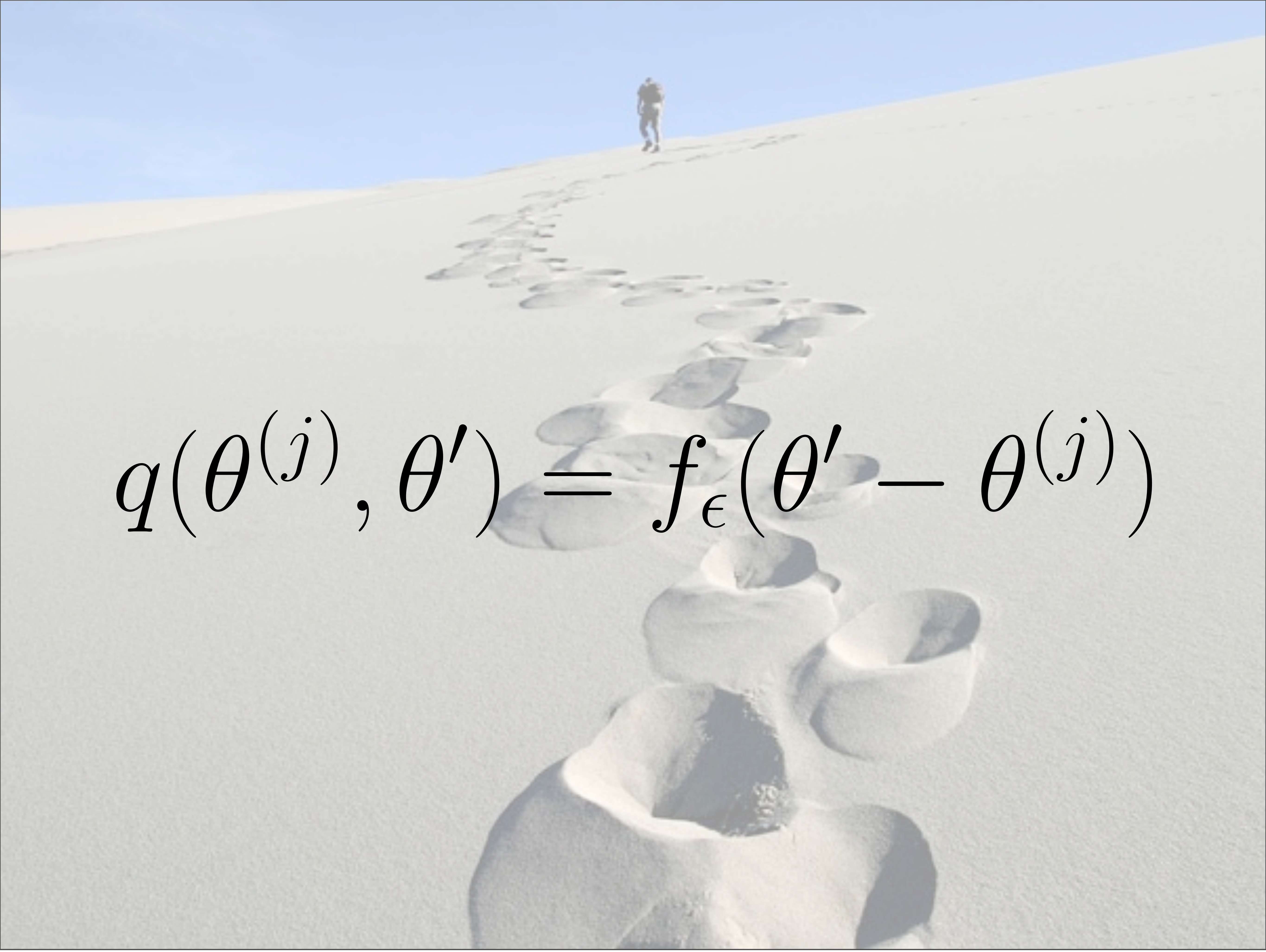
Choice of proposal



Random walk chains

$$\theta^{(j+1)} = \theta^{(j)} + \epsilon_j$$

$$\epsilon_j \sim N(0, \sigma^2)$$

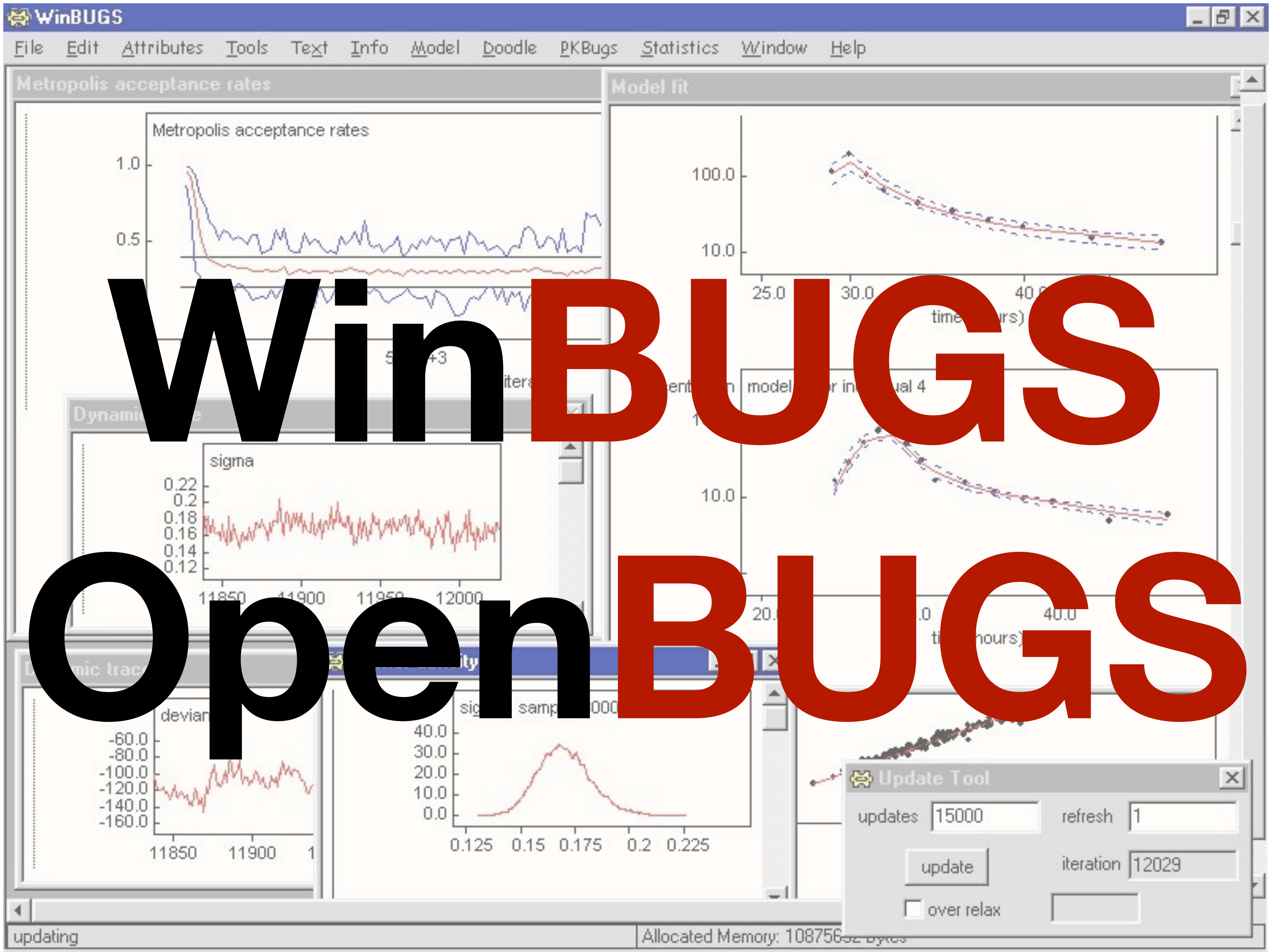


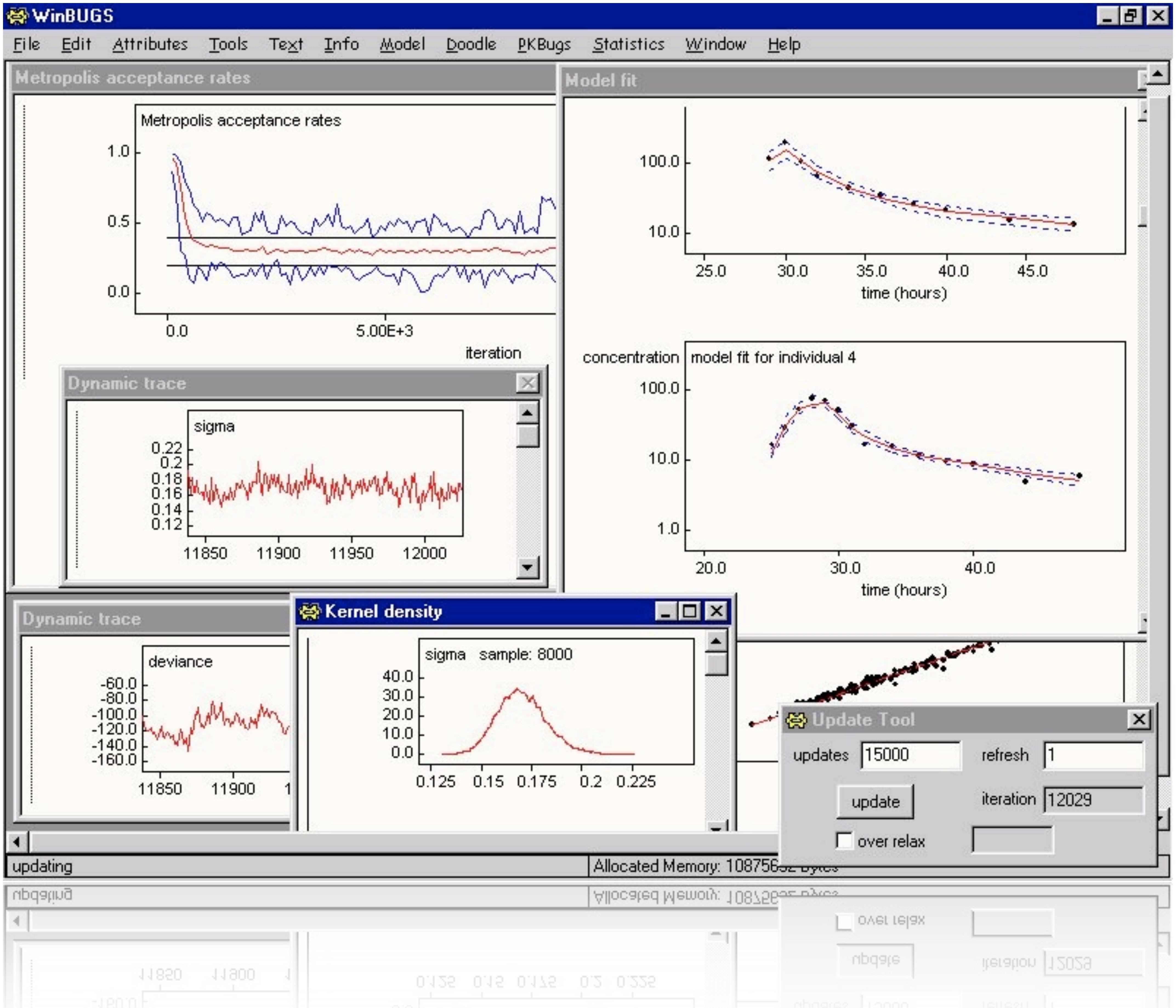
$$q(\theta^{(j)}, \theta') = f_\epsilon(\theta' - \theta^{(j)})$$

Software for Bayesian Computation

WinBUGS

OpenBUGS




WinBUGS
File Edit Attributes Tools Text Info Model Doodle PKBugs Statistics Window Help
Metropolis acceptance rates
Metropolis acceptance rates
1.0
0.5
0.0
0.0
5.00E+3
iteration
Model fit
100.0
10.0
25.0
30.0
35.0
40.0
45.0
time (hours)
concentration
model fit for individual 4
100.0
10.0
1.0
20.0
30.0
40.0
time (hours)
Dynamic trace
sigma
0.22
0.2
0.18
0.16
0.14
0.12
11850
11900
11950
12000
Dynamic trace
deviance
-60.0
-80.0
-100.0
-120.0
-140.0
-160.0
11850
11900
Kernel density
sigma sample: 8000
40.0
30.0
20.0
10.0
0.0
0.125
0.15
0.175
0.2
0.225
Update Tool
updates 15000
refresh 1
update
iteration 12029
over relax
updating

```
model {
    for (i in 1:N) {
        r[i] ~ dbin(p[i], n[i])
        b[i] ~ dnorm(0, tau)
        logit(p[i]) <- alpha0 + alpha1 * x1[i] + alpha2 * x2[i]
            + alpha12 * x1[i] * x2[i] + b[i]
    }
    alpha0 ~ dnorm(0, 1.0E-6)
    alpha1 ~ dnorm(0, 1.0E-6)
    alpha2 ~ dnorm(0, 1.0E-6)
    alpha12 ~ dnorm(0, 1.0E-6)
    tau ~ dgamma(0.001, 0.001)
    sigma <- 1 / sqrt(tau)
}
```

Lunn, D.J., Thomas, A., Best, N., and Spiegelhalter, D. (2000) WinBUGS -- a Bayesian modelling framework: concepts, structure, and extensibility. *Statistics and Computing*, **10**:325--337.

FrontPage - OpenBUGS

http://www.openbugs.info/w/

Pinboard Twitter Flickr Simplenote GitHub Shorten Send to Delibar Send to OmniFocus Open in Papers Squarespace

Login

OpenBUGS

Search Titles Text

Wiki

- Home Page
- Overview
- Downloads
- Getting Started
- Documentation
- Developers
- BugTracker
- Community
- Recent Changes
- search
- Site Navigation
- How to Use This Site
- Help Contents

Page

- Immutable Page
- Info
- Attachments

More Actions:

FrontPage [what links here?]

Overview

Overview...

BUGS is a software package for performing **B**ayesian inference **U**sing **G**ibbs **S**ampling. The user specifies a statistical model, of (almost) arbitrary complexity, by simply stating the relationships between related variables. The software includes an 'expert system', which determines an appropriate MCMC (Markov chain Monte Carlo) scheme (based on the Gibbs sampler) for analysing the specified model. The user then controls the execution of the scheme and is free to choose from a wide range of output types.

Versions...

There are two main versions of BUGS, namely WinBUGS and OpenBUGS. This site is dedicated to OpenBUGS, an open-source version of the package, on which all future development work will be focused. OpenBUGS, therefore, represents the future of the BUGS project. WinBUGS, on the other hand, is an established and stable, stand-alone version of the software, which will remain available but not further developed. The latest versions of OpenBUGS (from v3.0.7 onwards) have been designed to be at least as efficient and reliable as WinBUGS over a wide range of test applications. Please see here for more information on WinBUGS. OpenBUGS runs on x86 machines with MS Windows, Unix/Linux or Macintosh (using Wine).

Note that software exists to run OpenBUGS (and analyse its output) from within both R and SAS, amongst others.

For additional details on the differences between OpenBUGS and WinBUGS see the OpenVsWin manual page.

How it works...

The specified model belongs to a class known as *Directed Acyclic Graphs* (DAGs), for which there exists an elegant underlying mathematical theory. This allows us to break down the analysis of arbitrarily large and complex structures into a sequence of relatively simple computations. BUGS includes a range of algorithms that its expert system can assign to each such computational task.

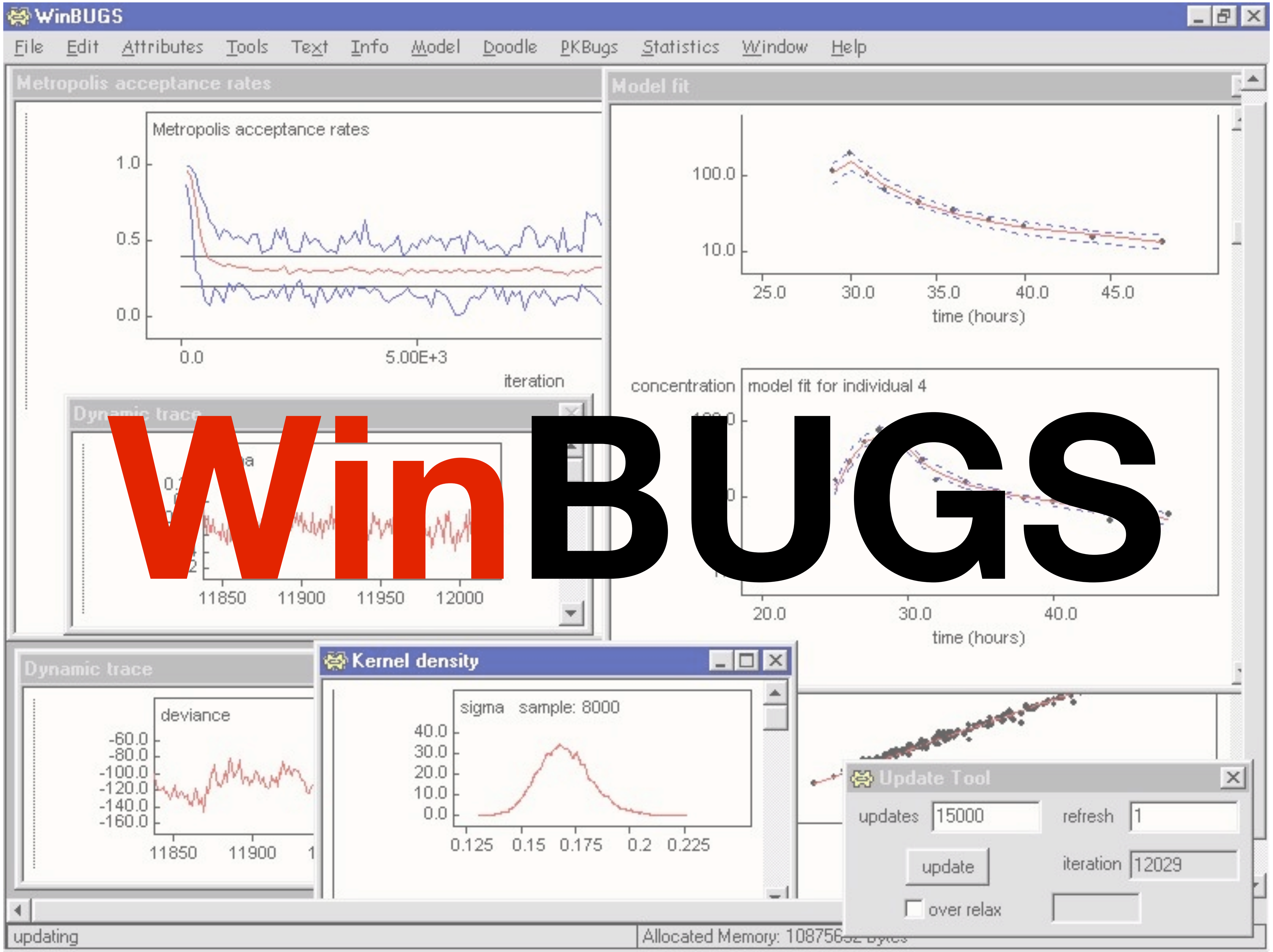
One of the main differences between OpenBUGS and WinBUGS is the way in which the expert system makes its decisions. WinBUGS defines one algorithm for each possible computation type whereas there is no limit to the number of algorithms that OpenBUGS can make use of, making for much greater flexibility and extensibility.

Trap

TRAP 0

◆ BugsMsg.StoreError [00000051H] ◆		
.msg	ARRAY 1024 OF CHAR	"unknown type of logical function" → ... ←
◆ BugsParser.Error [00000180H] ◆		
.errorMes	ARRAY 1024 OF CHAR	"unknown type of logical function" → ... ←
.errorNum	INTEGER	1
.numToString	ARRAY 8 OF CHAR	"1" → ... ←
◆ BugsParser.ParseFunction [00000C31H] ◆		
.dependent	BugsParser.Variable	NIL
.derivative	BugsParser.Variable	NIL
.fact	GraphNodes.Factory	[01029CF0H] ◆
.funcDesc	BugsGrammar.External	[010CAD50H] ◆
.function	BugsParser.Function	NIL
.functionVar	BugsParser.Variable	NIL
.i	INTEGER	2283448
.independent	BugsParser.Variable	NIL
.internal	BugsParser.Internal	NIL
.j	INTEGER	17407692
.loops	BugsParser.Statement	NIL
.numPar	INTEGER	1
.opDesc	BugsGrammar.Internal	NIL
.s	BugsMappers.Scanner	→ fields ←
.signature	ARRAY 64 OF CHAR	"vC" → ... ←
◆ BugsParser.ParseFactor [000013EEH] ◆		
.binary	BugsParser.Binary	NIL
.integer	BOOLEAN	FALSE
.loops	BugsParser.Statement	NIL
.node	BugsParser.Node	NIL
.pos	INTEGER	0
.s	BugsMappers.Scanner	→ fields ←
◆ BugsParser.ParseTerm [000011C3H] ◆		
.binary	BugsParser.Binary	NIL
.integer	BOOLEAN	FALSE
.loops	BugsParser.Statement	NIL
.node	BugsParser.Node	NIL
.op	INTEGER	1636074335
.s	BugsMappers.Scanner	→ fields ←
◆ BugsParser.Expression [000012C5H] ◆		
.binary	BugsParser.Binary	NIL

WinBUGS


WinBUGS
File Edit Attributes Tools Text Info Model Doodle PKBugs Statistics Window Help
Metropolis acceptance rates
Metropolis acceptance rates
1.0
0.5
0.0
0.0
5.00E+3
iteration
Model fit
100.0
10.0
25.0 30.0 35.0 40.0 45.0
time (hours)
concentration
model fit for individual 4
20.0 30.0 40.0
time (hours)
Dynamic trace
11850 11900 11950 12000
Dynamic trace
deviance
-60.0
-80.0
-100.0
-120.0
-140.0
-160.0
11850 11900
Kernel density
sigma sample: 8000
40.0
30.0
20.0
10.0
0.0
0.125 0.15 0.175 0.2 0.225
Update Tool
updates 15000
refresh 1
update
iteration 12029
over relax
updating
Allocated Memory: 108756

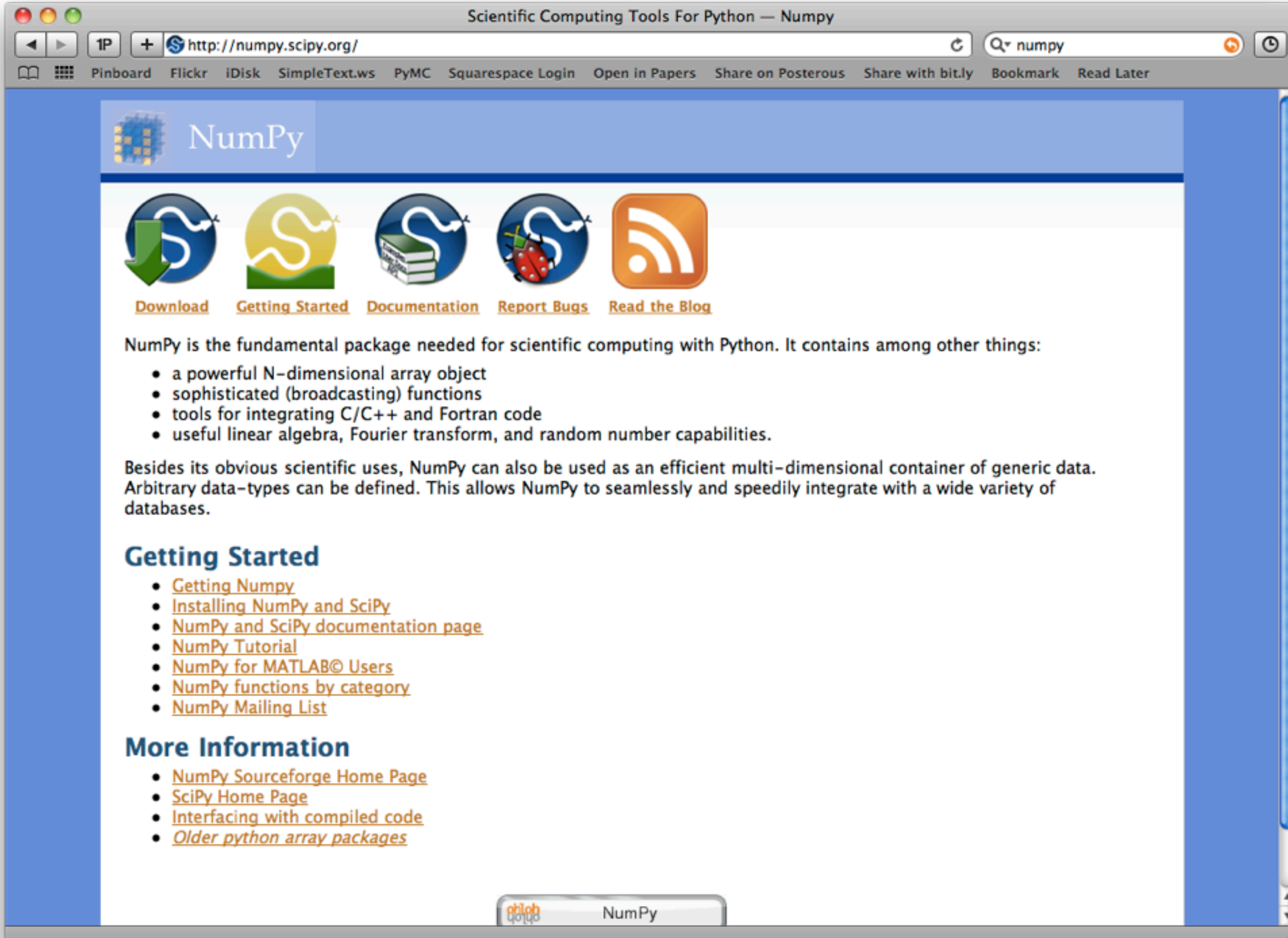
PyMC

Anand Patil

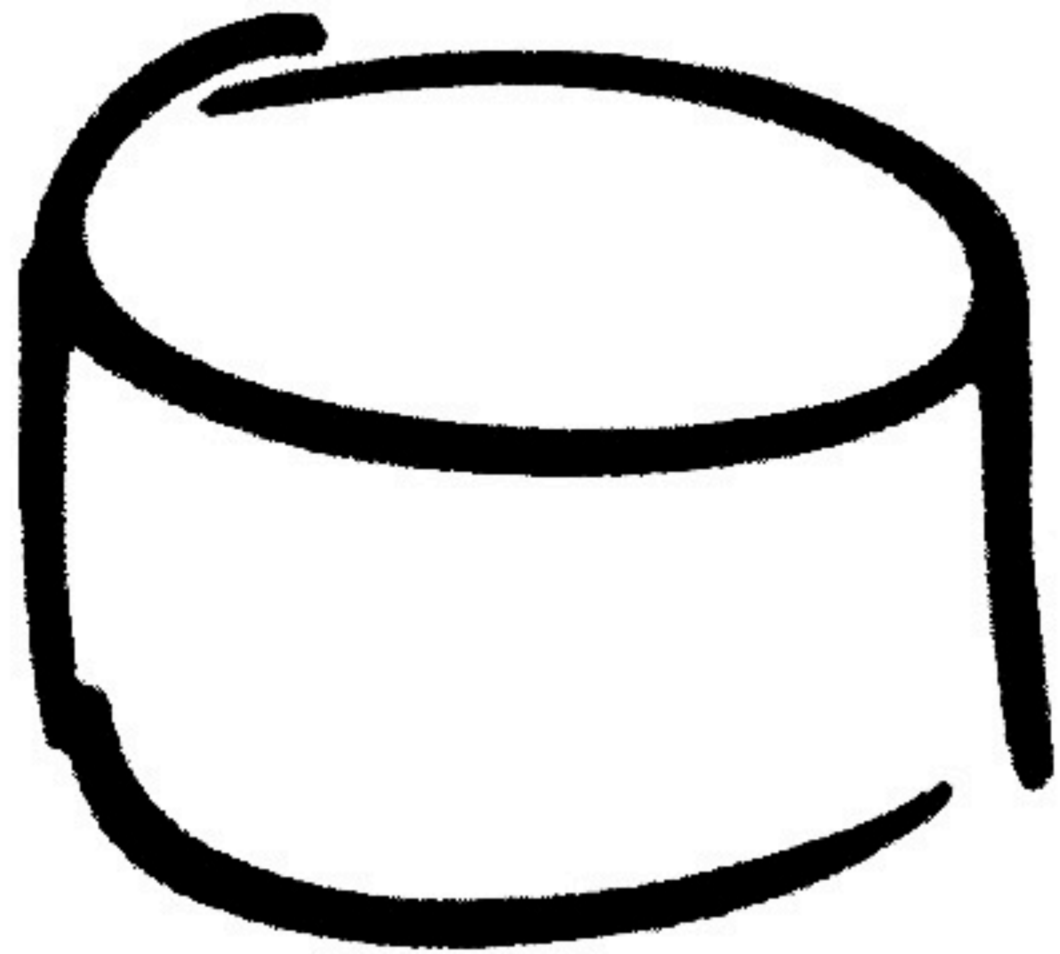
Oxford University

David Huard

McGill University

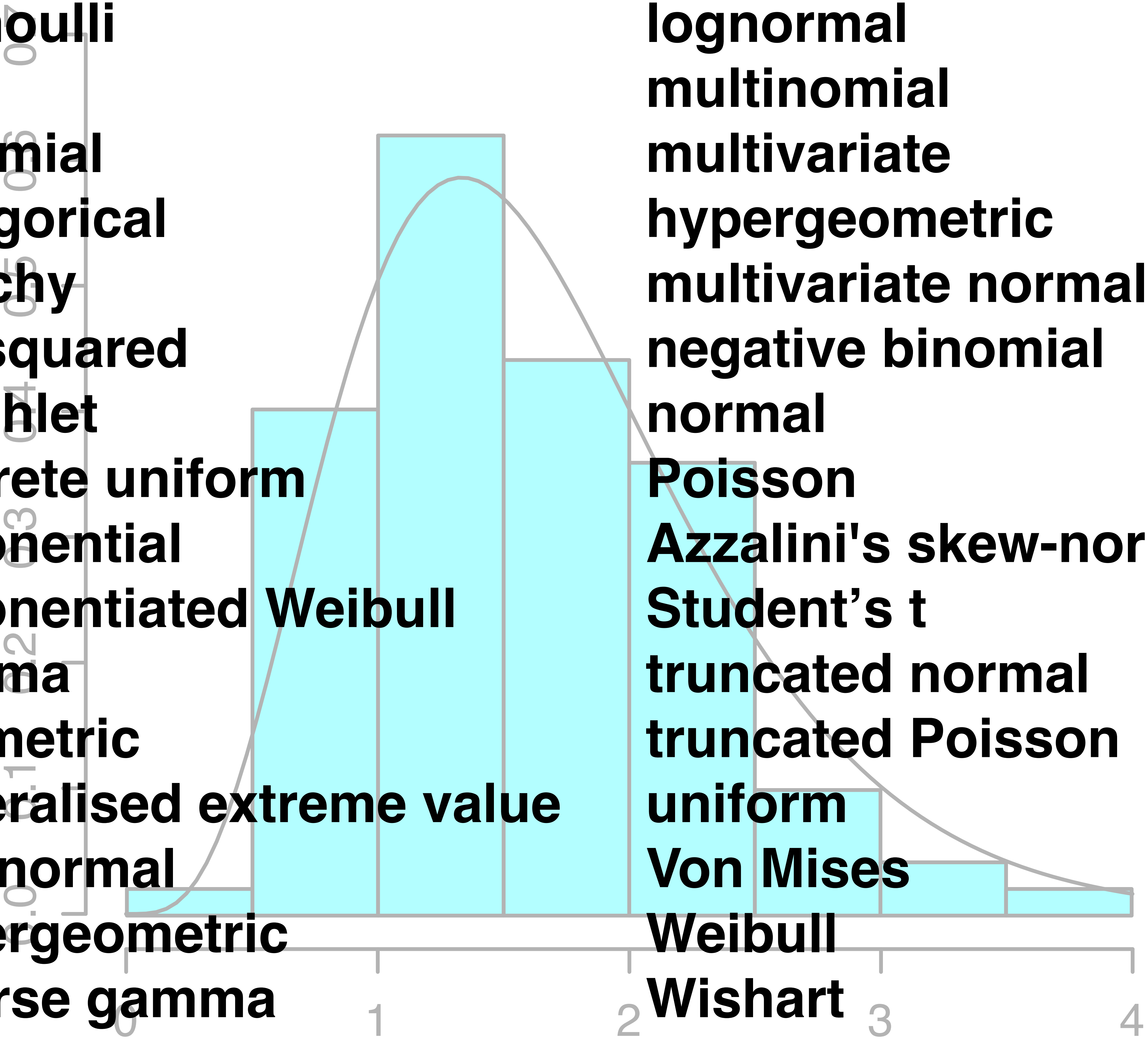

Scientific Computing Tools For Python — Numpy
http://numpy.scipy.org/
numpy
Pinboard Flickr iDisk SimpleText.ws PyMC Squarespace Login Open in Papers Share on Posterous Share with bit.ly Bookmark Read Later
NumPy
Download
Getting Started
Documentation
Report Bugs
Read the Blog
NumPy is the fundamental package needed for scientific computing with Python. It contains among other things:
• a powerful N-dimensional array object
• sophisticated (broadcasting) functions
• tools for integrating C/C++ and Fortran code
• useful linear algebra, Fourier transform, and random number capabilities.
Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases.
Getting Started
• Getting Numpy
• Installing NumPy and SciPy
• NumPy and SciPy documentation page
• NumPy Tutorial
• NumPy for MATLAB© Users
• NumPy functions by category
• NumPy Mailing List
More Information
• NumPy Sourceforge Home Page
• SciPy Home Page
• Interfacing with compiled code
• Older python array packages
NumPy

Database Backends



autoregressive lognormal
Bernoulli
beta
binomial
categorical
Cauchy
chi-squared
Dirichlet
discrete uniform
exponential
exponentiated Weibull
gamma
geometric
generalised extreme value
half-normal
hypergeometric
inverse gamma

Laplace
lognormal
multinomial
multivariate
hypergeometric
multivariate normal
negative binomial
normal
Poisson
Azzalini's skew-normal
Student’s t
truncated normal
truncated Poisson
uniform
Von Mises
Weibull
Wishart



FORTRAN
log-probabilities

```
      SUBROUTINE t(x,nu,n,nnu,like)

c Student's t log-likelihood function

cf2py integer dimension(n),intent(in) :: x
cf2py double precision dimension(nnu),intent(in) :: nu
cf2py double precision intent(out) :: like
cf2py integer intent(hide),depend(x) :: n=len(x)
cf2py integer intent(hide),depend(nu) :: nnu=len(nu)
cf2py threadsafe

      IMPLICIT NONE
      INTEGER n, i, nnu
      INTEGER x(n)
      DOUBLE PRECISION nu(nnu), like, infinity, nut
      PARAMETER (infinity = [illegible])
      DOUBLE PRECISION gammln
      DOUBLE PRECISION PI
      PARAMETER (PI=3.141592653589793238462643d0)

      nut = nu(1)

      like = 0.0
      do i=1,n
        if (nnu .GT. 1) then
          nut = nu(i)
        endif
```

bioassay.py

```python
from pymc import *
from numpy import ones, array

n = 5*ones(4,dtype=int)
dose = array([-.86,-.3,-.05,.73])
response = array([0,1,3,5])

# Priors on unknown parameters
alpha = Normal('alpha', mu=0.0, tau=0.01)
beta = Normal('beta', mu=0.0, tau=0.01)

@deterministic
def theta(a=alpha, b=beta, d=dose):
    """theta = inv_logit(a+b)"""
    return invlogit(a+b*d)

# deaths ~ binomial(n, p)
deaths = Binomial('deaths', n=n, p=theta, value=response, observed=True)

```

Line: 10 Column: 39 Python Soft Tabs: 4