# A Bayesian Matrix Model for Relational Data

**Ajit P. Singh**
Machine Learning Department
Carnegie Mellon University
Pittsburgh, PA 15213, USA
ajit@cs.cmu.edu

**Geoffrey J. Gordon**
Machine Learning Department
Carnegie Mellon University
Pittsburgh, PA 15213, USA
ggordon@cs.cmu.edu

## Abstract

Relational learning can be used to augment one data source with other correlated sources of information, to improve predictive accuracy. We frame a large class of relational learning problems as matrix factorization problems, and propose a hierarchical Bayesian model. Training our Bayesian model using random-walk Metropolis-Hastings is impractically slow, and so we develop a block Metropolis-Hastings sampler which uses the gradient and Hessian of the likelihood to dynamically tune the proposal. We demonstrate that a predictive model of brain response to stimuli can be improved by augmenting it with side information about the stimuli.

## 1 Introduction

In attribute-value learning, entities are represented by a predetermined set of features, or *attributes*. Entities are mapped to records, or assignments to attributes, and are assumed to be fully described by their attributes (e.g., exchangeable records). In relational learning, objects have both attributes and *relations*, properties involving sets of attributes. We take a broad view of relations, as functional mappings from a set of two or more entities to a subset of the real numbers. Relations correlate entities, and we seek to exploit these correlations to improve the quality of models that predict the value of unobserved relations.[1]

**Target application for relational learning**. Functional Magnetic Resonance Imaging (fMRI) is often used to measure responses in small regions of the brain (i.e., voxels) given external stimuli. Given enough experiments, on a sufficiently broad range of stimuli, one can build models that predict patterns of brain activation given new stimuli [4]. The fMRI data can be viewed as a relation, $\texttt{Response}(stimulus, voxel) \in [0, 1]$, measuring the response in a region of the brain under a particular stimulus, averaged over all patients. Running enough experiments is costly, but we can often collect cheap side information about the stimuli. In this paper, we consider an experiment where the stimulus is a word-picture pair displayed on a screen. We can collect statistics of whether the stimulus word co-occurs with other commonly used words in large, freely available text corpora. The stimulus side-information can be viewed as a relation $\texttt{Co-occurs}(word, stimulus) \in \{0, 1\}$, measuring whether or not a word co-occurs near a stimulus word in the corpus. Both relations provide information about the same stimuli, and we seek to improve the quality of a predictive model of brain activity, the `Response` relation, using word co-occurrence data, the `Co-occurs` relation.

## 2 Collective Matrix Factorization

*We introduce the idea of modeling certain kinds of relational data using sets of matrices, and frame relational learning as a form of low-rank matrix factorization with parameter tying.*

[1] An extended version of this paper appears in Singh [9].

*The maximum likelihood objective is not convex, but has convex substructure, which we later use to guide a Metropolis-Hastings sampler for Bayesian inference.*

An arity-two relation can be represented as a matrix, and sets of correlated relations can be represented as sets of matrices which share dimensions. In our fMRI example, the `Co-occurs` relation is represented as a $m \times n$ matrix $X$; the `Response` relation as a $n \times r$ matrix $Y$. We embed the entities in a $k$-dimensional space, by factoring an indirect representation of each matrix, $X \approx f(UV^T)$ and $Y \approx g(VZ^T)$. Thus, $U$ is a $m \times k$ factor representing words, $Z$ is a $r \times k$ factor representing voxels, and $V$ is a $n \times k$ factor representing stimuli. $f$ and $g$ are link functions (Section 2). Given parameters $\mathcal{F} = \{U, V, Z\}$, and data $\mathcal{D} = \{X, Y\}$, the likelihood of each matrix is

$$p(X \mid U, V) = \prod_{i=1}^{m} \prod_{j=1}^{n} \left[ p_X \left( X_{ij} \mid U_{i\cdot} V_{j\cdot}^T \right) \right]^{W_{ij}}, \tag{1}$$

$$p(Y \mid V, Z) = \prod_{j=1}^{n} \prod_{r=1}^{r} \left[ p_Y \left( Y_{jr} \mid V_{j\cdot} Z_{r\cdot}^T \right) \right]^{\tilde{W}_{jr}}. \tag{2}$$

The per-entry distributions $p_X$ and $p_Y$ are one-parameter exponential families with natural parameter $U_{i\cdot}V_{j\cdot}^T$ and $V_{j\cdot}Z_{r\cdot}^T$, respectively. The modeler chooses $p_X$ and $p_Y$, and they need not be from the same exponential family; this allows us to integrate relations with different response types: e.g., `Co-occurs` is well-modelled by the Bernoulli distribution, but `Response` is better modelled by a Gaussian. The fixed weights $W_{ij} \in \{0, 1\}$ and $\tilde{W}_{jr} \in \{0, 1\}$ allow for missing data: set a weight to zero when the corresponding value in the data matrix is unobserved. Maximizing Equation 1 is a weighted version of Exponential Family PCA [2].

Maximizing the product of Equations 1 and 2 with respect to the factors $\mathcal{F}$ is an example of collective matrix factorization [10]. Given that the number of parameters grow with the data, we place a multivariate Gaussian prior on each row of U:

$$p(U \mid \Theta_U) = \prod_{i=1}^{m} \mathcal{N}(U_{i\cdot} \mid \mu_U, \Sigma_U), \tag{3}$$

where $\mathcal{N}(\cdot \mid \mu_U, \Sigma_U)$ is a Gaussian with mean vector $\mu_U$ and covariance matrix $\Sigma_U$. We assume that $\Theta_U = \{\mu_U, \Sigma_U\}$ is known. The priors over $V$ and $Z$ are defined similarly, with $\Theta = \{\Theta_U, \Theta_V, \Theta_Z\}$. Equations 1–3 define the posterior distribution

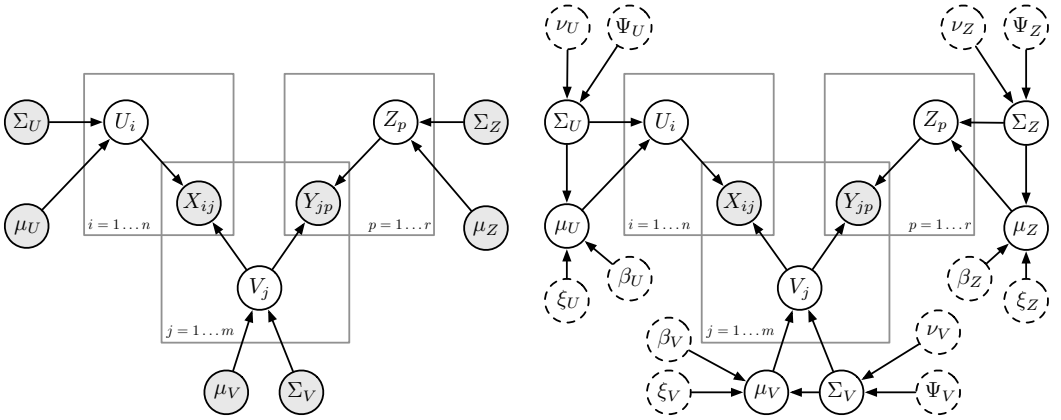
$$p(U, V, Z \mid X, Y, \Theta) = c \cdot p(X \mid U, V, W)\, p(Y \mid V, Z, \tilde{W})\, p(U \mid \Theta_U)\, p(V \mid \Theta_V)\, p(Z \mid \Theta_Z), \tag{4}$$

where $c$ is a normalizing constant. Figure 1(a) is a plate model representation of Equation 4. Maximum a posteriori inference (a.k.a. regularized maximum likelihood) involves searching for the parameters which minimize the negative log-posterior

$$\mathcal{L} = -\log p(U, V, Z \mid X, Y, W, \tilde{W}, \Theta). \tag{5}$$

The prediction link $f$ is equivalent to the choice of $p_X$: $E[X_{ij}] = f(U_{i\cdot}V_{j\cdot}^T)$ where $f$ is the derivative of the log-partition function of the exponential family of $p_X$ (likewise $g$ and $p_Y$). It can be shown that $\mathcal{L}$, a high-dimensional non-convex function, is *componentwise convex*: i.e., convex in one low-rank factor when the others are fixed.

Componentwise convexity leads to an elegant algorithm for maximum a posteriori estimation. Consider the graphical model form of collective matrix factorization (Figure 1(a)). Using $d$-separation [5], it is easy to deduce that if only one factor is free (say $U$) then the rows of that factor are independent of one other. Therefore, the projection over a large factor matrix can be reduced into parallel optimizations over each row of that factor. Each row of a factor has only $k \ll \min\{m, n, r\}$ parameters, and so both the gradient and Hessian may be used to minimize $\mathcal{L}$ with respect to a factor row. We call this approach alternating Newton-projections. The per-row optimization is convex. In Section 4, the same decomposition into per-row updates leads to a block Metropolis-Hastings sampler, where the gradient and Hessian computed here are used to select the proposal distribution.




(a) Collective Matrix Factorization

(b) Hierarchical (and Hierarchical Bayesian) Collective Matrix Factorization

Figure 1: Plate graphical models for collective matrix factorization (Section 2) and its analogue with hierarchical priors (Section 3). Shaded nodes indicate known quantities; dashed nodes are fixed parameters in the hyperprior. Weight matrices $W$ and $\tilde{W}$ are elided.

## 3 Hierarchical Collective Matrix Factorization

*Here, we discuss the modeling limitations of Collective Matrix Factorization, and propose solving them through hierarchical priors.*

Collective Matrix Factorization requires choosing a good fixed value for the hyperparameters $\Theta = \{\Theta_U, \Theta_V, \Theta_Z\}$, where $\forall F \in \{U, V, Z\}, \Theta_F = \{\mu_F, \Sigma_F\}$. Finding a good value for $\Theta$ is a difficult task, even if we assume that the prior means are zero and the covariances spherical.

Another concern is that information between entities can only be shared indirectly, through another factor: e.g., in $f(UV^T)$, two distinct rows of $U$ are correlated only through $V$. Computationally, the independence of rows in the free factor is useful. Statistically, we want a more direct way of pooling shared behaviour among entities. Both concerns can be addressed by extending Collective Matrix Factorization (Figure 1(a)) to include hierarchical priors (Figure 1(b)), as discussed below.

Since we do not know $\Theta$, we place a weak prior on it and treat it as a quantity to be learned: i.e., a hierarchical model. We design the hierarchical prior so that independence of factor rows is preserved under alternating Newton-projections. We place separate hierarchical priors on $\Theta_U$, $\Theta_V$, and $\Theta_Z$, electing to use the conjugate prior for Gaussian parameters: the normal-Inverse-Wishart distribution [3]. The normal-Inverse-Wishart prior on $\Theta_F = \{\mu_F, \Sigma_F\}$ is defined by first sampling the covariance from a Wishart distribution, $\mathcal{W}$, then conditionally sampling the mean from a Gaussian distribution, $\mathcal{N}$:

$$\Sigma_F^{-1} \sim \mathcal{W}(\nu_F, \Psi_F),$$
$$\mu_F \,|\, \Sigma_F \sim \mathcal{N}(\xi_F, \Sigma_F/\beta_0).$$

The fixed hyperprior parameters $\nu_F > k$, $\Psi_F \in \mathbb{R}_+^{k \times k}$, $\xi_F \in \mathbb{R}^k$, $\beta_0 > 0$ are chosen by the modeler.[2] Our choice of hierarchical priors for matrix factorization is identical to that of Salakhutdinov and Mnih [7], though we do not make the restrictive simplifying assumption that the likelihood is conjugate to the hierarchical prior.

The hierarchical prior acts as a shrinkage estimator for the rows of a factor, pooling information indirectly, through $\Theta$. Shrinkage may be especially useful when some entities are associated with only a few observations: in the absence of data, the low-rank representation of an entity tends towards the population mean.

[2] In all our experiments, for each factor $F$, $\nu_F = k$ is the embedding dimension, $\Psi_F$ is a $k \times k$ identity matrix, $\xi_U = 0$, and $\beta_0 = 1$.

Maximum a posteriori estimation of $\{\mathcal{F}, \Theta\}$ is straightforward: alternate between optimizing $\mathcal{F}$ given fixed $\Theta$, and optimizing $\Theta$ given fixed $\mathcal{F}$. Given fixed $\mathcal{F}$, the most likely value of $(\mu_F, \Sigma_F)$ is the mode of a normal-Inverse-Wishart distribution with parameters

$$\bar{F} = \frac{1}{n_F}\sum_{i=1}^{n_F} F_{i\cdot}, \quad S_F = \sum_{i=1}^{n_F}(F_{i\cdot} - \bar{F})^T(F_{i\cdot} - \bar{F}), \quad \xi_F^* = \frac{n_F}{\beta_F + n_F}\bar{F} + \frac{\beta_F \xi_F}{\beta_F + n_F}, \tag{6}$$

$$\Psi_F^* = \Psi_F^{-1} + S_F + \frac{\beta_F n_F}{\beta_F + n_F}(\bar{F} - \xi_F)^T(\bar{F} - \xi_F), \quad \nu_F^* = \nu_F + n_F, \quad \beta_F^* = \beta_F + n_F.$$

The fixed hyperprior parameters are referred to as

$$\Theta_0 = \{\nu_U, \Psi_U, \xi_U, \nu_V, \Psi_V, \xi_V, \nu_Z, \Psi_Z, \xi_Z, \beta_0\}.$$

Factoring the posterior distribution over $\{\mathcal{F}, \Theta\}$, according to the hierarchical model, yields the following objective for maximum a posteriori:

$$\mathcal{O} \quad = \quad \mathcal{L} + \sum_{F \in \mathcal{F}} \log p(\Theta_F \,|\, \Theta_0). \tag{7}$$

## 4 Bayesian Inference for the Hierarchical Model

*We propose a block Metropolis-Hastings algorithm for sampling from the model in Figure 1(b). Random-walk proposals would take too long to mix, so we develop an adaptive proposal distribution that uses convex substructure in matrix factorizations. The result is an fast Bayesian algorithm with no proposal tuning, and no restrictive conjugacy assumptions.*

Thus far we have discussed only maximum a posteriori (point) estimation, but there may be substantial posterior uncertainty in $\{\mathcal{F}, \Theta\}$, especially when each entity participates only in a few relationships. The Frequentist argument of asymptotic consistency does not hold for matrix factorization models: the number of parameters grow with the size of $X$ and $Y$.

Point estimation tends to perform poorly when predicting the behaviour of new entities, new rows or columns in the data matrices, which did not appear in the training data—Welling et al. [11] provides a compelling theoretical justification for this behaviour.

Finally, point estimation is limited in how it can exploit correlations between the data matrices. Consider Collective Matrix Factorization on the three entity-type example, where the posterior distribution is $p(U, V, Z \,|\, X, Y, \Theta)$. The analogous posterior involving only the $X$ data matrix is $p(U, V \,|\, X, \Theta_U, \Theta_V)$. In each case we can compute the marginal distribution of an element of a factor, say $U_{i\ell} \in U$. If we compare the posterior distribution over $U_{i\ell}$ in the single and two-matrix cases, it is clear that the two distributions $p(U_{i\ell} \,|\, X, Y, \Theta)$ and $p(U_{i\ell} \,|\, X, \Theta_U, \Theta_V)$ can, and usually will, differ. Under maximum a posteriori inference, the only difference we see between the two distributions is a difference in the mode: changes in the variance, skew, and other properties of the distribution are not accounted for. Changes in the posterior mode account for some of the effect of information sharing between matrices; changes in the posterior distribution account for all of the effect.

A fully Bayesian approach to training the hierarchical model introduced in Section 3 addresses the aforementioned concerns. Instead of approximating the posterior $p(\mathcal{F}, \Theta \,|\, \mathcal{D}, \Theta_0)$ by its mode, we approximate it using samples drawn from it.

### 4.1 Block Metropolis-Hastings

In a block Metropolis-Hastings sampler we partition the unknowns $\Omega = \{\mathcal{F}, \Theta\}$ into subsets (blocks) of correlated variables, cyclically sampling from each block given that the others are fixed. We choose the following blocks for Metropolis-Hastings:

$$\forall F \in \{U, V, Z\}\ \forall i = 1 \ldots n_F : \ F_{i\cdot} \sim p(F_i \,|\, \Omega - F_{i\cdot}), \tag{8}$$

$$\forall F \in \{U, V, Z\} : \ \Theta_F \sim p(\Theta_F \,|\, \Omega - \Theta_F). \tag{9}$$

The way the parameters are grouped in block Metropolis-Hastings is similar to that of alternating Newton-projections: sample the hyperparameters, then sample each factor by

sampling each row of the factor in parallel. Equation 9 is simply sampling from a normal-Inverse-Wishart; Equation 8 cannot be sampled from directly, unless we assume that $X$ and $Y$ are Gaussian—i.e., that the data distributions and priors are mutually conjugate. In the single matrix case, with $p_X$ Gaussian, the model is that of Salakhutdinov and Mnih [8].

### 4.2 Hessian Metropolis-Hastings

In our motivating fMRI example, the entries of different data matrices have different response types: binary word co-occurrence, real-valued voxel responses. This flexibility in response type significantly improves predictive accuracy (see Section 5). We do not wish to sacrifice the flexibility of our model, which supports multiple response types, to reap the benefits of Bayesian inference. Instead of assuming that $p_X$ and $p_Y$ are Gaussian, and resorting to Gibbs sampling; we allow $p_X$ and $p_Y$ to be any rank-one exponential family distribution, and consider Metropolis-Hastings.

In Metropolis-Hastings, one often resorts to sampling from a Gaussian proposal distribution whose mean is the sample at time $t$, $F_{i\cdot}^{(t)}$, with covariance matrix $v_i \cdot I$. The user must choose $v_i$, for each row, such that the Markov chain mixes quickly. Tuning one proposal distribution is tedious; tuning a proposal distribution for each entity is masochistic. If the Hessian of the target distribution, with respect to $F_{i\cdot}^{(t)}$, is far from spherical, then the rate at which the underlying Markov chain mixes can be slow, regardless of how $v_i$ is tuned.

The distribution in Equation 8 may not be easy to sample from; but given a point, namely $F_{i\cdot}^{(t)}$, we can easily compute the local gradient and Hessian of the distribution with respect to $F_{i\cdot}$. By using the gradient and Hessian, we can create a proposal distribution that better approximates $p(F_{i\cdot} \mid \Omega - F_{i\cdot})$.

Once one realizes that $p(F_{i\cdot} \mid \Omega - F_{i\cdot})$ is the likelihood of a Bayesian Generalized Linear Model, we can use what we know about efficient inference in Bayesian GLMs to accelerate sampling from Equation 4. A contribution of this work is the insight that we can use Hessian Metropolis-Hastings (HMH) [6] in Bayesian matrix factorization. HMH uses both the gradient and Hessian to automatically construct a proposal distribution at each sampling step. To define a Metropolis-Hastings sampler, we need to define the forward sampling distribution, from which a proposal value $F_{i\cdot}^{(*)}$ is drawn given the previous value in the chain, $F_{i\cdot}^{(t)}$. The choice of a forward sampling distribution leads to a corresponding backward sampling distribution, which defines the probability of returning to $F_{i\cdot}^{(t)}$ from the proposal value $F_{i\cdot}^{(*)}$. The forward sampling distribution in HMH is a Gaussian whose mean is determined by taking one Newton step from $F_{i\cdot}^{(t)}$, and whose covariance is derived from the Hessian used to take the Newton step. The backward sampling distribution in HMH is a Gaussian whose mean is determined by taking one Newton step from $F_{i\cdot}^{(*)}$, and whose covariance is derived from the Hessian used to take the Newton step. Instead of searching over step lengths $\eta$, we sample from a fixed distribution over step lengths (here, uniformly at random). Algorithms 2 and 3 describe the Hessian Metropolis-Hastings sampler for Equation 8.[3] Since we have a efficient Newton-projection for finding the mode of $p(F_{i\cdot} \mid \Omega - F)$, we should use it to pick a proposal that is closer to the mode. After all, for any smooth distribution on $F_{i\cdot}$, the mode is in a region of high probability.

### 4.3 Generalization to an Arbitrary Number of Relations

Algorithms 1-3 are presented for the general case, which can involve any number of related matrices. Here, we describe the general notation. Entity-types, the different kinds of relation arguments, are indexed by $i = 1 \ldots t$. The number of entities of type $i$ in the training set is denoted $n_i$. A matrix corresponding to a relation between entity-types $i$ and $j$ is denoted $X^{(ij)}$. Each relation matrix is represented as the product of low-rank factors:

$$X^{(ij)} \approx f^{(ij)}\left(U^{(i)}\left(U^{(j)}\right)^T\right),$$

[3]The extended notation used in Algorithms 1–3 are described in Section 4.3.

**Algorithm 1**: Decomposition Algorithm for Maximum a Posteriori and Bayesian Inference

**Input**: Data matrices, $\{X^{(ee')}\}$. The model (embedding dimension $k$, the choice of exponential family for each matrix, and the hyperparameters $\Theta$, if they are fixed).
**Output**: Low rank factors for each entity-type: $U^{(1)} \dots U^{(E)}$.
**for** $e = 1 \dots E$ **do**
  Initialize $U^{(e,0)}$ using Algorithm 2 with prior mean $\mu_e = 0$ and $\Sigma_e = I$. This initialization works well for either maximum a posteriori or Bayesian inference.
**while** *not converged/mixed* **do**
  **for** $e = 1 \dots E$ **do**
    **foreach** *row of* $U^{(e,t)} : U^{(e,t)}_{i\cdot}$ **do**
      Update $U^{(e,t+1)}_{i\cdot}$ using $U^{(e,t)}_{i\cdot}$ and all the observations involving the current entity: i.e., $\forall e', e,\ X^{(ee')}_{i\cdot}$. For maximum a posteriori, the update is a convex optimization, approximated by one Newton step; for Bayesian inference, we sample from the conditional sampling distribution for each factor row (Equation 8) using Hessian Metropolis-Hastings (Algorithm 3).
    If the hyperparameters for factor $U^{(e)}$, i.e., $(\mu_e, \Sigma_e)$, are not fixed, then compute the normal-Inverse-Wishart posterior with parameters defined in Equation 6. Use the posterior mode for maximum a posteriori, or a Gibbs sample for Bayesian inference.
  $t = t + 1$

where $f^{(ij)}$ is the element-wise link function that maps the low-rank latent representation into predictions. Each factor $U^{(i)}$ has its own Gaussian prior, defined over factor rows. In the hierarchical case, each of these priors is assigned a normal-Inverse-Wishart hyperprior.

### 4.4 Bayesian Prediction

*Hold-out prediction*: Given the low-rank representation for an entity, we want to predict the value of relations that the entity participates in, e.g., predict $X_{ij}$. For a point estimate, the prediction is $\hat{X}_{ij} = f(U_{i\cdot}V^T_{j\cdot})$. Given a posterior distribution, we integrate out uncertainty:

$$p(X_{ij} \mid \mathcal{D}) = \int p(X_{ij} \mid \mathcal{F}, \Theta) p(\mathcal{F}, \Theta \mid X, Y, W, \tilde{W})\, d\{\mathcal{F}, \Theta\}. \tag{10}$$

Equation 10 is known as the posterior predictive distribution. Since we have only samples from the posterior, $\{\mathcal{F}^{(s)}, \Theta^{(s)}\}$, we use a Monte Carlo approximation of Equation 10,

$$p(X_{ij} \mid \mathcal{D}) = \frac{1}{S} \sum_{s=1}^{S} p(X_{ij} \mid \mathcal{F}^{(s)}, \Theta^{(s)}). \tag{11}$$

*Fold-in prediction*: To generate a low-rank representation for an entity not in the training set, we fix the value of $\{\mathcal{F}^{(s)}, \Theta^{(s)}\}$ for each sample, and use HMH to generate five samples from the posterior over the new factor row.[4] Under maximum a posteriori, fold-in reduces to finding the most likely value for a new factor row, given that the learned $\{\mathcal{F}, \Theta\}$ is fixed.

## 5 Experiments

*Claim: Our hierarchical Bayesian model can use inexpensive text data to improve the predictive accuracy of a model of fMRI brain activity, even when the voxels being tested never appeared in the training data. The hierarchical Bayesian variant of Collective Matrix Factorization can have significantly better predictive accuracy than either the hierarchical or non-hierarchical analogues under maximum a posteriori.*

[4] We discard the first twenty samples (burn-in), and every other sample after that (subsampling). We chose $S = 20$ for hold-out experiments, and $S = 10$ on fold-in experiments. The predictive performance was not significantly improved by increasing $S$.

**Algorithm 2**: Initialization for Hessian Metropolis-Hastings

**Input**: Number of entities of type $e$, $n_e$. Prior mean $\mu_e$ and covariance $\Sigma_e$.

**Output**: Initial value for low-rank factor $U^{(e,0)}$. Mean and negative precision matrix for the first forward sampling distribution: $\bar{U}_{i\cdot}^{(e,0)}$ and $\nabla^2\mathcal{O}\left(U_{i\cdot}^{(e,0)}\right)$.

**for** $i = 1 \dots n_t$ **do**

Sample from the prior: $U_{i\cdot}^{(e,0)} \sim \mathcal{N}(\mu_e, \Sigma_e)$.

Choose a random step length: $\eta \sim \mathcal{U}[0,1]$.

Compute gradient and Hessian. Estimate posterior mean using one Newton step:

$\bar{U}_{i\cdot}^{(e,0)} = U_{i\cdot}^{(e,0)} + \eta \cdot \left[\nabla\mathcal{O}\left(U_{i\cdot}^{(e,0)}\right)\right]\left[\nabla^2\mathcal{O}\left(U_{i\cdot}^{(e,0)}\right)\right]^{-1}$.

**Algorithm 3**: Hessian Metropolis-Hastings

**Input**: Previous sample from the Markov chain, $U_{i\cdot}^{(e,t)}$. Observations involving entity $i$.

**Output**: Next sample: $U_{i\cdot}^{(e,t+1)}$. Mean and negative precision of the next proposal.

Sample the proposal: $U_{i\cdot}^{(e,*)} \sim \mathcal{N}\left(\bar{U}_{i\cdot}^{(e,t)}, \left[-\nabla^2\mathcal{O}\left(U_{i\cdot}^{(e,t)}\right)\right]^{-1}\right)$.

Compute the gradient $[\nabla\mathcal{O}(U_{i\cdot})]$ and Hessian $\left[\nabla^2\mathcal{O}(U_{i\cdot})\right]$ at $U_{i\cdot}^{(e,*)}$.

Estimate the posterior mean using one Newton step with random step length:

$\bar{U}_{i\cdot}^{(e,*)} = U_{i\cdot}^{(e,*)} + \eta \cdot \left[\nabla\mathcal{O}\left(U_{i\cdot}^{(e,*)}\right)\right]\left[\nabla^2\mathcal{O}\left(U_{i\cdot}^{(e,*)}\right)\right]^{-1}$, where $\eta \sim \mathcal{U}[0,1]$.

Compute the acceptance probability

$$\alpha = \min\left\{1, \frac{p\left(U_{i\cdot}^{(e,*)} \mid \mathcal{D}, F^{(t)}, \Theta^{(t)}\right)}{p\left(U_{i\cdot}^{(e,t)} \mid \mathcal{D}, \mathcal{F}^{(t)}, \Theta^{(t)}\right)} \times \frac{\mathcal{N}\left(U_{i\cdot}^{(e,t)} \mid \bar{U}_{i\cdot}^{(e,*)}, \left[-\nabla^2\mathcal{O}\left(U_{i\cdot}^{(e,*)}\right)\right]^{-1}\right)}{\mathcal{N}\left(U_{i\cdot}^{(e,*)} \mid \bar{U}_{i\cdot}^{(e,t)}, \left[-\nabla^2\mathcal{O}\left(U_{i\cdot}^{(e,t)}\right)\right]^{-1}\right)}\right\}.$$

**if** $r \sim \mathcal{U}[0,1] \leq \alpha$ **then** $k = *$ **else** $k = t$.

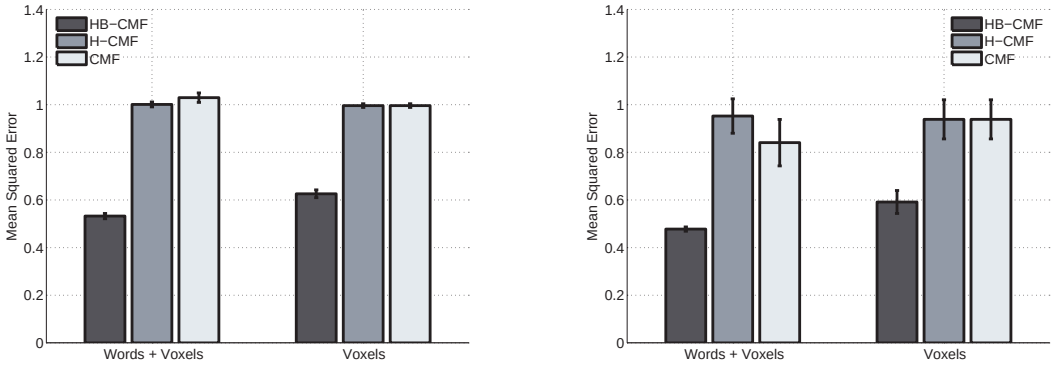
Collect outputs: $U_{i\cdot}^{(e,t+1)} = U_{i\cdot}^{(e,k)}$, $\bar{U}_{i\cdot}^{(e,t+1)} = \bar{U}_{i\cdot}^{(e,k)}$, $\nabla^2\mathcal{O}\left(U_{i\cdot}^{(e,t+1)}\right) = \nabla^2\mathcal{O}\left(U_{i\cdot}^{(e,k)}\right)$.

**Data**: The data collection protocol is described in Mitchell et al. [4]. Stimuli, the shared entities, consist of word-picture pairs flashed onto a screen (e.g., bear, barn, pliers); stimuli are chosen to be category exemplars (e.g., animals, buildings, tools). Nine subjects are presented with sixty stimuli. Each subject was presented with each stimulus six times—the fMRI image for a subject given a stimulus is the average of the six presentations. Averaging the voxel response over patients yields the `Response`($stimulus, voxel$) relation. While a fMRI image contains >20,000 voxels, we use only the 500 most stable voxels, following [4].

The `Co-occurs`($word, stimulus$) relation is collected by measuring whether or not the stimulus word occurs within five tokens of a word found in the Google Tera-word corpus [1]. Of the ∼50,000 common words in the text corpus, we select 20,000 uniformly at random to reduce the cost of learning, allowing for repeated trials in our results. The relations map into two matrices: $X =$ `Co-occurs` and $Y =$ `Response`. Unless stated otherwise, we assume that $X_{ij}$ is Bernoulli distributed, and that $Y_{ij}$ is Gaussian. The embedding dimension is $k = 25$ throughout. We standardize the entries of the $Y$ matrix, to avoid estimating a per-matrix variance as part of the link function $g(\cdot)$.

**Evaluation**: In both fold-in and hold-out experiments, our concern is predicting voxel response, i.e., entries of $Y$, under mean squared error. Fold-in experiments involve testing on voxels, columns of $Y$, which did not appear in the training data. In fold-in experiments, two-thirds of a new entity's observations are used in folding-in; the rest for estimating test error. In hold-out experiments, one-tenth of the observations are used to estimate test error.

**Models Compared**: We compare three variants of Collective Matrix Factorization:




(a) Hold-out (b) Fold-in

Figure 2: Performance on predicting `Response(stimulus,voxel)` using just the `Response` relation (Voxel) and augmenting with `Co-occurs` (Words + Voxels). The bars represent algorithms (error bars are 2-standard deviations, acronyms are described in the body text).

- Hierarchical Bayesian Collective Matrix Factorization (HB-CMF), where we use multiple samples from the posterior distribution over $\{\mathcal{F}, \Theta\}$ in prediction.
- Hierarchical Collective Matrix Factorization (H-CMF), which has the same structure as HB-CMF (i.e., Figure 1(b)), but where we use only the posterior mode in prediction.
- Collective Matrix Factorization (CMF), our baseline model, where we learn the maximum likelihood value of $\mathcal{F}$ given a fixed $\Theta$. For each factor $F$, $\Theta_F = (\mu_F, \Sigma_F)$ is assumed to be zero mean with diagonal covariance. We use psychic initialization, where the $j^{th}$ diagonal element of $\Sigma_F$ is the variance of the $j^{th}$ column of the estimate of $F$ produced by H-CMF. Psychic initialization avoids the computational burden of finding a good value for $\Sigma_F$ by grid search and cross-validation.

The only difference between CMF and H-CMF is the hierarchical prior; the only difference between H-CMF and HB-CMF is that the former approximates the posterior by its mode, the latter by multiple samples drawn from the posterior.

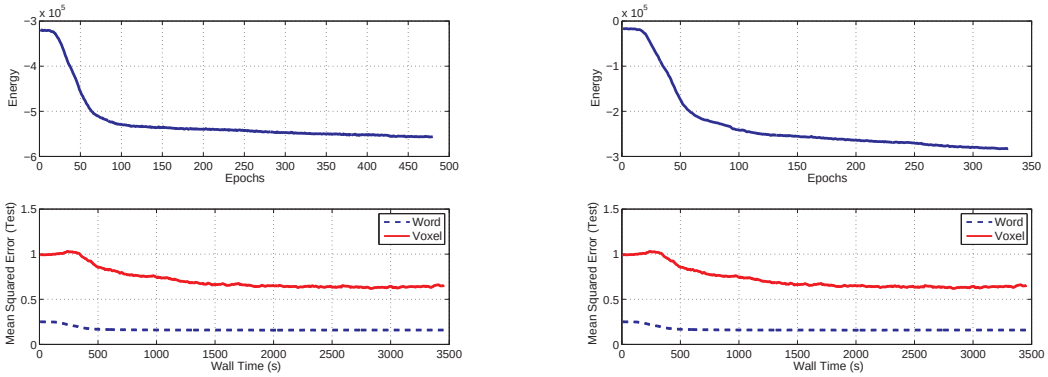
**Relationship to existing work**: The three algorithms we have described generalize a large literature on single- and multiple-matrix factorization. Our primary contribution is a hierarchical Bayesian model for multiple-matrix factorization and a sample-based learning algorithm which (i) scales to a moderately large number of entities, (ii) does so by exploiting structure in matrix factorizations that has been largely neglected in the literature on Bayesian matrix factorization. We refer the reader to Singh [9] for details on related work.

**Predictive Accuracy**: Figure 2 illustrates how much better the hierarchical Bayesian approach is than point estimate alternatives, on both hold-out and fold-in tasks. Using HB-CMF, a statistically significant improvement is achieved by augmenting the `Response` relation with the `Co-occurs` relation (on both hold-out and fold-in tasks).

The results on the fold-in experiment (Figure 2(b)) show that we can achieve high prediction accuracy even when testing voxels which never appeared in the training data. The training and test voxels are drawn from both hemispheres of the brain, with weak spatial correlation: voxels in the visual processing centers tend to be more stable.

The mean over voxel responses (entries in $Y$) is used as a baseline. Since $Y$ has been standardized, the mean predictor is 0, and the mean squared error equals the variance, i.e., 1. Modeling the entire posterior, not just the mode, seems to be critical to prediction here.

Non-Gaussianity is important for good predictive accuracy. If, in HB-CMF, we assume that $p_X$ is Gaussian instead of Bernoulli, prediction accuracy decreases: by 26% on hold-out; by 39% on fold-in. The need for a better proposal distribution was motivated by the desire to avoid choosing between conjugacy assumptions that would have hurt predictive accuracy, or a model without conjugacy assumptions where learning is impractically slow.




(a) Hold-out (b) Fold-in

Figure 3: Mixing behavior of Algorithm 3. The slowest mixing instance of the hold-out and fold-in experiments are shown. Each point on the energy vs. epochs plots (top) measures the loss of a sample. Each point on the test error vs. time plots (bottom) measures the test error of a sample on predicting word co-occurrence or voxel activation.

**Performance**: Alternating Newton-projections scales well for maximum a posteriori—while each iteration is slower, due to line searches, the number of iterations to convergence is less than the Hessian Metropolis-Hastings approach. Sampling the parameters and hyperparameters in HB-CMF takes, on average, $7.2s$ for the hold-out experiment using a parallel implementation on four processors.[5] Our implementation was developed in MATLAB, using the Distributed Computing Environment toolkit. An analogous iteration of H-CMF takes $< 10s$. That said, H-CMF converges in less than 20 iterations; HB-CMF takes over 100 iterations to converge. The practicality of the Bayesian approach depends largely on how many iterations are required (Figure 3). The energy vs. epochs plots suggest that the underlying Markov chain over states mixes quickly.

**Comments**: The premise of this paper is that the world abounds with entities that are related, but not by any readily apparent set of structured rules. Our matrix factorization approach allows for relational learning in such scenarios. We have found that a Bayesian approach is desirable when transferring information between matrix models, which leads to the computational contribution of this paper: an automatically adaptive proposal distribution for Metropolis-Hastings on matrix factorization models.

**Acknowledgements:** We thank Tom Mitchell and Mark Palatucci for their assistance in using the fMRI data; and Tom Mitchell, Christos Faloutsos, Pedro Domingos, Zoubin Gharahamani, and Mark Palatucci for their insightful comments.

## References

[1] T. Brants and A. Franz. Web 1T 5-gram corpus, version 1. Electronic, Sept. 2006.
[2] M. Collins, S. Dasgupta, and R. E. Schapire. A generalization of principal component analysis to the exponential family. In *NIPS 13*, 2001.
[3] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian Data Analysis*. CRC Press, 2nd ed, 2004.
[4] T. M. Mitchell, S. V. Shinkareva, A. Carlson, K.-M. Chang, V. L. Malave, R. A. Mason, and M. A. Just. Predicting human brain activity associated with the meanings of nouns. *Science*, 2008.
[5] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.
[6] Y. Qi and T. P. Minka. Hessian-based Markov Chain Monte Carlo algorithms. In *First Cape Cod Workshop on Monte Carlo Methods, Cape Cod, Massachusetts*. 2002.
[7] R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. In *NIPS 20*, 2007.
[8] R. Salakhutdinov and A. Mnih. Bayesian probabilistic matrix factorization using MCMC. In *ICML*, 2008.
[9] A. P. Singh. *Efficient Matrix Models for Relational Learning*. PhD thesis, Machine Learning Department, Carnegie Mellon University, 2009.
[10] A. P. Singh and G. J. Gordon. Relational learning via collective matrix factorization. In *KDD*, 2008.
[11] M. Welling, C. Chemudugunta, and N. Sutter. Deterministic latent variable models and their pitfalls. In *SDM*, 2008.

[5] The four processors are cores on an AMD Opteron 2384 CPU, operating at 2.7GHz.