Probabilistic Graphical Models for Image Analysis - Lecture 5

Stefan Bauer

19 October 2018

Max Planck ETH Center for Learning Systems

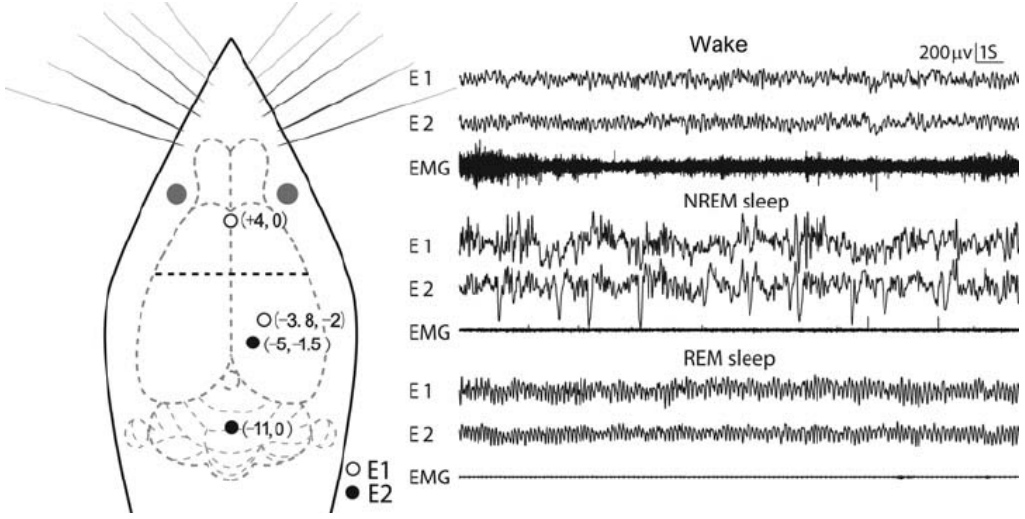
Overview

1. Motivational Examples

2. Sequential Data

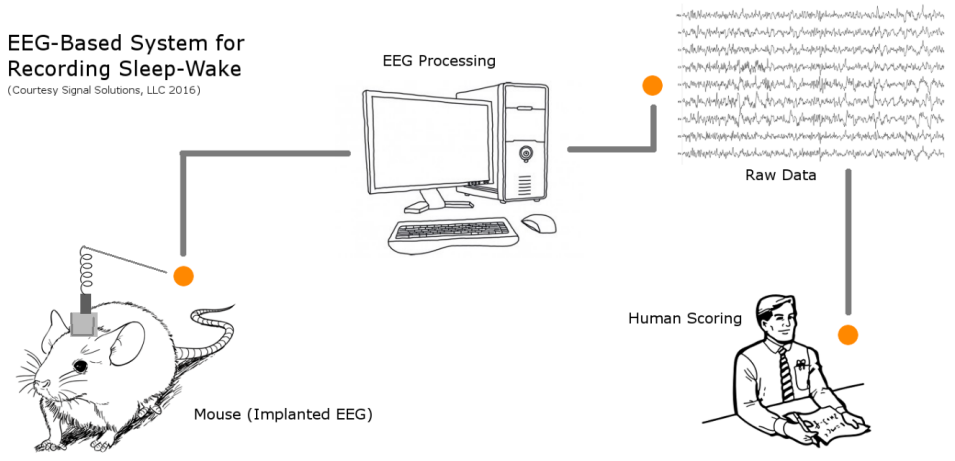
Motivational Examples

Sleep scoring in animals



- Sleep monitoring in animals is commonly done through vigilance state classification of EEG/EMG recordings
- EEG/EMG signals are partitioned into short epochs of equal size
- Each epoch is then individually scored accordingly, w.r.t. corresponding vigilance state

Sleep scoring in animals



Typical experimental pipeline:

1. Perform "intervention" on an animal subset
2. Record EEG/EMG signals over some period of time
3. Manually score EEG/EMG
4. Perform statistical posthoc analysis on scored data

Manual sleep scoring is a bottleneck

- Slow!
- Laborious
- Prone to human errors
- Non-standardized
- Decoupled from posthoc analysis

Automating sleep scoring

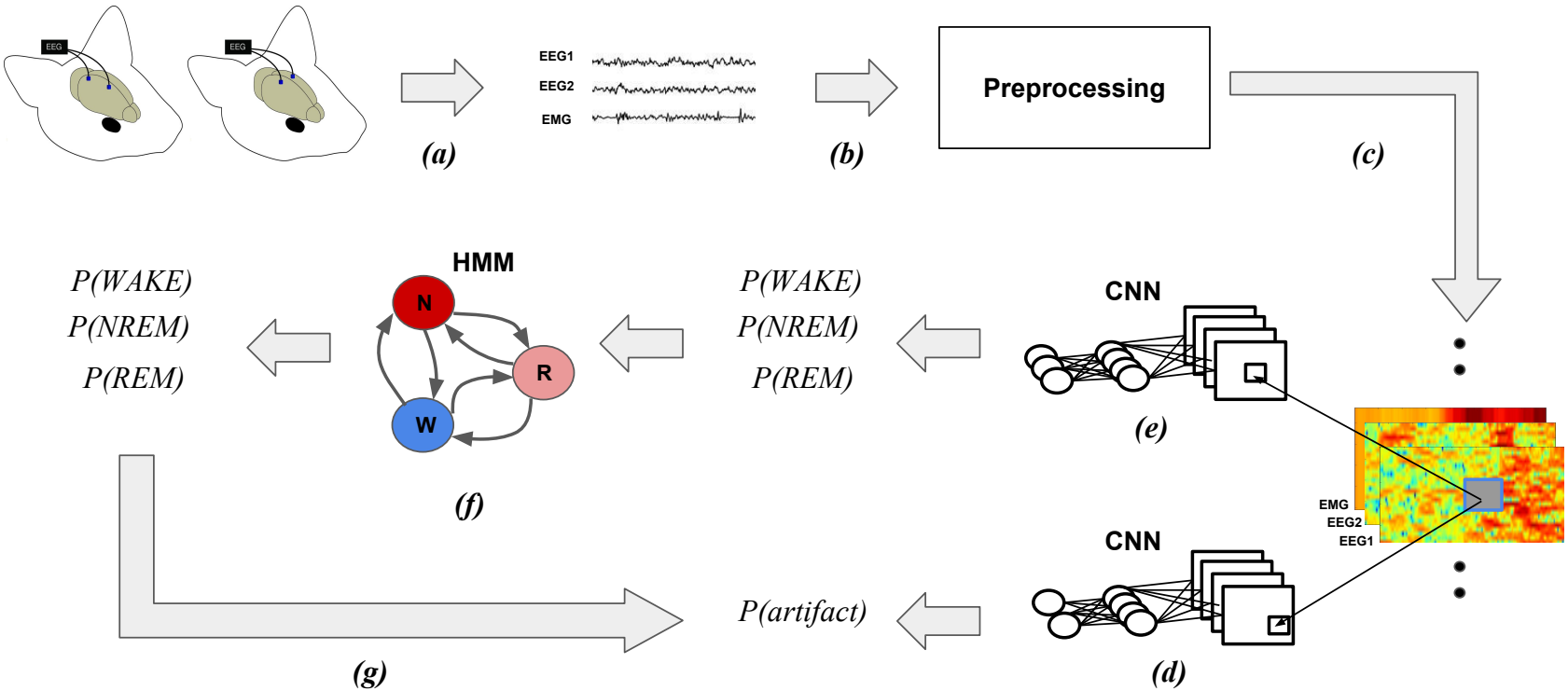
Some research efforts aim to replace visual inspection

- Automation of sleep scoring for both animals* and humans
- Current state-of-the-art solution offer promising prediction performance
- Some generalization issues of current solutions still remain

---

*Sunagawa, G. A., Sei, H., Shimba, S., Urade, Y., & Ueda, H. R. (2013). FASTER: an unsupervised fully automated sleep staging method for mice. Genes to Cells, 18(6), 502-518.

State of the art - Sleep Scoring (paper under review)



Djordje Miladinovic et.al. End-to-end learning from EEG/EMG to extrapolate animal sleep scoring across experimental settings, labs and species, under submission

Images to Torques

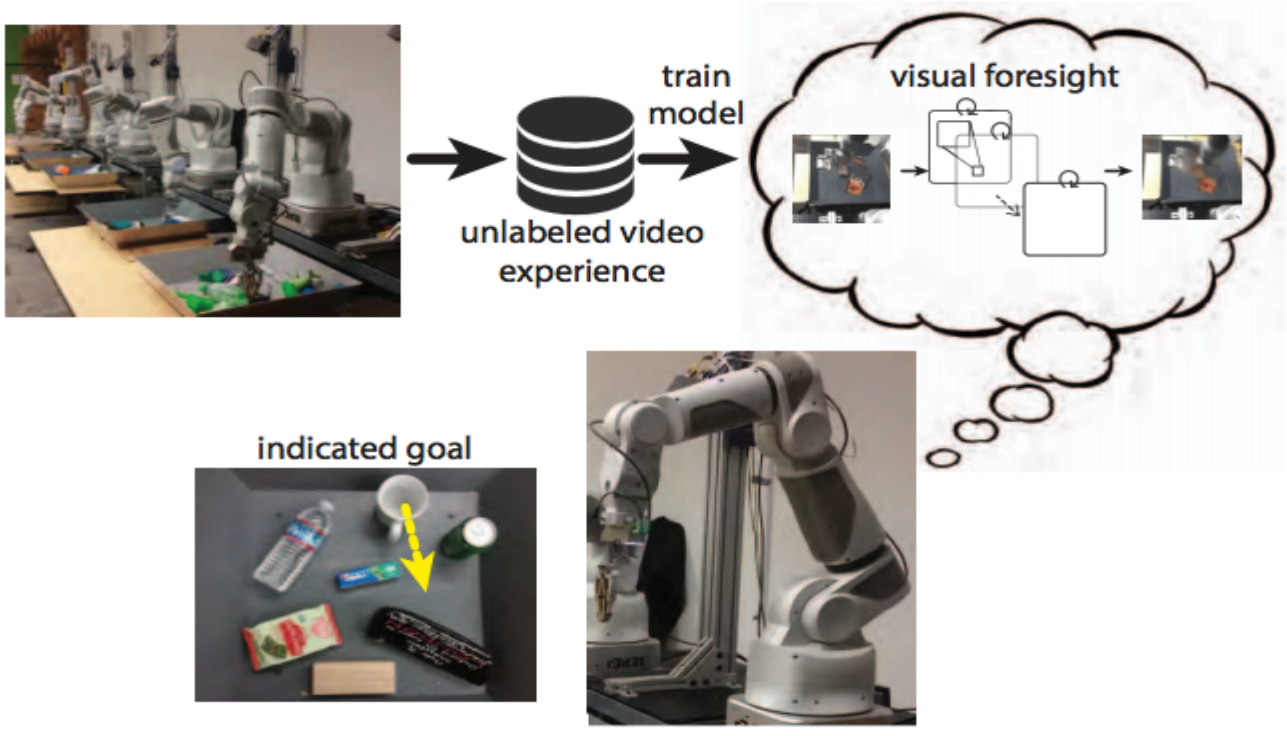
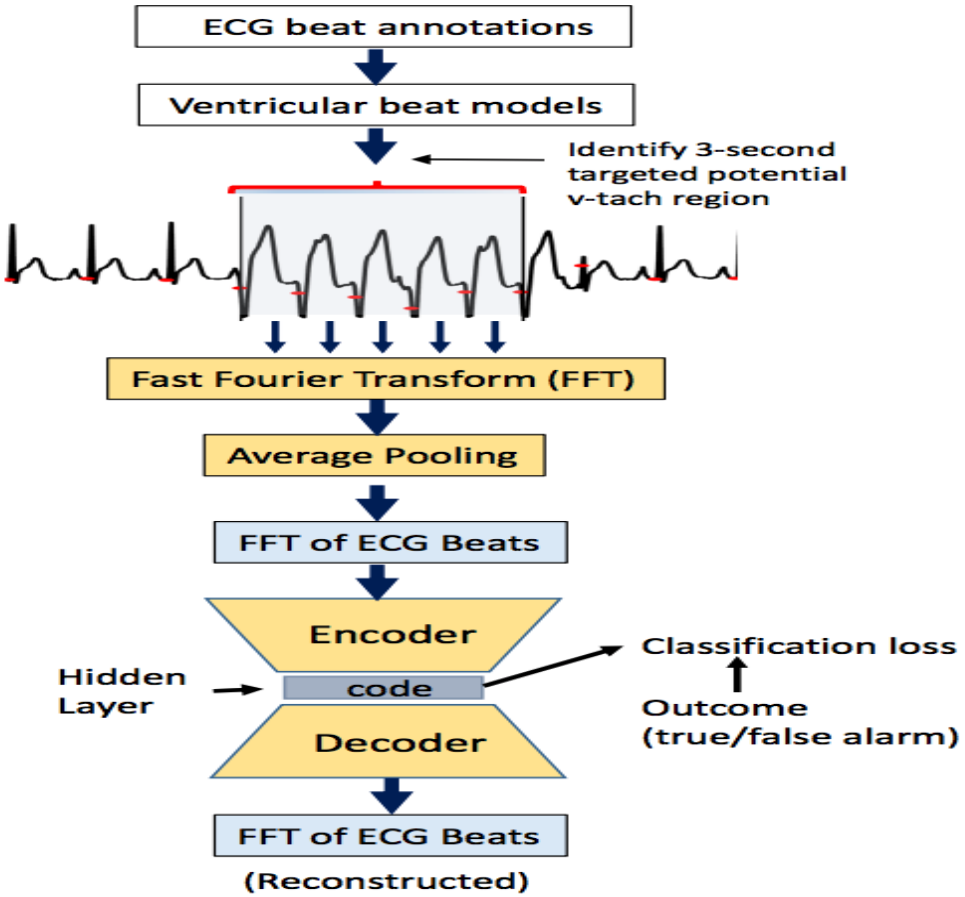


Fig. 1. Using our approach, a robot uses a learned predictive model of images, i.e. a visual imagination, to push objects to desired locations.

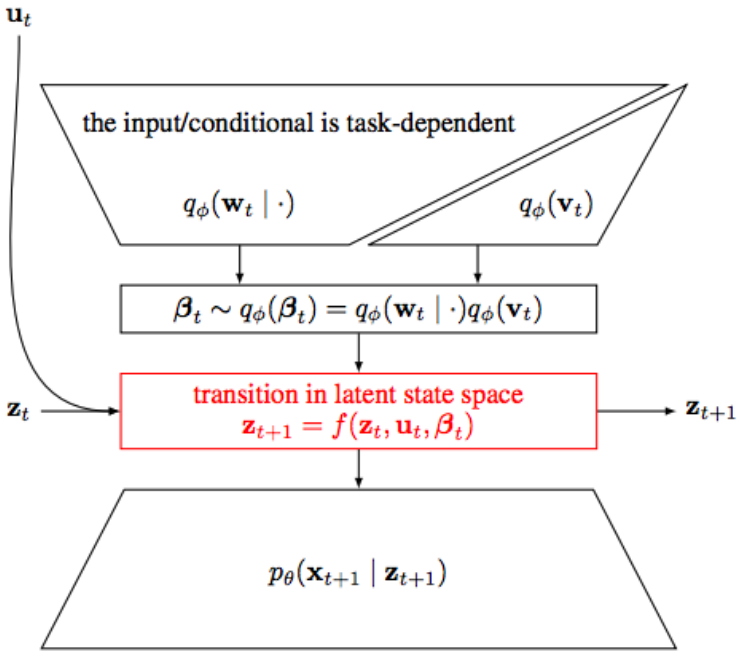
Finn and Levine, Deep Visual Foresight for Planning Robot Motion, ICRA 2017

False Alarms ICU

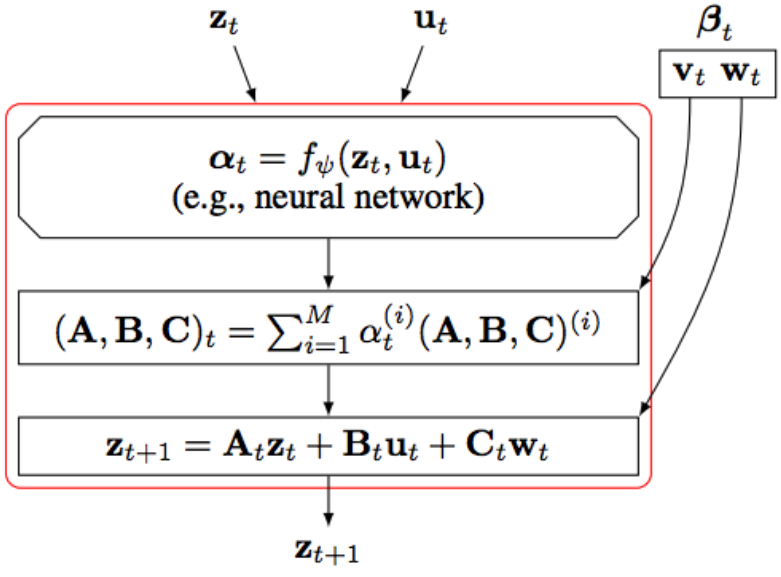


Lehman et.al Representation Learning Approaches to Detect False Arrhythmia Alarms from ECG Dynamics, MLHC 2018

Deep Bayes



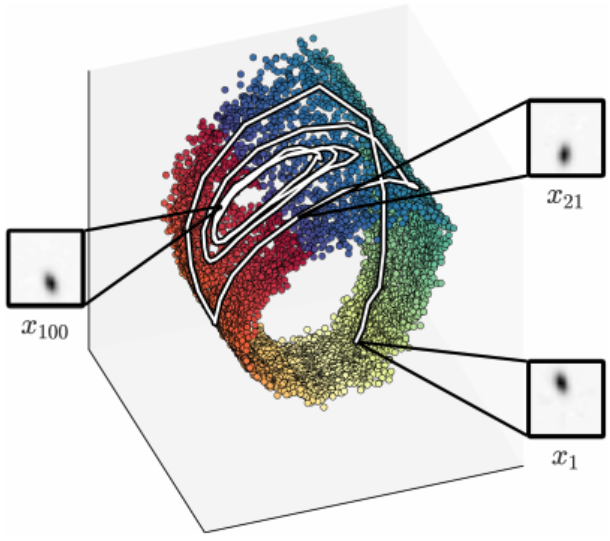
(a) General scheme for arbitrary transitions.



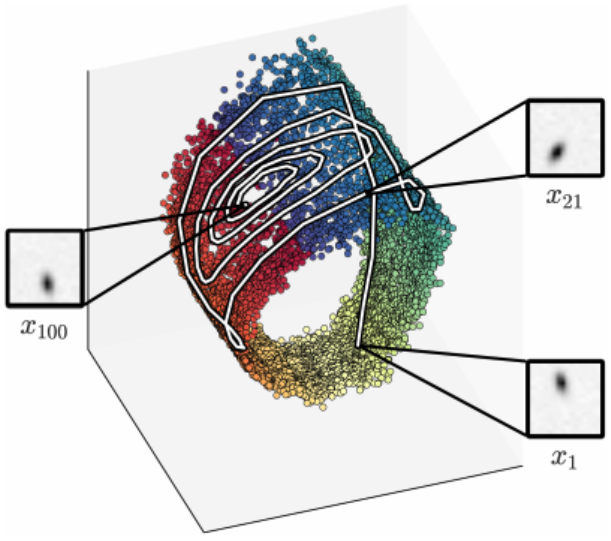
(b) One particular example of a latent transition: local linearity.

Karl et. al. Deep Variational Bayes Filters: Unsupervised Learning of State Space Models from Raw Data, ICLR 2017

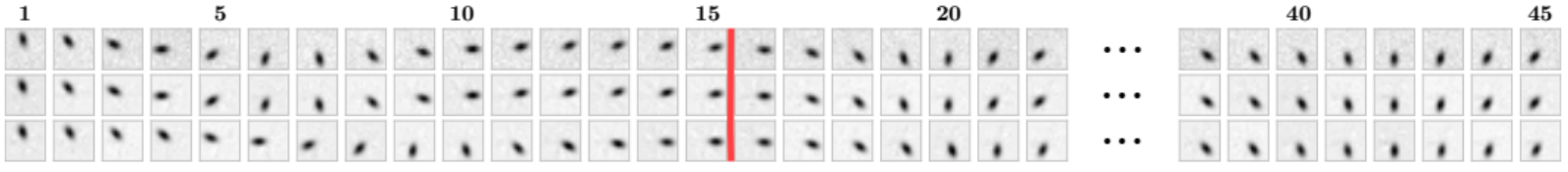
Deep Bayes



(a) Generative latent walk.



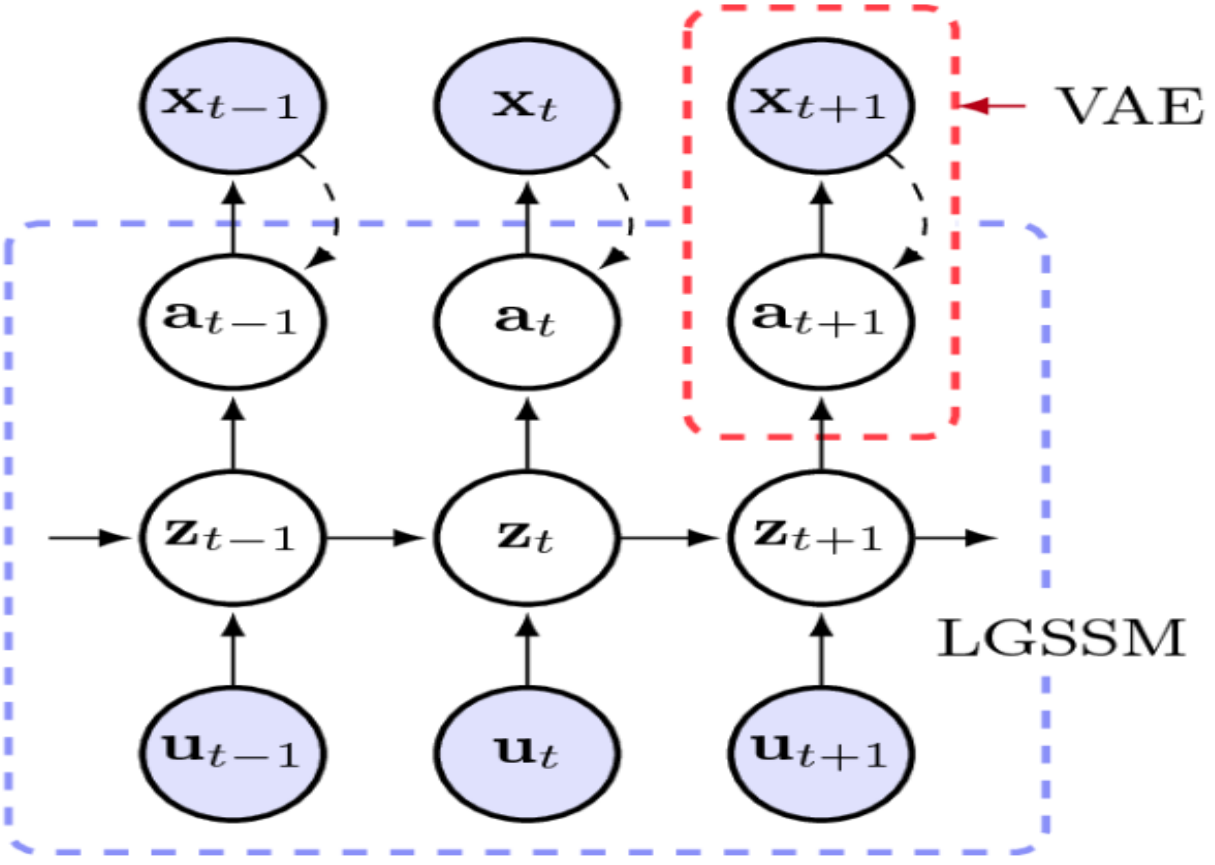
(b) Reconstructive latent walk.



(c) Ground truth (top), reconstructions (middle), generative samples (bottom) from identical initial latent state.

Karl et. al. Deep Variational Bayes Filters: Unsupervised Learning of State Space Models from Raw Data, ICLR 2017

Kalman Variational Autoencoders



Fraccaro et. al. A Disentangled Recognition and Nonlinear Dynamics Model for Unsupervised Learning, NIPS 2017

Additional Papers (random selection and order)

- Linderman et.al Bayesian Learning and Inference in Recurrent Switching Linear Dynamical Systems, AISTATS 2017
- Archer et.al Black box variational inference for state space models, Workshop ICLR 2016
- Doerr et.al Probabilistic Recurrent State-Space Models, ICML 2018
- Krishnan et.al, Deep Kalman Filters, Workshop NIPS 2015
- Krishnan et.al, Structured Inference Networks for Nonlinear State Space Models, AAAI 2017
- Johnson et.al. Composing graphical models with neural networks for structured representations and fast inference, NIPS 2016
- . . .

Sequential Data

Motivation

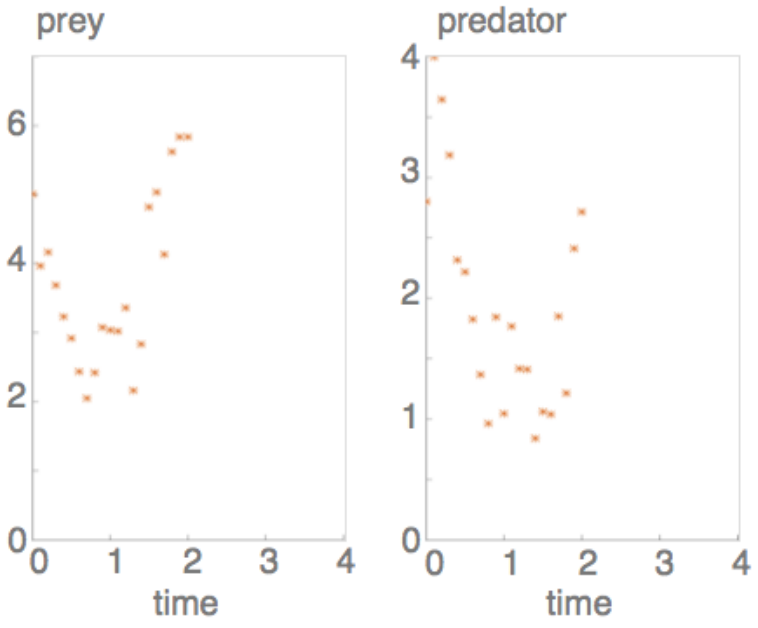
$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \theta)$$

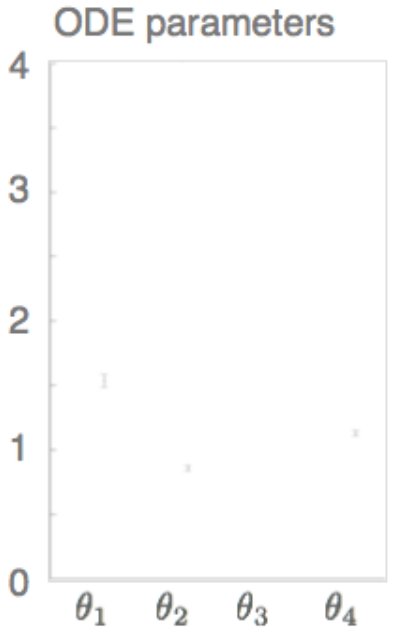
$$\dot{x}_1(t) := \theta_1 x_1(t) - \theta_2 x_1(t) x_2(t)$$
$$\dot{x}_2(t) := -\theta_3 x_2(t) + \theta_4 x_1(t) x_2(t)$$

Motivation

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \theta)$$
$$\mathbf{y}(t) = \mathbf{x}(t) + \varepsilon(t),$$

$$\dot{x}_1(t) := \theta_1 x_1(t) - \theta_2 x_1(t) x_2(t)$$
$$\dot{x}_2(t) := -\theta_3 x_2(t) + \theta_4 x_1(t) x_2(t)$$

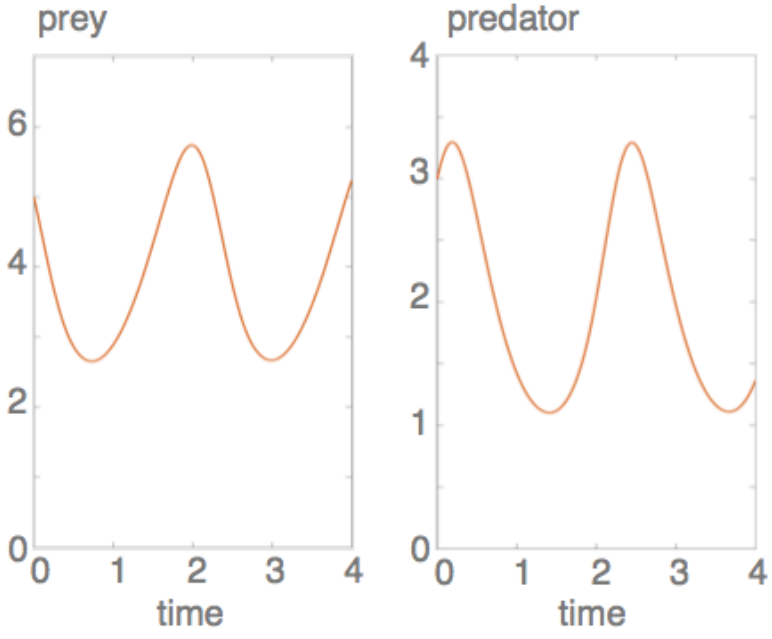


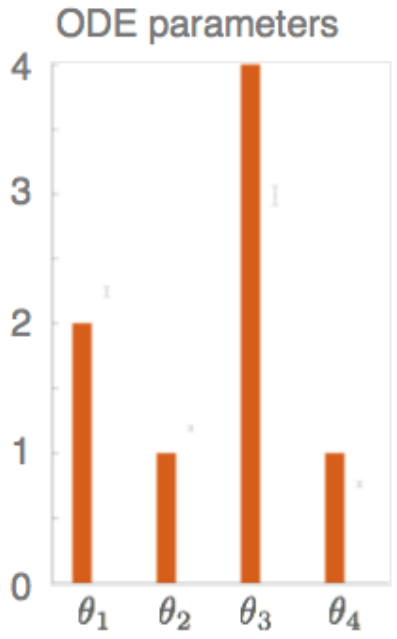


Motivation

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \theta)$$
$$\mathbf{y}(t) = \mathbf{x}(t) + \varepsilon(t),$$

$$\dot{x}_1(t) := \theta_1 x_1(t) - \theta_2 x_1(t) x_2(t)$$
$$\dot{x}_2(t) := -\theta_3 x_2(t) + \theta_4 x_1(t) x_2(t)$$

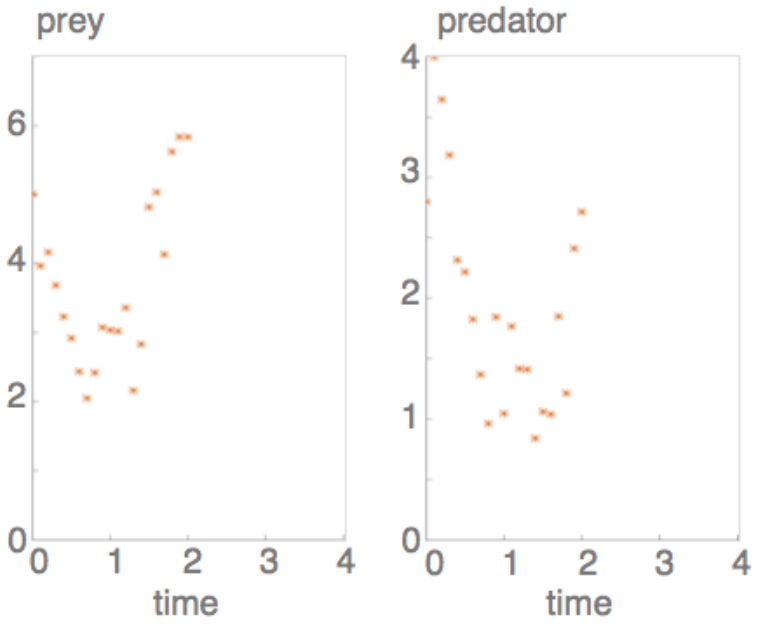


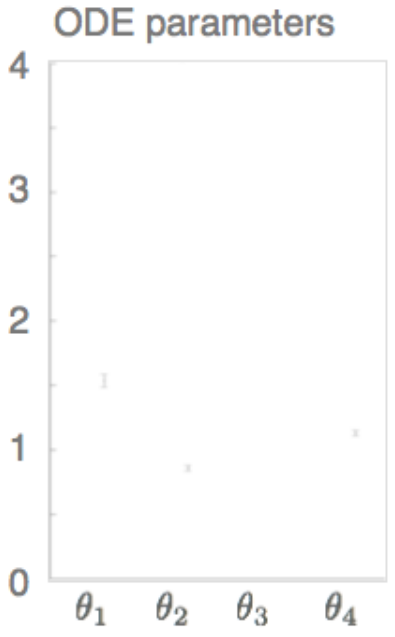


Lotka-Volterra

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \theta)$$
$$\mathbf{y}(t) = \mathbf{x}(t) + \varepsilon(t),$$

$$\dot{x}_1(t) := \theta_1 x_1(t) - \theta_2 x_1(t) x_2(t)$$
$$\dot{x}_2(t) := -\theta_3 x_2(t) + \theta_4 x_1(t) x_2(t)$$

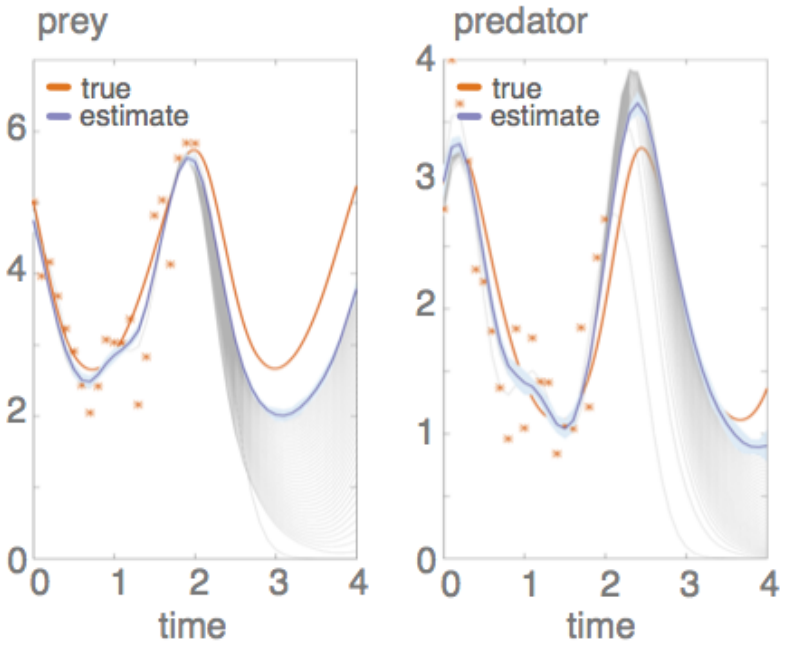


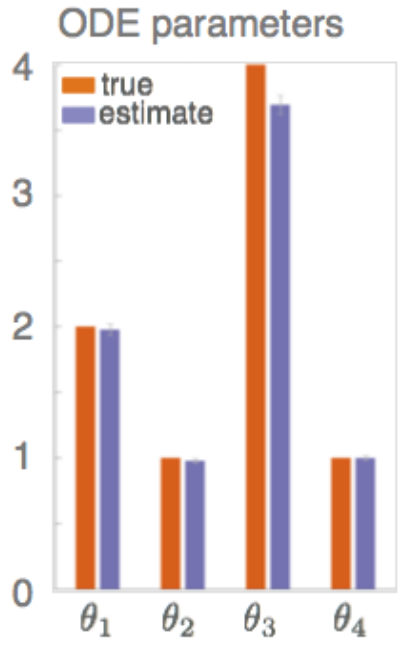


Lotka-Volterra

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \theta)$$
$$\mathbf{y}(t) = \mathbf{x}(t) + \varepsilon(t),$$

$$\dot{x}_1(t) := \theta_1 x_1(t) - \theta_2 x_1(t) x_2(t)$$
$$\dot{x}_2(t) := -\theta_3 x_2(t) + \theta_4 x_1(t) x_2(t)$$





Algorithm

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \boldsymbol{\theta})$$
$$\mathbf{y}(t) = \mathbf{x}(t) + \boldsymbol{\varepsilon}(t),$$

$$\dot{x}_1(t) := \theta_1 x_1(t) - \theta_2 x_1(t) x_2(t)$$
$$\dot{x}_2(t) := -\theta_3 x_2(t) + \theta_4 x_1(t) x_2(t)$$

Dynamic Systems - State Space Models

$$\dot{x}(t) = f(x(t), u(t)), \quad \text{state evolution}$$
$$y(t) = g(x(t), u(t)), \quad \text{observations}$$

Most often used in practice are linear, discrete Systems

$$x(t+1) = Ax(t) + Bu(t)$$
$$y(t) = Cx(t) + Du(t)$$

Black Board: Examples and Connections

New reference: Linear Gaussian Models

A Unifying Review of Linear Gaussian Models

Sam Roweis **Zoubin Ghahramani**

{roweis,zoubin}@gatsby.ucl.ac.uk
Gatsby Computational Neuroscience Unit
University College London
17 Queen Square, London WC1N 3AR, United Kingdom

October 1998
— In Press —
(Neural Computation, Vol. 11 No. 2, 1999)

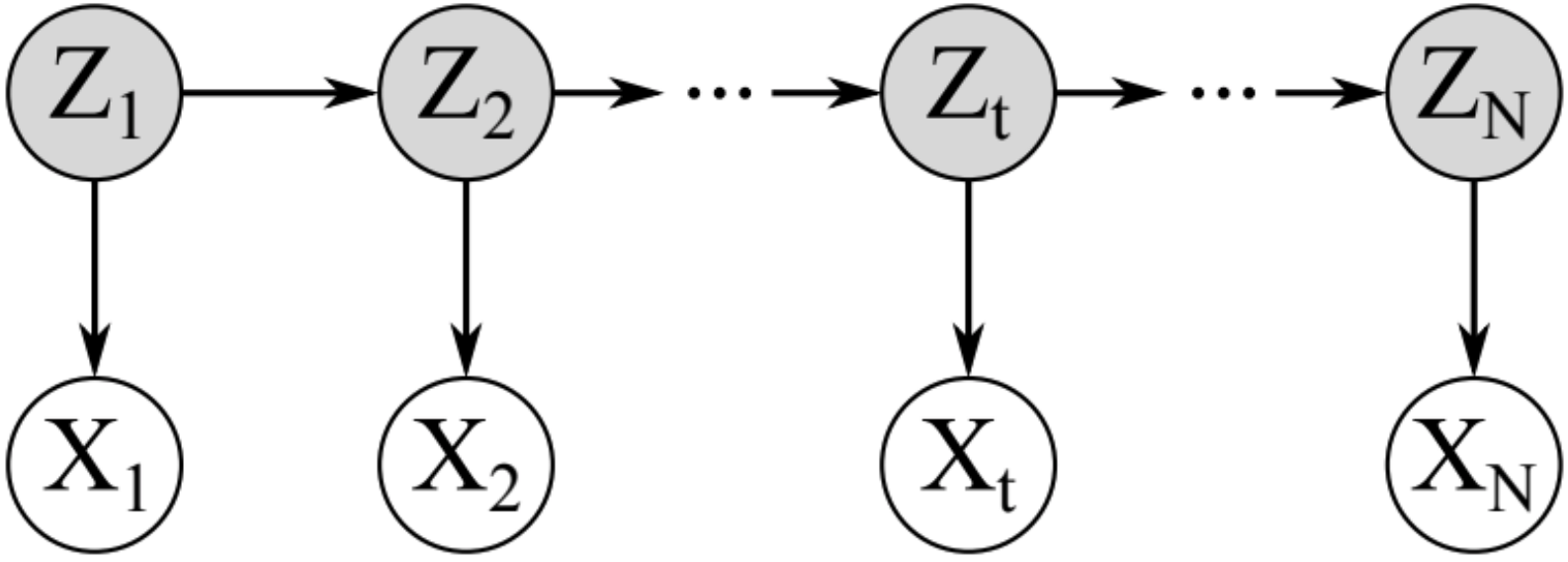
Graphical Models

Most popular Graphical Models

- Hidden Markov Models
  - Speech Recognition
  - Sequence analysis in computational biology
  - activity recognition
  - ...
- Kalman Filter
  - Cruise control in cars
  - GPS navigation devices
  - Tracking
  - ....

Very simple models but very powerful!

Inference tasks

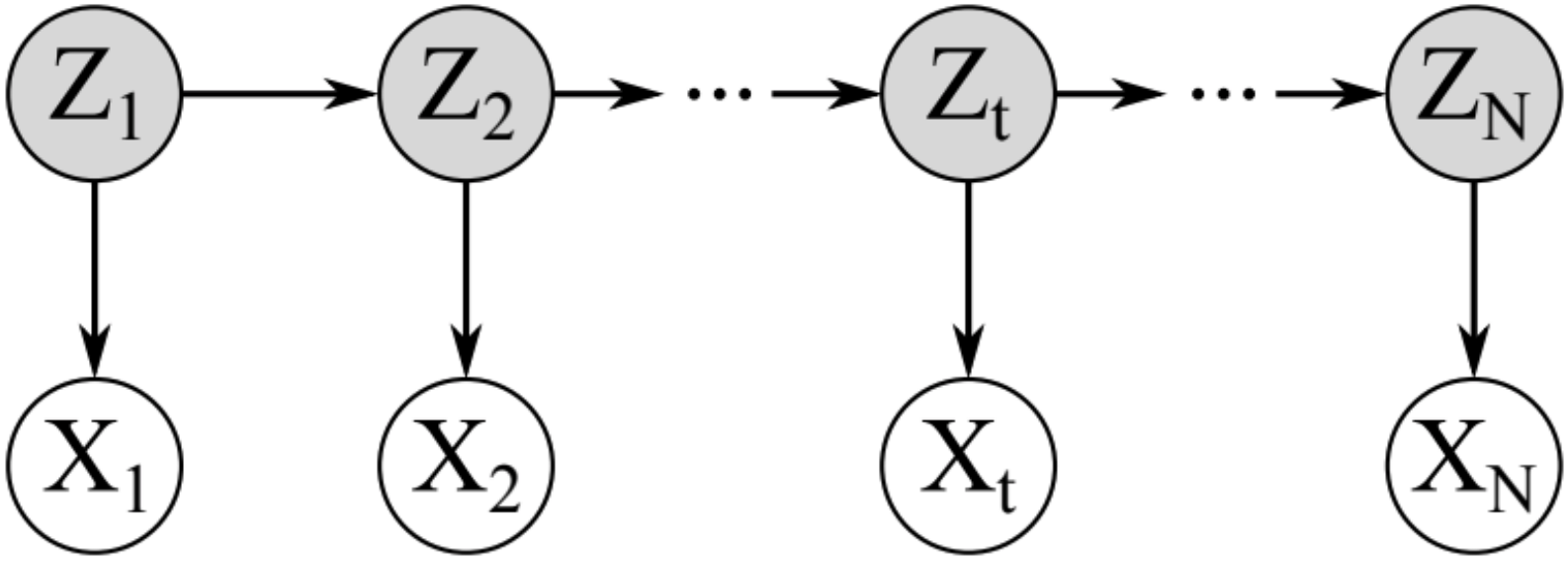


Filtering $P(Z_t|X_{1:t})$

Prediction $P(Z_{t+\tau}|X_{1:t})$

Smoothing $P(Z_t|X_{1:t})$ for $1 \leq t \leq T$

HMM and Kalman Filter



HMM: Z_i Multinomial, X_i arbitrary

Kalman: Z_i, X_i Gaussian

Extended Kalman: Z_i Gaussian, X_i arbitrary

Recall EM Algorithm (Lecture 2)

Need to maximize

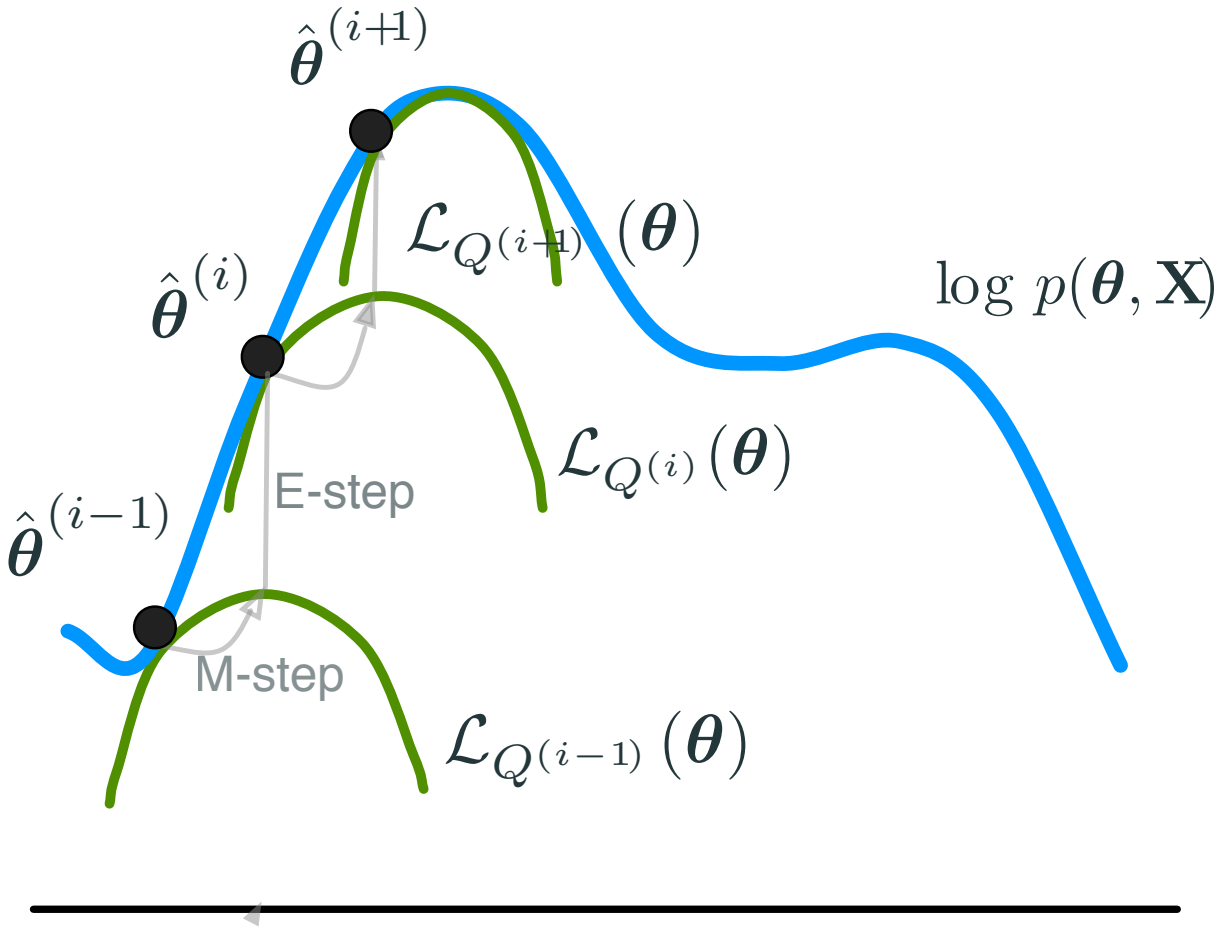
$$\log p(\mathcal{D}) = \sum_{x \in D} \log p(x) = \sum_{x \in D} \log \left(\sum_{z} p(x|z)p(z) \right)$$

Problem: Only x is observed but we have parameters θ and latent variables z

The Expectation Maximization (EM) algorithm:

- **Expectation**: Assign values to hidden/missing variables i.e. compute $p(z|x;\theta_t)$
- **Maximization**: Maximize parameter log likelihood via $\theta_{t+1} = \arg\max_\theta \sum_{x \in D} \mathbb{E}_{z \sim p(z|x,\theta_t)} \log p(x,z,\theta)$
- Repeat until convergence for $t = 1, 2, \cdots$, starting with θ_0

Illustration EM



EM using Jensen

Y observations, X latent states, θ parameters.

$$
\begin{aligned}
\log P(Y|\theta) &= \log \sum_{\mathcal{X}} P(Y, X|\theta) \\
&= \log \sum_{\mathcal{X}} p(X, Y|\theta) \frac{q(X)}{q(X)} \\
&\geq \sum_{\mathcal{X}} q(X) \log \frac{p(X, Y|\theta)}{q(X)} \\
&= \sum_{\mathcal{X}} q(X) \log p(X, Y|\theta) - \sum_{\mathcal{X}} q(X) \log q(X) \\
&= \mathcal{L}(q, \theta)
\end{aligned}
$$

Learning HMMs using the EM algorithm

$$\log P(X_{1:T}, Y_{1:t}) = \log P(X_1) + \sum_{t=1}^{T} \log P(Y_t|X_t) + \sum_{t=2}^{T} \log P(X_t|X_{t-1})$$

Hidden Markov Model i.e. X_t categorical (with K values). Thus we can represent X_t as a K dimensional unit vector e.g. for taking on the second value:

$$X_t = [010 \cdots 0]^{\mathsf{T}}$$

The transition probability can then be written as:

$$P(X_t|X_{t-1}) = \prod_{i=1}^{K} \prod_{j=1}^{K} A_{ij}^{X_{t,i}, X_{t-1,j}}$$

where A_{ij} is the transition matrix, with non-negative entries and each row sums to 1.

State transition models

$$\log P(X_t|X_{t-1}) = \sum_{i=1}^{K}\sum_{j=1}^{K} X_{t,i}X_{t-1,j}\log A_{ij} = X_t^{\mathsf{T}}(\log A)X_{t-1}$$

Similarly if initial state probabilities are arranged in a vector π, of dimension $K \times 1$ with $\pi_i = P(X_{1i=1})$, then

$$P(X_1|\pi) = \prod_{i=1}^{K} \pi_i^{X_{1i}}$$

and

$$\log P(X_1) = X_1^{\mathsf{T}} \log \pi$$

Observation model

If Y_t is discrete and can take on D values, we can again write

$$\log P(Y_t|X_t) = Y_t^\mathsf{T}(\log B)X_t$$

where B is a $D \times K$ dimensional emission probability matrix.

The final parameter set of the model is then

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

Goal: $\arg\max_\theta \log P(Y)$

Expectation Maximization for HMM

M-Step

$$A_{ij} \propto \sum_{t=2}^{T} \mathbb{E}[X_{t,i} X_{t-1,j}] \leftarrow \frac{\sum_{t=2}^{T} \mathbb{E}[X_{t,i} X_{t-1,j}]}{\sum_{t=2}^{T} \mathbb{E}[X_{t-1,j}]} \tag{1}$$

$$\pi \leftarrow \mathbb{E}[X_{1,i}] \tag{2}$$

$$B_{di} \leftarrow \frac{\sum_{t=1}^{T} Y_{t,d} \mathbb{E}[X_{t,i}]}{\sum_{t=1}^{T} \mathbb{E}[X_{t,i}]} \tag{3}$$

The forward algorithm

$$\alpha_t = P(Y_{t+1:T}|X_t) =$$

$$= \left[\sum_{X_{t-1}} P(X_{t-1}, Y_{1:t-1})P(X_t|X_{t-1})\right] P(Y_t|X_t)$$

$$= \left[\sum_{X_{t-1}} \alpha_{t-1}P(X_t|X_{t-1})\right] P(Y_t|X_t)$$

The backward algorithm

$$
\begin{aligned}
\beta_t &= P(X_t, Y_{1:t}) = \\
&= \sum_{X_{t+1}} P(Y_{t+2:T}|X_{t+1}) P(X_{t+1}|X_t) P(Y_{t+1}|X_{t+1}) \\
&= \sum_{X_{t+1}} \beta_{t+1} P(X_{t+1}|X_t) P(Y_{t+1}|X_{t+1})
\end{aligned}
$$

Expectations for EM

E-Step

$$\mathbb{E}[X_{t,i}] = \gamma_{ti} = \frac{\alpha_{t,i}\beta_{t,i}}{\sum_j \alpha_{t,j}\beta_{t,j}}$$

$$\mathbb{E}[X_{t,i}X_{t-1,j}] = \zeta_{tij} = \frac{\alpha_{t-1,j}A_{ij}P(Y_t|X_{t,i})\beta_{t,i}}{\sum_{k,l} \alpha_{t-1,k}A_{kl}P(Y_t|X_{t,l})\beta_{t,l}}$$

Exercise: Kalman Filter (update equations, Bishop Chapter 13, Appendix 3 of review paper, idea next slide)

Linear Gaussian State Space Models

Assumption: Initial states are Gaussian distributed:

$$x_1 \sim \mathcal{N}(\mu_1, Q_1)$$

With linear dynamics all future states x_t and observations will be Gaussian distributed:

$$P(x_{t+1}|x_t) = \mathcal{N}(Ax_t, Q)$$
$$P(y_t|x_t) = \mathcal{N}(Cx_t, R)$$

With Markov property it follows:

$$P(X_{1:T}, Y_{1:T}) = P(x_1) \prod_{t=2}^{T} P(x_t|x_{t-1}) \prod_{t=1}^{T} P(y_t|x_t)$$

Linear Gaussian State Space Models II

From before

$$P(X_{1:T}, Y_{1:T}) = P(x_1) \prod_{t=2}^{T} P(x_t|x_{t-1}) \prod_{t=1}^{T} P(y_t|x_t)$$

Each of the above densities is Gaussian, thus:

$$\begin{aligned} -2 \log P(X_{1:T}, Y_{1:T}) = & \sum_{t=1}^{T} [(y_t - Cx_t)^\intercal R^{-1} (y_t - Cx_t) + \log |R|] \\ & + \sum_{t=1}^{T-1} [(x_{t+1} - Ax_t)^\intercal Q^{-1} (x_{t+1} - Ax_t) + log|Q|] \\ & + (x_1 - \mu_1)^\intercal Q_1^{-1} (x_1 - \mu_1) + \text{const.} \end{aligned}$$

Method: Again EM, M-Step e.g. $C \leftarrow (\sum_t y_t x_t^\intercal)(\sum_t x_t x_t^\intercal)^{-1}$

Problem x is hidden <- use expectations! (kalman smoother)

Summary

Linear Gaussian State Space Models

Problems:

- state dynamics can be non-linear
- relations between observed and latent states can be non-linear
- noise can be non-Gaussian

Extensions and Generalisations (Next week)

Next week

Plan for next week

- Factor Graph
- Viterbi Algorithm
- Extensions: Switching, Factorial, Recurrency

Questions?