Latent Variable Models (LVMs) and Inference in LVMs

Piyush Rai

Topics in Probabilistic Modeling and Inference (CS698X)

Feb 9, 2019

Recap: Conditional Posterior and Local Conjugacy

- Consider some model with multiple unknowns $\theta_1, \dots, \theta_K$ and observations $\mathbf{X}$

Recap: Conditional Posterior and Local Conjugacy

- Consider some model with multiple unknowns $\theta_1, \ldots, \theta_K$ and observations $\mathbf{X}$
- The joint posterior $p(\Theta|\mathbf{X})$ may not be computable but often conditional posteriors are..

Recap: Conditional Posterior and Local Conjugacy

- Consider some model with multiple unknowns $\theta_1, \ldots, \theta_K$ and observations $\mathbf{X}$
- The joint posterior $p(\Theta|\mathbf{X})$ may not be computable but often conditional posteriors are..
- The conditional posterior, a.k.a. local posterior, a.k.a. complete conditional

$$p(\theta_k|\mathbf{X}, \Theta_{-k}) = \frac{p(\mathbf{X}|\theta_k, \Theta_{-k})p(\theta_k)}{\int p(\mathbf{X}|\theta_k, \Theta_{-k})p(\theta_k)d\theta_k} \propto p(\mathbf{X}|\theta_k, \Theta_{-k})p(\theta_k)$$

Recap: Conditional Posterior and Local Conjugacy

- Consider some model with multiple unknowns $\theta_1, \ldots, \theta_K$ and observations $\mathbf{X}$
- The joint posterior $p(\Theta|\mathbf{X})$ may not be computable but often conditional posteriors are..
- The conditional posterior, a.k.a. local posterior, a.k.a. complete conditional

$$p(\theta_k|\mathbf{X}, \Theta_{-k}) = \frac{p(\mathbf{X}|\theta_k, \Theta_{-k})p(\theta_k)}{\int p(\mathbf{X}|\theta_k, \Theta_{-k})p(\theta_k)d\theta_k} \propto p(\mathbf{X}|\theta_k, \Theta_{-k})p(\theta_k)$$

- This is computable easily if the model has Local Conjugacy

Recap: Conditional Posterior and Local Conjugacy

- Consider some model with multiple unknowns $\theta_1, \dots, \theta_K$ and observations $\mathbf{X}$
- The joint posterior $p(\Theta|\mathbf{X})$ may not be computable but often conditional posteriors are..
- The conditional posterior, a.k.a. local posterior, a.k.a. complete conditional

$$p(\theta_k|\mathbf{X}, \Theta_{-k}) = \frac{p(\mathbf{X}|\theta_k, \Theta_{-k})p(\theta_k)}{\int p(\mathbf{X}|\theta_k, \Theta_{-k})p(\theta_k)d\theta_k} \propto p(\mathbf{X}|\theta_k, \Theta_{-k})p(\theta_k)$$

- This is computable easily if the model has Local Conjugacy
  - $p(\mathbf{X}|\theta_k, \Theta_{-k})$ and $p(\theta_k)$ are conjugate to each other

Recap: Conditional Posterior and Local Conjugacy

- Consider some model with multiple unknowns $\theta_1, \ldots, \theta_K$ and observations $\mathbf{X}$
- The joint posterior $p(\Theta|\mathbf{X})$ may not be computable but often conditional posteriors are..
- The conditional posterior, a.k.a. local posterior, a.k.a. complete conditional

$$p(\theta_k|\mathbf{X}, \Theta_{-k}) = \frac{p(\mathbf{X}|\theta_k, \Theta_{-k})p(\theta_k)}{\int p(\mathbf{X}|\theta_k, \Theta_{-k})p(\theta_k)d\theta_k} \propto p(\mathbf{X}|\theta_k, \Theta_{-k})p(\theta_k)$$

- This is computable easily if the model has Local Conjugacy
  - $p(\mathbf{X}|\theta_k, \Theta_{-k})$ and $p(\theta_k)$ are conjugate to each other
  - The conditional posterior will also be in the same distribution family as $p(\theta_k)$

Recap: Conditional Posterior and Local Conjugacy

- Consider some model with multiple unknowns $\theta_1, \ldots, \theta_K$ and observations $\mathbf{X}$
- The joint posterior $p(\Theta|\mathbf{X})$ may not be computable but often conditional posteriors are..
- The conditional posterior, a.k.a. local posterior, a.k.a. complete conditional

$$p(\theta_k|\mathbf{X}, \Theta_{-k}) = \frac{p(\mathbf{X}|\theta_k, \Theta_{-k})p(\theta_k)}{\int p(\mathbf{X}|\theta_k, \Theta_{-k})p(\theta_k)d\theta_k} \propto p(\mathbf{X}|\theta_k, \Theta_{-k})p(\theta_k)$$

- This is computable easily if the model has Local Conjugacy
  - $p(\mathbf{X}|\theta_k, \Theta_{-k})$ and $p(\theta_k)$ are conjugate to each other
  - The conditional posterior will also be in the same distribution family as $p(\theta_k)$
- The conditional posteriors are used within various inference algorithms

Recap: Conditional Posterior and Local Conjugacy

- Consider some model with multiple unknowns $\theta_1, \ldots, \theta_K$ and observations $\mathbf{X}$
- The joint posterior $p(\Theta|\mathbf{X})$ may not be computable but often conditional posteriors are..
- The conditional posterior, a.k.a. local posterior, a.k.a. complete conditional

$$p(\theta_k|\mathbf{X}, \Theta_{-k}) = \frac{p(\mathbf{X}|\theta_k, \Theta_{-k})p(\theta_k)}{\int p(\mathbf{X}|\theta_k, \Theta_{-k})p(\theta_k)d\theta_k} \propto p(\mathbf{X}|\theta_k, \Theta_{-k})p(\theta_k)$$

- This is computable easily if the model has Local Conjugacy
  - $p(\mathbf{X}|\theta_k, \Theta_{-k})$ and $p(\theta_k)$ are conjugate to each other
  - The conditional posterior will also be in the same distribution family as $p(\theta_k)$
- The conditional posteriors are used within various inference algorithms, e.g.,
  - Expectation-Maximization (EM)

Recap: Conditional Posterior and Local Conjugacy

- Consider some model with multiple unknowns $\theta_1, \ldots, \theta_K$ and observations $\mathbf{X}$
- The joint posterior $p(\Theta|\mathbf{X})$ may not be computable but often conditional posteriors are..
- The conditional posterior, a.k.a. local posterior, a.k.a. complete conditional

$$p(\theta_k|\mathbf{X}, \Theta_{-k}) = \frac{p(\mathbf{X}|\theta_k, \Theta_{-k})p(\theta_k)}{\int p(\mathbf{X}|\theta_k, \Theta_{-k})p(\theta_k)d\theta_k} \propto p(\mathbf{X}|\theta_k, \Theta_{-k})p(\theta_k)$$

- This is computable easily if the model has Local Conjugacy
  - $p(\mathbf{X}|\theta_k, \Theta_{-k})$ and $p(\theta_k)$ are conjugate to each other
  - The conditional posterior will also be in the same distribution family as $p(\theta_k)$
- The conditional posteriors are used within various inference algorithms, e.g.,
  - Expectation-Maximization (EM)
  - Gibbs Sampling (an MCMC sampling algorithm)

Recap: Conditional Posterior and Local Conjugacy

- Consider some model with multiple unknowns $\theta_1, \ldots, \theta_K$ and observations $\mathbf{X}$
- The joint posterior $p(\Theta|\mathbf{X})$ may not be computable but often conditional posteriors are..
- The conditional posterior, a.k.a. local posterior, a.k.a. complete conditional

$$p(\theta_k|\mathbf{X}, \Theta_{-k}) = \frac{p(\mathbf{X}|\theta_k, \Theta_{-k})p(\theta_k)}{\int p(\mathbf{X}|\theta_k, \Theta_{-k})p(\theta_k)d\theta_k} \propto p(\mathbf{X}|\theta_k, \Theta_{-k})p(\theta_k)$$

- This is computable easily if the model has Local Conjugacy
  - $p(\mathbf{X}|\theta_k, \Theta_{-k})$ and $p(\theta_k)$ are conjugate to each other
  - The conditional posterior will also be in the same distribution family as $p(\theta_k)$
- The conditional posteriors are used within various inference algorithms, e.g.,
  - Expectation-Maximization (EM)
  - Gibbs Sampling (an MCMC sampling algorithm)
  - Mean-field variational inference

Recap: Gibbs Sampling

- Idea: Approximate a joint distribution using random samples from conditional distributions
- For the simple two-parameter case $\theta = (\theta_1, \theta_2)$, the Gibb sampler looks like this

Recap: Gibbs Sampling

- Idea: Approximate a joint distribution using random samples from conditional distributions
- For the simple two-parameter case $\theta = (\theta_1, \theta_2)$, the Gibb sampler looks like this
  - Initialize $\theta_2^{(0)}$

Recap: Gibbs Sampling

- Idea: Approximate a joint distribution using random samples from conditional distributions
- For the simple two-parameter case $\theta = (\theta_1, \theta_2)$, the Gibb sampler looks like this
  - Initialize $\theta_2^{(0)}$
  - For $s = 1, \ldots, S$

Recap: Gibbs Sampling

- Idea: Approximate a joint distribution using random samples from conditional distributions
- For the simple two-parameter case $\theta = (\theta_1, \theta_2)$, the Gibb sampler looks like this
  - Initialize $\theta_2^{(0)}$
  - For $s = 1, \ldots, S$
    - Draw a random sample for θ_1 as $\theta_1^{(s)} \sim p(\theta_1 | \theta_2^{(s-1)}, \boldsymbol{y})$

Recap: Gibbs Sampling

- Idea: Approximate a joint distribution using random samples from conditional distributions
- For the simple two-parameter case $\theta = (\theta_1, \theta_2)$, the Gibb sampler looks like this
  - Initialize $\theta_2^{(0)}$
  - For $s = 1, \ldots, S$
    - Draw a random sample for θ_1 as $\theta_1^{(s)} \sim p(\theta_1|\theta_2^{(s-1)}, \boldsymbol{y})$
    - Draw a random sample for θ_2 as $\theta_2^{(s)} \sim p(\theta_2|\theta_1^{(s)}, \boldsymbol{y})$

Recap: Gibbs Sampling

- Idea: Approximate a joint distribution using random samples from conditional distributions
- For the simple two-parameter case $\theta = (\theta_1, \theta_2)$, the Gibb sampler looks like this
  - Initialize $\theta_2^{(0)}$
  - For $s = 1, \ldots, S$
    - Draw a random sample for θ_1 as $\theta_1^{(s)} \sim p(\theta_1|\theta_2^{(s-1)}, \boldsymbol{y})$
    - Draw a random sample for θ_2 as $\theta_2^{(s)} \sim p(\theta_2|\theta_1^{(s)}, \boldsymbol{y})$
- The set of S random samples $\{\theta_1^{(s)}, \theta_2^{(s)}\}_{s=1}^{S}$ represent the joint posterior distribution $p(\theta_1, \theta_2|\boldsymbol{y})$

Recap: Gibbs Sampling

- Idea: Approximate a joint distribution using random samples from conditional distributions
- For the simple two-parameter case $\theta = (\theta_1, \theta_2)$, the Gibb sampler looks like this
  - Initialize $\theta_2^{(0)}$
  - For $s = 1, \ldots, S$
    - Draw a random sample for θ_1 as $\theta_1^{(s)} \sim p(\theta_1 | \theta_2^{(s-1)}, \boldsymbol{y})$
    - Draw a random sample for θ_2 as $\theta_2^{(s)} \sim p(\theta_2 | \theta_1^{(s)}, \boldsymbol{y})$
- The set of S random samples $\{\theta_1^{(s)}, \theta_2^{(s)}\}_{s=1}^{S}$ represent the joint posterior distribution $p(\theta_1, \theta_2 | \boldsymbol{y})$
- Can think of this as an empirical distribution with support only at the samples genered

$$p(\theta_1, \theta_2 | \boldsymbol{y}) \approx \frac{1}{S} \sum_{s=1}^{S} \delta_{\theta_1^{(s)}, \theta_2^{(s)}}(.)$$

where $\delta_x(.)$ denotes the Dirac distribution with mass only at x

Recap: Gibbs Sampling for Bayesian Matrix Factorization

- Randomly initialize the latent factors of users $\{\boldsymbol{u}_i\}_{i=1}^N$ and items $\{\boldsymbol{v}_j\}_{j=1}^M$

Recap: Gibbs Sampling for Bayesian Matrix Factorization

- Randomly initialize the latent factors of users $\{\boldsymbol{u}_i\}_{i=1}^N$ and items $\{\boldsymbol{v}_j\}_{j=1}^M$
- For $s = 1, \ldots, S$

Recap: Gibbs Sampling for Bayesian Matrix Factorization

- Randomly initialize the latent factors of users $\{\boldsymbol{u}_i\}_{i=1}^N$ and items $\{\boldsymbol{v}_j\}_{j=1}^M$
- For $s = 1, \ldots, S$
  - For each user $i = 1 : N$, draw a random sample for $\boldsymbol{u}_i$ as $\boldsymbol{u}_i^{(s)} \sim p(\boldsymbol{u}_i|\mathbf{R}, \mathbf{U}_{-i}, \mathbf{V})$

Recap: Gibbs Sampling for Bayesian Matrix Factorization

- Randomly initialize the latent factors of users $\{\boldsymbol{u}_i\}_{i=1}^N$ and items $\{\boldsymbol{v}_j\}_{j=1}^M$
- For $s = 1, \ldots, S$
  - For each user $i = 1 : N$, draw a random sample for $\boldsymbol{u}_i$ as $\boldsymbol{u}_i^{(s)} \sim p(\boldsymbol{u}_i | \mathbf{R}, \mathbf{U}_{-i}, \mathbf{V})$
  - For each item $j = 1 : N$, draw a random sample for $\boldsymbol{v}_j$ as $\boldsymbol{v}_j^{(s)} \sim p(\boldsymbol{v}_j | \mathbf{R}, \mathbf{V}_{-j}, \mathbf{U})$

Recap: Gibbs Sampling for Bayesian Matrix Factorization

- Randomly initialize the latent factors of users $\{\boldsymbol{u}_i\}_{i=1}^N$ and items $\{\boldsymbol{v}_j\}_{j=1}^M$
- For $s = 1, \ldots, S$
  - For each user $i = 1 : N$, draw a random sample for $\boldsymbol{u}_i$ as $\boldsymbol{u}_i^{(s)} \sim p(\boldsymbol{u}_i | \mathbf{R}, \mathbf{U}_{-i}, \mathbf{V})$
  - For each item $j = 1 : N$, draw a random sample for $\boldsymbol{v}_j$ as $\boldsymbol{v}_j^{(s)} \sim p(\boldsymbol{v}_j | \mathbf{R}, \mathbf{V}_{-j}, \mathbf{U})$
- Note: On the conditioning side, the most recent values of the latent factors are used

Recap: Gibbs Sampling for Bayesian Matrix Factorization

- Randomly initialize the latent factors of users $\{\boldsymbol{u}_i\}_{i=1}^N$ and items $\{\boldsymbol{v}_j\}_{j=1}^M$
- For $s = 1, \ldots, S$
  - For each user $i = 1 : N$, draw a random sample for $\boldsymbol{u}_i$ as $\boldsymbol{u}_i^{(s)} \sim p(\boldsymbol{u}_i|\mathbf{R}, \mathbf{U}_{-i}, \mathbf{V})$
  - For each item $j = 1 : N$, draw a random sample for $\boldsymbol{v}_j$ as $\boldsymbol{v}_j^{(s)} \sim p(\boldsymbol{v}_j|\mathbf{R}, \mathbf{V}_{-j}, \mathbf{U})$
- Note: On the conditioning side, the most recent values of the latent factors are used
- The posterior distribution is approximated as $p(\mathbf{U}, \mathbf{V}|\mathbf{R}) \approx \frac{1}{S}\sum_{s=1}^{S} \delta_{\mathbf{U}^{(s)},\mathbf{V}^{(s)}}(.)$

Recap: Gibbs Sampling for Bayesian Matrix Factorization

- Randomly initialize the latent factors of users $\{\boldsymbol{u}_i\}_{i=1}^N$ and items $\{\boldsymbol{v}_j\}_{j=1}^M$
- For $s = 1, \ldots, S$
  - For each user $i = 1 : N$, draw a random sample for $\boldsymbol{u}_i$ as $\boldsymbol{u}_i^{(s)} \sim p(\boldsymbol{u}_i | \mathbf{R}, \mathbf{U}_{-i}, \mathbf{V})$
  - For each item $j = 1 : N$, draw a random sample for $\boldsymbol{v}_j$ as $\boldsymbol{v}_j^{(s)} \sim p(\boldsymbol{v}_j | \mathbf{R}, \mathbf{V}_{-j}, \mathbf{U})$
- Note: On the conditioning side, the most recent values of the latent factors are used
- The posterior distribution is approximated as $p(\mathbf{U}, \mathbf{V} | \mathbf{R}) \approx \frac{1}{S} \sum_{s=1}^S \delta_{\mathbf{U}^{(s)}, \mathbf{V}^{(s)}}(.)$
- The posterior predictive distribution: $p(r_{ij} | \mathbf{R}) = \int \int p(r_{ij} | \boldsymbol{u}_i, \boldsymbol{v}_j) p(\boldsymbol{u}_i, \boldsymbol{v}_j | \mathbf{R}) d\boldsymbol{u}_i d\boldsymbol{v}_j$

Recap: Gibbs Sampling for Bayesian Matrix Factorization

- Randomly initialize the latent factors of users $\{\boldsymbol{u}_i\}_{i=1}^N$ and items $\{\boldsymbol{v}_j\}_{j=1}^M$
- For $s = 1, \ldots, S$
  - For each user $i = 1 : N$, draw a random sample for $\boldsymbol{u}_i$ as $\boldsymbol{u}_i^{(s)} \sim p(\boldsymbol{u}_i|\mathbf{R}, \mathbf{U}_{-i}, \mathbf{V})$
  - For each item $j = 1 : N$, draw a random sample for $\boldsymbol{v}_j$ as $\boldsymbol{v}_j^{(s)} \sim p(\boldsymbol{v}_j|\mathbf{R}, \mathbf{V}_{-j}, \mathbf{U})$
- Note: On the conditioning side, the most recent values of the latent factors are used
- The posterior distribution is approximated as $p(\mathbf{U}, \mathbf{V}|\mathbf{R}) \approx \frac{1}{S}\sum_{s=1}^{S} \delta_{\mathbf{U}^{(s)}, \mathbf{V}^{(s)}}(.)$
- The posterior predictive distribution: $p(r_{ij}|\mathbf{R}) = \int\int p(r_{ij}|\boldsymbol{u}_i, \boldsymbol{v}_j)p(\boldsymbol{u}_i, \boldsymbol{v}_j|\mathbf{R})d\boldsymbol{u}_i d\boldsymbol{v}_j$
- In general, posterior predictive is also hard to compute and needs approximation

Recap: Gibbs Sampling for Bayesian Matrix Factorization

- Randomly initialize the latent factors of users $\{\boldsymbol{u}_i\}_{i=1}^N$ and items $\{\boldsymbol{v}_j\}_{j=1}^M$
- For $s = 1, \dots, S$
  - For each user $i = 1 : N$, draw a random sample for $\boldsymbol{u}_i$ as $\boldsymbol{u}_i^{(s)} \sim p(\boldsymbol{u}_i|\mathbf{R}, \mathbf{U}_{-i}, \mathbf{V})$
  - For each item $j = 1 : N$, draw a random sample for $\boldsymbol{v}_j$ as $\boldsymbol{v}_j^{(s)} \sim p(\boldsymbol{v}_j|\mathbf{R}, \mathbf{V}_{-j}, \mathbf{U})$
- Note: On the conditioning side, the most recent values of the latent factors are used
- The posterior distribution is approximated as $p(\mathbf{U}, \mathbf{V}|\mathbf{R}) \approx \frac{1}{S}\sum_{s=1}^S \delta_{\mathbf{U}^{(s)},\mathbf{V}^{(s)}}(.)$
- The posterior predictive distribution: $p(r_{ij}|\mathbf{R}) = \int\int p(r_{ij}|\boldsymbol{u}_i, \boldsymbol{v}_j)p(\boldsymbol{u}_i, \boldsymbol{v}_j|\mathbf{R})d\boldsymbol{u}_i d\boldsymbol{v}_j$
- In general, posterior predictive is also hard to compute and needs approximation
  - We can use the S samples $\{\boldsymbol{u}_i^{(s)}, \boldsymbol{v}_j^{(s)}\}_{s=1}^S$ to compute the predictive mean via Monte-Carlo averaging

Recap: Gibbs Sampling for Bayesian Matrix Factorization

- Randomly initialize the latent factors of users $\{\boldsymbol{u}_i\}_{i=1}^N$ and items $\{\boldsymbol{v}_j\}_{j=1}^M$
- For $s = 1, \ldots, S$
  - For each user $i = 1 : N$, draw a random sample for $\boldsymbol{u}_i$ as $\boldsymbol{u}_i^{(s)} \sim p(\boldsymbol{u}_i|\mathbf{R}, \mathbf{U}_{-i}, \mathbf{V})$
  - For each item $j = 1 : N$, draw a random sample for $\boldsymbol{v}_j$ as $\boldsymbol{v}_j^{(s)} \sim p(\boldsymbol{v}_j|\mathbf{R}, \mathbf{V}_{-j}, \mathbf{U})$
- Note: On the conditioning side, the most recent values of the latent factors are used
- The posterior distribution is approximated as $p(\mathbf{U}, \mathbf{V}|\mathbf{R}) \approx \frac{1}{S}\sum_{s=1}^S \delta_{\mathbf{U}^{(s)},\mathbf{V}^{(s)}}(.)$
- The posterior predictive distribution: $p(r_{ij}|\mathbf{R}) = \int\int p(r_{ij}|\boldsymbol{u}_i, \boldsymbol{v}_j)p(\boldsymbol{u}_i, \boldsymbol{v}_j|\mathbf{R})d\boldsymbol{u}_i d\boldsymbol{v}_j$
- In general, posterior predictive is also hard to compute and needs approximation
  - We can use the S samples $\{\boldsymbol{u}_i^{(s)}, \boldsymbol{v}_j^{(s)}\}_{s=1}^S$ to compute the predictive mean via Monte-Carlo averaging
  - For the Gaussian likelihood case in matrix factorization, the mean can be computed as

$$\mathbb{E}[r_{ij}] \approx \frac{1}{S}\sum_{s=1}^S \boldsymbol{u}_i^{(s)\top} \boldsymbol{v}_j^{(s)}$$

Recap: Gibbs Sampling for Bayesian Matrix Factorization

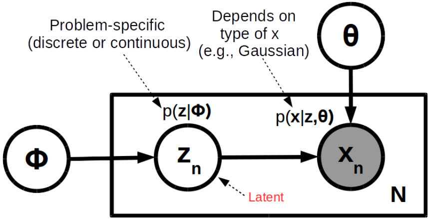
- Randomly initialize the latent factors of users $\{\boldsymbol{u}_i\}_{i=1}^N$ and items $\{\boldsymbol{v}_j\}_{j=1}^M$
- For $s = 1, \ldots, S$
  - For each user $i = 1 : N$, draw a random sample for $\boldsymbol{u}_i$ as $\boldsymbol{u}_i^{(s)} \sim p(\boldsymbol{u}_i|\mathbf{R}, \mathbf{U}_{-i}, \mathbf{V})$
  - For each item $j = 1 : N$, draw a random sample for $\boldsymbol{v}_j$ as $\boldsymbol{v}_j^{(s)} \sim p(\boldsymbol{v}_j|\mathbf{R}, \mathbf{V}_{-j}, \mathbf{U})$
- Note: On the conditioning side, the most recent values of the latent factors are used
- The posterior distribution is approximated as $p(\mathbf{U}, \mathbf{V}|\mathbf{R}) \approx \frac{1}{S}\sum_{s=1}^{S} \delta_{\mathbf{U}^{(s)},\mathbf{V}^{(s)}}(.)$
- The posterior predictive distribution: $p(r_{ij}|\mathbf{R}) = \int\int p(r_{ij}|\boldsymbol{u}_i, \boldsymbol{v}_j)p(\boldsymbol{u}_i, \boldsymbol{v}_j|\mathbf{R})d\boldsymbol{u}_i d\boldsymbol{v}_j$
- In general, posterior predictive is also hard to compute and needs approximation
  - We can use the S samples $\{\boldsymbol{u}_i^{(s)}, \boldsymbol{v}_j^{(s)}\}_{s=1}^S$ to compute the predictive mean via Monte-Carlo averaging
  - For the Gaussian likelihood case in matrix factorization, the mean can be computed as

$$\mathbb{E}[r_{ij}] \approx \frac{1}{S}\sum_{s=1}^{S} \boldsymbol{u}_i^{(s)\top} \boldsymbol{v}_j^{(s)}$$

  - Can also compute the variance of r_{ij} (think how)

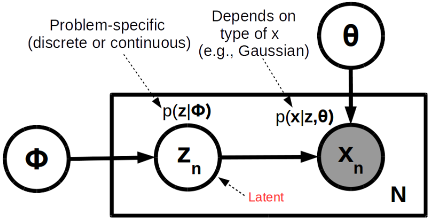
Latent Variable Models

Why Latent Variable Models?



- Application 1: Can use these to model latent properties/features of data, e.g.,
  - Cluster assignment of each observation (in mixture models)
  - Topic assignment of each word (in topic models such as Latent Dirichlet Allocation)
  - Low-dim rep. or "code" of each observation (e.g., prob. PCA, variational autoencoders, etc)

Why Latent Variable Models?



- Application 1: Can use these to model latent properties/features of data, e.g.,
  - Cluster assignment of each observation (in mixture models)
  - Topic assignment of each word (in topic models such as Latent Dirichlet Allocation)
  - Low-dim rep. or "code" of each observation (e.g., prob. PCA, variational autoencoders, etc)
- In such apps, latent variables ($\boldsymbol{z}_n$'s above) are called "local variables" (specific to individual obs.), and other unknown parameters/hyperparams (θ, ϕ above) are called "global variables"

Why Latent Variable Models?

- Application 2: Sometimes, augmenting a model by latent variables simplies inference
  - .. even if these latent variables aren't part of the original model definition

Why Latent Variable Models?

- Application 2: Sometimes, augmenting a model by latent variables simplies inference
  - .. even if these latent variables aren't part of the original model definition
- Some of the popular examples include
  - In Probit regression for binary classification, we can model each label $y_n \in \{0, 1\}$ as

$$y_n = \mathbb{I}[z_n > 0] \qquad \text{where} \quad z_n \sim \mathcal{N}(\boldsymbol{w}^\top \boldsymbol{x}_n, 1) \quad \text{is a Gaussian latent variable}$$

.. and use EM etc, to infer the unknowns $\boldsymbol{w}$ and z_n's (MLAPP 11.4.6, EM for Probit Regression)

Why Latent Variable Models?

- Application 2: Sometimes, augmenting a model by latent variables simplies inference
  - .. even if these latent variables aren't part of the original model definition
- Some of the popular examples include
  - In Probit regression for binary classification, we can model each label $y_n \in \{0, 1\}$ as

$$y_n = \mathbb{I}[z_n > 0] \qquad \text{where} \quad z_n \sim \mathcal{N}(\boldsymbol{w}^\top \boldsymbol{x}_n, 1) \quad \text{is a Gaussian latent variable}$$

  .. and use EM etc, to infer the unknowns $\boldsymbol{w}$ and z_n's (MLAPP 11.4.6, EM for Probit Regression)
  - Many sparse priors on weights can be thought of as Gaussian "scale-mixtures" (scale is variance)

$$\text{Laplace}(w_d|0, 1/\gamma) = \frac{\gamma}{2}\exp(-\gamma|w_d|) = \int \mathcal{N}(w_d|0, \tau_d^2)\text{Gamma}(\tau_d^2|1, \gamma^2/2)d\tau_d^2$$

  .. where τ_d's are latent vars. Can use EM to infer $\boldsymbol{w}$ and τ (MLAPP 13.4.4 - EM for LASSO)

Why Latent Variable Models?

- Application 2: Sometimes, augmenting a model by latent variables simplies inference
  - .. even if these latent variables aren't part of the original model definition
- Some of the popular examples include
  - In Probit regression for binary classification, we can model each label $y_n \in \{0, 1\}$ as

    $$y_n = \mathbb{I}[z_n > 0] \qquad \text{where} \quad z_n \sim \mathcal{N}(\boldsymbol{w}^\top \boldsymbol{x}_n, 1) \quad \text{is a Gaussian latent variable}$$

    .. and use EM etc, to infer the unknowns $\boldsymbol{w}$ and z_n's (MLAPP 11.4.6, EM for Probit Regression)
  - Many sparse priors on weights can be thought of as Gaussian "scale-mixtures" (scale is variance)

    $$\text{Laplace}(w_d|0, 1/\gamma) = \frac{\gamma}{2}\exp(-\gamma|w_d|) = \int \mathcal{N}(w_d|0, \tau_d^2)\text{Gamma}(\tau_d^2|1, \gamma^2/2)d\tau_d^2$$

    .. where τ_d's are latent vars. Can use EM to infer $\boldsymbol{w}$ and τ (MLAPP 13.4.4 - EM for LASSO)
- Such augmentation can make an originally non-conjugate model a conditionally conjugate one!

Why Latent Variable Models?

- Application 2: Sometimes, augmenting a model by latent variables simplies inference
  - .. even if these latent variables aren't part of the original model definition
- Some of the popular examples include
  - In Probit regression for binary classification, we can model each label $y_n \in \{0, 1\}$ as

$$y_n = \mathbb{I}[z_n > 0] \qquad \text{where} \quad z_n \sim \mathcal{N}(\boldsymbol{w}^\top \boldsymbol{x}_n, 1) \quad \text{is a Gaussian latent variable}$$

    .. and use EM etc, to infer the unknowns $\boldsymbol{w}$ and z_n's (MLAPP 11.4.6, EM for Probit Regression)
  - Many sparse priors on weights can be thought of as Gaussian "scale-mixtures" (scale is variance)

$$\text{Laplace}(w_d|0, 1/\gamma) = \frac{\gamma}{2}\exp(-\gamma|w_d|) = \int \mathcal{N}(w_d|0, \tau_d^2)\text{Gamma}(\tau_d^2|1, \gamma^2/2)d\tau_d^2$$

    .. where τ_d's are latent vars. Can use EM to infer $\boldsymbol{w}$ and τ (MLAPP 13.4.4 - EM for LASSO)
- Such augmentation can make an originally non-conjugate model a conditionally conjugate one!
  - Can then use Gibbs sampling, EM, and various other conditional posterior based inference algos

Nomenclature/Notation Alert

- Why call some unknowns as parameters and others as latent variables?

Nomenclature/Notation Alert

- Why call some unknowns as parameters and others as latent variables?
- Well, no specific reason. Sort of a convention adopted by some algorithms

Nomenclature/Notation Alert

- Why call some unknowns as parameters and others as latent variables?
- Well, no specific reason. Sort of a convention adopted by some algorithms
- EM refers to the unknowns estimated in E step as latent vars and those in M step as params

Nomenclature/Notation Alert

- Why call some unknowns as parameters and others as latent variables?
- Well, no specific reason. Sort of a convention adopted by some algorithms
- EM refers to the unknowns estimated in E step as latent vars and those in M step as params
  - Here the distinction is: Infer the posterior for latent vars and point estimates of parameters

Nomenclature/Notation Alert

- Why call some unknowns as parameters and others as latent variables?
- Well, no specific reason. Sort of a convention adopted by some algorithms
- EM refers to the unknowns estimated in E step as latent vars and those in M step as params
  - Here the distinction is: Infer the posterior for latent vars and point estimates of parameters
- In contrast, some algos that infer posteriors for all unknowns refer to everything as latent vars

Nomenclature/Notation Alert

- Why call some unknowns as parameters and others as latent variables?
- Well, no specific reason. Sort of a convention adopted by some algorithms
- EM refers to the unknowns estimated in E step as latent vars and those in M step as params
  - Here the distinction is: Infer the posterior for latent vars and point estimates of parameters
- In contrast, some algos that infer posteriors for all unknowns refer to everything as latent vars
- Sometimes the "global" or "local" unknown distinction makes it clear

Nomenclature/Notation Alert

- Why call some unknowns as parameters and others as latent variables?
- Well, no specific reason. Sort of a convention adopted by some algorithms
- EM refers to the unknowns estimated in E step as latent vars and those in M step as params
  - Here the distinction is: Infer the posterior for latent vars and point estimates of parameters
- In contrast, some algos that infer posteriors for all unknowns refer to everything as latent vars
- Sometimes the "global" or "local" unknown distinction makes it clear
  - Local variables = latent variables, global variables = parameters

Nomenclature/Notation Alert

- Why call some unknowns as parameters and others as latent variables?
- Well, no specific reason. Sort of a convention adopted by some algorithms
- EM refers to the unknowns estimated in E step as latent vars and those in M step as params
  - Here the distinction is: Infer the posterior for latent vars and point estimates of parameters
- In contrast, some algos that infer posteriors for all unknowns refer to everything as latent vars
- Sometimes the "global" or "local" unknown distinction makes it clear
  - Local variables = latent variables, global variables = parameters
- But remember that this nomenclature isn't really cast in stone, no need to be confused so long as you are clear as to what the role of each unknown is, and how we want to estimate it (posterior or point estimate) and using what type of inference algorithm

Hybrid Inference (posterior inference + point estimation)

Hybrid Inference (posterior inference + point estimation)

- In many models, we infer the posterior on some unknowns and do point estimation for the rest

Hybrid Inference (posterior inference + point estimation)

- In many models, we infer the posterior on some unknowns and do point estimation for the rest
- We have already seen that MLE-II based inference does that

Hybrid Inference (posterior inference + point estimation)

- In many models, we infer the posterior on some unknowns and do point estimation for the rest
- We have already seen that MLE-II based inference does that
  - Maximize the marginal likelihood to do point estimation for hyperparams

Hybrid Inference (posterior inference + point estimation)

- In many models, we infer the posterior on some unknowns and do point estimation for the rest
- We have already seen that MLE-II based inference does that
  - Maximize the marginal likelihood to do point estimation for hyperparams
  - Infer the conditional posterior over the main parameter given the point estimates of hyperparams

Hybrid Inference (posterior inference + point estimation)

- In many models, we infer the posterior on some unknowns and do point estimation for the rest
- We have already seen that MLE-II based inference does that
  - Maximize the marginal likelihood to do point estimation for hyperparams
  - Infer the conditional posterior over the main parameter given the point estimates of hyperparams
- The Expectation-Maximization algorithm (will see today) also does something similar

Hybrid Inference (posterior inference + point estimation)

- In many models, we infer the posterior on some unknowns and do point estimation for the rest
- We have already seen that MLE-II based inference does that
  - Maximize the marginal likelihood to do point estimation for hyperparams
  - Infer the conditional posterior over the main parameter given the point estimates of hyperparams
- The Expectation-Maximization algorithm (will see today) also does something similar
  - In E step, the conditional posterior of latent variables is inferred

Hybrid Inference (posterior inference + point estimation)

- In many models, we infer the posterior on some unknowns and do point estimation for the rest
- We have already seen that MLE-II based inference does that
  - Maximize the marginal likelihood to do point estimation for hyperparams
  - Infer the conditional posterior over the main parameter given the point estimates of hyperparams
- The Expectation-Maximization algorithm (will see today) also does something similar
  - In E step, the conditional posterior of latent variables is inferred
  - In M step, the expected complete data log-lik. is maximized to get point estimates of parameters

Hybrid Inference (posterior inference + point estimation)

- In many models, we infer the posterior on some unknowns and do point estimation for the rest
- We have already seen that MLE-II based inference does that
  - Maximize the marginal likelihood to do point estimation for hyperparams
  - Infer the conditional posterior over the main parameter given the point estimates of hyperparams
- The Expectation-Maximization algorithm (will see today) also does something similar
  - In E step, the conditional posterior of latent variables is inferred
  - In M step, the expected complete data log-lik. is maximized to get point estimates of parameters
- If we can't afford to (due to computational or other reasons) infer the posterior over all unknowns, how to decide which variables to infer posterior on, and for which to do point estimation?

Hybrid Inference (posterior inference + point estimation)

- In many models, we infer the posterior on some unknowns and do point estimation for the rest
- We have already seen that MLE-II based inference does that
  - Maximize the marginal likelihood to do point estimation for hyperparams
  - Infer the conditional posterior over the main parameter given the point estimates of hyperparams
- The Expectation-Maximization algorithm (will see today) also does something similar
  - In E step, the conditional posterior of latent variables is inferred
  - In M step, the expected complete data log-lik. is maximized to get point estimates of parameters
- If we can't afford to (due to computational or other reasons) infer the posterior over all unknowns, how to decide which variables to infer posterior on, and for which to do point estimation?
- Rule of thumb: Infer posterior over local variables and point estimates for global variables

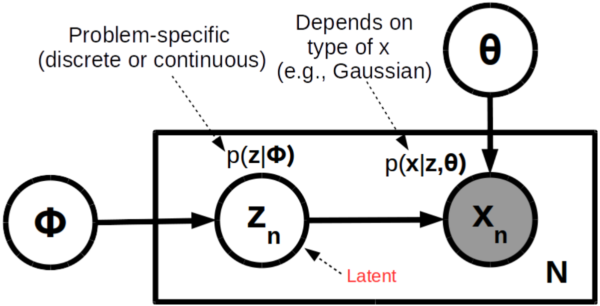
Hybrid Inference (posterior inference + point estimation)

- In many models, we infer the posterior on some unknowns and do point estimation for the rest
- We have already seen that MLE-II based inference does that
  - Maximize the marginal likelihood to do point estimation for hyperparams
  - Infer the conditional posterior over the main parameter given the point estimates of hyperparams
- The Expectation-Maximization algorithm (will see today) also does something similar
  - In E step, the conditional posterior of latent variables is inferred
  - In M step, the expected complete data log-lik. is maximized to get point estimates of parameters
- If we can't afford to (due to computational or other reasons) infer the posterior over all unknowns, how to decide which variables to infer posterior on, and for which to do point estimation?
- Rule of thumb: Infer posterior over local variables and point estimates for global variables
  - Reason: We typically have plenty of data to reliably estimate the global variables so it is okay even if we just do point estimation for those (recall the schools problem in HW1)

Inference/Parameter Estimation in Latent Variable Models using Expectation-Maximization

Parameter Estimation in Latent Variable Models

- Assume each observation $\boldsymbol{x}_n$ to be associated with a "local" latent variable $\boldsymbol{z}_n$



- Although we can do fully Bayesian inference for all the unknowns, suppose we only want a point estimate of the "global" parameters $\Theta = (\theta, \phi)$ via MLE/MAP

Why Estimation is Difficult in LVMs?

- Suppose we want to estimate parameters Θ via MLE

Why Estimation is Difficult in LVMs?

- Suppose we want to estimate parameters Θ via MLE. If we knew both $\boldsymbol{x}_n$ and $\boldsymbol{z}_n$ then we could do

$$\Theta_{MLE} = \arg\max_{\Theta} \sum_{n=1}^{N} \log p(\boldsymbol{x}_n, \boldsymbol{z}_n | \Theta)$$

Why Estimation is Difficult in LVMs?

- Suppose we want to estimate parameters Θ via MLE. If we knew both $\boldsymbol{x}_n$ and $\boldsymbol{z}_n$ then we could do

$$\Theta_{MLE} = \arg\max_{\Theta} \sum_{n=1}^{N} \log p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta) = \arg\max_{\Theta} \sum_{n=1}^{N} [\log p(\boldsymbol{z}_n|\phi) + \log p(\boldsymbol{x}_n|\boldsymbol{z}_n, \theta)]$$

Why Estimation is Difficult in LVMs?

- Suppose we want to estimate parameters Θ via MLE. If we knew both $\boldsymbol{x}_n$ and $\boldsymbol{z}_n$ then we could do

$$\Theta_{MLE} \quad = \quad \arg\max_{\Theta} \sum_{n=1}^{N} \log p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta) = \arg\max_{\Theta} \sum_{n=1}^{N} [\log p(\boldsymbol{z}_n|\phi) + \log p(\boldsymbol{x}_n|\boldsymbol{z}_n, \theta)]$$

- Simple to solve (usually closed form) if $p(\boldsymbol{z}_n|\phi)$ and $p(\boldsymbol{x}_n|\boldsymbol{z}_n, \theta)$ are "simple" (e.g., exp-fam. dist.)

Why Estimation is Difficult in LVMs?

- Suppose we want to estimate parameters Θ via MLE. If we knew both $\boldsymbol{x}_n$ and $\boldsymbol{z}_n$ then we could do

$$\Theta_{MLE} = \arg\max_{\Theta} \sum_{n=1}^{N} \log p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta) = \arg\max_{\Theta} \sum_{n=1}^{N} [\log p(\boldsymbol{z}_n|\phi) + \log p(\boldsymbol{x}_n|\boldsymbol{z}_n, \theta)]$$

- Simple to solve (usually closed form) if $p(\boldsymbol{z}_n|\phi)$ and $p(\boldsymbol{x}_n|\boldsymbol{z}_n, \theta)$ are "simple" (e.g., exp-fam. dist.)

- However, in LVMs where $\boldsymbol{z}_n$ is "hidden", the MLE problem will be the following

$$\Theta_{MLE} = \arg\max_{\Theta} \sum_{n=1}^{N} \log p(\boldsymbol{x}_n|\Theta)$$

Why Estimation is Difficult in LVMs?

- Suppose we want to estimate parameters Θ via MLE. If we knew both $\boldsymbol{x}_n$ and $\boldsymbol{z}_n$ then we could do

$$\Theta_{MLE} = \arg\max_{\Theta} \sum_{n=1}^{N} \log p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta) = \arg\max_{\Theta} \sum_{n=1}^{N} [\log p(\boldsymbol{z}_n|\phi) + \log p(\boldsymbol{x}_n|\boldsymbol{z}_n, \theta)]$$

- Simple to solve (usually closed form) if $p(\boldsymbol{z}_n|\phi)$ and $p(\boldsymbol{x}_n|\boldsymbol{z}_n, \theta)$ are "simple" (e.g., exp-fam. dist.)
- However, in LVMs where $\boldsymbol{z}_n$ is "hidden", the MLE problem will be the following

$$\Theta_{MLE} = \arg\max_{\Theta} \sum_{n=1}^{N} \log p(\boldsymbol{x}_n|\Theta) = \arg\max_{\Theta} \log p(\mathbf{X}|\Theta)$$

Why Estimation is Difficult in LVMs?

- Suppose we want to estimate parameters Θ via MLE. If we knew both $\boldsymbol{x}_n$ and $\boldsymbol{z}_n$ then we could do

$$\Theta_{MLE} = \arg\max_{\Theta} \sum_{n=1}^{N} \log p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta) = \arg\max_{\Theta} \sum_{n=1}^{N} [\log p(\boldsymbol{z}_n|\phi) + \log p(\boldsymbol{x}_n|\boldsymbol{z}_n, \theta)]$$

- Simple to solve (usually closed form) if $p(\boldsymbol{z}_n|\phi)$ and $p(\boldsymbol{x}_n|\boldsymbol{z}_n, \theta)$ are "simple" (e.g., exp-fam. dist.)
- However, in LVMs where $\boldsymbol{z}_n$ is "hidden", the MLE problem will be the following

$$\Theta_{MLE} = \arg\max_{\Theta} \sum_{n=1}^{N} \log p(\boldsymbol{x}_n|\Theta) = \arg\max_{\Theta} \log p(\mathbf{X}|\Theta)$$

- The form of $p(\boldsymbol{x}_n|\Theta)$ may not be simple since we need to sum over unknown $\boldsymbol{z}_n$'s possible values

Why Estimation is Difficult in LVMs?

- Suppose we want to estimate parameters Θ via MLE. If we knew both $\boldsymbol{x}_n$ and $\boldsymbol{z}_n$ then we could do

$$\Theta_{MLE} = \arg\max_{\Theta} \sum_{n=1}^{N} \log p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta) = \arg\max_{\Theta} \sum_{n=1}^{N} [\log p(\boldsymbol{z}_n|\phi) + \log p(\boldsymbol{x}_n|\boldsymbol{z}_n, \theta)]$$

- Simple to solve (usually closed form) if $p(\boldsymbol{z}_n|\phi)$ and $p(\boldsymbol{x}_n|\boldsymbol{z}_n, \theta)$ are "simple" (e.g., exp-fam. dist.)
- However, in LVMs where $\boldsymbol{z}_n$ is "hidden", the MLE problem will be the following

$$\Theta_{MLE} = \arg\max_{\Theta} \sum_{n=1}^{N} \log p(\boldsymbol{x}_n|\Theta) = \arg\max_{\Theta} \log p(\mathbf{X}|\Theta)$$

- The form of $p(\boldsymbol{x}_n|\Theta)$ may not be simple since we need to sum over unknown $\boldsymbol{z}_n$'s possible values

$$p(\boldsymbol{x}_n|\Theta) = \sum_{\boldsymbol{z}_n} p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta)$$

Why Estimation is Difficult in LVMs?

- Suppose we want to estimate parameters Θ via MLE. If we knew both $\boldsymbol{x}_n$ and $\boldsymbol{z}_n$ then we could do

$$\Theta_{MLE} = \arg\max_{\Theta} \sum_{n=1}^{N} \log p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta) = \arg\max_{\Theta} \sum_{n=1}^{N} [\log p(\boldsymbol{z}_n|\phi) + \log p(\boldsymbol{x}_n|\boldsymbol{z}_n, \theta)]$$

- Simple to solve (usually closed form) if $p(\boldsymbol{z}_n|\phi)$ and $p(\boldsymbol{x}_n|\boldsymbol{z}_n, \theta)$ are "simple" (e.g., exp-fam. dist.)
- However, in LVMs where $\boldsymbol{z}_n$ is "hidden", the MLE problem will be the following

$$\Theta_{MLE} = \arg\max_{\Theta} \sum_{n=1}^{N} \log p(\boldsymbol{x}_n|\Theta) = \arg\max_{\Theta} \log p(\mathbf{X}|\Theta)$$

- The form of $p(\boldsymbol{x}_n|\Theta)$ may not be simple since we need to sum over unknown $\boldsymbol{z}_n$'s possible values

$$p(\boldsymbol{x}_n|\Theta) = \sum_{\boldsymbol{z}_n} p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta) \quad \text{... or if } \boldsymbol{z}_n \text{ is continuous:} \quad p(\boldsymbol{x}_n|\Theta) = \int p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta) d\boldsymbol{z}_n$$

Why Estimation is Difficult in LVMs?

- Suppose we want to estimate parameters Θ via MLE. If we knew both $\boldsymbol{x}_n$ and $\boldsymbol{z}_n$ then we could do

$$\Theta_{MLE} = \arg\max_{\Theta} \sum_{n=1}^{N} \log p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta) = \arg\max_{\Theta} \sum_{n=1}^{N} [\log p(\boldsymbol{z}_n|\phi) + \log p(\boldsymbol{x}_n|\boldsymbol{z}_n, \theta)]$$

- Simple to solve (usually closed form) if $p(\boldsymbol{z}_n|\phi)$ and $p(\boldsymbol{x}_n|\boldsymbol{z}_n, \theta)$ are "simple" (e.g., exp-fam. dist.)
- However, in LVMs where $\boldsymbol{z}_n$ is "hidden", the MLE problem will be the following

$$\Theta_{MLE} = \arg\max_{\Theta} \sum_{n=1}^{N} \log p(\boldsymbol{x}_n|\Theta) = \arg\max_{\Theta} \log p(\mathbf{X}|\Theta)$$

- The form of $p(\boldsymbol{x}_n|\Theta)$ may not be simple since we need to sum over unknown $\boldsymbol{z}_n$'s possible values

$$p(\boldsymbol{x}_n|\Theta) = \sum_{\boldsymbol{z}_n} p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta) \quad \text{... or if } \boldsymbol{z}_n \text{ is continuous:} \quad p(\boldsymbol{x}_n|\Theta) = \int p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta) d\boldsymbol{z}_n$$

- The summation/integral may be intractable

Why Estimation is Difficult in LVMs?

- Suppose we want to estimate parameters Θ via MLE. If we knew both $\boldsymbol{x}_n$ and $\boldsymbol{z}_n$ then we could do

$$\Theta_{MLE} = \arg\max_{\Theta} \sum_{n=1}^{N} \log p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta) = \arg\max_{\Theta} \sum_{n=1}^{N} [\log p(\boldsymbol{z}_n|\phi) + \log p(\boldsymbol{x}_n|\boldsymbol{z}_n, \theta)]$$

- Simple to solve (usually closed form) if $p(\boldsymbol{z}_n|\phi)$ and $p(\boldsymbol{x}_n|\boldsymbol{z}_n, \theta)$ are "simple" (e.g., exp-fam. dist.)
- However, in LVMs where $\boldsymbol{z}_n$ is "hidden", the MLE problem will be the following

$$\Theta_{MLE} = \arg\max_{\Theta} \sum_{n=1}^{N} \log p(\boldsymbol{x}_n|\Theta) = \arg\max_{\Theta} \log p(\mathbf{X}|\Theta)$$

- The form of $p(\boldsymbol{x}_n|\Theta)$ may not be simple since we need to sum over unknown $\boldsymbol{z}_n$'s possible values

$$p(\boldsymbol{x}_n|\Theta) = \sum_{\boldsymbol{z}_n} p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta) \quad \text{... or if } \boldsymbol{z}_n \text{ is continuous:} \quad p(\boldsymbol{x}_n|\Theta) = \int p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta) d\boldsymbol{z}_n$$

- The summation/integral may be intractable + may lead to complex expressions for $p(\boldsymbol{x}_n|\Theta)$

Why Estimation is Difficult in LVMs?

- Suppose we want to estimate parameters Θ via MLE. If we knew both $\boldsymbol{x}_n$ and $\boldsymbol{z}_n$ then we could do

$$\Theta_{MLE} = \arg\max_{\Theta} \sum_{n=1}^{N} \log p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta) = \arg\max_{\Theta} \sum_{n=1}^{N} [\log p(\boldsymbol{z}_n|\phi) + \log p(\boldsymbol{x}_n|\boldsymbol{z}_n, \theta)]$$

- Simple to solve (usually closed form) if $p(\boldsymbol{z}_n|\phi)$ and $p(\boldsymbol{x}_n|\boldsymbol{z}_n, \theta)$ are "simple" (e.g., exp-fam. dist.)
- However, in LVMs where $\boldsymbol{z}_n$ is "hidden", the MLE problem will be the following

$$\Theta_{MLE} = \arg\max_{\Theta} \sum_{n=1}^{N} \log p(\boldsymbol{x}_n|\Theta) = \arg\max_{\Theta} \log p(\mathbf{X}|\Theta)$$

- The form of $p(\boldsymbol{x}_n|\Theta)$ may not be simple since we need to sum over unknown $\boldsymbol{z}_n$'s possible values

$$p(\boldsymbol{x}_n|\Theta) = \sum_{\boldsymbol{z}_n} p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta) \quad \text{... or if } \boldsymbol{z}_n \text{ is continuous:} \quad p(\boldsymbol{x}_n|\Theta) = \int p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta) d\boldsymbol{z}_n$$

- The summation/integral may be intractable + may lead to complex expressions for $p(\boldsymbol{x}_n|\Theta)$, in fact almost never an exponential family distribution

Why Estimation is Difficult in LVMs?

- Suppose we want to estimate parameters Θ via MLE. If we knew both $\boldsymbol{x}_n$ and $\boldsymbol{z}_n$ then we could do

$$\Theta_{MLE} = \arg\max_{\Theta} \sum_{n=1}^{N} \log p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta) = \arg\max_{\Theta} \sum_{n=1}^{N} [\log p(\boldsymbol{z}_n|\phi) + \log p(\boldsymbol{x}_n|\boldsymbol{z}_n, \theta)]$$

- Simple to solve (usually closed form) if $p(\boldsymbol{z}_n|\phi)$ and $p(\boldsymbol{x}_n|\boldsymbol{z}_n, \theta)$ are "simple" (e.g., exp-fam. dist.)
- However, in LVMs where $\boldsymbol{z}_n$ is "hidden", the MLE problem will be the following

$$\Theta_{MLE} = \arg\max_{\Theta} \sum_{n=1}^{N} \log p(\boldsymbol{x}_n|\Theta) = \arg\max_{\Theta} \log p(\mathbf{X}|\Theta)$$

- The form of $p(\boldsymbol{x}_n|\Theta)$ may not be simple since we need to sum over unknown $\boldsymbol{z}_n$'s possible values

$$p(\boldsymbol{x}_n|\Theta) = \sum_{\boldsymbol{z}_n} p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta) \quad \text{... or if } \boldsymbol{z}_n \text{ is continuous:} \quad p(\boldsymbol{x}_n|\Theta) = \int p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta) d\boldsymbol{z}_n$$

- The summation/integral may be intractable + may lead to complex expressions for $p(\boldsymbol{x}_n|\Theta)$, in fact almost never an exponential family distribution. MLE for Θ won't have closed form solutions!

An Important Identity

- Define $p_z = p(\mathbf{Z}|\mathbf{X}, \Theta)$ and let $q(\mathbf{Z})$ be some distribution over $\mathbf{Z}$

An Important Identity

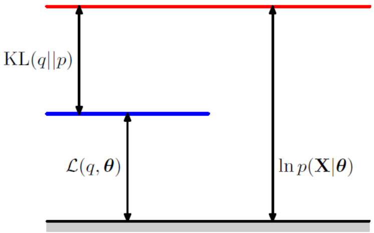
- Define $p_z = p(\mathbf{Z}|\mathbf{X}, \Theta)$ and let $q(\mathbf{Z})$ be some distribution over $\mathbf{Z}$
- Assume discrete $\mathbf{Z}$, the identity below holds for any choice of the distribution $q(\mathbf{Z})$

$$\boxed{\log p(\mathbf{X}|\Theta) = \mathcal{L}(q, \Theta) + \mathrm{KL}(q||p_z)}$$

$$\mathcal{L}(q, \Theta) = \sum_{\mathbf{Z}} q(\mathbf{Z}) \log \left\{ \frac{p(\mathbf{X}, \mathbf{Z}|\Theta)}{q(\mathbf{Z})} \right\}$$

$$\mathrm{KL}(q||p_z) = -\sum_{\mathbf{Z}} q(\mathbf{Z}) \log \left\{ \frac{p(\mathbf{Z}|\mathbf{X}, \Theta)}{q(\mathbf{Z})} \right\}$$

(Exercise: Verify the above identity)



An Important Identity

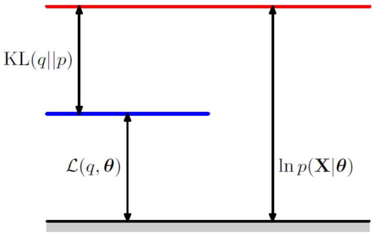
- Define $p_z = p(\mathbf{Z}|\mathbf{X}, \Theta)$ and let $q(\mathbf{Z})$ be some distribution over $\mathbf{Z}$
- Assume discrete $\mathbf{Z}$, the identity below holds for any choice of the distribution $q(\mathbf{Z})$

$$\boxed{\log p(\mathbf{X}|\Theta) = \mathcal{L}(q, \Theta) + \text{KL}(q||p_z)}$$

$$\mathcal{L}(q, \Theta) = \sum_{\mathbf{Z}} q(\mathbf{Z}) \log \left\{ \frac{p(\mathbf{X}, \mathbf{Z}|\Theta)}{q(\mathbf{Z})} \right\}$$

$$\text{KL}(q||p_z) = -\sum_{\mathbf{Z}} q(\mathbf{Z}) \log \left\{ \frac{p(\mathbf{Z}|\mathbf{X}, \Theta)}{q(\mathbf{Z})} \right\}$$

(Exercise: Verify the above identity)



- Since $\text{KL}(q||p_z) \geq 0$, $\mathcal{L}(q, \Theta)$ is a lower-bound on $\log p(\mathbf{X}|\Theta)$

An Important Identity

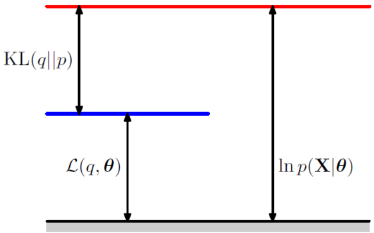
- Define $p_z = p(\mathbf{Z}|\mathbf{X}, \Theta)$ and let $q(\mathbf{Z})$ be some distribution over $\mathbf{Z}$
- Assume discrete $\mathbf{Z}$, the identity below holds for any choice of the distribution $q(\mathbf{Z})$

$$\boxed{\log p(\mathbf{X}|\Theta) = \mathcal{L}(q, \Theta) + \text{KL}(q||p_z)}$$

$$\mathcal{L}(q, \Theta) = \sum_{\mathbf{Z}} q(\mathbf{Z}) \log \left\{ \frac{p(\mathbf{X}, \mathbf{Z}|\Theta)}{q(\mathbf{Z})} \right\}$$

$$\text{KL}(q||p_z) = -\sum_{\mathbf{Z}} q(\mathbf{Z}) \log \left\{ \frac{p(\mathbf{Z}|\mathbf{X}, \Theta)}{q(\mathbf{Z})} \right\}$$

(Exercise: Verify the above identity)



- Since $\text{KL}(q||p_z) \geq 0$, $\mathcal{L}(q, \Theta)$ is a lower-bound on $\log p(\mathbf{X}|\Theta)$

$$\log p(\mathbf{X}|\Theta) \geq \mathcal{L}(q, \Theta)$$

An Important Identity

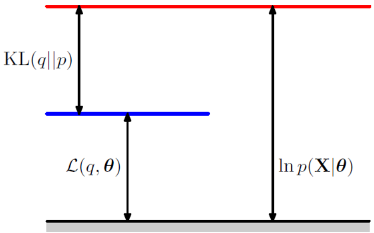
- Define $p_z = p(\mathbf{Z}|\mathbf{X}, \Theta)$ and let $q(\mathbf{Z})$ be some distribution over $\mathbf{Z}$
- Assume discrete $\mathbf{Z}$, the identity below holds for any choice of the distribution $q(\mathbf{Z})$

$$\boxed{\log p(\mathbf{X}|\Theta) = \mathcal{L}(q, \Theta) + \mathrm{KL}(q||p_z)}$$

$$\mathcal{L}(q, \Theta) = \sum_{\mathbf{Z}} q(\mathbf{Z}) \log \left\{ \frac{p(\mathbf{X}, \mathbf{Z}|\Theta)}{q(\mathbf{Z})} \right\}$$

$$\mathrm{KL}(q||p_z) = -\sum_{\mathbf{Z}} q(\mathbf{Z}) \log \left\{ \frac{p(\mathbf{Z}|\mathbf{X}, \Theta)}{q(\mathbf{Z})} \right\}$$

(Exercise: Verify the above identity)



- Since $\mathrm{KL}(q||p_z) \geq 0$, $\mathcal{L}(q, \Theta)$ is a lower-bound on $\log p(\mathbf{X}|\Theta)$

$$\log p(\mathbf{X}|\Theta) \geq \mathcal{L}(q, \Theta)$$

- Maximizing $\mathcal{L}(q, \Theta)$ will also improve $\log p(\mathbf{X}|\Theta)$

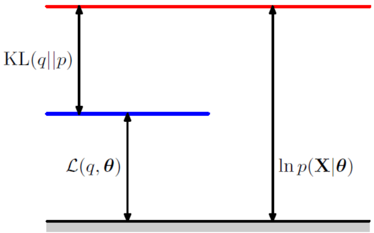
An Important Identity

- Define $p_z = p(\mathbf{Z}|\mathbf{X},\Theta)$ and let $q(\mathbf{Z})$ be some distribution over $\mathbf{Z}$
- Assume discrete $\mathbf{Z}$, the identity below holds for any choice of the distribution $q(\mathbf{Z})$

$$\boxed{\log p(\mathbf{X}|\Theta) = \mathcal{L}(q,\Theta) + \mathrm{KL}(q||p_z)}$$

$$\begin{aligned} \mathcal{L}(q,\Theta) &= \sum_{\mathbf{Z}} q(\mathbf{Z}) \log\left\{\frac{p(\mathbf{X},\mathbf{Z}|\Theta)}{q(\mathbf{Z})}\right\} \\ \mathrm{KL}(q||p_z) &= -\sum_{\mathbf{Z}} q(\mathbf{Z}) \log\left\{\frac{p(\mathbf{Z}|\mathbf{X},\Theta)}{q(\mathbf{Z})}\right\} \end{aligned}$$

(Exercise: Verify the above identity)



- Since $\mathrm{KL}(q||p_z) \geq 0$, $\mathcal{L}(q,\Theta)$ is a lower-bound on $\log p(\mathbf{X}|\Theta)$

$$\log p(\mathbf{X}|\Theta) \geq \mathcal{L}(q,\Theta)$$

- Maximizing $\mathcal{L}(q,\Theta)$ will also improve $\log p(\mathbf{X}|\Theta)$. Also, as we'll see, it's easier to maximize $\mathcal{L}(q,\Theta)$

Maximizing $\mathcal{L}(q, \Theta)$

- Note that $\mathcal{L}(q, \Theta)$ depends on two things $q(\mathbf{Z})$ and Θ. Let's do ALT-OPT for these

Maximizing $\mathcal{L}(q, \Theta)$

- Note that $\mathcal{L}(q, \Theta)$ depends on two things $q(\mathbf{Z})$ and Θ. Let's do ALT-OPT for these
- First recall the identity we had: $\log p(\mathbf{X}|\Theta) = \mathcal{L}(q, \Theta) + \text{KL}(q||p_z)$ with

$$\mathcal{L}(q, \Theta) = \sum_{\mathbf{Z}} q(\mathbf{Z}) \log \left\{ \frac{p(\mathbf{X}, \mathbf{Z}|\Theta)}{q(\mathbf{Z})} \right\}$$

Maximizing $\mathcal{L}(q, \Theta)$

- Note that $\mathcal{L}(q, \Theta)$ depends on two things $q(\mathbf{Z})$ and Θ. Let's do ALT-OPT for these
- First recall the identity we had: $\log p(\mathbf{X}|\Theta) = \mathcal{L}(q, \Theta) + \text{KL}(q||p_z)$ with

$$\mathcal{L}(q, \Theta) = \sum_{\mathbf{Z}} q(\mathbf{Z}) \log \left\{ \frac{p(\mathbf{X}, \mathbf{Z}|\Theta)}{q(\mathbf{Z})} \right\} \quad \text{and} \quad \text{KL}(q||p_z) = -\sum_{\mathbf{Z}} q(\mathbf{Z}) \log \left\{ \frac{p(\mathbf{Z}|\mathbf{X}, \Theta)}{q(\mathbf{Z})} \right\}$$

Maximizing $\mathcal{L}(q, \Theta)$

- Note that $\mathcal{L}(q, \Theta)$ depends on two things $q(\mathbf{Z})$ and Θ. Let's do ALT-OPT for these
- First recall the identity we had: $\log p(\mathbf{X}|\Theta) = \mathcal{L}(q, \Theta) + \text{KL}(q||p_z)$ with

$$\mathcal{L}(q, \Theta) = \sum_{\mathbf{Z}} q(\mathbf{Z}) \log \left\{ \frac{p(\mathbf{X}, \mathbf{Z}|\Theta)}{q(\mathbf{Z})} \right\} \quad \text{and} \quad \text{KL}(q||p_z) = -\sum_{\mathbf{Z}} q(\mathbf{Z}) \log \left\{ \frac{p(\mathbf{Z}|\mathbf{X}, \Theta)}{q(\mathbf{Z})} \right\}$$

- Maximize $\mathcal{L}$ w.r.t. q with Θ fixed at Θ^{old}: Since $\log p(\mathbf{X}|\Theta)$ will be a constant in this case,

$$\hat{q} = \arg\max_q \mathcal{L}(q, \Theta^{old})$$

Maximizing $\mathcal{L}(q, \Theta)$

- Note that $\mathcal{L}(q, \Theta)$ depends on two things $q(\mathbf{Z})$ and Θ. Let's do ALT-OPT for these
- First recall the identity we had: $\log p(\mathbf{X}|\Theta) = \mathcal{L}(q, \Theta) + \text{KL}(q||p_z)$ with

$$\mathcal{L}(q, \Theta) = \sum_{\mathbf{Z}} q(\mathbf{Z}) \log \left\{ \frac{p(\mathbf{X}, \mathbf{Z}|\Theta)}{q(\mathbf{Z})} \right\} \quad \text{and} \quad \text{KL}(q||p_z) = -\sum_{\mathbf{Z}} q(\mathbf{Z}) \log \left\{ \frac{p(\mathbf{Z}|\mathbf{X}, \Theta)}{q(\mathbf{Z})} \right\}$$

- Maximize $\mathcal{L}$ w.r.t. q with Θ fixed at Θ^{old}: Since $\log p(\mathbf{X}|\Theta)$ will be a constant in this case,

$$\hat{q} = \arg\max_q \mathcal{L}(q, \Theta^{old}) = \arg\min_q \text{KL}(q||p_z)$$

Maximizing $\mathcal{L}(q, \Theta)$

- Note that $\mathcal{L}(q, \Theta)$ depends on two things $q(\mathbf{Z})$ and Θ. Let's do ALT-OPT for these
- First recall the identity we had: $\log p(\mathbf{X}|\Theta) = \mathcal{L}(q, \Theta) + \text{KL}(q||p_z)$ with

$$\mathcal{L}(q, \Theta) = \sum_{\mathbf{Z}} q(\mathbf{Z}) \log \left\{ \frac{p(\mathbf{X}, \mathbf{Z}|\Theta)}{q(\mathbf{Z})} \right\} \quad \text{and} \quad \text{KL}(q||p_z) = -\sum_{\mathbf{Z}} q(\mathbf{Z}) \log \left\{ \frac{p(\mathbf{Z}|\mathbf{X}, \Theta)}{q(\mathbf{Z})} \right\}$$

- Maximize $\mathcal{L}$ w.r.t. q with Θ fixed at Θ^{old}: Since $\log p(\mathbf{X}|\Theta)$ will be a constant in this case,

$$\hat{q} = \arg\max_q \mathcal{L}(q, \Theta^{old}) = \arg\min_q \text{KL}(q||p_z) = p_z = p(\mathbf{Z}|\mathbf{X}, \Theta^{old})$$

Maximizing $\mathcal{L}(q, \Theta)$

- Note that $\mathcal{L}(q, \Theta)$ depends on two things $q(\mathbf{Z})$ and Θ. Let's do ALT-OPT for these
- First recall the identity we had: $\log p(\mathbf{X}|\Theta) = \mathcal{L}(q, \Theta) + \text{KL}(q||p_z)$ with

$$\mathcal{L}(q, \Theta) = \sum_{\mathbf{Z}} q(\mathbf{Z}) \log \left\{ \frac{p(\mathbf{X}, \mathbf{Z}|\Theta)}{q(\mathbf{Z})} \right\} \quad \text{and} \quad \text{KL}(q||p_z) = -\sum_{\mathbf{Z}} q(\mathbf{Z}) \log \left\{ \frac{p(\mathbf{Z}|\mathbf{X}, \Theta)}{q(\mathbf{Z})} \right\}$$

- Maximize $\mathcal{L}$ w.r.t. q with Θ fixed at Θ^{old}: Since $\log p(\mathbf{X}|\Theta)$ will be a constant in this case,

$$\hat{q} = \arg\max_q \mathcal{L}(q, \Theta^{old}) = \arg\min_q \text{KL}(q||p_z) = p_z = p(\mathbf{Z}|\mathbf{X}, \Theta^{old})$$

- Maximize $\mathcal{L}$ w.r.t. Θ with q fixed at $\hat{q} = p(\mathbf{Z}|\mathbf{X}, \Theta^{old})$

$$\Theta^{new} = \arg\max_\Theta \mathcal{L}(\hat{q}, \Theta)$$

Maximizing $\mathcal{L}(q, \Theta)$

- Note that $\mathcal{L}(q, \Theta)$ depends on two things $q(\mathbf{Z})$ and Θ. Let's do ALT-OPT for these
- First recall the identity we had: $\log p(\mathbf{X}|\Theta) = \mathcal{L}(q, \Theta) + \text{KL}(q||p_z)$ with

$$\mathcal{L}(q, \Theta) = \sum_{\mathbf{Z}} q(\mathbf{Z}) \log \left\{ \frac{p(\mathbf{X}, \mathbf{Z}|\Theta)}{q(\mathbf{Z})} \right\} \quad \text{and} \quad \text{KL}(q||p_z) = -\sum_{\mathbf{Z}} q(\mathbf{Z}) \log \left\{ \frac{p(\mathbf{Z}|\mathbf{X}, \Theta)}{q(\mathbf{Z})} \right\}$$

- Maximize $\mathcal{L}$ w.r.t. q with Θ fixed at Θ^{old}: Since $\log p(\mathbf{X}|\Theta)$ will be a constant in this case,

$$\hat{q} = \arg\max_q \mathcal{L}(q, \Theta^{old}) = \arg\min_q \text{KL}(q||p_z) = p_z = p(\mathbf{Z}|\mathbf{X}, \Theta^{old})$$

- Maximize $\mathcal{L}$ w.r.t. Θ with q fixed at $\hat{q} = p(\mathbf{Z}|\mathbf{X}, \Theta^{old})$

$$\Theta^{new} = \arg\max_\Theta \mathcal{L}(\hat{q}, \Theta) = \arg\max_\Theta \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}, \Theta^{old}) \log \frac{p(\mathbf{X}, \mathbf{Z}|\Theta)}{p(\mathbf{Z}|\mathbf{X}, \Theta^{old})}$$

Maximizing $\mathcal{L}(q, \Theta)$

- Note that $\mathcal{L}(q, \Theta)$ depends on two things $q(\mathbf{Z})$ and Θ. Let's do ALT-OPT for these
- First recall the identity we had: $\log p(\mathbf{X}|\Theta) = \mathcal{L}(q, \Theta) + \text{KL}(q||p_z)$ with

$$\mathcal{L}(q, \Theta) = \sum_{\mathbf{Z}} q(\mathbf{Z}) \log \left\{ \frac{p(\mathbf{X}, \mathbf{Z}|\Theta)}{q(\mathbf{Z})} \right\} \quad \text{and} \quad \text{KL}(q||p_z) = -\sum_{\mathbf{Z}} q(\mathbf{Z}) \log \left\{ \frac{p(\mathbf{Z}|\mathbf{X}, \Theta)}{q(\mathbf{Z})} \right\}$$

- Maximize $\mathcal{L}$ w.r.t. q with Θ fixed at Θ^{old}: Since $\log p(\mathbf{X}|\Theta)$ will be a constant in this case,

$$\hat{q} = \arg\max_q \mathcal{L}(q, \Theta^{old}) = \arg\min_q \text{KL}(q||p_z) = p_z = p(\mathbf{Z}|\mathbf{X}, \Theta^{old})$$

- Maximize $\mathcal{L}$ w.r.t. Θ with q fixed at $\hat{q} = p(\mathbf{Z}|\mathbf{X}, \Theta^{old})$

$$\Theta^{new} = \arg\max_\Theta \mathcal{L}(\hat{q}, \Theta) = \arg\max_\Theta \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}, \Theta^{old}) \log \frac{p(\mathbf{X}, \mathbf{Z}|\Theta)}{p(\mathbf{Z}|\mathbf{X}, \Theta^{old})} = \arg\max_\Theta \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}, \Theta^{old}) \log p(\mathbf{X}, \mathbf{Z}|\Theta)$$

Maximizing $\mathcal{L}(q, \Theta)$

- Note that $\mathcal{L}(q, \Theta)$ depends on two things $q(\mathbf{Z})$ and Θ. Let's do ALT-OPT for these
- First recall the identity we had: $\log p(\mathbf{X}|\Theta) = \mathcal{L}(q, \Theta) + \text{KL}(q||p_z)$ with

$$\mathcal{L}(q, \Theta) = \sum_{\mathbf{Z}} q(\mathbf{Z}) \log \left\{ \frac{p(\mathbf{X}, \mathbf{Z}|\Theta)}{q(\mathbf{Z})} \right\} \quad \text{and} \quad \text{KL}(q||p_z) = -\sum_{\mathbf{Z}} q(\mathbf{Z}) \log \left\{ \frac{p(\mathbf{Z}|\mathbf{X}, \Theta)}{q(\mathbf{Z})} \right\}$$

- Maximize $\mathcal{L}$ w.r.t. q with Θ fixed at Θ^{old}: Since $\log p(\mathbf{X}|\Theta)$ will be a constant in this case,

$$\hat{q} = \arg\max_q \mathcal{L}(q, \Theta^{old}) = \arg\min_q \text{KL}(q||p_z) = p_z = p(\mathbf{Z}|\mathbf{X}, \Theta^{old})$$

- Maximize $\mathcal{L}$ w.r.t. Θ with q fixed at $\hat{q} = p(\mathbf{Z}|\mathbf{X}, \Theta^{old})$

$$\Theta^{new} = \arg\max_\Theta \mathcal{L}(\hat{q}, \Theta) = \arg\max_\Theta \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}, \Theta^{old}) \log \frac{p(\mathbf{X}, \mathbf{Z}|\Theta)}{p(\mathbf{Z}|\mathbf{X}, \Theta^{old})} = \arg\max_\Theta \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}, \Theta^{old}) \log p(\mathbf{X}, \mathbf{Z}|\Theta)$$

.. therefore, $\boxed{\Theta^{new} = \arg\max_\theta \mathcal{Q}(\Theta, \Theta^{old})}$ where $\mathcal{Q}(\Theta, \Theta^{old}) = \mathbb{E}_{p(\mathbf{Z}|\mathbf{X}, \Theta^{old})}[\log p(\mathbf{X}, \mathbf{Z}|\Theta)]$

Maximizing $\mathcal{L}(q, \Theta)$

- Note that $\mathcal{L}(q, \Theta)$ depends on two things $q(\mathbf{Z})$ and Θ. Let's do ALT-OPT for these
- First recall the identity we had: $\log p(\mathbf{X}|\Theta) = \mathcal{L}(q, \Theta) + \text{KL}(q||p_z)$ with

$$\mathcal{L}(q, \Theta) = \sum_{\mathbf{Z}} q(\mathbf{Z}) \log \left\{ \frac{p(\mathbf{X}, \mathbf{Z}|\Theta)}{q(\mathbf{Z})} \right\} \quad \text{and} \quad \text{KL}(q||p_z) = -\sum_{\mathbf{Z}} q(\mathbf{Z}) \log \left\{ \frac{p(\mathbf{Z}|\mathbf{X}, \Theta)}{q(\mathbf{Z})} \right\}$$

- Maximize $\mathcal{L}$ w.r.t. q with Θ fixed at Θ^{old}: Since $\log p(\mathbf{X}|\Theta)$ will be a constant in this case,

$$\hat{q} = \arg\max_q \mathcal{L}(q, \Theta^{old}) = \arg\min_q \text{KL}(q||p_z) = p_z = p(\mathbf{Z}|\mathbf{X}, \Theta^{old})$$

- Maximize $\mathcal{L}$ w.r.t. Θ with q fixed at $\hat{q} = p(\mathbf{Z}|\mathbf{X}, \Theta^{old})$

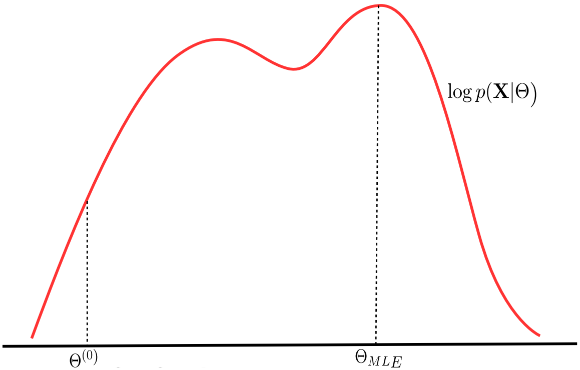
$$\Theta^{new} = \arg\max_\Theta \mathcal{L}(\hat{q}, \Theta) = \arg\max_\Theta \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}, \Theta^{old}) \log \frac{p(\mathbf{X}, \mathbf{Z}|\Theta)}{p(\mathbf{Z}|\mathbf{X}, \Theta^{old})} = \arg\max_\Theta \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}, \Theta^{old}) \log p(\mathbf{X}, \mathbf{Z}|\Theta)$$

.. therefore, $\boxed{\Theta^{new} = \arg\max_\theta \mathcal{Q}(\Theta, \Theta^{old})}$ where $\mathcal{Q}(\Theta, \Theta^{old}) = \mathbb{E}_{p(\mathbf{Z}|\mathbf{X}, \Theta^{old})}[\log p(\mathbf{X}, \mathbf{Z}|\Theta)]$

- $\mathcal{Q}(\Theta, \Theta^{old}) = \mathbb{E}_{p(\mathbf{Z}|\mathbf{X}, \Theta^{old})}[\log p(\mathbf{X}, \mathbf{Z}|\Theta)]$ is known as expected complete data log-likelihood (CLL)

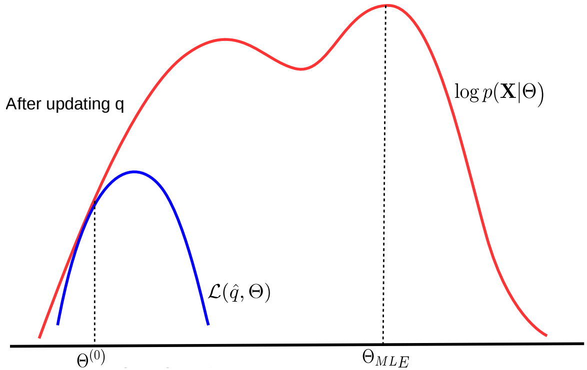
What's Going On: A Visual Illustration..

- Step 1: We set $\hat{q} = p(\mathbf{Z}|\mathbf{X}, \Theta^{old})$, $\mathcal{L}(\hat{q}, \Theta)$ touches $\log p(\mathbf{X}|\Theta)$ at Θ^{old}
- Step 2: We maximize $\mathcal{L}(\hat{q}, \Theta)$ w.r.t. Θ (equivalent to maximizing $\mathcal{Q}(\Theta, \Theta^{old})$)



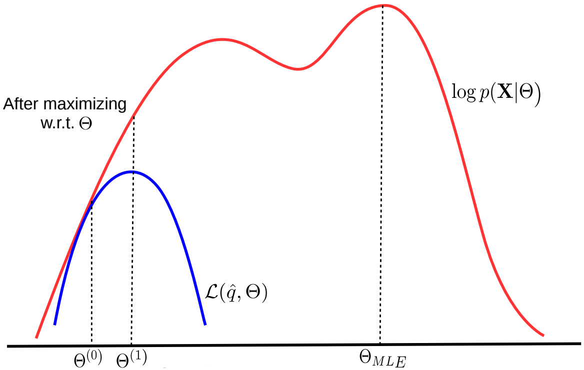
What's Going On: A Visual Illustration..

- Step 1: We set $\hat{q} = p(\mathbf{Z}|\mathbf{X}, \Theta^{old})$, $\mathcal{L}(\hat{q}, \Theta)$ touches $\log p(\mathbf{X}|\Theta)$ at Θ^{old}
- Step 2: We maximize $\mathcal{L}(\hat{q}, \Theta)$ w.r.t. Θ (equivalent to maximizing $\mathcal{Q}(\Theta, \Theta^{old})$)



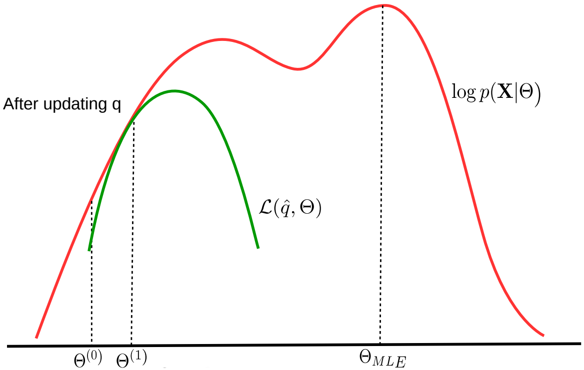
What's Going On: A Visual Illustration..

- Step 1: We set $\hat{q} = p(\mathbf{Z}|\mathbf{X}, \Theta^{old})$, $\mathcal{L}(\hat{q}, \Theta)$ touches $\log p(\mathbf{X}|\Theta)$ at Θ^{old}
- Step 2: We maximize $\mathcal{L}(\hat{q}, \Theta)$ w.r.t. Θ (equivalent to maximizing $\mathcal{Q}(\Theta, \Theta^{old})$)



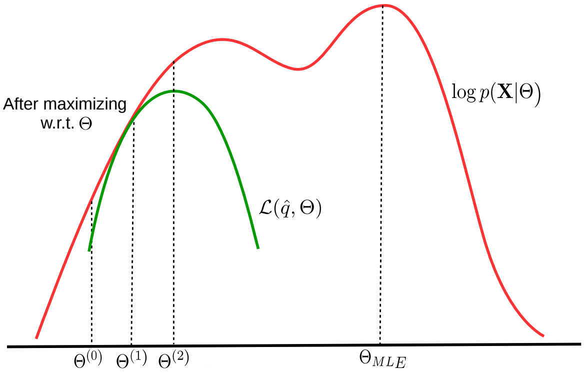
What's Going On: A Visual Illustration..

- Step 1: We set $\hat{q} = p(\mathbf{Z}|\mathbf{X}, \Theta^{old})$, $\mathcal{L}(\hat{q}, \Theta)$ touches $\log p(\mathbf{X}|\Theta)$ at Θ^{old}
- Step 2: We maximize $\mathcal{L}(\hat{q}, \Theta)$ w.r.t. Θ (equivalent to maximizing $\mathcal{Q}(\Theta, \Theta^{old})$)



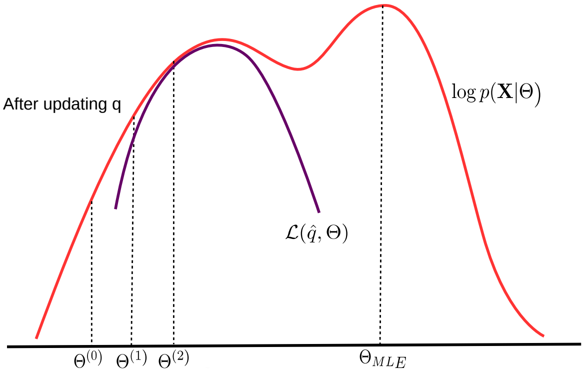
What's Going On: A Visual Illustration..

- Step 1: We set $\hat{q} = p(\mathbf{Z}|\mathbf{X}, \Theta^{old})$, $\mathcal{L}(\hat{q}, \Theta)$ touches $\log p(\mathbf{X}|\Theta)$ at Θ^{old}
- Step 2: We maximize $\mathcal{L}(\hat{q}, \Theta)$ w.r.t. Θ (equivalent to maximizing $\mathcal{Q}(\Theta, \Theta^{old})$)



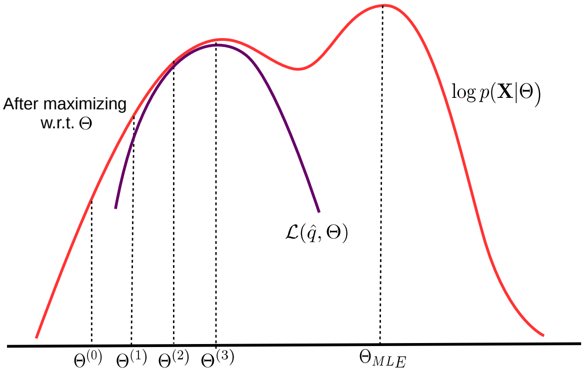
What's Going On: A Visual Illustration..

- Step 1: We set $\hat{q} = p(\mathbf{Z}|\mathbf{X}, \Theta^{old})$, $\mathcal{L}(\hat{q}, \Theta)$ touches $\log p(\mathbf{X}|\Theta)$ at Θ^{old}
- Step 2: We maximize $\mathcal{L}(\hat{q}, \Theta)$ w.r.t. Θ (equivalent to maximizing $\mathcal{Q}(\Theta, \Theta^{old})$)



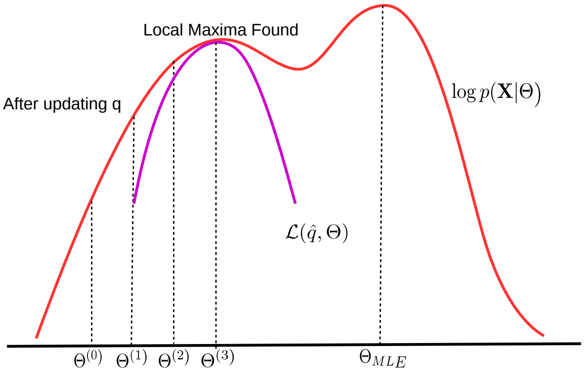
What's Going On: A Visual Illustration..

- Step 1: We set $\hat{q} = p(\mathbf{Z}|\mathbf{X}, \Theta^{old})$, $\mathcal{L}(\hat{q}, \Theta)$ touches $\log p(\mathbf{X}|\Theta)$ at Θ^{old}
- Step 2: We maximize $\mathcal{L}(\hat{q}, \Theta)$ w.r.t. Θ (equivalent to maximizing $\mathcal{Q}(\Theta, \Theta^{old})$)



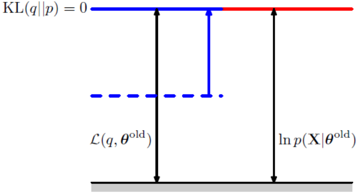
What's Going On: A Visual Illustration..

- Step 1: We set $\hat{q} = p(\mathbf{Z}|\mathbf{X}, \Theta^{old})$, $\mathcal{L}(\hat{q}, \Theta)$ touches $\log p(\mathbf{X}|\Theta)$ at Θ^{old}
- Step 2: We maximize $\mathcal{L}(\hat{q}, \Theta)$ w.r.t. Θ (equivalent to maximizing $\mathcal{Q}(\Theta, \Theta^{old})$)

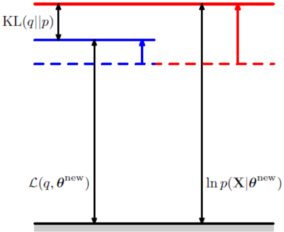


What's Going On: Another Illustration

- The two-step alternating optimzation scheme we saw can never decrease $p(\mathbf{X}|\Theta)$ (good thing)
- To see this consider both steps: (1) Optimize q given $\Theta = \Theta^{old}$; (2) Optimize Θ given this q



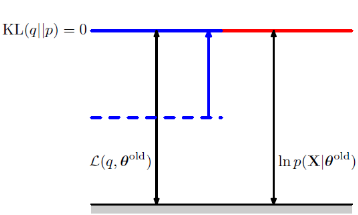
(Step 1)



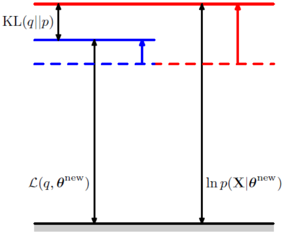
(Step 2)

What's Going On: Another Illustration

- The two-step alternating optimzation scheme we saw can never decrease $p(\mathbf{X}|\Theta)$ (good thing)
- To see this consider both steps: (1) Optimize q given $\Theta = \Theta^{old}$; (2) Optimize Θ given this q



(Step 1)

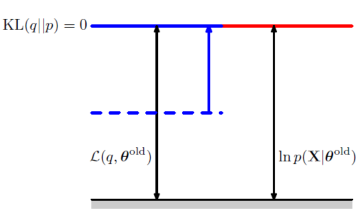


(Step 2)

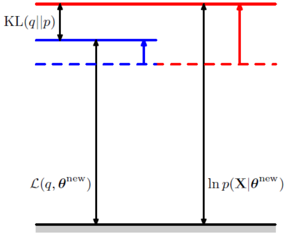
- Step 1 keeps Θ fixed, so $p(\mathbf{X}|\Theta)$ obviously can't decrease (stays unchanged in this step)

What's Going On: Another Illustration

- The two-step alternating optimzation scheme we saw can never decrease $p(\mathbf{X}|\Theta)$ (good thing)
- To see this consider both steps: (1) Optimize q given $\Theta = \Theta^{old}$; (2) Optimize Θ given this q



(Step 1)



(Step 2)

- Step 1 keeps Θ fixed, so $p(\mathbf{X}|\Theta)$ obviously can't decrease (stays unchanged in this step)
- Step 2 maximizes the lower bound $\mathcal{L}(q, \Theta)$ w.r.t Θ. Thus $p(\mathbf{X}|\Theta)$ can't decrease!

The Expectation Maximization (EM) Algorithm

Initialize the parameters: Θ^{old}. Then alternate between these steps:

- **E (Expectation) step:**

- **M (Maximization) step:**

The Expectation Maximization (EM) Algorithm

Initialize the parameters: Θ^{old}. Then alternate between these steps:

- **E (Expectation) step:**
  - Compute the posterior distribution $p(\mathbf{Z}|\mathbf{X}, \Theta^{old})$ over latent variables $\mathbf{Z}$ using Θ^{old}
- **M (Maximization) step:**

The Expectation Maximization (EM) Algorithm

Initialize the parameters: Θ^{old}. Then alternate between these steps:

- **E (Expectation) step:**
  - Compute the posterior distribution $p(\mathbf{Z}|\mathbf{X}, \Theta^{old})$ over latent variables $\mathbf{Z}$ using Θ^{old}
  - Compute the expected complete data log-likelihood w.r.t. *this* posterior distribution

$$\begin{aligned}
\mathcal{Q}(\Theta, \Theta^{old}) &= \mathbb{E}_{p(\mathbf{Z}|\mathbf{X},\Theta^{old})}[\log p(\mathbf{X}, \mathbf{Z}|\Theta)] = \sum_{n=1}^{N} \mathbb{E}_{p(\boldsymbol{z}_n|\boldsymbol{x}_n,\Theta^{old})}[\log p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta)] \\
&= \sum_{n=1}^{N} \mathbb{E}_{p(\boldsymbol{z}_n|\boldsymbol{x}_n,\Theta^{old})}[\log p(\boldsymbol{x}_n|\boldsymbol{z}_n, \Theta) + \log p(\boldsymbol{z}_n|\Theta)]
\end{aligned}$$

- **M (Maximization) step:**

The Expectation Maximization (EM) Algorithm

Initialize the parameters: Θ^{old}. Then alternate between these steps:

- **E (Expectation) step:**
  - Compute the posterior distribution $p(\mathbf{Z}|\mathbf{X}, \Theta^{old})$ over latent variables $\mathbf{Z}$ using Θ^{old}
  - Compute the expected complete data log-likelihood w.r.t. *this* posterior distribution

$$\begin{aligned}\mathcal{Q}(\Theta, \Theta^{old}) &= \mathbb{E}_{p(\mathbf{Z}|\mathbf{X},\Theta^{old})}[\log p(\mathbf{X}, \mathbf{Z}|\Theta)] = \sum_{n=1}^{N} \mathbb{E}_{p(\boldsymbol{z}_n|\boldsymbol{x}_n,\Theta^{old})}[\log p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta)] \\ &= \sum_{n=1}^{N} \mathbb{E}_{p(\boldsymbol{z}_n|\boldsymbol{x}_n,\Theta^{old})}[\log p(\boldsymbol{x}_n|\boldsymbol{z}_n, \Theta) + \log p(\boldsymbol{z}_n|\Theta)]\end{aligned}$$

- **M (Maximization) step:**

The Expectation Maximization (EM) Algorithm

Initialize the parameters: Θ^{old}. Then alternate between these steps:

- **E (Expectation) step:**
  - Compute the posterior distribution $p(\mathbf{Z}|\mathbf{X}, \Theta^{old})$ over latent variables $\mathbf{Z}$ using Θ^{old}
  - Compute the expected complete data log-likelihood w.r.t. *this* posterior distribution

$$
\begin{aligned}
\mathcal{Q}(\Theta, \Theta^{old}) &= \mathbb{E}_{p(\mathbf{Z}|\mathbf{X},\Theta^{old})}[\log p(\mathbf{X}, \mathbf{Z}|\Theta)] = \sum_{n=1}^{N} \mathbb{E}_{p(\boldsymbol{z}_n|\boldsymbol{x}_n,\Theta^{old})}[\log p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta)] \\
&= \sum_{n=1}^{N} \mathbb{E}_{p(\boldsymbol{z}_n|\boldsymbol{x}_n,\Theta^{old})}[\log p(\boldsymbol{x}_n|\boldsymbol{z}_n, \Theta) + \log p(\boldsymbol{z}_n|\Theta)]
\end{aligned}
$$

- **M (Maximization) step:**
  - Maximize the expected complete data log-likelihood w.r.t. Θ

$$
\Theta^{new} = \arg\max_{\Theta} \mathcal{Q}(\Theta, \Theta^{old})
$$

The Expectation Maximization (EM) Algorithm

Initialize the parameters: Θ^{old}. Then alternate between these steps:

- **E (Expectation) step:**
  - Compute the posterior distribution $p(\mathbf{Z}|\mathbf{X}, \Theta^{old})$ over latent variables $\mathbf{Z}$ using Θ^{old}
  - Compute the expected complete data log-likelihood w.r.t. *this* posterior distribution

$$\begin{aligned} \mathcal{Q}(\Theta, \Theta^{old}) &= \mathbb{E}_{p(\mathbf{Z}|\mathbf{X}, \Theta^{old})}[\log p(\mathbf{X}, \mathbf{Z}|\Theta)] = \sum_{n=1}^{N} \mathbb{E}_{p(\boldsymbol{z}_n|\boldsymbol{x}_n, \Theta^{old})}[\log p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta)] \\ &= \sum_{n=1}^{N} \mathbb{E}_{p(\boldsymbol{z}_n|\boldsymbol{x}_n, \Theta^{old})}[\log p(\boldsymbol{x}_n|\boldsymbol{z}_n, \Theta) + \log p(\boldsymbol{z}_n|\Theta)] \end{aligned}$$

- **M (Maximization) step:**
  - Maximize the expected complete data log-likelihood w.r.t. Θ

$$\Theta^{new} = \arg\max_{\Theta} \mathcal{Q}(\Theta, \Theta^{old})$$

- If the incomplete log-lik $p(\mathbf{X}|\Theta)$ not yet converged then set $\Theta^{old} = \Theta^{new}$ and go to the E step.

The Expectation Maximization (EM) Algorithm as Psuedo-code

The EM Algorithm

- Initialize Θ as $\Theta^{(0)}$, set $t = 1$
- Step 1: Compute conditional posterior of latent vars given current params $\Theta^{(t-1)}$

$$p(\boldsymbol{z}_n^{(t)}|\boldsymbol{x}_n, \Theta^{(t-1)}) = \frac{p(\boldsymbol{z}_n^{(t)}|\Theta^{(t-1)})p(\boldsymbol{x}_n|\boldsymbol{z}_n^{(t)}, \Theta^{(t-1)})}{p(\boldsymbol{x}_n|\Theta^{(t-1)})} \propto \text{prior} \times \text{likelihood}$$

- Step 2: Now maximize the expected complete data log-likelihood w.r.t. Θ

$$\Theta^{(t)} = \arg\max_{\Theta} \mathcal{Q}(\Theta, \Theta^{(t-1)}) = \arg\max_{\Theta} \sum_{n=1}^{N} \mathbb{E}_{p(\boldsymbol{z}_n^{(t)}|\boldsymbol{x}_n, \Theta^{(t-1)})}[\log p(\boldsymbol{x}_n, \boldsymbol{z}_n^{(t)}|\Theta)]$$

- If not yet converged, set $t = t + 1$ and go to Step 1.

The Expected CLL

- Deriving the EM algorithm requires finding the expression of the expected CLL

$$\mathcal{Q}(\Theta, \Theta^{old}) = \sum_{n=1}^{N} \mathbb{E}_{p(\boldsymbol{z}_n|\boldsymbol{x}_n,\Theta^{old})}[\log p(\boldsymbol{x}_n|\boldsymbol{z}_n, \Theta) + \log p(\boldsymbol{z}_n|\Theta)]$$

The Expected CLL

- Deriving the EM algorithm requires finding the expression of the expected CLL

$$\mathcal{Q}(\Theta, \Theta^{old}) = \sum_{n=1}^{N} \mathbb{E}_{p(\boldsymbol{z}_n|\boldsymbol{x}_n,\Theta^{old})}[\log p(\boldsymbol{x}_n|\boldsymbol{z}_n, \Theta) + \log p(\boldsymbol{z}_n|\Theta)]$$

- If $p(\boldsymbol{x}_n|\boldsymbol{z}_n, \Theta)$ and $p(\boldsymbol{z}_n|\Theta)$ are exp-family distributions, expected CLL will have a simple form

The Expected CLL

- Deriving the EM algorithm requires finding the expression of the expected CLL

$$\mathcal{Q}(\Theta, \Theta^{old}) = \sum_{n=1}^{N} \mathbb{E}_{p(\boldsymbol{z}_n|\boldsymbol{x}_n,\Theta^{old})}[\log p(\boldsymbol{x}_n|\boldsymbol{z}_n, \Theta) + \log p(\boldsymbol{z}_n|\Theta)]$$

- If $p(\boldsymbol{x}_n|\boldsymbol{z}_n, \Theta)$ and $p(\boldsymbol{z}_n|\Theta)$ are exp-family distributions, expected CLL will have a simple form
- Finding the expression for the expected CLL in such cases is fairly straightforward

The Expected CLL

- Deriving the EM algorithm requires finding the expression of the expected CLL

$$\mathcal{Q}(\Theta, \Theta^{old}) = \sum_{n=1}^{N} \mathbb{E}_{p(\boldsymbol{z}_n|\boldsymbol{x}_n,\Theta^{old})}[\log p(\boldsymbol{x}_n|\boldsymbol{z}_n, \Theta) + \log p(\boldsymbol{z}_n|\Theta)]$$

- If $p(\boldsymbol{x}_n|\boldsymbol{z}_n, \Theta)$ and $p(\boldsymbol{z}_n|\Theta)$ are exp-family distributions, expected CLL will have a simple form
- Finding the expression for the expected CLL in such cases is fairly straightforward
  - First write down the expressions for $p(\boldsymbol{x}_n|\boldsymbol{z}_n, \Theta)$ and $p(\boldsymbol{z}_n|\Theta)$ and simplify as much as possible

The Expected CLL

- Deriving the EM algorithm requires finding the expression of the expected CLL

$$\mathcal{Q}(\Theta, \Theta^{old}) = \sum_{n=1}^{N} \mathbb{E}_{p(\boldsymbol{z}_n|\boldsymbol{x}_n,\Theta^{old})}[\log p(\boldsymbol{x}_n|\boldsymbol{z}_n, \Theta) + \log p(\boldsymbol{z}_n|\Theta)]$$

- If $p(\boldsymbol{x}_n|\boldsymbol{z}_n, \Theta)$ and $p(\boldsymbol{z}_n|\Theta)$ are exp-family distributions, expected CLL will have a simple form
- Finding the expression for the expected CLL in such cases is fairly straightforward
  - First write down the expressions for $p(\boldsymbol{x}_n|\boldsymbol{z}_n, \Theta)$ and $p(\boldsymbol{z}_n|\Theta)$ and simplify as much as possible
  - In the resulting expressions, replace all terms containing $\boldsymbol{z}_n$'s by their respective expectations

The Expected CLL

- Deriving the EM algorithm requires finding the expression of the expected CLL

$$\mathcal{Q}(\Theta, \Theta^{old}) = \sum_{n=1}^{N} \mathbb{E}_{p(\boldsymbol{z}_n|\boldsymbol{x}_n,\Theta^{old})}[\log p(\boldsymbol{x}_n|\boldsymbol{z}_n, \Theta) + \log p(\boldsymbol{z}_n|\Theta)]$$

- If $p(\boldsymbol{x}_n|\boldsymbol{z}_n, \Theta)$ and $p(\boldsymbol{z}_n|\Theta)$ are exp-family distributions, expected CLL will have a simple form
- Finding the expression for the expected CLL in such cases is fairly straightforward
  - First write down the expressions for $p(\boldsymbol{x}_n|\boldsymbol{z}_n, \Theta)$ and $p(\boldsymbol{z}_n|\Theta)$ and simplify as much as possible
  - In the resulting expressions, replace all terms containing $\boldsymbol{z}_n$'s by their respective expectations, e.g.,
    - $\boldsymbol{z}_n$ replaced by $\mathbb{E}_{p(\boldsymbol{z}_n|\boldsymbol{x}_n,\Theta^{old})}[\boldsymbol{z}_n]$, i.e., the posterior mean of $\boldsymbol{z}_n$

The Expected CLL

- Deriving the EM algorithm requires finding the expression of the expected CLL

$$\mathcal{Q}(\Theta, \Theta^{old}) = \sum_{n=1}^{N} \mathbb{E}_{p(\boldsymbol{z}_n|\boldsymbol{x}_n,\Theta^{old})}[\log p(\boldsymbol{x}_n|\boldsymbol{z}_n, \Theta) + \log p(\boldsymbol{z}_n|\Theta)]$$

- If $p(\boldsymbol{x}_n|\boldsymbol{z}_n, \Theta)$ and $p(\boldsymbol{z}_n|\Theta)$ are exp-family distributions, expected CLL will have a simple form
- Finding the expression for the expected CLL in such cases is fairly straightforward
  - First write down the expressions for $p(\boldsymbol{x}_n|\boldsymbol{z}_n, \Theta)$ and $p(\boldsymbol{z}_n|\Theta)$ and simplify as much as possible
  - In the resulting expressions, replace all terms containing $\boldsymbol{z}_n$'s by their respective expectations, e.g.,
    - $\boldsymbol{z}_n$ replaced by $\mathbb{E}_{p(\boldsymbol{z}_n|\boldsymbol{x}_n,\Theta^{old})}[\boldsymbol{z}_n]$, i.e., the posterior mean of $\boldsymbol{z}_n$
    - $\boldsymbol{z}_n\boldsymbol{z}_n^\top$ replaced by $\mathbb{E}_{p(\boldsymbol{z}_n|\boldsymbol{x}_n,\Theta^{old})}[\boldsymbol{z}_n\boldsymbol{z}_n^\top]$

The Expected CLL

- Deriving the EM algorithm requires finding the expression of the expected CLL

$$\mathcal{Q}(\Theta, \Theta^{old}) = \sum_{n=1}^{N} \mathbb{E}_{p(\boldsymbol{z}_n|\boldsymbol{x}_n,\Theta^{old})}[\log p(\boldsymbol{x}_n|\boldsymbol{z}_n, \Theta) + \log p(\boldsymbol{z}_n|\Theta)]$$

- If $p(\boldsymbol{x}_n|\boldsymbol{z}_n, \Theta)$ and $p(\boldsymbol{z}_n|\Theta)$ are exp-family distributions, expected CLL will have a simple form
- Finding the expression for the expected CLL in such cases is fairly straightforward
  - First write down the expressions for $p(\boldsymbol{x}_n|\boldsymbol{z}_n, \Theta)$ and $p(\boldsymbol{z}_n|\Theta)$ and simplify as much as possible
  - In the resulting expressions, replace all terms containing $\boldsymbol{z}_n$'s by their respective expectations, e.g.,
    - $\boldsymbol{z}_n$ replaced by $\mathbb{E}_{p(\boldsymbol{z}_n|\boldsymbol{x}_n,\Theta^{old})}[\boldsymbol{z}_n]$, i.e., the posterior mean of $\boldsymbol{z}_n$
    - $\boldsymbol{z}_n\boldsymbol{z}_n^\top$ replaced by $\mathbb{E}_{p(\boldsymbol{z}_n|\boldsymbol{x}_n,\Theta^{old})}[\boldsymbol{z}_n\boldsymbol{z}_n^\top]$
    - .. and so on..

Online or Incremental EM

- Needn't compute $p(\boldsymbol{z}_n|\boldsymbol{x}_n)$ for every $\boldsymbol{x}_n$ in each EM iteration (computational/storage efficiency)

Online or Incremental EM

- Needn't compute $p(\boldsymbol{z}_n|\boldsymbol{x}_n)$ for every $\boldsymbol{x}_n$ in each EM iteration (computational/storage efficiency)
  - Recall that the expected CLL is often a sum over all data points

$$\mathcal{Q}(\Theta, \Theta^{old}) = \mathbb{E}[\log p(\mathbf{X}, \mathbf{Z}|\Theta) = \sum_{n=1}^{N} \mathbb{E}[\log p(\boldsymbol{x}_n|\boldsymbol{z}_n, \theta)] + \mathbb{E}[\log p(\boldsymbol{z}_n|\phi)]$$

Online or Incremental EM

- Needn't compute $p(\boldsymbol{z}_n|\boldsymbol{x}_n)$ for every $\boldsymbol{x}_n$ in each EM iteration (computational/storage efficiency)
  - Recall that the expected CLL is often a sum over all data points

$$\mathcal{Q}(\Theta, \Theta^{old}) = \mathbb{E}[\log p(\mathbf{X}, \mathbf{Z}|\Theta) = \sum_{n=1}^{N} \mathbb{E}[\log p(\boldsymbol{x}_n|\boldsymbol{z}_n, \theta)] + \mathbb{E}[\log p(\boldsymbol{z}_n|\phi)]$$

  - Can compute this quantity recursively using small minibatches of data

$$\mathcal{Q}_t = (1-\gamma_t)\mathcal{Q}_{t-1} + \gamma_t \left[\sum_{n=1}^{N_t} \mathbb{E}[\log p(\boldsymbol{x}_n|\boldsymbol{z}_n, \theta)] + \mathbb{E}[\log p(\boldsymbol{z}_n|\phi)] \right]$$

  .. where $\gamma_t = (1+t)^{-\kappa}, 0.5 < \kappa \leq 1$ is a decaying learning rate

Online or Incremental EM

- Needn't compute $p(\boldsymbol{z}_n|\boldsymbol{x}_n)$ for every $\boldsymbol{x}_n$ in each EM iteration (computational/storage efficiency)
  - Recall that the expected CLL is often a sum over all data points

$$\mathcal{Q}(\Theta, \Theta^{old}) = \mathbb{E}[\log p(\mathbf{X}, \mathbf{Z}|\Theta) = \sum_{n=1}^{N} \mathbb{E}[\log p(\boldsymbol{x}_n|\boldsymbol{z}_n, \theta)] + \mathbb{E}[\log p(\boldsymbol{z}_n|\phi)]$$

  - Can compute this quantity recursively using small minibatches of data

$$\mathcal{Q}_t = (1-\gamma_t)\mathcal{Q}_{t-1} + \gamma_t \left[\sum_{n=1}^{N_t} \mathbb{E}[\log p(\boldsymbol{x}_n|\boldsymbol{z}_n, \theta)] + \mathbb{E}[\log p(\boldsymbol{z}_n|\phi)]\right]$$

  .. where $\gamma_t = (1+t)^{-\kappa}, 0.5 < \kappa \leq 1$ is a decaying learning rate
  - Requires computing $p(\boldsymbol{z}_n|\boldsymbol{x}_n)$ only for data in current mini-batch (computational/storage efficiency)

Online or Incremental EM

- Needn't compute $p(\boldsymbol{z}_n|\boldsymbol{x}_n)$ for every $\boldsymbol{x}_n$ in each EM iteration (computational/storage efficiency)
  - Recall that the expected CLL is often a sum over all data points

$$\mathcal{Q}(\Theta, \Theta^{old}) = \mathbb{E}[\log p(\mathbf{X}, \mathbf{Z}|\Theta) = \sum_{n=1}^{N} \mathbb{E}[\log p(\boldsymbol{x}_n|\boldsymbol{z}_n, \theta)] + \mathbb{E}[\log p(\boldsymbol{z}_n|\phi)]$$

  - Can compute this quantity recursively using small minibatches of data

$$\mathcal{Q}_t = (1 - \gamma_t)\mathcal{Q}_{t-1} + \gamma_t \left[\sum_{n=1}^{N_t} \mathbb{E}[\log p(\boldsymbol{x}_n|\boldsymbol{z}_n, \theta)] + \mathbb{E}[\log p(\boldsymbol{z}_n|\phi)] \right]$$

  .. where $\gamma_t = (1+t)^{-\kappa}, 0.5 < \kappa \leq 1$ is a decaying learning rate
  - Requires computing $p(\boldsymbol{z}_n|\boldsymbol{x}_n)$ only for data in current mini-batch (computational/storage efficiency)
  - MLE on above $\mathcal{Q}_t$ can be shown to be equivalent to a simple recursive updates for Θ

$$\Theta^{(t)} = (1 - \gamma_t) \times \Theta^{(t-1)} + \gamma_t \times \arg\max_{\Theta} \underbrace{\mathcal{Q}(\Theta, \Theta^{t-1})}_{\substack{\text{computed using only} \\ \text{the } N_t \text{ examples} \\ \text{from this minibatch}}}$$

Some Applications of EM

- Mixture of (multivariate) Gaussians/Bernoullis,multinoullis, Mixture of experts models

Some Applications of EM

- Mixture of (multivariate) Gaussians/Bernoullis,multinoullis, Mixture of experts models
- Problems with missing labels/features (treat these as latent variables)

Some Applications of EM

- Mixture of (multivariate) Gaussians/Bernoullis,multinoullis, Mixture of experts models
- Problems with missing labels/features (treat these as latent variables)
- Note that EM not only gives estimates of the parameters Θ but also infers latent variables $\mathbf{Z}$

Some Applications of EM

- Mixture of (multivariate) Gaussians/Bernoullis,multinoullis, Mixture of experts models
- Problems with missing labels/features (treat these as latent variables)
- Note that EM not only gives estimates of the parameters Θ but also infers latent variables $\mathbf{Z}$
- Hyperparameter estimation in probabilistic models (an alternative to MLE-II)

Some Applications of EM

- Mixture of (multivariate) Gaussians/Bernoullis,multinoullis, Mixture of experts models
- Problems with missing labels/features (treat these as latent variables)
- Note that EM not only gives estimates of the parameters Θ but also infers latent variables $\mathbf{Z}$
- Hyperparameter estimation in probabilistic models (an alternative to MLE-II)
    - We've already seen MLE-II where we did MLE on marginal likelihood, e.g., for linear regression

$$p(\boldsymbol{y}|\mathbf{X}, \lambda, \beta) = \int p(\boldsymbol{y}|\mathbf{X}, \boldsymbol{w}, \beta)p(\boldsymbol{w}|\lambda)d\boldsymbol{w}$$

Some Applications of EM

- Mixture of (multivariate) Gaussians/Bernoullis,multinoullis, Mixture of experts models
- Problems with missing labels/features (treat these as latent variables)
- Note that EM not only gives estimates of the parameters Θ but also infers latent variables $\mathbf{Z}$
- Hyperparameter estimation in probabilistic models (an alternative to MLE-II)
  - We've already seen MLE-II where we did MLE on marginal likelihood, e.g., for linear regression

$$p(\boldsymbol{y}|\mathbf{X}, \lambda, \beta) = \int p(\boldsymbol{y}|\mathbf{X}, \boldsymbol{w}, \beta)p(\boldsymbol{w}|\lambda)d\boldsymbol{w}$$

  - As an alternative, can treat $\boldsymbol{w}$ as a latent variable and β, λ as parameters and use EM to learn these

Some Applications of EM

- Mixture of (multivariate) Gaussians/Bernoullis,multinoullis, Mixture of experts models
- Problems with missing labels/features (treat these as latent variables)
- Note that EM not only gives estimates of the parameters Θ but also infers latent variables $\mathbf{Z}$
- Hyperparameter estimation in probabilistic models (an alternative to MLE-II)
  - We've already seen MLE-II where we did MLE on marginal likelihood, e.g., for linear regression

$$p(\boldsymbol{y}|\mathbf{X}, \lambda, \beta) = \int p(\boldsymbol{y}|\mathbf{X}, \boldsymbol{w}, \beta)p(\boldsymbol{w}|\lambda)d\boldsymbol{w}$$

  - As an alternative, can treat $\boldsymbol{w}$ as a latent variable and β, λ as parameters and use EM to learn these
  - Note: In this case, the latent variable $\boldsymbol{w}$ is not "local" (but EM still applies)

Some Applications of EM

- Mixture of (multivariate) Gaussians/Bernoullis,multinoullis, Mixture of experts models
- Problems with missing labels/features (treat these as latent variables)
- Note that EM not only gives estimates of the parameters Θ but also infers latent variables $\mathbf{Z}$
- Hyperparameter estimation in probabilistic models (an alternative to MLE-II)
  - We've already seen MLE-II where we did MLE on marginal likelihood, e.g., for linear regression

$$p(\boldsymbol{y}|\mathbf{X}, \lambda, \beta) = \int p(\boldsymbol{y}|\mathbf{X}, \boldsymbol{w}, \beta) p(\boldsymbol{w}|\lambda) d\boldsymbol{w}$$

  - As an alternative, can treat $\boldsymbol{w}$ as a latent variable and β, λ as parameters and use EM to learn these
  - Note: In this case, the latent variable $\boldsymbol{w}$ is not "local" (but EM still applies)
  - E step computes posterior $p(\boldsymbol{w}|\mathbf{X}, \boldsymbol{y}, \beta, \lambda)$ assuming β, λ fixed from the previous M step

Some Applications of EM

- Mixture of (multivariate) Gaussians/Bernoullis,multinoullis, Mixture of experts models
- Problems with missing labels/features (treat these as latent variables)
- Note that EM not only gives estimates of the parameters Θ but also infers latent variables $\mathbf{Z}$
- Hyperparameter estimation in probabilistic models (an alternative to MLE-II)
  - We've already seen MLE-II where we did MLE on marginal likelihood, e.g., for linear regression

$$p(\boldsymbol{y}|\mathbf{X}, \lambda, \beta) = \int p(\boldsymbol{y}|\mathbf{X}, \boldsymbol{w}, \beta)p(\boldsymbol{w}|\lambda)d\boldsymbol{w}$$

  - As an alternative, can treat $\boldsymbol{w}$ as a latent variable and β, λ as parameters and use EM to learn these
  - Note: In this case, the latent variable $\boldsymbol{w}$ is not "local" (but EM still applies)
  - E step computes posterior $p(\boldsymbol{w}|\mathbf{X}, \boldsymbol{y}, \beta, \lambda)$ assuming β, λ fixed from the previous M step
  - M step maximizes $\mathbb{E}[\log p(\boldsymbol{y}, \boldsymbol{w}|\mathbf{X}, \beta, \lambda)] = \mathbb{E}[\log p(\boldsymbol{y}|\boldsymbol{w}, \mathbf{X}, \beta) + \log p(\boldsymbol{w}|\lambda)]$ w.r.t. λ, β

Some Applications of EM

- Mixture of (multivariate) Gaussians/Bernoullis,multinoullis, Mixture of experts models
- Problems with missing labels/features (treat these as latent variables)
- Note that EM not only gives estimates of the parameters Θ but also infers latent variables $\mathbf{Z}$
- Hyperparameter estimation in probabilistic models (an alternative to MLE-II)
  - We've already seen MLE-II where we did MLE on marginal likelihood, e.g., for linear regression

$$p(\boldsymbol{y}|\mathbf{X}, \lambda, \beta) = \int p(\boldsymbol{y}|\mathbf{X}, \boldsymbol{w}, \beta)p(\boldsymbol{w}|\lambda)d\boldsymbol{w}$$

  - As an alternative, can treat $\boldsymbol{w}$ as a latent variable and β, λ as parameters and use EM to learn these
  - Note: In this case, the latent variable $\boldsymbol{w}$ is not "local" (but EM still applies)
  - E step computes posterior $p(\boldsymbol{w}|\mathbf{X}, \boldsymbol{y}, \beta, \lambda)$ assuming β, λ fixed from the previous M step
  - M step maximizes $\mathbb{E}[\log p(\boldsymbol{y}, \boldsymbol{w}|\mathbf{X}, \beta, \lambda)] = \mathbb{E}[\log p(\boldsymbol{y}|\boldsymbol{w}, \mathbf{X}, \beta) + \log p(\boldsymbol{w}|\lambda)]$ w.r.t. λ, β
    - This requires using expectations of quantities like $\boldsymbol{w}$ and $\boldsymbol{w}\boldsymbol{w}^\top$ which can be obtained easily from the posterior $p(\boldsymbol{w}|\mathbf{X}, \boldsymbol{y}, \beta, \lambda)$ which we compute in the E step

The EM Algorithm: Some Comments

- The E and M steps may not always be possible to perform exactly. Some reasons

The EM Algorithm: Some Comments

- The E and M steps may not always be possible to perform exactly. Some reasons
  - The posterior of latent variables $p(\mathbf{Z}|\mathbf{X}, \Theta)$ may not be easy to find

The EM Algorithm: Some Comments

- The E and M steps may not always be possible to perform exactly. Some reasons
  - The posterior of latent variables $p(\mathbf{Z}|\mathbf{X},\Theta)$ may not be easy to find
    - Would need to approximate $p(\mathbf{Z}|\mathbf{X},\Theta)$ in such a case

The EM Algorithm: Some Comments

- The E and M steps may not always be possible to perform exactly. Some reasons
  - The posterior of latent variables $p(\mathbf{Z}|\mathbf{X},\Theta)$ may not be easy to find
    - Would need to approximate $p(\mathbf{Z}|\mathbf{X},\Theta)$ in such a case
  - Even if $p(\mathbf{Z}|\mathbf{X},\Theta)$ is easy, the expected CLL, i.e., $\mathbb{E}[\log p(\mathbf{X},\mathbf{Z}|\Theta)]$ may still not be tractabe

$$\mathbb{E}[\log p(\mathbf{X},\mathbf{Z}|\Theta)] = \int \log p(\mathbf{X},\mathbf{Z}|\Theta)p(\mathbf{Z}|\mathbf{X},\Theta)d\mathbf{Z}$$

The EM Algorithm: Some Comments

- The E and M steps may not always be possible to perform exactly. Some reasons
  - The posterior of latent variables $p(\mathbf{Z}|\mathbf{X}, \Theta)$ may not be easy to find
    - Would need to approximate $p(\mathbf{Z}|\mathbf{X}, \Theta)$ in such a case
  - Even if $p(\mathbf{Z}|\mathbf{X}, \Theta)$ is easy, the expected CLL, i.e., $\mathbb{E}[\log p(\mathbf{X}, \mathbf{Z}|\Theta)]$ may still not be tractabe

$$\mathbb{E}[\log p(\mathbf{X}, \mathbf{Z}|\Theta)] = \int \log p(\mathbf{X}, \mathbf{Z}|\Theta) p(\mathbf{Z}|\mathbf{X}, \Theta) d\mathbf{Z}$$

.. which can be approximated, e.g., using Monte-Carlo expectation (called Monte-Carlo EM)

The EM Algorithm: Some Comments

- The E and M steps may not always be possible to perform exactly. Some reasons
  - The posterior of latent variables $p(\mathbf{Z}|\mathbf{X},\Theta)$ may not be easy to find
    - Would need to approximate $p(\mathbf{Z}|\mathbf{X},\Theta)$ in such a case
  - Even if $p(\mathbf{Z}|\mathbf{X},\Theta)$ is easy, the expected CLL, i.e., $\mathbb{E}[\log p(\mathbf{X},\mathbf{Z}|\Theta)]$ may still not be tractabe

$$\mathbb{E}[\log p(\mathbf{X},\mathbf{Z}|\Theta)] = \int \log p(\mathbf{X},\mathbf{Z}|\Theta)p(\mathbf{Z}|\mathbf{X},\Theta)d\mathbf{Z}$$

    .. which can be approximated, e.g., using Monte-Carlo expectation (called Monte-Carlo EM)
  - Maximization of the expected CLL may not be possible in closed form

The EM Algorithm: Some Comments

- The E and M steps may not always be possible to perform exactly. Some reasons
  - The posterior of latent variables $p(\mathbf{Z}|\mathbf{X},\Theta)$ may not be easy to find
    - Would need to approximate $p(\mathbf{Z}|\mathbf{X},\Theta)$ in such a case
  - Even if $p(\mathbf{Z}|\mathbf{X},\Theta)$ is easy, the expected CLL, i.e., $\mathbb{E}[\log p(\mathbf{X},\mathbf{Z}|\Theta)]$ may still not be tractabe

$$\mathbb{E}[\log p(\mathbf{X},\mathbf{Z}|\Theta)] = \int \log p(\mathbf{X},\mathbf{Z}|\Theta)p(\mathbf{Z}|\mathbf{X},\Theta)d\mathbf{Z}$$

    .. which can be approximated, e.g., using Monte-Carlo expectation (called Monte-Carlo EM)
  - Maximization of the expected CLL may not be possible in closed form
- EM works even if the M step is only solved approximately (Generalized EM)

The EM Algorithm: Some Comments

- The E and M steps may not always be possible to perform exactly. Some reasons
  - The posterior of latent variables $p(\mathbf{Z}|\mathbf{X},\Theta)$ may not be easy to find
    - Would need to approximate $p(\mathbf{Z}|\mathbf{X},\Theta)$ in such a case
  - Even if $p(\mathbf{Z}|\mathbf{X},\Theta)$ is easy, the expected CLL, i.e., $\mathbb{E}[\log p(\mathbf{X},\mathbf{Z}|\Theta)]$ may still not be tractabe

$$\mathbb{E}[\log p(\mathbf{X},\mathbf{Z}|\Theta)] = \int \log p(\mathbf{X},\mathbf{Z}|\Theta)p(\mathbf{Z}|\mathbf{X},\Theta)d\mathbf{Z}$$

    .. which can be approximated, e.g., using Monte-Carlo expectation (called Monte-Carlo EM)
  - Maximization of the expected CLL may not be possible in closed form
- EM works even if the M step is only solved approximately (Generalized EM)
- If M step has multiple parameters whose updates depend on each other, they are updated in an alternating fashion - called Expectation Conditional Maximization (ECM) algorithm

The EM Algorithm: Some Comments

- Can also use the idea of "annealing"† to avoid local optima problem of EM

† Deterministic annealing EM algorithm (Ueda and Nakano, 1998)

The EM Algorithm: Some Comments

- Can also use the idea of "annealing" † to avoid local optima problem of EM
  - Basic idea: Use a modified form of the posterior over the latent variables

$$f(\boldsymbol{z}_n|\boldsymbol{x}_n) = \frac{p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta)^\beta}{\int p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta)^\beta \boldsymbol{z}_n}$$

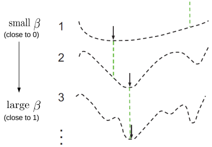
† Deterministic annealing EM algorithm (Ueda and Nakano, 1998)

The EM Algorithm: Some Comments

- Can also use the idea of "annealing"† to avoid local optima problem of EM
  - Basic idea: Use a modified form of the posterior over the latent variables

$$f(\boldsymbol{z}_n|\boldsymbol{x}_n) = \frac{p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta)^\beta}{\int p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta)^\beta \boldsymbol{z}_n}$$

  - Here $\beta \in (0, 1)$ is a "temperature" parameter (start very small and gradually increase)



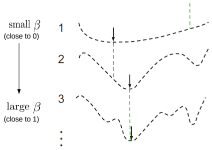
† Deterministic annealing EM algorithm (Ueda and Nakano, 1998)

The EM Algorithm: Some Comments

- Can also use the idea of "annealing"[†] to avoid local optima problem of EM
  - Basic idea: Use a modified form of the posterior over the latent variables

$$f(\boldsymbol{z}_n|\boldsymbol{x}_n) = \frac{p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta)^\beta}{\int p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta)^\beta \boldsymbol{z}_n}$$

  - Here $\beta \in (0, 1)$ is a "temperature" parameter (start very small and gradually increase)



- Other advanced probabilistic inference algorithms are based on ideas similar to EM

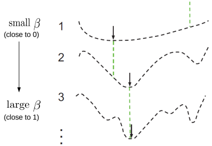
† Deterministic annealing EM algorithm (Ueda and Nakano, 1998)

The EM Algorithm: Some Comments

- Can also use the idea of "annealing"† to avoid local optima problem of EM
  - Basic idea: Use a modified form of the posterior over the latent variables

$$f(\boldsymbol{z}_n|\boldsymbol{x}_n) = \frac{p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta)^\beta}{\int p(\boldsymbol{x}_n, \boldsymbol{z}_n|\Theta)^\beta \boldsymbol{z}_n}$$

  - Here $\beta \in (0,1)$ is a "temperature" parameter (start very small and gradually increase)



- Other advanced probabilistic inference algorithms are based on ideas similar to EM
  - E.g., Variational Bayes (VB) inference, a.k.a. Variational Inference (VI): Also maximizes a lower bound on log evidence $\log p(\mathbf{X})$ (and unlike EM, treats all unknowns as latent vars). Will see it soon.

† Deterministic annealing EM algorithm (Ueda and Nakano, 1998)