Yet another intro to MCMC
Focusing on 2 main methods

Konstantinos Pateras

1. University of Thessaly, Department of Public Health
2. University of Utrecht, Department of Biostatistics

kostas.pateras@gmail.com

September 20, 2021

What to expect?

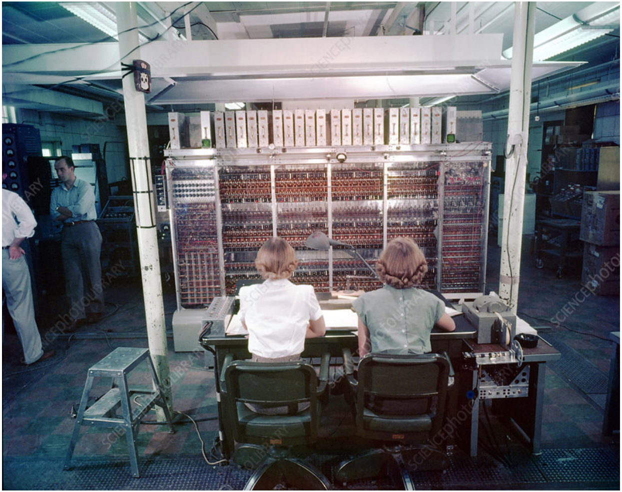
Go to www.menti.com

Why bother?

Why bother?

- Monte Carlo: Physical sciences (Astrophysics), Engineering (Thermodynamics), Climate change, Computational biology (Phylogenetics), Applied statistics (Bayesian approach), Artificial intelligence, among many
- MCMC: Computational statistics, computational physics and computational biology among others. Hierarchical modelling in Bayesian statistics

MANIAC I




MANHATTAN PROJECT
A
BOMB

Charles III of Monaco

Historical notes

- Monte carlo methods since 1920 (Enrico Fermi)
- Named after Nicholas Metropolis, Monte Carlo casino.

Historical notes

- Monte carlo methods since 1920 (Enrico Fermi)
- Named after Nicholas Metropolis, Monte Carlo casino.
- Los Alamos labs 1940s (Stanislaw Ulam, John von Neumann, Nicholas Metropolis and others)

Historical notes

- Monte carlo methods since 1920 (Enrico Fermi)
- Named after Nicholas Metropolis, Monte Carlo casino.
- Los Alamos labs 1940s (Stanislaw Ulam, John von Neumann, Nicholas Metropolis and others)
- Metropolis-Hastings 1970, Geman Geman Gibbs (1984)
- Mainstream in stats by Gelman and Rubin 1992 among others
- Mainstream software (BUGS 1996 (Gibbs), JAGS (Slice), Stan (Hamiltonian))

Basic notions

1. How to generate completely random numbers (not easy). R uses pseudo-random number generators.
2. Main assumption: Be able to generate from a Uniform(0,1)

Basic notions

1. How to generate completely random numbers (not easy). R uses pseudo-random number generators.
2. Main assumption: Be able to generate from a Uniform(0,1)
3. Inverse transform method, inv-CDF [when easy to implement] - e.g. (1. $f(x) = 3x^2$ or 2. $x \sim Exp(\lambda)$), (Rizzo pg 49)
4. Quiz: Exercise 1 in GitHub "IntroMCMC.r". Interact in groups of 2-3.

Basic notions

1. How to generate completely random numbers (not easy). R uses pseudo-random number generators.
2. Main assumption: Be able to generate from a Uniform(0,1)
3. Inverse transform method, inv-CDF [when easy to implement] - e.g. (1. $f(x) = 3x^2$ or 2. $x \sim Exp(\lambda)$), (Rizzo pg 49)
4. Quiz: Exercise 1 in GitHub "IntroMCMC.r". Interact in groups of 2-3.
5. Bayes Theorem, Simple Monte Carlo

- $p(\theta|x) = \dfrac{p(\theta) \cdot p(x|\theta)}{p(x)} = \dfrac{p(\theta) \cdot p(x|\theta)}{\int_\theta p(x|\theta)p(\theta)}$
- $p(\theta|x) \propto p(\theta) \cdot p(x|\theta)$
- (Quiz) Compute a Monte Carlo estimate of $\int_0^1 e^{-x}dx$ and compare it to its exact value $(1 - e^{-1})$.

Acceptance-Rejection method

To generate a random variable Y, suppose Y and X are random variables with density f and g. Also, there exists a constant c such that $f(t)/g(t) \leq c, f(t) > 0$

Acceptance-Rejection method

To generate a random variable Y, suppose Y and X are random variables with density f and g. Also, there exists a constant c such that $f(t)/g(t) \leq c, f(t) > 0$

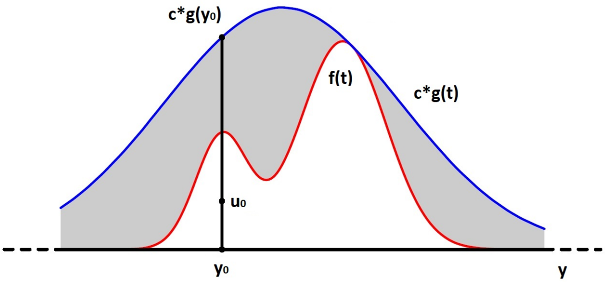
1. Find and generate Y with g for which $f(t) \leq c \cdot g(t), f(t) > 0$
    2. Generate a random $y^* \sim g$
    3. Generate a random $u \sim Uniform(0, 1)$

Acceptance-Rejection method

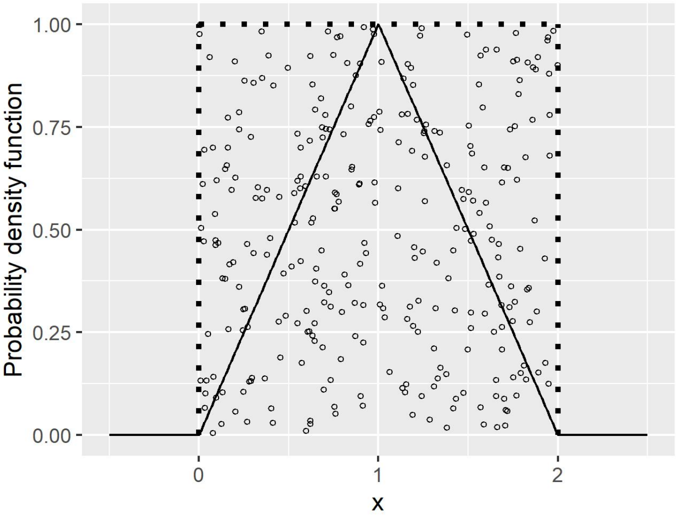
To generate a random variable Y, suppose Y and X are random variables with density f and g. Also, there exists a constant c such that $f(t)/g(t) \leq c, f(t) > 0$

1. Find and generate Y with g for which $f(t) \leq c \cdot g(t), f(t) > 0$
   2. Generate a random $y^* \sim g$
   3. Generate a random $u \sim Uniform(0, 1)$
   4. If $u < f(y^*)/(c \cdot g(y^*))$ accept y^* as sample, otherwise reject y^*.
   5. Repeat from step 2).

Acceptance-Rejection figure



Acceptance-Rejection (quiz)



Acceptance-Rejection method

Example (Triangle probability density)

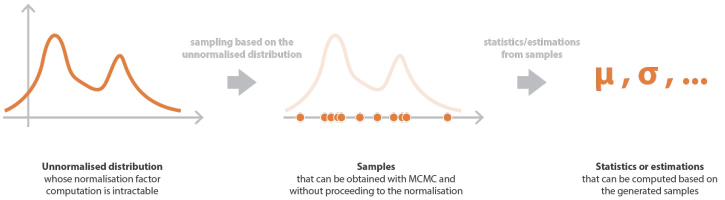
```
triangle.pdf = function(x){ # Definition of triangle pdf.
ifelse((0 < x) & (x < 1), x,
ifelse((1 <= x) & (x < 2), 2 - x, 0))
}

#simulates a sample of size (n) from the pdf f(x) via the acceptance/rejection algorithm
accept_reject = function(fx, n = 100, prop.val=1) {

x = numeric(n);count = 0

while(count < n) {
temp.x <- runif(1, 0, 2)
y <- runif(1, 0, 1)

if (y < fx(temp.x)) {
count = count + 1
x[count] <- temp.x
}

}

return(x)
}
sample = accept_reject(triangle.pdf, 50000)
length(sample)/50000 # Probability of acceptance
plot(density(sample), adjust=0.5)
```

Markov Chains

Some brief notions

- First order Markov Chains (Depend on last value)
- Irreducibility (Any state can reach another state)
- Aperiodicity
- (Positive) recurrent
- Stationary distribution

Markov Chains Concept



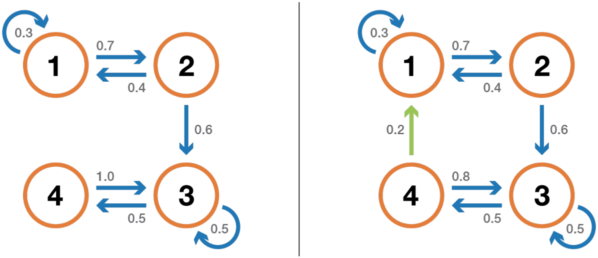
Source: Joseph Rocca, Bayesian inference problem, MCMC and variational inference

Markov Chains Concept 2

Source: Joseph Rocca, Bayesian inference problem, MCMC and variational inference

Markov Chains Assumption 1

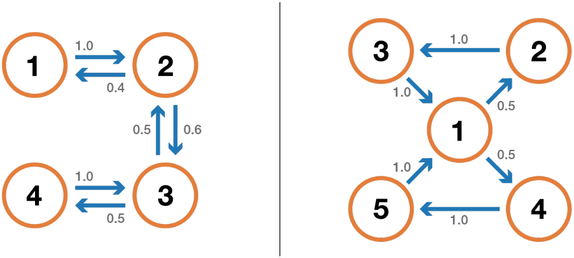
Illustration of the irreducibility property. The chain on the left is not irreducible: from 3 or 4 we can't reach 1 or 2. The chain on the right (one edge has been added) is irreducible: each state can be reached from any other state.



Source: Joseph Rocca, Bayesian inference problem, MCMC and variational inference

Markov Chains Assumption 2

Illustration of the aperiodicity property. The chain on the left is 2-periodic: when leaving any state, it always takes a multiple of 2 steps to come back to it. The chain on the right is 3-periodic.



Source: Joseph Rocca, Bayesian inference problem, MCMC and variational inference

Metropolis background

Metropolis with symmetric proposal (Random-walk metropolis) - If transition is reversible then target is invariant/stationary.

Metropolis background

Metropolis with symmetric proposal (Random-walk metropolis) - If transition is reversible then target is invariant/stationary.

1. Choose a proposal distribution $q(\theta|\theta^*) = q(\theta^*|\theta) = N(\theta, s_\theta)$ [Irreducible, positive recurrent, aperiodic] and set initial $\theta = \theta_0$
2. Repeat until chain has converged

Metropolis background

Metropolis with symmetric proposal (Random-walk metropolis) - If transition is reversible then target is invariant/stationary.

1 Choose a proposal distribution $q(\theta|\theta^*) = q(\theta^*|\theta) = N(\theta, s_\theta)$ [Irreducible, positive recurrent, aperiodic] and set initial $\theta = \theta_0$

2 Repeat until chain has converged

  2a Generate θ^* from $N(\theta, s_\theta)$, what about s_θ?

  2b Generate U from Uniform(0,1)

Metropolis background

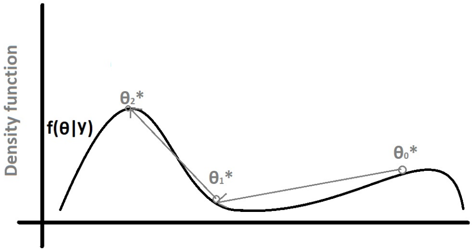
Metropolis with symmetric proposal (Random-walk metropolis) - If transition is reversible then target is invariant/stationary.

1 Choose a proposal distribution $q(\theta|\theta^*) = q(\theta^*|\theta) = N(\theta, s_\theta)$ [Irreducible, positive recurrent, aperiodic] and set initial $\theta = \theta_0$

2 Repeat until chain has converged

- 2a Generate θ^* from $N(\theta, s_\theta)$, what about s_θ?
- 2b Generate U from Uniform(0,1)
- 2c If $U \leq \frac{f(\theta^*|y)}{f(\theta|y)} = \frac{f(y|\theta^*)p(\theta^*)}{f(y|\theta)p(\theta)}$, accept θ^* and set $\theta = \theta^*$, otherwise set $\theta = \theta$
- 2d Increment t and repeat

Metropolis figure

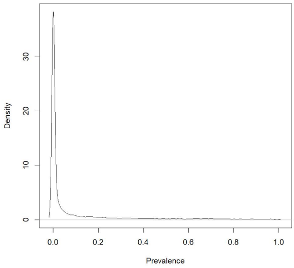


Metropolis algorithm

- Based on expert beliefs the prior probability of an adverse effect can be summarized via a Beta distribution opinion $\theta \sim Beta(\alpha = 0.1, \beta = 1)$. Prior beliefs can be expressed as

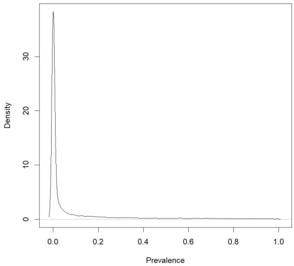
Metropolis algorithm

- Based on expert beliefs the prior probability of an adverse effect can be summarized via a Beta distribution opinion $\theta \sim Beta(\alpha = 0.1, \beta = 1)$. Prior beliefs can be expressed as



Metropolis algorithm

- Based on expert beliefs the prior probability of an adverse effect can be summarized via a Beta distribution opinion $\theta \sim Beta(\alpha = 0.1, \beta = 1)$. Prior beliefs can be expressed as



- Toy example: Prevalence of moderate adverse events of Covid-19 vaccine. Observed 50 events in a sample of 10,000, $y \sim B(50, 10^4)$.

Metropolis algorithm

Example (Theorem Slide Code)

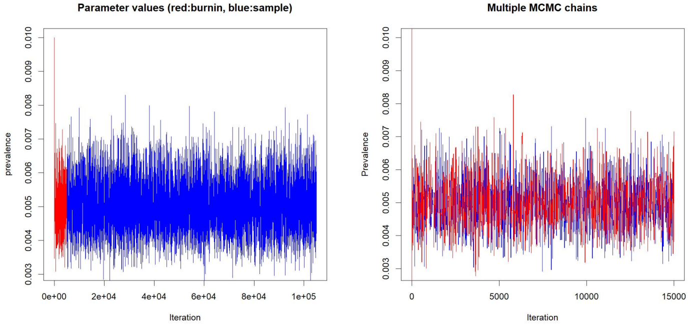
```
metropolis <- function(burnin = 0, sample = 10000,
sigma = 0.05, initial_value = 0.05, plot=TRUE)

stopifnot(initial_value > 0, initial_value < 1)
stopifnot(sigma > 0)
burnin <- as.integer(burnin)
sample <- as.integer(sample)
stopifnot(burnin >= 0)
stopifnot(sample > 0)

# Redefine these to work on the log scale:
llikelihood_fun <- function(prevalence) dbinom(50, 10^4, prevalence, log=TRUE)
lprior_fun <- function(prevalence) dbeta(prevalence, 0.1, 1, log=TRUE)

parameters <- numeric(burnin+sample)
parameters[1] <- initial_value
current <- initial_value

post <- llikelihood_fun(current) + lprior_fun(current)
```

Metropolis algorithm

Example (Theorem Slide Code)

```
for(i in 2:(burnin+sample)){

proposal <- rnorm(1, current, sigma)

if(proposal > 0 \& proposal < 1){
U=log(runif(1,0,1))
newpost <- llikelihood_fun(proposal) + lprior_fun(proposal)
accept <- U <= newpost-post

if(accept){
current <- proposal
post <- newpost
}
}
parameters[i] <- current
}
```

Metropolis results



- ESS: 10000/647.67 = 15.43 iterations for an independent sample
- Geweke diagnostic: p-value=0.67
- Autocorrelation? How to fix this?

Gibbs sampler background

Named after JW Gibbs, invented 80 years after his death.
Why Gibbs? When it should be preferred?

Gibbs sampler background

Named after JW Gibbs, invented 80 years after his death.
Why Gibbs? When it should be preferred?

1 Set initial θ_0 at time $t = 0$

2 For each iteration $t = 1, 2, ...T$ repeat

    2a Set $\theta = \theta^{(t-1)}$

Gibbs sampler background

Named after JW Gibbs, invented 80 years after his death.
Why Gibbs? When it should be preferred?

1 Set initial θ_0 at time $t = 0$

2 For each iteration $t = 1, 2, ...T$ repeat

  2a Set $\theta = \theta^{(t-1)}$

  2b For each parameter $j = 1, 2..., d$

    2bi Generate θ_j from $\theta_j \sim f(\theta_j|\theta_{/j}, y)$

Gibbs sampler background

Named after JW Gibbs, invented 80 years after his death. Why Gibbs? When it should be preferred?

1 Set initial θ_0 at time $t = 0$

2 For each iteration $t = 1, 2, ...T$ repeat

  2a Set $\theta = \theta^{(t-1)}$

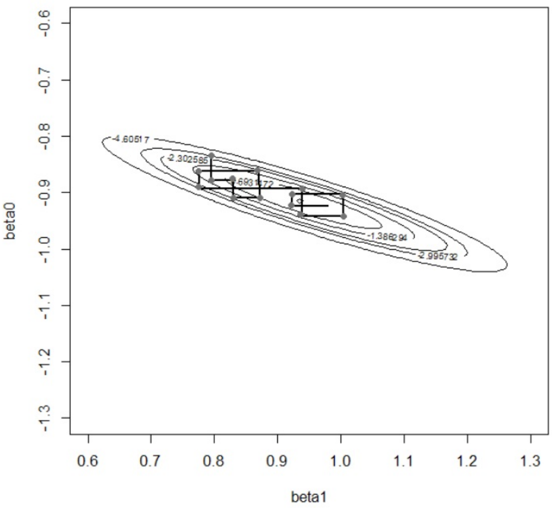
  2b For each parameter $j = 1, 2..., d$

    2bi Generate θ_j from $\theta_j \sim f(\theta_j | \theta_{/j}, y)$

  2c Increase t and repeat

One component is updated at a time. Conditionally conjugate models and others.

Gibbs sampler figure



Gibbs sampler

1. Example with a normal variable indicating average temperature for July in a time-span of 12 years. (Ntzoufras 2009)
   - $\mu \sim N(\mu_0, \sigma_0^2)$ and $\sigma^2 \sim IG(a_0, b_0)$
   - Compute conditional distributions $f(\mu|\sigma^2, y)$ and $f(\sigma^2|\mu, y)$
   - $\mu|\sigma^2, y \sim N(w\tilde{y} + (1-w)\mu_0, w\dfrac{\sigma^2}{n})$, $w = \dfrac{\sigma_0^2}{\sigma^2/n + \sigma_0^2}$
     $\sigma|\mu, y \sim IG[(a_0) + n/2, b_0 + 1/2\sum_1^n (y_i - \mu)^2]$

Gibbs sampler

Example (Gibbs with non-conjugate priors)

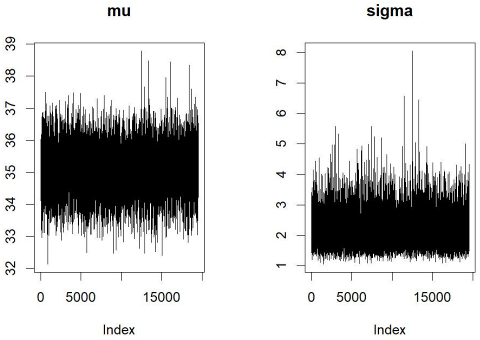
```
y<-c(32,36,37,34,38,36,33,36,37,35,32,35)
bary<-mean(y); n<-length(y)  # Set up data

# Set number of iterations
Iterations<-5000

# Set prior parameters.
mu0<-0; s0<-100; a0<-0.001; b0<-0.001

# Initialize vectors of sample values mu and sigma
theta <- matrix(nrow=Iterations, ncol=2)

# Set initial current mu(0) and sigma(0)
cur.mu<-0; cur.tau<-1; cur.s<-sqrt(1/cur.tau)
```

Gibbs sampler

Example (Gibbs with non-conjugate priors)

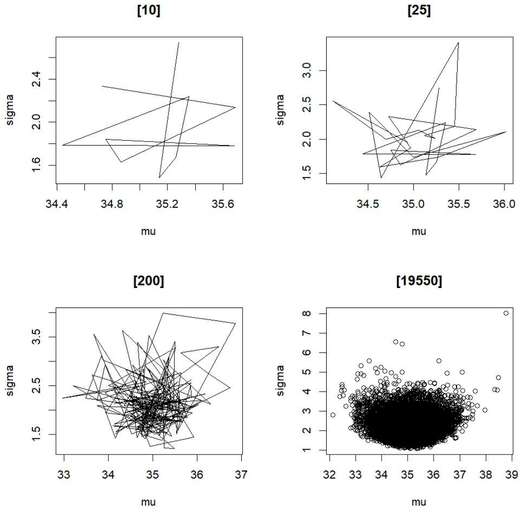
```
for (t in 1:Iterations){  # Repeat for t=1,...,T iterations

w<- s0^2/( cur.s^2/n+ s0^2) # Calculate w

m <- w*bary + (1-w)*mu0  # Calculate m
s <- sqrt( w * cur.s^2/n) # Calculate s

cur.mu <- rnorm(1, m, s) # Calculate mu

a <- a0 + 0.5*n # Calculate alpha
b <- b0 + 0.5 * sum( (y-cur.mu)^2) # Calculate beta

cur.tau <- rgamma( 1, a, b) # Calculate tau from gamma
cur.s <- sqrt(1/cur.tau) # Calculate sigma (inversed tau)

theta[t,]<-c( cur.mu, cur.s) # Set mu(t)=mu, sigma(t)=sigma
} # End of for loop
```

Gibbs results [univariate]

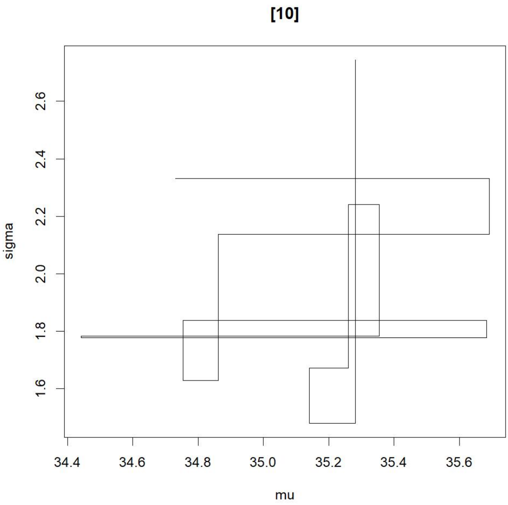


- ESS: $20000/18757.47 = 1.07$ iterations for an independent sample of μ
- Geweke diagnostic: p-value $= 0.49$

Gibbs results [both simultaneously]



Gibbs results [both step by step]



JAGS/WinBUGS code - Examples

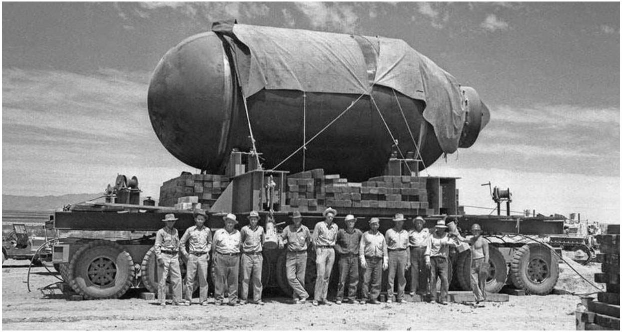
Metropolis example with $y = 50$ and $n = 10^4$

```
"model{
y ~ dbinom(main.ap, n)
main.ap  ~ dbeta(0.1,1)
}"
```

Gibbs example with y temperatures

```
"model{
for (i in 1:12){
y[i] ~ dnorm (mu, prec)
}
mu ~ dnorm (0, 0.01)
prec ~ dgamma (0.001, 0.001)
s2 <- 1/prec
}
}"
```

Epilogue



Bibliography

- Maria L Rizzo, Statistical computing with R, 2008
- Ioannis Ntzoufras, Bayesian modelling using WinBUGS, 2009
- Martin Haugh, Monte-Carlo Simulation, MCMC and Bayesian Modelling
- Joseph Rocca, Bayesian inference problem, MCMC and variational inference
- Pre-course material - HARMONY Training School - Larissa Greece 2020