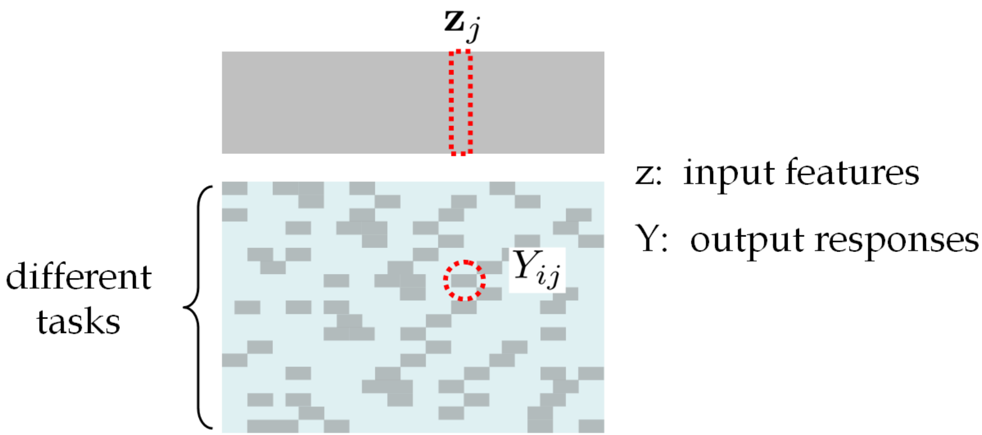
Large-scale Collaborative Prediction Using a Nonparametric Random Effects Model

Kai Yu†

Joint work with John Lafferty‡ and Shenghuo Zhu†

†NEC Laboratories America, ‡Carnegie Mellon University

Learning multiple tasks



- For input vector $\mathbf{z}_j$, and its outputs Y_{ij} under various conditions (tasks), the standard regression model is

$$Y_{ij} = \mu + m_i(\mathbf{z}_j) + \epsilon_{ij},$$

where $\epsilon_{ij} \overset{\text{iid}}{\sim} \mathrm{N}(0, \sigma^2)$, $i = 1, \ldots, M$, and $j = 1, \ldots, N$.

A kernel approach to multi-task learning

- To model the dependency between tasks, a hierarchical Bayesian approach may assume

$$m_i \overset{\text{iid}}{\sim} \mathrm{GP}(0, \Sigma)$$

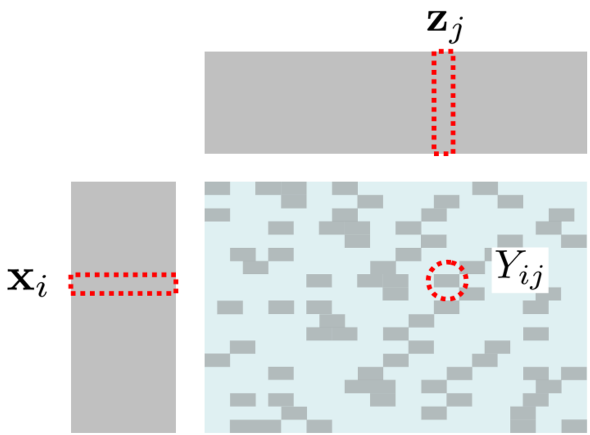
  where

  – $\Sigma(\mathbf{z}_j, \mathbf{z}_{j'}) \succ 0$ is a **shared covariance function among inputs**;

  – Many multi-task learning approaches are similar to this.[a]

[a] ICML-09 tutorial, Tresp & Yu

Using task-specific features



z: input features

x: task-specific features

Y: output responses

- Assuming **task-specific features** x are available, a more flexible approach is to model the data jointly, as

$$Y_{ij} = \mu + m(\mathbf{x}_i, \mathbf{z}_j) + \epsilon_{ij},$$

where $\epsilon_{ij} \overset{\text{iid}}{\sim} \mathrm{N}(0, \sigma^2)$, $m_{ij} = m(\mathbf{x}_i, \mathbf{z}_j)$ is a **relational function**.

A nonparametric kernel-based approach

- Assume the relational function follows[a]

$$m \sim \mathrm{GP}(0, \Omega \otimes \Sigma)$$

  where

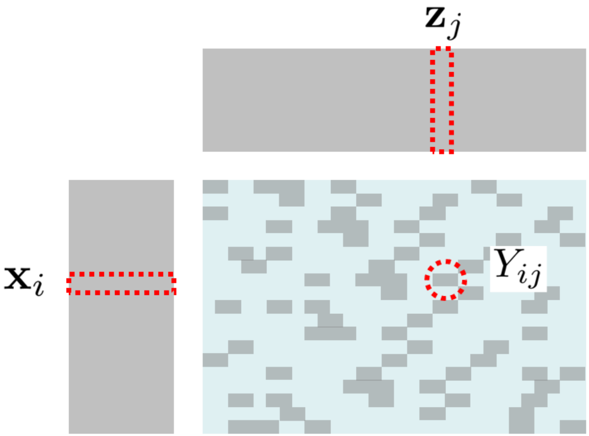
  - $\Omega(\mathbf{x}_i, \mathbf{x}_{i'})$ is a **covariance function on tasks**;
  - $\Sigma(\mathbf{z}_j, \mathbf{z}_{j'})$ is a **covariance function on inputs**;
  - any sub matrix follows

$$\mathbf{m} \sim \mathrm{N}(0, \mathbf{\Omega} \otimes \mathbf{\Sigma}) \;\; \Rightarrow \;\; \mathrm{Cov}(m_{ij}, m_{i'j'}) = \Omega_{ii'}\Sigma_{jj'};$$

  - If $\Omega = \delta$, the prior reduces to $m_i \overset{\text{iid}}{\sim} \mathrm{GP}(0, \Sigma)$.

[a] Yu et al., 2007; Bonilla et al., 2008

The collaborative prediction problem



z: movie features

x: user features

Y: ratings

- This essentially a multi-task learning problem with task features;
- Matrix factorization using additional row/column attributes;
- The formulation applies to many **relational prediction** problems.

Challenges to the kernel approach

- **Computation**: the cost $O(M^3N^3)$ is prohibitive.
  - Netflix data: $M = 480,189$ and $N = 17,770$.
- **Dependent "noise"**: when Y_{ij} cannot be fully explained by the predictors $\mathbf{x}_i$ and $\mathbf{z}_j$, the conditional independence assumption is invalid, which means,

$$\mathrm{p}(\mathbf{Y} \mid m, \mathbf{x}, \mathbf{z}) \neq \prod_{i,j} \mathrm{p}(Y_{ij}|m, \mathbf{x}_i, \mathbf{z}_j)$$

  - User and movie features are weak predictors;
  - The relational observations Y_{ij} alone are informative to each other.

This work

- Novel multi-task model using both input and task attributes;
- Nonparametric random effects to resolve dependent "noises";
- Efficient algorithm for large-scale collaborative prediction problems.

Nonparametric random effects

$$Y_{ij} = \mu + m_{ij} + f_{ij} + \epsilon_{ij},$$

- $m(\mathbf{x}_i, \mathbf{z}_j)$: a function depending on known attributes;
- f_{ij}: **random effects** for dependent "noises";
  - modeling dependency in observations with repeated structures.
- Let f_{ij} be **nonparametric**: dimensionality increases with data size;
  - "**nonparametric matrix factorization**"

Efficiency considerations in modeling

- To save computation, we absorb ϵ into f and obtain

$$Y_{ij} = \mu + m_{ij} + f_{ij},$$

- Introduce a special generative process for m and f ...

$$m, f \sim \cdot, \cdot | \Omega_0(\mathbf{x}_i, \mathbf{x}_{i'}), \Sigma_0(\mathbf{z}_j, \mathbf{z}_{j'}),$$

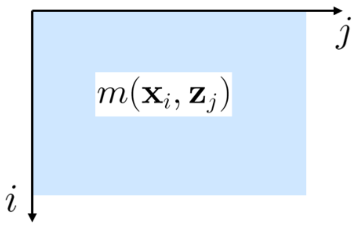
The row-wise generative model

$$\Sigma \sim \mathrm{IWP}(\kappa, \Sigma_0 + \lambda\delta)$$

The row-wise generative model

$$\Sigma \sim \mathrm{IWP}(\kappa, \Sigma_0 + \lambda\delta)$$

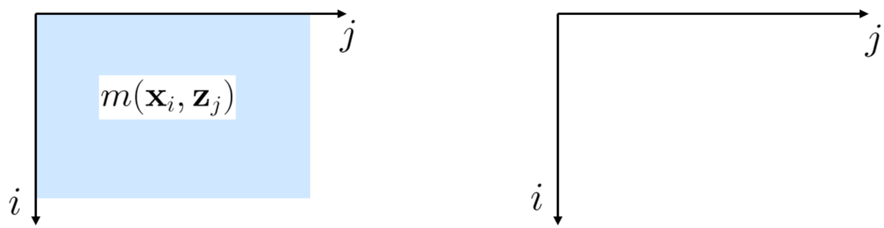
$$m \sim \mathrm{GP}(0, \Omega_0 \otimes \Sigma)$$



The row-wise generative model

$$\Sigma \sim \mathrm{IWP}(\kappa, \Sigma_0 + \lambda\delta)$$

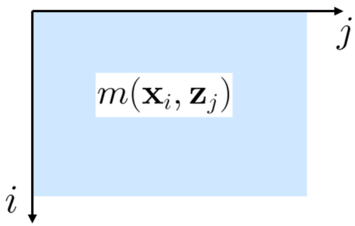
$$m \sim \mathrm{GP}(0, \Omega_0 \otimes \Sigma) \qquad f_i \overset{\text{iid}}{\sim} \mathrm{GP}(0, \tau\Sigma)$$



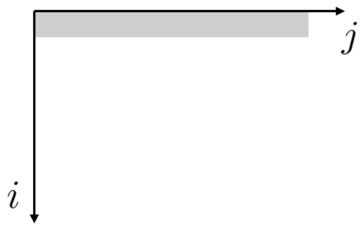
The row-wise generative model

$$\Sigma \sim \mathrm{IWP}(\kappa, \Sigma_0 + \lambda\delta)$$

$$m \sim \mathrm{GP}(0, \Omega_0 \otimes \Sigma)$$



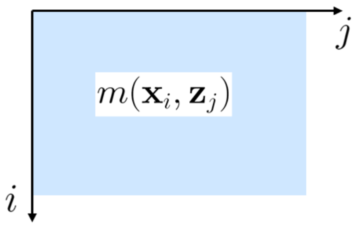
$$f_i \overset{\mathrm{iid}}{\sim} \mathrm{GP}(0, \tau\Sigma)$$

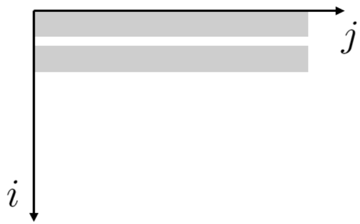


The row-wise generative model

$$\Sigma \sim \mathrm{IWP}(\kappa, \Sigma_0 + \lambda\delta)$$

$$m \sim \mathrm{GP}(0, \Omega_0 \otimes \Sigma) \qquad f_i \stackrel{\mathrm{iid}}{\sim} \mathrm{GP}(0, \tau\Sigma)$$

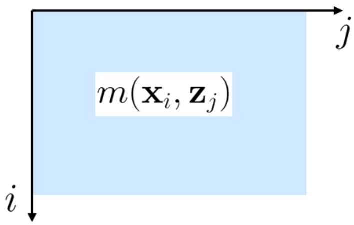


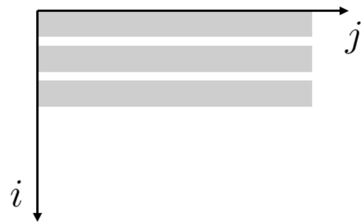


The row-wise generative model

$$\Sigma \sim \text{IWP}(\kappa, \Sigma_0 + \lambda\delta)$$

$$m \sim \text{GP}(0, \Omega_0 \otimes \Sigma) \qquad f_i \overset{\text{iid}}{\sim} \text{GP}(0, \tau\Sigma)$$

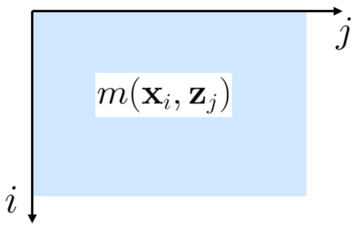




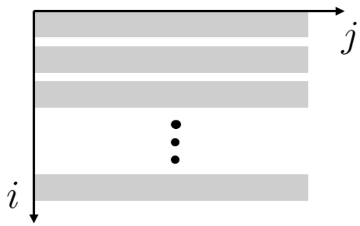
The row-wise generative model

$$\Sigma \sim \text{IWP}(\kappa, \Sigma_0 + \lambda\delta)$$

$$m \sim \text{GP}(0, \Omega_0 \otimes \Sigma)$$



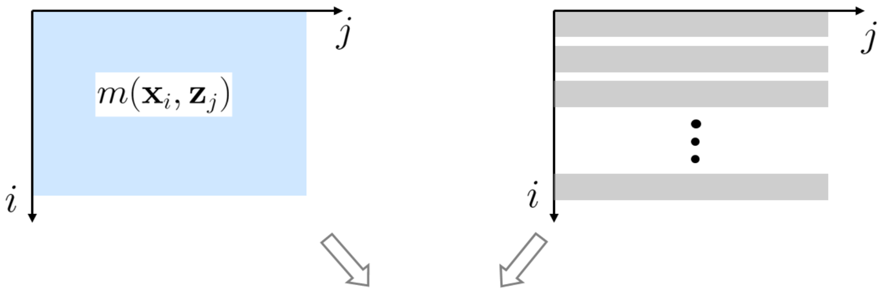
$$f_i \overset{\text{iid}}{\sim} \text{GP}(0, \tau\Sigma)$$



The row-wise generative model

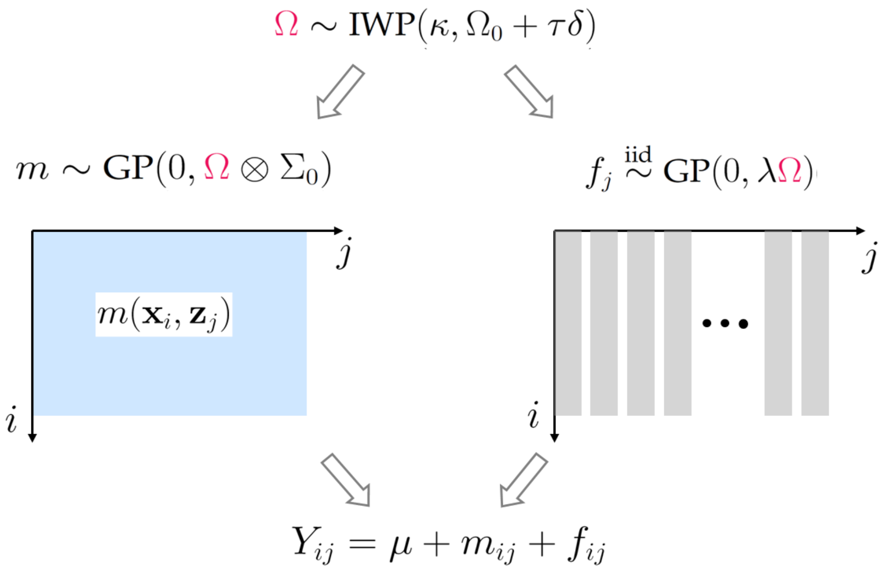
$$\Sigma \sim \mathrm{IWP}(\kappa, \Sigma_0 + \lambda\delta)$$

$$m \sim \mathrm{GP}(0, \Omega_0 \otimes \Sigma) \qquad f_i \overset{\mathrm{iid}}{\sim} \mathrm{GP}(0, \tau\Sigma)$$



$$Y_{ij} = \mu + m_{ij} + f_{ij}$$

The column-wise generative model



Two generative models

$$Y_{ij} = \mu + m_{ij} + f_{ij},$$

column-wise model:

$$\Omega \sim \mathrm{IWP}(\kappa, \Omega_0 + \tau\delta),$$
$$m \sim \mathrm{GP}(0, \Omega \otimes \Sigma_0),$$
$$f_j \overset{\text{iid}}{\sim} \mathrm{GP}(0, \lambda\Omega),$$

row-wise model:

$$\Sigma \sim \mathrm{IWP}(\kappa, \Sigma_0 + \lambda\delta),$$
$$m \sim \mathrm{GP}(0, \Omega_0 \otimes \Sigma),$$
$$f_i \overset{\text{iid}}{\sim} \mathrm{GP}(0, \tau\Sigma),$$

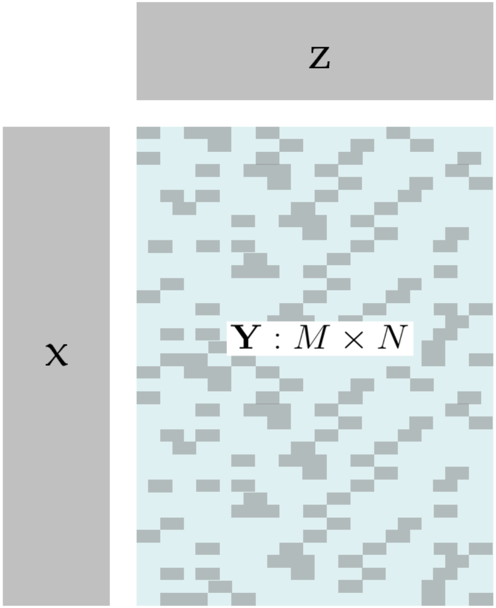
Two generative models are equivalent

- Both models lead to the same **matrix-variate Student-t process**

$$Y \sim \mathrm{MTP}\big(\kappa, 0, (\Omega_0 + \tau\delta), (\Sigma_0 + \lambda\delta)\big),$$

- The model **“learns” both Ω and Σ simultaneously**;
- Sometimes one model is computationally cheaper than the other.

An idea of large-scale modeling



Assuming $M \gg N$, we

- choose the row-wise model,
- let $\Omega_0(\mathbf{x}_i, \mathbf{x}_{i'})$ be low-rank.

$$\mathbf{Y} \sim \mathrm{MT}\big(\kappa, 0, (\mathbf{\Omega}_0 + \tau \mathbf{I}_M), (\mathbf{\Sigma}_0 + \lambda \mathbf{I}_N)\big),$$

Modeling large-scale data

- If $\Omega_0(\mathbf{x}_i, \mathbf{x}_{i'}) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_{i'}) \rangle$, $m \sim \mathrm{GP}(0, \Omega_0 \otimes \Sigma)$ implies

$$m_{ij} = \langle \phi(\mathbf{x}_i), \beta_j \rangle$$

- Without loss of generality, let $\Omega_0(\mathbf{x}_i, \mathbf{x}_{i'}) = \langle p^{\frac{1}{2}}\mathbf{x}_i, p^{\frac{1}{2}}\mathbf{x}_{i'} \rangle$, $\mathbf{x}_i \in \mathbb{R}^p$. On a finite observational matrix $\mathbf{Y} \in \mathbb{R}^{M \times N}$, $M \gg N$, the row-wise model becomes

$$\begin{aligned} \boldsymbol{\Sigma} &\sim \mathrm{IW}(\kappa, \boldsymbol{\Sigma}_0 + \lambda \mathbf{I}_N), \\ \boldsymbol{\beta} &\sim \mathrm{N}(0, \mathbf{I}_p \otimes \boldsymbol{\Sigma}), \\ \mathbf{Y}_i &\sim \mathrm{N}(\boldsymbol{\beta}^\top \mathbf{x}_i, \tau \boldsymbol{\Sigma}), \quad i = 1, \ldots, M \end{aligned}$$

where $\boldsymbol{\beta}$ is $p \times N$ random matrix.

Approximate Inference - EM

- E-step: compute the sufficient statistics $\{\boldsymbol{v}_i, \mathbf{C}_i\}$ for the posterior of $\mathbf{Y}_i$ given the current $\boldsymbol{\beta}$ and $\boldsymbol{\Sigma}$:

$$Q(\mathbf{Y}) = \prod_{i=1}^{M} \mathrm{p}(\mathbf{Y}_i | \mathbf{Y}_{O_i}, \boldsymbol{\beta}, \boldsymbol{\Sigma}) = \prod_{i=1}^{M} \mathrm{N}(\mathbf{Y}_i | \boldsymbol{v}_i, \mathbf{C}_i),$$

- M-step: optimize $\boldsymbol{\beta}$ and $\boldsymbol{\Sigma}$:

$$\widehat{\boldsymbol{\beta}}, \widehat{\boldsymbol{\Sigma}} = \arg\min_{\boldsymbol{\beta}, \boldsymbol{\Sigma}} \left\{ \mathbb{E}_{Q(\mathbf{Y})} \left[- \log \mathrm{p}(\mathbf{Y}, \boldsymbol{\beta}, \boldsymbol{\Sigma} | \theta) \right] \right\}$$

and then let $\boldsymbol{\beta} \leftarrow \widehat{\boldsymbol{\beta}}, \boldsymbol{\Sigma} \leftarrow \widehat{\boldsymbol{\Sigma}}$.

Some notation

- Let $J_i \subset \{1, \ldots, N\}$ be the index set of the N_i observed elements in the row $\mathbf{Y}_i$;
- $\boldsymbol{\Sigma}_{[:,J_i]} \in \mathbb{R}^{N \times N_i}$ is the matrix obtained by keeping the columns of $\boldsymbol{\Sigma}$ indexed by J_i;
- $\boldsymbol{\Sigma}_{[J_i,J_i]} \in \mathbb{R}^{N_i \times N_i}$ is obtained from $\boldsymbol{\Sigma}_{[:,J_i]}$ by further keeping only the rows indexed by J_i;
- Similarly, we can define $\boldsymbol{\Sigma}_{[J_i,:]}$, $\mathbf{Y}_{[i,J_i]}$ and $\mathbf{m}_{[i,J_i]}$.

The EM algorithm

- E-step: for $i = 1, \ldots, M$

$$\mathbf{m}_i = \boldsymbol{\beta}^\top \mathbf{x}_i,$$
$$\boldsymbol{\upsilon}_i = \mathbf{m}_i + \boldsymbol{\Sigma}_{[:,J_i]}\boldsymbol{\Sigma}^{-1}_{[J_i,J_i]}(\mathbf{Y}_{[i,J_i]} - \mathbf{m}_{[i,J_i]})^\top,$$
$$\mathbf{C}_i = \tau\boldsymbol{\Sigma} - \tau\boldsymbol{\Sigma}_{[:,J_i]}\boldsymbol{\Sigma}^{-1}_{[J_i,J_i]}\boldsymbol{\Sigma}_{[J_i,:]}.$$

- M-step:

$$\widehat{\boldsymbol{\beta}} = (\mathbf{x}^\top\mathbf{x} + \tau\mathbf{I}_p)^{-1}\mathbf{x}^\top\boldsymbol{\upsilon},$$
$$\widehat{\boldsymbol{\Sigma}} = \frac{\tau^{-1}\left[\sum_{i=1}^{M}(\mathbf{C}_i + \boldsymbol{\upsilon}_i\boldsymbol{\upsilon}_i) - \boldsymbol{\upsilon}^\top\mathbf{x}(\mathbf{x}^\top\mathbf{x} + \tau\mathbf{I}_p)^{-1}\mathbf{x}^\top\boldsymbol{\upsilon}\right] + \boldsymbol{\Sigma}_0 + \lambda\mathbf{I}_N}{M + 2N + p + \kappa}$$

On Netflix, each EM iteration takes **several thousands of hours** .

Fast implementation

- Let $\mathbf{U}_i \in \mathbb{R}^{N \times N_i}$ be a **column selection operator**, such that $\mathbf{\Sigma}_{[:,J_i]} = \mathbf{\Sigma}\mathbf{U}_i$ and $\mathbf{\Sigma}_{[J_i,J_i]} = \mathbf{U}_i^\top \mathbf{\Sigma}\mathbf{U}_i$.

- The M-step only needs $\mathbf{C} = \sum_{i=1}^M \mathbf{C}_i + \boldsymbol{\upsilon}^\top \boldsymbol{\upsilon}$ and $\boldsymbol{\upsilon}^\top \mathbf{x}$ from the previous E-step. To obtain them, it's **unnecessary to compute $\boldsymbol{\upsilon}_i$ and $\mathbf{C}_i$**. For example,

$$
\begin{aligned}
\sum_{i=1}^M \mathbf{C}_i &= \sum_{i=1}^M \left(\tau \mathbf{\Sigma} - \tau \mathbf{\Sigma}_{[:,J_i]} \mathbf{\Sigma}_{[J_i,J_i]}^{-1} \mathbf{\Sigma}_{[J_i,:]}\right) \\
&= \sum_{i=1}^M \left(\tau \mathbf{\Sigma} - \tau \mathbf{\Sigma} \mathbf{U}_i \mathbf{\Sigma}_{[J_i,J_i]}^{-1} \mathbf{U}_i^\top \mathbf{\Sigma}\right) \\
&= \tau M \mathbf{\Sigma} - \tau \mathbf{\Sigma} \left(\sum_{i=1}^M \mathbf{U}_i \mathbf{\Sigma}_{[J_i,J_i]}^{-1} \mathbf{U}_i^\top \right) \mathbf{\Sigma}
\end{aligned}
$$

- Similar tricks can be applied to $\boldsymbol{\upsilon}^\top \boldsymbol{\upsilon}$ and $\boldsymbol{\upsilon}^\top \mathbf{x}$. Time for each iteration is reduced from **thousands of hours** to **5 hours only**.

EachMovie Data

- 74424 users, 1648 movies;
- $2,811,718$ numeric ratings $Y_{ij} \in \{1, \ldots, 6\}$;
- 97.17% of the elements are missing;
- Use 80% ratings of each user for training and the rest for testing;
- This random selection is repeated 10 times independently.

Compared methods

- **User Mean** & **Movie Mean**: prediction by the empirical mean;
- **FMMMF**: fast max-margin matrix factorization [a];
- **PPCA**: probabilistic principal component analysis [b];
- **BSRM**: Bayesian stochastic relational model [c], BSRM-1 uses no additional user/movie attributes [d];
- **NREM**: Nonparametric random effects model, NREM-1 uses no additional attributes.

[a] Rennie & Srebro (2005).
[b] Tipping & Bishop (1999).
[c] Zhu, Yu, & Gong (2009).
[d] Top 20 eigenvectors from of binary matrix indicating if a rating is observed or not.

Results on EachMovie

TABLE: Prediction Error on EachMovie Data

Method	RMSE	Standard Error	Run Time (hours)
User Mean	1.4251	0.0004	
Movie Mean	1.3866	0.0004	
FMMMF	1.1552	0.0008	4.94
PPCA	1.1045	0.0004	1.26
BSRM-1	1.0902	0.0003	1.67
BSRM-2	1.0852	0.0003	1.70
NREM-1	**1.0816**	0.0003	**0.59**
NREM-2	**1.0758**	0.0003	**0.59**

Netflix Data

- $100,480,507$ ratings from $480,189$ users on $17,770$ movies;
- $Y_{ij} \in \{1,2,3,4,5\}$;
- A set of validation data contain $1,408,395$ ratings;
- Therefore there are 98.81% of elements missing in the rating matrix;
- The test set includes $2,817,131$ ratings;

Compared methods

In addition to those compared in EachMovie experiment, there are several other methods:

- **SVD**: a method almost the same as FMMMF, using a gradient-based method for optimization [a].
- **RBM**: Restricted Boltzmann Machine using contrast divergence [b].
- **PMF** and **BPMF**: probabilistic matrix factorization [c], and its Bayesian version [d].
- **PMF-VB**: probabilistic matrix factorization using a variational Bayes method for inference [e].

---

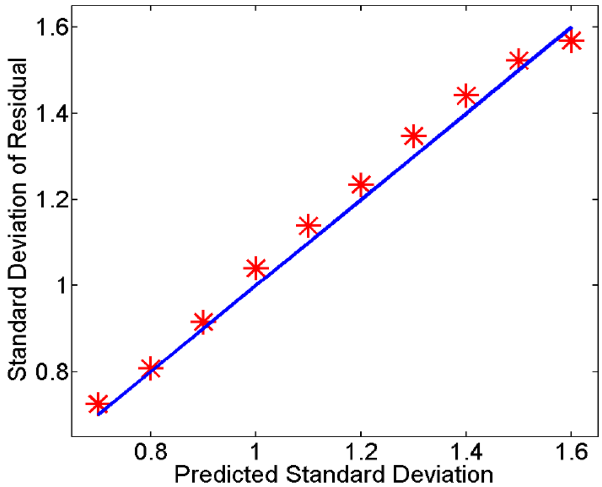
[a] Kurucz, Benczur, & Csalogany, (2007).
[b] Salakhutdinov, Mnih & Hinton (2007).
[c] Salakhutdinov & Mnih (2008b).
[d] Salakhutdinov & Mnih (2008a).
[e] Lim & Teh (2007).

Results on Netflix

TABLE: Performance on Netflix Data

Method	RMSE	Run Time (hours)
Cinematch	0.9514	-
SVD	0.920	300
PMF	0.9265	-
RBM	0.9060	-
PMF-VB	0.9141	-
BPMF	0.8954	1100
BSRM-2	0.8881	350
NREM-1	**0.8876**	**148**
NREM-2	**0.8853**	**150**

Predictive Uncertainty



Standard deviations of prediction residuals vs. standard deviations predicted by our model on EachMovie

Related work

- Multi-task learning using Gaussian processes, those learn the covariance Σ shared across tasks [a], and those that additionally consider the covariance Ω between tasks [b]
- Application of GP models to collaborative filtering [c]
- Low-rank matrix factorization, e.g., [d]. Our model is nonparametric in the sense no rank constraint is imposed.
- Very few matrix factorization methods use known predictors. One such a work [e] introduces low-rank multiplicative random effects in modeling networked observations.

[a] Lawrence & Platt (2004); Schwaighofer, Tresp & Yu (2004); Yu, Tresp & Schwaighofer (2005).
[b] Yu, Chu, Yu, Tresp, & Xu, (2007); Bonilla, Chai, & Williams (2008).
[c] Schwaighofer, Tresp & Yu (2004), Yu & Chu (2007)
[d] Salakhutdinov & Mnih (2008b).
[e] Hoff (2005)

Summary

- The model provides a novel way to use random effects and known attributes to explain the complex dependence of data;
- We make the nonparametric model scalable and efficient on very large-scale problems;
- Our experiments demonstrate that the algorithm works very well on two challenging collaborative prediction problems;
- In the near future, it will be promising to perform a full Bayesian inference by a parallel Gibbs sampling method.