Inference in Multi-parameter Models, Conditional Posterior, Local Conjugacy

CS772A: Probabilistic Machine Learning

Piyush Rai

Plan for today

- Wrapping up GPs
- Inference in multi-parameter models
  - Conditional posterior
  - Local conjugacy
  - An example: Bayesian matrix factorization (BMF)

Scalability of GPs

- Computational costs in some steps of GP models scale in the size of training data
- For example, prediction cost is $O(N)$

$O(N)$ cost assuming C_N is already inverted

$$p(y_*|\boldsymbol{y}) = \mathcal{N}(\mu_*, \sigma_*^2) \qquad \mu_* = \mathbf{k}_*^\top \mathbf{C}_N^{-1}\boldsymbol{y} \qquad \sigma_*^2 = \kappa(x_*, x_*) - \mathbf{k}_*^\top \mathbf{C}_N^{-1}\mathbf{k}_* + \beta^{-1}$$

- GP models often require matrix inversions (e.g., in marg-lik computation when estimating hyperparameters) – takes $O(N^3)$
- Storage also requires $O(N^2)$ since need to store the covariance matrix
- A lot of work on speeding up GPs[1]. Some prominent approaches include
  - Inducing Point Methods (condition predictions only on a small set of "learnable" points)
  - Divide-and-Conquer (learn GP on small subsets of data and aggregate predictions)
  - Kernel approximations

$M \ll N$ pseudo-inputs and pseudo-outputs

- Note that nearest neighbor methods and kernel methods also face similar issues
  - Many tricks to speed up kernel methods can be used for speeding up GPs too

[1]When Gaussian Process Meets Big Data: A Review of Scalable GPs - Liu et al, 2018

GP: Some Comments

- GP is sometimes referred to as a nonparametric model because
  - Complexity (representation size) of the function f grows in the size of training data
  - To see this, note the form of the GP predictions, e.g., predictive mean in GP regression

$$\mu_* = f(\boldsymbol{x}_*) = \mathbf{k}_*^\top \mathbf{C}_N^{-1} \boldsymbol{y} = \mathbf{k}_*^\top \boldsymbol{\alpha} = \sum_{n=1}^{N} \alpha_n k(\boldsymbol{x}_*, \boldsymbol{x}_n)$$

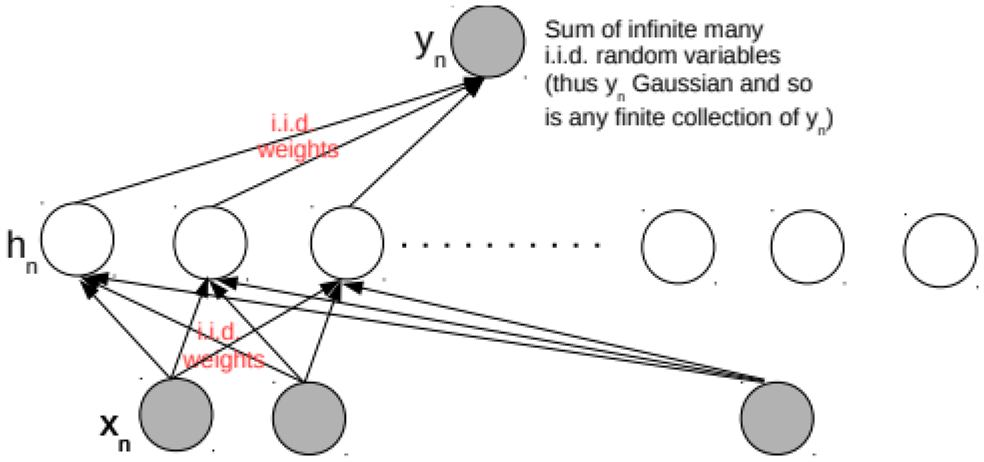
  - It implies that $f(.) = \sum_{n=1}^{N} \alpha_n k(., \boldsymbol{x}_n)$ which means f is written in terms of all training examples
  - Thus the representation size of f depends on the number of training examples

- In contrast, a parametric model has a size that doesn't grow with training data
  - E.g., a linear model learns a weight vector $\boldsymbol{w} \in \mathbb{R}^D$ (D parameters, size independent of N)

- Nonparametric models more flexible since their complexity is not limited beforehand
  - Note: Methods like nearest neighbors and kernel SVMs are also nonparametric (but not Bayesian)

Neural Networks and Gaussian Process

- An infinitely-wide single hidden layer NN with i.i.d. weights is <u>equivalent</u> to a GP
- Shown formally by (Neal[2], 1994). Based on applying the central limit theorem



- This equivalence is useful for several reasons
  - Can use a GP instead of an infinitely wide Bayesian NN (which is impractical anyway)
  - With GPs, inference is easy (at least for regression and with known hyperparams)
  - A proof that GPs can also learn any function (just like infinitely wide neural nets - Hornik's theorem)
- Connection generalized to infinitely wide multiple hidden layer NN (Lee et al[3], 2018)

[2]Priors for infinite networks, Tech Report, 1994
[3]Deep Neural Networks as Gaussian Processes (ICLR 2018)

GP: A Few Other Comments

- GPs can be thought of as Bayesian analogues of kernel methods
- Can get estimate in the uncertainty in the function and its predictions

  - Can learn the kernel (by learning the hyperparameters of the kernels)
- Not limited to supervised learning problems
  - f could even define a mapping of low-dim latent variable z_n to an observation x_n

$$x_n = f(z_n) + \text{"noise"}$$

GP latent variable model for dimensionality reduction (like a kernel version of probabilistic PCA)

- Many mature implementations of GP exist. You may check out
  - GPyTorch (PyTorch), GPFlow (Tensorflow)
  - GPML (MATLAB), GPsuff (MATLAB/Octave)

Coming up next

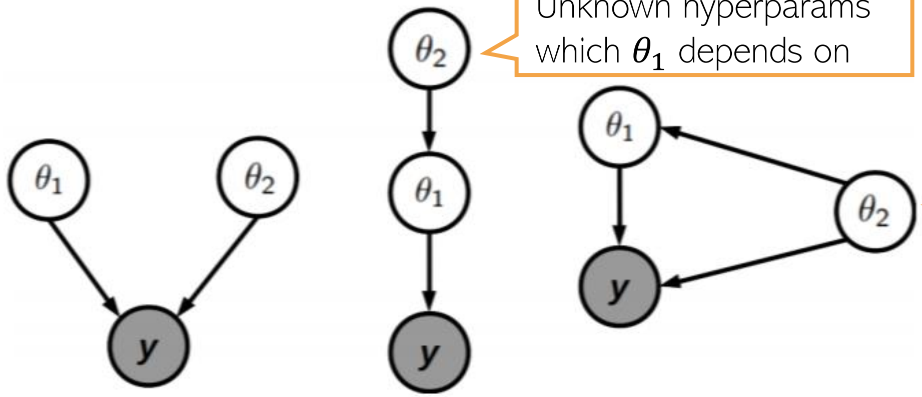
- Foray into models with several parameters
- Goal: Infer the posterior over all of them (not posterior for some, MLE-II for others)
- Idea of conditional/local posteriors in such problems
- Local conjugacy (which helps in computing conditional posteriors)
- Gibbs sampling (an algorithm that infer the joint posterior via conditional posteriors)
- An example: Bayesian matrix factorization (a model with many parameters)
- Conditional/local posterior, local conjugacy, etc are important ideas (will appear in many inference algorithms that we will see later)

Moving Beyond Simple Models..

- So far, our models usually had a "main" parameter and maybe a few hyperparams, e.g.,
  - For a Gaussian, infer the mean assuming variance known (or vice-versa)
  - Bayesian linear regression with weight vector $\boldsymbol{w}$ and noise/prior precision hyperparams β, λ
  - GP regression with one function to be learned, and fixed hyperparams
- Easy posterior inference if the likelihood and prior are conjugate to each other
- Hyperparams were estimated via MLE-II (since full posterior is much harder)
- For non-conjugate models or models with many parameters, need posterior approx
  - Can use Laplace approx but it has limitations (unimodal posterior, model should be differentiable)
- We will now look at more general inference schemes for such "difficult" cases
  - Difficult = Models with many params/hyperparams, non-conjugacy, non-differ., etc
  - Will be intractable in general. We will study approx. inference methods to handle such cases

Multiparameter Models

- Multiparameter models consist of two or more unknowns, say $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$
- Given some data $\boldsymbol{y}$, some examples for the simple two parameter case



Unknown hyperparams which $\boldsymbol{\theta}_1$ depends on

Unknown hyperparams which $\boldsymbol{\theta}_1$ and the likelihood depends on

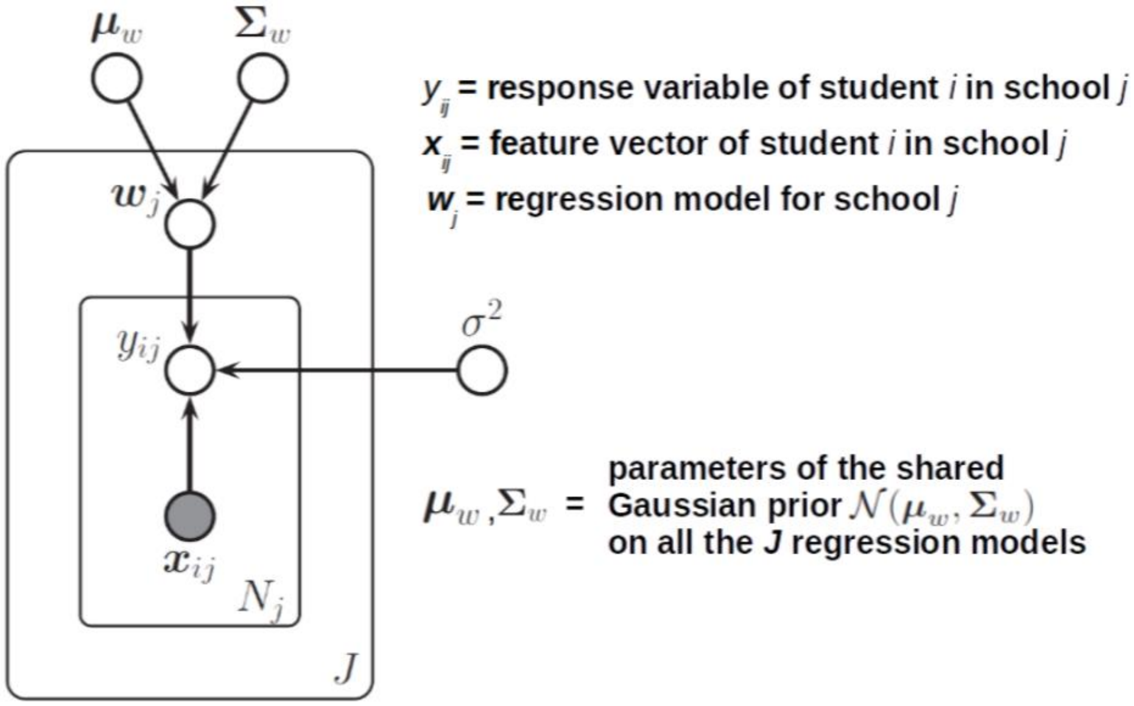
- Assume the likelihood model to be of the form $p(\boldsymbol{y}|\boldsymbol{\theta}_1, \boldsymbol{\theta}_2)$
- Assume a joint prior distribution $p(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2)$

This prior may still be conditioned on some fixed hyperparams

- The joint posterior $p(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2|\boldsymbol{y}) \propto p(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2)p(\boldsymbol{y}|\boldsymbol{\theta}_1, \boldsymbol{\theta}_2)$
  - Easy if the joint prior is conjugate to the likelihood (e.g., NIW prior for Gaussian likelihood)
  - Otherwise needs more work, e.g., MLE-II, MCMC, VI, etc. (already saw MLE-II)

Multiparameter Models: Some Examples

- Multiparameter models arise in many situations, e.g.,
  - Probabilistic models with unknown hyperparams (e.g., Bayesian linear regression)
  - Joint analysis of data from multiple (and possibly related) groups: Hierarchical models



  - .. and in fact, pretty much in any non-toy example of probabilistic model

Another Example: Matrix Factorization/Completion

- Given: Data $\boldsymbol{R} = \{r_{ij}\}$ of "interactions" (e.g., ratings)
  - Here $i = 1,2,\ldots,N$ denotes users, $j = 1,2,\ldots,M$ denotes items

Assume each item $j = 1,2,\ldots,M$ modeled by an unknown parameter vector $\boldsymbol{v}_j \in \mathbb{R}^K$

Only a small fraction of user-item ratings are observed (say user-item index set Ω)

Need to predict the rest using these

Many methods exist

Assume each user $i = 1,2,\ldots,N$ modeled by an unknown parameter vector $\boldsymbol{u}_i \in \mathbb{R}^K$ for some small K

4	3			5	
5		4		4	
4		5	3	4	
	3				5
	4				4
		2	4		5

Similar to assuming a low-rank structure on the ratings matrix

Can assume zero mean Gaussian noise $\mathcal{N}(0, \beta^{-1})$ for real-valued ratings

$$r_{ij} = \boldsymbol{u}_i^\top \boldsymbol{v}_j + \epsilon_{ij}$$

Likelihood

$$p(r_{ij} | \boldsymbol{u}_i, \boldsymbol{v}_j) = \mathcal{N}(r_{ij} | \boldsymbol{u}_i^\top \boldsymbol{v}_j, \beta^{-1})$$

Individual priors for each user and item

$$p(\boldsymbol{u}_i) = \mathcal{N}(\boldsymbol{u}_i | \boldsymbol{0}, \lambda_u^{-1})$$

$$p(\boldsymbol{v}_j) = \mathcal{N}(\boldsymbol{v}_j | \boldsymbol{0}, \lambda_v^{-1})$$

$$p(\{\boldsymbol{u}_i\}_{i=1}^N, \{\boldsymbol{v}_j\}_{j=1}^M | \boldsymbol{R})$$

Posterior

For simplicity, hyperparams $\beta, \lambda_u, \lambda_v$ are fixed or unknown but we may only want to do point estimation for them

Bayesian Matrix Factorization (BMF): The Posterior

- Our target posterior distribution for this model is

Due to conditional independence of the observations given params

Assume that the joint prior factorizes into individual priors

$$p(\mathbf{U},\mathbf{V}|\mathbf{R}) = \frac{p(\mathbf{R}|\mathbf{U},\mathbf{V})p(\mathbf{U},\mathbf{V})}{\int\int p(\mathbf{R}|\mathbf{U},\mathbf{V})p(\mathbf{U},\mathbf{V})d\mathbf{U}d\mathbf{V}}$$

$$= \frac{\prod_{(i,j)\in\Omega} p(r_{ij}|\boldsymbol{u}_i,\mathbf{v}_j)\prod_i p(\boldsymbol{u}_i)\prod_j p(\mathbf{v}_j)}{\int\dots\int\prod_{(i,j)\in\Omega} p(r_{ij}|\boldsymbol{u}_i,\mathbf{v}_j)\prod_i p(\boldsymbol{u}_i)\prod_j p(\mathbf{v}_j)d\boldsymbol{u}_1\dots d\boldsymbol{u}_N d\mathbf{v}_1\dots d\mathbf{v}_M}$$

- Posterior still intractable since integrals here are intractable
- Need to approx. the posterior. One way is via conditional posteriors (CP), e.g.,

All of $\boldsymbol{U}$ except $\boldsymbol{u}_i$

All of $\boldsymbol{V}$ except $\boldsymbol{v}_j$

$$p(\boldsymbol{u}_i|\boldsymbol{R},\boldsymbol{V},\boldsymbol{U}_{-i}) \qquad p(\boldsymbol{v}_j|\boldsymbol{R},\boldsymbol{U},\boldsymbol{V}_{-j})$$

- CP of each unknown is conditioned on fixed values of all other unknowns
  - The different CPs can be computed in an alternating fashion (like ALT-OPT/EM)
- Note: CP individually won't give us joint posterior. Need to combine them

E.g., using MCMC, variational inference, EM, etc

Conditional Posterior and Local Conjugacy

- Conditional Posteriors are easy to compute for model that are locally conjugate
  - Note: CP is sometimes also referred to as Complete Conditional or Local Posterior
- Consider a model with data $\boldsymbol{X}$ and K unknown params/h.p. $\boldsymbol{\Theta} = (\theta_1, \theta_2, \ldots, \theta_K)$
- Suppose the joint posterior $p(\Theta|\boldsymbol{X}) = \frac{p(\Theta)p(\boldsymbol{X}|\Theta)}{p(\boldsymbol{X})}$ is intractable (like in BMF)
- Local Conjugacy: If we can compute each CP tractably

Possible if the likelihood $p(\boldsymbol{X}|\theta_k, \Theta_{-k})$ and prior $p(\theta_k)$ are conjugate to each other

This is the notion of local conjugacy as opposed to full/joint conjugacy

Θ_{-k} is assumed known while computing this CP

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

- Important: In the above context, when considering the likelihood $p(\boldsymbol{X}|\theta_k, \Theta_{-k})$
  - $\boldsymbol{X}$ actually refers to only that part of data $\boldsymbol{X}$ that depends on θ_k
  - Θ_{-k} refers to only those unknowns that "interact" with θ_k in generating that part of data

In the likelihood model

Approximating Joint Posterior via CPs

- With the conditional posterior based approximation, the target joint posterior

$$p(\Theta|\mathbf{X}) = \frac{p(\mathbf{X}|\Theta)p(\Theta)}{p(\mathbf{X})}$$

... is represented by several conditional posteriors $p(\theta_k, |\mathbf{X}, \Theta_{-k})$, $k = 1,2, \dots, K$

- Each CP is a distribution over one unknown θ_k , given all other unknowns
- Need a way to "combine" these CPs to get the overall posterior
  - MCMC (e.g., Gibbs sampling): Based on generating samples from the CPs
  - Variational Inference (VI): Based on cyclic estimation of each CP
  - Note: Expectation Maximization also computes CP of latent variables in its E step
- More on this when we discuss MCMC, VI, EM, etc

Will revisit Gibbs sampling again when discussing MCMC algos

- But let's look at Gibbs sampling (an MCMC algo) right away as it is fairly simple

Gibbs Sampling (Geman and Geman, 1982)

- A general algo to generate samples from multivar. distr. <u>one component at a time</u>
  - Not limited to sampling from intractable posteriors only
  - Sometimes can be used even if we can draw from the multivar distr. directly

Note: If posterior, it will be conditioned on other stuff too (e.g., data, other param, etc)

- Assume we want to sample from a joint distribution $p(\theta_1, \theta_2, \ldots, \theta_K)$
- It generates one component θ_k at a time using its conditional $p(\theta_k | \Theta_{-k})$
- Each conditional is assumed to be available in closed form. An example below:

Suppose

A 2-dim Gaussian

$$\theta \sim N_2(0, \Sigma) \qquad \Sigma = \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}$$

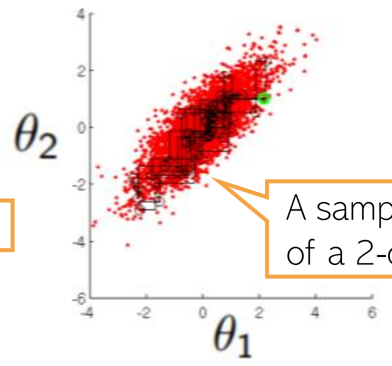
Then

$$\begin{aligned} \theta_1 | \theta_2 &\sim N\left(\rho\theta_2, [1-\rho^2]\right) \\ \theta_2 | \theta_1 &\sim N\left(\rho\theta_1, [1-\rho^2]\right) \end{aligned}$$

A 1-dim Gaussian

A 1-dim Gaussian

are the conditional distributions.



A sample based representation of a 2-dim Gaussian distribution

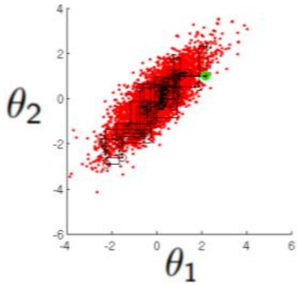
This example is a toy illustration of sampling from a 2-dim Gaussian by sampling from 1-dim Gaussians

Gibbs Sampling (Geman and Geman, 1982)

- Can be used to get a sampling-based approx. of a multiparam. posterior
- Gibbs sampler iteratively draws random samples from CPs
- When run long enough, the sampler produces samples from the joint posterior
- For the simple two-param case $\theta = (\theta_1, \theta_2)$, the Gibb sampler looks like this
  - Initialize $\theta_2^{(0)}$
  - For $s = 1,2,\ldots,S$
    - Draw a random sample for θ_1 as $\theta_1^{(s)} \sim p(\theta_1 | X, \theta_2^{(s-1)})$
    - Draw a random sample for θ_2 as $\theta_2^{(s)} \sim p(\theta_2 | X, \theta_1^{(s)})$

This CP uses the most recent value of θ_2

This CP uses the most recent value of θ_1



- These S random samples $\left(\theta_1^{(s)}, \theta_1^{(s)}\right)_{s=1}^{S}$ represent joint posterior $p(\theta_1, \theta_2 | X)$
- This is just a high-level idea. More on this when we discuss MCMC

Back to Bayesian Matrix Factorization

Bayesian Matrix Factorization: The CPs

- BMF with Gaussian likelihood and Gaussian prior on each user/item params is not fully conjugate but has local conjugacy
- To see this, note that the conditional posterior (CP) for user parameter $\boldsymbol{u}_i$

$$p(\boldsymbol{u}_i|\mathbf{R},\mathbf{V},\mathbf{U}_{-i}) \propto \prod_{j:(i,j)\in\Omega} p(r_{ij}|\boldsymbol{u}_i,\mathbf{v}_j)p(\boldsymbol{u}_i)$$

Only depends on the ratings of user $\boldsymbol{i}$. Also doesn't depend on $\mathbf{U}_{-i}$

This is due to the structure of the problem and the $\boldsymbol{u}_i$'s being independent a priori

$$= \prod_{j:(i,j)\in\Omega} \mathcal{N}(r_{ij}|\boldsymbol{u}_i^\top \mathbf{v}_j,\beta^{-1})\mathcal{N}(\boldsymbol{u}_i|\mathbf{0},\lambda_u^{-1}\mathbf{I}_K)$$

- The above is just like Bayesian linear regression, given $\boldsymbol{R}$ and fixed $\mathbf{V}$
  - Weight vector is $\boldsymbol{u}_i$, training data is $\{(\boldsymbol{v}_j, r_{ij})\}_{j:(i,j)\in\Omega}$, given
  - Also have local conjugacy since likelihood and prior are conjugate (assuming hyperparams fixed)
- Likewise, the CP for the item parameter $\boldsymbol{v}_j$ can be computed as

$$p(\mathbf{v}_j|\mathbf{R},\mathbf{U}) \propto \prod_{i:(i,j)\in\Omega} \mathcal{N}(r_{ij}|\boldsymbol{u}_i^\top \mathbf{v}_j,\beta^{-1})\mathcal{N}(\mathbf{v}_j|\mathbf{0},\lambda_v^{-1}\mathbf{I}_K)$$

Another Bayesian linear regression problem with weight vector $\boldsymbol{v}_j$

Bayesian Matrix Factorization: The CPs

- The CPs will have forms similar to solution of Bayesian linear regression

$$p(\boldsymbol{u}_i|\mathbf{R},\mathbf{V}) = \mathcal{N}(\boldsymbol{u}_i|\boldsymbol{\mu}_{u_i},\boldsymbol{\Sigma}_{u_i}) \qquad p(\boldsymbol{v}_j|\mathbf{R},\mathbf{U}) = \mathcal{N}(\boldsymbol{v}_j|\boldsymbol{\mu}_{v_j},\boldsymbol{\Sigma}_{v_j})$$

$$\boldsymbol{\Sigma}_{u_i} = (\lambda_u \mathbf{I} + \beta \textstyle\sum_{j:(i,j)\in\Omega} \boldsymbol{v}_j \boldsymbol{v}_j^\top)^{-1} \qquad \boldsymbol{\Sigma}_{v_j} = (\lambda_v \mathbf{I} + \beta \textstyle\sum_{i:(i,j)\in\Omega} \boldsymbol{u}_i \boldsymbol{u}_i^\top)^{-1}$$

$$\boldsymbol{\mu}_{u_i} = \boldsymbol{\Sigma}_{u_i}(\beta \textstyle\sum_{j:(i,j)\in\Omega} r_{ij} \boldsymbol{v}_j) \qquad \boldsymbol{\mu}_{v_j} = \boldsymbol{\Sigma}_{v_j}(\beta \textstyle\sum_{i:(i,j)\in\Omega} r_{ij} \boldsymbol{u}_i)$$

- These CPs can be updated in an alternating fashion until convergence
  - Many ways. One popular way is to use Gibbs sampling
  - Note: Hyperparameters can also be inferred by computing their CPs[1]
- Can extend Gaussian BMF easily to other exp. family distr. while maintaining local conj.
  - Example: Poisson likelihood and gamma priors on user/item parameters[2]

[1] "Bayesian Probabilistic Matrix Factorization using Markov Chain Monte Carlo" by Salakhutdinov and Mnih (2008)

[2] "Scalable recommendation with Poisson factorization" by Gopalan et al (2013)

BMF: Making Predictions

- PPD for each missing entry of the matrix (assuming hyperparams known)

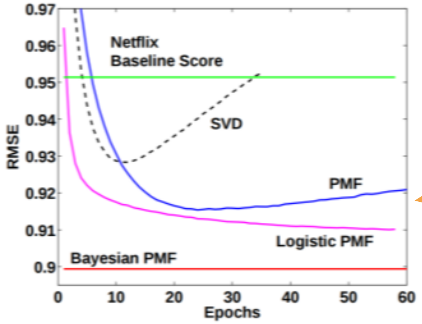
$$p(r_{ij}|\mathbf{R}) = \int\int p(r_{ij}|\boldsymbol{u}_i, \boldsymbol{v}_j)p(\boldsymbol{u}_i, \boldsymbol{v}_j|\mathbf{R})d\boldsymbol{u}_i d\boldsymbol{v}_j$$

- In general, this is intractable and needs approximation
  - If using Gibbs sampling, we can use S samples $(\boldsymbol{u}_i^{(s)}, \boldsymbol{v}_j^{(s)})_{s=1}^S$ to compute mean of r_{ij}
  - For the Gaussian likelihood case, the mean can be computed as

Can also compute the variance in the predicted r_{ij} using these S samples (think how)

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

- Comparison of Bayesian MF with others (from Salakhutdinov and Mnih (2008))



All other baselines are optimization based or point estimation based probabilistic models (PMF = probabilistic matrix factorization with point estimation)

Summary

- Local conjugacy is helpful even for complex prob. models with many params
  - CPs will have a closed form
  - Easy to implement Gibbs sampling can be used to get the (approx.) posterior
  - Many other approx. inference algos (like variational inference) benefit from local conjugacy
- Helps to choose likelihood and priors on each param as exp. family distr.
  - So if we can't get a globally conjugate model, we can still get a model with local conjugacy
- Even if we can't have local conjugacy, the notion of CPs is applicable
  - We can break the inference problem into estimating CPs (exactly if we have local conjugacy, or approximately if we don't have local conjugacy)
  - Almost all approx. algorithms work by estimating CPs exactly or approximately

Coming Up

- Latent Variable Models
- Expectation Maximization