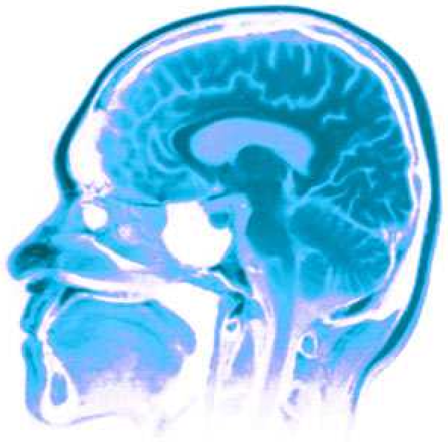
CPSC540



Monte Carlo for Stochastic Inference and Learning

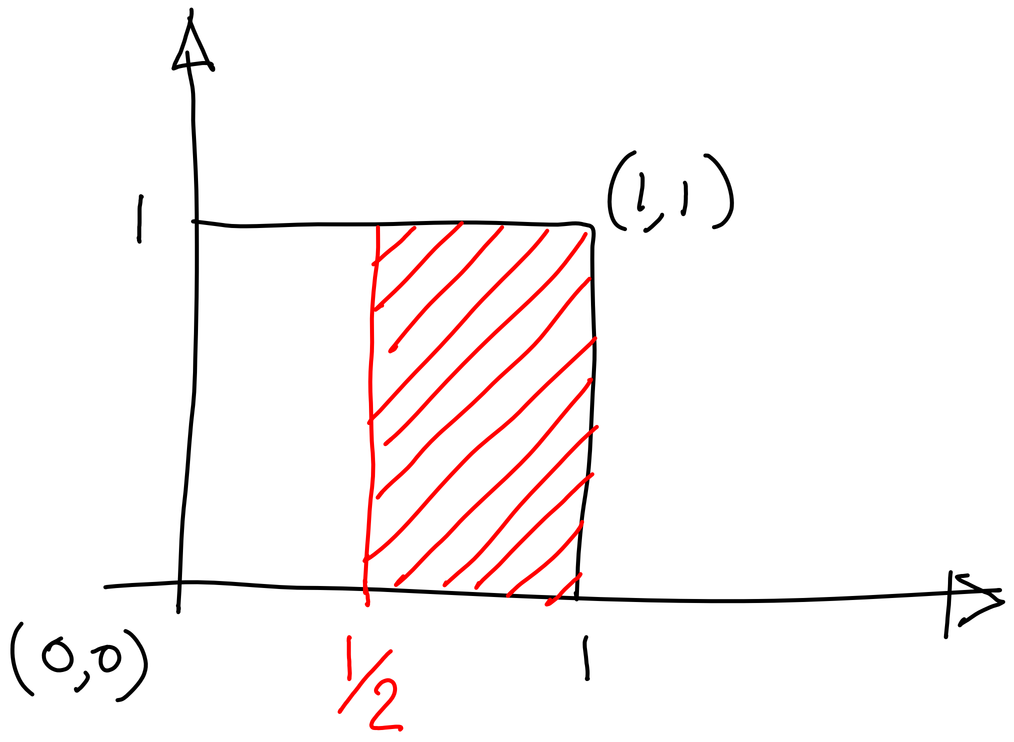
Nando de Freitas

2011

KPM Book Sections: 12

The idea

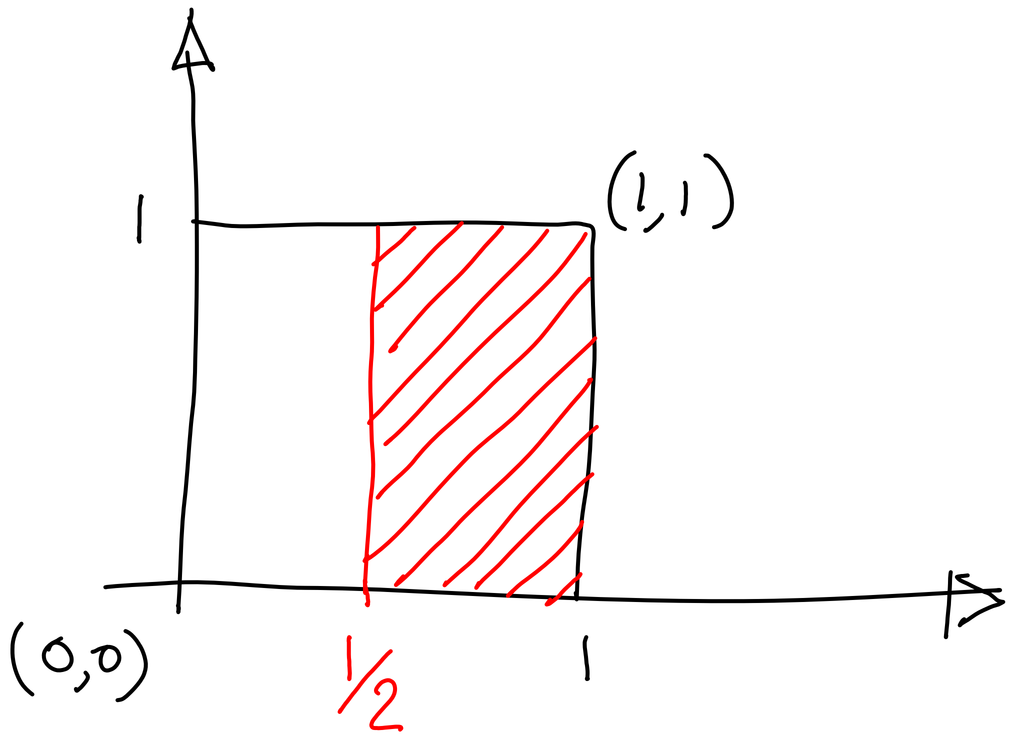
What is the probability that a dart thrown uniformly at random will hit the red area?



$P(\text{area}) =$

The idea

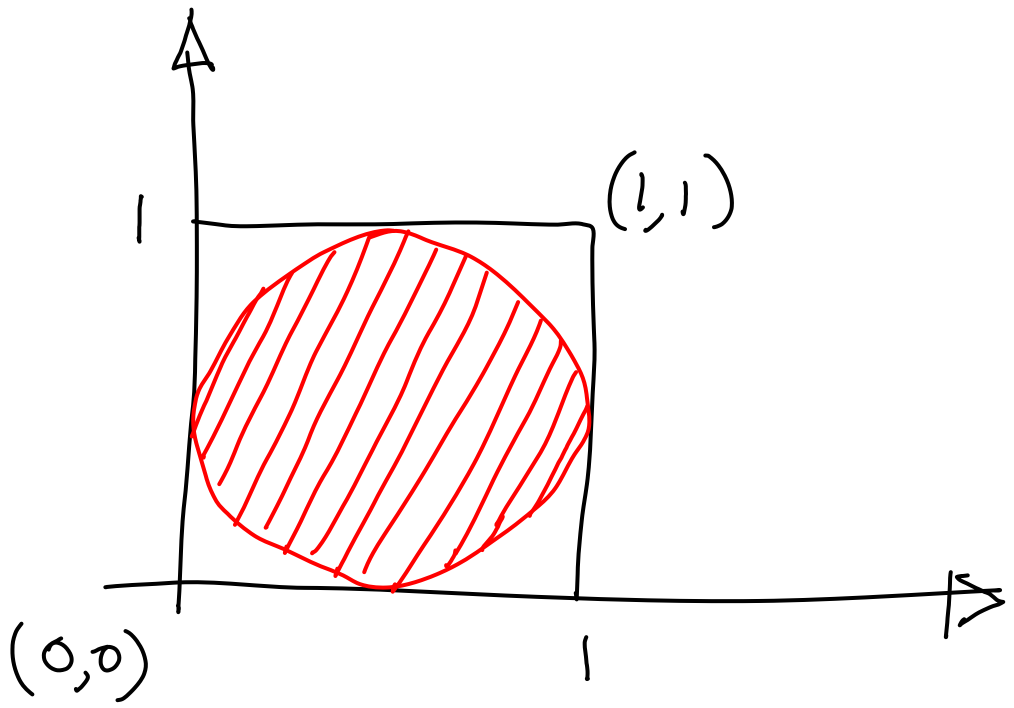
What is the probability that a dart thrown uniformly at random will hit the red area?



$$P(\text{area}) = \tfrac{1}{2}$$

The idea

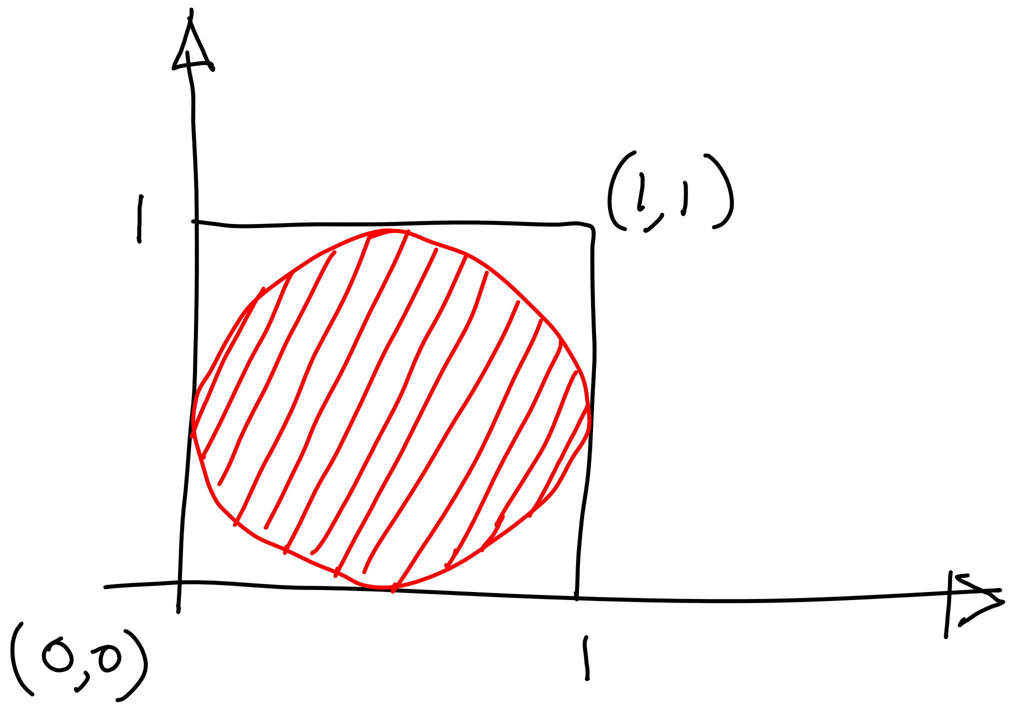
What is the probability that a dart thrown uniformly at random will hit the red area?



$$P(\text{area}) =$$

The idea

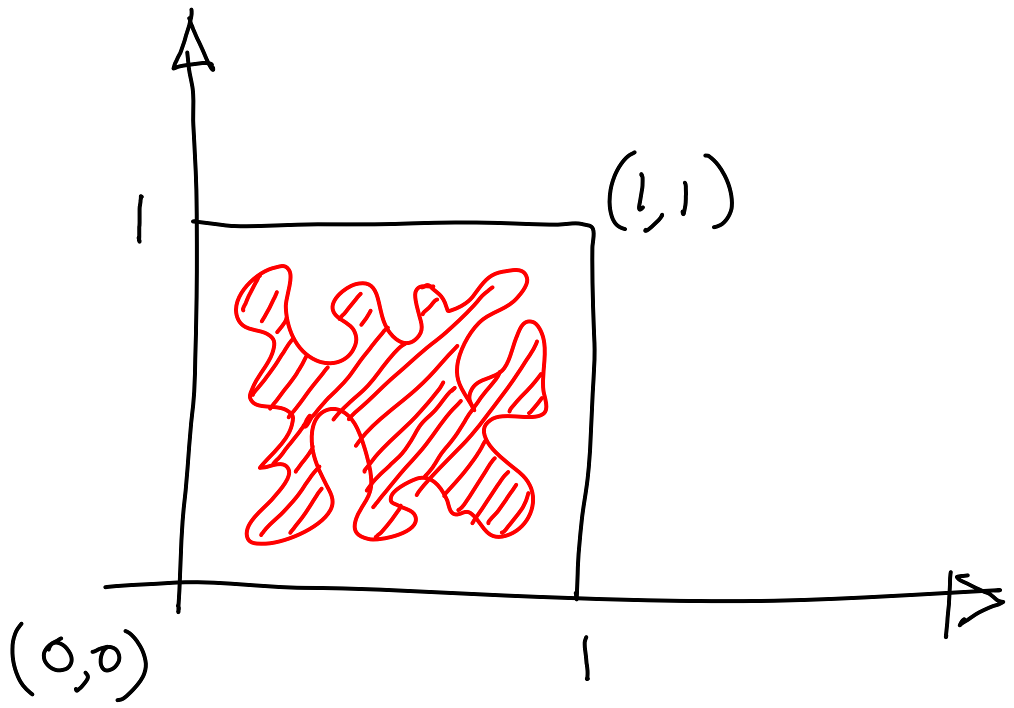
What is the probability that a dart thrown uniformly at random will hit the red area?



$$P(\text{area}) = \pi\left(\tfrac{1}{2}\right)^2 = \frac{\pi}{4}$$

The idea

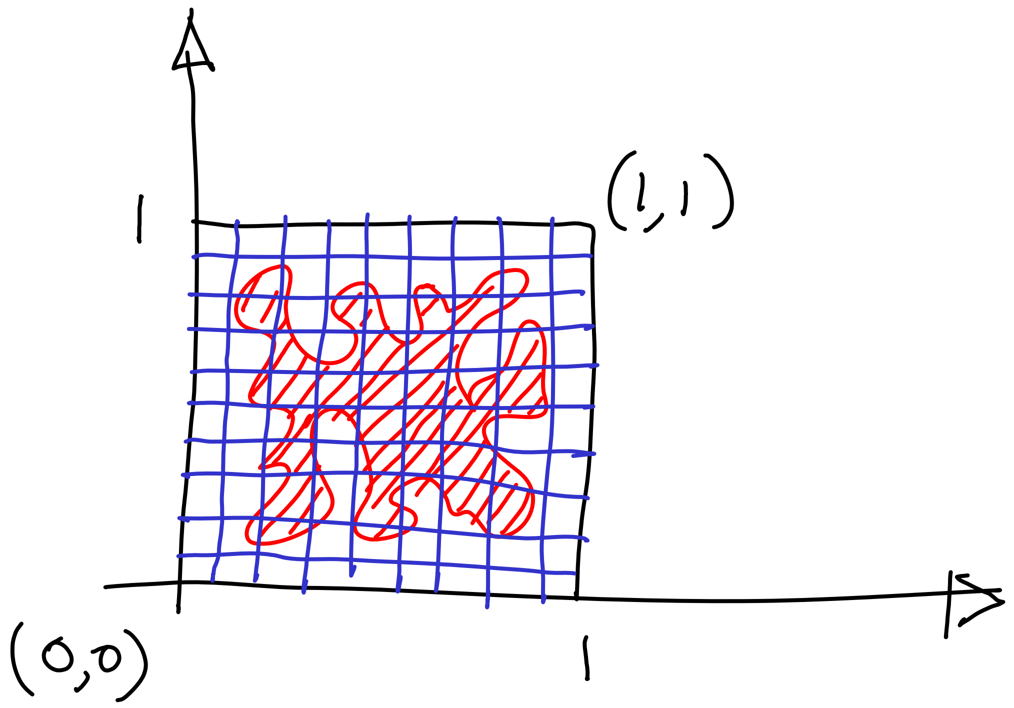
What is the probability that a dart thrown uniformly at random will hit the red area?



$P(\text{area}) =$

The idea

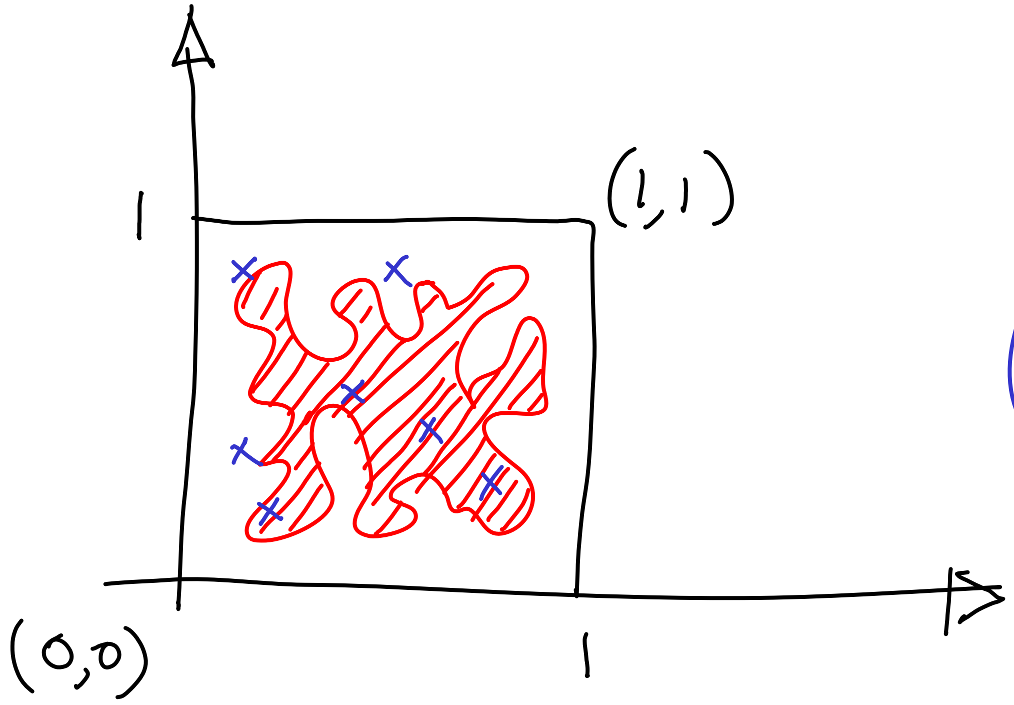
What is the probability that a dart thrown uniformly at random will hit the red area?

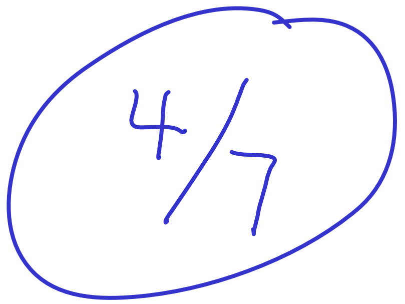


$$P(\text{area}) = \frac{\#\text{ red boxes}}{\#\text{ white boxes}}$$

The idea

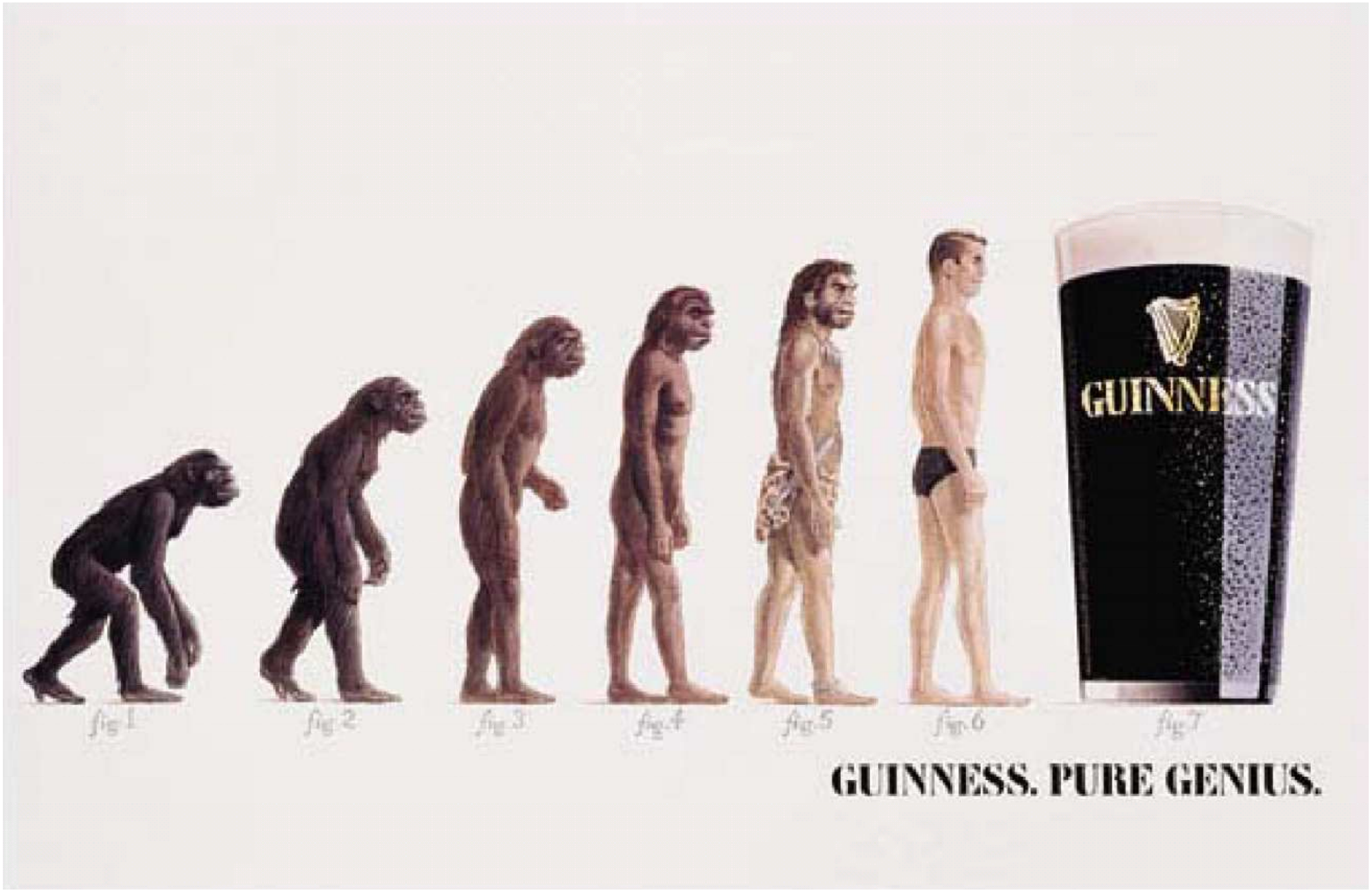
What is the probability that a dart thrown uniformly at random will hit the red area?



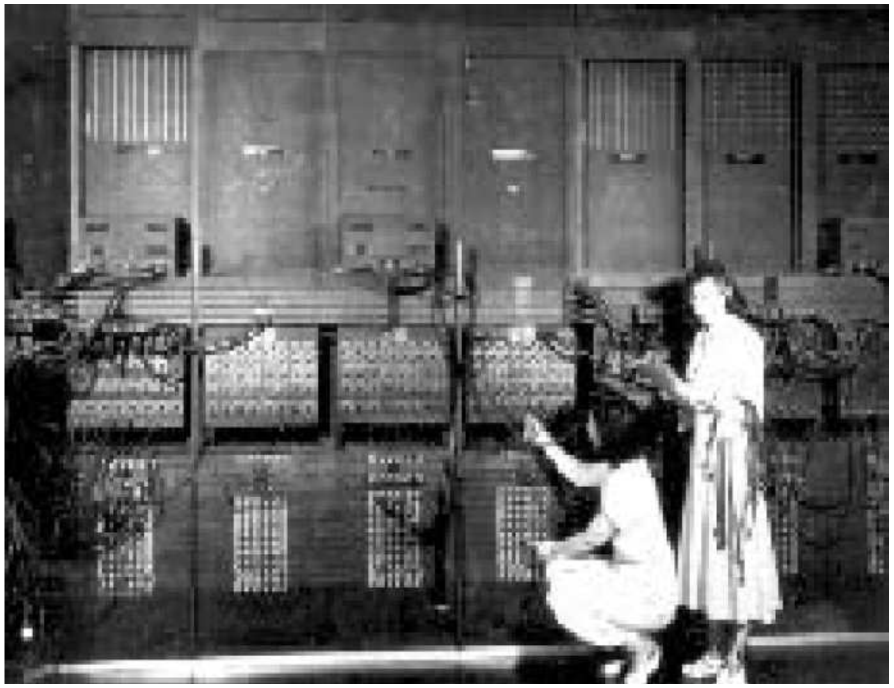


$$P(\text{area}) = \frac{\#\text{ darts in }}{\#\text{ darts in } \square}$$

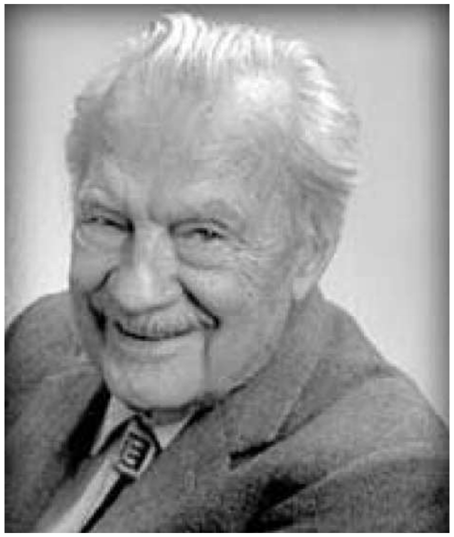
History of the Monte Carlo method



History of the Monte Carlo method: The bomb and ENIAC



History of the Monte Carlo method



Integrals in Probabilistic Inference

1. *Normalisation*:

$$p(x|y) = \frac{p(y|x)p(x)}{\int_X p(y|x^\star)p(x^\star)dx^\star}$$

2. *Marginalisation*:

$$p(x|y) = \int_Z p(x, z|y)dz$$

3. *Expectation*:

$$\mathbb{E}_{p(x|y)}(f(x)) = \int_X f(x)p(x|y)dx$$

Monte Carlo Integration

Suppose we want to compute

$$I = \int f(x)\, p(x \mid \text{data})\, dx$$

Monte Carlo Integration

Suppose we want to compute

$$I = \int f(x)\, P(x \mid \text{data})\, dx$$

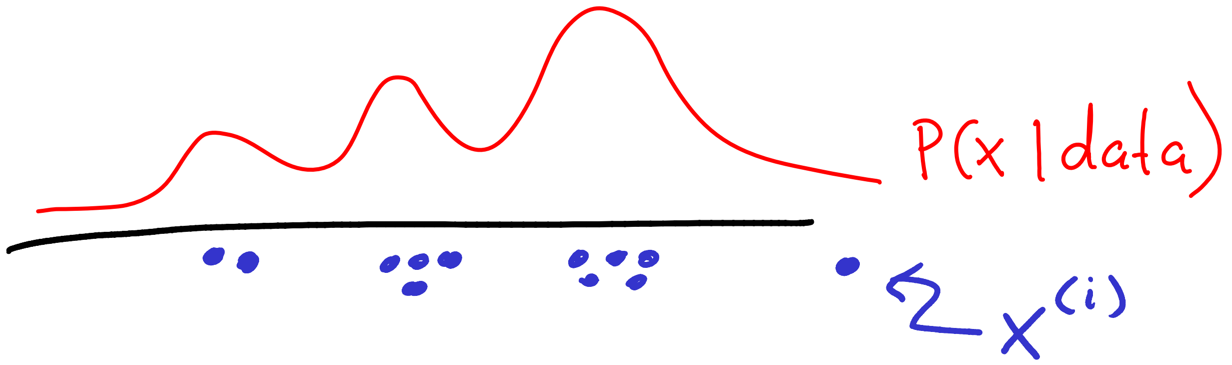
(i) Simulate $x^{(i)}\Big|_{i=1}^{N}$ from $P(x \mid \text{data})$

Monte Carlo Integration

Suppose we want to compute

$$I = \int f(x)\, p(x|\text{data})\, dx$$

(i) Simulate $x^{(i)}\Big|_{i=1}^{N}$ from $p(x|\text{data})$

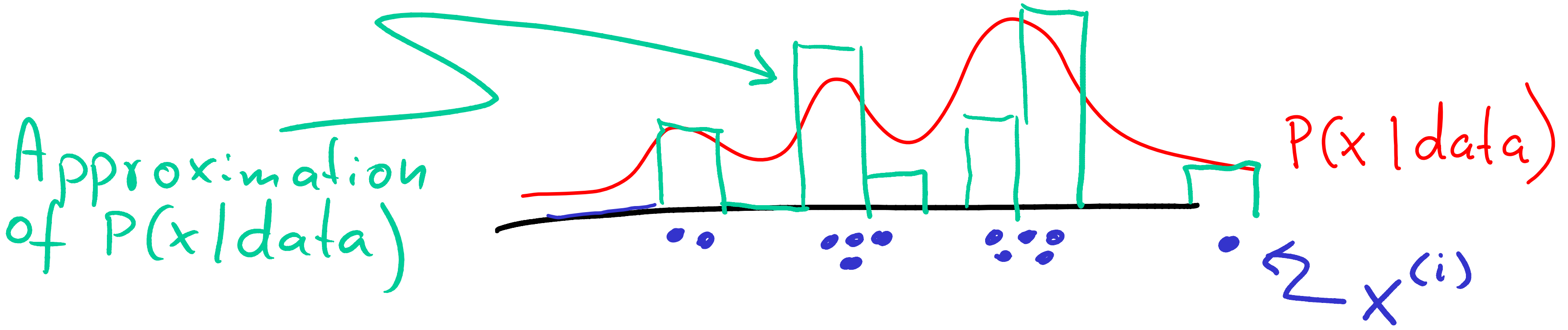


Monte Carlo Integration

Suppose we want to compute

$$I = \int f(x)\, P(x \mid \text{data})\, dx$$

(i) Simulate $x^{(i)}\Big|_{i=1}^{N}$ from $P(x \mid \text{data})$

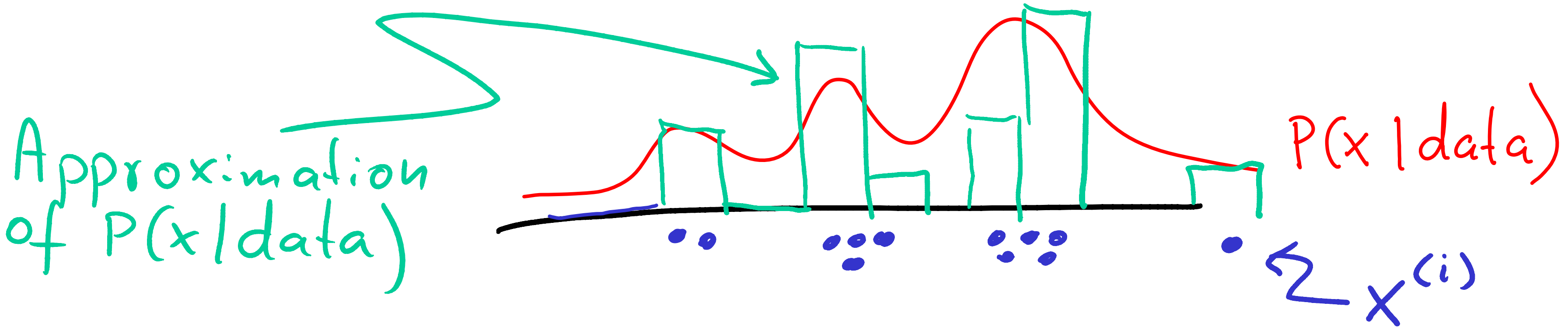


Monte Carlo Integration

Suppose we want to compute

$$I = \int f(x)\, p(x|\text{data})\, dx$$

(i) Simulate $x^{(i)}\Big|_{i=1}^{N}$ from $p(x|\text{data})$



(ii) Replace nasty integral with simple sum: $I \approx \frac{1}{N} \sum_{i=1}^{N} f(x^{(i)})$ ▨

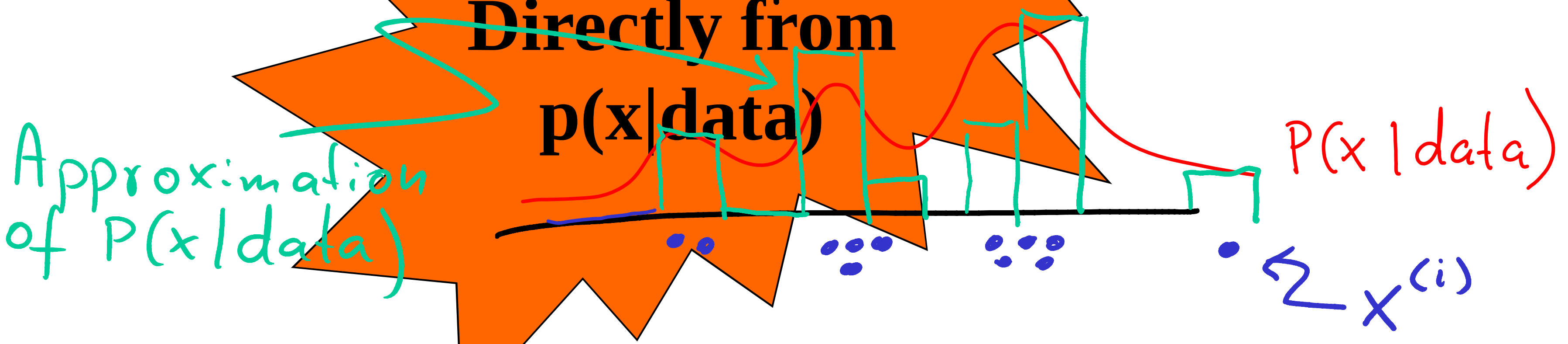
Monte Carlo Integration

Suppose we want to compute

$$I = \int f(x)\, p(x|\text{data})\, dx$$

(i) Simulate $\{x^{(i)}\}_{i=1}^{N}$ from $p(x|\text{data})$

Cannot sample Directly from p(x|data)



(ii) Replace nasty integral with simple sum:

$$I \approx \frac{1}{N} \sum_{i=1}^{N} f(x^{(i)})$$

Monte Carlo Integration Formally

The idea of Monte Carlo simulation is to draw an i.i.d. set of samples $\{x^{(i)}\}_{i=1}^{N}$ from a target density $p(x)$ defined on a high-dimensional space $\mathcal{X}$. These N samples can be used to approximate the target distribution with the following empirical point-mass function (think of it as a histogram):

$$p_N(dx) = \frac{1}{N}\sum_{i=1}^{N} \delta_{x^{(i)}}(dx),$$

where $\delta_{x^{(i)}}(dx)$ denotes the delta-Dirac mass located at $x^{(i)}$.

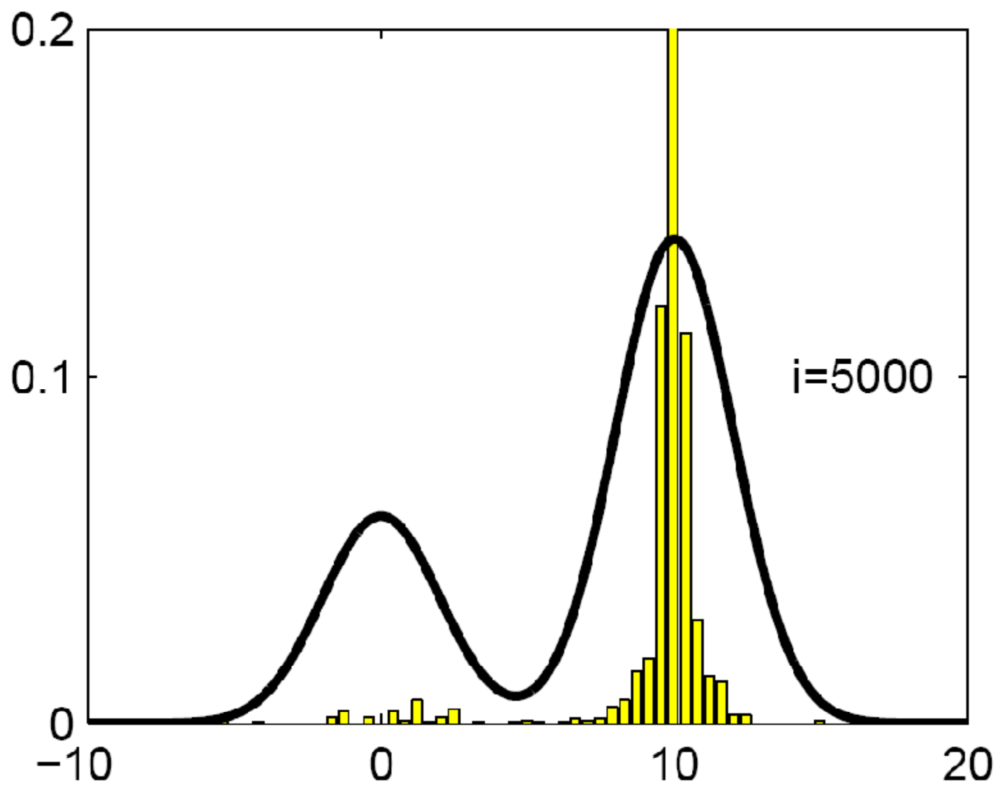
Monte Carlo Integration Formally

Consequently, one can approximate the integrals (or very large sums) $I\left(f\right)$ with tractable sums $I_N\left(f\right)$ as follows

$$I\left(f\right)=\int_{\mathcal{X}} f(x)p(x)dx.$$

Optimisation:
Concentrate Samples on Modes

$$\hat{x} = \underset{x^{(i)};i=1,\ldots,N}{\arg\max}\; p\left(x^{(i)}\right)$$

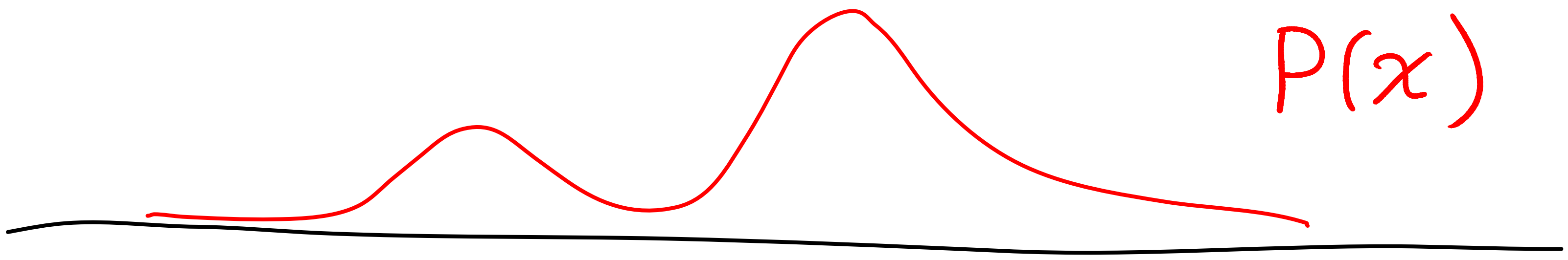


Rejection sampling

Set $i = 1$

Repeat until $i = N$

1. Sample $x^{(i)} \sim q(x)$ and $u \sim \mathcal{U}_{(0,1)}$.
2. If $u < \frac{p(x^{(i)})}{Mq(x^{(i)})}$ then accept $x^{(i)}$ and increment the counter i by 1. Otherwise, reject.

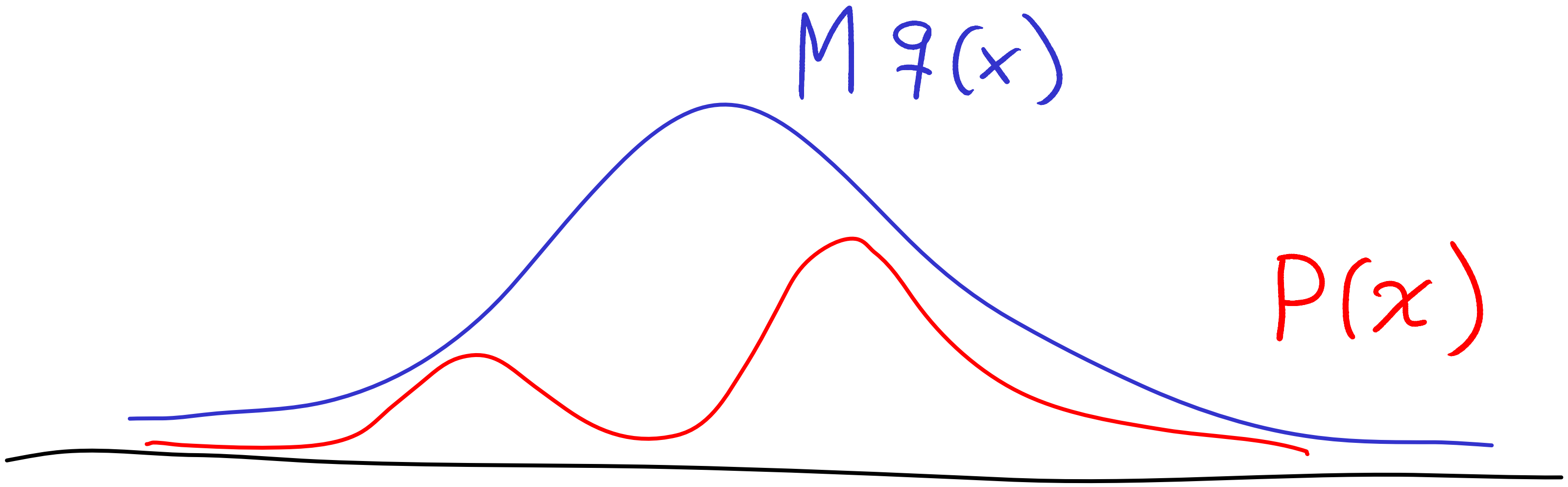


Rejection sampling

Set $i = 1$

Repeat until $i = N$

1. Sample $x^{(i)} \sim q(x)$ and $u \sim \mathcal{U}_{(0,1)}$.
2. If $u < \frac{p(x^{(i)})}{Mq(x^{(i)})}$ then accept $x^{(i)}$ and increment the counter i by 1. Otherwise, reject.

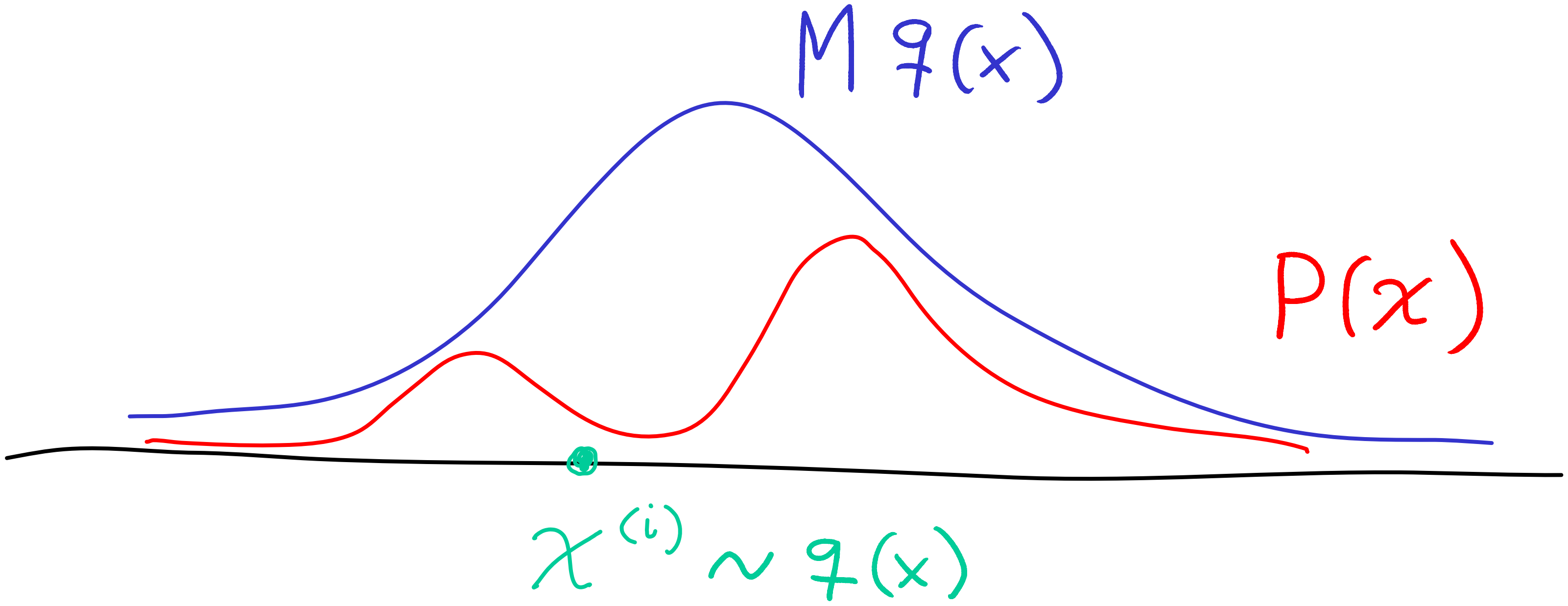


Rejection sampling

Set $i = 1$

Repeat until $i = N$

1. Sample $x^{(i)} \sim q(x)$ and $u \sim \mathcal{U}_{(0,1)}$.
2. If $u < \frac{p(x^{(i)})}{Mq(x^{(i)})}$ then accept $x^{(i)}$ and increment the counter i by 1. Otherwise, reject.

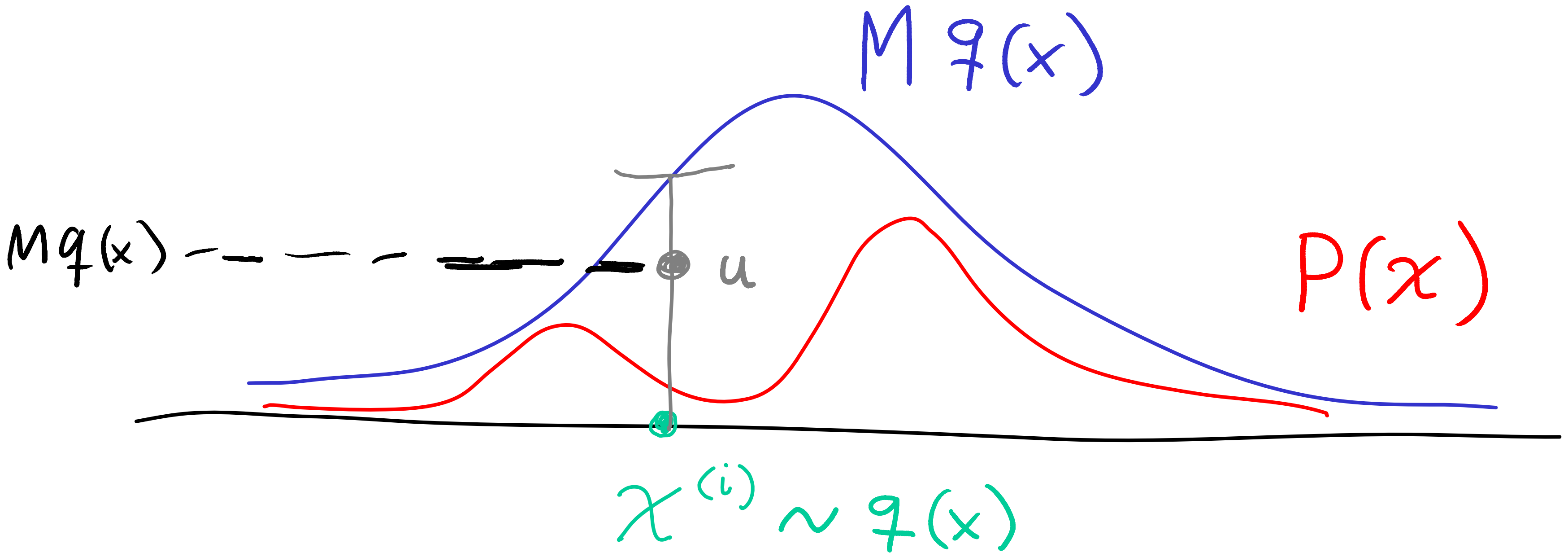


Rejection sampling

Set $i = 1$

Repeat until $i = N$

1. Sample $x^{(i)} \sim q(x)$ and $u \sim \mathcal{U}_{(0,1)}$.
2. If $u < \frac{p(x^{(i)})}{Mq(x^{(i)})}$ then accept $x^{(i)}$ and increment the counter i by 1. Otherwise, reject.

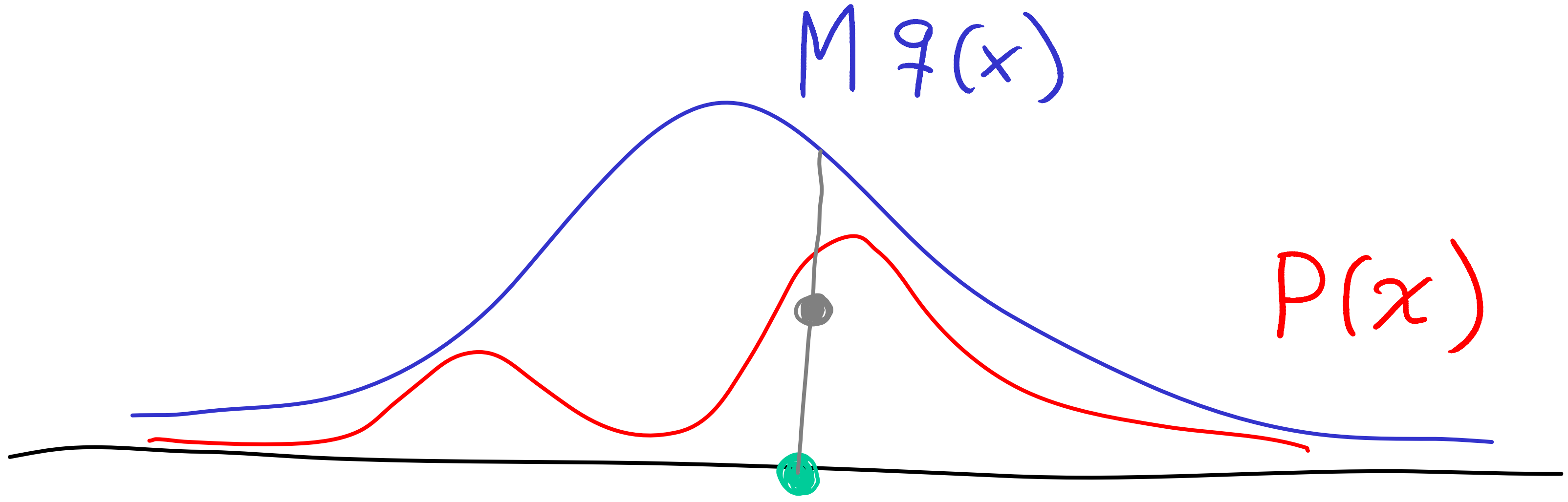


Rejection sampling

Set $i = 1$

Repeat until $i = N$

1. Sample $x^{(i)} \sim q(x)$ and $u \sim \mathcal{U}_{(0,1)}$.
2. If $u < \frac{p(x^{(i)})}{Mq(x^{(i)})}$ then accept $x^{(i)}$ and increment the counter i by 1. Otherwise, reject.

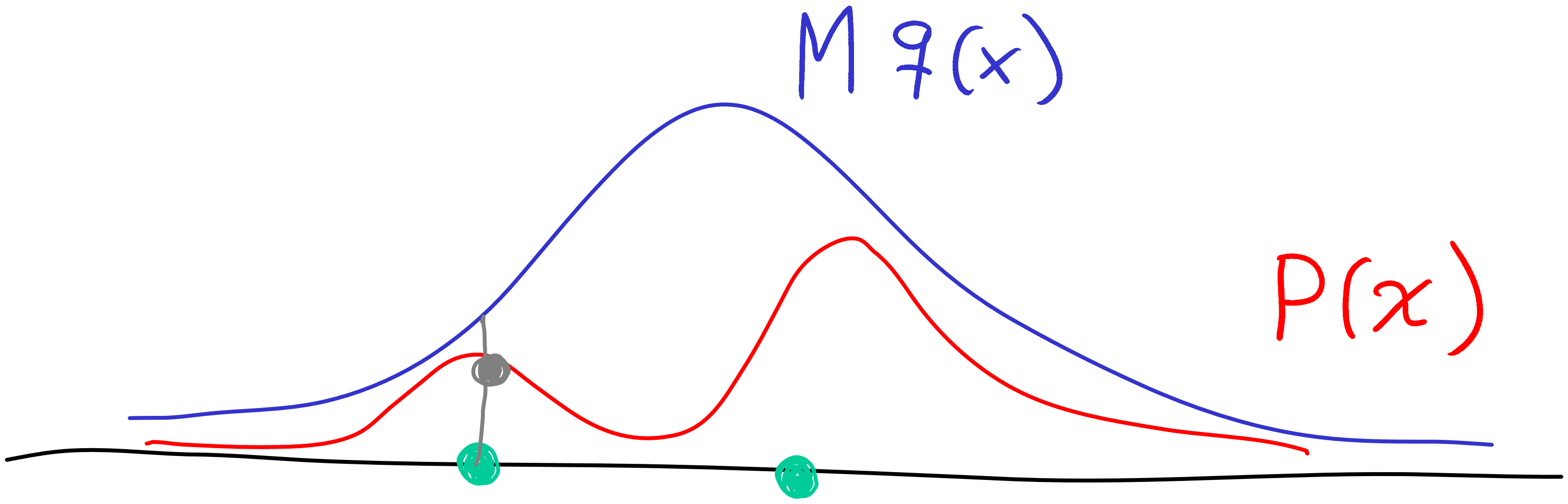


Rejection sampling

Set $i = 1$

Repeat until $i = N$

1. Sample $x^{(i)} \sim q(x)$ and $u \sim \mathcal{U}_{(0,1)}$.
2. If $u < \frac{p(x^{(i)})}{Mq(x^{(i)})}$ then accept $x^{(i)}$ and increment the counter i by 1. Otherwise, reject.

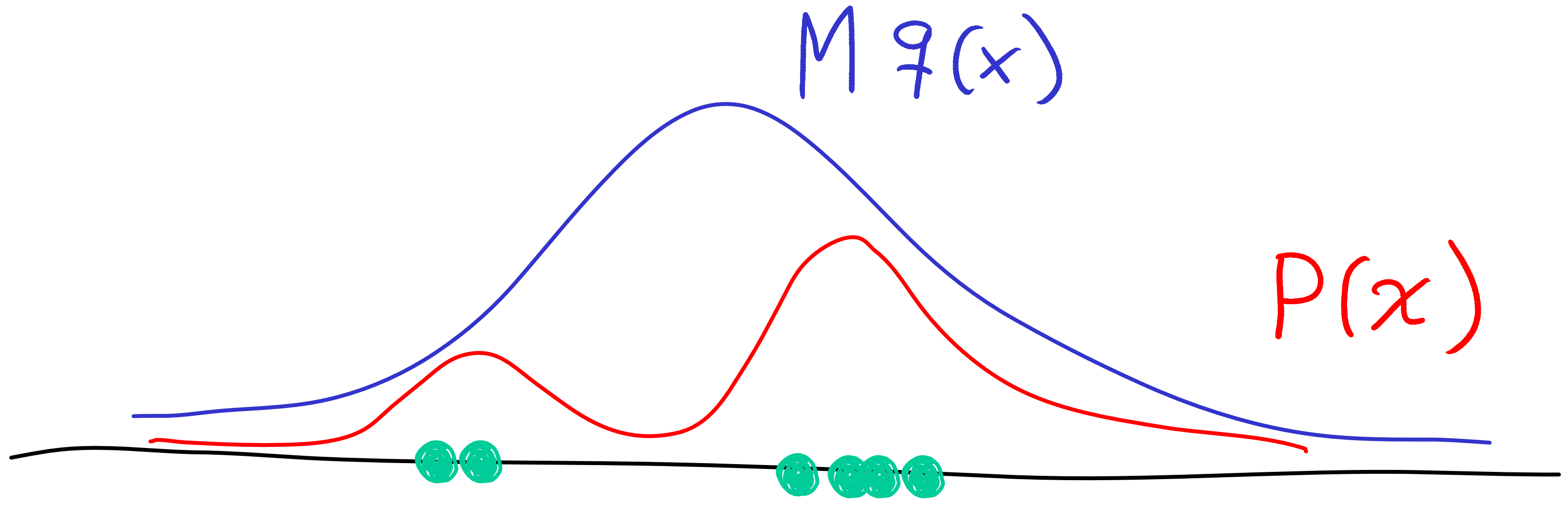


Rejection sampling

Set $i = 1$

Repeat until $i = N$

1. Sample $x^{(i)} \sim q(x)$ and $u \sim \mathcal{U}_{(0,1)}$.
2. If $u < \frac{p(x^{(i)})}{Mq(x^{(i)})}$ then accept $x^{(i)}$ and increment the counter i by 1. Otherwise, reject.



Importance Sampling

Importance sampling is a "classical" solution that goes back to the 1940's. Let us introduce an arbitrary importance proposal distribution $q(x)$ such that its support includes the support of $p(x)$ and such that we can sample from it. Then we can rewrite $I(f)$ as follows

$$I\left(f\right) = \int f\left(x\right) w\left(x\right) q\left(x\right) dx$$

where $w\left(x\right) \triangleq \frac{p(x)}{q(x)}$ is known as the *importance weight.*

Importance Sampling

Importance Sampling

Consequently, if one can simulate N i.i.d. samples $\{x^{(i)}\}_{i=1}^{N}$ according to $q(x)$ and evaluate $w(x^{(i)})$, a possible Monte Carlo estimate of $I(f)$ is

$$\widehat{I}_N(f) =$$

Importance Sampling

This estimator is unbiased and, under weak assumptions, the strong law of large numbers applies, that is $\widehat{I}_N(f) \xrightarrow[N\to\infty]{a.s.} I(f)$. It is clear that this integration method can also be interpreted as a sampling method where the posterior density $p(x)$ is approximated by:

$$\widehat{p}_N(dx) = \frac{1}{N}\sum_{i=1}^{N} w(x^{(i)})\delta_{x^{(i)}}(dx)$$

Some proposal distributions $q(x)$ will obviously be preferable to others.

Normalized Importance Sampling

When the normalising constant of $p(x)$ is unknown, it is still possible to apply the importance sampling method:

Normalized Importance Sampling

The Monte Carlo estimate of $I(f)$ becomes

$$\widetilde{I}_N(f) = \frac{\frac{1}{N}\sum_{i=1}^{N} f\left(x^{(i)}\right) w(x^{(i)})}{\frac{1}{N}\sum_{j=1}^{N} w\left(x^{(i)}\right)} = \sum_{i=1}^{N} f\left(x^{(i)}\right) \widetilde{w}(x^{(i)})$$

where $\widetilde{w}(x^{(i)})$ is a normalised importance weight. For N finite, $\widetilde{I}_N(f)$ is biased (ratio of two estimates) but asymptotically, under weak assumptions, the strong law of large numbers applies, that is $\widetilde{I}_N(f) \xrightarrow[N\to\infty]{a.s.} I(f)$.

Sampling-Importance Sampling (SIR)

If one is interested in obtaining M $i.i.d.$ samples from $\widehat{p}_N(x)$, then an asymptotically $(N/M \rightarrow \infty)$ valid method consists of resampling M times according to the discrete distribution $\widehat{p}_N(x)$.

Sampling-Importance Sampling (SIR)

This procedure results in M samples $\widetilde{x}^{(i)}$ with the possibility that $\widetilde{x}^{(i)} = \widetilde{x}^{(j)}$ for $i \neq j$. After resampling, the approximation of the target density is

$$\widetilde{p}_M(dx) = \frac{1}{M}\sum_{i=1}^{M} \delta_{\widetilde{x}^{(i)}}(dx)$$

Sampling-Importance Sampling (SIR)

Set $i = 1$

Repeat until $i = N$

1. Sample $x^{(i)} \sim q\left(x\right)$
2. Evaluate $p(x^{(i)}|y)$ up to a normalising constant.
3. Evaluate $q(x^{(i)})$ up to a normalising constant.
4. Compute $w(x^{(i)})$.

Normalise $w(x^{(i)})$ to obtain $\widetilde{w}(x^{(i)})$.

Resample $\{x^{(i)}, \widetilde{w}(x^{(i)})\}_{i=1}^{N} \quad \longrightarrow \quad \{\widetilde{x}^{(i)}, 1/N\}_{i=1}^{N}$

What is the best proposal?

The IS estimator is unbiased, but has variance

$$\mathrm{var}_{q(x)}\left(\widehat{I}_N(f)\right) = \mathbb{E}_{q(x)}\left(f^2(x)w^2(x)\right) - I^2(f)$$

This variance is minimised when

$$q^\star(x) = \frac{|f(x)|p(x)}{\int |f(x)|p(x)dx}$$

What is the best proposal?

Introduce parametric proposals and adapt the parameters so as to minimise the variance

$$\theta_{t+1} = \theta_t - \alpha \frac{1}{N} \sum_{i=1}^{N} f^2(x^{(i)}) w(x^{(i)}, \theta_t) \frac{\partial w(x^{(i)}, \theta_t)}{\partial \theta_t}$$

where α is a learning rate and $x^{(i)} \sim q(x, \theta)$.

Proposal distributions that adapt to the data are also very widely used.

IS Example: Logistic Regression

Given the input-output i.i.d. data sets $x \triangleq \underline{x_{1:T} \triangleq \{x_0, x_1, \ldots}, x_T\}$ and $y \triangleq y_{1:T} \triangleq \{y_0, y_1, \ldots, y_T\}$, where $x_t \in \mathbb{R}$ and $y_t \in \{0, 1\}$. The idea is to come up with a model that takes a new input x_{T+1} and produces as output $p(y_{T+1} = 1 | x_{T+1})$ and $p(y_{T+1} = 0 | x_{T+1})$. This classification problem arises in several areas of technology, including condition monitoring and binary decision systems. For example, when monitoring patients, we might wish to decide whether they require an increase in drug intake based on new evidence.

IS Example: Logistic Regression

For practical reasons, we parameterise our model. In particular, we introduce the following Bernoulli likelihood function:

$$p(y_t|x_t,\theta) = \left[\frac{1}{1+\exp(-\theta x_t)}\right]^{y_t}\left[1-\frac{1}{1+\exp(-\theta x_t)}\right]^{1-y_t}$$

where θ are the model parameters. The logistic function $p(y_t = 1|x_t) = \frac{1}{1+\exp(-\theta x_t)}$ is conviniently bounded between 0 and 1.

IS Example: Logistic Regression

We also assume a Gaussian prior

$$p(\theta) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}(\theta - \mu)'(\theta - \mu)\right)$$

The goal of the analysis is then to compute the posterior distribution $p(\theta|x_{1:T}, y_{1:T})$. This distribution will enable us to classify new data as follows

$$p(y_{T+1}|x_{1:T+1}) = \int_{\Theta} p(y_{T+1}|x_{T+1}, \theta)p(\theta|x_{1:T}, y_{1:T})d\theta$$

IS Example: Logistic Regression

Bayes' rule gives us the following expression for the posterior

$$p(\theta|x_{1:T}, y_{1:T}) \;\propto\; \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}(\theta-\mu)'(\theta-\mu)\right)$$

$$\times \prod_{t=1}^{T} \left[\frac{1}{1+\exp\left(-\theta' x\right)}\right]^{y_t} \left[1 - \frac{1}{1+\exp\left(-\theta' x\right)}\right]^{1-y_t}$$

IS Example: Logistic Regression

$$p(y_t|x_t,\theta) = \left[\frac{1}{1+\exp(-\theta x_t)}\right]^{y_t} \left[1 - \frac{1}{1+\exp(-\theta x_t)}\right]^{1-y_t}$$

$$p(\theta) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}(\theta-\mu)'(\theta-\mu)\right)$$

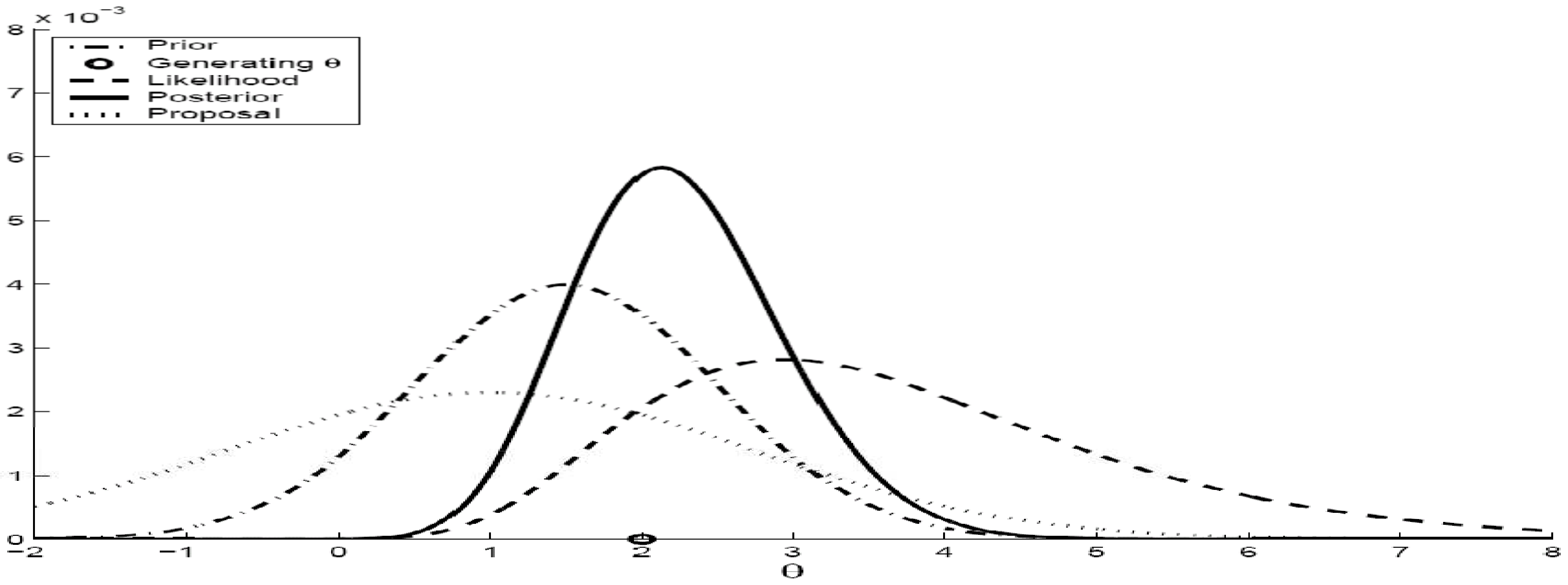
$$\begin{aligned} p(\theta|x_{1:T}, y_{1:T}) \quad &\propto \quad \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}(\theta-\mu)'(\theta-\mu)\right) \\ &\times \prod_{t=1}^{T} \left[\frac{1}{1+\exp(-\theta' x)}\right]^{y_t} \left[1 - \frac{1}{1+\exp(-\theta' x)}\right]^{1-y_t} \end{aligned}$$

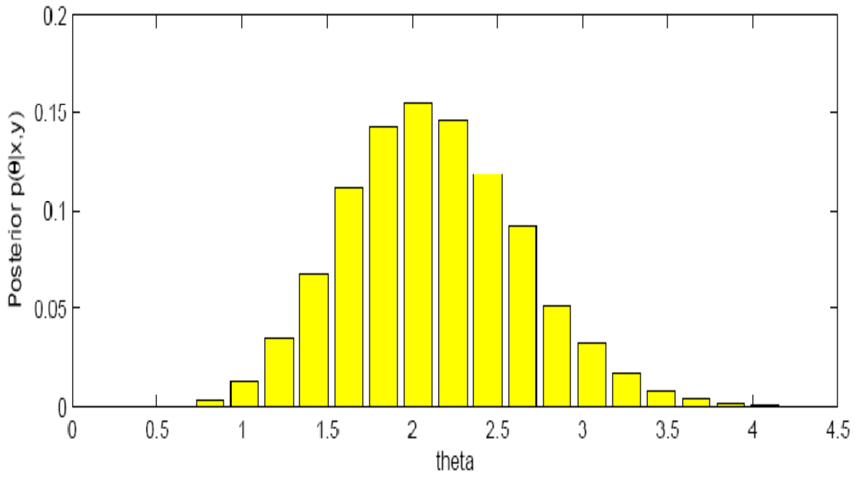
$x_t \in \mathbb{R}$ and $y_t \in \{0,1\}$

IS Example: Logistic Regression

The problem is that in this case we can't solve the normalising integral analytically. So we have to use numerical methods — in this case importance sampling — to approximate $p(\theta|x_{1:T}, y_{1:T})$. Note that we cannot sample from $p(\theta|x_{1:T}, y_{1:T})$ directly because we don't know the normalising constant. So instead we sample from a proposal distribution $q(\theta)$ (say a Gaussian) and weight the samples using importance sampling. After obtaining N samples of θ from the posterior, we can classify new data as follows

IS Example: Logistic Regression

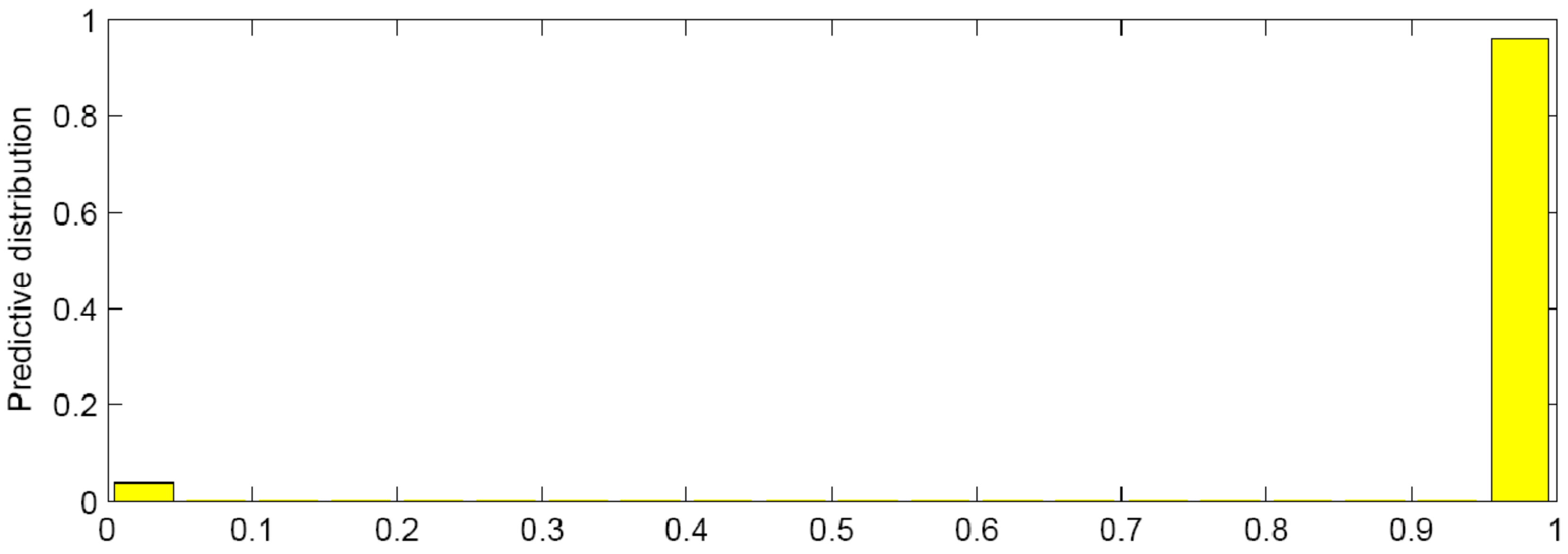




IS Example: Logistic Regression

$$p(y_{T+1}|x_{1:T+1}) = \int_\Theta p(y_{T+1}|x_{T+1},\theta)p(\theta|x_{1:T},y_{1:T})d\theta$$

$$p(y_{T+1}|x_{1:T+1}) = \frac{1}{N}\sum_{i=1}^{N} p(y_{T+1}|x_{T+1},\theta^{(i)})$$



Dynamic models and particle filtering

- Unknown states: $\mathbf{x}_{0:t} = \{\mathbf{x}_0, ..., \mathbf{x}_t\}$.
- Observations: $\mathbf{y}_{1:t} = \{\mathbf{y}_1, ..., \mathbf{y}_t\}$.
- Model:

$$p(\mathbf{x}_0)$$

$$p(\mathbf{x}_t | \mathbf{x}_{t-1}) \quad \text{for } t \geq 1$$

$$p(\mathbf{y}_t | \mathbf{x}_t) \quad \text{for } t \geq 1$$

Dynamic models and particle filtering

Stochastic Volatility

$$
\begin{aligned}
y_t &= e^{\alpha_t/2}\sigma_t\varepsilon_t \\
\log\sigma_t^2 &= \beta\log\sigma_{t-1}^2 + v_t \\
\alpha_t &= \alpha_{t-1} + u_{t1} \\
\beta_t &= \beta_{t-1} + u_{t2}
\end{aligned}
$$

Dynamic models and particle filtering

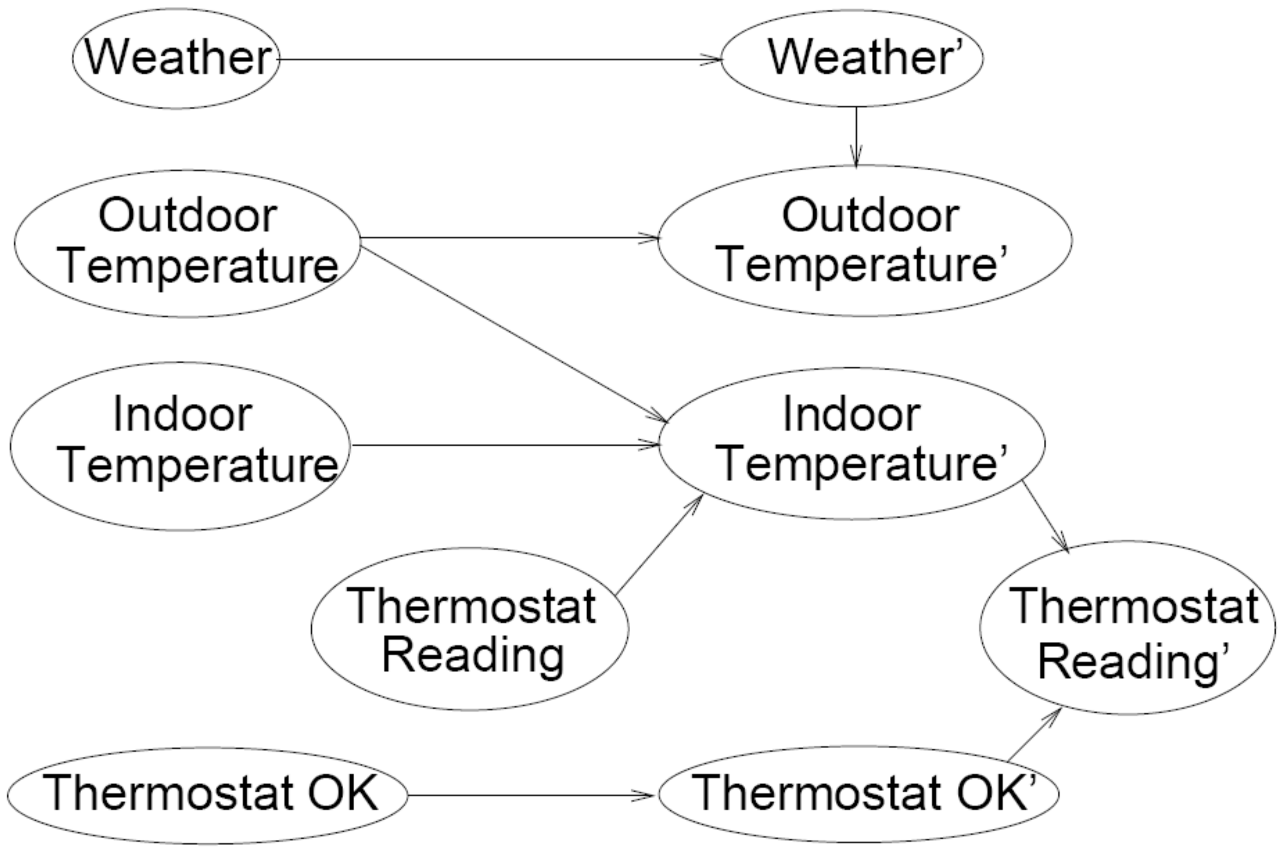
Target Tracking

$$\begin{pmatrix} s_{t,1} \\ s_{t,2} \\ v_{t,1} \\ v_{t,2} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} s_{t-1,1} \\ s_{t-1,2} \\ v_{t-1,1} \\ v_{t-1,2} \end{pmatrix} + \begin{pmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \epsilon_{t,1} \\ \epsilon_{t,2} \end{pmatrix}$$

$$\begin{pmatrix} y_{t,1} \\ y_{t,2} \end{pmatrix} = \begin{pmatrix} s_{t,1} \\ s_{t,2} \end{pmatrix} + \begin{pmatrix} e_{t,1} \\ e_{t,2} \end{pmatrix}$$

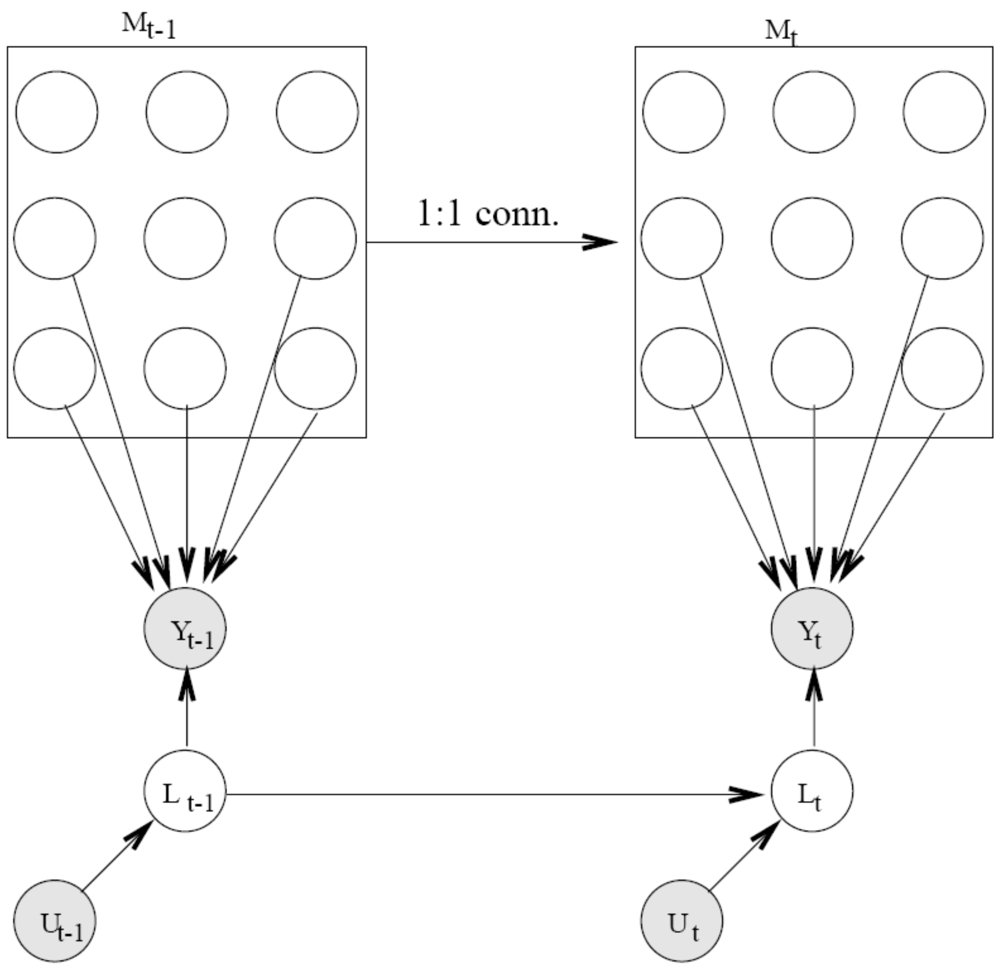
Dynamic models and particle filtering

Dynamic Bayesian Networks

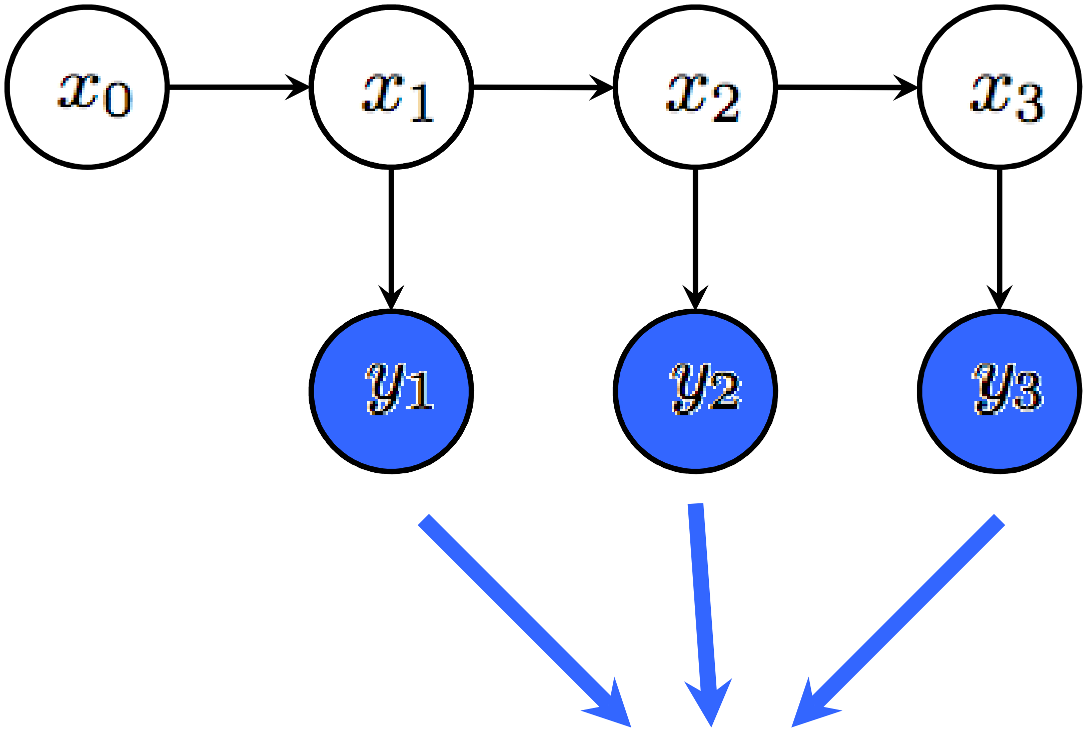


Dynamic models and particle filtering

Robot Localisation and Map Learning



Non-linear non-Gaussian filtering



$$p(x_3|y_{1:3})$$

$$p(x_t|y_{1:t}) \propto p(y_t|x_t) \int p(x_t|x_{t-1}) p(x_t|y_{1:t-1}) dx_{t-1}$$

Nasty integral

Importance Sampling for Optimal Filtering / Tracking

- Input: $p\left(\mathbf{x}_{t-1} \mid \mathbf{y}_{1:t-1}\right)$
- Prediction:

$$p\left(\mathbf{x}_t \mid \mathbf{y}_{1:t-1}\right) = \int p\left(\mathbf{x}_t \mid \mathbf{x}_{t-1}\right) p\left(\mathbf{x}_{t-1} \mid \mathbf{y}_{1:t-1}\right) d\mathbf{x}_{t-1}$$

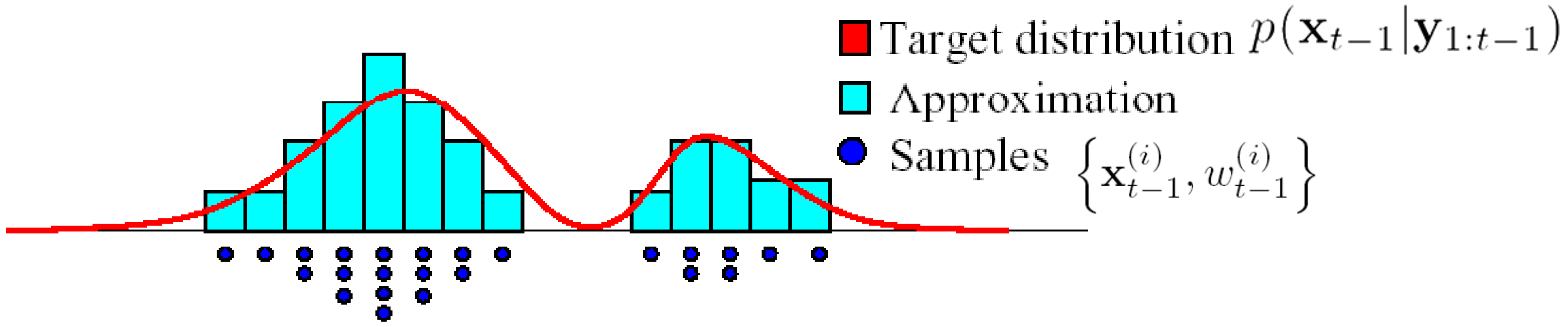
- Bayes update:

$$p\left(\mathbf{x}_t \mid \mathbf{y}_{1:t}\right) = \frac{p\left(\mathbf{y}_t \mid \mathbf{x}_t\right) p\left(\mathbf{x}_t \mid \mathbf{y}_{1:t-1}\right)}{\int p\left(\mathbf{y}_t \mid \mathbf{x}_t\right) p\left(\mathbf{x}_t \mid \mathbf{y}_{1:t-1}\right) d\mathbf{x}_t}$$

- Output: $p\left(\mathbf{x}_t \mid \mathbf{y}_{1:t}\right)$

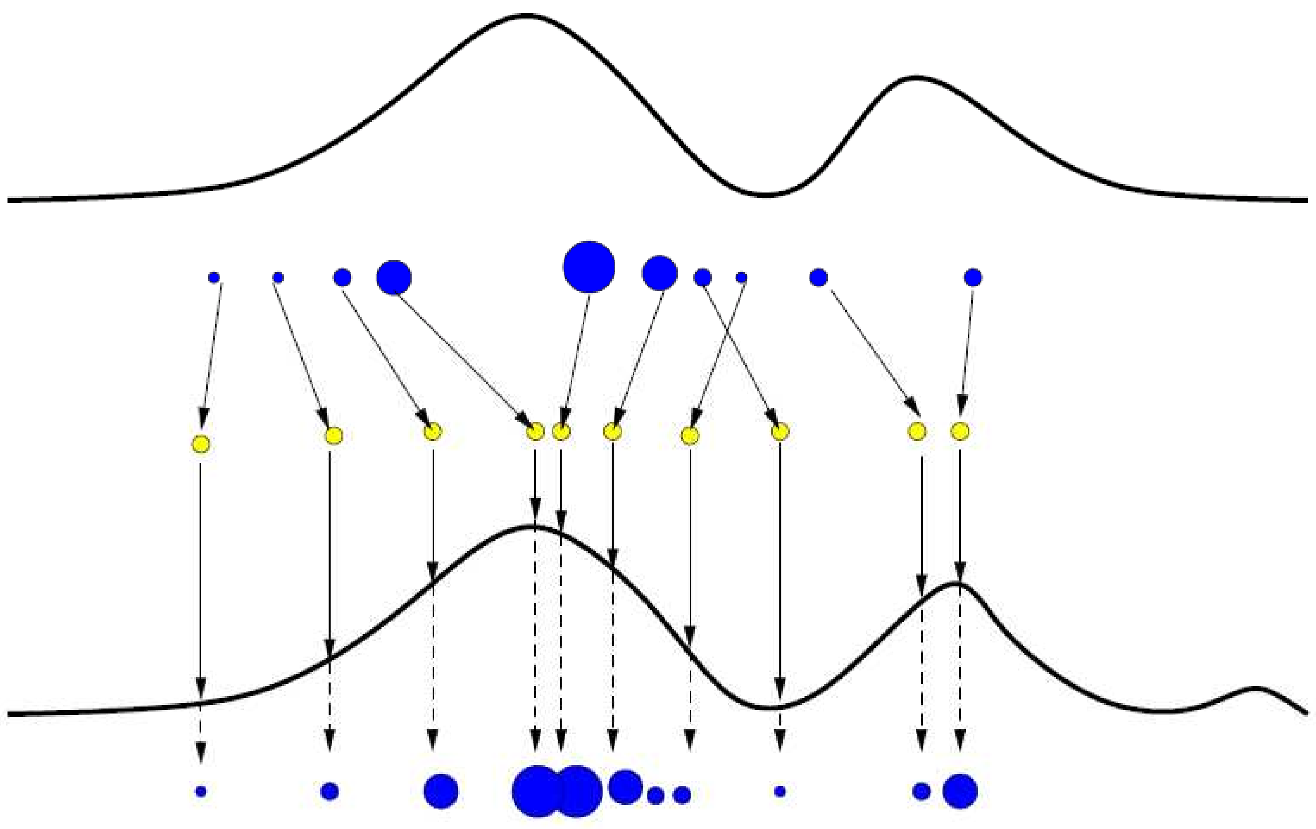
One can Compute the Integrals Recursively in Time

Given the samples $\left\{\mathbf{x}_{t-1}^{(i)}, w_{t-1}^{(i)}\right\}$ from $p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})$

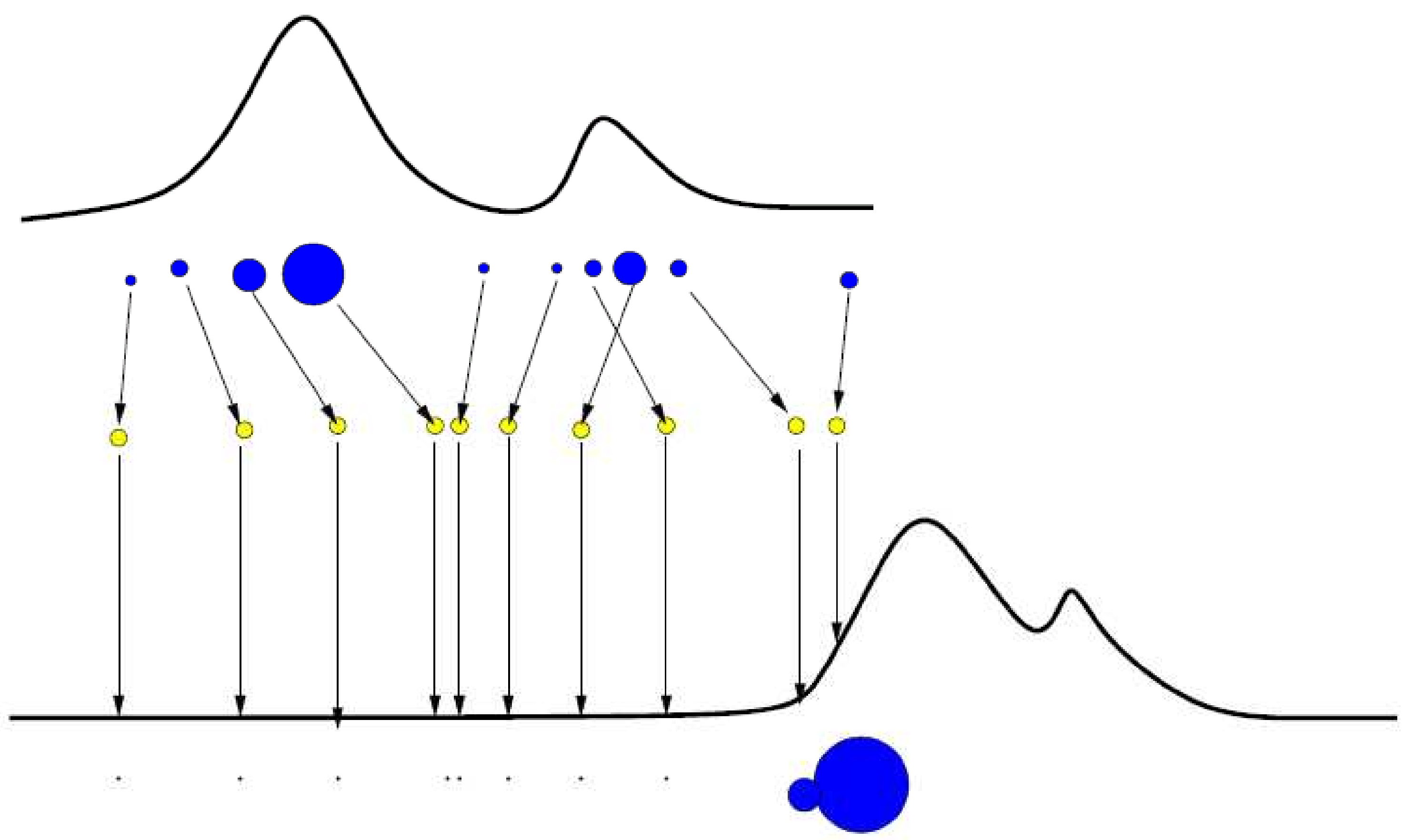


$$\int p(\mathbf{x}_t|\mathbf{x}_{t-1})\,p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})\,d\mathbf{x}_{t-1} \rightarrow \sum_{j=1}^{N} w_{t-1}^{(j)} p\left(\mathbf{x}_t \middle| \mathbf{x}_{t-1}^{(j)}\right)$$

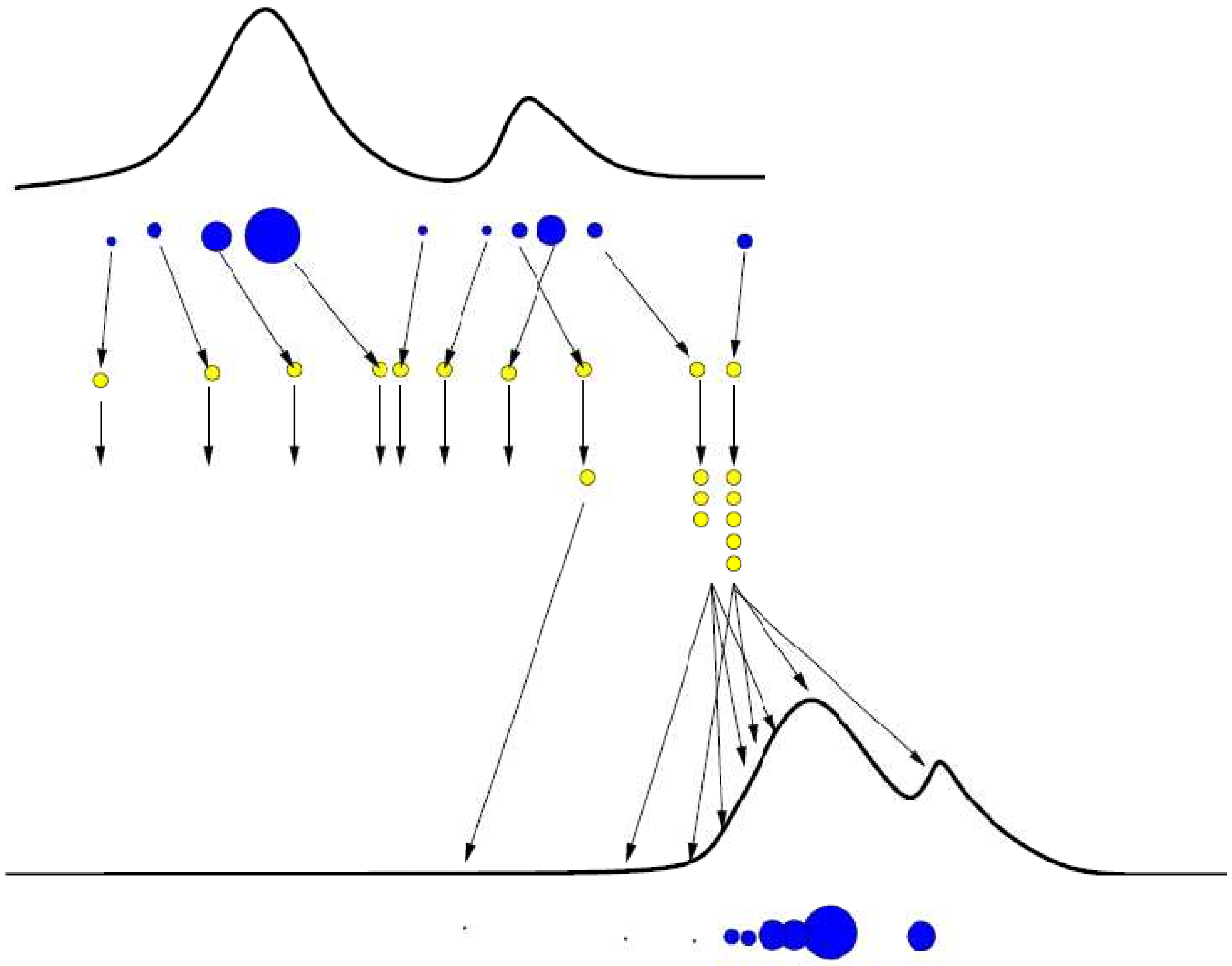
Particle Filtering (SIS)



Particle Filtering (SIS)



Particle Filtering (SIR)



Particle Filtering Code

- For $i = 1, ..., N$, sample $\mathbf{x}_0^{(i)} \sim p(\mathbf{x}_0)$ and set $t = 1$.

- For $i = 1, ..., N$, sample $\widetilde{\mathbf{x}}_t^{(i)} \sim p\left(\mathbf{x}_t \middle| \mathbf{x}_{t-1}^{(i)}\right)$.
- For $i = 1, ..., N$, evaluate the importance weights

$$\widetilde{w}_t^{(i)} = p\left(\mathbf{y}_t \middle| \widetilde{\mathbf{x}}_t^{(i)}\right)$$

- Normalise the importance weights.
- Select • ttest samples (black-box).

Particle Filtering Example

$$\begin{aligned} x_t &= \frac{1}{2}x_{t-1} + 25\frac{x_{t-1}}{1+x_{t-1}^2} + 8\cos(1.2t) + v_t \\ y_t &= \frac{x_t^2}{20} + w_t \end{aligned}$$

where $x_0 \sim \mathcal{N}\left(0, \sigma_1^2\right)$, v_t and w_t are mutually independent white Gaussian noises, $v_t \sim \mathcal{N}\left(0, \sigma_v^2\right)$ and $w_t \sim \mathcal{N}\left(0, \sigma_w^2\right)$

Particle Filtering Example

- For $i = 1, \ldots, N$, sample $\mathbf{x}_0^{(i)} \sim \mathcal{N}\left(0, \sigma_1^2\right)$
- For $i = 1, \ldots, N$, sample

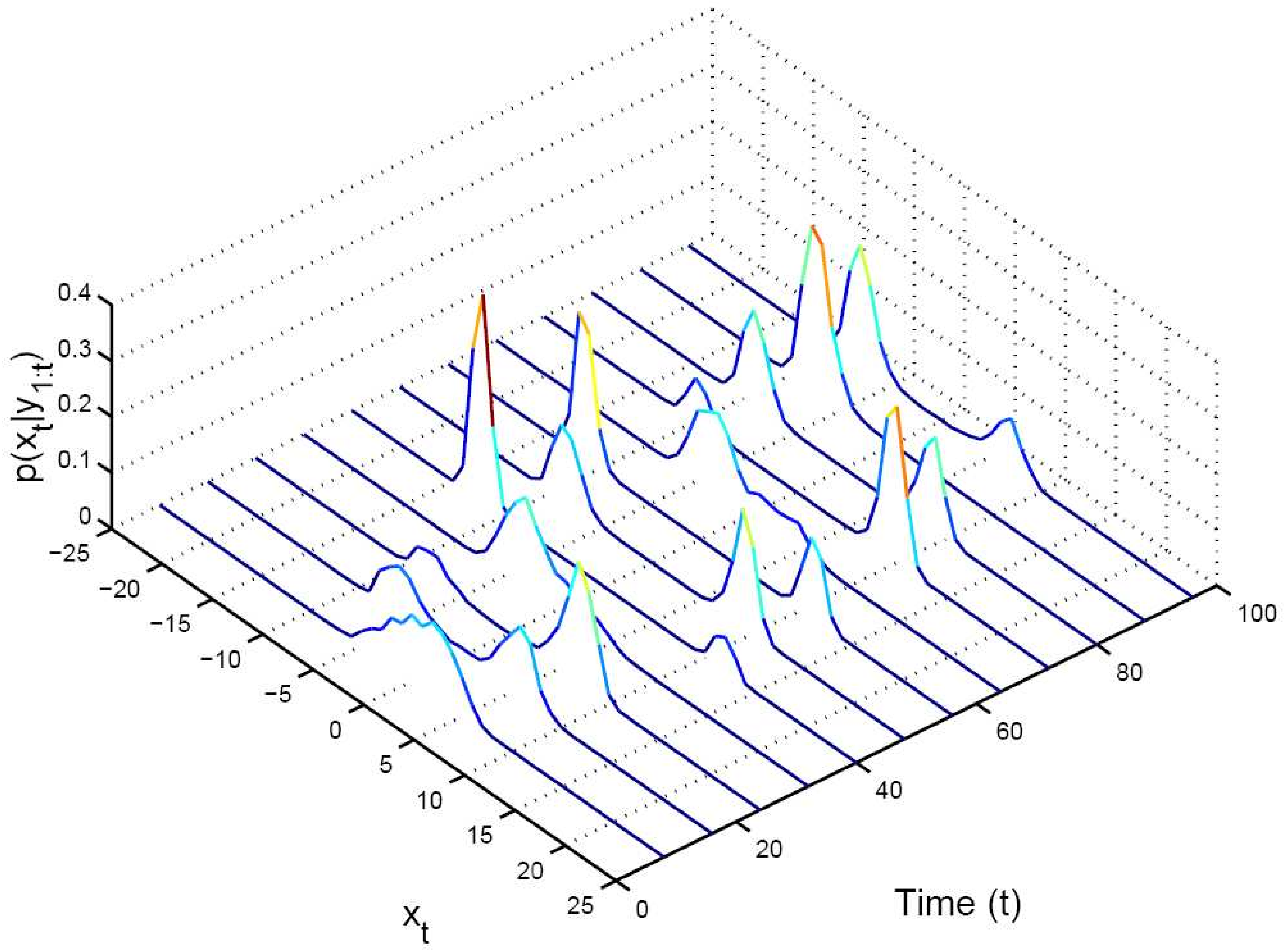
$$x_t^{(i)} = \frac{1}{2} x_{t-1}^{(i)} + 25 \frac{x_{t-1}^{(i)}}{1 + x_{t-1}^{2(i)}} + 8 \cos(1.2t) + \mathcal{N}\left(0, \sigma_v^2\right)$$

- For $i = 1, \ldots, N$, evaluate the importance weights

$$\widetilde{w}_t^{(i)} = \frac{1}{\sqrt{2\pi\sigma_w^2}} e^{-\frac{1}{2\sigma_w^2}\left(y - \frac{x_t^{2(i)}}{20}\right)^2}$$

- Normalise the importance weights.
- Resample fittest samples (black-box).

Particle Filtering Example



Particle Methods More Generally

The goal is to approximate a target distribution over a sequence of states $\mathbf{x}_{1:n} \triangleq \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ that is growing with “time” as well as the partition function.

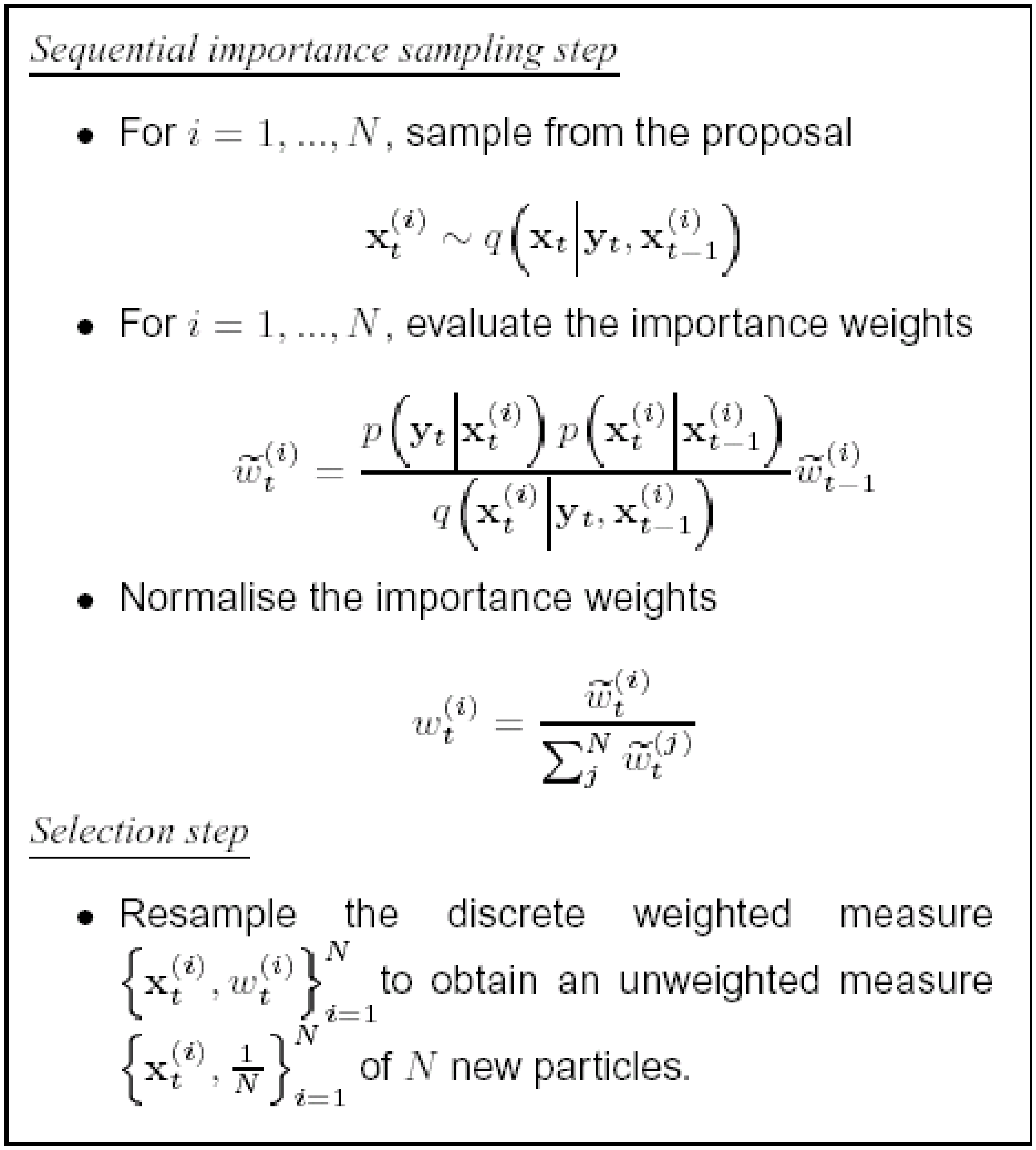
$$\pi_n(\mathbf{x}_{1:n}) = Z_n^{-1} f_n(\mathbf{x}_{1:n}) \qquad Z_n \triangleq \int f_n(\mathbf{x}_{1:n}) d\mathbf{x}_{1:n}$$

We do this using sequential importance sampling (M&U, 49)

$$w_n = \frac{f_n(\mathbf{x}_{1:n})}{q_n(\mathbf{x}_{1:n})} = \frac{f_n(\mathbf{x}_{1:n})}{f_{n-1}(\mathbf{x}_{1:n-1})} \frac{1}{q(\mathbf{x}_n | \mathbf{x}_{1:n-1})} w_{n-1}$$

e.g. For filtering, we use: $f_n(\mathbf{x}_{1:n}) = \prod_{t=1}^{n} p(\mathbf{x}_t | \mathbf{x}_{t-1}) p(\mathbf{y}_t | \mathbf{x}_t)$

Particle Filtering



Sequential importance sampling step

- For $i = 1, ..., N$, sample from the proposal

$$\mathbf{x}_t^{(i)} \sim q\left(\mathbf{x}_t \middle| \mathbf{y}_t, \mathbf{x}_{t-1}^{(i)}\right)$$

- For $i = 1, ..., N$, evaluate the importance weights

$$\widetilde{w}_t^{(i)} = \frac{p\left(\mathbf{y}_t \middle| \mathbf{x}_t^{(i)}\right) p\left(\mathbf{x}_t^{(i)} \middle| \mathbf{x}_{t-1}^{(i)}\right)}{q\left(\mathbf{x}_t^{(i)} \middle| \mathbf{y}_t, \mathbf{x}_{t-1}^{(i)}\right)} \widetilde{w}_{t-1}^{(i)}$$

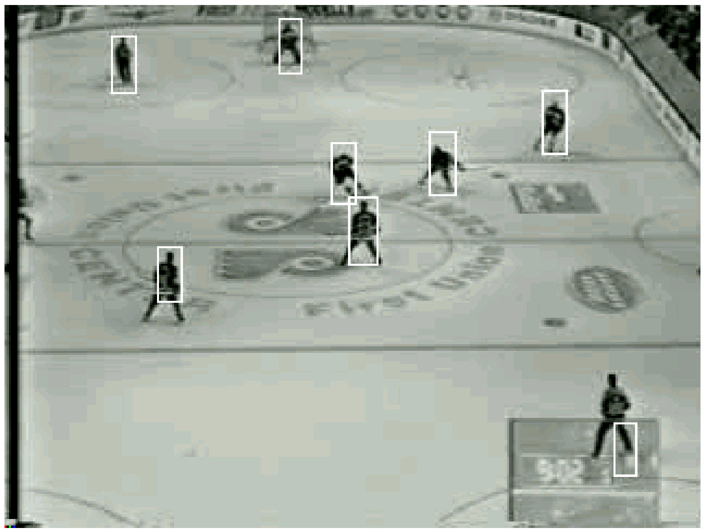
- Normalise the importance weights

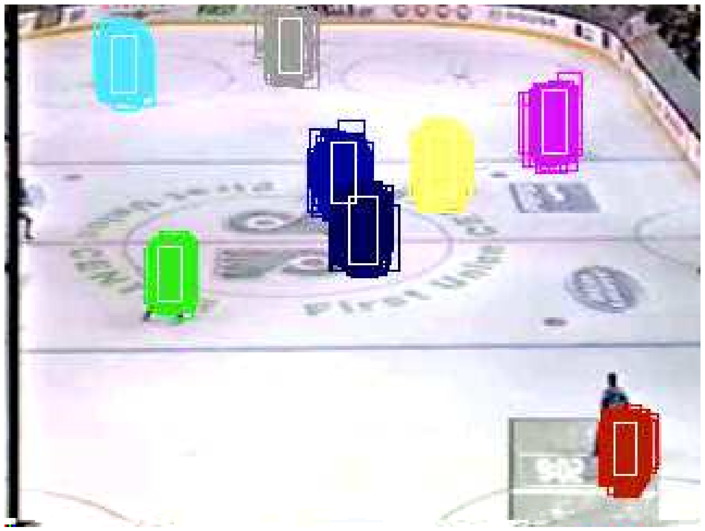
$$w_t^{(i)} = \frac{\widetilde{w}_t^{(i)}}{\sum_j^N \widetilde{w}_t^{(j)}}$$

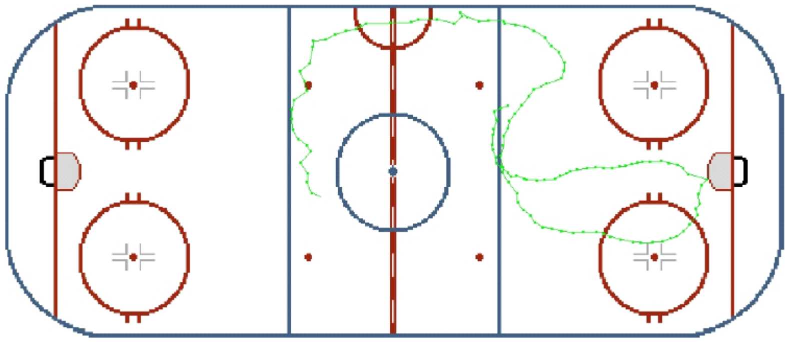
Selection step

- Resample the discrete weighted measure $\left\{\mathbf{x}_t^{(i)}, w_t^{(i)}\right\}_{i=1}^{N}$ to obtain an unweighted measure $\left\{\mathbf{x}_t^{(i)}, \frac{1}{N}\right\}_{i=1}^{N}$ of N new particles.

Using Clever Proposals: e.g. Boosting







Autonomous robots and self-diagnosis

$$\begin{tikzcd}
z_1 \arrow[r] \arrow[d] & z_2 \arrow[r] \arrow[d] & z_3 \arrow[d] \\
X_1 \arrow[r] \arrow[d] & X_2 \arrow[r] \arrow[d] & X_3 \arrow[d] \\
Y_1 & Y_2 & Y_3
\end{tikzcd}$$

Unknown internal discrete state

Unknown continuous signals

Sensor readings

Rao-Blackwellised Particle Filtering

- Robot gathers **observations** $y_t \in \mathbb{R}^{n_y}$ one-at-a-time using internal and external sensors.
- Robot has internal **continuous states** $x_t \in \mathbb{R}^{n_x}$.
- Robot has internal **discrete states** $z_t \in \mathcal{Z} = \{1, \ldots, n_z\}$ (*e.g.* “`stuck rear wheel`”, “`walking`”, “`damaged camera`”, “`spotting alliens`”).
- **Goal**: Obtain a recursive estimate of $p(x_{0:t}, z_{0:t} | y_{1:t})$ from which we can derive $p(z_t | y_{1:t})$, where $x_{0:t} \triangleq \{x_0, x_1, \ldots, x_t\}$.

Rao-Blackwellised Particle Filtering

$$\begin{aligned} z_t &\sim P(z_t|z_{t-1}) \\ x_t &= A(z_t)x_{t-1} + B(z_t)w_t + F(z_t)u_t \\ y_t &= C(z_t)x_t + D(z_t)v_t + G(z_t)u_t \end{aligned}$$

- $u_t \in \mathcal{U}$ is a known control signal.
- i.i.d noise processes: $w_t \sim \mathcal{N}(0, I)$ and $v_t \sim \mathcal{N}(0, I)$.
- The parameters $(A, B, C, D, E, F, P(z_t|z_{t-1}))$ are known matrices; see [Andrieu, de Freitas, Doucet, 1999] for parameter estimation and model selection.
- Initial states: $x_0 \sim \mathcal{N}(\mu_0, \Sigma_0)$ and $z_0 \sim P(z_0)$.

Naïve solution with PF

- For $i = 1, \ldots, N$, sample from the transition priors

$$\tilde{z}_t^{(i)} \sim \Pr(z_t | z_{t-1}^{(i)}) \qquad \tilde{x}_t^{(i)} \sim p(x_t | x_{t-1}^{(i)}, \tilde{z}_t^{(i)})$$

- For $i = 1, \ldots, N$, evaluate and normalize the weights

$$\tilde{w}_t^{(i)} \propto p\left(y_t \middle| \tilde{x}_t^{(i)}, \tilde{z}_t^{(i)}\right)$$

- Select fittest particles.

RBPF: The conditioning argument

- Exploit the following factorisation

$$p\left(x_{0:t}, z_{0:t} \middle| y_{1:t}\right) = p\left(x_{0:t} \middle| y_{1:t}, z_{0:t}\right) p\left(z_{0:t} \middle| y_{1:t}\right)$$

- The density $p\left(x_{0:t} \middle| y_{1:t}, z_{0:t}\right)$ is Gaussian and can be computed analytically if we know the marginal posterior density $p\left(z_{0:t} \middle| y_{1:t}\right)$.

- This marginal density satisfies the alternative recursion

$$p\left(z_{0:t} | y_{1:t}\right) = p\left(z_{0:t-1} | y_{1:t-1}\right) \frac{p\left(y_t | y_{1:t-1}, z_{0:t}\right) p\left(z_t | z_{t-1}\right)}{p\left(y_t | y_{1:t-1}\right)}$$

RBPF: Do it Analytically if you Can!

Given $\{z_{0:t}^{(i)}, w_t^{(i)}\}_{i=1}^{N}$, we have

$$\widehat{P}_N(z_{0:t}|y_{1:t}) = \sum_{i=1}^{N} w_t^{(i)} \delta_{z_{0:t}^{(i)}}(z_{0:t})$$

and the marginal density of $x_{0:t}$ is a Gaussian mixture:

$$\widehat{p}_N(x_{0:t}|y_{1:t}) = \sum_{\mathcal{Z}^{t+1}} p(x_{0:t}|z_{0:t}, y_{1:t}) \widehat{P}_N(z_{0:t}|y_{1:t})$$

$$= \sum_{i=1}^{N} w_t^{(i)} p(x_{0:t}|y_{1:t}, z_{0:t}^{(i)})$$

that can be computed efficiently with a bank of Kalman filters

RBPF: Do it Analytically if you Can!

We sample $z_t^{(i)}$ and then propagate the mean $\mu_t^{(i)}$ and covariance $\Sigma_t^{(i)}$ of x_t with a Kalman filter

$$
\begin{aligned}
\mu_{t|t-1}^{(i)} &= A(z_t^{(i)})\mu_{t-1|t-1}^{(i)} + F(z_t^{(i)})u_t \\
\Sigma_{t|t-1}^{(i)} &= A(z_t^{(i)})\Sigma_{t-1|t-1}^{(i)}A(z_t^{(i)})^{\mathrm{T}} + B(z_t^{(i)})B(z_t^{(i)})^{\mathrm{T}} \\
S_t^{(i)} &= C(z_t^{(i)})\Sigma_{t|t-1}^{(i)}C(z_t^{(i)})^{\mathrm{T}} + D(z_t^{(i)})D(z_t^{(i)})^{\mathrm{T}} \\
y_{t|t-1}^{(i)} &= C(z_t^{(i)})\mu_{t|t-1}^{(i)} + G(z_t^{(i)})u_t \\
\mu_{t|t}^{(i)} &= \mu_{t|t-1}^{(i)} + \Sigma_{t|t-1}^{(i)}C(z_t^{(i)})^{\mathrm{T}}S_t^{-1(i)}(y_t - y_{t|t-1}^{(i)}) \\
\Sigma_{t|t}^{(i)} &= \Sigma_{t|t-1}^{(i)} - \Sigma_{t|t-1}^{(i)}C(z_t^{(i)})^{\mathrm{T}}S_t^{-1(i)}C(z_t^{(i)})\Sigma_{t|t-1}^{(i)},
\end{aligned}
$$

and with $\widehat{z}_t^{(i)} \sim \Pr(z_t|z_{t-1}^{(i)})$, we have

$$w_t = p\left(y_t|\, y_{1:t-1}, z_{0:t}^{(i)}\right) = \mathcal{N}\left(y_{t|t-1}^{(i)}, S_t^{(i)}\right)$$

RBPF Algorithm

- For $i = 1, \ldots, N$ sample $\widetilde{z}_t^{(i)} \sim \Pr(z_t | z_{t-1}^{(i)})$

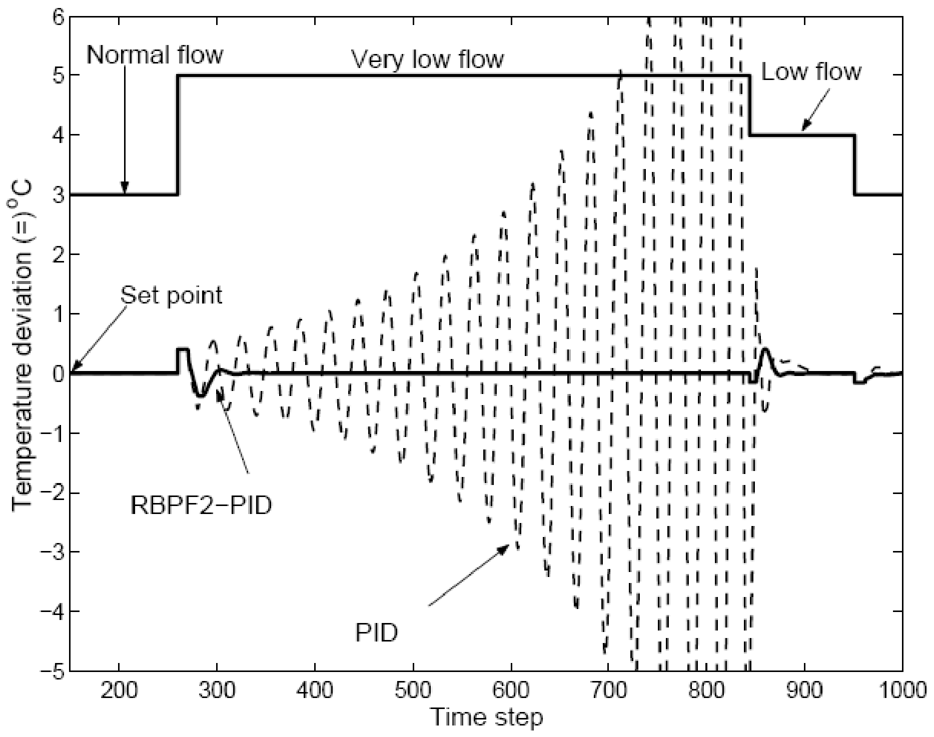
- For $i = 1, \ldots, N$, evaluate and normalize the weights

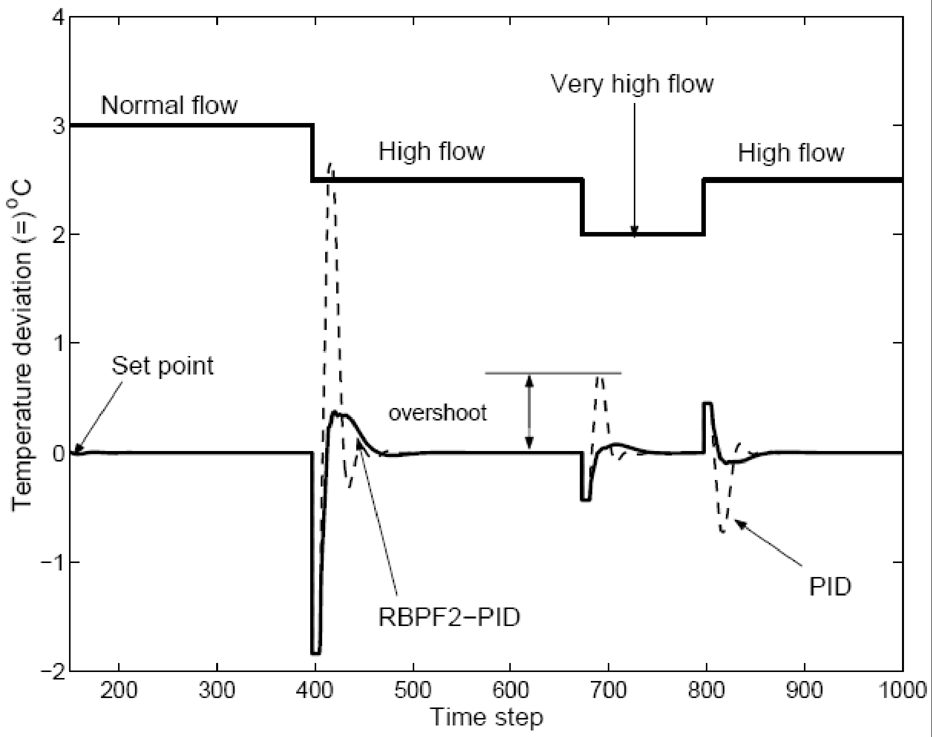
$$\widetilde{w}_t^{(i)} \propto p\left(y_t | y_{1:t-1}, \widetilde{z}_t^{(i)}\right)$$

- Select fittest particles.

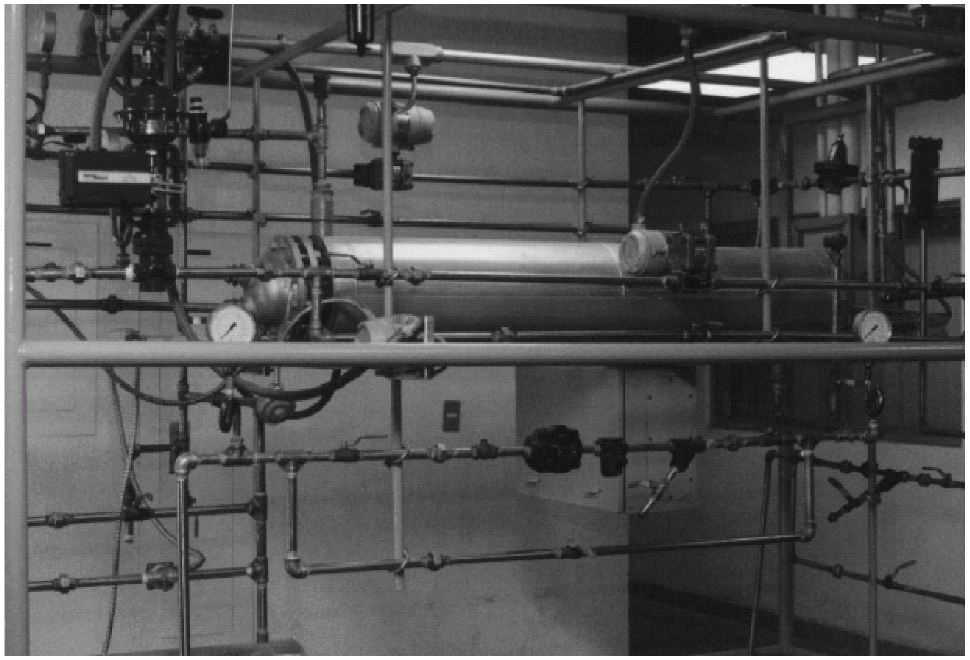
- For $i = 1, \ldots, N$, use one step of the Kalman recursion to compute the minimum statistics $\left\{\mu_{t+1|t}^{(i)}, \Sigma_{t+1|t}^{(i)}, y_{t+1|t}^{(i)}, S_{t+1}^{(i)}\right\}$ given $\left\{z_t^{(i)}, \mu_{t|t-1}^{(i)}, \Sigma_{t|t-1}^{(i)}\right\}$.

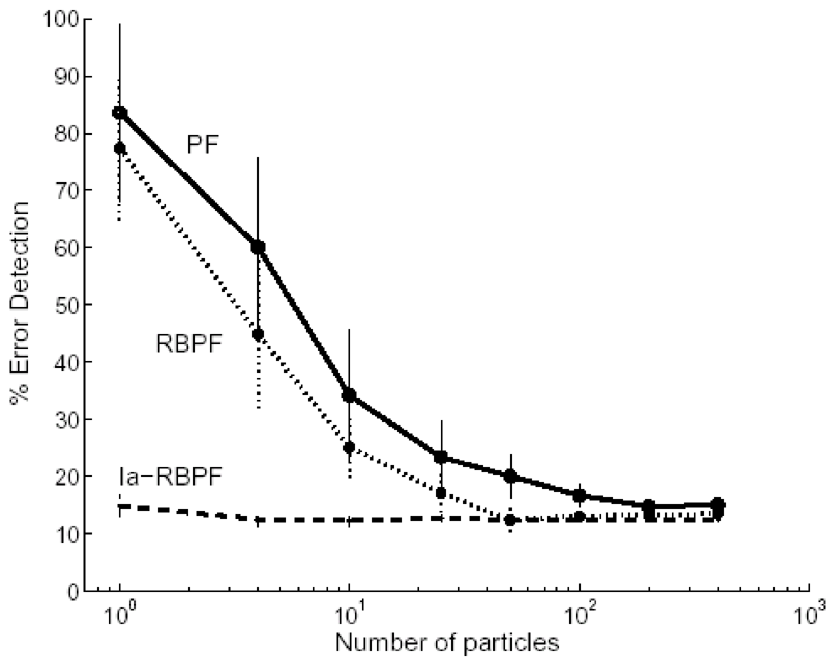
RBPF for Hybrid Control with PIDs





RBPF Real-time diagnosis





$$\begin{tikzcd}
z_0 \arrow[r] \arrow[d] & z_1 \arrow[r] \arrow[d] \arrow[dd, bend left] & z_2 \arrow[r] \arrow[d] \arrow[dd, bend left] & z_3 \arrow[r] \arrow[d] \arrow[dd, bend left] & {} \\
x_0 \arrow[r] & x_1 \arrow[r] \arrow[d] & x_2 \arrow[r] \arrow[d] & x_3 \arrow[r] \arrow[d] & {} \\
 & y_1 \arrow[r] & y_2 \arrow[r] & y_3 \arrow[r] & {}
\end{tikzcd}$$

Given samples of z, we can solve for x exactly with a mixture of Kalman filters.

Beyond Filtering

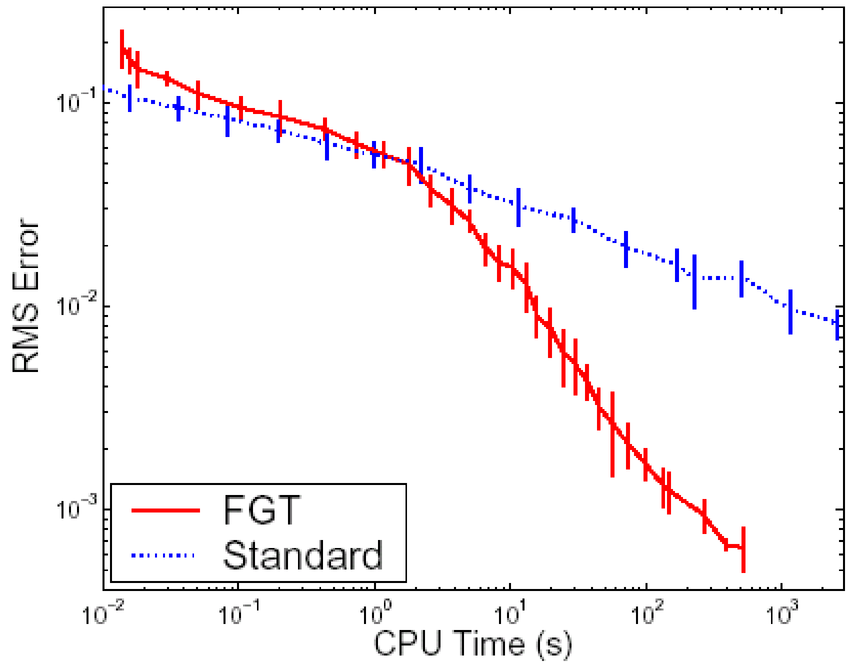
Filtering: $p\left(x_t \middle| y_{1:t}\right)$

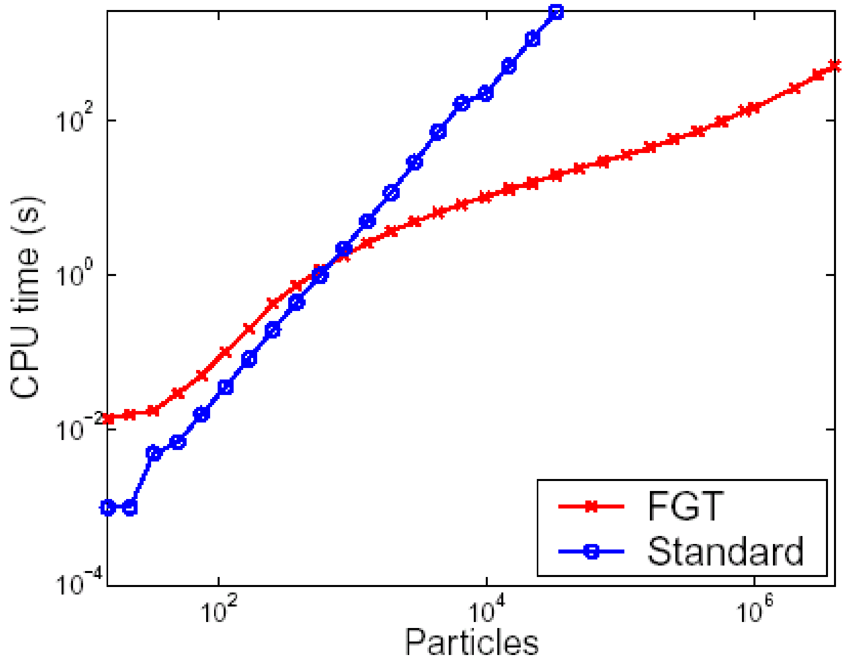
Smoothing: $p\left(x_t \middle| y_{1:T}\right)$

Viterbi: $\arg\max_{x_{1:T}} p\left(x_{1:T} \middle| y_{1:T}\right)$

Filtering is O(N), but smoothing and Viterbi are O(N ^{2})

Solution: **Fast multipole methods** (Greengard and Rohklin), **dual metric trees** (Gray and Moore) and **Huttenlocher's tricks** – no FFT.

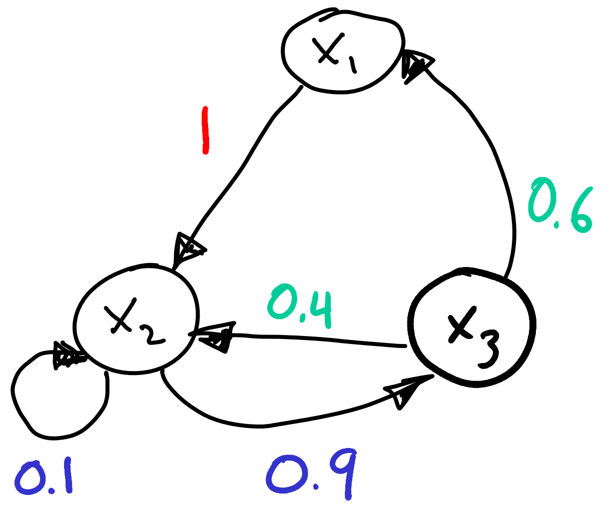




Markov Chain Monte Carlo

For simplicity, let's consider only 3 states:

$x_t \in \mathcal{X} = \{x_1, x_2, x_3\}$



$$T = p(x_t \mid x_{t-1}) = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0.1 & 0.9 \\ 0.6 & 0.4 & 0 \end{bmatrix}$$

Think of this as a webgraph. Our goal is to crawl it to find the "relevance" of each node.

Markov Chain Monte Carlo

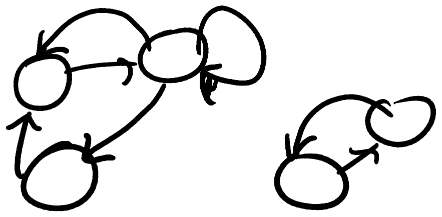
T is a stochastic matrix. As long as the graph (state space) is aperiodic and irreducible, we have that for any initial vector of probabilities ν:

$$\nu' T^t \longrightarrow \pi' \qquad \text{as} \quad t \to \infty$$

where π is the invariant or stationary distribution of the chain. It is unique.

Markov Chain Monte Carlo

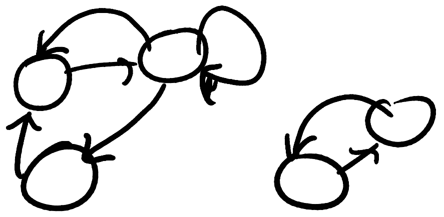
Need for irreducibility:



One cluster might never be visited!

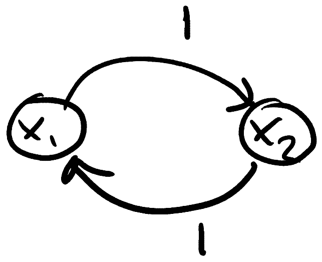
Markov Chain Monte Carlo

Need for irreducibility:



One cluster might never be visited!

Need for aperiodicity:



$T = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$

Let $\pi = \begin{bmatrix} 1/3 & 2/3 \end{bmatrix}$

$\pi' T = \begin{bmatrix} 2/3 & 1/3 \end{bmatrix}$

$\pi' T^2 = \begin{bmatrix} 1/3 & 2/3 \end{bmatrix}$

$\vdots$

Oscillation!

Markov Chain Monte Carlo

In the limit:

$$\pi' T = \pi'$$

π is the left eigenvector of T with corresponding eigenvalue 1.

Markov Chain Monte Carlo

In the limit:

$$\pi' T = \pi'$$

π is the left eigenvector of T with corresponding eigenvalue 1. Componentwise, we have:

$$\sum_{i=1}^{3} \pi_i T_{ij} = \pi_j$$

Markov Chain Monte Carlo

In the limit:

$$\pi' T = \pi'$$

π is the left eigenvector of T with corresponding eigenvalue 1. Componentwise, we have:

$$\sum_{i=1}^{3} \pi_i T_{ij} = \pi_j$$

As the state space grows:

$$\int \pi(x) \underbrace{p(y|x)}_{\text{Markov Chain Kernel}} dx = \pi(y)$$

Markov Chain Monte Carlo

Detailed Balance:

If $$\pi(x_t)\,P(x_{t+1}|x_t) = \pi(x_{t+1})\,P(x_t|x_{t+1})$$

Integrating over x_t yields

$$\int \pi(x_t)\,P(x_{t+1}|x_t) = \pi(x_{t+1})$$

Which is the ergodic behaviour we want.
Now we have a sufficient condition for designing $P(x_{t+1}|x_t)$ so as to get samples from π

MCMC: Metropolis-Hastings

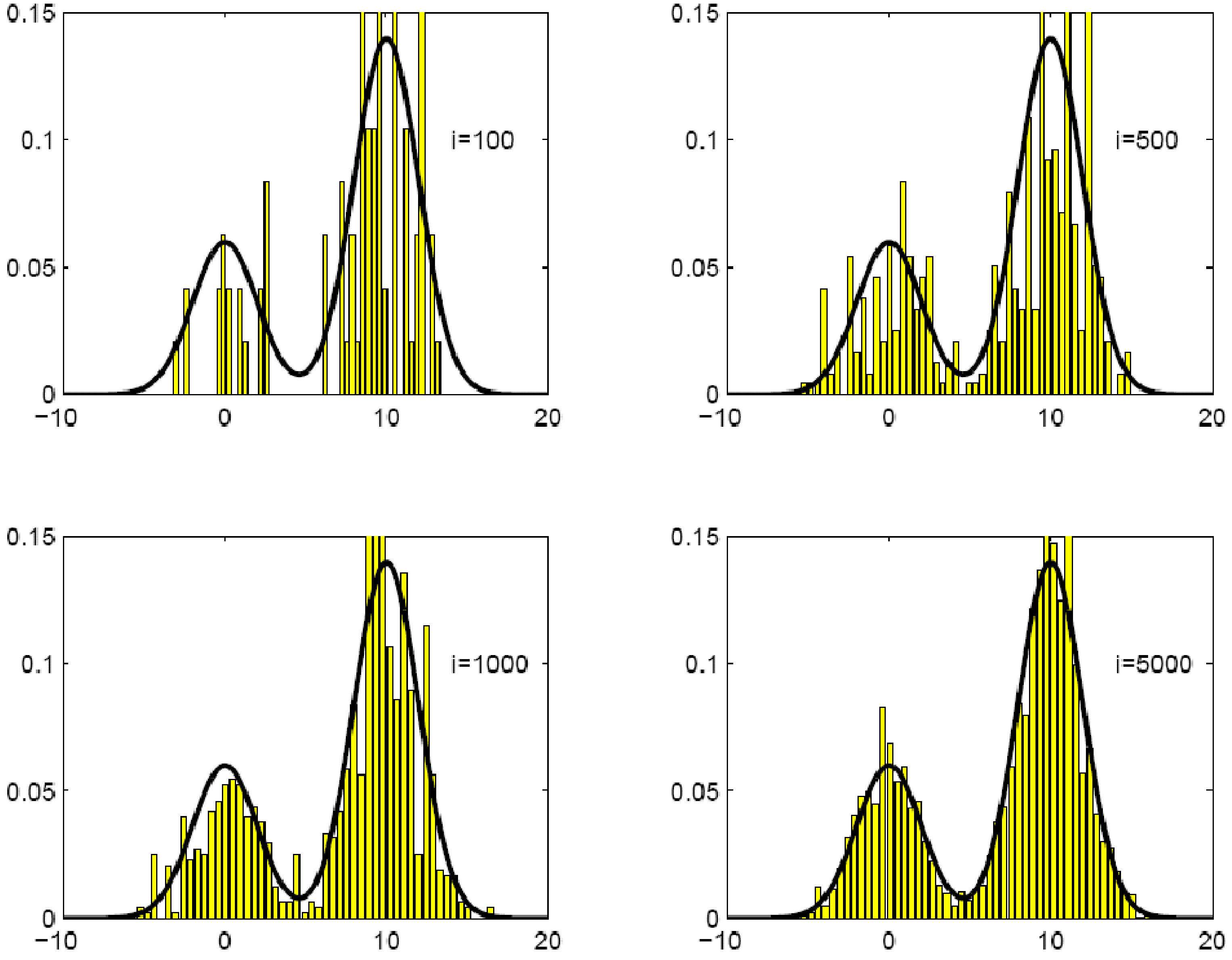
- Initialise $x^{(0)}$.
- For $i = 0$ to $N-1$
    - Sample $u \sim U_{[0,1]}$.
    - Sample $x^\star \sim q(x^\star | x^{(i)})$.
    - If $u < A(x^{(i)}, x^\star) = \min\left\{1, \frac{p(x^\star) q(x^{(i)} | x^\star)}{p(x^{(i)}) q(x^\star | x^{(i)})}\right\}$

      $$x^{(i+1)} = x^\star$$

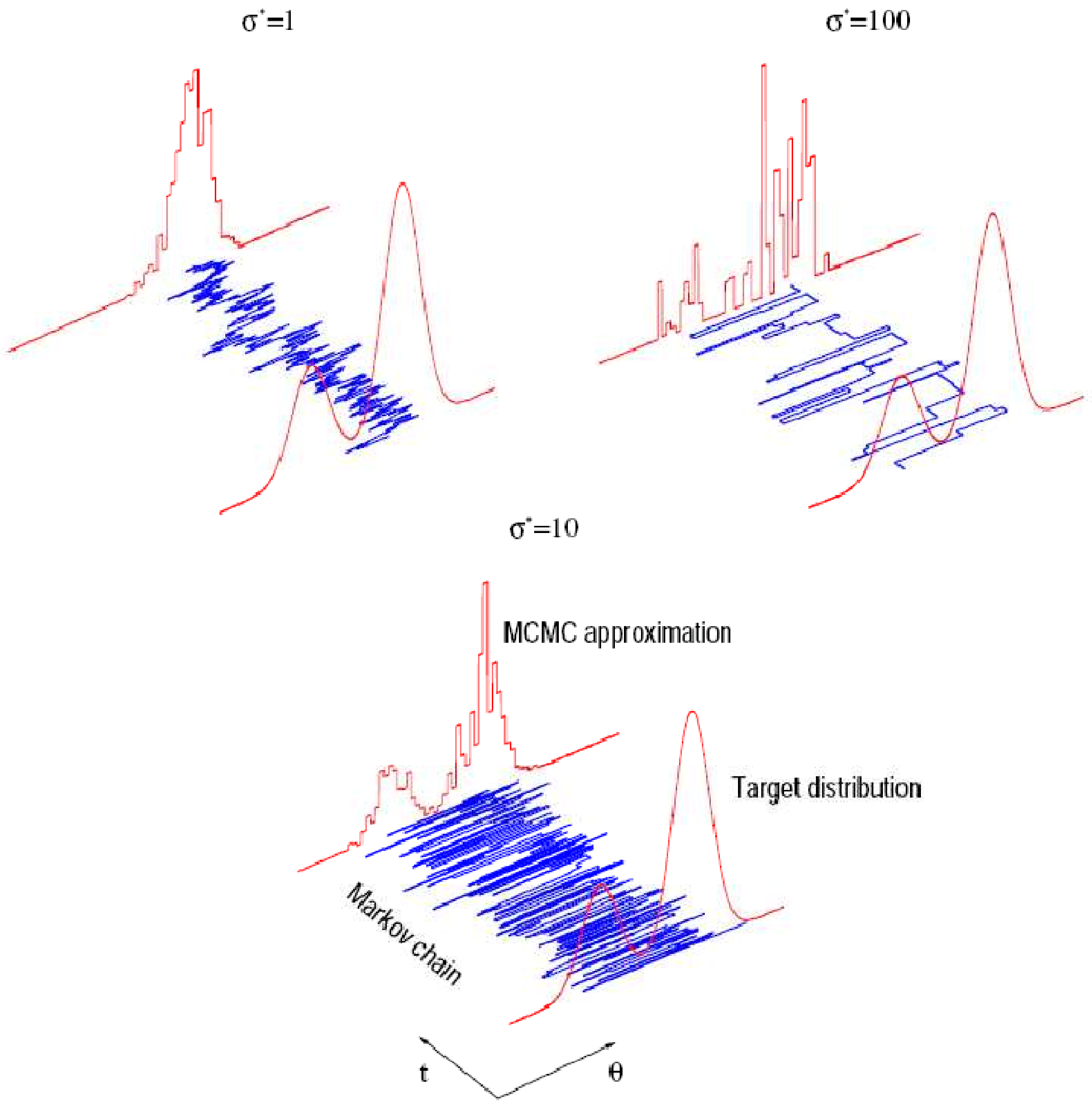
      else

      $$x^{(i+1)} = x^{(i)}$$

MCMC: Metropolis-Hastings



MCMC: Choosing the Right Proposal



MCMC: Theory

Kernel:

$$K(x,B) = \begin{cases} q(B|x)\,A(x,B) & x \notin B \\ 1 - \displaystyle\int_{x' \in \{\mathcal{X} \setminus B\}} q(x'|x)\,A(x,x') & x \in B \end{cases}$$

$$\therefore\; K(x,B) = q(B|x)\,A(x,B) + \mathbb{I}_{x \in B} \left\{ \begin{array}{l} 1 - q(B|x)\,A(x,B) \\ - \displaystyle\int_{x' \in \{\mathcal{X} \setminus B\}} q(x'|x)\,A(x,x') \end{array} \right\}$$

$$K(x,B) = q(B|x)\,A(x,B) + \mathbb{I}_{x \in B} \left\{ 1 - \int_{x' \in \mathcal{X}} q(x'|x)\,A(x,x') \right\}$$

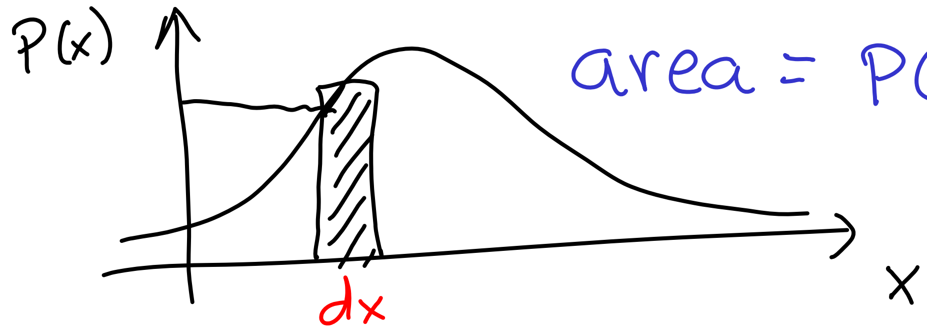
MCMC: Theory

Detailed balance :

$$\pi(A)K(A,B) = \pi(B)K(B,A)$$

$$\int_{x\in A} \pi(dx)\,K(x,B) = \int_{y\in B} \pi(dy)\,K(y,A)$$

Note: $\int f(x)\,p(x)\,dx \equiv \int f(x)\,p(dx)$



area $= P(dx) = P(x)\,dx$

MCMC: MH Annealed

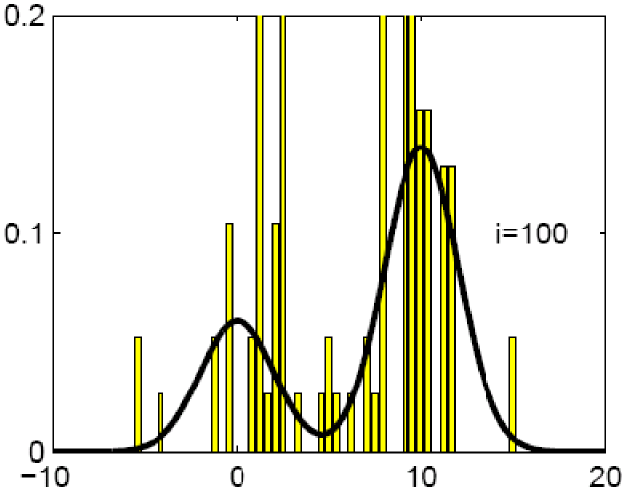
- Initialise $x^{(0)}$ and set $T_0 = 1$.
- For $i = 0$ to $N - 1$
  - Sample $u \sim U_{[0,1]}$.
  - Sample $x^\star \sim q(x^\star | x^{(i)})$.
  - If $u < A(x^{(i)}, x^\star) = \min\left\{1, \dfrac{p^{\frac{1}{T_i}}(x^\star) q(x^{(i)} | x^\star)}{p^{\frac{1}{T_i}}(x^{(i)}) q(x^\star | x^{(i)})}\right\}$

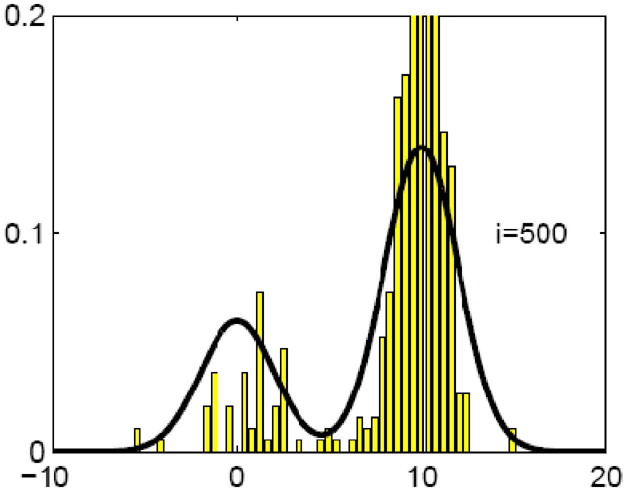
    $$x^{(i+1)} = x^\star$$

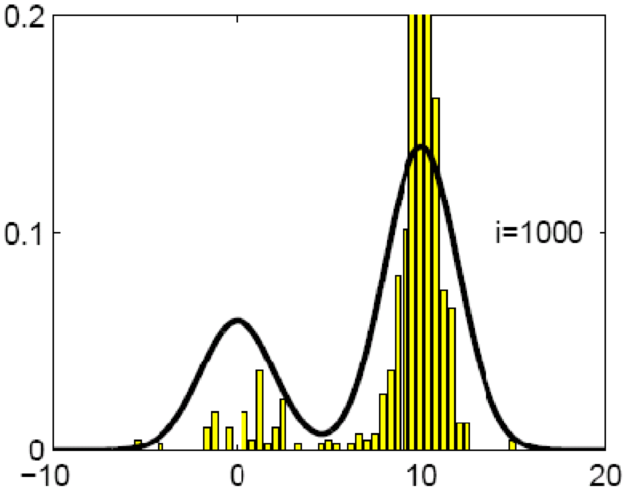
    else

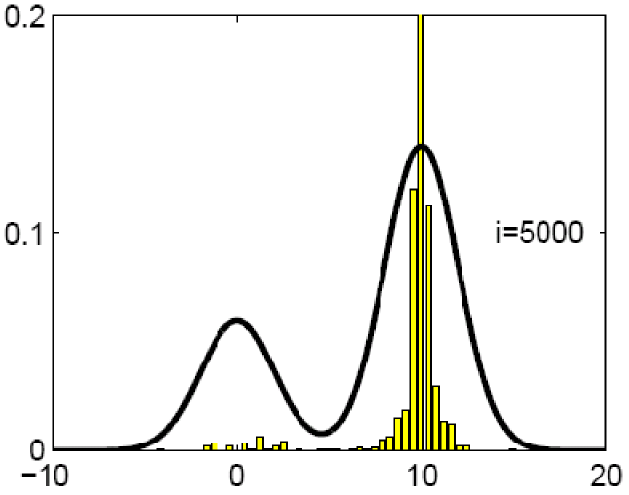
    $$x^{(i+1)} = x^{(i)}$$
  - Set T_{i+1} according to a chosen cooling schedule.

MCMC: MH Annealed

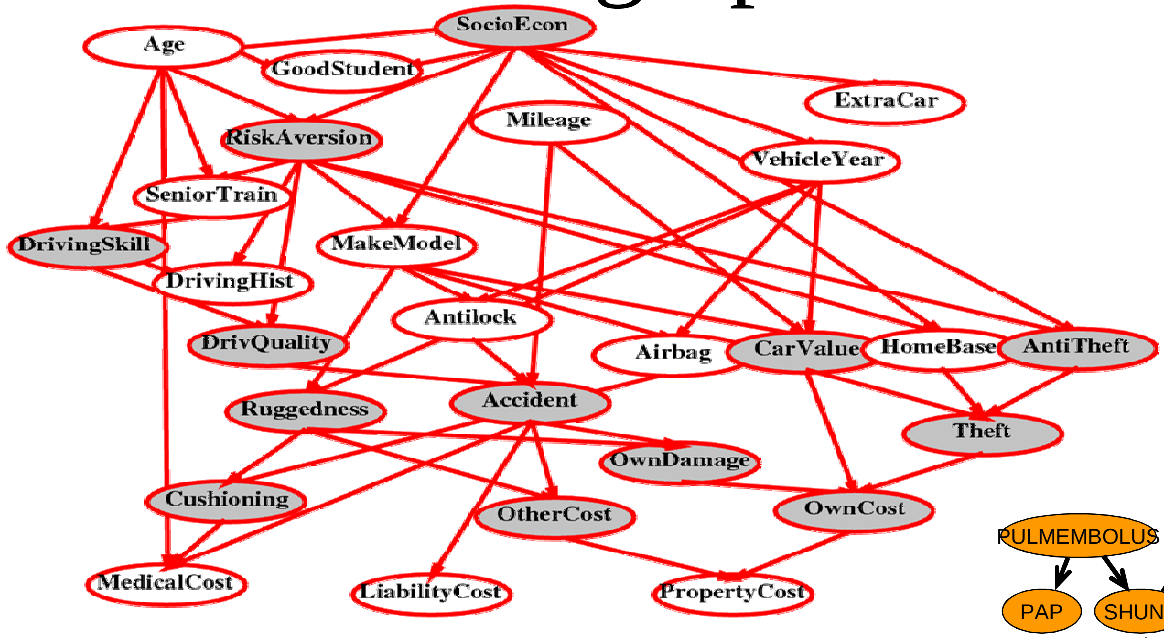


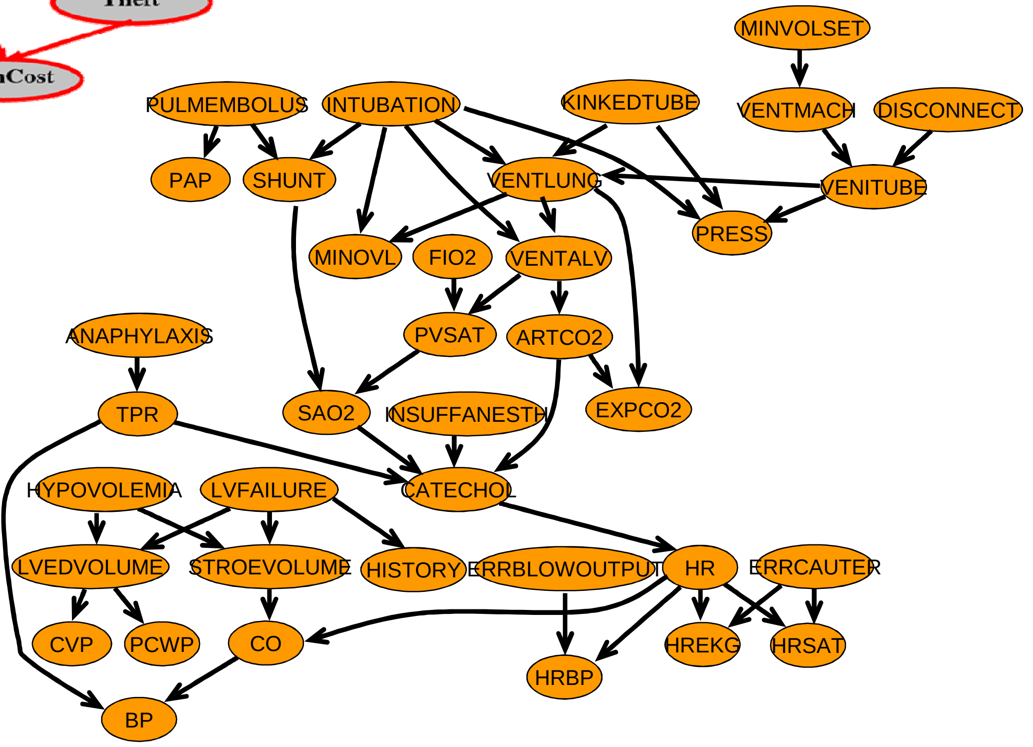






Extending MH to directed probabilistic graphical models





Gibbs Sampling

Choose the following proposal:

$$q(x^\star|x^{(i)}) = \begin{cases} p(x_j^\star|x_{-j}^{(i)}) & \text{If } x_{-j}^\star = x_{-j}^{(i)} \\ 0 & \text{Otherwise.} \end{cases}$$

where $x_{-j} = \{x_1, \ldots, x_{j-1}, x_{j+1}, \ldots, x_n\}$.

Then the acceptance is:

$$A(x^{(i)}, x^\star) \quad = \quad \min\left\{1, \frac{p(x^\star)q(x^{(i)}|x^\star)}{p(x^{(i)})q(x^\star|x^{(i)})}\right\} = 1.$$

Gibbs Sampling

- Initialise $x_{1:n}^{(0)}$.
- For $i = 0$ to $N - 1$
  - Sample $x_1^{(i+1)} \sim p(x_1 | x_2^{(i)}, x_3^{(i)}, \ldots, x_n^{(i)})$.
  - Sample $x_2^{(i+1)} \sim p(x_2 | x_1^{(i+1)}, x_3^{(i)}, \ldots, x_n^{(i)})$.

$$\vdots$$

  - Sample $x_j^{(i+1)} \sim p(x_j | x_1^{(i+1)}, \ldots, x_{j-1}^{(i+1)}, x_{j+1}^{(i)}, \ldots, x_n^{(i)})$.

$$\vdots$$

  - Sample $x_n^{(i+1)} \sim p(x_n | x_1^{(i+1)}, x_2^{(i+1)}, \ldots x_{n-1}^{(i+1)})$.

Gibbs Sampling For Graphical models

A large-dimensional joint distribution is factored into a directed graph that encodes the conditional independencies in the model. In particular, if $x_{pa(j)}$ denotes the parent nodes of node x_j, we have

$$p(x) = \prod_j p(x_j | x_{pa(j)}).$$

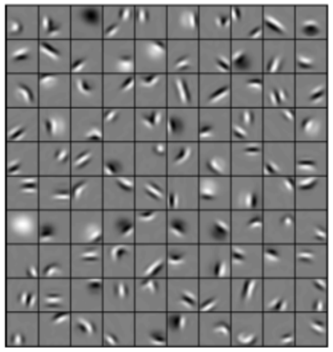
It follows that the full conditionals simplify as follows

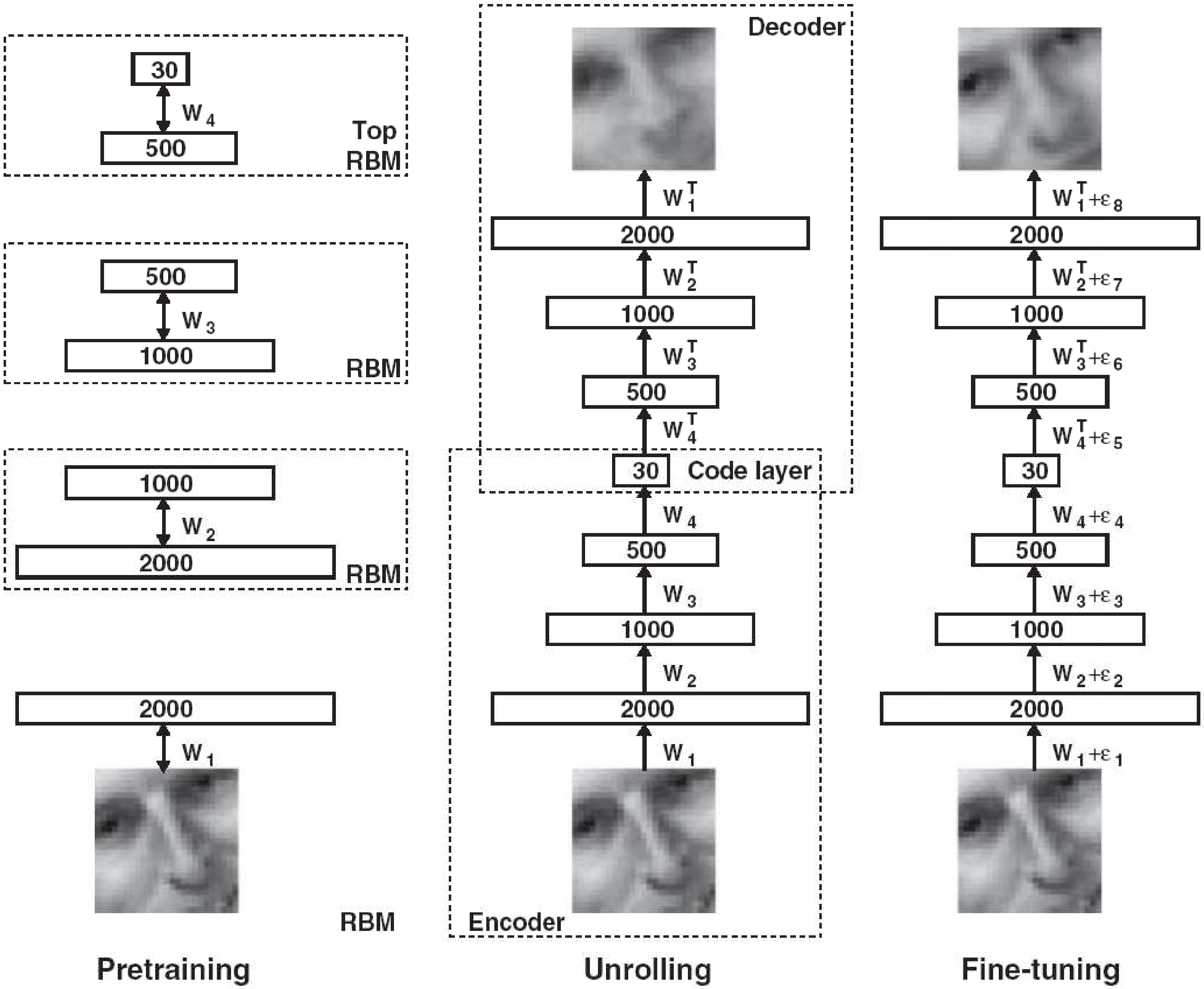
$$p(x_j | x_{-j}) = p(x_j | x_{pa(j)}) \prod_{k \in ch(j)} p(x_k | x_{pa(k)})$$

where $ch(j)$ denotes the children nodes of x_j.

☜

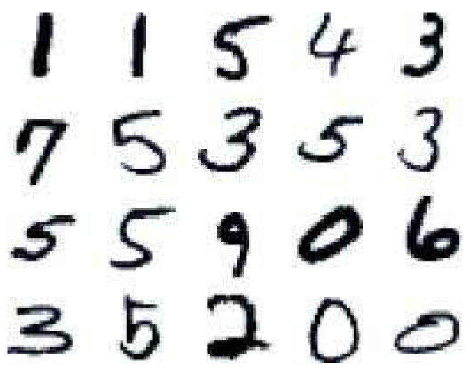
Deep learning (Hinton and collaborators)

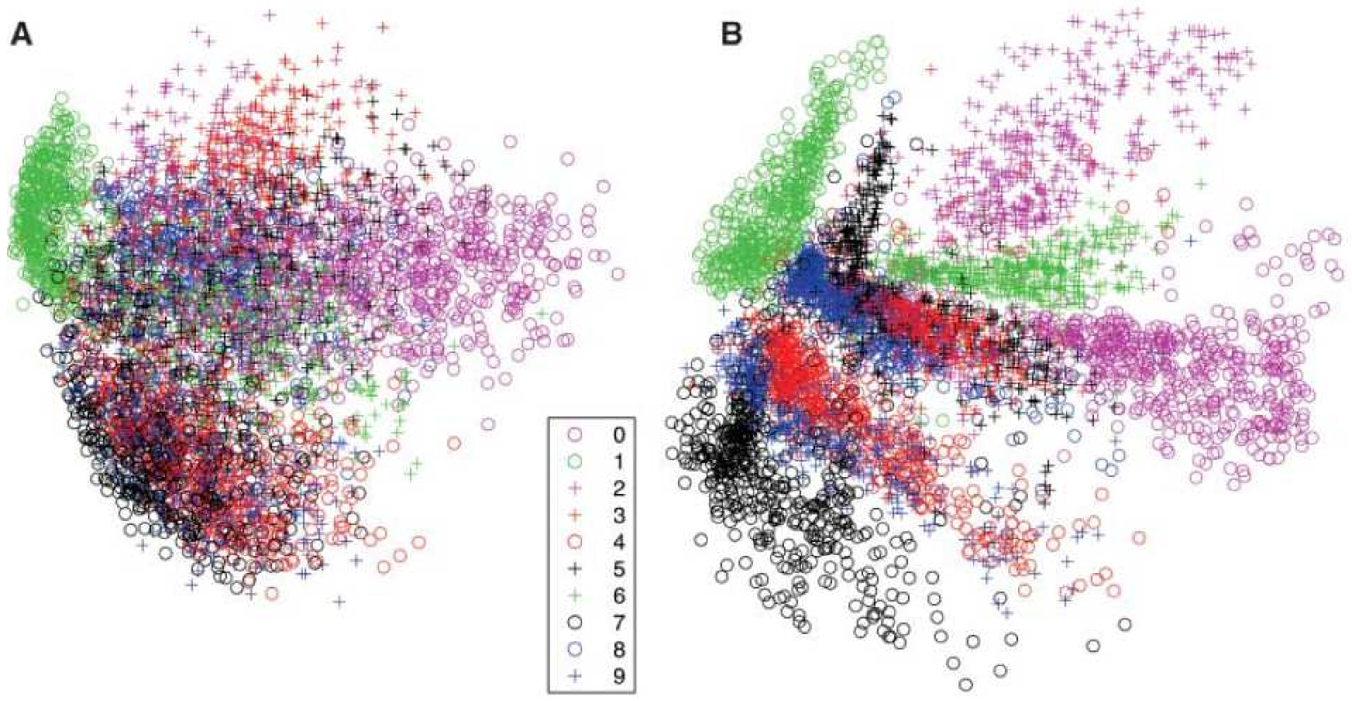




Encoding digits

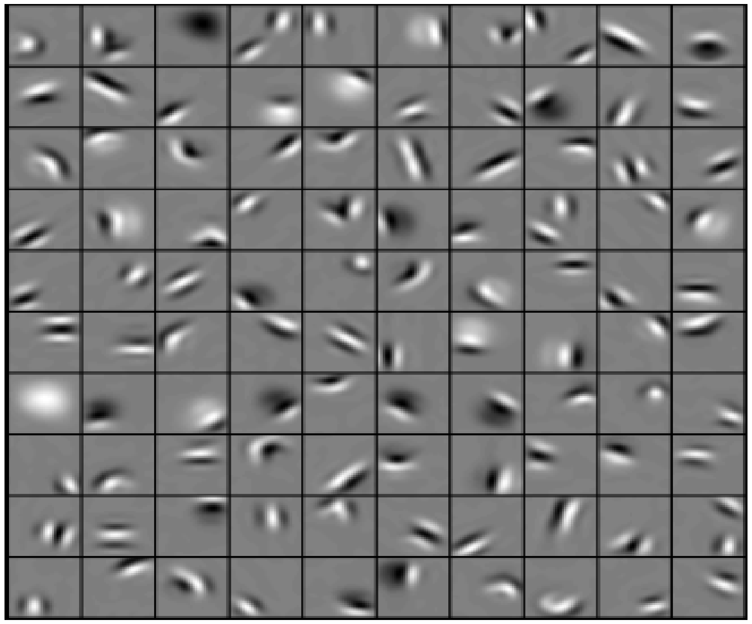
(A) The two-dimensional codes for 500 digits of each class produced by taking the first two principal components of all 60,000 training images.
(B) The two-dimensional codes found by a 784-1000-500-250-2 autoencoder.



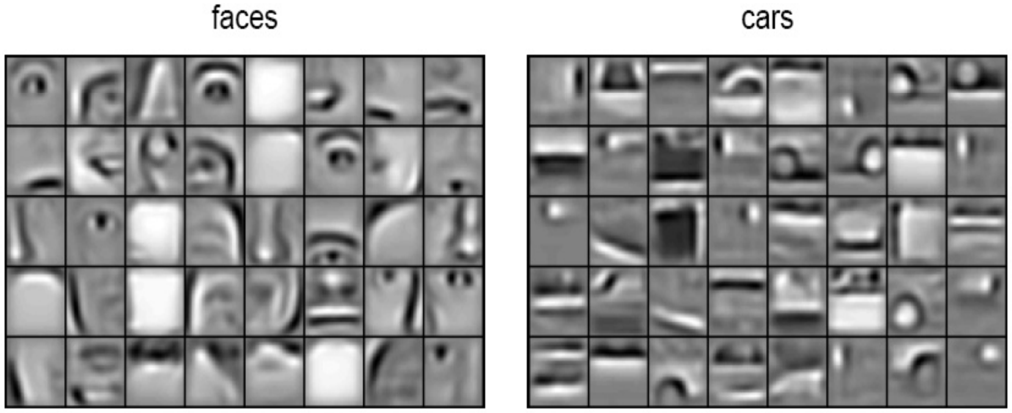


These 2-dimensional embeddings of images of digits enable us to make predictions (classification)

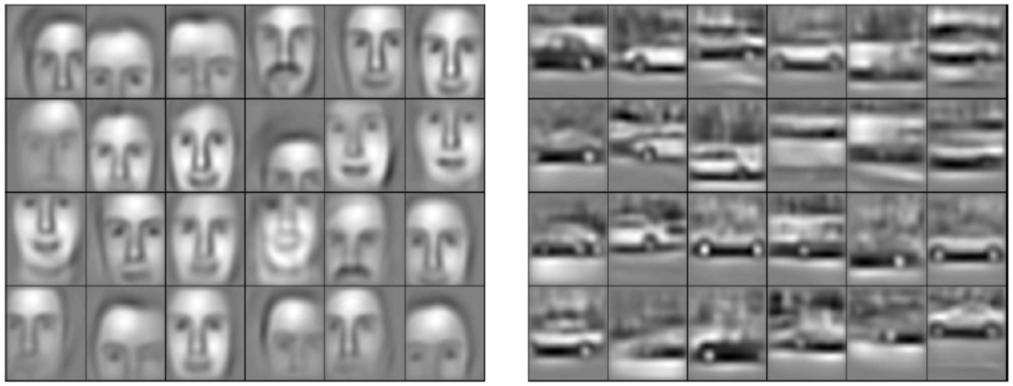
Layer 1



Layer 2



Layer 3



[Honglak Lee et al 2009]

In the binary case where $v \in \{0,1\}^D$ and $h \in \{0,1\}^K$ the energy function can be expressed as:

$$E(v, h, W) = -\sum_{i=1}^{D}\sum_{j=1}^{K} v_i W_{ij} h_j - \sum_{i=1}^{D} v_i b_i - \sum_{j=1}^{K} h_j b_j.$$

The probabilities of each node can be easily obtained.

$$p(v_i = 1 | h, W) = sigmoid\left(\sum_{j=1}^{K} W_{ij} h_j + b_i\right)$$

$$p(h_j = 1 | v, W) = sigmoid\left(\sum_{i=1}^{D} W_{ij} v_i + b_j\right),$$

where $sigmoid(a) = \frac{1}{1+exp(-a)}$. The model is therefore easy to sample: One simply flips K coins for the hidden units and D coins for the visible units.

Contrastive divergence learning

1. Sample hidden units $\widetilde{h_n}$ from $p(h|v_n, W^{(t)})$.
2. Sample imaginary data $\widetilde{v_n}$ from $p(v|\widetilde{h_n}, W^{(t)})$.
3. Sample hidden units again $\widetilde{\widetilde{h_n}}$ from $p(h|\widetilde{v_n}, W^{(t)})$.
4. Update the parameters:

$$W_{dk}^{(t+1)} = W_{dk}^{(t)} + \eta^{(t)}\left[\overset{\text{Real data}}{\frac{1}{N}\sum_{n=1}^{N} v_{dn}\widetilde{h_{kn}}} - \overset{\text{Confabulation}}{\frac{1}{N}\sum_{n=1}^{N} \widetilde{v_{dn}}\widetilde{\widetilde{h_{kn}}}}\right]$$

5. Increase t to $t+1$ and go to step 2.

MH is a Building Block

- Mixtures of MCMC algorithms.
  - Global and local exploration.
  - Reversible jump MCMC.

$$\pi T_1 = \pi$$

$$\pi T_2 = \pi$$

$$\Downarrow$$

$$\alpha \pi T_1 + (1-\alpha)\pi T_2 = \pi$$

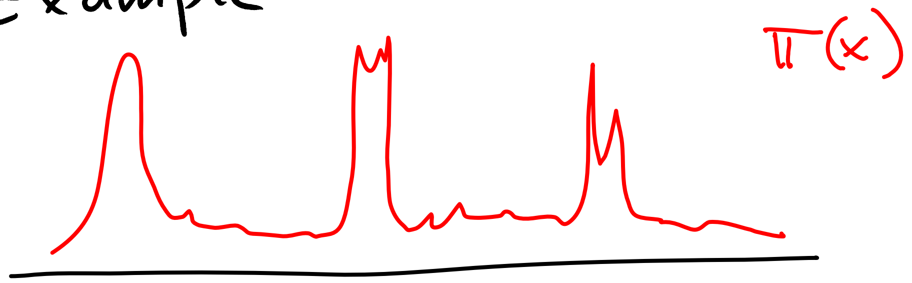
$$\pi\left[\alpha T_1 + (1-\alpha) T_2\right] = \pi$$

The mixture is also a kernel of π.

MH is a Building Block

- Mixtures of MCMC algorithms.
  - Global and local exploration.
  - Reversible jump MCMC.

Example



$$\pi T_1 = \pi$$

$$\pi T_2 = \pi$$

$$\Downarrow$$

$$\alpha \pi T_1 + (1-\alpha)\pi T_2 = \pi$$

$$\pi\left[\alpha T_1 + (1-\alpha)T_2\right] = \pi$$

The mixture is also a kernel of π.

MH is a Building Block

- Mixtures of MCMC algorithms.
  - Global and local exploration.
  - Reversible jump MCMC.

$$\pi T_1 = \pi$$

$$\pi T_2 = \pi$$

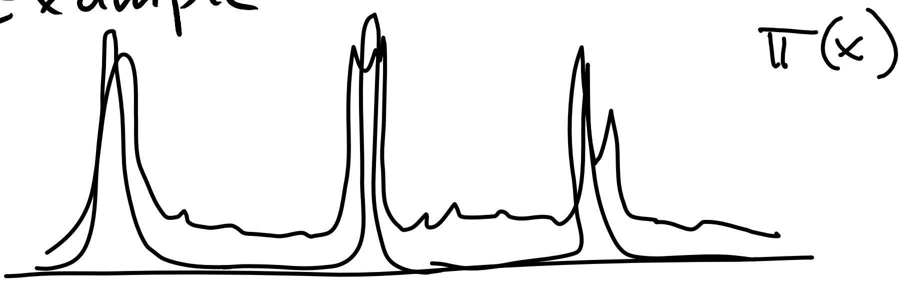
$$\Downarrow$$

$$\alpha \pi T_1 + (1-\alpha)\pi T_2 = \pi$$

$$\pi \left[\alpha T_1 + (1-\alpha) T_2 \right] = \pi$$

The mixture is also a kernel of π.

Example



$q_1(x) \equiv$ FFT proposal

$q_2(x) \equiv$ random walk to explore local modes and discover detail

MH is a Building Block

At iteration $(i + 1)$:

- Flip a coin.
- If heads

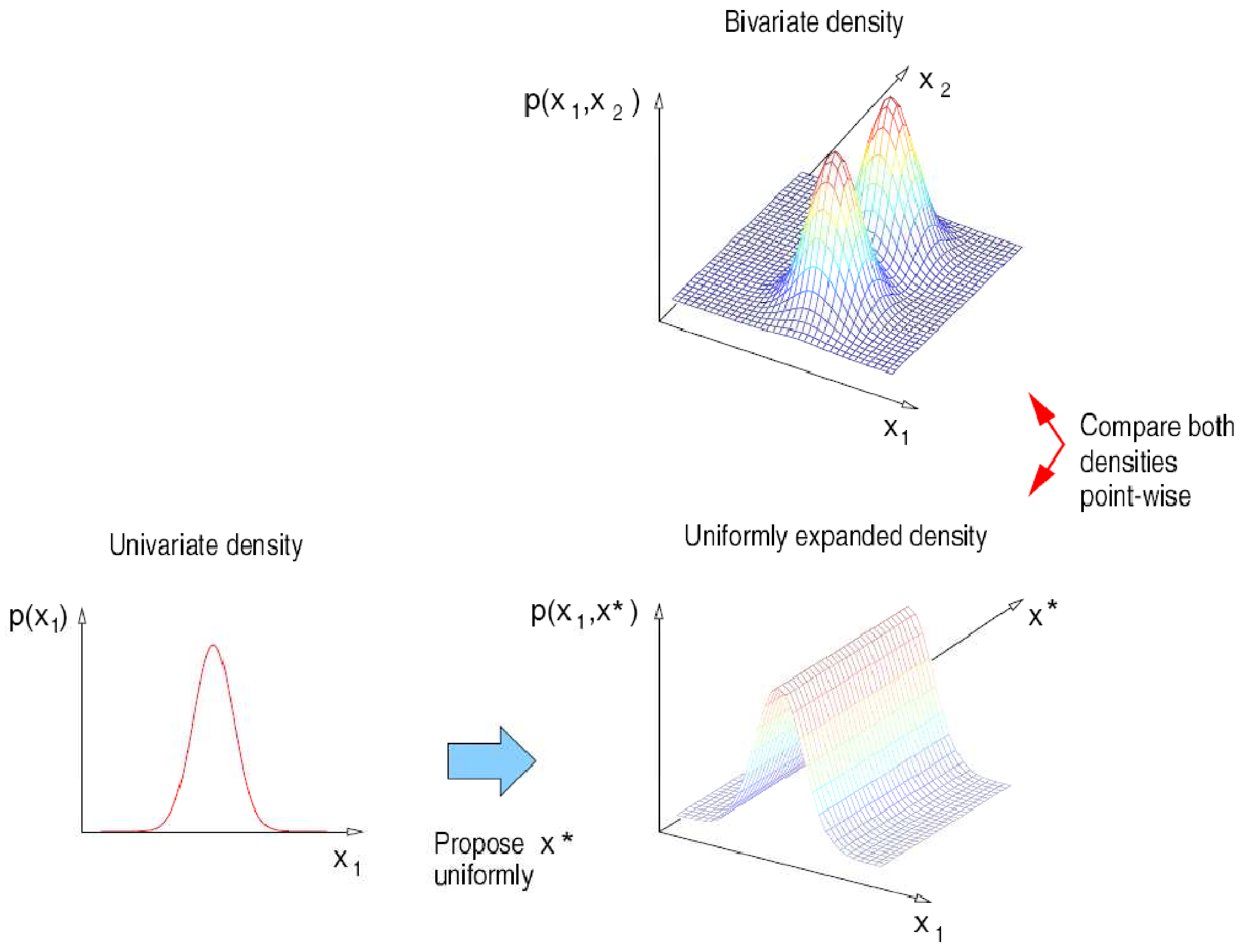
  Apply the M-H algorithm with global proposal.
- else

  Apply the M-H algorithm with a local proposal.

Trans-Dimensional MCMC and the Reversible Jump Algorithm of Peter Green: Use a mixture of dimension jumping algorithms

```
1. Initialisation: set $(k^{(0)}, \mu^{(0)})$.
2. For $i = 0$ to $N-1$
   – Sample $u \sim \mathcal{U}_{[0,1]}$.
   – If $(u \leq b_k)$
       • then "birth" move.
       • else if $(u \leq b_k + d_k)$ then "death" move.
       • else if $(u \leq b_k + d_k + s_k)$ then "split" move.
       • else if $(u \leq b_k + d_k + s_k + m_k)$ then "merge" move.
       • else update.
     End If.
   – Sample other parameters.
```

Must be Careful with Measures !



Trans-Dimensional MCMC

$$K(x,B) = \underbrace{\sum_{k}}_{\text{mixture index}} \int_B \alpha_k \, q_k(dy \mid x) + \mathbb{I}_{x \in B} \left\{ 1 - \sum_k \int_{\mathcal{X}} \alpha_k \, q_k(dx' \mid x) \right\}$$

So we have a mixture of moves (or dimensionally jumping algorithms)

Next, we need to check that this is a valid Metropolis-Hastings.

Trans-Dimensional MCMC

Since we need to compare distributions, not densities, we need a commos space.

so, if

Θ^{k_1} is completed by $u_1 \sim g_1(\cdot)$

Θ^{k_2} " " " $u_2 \sim g_2(\cdot)$

we have a $\overset{\text{deterministic}}{\text{bijection}}$ $(\Theta^{k_2}, u_2) = T(\Theta^{k_1}, u_1)$

and the acceptance term becomes:

$$A = \min\left\{1, \frac{\pi(k_2, \Theta^{k_2})}{\pi(k_1, \Theta^{k_1})} \frac{\pi_{21}}{\pi_{12}} \frac{g_2(u_2)}{g_1(u_1)} \overset{\text{Jacobian}}{\left| \frac{\partial T(\Theta^{k_1}, u_1)}{\partial(\Theta^{k_2}, u_2)} \right|}\right\}.$$

Trans-Dimensional MCMC

Example: Mixture of Gaussians with unknown number of components

$$P(x|\mu) = \sum_{i=1}^{c} \pi_i \mathcal{N}(x_j \mu_i, 1)$$

Merge Move

$$\mu = \frac{\mu_1 + \mu_2}{2}$$

Split move

$$\begin{cases} \mu_1 = \mu - uB \\ \mu_2 = \mu + uB \end{cases}$$

$u \sim N(0,1)$, $B \equiv$ parameter

For reversibility, $\|\mu_1 - \mu_2\| \leq 2B$ in merge.

Trans-Dimensional MCMC

SPLIT MOVE:

$$A_{split} = \min\left\{1, \frac{P(\mu_{k+1}, k+1)}{P(\mu_k, k)} \cdot \frac{\overbrace{\frac{1}{k+1}}^{\text{proposal to merge}} \; m_{k+1} \; 1}{\underbrace{\frac{1}{k} \; s_k \; p(u)}_{\text{proposal to split}}} \, J_{split}\right\}$$

$$J_{split} = \left| \frac{\partial(\mu_1, \mu_2)}{\partial(\mu, u)} \right| = \begin{vmatrix} \partial\mu_1/\partial\mu & \partial\mu_2/\partial\mu \\ \partial\mu_1/\partial u & \partial\mu_2/\partial u \end{vmatrix} = \begin{vmatrix} 1 & 1 \\ -\beta & \beta \end{vmatrix} = 2\beta$$

$$\begin{cases} \mu_1 = \mu - u\beta \\ \mu_2 = \mu + u\beta \end{cases}$$

Trans-Dimensional MCMC

Merge Move

$$A_{merge} = \min\left\{1, \frac{P(K-1, M_{K-1})}{P(K, M_K)} \frac{S_{K-1}}{m_K} \frac{\frac{1}{K-1}}{\frac{1}{K}} \times J_{merge}\right\}$$

$$J_{merge} = \left|\frac{\partial(m, u)}{\partial(m_1, m_2)}\right| = \begin{vmatrix} \frac{\partial m}{\partial m_1} & \frac{\partial u}{\partial m_1} \\ \frac{\partial m}{\partial m_2} & \frac{\partial u}{\partial m_2} \end{vmatrix} = \frac{1}{2\beta}$$

Trans-Dimensional MCMC

$\boxed{\text{BIRTH}}$ $\quad \mu_{1:k+1} = \{\mu_{1:k}, \mu^*\}$

$$A_{birth} = \min\left\{1, \frac{P(k+1, \mu_{k+1})}{P(k, \mu_k)} \frac{d_{k+1}}{\cancel{b}_k} \frac{\frac{1}{k+1}}{P(\mu^*)} J_{Birth}\right\}$$

$\uparrow$ new component

$$J_{Birth} = \left| \frac{\partial \mu_{1:k+1}}{\partial(\mu_{1:k}, \mu^*)} \right| = \begin{vmatrix} \frac{\partial \mu_{1:k}}{\partial \mu_{1:k}} \xrightarrow{} 1 & 0 \\ \partial \xrightarrow{} 0 & 1 \end{vmatrix} = 1$$

$\boxed{\text{DEATH}}$

This is one of the μ in $\mu_{1:k}$!

$$A_{death} = \min\left\{1, \frac{P(k-1, \mu_{k-1})}{P(k, \mu_k)} \frac{P(\mu^*)}{\frac{1}{k}} \times \frac{b_{k-1}}{d_k} \times \cancel{J}_{death}\right\}$$

$\nearrow 1$

MH is a Building Block

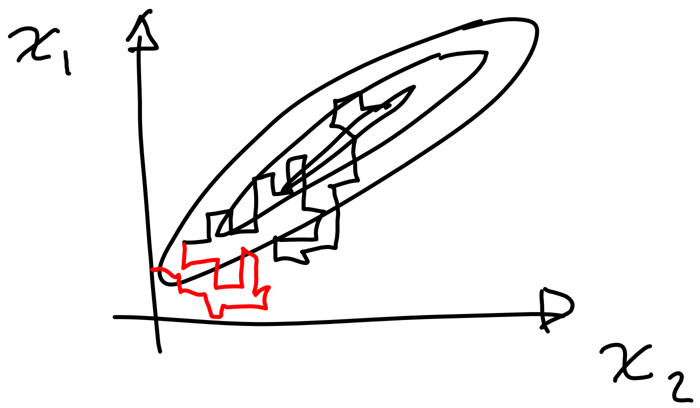
- Cycles of MCMC algorithms.
  - Large dimensional vectors.
  - Gibbs sampling.

MH is a Building Block

- **Idea**: Split the high dimensional vector x into blocks $\{x_{b1}, \ldots, x_{bn}\}$.

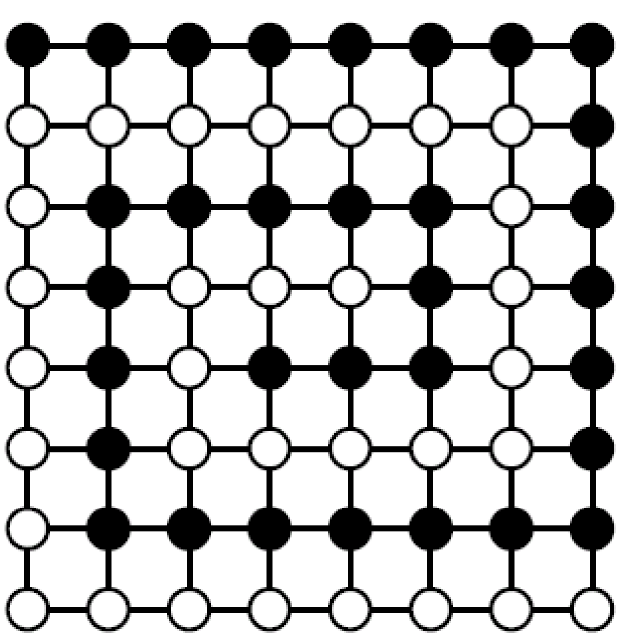
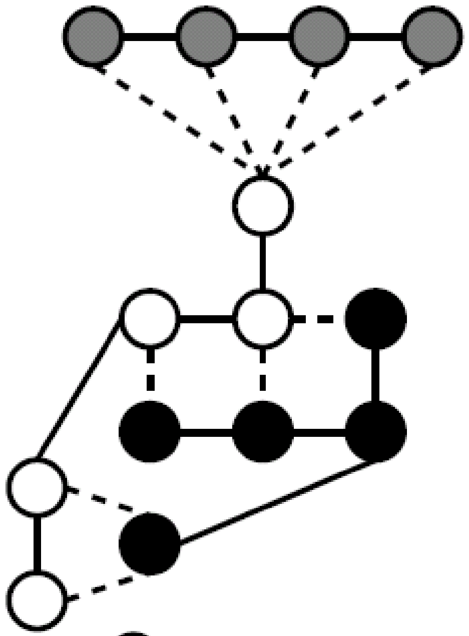
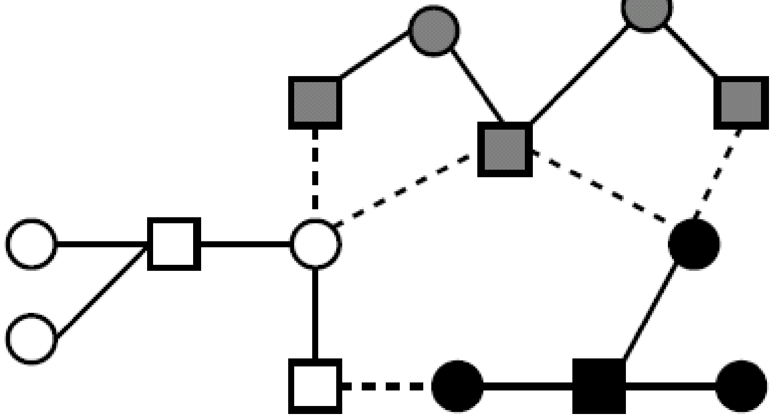
- **Cycle**: sample each block using an MH algorithm with invariant distribution $p(x_{bi}|x_{-bi})$ and proposal distribution $q(x_{bi})$, where $x_{-bi} = \{\text{All blocks except } x_{bi}\}$.

- Block highly correlated variables.



Chain can take a long time to mix when variables are correlated.

Collapsing and Blocking

Auxiliary Variable Samplers

- It is often easier to sample from an augmented distribution $p(x, u)$, where u is an auxiliary variable, than from $p(x)$.
- It is possible to obtain marginal samples $x^{(i)}$ by sampling $(x^{(i)}, u^{(i)})$ according to $p(x, u)$ and, then, ignoring the samples $u^{(i)}$.
- This very useful idea was proposed in the physics literature (Swendsen and Wang, 1987).

Hybrid Monte Carlo

- The idea is to exploit gradient information.
- Define the extended target distribution:

$$p(x,u) = p(x)N(u;0,I_{n_x}).$$

- Introduce the gradient vector: $\Delta(x) = \partial \log p(x)/\partial x$
- Introduce the parameters ρ and L.
- Next we “leapfrog”.

Hybrid Monte Carlo

- Sample $v \sim U_{[0,1]}$ and $u^{\star} \sim N(0, I_{n_x})$.
- Let $x_0 = x^{(i)}$ and $u_0 = u^{\star} + \rho\Delta(x_0)/2$.
- For $l = 1, \ldots, L$, take steps

$$x_l = x_{l-1} + \rho u_{l-1}$$

$$u_l = u_{l-1} + \rho_l \Delta(x_l)$$

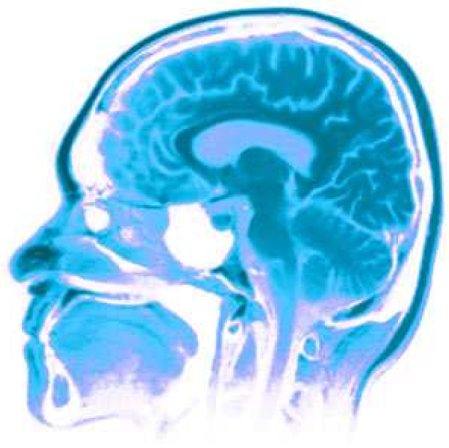
where $\rho_l = \rho$ for $l < L$ and $\rho_L = \rho/2$.

- If $v < A = \min\left\{1, \frac{p(x_L)}{p(x^{(i)})} \exp\left(-\frac{1}{2}(u_L^{\mathrm{T}} u_L - u^{\star\mathrm{T}} u^{\star})\right)\right\}$

$$(x^{(i+1)}, u^{(i+1)}) = (x_L, u_L)$$

else $$(x^{(i+1)}, u^{(i+1)}) = (x^{(i)}, u^{\star})$$

Next class



The EM algorithm

Nando de Freitas
2011
KPM Book Sections: *10*