Introduction to Numerical Bayesian Methods

A. Taylan Cemgil

Signal Processing and Communications Lab.

UNIVERSITY OF CAMBRIDGE

Department of Engineering

Boğaziçi Üniversitesi, Bilgisayar Mühendisliği Bölümü
7-8 Eylül 2006, İstanbul, Türkiye

Thanks to

- Nick Whiteley
- Simon Godsill
- Bill Fitzgerald

(Bu slaytlar muhtemelen değişebilir, en son versyon için aşağıdaki link'e bakın)

`http://www-sigproc.eng.cam.ac.uk/~atc27/papers/cemgil-bu-pres.pdf`

Outline, Day 1

- Introduction, Bayes' Theorem,
- Probability models, Bayesian Networks and Factor graphs
- Applications
- Deterministic Inference Techniques
  - Variational Methods: Variational Bayes, EM, ICM
- Stochastic (Sampling Based) Methods
  - Markov Chain Monte Carlo (MCMC)
    - Gibbs Sampler
    - Simulated Annealing
    - Iterative Improvement

Outline, Day 2

- Time Series models
  - Hidden Markov Models, Kalman Filter Models
  - Switching State Space models, Changepoint models
  - Nonlinear Dynamical Systems
- Applications
- Exact inference in time series models
  - Filtering
  - Smoothing
- Online Approximate Inference
  - Importance Sampling
  - Sequential Monte Carlo, Particle Filtering
- Yetsin artık bu kadar

Bayes' Theorem [4, 6]

Thomas Bayes (1702-1761)

What you know about a parameter θ after the data $\mathcal{D}$ arrive is what you knew before about θ and what the data $\mathcal{D}$ told you.

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{p(\mathcal{D})}$$

$$\text{Posterior} = \frac{\text{Likelihood} \times \text{Prior}}{\text{Evidence}}$$

Bayes' Theorem

- This rather simple looking formula has surprisingly many applications

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{p(\mathcal{D})}$$

  - Medical Diagnosis (Symptoms/Diseases)
  - Computer Vision (Pixels/Object)
  - Speech Recognition (Signal/Phoneme)
  - Music Transcription (Audio/Score)
  - Robotics/Navigation (Sensor Reading/Position)
  - Finance (Past Price/Future Price)
  - . . .

An application of Bayes' Theorem: "Parameter Estimation"

Given two fair dice with outcomes λ and y,

$$\mathcal{D} = \lambda + y$$

What is λ when $\mathcal{D} = 9$?

An application of Bayes' Theorem: "Parameter Estimation"

$$\mathcal{D} = \lambda + y = 9$$

$\mathcal{D} = \lambda + y$	$y = 1$	$y = 2$	$y = 3$	$y = 4$	$y = 5$	$y = 6$
$\lambda = 1$	2	3	4	5	6	7
$\lambda = 2$	3	4	5	6	7	8
$\lambda = \mathbf{3}$	4	5	6	7	8	**9**
$\lambda = \mathbf{4}$	5	6	7	8	**9**	10
$\lambda = \mathbf{5}$	6	7	8	**9**	10	11
$\lambda = \mathbf{6}$	7	8	**9**	10	11	12

Bayes theorem "upgrades" $p(\lambda)$ into $p(\lambda|\mathcal{D})$.

But you have to provide an observation model: $p(\mathcal{D}|\lambda)$

An application of Bayes' Theorem: "Parameter Estimation"

Formally we write

$$\begin{aligned} p(\lambda) &= \mathcal{C}(\lambda; [\ 1/6 \quad 1/6 \quad 1/6 \quad 1/6 \quad 1/6 \quad 1/6\ ]) \\ p(y) &= \mathcal{C}(y; [\ 1/6 \quad 1/6 \quad 1/6 \quad 1/6 \quad 1/6 \quad 1/6\ ]) \\ p(\mathcal{D}|\lambda, y) &= \delta(\mathcal{D} - (\lambda + y)) \end{aligned}$$

$$\begin{aligned} p(\lambda|\mathcal{D}) &= \frac{1}{p(\mathcal{D})} \times p(\mathcal{D}|\lambda, y) \times p(y)p(\lambda) \\ &= \frac{1}{\text{Evidence}} \times \text{Likelihood} \times \text{Prior} \end{aligned}$$

Kronecker delta function denoting a degenerate (deterministic) distribution $\delta(x) = \begin{cases} 1 & x = 0 \\ 0 & x \neq 0 \end{cases}$

Prior

$$p(y)p(\lambda)$$

$p(y) \times p(\lambda)$	$y = 1$	$y = 2$	$y = 3$	$y = 4$	$y = 5$	$y = 6$
$\lambda = 1$	$1/36$	$1/36$	$1/36$	$1/36$	$1/36$	$1/36$
$\lambda = 2$	$1/36$	$1/36$	$1/36$	$1/36$	$1/36$	$1/36$
$\lambda = 3$	$1/36$	$1/36$	$1/36$	$1/36$	$1/36$	$1/36$
$\lambda = 4$	$1/36$	$1/36$	$1/36$	$1/36$	$1/36$	$1/36$
$\lambda = 5$	$1/36$	$1/36$	$1/36$	$1/36$	$1/36$	$1/36$
$\lambda = 6$	$1/36$	$1/36$	$1/36$	$1/36$	$1/36$	$1/36$

Likelihood

$$p(\mathcal{D} = 9|\lambda, y)$$

$p(\mathcal{D} = 9\|\lambda, y)$	$y = 1$	$y = 2$	$y = 3$	$y = 4$	$y = 5$	$y = 6$
$\lambda = 1$	0	0	0	0	0	0
$\lambda = 2$	0	0	0	0	0	0
$\lambda = 3$	0	0	0	0	0	$\mathbf{1}$
$\lambda = 4$	0	0	0	0	$\mathbf{1}$	0
$\lambda = 5$	0	0	0	$\mathbf{1}$	0	0
$\lambda = 6$	0	0	$\mathbf{1}$	0	0	0

Likelihood $\times$ Prior

$$\phi_{\mathcal{D}}(\lambda, y) = p(\mathcal{D} = 9|\lambda, y)p(\lambda)p(y)$$

$p(\mathcal{D} = 9\|\lambda, y)$	$y = 1$	$y = 2$	$y = 3$	$y = 4$	$y = 5$	$y = 6$
$\lambda = 1$	0	0	0	0	0	0
$\lambda = 2$	0	0	0	0	0	0
$\lambda = 3$	0	0	0	0	0	$\mathbf{1/36}$
$\lambda = 4$	0	0	0	0	$\mathbf{1/36}$	0
$\lambda = 5$	0	0	0	$\mathbf{1/36}$	0	0
$\lambda = 6$	0	0	$\mathbf{1/36}$	0	0	0

Posterior

$$p(\lambda, y|\mathcal{D}=9) \;=\; \frac{1}{p(\mathcal{D})} p(\mathcal{D}=9|\lambda, y)p(\lambda)p(y)$$

$p(\mathcal{D}=9\|\lambda,y)$	$y=1$	$y=2$	$y=3$	$y=4$	$y=5$	$y=6$
$\lambda=1$	0	0	0	0	0	0
$\lambda=2$	0	0	0	0	0	0
$\lambda=3$	0	0	0	0	0	$\mathbf{1/4}$
$\lambda=4$	0	0	0	0	$\mathbf{1/4}$	0
$\lambda=5$	0	0	0	$\mathbf{1/4}$	0	0
$\lambda=6$	0	0	$\mathbf{1/4}$	0	0	0

$$p(\mathcal{D}=9) \;=\; \sum_{\lambda,y} p(\mathcal{D}=9|\lambda,y)p(\lambda)p(y) = 0+0+\cdots+1/36+1/36+1/36+1/36+0+\cdots+0 = 1/9$$

$$1/4 \;=\; (1/36)/(1/9)$$

Marginal Posterior

$$p(\lambda|\mathcal{D}) \quad = \quad \sum_y \frac{1}{p(\mathcal{D})} p(\mathcal{D}|\lambda, y) p(\lambda) p(y)$$

	$p(\lambda\|\mathcal{D}=9)$	$y=1$	$y=2$	$y=3$	$y=4$	$y=5$	$y=6$
$\lambda=1$	0	0	0	0	0	0	0
$\lambda=2$	0	0	0	0	0	0	0
$\lambda=3$	$1/4$	0	0	0	0	0	$\mathbf{1/4}$
$\lambda=4$	$1/4$	0	0	0	0	$\mathbf{1/4}$	0
$\lambda=5$	$1/4$	0	0	0	$\mathbf{1/4}$	0	0
$\lambda=6$	$1/4$	0	0	$\mathbf{1/4}$	0	0	0

The “proportional to” notation

$$p(\lambda|\mathcal{D}) \quad \propto \quad \sum_y p(\mathcal{D}|\lambda, y)p(\lambda)p(y)$$

	$p(\lambda\|\mathcal{D}=9)$	$y=1$	$y=2$	$y=3$	$y=4$	$y=5$	$y=6$
$\lambda=1$	0	0	0	0	0	0	0
$\lambda=2$	0	0	0	0	0	0	0
$\lambda=3$	$1/36$	0	0	0	0	0	$\mathbf{1/36}$
$\lambda=4$	$1/36$	0	0	0	0	$\mathbf{1/36}$	0
$\lambda=5$	$1/36$	0	0	0	$\mathbf{1/36}$	0	0
$\lambda=6$	$1/36$	0	0	$\mathbf{1/36}$	0	0	0

Another application of Bayes' Theorem: "Model Selection"

Given an unknown number of fair dice with outcomes $\lambda_1, \lambda_2, \ldots, \lambda_n$,

$$\mathcal{D} = \sum_{i=1}^{n} \lambda_i$$

How many dice are there when $\mathcal{D} = 9$?

Assume that any number n is equally likely

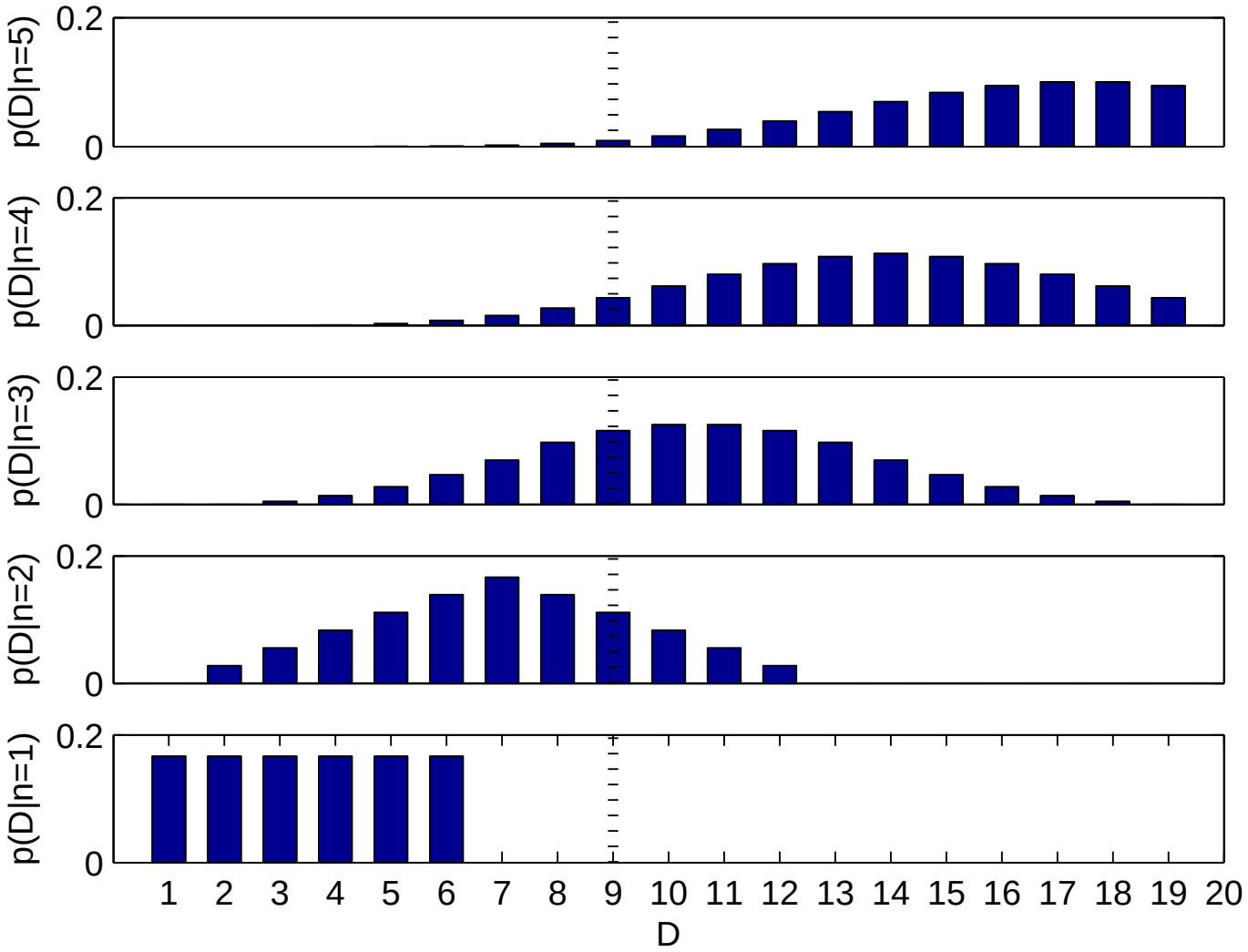
Another application of Bayes' Theorem: "Model Selection"

Given all n are equally likely (i.e., $p(n)$ is flat), we calculate (formally)

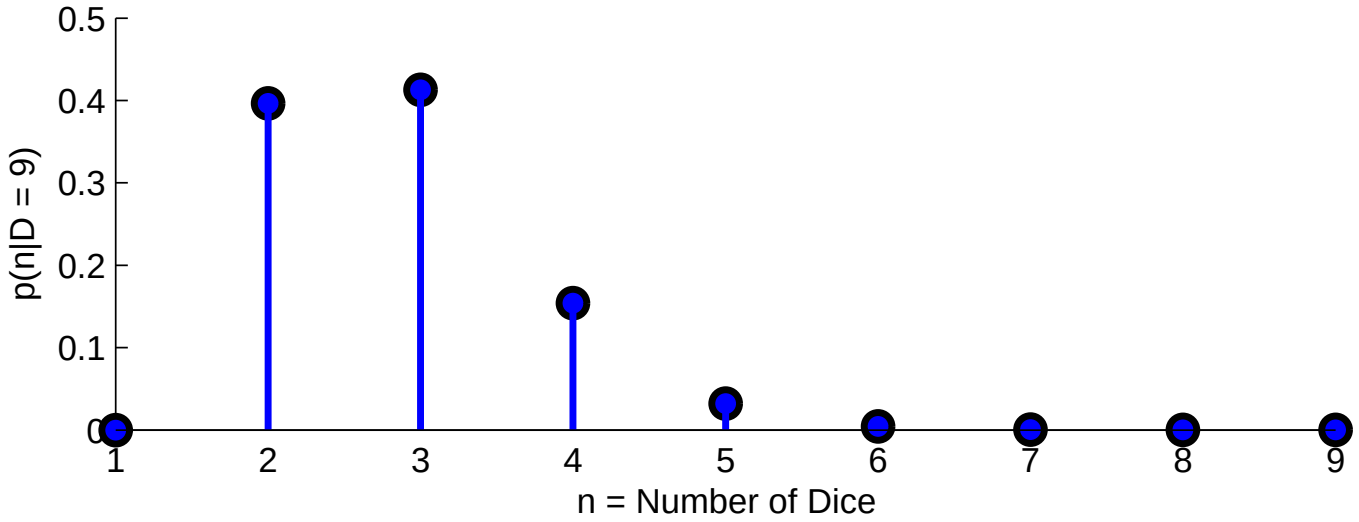
$$p(n|\mathcal{D}=9) = \frac{p(\mathcal{D}=9|n)p(n)}{p(\mathcal{D})} \propto p(\mathcal{D}=9|n)$$

$$\begin{aligned}
p(\mathcal{D}|n=1) &= \sum_{\lambda_1} p(\mathcal{D}|\lambda_1)p(\lambda_1) \\
p(\mathcal{D}|n=2) &= \sum_{\lambda_1}\sum_{\lambda_2} p(\mathcal{D}|\lambda_1,\lambda_2)p(\lambda_1)p(\lambda_2) \\
&\cdots \\
p(\mathcal{D}|n=n') &= \sum_{\lambda_1,\ldots,\lambda_{n'}} p(\mathcal{D}|\lambda_1,\ldots,\lambda_{n'}) \prod_{i=1}^{n'} p(\lambda_i)
\end{aligned}$$

$$p(\mathcal{D}|n) = \sum_{\boldsymbol{\lambda}} p(\mathcal{D}|\boldsymbol{\lambda}, n) p(\boldsymbol{\lambda}|n)$$



Another application of Bayes' Theorem: "Model Selection"



- Complex models are more flexible but they spread their probability mass
- Bayesian inference inherently prefers "simpler models" – Occam's razor
- Computational burden: We need to sum over all parameters λ

Tutorial’ımız Bitmiştir, İlginize teşekkürler

Probability Models

$+$

Inference Algorithms

$=$

Bayesian Numerical Methods

Formal Languages for specification of Probability Models and Inference Algorithms

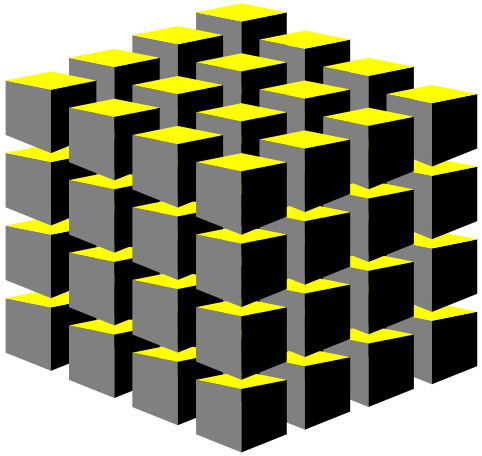
- **Directed Graphical Models, Directed Acyclic Graphs (DAG), Bayesian Networks**
- Undirected Graphs, Markov Networks, Random Fields
- **Factor Graphs**

Conditional Independence

Discrete conditional probability tables

- Assume all x_i are discrete with $|x_i| = k$. If N is large, a naive table representation is HUGE: k^N entries

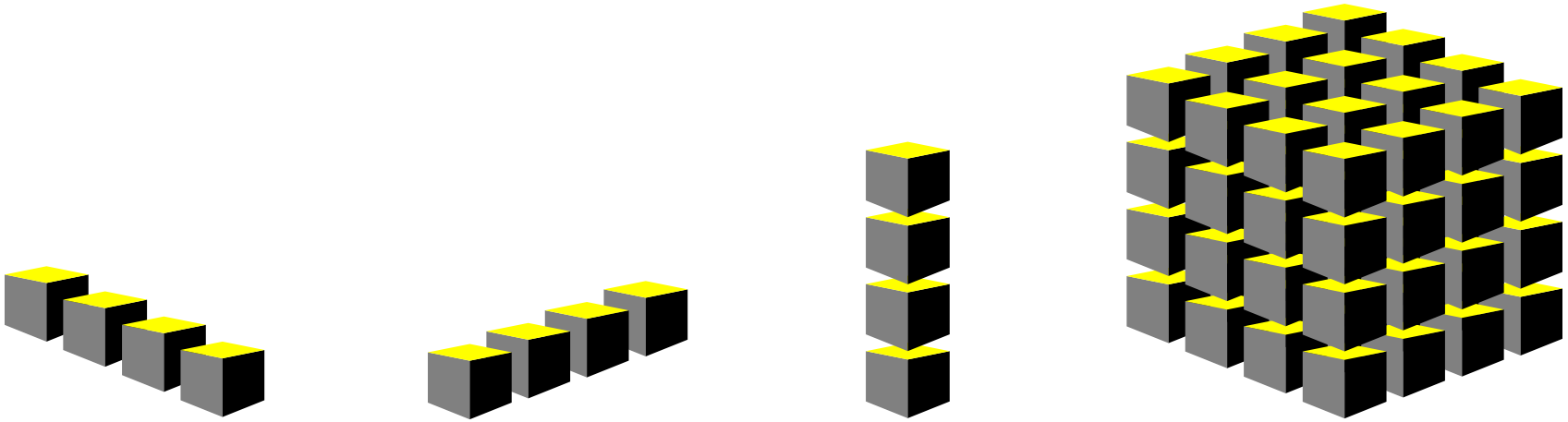
Example: $p(x_1, x_2, x_3)$ with $|x_i| = 4$



Each cell is a positive number s.t. $\sum_{x_1,x_2,x_3} p(x_1, x_2, x_3) = 1$

Independence Assumption == Complete Factorization

- Assume $p(x_1, x_2, \ldots, x_N) = \prod_k p(x_k)$.

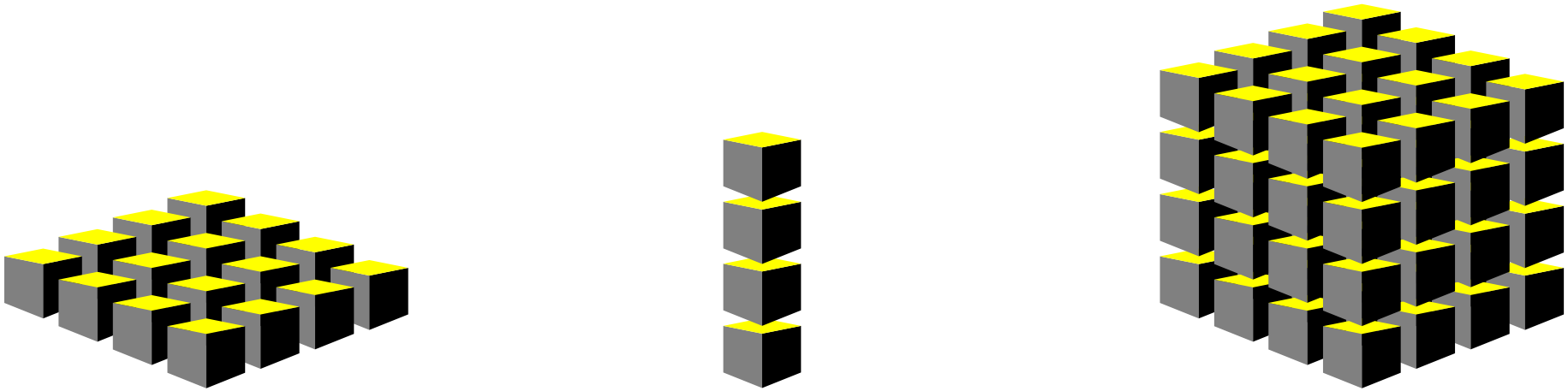


$p(x_1) \quad \times \quad p(x_2) \quad \times \quad p(x_3) \quad = \quad p(x_1, x_2, x_3)$

We need to store 4×3 numbers instead of 4^3 !

- However, complete independence may be too restrictive.

An alternative Factorization



$$p(x_1, x_2) \quad \times \quad p(x_3) \quad = \quad p(x_1, x_2, x_3)$$

- We need to store $4^2 + 4$ numbers instead of 4^3.
- Still some variables are independent from rest. It is possible to introduce conditional independence relations to design “richer” distributions.

Conditional Independence

- Two sets of variables A and B are conditionally independent given a third set C if

$$p(A, B|C) \quad = \quad p(A|C)p(B|C)$$

- This is equivalent to

$$p(A|BC) \quad = \quad p(A|C)$$

Exercise

	$x_2 = 1$	$x_2 = 2$
$x_1 = 1$	0.6	0.1
$x_1 = 2$	0.2	0.1

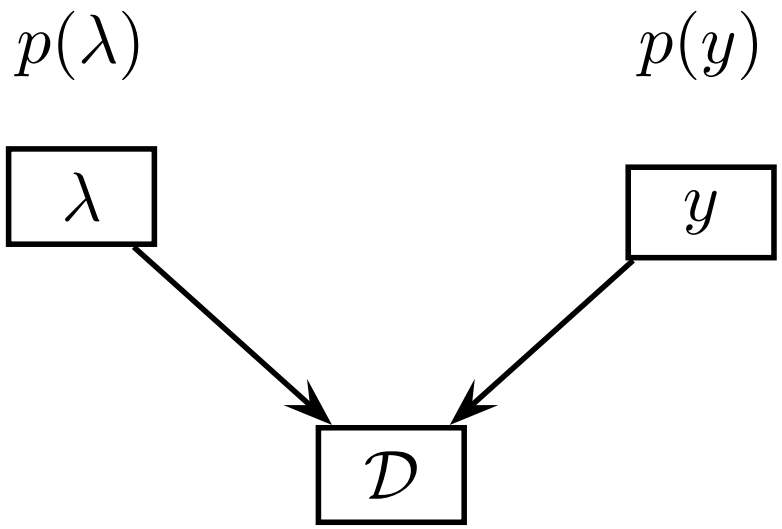
1. Find the following quantities
   - Marginals: $p(x_1)$, $p(x_2)$
   - Conditionals: $p(x_1|x_2)$, $p(x_2|x_1)$
   - Posterior: $p(x_1, x_2 = 2)$, $p(x_1|x_2 = 2)$
   - Evidence $p(x_2 = 2)$
   - $p(\{\})$

2. Are x_1 and x_2 independent ? If not, construct a new probability table where x_1 and x_2 are independent but still have the same marginals.

3. Construct a new probability table $p(x_1, x_3)$ such that $p(x_2, x_3|x_1) = p(x_2|x_1)p(x_3|x_1)$ but $p(x_3) = [0.5 \;\; 0.5]$. Do you have any freedom in choosing the new table ?

DAG Example: Two dice

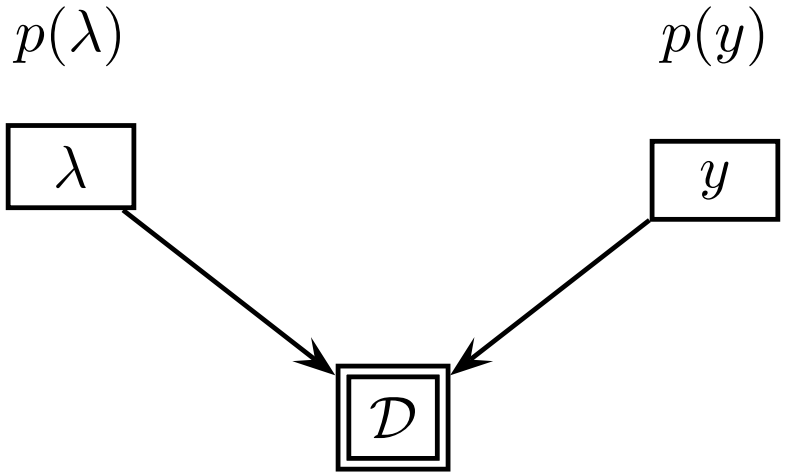
Given two fair dice with outcomes λ and y where $\mathcal{D} = \lambda + y$...



$$p(\mathcal{D}, \lambda, y) \quad = \quad p(\mathcal{D}|\lambda, y)p(\lambda)p(y)$$

DAG with observations

Given two fair dice with outcomes λ and y when $\mathcal{D} = \lambda + y = 9$



$$p(\mathcal{D} = 9|\lambda, y)$$

$$\phi_{\mathcal{D}}(\lambda, y) \quad = \quad p(\mathcal{D} = 9|\lambda, y)p(\lambda)p(y)$$

Directed Graphical models

- Each random variable is associated with a node in the graph,
- We draw an arrow from $x_j \rightarrow x_i$ each parent node $x_j \in \text{parent}(x_i)$,

$$p(x_1, \ldots, x_N) \quad = \quad \prod_{i=1}^{N} p(x_i|\text{parent}(x_i))$$

- Every joint probability distribution over finite number of variables can be written in this form, but this is not necessarily the minimal representation,
- Describes in a compact way how data is “generated”,
- Technically, missing links denote conditional independence relations between variables. This turns out to be very important in developing efficient inference algorithms.

Graphical Models

- Graphical models represent joint distributions compactly using a set of local variables
- Each variable corresponds to a node in the graph
- The edges tell us *qualitatively* about the factorization of the joint probability
- There are *functions* at the nodes that tell us the quantitative details of the factors

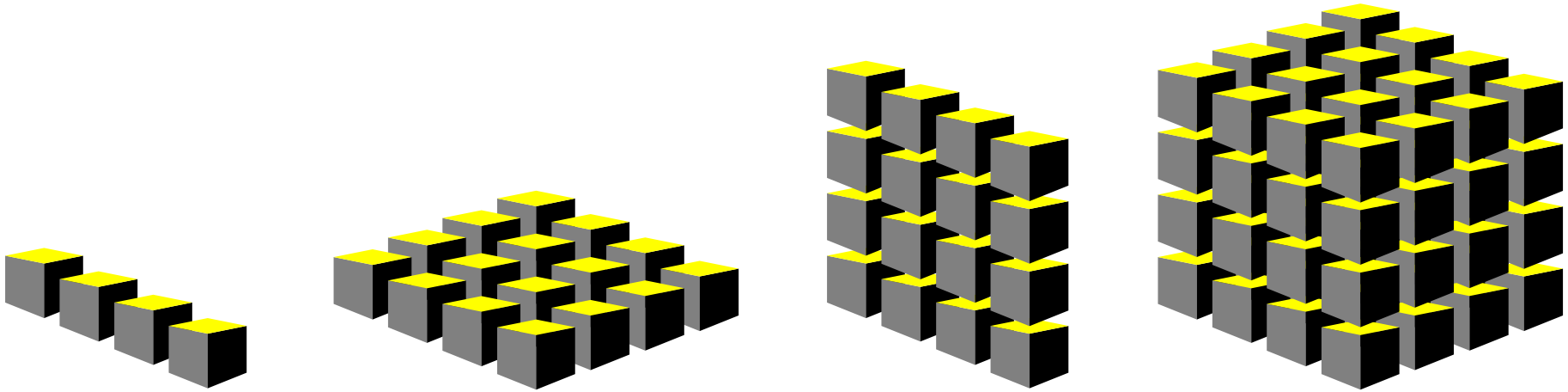
Directed Graphical Models

- Consider *directed acyclic graphs* over N variables
- Each node x_i has a (possibly empty) set of parents denoted by $pa(x_i)$
- Each node has a function $p(x_i|pa(x_i))$.
- The joint probability is given by

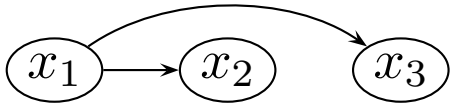
$$p(x_1, x_2, \ldots, x_N) = \prod_i p(x_i|pa(x_i))$$

- Factorization in terms of local functions

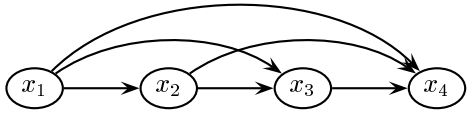
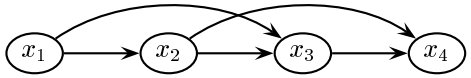
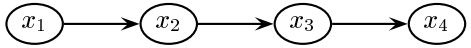
Examples



$$p(x_1) \times p(x_2|x_1) \times p(x_3|x_1) = p(x_1, x_2, x_3)$$



Examples

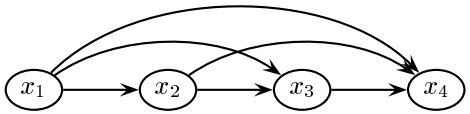
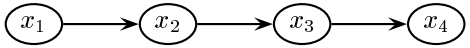
Model	Structure	factorization
Full		$p(x_1)p(x_2\|x_1)p(x_3\|x_1,x_2)p(x_4\|x_1,x_2,x_3)$
Markov(2)		$p(x_1)p(x_2\|x_1)p(x_3\|x_1,x_2)p(x_4\|x_2,x_3)$
Markov(1)		$p(x_1)p(x_2\|x_1)p(x_3\|x_2)p(x_4\|x_3)$
		$p(x_1)p(x_2\|x_1)p(x_3\|x_1)p(x_4)$
Factorized		$p(x_1)p(x_2)p(x_3)p(x_4)$

Removing edges eliminates a term from the conditional probability factors.

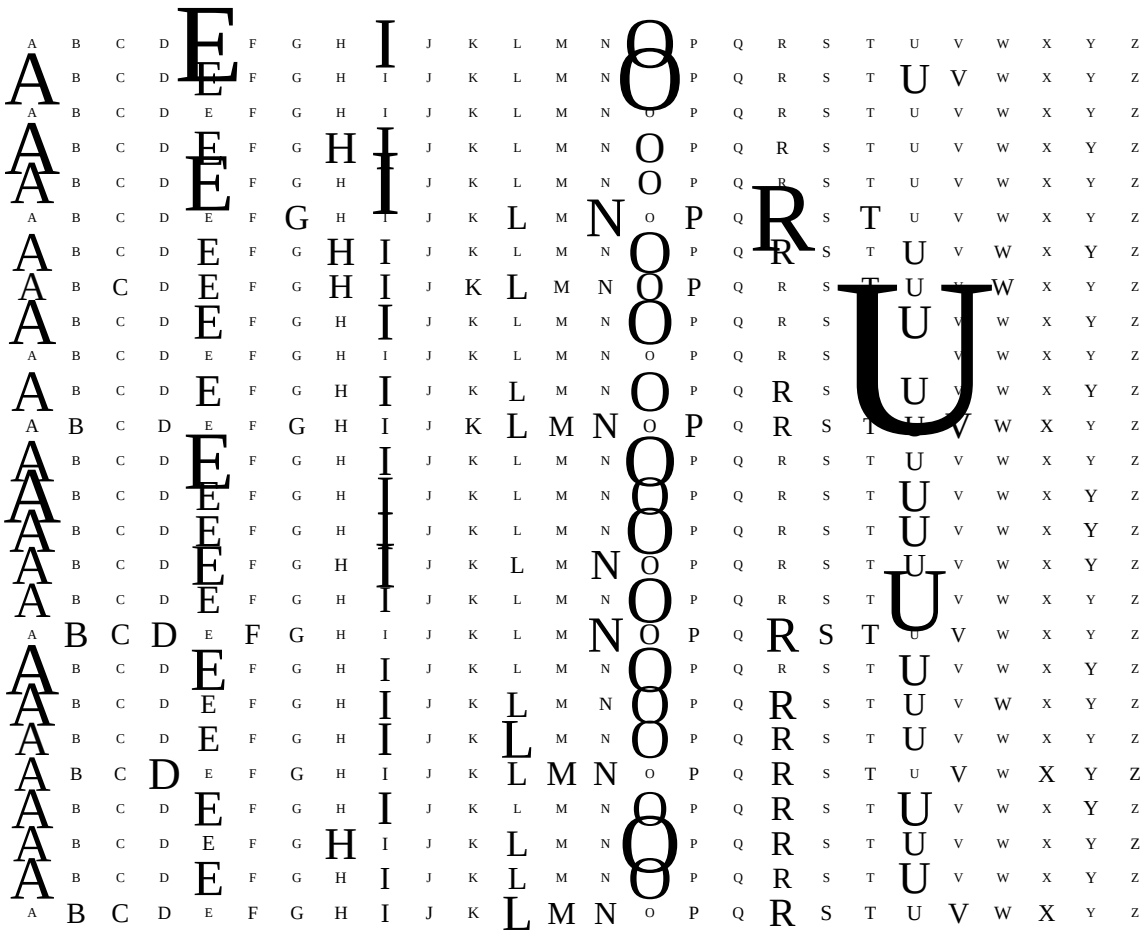
Examples

Dataset (From Sayood): All four letter English words (2149) of a Sun-Sparc spell checker.

`(abbe, abed, abel, abet, able, ...  zion, zone, zoom, zorn)`

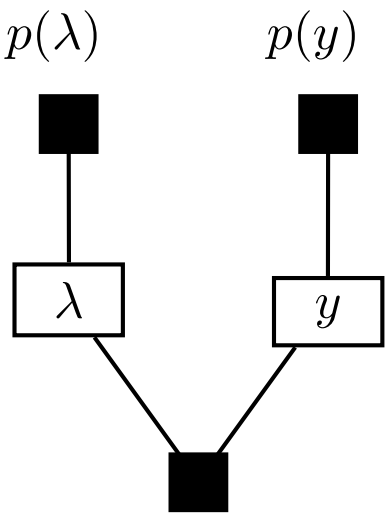
Model	Structure	Random Samples
Full		**tabu, else, duly, crib, bohr, seal, tome, free, bern**
Markov(1)		**miro**, `jaid, saun, trol, bale, liro, pibo, brox, heth`
Factorized		`yiij, ekmy, vguo, addn, ecmi, miui, bhin, hnri, roia, azfa`

Estimated model $p(x_k|x_{k-1})$ for the four letter words dataset



Factor graphs

Factor graph for two dice example [5]



$$p(\mathcal{D} = 9|\lambda, y)$$

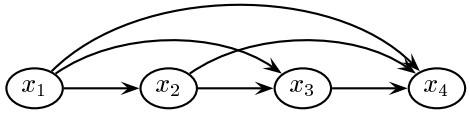
$$\phi_{\mathcal{D}}(\lambda, y) \quad = \quad p(\mathcal{D} = 9|\lambda, y)p(\lambda)p(y) = \phi_1(\lambda, y)\phi_2(\lambda)\phi_3(y)$$

- A bipartite graph. A powerful graphical representation of the inference problem
  - **Factor nodes**: Black squares. Factor potentials (local functions) defining the posterior.
  - **Variable nodes**: White Circles.
  - **Edges**: denote membership. A variable is connected to a factor if it is an argument of the local function.

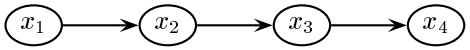
Exercise

- For the following Graphical models, write down the factors of the joint distribution and plot the corresponding factor graphs.

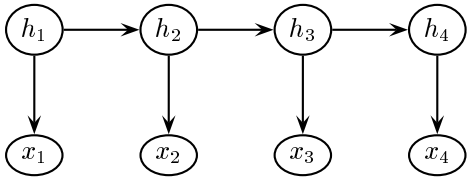
Full



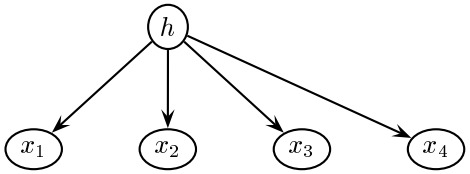
Markov(1)



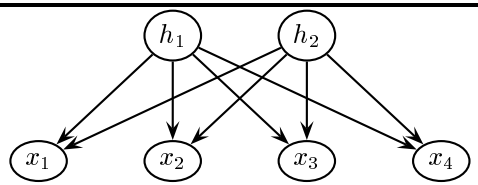
HMM



MIX



IFA



Factorized

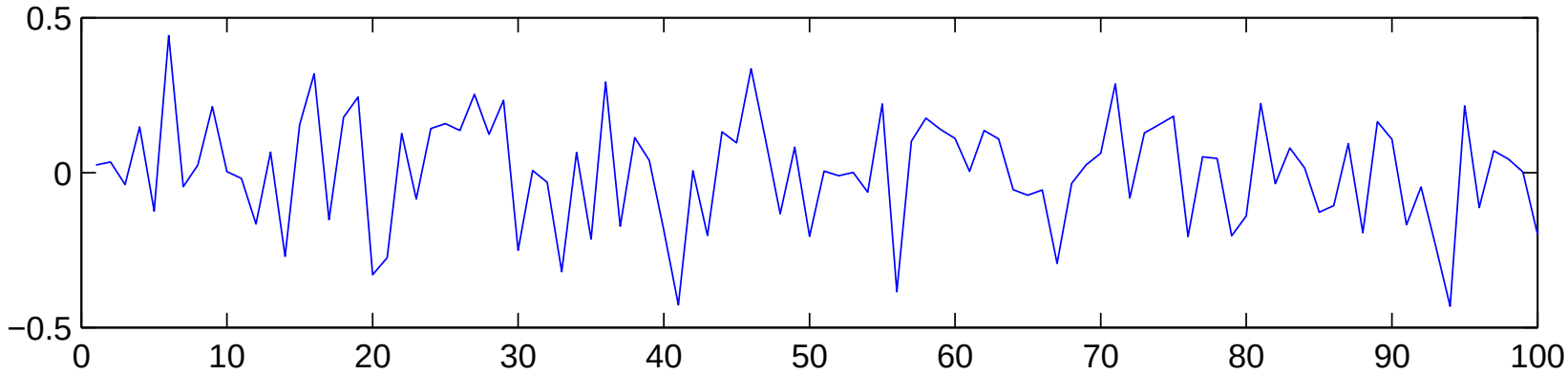
Example: AR(1) model

$$x_k = Ax_{k-1} + \epsilon_k \qquad\qquad k = 1 \ldots K$$

ϵ_k is i.i.d., zero mean and normal with variance R.

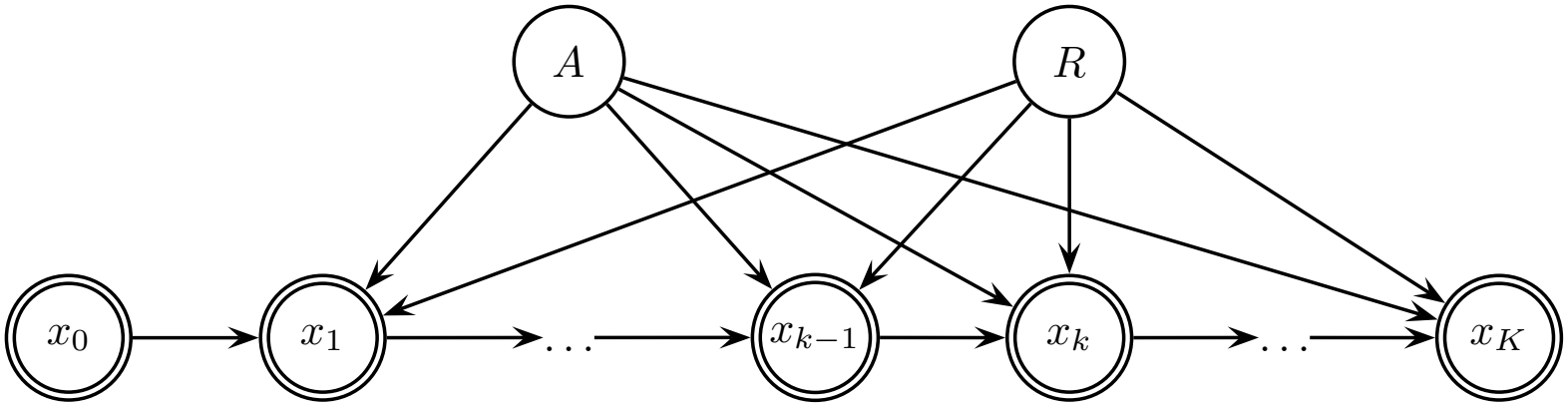
Estimation problem:

Given $x_0, \ldots, x_K$, determine coefficient A and variance R (both scalars).



AR(1) model, Generative Model notation

$$
\begin{aligned}
A &\sim \mathcal{N}(A; 0, P) \\
R &\sim \mathcal{IG}(R; \nu, \beta/\nu) \\
x_k | x_{k-1}, A, R &\sim \mathcal{N}(x_k; Ax_{k-1}, R) \qquad x_0 = \hat{x}_0
\end{aligned}
$$



Gaussian : $\mathcal{N}(x; \mu, V) \equiv |2\pi V|^{-\frac{1}{2}} \exp(-\frac{1}{2}(x-\mu)^2/V)$

Inverse-Gamma distribution: $\mathcal{IG}(x; a, b) \equiv \Gamma(a)^{-1} b^{-a} x^{-(a+1)} \exp(-1/(bx)) \quad x \geq 0$

Observed variables are shown with double circles

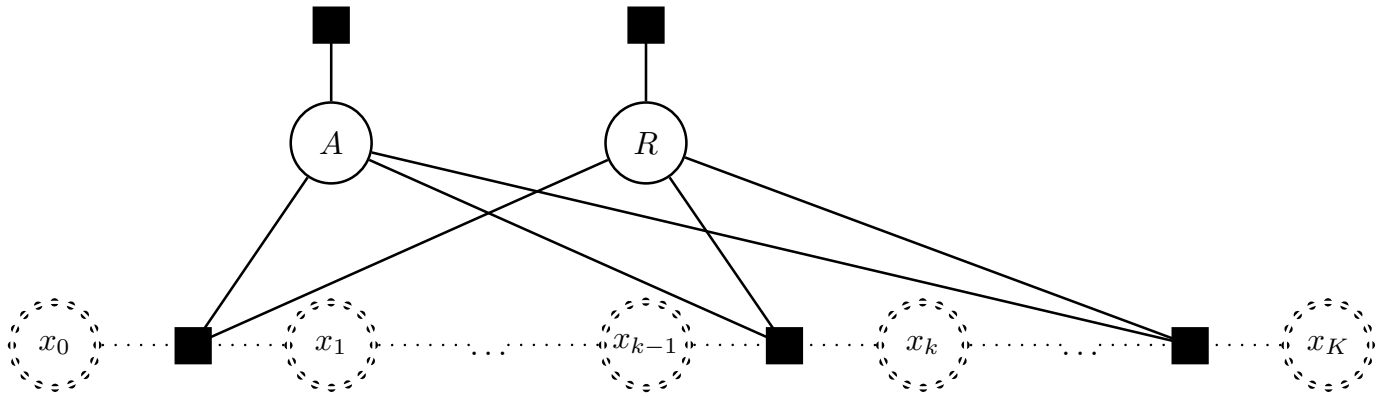
AR(1) Model. Bayesian Posterior Inference

$$p(A, R|x_0, x_1, \ldots, x_K) \quad \propto \quad p(x_1, \ldots, x_K|x_0, A, R)p(A, R)$$

$$\text{Posterior} \quad \propto \quad \text{Likelihood} \times \text{Prior}$$

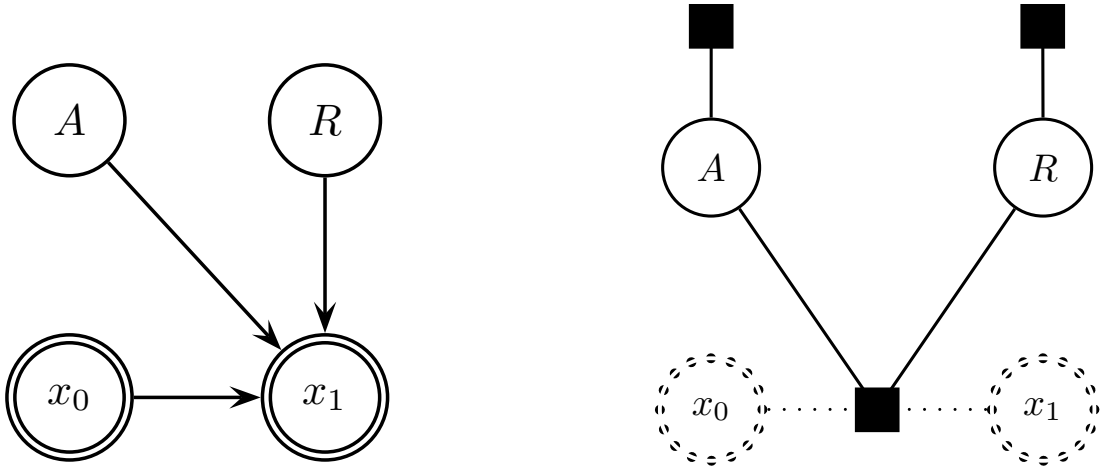
Using the Markovian (conditional independence) structure we have

$$p(A, R|x_0, x_1, \ldots, x_K) \quad \propto \quad \left(\prod_{k=1}^{K} p(x_k|x_{k-1}, A, R) \right) p(A)p(R)$$



Numerical Example

Suppose $K = 1$,

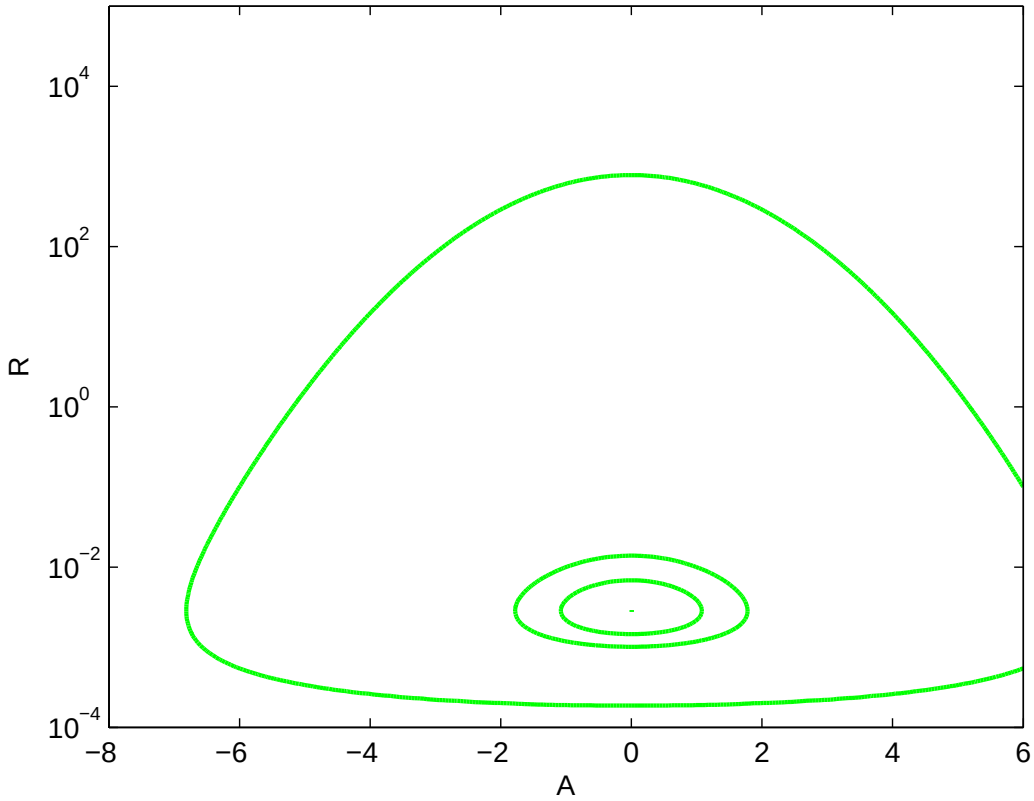


By Bayes' Theorem and the structure of AR(1) model

$$\begin{aligned} p(A, R|x_0, x_1) &\propto p(x_1|x_0, A, R)p(A)p(R) \\ &= \mathcal{N}(x_1; Ax_0, R)\mathcal{N}(A; 0, P)\mathcal{IG}(R; \nu, \beta/\nu) \end{aligned}$$

Numerical Example, the prior $p(A, R)$

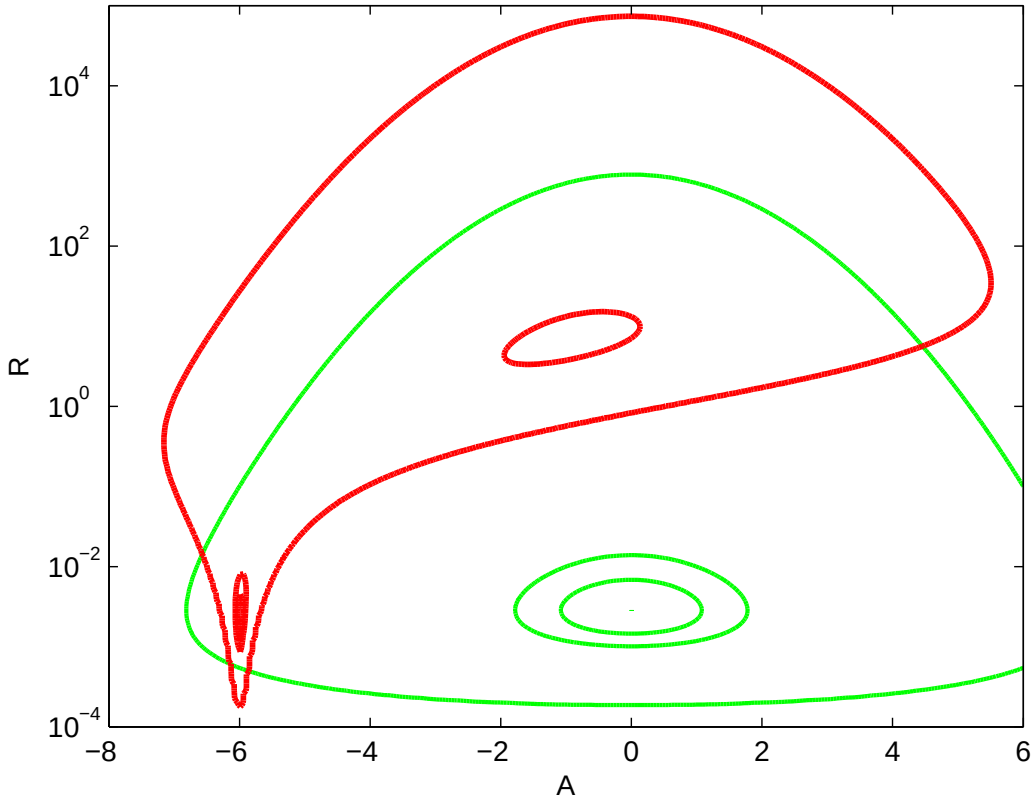
Equiprobability contour of $p(A)p(R)$



$$A \sim \mathcal{N}(A; 0, 1.2) \qquad R \sim \mathcal{IG}(R; 0.4, 250)$$

Suppose: $x_0 = 1$ $\qquad x_1 = -6$ $\qquad x_1 \sim \mathcal{N}(x_1; Ax_0, R)$

Numerical Example, the posterior $p(A, R|x)$



Note the bimodal posterior with $x_0 = 1, x_1 = -6$

- $A \approx -6 \Leftrightarrow$ low noise variance R.
- $A \approx 0 \Leftrightarrow$ high noise variance R.

Remarks

- The maximum likelihood solution (or any other point estimate) is not always representative about the solution
- (Unfortunately), exact posterior inference is only possible for few special cases
- Even very simple models can lead easily to complicated posterior distributions
- *A-priori* independent variables often become dependent *a-posteriori* (“Explaining away”)
- Ambiguous data usually leads to a multimodal posterior, each mode corresponding to one possible explanation
- The complexity of an inference problem depends, among others, upon the particular “parameter regime” and observed data sequence

Probabilistic Inference

A huge spectrum of applications – all boil down to computation of

- **expectations** of functions under probability distributions: **Integration**

$$\langle f(x) \rangle \quad = \quad \int_{\mathcal{X}} dx p(x) f(x)$$

- **modes** of functions under probability distributions: **Optimization**

$$x^* \quad = \quad \underset{x \in \mathcal{X}}{\operatorname{argmax}}\, p(x) f(x)$$

- any “mix” of the above: e.g.,

$$x^* \quad = \quad \underset{x \in \mathcal{X}}{\operatorname{argmax}}\, p(x) = \underset{x \in \mathcal{X}}{\operatorname{argmax}} \int_{\mathcal{Z}} dz p(z) p(x|z)$$

Divide and Conquer

Probabilistic modelling provides a methodology that puts a clear division between

- What to solve : Model Construction
  - Both an Art and Science
  - Highly domain specific
- How to solve : Inference Algorithm
  - (In principle) Mechanical
  - Generic

"*An approximate solution of the exact problem is often more useful than the exact solution of an approximate problem*",

J. W. Tukey (1915-2000).

Attributes of Probabilistic Inference

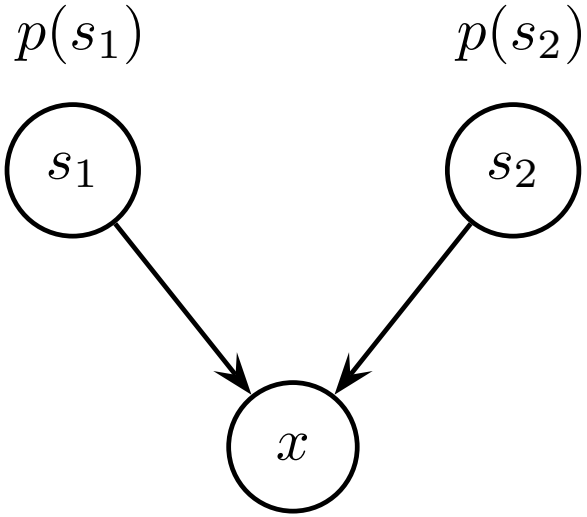
- Exact $\leftrightarrow$ **Approximate**
- **Deterministic** $\leftrightarrow$ **Stochastic**
- **Online** $\leftrightarrow$ **Offline**
- **Centralized** $\leftrightarrow$ Distributed

This talk focuses on the bold ones

Deterministic Inference

Mean Field – Variational Bayes

Toy Model : “One sample source separation (OSSS)”



This graph encodes the joint: $p(x, s_1, s_2) = p(x|s_1, s_2)p(s_1)p(s_2)$

$$
\begin{aligned}
s_1 &\sim p(s_1) = \mathcal{N}(s_1; \mu_1, P_1) \\
s_2 &\sim p(s_2) = \mathcal{N}(s_2; \mu_2, P_2) \\
x|s_1, s_2 &\sim p(x|s_1, s_2) = \mathcal{N}(x; s_1 + s_2, R)
\end{aligned}
$$

The Gaussian Distribution

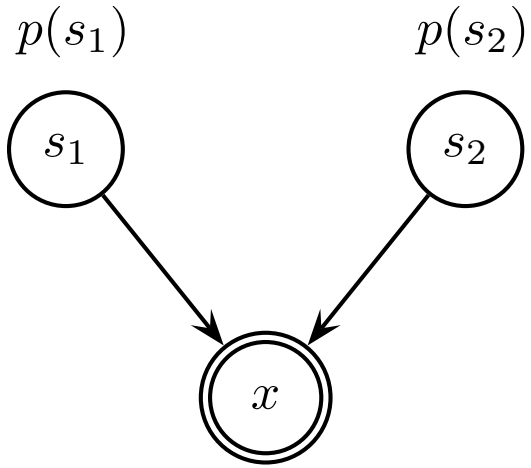
μ is the mean and P is the covariance:

$$
\begin{aligned}
\mathcal{N}(s;\mu,P) &= |2\pi P|^{-1/2}\exp\left(-\frac{1}{2}(s-\mu)^T P^{-1}(s-\mu)\right) \\
&= \exp\left(-\frac{1}{2}s^T P^{-1}s + \mu^T P^{-1}s - \frac{1}{2}\mu^T P^{-1}\mu - \frac{1}{2}|2\pi P|\right) \\
\log\mathcal{N}(s;\mu,P) &= -\frac{1}{2}s^T P^{-1}s + \mu^T P^{-1}s + \text{ const} \\
&= -\frac{1}{2}\mathbf{Tr}\, P^{-1}ss^T + \mu^T P^{-1}s + \text{ const} \\
&=^{+} -\frac{1}{2}\mathbf{Tr}\, P^{-1}ss^T + \mu^T P^{-1}s
\end{aligned}
$$

Notation: $\log f(x) =^{+} g(x) \iff f(x) \propto \exp(g(x)) \iff \exists c \in \mathbb{R} : f(x) = c\exp(g(x))$

OSSS example

Suppose, we observe $x = \hat{x}$.



$$p(x = \hat{x}|s_1, s_2)$$

- By Bayes' theorem, the posterior is given by:

$$\mathcal{P} \equiv p(s_1, s_2|x = \hat{x}) = \frac{1}{Z_{\hat{x}}} p(x = \hat{x}|s_1, s_2) p(s_1) p(s_2) \equiv \frac{1}{Z_{\hat{x}}} \phi(s_1, s_2)$$

- The function $\phi(s_1, s_2)$ is proportional to the exact posterior. ($Z_{\hat{x}} \equiv p(x = \hat{x})$)

OSSS example, cont.

$$\begin{aligned}
\log p(s_1) &= \mu_1^T P_1^{-1} s_1 - \frac{1}{2} s_1^T P_1^{-1} s_1 + \text{const} \\
\log p(s_2) &= \mu_2^T P_2^{-1} s_2 - \frac{1}{2} s_2^T P_2^{-1} s_2 + \text{const} \\
\log p(x|s_1, s_2) &= \hat{x}^T R^{-1} (s_1 + s_2) - \frac{1}{2} (s_1 + s_2)^T R^{-1} (s_1 + s_2) + \text{const}
\end{aligned}$$

$$\begin{aligned}
\log \phi(s_1, s_2) &= \log p(x = \hat{x}|s_1, s_2) + \log p(s_1) + \log p(s_2) \\
&=^+ \left(\mu_1^T P_1^{-1} + \hat{x}^T R^{-1}\right) s_1 + \left(\mu_2^T P_2^{-1} + \hat{x}^T R^{-1}\right) s_2 \\
&\quad -\frac{1}{2} \mathbf{Tr}\left(P_1^{-1} + R^{-1}\right) s_1 s_1^T - \underbrace{s_1^T R^{-1} s_2}_{(*)} - \frac{1}{2} \mathbf{Tr}\left(P_2^{-1} + R^{-1}\right) s_2 s_2^T
\end{aligned}$$

- The (*) term is the cross correlation term that makes s_1 and s_2 a-posteriori dependent.

OSSS example, cont.

Completing the square

$$\log \phi(s_1, s_2) \quad =^+ \quad \begin{pmatrix} P_1^{-1}\mu_1 + R^{-1}\hat{x} \\ P_2^{-1}\mu_2 + R^{-1}\hat{x} \end{pmatrix}^\top \begin{pmatrix} s_1 \\ s_2 \end{pmatrix}$$

$$-\frac{1}{2} \begin{pmatrix} s_1 \\ s_2 \end{pmatrix}^\top \begin{pmatrix} P_1^{-1} + R^{-1} & R^{-1} \\ R^{-1} & P_2^{-1} + R^{-1} \end{pmatrix} \begin{pmatrix} s_1 \\ s_2 \end{pmatrix}$$

Remember: $\log \mathcal{N}(s; m, \Sigma) \quad =^+ \quad (\Sigma^{-1}m)^\top s - \frac{1}{2} s^\top \Sigma^{-1} s$

$$\Sigma \quad = \quad \begin{pmatrix} P_1^{-1} + R^{-1} & R^{-1} \\ R^{-1} & P_2^{-1} + R^{-1} \end{pmatrix}^{-1} \qquad m = \Sigma \quad \begin{pmatrix} P_1^{-1}\mu_1 + R^{-1}\hat{x} \\ P_2^{-1}\mu_2 + R^{-1}\hat{x} \end{pmatrix}$$

Variational Bayes (VB), mean field

We will approximate the posterior $\mathcal{P}$ with a simpler distribution $\mathcal{Q}$.

$$\begin{aligned} \mathcal{P} &= \frac{1}{Z_x} p(x = \hat{x}|s_1, s_2) p(s_1) p(s_2) \\ \mathcal{Q} &= q(s_1) q(s_2) \end{aligned}$$

Here, we choose

$$q(s_1) = \mathcal{N}(s_1; m_1, S_1) \qquad q(s_2) = \mathcal{N}(s_2; m_2, S_2)$$

A “measure of fit” between distributions is the KL divergence

Kullback-Leibler (KL) Divergence

- A “quasi-distance” between two distributions $\mathcal{P} = p(x)$ and $\mathcal{Q} = q(x)$.

$$KL(\mathcal{P}||\mathcal{Q}) \equiv \int_{\mathcal{X}} dx p(x) \log \frac{p(x)}{q(x)} = \langle \log \mathcal{P} \rangle_{\mathcal{P}} - \langle \log \mathcal{Q} \rangle_{\mathcal{P}}$$

- Unlike a metric, (in general) it is not symmetric,

$$KL(\mathcal{P}||\mathcal{Q}) \neq KL(\mathcal{Q}||\mathcal{P})$$

- But it is non-negative (by Jensen’s Inequality)

$$\begin{aligned} KL(\mathcal{P}||\mathcal{Q}) &= -\int_{\mathcal{X}} dx p(x) \log \frac{q(x)}{p(x)} \\ &\geq -\log \int_{\mathcal{X}} dx p(x) \frac{q(x)}{p(x)} = -\log \int_{\mathcal{X}} dx q(x) = -\log 1 = 0 \end{aligned}$$

OSSS example, cont.

Let the approximating distribution be factorized as

$$\mathcal{Q} \quad = \quad q(s_1)q(s_2)$$

$$q(s_1) = \mathcal{N}(s_1; m_1, S_1) \qquad q(s_2) = \mathcal{N}(s_2; m_2, S_2)$$

The m_i and S_j are the *variational* parameters to be optimized to minimize

$$KL(\mathcal{Q}||\mathcal{P}) \quad = \quad \langle \log \mathcal{Q} \rangle_{\mathcal{Q}} - \left\langle \log \underbrace{\frac{1}{Z_x}\phi(s_1, s_2)}_{=\mathcal{P}} \right\rangle_{\mathcal{Q}} \tag{1}$$

The form of the mean field solution

$$
\begin{aligned}
0 &\leq \left\langle \log q(s_1)q(s_2) \right\rangle_{q(s_1)q(s_2)} + \log Z_x - \left\langle \log \phi(s_1, s_2) \right\rangle_{q(s_1)q(s_2)} \\
\log Z_x &\geq \left\langle \log \phi(s_1, s_2) \right\rangle_{q(s_1)q(s_2)} - \left\langle \log q(s_1)q(s_2) \right\rangle_{q(s_1)q(s_2)} \\
&\equiv -F(p; q) + H(q) \qquad (2)
\end{aligned}
$$

Here, F is the *energy* and H is the *entropy*. We need to maximize the right hand side.

$$\text{Evidence} \geq -\text{Energy} + \text{Entropy}$$

Note r.h.s. is a **lower bound** [7]. The mean field equations **monotonically** increase this bound. Good for assessing convergence and debugging computer code.

Details of derivation

- Define the Lagrangian

$$\begin{aligned}\Lambda &= \int ds_1 q(s_1)\log q(s_1) + \int ds_2 q(s_2)\log q(s_2) + \log Z_x - \int ds_1 ds_2 q(s_1)q(s_2)\log\phi(s_1,s_2) \\ &\quad +\lambda_1(1-\int ds_1 q(s_1)) + \lambda_2(1-\int ds_2 q(s_2)) \end{aligned} \tag{3}$$

- Calculate the functional derivatives w.r.t. $q(s_1)$ and set to zero

$$\frac{\delta}{\delta q(s_1)}\Lambda = \log q(s_1) + 1 - \langle\log\phi(s_1,s_2)\rangle_{q(s_2)} - \lambda_1$$

- Solve for $q(s_1)$,

$$\begin{aligned}\log q(s_1) &= \lambda_1 - 1 + \langle\log\phi(s_1,s_2)\rangle_{q(s_2)} \\ q(s_1) &= \exp(\lambda_1-1)\exp(\langle\log\phi(s_1,s_2)\rangle_{q(s_2)}) \end{aligned} \tag{4}$$

- Use the fact that

$$\begin{aligned}1 &= \int ds_1 q(s_1) = \exp(\lambda_1-1)\int ds_1 \exp(\langle\log\phi(s_1,s_2)\rangle_{q(s_2)}) \\ \lambda_1 &= 1 - \log\int ds_1 \exp(\langle\log\phi(s_1,s_2)\rangle_{q(s_2)})\end{aligned}$$

The form of the solution

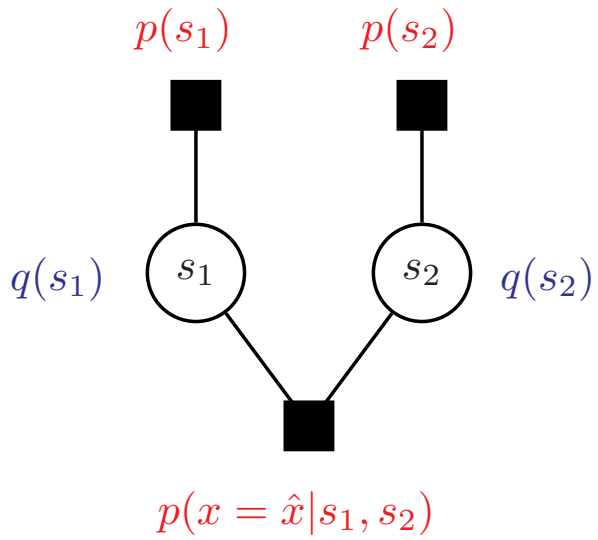
- No direct analytical solution
- We obtain fixed point equations in closed form

$$q(s_1) \propto \exp(\langle \log \phi(s_1, s_2) \rangle_{q(s_2)})$$

$$q(s_2) \propto \exp(\langle \log \phi(s_1, s_2) \rangle_{q(s_1)})$$

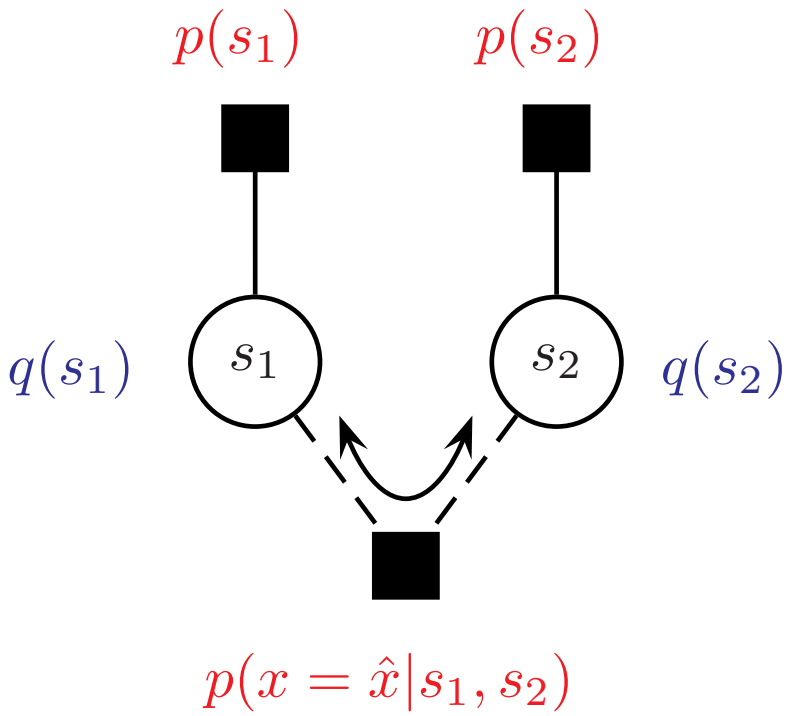
Note the nice symmetry

OSSS: Factor Graph



- A graphical representation of the inference problem
  - **Factor nodes**: Black squares. Factor potentials (local functions) defining the posterior $\mathcal{P}$.
  - **Variable nodes**: Circles. Think of them as "factors" of the approximating distribution $\mathcal{Q}$. (Caution – non standard interpretation!)
  - **Edges**: denote membership. A variable is connected to a factor if it is a variable of the local function.

Fixed Point Iteration for OSSS



$$\log q(s_1) \leftarrow \log p(s_1) + \langle \log p(x = \hat{x}|s_1, s_2) \rangle_{q(s_2)}$$

$$\log q(s_2) \leftarrow \log p(s_2) + \langle \log p(x = \hat{x}|s_1, s_2) \rangle_{q(s_1)}$$

Fixed Point Iteration for the Gaussian Case

$$\log q(s_1) \quad \leftarrow \quad -\frac{1}{2}\,\mathbf{Tr}\left(P_1^{-1}+R^{-1}\right)s_1 s_1^\top - s_1^\top R^{-1}\underbrace{\langle s_2\rangle_{q(s_2)}}_{=m_2} + \left(\mu_1^\top P_1^{-1}+\hat{x}^\top R^{-1}\right)s_1$$

$$\log q(s_2) \quad \leftarrow \quad -\underbrace{\langle s_1\rangle_{q(s_1)}^\top}_{=m_1^\top} R^{-1}s_2 - \frac{1}{2}\,\mathbf{Tr}\left(P_2^{-1}+R^{-1}\right)s_2 s_2^\top + \left(\mu_2^\top P_2^{-1}+\hat{x}^\top R^{-1}\right)s_2$$

Remember $q(s)=\mathcal{N}(s;m,S)$

$$\log q(s) \quad =^+ \quad -\frac{1}{2}\,\mathbf{Tr}\,Kss^\top + h^\top s$$

$$\Downarrow$$

$$S=K^{-1} \qquad\qquad m=K^{-1}h$$

Fixed Point Equations for the Gaussian Case

- Covariances are obtained directly

$$S_1 = \left(P_1^{-1} + R^{-1}\right)^{-1} \qquad S_2 = \left(P_2^{-1} + R^{-1}\right)^{-1}$$

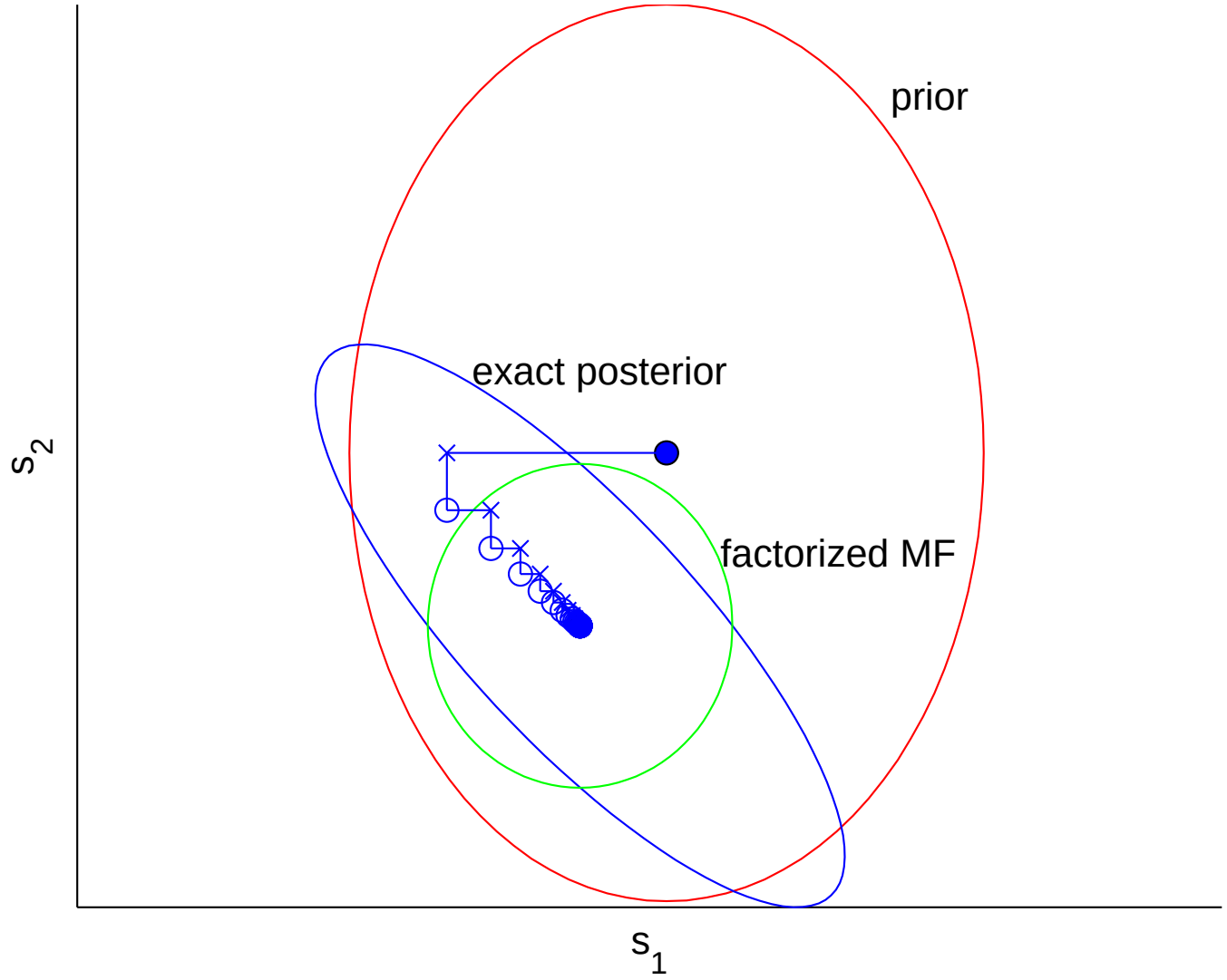
- To compute the means, we should iterate:

$$\begin{aligned} m_1 &= S_1\left(P_1^{-1}\mu_1 + R^{-1}\left(\hat{x} - m_2\right)\right) \\ m_2 &= S_2\left(P_2^{-1}\mu_2 + R^{-1}\left(\hat{x} - m_1\right)\right) \end{aligned}$$

- Intuitive algorithm:
  - Substract from the observation $\hat{x}$ the prediction of the other factors of $\mathcal{Q}$.
  - Compute a fit to this residual (e.g. "fit" m_2 to $\hat{x} - m_1$).

- Equivalent to Gauss-Seidel, an iterative method for solving linear systems of equations.

OSSS example, cont.



Direct Link to Expectation-Maximisation (EM) [3]

Suppose we choose one of the distributions degenerate, i.e.

$$\tilde{q}(s_2) \quad = \quad \delta(s_2 - \tilde{m})$$

where $\tilde{m}$ corresponds to the "location parameter" of $\tilde{q}(s_2)$. We need to find the closest degenerate distribution to the actual mean field solution $q(s_2)$, hence we take one more KL and minimize

$$\tilde{m} \quad = \quad \underset{\xi}{\operatorname{argmin}}\, KL(\delta(s_2 - \xi)||q(s_2))$$

It can be shown that this leads exactly to the EM fixed point iterations.

Iterated Conditional Modes (ICM) [1, 2]

If we choose both distributions degenerate, i.e.

$$\begin{aligned} \tilde{q}(s_1) &= \delta(s_1 - \tilde{m}_1) \\ \tilde{q}(s_2) &= \delta(s_2 - \tilde{m}_2) \end{aligned}$$

It can be shown that this leads exactly to the ICM fixed point iterations. This algorithm is equivalent to coordinate ascent in the original posterior surface $\phi(s_1, s_2)$.

$$\begin{aligned} \tilde{m}_1 &= \operatorname*{argmax}_{s_1} \phi(s_1, s_2 = \tilde{m}_2) \\ \tilde{m}_2 &= \operatorname*{argmax}_{s_2} \phi(s_1 = \tilde{m}_1, s_2) \end{aligned}$$

ICM, EM, VB ...

For OSSS, all algorithms are identical. This is in general not true.

While algorithmic details are very similar, there can be big qualitative differences in terms of fixed points.

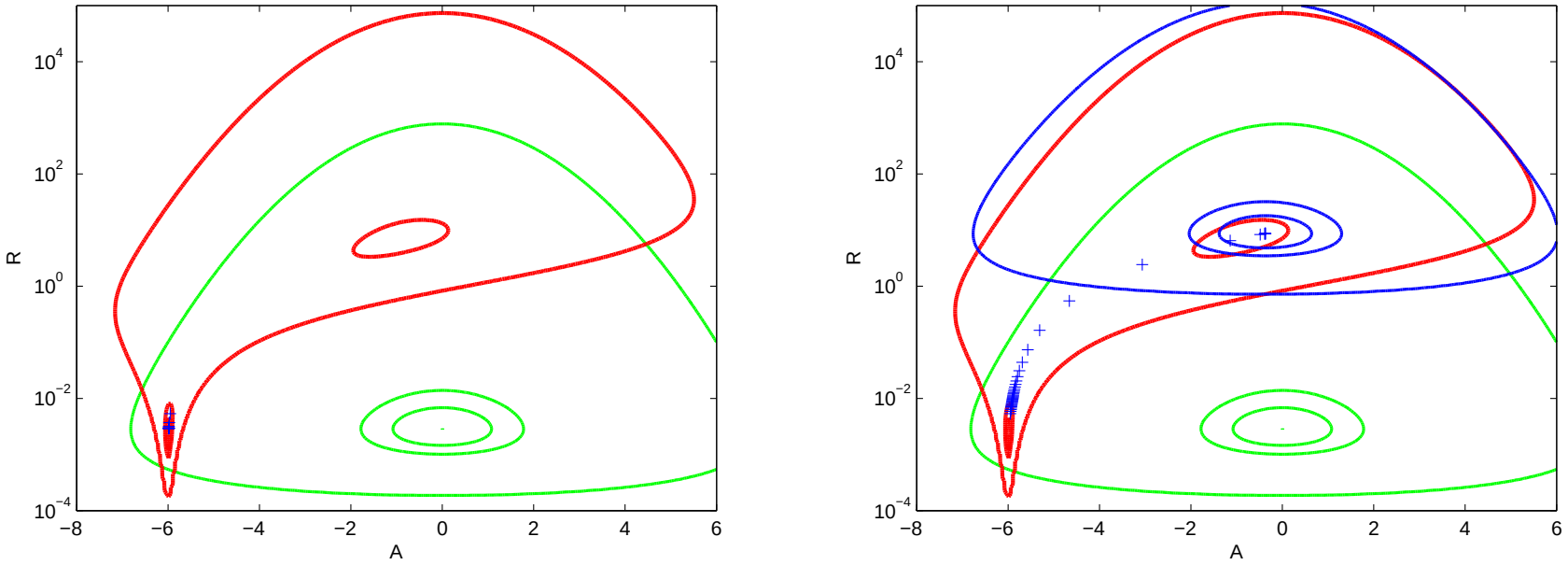


Figure 1: Left, ICM, Right VB. EM is similar to ICM in this AR(1) example.

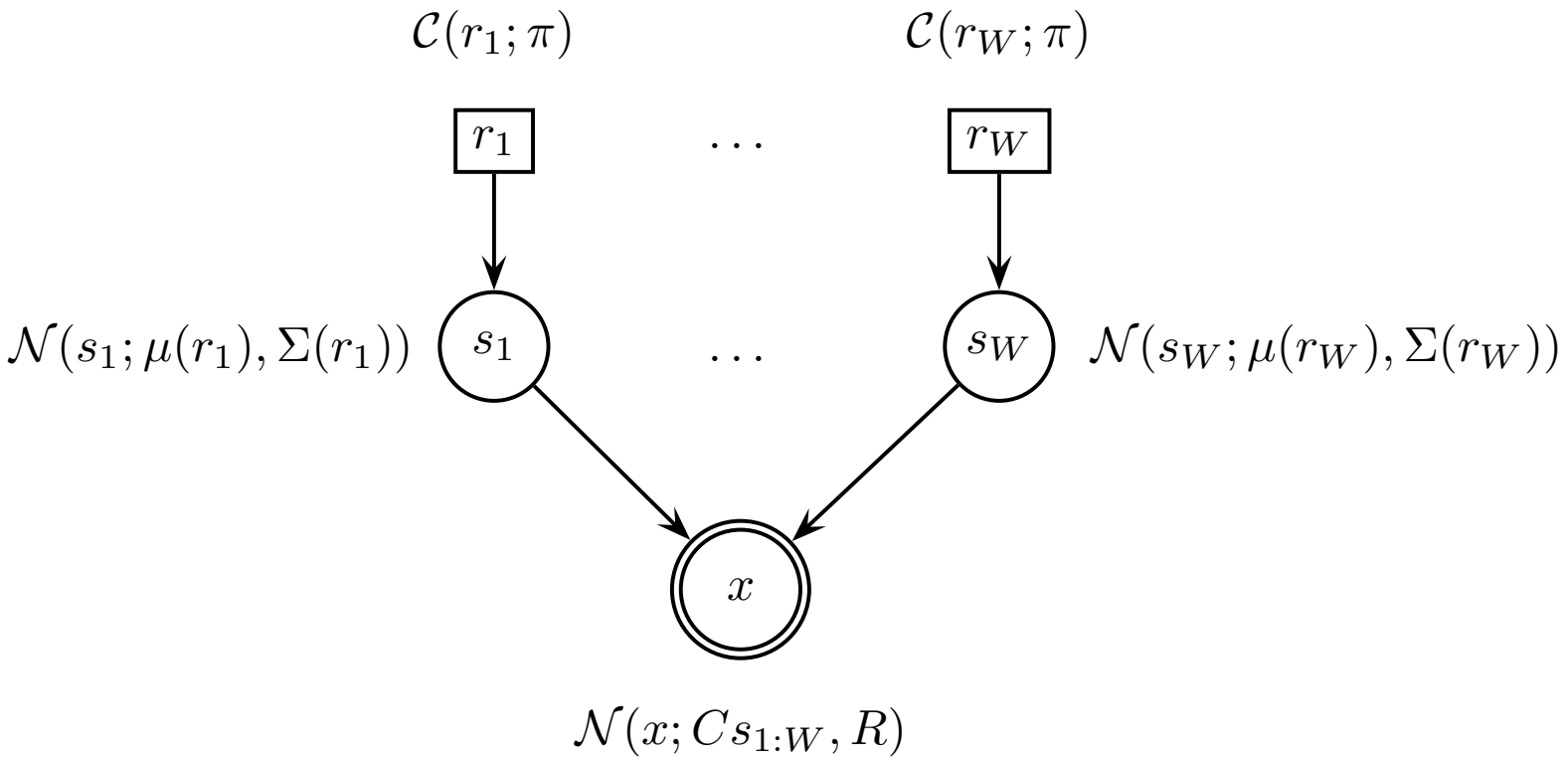
Structured Mean Field

Main Idea

- Identify tractable substructures to construct richer approximating distributions
- Tradeoff between approximation quality and computation time

The OSSS model is too simple; a richer approximation $\mathcal{Q}(s_1, s_2)$ would be equivalent to the exact posterior.

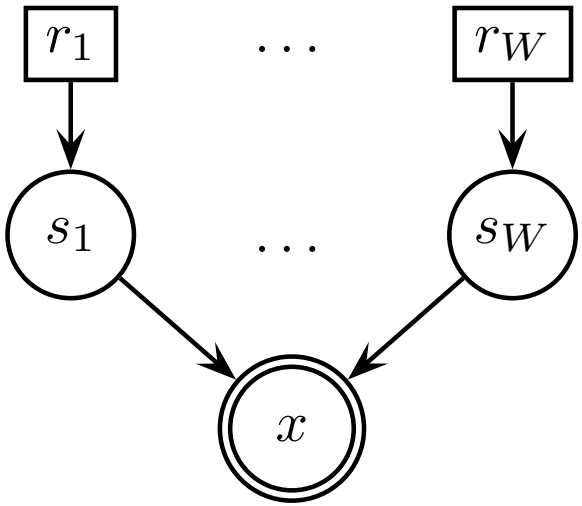
Bayesian Variable Selection



- Generalized Linear Model – Column's of C are the basis vectors
- The exact posterior is a mixture of 2^W Gaussians
- When W is large, computation of posterior features becomes intractable.

Generative model

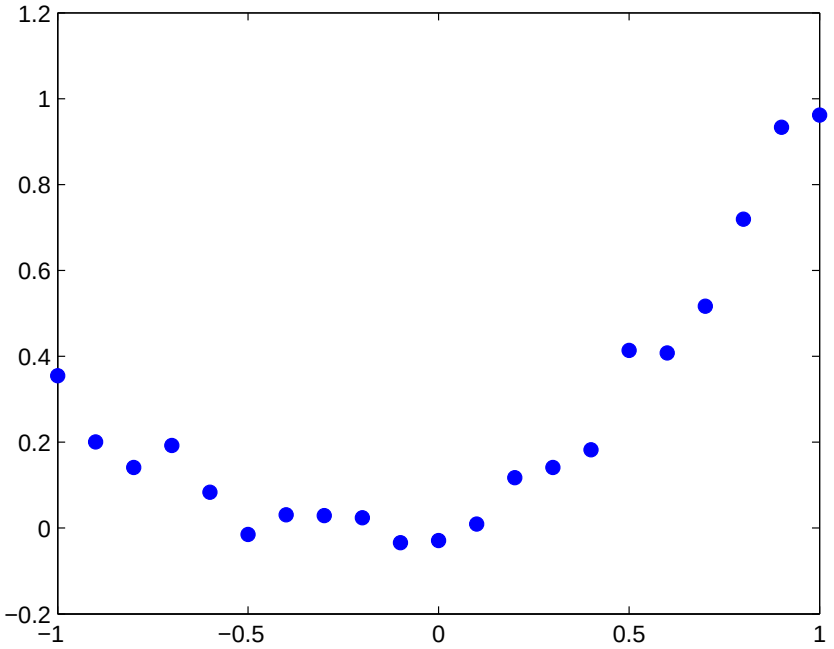
$$
\begin{aligned}
r_i &\sim \mathcal{C}(r_i;\pi) \\
s_i|r_i &\sim \mathcal{N}(s_i;\mu(r_i),\Sigma(r_i)) \\
\mathbf{x}|s_{1:W} &\sim \mathcal{N}(\mathbf{x};Cs_{1:W},R) \\
C &\equiv \begin{bmatrix} C_1 & \dots & C_i & \dots & C_W \end{bmatrix}
\end{aligned}
$$



$$
p(\mathbf{x}, s_{1:W}, r_{1:W}) = p(\mathbf{x}|s_{1:W}, r_{1:W}) \prod_{i=1}^{W} p(s_i|r_i)p(r_i)
$$

Example 1: Variable selection in Polynomial Regression

Given $\{t_j, x(t_j)\}_{j=1\ldots J}$, what is the order N of the polynomial?



$$x(t) = \sum_{i=0}^{N} s_{i+1} t^i + \epsilon(t)$$

Ex1: Regression

$$\begin{aligned} \mathbf{t} &= \begin{pmatrix} t_1 & t_2 & \dots & t_J \end{pmatrix}^\top \\ C &\equiv \begin{pmatrix} \mathbf{t}^0 & \mathbf{t}^1 & \dots & \mathbf{t}^{W-1} \end{pmatrix} \end{aligned}$$

$$\begin{aligned} r_i &\sim \mathcal{C}(r_i; 0.5, 0.5) \qquad r_i \in \{\text{on}, \text{off}\} \\ s_i | r_i &\sim \mathcal{N}(s_i; 0, \Sigma(r_i)) \\ \mathbf{x} | s_{1:W} &\sim \mathcal{N}(\mathbf{x}; C s_{1:W}, R) \end{aligned}$$

$$\Sigma(r_i = \text{on}) \gg \Sigma(r_i = \text{off})$$

Ex1: Regression

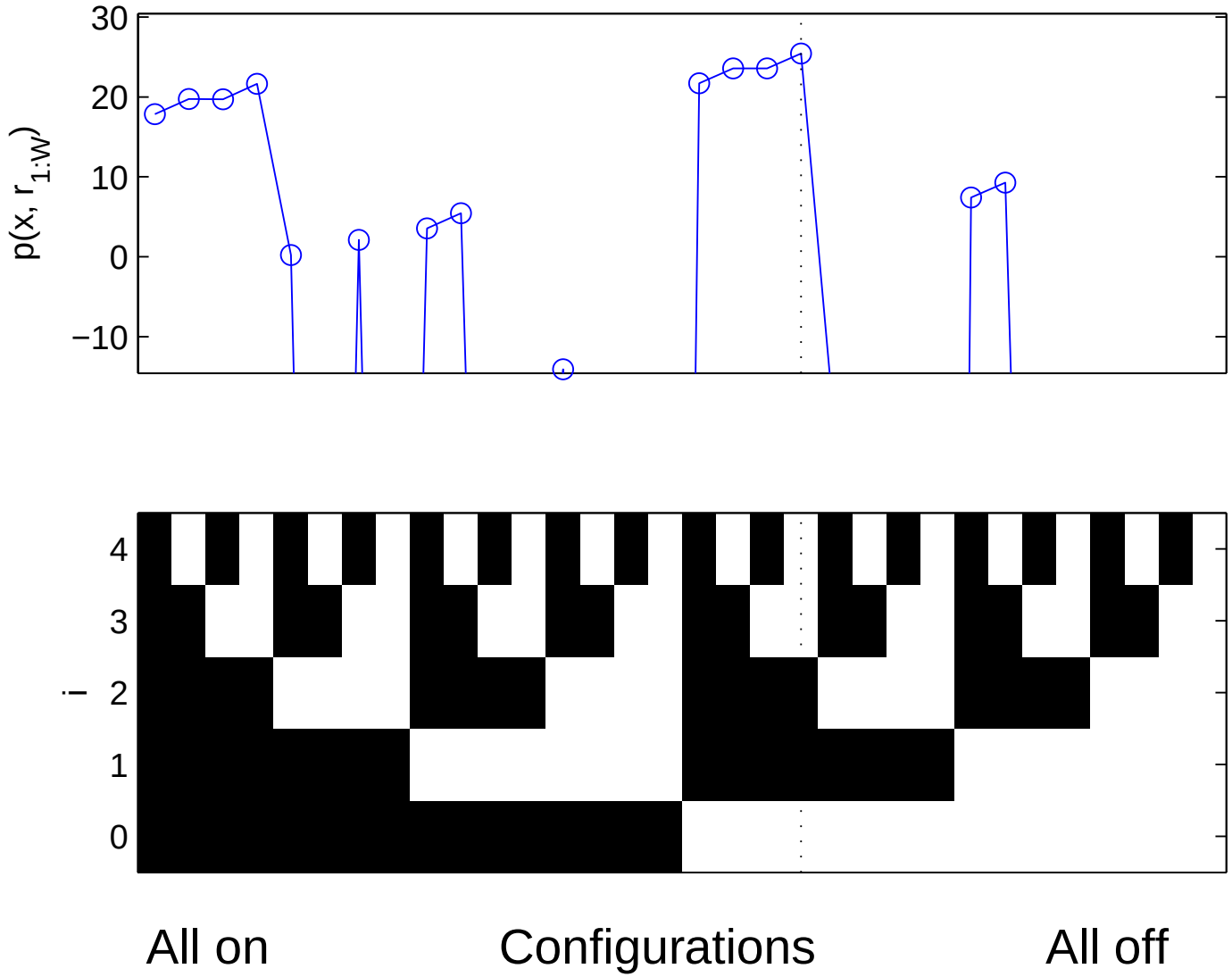
To find the “active” basis functions we need to calculate

$$r^*_{1:W} \equiv \underset{r_{1:W}}{\operatorname{argmax}}\, p(r_{1:W}|\mathbf{x}) \quad = \quad \underset{r_{1:W}}{\operatorname{argmax}} \int ds_{1:W} p(\mathbf{x}|s_{1:W}) p(s_{1:W}|r_{1:W}) p(r_{1:W})$$

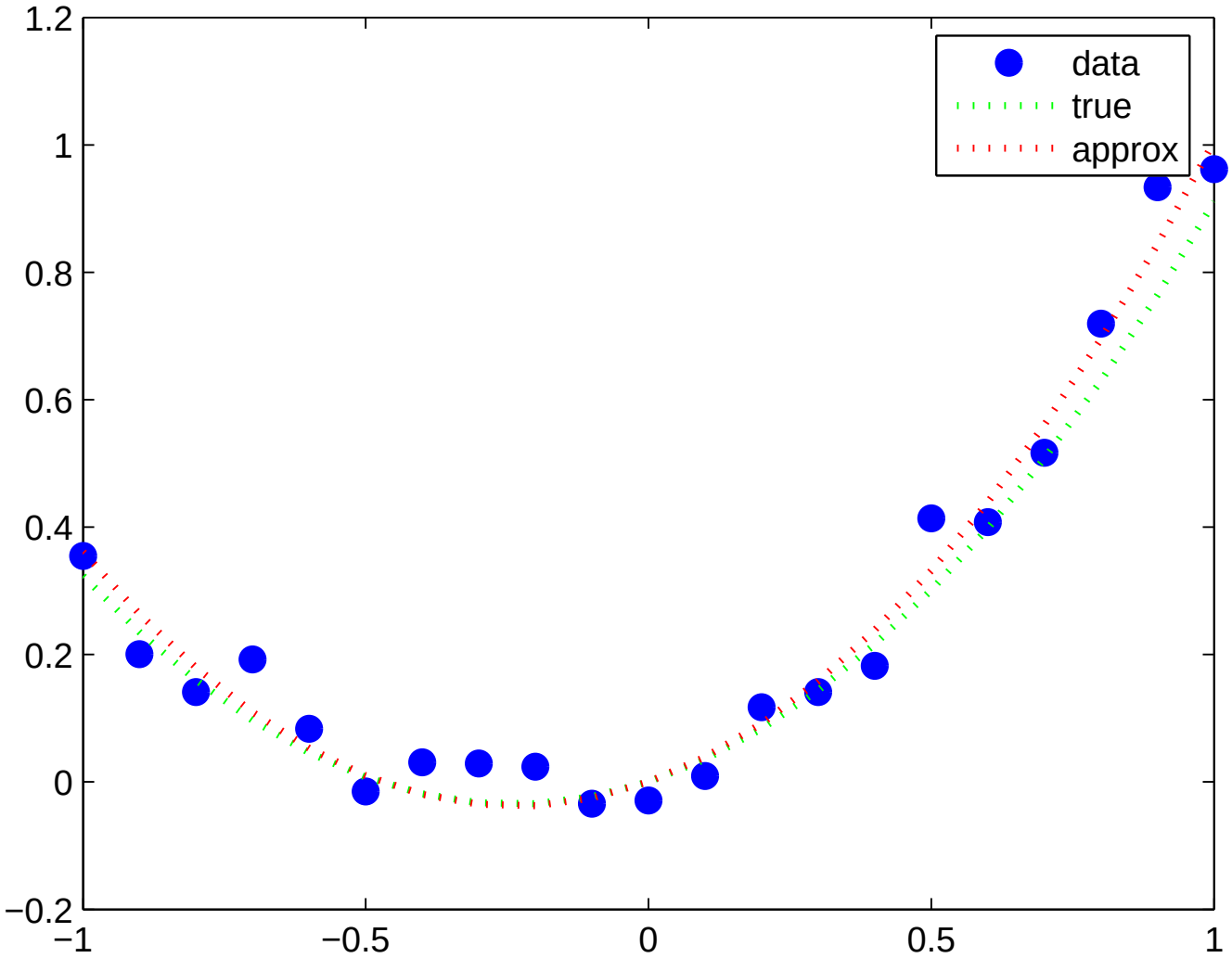
Then, the reconstruction is given by

$$\begin{aligned} \hat{x}(t) &= \left\langle \sum_{i=0}^{W-1} s_{i+1} t^i \right\rangle_{p(s_{1:W}|\mathbf{x}, r^*_{1:W})} \\ &= \sum_{i=0}^{W-1} \left\langle s_{i+1} \right\rangle_{p(s_{i+1}|\mathbf{x}, r^*_{1:W})} t^i \end{aligned}$$

Ex1: Regression

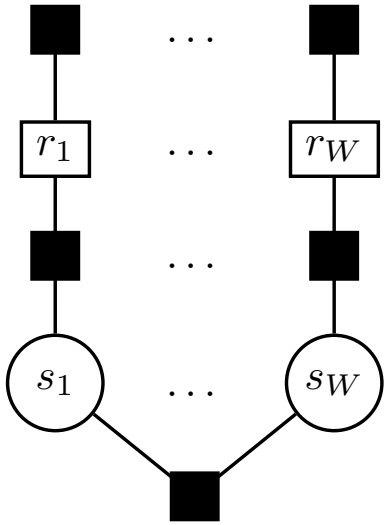


Ex1: Regression

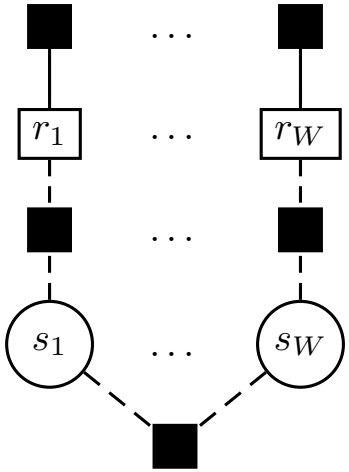


Factor graph

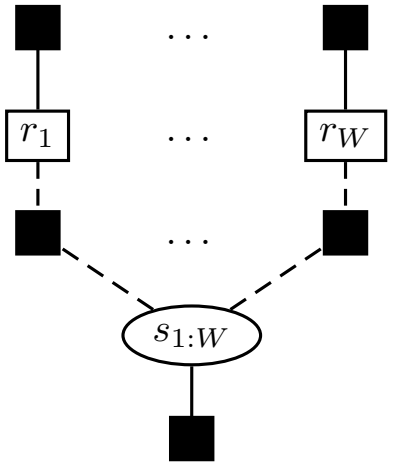
$$
\begin{aligned}
\log \phi(r_{1:W}, s_{1:W}) \quad = \quad & \sum_{i=1}^{W} (\log \pi(r_i)) \\
& + \sum_{i=1}^{W} \Big(-\frac{1}{2} s_i^\top \Sigma(r_i)^{-1} s_i + \mu(r_i)^\top \Sigma(r_i)^{-1} s_i \\
& \quad -\frac{1}{2} \mu(r_i)^\top \Sigma(r_i)^{-1} \mu(r_i) - \frac{1}{2} \log |2\pi\Sigma(r_i)| \Big) \\
& -\frac{1}{2} \mathbf{x}^\top R^{-1} \mathbf{x} + s_{1:W}^\top C^\top R^{-1} \mathbf{x} - \frac{1}{2} s_{1:W}^\top C^\top R^{-1} C s_{1:W} - \frac{1}{2} \log |2\pi R|
\end{aligned}
$$



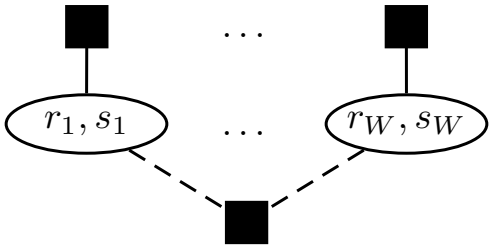
Approximating Distributions



$$\mathcal{Q}_1 = \prod_{i=1}^{W} \mathcal{Q}(s_i)\mathcal{Q}(r_i)$$



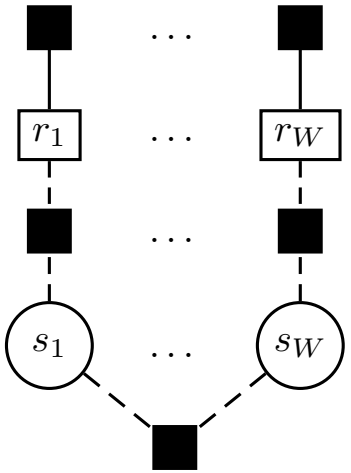
$$\mathcal{Q}_2 = \mathcal{Q}(s_{1:W}) \prod_{i=1}^{W} \mathcal{Q}(r_i)$$



$$\mathcal{Q}_3 = \prod_{i=1}^{W} \mathcal{Q}(s_i, r_i)$$

Update Equations, $\mathcal{Q}_1 = \prod_{i=1}^{W} \mathcal{Q}(s_i)\mathcal{Q}(r_i)$

$$\log \mathcal{Q}(r_i) \quad =^{+} \quad \log \pi(r_i) - \frac{1}{2}(\langle s_i \rangle - \mu(r_i))^{\top} \Sigma(r_i)^{-1} (\langle s_i \rangle - \mu(r_i))$$



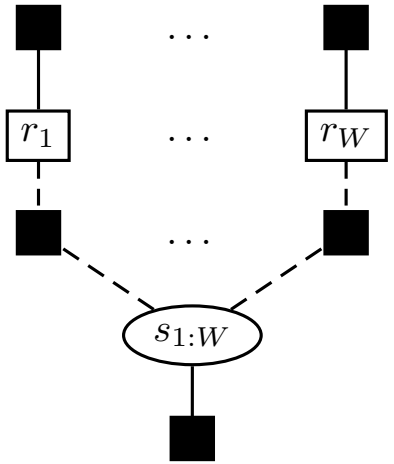
$$\log \mathcal{Q}(s_i) =^{+} \left(\left\langle \Sigma(r_i)^{-1} \mu(r_i) \right\rangle + C_i^{\top} R^{-1} (\mathbf{x} - C_{\neg i} \langle s_{\neg i} \rangle) \right)^{\top} s_i - \frac{1}{2} s_i^{\top} \left(\left\langle \Sigma(r_i)^{-1} \right\rangle + C_i^{\top} R^{-1} C_i \right) s_i$$

$$C_{\neg i} \equiv \begin{pmatrix} C_1 & \dots & C_{i-1} & C_{i+1} & \dots & C_W \end{pmatrix}$$

$$s_{\neg i} \equiv \begin{pmatrix} s_1^{\top} & \dots & s_{i-1}^{\top} & s_{i+1}^{\top} & \dots & s_W^{\top} \end{pmatrix}^{\top}$$

Update Equations: $\mathcal{Q}_2 = \mathcal{Q}(s_{1:W}) \prod_{i=1}^{W} \mathcal{Q}(r_i)$

$$\log \mathcal{Q}(r_i) \quad =^{+} \quad \log \pi(r_i) - \frac{1}{2}(\langle s_i \rangle - \mu(r_i))^{\top} \Sigma(r_i)^{-1} (\langle s_i \rangle - \mu(r_i))$$

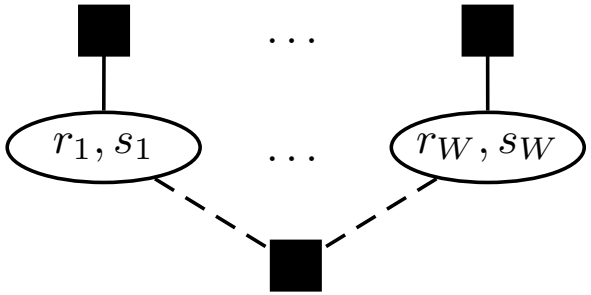


$$\log \mathcal{Q}(s_{1:W}) \quad =^{+} \quad \left(\left\langle \Sigma(\mathbf{r})^{-1} \mu(\mathbf{r}) \right\rangle + C^{\top} R^{-1} \mathbf{x} \right)^{\top} s_{1:W} - \frac{1}{2} s_{1:W}^{\top} \left(\left\langle \Sigma(\mathbf{r})^{-1} \right\rangle + C^{\top} R^{-1} C \right) s_{1:W}$$

$$\Sigma(\mathbf{r})^{-1} \equiv \begin{pmatrix} \Sigma(r_1)^{-1} & & \\ & \ddots & \\ & & \Sigma(r_W)^{-1} \end{pmatrix} \qquad \mu(\mathbf{r}) \equiv \begin{pmatrix} \mu(r_1) \\ \vdots \\ \mu(r_W) \end{pmatrix}$$

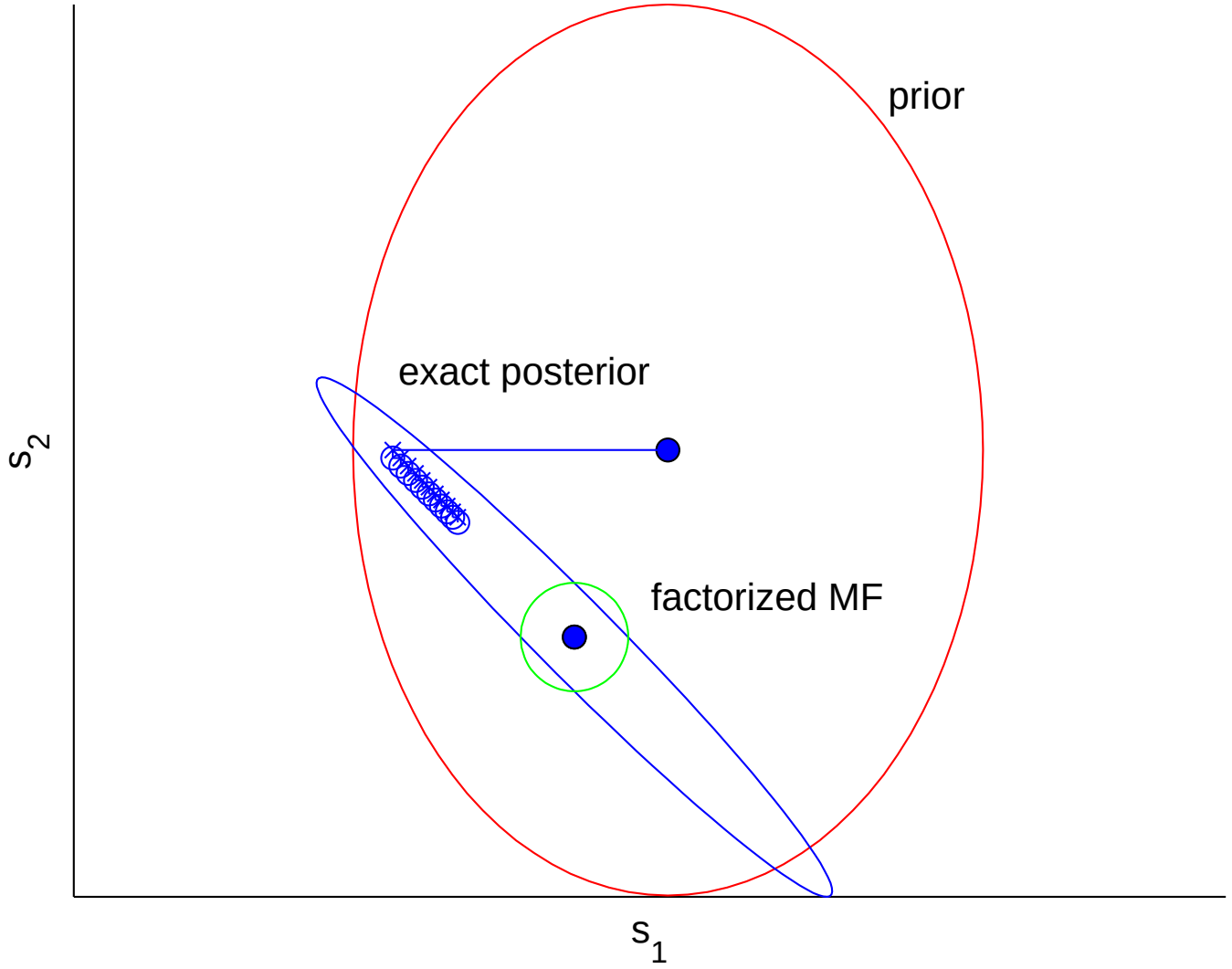
Update Equations: $\mathcal{Q}_3 = \prod_{i=1}^{W} \mathcal{Q}(r_i, s_i)$

Left as an exercise to the interested reader...



Convergence Issues

OSSS example, Slow Convergence



Annealing, Bridging, Relaxation, Tempering

Main idea:

- If the original target $\mathcal{P}$ is too complex, relax it.
- First solve a simple version $\mathcal{P}_{\tau_1}$. Call the solution m_{τ_1}
- Make the problem little bit harder $\mathcal{P}_{\tau_1} \rightarrow \mathcal{P}_{\tau_2}$, and improve the solution $m_{\tau_1} \rightarrow m_{\tau_2}$.
- While $\mathcal{P}_{\tau_1} \rightarrow \mathcal{P}_{\tau_2}, \ldots, \rightarrow \mathcal{P}_T = \mathcal{P}$, we hope to get better and better solutions.

The sequence $\tau_1, \tau_2, \ldots, \tau_T$ is called annealing schedule if

$$\mathcal{P}_{\tau_i} \quad \propto \quad \mathcal{P}^{\tau_i}$$

OSSS example: Annealing, Bridging, ...

- Remember the cross term $(*)$ of the posterior:

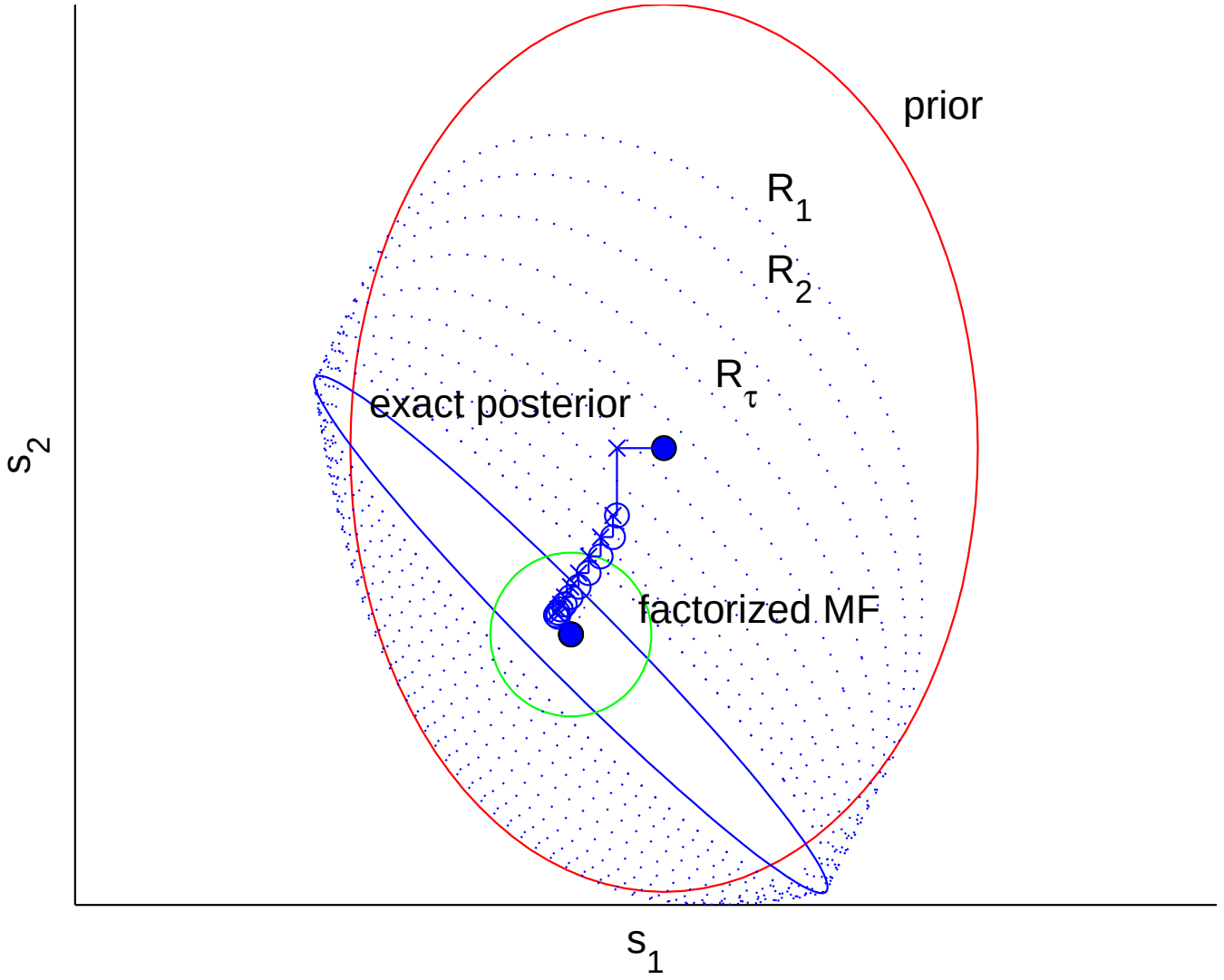
$$\cdots - \underbrace{s_1^\top R^{-1} s_2}_{(*)} \cdots$$

- When the noise variance is low, the coupling is strong.
- If we choose a decreasing sequence of noise covariances

$$R_{\tau_1} > R_{\tau_2} > \cdots > R_{\tau_T} = R$$

we increase correlations gradually.

OSSS example: Annealing, Bridging, ...



Stochastic Inference

Deterministic versus Stochastic

Let θ denote the parameter vector of $\mathcal{Q}$.

- Given the fixed point equation F and an initial parameter $\theta^{(0)}$, the inference algorithm is simply

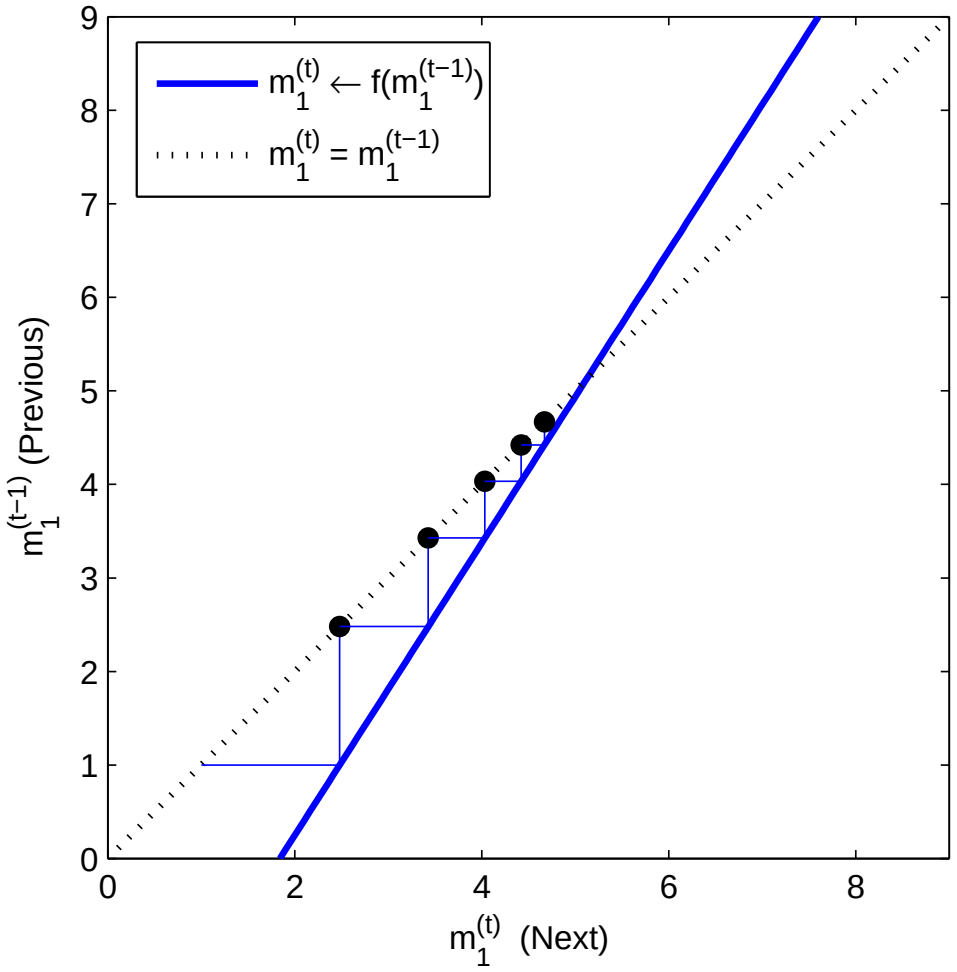
$$\theta^{(t+1)} \leftarrow F(\theta^{(t)})$$

For OSSS $\theta = (m_1, m_2)^\top$ (S_1, S_2 were constant, so we exclude them). The update equations were

$$\begin{aligned} m_1^{(t+1)} &\leftarrow F_1(m_2^{(t)}) \\ m_2^{(t+1)} &\leftarrow F_2(m_1^{(t+1)}) \end{aligned}$$

This is a deterministic dynamical system in the parameter space.

OSSS: Fixed Point iteration for m_1



Stochastic Inference

Stochastic inference is similar, but everything happens directly in the configuration space (= domain) of variables $\mathbf{s}$.

- Given a transition kernel T (=a collection of probability distributions conditioned on each $\mathbf{s}$) and an initial configuration $\mathbf{s}^{(0)}$

$$\mathbf{s}^{(t+1)} \quad \sim \quad T(\mathbf{s}|\mathbf{s}^{(t)}) \qquad\qquad t = 1, \ldots, \infty$$

- This is a stochastic dynamical system in the configuration space.
- A remarkable fact is that we can estimate any desired expectation by ergodic averages

$$\langle f(\mathbf{s}) \rangle_{\mathcal{P}} \approx \frac{1}{t - t_0} \sum_{n=t_0}^{t} f(\mathbf{s}^{(n)})$$

- Consecutive samples $\mathbf{s}^{(t)}$ are dependent but we can "pretend" as if they are independent!

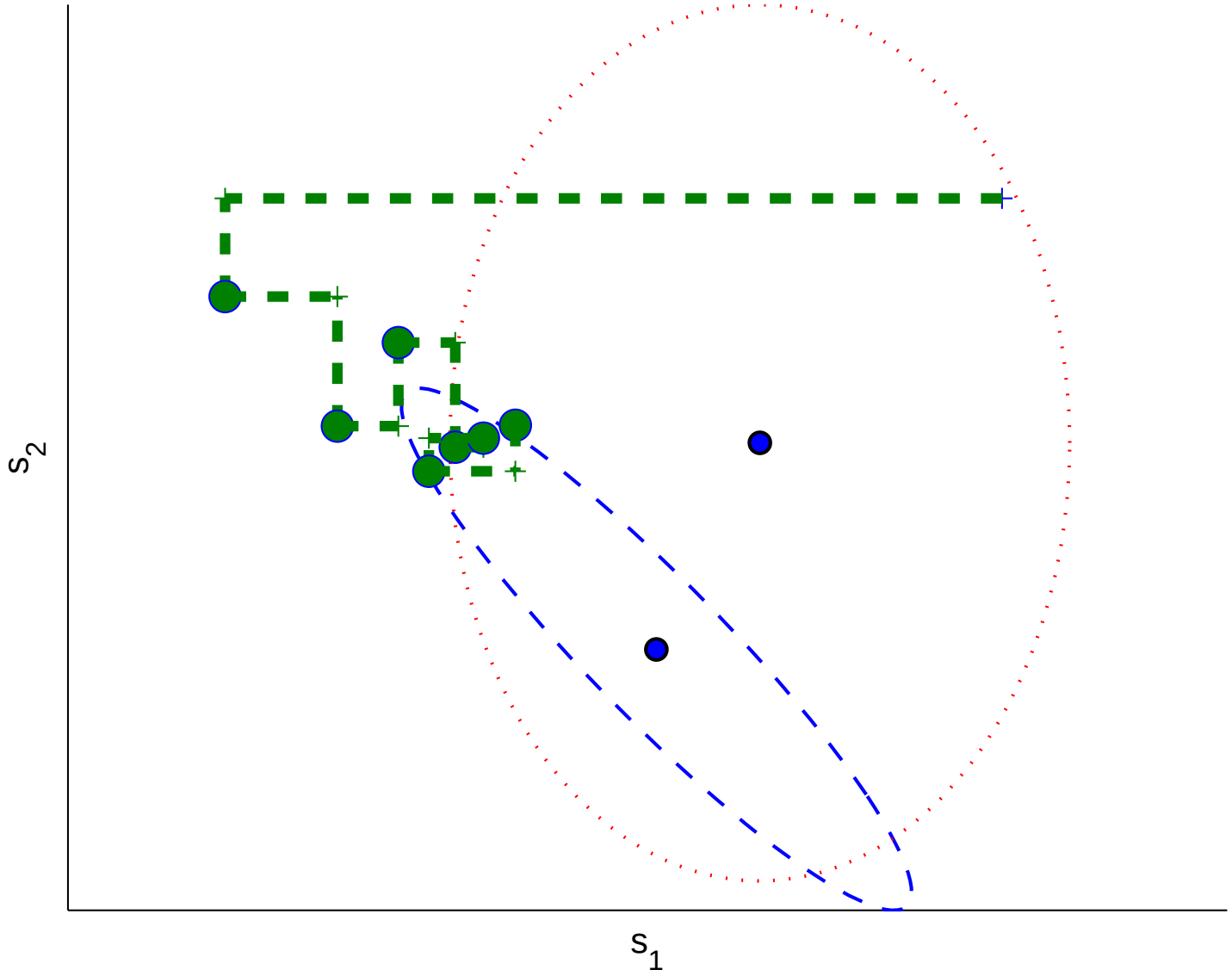
Looking ahead...

- For OSSS, the configuration space is $\mathbf{s} = (s_1, s_2)^\top$.
- A possible transition kernel T is specified by

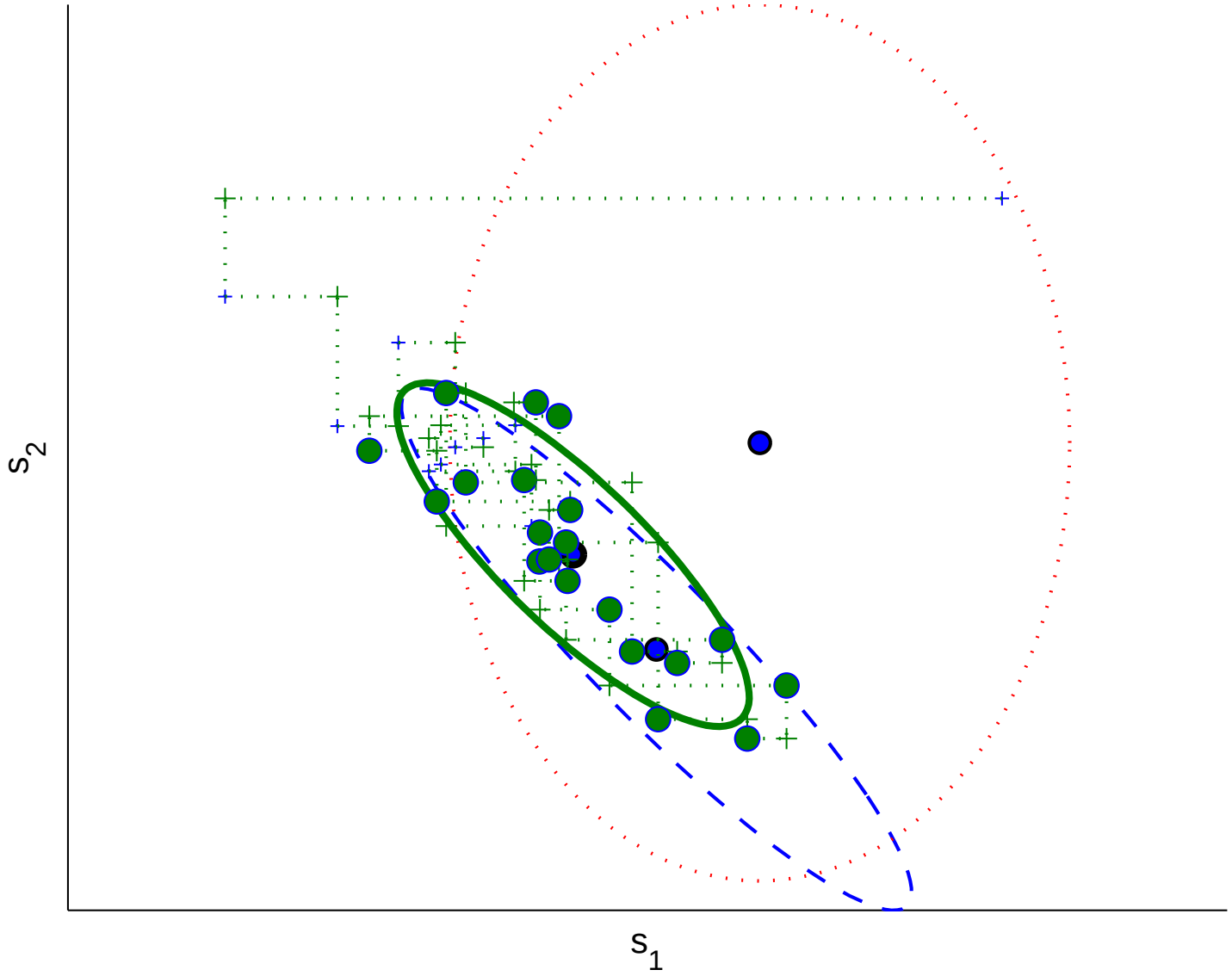
$$\begin{aligned} s_1^{(t+1)} &\sim p(s_1 | s_2^{(t)}, x = \hat{x}) &&\propto \phi(s_1, s_2^{(t)}) \\ s_2^{(t+1)} &\sim p(s_2 | s_1^{(t+1)}, x = \hat{x}) &&\propto \phi(s_1^{(t+1)}, s_2) \end{aligned}$$

- This algorithm, that samples from above conditional marginals is a particular instance of the **Gibbs sampler**.
- The desired posterior $\mathcal{P}$ is the stationary distribution of T (why? – later...).
- Note the algorithmic similarity to ICM. In Gibbs, we make a random move instead of directly going to the conditional mode.

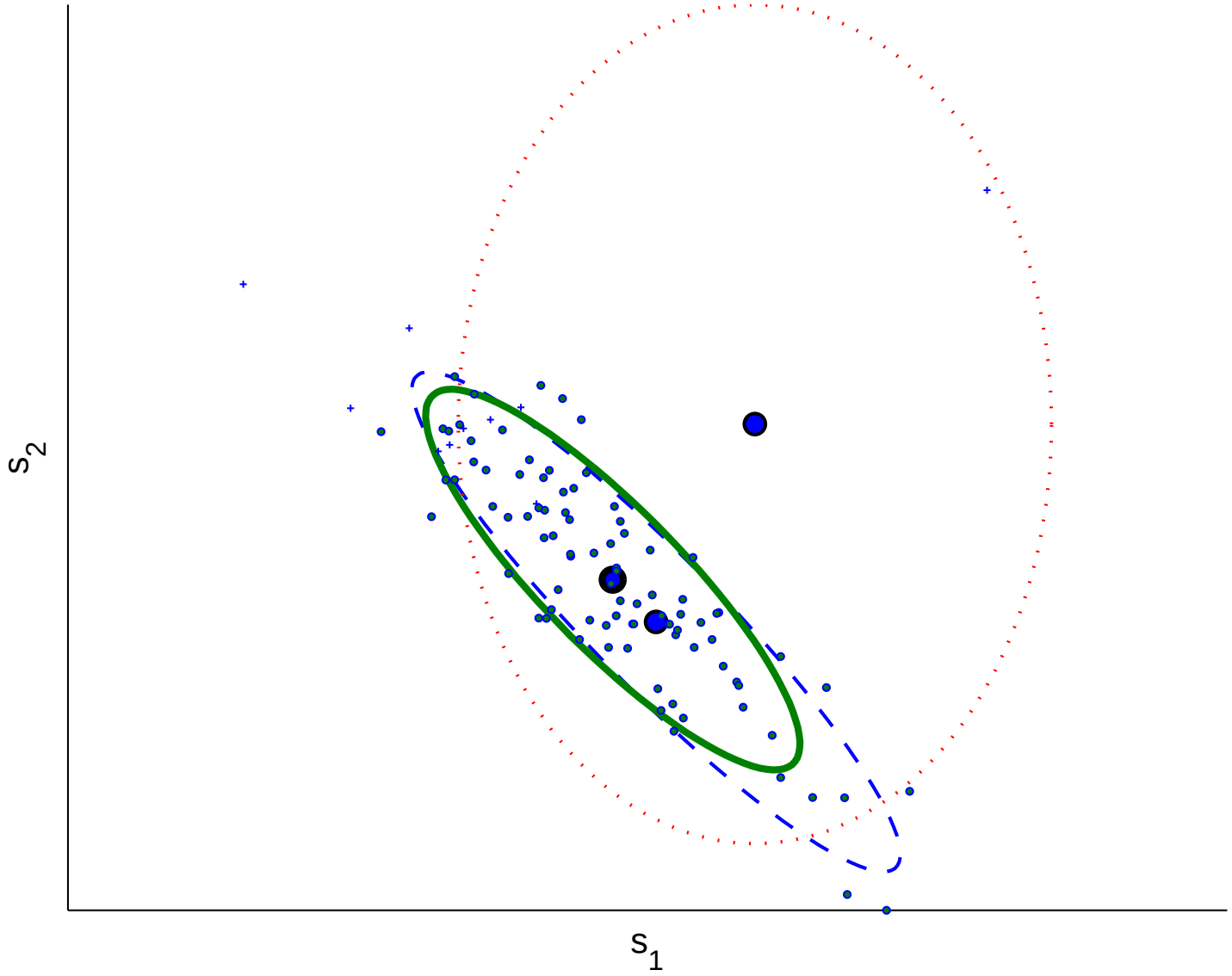
Gibbs Sampling



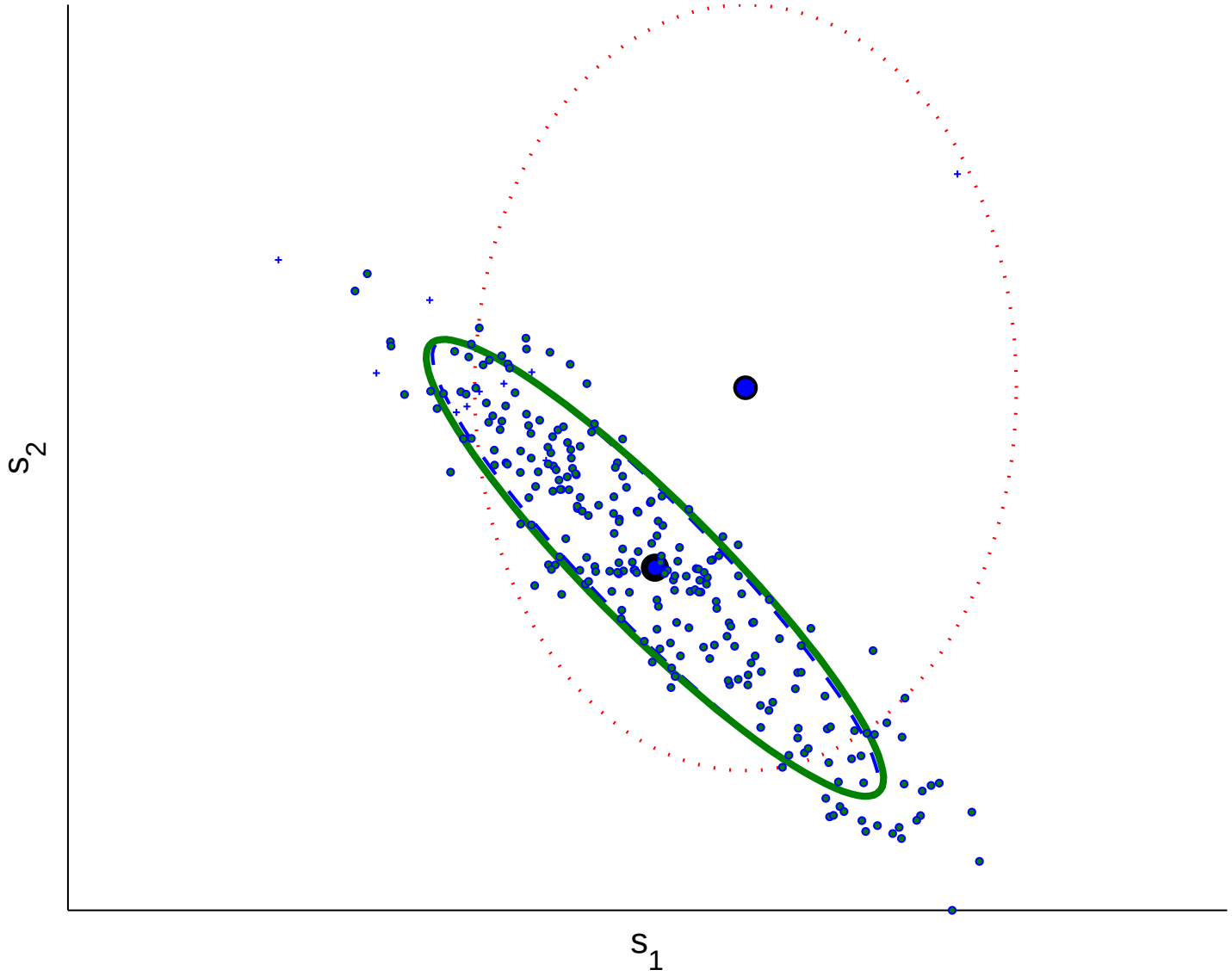
Gibbs Sampling, $t = 20$



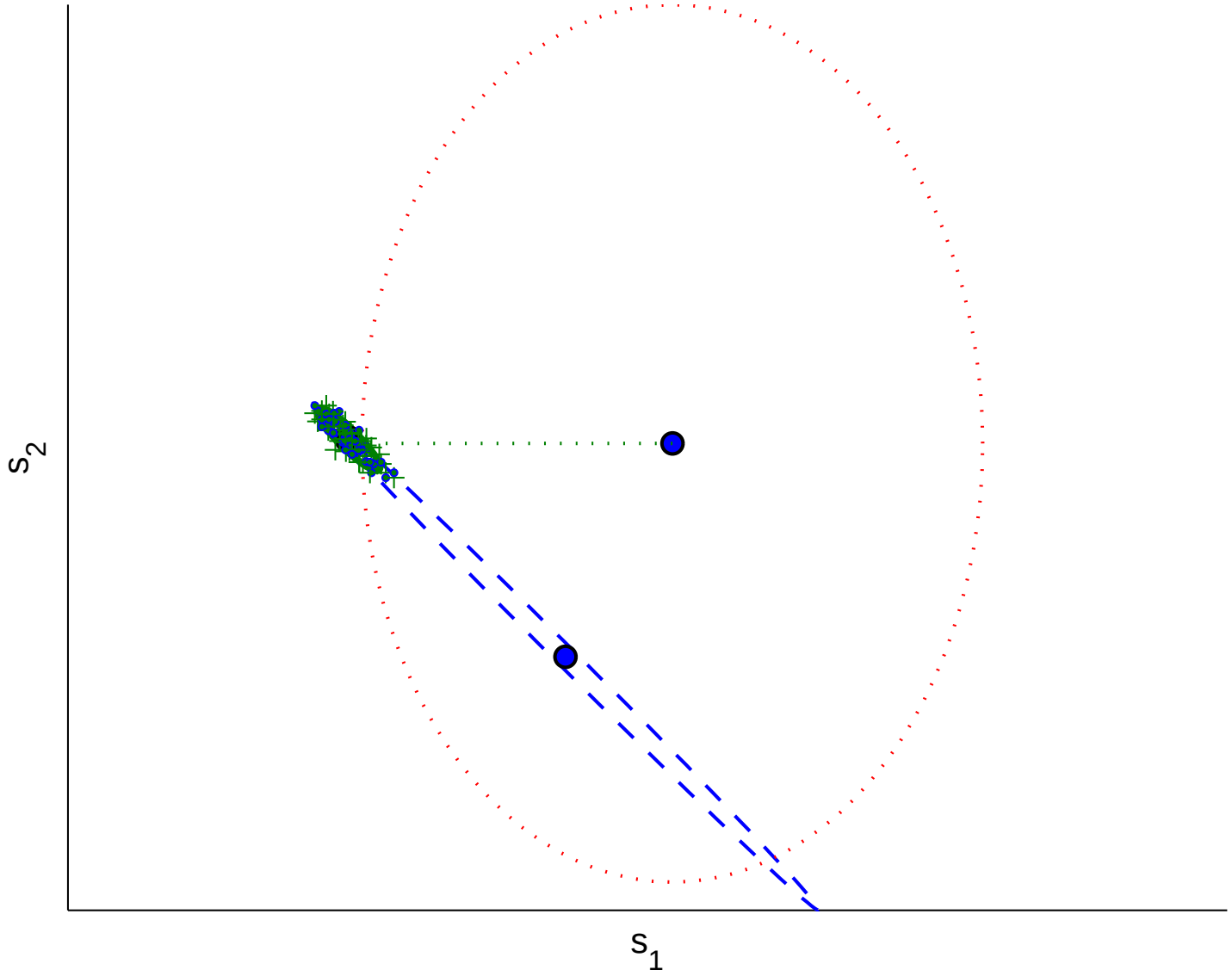
Gibbs Sampling, $t = 100$



Gibbs Sampling, $t = 250$



Gibbs Sampling, Slow convergence



Markov Chain Monte Carlo (MCMC)

- Construct a transition kernel $T(\mathbf{s}'|\mathbf{s})$ with the stationary distribution $\mathcal{P} = \phi(\mathbf{s})/Z_x \equiv \pi(\mathbf{s})$ for any initial distribution $r(\mathbf{s})$.

$$\pi(\mathbf{s}) = T^{\infty} r(\mathbf{s}) \tag{5}$$

- Sample $\mathbf{s}^{(0)} \sim r(\mathbf{s})$
- For $t = 1 \ldots \infty$, Sample $\mathbf{s}^{(t)} \sim T(\mathbf{s}|\mathbf{s}^{(t-1)})$
- Estimate any desired expectation by the average

$$\langle f(\mathbf{s}) \rangle_{\pi(\mathbf{s})} \approx \frac{1}{t - t_0} \sum_{n=t_0}^{t} f(\mathbf{s}^{(n)})$$

  where t_0 is a preset burn-in period.

But how to construct T and verify that $\pi(\mathbf{s})$ is indeed its stationary distribution ?

Equilibrium condition = Detailed Balance

$$T(\mathbf{s}|\mathbf{s}')\pi(\mathbf{s}') \quad = \quad T(\mathbf{s}'|\mathbf{s})\pi(\mathbf{s})$$

If detailed balance is satisfied then $\pi(\mathbf{s})$ is a stationary distribution

$$\pi(\mathbf{s}) \quad = \quad \int d\mathbf{s}' T(\mathbf{s}|\mathbf{s}')\pi(\mathbf{s}')$$

If the configuration space is discrete, we have

$$\begin{aligned} \pi(\mathbf{s}) &= \sum_{\mathbf{s}'} T(\mathbf{s}|\mathbf{s}')\pi(\mathbf{s}') \\ \pi &= T\pi \end{aligned}$$

π has to be a (right) eigenvector of T.

Conditions on T

- Irreducibility (probabilisic connectedness): Every state s' can be reached from every s

$$T(s'|s) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$ is **not** irreducible

- Aperiodicity : Cycling around is not allowed

$$T(s'|s) = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$ is **not** aperiodic

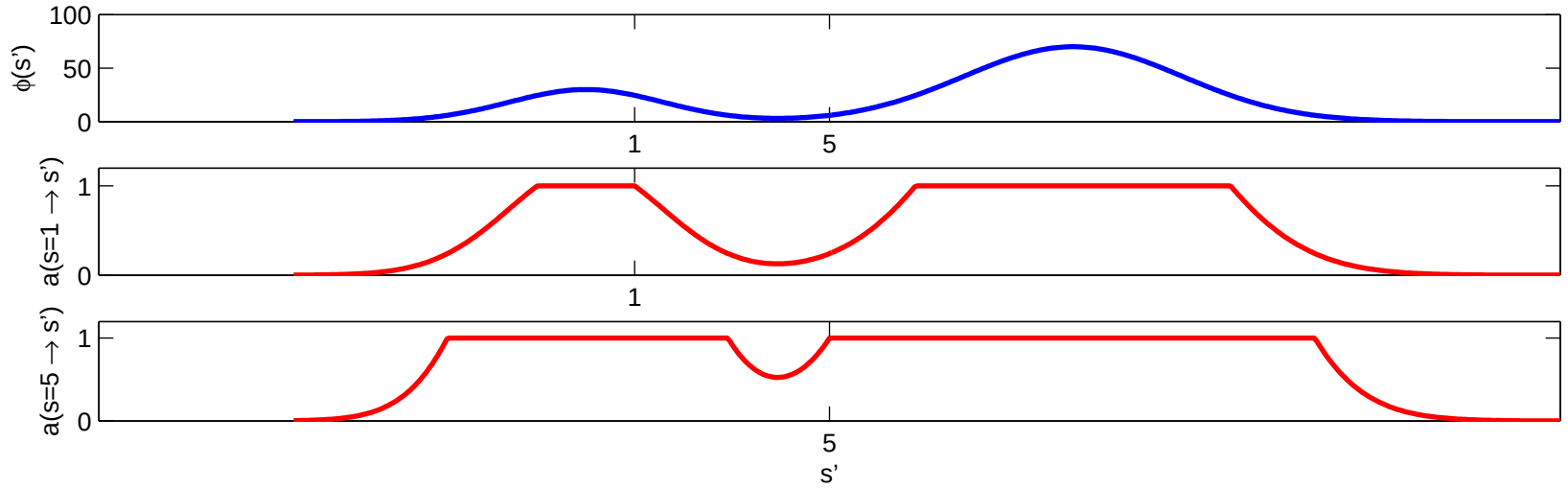
Surprisingly, it is easy to construct a transition kernel with these properties by following the recipe provided by Metropolis (1953) and Hastings (1970).

Metropolis-Hastings Kernel

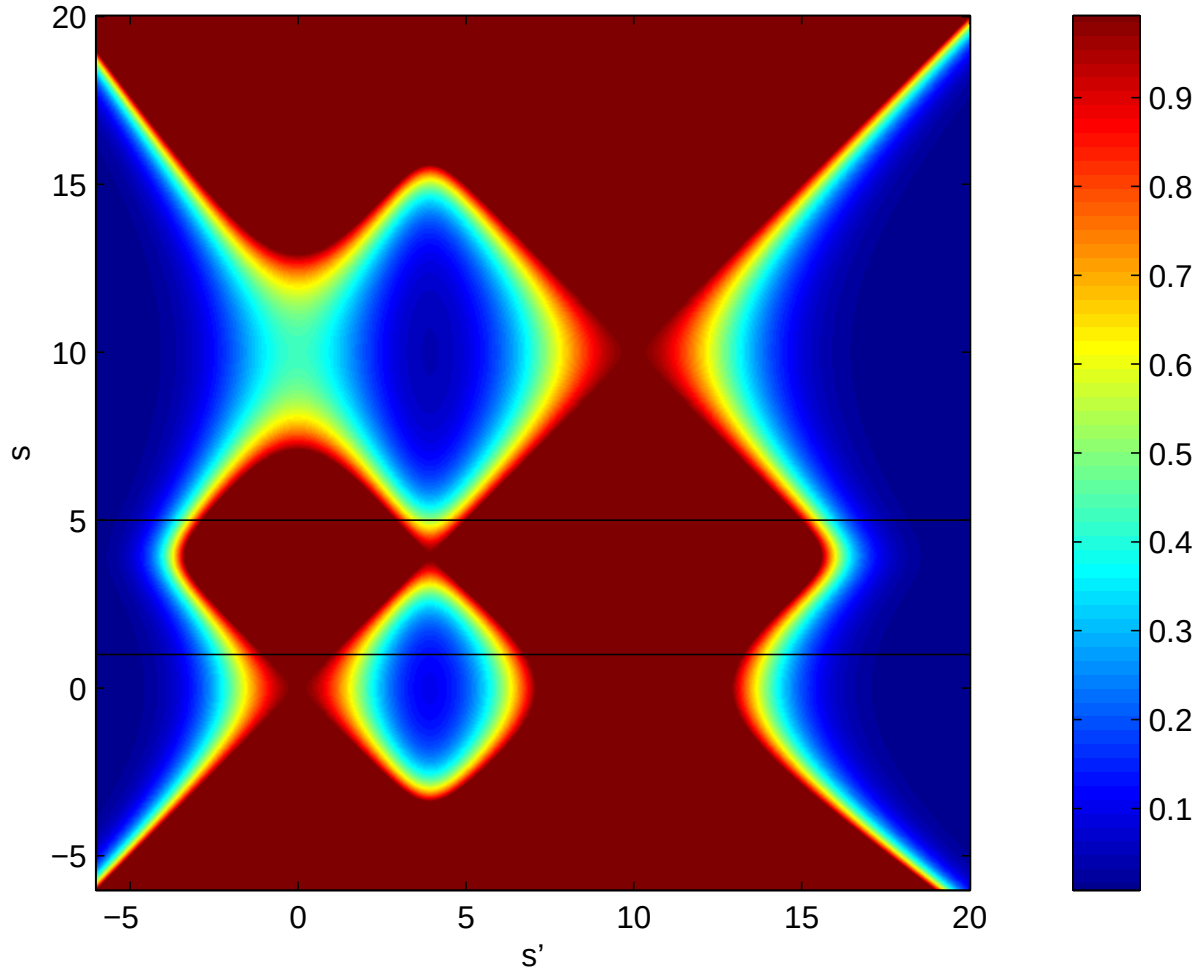
- We choose an arbitrary proposal distribution $q(s'|s)$ (that satisfies mild regularity conditions).
  (When q is symmetric, i.e., $q(s'|s) = q(s|s')$, we have a Metropolis algorithm.)

- We define the *acceptance probability* of a jump from s to s' as

$$a(s \to s') \quad \equiv \quad \min\{1, \frac{q(s|s')\pi(s')}{q(s'|s)\pi(s)}\}$$



Acceptance Probability $a(s \rightarrow s')$



Basic MCMC algorithm: Metropolis-Hastings

1. Initialize: $s^{(0)} \sim r(s)$

2. For $t = 1, 2, \ldots$

   - Propose:

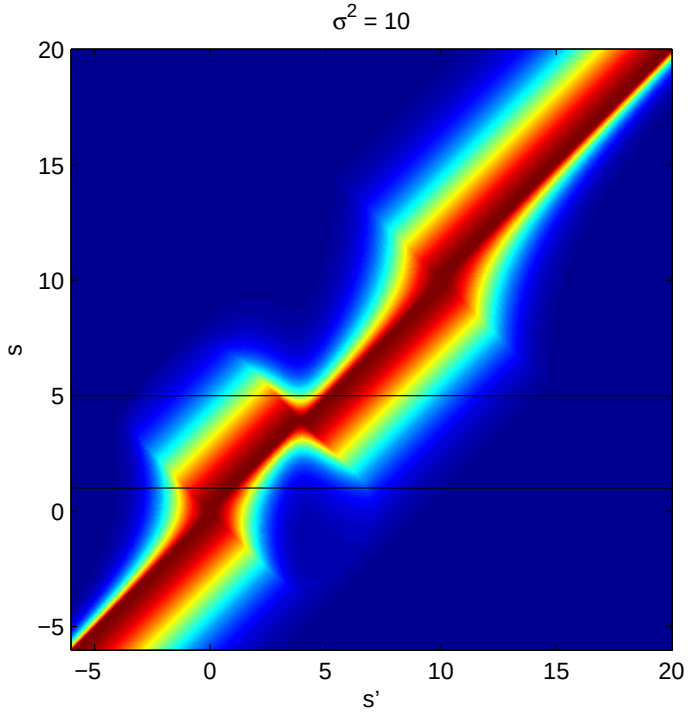
$$s' \sim q(s'|s^{(t-1)})$$

   - Evaluate Proposal: $u \sim \text{Uniform}[0, 1]$

$$s^{(t)} := \begin{cases} s' & u < a(s^{(t-1)} \rightarrow s') \quad \text{Accept} \\ s^{(t-1)} & \text{otherwise} \quad \text{Reject} \end{cases}$$

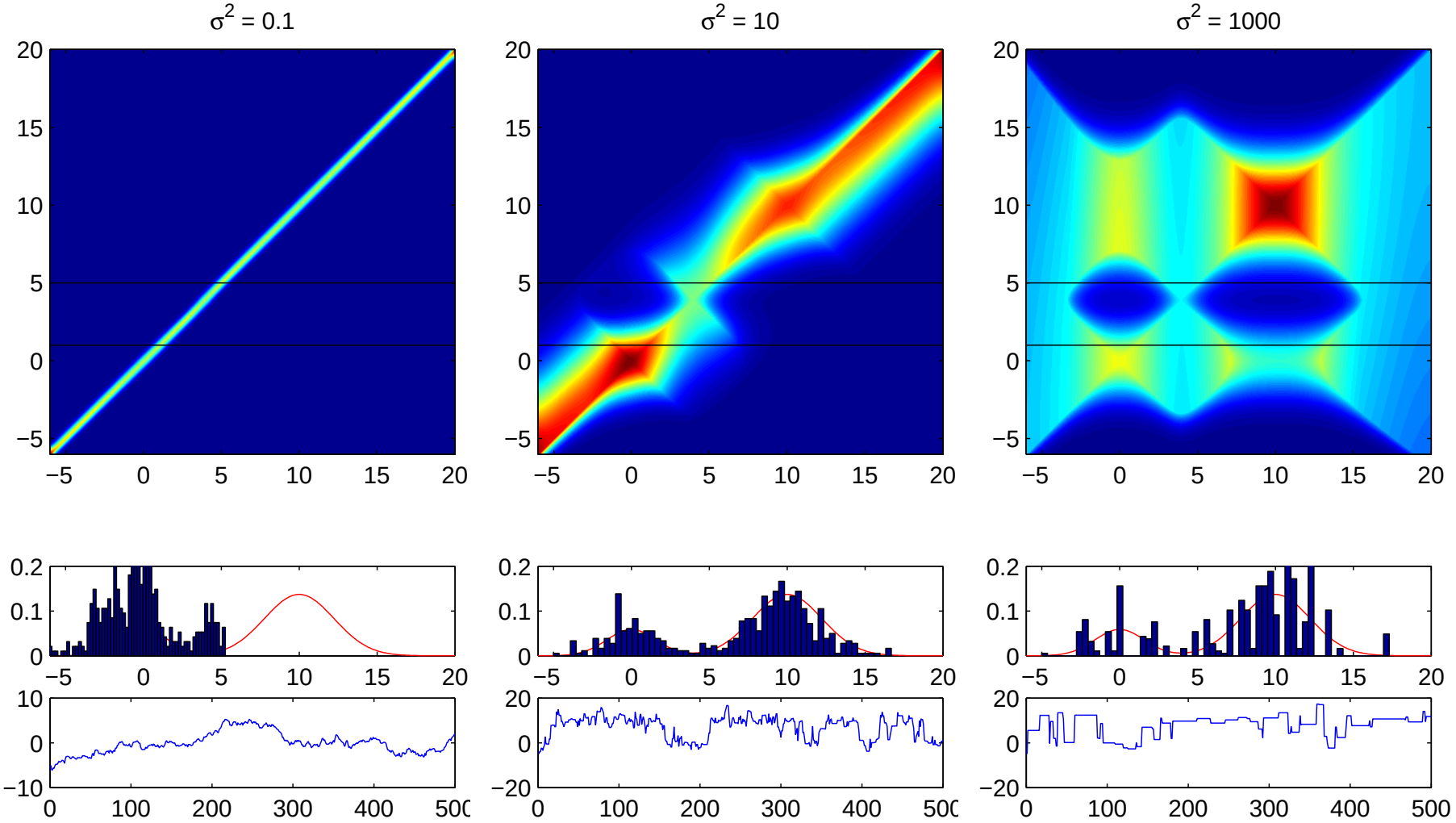
Transition Kernel of the Metropolis Algorithm

$$T(s'|s) = \underbrace{q(s'|s)a(s \to s')}_{\text{Accept}} + \underbrace{\delta(s'-s)\int ds' q(s'|s)(1-a(s \to s'))}_{\text{Reject}}$$



Only Accept part for visual convenience

Various Kernels with the same stationary distribution



$$q(s'|s) = \mathcal{N}(s'; s, \sigma^2)$$

Cascades and Mixtures of Transition Kernels

Let T_1 and T_2 have the same stationary distribution $p(s)$.

Then:

$$\begin{aligned} T_c &= T_1 T_2 \\ T_m &= \nu T_1 + (1-\nu) T_2 \quad 0 \leq \nu \leq 1 \end{aligned}$$

are also transition kernels with stationary distribution $p(s)$.

This opens up many possibilities to “tailor” application specific algorithms.

For example let

T_1 : global proposal (allows large “jumps”)
T_2 : local proposal (investigates locally)

We can use T_m and adjust ν as a function of rejection rate.

Optimization : Simulated Annealing and Iterative Improvement

For optimization, (e.g. to find a MAP solution)

$$s^* = \arg\max_{s \in \mathcal{S}} \pi(s)$$

The MCMC sampler may not visit s^*.

Simulated Annealing: We define the target distribution as

$$\pi(s)^{\tau_i}$$

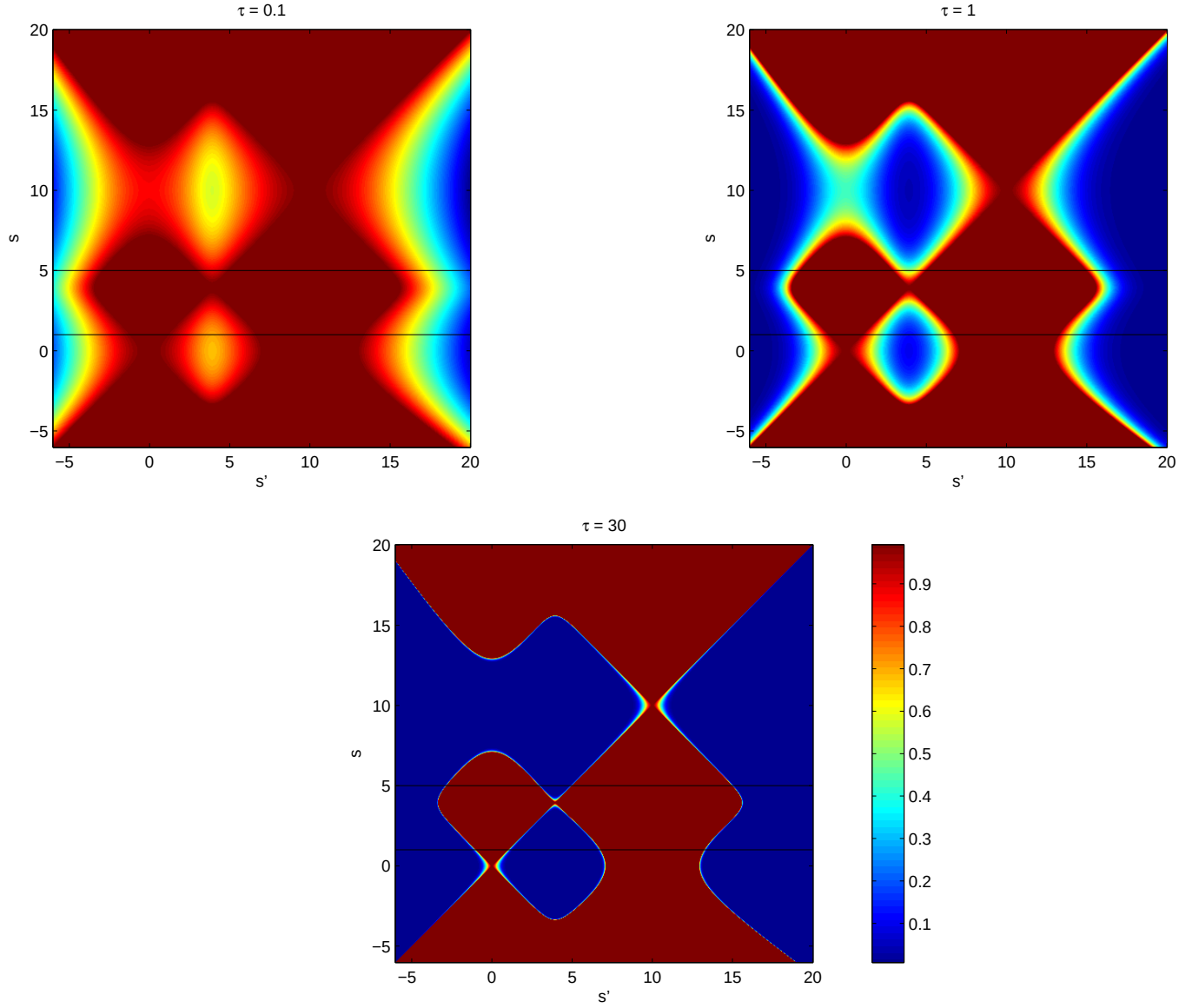
where τ_i is an annealing schedule. For example,

$$\tau_1 = 0.1, \ldots, \tau_N = 10, \tau_{N+1} = \infty \ldots$$

Iterative Improvement (greedy search) is a special case of SA

$$\tau_1 = \tau_2 = \cdots = \tau_N = \infty$$

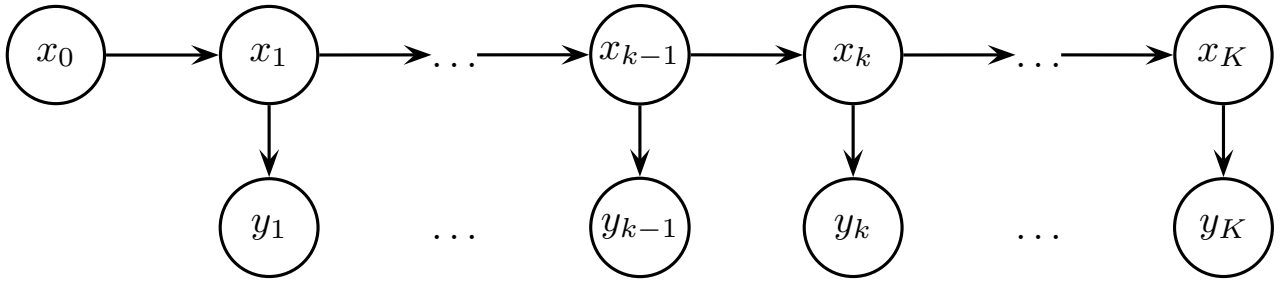
Acceptance probabilities $a(s \rightarrow s')$ at different τ



Time series models with latent variables

Online Inference, Terminology

In signal processing and machine learning many phenomena can be modelled by dynamical state space models (SSM)



$$x_k \sim p(x_k|x_{k-1}) \qquad \text{Transition Model}$$

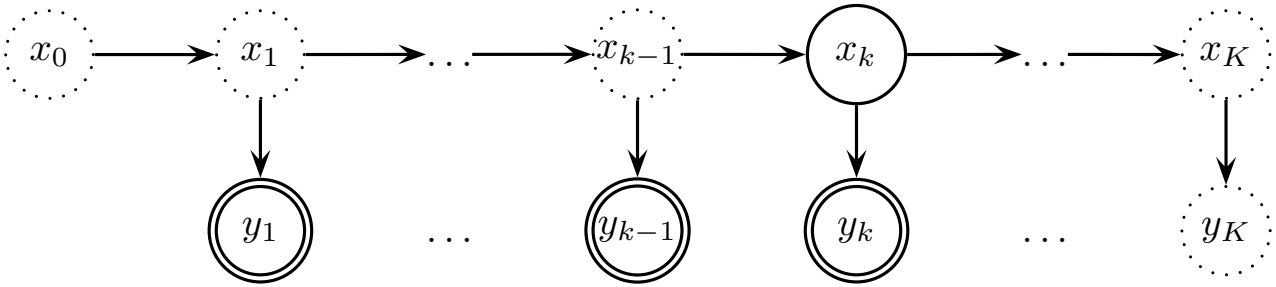
$$y_k \sim p(y_k|x_k) \qquad \text{Observation Model}$$

Here, x is the latent state and y are observations. In a Bayesian setting, x can also include unknown model parameters. This model is very generic and includes as special cases:

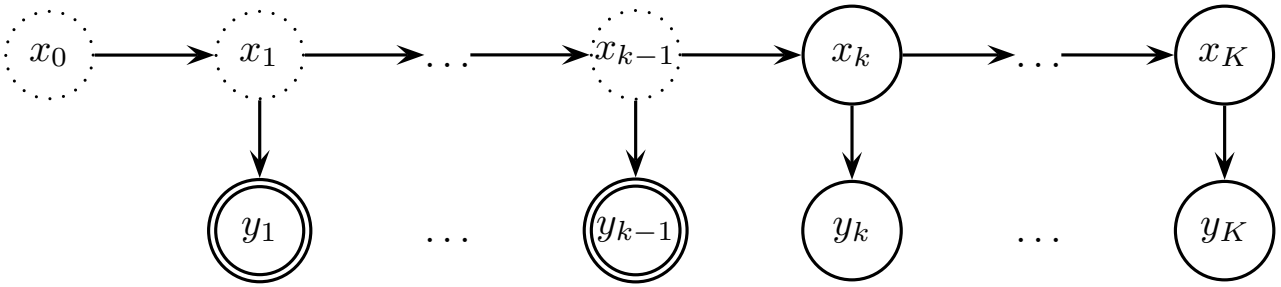
- Linear Dynamical Systems (Kalman Filter models)
- (Time varying) AR, ARMA, MA models
- Hidden Markov Models, Switching state space models
- Dynamic Bayesian networks, Nonlinear Stochastic Dynamical Systems

Online Inference, Terminology

- **Filtering** $p(x_k|y_{1:k})$
  belief state—distribution of current state given all past information

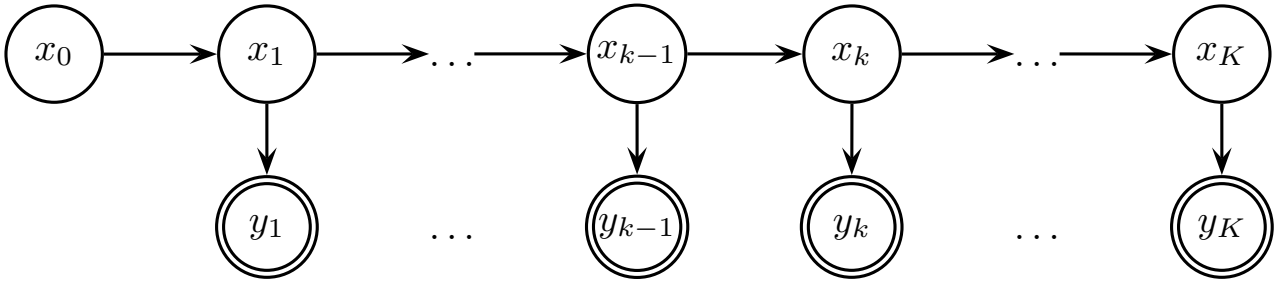


- **Prediction** $p(y_{k:K}, x_{k:K}|y_{1:k-1})$
  evaluation of possible future outcomes; like filtering without observations

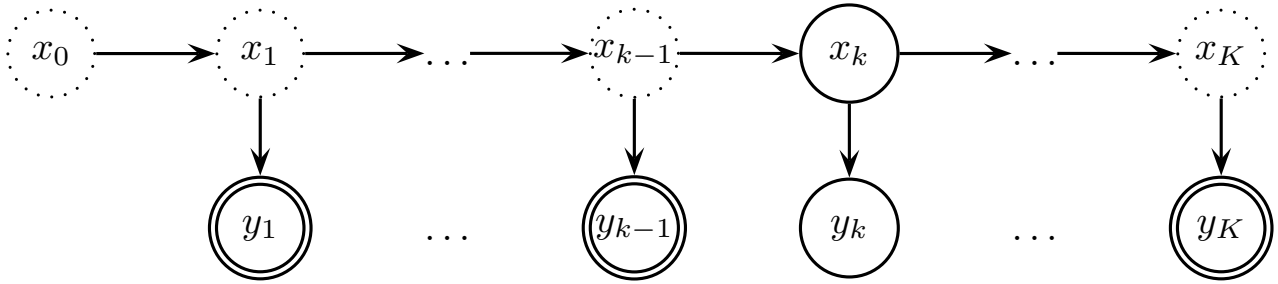


Online Inference, Terminology

- **Smoothing** $p(x_{0:K}|y_{1:K})$,
  Most likely trajectory – Viterbi path $\arg\max_{x_{0:K}} p(x_{0:K}|y_{1:K})$
  better estimate of past states, essential for learning



- **Interpolation** $p(y_k, x_k|y_{1:k-1}, y_{k+1:K})$
  fill in lost observations given past and future

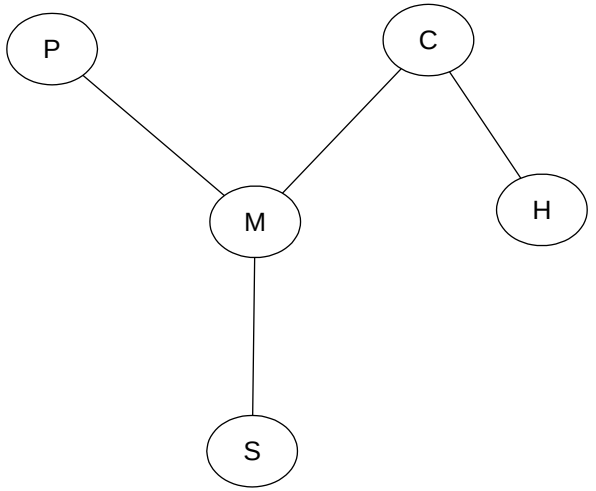


Goals and uses of Probabilistic Models

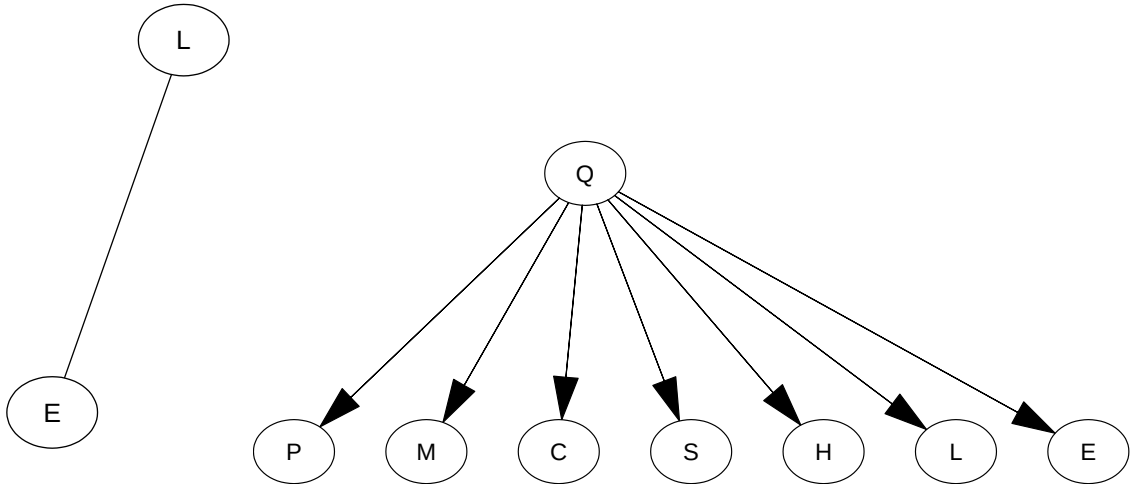
- Finding some interesting (hidden) structure
  Clustering
  Dimensionality Reduction
- Finding a compact representation for data $=$ Data Compression
- Outlier Detection
- Prediction
- Classification
- Optimal Decision (given a loss function)

Why are Hidden Variable models Useful ?

Example: Highschool grades (inspired by J. Whittaker)
Consider the grades that students get from 7 different subjects:
Maths, Physics, Chemistry, History, Sports, Literature, English.
We wish to tell some interesting story about data.

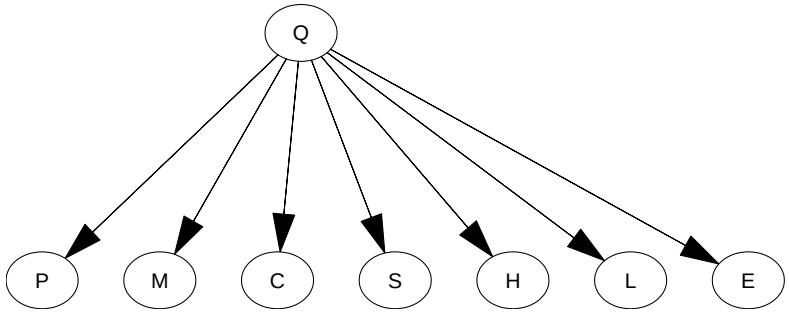


(a) “Visible” Model: Subjects contain related material or require similar abilities.



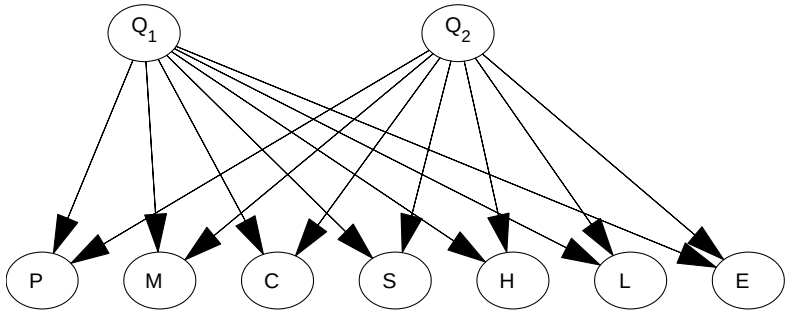
(b) Hidden Variable Model: Students have some hidden interests, e.g. Science, Languages, Art.

Mixture Models



Hidden	Visible	Model
Discrete	Discrete	Discrete Mixture
Discrete	Gaussian	Mixture of Gaussians (MOG)
Gaussian	Gaussian	Factor Analysis (Constrained Gaussian)
Gaussian	Discrete	Clipped Gaussian

Factorized (Distributed) Representations



Discrete Factors

- Possible to code $O(2^q)$ states, however intractable for large q.

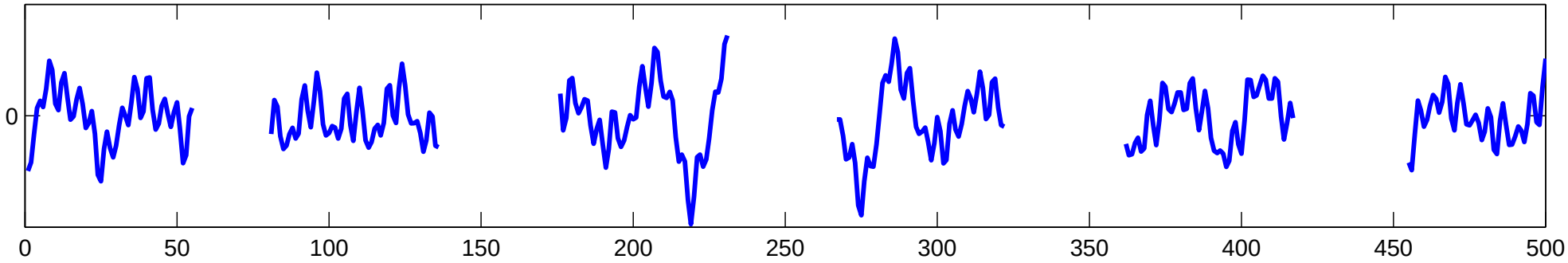
Continuous Factors

- Gaussian Factors $\Rightarrow$ FA, PCA, PPCA ..

- Non Gaussian Factors $\Rightarrow$ ICA, IFA

Some Applications: Audio Restoration

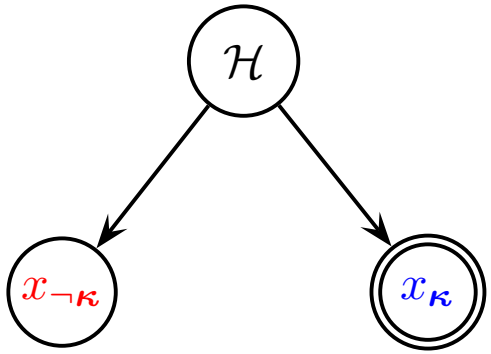
- During download or transmission, some samples of audio are lost
- Estimate missing samples given clean ones



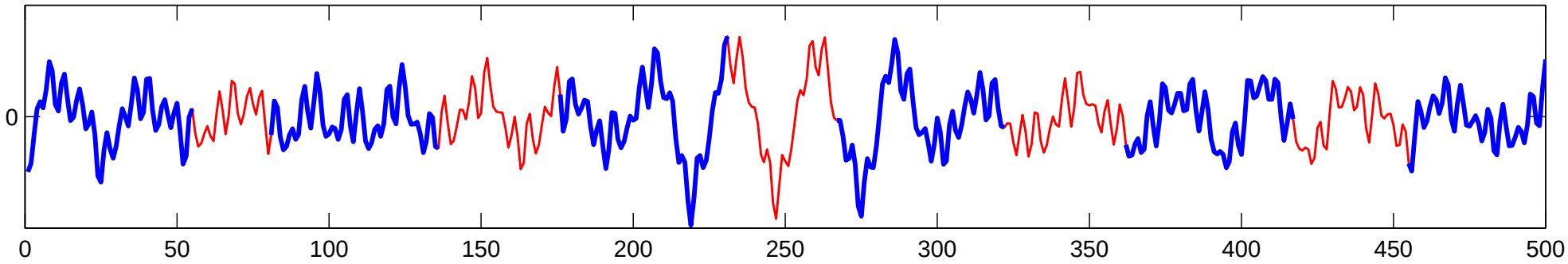
Examples: Audio Restoration

$$p(x_{\neg\boldsymbol{\kappa}}|x_{\boldsymbol{\kappa}}) \propto \int d\mathcal{H} p(x_{\neg\boldsymbol{\kappa}}|\mathcal{H})p(x_{\boldsymbol{\kappa}}|\mathcal{H})p(\mathcal{H})$$

$$\mathcal{H} \equiv (\text{parameters, hidden states})$$

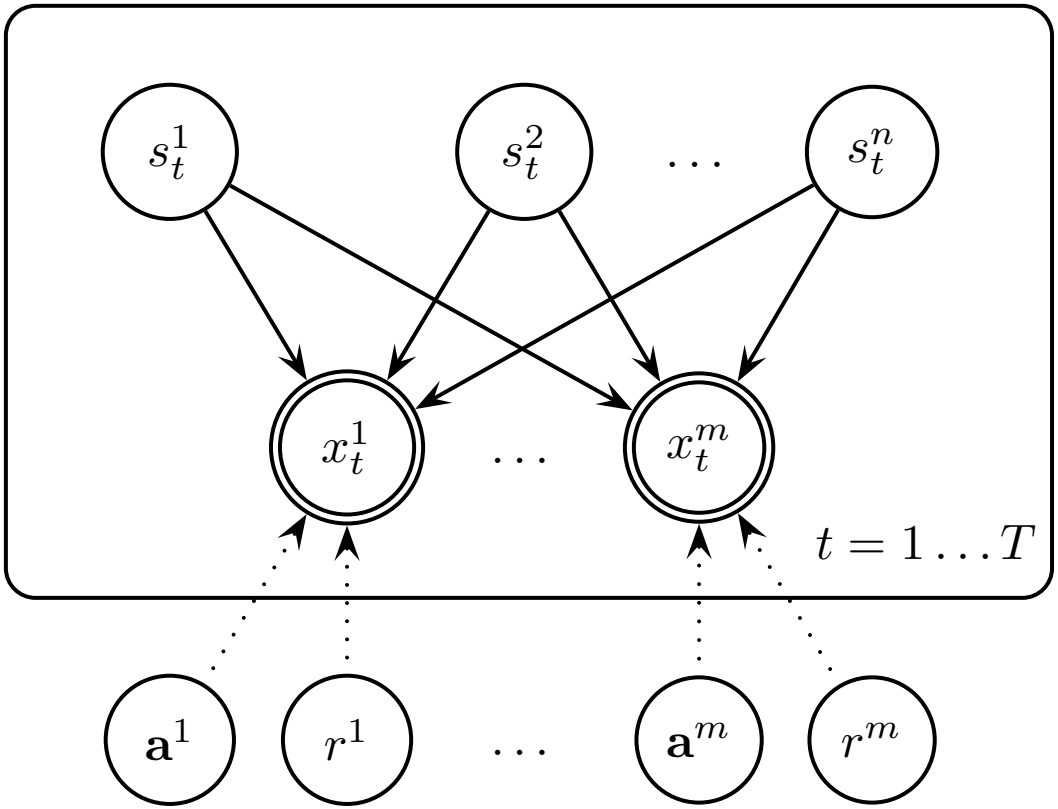


Missing Observed



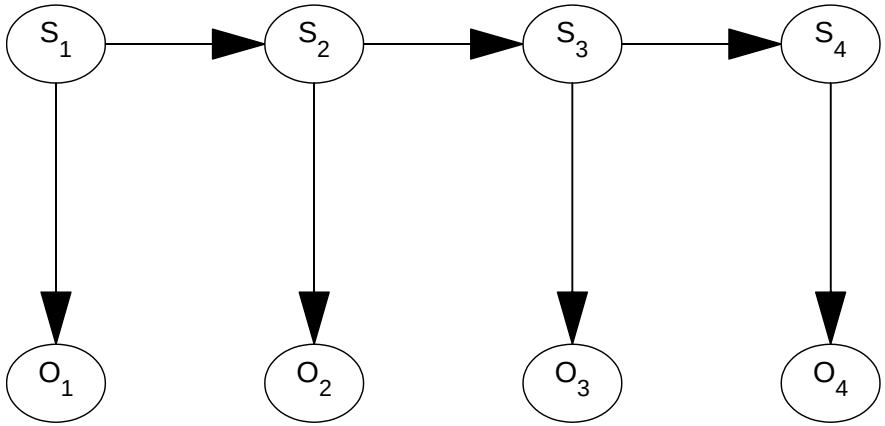
Some Applications: Source Separation

Estimate n hidden signals $\mathbf{s}_t$ from m observed signals $\mathbf{x}_t$.



$$
\begin{aligned}
s_t^i &\sim p(s_t^i) \\
x_t^j &\sim \mathcal{N}(x; \mathbf{a}^j s_t^{1:n}, r^j)
\end{aligned}
$$

Time Series models: Introduce Dynamics



S	O	Static	Dynamic
Discrete	Discrete	DM	Discrete HMM
Discrete	Gaussian	MOG	Continuous HMM
Gaussian	Gaussian	FA	Linear Dynamical System

Inference in HMMs

Compute

$$p(S|O) = \frac{p(O|S)p(S)}{p(O)}$$

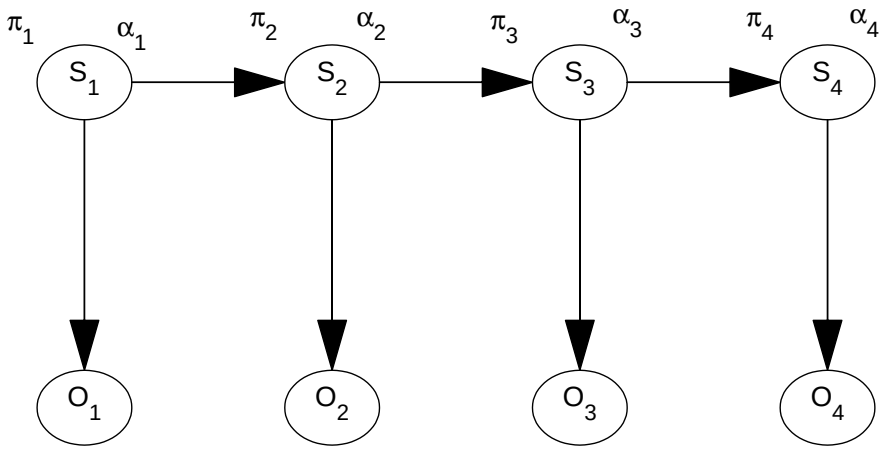
The crux is to compute $p(O)$.

$$\begin{aligned} p(\text{Observations}) &= \sum_{\text{State Seq}} p(\text{Observations}|\text{State Seq})p(\text{State Seq}) \\ p(O) &= \sum_{S} p(O|S)p(S) \\ &= \sum_{S}\prod_{t} p(S_t|S_{t-1})p(O_t|S_t) \\ & \quad p(S_1|S_0) = p(S_1) \end{aligned}$$

Inference in HMMs: Forward

$$p(O) = \underbrace{\sum_{S_T} p(O_T|S_T) \sum_{S_{T-1}} p(S_T|S_{T-1})}_{\alpha_T} p(O_{T-1}|S_{T-1}) \cdots \sum_{S_2} p(S_3|S_2)$$

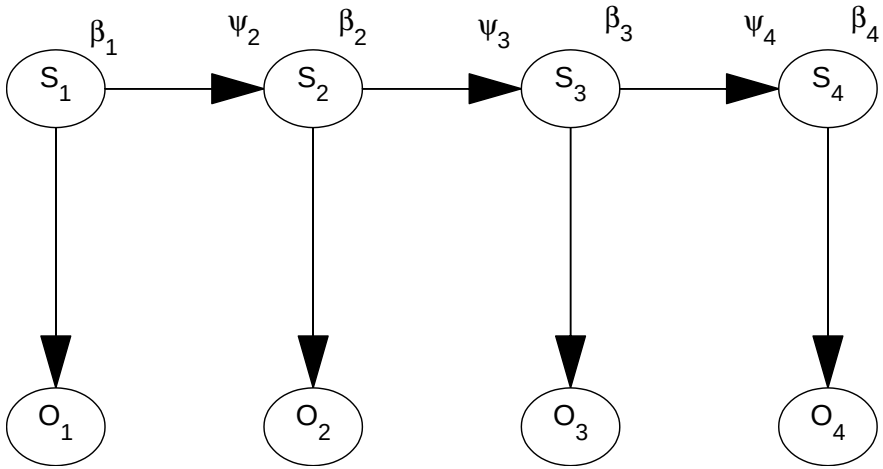
$$\underbrace{p(O_2|S_2) \overbrace{\sum_{S_1} p(S_2|S_1)}^{\pi_2}}_{\alpha_2} \underbrace{p(O_1|S_1) \overbrace{p(S_1)}^{\pi_1}}_{\alpha_1}$$



Inference in HMMs: Backward

$$p(O) = \sum_{S_1} p(S_1)p(O_1|S_1) \underbrace{\sum_{S_2} p(S_2|S_1)p(O_2|S_2) \ldots}_{\beta_1}$$

$$\underbrace{\sum_{S_{T-1}} p(S_{T-1}|S_{T-2})p(O_{T-1}|S_{T-1})}_{\beta_{T-2}} \underbrace{\sum_{S_T} p(S_T|S_{T-1})p(O_T|S_T)}_{\beta_{T-1}} \underbrace{\mathbf{1}}_{\beta_T}$$



Forward-Backward as an instance of Belief Propagation on a factor graph

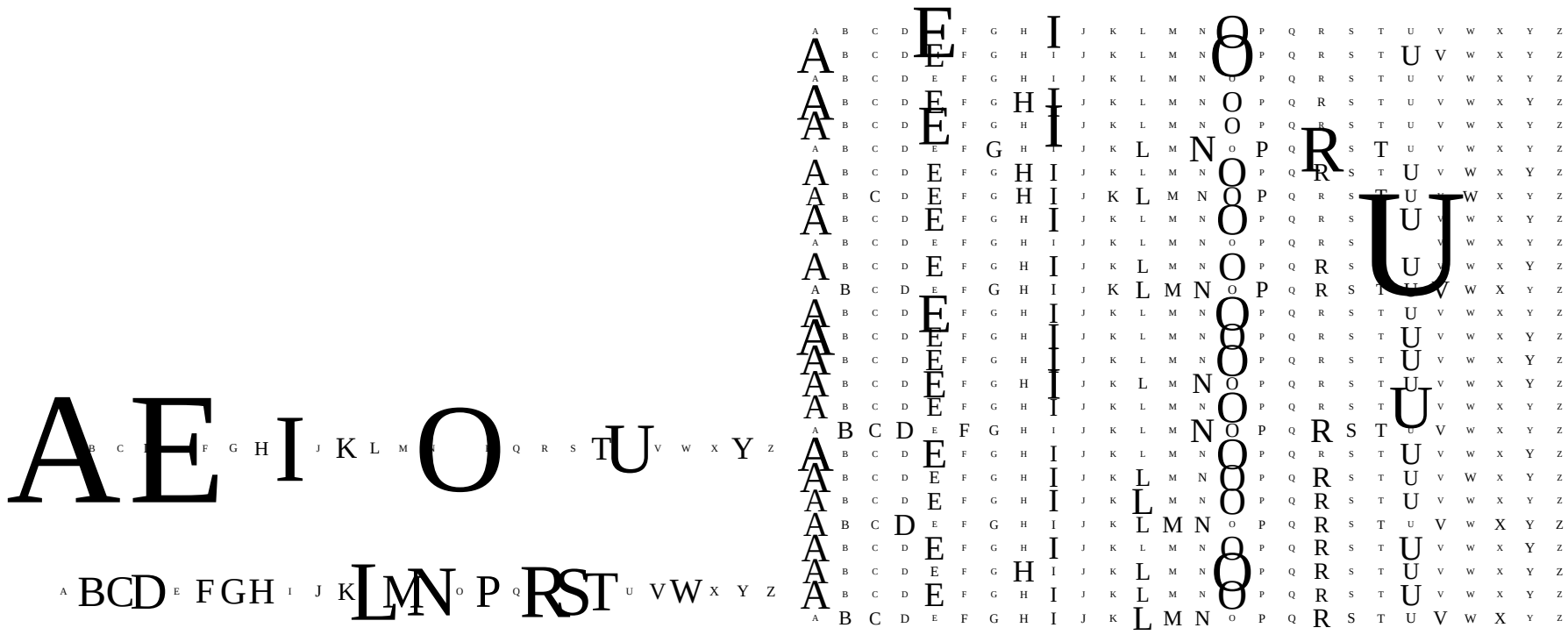
Four Letter Words

Dataset (From Sayood): All four letter English words (2149) of a Sun-Sparc spell checker.

`(abbe, abed, abel, abet, able, ...  zion, zone, zoom, zorn)`

Model	loglik	params	random samples
Full	-16488	456975	**tabu, else, duly, crib, bohr, seal, tome, free, bern**
Markov(2)	-19216	17575	
Markov(1)	-22296	675	**miro**, jaid, saun, trol, bale, liro, pibo, brox, heth
HMM	-24351	53	sehe, reah, vefa, tlil, **hutu**, stec, **make**, otod, **pose**
Factorized	-25909	25	yiij, ekmy, vguo, addn, ecmi, miui, bhin, hnri, roia, azfa

HMM captures an interesting structure



(c) HMM $p(c|s)$

(d) Markov(1) $p(c_i|c_{i-1})$

- For the HMM, the latent states correspond to vowel/non-vowel
- Clustering

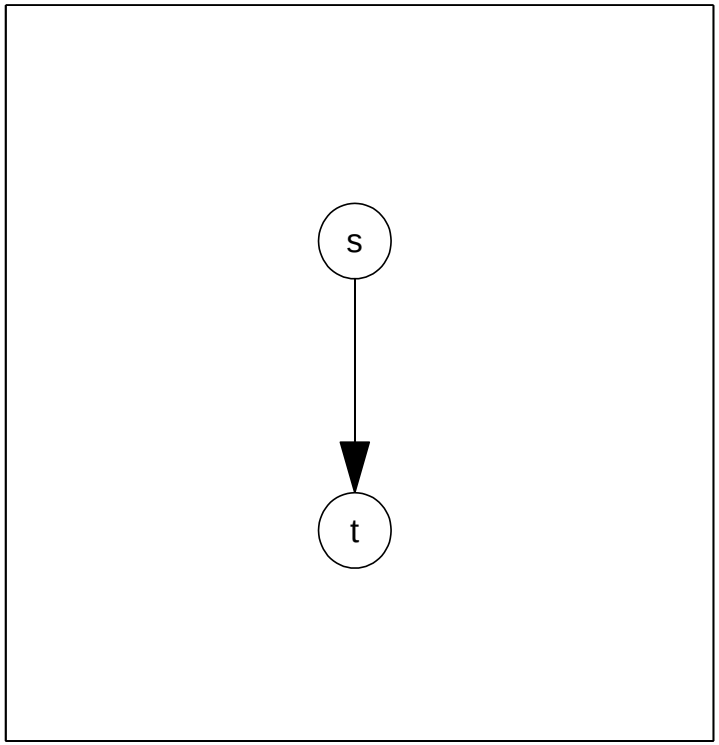
Linear Dynamical Systems and Kalman Filter Models

Partitioned Inverse Equations

$$
\begin{aligned}
K &= \begin{pmatrix} S & D \\ D^T & F \end{pmatrix} \quad K^{-1} = \begin{pmatrix} \Sigma & \Delta \\ \Delta^T & \Phi \end{pmatrix} \\
\Phi &= (F - D^T S^{-1} D)^{-1} \\
\Delta &= -S^{-1} D \Phi \\
\Sigma &= S^{-1}(I - D\Delta^T)
\end{aligned}
$$

$$
\begin{aligned}
p(t,s) &= \mathcal{N}([\mu_t,\ \mu_s]^T, K) \\
&\Downarrow \\
p(s|t) &= \mathcal{N}(\mu_s + D^T S^{-1}(t - \mu_t), \Phi^{-1})
\end{aligned}
$$

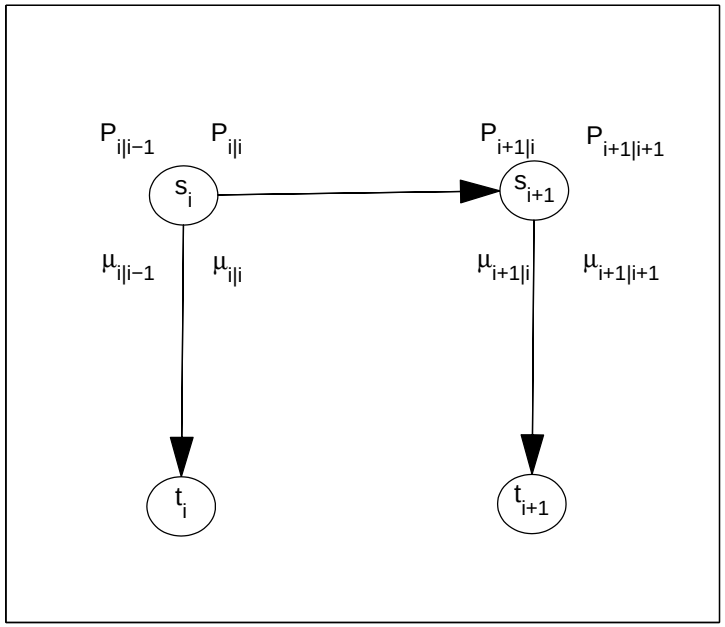
Factor Analysis



$$
\begin{aligned}
t &= Cs + \omega \\
\omega &\sim \mathcal{N}(0, R) \qquad s \sim \mathcal{N}(\mu_s, P) \\
&\Downarrow \\
p(t, s) &= \mathcal{N}(\mu, K) \\
\mu &= \begin{pmatrix} C\mu_s \\ \hline \mu_s \end{pmatrix} \\
K &= \left(\begin{array}{c|c} CPC^T + R & CP \\ \hline PC^T & P \end{array}\right)
\end{aligned}
$$

$$
\begin{aligned}
p(s|t) = \mathcal{N}(\quad & \mu_s + PC^T(CPC^T + R)^{-1}(t - C\mu_s), \\
& P - PC^T(CPC^T + R)^{-1}CP)
\end{aligned}
$$

Kalman Filtering



$$
\begin{aligned}
t_i &= Cs_i + \omega \\
s_{i+1} &= As_i + \nu \\
\omega &\sim \mathcal{N}(0, R) \qquad \nu \sim \mathcal{N}(0, Q) \\
s &\sim \mathcal{N}(\mu_s, P) \\
&\Downarrow \\
p(s_i, s_{i+1} | t_{1:i}) &= \mathcal{N}(\mu, K) \\
\mu &= \begin{pmatrix} \mu_{i|i} \\ \hline A\mu_{i|i} \end{pmatrix} \\
K &= \left(\begin{array}{c|c} P & PA^T \\ \hline AP & APA^T + Q \end{array}\right)
\end{aligned}
$$

Kalman Filtering Equations

$$p(s_i|t_j \ldots, t_1) \quad = \quad \mathcal{N}(\mu_{i|j}, P_{i|j})$$

(by Partitioned Inverse Equations)

$$\begin{aligned}
\mu_{i|i} &= \mu_{i|i-1} + P_{i|i-1}C^T(CP_{i|i-1}C^T + R)^{-1}(\hat{t} - C\mu_{i|i-1}) \\
P_{i|i} &= P_{i|i-1} - P_{i|i-1}C^T(CP_{i|i-1}C^T + R)^{-1}CP_{i|i-1}
\end{aligned}$$

(by the parametric form of $p(s_{i+1}, s_i)$)

$$\begin{aligned}
\mu_{i+1|i} &= A\mu_{i|i} \\
P_{i+1|i} &= AP_{i|i}A^T + Q
\end{aligned}$$

Kalman Smoothing

- Computes $p(s_i|t_1, \ldots t_N)$.
- The state estimates are “more smooth” since all observations are available.
- Analog of forward-backward algorithm in HMM’s.

Example: Point moving on the line

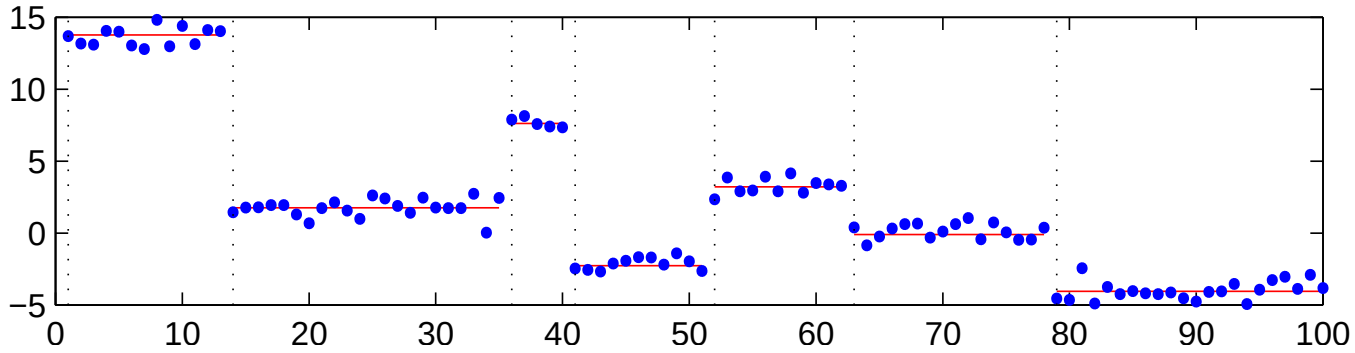
$$
\begin{aligned}
s_i &\sim \mathcal{N}(s_i; \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} s_{i-1}, Q) \\
t_i &\sim \mathcal{N}(t_i; \begin{pmatrix} 1 & 0 \end{pmatrix} s_i, R)
\end{aligned}
$$

run filter_demo.m

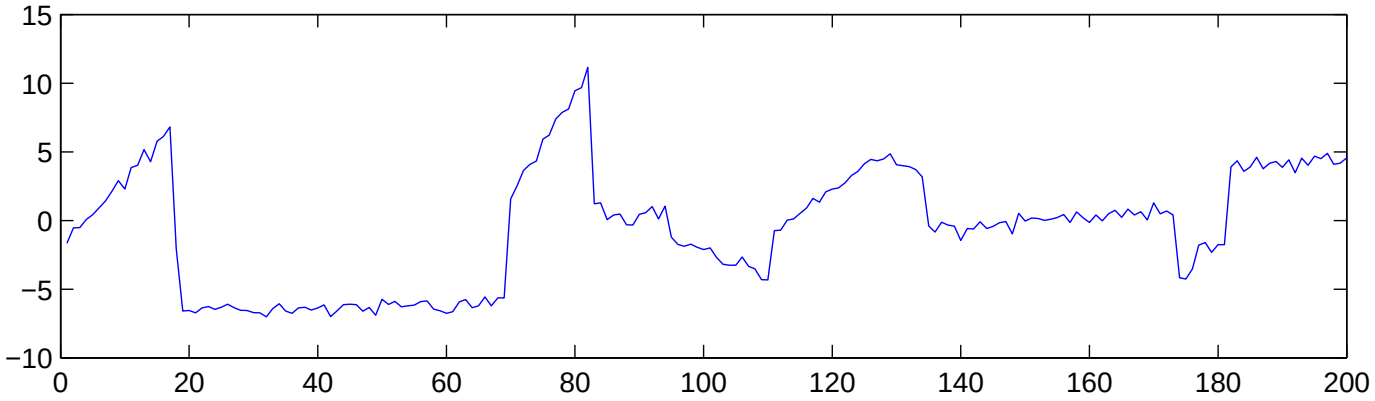
Switching State space models - Segmentation - Changepoint detection

Segmentation and Changepoint detection

- Data are modelled by using simple processes with occasional regime switches
- Piecewise constant



- Piecewise linear



Bayesian Model Selection by Marginal MAP (MMAP)

Integrating out unknown model parameters:

$$M^* = \operatorname*{argmax}_M p(D|M)p(M) = \operatorname*{argmax}_M \int d\theta_M p(D|\theta_M)p(\theta_M|M)p(M)$$

where M: Model, D: Data, θ_M: Model Parameters

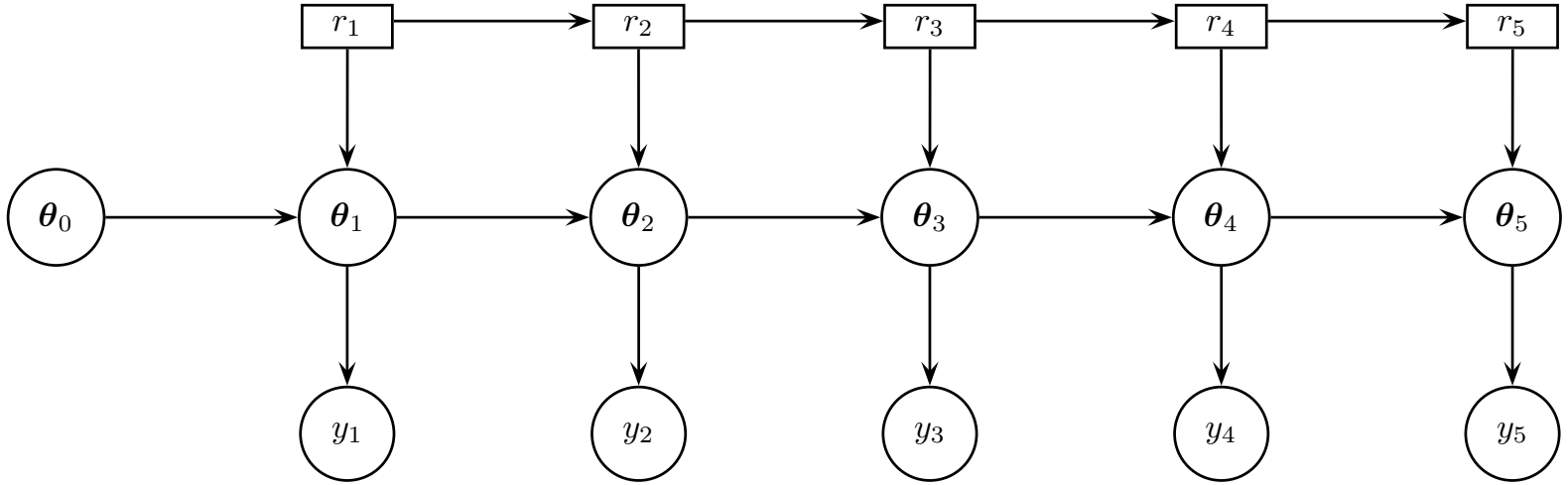
- How do we calculate $\int d\theta_M p(D|\theta_M)p(\theta_M|M)$ efficiently ?
  - When dimensionality of θ_M varies with M the standard choice is reversible Jump Markov Chain Monte Carlo (Green 1995)
  - It is possible to cast the problem to a fixed dimensional problem by introducing indicators that “switch on and off” model parameters (e.g. Godsill 1998)

Conditionally Gaussian Changepoint Model

$r_k \sim p(r_k|r_{k-1})$ Changepoint flags $\in \{\text{new}, \text{reg}\}$

$\theta_k = [r_k = \text{reg}] \underbrace{f(\theta_k|\theta_{k-1})}_{\text{Transition}} + [r_k = \text{new}] \underbrace{\pi(\theta_k)}_{\text{Reinitialization}}$ Latent State

$y_k \sim p(y_k|\theta_k)$ Observations



Sequential Inference Problems

- Filtering $p(\theta_k|y_{1:k}) = \sum_{r_{1:k}} \int d\theta_{0:k-1} p(y_{1:k}|\theta_{0:k}) p(\theta_{0:k}|r_{1:k}) p(r_{1:k})$
- Viterbi path (e.g. Raphael 2001)

$$(r_{1:k}, \theta_{1:k})^* = \underset{r_{1:k}, \theta_{1:k}}{\operatorname{argmax}} p(y_{1:k}|\theta_{0:k}) p(\theta_{0:k}|r_{1:k}) p(r_{1:k})$$

- Best segmentation (MMAP)

$$r^*_{1:k} = \underset{r_{1:k}}{\operatorname{argmax}} \int d\theta_{0:k} p(y_{1:k}|\theta_{0:k}) p(\theta_{0:k}|r_{1:k}) p(r_{1:k})$$

  - Each configuration of $r_{1:K}$ encodes one of the possible 2^K possible models, *i.e.*, segmentation.

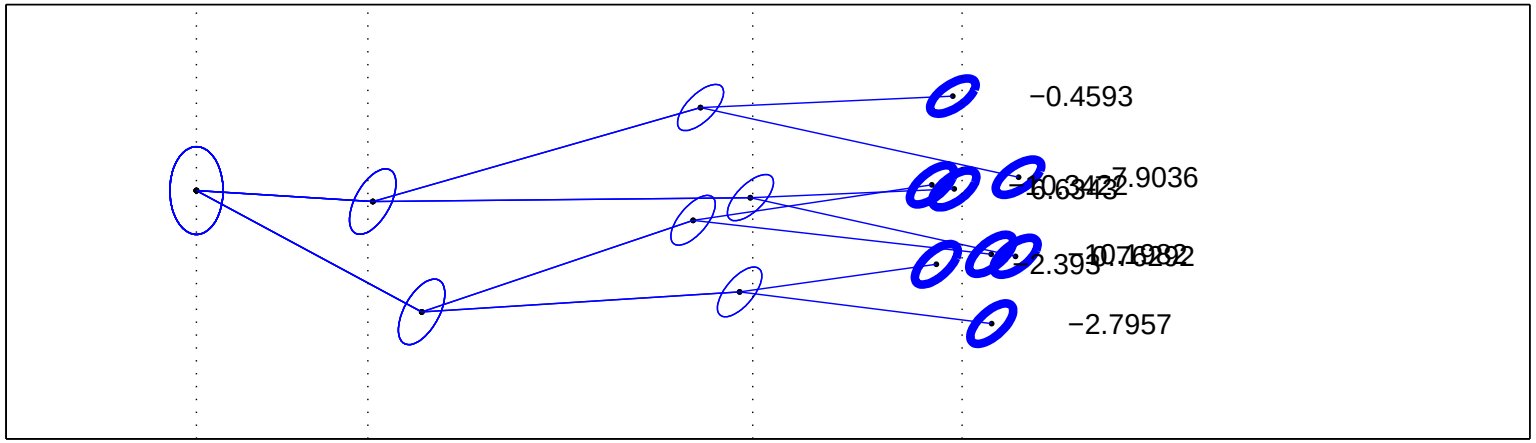
- All problems are similar, but MMAP is usually harder because $\max$ and $\int$ do not commute

Exact Inference in switching state space models

- In general, exact inference is intractable (NP hard)
  - Conditional Gaussians are not closed under marginalization

$\Rightarrow$ Unlike HMM's or KFM's, summing over r_k does not simplify the filtering density

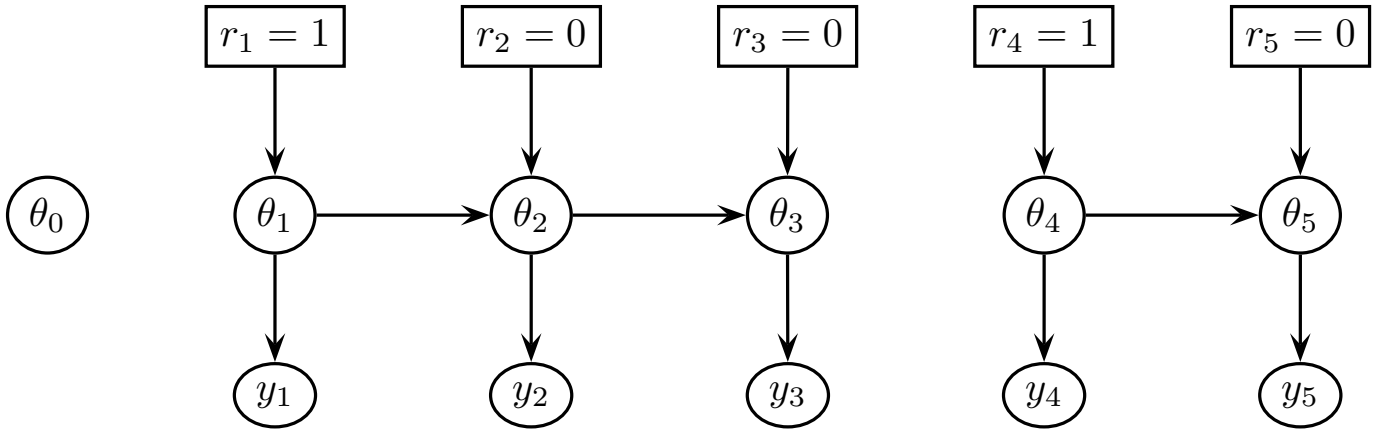
$\Rightarrow$ Number of Gaussian kernels to represent exact filtering density $p(r_k, \theta_k | y_{1:k})$ increases exponentially



Exact Inference for Changepoint detection?

- Exact inference is achievable in polynomial time/space
  - Intuition: When a changepoint occurs, the old state vector is reinitialized

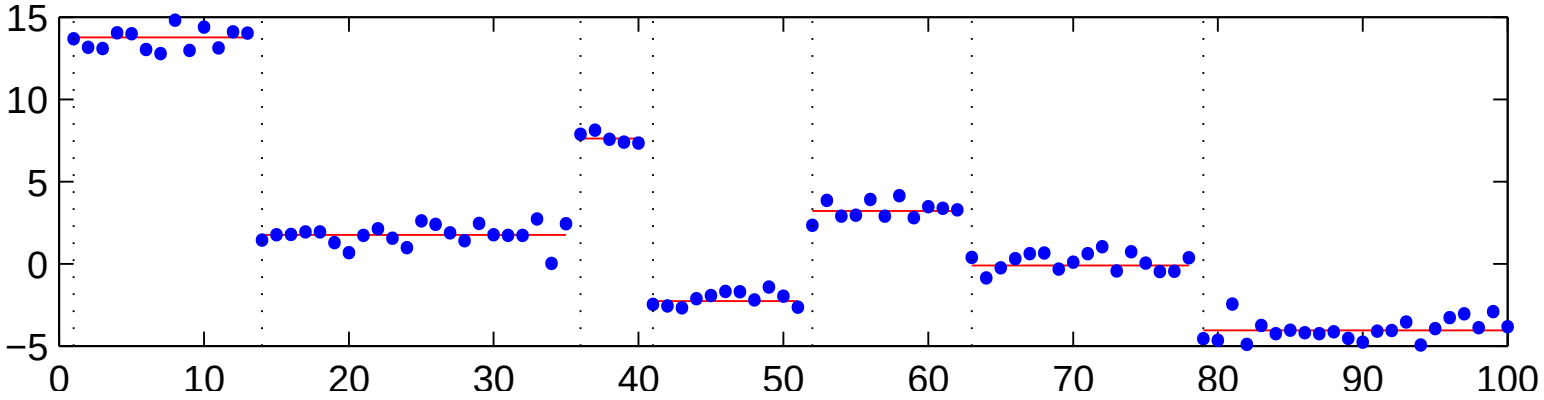
    $\Rightarrow$ Number of Gaussians kernels grows only polynomially (See, e.g., Barry and Hartigan 1992, Digalakis et. al. 1993, Ò Ruanaidh and Fitzgerald 1996, Gustaffson 2000, Fearnhead 2003)



  - The same structure can be exploited for the MMAP problem

    $\Rightarrow$ Trajectories $r_{1:k}^{(i)}$ which are dominated in terms of conditional evidence $p(y_{1:k}, r_{1:k}^{(i)})$ can be discarded without destroying optimality

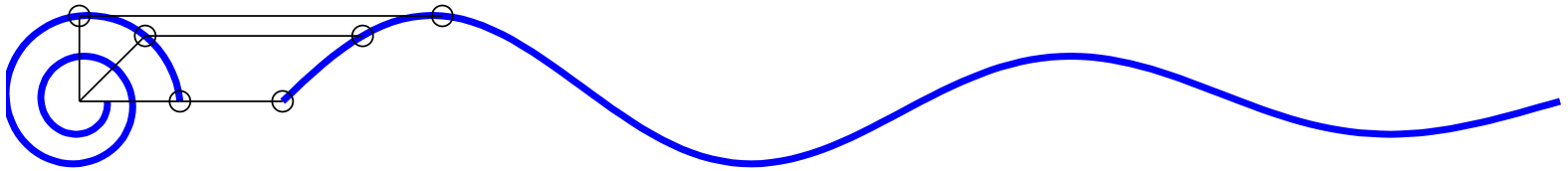
Example 1: Piecewise constant signal



$$
\begin{aligned}
\theta_0 &\sim \mathcal{N}(\mu, P) \\
r_k|r_{k-1} &\sim p(r_k|r_{k-1}) \\
\theta_k|\theta_{k-1}, r_k &\sim \underbrace{[r_k = 0]\delta(\theta_k - \theta_{k-1})}_{\text{reg}} + \underbrace{[r_k = 1]\mathcal{N}(m, V)}_{\text{new}} \\
y_k|\theta_k &\sim \mathcal{N}(\theta_k, R)
\end{aligned}
$$

Example 2: Audio Signal Analysis

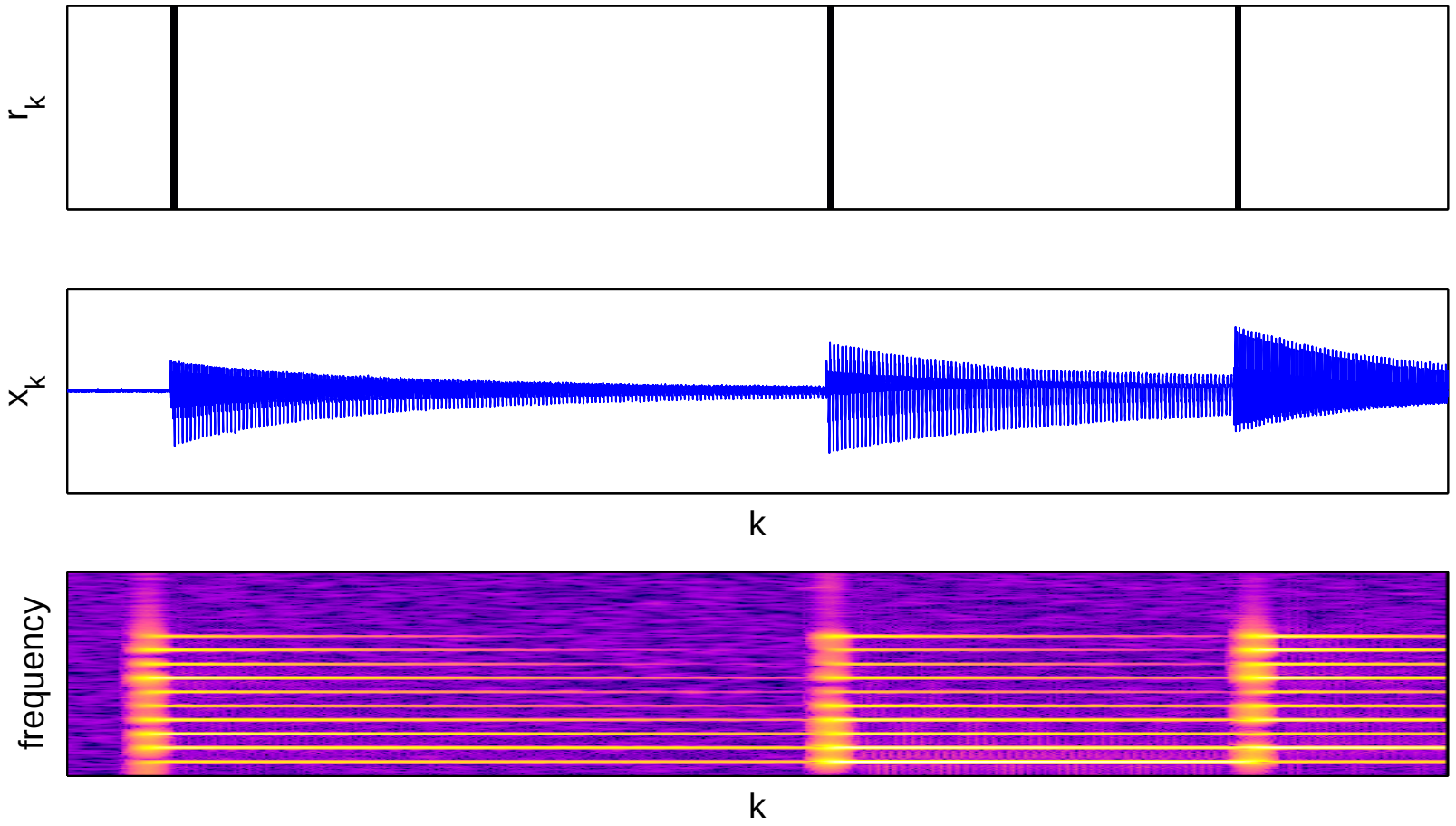
$$
\begin{aligned}
r_k|r_{k-1} &\sim p(r_k|r_{k-1}) \\
\theta_k|\theta_{k-1}, r_k &\sim \underbrace{[r_k = 0]\mathcal{N}(A\theta_{k-1}, Q)}_{\text{reg}} + \underbrace{[r_k = 1]\mathcal{N}(0, S)}_{\text{new}} \\
y_k|\theta_k &\sim \mathcal{N}(C\theta_k, R)
\end{aligned}
$$



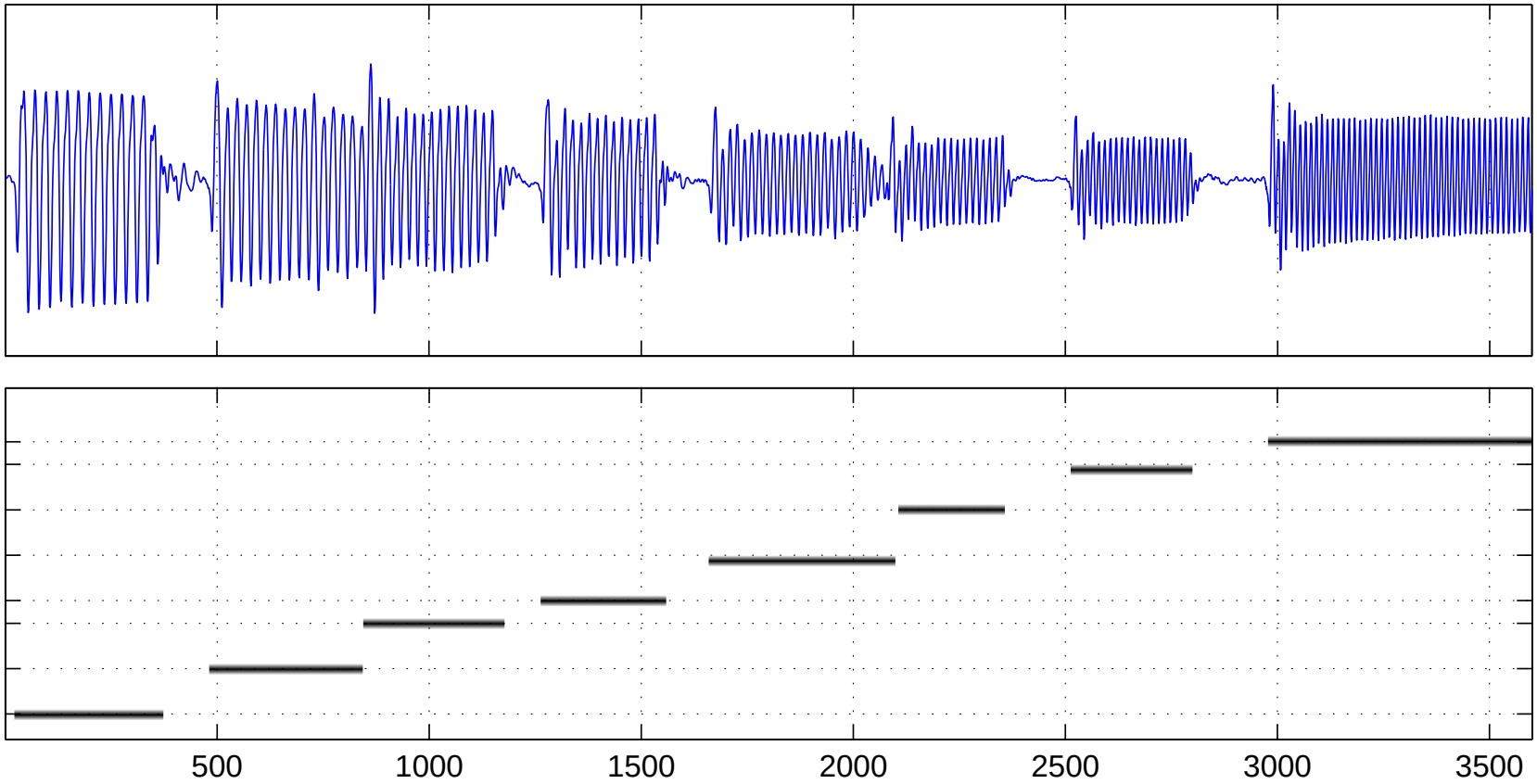
$$
A = \begin{pmatrix} G_\omega & & & \\ & G_\omega^2 & & \\ & & \ddots & \\ & & & G_\omega^H \end{pmatrix}
\qquad
G_\omega = \rho_k \begin{pmatrix} \cos(\omega) & -\sin(\omega) \\ \sin(\omega) & \cos(\omega) \end{pmatrix}
$$

$0 < \rho_k < 1$ is a damping factor and $C = \begin{bmatrix} 1 & 0 & 1 & 0 & \dots & 1 & 0 \end{bmatrix}$ is a projection matrix.

Audio Signal Analysis



Application to music transcription



Cemgil et. al. 2006, IEEE TSALP

Factorial Changepoint model

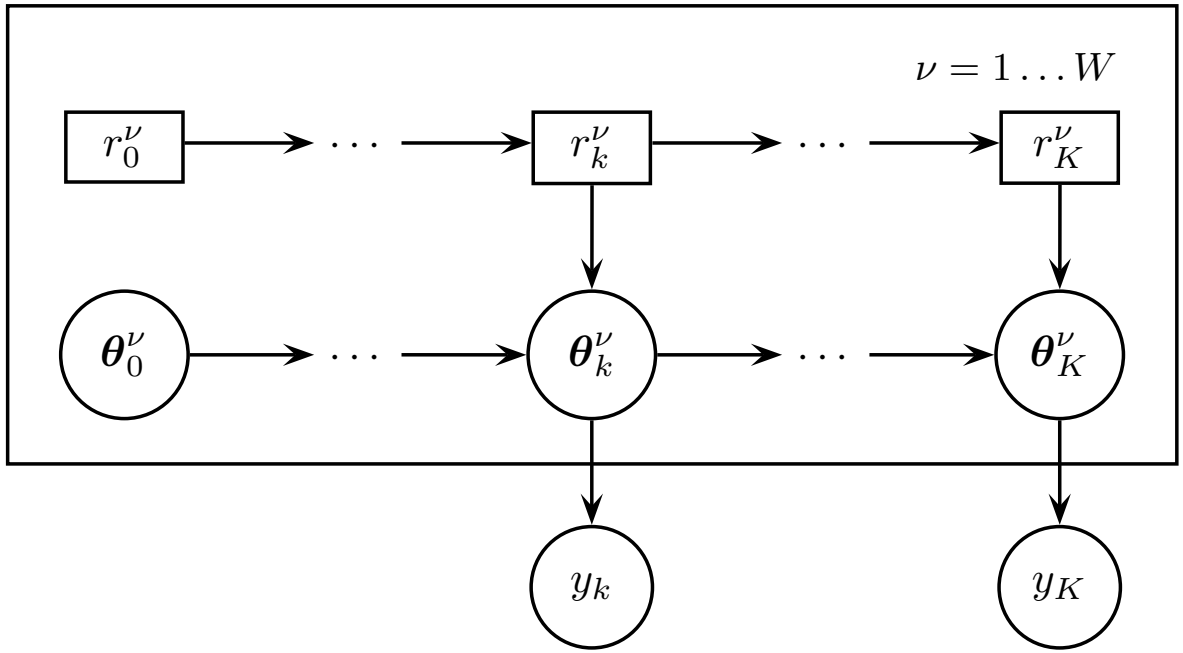
$$r_{0,\nu} \sim \mathcal{C}(r_{0,\nu};\, \pi_{0,\nu})$$

$$\theta_{0,\nu} \sim \mathcal{N}(\theta_{0,\nu};\, \mu_\nu,\, P_\nu)$$

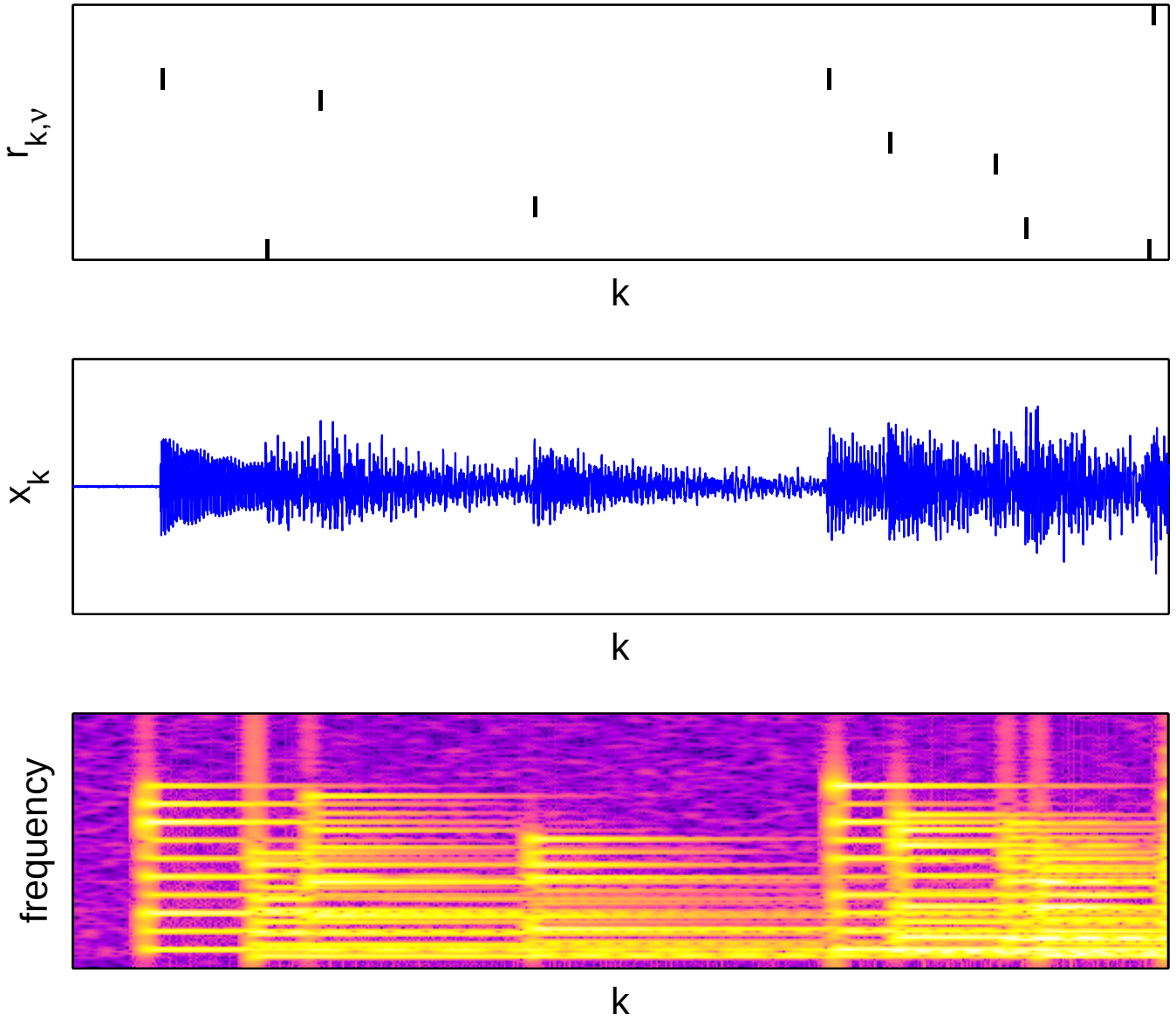
$$r_{k,\nu}|r_{k-1,\nu} \sim \mathcal{C}(r_{k,\nu};\, \pi_\nu(r_{t-1,\nu}))$$ Changepoint indicator

$$\theta_{k,\nu}|\theta_{k-1,\nu} \sim \mathcal{N}(\theta_{k,\nu};\, A_\nu(r_k)\theta_{k-1,\nu},\, Q_\nu(r_k))$$ Latent state

$$y_k|\theta_{k,1:W} \sim \mathcal{N}(y_k;\, C_k\theta_{k,1:W},\, R)$$ Observation



Polyphonic Pitch tracking



Importance Sampling,

Online Inference, Sequential Monte Carlo

Importance Sampling

Consider a probability distribution with $Z = \int d\mathbf{x}\phi(\mathbf{x})$

$$p(\mathbf{x}) = \frac{1}{Z}\phi(\mathbf{x}) \tag{6}$$

Estimate expectations (or features) of $p(\mathbf{x})$ by a weighted sample

$$\langle f(\mathbf{x})\rangle_{p(\mathbf{x})} = \int dx f(\mathbf{x})p(\mathbf{x})$$

$$\langle f(\mathbf{x})\rangle_{p(\mathbf{x})} \approx \sum_{i=1}^{N} \tilde{w}^{(i)} f(\mathbf{x}^{(i)}) \tag{7}$$

Importance Sampling (cont.)

- Change of measure with **weight function** $W(\mathbf{x}) \equiv \phi(x)/q(x)$

$$\langle f(\mathbf{x}) \rangle_{p(\mathbf{x})} = \frac{1}{Z} \int d\mathbf{x} f(\mathbf{x}) \frac{\phi(\mathbf{x})}{q(\mathbf{x})} q(\mathbf{x}) = \frac{1}{Z} \left\langle f(\mathbf{x}) \frac{\phi(\mathbf{x})}{q(\mathbf{x})} \right\rangle_{q(\mathbf{x})} \equiv \frac{1}{Z} \langle f(\mathbf{x}) W(\mathbf{x}) \rangle_{q(\mathbf{x})}$$

- If Z is unknown, as is often the case in Bayesian inference

$$Z = \int d\mathbf{x} \phi(\mathbf{x}) = \int d\mathbf{x} \frac{\phi(\mathbf{x})}{q(\mathbf{x})} q(\mathbf{x}) = \langle W(\mathbf{x}) \rangle_{q(\mathbf{x})}$$

$$\langle f(\mathbf{x}) \rangle_{p(\mathbf{x})} = \frac{\langle f(\mathbf{x}) W(\mathbf{x}) \rangle_{q(\mathbf{x})}}{\langle W(\mathbf{x}) \rangle_{q(\mathbf{x})}}$$

Importance Sampling (cont.)

- Draw $i = 1, \ldots N$ independent samples from q

$$\mathbf{x}^{(i)} \sim q(\mathbf{x})$$

- We calculate the **importance weights**

$$W^{(i)} \quad = \quad W(\mathbf{x}^{(i)}) = \phi(\mathbf{x}^{(i)})/q(\mathbf{x}^{(i)})$$

- Approximate the normalizing constant

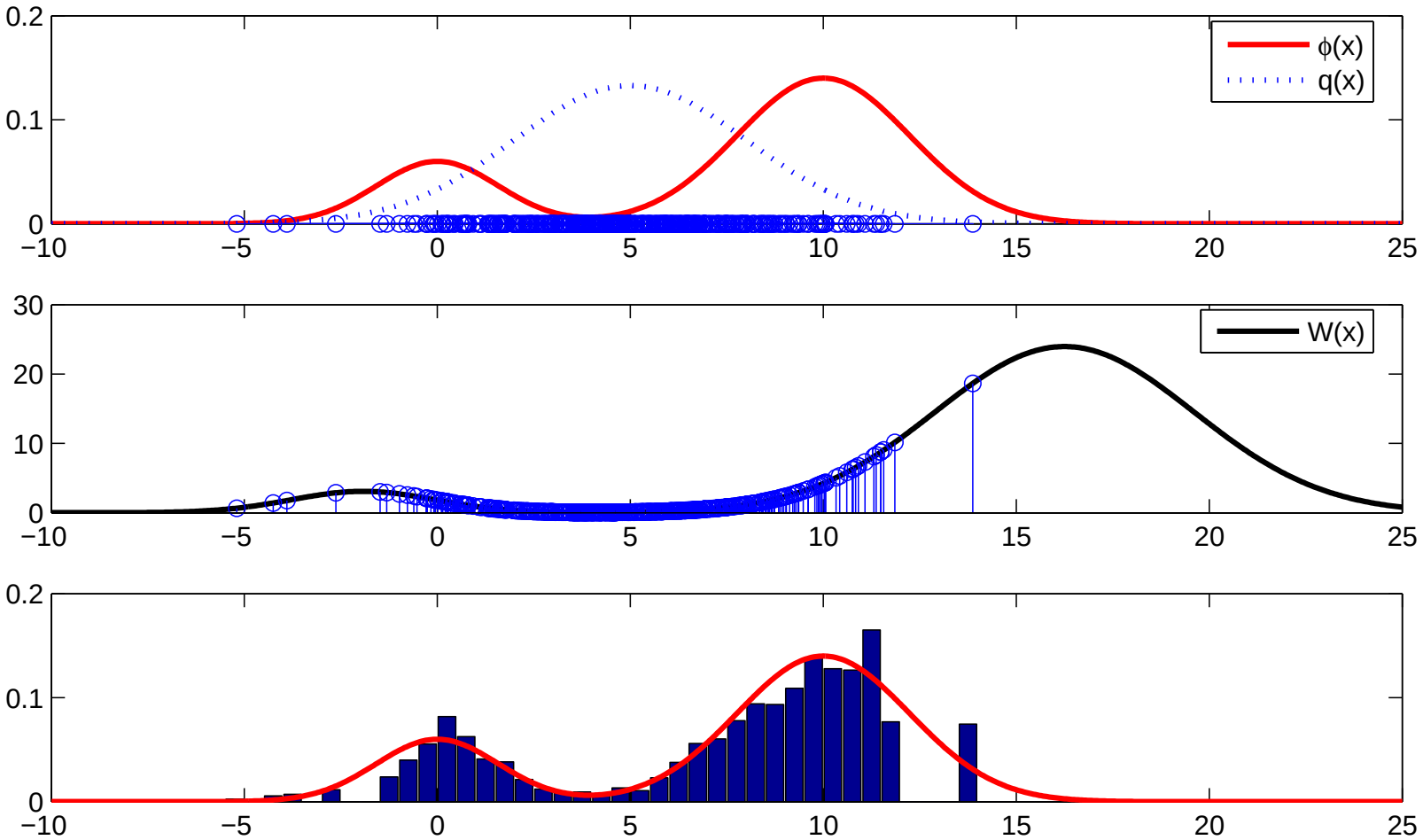
$$Z \quad = \quad \langle W(\mathbf{x}) \rangle_{q(\mathbf{x})} \approx \sum_{i=1}^{N} W^{(i)}$$

- Desired expectation is approximated by

$$\langle f(\mathbf{x}) \rangle_{p(\mathbf{x})} \quad = \quad \frac{\langle f(\mathbf{x}) W(\mathbf{x}) \rangle_{q(\mathbf{x})}}{\langle W(\mathbf{x}) \rangle_{q(\mathbf{x})}} \approx \frac{\sum_{i=1}^{N} W^{(i)} f(\mathbf{x}^{(i)})}{\sum_{i=1}^{N} W^{(i)}} \equiv \sum_{i=1}^{N} \tilde{w}^{(i)} f(\mathbf{x}^{(i)})$$

Here $\tilde{w}^{(i)} = W^{(i)} / \sum_{j=1}^{N} W^{(j)}$ are *normalized importance weights*.

Importance Sampling (cont.)



Resampling

- Importance sampling computes an approximation with weighted delta functions

$$p(x) \approx \sum_i \tilde{W}^{(i)} \delta(x - x^{(i)})$$

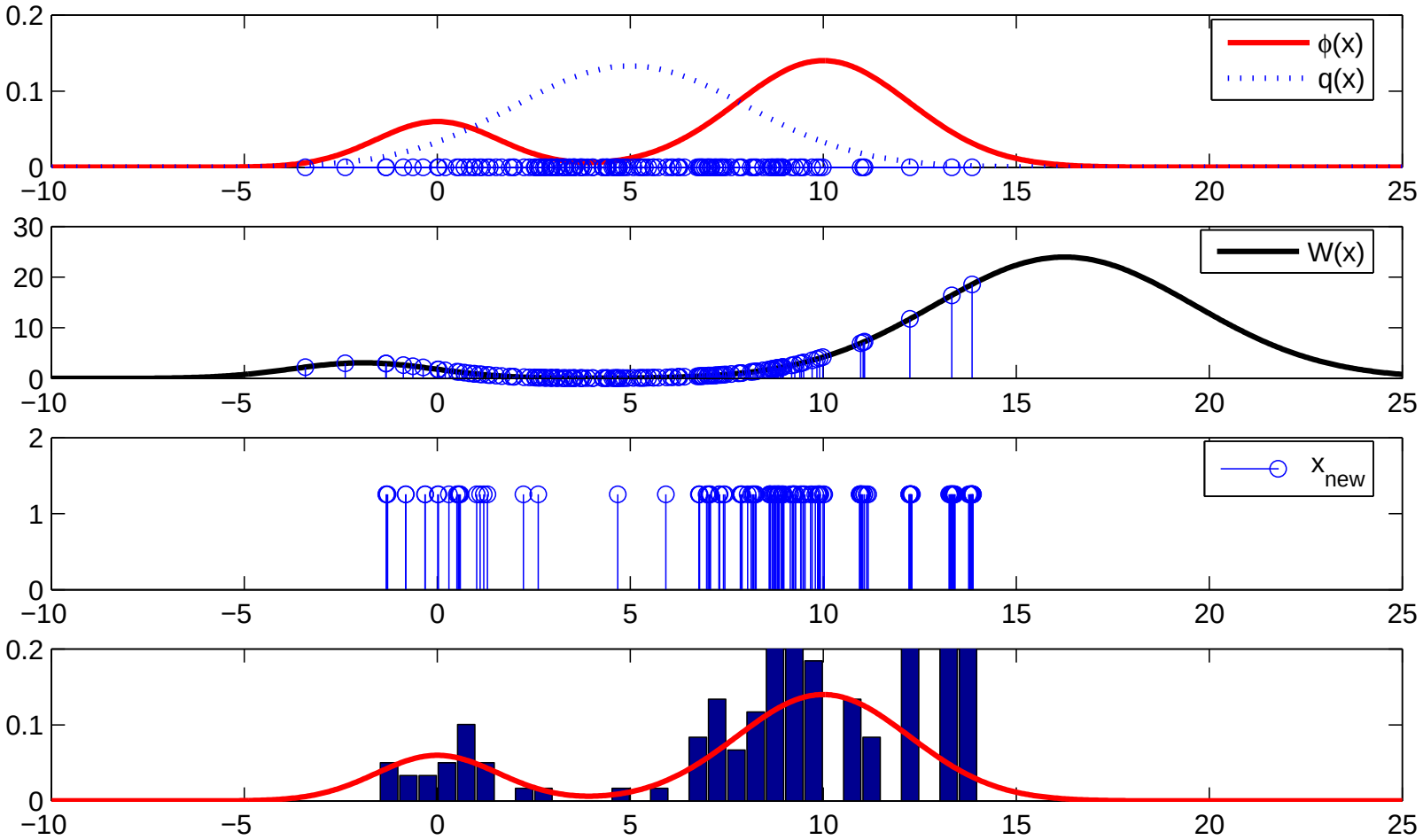
- In this representation, most of $\tilde{W}^{(i)}$ will be very close to zero and the representation may be dominated by few large weights.
- Resampling samples a set of new “particles”

$$x_{\text{new}}^{(j)} \sim \sum_i \tilde{W}^{(i)} \delta(x - x^{(i)})$$

$$p(x) \approx \frac{1}{N} \sum_j \delta(x - x_{\text{new}}^{(j)})$$

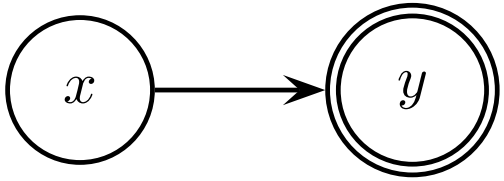
- Since we sample from a degenerate distribution, particle locations stay unchanged. We merely dublicate (, triplicate, ...) or discard particles according to their weight.
- This process is also named “selection”, “survival of the fittest”, e.t.c., in various fields (Genetic algorithms, AI..).

Resampling



$$x_{\text{new}}^{(j)} \sim \sum_i \tilde{W}^{(i)} \delta\left(x - x^{(i)}\right)$$

Examples of Proposal Distributions



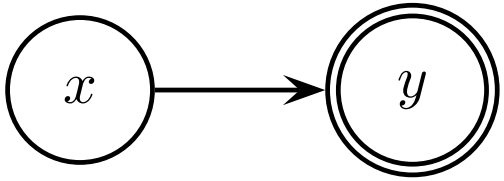
$$p(x|y) \propto p(y|x)p(x)$$

Task: Obtain samples from the posterior $p(x|y)$

- Prior as the proposal. $q(x) = p(x)$

$$W(x) = \frac{p(y|x)p(x)}{p(x)} = p(y|x)$$

Examples of Proposal Distributions



$$p(x|y) \propto p(y|x)p(x)$$

Task: Obtain samples from the posterior $p(x|y)$

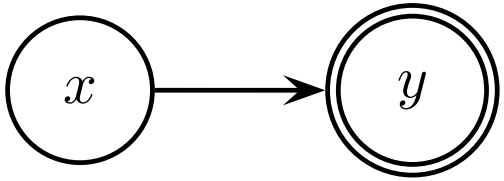
- Likelihood as the proposal. $q(x) = p(y|x)/\int dx p(y|x) = p(y|x)/c(y)$

$$W(x) = \frac{p(y|x)p(x)}{p(y|x)/c(y)} = p(x)c(y) \propto p(x)$$

- Interesting when sensors are very accurate and $\dim(y) \gg \dim(x)$. Idea behind "Dual-PF" (Thrun et.al.. 2000)

Since there are many proposals, is there a "best" proposal distribution?

Optimal Proposal Distribution



$$p(x|y) \propto p(y|x)p(x)$$

Task: Estimate $\langle f(x) \rangle_{p(x|y)}$

- IS constructs the estimator $I(f) = \langle f(x)W(x) \rangle_{q(x)}$ (where $W(x) = p(x|y)/q(x)$)
- Minimize the variance of the estimator

$$\begin{aligned} \left\langle \left(f(x)W(x) - \langle f(x)W(x) \rangle\right)^2 \right\rangle_{q(x)} &= \left\langle f^2(x)W^2(x) \right\rangle_{q(x)} - \langle f(x)W(x) \rangle^2_{q(x)} && (8) \\ &= \left\langle f^2(x)W^2(x) \right\rangle_{q(x)} - \langle f(x) \rangle^2_{p(x)} && (9) \\ &= \left\langle f^2(x)W^2(x) \right\rangle_{q(x)} - I^2(f) && (10) \end{aligned}$$

- Minimize the first term since only it depends upon q

Optimal Proposal Distribution

- (By Jensen's inequality) The first term is lower bounded:

$$\left\langle f^2(x)W^2(x)\right\rangle_{q(x)} \quad \geq \quad \langle |f(x)|W(x)\rangle^2_{q(x)} = \left(\int |f(x)|\, p(x|y)dx\right)^2$$

- We well look for a distribution q^* that attains this lower bound. Take

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

Optimal Proposal Distribution (cont.)

- The weight function for this particular proposal q^* is

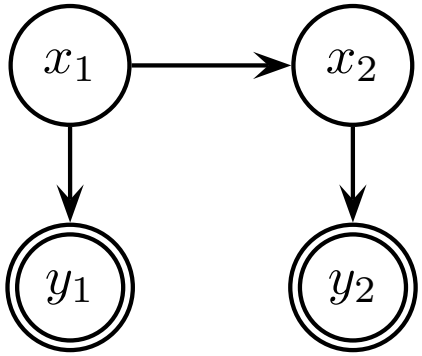
$$W_*(x) = p(x|y)/q^*(x) = \frac{\int |f(x')|p(x'|y)dx'}{|f(x)|}$$

- We show that q^* attains its lower bound

$$\begin{aligned}
\left\langle f^2(x)W_*^2(x)\right\rangle_{q^*(x)} &= \left\langle f^2(x)\frac{\left(\int |f(x')|p(x'|y)dx'\right)^2}{|f(x)|^2}\right\rangle_{q^*(x)} \\
&= \left(\int |f(x')|p(x'|y)dx'\right)^2 = \langle|f(x)|\rangle^2_{p(x|y)} \\
&= \langle|f(x)|W_*(x)\rangle^2_{q^*(x)}
\end{aligned}$$

- $\Rightarrow$ There are distributions q^* that are even "better" than the exact posterior!

Examples of Proposal Distributions



$$p(x|y) \propto p(y_1|x_1)p(x_1)p(y_2|x_2)p(x_2|x_1)$$

Task: Obtain samples from the posterior $p(x_{1:2}|y_{1:2})$

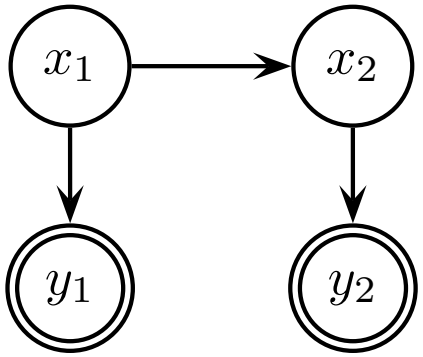
- Prior as the proposal. $q(x_{1:2}) = p(x_1)p(x_2|x_1)$

$$W(x_1, x_2) = p(y_1|x_1)p(y_2|x_2)$$

- We sample from the prior as follows:

$$x_1^{(i)} \sim p(x_1) \qquad x_2^{(i)} \sim p(x_2|x_1 = x_1^{(i)}) \qquad W(\mathbf{x}^{(i)}) = p(y_1|x_1^{(i)})p(y_2|x_2^{(i)})$$

Examples of Proposal Distributions



$$p(x|y) \propto p(y_1|x_1)p(x_1)p(y_2|x_2)p(x_2|x_1)$$

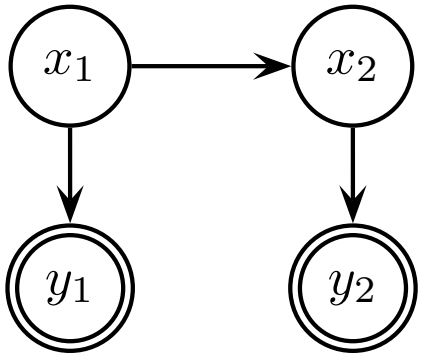
- State prediction as the proposal. $q(x_{1:2}) = p(x_1|y_1)p(x_2|x_1)$

$$W(x_1, x_2) = \frac{p(y_1|x_1)p(x_1)p(y_2|x_2)p(x_2|x_1)}{p(x_1|y_1)p(x_2|x_1)} = p(y_1)p(y_2|x_2)$$

- Note that this proposal does not depend on x_1
- We sample from the proposal and compute the weight

$$x_1^{(i)} \sim p(x_1|y_1) \qquad x_2^{(i)} \sim p(x_2|x_1 = x_1^{(i)}) \qquad W(\mathbf{x}^{(i)}) = p(y_1)p(y_2|x_2^{(i)})$$

Examples of Proposal Distributions



$$p(x|y) \propto p(y_1|x_1)p(x_1)p(y_2|x_2)p(x_2|x_1)$$

- Filtering distribution as the proposal. $q(x_{1:2}) = p(x_1|y_1)p(x_2|x_1, y_2)$

$$W(x_1, x_2) = \frac{p(y_1|x_1)p(x_1)p(y_2|x_2)p(x_2|x_1)}{p(x_1|y_1)p(x_2|x_1, y_2)} = p(y_1)p(y_2|x_1)$$

- Note that this proposal does not depend on x_2
- We sample from the proposal and compute the weight

$$x_1^{(i)} \sim p(x_1|y_1) \qquad x_2^{(i)} \sim p(x_2|x_1 = x_1^{(i)}, y_2) \qquad W(\mathbf{x}^{(i)}) = p(y_1)p(y_2|x_1^{(i)})$$

Sequential Importance Sampling, Particle Filtering

Apply importance sampling to the SSM to obtain some samples from the posterior $p(x_{0:K}|y_{1:K})$.

$$p(x_{0:K}|y_{1:K}) \quad = \quad \frac{1}{p(y_{1:K})} p(y_{1:K}|x_{0:K}) p(x_{0:K}) \equiv \frac{1}{Z_y} \phi(x_{0:K}) \tag{11}$$

Key idea: sequential construction of the proposal distribution q, possibly using the available observations $y_{1:k}$, i.e.

$$q(x_{1:K}|y_{1:K}) = q(x_0) \prod_{k=1}^{K} q(x_k|x_{1:k-1} y_{1:k})$$

Sequential Importance Sampling

Due to the sequential nature of the model and the proposal, the importance weight function $W(x_{0:k}) \equiv W_k$ admits *recursive* computation

$$W_k = \frac{\phi(x_{0:k})}{q(x_{0:k}|y_{1:k})} = \frac{p(y_k|x_k)p(x_k|x_{k-1})}{q(x_k|x_{0:k-1}y_{1:k})} \frac{\phi(x_{0:k-1})}{q(x_{0:k-1}|y_{1:k-1})} \quad (12)$$

$$= \frac{p(y_k|x_k)p(x_k|x_{k-1})}{q(x_k|x_{0:k-1}, y_{1:k})} W_{k-1} \equiv u_{k|0:k-1} W_{k-1} \quad (13)$$

Suppose we had an approximation to the posterior (in the sense $\langle f(x) \rangle_\phi \approx \sum_i W_{k-1}^{(i)} f(x_{0:k-1}^{(i)})$)

$$\phi(x_{0:k-1}) \approx \sum_i W_{k-1}^{(i)} \delta(x_{0:k-1} - x_{0:k-1}^{(i)})$$

$$x_k^{(i)} \sim q(x_k|x_{0:k-1}^{(i)}, y_{1:k}) \qquad \text{Extend trajectory}$$

$$W_k^{(i)} = u_{k|0:k-1}^{(i)} W_{k-1} \qquad \text{Update weight}$$

$$\phi(x_{0:k}) \approx \sum_i W_k^{(i)} \delta(x_{0:k} - x_{0:k}^{(i)})$$

Example

- Prior as the proposal density

$$q(x_k|x_{0:k-1}, y_{1:k}) \quad = \quad p(x_k|x_{k-1})$$

- The weight is given by

$$\begin{aligned} x_k^{(i)} &\sim p(x_k|x_{k-1}^{(i)}) && \text{Extend trajectory} \\ W_k^{(i)} &= u_{k|0:k-1}^{(i)} W_{k-1} && \text{Update weight} \\ &= \frac{p(y_k|x_k^{(i)})p(x_k^{(i)}|x_{k-1}^{(i)})}{p(x_k^{(i)}|x_{k-1}^{(i)})} W_{k-1}^{(i)} = p(y_k|x_k^{(i)}) W_{k-1}^{(i)} \end{aligned}$$

- However, this schema will **not** work, since we blindly sample from the prior. But ...

Example (cont.)

- Perhaps surprisingly, interleaving importance sampling steps with (occasional) resampling steps makes the approach work quite well !!

$$x_k^{(i)} \sim p(x_k|x_{k-1}^{(i)}) \qquad \text{Extend trajectory}$$

$$W_k^{(i)} = p(y_k|x_k^{(i)})W_{k-1}^{(i)} \qquad \text{Update weight}$$

$$\tilde{W}_k^{(i)} = W_k^{(i)}/\tilde{Z}_k \qquad \text{Normalize } (\tilde{Z}_k \equiv \sum_{i'} W_k^{(i')})$$

$$x_{0:k,\text{new}}^{(j)} \sim \sum_{i=1}^{N} \tilde{W}^{(i)} \delta(x_{0:k} - x_{0:k}^{(i)}) \qquad \text{Resample } \; j = 1 \ldots N$$

- This results in a new representation as

$$\phi(x) \quad \approx \quad \frac{1}{N} \sum_j \tilde{Z}_k \delta(x_{0:k} - x_{0:k,\text{new}}^{(j)})$$

$$x_{0:k}^{(i)} \leftarrow x_{0:k,\text{new}}^{(j)} \qquad W_k^{(i)} \leftarrow \tilde{Z}_k/N$$

Optimal proposal distribution

- The algorithm in the previous example is known as *Bootstrap particle filter* or *Sequential Importance Sampling/Resampling* (SIS/SIR).

- Can we come up with a better proposal in a sequential setting?
  - We are not allowed to move previous sampling points $x_{1:k-1}^{(i)}$ (because in many applications we can't even store them)
  - Better in the sense of minimizing the variance of weight function $W_k(x)$. (remember the optimality story in Eq.(10) and set $f(x) = 1$).

- The answer turns out to be the filtering distribution

$$q(x_k|x_{1:k-1}, y_{1:k}) = p(x_k|x_{k-1}, y_k) \tag{14}$$

Optimal proposal distribution (cont.)

- The weight is given by

$$
\begin{aligned}
x_k^{(i)} &\sim p(x_k|x_{k-1}^{(i)}, y_k) && \text{Extend trajectory} \\
W_k^{(i)} &= u_{k|0:k-1}^{(i)} W_{k-1}^{(i)} && \text{Update weight} \\
u_{k|0:k-1}^{(i)} &= \frac{p(y_k|x_k^{(i)})p(x_k^{(i)}|x_{k-1}^{(i)})}{p(x_k^{(i)}|x_{k-1}^{(i)}, y_k)} \times \frac{p(y_k|x_{k-1}^{(i)})}{p(y_k|x_{k-1}^{(i)})} \\
&= \frac{p(y_k, x_k^{(i)}|x_{k-1}^{(i)})p(y_k|x_{k-1}^{(i)})}{p(x_k^{(i)}, y_k|x_{k-1}^{(i)})} = p(y_k|x_{k-1}^{(i)})
\end{aligned}
$$

A Generic Particle Filter

1. **Generation**:
   Compute the proposal distribution $q(x_k|x_{0:k-1}^{(i)}, y_{1:k})$.
   Generate offsprings for $i = 1 \ldots N$

$$\hat{x}_k^{(i)} \quad \sim \quad q(x_k|x_{0:k-1}^{(i)}, y_{1:k})$$

2. **Evaluate** importance weights

$$W_k^{(i)} \quad = \quad \frac{p(y_k|\hat{x}_k^{(i)})p(\hat{x}_k^{(i)}|x_{k-1}^{(i)})}{q(\hat{x}_k^{(i)}|x_{0:k-1}^{(i)}, y_{1:k})} W_{k-1}^{(i)} \qquad x_{0:k}^{(i)} = (\hat{x}_k^{(i)}, x_{0:k-1}^{(i)})$$

3. **Resampling** (optional but recommended)

Normalize weigts	$\tilde{W}_k^{(i)} = W_k^{(i)} / \tilde{Z}_k$	$\tilde{Z}_k \equiv \sum_j W_k^{(j)}$
Resample	$x_{0:k,\text{new}}^{(j)} \sim \sum_{i=1}^{N} \tilde{W}^{(i)} \delta(x_{0:k} - x_{0:k}^{(i)})$	$j = 1 \ldots N$
Reset	$x_{0:k}^{(i)} \leftarrow x_{0:k,\text{new}}^{(j)}$	$W_k^{(i)} \leftarrow \tilde{Z}_k / N$

Summary of what we have (hopefully) covered

- Deterministic
  - Variational Bayes, Mean field
  - Expectation/Maximization (EM), Iterative Conditional Modes (ICM)
- Stochastic
  - Markov Chain Monte Carlo
  - Importance Sampling,
  - Particle filtering

Summary of what we have not covered

- Exact Inference (Belief Propagation, Junction Tree ...)
- Deterministic
  - Assumed Density Filter (ADF), Extended Kalman Filter (EKF), Unscented Particle Filter
  - Structured Mean field
  - Loopy Belief Propagation, Expectation Propagation, Generalized Belief Propagation
  - Fractional Belief propagation, Bound Propagation, $<$your favorite name$>$ Propagation
  - Graph cuts ...
- Stochastic
  - Unscented Particle Filter, Nonparametric Belief Propagation
  - Annealed Importance Sampling, Adaptive Importance Sampling
  - Hybrid Monte Carlo, Exact sampling, Coupling from the past

Variational or Sampling?

- Possible criteria
  - **How accurate**
  - **How fast**
  - **How easy to learn**
  - **How easy to code/test/maintain**

When all you own is a hammer, every problem looks like a nail

Variational or Sampling?

- Depends upon application domain. My personal impression is:
  - Sampling dominated
    - Bayesian statistics, Scientific data analysis
    - Finance/auditing
    - Operations research
    - Genetics
    - Tracking
  - Variational dominated
    - Communications/error correcting codes
  - Mixed territory
    - Machine Learning, Robotics
    - Computer Vision
    - Human-Computer Interaction
    - Speech/audio/multimedia analysis/information retrieval
    - Statistical Signal processing

Further Reading

Variational tutorials and overviews

- Tommi Jaakkola. *Tutorial on variational approximation methods.* (2000). `http://people.csail.mit.edu/tommi/papers/Jaa-var-tutorial.ps`
- Frey and Jojic [2]
- Wainwright and Jordan [8]

MCMC and SMC tutorials and overviews

- Andrieu, de Freitas, Doucet, Jordan. *An Introduction to MCMC for Machine Learning*, 2001
- Andrieu. *Monte Carlo Methods for Absolute beginners*, 2004
- Doucet, Godsill, Andrieu. ”On Sequential Monte Carlo Sampling Methods for Bayesian Filtering”, Statistics and Computing, vol. 10, no. 3, pp. 197-208, 2000

The “in Practice” Books

- Gilks, Richardson, Spiegelhalter, *Markov Chain Monte Carlo in Practice*, Chapman Hall, 1996
- Doucet, de Freitas, Gordon, *Sequential Monte Carlo Methods in Practice*, Springer, 2001

References

[1] J.E. Besag. On the statistical analysis of dirty pictures (with discussion). Jr. R. Stat. Soc. B, 48:259–302, 1986.

[2] B. J. Frey and N. Jojic. A comparison of algorithms for inference and learning in probabilistic graphical models. IEEE Transactions on Pattern Analysis and Machine Intelligence, 27(9), 2005.

[3] Z. Ghahramani and M. Beal. Propagation algorithms for variational Bayesian learning. In Neural Information Processing Systems 13, 2000.

[4] E. T. Jaynes. Probability Theory, The Logic of Science. Cambridge University Press, edited by G. L. Bretthorst, 2003.

[5] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. IEEE Transactions on Information Theory, 47(2):498–519, February 2001.

[6] D. J. C. MacKay. Information Theory, Inference and Learning Algorithms. Cambridge University Press, 2003.

[7] Radford M. Neal and Geoffrey E. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In Learning in graphical models, pages 355–368. MIT Press, 1999.

[8] M. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. Technical Report 649, Department of Statistics, UC Berkeley, September 2003.