STA 414/2104:
Statistical Methods in Machine Learning II

Week 6: HMMs and Variational Inference I

Piotr Zwiernik

University of Toronto

Table of contents

Hidden Markov Models (HMMs)

Sequential data

We generally assume data was i.i.d, however this may be a poor assumption:

- Sequential data is common in time-series modelling (e.g. stock prices, speech, video analysis) or ordered (e.g. textual data, gene sequences).
- Recall the general joint factorization via the chain rule

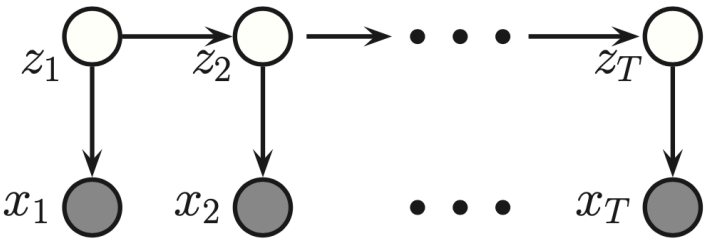
$$p(x_{1:T}) = \prod_{t=1}^{T} p(x_t|x_{t-1}, ..., x_1) \quad \text{where } p(x_1|x_0) = p(x_1).$$

- But this quickly becomes intractable for high-dimensional data -each factor requires exponentially many parameters to specify as a function of T.
- So we **made** the simplifying assumption that our data can be modeled as a **first-order Markov chain**

$$p(x_t|x_{1:(t-1)}) = p(x_t|x_{t-1})$$

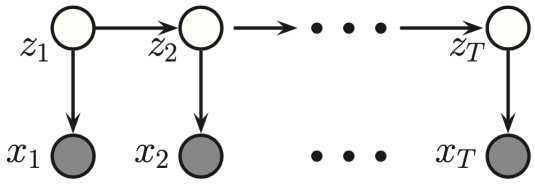
Sequential data

- In certain cases, Markov chain assumption is also restrictive.
- The state of our variables is fully observed. Hence, we introduce Hidden Markov Models



Hidden Markov Models (HMMs)

- HMMs hide the temporal dependence by keeping it in the unobserved state.
- No assumptions on the temporal dependence of observations is made.
- For each observation x_t, we associate a corresponding unobserved hidden/latent variable z_t



- The joint distribution of the model becomes

$$p(x_{1:T}, z_{1:T}) = p(z_1) \prod_{t=2}^{T} p(z_t|z_{t-1}) \prod_{t=1}^{T} p(x_t|z_t)$$

Hidden Markov Models (HMMs)

In HMMs the observations are not limited by a Markov assumption of any order.

Assuming we have a homogeneous model, we only have to know three sets of distributions

1. **Initial distribution**: $\pi(i) = p(z_1 = i)$. The probability of the first hidden variable being in state i; often denoted $\pi \in \mathbb{R}^K$.
2. **Transition distribution**: $A_{ij} = p(z_{t+1} = j | z_t = i) \quad i \in \{1, ..., K\}$. The probability of moving from hidden state i to hidden state j; $A \in \mathbb{R}^{K \times K}$.
3. **Emission probability**: $p(x_t = j | z_t = i)$. The probability of an observed random variable x_t given the state of the hidden variable that "emitted" it.
   (Often think about x_t as discrete but all that follows works in the continuous case)

HMMs: Objectives

We consider the following objectives:

1. Compute the probability of a latent sequence given an observation sequence. That is, we want to be able to compute $p(z_{1:t}|x_{1:t})$. This is achieved with the **Forward-Backward algorithm**.
2. Infer the most likely sequence of hidden states. That is, we want to be able to compute

$$z^\star = \underset{z_{1:T}}{\text{argmax}}\ p(z_{1:T}|x_{1:T}).$$

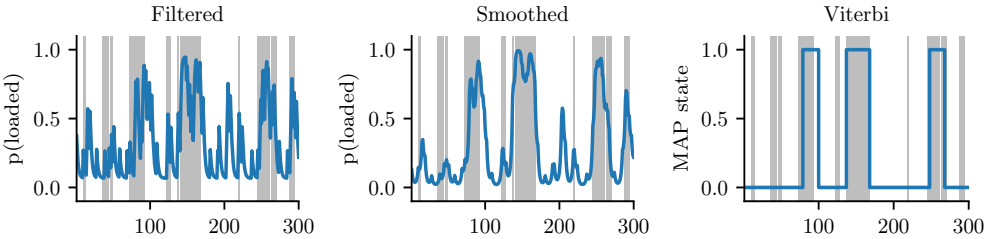
This is achieved using the **Viterbi algorithm**.

Example: Ocasionally dishonest casino

Series of die rolls $x_t \in \{1, \ldots, 6\}$, $z_t \in \{1, 2\}$ (fair/loaded)
$p(x_t|z_t = 1) = (\frac{1}{6}, \frac{1}{6}, \frac{1}{6}, \frac{1}{6}, \frac{1}{6}, \frac{1}{6})$, $p(x_t|z_t = 1) = (\frac{1}{10}, \frac{1}{10}, \frac{1}{10}, \frac{1}{10}, \frac{1}{10}, \frac{1}{2})$

Possible realization:

```
hid:  11111111112222111111111111111111111112222222221222211111111111111111111
obs:  13555345265533663163515515262321121134622212632642654223446453232423 61
```



Forward algorithm

- The goal is to recursively compute the **filtered marginals**,

$$\alpha_t(j) = p(z_t = j | x_{1:t}).$$

- Assuming that we know the initial $p(z_1)$, transition $p(z_t|z_{t-1})$, and emission $p(x_t|z_t)$ probabilities for all $1 \leq t \leq T$.
- This is a step in the **forward-backward algorithm**.

Forward algorithm has two steps

- **Prediction step**: compute the one-step-ahead predictive density:

$$
\begin{aligned}
p(z_t = j | x_{1:(t-1)}) &= \sum_{i=1}^{K} p(z_{t-1} = i, z_t = j | x_{1:(t-1)}) \\
&= \sum_{i=1}^{K} p(z_t = j | z_{t-1} = i, x_{1:(t-1)}) p(z_{t-1} = i | x_{1:(t-1)}) \\
&= \sum_{i=1}^{K} p(z_t = j | z_{t-1} = i) p(z_{t-1} = i | x_{1:(t-1)}) = \sum_{i=1}^{K} A_{ij} \alpha_{t-1}(i) = (A^\top \alpha_{t-1})_j
\end{aligned}
$$

- **Update step:** Denoting $\lambda_t(j) = p(x_t | z_t = j)$ (here x_t is fixed and $\lambda_t \in \mathbb{R}^K$)

$$
\begin{aligned}
\alpha_t(j) &= p(z_t = j | x_{1:t}) = p(z_t = j | x_{1:(t-1)}, x_t) \propto p(z_t = j, x_t | x_{1:(t-1)}) \\
&= p(x_t | z_t = j, x_{1:(t-1)}) p(z_t = j | x_{1:(t-1)}) \\
&= p(x_t | z_t = j) p(z_t = j | x_{1:(t-1)}) = \lambda_t(j) p(z_t = j | x_{1:(t-1)})
\end{aligned}
$$

Using matrix notation: $\alpha_t \propto \lambda_t \odot (A^\top \alpha_{t-1})$ [$\odot$ is the Hadamard (entrywise) product]

Forward-Backward algorithm

This task of hidden state inference breaks down into the following:

- **Filtering**: compute posterior over current hidden state, $p(z_t|x_{1:t})$.
- **Prediction**: compute posterior over future hidden state, $p(z_{t+k}|x_{1:t})$.
- **Smoothing**: compute posterior over past hidden state, $p(z_t|x_{1:T}) \quad 1 \leq t < T$.

The probability of interest, $p(z_t|x_{1:T})$ is computed using a forward and backward recursion

- **Forward Recursion**: $\alpha_t(j) = p(z_t = j|x_{1:t})$ [this was computed earlier]
- **Backward Recursion**: $\beta_t(j) := p(x_{(t+1):T}|z_t = j)$

We assume that we know the initial $\pi(j) = p(z_1 = j)$, transition $A_{ij} = p(z_t = j|z_{t-1} = i)$, and emission $\lambda_t(j) = p(x_t|z_t = j)$ probabilities for all t.

Forward-Backward algorithm

We can break the chain into two parts, the past and the future, by conditioning on z_t:

- We have

$$\begin{aligned}\gamma_t := p(z_t|x_{1:T}) &\propto p(z_t, x_{1:T}) = p(z_t, x_{1:t}, x_{(t+1):T}) \\ &= p(z_t, x_{1:t})p(x_{(t+1):T}|z_t, x_{1:t}) = p(z_t, x_{1:t})p(x_{(t+1):T}|z_t) \\ &\propto p(z_t|x_{1:t})p(x_{(t+1):T}|z_t) \\ &= (\text{Forward Recursion})(\text{Backward Recursion})\end{aligned}$$

- Here we use the conditional independence $x_{(t+1):T} \perp x_{1:t}|z_t$.
- We know how to perform forward recursion from the previous part.

Backward recursion

In the backward pass,

$$\begin{aligned}
\beta_t(i) &= p(x_{(t+1):T}|z_t = i) = \sum_{j=1}^{K} p(z_{t+1} = j, x_{t+1}, x_{(t+2):T}|z_t = i) \\
&= \sum_j p(x_{(t+2):T}|z_{t+1} = j, z_t = i, x_{t+1}) p(x_{t+1}|z_{t+1} = j, z_t = i) p(z_{t+1} = j|z_t = i) \\
&= \sum_j p(x_{(t+2):T}|z_{t+1} = j) p(x_{t+1}|z_{t+1} = j) p(z_{t+1} = j|z_t = i) \\
&= \sum_j \beta_{t+1}(j) \lambda_{t+1}(j) A_{ij}
\end{aligned}$$

In vector notation $\beta_t = A(\lambda_{t+1} \odot \beta_{t+1})$, where $\beta_T(i) = 1$.

Forward-backward algorithm [$\gamma_t(j) = p(z_t = j|x_{1:T})$]

Once we have the forward and the backward steps complete, we can compute $\gamma_t \propto \alpha_t \odot \beta_t$.

Viterbi algorithm

- The Viterbi algorithm (Viterbi 1967) is used to compute the most probable sequence.

$$\hat{z} = \arg\max_{z_{1:T}} \; p(z_{1:T}|x_{1:T})$$

- Since this is MAP inference, we might think of replacing sum-operators with max-operators, just like we did in sum-product and max-product.
- Viterbi algorithm is a specialized version of max-product: the forward pass uses max-product, and the backward pass uses a traceback procedure to recover the most probable path.

Viterbi algorithm

- Define δ_t via

$$\delta_t(j) = \max_{z_1,..,z_{t-1}} p(z_{1:(t-1)}, z_t = j, x_{1:t})$$

  which is the probability of ending up in state j at time t, by taking the most probable path.

- We notice that

$$\begin{aligned}
\delta_t(j) &= \max_{z_1,..,z_{t-1}} p(z_{1:(t-1)}, z_t = j, x_{1:(t-1)}, x_t) \\
&= \max_{z_1,..,z_{t-1}} p(z_{1:(t-1)}, x_{1:(t-1)}) p(z_t = j | z_{t-1}) p(x_t | z_t = j) \\
&= \max_i \max_{z_1,..,z_{t-2}} p(z_{1:(t-2)}, z_{t-1} = i, x_{1:(t-1)}) p(z_t = j | z_{t-1} = i) p(x_t | z_t = j) \\
&= \max_i \delta_{t-1}(i) A_{ij} \lambda_t(j)
\end{aligned}$$

- Keep track of the most likely previous state: $\theta_t(j) = \arg\max_i \delta_{t-1}(i) A_{ij} \lambda_t(j)$.

Viterbi algorithm

- Initialize the algorithm with

$$\delta_1(j) \;=\; p(z_1 = j, x_1) \;=\; \pi_j \lambda_1(j).$$

- and terminate with

$$z_T^* = \arg\max_i \delta_T(i)$$

- Then, we compute the most probable sequence of states using traceback:

$$z_t^* = \theta_{t+1}(z_{t+1}^*)$$

Summary: HMMs

- HMMs hide the temporal dependence by keeping it in the unobserved state.
- No assumptions on the temporal dependence of observations is made.
- Forward-backward algorithm can be used to find "beliefs".
- Viterbi algorithm can be used to do MAP.

Variational Inference I

Overview: Second hour

- Variational Inference
- M-projection
- I-projection
- Naive mean-field approach

Posterior Inference for Latent Variable Models

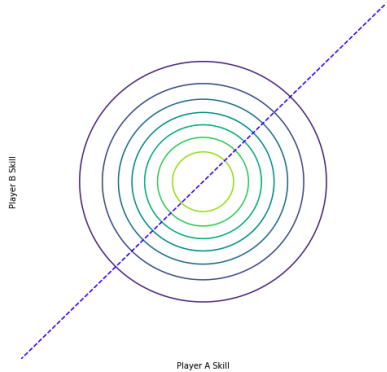
Examples of latent variable models: the generative image model and the TrueSkill model.

These models have a factorization $p(x, z) = p(z)p(x|z)$ where

- x are the observations or data,
- z are the unobserved (latent) variables.
- $p(z)$ is usually called the **prior**
- $p(x|z)$ is usually called the **likelihood**
- The conditional distribution of the unobserved variables given the observed variables (aka the **posterior**) is

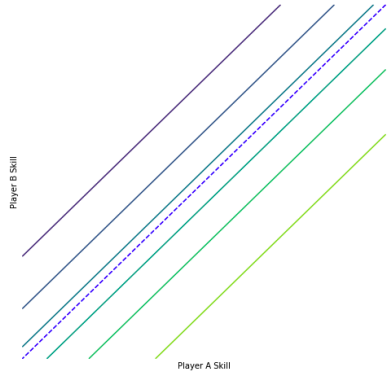
$$p(z|x) = \frac{p(x|z)p(z)}{p(x)} = \frac{p(x|z)p(z)}{\int p(x, z)dz}$$

Prior:



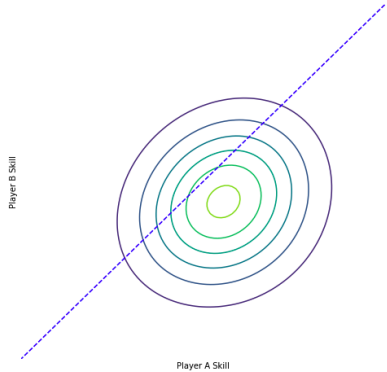
Says we are very uncertain about both players' skill.

Likelihood:



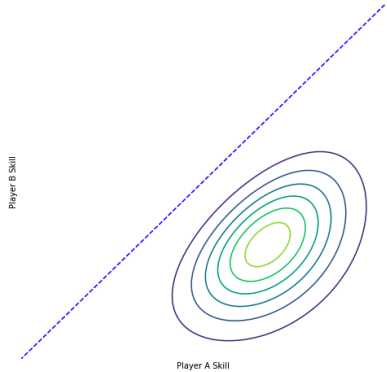
This is the part of the model that gives meaning to the latent variables.

Posterior:



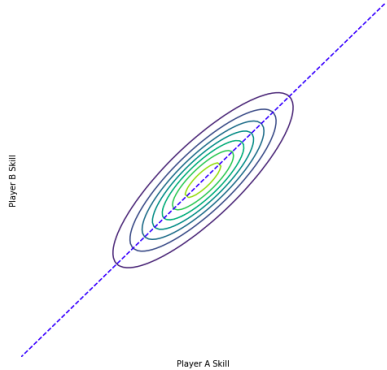
The posterior is not Gaussian anymore.

Posterior after A beats B 10 times:



Now the posterior is certain that A is better than B.

Posterior after both beat each other 10 times:



Now the posterior is certain that neither player is much better than the other, but is uncertain how good they both are in an absolute sense.

What is hard to compute about the posterior?

The integral $p(x) = \int p(x, z)dz$ is intractable when z is large.

These are operations that become expensive:

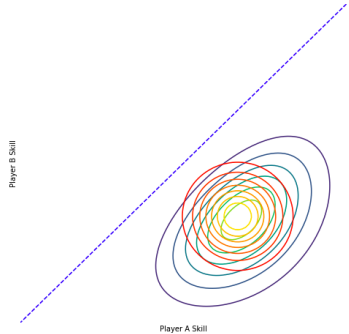
- Computing a posterior probability: $p(z|x) = \frac{p(z)p(x|z)}{p(x)}$
- Computing the evidence / marginal likelihood $p(x) = \int p(z, x)dz$
  - Useful for choosing between models, or fitting model parameters.
- Computing marginals of $p(z_1|x) = \int p(z_1, z_2, \ldots z_D|x)dz_2, dz_3, \ldots dz_D$
  - E.g. finding the posterior over a single tennis player's skill given all games.
- Sampling $z \sim p(z|x)$
  - e.g. summarizing which hypotheses are likely given the data, making predictions, and decisions.

Variational methods

Variational inference is closely related to the calculus of variations, developed in the 1700s by Euler, Lagrange. It is an approximate inference method where we seek a tractable (e.g., factorized) approximation to the target intractable distribution.

To be more formal, variational inference works as follows:

- Choose a tractable distribution $q(z) \in Q$ from a feasible set Q. This distribution will be used to approximate $p(z|x)$.
  - For example, $q(z) = \mathcal{N}(z|\mu, \Sigma)$. Try choose a Q that makes $q(z)$ a good approximation of the true posterior $p(z|x)$.
- Encode some notion of "difference" between $p(z|x)$ and q that can be effciently estimated. Usually we will use the KL divergence.
- Minimize this difference. Usually we will use an iterative optimization method.



- Whatever feasible set we choose for Q, it's usually not the case that there is any $q \in Q$ that exactly matches the true posterior.
- But computing the true posterior is intractable, so we have to take a shortcut somewhere.

How to measure closeness: KL divergence

We will measure the difference between q and p using the **Kullback-Leibler divergence**

$$\mathrm{KL}(q(z)\|p(z|x)) \;=\; \int q(z)\log\frac{q(z)}{p(z|x)}dz \;=\; \mathop{\mathbb{E}}_{z\sim q}\log\frac{q(z)}{p(z|x)}$$

Properties of the KL Divergence (see the tutorial):

- $\mathrm{KL}(q\|p) \geq 0$
- $\mathrm{KL}(q\|p) = 0 \Leftrightarrow q = p$
- $\mathrm{KL}(q\|p) \neq \mathrm{KL}(p\|q)$
- KL divergence is not a metric, since it is not symmetric.

Which direction of KL to use? $\mathrm{KL}(q\|p)$ **vs** $\mathrm{KL}(p\|q)$

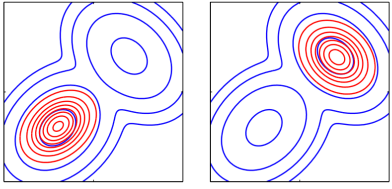
We could minimize $\mathrm{KL}(q\|p)$ or $\mathrm{KL}(p\|q)$. We will go with the tractable one.

Information (I-)Projection:

I-projection: $q^* = \arg\min_{q \in Q} \mathrm{KL}(q\|p) = \mathbb{E}_{x\sim q(x)} \log \frac{q(x)}{p(x)}$:

- $p \approx q \implies \mathrm{KL}(q\|p)$ small
- I-projection underestimates support, and does not yield the correct moments.
- $\mathrm{KL}(q\|p)$ penalizes q having mass where p has none (but not vice versa).

$p(x)$ is mixture of two 2D Gaussians and Q is the set of all 2D Gaussian distributions (with arbitrary covariance matrices)



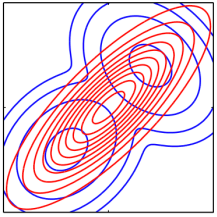
p=Blue, q^*=Red (two equivalently good solutions!)

Moment (M-)projection

M-projection: $q^* = \arg\min_{q \in Q} \mathrm{KL}(p\|q) = \mathbb{E}_{x\sim p(x)} \log \frac{p(x)}{q(x)}$:

- $p \approx q \implies \mathrm{KL}(p\|q)$ small
- $\mathrm{KL}(p\|q)$ penalizes q missing mass where p has some.
- M-projection yields a distribution $q(x)$ with the correct mean and covariance.

$p(x)$ is mixture of two 2D Gaussians and Q is the set of all 2D Gaussian distributions (with arbitrary covariance matrices)



p=Blue, q^*=Red

Maximum entropy interpretation

A related quantity is the **entropy** $H(p) = -\mathbb{E}_{x\sim p(x)} \log p(x)$ measuring uncertainty in p. Consider the optimization problem

$$\begin{aligned} &\text{maximize } H(p) \\ &\text{subject to } \mathbb{E}_{x\sim p(x)}[f_i(x)] = t_i \quad \text{for } i = 1, .., k. \end{aligned}$$

Theorem: Maximum entropy principle

Exponential family of distributions maximize the entropy $H(p)$ over all distributions satisfying: $\mathbb{E}_{x\sim p(x)}[f_i(x)] = t_i$ for $i = 1, .., k$.

- In M-projection, if Q is set of exponential families, then the expected sufficient statistics wrt $q^*(x)$ is the same as that wrt $p(x)$.
- M-projection require expectation wrt p, hence intractable.
- Most variational inference algorithms make use of the I-projection.

Mean-field approach

- Say we have an arbitrary MRF:

$$p(x|\theta) = \exp\Big\{ \sum_{C \in \mathcal{C}} \phi_C(x_C) - \log Z(\theta) \Big\}$$

- We find an approximate distribution $q(x) \in Q$ by performing I-projection to $p(x)$.

$$\mathrm{KL}(q\|p) \;=\; \mathbb{E}_{x\sim q(x)} \log \frac{q(x)}{p(x|\theta)} \;=\; \mathbb{E}_{x\sim q(x)} \Big[\log q(x) - \sum_{C \in \mathcal{C}} \phi_C(x_C) + \log Z(\theta) \Big]$$

  Thus $\arg\min_{q\in Q} \mathrm{KL}(q\|p) \;=\; \arg\max_{q\in Q} \{ \sum_{C\in\mathcal{C}} \mathbb{E}_q[\phi_C(x_C)] + H(q) \}$

- For tractability, we need a nice set Q. If $p \in Q$, then $q^* = p$. But this almost never happens.

Naive Mean-Field

- One way to proceed is the mean-field approach where we assume: $q(x) = \prod_{i \in V} q_i(x_i)$. (the set Q is composed of those distributions that factor out)
- Using this in the maximization problem, we can simplify things

$$q^* = \arg\max_{q \in Q} \sum_{C \in \mathcal{C}} \sum_{x_C} q_C(x_C)\phi_C(x_C) + H(q)$$

We notice $q_C(x_C) = \prod_{i \in C} q_i(x_i)$ and also $H(q)$ decomposes

$$\begin{aligned} H(q) &= \mathbb{E}_q[-\log q(x)] = -\sum_x q(x) \log q(x) = -\sum_x q(x) \Big[\sum_i \log q_i(x_i)\Big] \\ &= -\sum_i \sum_x \Big[q_i(x_i) \log q_i(x_i)\Big] \frac{q(x)}{q_i(x_i)} = -\sum_i \sum_{x_i} \Big[q_i(x_i) \log q_i(x_i)\Big] \sum_{x_{\setminus i}} \frac{q(x)}{q_i(x_i)} \\ &= -\sum_i \sum_{x_i} \Big[q_i(x_i) \log q_i(x_i)\Big] = \sum_i H(q_i) \end{aligned}$$

We can thus write everything as a function of q_i for $i \in V$.

Example: Pairwise MRF

- Thus the final optimization problem reduces to

$$q^* = \arg\max_q \sum_{C \in \mathcal{C}} \sum_{x_C} \phi_C(x_C) \prod_{i \in C} q_i(x_i) + \sum_{i \in V} H(q_i)$$
$$\text{subject to} \quad q_i(x_i) \geq 0 \text{ and } \sum_{x_i} q_i(x_i) = 1.$$

- Further simplify the setting and assume that we have a pairwise MRF. Then the optimization problem becomes

$$q^* = \arg\max_q \sum_{(i,j) \in \mathcal{E}} \sum_{x_i, x_j} \phi_{ij}(x_i, x_j) q_i(x_i) q_j(x_j) - \sum_i \sum_{x_i} q_i(x_i) \log(q_i(x_i))$$
$$\text{subject to: } q_i(x_i) \geq 0 \text{ and } \sum_{x_i} q_i(x_i) = 1.$$

Coordinate maximization

This problem is hard as it has many local maxima! But we can still try to optimize using block coordinate ascent.

- Initialize $\{q_i(x_i)\}_{i \in V}$ uniformly
- Iterate over $i \in V$
  - Greedily maximize the objective over $q_i(x_i)$
  - This is equivalent to: $q_i(x_i) \propto \exp\left\{ \sum_{j \in N(i)} \sum_{x_j} q_j(x_j) \phi_{ij}(x_i, x_j) \right\}$
    - which follows from: write the Lagrangian, take the derivative, set to zero, and solve
  - Repeat until convergence.

This is guaranteed to converge but can converge to local optima.

Summary

- Approximate the complex (intractable) distribution with a simpler (tractable) distribution
- I-projection & M-projection measure the distance to true posterior
- Mean field approximation is a way to simplify the set of distributions
- More variational inference after midterm (which is in 2 weeks).