

Optimization and Simulation

Markov Chain Monte Carlo Methods

Michel Bierlaire

Transport and Mobility Laboratory
School of Architecture, Civil and Environmental Engineering
Ecole Polytechnique Fédérale de Lausanne



EPFL

Outline

- 1 Motivation
- 2 Metropolis-Hastings
- 3 Gibbs sampling
- 4 Simulated annealing
- 5 Appendix: Introduction to Markov chains
- 6 Appendix: Stationary distributions

The knapsack problem

- Patricia prepares a hike in the mountain.
- She has a knapsack with capacity W kg.
- She considers carrying a list of n items.
- Each item has a utility u_i and a weight w_i .
- What items should she take to maximize the total utility, while fitting in the knapsack?



Knapsack problem



Simulation

- Let \mathcal{X} be the set of all possible configurations (2^n).
- Define a probability distribution:

$$P(x) = \frac{e^{U(x)}}{\sum_{y \in \mathcal{X}} e^{U(y)}}$$

- Question: how to draw from this discrete random variable?

Bayesian inference

Choice model

- Consider a commuter n .
- Possible modes: $\mathcal{C}_n = \{car, bus, bike\}$.
- Utility

$$U_{in} = V_{in}(time, cost, weather, \dots; \beta) + \varepsilon_{in}$$

- Choice model:

$$P_n(i) = \Pr(U_{in} \geq U_{jn}, j \in \mathcal{C}_n).$$

- If ε_{in} is EV distributed, we have the logit model:

$$P_n(i; x, \beta) = \frac{e^{V_{in}(x; \beta)}}{\sum_{j \in \mathcal{C}_n} e^{V_{jn}(x; \beta)}}.$$



Bayesian inference



Inference

- Data: $Y = (i_n, x_n)_{n=1}^N$.
- Inference: estimate the true value of β .
- Likelihood:

$$L(Y|\beta) = \prod_{n=1}^N P_n(i_n; x_n, \beta).$$

- Frequentist inference: maximum likelihood estimation.
- Bayesian inference.

Bayesian inference

Bayesian concepts

- Likelihood:

$$L(Y|\beta) = \prod_{n=1}^N P_n(i_n; x_n, \beta).$$

- Prior: $f(\beta)$.
- Posterior:

$$f(\beta|Y) = \frac{L(Y|\beta)f(\beta)}{L(Y)} = \frac{L(Y|\beta)f(\beta)}{\int L(Y|\beta)f(\beta)d\beta}.$$

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

Bayesian inference

Prior: $N(\mu, \Sigma)$

$$f(\beta) = (2\pi)^{-\frac{K}{2}} \det(\Sigma)^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\beta - \mu)^T \Sigma^{-1}(\beta - \mu)\right)$$

Posterior for logit

$$f(\beta|Y) = \frac{f(\beta) \prod_{n=1}^N \frac{e^{V_{in}(x_n; \beta)}}{\sum_{j \in C_n} e^{V_{jn}(x_n; \beta)}}}{\int_{\gamma} f(\gamma) \prod_{n=1}^N \frac{e^{V_{in}(x_n; \gamma)}}{\sum_{j \in C_n} e^{V_{jn}(x_n; \gamma)}}}$$

Prediction



Frequentist way

$$\bar{\beta} = \int \beta f(\beta|Y) d\beta.$$

Bayesian way

- Future unobserved data: Y_f

$$f(Y_f|Y) = \int_{\beta} f(Y_f, \beta|Y) d\beta = \int_{\beta} f(Y_f|\beta, Y) f(\beta|Y) d\beta.$$

- Assumption for prediction: Y and Y_f are independent, cond. on β :

$$f(Y_f|Y) = \int_{\beta} f(Y_f|\beta) f(\beta|Y) d\beta.$$

Average of the likelihood on Y_f over the posterior of β .

Bayesian inference

Difficulties

- Complicated integrals.
- Critical to draw from the posterior.
- Must rely on simulation.
- But how do we draw from such complex distributions?

Outline

- 1 Motivation
- 2 Metropolis-Hastings**
- 3 Gibbs sampling
- 4 Simulated annealing
- 5 Appendix: Introduction to Markov chains
- 6 Appendix: Stationary distributions

Markov chains

Stochastic process

X_t , $t = 0, 1, \dots$, collection of r.v. with same support, or *states space* $\{1, \dots, i, \dots, J\}$.

Markov process: (short memory)

$$\Pr(X_t = i | X_0, \dots, X_{t-1}) = \Pr(X_t = i | X_{t-1})$$

Homogeneous Markov process

$$\Pr(X_t = j | X_{t-1} = i) = \Pr(X_{t+k} = j | X_{t-1+k} = i) = P_{ij} \quad \forall t \geq 1, k \geq 0.$$

Markov chains

Stationary distribution

Unique solution of the system:

$$\pi_j = \sum_{i=1}^J P_{ij} \pi_i, \forall j = 1, \dots, J,$$

$$\sum_{j=1}^J \pi_j = 1.$$

Markov chains

We assume...

- homogeneous: constant transition probability,
- irreducible and aperiodic: any state can be reached from any state in one step with non zero probability,
- time reversible: can be traversed forward and backward:

$$\pi_i P_{ij} = \pi_j P_{ji}, i \neq j.$$

Simulation with Markov chains

Procedure

- We want to simulate a r.v. X with pmf

$$\Pr(X = j) = p_j.$$

- We generate a Markov process with stationary probability p_j (how?)
- We simulate the evolution of the process.

$$p_j = \pi_j = \lim_{t \rightarrow \infty} \Pr(X_t = j) \quad j = 1, \dots, J.$$

Example

- A machine can be in 4 states with respect to wear
 - perfect condition,
 - partially damaged,
 - seriously damaged,
 - completely useless.
- The degradation process can be modeled by an irreducible aperiodic homogeneous Markov process, with the following transition matrix:

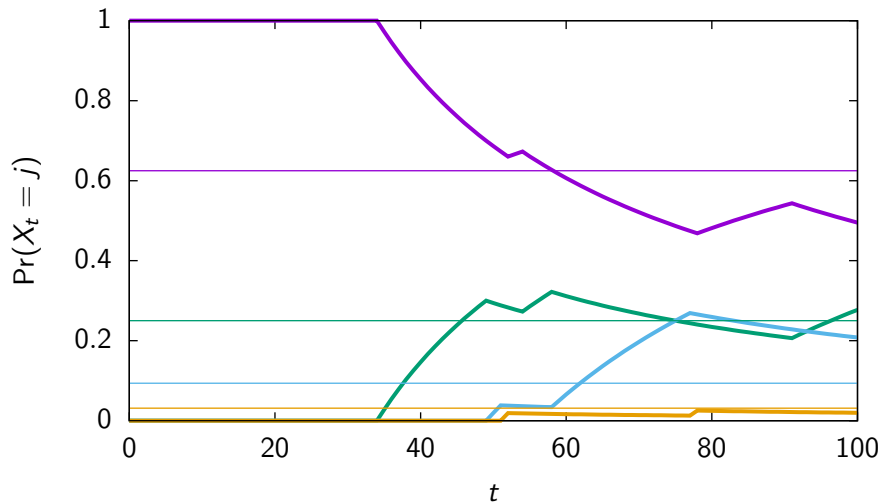
$$P = \begin{pmatrix} 0.95 & 0.04 & 0.01 & 0.0 \\ 0.0 & 0.90 & 0.05 & 0.05 \\ 0.0 & 0.0 & 0.80 & 0.20 \\ 1.0 & 0.0 & 0.0 & 0.0 \end{pmatrix}$$

Example

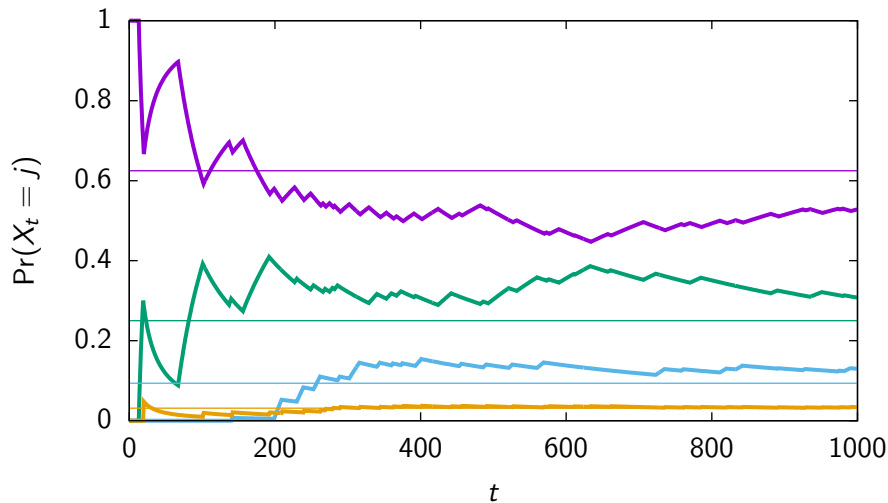
Stationary distribution: $\left(\frac{5}{8}, \frac{1}{4}, \frac{3}{32}, \frac{1}{32}\right)$

$$\left(\frac{5}{8}, \frac{1}{4}, \frac{3}{32}, \frac{1}{32}\right) \begin{pmatrix} 0.95 & 0.04 & 0.01 & 0.0 \\ 0.0 & 0.90 & 0.05 & 0.05 \\ 0.0 & 0.0 & 0.80 & 0.20 \\ 1.0 & 0.0 & 0.0 & 0.0 \end{pmatrix} = \left(\frac{5}{8}, \frac{1}{4}, \frac{3}{32}, \frac{1}{32}\right)$$

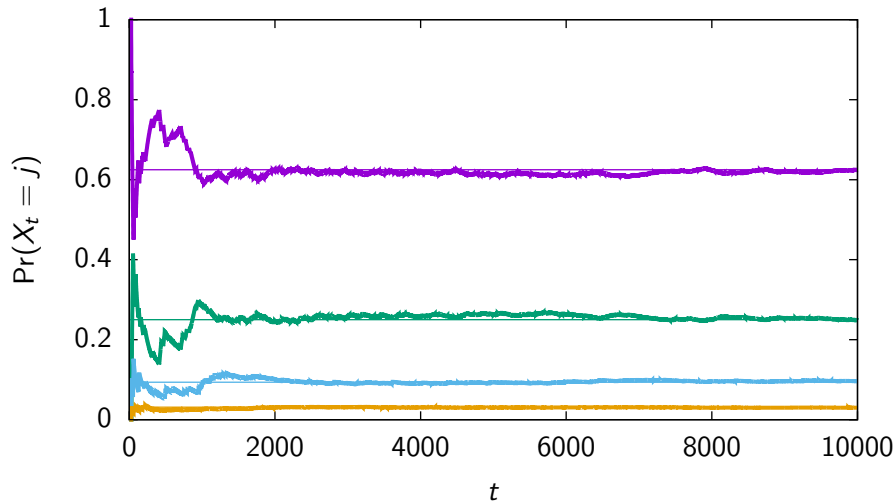
- Machine in perfect condition 5 days out of 8, in average.
- Repair occurs in average every 32 days

Example: $T = 100$ 

Example: $T = 1000$



Example: $T = 10000$



Simulation

Assume that we are interested in simulating

$$E[f(X)] = \sum_{j=1}^J f(j)p_j.$$

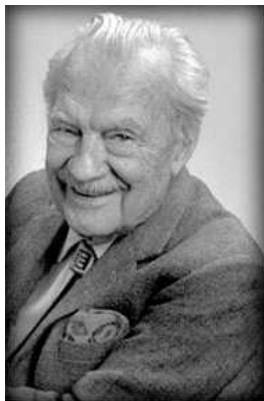
Property of Markov chain: ergodicity

$$E[f(X)] = \lim_{T \rightarrow \infty} \frac{1}{T} \sum_{t=1}^T f(X_t).$$

Drop early states (see above example)

$$E[f(X)] \approx \frac{1}{T} \sum_{t=1+k}^{T+k} f(X_t).$$

Metropolis-Hastings



Nicholas Metropolis
1915 – 1999



Wilfred Keith Hastings
1930 – 2016

Metropolis-Hastings

Context

- Let $b_j, j = 1, \dots, J$ be positive numbers.
- Let $B = \sum_j b_j$. If J is huge, B cannot be computed.
- Let $\pi_j = b_j/B$.
- We want to simulate a r.v. with pmf π_j .

Explore the set

- Consider a Markov process on $\{1, \dots, J\}$ with transition probability Q .
- Designed to explore the space $\{1, \dots, J\}$ efficiently
- Not too fast (and miss important points to sample)
- Not too slowly (and take forever to reach important points)

Metropolis-Hastings

Define another Markov process

- Based on the exact same states $\{1, \dots, J\}$ as the previous ones
- Assume the process is in state i , that is $X_t = i$.
- Simulate the (candidate) next state j according to Q .
- Define

$$X_{t+1} = \begin{cases} j & \text{with probability } \alpha_{ij} \\ i & \text{with probability } 1 - \alpha_{ij} \end{cases}$$

Metropolis-Hastings

Transition probability P

$$\begin{aligned} P_{ij} &= Q_{ij}\alpha_{ij} && \text{if } i \neq j \\ P_{ii} &= Q_{ii}\alpha_{ii} + \sum_{\ell \neq i} Q_{i\ell}(1 - \alpha_{i\ell}) && \text{otherwise} \end{aligned}$$

Must verify the property

$$\begin{aligned} 1 = \sum_j P_{ij} &= P_{ii} + \sum_{j \neq i} P_{ij} \\ &= Q_{ii}\alpha_{ii} + \sum_{\ell \neq i} Q_{i\ell}(1 - \alpha_{i\ell}) + \sum_{j \neq i} Q_{ij}\alpha_{ij} \\ &= Q_{ii}\alpha_{ii} + \sum_{\ell \neq i} Q_{i\ell} - \sum_{\ell \neq i} Q_{i\ell}\alpha_{i\ell} + \sum_{j \neq i} Q_{ij}\alpha_{ij} \\ &= Q_{ii}\alpha_{ii} + \sum_{\ell \neq i} Q_{i\ell} \end{aligned}$$

As $\sum_j Q_{ij} = 1$, we have $\alpha_{ii} = 1$.

Metropolis-Hastings

Time reversibility

$$\pi_i P_{ij} = \pi_j P_{ji}, \quad i \neq j$$

that is

$$\pi_i Q_{ij} \alpha_{ij} = \pi_j Q_{ji} \alpha_{ji}, \quad i \neq j$$

It is satisfied if

$$\alpha_{ij} = \frac{\pi_j Q_{ji}}{\pi_i Q_{ij}} \text{ and } \alpha_{ji} = 1$$

or

$$\frac{\pi_i Q_{ij}}{\pi_j Q_{ji}} = \alpha_{ji} \text{ and } \alpha_{ij} = 1$$

Metropolis-Hastings

As α_{ij} is a probability

$$\alpha_{ij} = \min \left(\frac{\pi_j Q_{ji}}{\pi_i Q_{ij}}, 1 \right)$$

Simplification

Remember: $\pi_j = b_j/B$. Therefore

$$\alpha_{ij} = \min \left(\frac{b_j B Q_{ji}}{b_i B Q_{ij}}, 1 \right) = \min \left(\frac{b_j Q_{ji}}{b_i Q_{ij}}, 1 \right)$$

The normalization constant B does not play a role in the computation of α_{ij} .

Metropolis-Hastings

In summary

- Given Q and b_j
- defining α as above
- creates a Markov process characterized by P
- with stationary distribution π .

Metropolis-Hastings

Algorithm

- 1 Choose a Markov process characterized by Q .
- 2 Initialize the chain with a state i : $t = 0$, $X_0 = i$.
- 3 Simulate the (candidate) next state j based on Q .
- 4 Let r be a draw from $U[0, 1[$.
- 5 Compare r with $\alpha_{ij} = \min\left(\frac{b_j Q_{ji}}{b_i Q_{ij}}, 1\right)$. If

$$r < \alpha_{ij}$$

then $X_{t+1} = j$, else $X_{t+1} = i$.

- 6 Increase t by one.
- 7 Go to step 3.

Metropolis-Hastings

Implementation note

Preferable to work in the log-space

$$\ln r < \ln \alpha_{ij}$$

where

$$\ln \alpha_{ij} = \min(\ln b_j + \ln Q_{ji} - \ln b_i - \ln Q_{ij}, 0).$$

It is equivalent to the condition

$$\ln r < \ln b_j + \ln Q_{ji} - \ln b_i - \ln Q_{ij}.$$

Simple example

$$b = (20, 8, 3, 1)$$

$$\pi = \left(\frac{5}{8}, \frac{1}{4}, \frac{3}{32}, \frac{1}{32} \right)$$

$$Q = \begin{pmatrix} \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \end{pmatrix}$$

Run MH for 10000 iterations. Collect statistics after 1000.

- Accept: [2488, 1532, 801, 283]
- Reject: [0, 952, 1705, 2239]
- Simulated: [0.627, 0.250, 0.095, 0.028]
- Target: [0.625, 0.250, 0.09375, 0.03125]

Bayesian inference

Prior: $N(\mu, \Sigma)$

$$f(\beta) = (2\pi)^{-\frac{K}{2}} \det(\Sigma)^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\beta - \mu)^T \Sigma^{-1}(\beta - \mu)\right)$$

We need to draw from

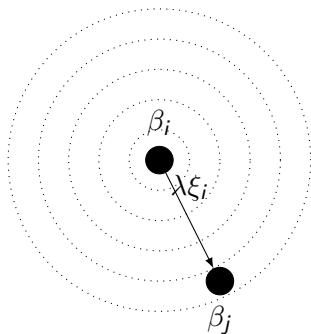
$$\begin{aligned} f(\beta|Y) &= \frac{f(\beta) \prod_{n=1}^N \frac{e^{V_{inn}(x_n; \beta)}}{\sum_{j \in \mathcal{C}_n} e^{V_{jn}(x_n; \beta)}}}{\int_{\gamma} f(\gamma) \prod_{n=1}^N \frac{e^{V_{inn}(x_n; \gamma)}}{\sum_{j \in \mathcal{C}_n} e^{V_{jn}(x_n; \gamma)}}} \\ &\propto f(\beta) \prod_{n=1}^N \frac{e^{V_{inn}(x_n; \beta)}}{\sum_{j \in \mathcal{C}_n} e^{V_{jn}(x_n; \beta)}} \\ &\propto f(\beta) L(Y|\beta). \end{aligned}$$

Markov chain Q : continuous case

Random walk

- Current state: $\beta_i \in \mathbb{R}^K$.
- Draw $\xi_i \in \mathbb{R}^K$ from $N(0, I)$.
- Next state: $\beta_j = \beta_i + \lambda \xi_i$.

$$\begin{aligned}
 Q_{ij} &= Q_{ji} = \phi(\xi_i) \\
 &= \phi\left(\frac{\beta_j - \beta_i}{\lambda}\right).
 \end{aligned}$$



Markov chain Q : continuous case

Reject criterion of MH

$$\begin{aligned}\alpha_{ij} &= \min \left(\frac{b_j Q_{jj}}{b_i Q_{ij}}, 1 \right) \\ &= \min \left(\frac{b_j}{b_i}, 1 \right) \\ &= \min \left(\frac{f(\beta_j)L(Y|\beta_j)}{f(\beta_i)L(Y|\beta_i)}, 1 \right)\end{aligned}$$

- Ratio of posteriors.
- In the log-space, difference of log of posteriors.

Case study

Swissmetro

- a revolutionary mag-lev underground system in Switzerland,
- 500 km/h.



swissmetro.ch

Transportation mode choice

- 1 Train
- 2 Swissmetro
- 3 Car

The model

Variables

- Travel time: TRAIN_TT, SM_TT, CAR_TT
- Travel cost: TRAIN_CO, SM_CO, CAR_CO
- Yearly subscription: GA

Utility functions

- $ASC_TRAIN + B_TIME * TRAIN_TT + B_COST * TRAIN_CO * (GA = 0)$
- $B_TIME * SM_TT + B_COST * SM_CO * (GA = 0)$
- $ASC_CAR + B_TIME * CAR_TT + B_COST * CAR_CO$

Four unknown parameters

Data

Stated preferences

- Collected in March 1998.
- 750 respondents.
- 6768 choice data.

Python code

```
def logPosteriorDensity(beta, loglike):  
    prior = np.array([0, 0, 0, 0])  
    variance = 100  
    lognorm = lognormpdf(beta - prior)  
    return loglike + lognorm / variance
```

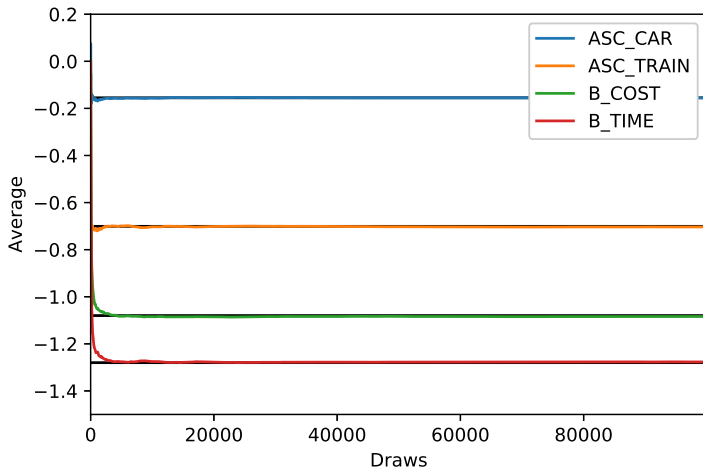
Python code

```
beta = np.array([0, 0, 0, 0])
loglike = biogeme.calculateLikelihood(beta)
logPosterior = logPosteriorDensity(beta, loglike)

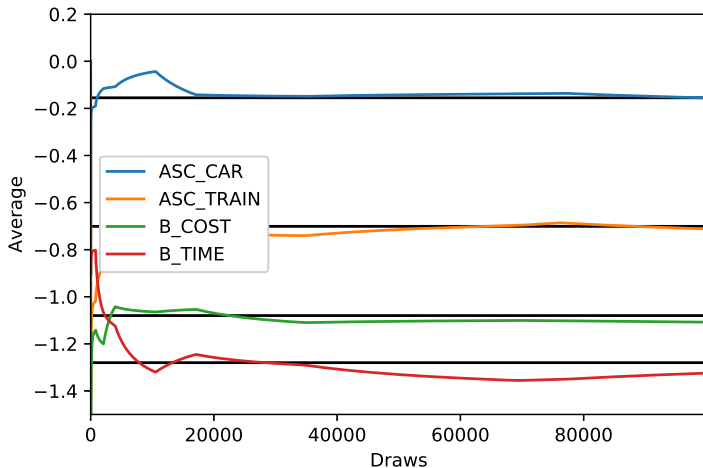
T = 100000
draws = []
for total in range(T):
    ksi = np.random.normal(size=len(beta))
    next = beta + stepRandomWalk * ksi
    nextLoglike = biogeme.calculateLikelihood(next)
    logPosteriorNext = logPosteriorDensity(next, nextLoglike)

    diff = logPosteriorNext - loglike
    r = np.random.uniform()
    if np.log(r) <= diff:
        beta = next
        loglike = lognext
    draws += [beta]
```

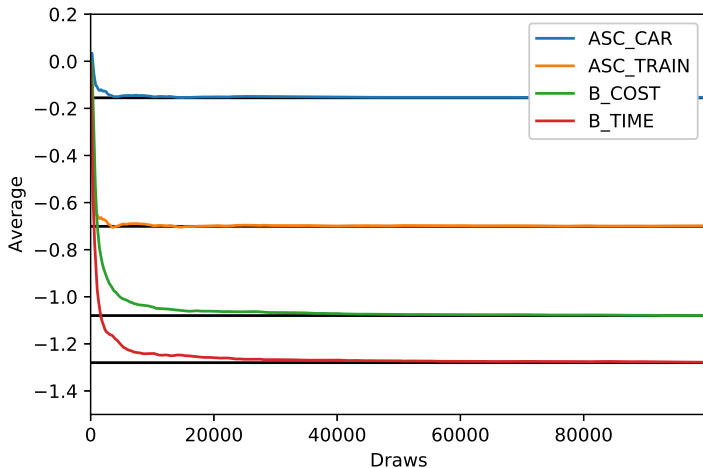
Step: $\lambda = 0.1$ — Accept rate: 7%



Step: $\lambda = 1$ — Accept rate: 0.02%

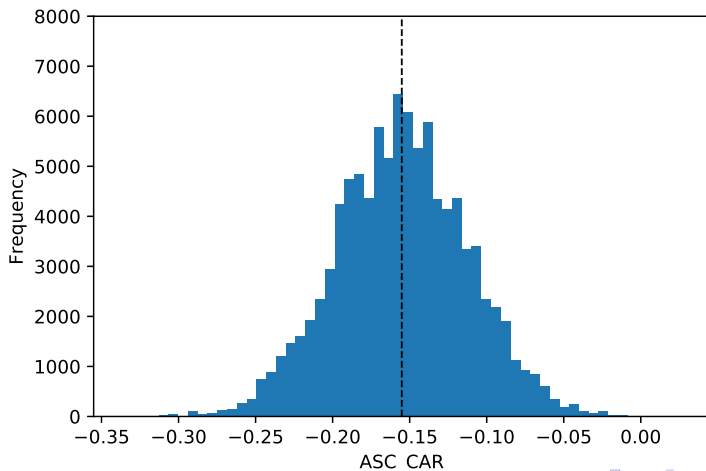


Step: $\lambda = 0.01$ — Accept rate: 78.2%



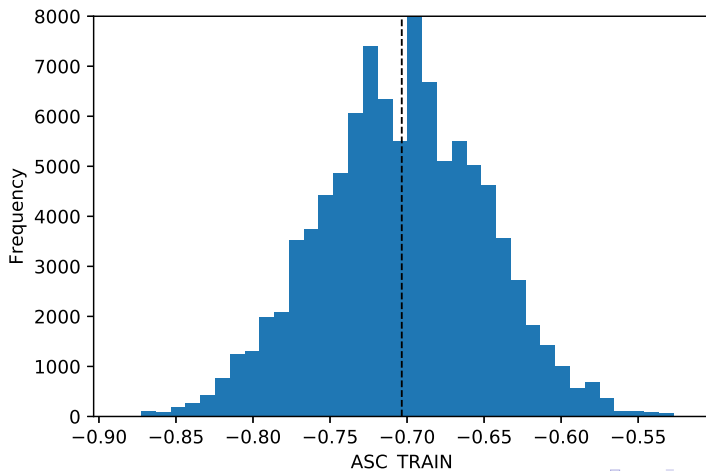
Distribution of the parameter: ASC_CAR

$\lambda = 0.1$, 2000 dropped



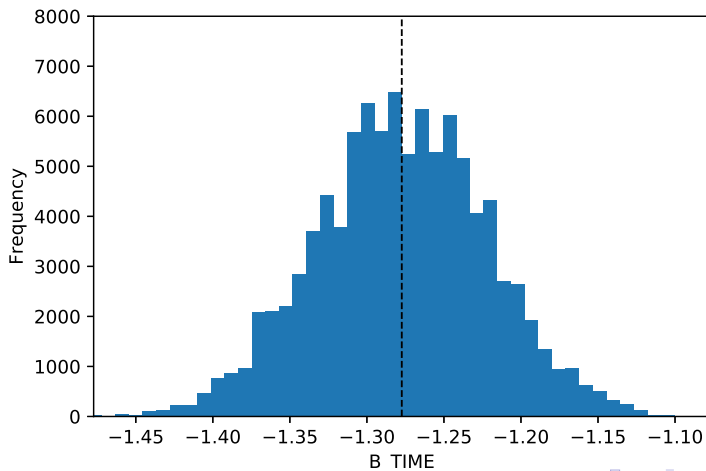
Distribution of the parameter: ASC_TRAIN

$\lambda = 0.1$, 2000 dropped



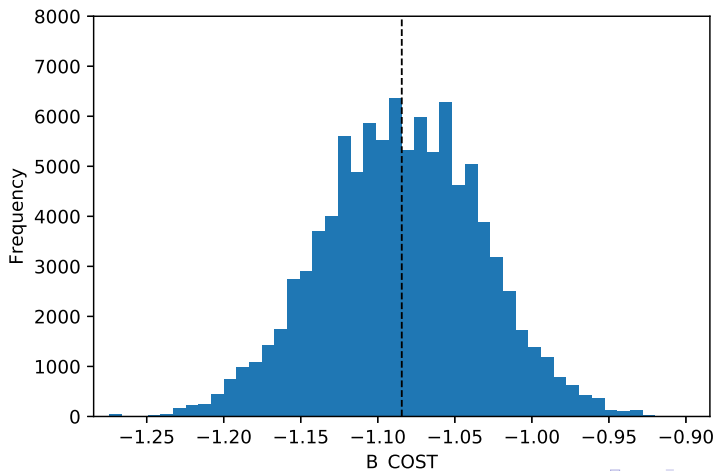
Distribution of the parameter: B_TIME

$\lambda = 0.1$, 2000 dropped



Distribution of the parameter: B_COST

$\lambda = 0.1$, 2000 dropped



Markov chain: gradient based

Idea

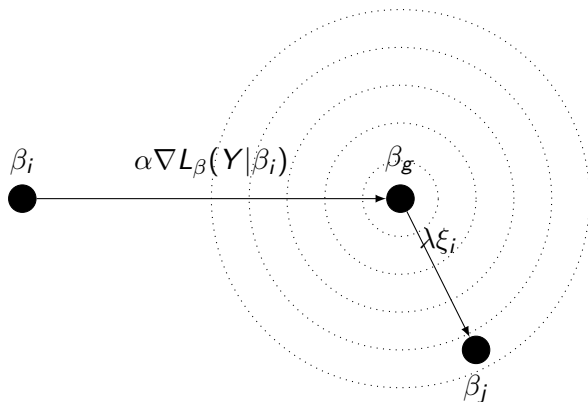
- The gradient $\nabla L_{\beta}(Y|\beta)$ of the likelihood is an ascent direction.
- Instead of performing a random walk around β_i , we perform a random walk around

$$\beta_g = \beta_i + \alpha \nabla L_{\beta}(Y|\beta_i).$$

- Motivation: we want to bias the search towards higher values of the likelihood.

Markov chain: gradient based

Idea

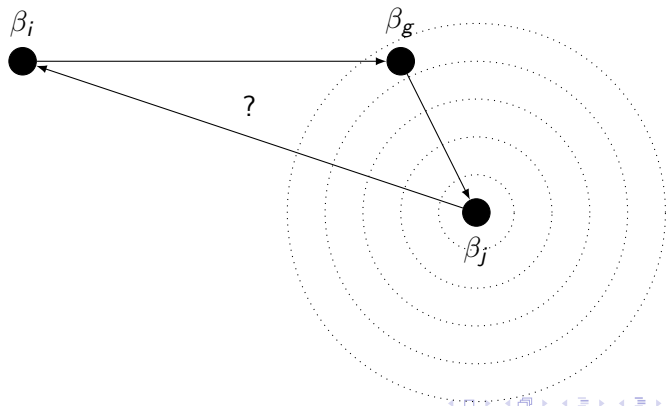


Reject criterion of MH

- Forward transition probability:

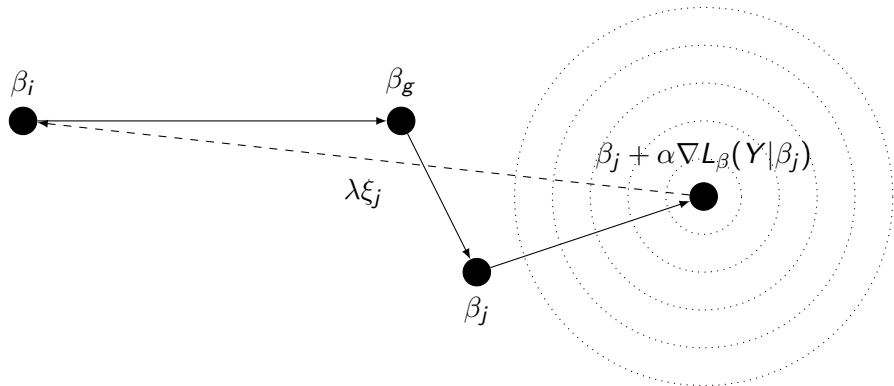
$$Q_{ij} = \phi(\xi_i).$$

- Backward transition probability:

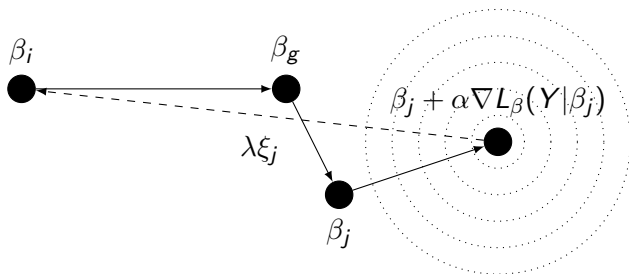


Reject criterion of MH

Backward transition probability



Reject criterion of MH



$$\beta_i = \beta_j + \alpha \nabla L_\beta(Y|\beta_j) + \lambda \xi_j$$

$$Q_{ji} = \phi(\xi_j) = \phi\left(\frac{\beta_i - \beta_j - \alpha \nabla L_\beta(Y|\beta_j)}{\lambda}\right)$$

Python code

```

beta = firstBeta
loglike, g, _, _ = biogeme.calculateLikelihoodAndDerivatives(beta)
betaGrad = beta + stepGradient * g
logPosterior = logPosteriorDensity(beta, loglike)

T = 5000
draws = []
for total in range(T):
    ksi = np.random.normal(size=len(beta))
    next = betaGrad + stepRandomWalk * ksi
    nextLoglike, nextg, _, _ = biogeme.calculateLikelihoodAndDerivatives(next)
    nextGrad = next + stepGradient * nextg
    logPosteriorNext = logPosteriorDensity(next, nextLoglike)

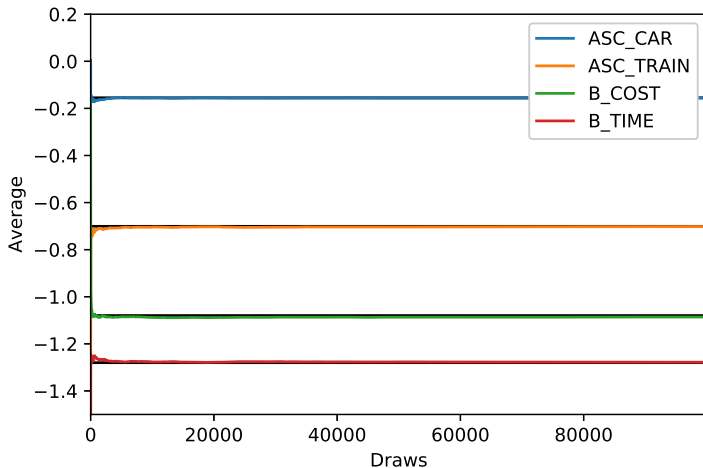
    logQij = lognormpdf(ksi)

    ksiback = (beta - nextGrad) / stepRandomWalk
    logQji = lognormpdf(ksiback)

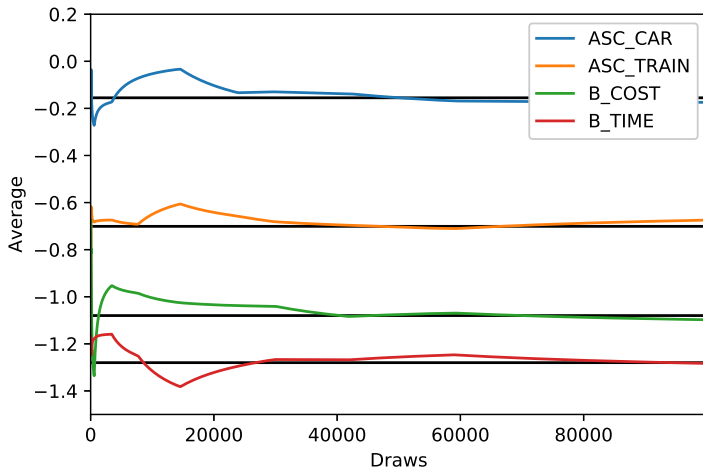
    diff = logPosteriorNext + logQji - logPosterior - logQij
    r = np.random.uniform()
    if np.log(r) <= diff:
        beta = next
        loglike = nextLoglike
        g = nextg
        betaGrad = nextGrad
        logPosterior = logPosteriorNext
    draws += [beta]

```

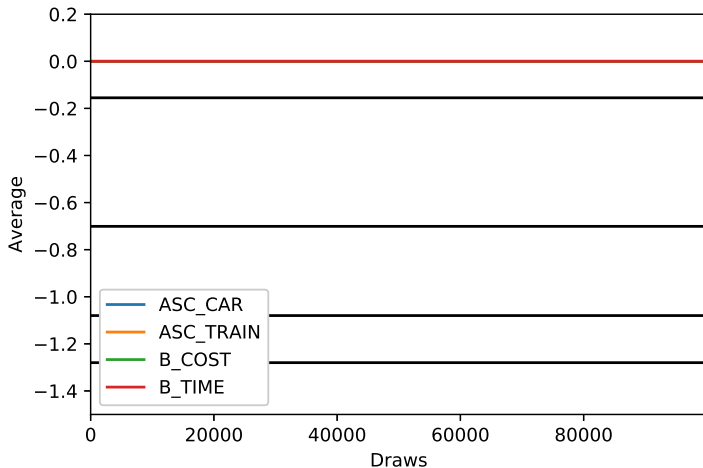
Step: $\lambda = 0.1$ — Accept rate: 8.6%



Step: $\lambda = 1$ — Accept rate: 0.01%

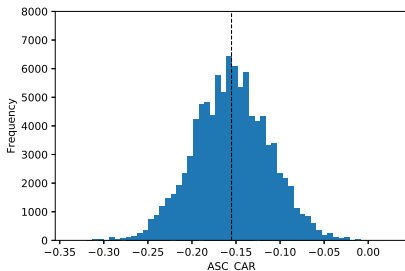


Step: $\lambda = 0.01$ — Accept rate: 0%

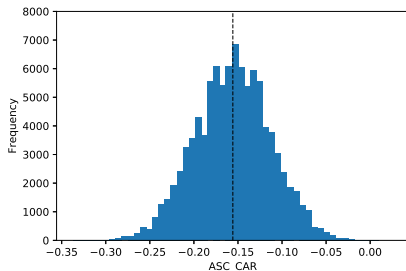


Distribution of the parameter: ASC_CAR

$\lambda = 0.1$, 2000 dropped



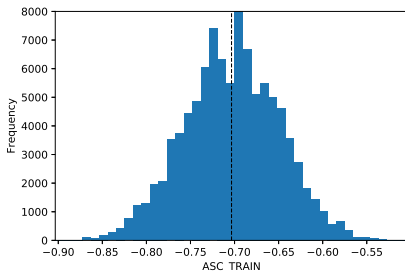
Random walk



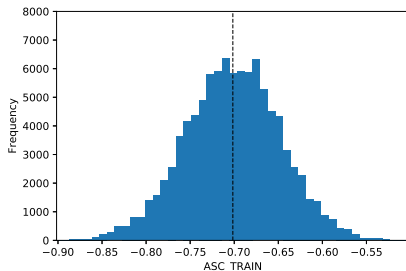
Gradient based

Distribution of the parameter: ASC_TRAIN

$\lambda = 0.1$, 2000 dropped



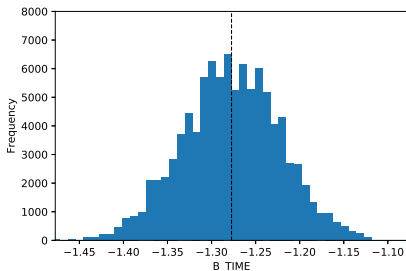
Random walk



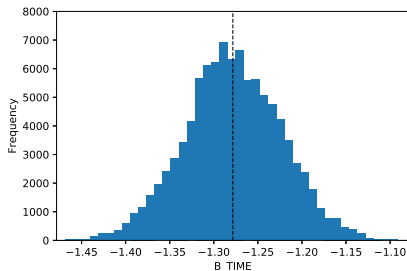
Gradient based

Distribution of the parameter: B_TIME

$\lambda = 0.1$, 2000 dropped



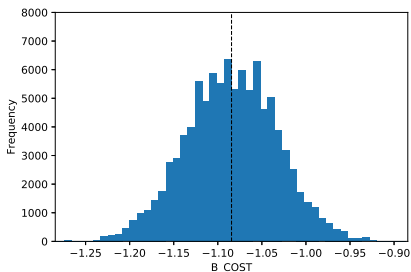
Random walk



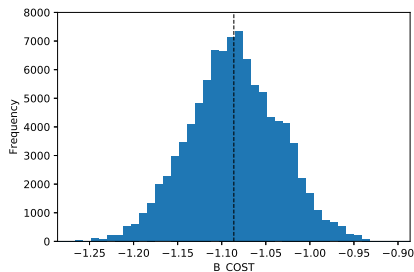
Gradient based

Distribution of the parameter: B_COST

$\lambda = 0.1$, 2000 dropped



Random walk



Gradient based

Mixed strategy

Combine different moves

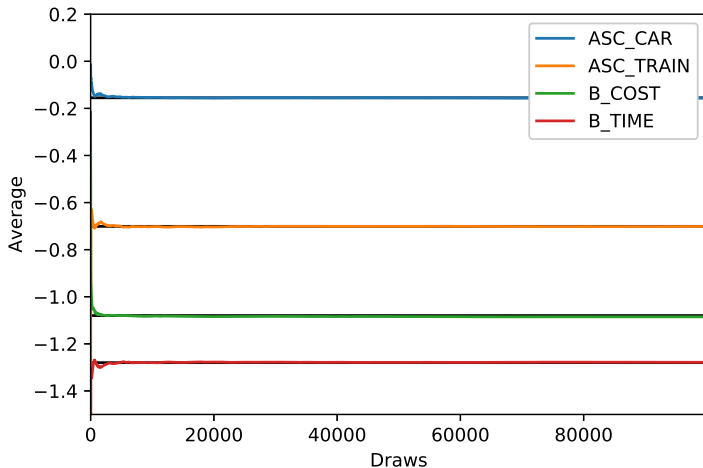
- With probability p , perform a random walk.
- With probability $1 - p$, perform a gradient step.

Transition probability

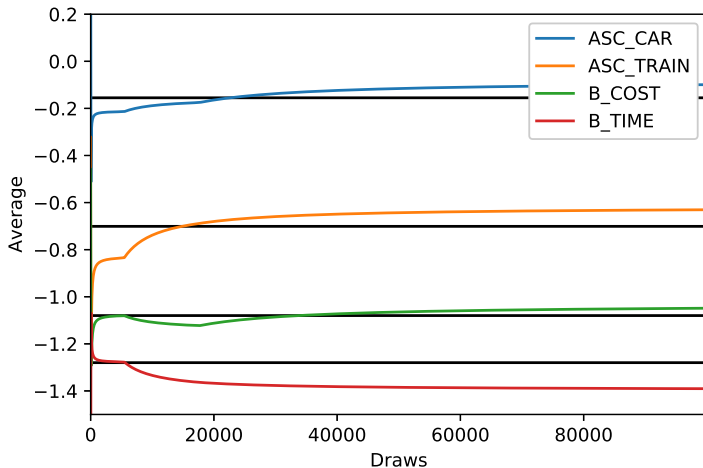
$$Q_{ij} = p\phi(\xi_i) + (1 - p)\phi(\xi_i) = \phi(\xi_i)$$

$$Q_{ji} = p\phi(\xi_j) + (1 - p)\phi\left(\frac{\beta_i - \beta_j - \alpha\nabla L_\beta(Y|\beta_j)}{\lambda}\right)$$

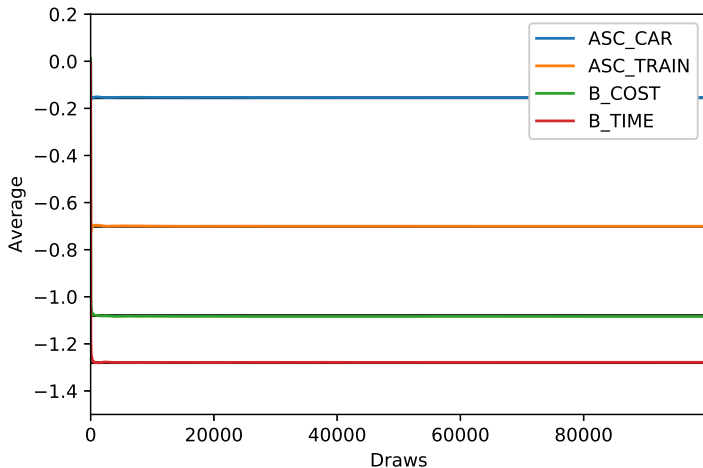
Step: $\lambda = 0.1$ — Accept rate: 8.3%



Step: $\lambda = 1$ — Accept rate: 0.009%

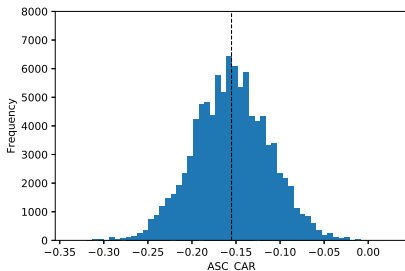


Step: $\lambda = 0.01$ — Accept rate: 63.24%

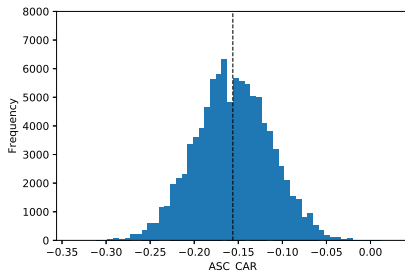


Distribution of the parameter: ASC_CAR

$\lambda = 0.1$, 2000 dropped



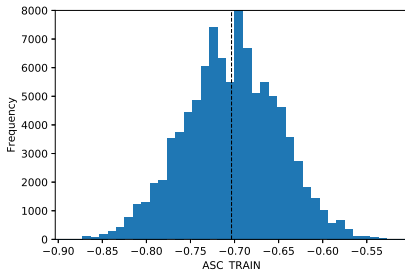
Random walk



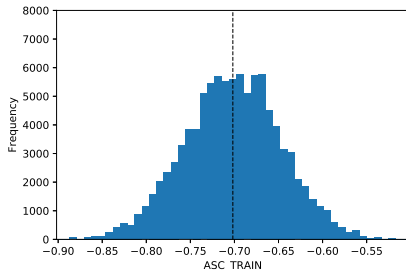
Mixed

Distribution of the parameter: ASC_TRAIN

$\lambda = 0.1$, 2000 dropped



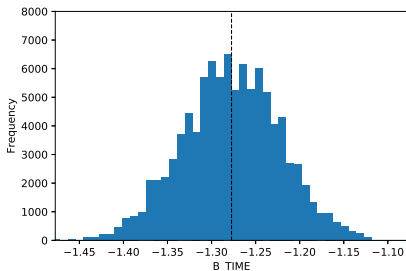
Random walk



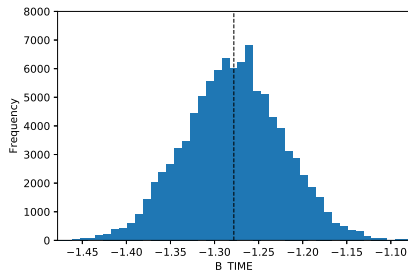
Mixed

Distribution of the parameter: B_TIME

$\lambda = 0.1$, 2000 dropped



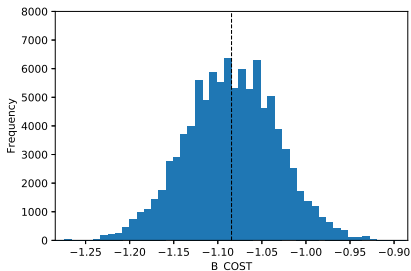
Random walk



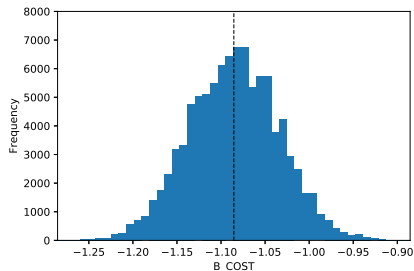
Mixed

Distribution of the parameter: B_COST

$\lambda = 0.1$, 2000 dropped



Random walk



Mixed

Practical considerations

Multiple starting points

- Generate multiple Markov chains.
- Initialize each sequence with a different value.

Stationarity

- Chains must have reached stationarity.
- How do we detect it?

Correlation

- Within sequences.
- Across sequences.
- It may generate inefficiencies in the simulation.

Sequences management

Generate S sequences of length N'

$$s = 1$$

$$s = 2$$

$$s = 3$$

$$s = 4$$

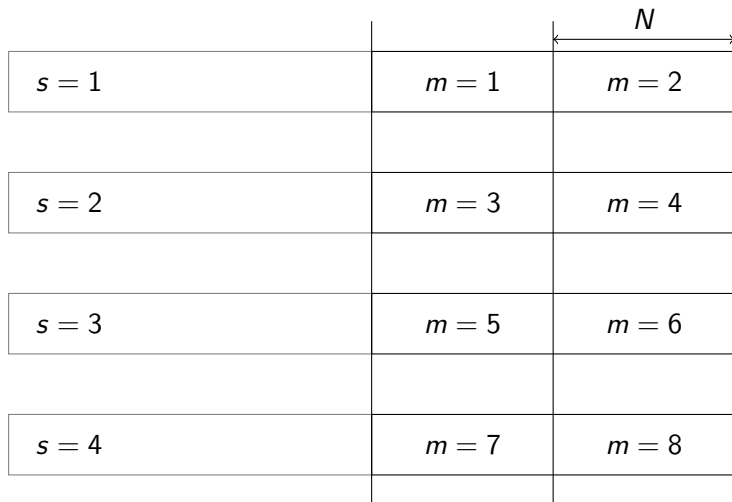
Sequences management

Warm-up: drop half of each sequence

$s = 1$	
$s = 2$	
$s = 3$	
$s = 4$	

Sequences management

Split each sequence into two to obtain M sequences of length $N = N'/4$



Between-sequence variance

Let θ be the parameter of interest, and θ_{nm} draw n from sequence m .

$$B = \frac{N}{M-1} \sum_{m=1}^M (\bar{\theta}_m - \bar{\theta})^2,$$

where

$$\bar{\theta}_m = \frac{1}{N} \sum_{n=1}^N \theta_{nm} \quad \text{mean of each sequence}$$

$$\bar{\theta} = \frac{1}{M} \sum_{m=1}^M \bar{\theta}_m \quad \text{mean of the mean}$$

Within-sequence variance

$$W = \frac{1}{M} \sum_{m=1}^M v_m^2$$

where

$$v_m^2 = \frac{1}{N-1} \sum_{n=1}^N (\theta_{nm} - \bar{\theta}_m)^2.$$

How long should the sequences be?

Potential scale reduction

$$\hat{R}_N = \sqrt{\frac{N-1}{N} + \frac{1}{N} \frac{B}{W}}$$
$$\lim_{N \rightarrow \infty} R_N = 1.$$

Choose N such that $R_n \leq 1.1$.

See Gelman et al. (2013) Section 11.4.

Outline

- 1 Motivation
- 2 Metropolis-Hastings
- 3 Gibbs sampling**
- 4 Simulated annealing
- 5 Appendix: Introduction to Markov chains
- 6 Appendix: Stationary distributions

Gibbs sampling

Motivation

- Draw from multivariate distributions.
- Main difficulty: deal with correlations.

Metropolis-Hastings

- Let $X = (X^1, X^2, \dots, X^n)$ be a random vector with pmf (or pdf) $p(x)$.
- Assume we can draw from the marginals:

$$\Pr(X^i | X^j = x^j, j \neq i), \quad i = 1, \dots, n.$$

- Markov process. Assume current state is x .
 - Draw randomly (equal probability) a coordinate i .
 - Draw r from the i th marginal.
 - New state: $y = (x^1, \dots, x^{i-1}, r, x^{i+1}, \dots, x^n)$.

Gibbs sampling

Transition probability

$$Q_{xy} = \frac{1}{n} \Pr(X^i = r | X^j = x^j, j \neq i) = \frac{p(y)}{n \Pr(X^j = x^j, j \neq i)}$$

- The denominator is independent of X^i .
- So Q_{xy} is proportional to $p(y)$.

Metropolis-Hastings

$$\alpha_{xy} = \min \left(\frac{p(y)Q_{yx}}{p(x)Q_{xy}}, 1 \right) = \min \left(\frac{p(y)p(x)}{p(x)p(y)}, 1 \right) = 1$$

The candidate state is **always** accepted.

Example: bivariate normal distribution

$$\begin{pmatrix} X \\ Y \end{pmatrix} \sim N \left(\begin{pmatrix} \mu_X \\ \mu_Y \end{pmatrix}, \begin{pmatrix} \sigma_X^2 & \rho\sigma_X\sigma_Y \\ \rho\sigma_X\sigma_Y & \sigma_Y^2 \end{pmatrix} \right)$$

Marginal distribution:

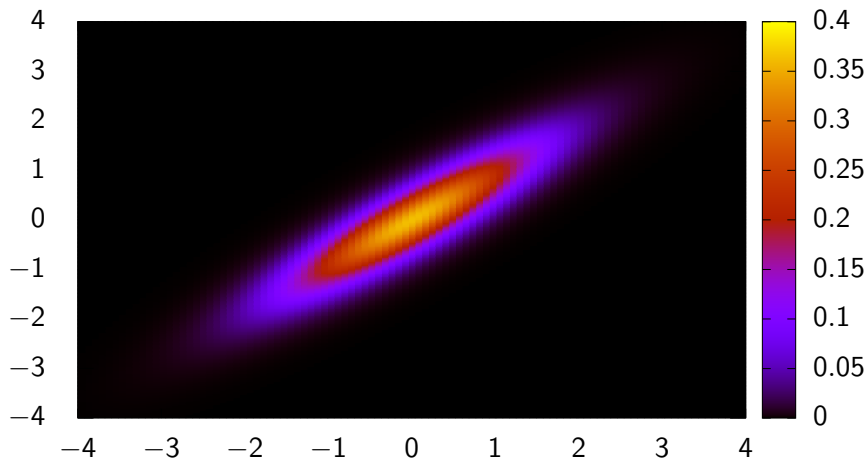
$$Y|(X=x) \sim N \left(\mu_Y + \frac{\sigma_Y}{\sigma_X} \rho(x - \mu_X), (1 - \rho^2)\sigma_Y^2 \right)$$

Apply Gibbs sampling to draw from:

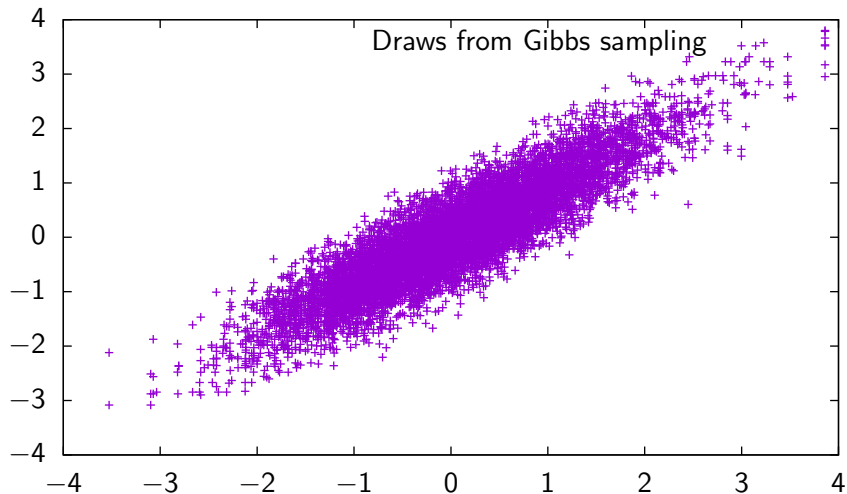
$$N \left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & 0.9 \\ 0.9 & 1 \end{pmatrix} \right)$$

Note: just for illustration. Should use Cholesky factor.

Example: pdf



Example: draws from Gibbs sampling



Outline

- 1 Motivation
- 2 Metropolis-Hastings
- 3 Gibbs sampling
- 4 Simulated annealing**
- 5 Appendix: Introduction to Markov chains
- 6 Appendix: Stationary distributions

Simulated annealing

Combinatorial optimization

$$\min_{x \in \mathcal{F}} f(x)$$

where the feasible set \mathcal{F} is a large finite set of vectors.

Set of optimal solutions

$$\mathcal{X}^* = \{x \in \mathcal{F} \mid f(x) \leq f(y), \forall y \in \mathcal{F}\} \text{ and } f(x^*) = f^*, \forall x^* \in \mathcal{X}^*.$$

Probability mass function on \mathcal{F}

$$p_\lambda(x) = \frac{e^{-\lambda f(x)}}{\sum_{y \in \mathcal{F}} e^{-\lambda f(y)}}, \lambda > 0.$$

Simulated annealing

$$p_\lambda(x) = \frac{e^{-\lambda f(x)}}{\sum_{y \in \mathcal{F}} e^{-\lambda f(y)}}$$

- Equivalently

$$p_\lambda(x) = \frac{e^{\lambda(f^* - f(x))}}{\sum_{y \in \mathcal{F}} e^{\lambda(f^* - f(y))}}$$

- As $f^* - f(x) \leq 0$, when $\lambda \rightarrow \infty$, we have

$$\lim_{\lambda \rightarrow \infty} p_\lambda(x) = \frac{\delta(x \in \mathcal{X}^*)}{|\mathcal{X}^*|},$$

where

$$\delta(x \in \mathcal{X}^*) = \begin{cases} 1 & \text{if } x \in \mathcal{X}^* \\ 0 & \text{otherwise.} \end{cases}$$

Example

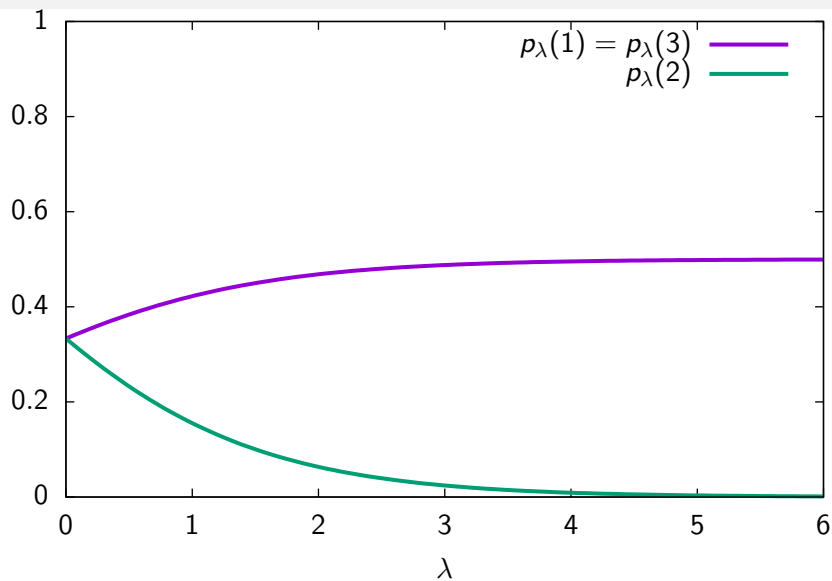
$$\mathcal{F} = \{1, 2, 3\} \quad f(\mathcal{F}) = \{0, 1, 0\}$$

$$p_\lambda(1) = \frac{1}{2 + e^{-\lambda}}$$

$$p_\lambda(2) = \frac{e^{-\lambda}}{2 + e^{-\lambda}}$$

$$p_\lambda(3) = \frac{1}{2 + e^{-\lambda}}$$

Example



Simulated annealing

- If λ is large,
- we generate a Markov chain with stationary distribution $p_\lambda(x)$.
- The mass is concentrated on optimal solutions.
- As the normalizing constant is not needed, only $e^{\lambda(f^* - f(x))}$ is used.
- Construction of the Markov process through the concept of *neighborhood*.
- A *neighbor* y of x is obtained by simple modifications of x .
- The Markov process will proceed from neighbors to neighbors.
- The neighborhood structure must be designed such that the chain is irreducible, that is the whole space \mathcal{F} must be covered.
- It must be designed also such that the size of the neighborhood is reasonably small.

Neighborhood

Metropolis-Hastings

- Denote $N(x)$ the set of neighbors of x .
- Define a Markov process where the next state is a randomly drawn neighbor.
- Transition probability:

$$Q_{xy} = \frac{1}{|N(x)|}$$

- Metropolis Hastings:

$$\alpha_{xy} = \min \left(\frac{p(y)Q_{yx}}{p(x)Q_{xy}}, 1 \right) = \min \left(\frac{e^{-\lambda f(y)}|N(x)|}{e^{-\lambda f(x)}|N(y)|}, 1 \right)$$

Neighborhood

Notes

- The neighborhood structure can always be arranged so that each vector has the same number of neighbors. In this case,

$$\alpha_{xy} = \min \left(\frac{e^{-\lambda f(y)}}{e^{-\lambda f(x)}}, 1 \right)$$

- If y is better than x , the next state is automatically accepted.
- Otherwise, it is accepted with a probability that depends on λ .
- If λ is high, the probability is small.
- When λ is small, it is easy to escape from local optima.

Heuristic

Issue

- The number of iterations needed to reach a stationary state and draw an optimal solution may exceed the number of feasible solutions in the set.
- The acceptance probability is very small.
- Therefore, a complete enumeration works better.
- The method is used as a heuristic.

Outline

- 1 Motivation
- 2 Metropolis-Hastings
- 3 Gibbs sampling
- 4 Simulated annealing
- 5 Appendix: Introduction to Markov chains**
- 6 Appendix: Stationary distributions

Markov Chains



Andrey Markov, 1856–1922, Russian mathematician.

Markov Chains: glossary

Stochastic process

X_t , $t = 0, 1, \dots$, collection of r.v. with same support, or *states space* $\{1, \dots, i, \dots, J\}$.

Markov process: (short memory)

$$\Pr(X_t = i | X_0, \dots, X_{t-1}) = \Pr(X_t = i | X_{t-1})$$

Homogeneous Markov process

$$\Pr(X_t = j | X_{t-1} = i) = \Pr(X_{t+k} = j | X_{t-1+k} = i) = P_{ij} \quad \forall t \geq 1, k \geq 0.$$

Markov Chains

Transition matrix

$$P \in \mathbb{R}^{J \times J}.$$

Properties:

$$\sum_{j=1}^J P_{ij} = 1, \quad i = 1, \dots, J, \quad P_{ij} \geq 0, \quad \forall i, j,$$

Ergodicity

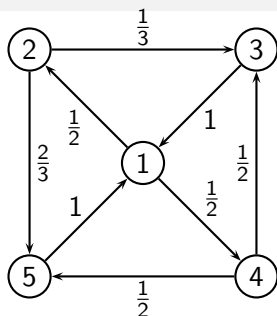
- If state j can be reached from state i with non zero probability, and i from j , we say that i *communicates* with j .
- Two states that communicate belong to the same *class*.
- A Markov chain is *irreducible* or *ergodic* if it contains only one class.
- With an ergodic chain, it is possible to go to every state from any state.

Markov Chains

Aperiodic

- P_{ij}^t is the probability that the process reaches state j from i after t steps.
- Consider all t such that $P_{ii}^t > 0$. The largest common divisor d is called the *period* of state i .
- A state with period 1 is *aperiodic*.
- If $P_{ii} > 0$, state i is aperiodic.
- The period is the same for all states in the same class.
- Therefore, if the chain is irreducible, if one state is aperiodic, they all are.

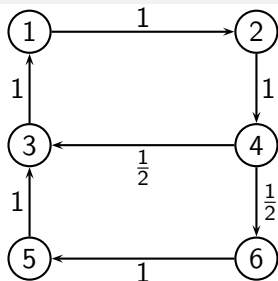
A periodic chain



$$P = \begin{pmatrix} 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{3} & 0 & \frac{2}{3} \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix}, \quad d = 3.$$

$$P_{ii}^t > 0 \text{ for } t = 3, 6, 9, 12, 15 \dots$$

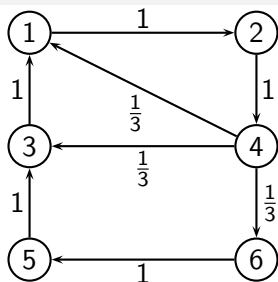
Another periodic chain



$$P = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 0 & 0 & \frac{1}{2} \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}, \quad d = 2.$$

$$P_{ii}^t > 0 \text{ for } t = 4, 6, 8, 10, 12, \dots$$

An aperiodic chain



$$P = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{3} & 0 & \frac{1}{3} & 0 & 0 & \frac{1}{3} \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}, \quad d = 1.$$

$$P_{ij}^t > 0 \text{ for } t = 3, 4, 6, 7, 8, 9, 10, 11, 12 \dots$$

Aperiodic chain

An equivalent definition

An irreducible Markov chain is said to be aperiodic if for some $t \geq 0$ and some state i , we have

$$\Pr(X_t = i | X_0 = i) > 0$$

and

$$\Pr(X_{t+1} = i | X_0 = i) > 0$$

Intuition

Oscillation

$$P = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

The chain will not “converge” to something stable.

Outline

- 1 Motivation
- 2 Metropolis-Hastings
- 3 Gibbs sampling
- 4 Simulated annealing
- 5 Appendix: Introduction to Markov chains
- 6 Appendix: Stationary distributions**

Markov Chains

Stationary probabilities

$$\Pr(j) = \sum_{i=1}^J \Pr(j|i) \Pr(i)$$

- Stationary probabilities: unique solution of the system

$$\pi_j = \sum_{i=1}^J P_{ij} \pi_i, \quad \forall j = 1, \dots, J. \quad (1)$$

$$\sum_{j=1}^J \pi_j = 1.$$

- Solution exists for any irreducible chain.

Example

- A machine can be in 4 states with respect to wear
 - perfect condition,
 - partially damaged,
 - seriously damaged,
 - completely useless.
- The degradation process can be modeled by an irreducible aperiodic homogeneous Markov process, with the following transition matrix:

$$P = \begin{pmatrix} 0.95 & 0.04 & 0.01 & 0.0 \\ 0.0 & 0.90 & 0.05 & 0.05 \\ 0.0 & 0.0 & 0.80 & 0.20 \\ 1.0 & 0.0 & 0.0 & 0.0 \end{pmatrix}$$

Example

Stationary distribution: $\left(\frac{5}{8}, \frac{1}{4}, \frac{3}{32}, \frac{1}{32}\right)$

$$\left(\frac{5}{8}, \frac{1}{4}, \frac{3}{32}, \frac{1}{32}\right) \begin{pmatrix} 0.95 & 0.04 & 0.01 & 0.0 \\ 0.0 & 0.90 & 0.05 & 0.05 \\ 0.0 & 0.0 & 0.80 & 0.20 \\ 1.0 & 0.0 & 0.0 & 0.0 \end{pmatrix} = \left(\frac{5}{8}, \frac{1}{4}, \frac{3}{32}, \frac{1}{32}\right)$$

- Machine in perfect condition 5 days out of 8, in average.
- Repair occurs in average every 32 days

From now on: Markov process = irreducible aperiodic homogeneous Markov process

Markov Chains

Detailed balance equations

Consider the following system of equations:

$$x_i P_{ij} = x_j P_{ji}, \quad i \neq j, \quad \sum_{i=1}^J x_i = 1 \quad (2)$$

We sum over i :

$$\sum_{i=1}^J x_i P_{ij} = x_j \sum_{i=1}^J P_{ji} = x_j.$$

If (2) has a solution, it is also a solution of (1). As π is the unique solution of (1) then $x = \pi$.

$$\pi_i P_{ij} = \pi_j P_{ji}, \quad i \neq j$$

The chain is said *time reversible*

Stationary distributions

Property of irreducible aperiodic Markov chains

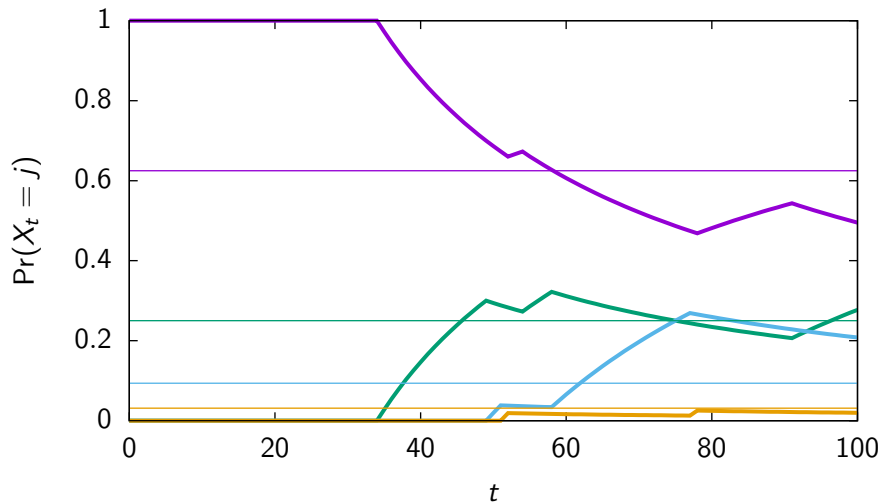
$$\pi_j = \lim_{t \rightarrow \infty} \Pr(X_t = j) \quad j = 1, \dots, J.$$

Ergodicity

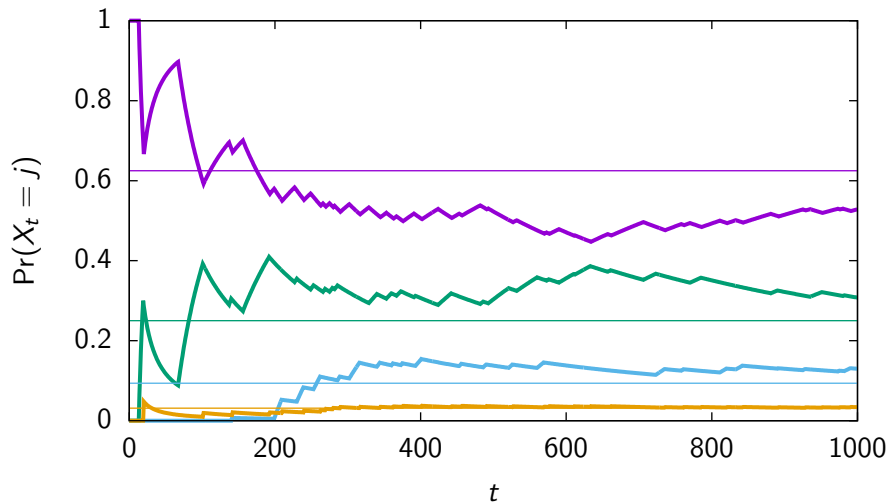
- Let f be any function on the state space.
- Then, with probability 1,

$$\lim_{T \rightarrow \infty} \frac{1}{T} \sum_{t=1}^T f(X_t) = \sum_{j=1}^J \pi_j f(j).$$

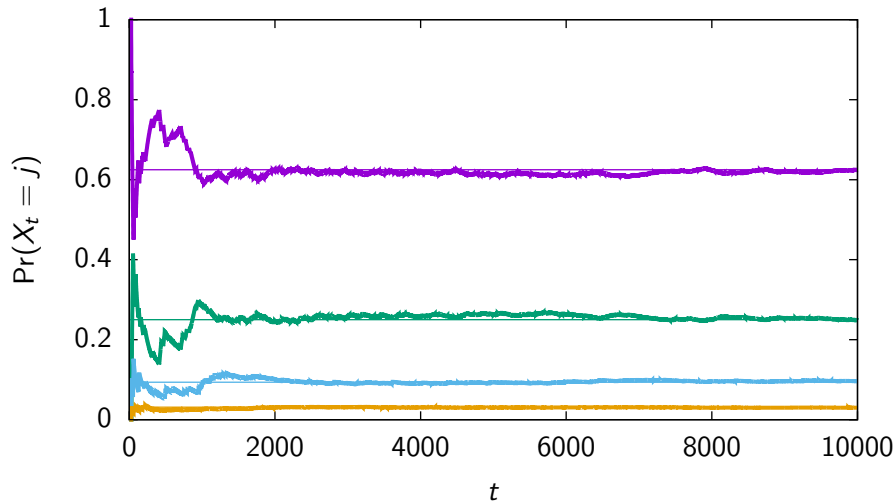
- Computing the expectation of a function of the stationary states is the same as to take the average of the values along a trajectory of the process.

Example: $T = 100$ 

Example: $T = 1000$



Example: $T = 10000$



A periodic example

It does not work for periodic chains

$$P = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

$$\Pr(X_t = 1) = \begin{cases} 1 & \text{if } t \text{ is odd} \\ 0 & \text{if } t \text{ is even} \end{cases}$$

$$\lim_{t \rightarrow \infty} \Pr(X_t = 1) \text{ does not exist}$$

Stationary distribution

$$\pi = \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix}$$