Monte Carlo integration

- Example of a Monte Carlo sampler in 2D:
  - imagine a circle (radius L/2) within a square of LxL.
  - If points are randomly generated over the square, what's the probability to hit within circle?
  - By algebra: $\pi(L/2)^2/L^2 = \pi/4$.
  - By simulation:

$$P(\theta \in S) \approx \frac{1}{K}\sum_{k=1}^{K} 1_{\{\theta \in S\}}(\theta^k)$$

  - This also provides a Monte Carlo approx of π.

Monte Carlo integration

- Wanted: e.g. posterior mean

$$E(\theta \mid X) = \int \theta p(\theta \mid X) d\theta$$

- But assume we do not have conjugate priors, no closed form solution.
- Could try numerical integration methods.
- Or Monte Carlo: draw random (i.i.d) samples $\boldsymbol{\theta}^{\mathbf{k}}$ from the distribution, k=1,….,K. (large K).

$$E(\theta \mid X) \approx \frac{1}{K} \sum_{k=1}^{K} \theta^k \qquad \theta^k \sim p(\theta \mid X)$$

Monte Carlo integration

- Even if we had solved the density, it can be difficult to evaluate E(g(θ) |X)
- Note also:

$$E(1_{\{\theta \in S\}}(\theta) \mid X) = 1P(\theta \in S \mid X) + 0P(\theta \notin S \mid X) = P(\theta \in S \mid X)$$

- So that we can approximate probabilities by $$P(\theta \in S \mid X) \approx \frac{1}{K} \sum_{k=1}^{K} 1_{\{\theta \in S\}}(\theta^k)$$

- And likewise any quantiles.

Monte Carlo integration

- What remains is to generate samples from the correct distribution.
- This is easy for known distributions, just read the manual of your software.
- N-dimensional distributions? Non-standard distributions?

- → Need other sampling methods than directly drawing i.i.d.

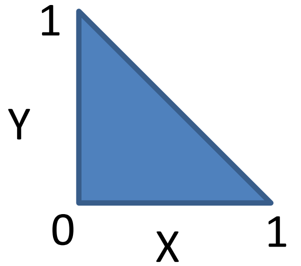
Monte Carlo Markov chain

Innovation:

- Construct a sampler that works as a Markov chain, for which stationary distribution exists, and this stationary distribution is the same as our target distribution.

Gibbs sampler

- Gibbs sampling in 2D
  - Example: uniform distribition in a triangle.



$$f(x,y) = 2 \times \mathbf{1}_{\{y<1-x,\, 0<x<1,\, 0<y<1\}}(x,y)$$

Gibbs sampler

- Gibbs sampling in 2D
  - Remember product rule:

    p(x,y) = p(x|y)p(y) = p(y|x)p(x)
  - Solve the marginal density p(x)

$$p(x) = \int_0^1 p(x,y)dy$$

$$= \int_0^1 2 \times \mathbf{1}_{\{y<1-x,\, 0<x<1,\, 0<y<1\}}(x,y)dy = \int_0^{1-x} 2\, dy = 2(1-x)$$

  - Then: p(y|x)=p(x,y)/p(x)

Gibbs sampler

- Gibbs sampling in 2D
  - Solve the conditional density:

$$p(y \mid x) = \frac{p(x,y)}{p(x)} = \frac{2 \times \mathbf{1}_{\{y<1-x,\,0<x<1,\,0<y<1\}}(x,y)}{2(1-x)}$$

$$= \frac{1}{1-x}\mathbf{1}_{\{y<1-x,\,0<y<1\}}(y) \quad = \; U(0,1-x)$$

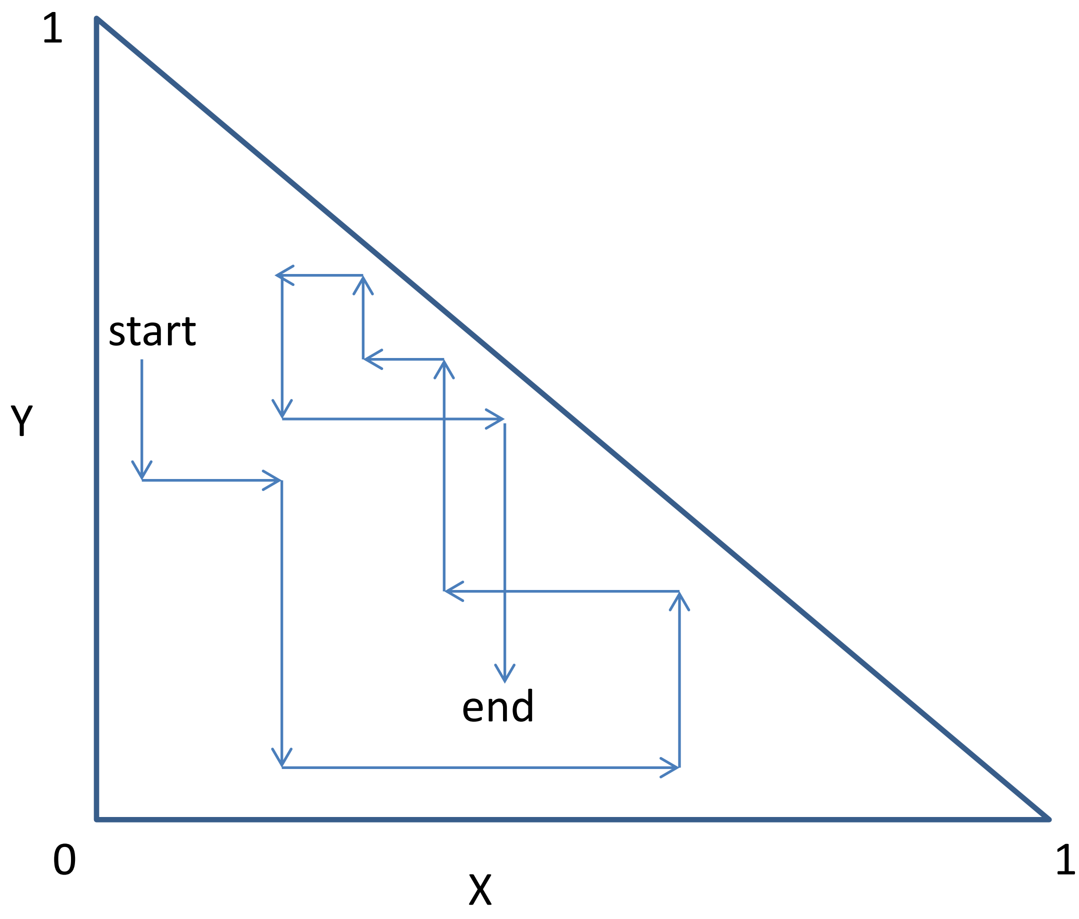
  - Note: above it would suffice to recognize p(y|x) up to a constant term, so that solving p(x) is not necessary.
  - Similarly, get p(x|y) = U(0,1-y).

Gibbs sampler

- Gibbs sampling in 2D
  - Starting from the joint density p(x,y), we have obtained two important conditional densities: p(x|y) and p(y|x)
  - Gibbs algorithm is then:
  - (1) start from x^0,y^0. Set k=1.
  - (2) sample x^k from $p(x|y^{k-1})$
  - (3) sample y^k from $p(y|x^k)$. Set k=k+1.
  - (4) go to (2) until sufficiently large sample.

  - These samples are no longer i.i.d.

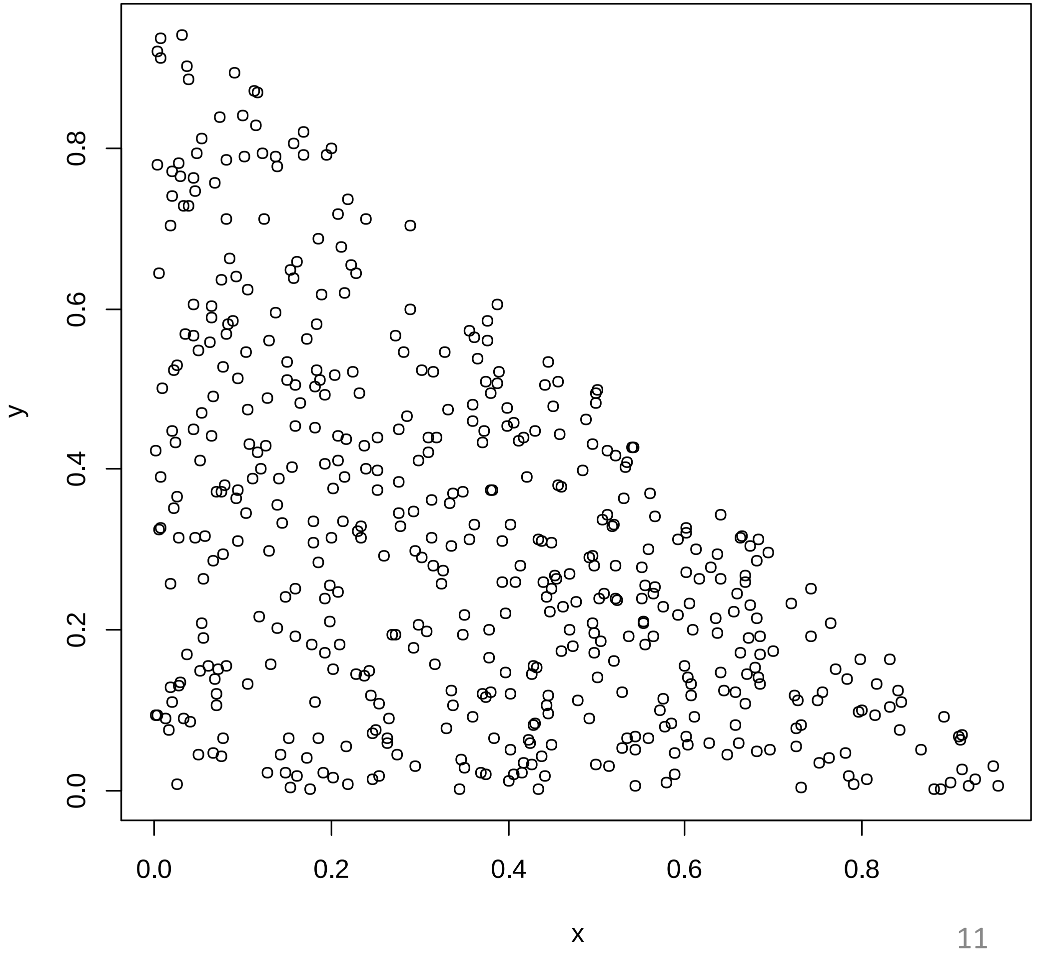
Gibbs sampler



Gibbs sampler

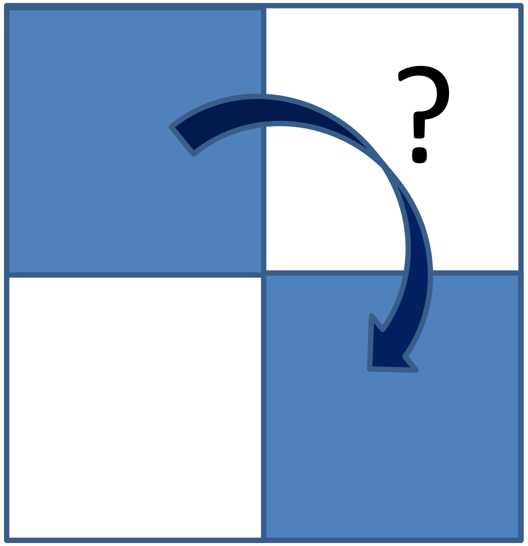
- **In R, you could:**

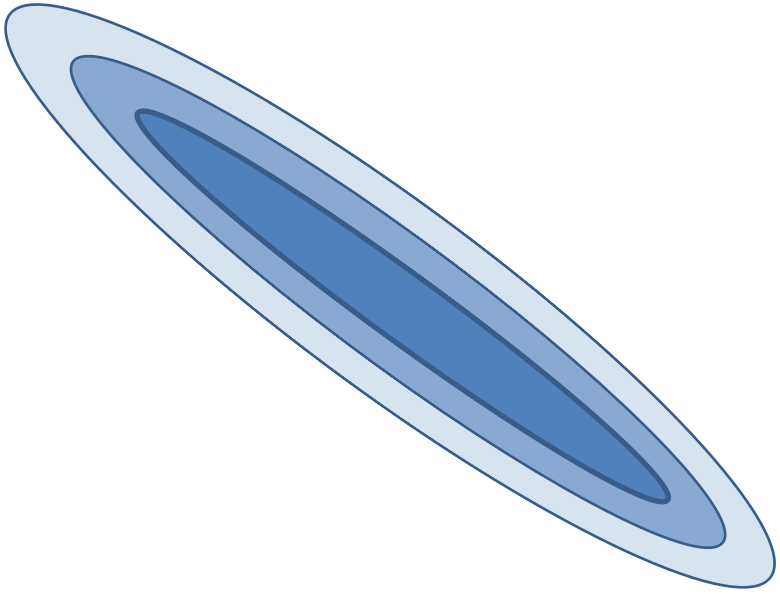
```
x <- numeric()
y <- numeric()
x[1] <- 0.5
y[1] <- 0.4
for(i in 2:500){
    y[i] <- runif(1,0,1-x[i-1])
    x[i] <- runif(1,0,1-y[i])
    }
plot(x,y)
```



Gibbs sampler

- **Jumping around? Possible problems.**





Gibbs sampler

- **Consider again the binomial model, ”conditional to N”**
  - **Joint distribution $p(\theta,X|N)$ can be expressed either as $p(X|\theta,N)p(\theta|N)$ or $p(\theta|X,N)p(X|N)$.**
  - **From the first, we recognize $p(X|\theta,N)=Bin(N,\theta)$ with e.g. uniform prior $p(\theta|N)=p(\theta)$. Then, we would know $p(\theta|X,N) = Beta(X+1,N-X+1)$.**

  - **This gives $p(\theta|X)$ and $p(X|\theta)$ for Gibbs.**

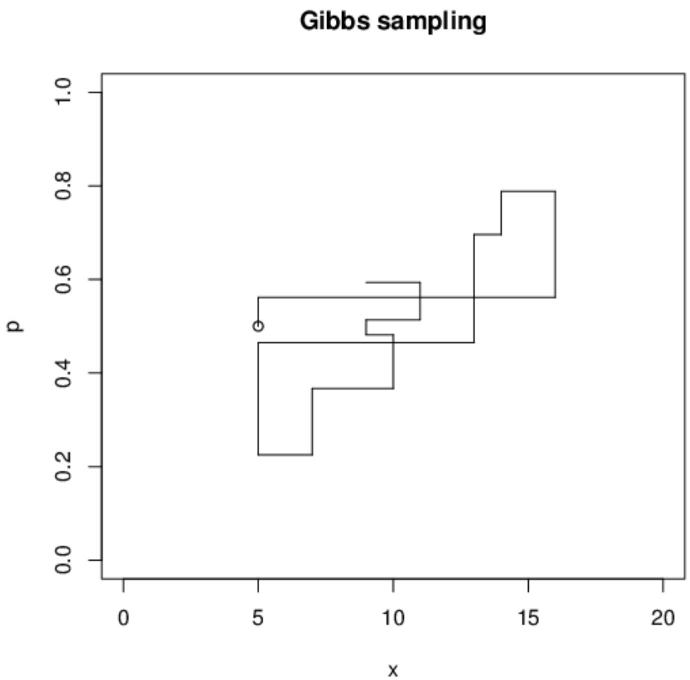
Gibbs sampler

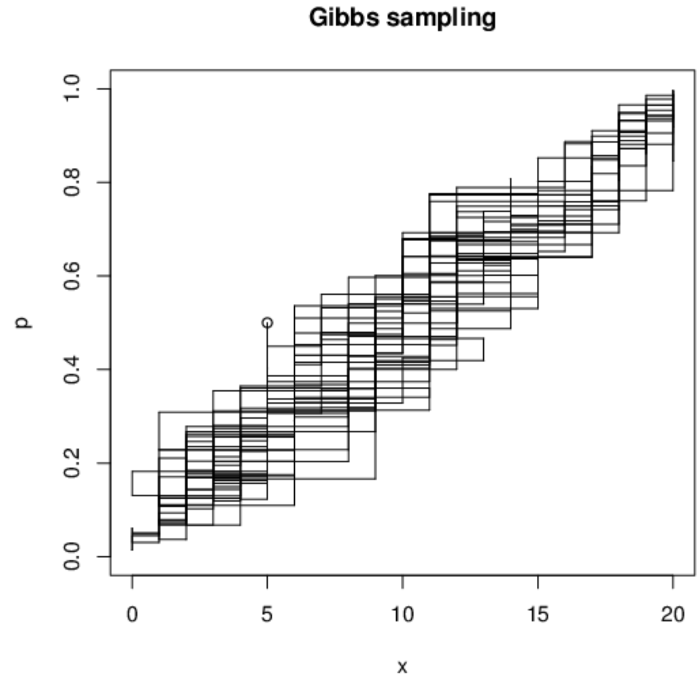
- **Consider again the binomial model, ”conditional to N”**
  - **Gibbs sampling (X,θ) gives the joint distribution of X and θ.**
  - **[We know both conditional densities, but it would be also possible to obtain p(θ|X) by Monte Carlo sampling from the joint p(θ,X), and then accepting only those (θ,X)-pairs for which X takes a given value. This idea is used in Approximate Bayesian Computation (ABC).]**

Gibbs sampler

- **Binomial model, ”conditional to N”, in R:**

```
n<-20; p <- numeric(); x<- numeric()
p[1] <- 0.5; x[1] <- 10   # initial values
for(i in 2:1000){
p[i] <- rbeta(1,x[i-1]+1,n-x[i-1]+1)
x[i] <- rbinom(1,n,p[i])
}
plot(x,p)
```





Gibbs and normal density

- **2D normal density:**

$$\begin{bmatrix} \mathbf{X} \\ \mathbf{Y} \end{bmatrix} \sim N\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix} \right)$$

- **Marg. densities p(x) and p(y) are both N(0,1)**

$$p(x,y) = \frac{1}{2\pi\sqrt{1-\rho^2}} \exp\left(-\frac{1}{2(1-\rho^2)}(x^2 - 2\rho xy + y^2)\right)$$

- **Conditional density p(y|x)=p(x,y)/p(x) is**

$$p(y \mid x) = \frac{1}{2\pi\sqrt{1-\rho^2}} \exp\left(-\frac{1}{2(1-\rho^2)}(\rho x - y)^2\right) = N(\rho x, 1-\rho^2)$$

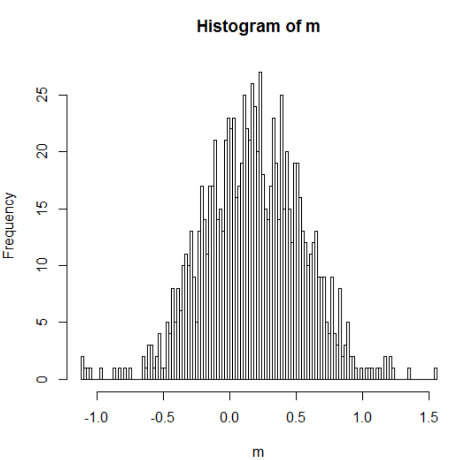
Gibbs and normal density

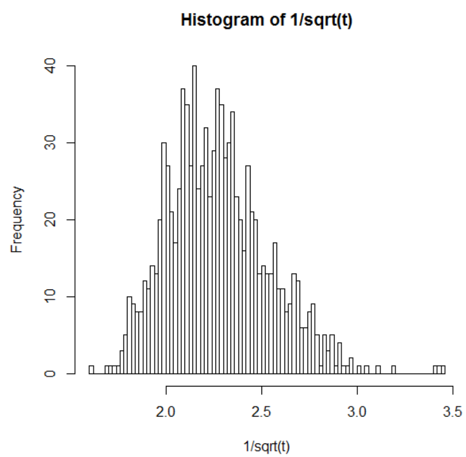
- **Gibbs would then be sampling from:**
  - **$p(y|x) = N(\rho x, 1-\rho^2)$**
  - **$p(x|y) = N(\rho y, 1-\rho^2)$**
  - **This can mix slowly if X & Y heavily correlated.**
- **Recall the posterior $p(\mu,\sigma \mid X_1,\ldots,X_n)$**
  - **This is a 2D problem.**
  - **Assume improper prior** $p(\mu,\sigma) \propto 1/\sigma^2$
  - **Then we can solve $p(\mu|\sigma, X) = N(\Sigma X_i/n, \sigma^2/n)$**
  - **And $p(\tau|\mu, X) = \text{gamma}(n/2, 0.5\,\Sigma(X_i-\mu)^2)$**

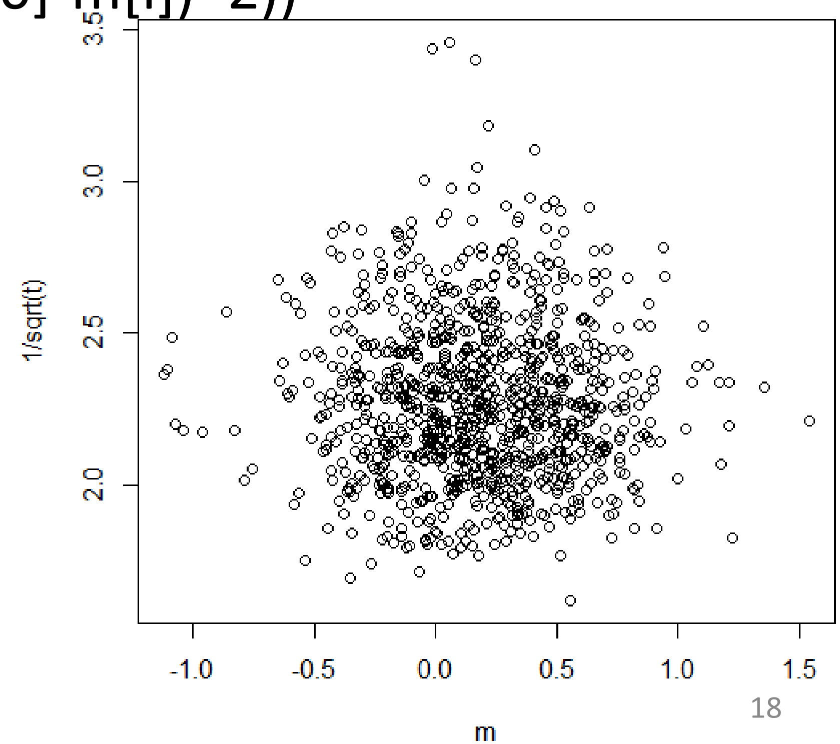
→ **This makes Gibbs! (try this with R)**

Next time you estimate μ,σ^2 from a sample $X_1,\ldots,X_n$, assuming normal model, try sampling the posterior distribution:

```
X <- rnorm(40,0,2) # generate example dataset, n=40, mean=0,sd=2
m[1] <- mean(x);  t[1] <- 1/(sd(x)*sd(x))   # initial values
for(i in 2:1000){   # Gibbs sampling
m[i] <- rnorm(1,mean(x),sqrt((1/t[i-1])/40));
t[i] <- rgamma(1,40/2,0.5*sum((x[1:40]-m[i])^2))
}
```







Metropolis-Hastings

- **This is a very general purpose sampler**
- **The core is: ’proposal distribution’ and ’acceptance probability’.**

- **At each iteration:**
  - **Random draw is obtained from proposal density $Q(\theta^* | \theta^{i-1})$, which can depend on previous iteration.**
  - **Simply, it could be $U(\theta^{i-1} - L/2, \theta^{i-1} + L/2)$.**

Metropolis-Hastings

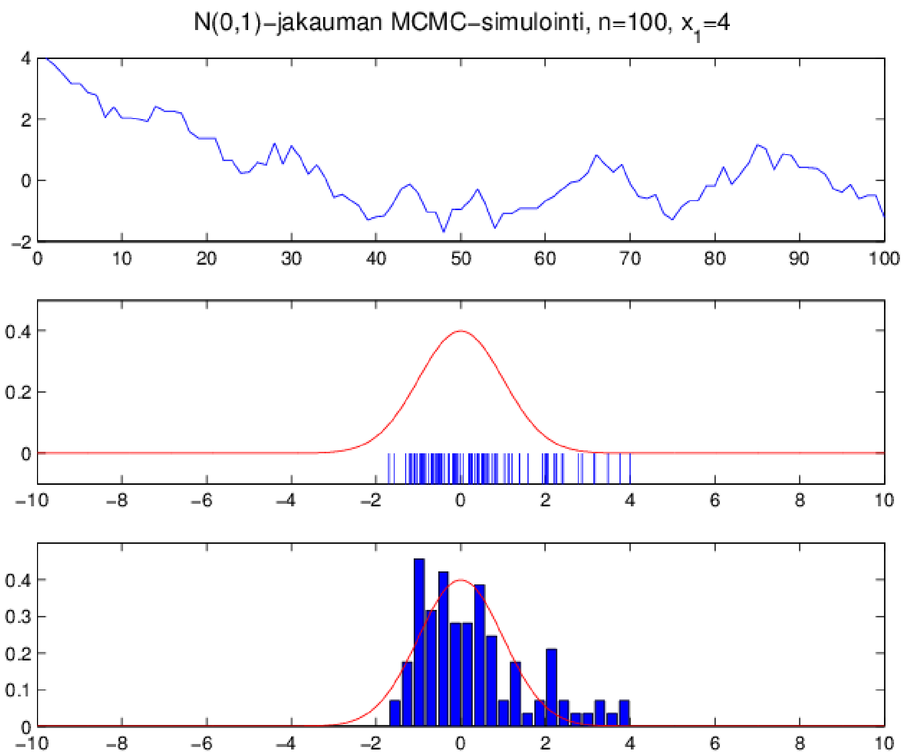
- **At each iteration:**
  - **Proposal is accepted with probability**

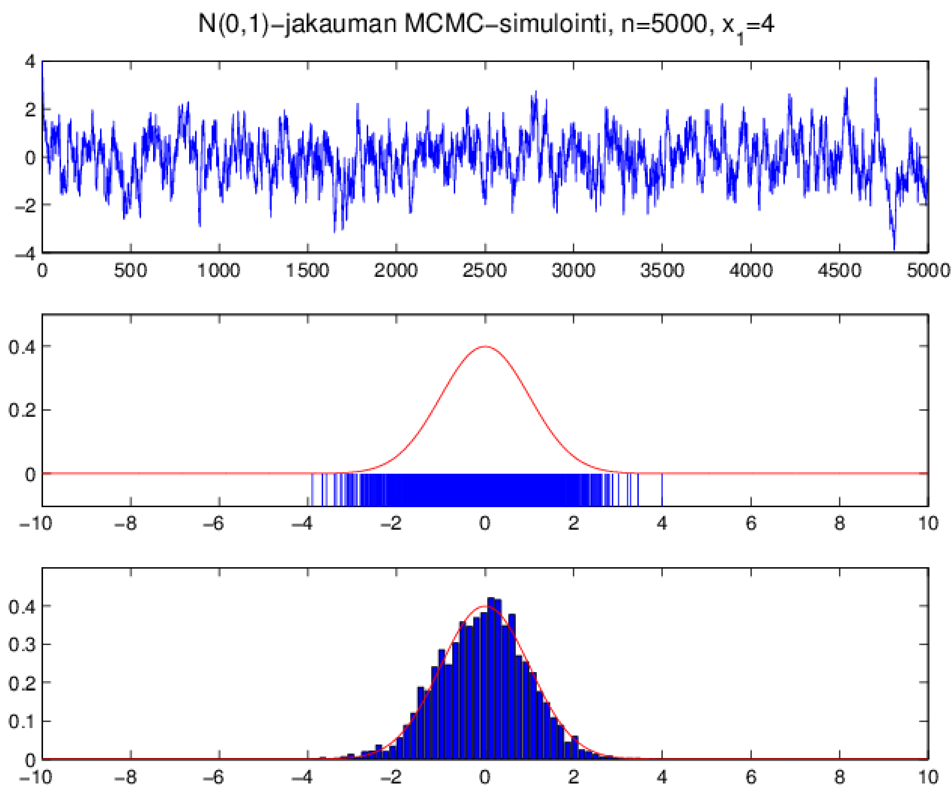
$$r = \min\left(\frac{p(\theta^* \mid data) Q(\theta^{i-1} \mid \theta^*)}{p(\theta^{i-1} \mid data) Q(\theta^* \mid \theta^{i-1})}, 1 \right)$$

  - **Note how little we need to know about p(θ|data)!**
    - **Normalizing constant cancels out from the ratio.**
    - **Enough to be able to evaluate prior and likelihood terms.**
    - **Proposals too far → accepted rarely → slow sampler**
    - **Proposals too near → small moves → slow sampler**
    - **Acceptance probability ideally about 20%-40%**
  - **Gibbs sampler is a special case of MH-sampler**
    - **In Gibbs, the acceptance probability is 1.**
    - **Block sampling also possible.**

Metropolis-Hastings

- **Sampling from N(0,1), using MH-algorithm:**





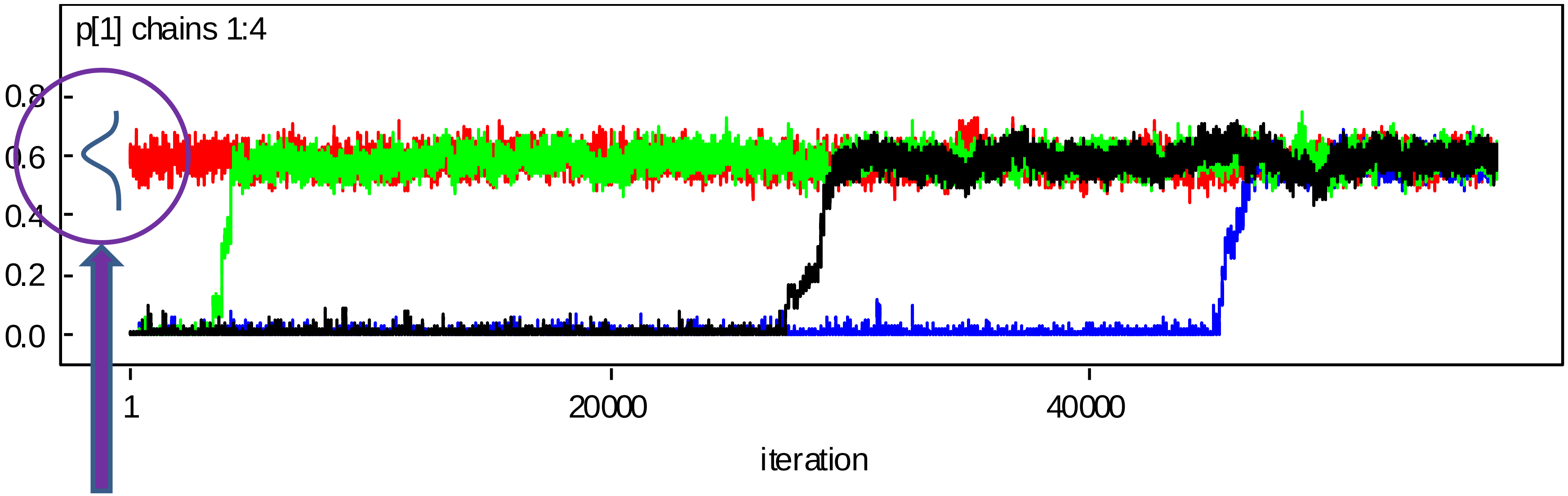
MCMC convergence

- **Remember to monitor for convergence!**
  - **Chain is only approaching the target density, when iterating a long time, k→∞.**
  - **Convergence can be very slow in some cases.**
  - **Autocorrelations between iterations are then large → makes sense to take a thinned sample.**
  - **Systematic patterns, trends, sticking, indicate problems.**

  - **Pay attention to starting values! Try different values in different MCMC chains. (discard burn-in period).**

MCMC convergence

- **Can only diagnose poor convergence, but cannot fully prove a good one! (e.g. multimodal densities).**



Target density

MCMC in BUGS

- **Many different samplers, some of them are implemented in WinBUGS/OpenBUGS.**
- → Next, we leave the sampling for BUGS, and only consider **building the models** (which define a posterior distribution), and **running the sampling** in BUGS, and **assessing the results**.