NUS National University of Singapore | School of Computing

CS5340
Uncertainty Modeling in AI

Lecture 10:
Variational Inference
(or "how to do approximate inference with optimization")

Asst. Harold Soh

AY 2022/23

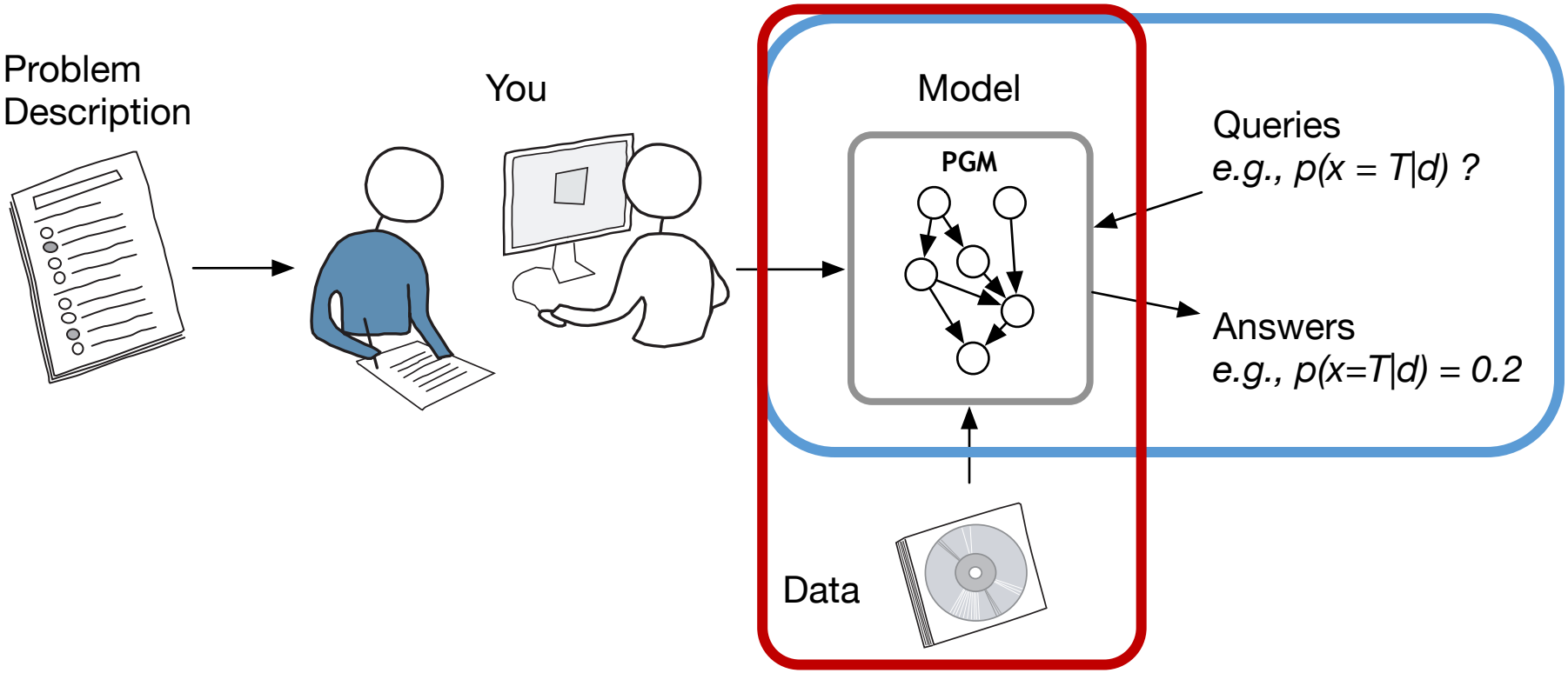
Semester 2

Course Schedule

Week	Date	Lecture Topic	Tutorial Topic
1	12 Jan	Introduction to Uncertainty Modeling + Probability Basics	Introduction
2	19 Jan	Simple Probabilistic Models	Probability Basics
3	26 Jan	Bayesian networks (Directed graphical models)	More Basic Probability
4	2 Feb	Markov random Fields (Undirected graphical models)	DGM modelling and d-separation
5	9 Feb	Variable elimination and belief propagation	MRF + Sum/Max Product
6	16 Feb	Factor graph and the junction tree algorithm	**Quiz 1**
-	-	RECESS WEEK	
7	2 Mar	Mixture Models and Expectation Maximization (EM)	Linear Gaussian Models
8	9 Mar	Hidden Markov Models (HMM)	Probabilistic PCA
9	**16** Mar	Monte-Carlo Inference (Sampling)	Linear Gaussian Dynamical System
10	23 Mar	Variational Inference	MCMC + Sequential VAE
11	30 Mar	Inference and Decision-Making (Special Topic)	**Quiz 2**
12	6 Apr	Gaussian Processes (Special Topic)	Wellness Day
13	13 Apr	**Project Presentations**	Closing

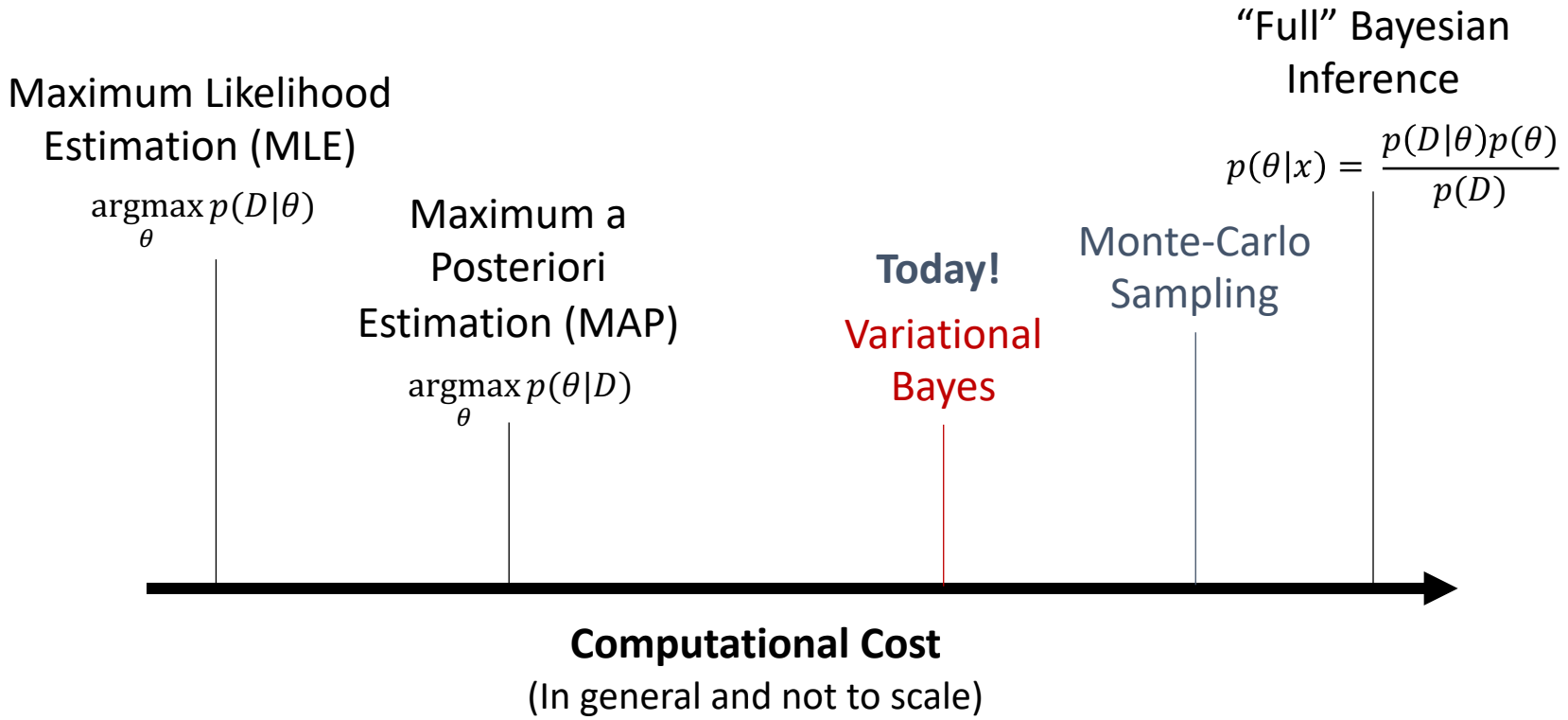
CS5340 in a nutshell

CS5340 is about how to "**represent**" and "**reason**" with uncertainty in a computer.



From Lecture 2: Learning Parameters

- Common approaches to learn the unknown parameters θ from a set of given data $\mathcal{D} = \{x[1], \ldots, x[N]\}$:



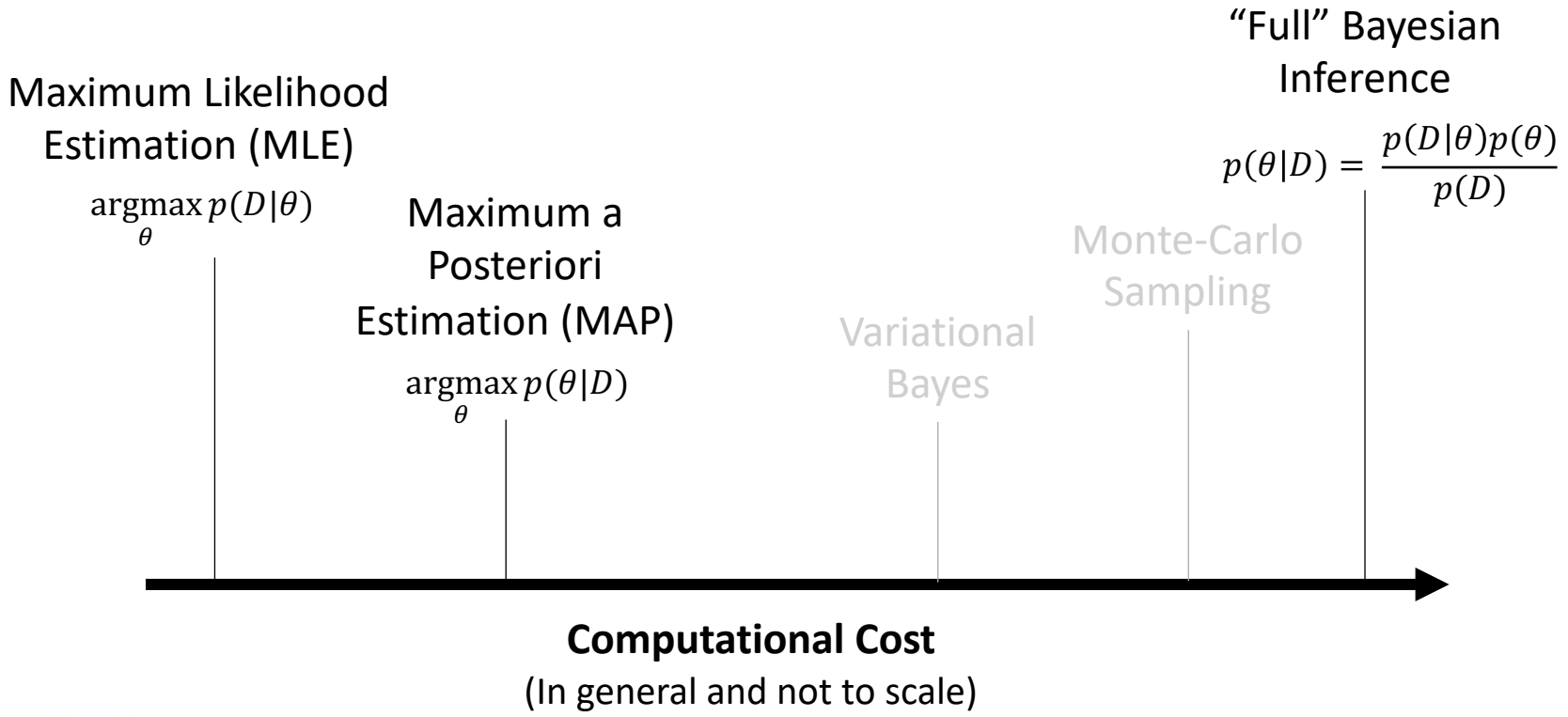
NUS National University of Singapore | School of Computing

Quick Recap and Motivation

EM, MCMC, Variational Inference

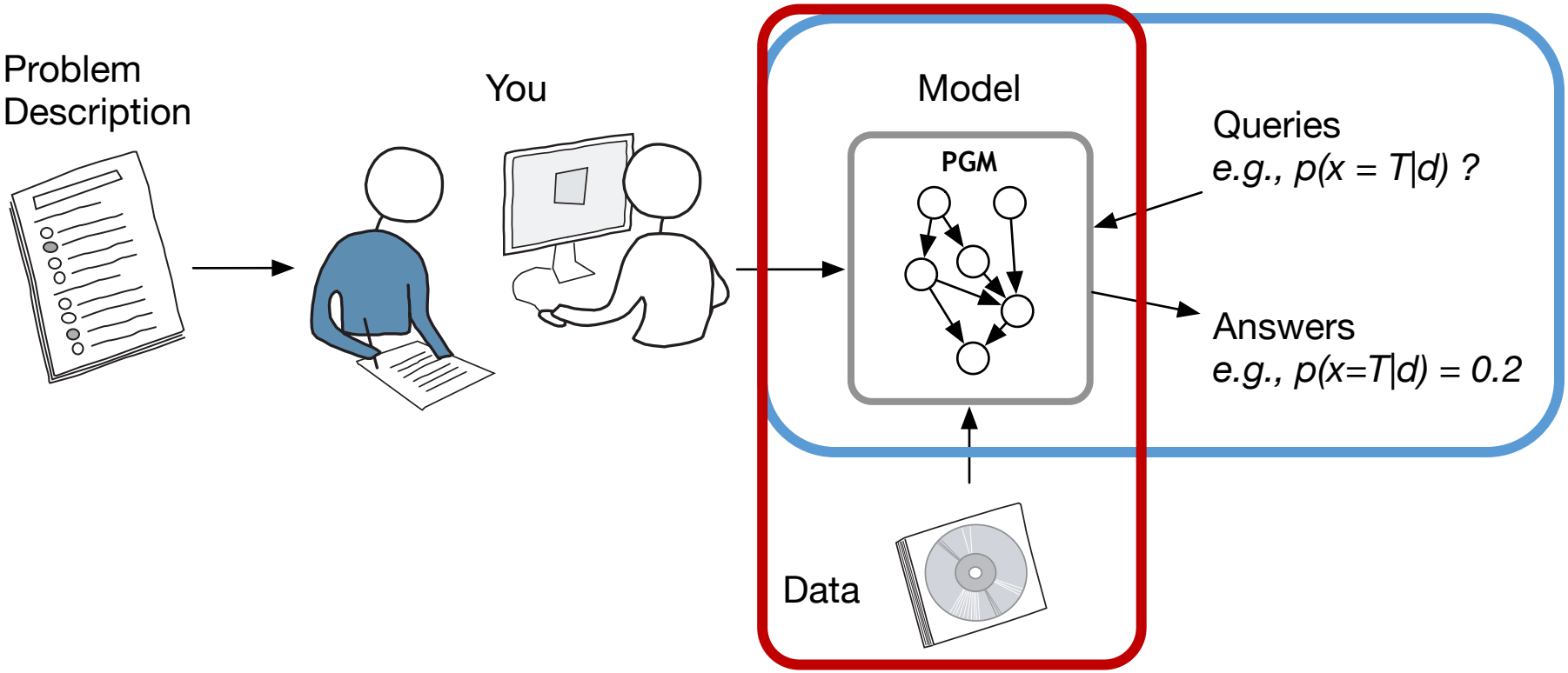
From Lecture 2: Learning Parameters

- Common approaches to learn the unknown parameters θ from a set of given data $\mathcal{D} = \{x[1], \ldots, x[N]\}$:



CS5340 in a nutshell

CS5340 is about how to "**represent**" and "**reason**" with uncertainty in a computer.



The General EM Algorithm

1. Choose an initial setting for the parameters θ^{old}.
2. Expectation step: Evaluate $p(\mathrm{Z}|\mathrm{X}, \theta^{old})$.
3. Maximization step: Evaluate θ^{new} given by:

$$\boldsymbol{\theta}^{\mathrm{new}} = \arg\max_{\boldsymbol{\theta}} \mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{\mathrm{old}})$$

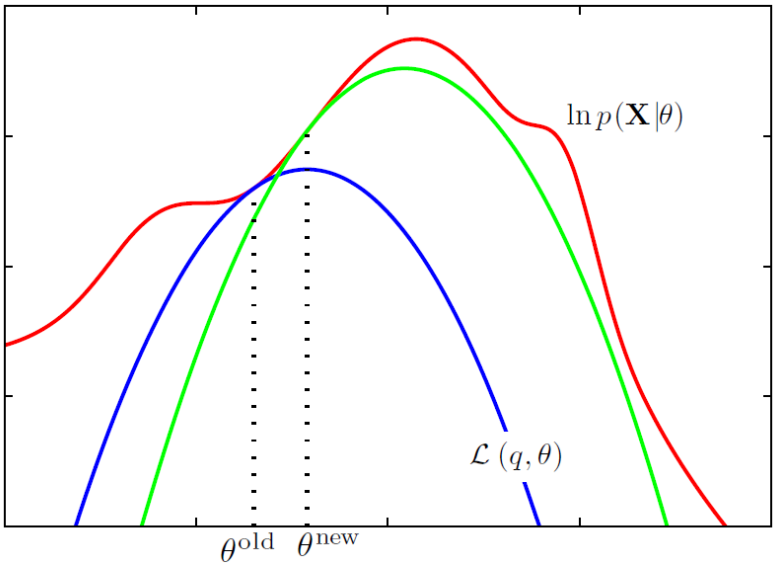
where

$$\mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{\mathrm{old}}) = \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}^{\mathrm{old}}) \ln p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})$$

4. Check for convergence of either the log likelihood or the parameter values, if not converged:

$$\boldsymbol{\theta}^{\mathrm{old}} \leftarrow \boldsymbol{\theta}^{\mathrm{new}}$$

The Theory Behind EM Algorithm



- **E-step**: we compute the convex lower bound given the old parameters θ^{old} (blue curve).
- **M-step**: we maximize this lower bound to get new parameters θ^{new}.
- This is repeated (green curve) until convergence.

Image source: "Pattern recognition and machine learning", Christopher Bishop

Sampling and the EM Algorithm

- Sampling methods can be used to approximate the E step of the EM algorithm for models in which the E step cannot be performed analytically.

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}) = \int p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}^{\text{old}}) \ln p(\mathbf{Z}, \mathbf{X}|\boldsymbol{\theta}) \, \mathrm{d}\mathbf{Z}$$

Cannot be computed analytically!

Sampling and the EM Algorithm

- Approximate integral by a finite sum over samples $\{Z^l\}$, which are drawn from $p(Z \mid X, \theta^{old})$

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}) \simeq \frac{1}{L} \sum_{l=1}^{L} \ln p(\mathbf{Z}^{(l)}, \mathbf{X} | \boldsymbol{\theta})$$

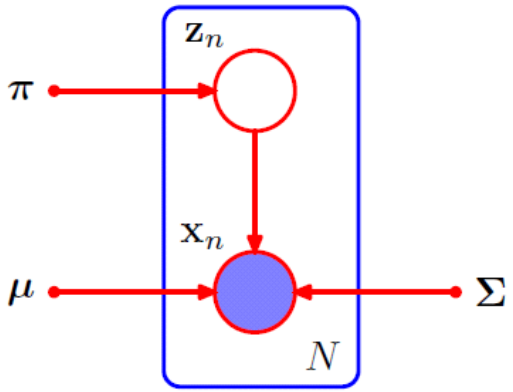
- The *Q* function is optimized in the usual way in the M step.

- Called the Monte Carlo EM algorithm.

Learning & Inference

- What is the difference?

Example: GMM

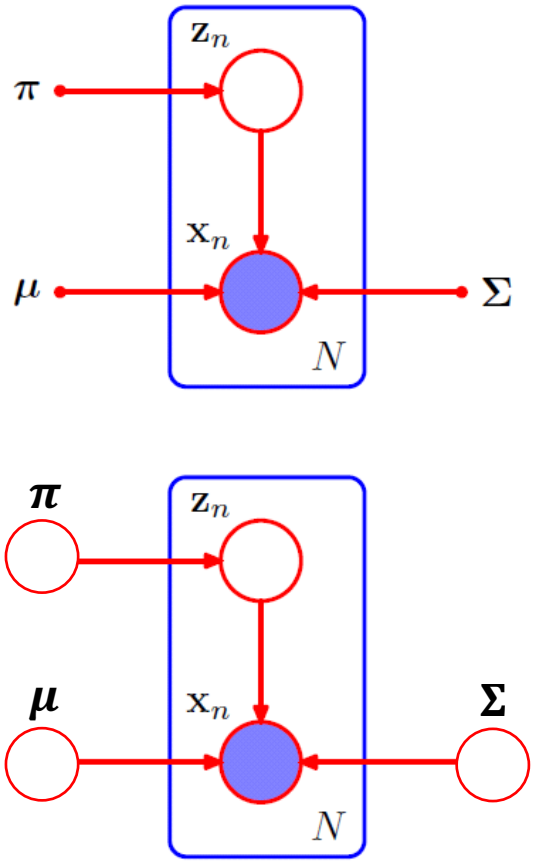


Learning & Inference

- What is the difference?

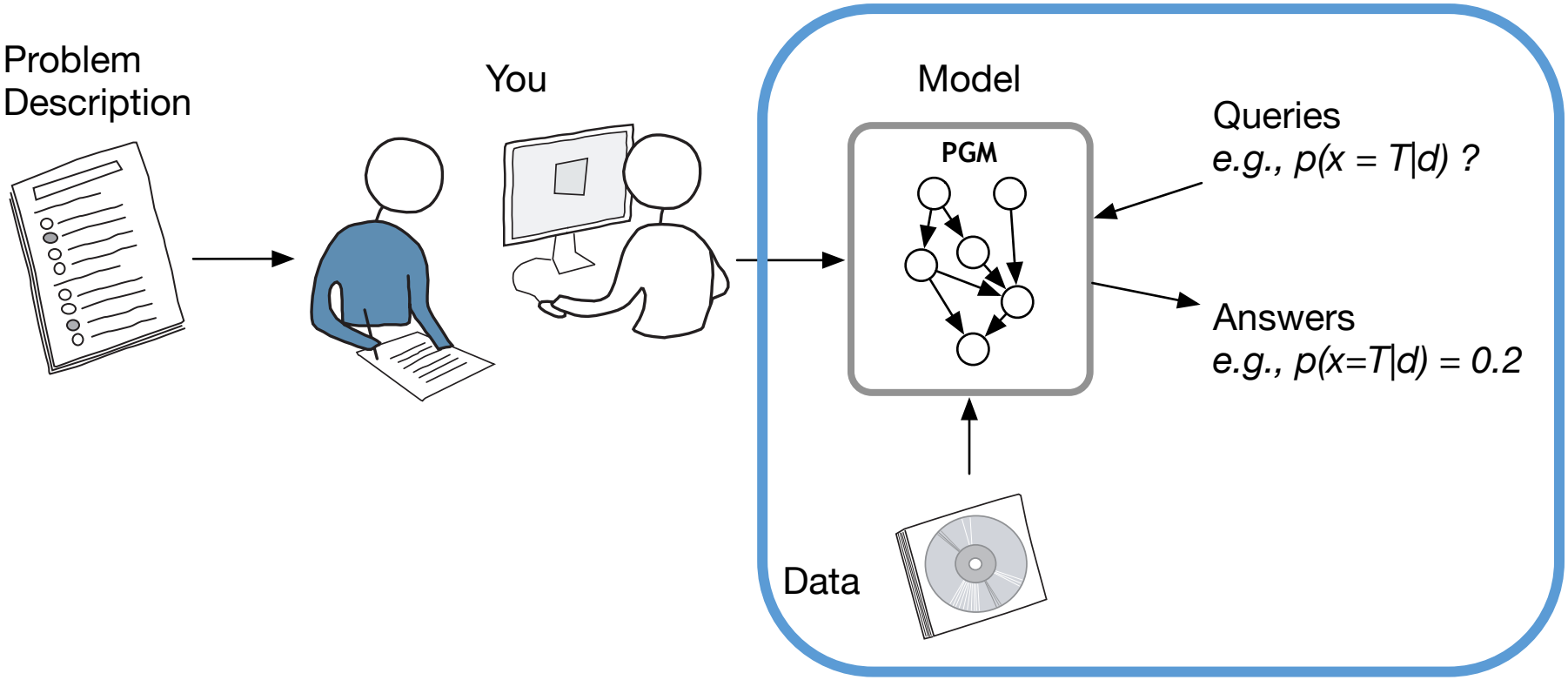
- If Bayesian, Learning is a form of Inference
  - Do not learn point parameters
    - Except: Empirical Bayes (wrong in theory, done in practice)
  - Put prior distribution and perform inference.

Example: GMM



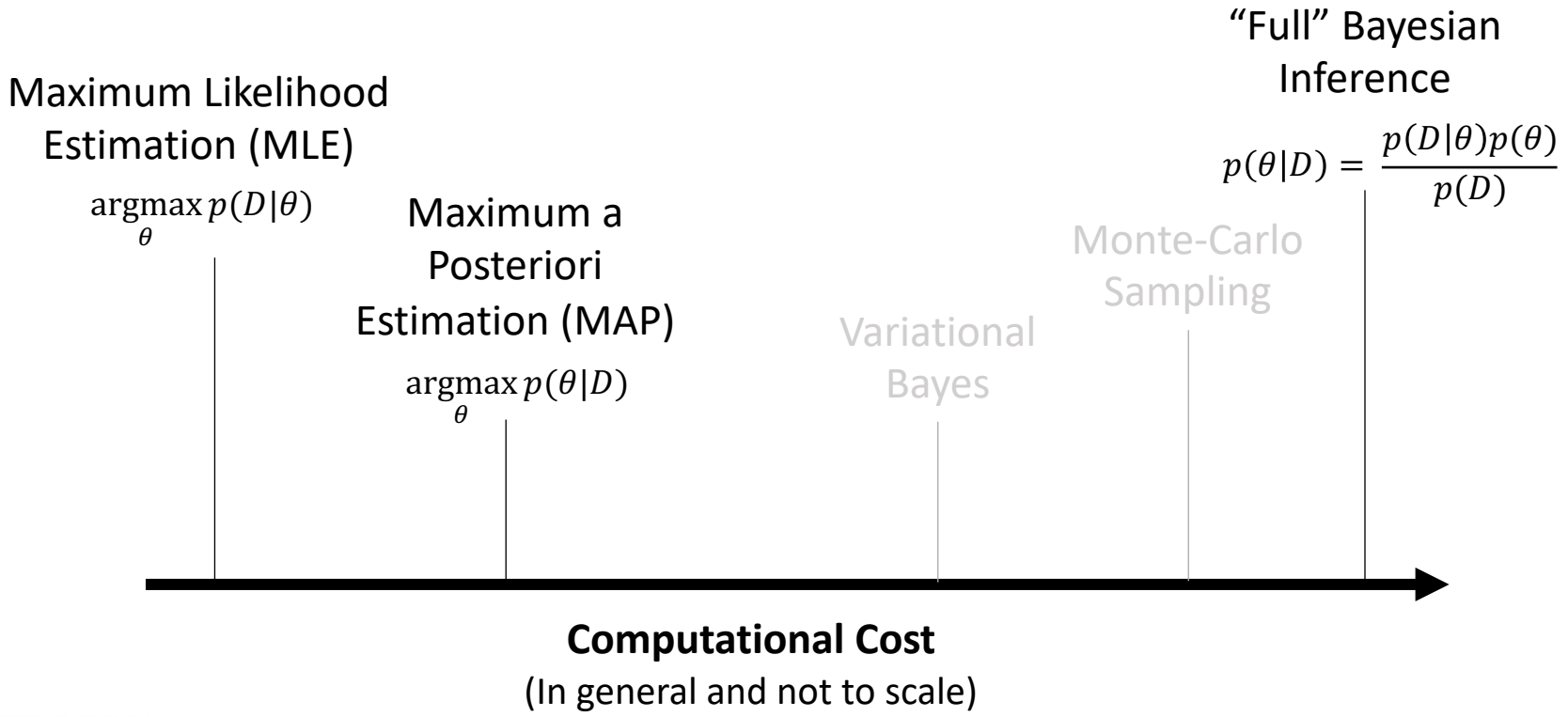
CS5340 in a nutshell

CS5340 is about how to "**represent**" and "**reason**" with uncertainty in a computer.



From Lecture 2: Learning Parameters

- Common approaches to learn the unknown parameters θ from a set of given data $\mathcal{D} = \{x[1], \ldots, x[N]\}$:



Inference

- Common tasks include evaluation of:
  - posterior distribution $p(z|x)$.
  - expectations computed with respect $p(z|x)$.

- Z are the latent variables (including the unknown parameters θ) and X is the observed variables.

$$p(Z|X) = \frac{p(X|\,Z)p(Z)}{p(X)}$$

Approximate Inference

- Could be infeasible to evaluate the posterior distribution or to compute expectations:

1. The dimensionality is too high in the latent space to work with directly.

2. The posterior distribution has a highly complex form and expectations are not analytically tractable.

Approximate Inference

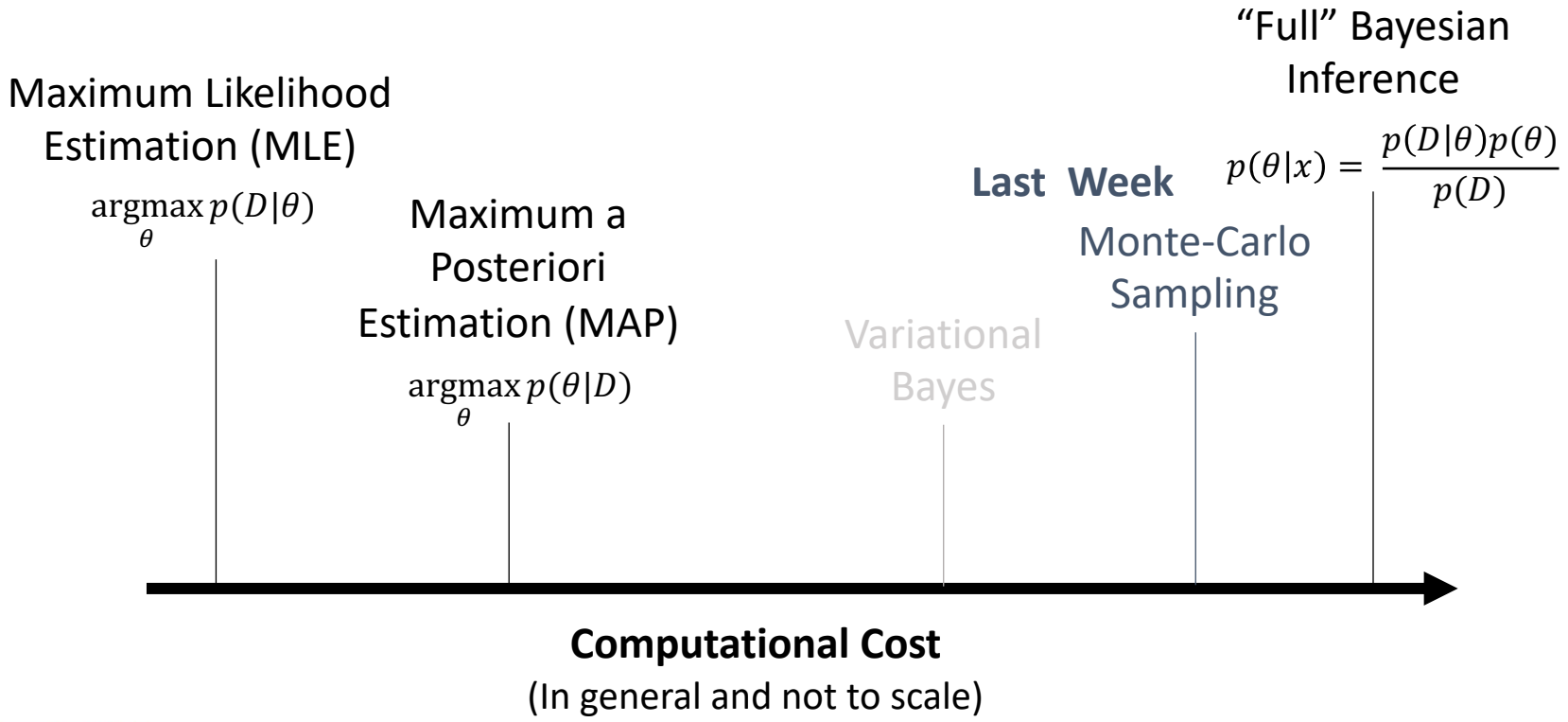
- In such situations, we resort to approximation schemes:

1. Stochastic approximation: Markov Chain Monte-Carlo (MCMC).

2. Deterministic approximation: Variational approach.

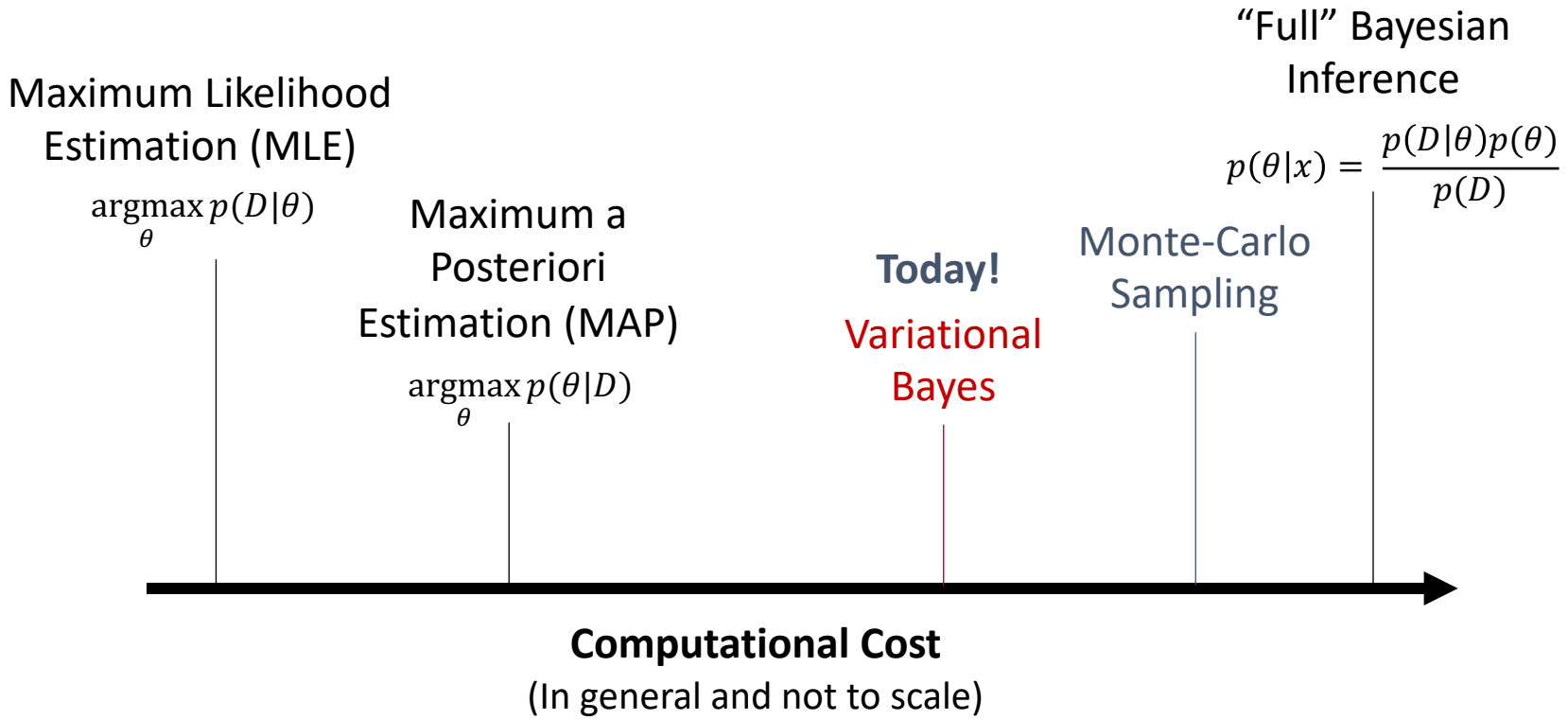
From Lecture 2: Learning Parameters

- Common approaches to learn the unknown parameters θ from a set of given data $\mathcal{D} = \{x[1], \ldots, x[N]\}$:



From Lecture 2: Learning Parameters

- Common approaches to learn the unknown parameters θ from a set of given data $\mathcal{D} = \{x[1], \ldots, x[N]\}$:



Acknowledgements

- A lot of slides and content of this lecture are adopted from:

1. Christopher Bishop, “Pattern Recognition and Machine Learning”, Chapter 10.
2. Kevin Murphy, "Machine learning: a probabilistic approach“, Chapter 21 and 22.
3. Carl Doersch, “Tutorial on Variational Autoencoders”
4. David Barber, "Bayesian reasoning and machine learning“, Chapter 28.
5. Daphne Koller and NirFriedman, "Probabilistic graphical models“, Chapter 11.
6. Gim Hee’s slides

Learning Outcomes

- Students should be able to:

1. Explain the concept of variational approach using Lower-Bound of maximum likelihood and KL-divergence.
2. Use variational approach to perform inference on graphical models containing hidden variables.
3. Explain the variational autoencoder (VAE).

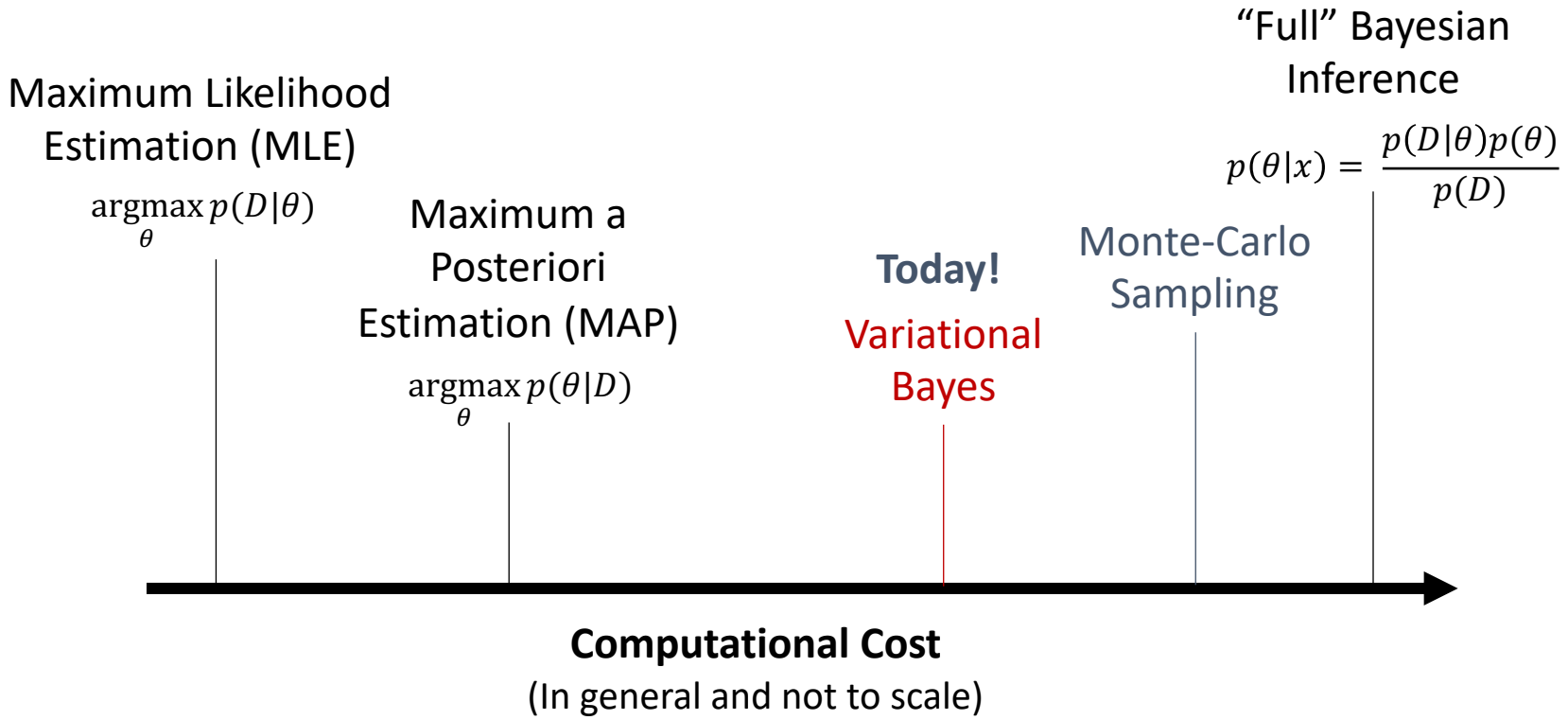
NUS National University of Singapore | School of Computing

Variational Inference

Intuition

From Lecture 2: Learning Parameters

- Common approaches to learn the unknown parameters θ from a set of given data $\mathcal{D} = \{x[1], \ldots, x[N]\}$:



Key Ideas

- **Want:** Obtain a distribution p
  - E.g., a posterior $p(\theta|D) = p(D|\theta)p(\theta)/p(D)$
- **Problem:** Intractable to compute

- **Variational Approach:**
  - Optimize a distribution q_ϕ to approximate p
    - ϕhere is the parameters of q
  - Minimize the KL Divergence
    - $\mathbb{D}_{\mathrm{KL}}[q \parallel p]$ or $\mathrm{KL}[q \parallel p]$
  - Transforms Inference to Optimization

Variational Approach

- Given $p(x, z)$, find an approximation $q(z)$ for the posterior distribution $p(z|x)$.
- **Key idea**: Choose the approximation $q(z)$ that minimizes the KL-divergence

$$\mathrm{KL}(q\|p) = -\int q(\mathbf{Z}) \ln \left\{ \frac{p(\mathbf{Z}|\mathbf{X})}{q(\mathbf{Z})} \right\} \mathrm{d}\mathbf{Z} \geq 0.$$

i.e. $q(z)$ that close to $p(z|x)$.

Variational Approach

$$\mathrm{KL}(q\|p) = -\int q(\mathbf{Z}) \ln \left\{ \frac{p(\mathbf{Z}|\mathbf{X})}{q(\mathbf{Z})} \right\} \mathrm{d}\mathbf{Z} \geq 0$$

- But: minimizing the KL-divergence is hard since $p(z|x)$ is intractable.
- **Solution:** Maximize the evidence lower-bound (ELBO) of log-likelihood

$$\mathcal{L}(q) = \int q(\mathbf{Z}) \ln \left\{ \frac{p(\mathbf{X}, \mathbf{Z})}{q(\mathbf{Z})} \right\} \mathrm{d}\mathbf{Z}$$

Recall: Lecture 7 on EM

Expectation Maximization (General Derivation)

Thursday, March 5, 2020 1:16 PM

$$\lg p(x|\theta) = \lg \sum_z p(x,z|\theta)$$

lg is on the outside!

$$= \sum_z q(z) \lg p(x|\theta)$$

$$= \sum_z q(z) \lg \frac{p(x|\theta)}{q(z)} \cdot \frac{q(z)}{p(z|x,\theta)} p(z|x,\theta)$$

$$= \sum_z q(z) \lg \left[\frac{p(x|\theta) p(z|x,\theta)}{q(z)} \frac{q(z)}{p(z|x,\theta)} \right]$$

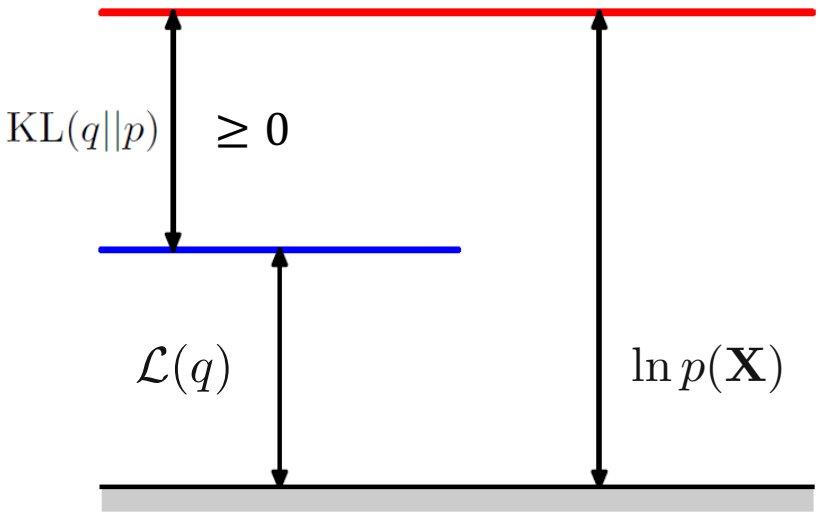
$$\lg p(x|\theta) = \sum_z q(z) \lg \frac{p(x,z|\theta)}{q(z)} + \sum_z q(z) \lg \frac{q(z)}{p(z|x,\theta)}$$

$\mathcal{L}(q,\theta)$ "lower bound"

$\mathbb{D}_{KL}[q \| p]$ (≥ 0) $(=0$ iff $p=q)$

Variational Approach

- Decompose the log marginal probability:



$$\ln p(\mathbf{X}) = \underbrace{\mathcal{L}(q)}_{\text{Lower bound}} + \underbrace{\mathrm{KL}(q\|p)}_{\text{KL-divergence}}$$

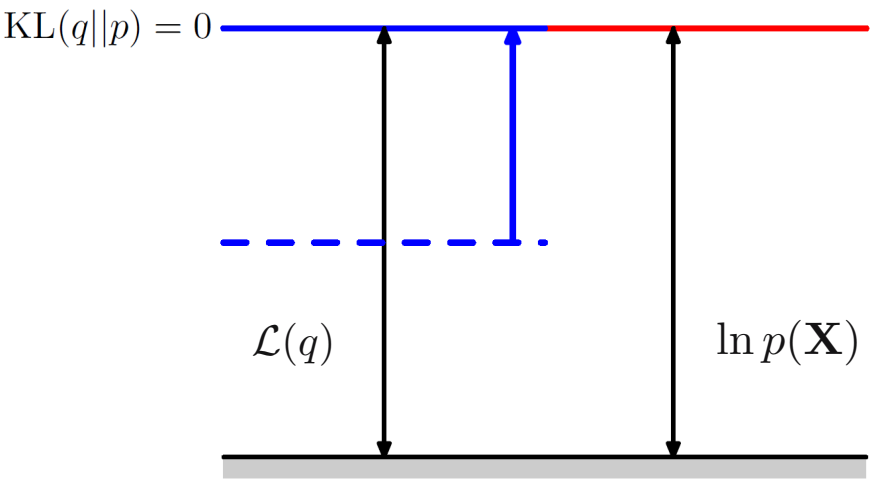
where

$$\mathcal{L}(q) = \int q(\mathbf{Z}) \ln \left\{ \frac{p(\mathbf{X}, \mathbf{Z})}{q(\mathbf{Z})} \right\} \mathrm{d}\mathbf{Z}$$

$$\mathrm{KL}(q\|p) = -\int q(\mathbf{Z}) \ln \left\{ \frac{p(\mathbf{Z}|\mathbf{X})}{q(\mathbf{Z})} \right\} \mathrm{d}\mathbf{Z}.$$

Image Source: "Pattern Recognition and Machine Learning", Christopher Bishop

Variational Approach



$$\ln p(\mathbf{X}) = \mathcal{L}(q) + \mathrm{KL}(q\|p)$$

where

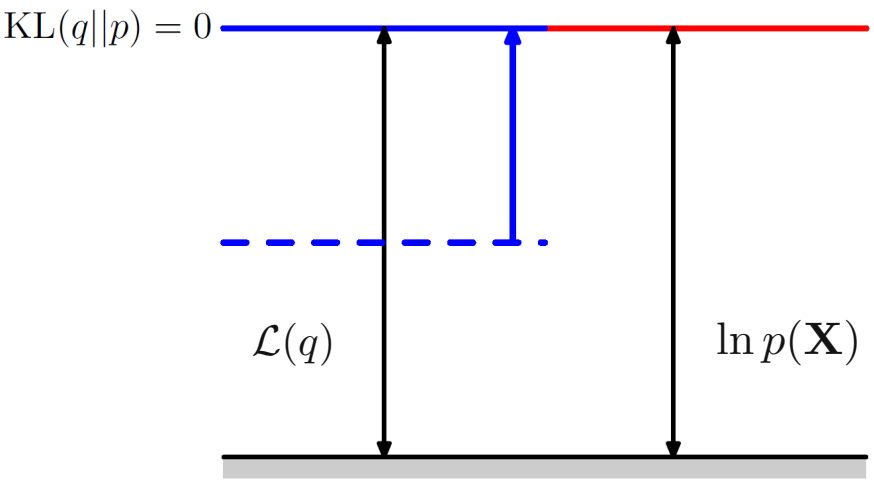
$$\mathcal{L}(q) = \int q(\mathbf{Z}) \ln \left\{ \frac{p(\mathbf{X}, \mathbf{Z})}{q(\mathbf{Z})} \right\} \mathrm{d}\mathbf{Z}$$

$$\mathrm{KL}(q\|p) = -\int q(\mathbf{Z}) \ln \left\{ \frac{p(\mathbf{Z}|\mathbf{X})}{q(\mathbf{Z})} \right\} \mathrm{d}\mathbf{Z}.$$

- Maximizing the lower bound $\mathcal{L}(q)$ with respect to the distribution $q(Z)$ is equivalent to minimizing the KL divergence.

Image Source: "Pattern Recognition and Machine Learning", Christopher Bishop

Variational Approach



$$\ln p(\mathbf{X}) = \mathcal{L}(q) + \mathrm{KL}(q\|p)$$

where

$$\mathcal{L}(q) = \int q(\mathbf{Z}) \ln \left\{ \frac{p(\mathbf{X}, \mathbf{Z})}{q(\mathbf{Z})} \right\} \mathrm{d}\mathbf{Z}$$

$$\mathrm{KL}(q\|p) = -\int q(\mathbf{Z}) \ln \left\{ \frac{p(\mathbf{Z}|\mathbf{X})}{q(\mathbf{Z})} \right\} \mathrm{d}\mathbf{Z}.$$

Image Source: "Pattern Recognition and Machine Learning", Christopher Bishop

Summary: Variational Approach

- Given $p(x, z)$, find an approximation $q(z)$ for the posterior distribution $p(z|x)$.
- **Key idea**: Choose the approximation $q(z)$ that minimizes the KL-divergence

$$\mathrm{KL}(q\|p) = -\int q(\mathbf{Z}) \ln \left\{ \frac{p(\mathbf{Z}|\mathbf{X})}{q(\mathbf{Z})} \right\} \mathrm{d}\mathbf{Z} \geq 0.$$

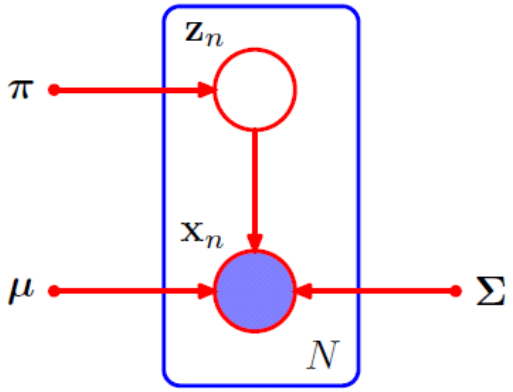
  i.e. $q(z)$ that close to $p(z|x)$.

- **Maximize** the ELBO $\mathcal{L}(q)$ wrt $q(Z)$ (equivalent to minimizing the KL divergence).

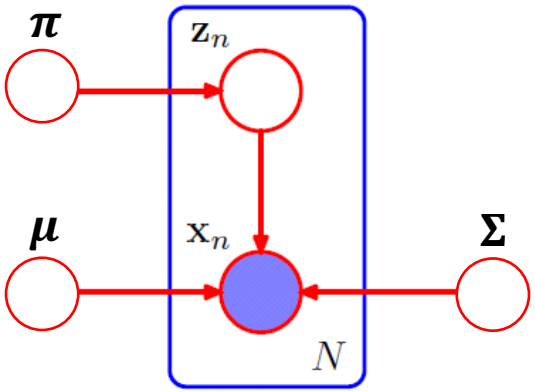
$$\mathcal{L}(q) = \int q(\mathbf{Z}) \ln \left\{ \frac{p(\mathbf{X}, \mathbf{Z})}{q(\mathbf{Z})} \right\} \mathrm{d}\mathbf{Z}$$

Difference from EM

- EM computes point estimates for the parameters

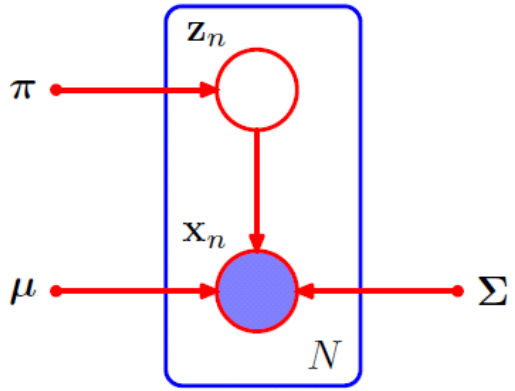


- Variational Bayes estimates the posterior distributions of latent variables (of which some are parameters).

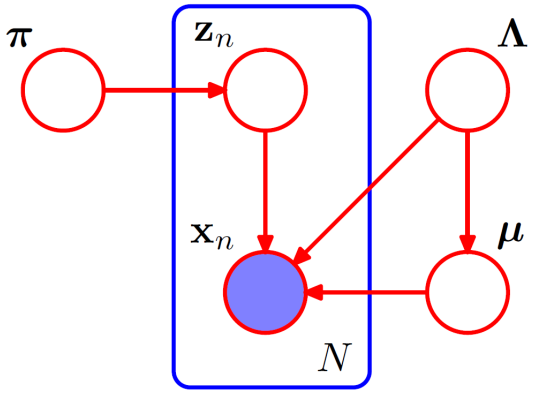


Difference from EM

- EM computes point estimates for the parameters

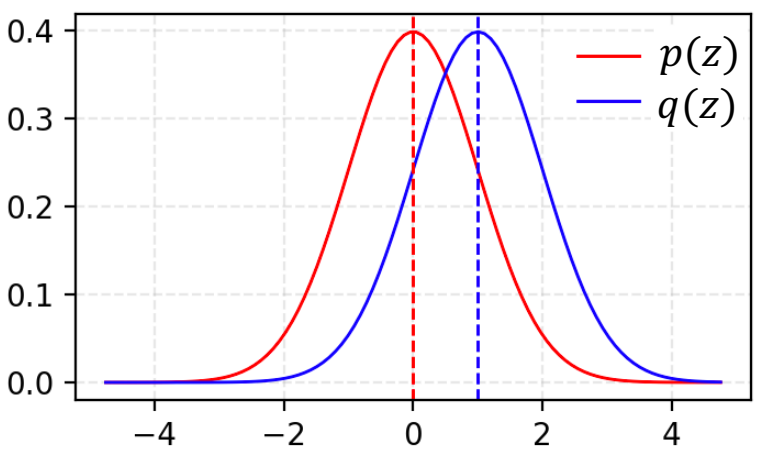


- Variational Bayes estimates the posterior distributions of latent variables (of which some are parameters).



Forward vs Backward KL-Divergence

- KL-divergence is not symmetric, minimizing $KL(q \parallel p)$ and $KL(p \parallel q)$ can give different results.



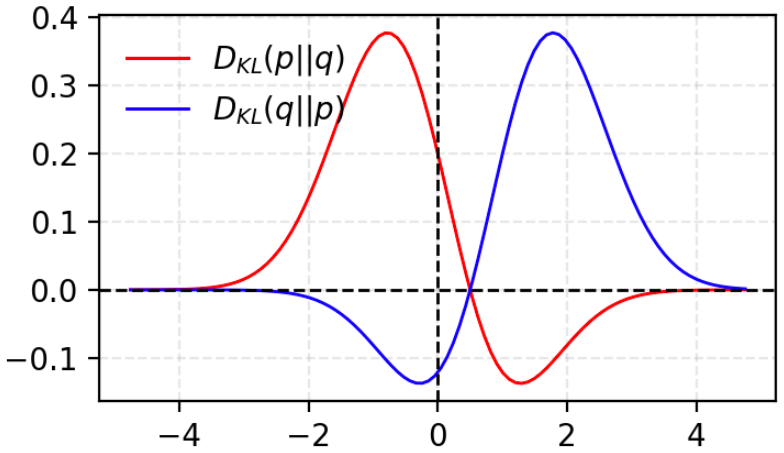


Image source: https://lilianweng.github.io/lil-log/2017/08/20/from-GAN-to-WGAN.html

Forward vs Backward KL-Divergence

- Reverse KL, $KL(q \parallel p)$, also known as an **I-projection** or **information projection**:

$$KL(q \parallel p) = \sum_{\mathrm{z}} q(\mathrm{z}) \ln \frac{q(\mathrm{z})}{p(\mathrm{z})}$$

- Infinite if $p(z) = 0$ and $q(z) > 0$
  - if $p(z) = 0$, must ensure $q(z) = 0$.
- Reverse KL is zero forcing for q
  - q will typically under-estimate the support of p.

Forward vs Backward KL-Divergence

- Forward KL, $KL(p \parallel q)$ also known as an **M-projection** or **moment projection**:

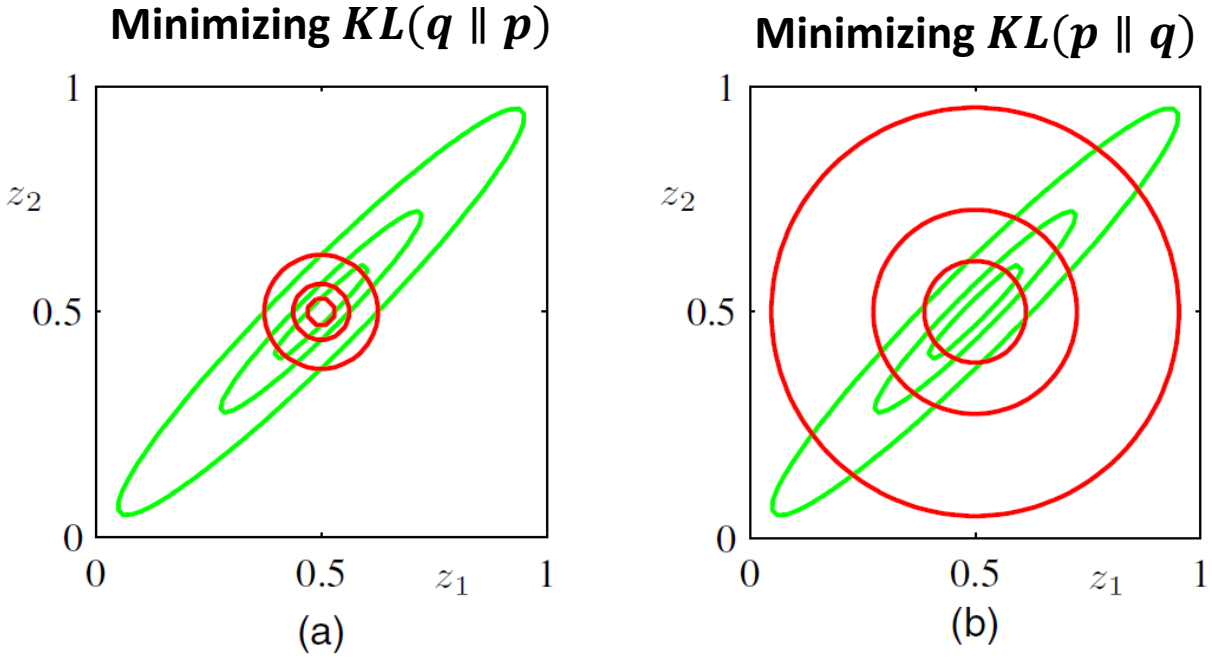
$$KL(p \parallel q) = \sum_{\mathrm{z}} p(\mathrm{z}) \ln \frac{p(\mathrm{z})}{q(\mathrm{z})}$$

- This is infinite if $q(x) = 0$ and $p(z) > 0$
  - if $p(z) > 0$, must ensure $q(z) > 0$.
- Forward KL is zero avoiding for q
  - q will typically over-estimate the support of p.

Forward vs Backward KL-Divergence

Red contours: approximating distribution $q(z)$

Green contours: Gaussian distribution $p(z)$



Under-estimation

Over-estimation

Image Source: "Pattern Recognition and Machine Learning", Christopher Bishop

Forward vs Backward KL-Divergence

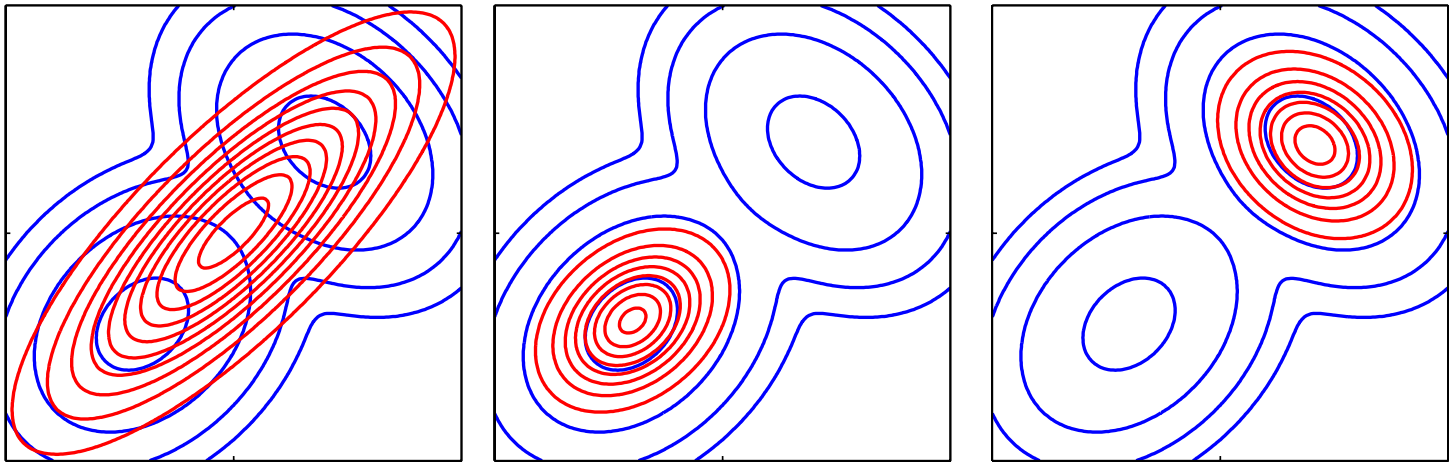
Blue contours: Bimodal distribution $p(Z)$
Red contours: $q(Z)$ that best approximates $p(Z)$

Minimizing $KL(p \parallel q)$

Minimizing $KL(q \parallel p)$

Local maxima 1

Local maxima 2



(a) (b) (c)

$q(Z)$ is nonzero in regions where $p(Z)$ is nonzero, i.e. zero avoiding

$q(Z)$ is small when $p(Z)$ is small, i.e. zero forcing

Image Source: "Pattern Recognition and Machine Learning", Christopher Bishop

Forward or Backward KL-Divergence?

- We use Backward KL-divergence $KL(q \parallel p)$ because it leads to a tractable lower-bound:

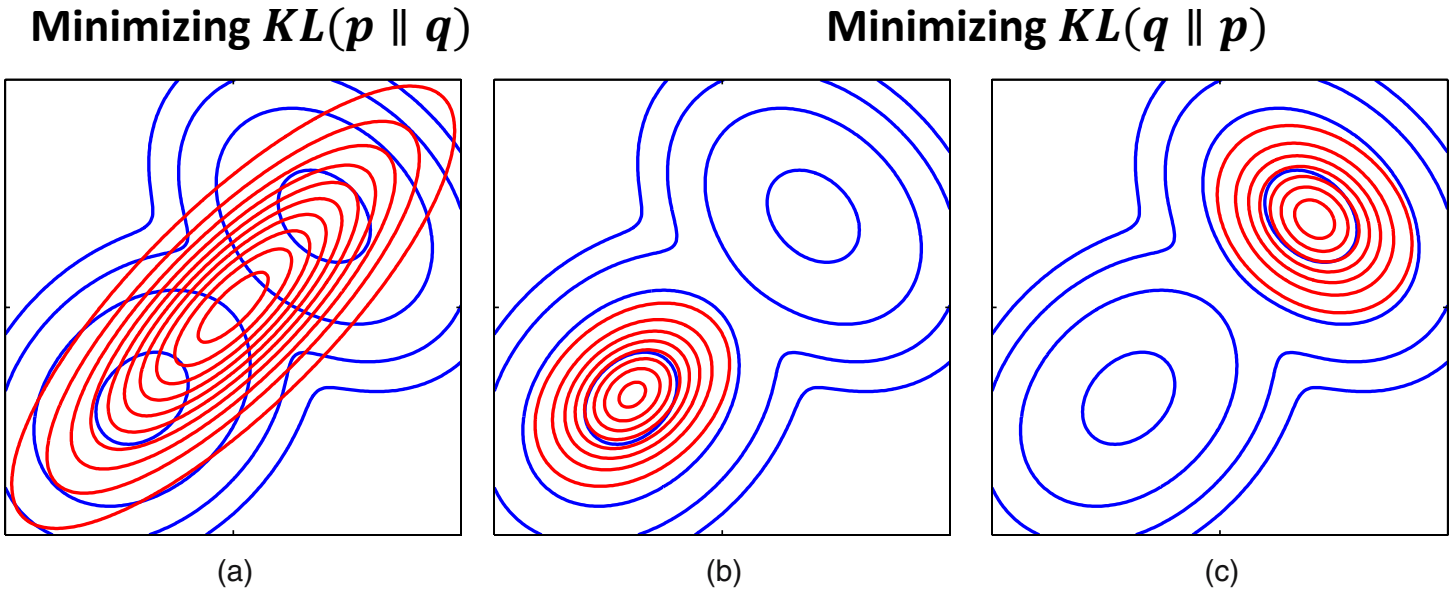
$$\mathcal{L}(q) = \int q(\mathbf{Z}) \ln \left\{ \frac{p(\mathbf{X}, \mathbf{Z})}{q(\mathbf{Z})} \right\} \mathrm{d}\mathbf{Z}$$

- Compared to forward KL-divergence $KL(p \parallel q)$ which leads to a intractable lower-bound:

$$\mathcal{L}(p) = \int p(\mathbf{Z}|\mathbf{X}) \ln \left\{ \frac{p(\mathbf{X}, \mathbf{Z})}{p(\mathbf{Z}|\mathbf{X})} \right\} d\mathbf{Z}$$

Intractable to compute $p(\mathbf{Z}|\mathbf{X})$

Forward or Backward KL-Divergence?



- It depends:
  - Backward KL-divergence is more sensible in the multi-modal example above (we will focus on Backward KL)
  - **But:** forward KL is also useful (see Expectation Propagation)

Image Source: "Pattern Recognition and Machine Learning", Christopher Bishop

Variational Inference

Two "Recipes"

Summary: Variational Approach

- Given $p(x, z)$, find an approximation $q(z)$ for the posterior distribution $p(z|x)$.
- **Key idea**: We choose the approximation $q(z)$ such that it minimizes the KL-divergence

$$\mathrm{KL}(q\|p) = -\int q(\mathbf{Z}) \ln \left\{ \frac{p(\mathbf{Z}|\mathbf{X})}{q(\mathbf{Z})} \right\} \mathrm{d}\mathbf{Z} \geq 0.$$

  i.e. $q(z)$ that close to $p(z|x)$.

- Maximizing the lower bound $\mathcal{L}(q)$ wrt $q(Z)$ is equivalent to minimizing the KL divergence.

$$\mathcal{L}(q) = \int q(\mathbf{Z}) \ln \left\{ \frac{p(\mathbf{X}, \mathbf{Z})}{q(\mathbf{Z})} \right\} \mathrm{d}\mathbf{Z}$$

Variational Inference: Two Approaches

1. Assume factorization for q, then derive the updates. Iterate until convergence.

OR

2. Assume factorization and parametric form for q, then derive the lower bound and optimize using off-the-shelf optimizer.

Mean-Field Approximation

- Mean-field theory: an approximation framework developed in physics.
- Partition the elements of Z into disjoint groups that we denote by Z_i where $i = 1, \ldots, M$.
- **Assume**: $q(z)$ factorizes with respect to these groups:

$$q(\mathbf{Z}) = \prod_{i=1}^{M} q_i(\mathbf{Z}_i).$$

Mean-Field Approximation

- Maximize $\mathcal{L}(q)$ w.r.t $q(\mathbf{Z})$:

$$\mathcal{L}(q) = \int q(\mathbf{Z}) \ln \left\{ \frac{p(\mathbf{X}, \mathbf{Z})}{q(\mathbf{Z})} \right\} \mathrm{d}\mathbf{Z} \qquad q(\mathbf{Z}) = \prod_{i=1}^{M} q_i(\mathbf{Z}_i).$$

- **Note:** Each $q_i(\mathbf{Z}_i; \phi_i)$ has parameters ϕ_i to optimize
- **Strategy:** Optimize $\mathcal{L}(q)$ w.r.t. factors $q_i(z_i; \phi_i)$

Variational Inference: Two Approaches

1. Assume factorization for q, then derive the updates. Iterate until convergence.

OR

2. Assume factorization and parametric form for q, then derive the lower bound and optimize using off-the-shelf optimizer.

Approach 1: Deriving the Updates

- Put the factorized distribution of $q(z)$ into the lower-bound $\mathcal{L}(q)$, and dissect out the dependence on one of the factors $q_j(z_j)$, we get:

$$\begin{aligned}
\mathcal{L}(q) &= \int \prod_i q_i \left\{ \ln p(\mathbf{X}, \mathbf{Z}) - \sum_i \ln q_i \right\} \mathrm{d}\mathbf{Z} \\
&= \int q_j \left\{ \int \ln p(\mathbf{X}, \mathbf{Z}) \prod_{i \neq j} q_i \, \mathrm{d}\mathbf{Z}_i \right\} \mathrm{d}\mathbf{Z}_j - \int q_j \ln q_j \, \mathrm{d}\mathbf{Z}_j + \text{const} \\
&= \int q_j \ln \widetilde{p}(\mathbf{X}, \mathbf{Z}_j) \, \mathrm{d}\mathbf{Z}_j - \int q_j \ln q_j \, \mathrm{d}\mathbf{Z}_j + \text{const}
\end{aligned}$$

We want to maximize w.r.t. each $q_j(z_j)$

Approach 1: Deriving the Updates

- We have defined a new distribution $\tilde{p}(x, z_j)$ by the relation:

$$\ln \widetilde{p}(\mathbf{X}, \mathbf{Z}_j) = \mathbb{E}_{i \neq j}[\ln p(\mathbf{X}, \mathbf{Z})] + \text{const.}$$

where

$$\mathbb{E}_{i \neq j}[\ln p(\mathbf{X}, \mathbf{Z})] = \int \ln p(\mathbf{X}, \mathbf{Z}) \prod_{i \neq j} q_i \, \mathrm{d}\mathbf{Z}_i$$

- $\mathbb{E}_{i \neq j}[\cdots]$ denotes an expectation w.r.t. q over all variables Z_i for $i \neq j$.

Approach 1: Deriving the Updates

$$\mathcal{L}(q) = \int q_j \ln \widetilde{p}(\mathbf{X}, \mathbf{Z}_j)\, \mathrm{d}\mathbf{Z}_j - \int q_j \ln q_j\, \mathrm{d}\mathbf{Z}_j + \text{const}$$

$$= \int q_j \ln \frac{\tilde{p}(\mathbf{x}, \mathbf{z}_j)}{q_j} d\mathbf{Z}_j + \text{const}$$

$$= -KL[q_j \parallel \tilde{p}(\mathbf{x}, \mathbf{z}_j)] + \text{const}$$

- The lower-bound is a negative KL-divergence that can be maximized by choosing

$$\ln q_j^{\star}(\mathbf{Z}_j) = \mathbb{E}_{i \neq j}[\ln p(\mathbf{X}, \mathbf{Z})] + \text{const}$$

Approach 1: Deriving the Updates

$$\ln q_j^\star(\mathbf{Z}_j) = \mathbb{E}_{i \neq j}[\ln p(\mathbf{X}, \mathbf{Z})] + \text{const}$$

- Since $q_j^*(z_j)$ must be a valid probability distribution, the constant is set by normalizing $q_j^*(z_j)$

$$q_j^\star(\mathbf{Z}_j) = \frac{\exp\left(\mathbb{E}_{i \neq j}[\ln p(\mathbf{X}, \mathbf{Z})]\right)}{\int \exp\left(\mathbb{E}_{i \neq j}[\ln p(\mathbf{X}, \mathbf{Z})]\right) \, \mathrm{d}\mathbf{Z}_j}.$$

Approach 1: Iterative Solution

1. Initialize factors $q_i(\mathbf{Z}_i)$
2. For each factor $q_j(\mathbf{Z}_j)$
   - Set $q_j(\mathbf{Z}_j) = \mathbb{E}_{i \neq j}[\ln p(\mathbf{X}, \mathbf{Z})] + \text{const}$
     with other factors q_i set to their *current* estimates
3. Repeat step 2 until convergence

- **Note:** Convergence is *guaranteed* since bound is convex wrt each $q_i(\mathbf{Z}_i)$

Variational Inference: Two Approaches

1. Assume factorization for q, then derive the updates. Iterate until convergence.

OR

2. Assume factorization and parametric form for q, then derive the lower bound and optimize using off-the-shelf optimizer.

Approach 2: Deriving $\mathcal{L}(q)$

- The ELBO is given by:

$$\mathcal{L}(q) = \int q(Z) \ln \frac{p(X,Z)}{q(Z)} dZ$$

$$= \mathbb{E}_{Z \sim q(Z)} \left[\ln \frac{p(X,Z)}{q(Z)} \right]$$

$$= \mathbb{E}_{Z \sim q(Z)} [\ln p(X,Z)] - \mathbb{E}_{Z \sim q(Z)} [\ln q(Z)]$$

$$= \mathbb{E}_{Z \sim q(Z)} [\ln p(X,Z)] + \mathbb{H}_q[Z]$$

Approach 2: Deriving $\mathcal{L}(q)$

- Leverage factorizations in the PGM and in q, e.g.,

  If $q(Z) = q_1(Z_1)q_2(Z_2)$ and

  $$p(X, Z_1, Z_2) = p(X|Z_1)p(Z_1|Z_2)p(Z_2)$$

  Then,

  $$\mathcal{L}(q) = \mathbb{E}_{Z\sim q(Z)}[\ln p(X, Z)] + \mathbb{H}_q[Z]$$
  $$= \mathbb{E}_{Z_1}[\ln p(X|Z_1)] + \mathbb{E}_{Z_1,Z_2}[\ln p(Z_1|Z_2)] + \mathbb{E}_{Z_2}[\ln p(Z_2)]$$
  $$+\mathbb{H}_{q_1}[Z_1] + \mathbb{H}_{q_2}[Z_2]$$

  Pick $q_1(Z_1)$ and $q_2(Z_2)$ so that terms are

  - Can represent the posterior sufficiently well
  - Easy to evaluate

  See upcoming examples.

Approach 2: Derive $L(q)$ and Optimize

1. Derive $\mathcal{L}(q) = \mathbb{E}_{Z \sim q(Z)}[\ln p(X, Z)] + \mathbb{H}_q[Z]$
2. Initialize all factors $q_i(\mathbf{Z}_i; \phi_i)$
3. Optimize $\mathcal{L}(q)$ wrt to parameters ϕ_i using your favorite optimizer

Variational Inference: Two Approaches

1. Assume factorization for q, then derive the updates. Iterate until convergence.

OR

2. Assume factorization and parametric form for q, then derive the lower bound and optimize using off-the-shelf optimizer.

NUS National University of Singapore | School of Computing

Variational Inference Example: Univariate Gaussian

Example 1: The Univariate Gaussian

- **Given**: a data set $\mathcal{D} = \{x_1, \ldots, x_N\}$ of observed values of X, which are assumed to be drawn independently from the Gaussian.

- **Goal**: infer the posterior distribution for the mean μ and precision τ (inverse of the covariance).

- The likelihood function is given by:

$$p(\mathcal{D}|\mu, \tau) = \left(\frac{\tau}{2\pi}\right)^{N/2} \exp\left\{-\frac{\tau}{2}\sum_{n=1}^{N}(x_n - \mu)^2\right\}.$$

Example 1: The Univariate Gaussian

- The conjugate prior distributions (Gaussian-Gamma) for μ and τ given by:

$$p(\mu|\tau) = \mathcal{N}\left(\mu|\mu_0, (\lambda_0\tau)^{-1}\right) \leftarrow (\mu_0, \lambda_0) \text{ : hyperparameters}$$
$$p(\tau) = \text{Gam}(\tau|a_0, b_0)$$

- Gamma distribution:

$$\text{Gam}(\tau|a_0, b_0) = \frac{1}{\Gamma(a_0)} b^{a_0} \tau^{a_0-1} \exp(-b_0\tau)$$

where a_0 and b_0 are the hyperparameters.

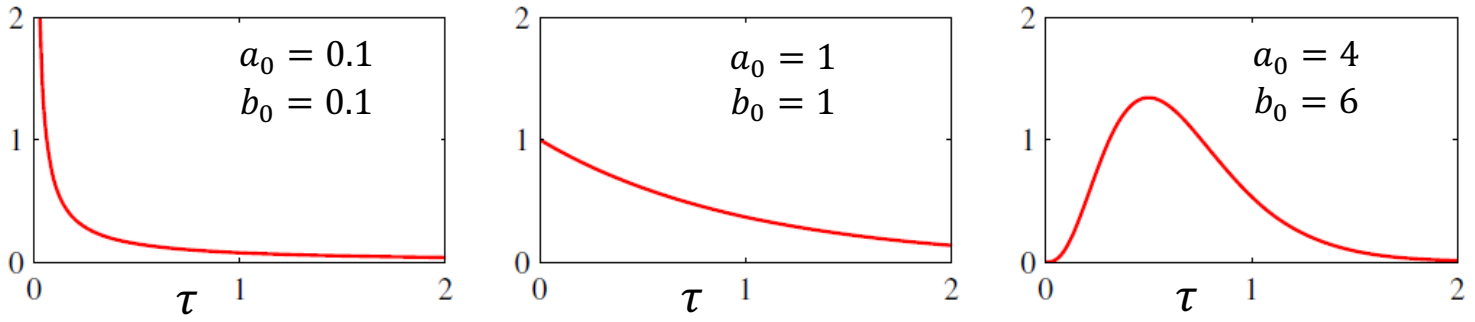


Image Source: "Pattern Recognition and Machine Learning", Christopher Bishop

Example 1: The Univariate Gaussian

Note: For this simple problem the posterior distribution can be found exactly, but we will do this with variational approach as an exercise.

Example 1: The Univariate Gaussian

- Consider a factorized variational approximation

$$q(\mu, \tau) = q_\mu(\mu) q_\tau(\tau) \,, \quad \text{Recall:} \quad q(\mathbf{Z}) = \prod_{i=1}^{M} q_i(\mathbf{Z}_i).$$

Variational Inference: Two Approaches

1. Assume factorization for q, then derive the updates. Iterate until convergence.

OR

2. Assume factorization and parametric form for q, then derive the lower bound and optimize using off-the-shelf optimizer.

Approach 1

- We consider a factorized variational approximation to the posterior distribution given by:

$$q(\mu, \tau) = q_\mu(\mu) q_\tau(\tau)\,, \quad \text{Recall:} \quad q(\mathbf{Z}) = \prod_{i=1}^{M} q_i(\mathbf{Z}_i).$$

- The optimum factors $q_\mu(\mu)$ and $q_\tau(\tau)$ can be obtained from the general result:

$$\ln q_j^\star(\mathbf{Z}_j) = \mathbb{E}_{i \neq j}[\ln p(\mathbf{X}, \mathbf{Z})] + \text{const}$$

Approach 1: $q_\mu(\mu)$

- The optimal solution for $\boldsymbol{q_\mu(\mu)}$ is given by:

$$\begin{aligned}\ln q_\mu^\star(\mu) &= \mathbb{E}_\tau\left[\ln p(\mathcal{D}|\mu,\tau) + \ln p(\mu|\tau)\right] + \text{const} \\ &= -\frac{\mathbb{E}[\tau]}{2}\left\{\lambda_0(\mu-\mu_0)^2 + \sum_{n=1}^{N}(x_n-\mu)^2\right\} + \text{const.}\end{aligned}$$

- Completing the square over μ, we see that $q_\mu(\mu)$ is a Gaussian $\mathcal{N}(\mu|\mu_N, \lambda_N^{-1})$ with mean and precision given by:

$$\begin{aligned}\mu_N &= \frac{\lambda_0\mu_0 + N\overline{x}}{\lambda_0 + N} \\ \lambda_N &= (\lambda_0 + N)\mathbb{E}[\tau].\end{aligned}$$

$N \to \infty \Longrightarrow \mu_N = \overline{x}$ and precision is infinite, i.e. maximum likelihood

Approach 1: $q_\tau(\tau)$

- Similarly, the optimal solution for the factor $\boldsymbol{q_\tau(\tau)}$ is given by:

$$\begin{aligned}\ln q_\tau^\star(\tau) &= \mathbb{E}_\mu\left[\ln p(\mathcal{D}|\mu,\tau) + \ln p(\mu|\tau)\right] + \ln p(\tau) + \text{const} \\ &= (a_0 - 1)\ln\tau - b_0\tau + \frac{N}{2}\ln\tau \\ &\quad -\frac{\tau}{2}\mathbb{E}_\mu\left[\sum_{n=1}^{N}(x_n - \mu)^2 + \lambda_0(\mu - \mu_0)^2\right] + \text{const}\end{aligned}$$

- Hence, $q_\tau(\tau)$ is a gamma distribution $\text{Gam}(\tau\,|a_N, b_N)$ with parameters:

$$a_N = a_0 + \frac{N}{2}$$

$$b_N = b_0 + \frac{1}{2}\mathbb{E}_\mu\left[\sum_{n=1}^{N}(x_n - \mu)^2 + \lambda_0(\mu - \mu_0)^2\right].$$

No functional forms specified!

- Did not assume any specific functional forms for the optimal distributions $q_\mu(\mu)$ and $q_\tau(\tau)$.

- They arose naturally from the structure of the likelihood function and the corresponding conjugate priors.

Solutions are Coupled

$q_\mu(\mu) = \mathcal{N}(\mu|\mu_N, \lambda_N^{-1})$, where

$$\begin{aligned} \mu_N &= \frac{\lambda_0\mu_0 + N\overline{x}}{\lambda_0 + N} \\ \lambda_N &= (\lambda_0 + N)\mathbb{E}[\tau]. \end{aligned}$$

Depends on $q_\tau(\tau)$!

$q_\tau(\tau) = \text{Gam}(\tau\,|a_N, b_N)$, where

Depends on $q_\mu(\mu)$!

$$\begin{aligned} a_N &= a_0 + \frac{N}{2} \\ b_N &= b_0 + \frac{1}{2}\mathbb{E}_\mu\left[\sum_{n=1}^{N}(x_n - \mu)^2 + \lambda_0(\mu - \mu_0)^2\right]. \end{aligned}$$

Recall: Iterative Solution

1. Initialize factors $q_i(\mathbf{Z}_i)$
2. For each factor $q_j(\mathbf{Z}_j)$
   - Set $q_j(\mathbf{Z}_j) = \mathbb{E}_{i \neq j}[\ln p(\mathbf{X}, \mathbf{Z})] + \text{const}$
     with other factors q_i set to their *current* estimates
3. Repeat step 2 until convergence

- **Note:** Convergence is *guaranteed* since bound is convex wrt each $q_i(\mathbf{Z}_i)$

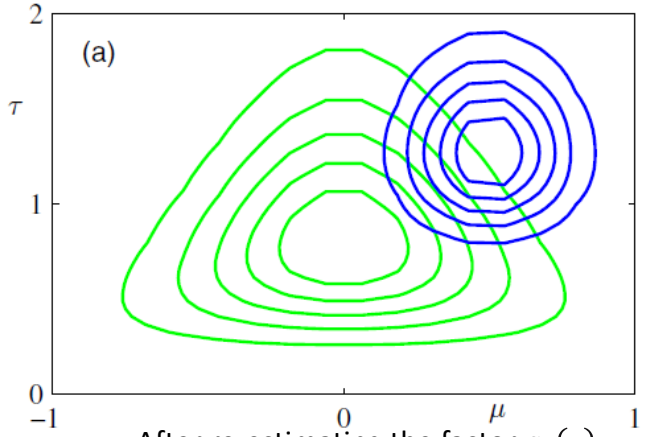
Iterative Solution

1. Use initial guess for $q_\tau(\tau)$ to re-compute $q_\mu(\mu)$.
2. Use the revised $q_\mu(\mu)$ to compute $\mathbb{E}[\mu]$ and $\mathbb{E}[\mu^2]$, and use these to re-compute $q_\tau(\tau)$.
3. Use revised $q_\tau(\tau)$ to compute $\mathbb{E}[\tau]$ and use this to re-compute $q_\mu(\mu)$.
4. Repeat until convergence.

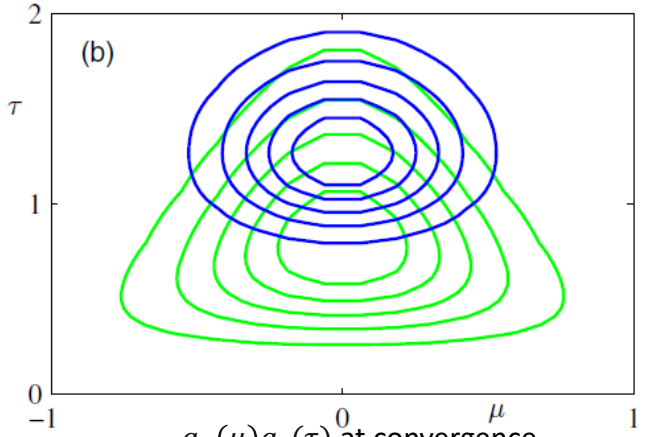
Example 1: The Univariate Gaussian

Contours of the true posterior distribution $p(\mu, \tau \mid \mathcal{D})$ are shown in green

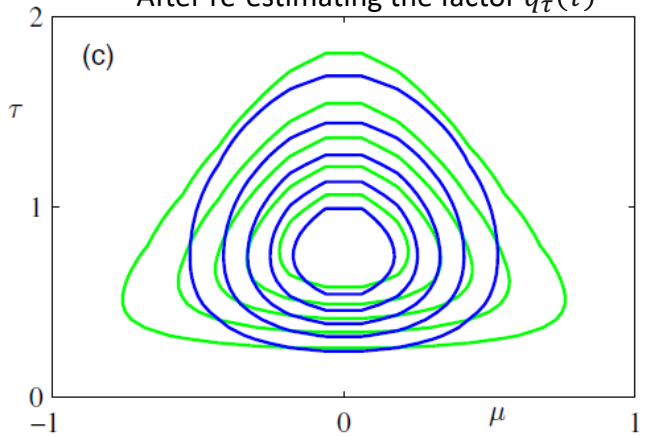
Initial factorized approximation $q_\mu(\mu)q_\tau(\tau)$



After re-estimating the factor $q_\mu(\mu)$



After re-estimating the factor $q_\tau(\tau)$



$q_\mu(\mu)q_\tau(\tau)$ at convergence

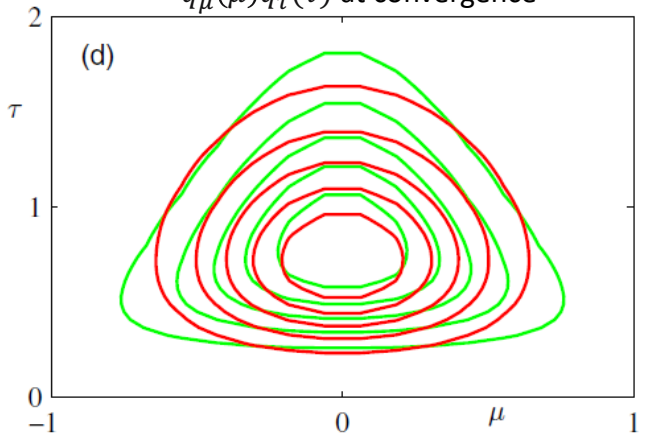


Image Source: "Pattern Recognition and Machine Learning", Christopher Bishop

Variational Bayes: Two Methods

1. Assume factorization for q, then derive the updates. Iterate until convergence.

OR

2. Assume factorization and parametric form for q, then derive the lower bound and optimize using off-the-shelf optimizer.

Deriving $L(q)$

$$\mathcal{L}(q) = \mathbb{E}_{\mu,\tau}[\ln p(D|\mu,\tau)] + \mathbb{E}_{\mu,\tau}[\ln p(\mu|\tau)] + \mathbb{E}_{\tau}[\ln p(\tau)] + \mathbb{E}_{\mu}[\ln q(\mu)] + \mathbb{E}_{\tau}[\ln q(\tau)]$$

Please see derivation sheet for full formula.

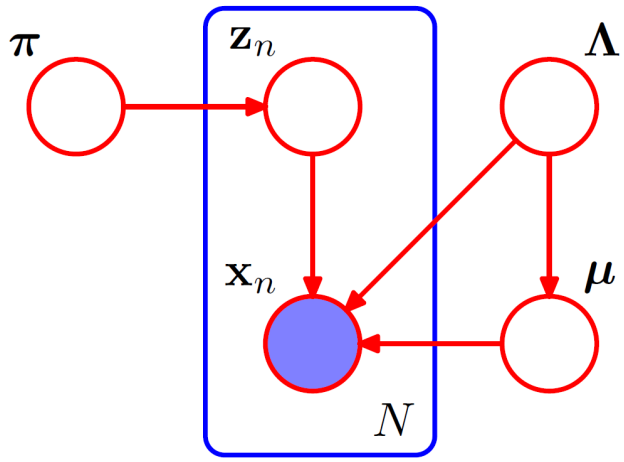
Maximize $\mathcal{L}(q)$ with your favorite optimizer.

Note: In the "old days", had to derive gradients, but today common to just use automatic differentiation (e.g., Autograd).

NUS National University of Singapore | School of Computing

Example:
Variational Mixture of Gaussians

Gaussian Mixture Model



- For each observation X_n, we have a corresponding latent variable Z_n comprising a 1-of-K binary vector with elements Z_{nk} for $k = 1, \ldots, K$.
- We denote the observed data set by $X = \{X_1, \ldots, X_N\}$, and similarly we denote the latent variables set by $Z = \{Z_1, \ldots, Z_N\}$.

Image Source: "Pattern Recognition and Machine Learning", Christopher Bishop

Gaussian Mixture Model

- Conditional distribution of Z, given the mixing coefficients π:

$$p(\mathbf{Z}|\boldsymbol{\pi}) = \prod_{n=1}^{N}\prod_{k=1}^{K}\pi_k^{z_{nk}}. \qquad \text{(Categorial distribution)}$$

- Dirichlet distribution conjugate prior over the mixing coefficients π:

Normalizing constant

$$p(\boldsymbol{\pi}) = \mathrm{Dir}(\boldsymbol{\pi}|\boldsymbol{\alpha}_0) = C(\boldsymbol{\alpha}_0)\prod_{k=1}^{K}\pi_k^{\alpha_0-1}$$

- Where we chose the same hyperparameter α_0 for each component.

Gaussian Mixture Model

- Conditional distribution of observed data vectors, given latent variables and component parameters:

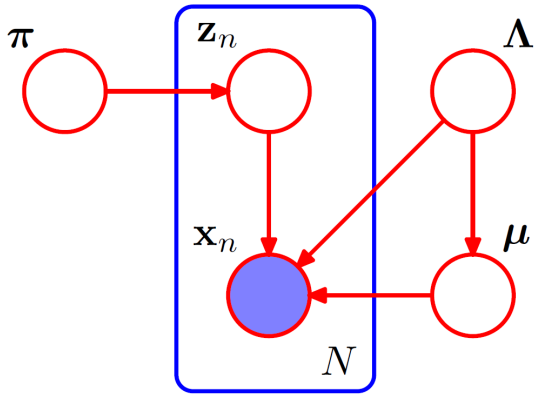
$$p(\mathbf{X}|\mathbf{Z},\boldsymbol{\mu},\boldsymbol{\Lambda}) = \prod_{n=1}^{N}\prod_{k=1}^{K}\mathcal{N}\left(\mathbf{x}_n|\boldsymbol{\mu}_k,\boldsymbol{\Lambda}_k^{-1}\right)^{z_{nk}}$$

- Independent Gaussian-Wishart prior governing the mean and precision of each Gaussian component:

$$\begin{aligned} p(\boldsymbol{\mu},\boldsymbol{\Lambda}) &= p(\boldsymbol{\mu}|\boldsymbol{\Lambda})p(\boldsymbol{\Lambda}) \\ &= \prod_{k=1}^{K}\mathcal{N}\left(\boldsymbol{\mu}_k|\mathbf{m}_0,(\beta_0\boldsymbol{\Lambda}_k)^{-1}\right)\mathcal{W}(\boldsymbol{\Lambda}_k|\mathbf{W}_0,\nu_0) \end{aligned}$$

- Where $\mathrm{m}_0, \mathrm{W}_0, \nu_0$ are the hyperparameters.

Gaussian Mixture Model



- Joint distribution:

$$p(\mathbf{X}, \mathbf{Z}, \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Lambda}) = p(\mathbf{X}|\mathbf{Z}, \boldsymbol{\mu}, \boldsymbol{\Lambda})p(\mathbf{Z}|\boldsymbol{\pi})p(\boldsymbol{\pi})p(\boldsymbol{\mu}|\boldsymbol{\Lambda})p(\boldsymbol{\Lambda})$$

- Consider a variational distribution which factorizes between the latent variables and parameters:

$$q(\mathbf{Z}, \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Lambda}) = q(\mathbf{Z})q(\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Lambda}).$$

Image Source: "Pattern Recognition and Machine Learning", Christopher Bishop

Variational Inference: Two Approaches

1. Assume factorization for q, then derive the updates. Iterate until convergence.

OR

2. Assume factorization and parametric form for q, then derive the lower bound and optimize using off-the-shelf optimizer.

Approach 1

- The log of the optimized factor for $\boldsymbol{q(z)}$ is given by:

$$\ln q^{\star}(\mathbf{Z}) = \mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Lambda}}[\ln p(\mathbf{X}, \mathbf{Z}, \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Lambda})] + \text{const.}$$

- Substituting the joint distribution and absorbing all terms that do not depend on Z into the additive constant, we get:

$$\ln q^{\star}(\mathbf{Z}) = \mathbb{E}_{\boldsymbol{\pi}}[\ln p(\mathbf{Z}|\boldsymbol{\pi})] + \mathbb{E}_{\boldsymbol{\mu},\boldsymbol{\Lambda}}[\ln p(\mathbf{X}|\mathbf{Z}, \boldsymbol{\mu}, \boldsymbol{\Lambda})] + \text{const.}$$

Approach 1

- Substituting the two conditional distributions, and again absorbing all terms independent of Z into the additive constant, we have:

$$\ln q^{\star}(\mathbf{Z}) = \sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} \ln \rho_{nk} + \text{const}$$

where we have defined

Dimensionality of data variable x

$$\begin{aligned} \ln \rho_{nk} \quad = \quad & \mathbb{E}[\ln \pi_k] + \frac{1}{2}\mathbb{E}\left[\ln |\mathbf{\Lambda}_k|\right] - \frac{D}{2}\ln(2\pi) \\ & -\frac{1}{2}\mathbb{E}_{\boldsymbol{\mu}_k, \mathbf{\Lambda}_k}\left[(\mathbf{x}_n - \boldsymbol{\mu}_k)^{\mathrm{T}} \mathbf{\Lambda}_k (\mathbf{x}_n - \boldsymbol{\mu}_k)\right] \end{aligned}$$

Approach 1

- Taking the exponential of both sides, we obtain:

$$q^{\star}(\mathbf{Z}) \propto \prod_{n=1}^{N} \prod_{k=1}^{K} \rho_{nk}^{z_{nk}}.$$

- Normalizing this distribution, and noting that $z_{nk} = \{0,1\}$ and sum to 1 over all values of k, we obtain:

$$q^{\star}(\mathbf{Z}) = \prod_{n=1}^{N} \prod_{k=1}^{K} r_{nk}^{z_{nk}}$$

← Same functional form as $p(\mathrm{z}|\pi)$

where

$$r_{nk} = \frac{\rho_{nk}}{\sum_{j=1}^{K} \rho_{nj}}.$$

Note that quantities r_{nk} are playing the role of responsibilities

Approach 1

- define three statistics of the observed data set evaluated with respect to the responsibilities:

$$
\begin{aligned}
N_k &= \sum_{n=1}^{N} r_{nk} \\
\overline{\mathbf{x}}_k &= \frac{1}{N_k} \sum_{n=1}^{N} r_{nk} \mathbf{x}_n \\
\mathbf{S}_k &= \frac{1}{N_k} \sum_{n=1}^{N} r_{nk} (\mathbf{x}_n - \overline{\mathbf{x}}_k)(\mathbf{x}_n - \overline{\mathbf{x}}_k)^{\mathrm{T}}.
\end{aligned}
$$

- These are analogous to quantities evaluated in the maximum likelihood EM algorithm for the GMM.

Approach 1: $q(\pi, \mu, \Lambda)$

- The log of the optimized factor for $\boldsymbol{q(\pi, \mu, \Lambda)}$ is given by:

Depends only on π

$$\ln q^{\star}(\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Lambda}) = \ln p(\boldsymbol{\pi}) + \sum_{k=1}^{K} \ln p(\boldsymbol{\mu}_k, \boldsymbol{\Lambda}_k) + \mathbb{E}_{\mathbf{Z}} \left[\ln p(\mathbf{Z}|\boldsymbol{\pi})\right]$$

$$+ \sum_{k=1}^{K} \sum_{n=1}^{N} \mathbb{E}[z_{nk}] \ln \mathcal{N}\left(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Lambda}_k^{-1}\right) + \text{const.}$$

Depends only on (μ, Λ)

Further factorizes into

$$q(\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Lambda}) = q(\boldsymbol{\pi}) \prod_{k=1}^{K} q(\boldsymbol{\mu}_k, \boldsymbol{\Lambda}_k)$$

Approach 1: $q(\pi)$

$\ln q^*(\pi)$ is given by:

$$\ln q^\star(\boldsymbol{\pi}) = (\alpha_0 - 1)\sum_{k=1}^{K} \ln \pi_k + \sum_{k=1}^{K}\sum_{n=1}^{N} r_{nk} \ln \pi_k + \text{const}$$

Taking exponential of both sides, we get:

$$q^\star(\boldsymbol{\pi}) = \text{Dir}(\boldsymbol{\pi}|\boldsymbol{\alpha})$$

Dirichlet distribution!

where $\boldsymbol{\alpha}$ has components α_k given by:

$$\alpha_k = \alpha_0 + N_k.$$

Approach 1: $q(\mu, \Lambda)$

And $\ln q^*(\mu, \Lambda)$ is given by:

r_{nk}: responsibility defined earlier

$$\ln q^\star(\boldsymbol{\mu}, \boldsymbol{\Lambda}) = \sum_{k=1}^{K} \ln p(\boldsymbol{\mu}_k, \boldsymbol{\Lambda}_k) + \sum_{k=1}^{K}\sum_{n=1}^{N} \mathbb{E}[z_{nk}] \ln \mathcal{N}\left(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Lambda}_k^{-1}\right) + \text{const.}$$

$$\mathcal{N}\left(\boldsymbol{\mu}_k | \mathbf{m}_0, (\beta_0 \boldsymbol{\Lambda}_k)^{-1}\right) \mathcal{W}(\boldsymbol{\Lambda}_k | \mathbf{W}_0, \nu_0)$$

Proof:

$$\mathbb{E}[z_{nk}] = \sum_z q(z) z_{nk}$$

$$= \sum_z \prod_n \prod_k r_{nk}^{z_{nk}} z_{nk}$$

$$= r_{nk}$$

where $q^\star(\mathbf{Z}) = \prod_{n=1}^{N}\prod_{k=1}^{K} r_{nk}^{z_{nk}}$

Approach 1: $q(\mu, \Lambda)$

Taking exponent on both sides, we get $q^*(\mu_k, \Lambda_k) = q^*(\mu_k|\Lambda_k)q^*(\Lambda_k)$, where:

$$q^\star(\boldsymbol{\mu}_k, \boldsymbol{\Lambda}_k) = \underbrace{\mathcal{N}\left(\boldsymbol{\mu}_k|\mathbf{m}_k, (\beta_k\boldsymbol{\Lambda}_k)^{-1}\right)}_{q^*(\mu_k|\Lambda_k)} \underbrace{\mathcal{W}(\boldsymbol{\Lambda}_k|\mathbf{W}_k, \nu_k)}_{q^*(\Lambda_k)}$$

Gaussian-Wishart distribution!

with:

$$\begin{aligned}
\beta_k &= \beta_0 + N_k \\
\mathbf{m}_k &= \frac{1}{\beta_k}(\beta_0\mathbf{m}_0 + N_k\overline{\mathbf{x}}_k) \\
\mathbf{W}_k^{-1} &= \mathbf{W}_0^{-1} + N_k\mathbf{S}_k + \frac{\beta_0 N_k}{\beta_0 + N_k}(\overline{\mathbf{x}}_k - \mathbf{m}_0)(\overline{\mathbf{x}}_k - \mathbf{m}_0)^{\mathrm{T}} \\
\nu_k &= \nu_0 + N_k.
\end{aligned}$$

Similar to M-step of the EM algorithm for mixture of Gaussians!

Dependent on responsibility $\mathbb{E}[z_{nk}]$

Iterative Solution

- As before, the solutions are coupled; we will use the iterative approach (similar to EM iterations).

$$\ln q^{\star}(\mathbf{Z}) = \sum_{n=1}^{N}\sum_{k=1}^{K} z_{nk} \ln \rho_{nk} + \text{const}$$

Depends on $q(\pi, \mu, \Lambda)$

where

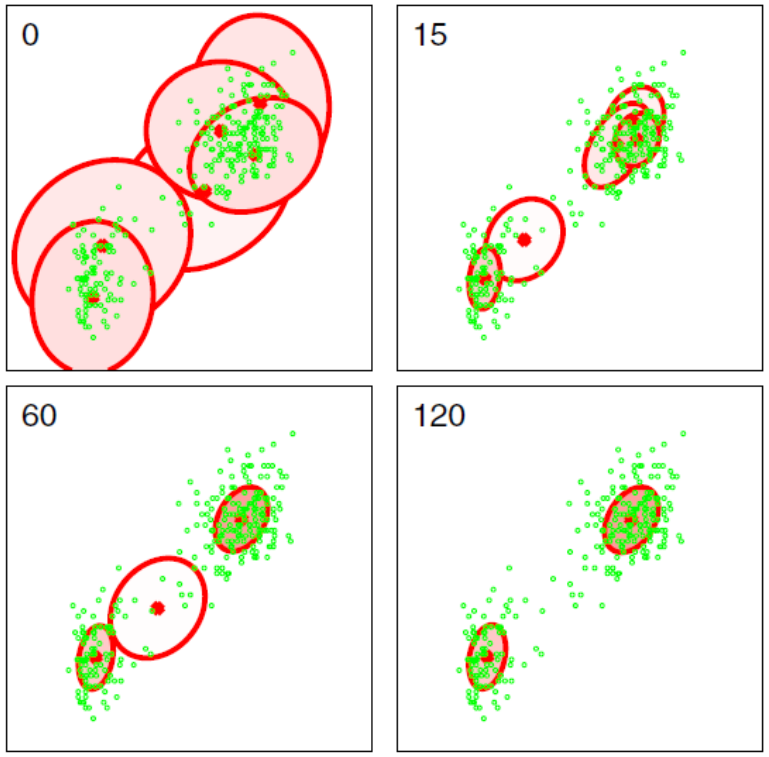
$$\ln \rho_{nk} = \mathbb{E}[\ln \pi_k] + \frac{1}{2}\mathbb{E}\left[\ln |\mathbf{\Lambda}_k|\right] - \frac{D}{2}\ln(2\pi) - \frac{1}{2}\mathbb{E}_{\boldsymbol{\mu}_k, \mathbf{\Lambda}_k}\left[(\mathbf{x}_n - \boldsymbol{\mu}_k)^{\mathrm{T}} \mathbf{\Lambda}_k (\mathbf{x}_n - \boldsymbol{\mu}_k)\right]$$

$$\ln q^{\star}(\boldsymbol{\pi}, \boldsymbol{\mu}, \mathbf{\Lambda}) = \ln p(\boldsymbol{\pi}) + \sum_{k=1}^{K} \ln p(\boldsymbol{\mu}_k, \mathbf{\Lambda}_k) + \mathbb{E}_{\mathbf{Z}}\left[\ln p(\mathbf{Z}|\boldsymbol{\pi})\right] + \sum_{k=1}^{K}\sum_{n=1}^{N} \mathbb{E}[z_{nk}] \ln \mathcal{N}\left(\mathbf{x}_n | \boldsymbol{\mu}_k, \mathbf{\Lambda}_k^{-1}\right) + \text{const.}$$

Depends on $q(Z)$

Example 2: Mixture of Gaussian

- Initialized with $K = 6$, but eventually converged to two unique clusters.
- All other unused components have the respective $r_{nk} \approx 0$.



Model selection!

Image Source: "Pattern Recognition and Machine Learning", Christopher Bishop

Variational Inference: Two Approaches

1. Assume factorization for q, then derive the updates. Iterate until convergence.

OR

2. Assume factorization and parametric form for q, then derive the lower bound and optimize using off-the-shelf optimizer.

Lower Bound: Mixture of Gaussians

$$\begin{aligned}
\mathcal{L} &= \sum_{\mathbf{Z}} \iiint q(\mathbf{Z}, \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Lambda}) \ln \left\{ \frac{p(\mathbf{X}, \mathbf{Z}, \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Lambda})}{q(\mathbf{Z}, \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Lambda})} \right\} \mathrm{d}\boldsymbol{\pi}\, \mathrm{d}\boldsymbol{\mu}\, \mathrm{d}\boldsymbol{\Lambda} \\
&= \mathbb{E}[\ln p(\mathbf{X}, \mathbf{Z}, \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Lambda})] - \mathbb{E}[\ln q(\mathbf{Z}, \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Lambda})] \\
&= \mathbb{E}[\ln p(\mathbf{X}|\mathbf{Z}, \boldsymbol{\mu}, \boldsymbol{\Lambda})] + \mathbb{E}[\ln p(\mathbf{Z}|\boldsymbol{\pi})] + \mathbb{E}[\ln p(\boldsymbol{\pi})] + \mathbb{E}[\ln p(\boldsymbol{\mu}, \boldsymbol{\Lambda})] \\
&\quad -\mathbb{E}[\ln q(\mathbf{Z})] - \mathbb{E}[\ln q(\boldsymbol{\pi})] - \mathbb{E}[\ln q(\boldsymbol{\mu}, \boldsymbol{\Lambda})]
\end{aligned}$$

$$\mathbb{E}[\ln p(\mathbf{X}|\mathbf{Z}, \boldsymbol{\mu}, \boldsymbol{\Lambda})] = \frac{1}{2} \sum_{k=1}^{K} N_k \left\{ \ln \widetilde{\Lambda}_k - D\beta_k^{-1} - \nu_k \mathrm{Tr}(\mathbf{S}_k \mathbf{W}_k) - \nu_k (\overline{\mathbf{x}}_k - \mathbf{m}_k)^{\mathrm{T}} \mathbf{W}_k (\overline{\mathbf{x}}_k - \mathbf{m}_k) - D \ln(2\pi) \right\}$$

$$\mathbb{E}[\ln p(\mathbf{Z}|\boldsymbol{\pi})] = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \ln \widetilde{\pi}_k$$

$$\mathbb{E}[\ln p(\boldsymbol{\pi})] = \ln C(\boldsymbol{\alpha}_0) + (\alpha_0 - 1) \sum_{k=1}^{K} \ln \widetilde{\pi}_k$$

$$\mathbb{E}[\ln p(\boldsymbol{\mu}, \boldsymbol{\Lambda})] = \frac{1}{2} \sum_{k=1}^{K} \left\{ D \ln(\beta_0/2\pi) + \ln \widetilde{\Lambda}_k - \frac{D\beta_0}{\beta_k} - \beta_0 \nu_k (\mathbf{m}_k - \mathbf{m}_0)^{\mathrm{T}} \mathbf{W}_k (\mathbf{m}_k - \mathbf{m}_0) \right\} + K \ln B(\mathbf{W}_0, \nu_0) + \frac{(\nu_0 - D - 1)}{2} \sum_{k=1}^{K} \ln \widetilde{\Lambda}_k - \frac{1}{2} \sum_{k=1}^{K} \nu_k \mathrm{Tr}(\mathbf{W}_0^{-1} \mathbf{W}_k)$$

$$\mathbb{E}[\ln q(\mathbf{Z})] = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \ln r_{nk}$$

$$\mathbb{E}[\ln q(\boldsymbol{\pi})] = \sum_{k=1}^{K} (\alpha_k - 1) \ln \widetilde{\pi}_k + \ln C(\boldsymbol{\alpha})$$

$$\mathbb{E}[\ln q(\boldsymbol{\mu}, \boldsymbol{\Lambda})] = \sum_{k=1}^{K} \left\{ \frac{1}{2} \ln \widetilde{\Lambda}_k + \frac{D}{2} \ln \left(\frac{\beta_k}{2\pi} \right) - \frac{D}{2} - \mathrm{H}\left[q(\boldsymbol{\Lambda}_k)\right] \right\}$$

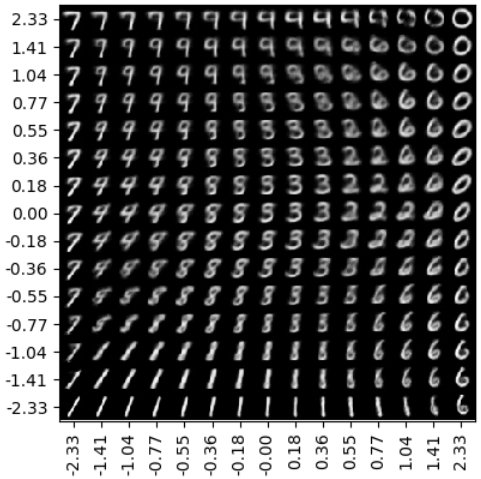
Image Source: “Pattern Recognition and Machine Learning”, Christopher Bishop

NUS National University of Singapore | School of Computing

Variational Inference Example: Variational Autoencoder

Generative Models with Variational Inference and Neural Networks

The Problem: Generative Modeling



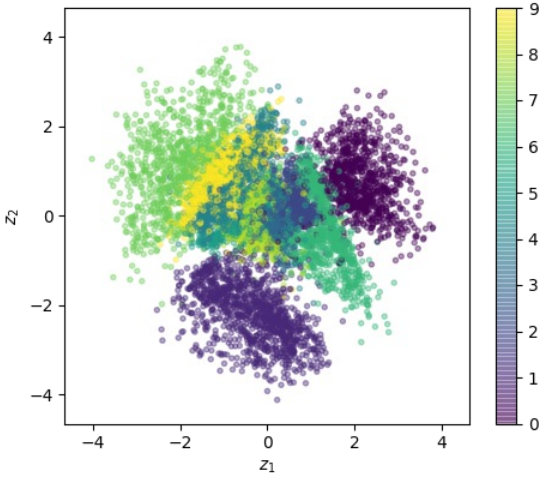


Image credit: https://tiao.io/post/tutorial-on-variational-autoencoders-with-a-concise-keraas-implementation/

Fatir and Soh, AAAI 2019

1 star	the food was good but the service was horrible . took forever to get our food . we had to ask twice for our check after we got our food . will not return .
2 star	the food was good , but the service was terrible . took forever to get someone to take our drink order . had to ask 3 times to get the check . food was ok , nothing to write about .
3 star	came here for the first time last night . food was good . service was a little slow . food was just ok .
4 star	food was good , service was a little slow , but the food was pretty good . i had the grilled chicken sandwich and it was really good . will definitely be back !
5 star	food was very good , service was fast and friendly . food was very good as well . will be back !

Yang, Zichao, et al. *ICML 2017*

Generative Model

- **The goal:** model a complex probability distribution that we can sample from.

$$\text{argmax}_\theta \log p_\theta(X)$$

- Introduce latent variable Z

$$p_\theta(X) = \int_z p_\theta(X|Z)p(Z)dZ$$

- **Note:** this is **not** Bayesian inference over the parameters. MLE estimation.

The Key Idea

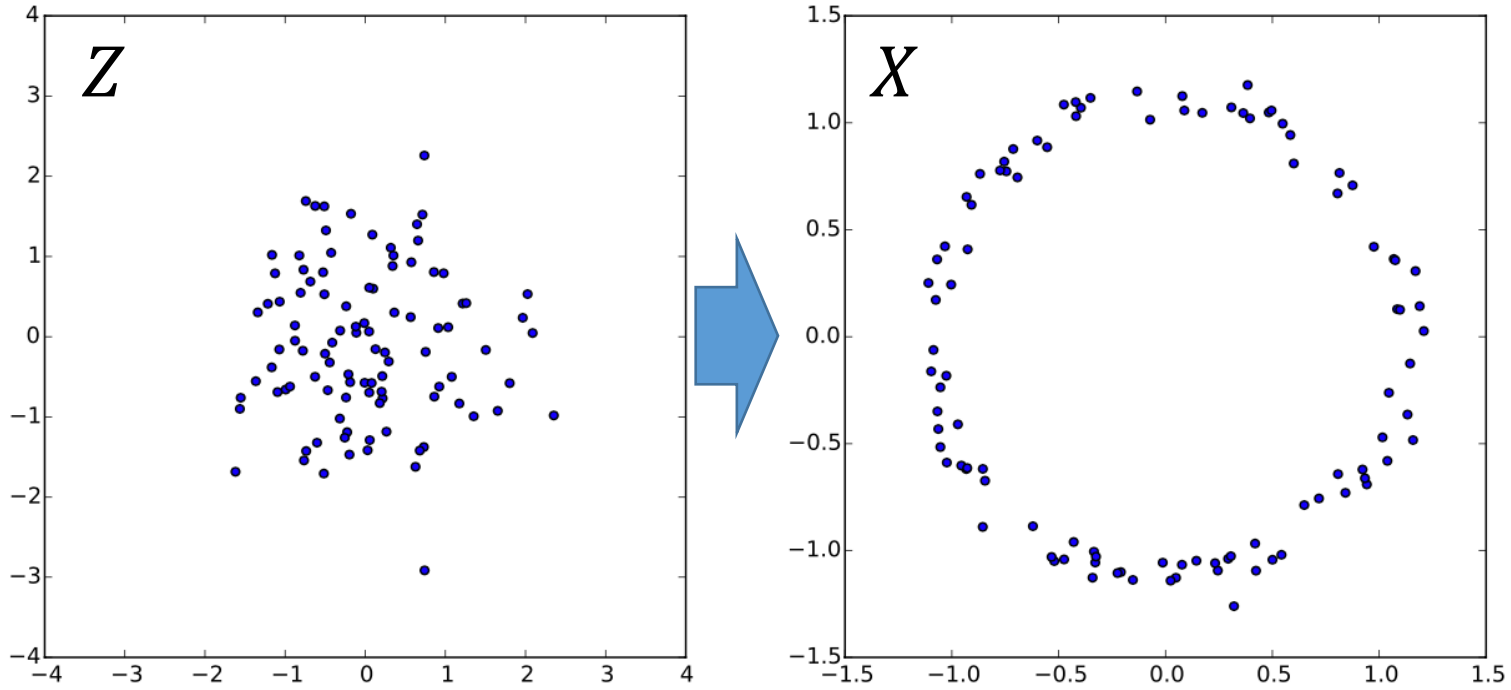
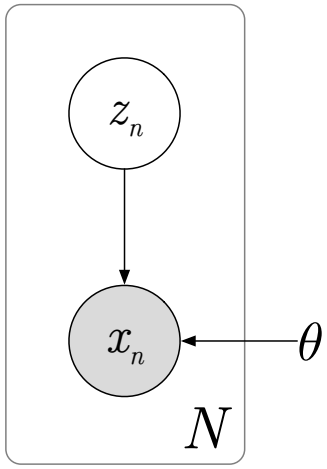
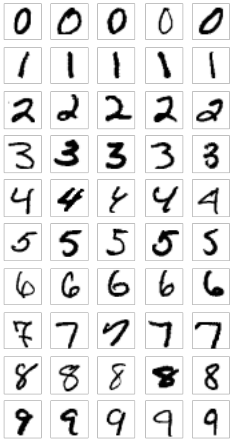


Image credit: **Tutorial on Variational Autoencoders,** Carl Doersch

Variational Autoencoder: The Graphical Model





Distributions

$$p(z_n) = N(z_n|0, I)$$

$$p(x_n|z_n) = N(x_n|\, f_\theta(z_n), \sigma^2 I)$$

$$p(x_n, z_n) = N(x_n|\, f_\theta(z_n), \sigma^2 I)N(z_n|0, I)$$

Or (for images)

$$p(x_n|z_n) = \text{Bern}\big(x_n|\, f_\theta(z_n)\big)$$

$$p(x_n, z_n) = \text{Bern}\big(x_n|\, f_\theta(z_n)\big)N(z_n|0, I)$$

Here, $f_\theta(z_n)$ is a complex function (e.g., neural network)

Inference is Difficult

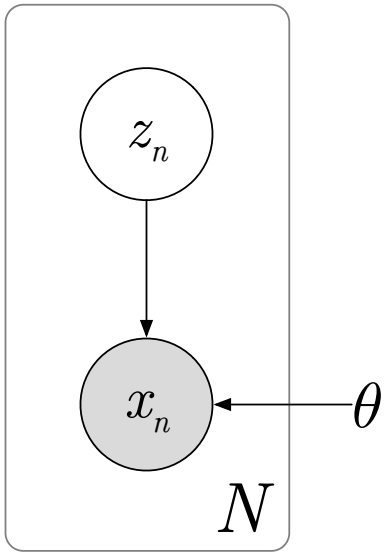
- Consider: $p(x_n, z_n) = \text{Bern}\big(x_n | f_\theta(z_n)\big) N(z_n | 0, I)$
- Given the specified distributions, computing

$$\log p_\theta(X) = \log \int_z p_\theta(X|Z) p(Z) dZ$$

  is intractable.

- **Difficulty:** Computing expectation of $p(X|Z)$ which contains a complicated function, e.g., a neural network

Inference is Difficult: EM?



Distributions

$$p(z_n) = N(z_n|0, I)$$

$$p(x_n|z_n) = N(x_n|\, f_\theta(z_n), \sigma^2 I)$$

$$p(x_n, z_n) = N(x_n|\, f_\theta(z_n), \sigma^2 I)N(z_n|0, I)$$

Here, $f_\theta(z_n)$ is a complex function (e.g., neural network)

Can we use EM to learn θ?

Variational Approach

Remember our lower bound:

$$\log p_\theta(x_n) = \mathcal{L}_n(\theta, \phi) + \mathbb{D}_{KL}[q_\phi(z_n|x_n)||\, p_\theta(z_n|x_n)]$$

- Here, introduce parameters ϕ for $q = q_\phi$
- **Let:** $q_\phi(z_n|x_n) = N(z_n|f_\phi^\mu(x_n), f_\phi^\Sigma(x_n))$
- Maximize the variational lower bound $\mathcal{L}_n(\theta, \phi)$
  - we will need gradients of $\mathcal{L}_n(\theta, \phi)$

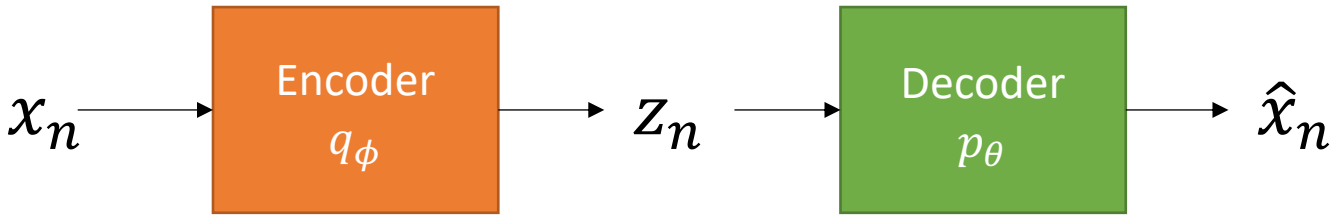
Analyzing $\mathcal{L}_n(\theta, \phi)$

- We rewrite:

$$\mathcal{L}_n(\theta,\phi) = \mathbb{E}_{q_\phi(z_n|x_n)}\left[\log p_\theta(x_n, z_n) - \log q_\phi(z_n|x_n)\right]$$

$$= \underbrace{\mathbb{E}_{q_\phi(z_n|x_n)}\left[\log p_\theta(x_n|z_n)\right]} - \mathbb{D}_{\mathrm{KL}}\left[q_\phi(z_n|x_n)\middle|p_\theta(z_n)\right]$$

Seen as: Encoder (q_ϕ) and Decoder (p_θ)



Maximizing $\mathcal{L}_n(\theta, \phi)$

- We rewrite:

$$\mathcal{L}_n(\theta, \phi) = \mathbb{E}_{q_\phi(z_n|x_n)}[\log p_\theta(x_n, z_n) - \log q_\phi(z_n|x_n)]$$

$$= \underbrace{\mathbb{E}_{q_\phi(z_n|x_n)}[\log p_\theta(x_n|z_n)]}_{\text{Hard to evaluate!}} - \underbrace{\mathbb{D}_{\mathrm{KL}}[q_\phi(z_n|x_n)|p_\theta(z_n)]}_{\text{Easy to evaluate!}}$$

Hard to evaluate!
Expectation of a complex function wrt a Normal

Easy to evaluate!
KL Divergence between two Normal distributions

$$p_\theta(x_n|z_n) = N(x_n|\, f_\theta(z_n), \sigma^2 I)$$

How to estimate $\mathbb{E}_{q_\phi(z_n|x_n)}[log\, p_\theta(x_n, |z_n)]$?

Lecture 9: The Monte Carlo Approach

- Approximate the expectation/integrals (or very large sums) $I(f)$ with tractable sums $I_N(f)$

$$I_N(f) = \frac{1}{N}\sum_{i}^{N} f(x^{(i)}) \approx I(f) = \int_{\mathcal{X}} f(x)p(x)dx$$

- $I_N(f)$ is an estimator for $I(f)$
- How "good" of an estimator is $I_N(f)$?

How to estimate $\mathbb{E}_{q_\phi(z_n|x_n)}[\log p_\theta(x_n, |z_n)]$?

- **Idea:** Use Sampling!

$$\mathbb{E}_{q_\phi(z_n|x_n)}[\log p_\theta(x_n|z_n)] \approx \frac{1}{L}\sum_{l}^{L} \log p_\theta(x_n|z_n^l)$$

Problem! ϕ no longer appears here! How can we get gradients to optimize q_ϕ?

Reparameterization Trick

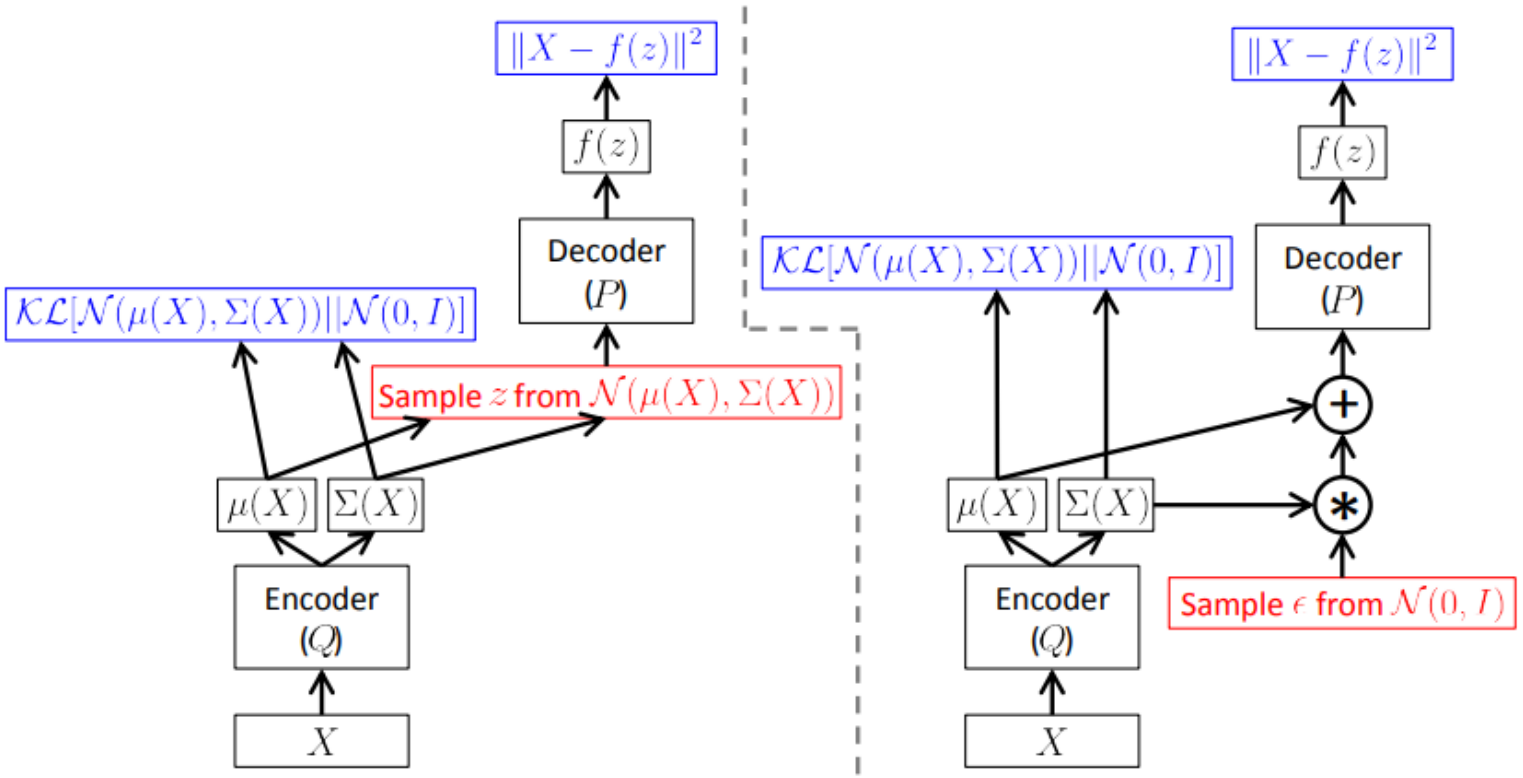


Image credit: Carl Doersch, "Tutorial on Variational Autoencoders"

Reparameterization Trick: Details

- Sample $\epsilon \sim N(0, I)$
- Transform ϵ to z via:

$$z = f_\phi^\mu(x_n) + \epsilon * \sqrt{f_\phi^\Sigma(x_n)}$$

- Then $z \sim N(z | f_\phi^\mu(x_n), f_\phi^\Sigma(x_n))$
- And the expectation becomes:

$$\mathbb{E}_{q_\phi(z_n|x_n)}[\log p_\theta(x_n|z_n)]$$

$$= \mathbb{E}_{\epsilon \sim \mathcal{N}(0,I)}\left[\log p_\theta\left(x_n | f_\phi^\mu(x_n) + \epsilon * \sqrt{f_\phi^\Sigma(x_n)}\right)\right]$$

Maximizing $\mathcal{L}_n(\theta, \phi)$

- We rewrite:

$$\mathcal{L}_n(\theta,\phi) = \mathbb{E}_{q_\phi(z_n|x_n)}\left[\log p_\theta(x_n, z_n) - \log q_\phi(z_n|x_n)\right]$$

$$= \underbrace{\mathbb{E}_{q_\phi(z_n|x_n)}\left[\log p_\theta(x_n|z_n)\right]}_{\text{Hard to evaluate! Expectation of a complex function wrt to a Normal}} - \underbrace{\mathbb{D}\left[q_\phi(z_n|x_n)\middle|p_\theta(z_n)\right]}_{\text{Easy to evaluate! KL Divergence between two Normal distributions}}$$

Sample using Re-parameterization trick!

Optimize using off-the-shelf optimizer (e.g., stochastic gradient descent, ADAM)

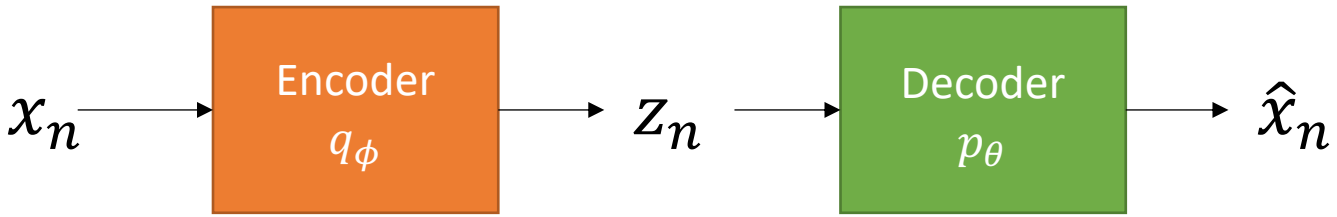
Putting it all-together

- We optimize:

$$\mathcal{L}_n(\theta, \phi) = \mathbb{E}_{q_\phi(z_n|x_n)}\left[\log p_\theta(x_n, z_n) - \log q_\phi(z_n|x_n)\right]$$

$$= \mathbb{E}_{\epsilon\sim\mathcal{N}(0,I)}\left[\log p_\theta\left(x_n|f_\phi^\mu(x_n) + \epsilon * \sqrt{f_\phi^\Sigma(x_n)}\right)\right] - \mathbb{D}_{\mathrm{KL}}\left[q_\phi(z_n|x_n)|p_\theta(z_n)\right]$$

where $\epsilon\sim\mathcal{N}(0, I)$



Learning Outcomes

- Students should be able to:

1. Explain the concept of variational approach using Lower-Bound of maximum likelihood and KL-divergence.
2. Use variational approach to perform inference on graphical models containing hidden variables.
3. Explain the variational autoencoder (VAE).