Machine Learning

Lecture 10 – MCMC and sampling methods

UPPSALA
UNIVERSITET

Thomas Schön
Division of Systems and Control
Department of Information Technology
Uppsala University.

Email: thomas.schon@it.uu.se,
www: `user.it.uu.se/~thosc112`

Contents – lecture 10

1. About the exam
2. Summary of lecture 9
3. Motivation for Monte Carlo methods
4. Basic sampling methods
   - Transformation methods
   - Rejection sampling
   - Importance sampling
5. Markov Chain Monte Carlo (MCMC)
   - General properties
   - Metropolis Hastings sampler
   - Gibbs sampler

(Chapter 11)

About the exam (I/II)

- If you have followed the course and completed the exercises you will not be surprised when you see the exam.
- You will learn new things during the exam.

Practicalities:

- Time frame: 3 days (72h), somewhere in the time frame April 4, 2014 - May 5, 2014 (May 5, 2014 is the last day to start the exam).
- You collect the exam from X.
- Within 72 hours after you have collected the exam, you put your solutions in an envelope (seal it) and hand it in to X.

About the exam (II/II)

As usual the **graduate exam honor code** applies. This means,

- The course books, other books and MATLAB are all allowed aids.
- Internet services such as email, web browsers and other communication with the surrounding world concerning the exam is NOT allowed.
- You are NOT allowed to actively search for the solutions in books, papers, the Internet or anywhere else.
- You are NOT allowed to talk to others (save for the responsible teacher) about the exam at all.
- You are NOT allowed to look at exams from earlier version of the course.
- If anything is unclear concerning what is allowed and not, just ask me.

Summary of lecture 9 (I/III)

Introduced the **Hammersley-Clifford theorem** to find the PDF for an undirected graph (Markov random field). The joint pdf,

$$p(x_{1:\mathcal{N}}) = \frac{1}{Z}\prod_{c\in C}\psi_c(x_c),$$

C is the set of all maximal cliques and Z is the **partition function**,

$$Z = \sum_{x_{1:N}}\prod_{c\in C}\psi_c(x_c).$$

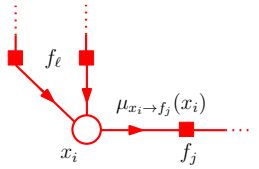
Inference in graphical models amounts to computing the posterior distribution of one or more of the nodes that are not observed.

Summary of lecture 9 (II/III)

The inference algorithm is expressed in terms of a **message passing** algorithm, where local messages are propagated around the graph. Two interconnected types of messages are considered:

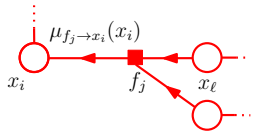
- Messages from variable nodes to factor nodes

$$\mu_{x_i\to f_j}(x_i) = \prod_{f_\ell\in \mathrm{ne}(x_i)\setminus f_j}\mu_{f_\ell\to x_i}(x_i)$$



- Messages from factor nodes to variable nodes

$$\mu_{f_j\to x_i}(x_i) = \sum_{x_\ell\in \mathrm{ne}(f_j)\setminus x_i} f_j \prod_{x_\ell\in \mathrm{ne}(f_j)\setminus x_i}\mu_{x_\ell\to f_j}(x_\ell)$$



Summary of lecture 9 (III/III)

Sum-Product Algorithm

- Calculate messages from variable nodes to factor nodes

$$\mu_{x_i\to f_j}(x_i) = \prod_{f_\ell\in \mathrm{ne}(x_i)\setminus f_j}\mu_{f_\ell\to x_i}(x_i)$$

- Calculate messages from factor nodes to variable nodes

$$\mu_{f_j\to x_i}(x_i) = \sum_{x_\ell\in \mathrm{ne}(f_j)\setminus x_i} f_j \prod_{x_\ell\in \mathrm{ne}(f_j)\setminus x_i}\mu_{x_\ell\to f_j}(x_\ell)$$

- Iterate messages until convergence. (Different iteration schemes can be designed.)
- After convergence, the marginals are calculated as

$$p(x_i) \propto \prod_{f_\ell\in \mathrm{ne}(x_i)}\mu_{f_\ell\to x_i}(x_i)$$

Motivation – Monte Carlo

In solving inference problems we are sooner or later typically faced with various **integration problems**, which tend to live in high dimensional spaces.

This hols for both Maximum likelihood and Bayesian approaches.

To be concrete, we have (for example) the following general problems

1. Expectation
2. Marginalization (includes normalization)

MC motivation 1 – expectation

An expected value often provides an interesting (and interpretable) point estimate.

Computing an expectation amounts to solving the following integral

$$\mathrm{E}\left[g(z)\right] = \int_{\mathcal{Z}} g(z)p(z)\mathrm{d}z,$$

for some function $g : \mathcal{Z} \to \mathrm{R}^{n_g}$.

Example: Computing a point estimate ($g(x_t) = x_t$).

MC motivation 2 – marginalization

If we are interested in the properties of a stochastic variable z_1 and have access to the PDF $p(z_1, z_2 \mid y_{1:T})$, then we can marginalize out the variable z_2, resulting in $p(z_1 \mid y_{1:T})$.

$$p(z_1 \mid y_{1:T}) = \int_{\mathcal{Z}_2} p(z_1, z_2 \mid y_{1:T})\mathrm{d}z_2$$

Examples: Normalization $p(y_{1:T}) = \int p(y_{1:T} \mid z)p(z)\mathrm{d}z$ (used in e.g., empirical Bayes). As another example (in using the EM algorithm for nonlinear ML identification) we need the two-step smoothing densities $p(x_{t:t+1} \mid y_{1:T})$, whereas several smoothing algorithms provides the entire joint smoothing density $p(x_{1:T} \mid y_{1:T})$.

Approximation methods

Many of the models we are currently interested in do **not** allow for closed form expressions. We are forced to approximations. Broadly speaking there are two classes,

1. **Deterministic analytical approximations:** Either approximate the model or restrict the solution to belong to an analytically tractable form. Examples, Laplace approximation, variational Bayes (VB), expectation propagation (EP).
2. **Stochastic approximations:** Keep the model and approximate the solution without imposing any restrictions other than the computational resources available.

In this lecture we are concerned with stochastic approximations (deterministic approximations are already covered).

Monte Carlo methods

Monte Carlo methods provides **computational solutions**, where the obtained accuracy is limited only by our computational resources.

Monte Carlo methods respects the model and the general solution. The approximation does not impose any restricting assumptions on the model or the solution.

The Monte Carlo idea (I/II)

(Very) restrictive assumption: Assume that we have N samples $\{z^i\}_{i=1}^N$ from the target density $\pi(z)$,

$$\widehat{\pi}(z) = \sum_{i=1}^{N} \frac{1}{N} \delta_{z^i}(z)$$

Allows for the following approximation of the integral,

$$\mathrm{E}\left[g(z)\right] = \int g(z)\pi(z)\mathrm{d}z \approx \int g(z) \sum_{i=1}^{N} \frac{1}{N} \delta_{z^i}(z)\mathrm{d}z = \frac{1}{N} \sum_{i=1}^{N} g(z^i)$$

$$\text{“} \int + \delta \rightarrow \sum \text{”}$$

The Monte Carlo idea (II/II)

The integral

$$I(g(z)) \triangleq \mathrm{E}_{\pi(z)}\left[g(z)\right] = \int g(z)\pi(z)\mathrm{d}z.$$

is approximated by

$$\widehat{I}^M(g(z)) = \frac{1}{M} \sum_{i=1}^{M} g(z^i).$$

The strong law of large numbers tells us that

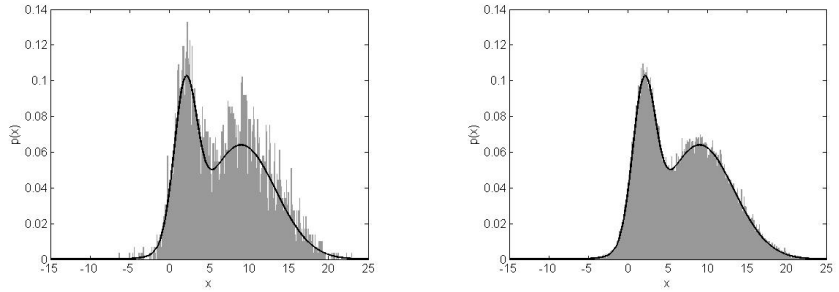
$$\widehat{I}^M(g(z)) \xrightarrow{\text{a.s.}} I(g(z)), \qquad M \rightarrow \infty,$$

and the central limit theorem state that

$$\frac{\sqrt{M}\left(\widehat{I}^M(g(z)) - I(g(z))\right)}{\sigma_g} \xrightarrow{\text{d}} \mathcal{N}(0,1), \qquad M \rightarrow \infty.$$

The Monte Carlo idea – toy illustration

$$\pi(z) = 0.3\mathcal{N}(z \mid 2, 2) + 0.7\mathcal{N}(z \mid 9, 19)$$



5 000 samples

50 000 samples

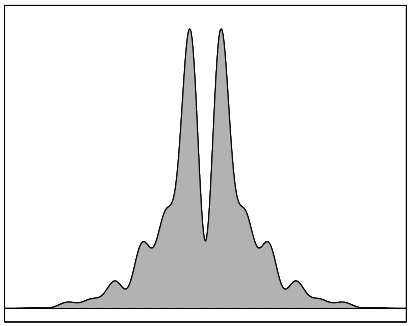
Obvious problem: In general we are **not** able to directly sample from the density we are interested in.

Rejection sampling (I/VII)

Rejection sampling is a Monte Carlo method that produce i.i.d. samples from a target distribution

$$\pi(z) = \frac{\widetilde{\pi}(z)}{C_\pi},$$

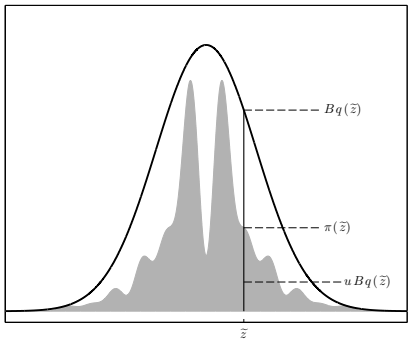
where $\widetilde{\pi}(z)$ can be evaluated and C_π is a normalization constant.



Key idea: Generate random numbers uniformly from the area under the graph of the target distribution $\pi(z)$.

Just as hard as the original problem, but what if...

Rejection sampling (II/VII)



Generate a sample $\widetilde{z}$ from a proposal distribution $q(z)$ and a sample $u \sim \mathcal{U}[0,1]$.

The sample $\widetilde{z}$ is then an i.i.d. sample from the target if

$$u \leq \frac{\widetilde{\pi}(\widetilde{z})}{Bq(\widetilde{z})}$$

Rejection sampling (III/VII)

Assumptions:

1. It is easy to sample from $q(z)$.
2. There exists a constant B such that $\pi(z) \leq Bq(z), \forall z \in \mathcal{Z}$.
3. The support of $q(z)$ includes the support of $\pi(z)$, i.e., $q(z) > 0$ when $\pi(z) > 0$.

Algorithm 1 Rejection sampling (RS)

1. Sample $\widetilde{z} \sim q(z)$.
2. Sample $u \sim \mathcal{U}[0,1]$.
3. If $u \leq \frac{\widetilde{\pi}(\widetilde{z})}{Bq(\widetilde{z})}$ accept $\widetilde{z}$ as a sample from $\pi(z)$ and go to 1.
4. Otherwise, reject $\widetilde{z}$ and go to 1.

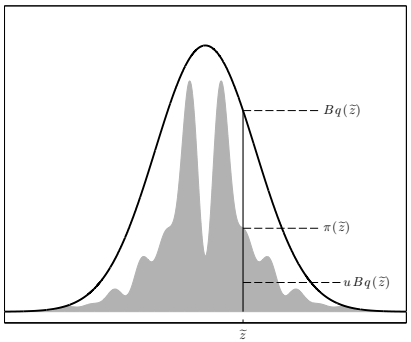
Rejection sampling (IV/VII)

- The procedure can be used with multivariate densities in the same way.
- The rejection rate depends on B, choose B as small as possible, while still satisfying $\pi(z) \leq Bq(z), \forall z \in \mathcal{Z}$.
- Choosing a good proposal distribution $q(z)$ is very important.
- Rejection sampling is used to construct fast **particle smoothers** via backward simulation.

Rejection sampling (V/VII)

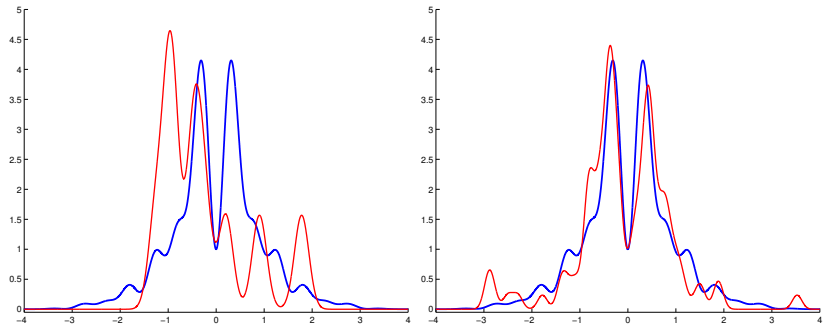
Task: Generate M i.i.d. samples from the following distribution,

$$\pi(z) = \frac{1}{C_\pi} e^{-\frac{1}{2}z^2} \left(\sin(6z)^2 + 3\cos(z)^2 \sin(4z)^2 + 1 \right),$$



Solution: Use rejection sampling where $q(z) = \mathcal{N}(z \mid 0, 1)$ and $B = 12$.

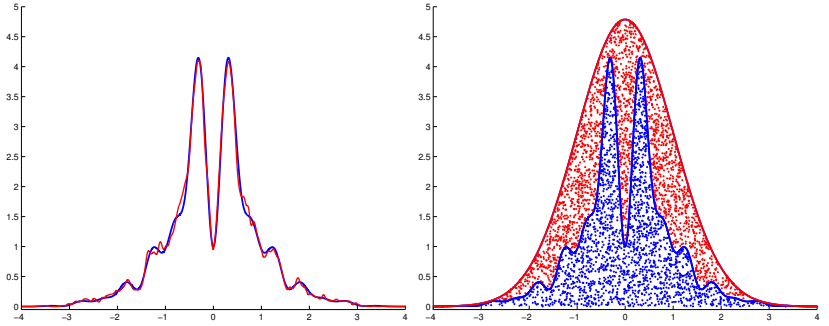
Rejection sampling (VI/VII)



(a) 10 samples

(b) 100 samples

Rejection sampling (VII/VII)



(c) 10 000 samples

(d) Showing rejected (red) and accepted (blue) samples

Importance sampling (IS)

Algorithm 2 Importance sampler (IS)

1. Sample $z^i \sim q(z)$.
2. Compute the weights $\widetilde{w}^i = \widetilde{\pi}(z^i)/q(z^i)$.
3. Normalize the weights $w^i = \widetilde{w}^i / \sum_{j=1}^{N} \widetilde{w}^j$.

Each step is carried out for $i = 1, \ldots M$.

Importance sampling – alternative interpretation

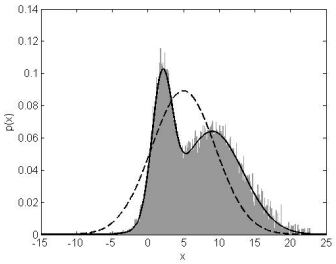
IS does not provide samples from the target density, but the samples $\{z^i\}_{i=1}^M$ together with the normalized weights $\{w^i\}_{i=1}^M$ provides an **empirical approximation** of the target density,

$$\widehat{\pi}(z) = \sum_{i=1}^{M} w^i \delta_{z^i}(z).$$

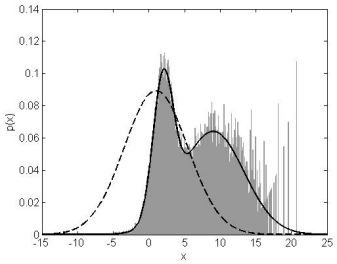
When this approximation is inserted into $I(g(z)) = \int g(z)\pi(z)\mathrm{d}z$ the resulting estimate is

$$\widehat{I}^M(g(z)) = \sum_{i=1}^{M} w^i g(z^i).$$

The importance of a good proposal density



$q_1(x) = \mathcal{N}(5, 20)$ (dashed curve)



$q_2(x) = \mathcal{N}(1, 20)$ (dashed curve)

$50\,000$ samples used in booth simulations.

Lesson learned: It is important to be careful in selecting the importance density.

Importance sampling – example (I/VII)

Let us revisit the same problem (scalar LGSS) used in illustrating the EM and the VB algorithms,

$$x_{t+1} = \theta x_t + v_t,$$
$$y_t = \frac{1}{2} x_t + e_t,$$
$$\begin{pmatrix} v_t \\ e_t \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0.1 & 0 \\ 0 & 0.1 \end{pmatrix} \right).$$

$p(x_1) = \mathcal{N}(x_1 \mid 0, 0.1)$. The true parameter value for θ is given by $\theta^\star = 0.9$. We use $p(\theta) = \mathcal{N}\left(\theta \mid \mu_\theta, \sigma_\theta^2\right)$ as prior distribution for θ.

The identification problem is now to determine the parameter θ on the basis of the observations $y_{1:T}$ and the above model, using the IS algorithm. The result will be an estimate of the posterior distribution $p(\theta \mid y_{1:T})$.

Importance sampling – example (II/VII)

The importance sampler will target

$$\pi(\theta) = p(\theta \mid y_{1:T}) = \frac{p(y_{1:T} \mid \theta) p(\theta)}{p(y_{1:T})} \propto p(y_{1:T} \mid \theta) p(\theta).$$

Chose the proposal distribution to be the same as the prior,

$$q(\theta) = \mathcal{N}\left(\theta \mid \mu_\theta, \sigma_\theta^2\right).$$

The importance weights are then computed according to

$$\widetilde{w}^i = \frac{\widetilde{\pi}(\theta^i)}{\widetilde{q}(\theta^i)} = p(y_{1:T} \mid \theta^i), \qquad i = 1, \ldots, M,$$

i.e., the likelihood.

Importance sampling – example (III/VII)

The Kalman filter straightforwardly allows us to evaluate the importance weights $\widetilde{w}^i = p(y_{1:T} \mid \theta^i)$,

$$p(y_{1:T} \mid \theta^i) = \prod_{t=1}^{T} p(y_t \mid y_{1:t-1}, \theta^i) = \prod_{t=1}^{T} \mathcal{N}\left(y_t \mid \widehat{y}_{t|t-1}(\theta^i), S_{t|t-1}(\theta^i)\right),$$
$$\widehat{y}_{t|t-1}(\theta^i) = 0.5 \widehat{x}_{t|t-1}(\theta^i),$$
$$S_{t|t-1}(\theta^i) = 0.5^2 P_{t|t-1}(\theta^i) + 0.1,$$

where $\widehat{x}_{t|t-1}(\theta^i)$ and $P_{t|t-1}(\theta^i)$ are provided by the Kalman filter.

Importance sampling – example (IV/VII)

Algorithm 3 Importance sampler targeting $p(\theta \mid y_{1:T})$

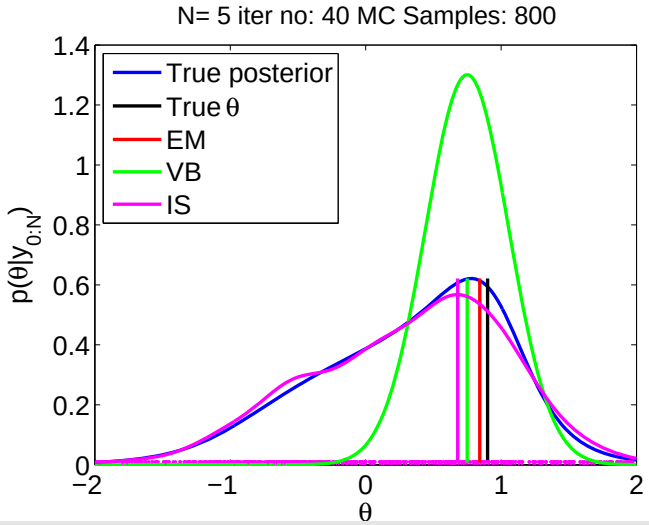
1. Generate M i.i.d. samples from the proposal distribution,

$$\theta^i \sim \mathcal{N}\left(\theta \mid \mu_\theta, \sigma_\theta^2\right).$$

2. Compute importance weights $\widetilde{w}^i = p(y_{1:T} \mid \theta^i)$.
3. Normalize the importance weights $w^i = \frac{\widetilde{w}^i}{\sum_{j=1}^M \widetilde{w}^j}$.

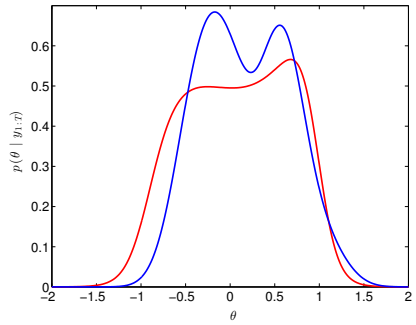
Each step is carried out for $i = 1, \ldots M$.

Importance sampling – example (V/VII)

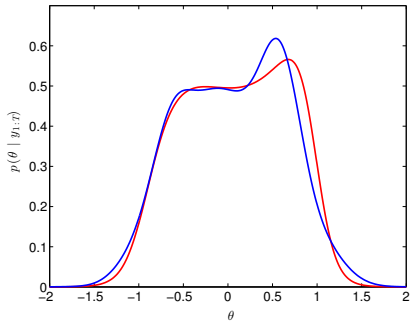


Importance sampling – example (VI/VII)

Using $T = 15$ measurements, $y_{1:15}$.

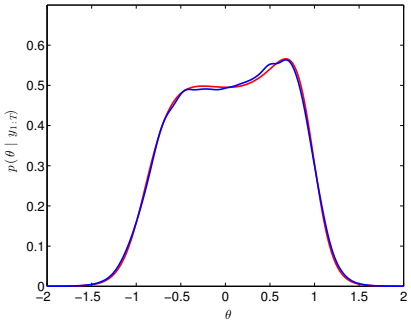


10 samples,
$q(\theta) = \mathcal{N}\left(\theta \mid 0, 1.5^2\right)$

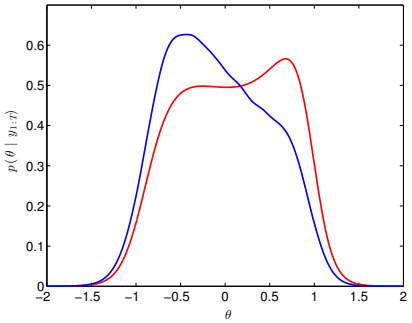


100 samples,
$q(\theta) = \mathcal{N}\left(\theta \mid 0, 1.5^2\right)$

Importance sampling – example (VII/VII)



50 000 samples,
$q(\theta) = \mathcal{N}\left(\theta \mid 0, 1.5^2\right)$



100 000 samples,
$q(\theta) = \mathcal{N}\left(\theta \mid -0.5, 1^2\right)$

Lesson learned (again): Note the different proposal distributions used above. It is very important to be careful in selecting the importance density.

Markov Chain Monte Carlo

What's the point with the AR(1) example? (I/II)

Task: How do we generate samples from the stationary distribution $\pi^{\mathrm{s}}(x) = \mathcal{N}\left(x \mid 0, \frac{q}{1-a^2}\right)$? Put in other words, the target distribution $\pi(x)$ is given by the stationary distribution $\pi^{\mathrm{s}}(x)$, i.e., $\pi(x) = \pi^{\mathrm{s}}(x)$.

Two solutions for this problem:

1. Simulate sufficiently many samples from the Markov chain and discard the initial samples. The remaining samples will then be approximately distributed according to the target distribution (we just proved that x^t is distributed according to $\pi(x)$ for a large enough t).
2. We proved that the stationary distribution is Gaussian. Generate samples directly from this distribution.

Clearly a somewhat contrived example (obviously solution 2 is preferred), **but** solution 1 is a simple illustration of the strategy underlying all MCMC methods.

What's the point with the AR(1) example? (II/II)

In the example, the Markov chain was fully specified and it was possible to explicitly compute the stationary distribution.

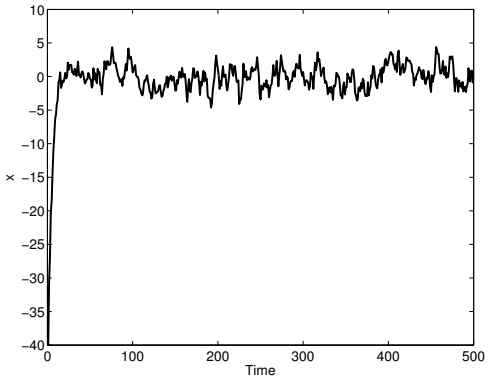
We are of course interested in the **reverse** situation, where we want to generate samples from a (typically rather complicated) target distribution $\pi(z)$.

The task is now to find a transition kernel such that the resulting Markov chain has the target distribution $\pi(z)$ as its stationary distribution.

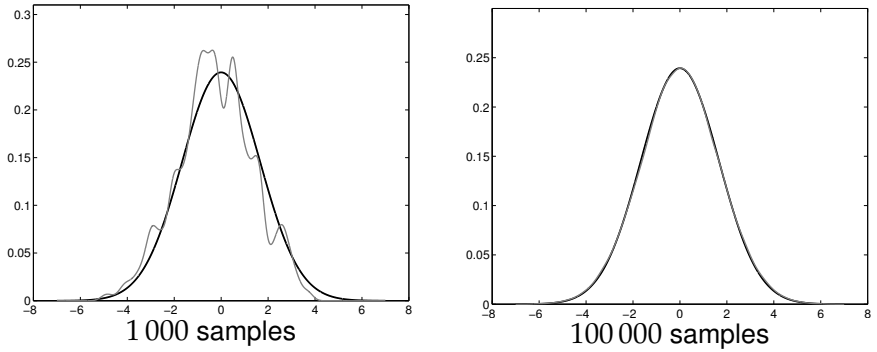
This can be done in many different ways and **constructive strategies** for doing this are provided by the Gibbs sampler and the Metropolis Hastings sampler.

AR(1) example again (I/II)

One realisation from $x^{t+1} = ax^t + v^t$ using $a = 0.8$, $v^t \sim \mathcal{N}(0, 1)$. The process is initialised in $x_0 = -40$.



AR(1) example again (II/II)



The true stationary distribution is showed in black and the empirical histogram obtained by simulating the Markov chain $x^{t+1} = ax^t + v^t$ is plotted in gray.

The initial $1\,000$ samples are discarded (burn-in).

Metropolis Hastings (MH) sampler

The Metropolis Hastings (MH) sampler provides a **constructive way** of producing a Markov chain that can be used to obtain samples approximately distributed according to the target distribution.

More pragmatically speaking, the MH sampler generates samples $\{z^m\}_{m=1}^M$ which can for example be used to approximately compute integrals.

Metropolis Hastings (MH) sampler – intuition

The basic idea underlying the Metropolis Hastings sampler is surprisingly simple.

Starting from an initial state of the Markov chain z^1, a new candidate sample z^* is generated using a **proposal distribution** $z^* \sim q(z \mid z^1)$.

This proposed sample z^* is then accepted with a certain probability, the so called **acceptance probability**

$$a(z^*, z^m) = \min\left(1, \frac{\pi(z^*)q(z^m \mid z^*)}{\pi(z^m)q(z^* \mid z^m)}\right).$$

If the sample is accepted, the new state of the Markov chain is set to the proposed sample $z^2 = z^*$, otherwise it is simply set to the previous value, $z^2 = z^1$.

Metropolis Hastings (MH) sampler – algorithm

Algorithm 4 Metropolis Hastings (MH) sampler

1. **Initialise:** Set the initial state of the Markov chain z^1.
2. **For** $m = 1$ **to** M**, iterate:**
   a. Sample $z^* \sim q(z \mid z^m)$.
   b. Sample $u \sim \mathcal{U}[0,1]$.
   c. Compute the acceptance probability
   $$a(z^*, z^m) = \min\left(1, \alpha(z^*, z^m)\right), \text{where } \alpha(z^*, z^m) = \frac{\pi(z^*)q(z^m \mid z^*)}{\pi(z^m)q(z^* \mid z^m)}$$
   d. Set the next state z^{m+1} of the Markov chain according to
   $$z^{m+1} = \begin{cases} z^* & u \leq a(z^*, z^m) \\ z^m & \text{otherwise} \end{cases}$$

Metropolis Hastings (MH) sampler

Note that the MH sampler only requires two things,

1. It requires the definition of a proposal distribution $q(\cdot \mid \cdot)$ that can be used to generate candidate samples.
2. It must be possible to point-wise evaluate the target distribution up to a possibly unknown normalization factor.

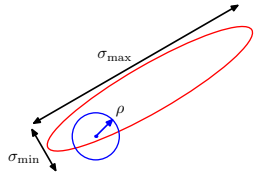
Point-wise evaluation of the target density $\pi(\theta)$ for a specific $\theta = \bar{\theta}$

$$\pi(\bar{\theta}) = p(\bar{\theta} \mid y_{1:T}) = \frac{p(y_{1:T} \mid \bar{\theta}) p(\bar{\theta})}{p(y_{1:T})}.$$

MH example – sampling from a Gaussian (I/II)

- Suppose the target density over $x \in \mathbb{R}^2$ is

$$\pi(x) = \mathcal{N}\left(x; \begin{bmatrix} 4 \\ 4 \end{bmatrix}, \begin{bmatrix} 1 & 0.8 \\ 0.8 & 1 \end{bmatrix}\right)$$



- Choose the proposal density $q(x|z)$ as

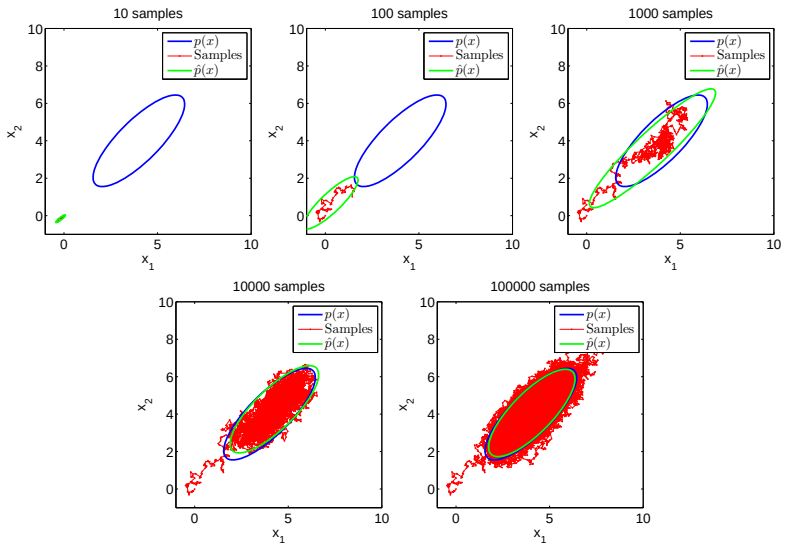
$$q(x|x^{(i-1)}) = \mathcal{N}\left(x; x^{(i-1)}, \begin{bmatrix} 0.01 & 0 \\ 0 & 0.01 \end{bmatrix}\right)$$

- Noticing that $q(x|x^{(i-1)}) = q(x^{(i-1)}|x)$ for all x, we have

$$\min\left(1, \frac{\pi(\bar{x})}{\pi(x^{(i-1)})} \frac{q(x^{(i-1)}|\bar{x})}{q(\bar{x}|x^{(i-1)})}\right) = \min\left(1, \frac{\pi(\bar{x})}{\pi(x^{(i-1)})}\right)$$

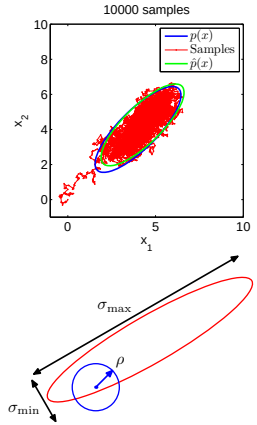
- This version of the Metropolis Hastings algorithm is called the Metropolis algorithm.

MH example – sampling from a Gaussian (II/II)



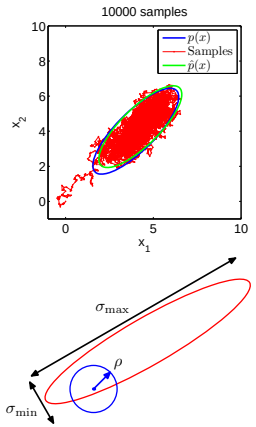
MH – general comments (I/II)

- The MH sampler generates samples that converge to the samples of a stationary distribution which is the target distribution.
- The time that passes before the samples start to represent the target density is called **burn-in period**.
- We generally have to use only the samples obtained after the burn-in period.
- Diagnosing convergence to the target distribution with MCMC algorithms is still an active area of research.
- After the burn-in period is over, the Markov chain is said to be **mixed**.



MH – general comments (II/II)

- Proposal selection is still an important problem.
- If the proposal is selected too narrow, then step-sizes get smaller and the burn-in period becomes longer.
- If the proposal is too wide, then the burn-in gets shorter, however, the acceptance rate is decreased significantly.



Gibbs Sampling

- Gibbs sampling is a special case of the Metropolis-Hastings algorithm where the proposal function is set to be the conditional distribution of the variables.
- It is especially useful when the dimension of the space to sample is very large e.g. images.
- Suppose, we are sampling in a two dimensional space $x = [x_1, x_2]^{\mathsf{T}}$. Then the Gibbs sampler works as follows.

Gibbs Sampler for 2D

- Sample $x^{(1)} \sim q(\cdot)$.
- For $i = 2, 3, \ldots,$
  - Sample $x_1^{(i)} \sim p(x_1 | x_2^{(i-1)})$.
  - Sample $x_2^{(i)} \sim p(x_2 | x_1^{(i)})$.
  - Set $x^{(i)} = [x_1^{(i)}, x_2^{(i)}]^{\mathsf{T}}$.

- Note that due to the special proposal, a Gibbs sampler does not have an accept-reject step as MH.

Outlook – MC for dynamical systems

Targeting $p(x_t \mid y_{1:t})$ in a nonlinear/non-Gaussian SSM using an importance sampler results in the particle filter (a member of the more general class of **Sequential Monte Carlo (SMC)** methods).

The Bayesian parameter inference problem in a general nonlinear/non-Gaussian SSM can be solved using the so called Particle Markov Chain Monte Carlo (PMCMC) methods. Here, an SMC algorithm is used as proposal to generate samples in an MCMC sampler.

Should you find this interesting I have a PhD course – *Computational learning in dynamical systems* – covering this material, see

`http://user.it.uu.se/~thosc112/CIDS.html`

A few concepts to summarize lecture 11

Monte Carlo Methods: Approximate inference tools using the samples from the target densities.

Basic Sampling Methods: The sampling methods to obtain independent samples from target densities. Though quite powerful, these would give bad results with high dimensions.

MCMC: Monte Carlo methods which produce dependent samples but more robust in high dimensions.

Metropolis Hastings Algorithm: The most well-known MCMC algorithm using arbitrary proposal densities.

Gibbs Sampler: A specific case of the MH sampler, which samples from conditionals iteratively and always accepts a new sample.