A Unified View of Matrix Factorization Models

Ajit P. Singh and Geoffrey J. Gordon

Machine Learning Department
Carnegie Mellon University
Pittsburgh PA 15213, USA
{ajit,ggordon}@cs.cmu.edu

Abstract. We present a unified view of matrix factorization that frames the differences among popular methods, such as NMF, Weighted SVD, E-PCA, MMMF, pLSI, pLSI-pHITS, Bregman co-clustering, and many others, in terms of a small number of modeling choices. Many of these approaches can be viewed as minimizing a generalized Bregman divergence, and we show that (i) a straightforward alternating projection algorithm can be applied to almost any model in our unified view; (ii) the Hessian for each projection has special structure that makes a Newton projection feasible, even when there are equality constraints on the factors, which allows for matrix co-clustering; and (iii) alternating projections can be generalized to simultaneously factor a set of matrices that share dimensions. These observations immediately yield new optimization algorithms for the above factorization methods, and suggest novel generalizations of these methods such as incorporating row and column biases, and adding or relaxing clustering constraints.

1 Introduction

Low-rank matrix factorization is a fundamental building block of machine learning, underlying many popular regression, factor analysis, dimensionality reduction, and clustering algorithms. We shall show that the differences between many of these algorithms can be viewed in terms of a small number of modeling choices. In particular, our unified view places dimensionality reduction methods, such as singular value decomposition [1], into the same framework as matrix co-clustering algorithms like probabilistic latent semantic indexing [2]. Moreover, recently-studied problems, such as relational learning [3] and supervised/semi-supervised matrix factorization [4], can be viewed as the simultaneous factorization of several matrices, where the low-rank representations share parameters. The modeling choices and optimization in the multiple-matrix models are very similar to the single-matrix case.

The first contribution of this paper is descriptive: our view of matrix factorization subsumes many single- and multiple-matrix models in the literature, using only a small set of modeling choices. Our basic single-matrix factorization model can be written $X \approx f(UV^T)$; choices include the prediction link f, the definition of $\approx$, and the constraints we place on the factors U and V. Different combinations of these choices also yield several new matrix factorization models.

W. Daelemans et al. (Eds.): ECML PKDD 2008, Part II, LNAI 5212, pp. 358–373, 2008.
© Springer-Verlag Berlin Heidelberg 2008

The second contribution of this paper is computational: we generalize the alternating projections technique for matrix factorization to handle constraints on the factors—*e.g.*, clustering or co-clustering, or the use of margin or bias terms. For most common choices of $\approx$, the loss has a special property, *decomposability*, which allows for an efficient Newton update for each factor. Furthermore, many constraints, such as non-negativity of the factors and clustering constraints, can be distributed across decomposable losses, and are easily incorporated into the per-factor update. This insight yields a common algorithm for most factorization models in our framework (including both dimensionality reduction and co-clustering models), as well as new algorithms for existing single-matrix models.

A parallel contribution [3] considers matrix factorization as a framework for relational learning, with a focus on multiple relations (matrices) and large-scale optimization using stochastic approximations. This paper, in contrast, focuses on modeling choices such as constraints, regularization, bias terms, and more elaborate link and loss functions in the single-matrix case.

2 Preliminaries

2.1 Notation

Matrices are denoted by capital letters, X, Y, Z. Elements, rows and columns of a matrix are denoted X_{ij}, $X_{i\cdot}$, and $X_{\cdot j}$. Vectors are denoted by lower case letters, and are assumed to be column vectors—*e.g.*, the columns of factor U are $(u_1, \ldots, u_k)$. Given a vector x, the corresponding diagonal matrix is $\mathrm{diag}(x)$. $A \odot B$ is the element-wise (Hadamard) product. $A \circ B$ is the matrix inner product $tr(A^T B) = \sum_{ij} A_{ij} B_{ij}$, which reduces to the dot product when the arguments are vectors. The operator $[A\ B]$ appends the columns of B to A, requiring that both matrices have the same number of rows. Non-negative and strictly positive restrictions of a set $\mathbb{F}$ are denoted $\mathbb{F}_+$ and $\mathbb{F}_{++}$. We denote matrices of natural parameters as Θ, and a single natural parameter as θ.

2.2 Bregman Divergences

A large class of matrix factorization algorithms minimize a Bregman divergence: *e.g.*, singular value decomposition [1], non-negative matrix factorization [5], exponential family PCA [6]. We generalize our presentation of Bregman divergences to include non-differentiable losses:

Definition 1. *For a closed, proper, convex function $F : \mathbb{R}^{m \times n} \to \mathbb{R}$ the generalized Bregman divergence [7,8] between matrices Θ and X is*

$$\mathbb{D}_F(\Theta \,||\, X) = F(\Theta) + F^*(X) - X \circ \Theta$$

where F^* is the convex conjugate, $F^*(\mu) = \sup_{\Theta \in \mathrm{dom}\, F} [\Theta \circ \mu - F(\Theta)]$. We overload the symbol F to denote an element-wise function over matrices. If

$F : \mathbb{R} \to \mathbb{R}$ is an element-wise function, and $W \in \mathbb{R}_{+}^{m \times n}$ is a constant weight matrix, then the weighted decomposable generalized Bregman divergence is

$$\mathbb{D}_F(\Theta \,||\, X, W) = \sum_{ij} W_{ij} \left(F(\Theta_{ij}) + F^*(X_{ij}) - X_{ij}\Theta_{ij} \right).$$

If $F : \mathbb{R} \to \mathbb{R}$ is additionally differentiable, $\nabla F = f$, and $w_{ij} = 1$, the decomposable divergence is equivalent to the standard definition [9,10]:

$$\begin{aligned}\mathbb{D}_F(\Theta \,||\, X, W) &= \sum_{ij} F^*(X_{ij}) - F^*(f(\Theta_{ij})) - \nabla F^*(f(\Theta_{ij}))(X_{ij} - f(\Theta_{ij})) \\ &= D_{F^*}(X \,||\, f(\Theta))\end{aligned}$$

Generalized Bregman divergences are important because (i) they include many common separable divergences, such as squared loss, $F(x) = \frac{1}{2}x^2$, and KL-divergence, $F(x) = x \log_2 x$; (ii) there is a close relationship between Bregman divergences and maximum likelihood estimation in regular exponential families:

Definition 2. *A parametric family of distributions $\psi_F = \{p_F(x|\theta) : \theta\}$ is a regular exponential family if each density has the form*

$$\log p_F(x|\theta) = \log p_0(x) + \theta \cdot x - F(\theta)$$

where θ is the vector of natural parameters for the distribution, x is the vector of minimal sufficient statistics, and $F(\theta) = \log \int p_0(x) \cdot \exp(\theta \cdot x)\, dx$ is the log-partition function.

A distribution in ψ_F is uniquely identified by its natural parameters. It has been shown that for regular exponential families

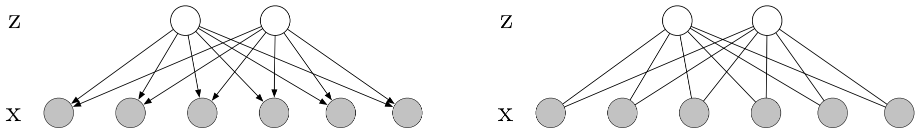
$$\log p_F(x|\theta) = \log p_0(x) + F^*(x) - D_{F^*}(x \,||\, f(\theta)),$$

where the prediction link $f(\theta) = \nabla F(\theta)$ is known as the matching link for F [11,12,6,13]. Generalized Bregman divergences assume that the link f and the loss match, though alternating projections generalizes to non-matching links.

The relationship between matrix factorization and exponential families is made clear by viewing the data matrix as a collection of samples $\{X_{11}, \ldots, X_{mn}\}$. Let $\Theta = UV^T$ be the parameters. For a decomposable regular Bregman divergence, minimizing $D_{F^*}(X \,||\, f(\Theta))$ is equivalent to maximizing the log-likelihood of the data under the assumption that X_{ij} is drawn from the distribution in ψ_F with natural parameter Θ_{ij}.

3 Unified View of Single-Matrix Factorization

Matrix factorization is both more principled and more flexible than is commonly assumed. Our arguments fall into the following categories:



(a) Latent marginal independence (b) Latent conditional independence

Fig. 1. Two layer model of matrix factorization

Decomposability: Matrix factorization losses tend to be decomposable, expressible as the sum of losses for each element in the matrix, which has both computational and statistical motivations. In many matrices the ordering of rows and columns is arbitrary, permuting the rows and columns separately would not change the distribution of the entries in the matrix. Formally, this idea is known as row-column exchangeability [14,15]. Moreover, for such matrices, there exists a function φ such that $X_{ij} = \varphi(\mu, \mu_i, \mu_j, \epsilon_{ij})$ where μ represents behaviour shared by the entire matrix (*e.g.*, a global mean), μ_i and μ_j per-row and per-column effects, and ϵ_{ij} per-entry effects. The ϵ_{ij} terms lead naturally to decomposable losses. The computational benefits of decomposability are discussed in Sect. 5.

Latent Independence: Matrix factorization can be viewed as maximum likelihood parameter estimation in a two layer graphical model, Fig. 1. If the rows of X are exchangeable, then each training datum corresponds to $x = X_{i\cdot}$, whose latent representation is $z = U_{i\cdot}$, and where $\Theta_{i\cdot} = U_{i\cdot}V^T$ are the parameters of $p(x|z)$. Most matrix factorizations assume that the latents are marginally independent of the observations, Fig. 1(a); an alternate style of matrix factorizations assumes that the latents are conditionally independent given the observations, Fig. 1(b), notably exponential family harmoniums [16].

Parameters vs. Predictions: Matrix factorizations can be framed as minimizing the loss with respect to model parameters; or minimizing the loss with respect to reconstruction error. Since many common losses are regular Bregman divergences, and there is a duality between expectation and natural parameters—$D_{F^*}(x \,||\, f(\theta)) = D_F(\theta \,||\, f^{-1}(x))$, the two views are usually equivalent. This allows one to view many plate models, such as pLSI, as matrix factorization.

Priors and Regularizers: Matrix factorizations allow for a wide variety of priors and regularizers, which can both address overfitting and the need for pooling information across different rows and columns. Standard regression regularizers, such as the ℓ_p norm of the factors, can be adapted. Hierarchical priors can be used to produce a fully generative model over rows and columns, without resorting to folding-in, which can easily lead to optimistic estimates of test errors [17].

Bayesian stance: The simplest distinction between the Bayesian and maximum a posteriori/maximum likelihood approaches is that the former computes a distribution over U and V, while the latter generates a point estimate. Latent Dirichlet allocation [18] is an example of Bayesian matrix factorization.

Collective matrix factorization assumes that the loss is decomposable and that the latents are marginally independent. Our presentation assumes that the prior is simple (non-hierarchical) and estimation is done via regularized maximum likelihood. Under these assumptions a matrix factorization can be defined by the following choices, which are sufficient to include many popular approaches:

1. Data weights $W \in \mathbb{R}_{+}^{m \times n}$.
2. Prediction link $f : \mathbb{R}^{m \times n} \to \mathbb{R}^{m \times n}$.
3. Hard constraints on factors, $U, V \in \mathcal{C}$.
4. Weighted loss between X and $\hat{X} = f(UV^T)$, $\mathcal{D}(X \,||\, \hat{X}, W) \geq 0$.
5. Regularization penalty, $\mathcal{R}(U, V) \geq 0$.

Given these choices the optimization for the model $X \approx f(UV^T)$ is

$$\underset{(U,V) \in \mathcal{C}}{\operatorname{argmin}} \mathcal{D}(X \,||\, f(UV^T), W) + \mathcal{R}(U, V)$$

Prediction links allow nonlinear relationships between $\Theta = UV^T$ and the data X. We focus on the case where $\mathcal{D}$ is a generalized Bregman divergences and f is the matching link. Constraints, weights, and regularizers, along with the ensuing optimization issues, are discussed in Sect. 5.

3.1 Models Subsumed by the Unified View

To justify our unified view we discuss a representative sample of single matrix factorizations (Table 1) and how they can be described by their choice of loss, link, constraints, weights, and regularization. Notice that all the losses in Table 1 are decomposable, and that many of the factorizations are closely related to linear models for independent, identically distributed (iid) data points: SVD is the matrix analogue of linear regression; E-PCA/G^{2}L^{2}M are the matrix analogues of generalized linear models; MMMF is the matrix analogue of ordinal regression under hinge loss[1] h; k-medians is the matrix analogue of quantile regression [19], where the quantile is the median; and ℓ_1-SVD is the matrix analogue of the LASSO [20]. The key difference between the regression/clustering algorithm and its matrix analogue is changing the assumption from iid observations, where each row of X is drawn from a single distribution, to row exchangeability.

Many of the models in Table 1 differ in the loss, the constraints, and the optimization. In many cases the loss and link do not match, and the optimization is non-convex in Θ, which is usually far harder than minimizing a similar convex problem. We speculate that replacing the non-matching link in pLSI with its matching link may yield an alternative that is easier to optimize.

Similarities between matrix factorizations have been noted elsewhere, such as the equivalence of pLSI and NMF with additional constraints [21]. pLSI requires that the matrix be parameters of a discrete distribution, $1 \circ X = 1$. Adding an orthogonality constraint to ℓ_2-NMF yields a relaxed form of k-means [22]. Orthogonality of a column factors $V^T V = I$ along with integrality of V_{ij} corresponds to hard clustering the columns of X, at most one entry in $V_{i\cdot}$ can be non-zero. Even without the integrality constraint, orthogonality plus non-negativity still implies

[1] In Fast-MMMF a smoothed, differentiable version of hinge loss is used, h_γ.

a stochastic (clustering) constraint: $\forall i \sum_\ell V_{i\ell} = 1, V_{i\ell} \geq 0$. In the k-means algorithm, U acts as the cluster centroids and V as the clustering indicators, where the rank of the decomposition and the number of clusters is k. In alternating projections, each row update for a factor with the clustering constraint reduces to assigning hard or soft cluster membership to each point/column of X.

A closely related characterization of matrix factorization models, which relates NMF, pLSI, as well as Bayesian methods like Latent Dirichlet Allocation, is Discrete PCA [23]. Working from a Bayesian perspective the regularizer and divergence are replaced with a prior and likelihood. This restricts one to representing models where the link and loss match, but affords the possibility of Bayesian averaging where we are restricted to regularized maximum likelihood. We speculate that a relationship exists between variational approximations to these models and Bregman information [12], which averages a Bregman divergence across the posterior distribution of predictions.

Our unified view of matrix factorization is heavily indebted to earlier work on exponential family PCA and $\mathrm{G}^2\mathrm{L}^2$M. Our approach on a single matrix differs in several ways from $\mathrm{G}^2\mathrm{L}^2$Ms: we consider extensions involving bias/margin terms, data weights, and constraints on the factors, which allows us to place matrix factorizations for dimensionality reduction and co-clustering into the same alternating Newton-projections approach. Even when the loss is a regular Bregman divergence, which corresponds to a regular exponential family, placing constraints on U and V, and thus on Θ, leads to models which do not correspond to regular exponential families. For the specific case of log-loss and its matching link, Logistic PCA proposes alternating projections on a lower bound of the log-likelihood. Max-margin matrix factorization is one of the more elaborate models: ordinal ratings $\{1, \ldots, R\}$[2] are modeled using $R-1$ parallel separating hyperplanes, corresponding to the binary decisions $X_{ij} \leq 1, X_{ij} \leq 2, X_{ij} \leq 3, \ldots, X_{ij} \leq R-1$. The per-row bias term B_{ir} allows the distance between hyperplanes to differ for each row. Since this technique was conceived for user-item matrices, the biases capture differences in each user. Predictions are made by choosing the value which minimizes the loss of the $R-1$ decision boundaries, which yields a number in $\{1, \ldots, R\}$ instead of $\mathbb{R}$.

4 Collective Matrix Factorization

A set of related matrices involves entity types $\mathcal{E}_1, \ldots, \mathcal{E}_t$, where the elements of each type are indexed by a row or column in at least one of the matrices. The number of entities of type $\mathcal{E}_i$ is denoted n_i. The matrices themselves are denoted $X^{(ij)}$ where each row corresponds to an element of type $\mathcal{E}_i$ and each column to an element of type $\mathcal{E}_j$. If there are multiple matrices on the same types we disambiguate them with the notation $X^{(ij,u)}, u \in \mathbb{N}$. Each data matrix is factored under the model $X^{(ij)} \approx f^{(ij)}(\Theta^{(ij)})$ where $\Theta^{(ij)} = U^{(i)}(U^{(j)})^T$ for low rank factors $U^{(i)} \in \mathbb{R}^{n_i \times k_{ij}}$. The embedding dimensions are $k_{ij} \in \mathbb{N}$. Let k be the largest embedding dimension. In low-rank factorization $k \ll \min(n_1, \ldots, n_t)$.

[2] Zeros in the matrix are considered missing values, and are assigned zero weight.

Table 1. Single matrix factorization models. dom X_{ij} describes the types of values allowed in the data matrix. Unweighted matrix factorizations are denoted $W_{ij} = 1$. If constraints or regularizers are not used, the entry is marked with a em-dash.

Method	dom X_{ij}	Link $f(\theta)$	Loss $\mathcal{D}(X \mid \hat{X} = f(\Theta), W)$	W_{ij}
SVD [1]	$\mathbb{R}$	θ	$\Vert W \odot (X - \hat{X})\Vert^2_{Fro}$	1
W-SVD [24,25]	$\mathbb{R}$	θ	$\Vert W \odot (X - \hat{X})\Vert^2_{Fro}$	≥ 0
k-means [26]	$\mathbb{R}$	θ	$\Vert W \odot (X - \hat{X})\Vert^2_{Fro}$	1
k-medians	$\mathbb{R}$	θ	$\sum_{ij} \lvert W_{ij}\left(X_{ij} - \hat{X}_{ij}\right)\rvert$	1
ℓ_1-SVD [27]	$\mathbb{R}$	θ	$\sum_{ij} \lvert W_{ij}\left(X_{ij} - \hat{X}_{ij}\right)\rvert$	≥ 0
pLSI [2]	$\mathbf{1} \circ X = 1$	θ	$\sum_{ij} W_{ij}\left(X_{ij} \log \frac{X_{ij}}{\hat{X}_{ij}}\right)$	1
NMF [5]	$\mathbb{R}_+$	θ	$\sum_{ij} W_{ij}\left(X_{ij} \log \frac{X_{ij}}{\Theta_{ij}} + \Theta_{ij} - X_{ij}\right)$	1
ℓ_2-NMF [28,5]	$\mathbb{R}_+$	θ	$\Vert W \odot (X - \hat{X})\Vert^2_{Fro}$	1
Logistic PCA [29]	$\{0, 1\}$	$(1 + e^{-\theta})^{-1}$	$\sum_{ij} W_{ij}\left(X_{ij} \log \frac{X_{ij}}{\hat{X}_{ij}} + (1 - X_{ij}) \log \frac{1-X_{ij}}{1-\hat{X}_{ij}}\right)$	1
E-PCA [6]	many	many	decomposable Bregman (D_F)	1
G^{2}L^{2}M [8]	many	many	decomposable Bregman ($\mathbb{D}_F$)	1
MMMF [30]	$\{0, \ldots, R\}$	min-loss	$\sum_{r=1}^{R-1} \sum_{ij : X_{ij} \neq 0} W_{ij} \cdot h(\Theta_{ij} - B_{ir})$	1
Fast-MMMF [31]	$\{0, \ldots, R\}$	min-loss	$\sum_{r=1}^{R-1} \sum_{ij : X_{ij} \neq 0} W_{ij} \cdot h_\gamma(\Theta_{ij} - B_{ir})$	1

Method	Constraints U	Constraints V	Regularizer $\mathcal{R}(U, V)$	Algorithm(s)
SVD	$U^T U = I$	$V^T V = \Lambda$	—	Gaussian Elimination, Power Method
W-SVD	—	—	—	Gradient, EM
k-means	—	$V^T V = I$, $V_{ij} \in \{0, 1\}$	—	EM
k-medians	—	$V^T V = I$, $V_{ij} \in \{0, 1\}$	—	Alternating
ℓ_1-SVD	—	—	—	Alternating
pLSI	$\mathbf{1}^T U \mathbf{1} = 1$ $U_{ij} \geq 0$	$\mathbf{1}^T V = \mathbf{1}$ $V_{ij} \geq 0$	—	EM
NMF	$U_{ij} \geq 0$	$V_{ij} \geq 0$	—	Multiplicative
ℓ_2-NMF	$U_{ij} \geq 0$	$V_{ij} \geq 0$	—	Multiplicative, Alternating
Logistic PCA	—	—	—	EM
E-PCA	—	—	—	Alternating
G^{2}L^{2}M	—	—	$\Vert U\Vert^2_{Fro} + \Vert V\Vert^2_{Fro}$	Alternating (Subgradient, Newton)
MMMF	—	—	$tr(UV^T)$	Semidefinite Program
Fast-MMMF	—	—	$\frac{1}{2}(\Vert U\Vert^2_{Fro} + \Vert V\Vert^2_{Fro})$	Conjugate Gradient

Collective matrix factorization addresses the problem of simultaneously factoring a set of matrices that are related, where the rows or columns of one matrix index the same type as the row or column of another matrix. An example of such data is joint document-citation models where one matrix consists of the word counts in each documents, and another matrix consists of hyperlinks or citations between documents. The types are documents ($\mathcal{E}_1$), words ($\mathcal{E}_2$), and cited documents ($\mathcal{E}_3$), which can include documents not in the set $\mathcal{E}_1$. The two relations (matrices) are denoted $\mathcal{E}_1 \sim \mathcal{E}_2$ and $\mathcal{E}_2 \sim \mathcal{E}_3$. If a matrix is unrelated to the others, it can be factored independently, and so we consider the case where the schema, the links between types $\{\mathcal{E}_i\}_i$, is fully connected. Denote the schema $E = \{(i,j) : \mathcal{E}_i \sim \mathcal{E}_j \wedge i < j\}$.

We assume that each matrix in the set $\{X^{(ij)}\}_{(i,j)\in E}$ is reconstructed under a weighted generalized Bregman divergence with factors $\{U^{(i)}\}_{i=1}^{t}$ and constant data weight matrices $\{W^{(ij)}\}_{(i,j)\in E}$. Our approach is trivially generalizable to any twice-differentiable and decomposable loss. The total reconstruction loss on all the matrices is the weighted sum of the losses for each reconstruction:

$$\mathcal{L}_u = \sum_{(i,j)\in E} \alpha^{(ij)} \mathbb{D}_F(\Theta^{(ij)} \,||\, X^{(ij)}, W^{(ij)})$$

where $\alpha^{(ij)} \geq 0$. We regularize on a per-factor basis to mitigate overfitting:

$$\mathcal{L} = \sum_{(i,j)\in E} \alpha^{(ij)} \mathbb{D}_F(\Theta^{(ij)} \,||\, X^{(ij)}, W^{(ij)}) + \sum_{i=1}^{t} \mathcal{R}(U^{(i)})$$

Learning consists of finding factors $U^{(i)}$ that minimize $\mathcal{L}$.

5 Parameter Estimation

The parameter space for a collective matrix factorization is large, $O(k \sum_{i=1}^{t} n_i)$, and $\mathcal{L}$ is non-convex, even in the single matrix case. One typically resorts to techniques that converge to a local optimum, like conjugate gradient or EM. A direct Newton step is infeasible due to the number of parameters in the Hessian. Another approximate approach is alternating projection, or block coordinate descent: iteratively optimize one factor $U^{(r)}$ at a time, fixing all the other factors. Decomposability of the loss implies that the Hessian is block diagonal, which allows a Newton coordinate update on $U^{(r)}$ to be reduced to a sequence of independent update over the rows $U_{i\cdot}^{(r)}$.

Ignoring terms that are constant with respect to the factors, the gradient of the objective with respect to one factor, $\nabla_r \mathcal{L} = \frac{\partial \mathcal{L}}{\partial U^{(r)}}$, is

$$\nabla_r \mathcal{L} \;=\; \sum_{(r,s)\in E} \alpha^{(rs)} \left(W^{(rs)} \odot \left(f^{(rs)} \left(\Theta^{(rs)} \right) - X^{(rs)} \right) \right) U^{(s)} + \nabla \mathcal{R}(U^{(r)}). \quad (1)$$

The gradient of a collective matrix factorization is the weighted sum of the gradients for each individual matrix reconstruction. If all the per-matrix losses, as well as the regularizers $\mathcal{R}(\cdot)$, are decomposable, then the Hessian of $\mathcal{L}$ with respect to $U^{(r)}$ is block-diagonal, with each block corresponding to a row of $U^{(r)}$. For a single matrix the result is proven by noting that a decomposable loss implies that the estimate of $X_{i\cdot}$ is determined entirely by $U_{i\cdot}$ and V. If V is fixed then the Hessian is block diagonal. An analogous argument applies when U is fixed and V is optimized. For a set of related matrices the result follows immediately by noting that Equation 1 is a linear function (sum) of per-matrix losses and the differential is a linear operator. Differentiating the gradient of the loss with respect to $U_{i\cdot}^{(r)}$,

$$\nabla_{r,i}\mathcal{L} = \sum_{(r,s)\in E} \alpha^{(rs)} \left(W_{i\cdot}^{(rs)} \odot \left(f^{(rs)}\left(\Theta_{i\cdot}^{(rs)} \right) - X^{(rs)} \right) \right) U^{(s)} + \nabla \mathcal{R}(U_{i\cdot}^{(r)}),$$

yields the Hessian for the row:

$$\nabla^2_{r,i}\mathcal{L} = \sum_{(r,s)\in E} \alpha^{(rs)} \left(U^{(s)} \right)^T \operatorname{diag}\left(W_{i\cdot}^{(rs)} \odot f^{(rs)}\left(\Theta_{i\cdot}^{(rs)} \right) \right) U^{(s)} + \nabla^2 \mathcal{R}(U_{i\cdot}^{(r)}).$$

Newton's rule yields the step direction $\nabla_{r,i}\mathcal{L} \cdot [\nabla^2_{r,i}\mathcal{L}]^{-1}$. We suggest using the Armijo criterion [32] to set the step length. While each projection can be computed fully, we found it suffices to take a single Newton step. For the single matrix case, with no constraints, weights, or bias terms, this approach is known as G^{2}L^{2}M [8]. The Hessian is a $k \times k$ matrix, regardless of how large the data matrix is. The cost of a gradient update for $U_{i\cdot}^{(r)}$ is $O(k \sum_{j:\mathcal{E}_i \sim \mathcal{E}_j} n_j)$. The cost of Newton update for the same row is $O(k^3 + k^2 \sum_{j:\mathcal{E}_i \sim \mathcal{E}_j} n_j)$. If the matrix is sparse, n_j can be replaced with the number of entries with non-zero weight. The incremental cost of a Newton update over the gradient is essentially only a factor of k more expensive when $k \ll \min(n_1, \ldots, n_t)$.

If $\mathcal{E}_j$ participates in more than one relationship, we allow our model to use only a subset of the columns of $U^{(j)}$ for each relationship. This flexibility allows us to have more than one relation between $\mathcal{E}_i$ and $\mathcal{E}_j$ without forcing ourselves to predict the same value for each one. In an implementation, we would store a list of participating column indices from each factor for each relation; but for simplicity, we ignore this possibility in our notation.

The advantages to our alternating-Newton approach include:

- **Memory Usage:** A solver that optimizes over all the factors simultaneously needs to compute residual errors to compute the gradient. Even if the data is sparse, the residuals rarely are. In contrast, our approach requires only that we store one row or column of a matrix in memory, plus $O(k^2)$ memory to perform the update. This make out-of-core factorizations, where the matrix cannot be stored in RAM, relatively easy.
- **Flexibility of Representation:** Alternating projections works for any link and loss, and can be applied to any of the models in Table 1[3]. In the following sections, we show that the alternating Newton step can also be used

[3] For integrally constrained factors, like V in k-means, the Newton projection, a continuous optimization, is replaced by an integer optimization, such as hard clustering.

with stochastic constraints, allowing one to handle matrix co-clustering algorithms. Additionally, the form of the gradient and Hessian make it easy to replace the per-matrix prediction link with different links for each element of a matrix.

5.1 Relationship to Linear Models

Single-matrix factorization $X \approx f(UV^T)$ is a bilinear form, *i.e.*, linear when one of the factors is fixed. An appealing property of alternating projection for collective matrix factorization is that the projection reduces an optimization over matrices $U^{(r)}$ into an optimization over data vectors $U^{(r)}_{i\cdot}$—essentially linear models where the "features" are defined by the fixed factors. Since the projection is a linear form, we can exploit many techniques from regression and clustering for iid data. Below, we use this relationship to adapt optimization techniques for ℓ_1-regularized regression to matrix factorization.

5.2 Weights

Weight $W_{ij} = 0$ implies that the corresponding entry in the data matrix has no influence on the reconstruction, which allows us to handle missing values. Moreover, weights can be used to scale terms so that the loss of each matrix reconstruction in $\mathcal{L}$ is a per-element loss, which prevents larger data matrices from exerting disproportionate influence. If the Bregman divergences correspond to exponential families, then we can use log-likelihood as a common scale. Even when the divergences are not regular, computing the average value of $\mathbb{D}_{F^{(ij)}}/\mathbb{D}_{F^{(rs)}}$, given uniform random natural parameters, provides an adequate estimate of the relative scale of the two divergences, which can be accounted for in the weights.

5.3 Regularization

The most common regularizers used in matrix factorization are ℓ_p regularizers: $\mathcal{R}(U) \propto \lambda \sum_{ij} |u_{ij}|^p$, where λ controls the strength of the regularizer, are decomposable. In our experiments we use ℓ_2-regularization:

$$\mathcal{R}(U) = \lambda ||U||^2_{Fro}/2, \quad \nabla \mathcal{R}(U_{i\cdot}) = U_{i\cdot}/\lambda, \quad \nabla^2 \mathcal{R}(U_{i\cdot}) = \mathrm{diag}(\lambda^{-1}\mathbf{1}).$$

The ℓ_1-regularization constraint can be reduced to an inequality constraint on each row of the factors: $|U^{(r)}_{i\cdot}| \leq t/\lambda, \exists t > 0$. One can exploit a variety of approaches for ℓ_1-regularized regression (see [33] for survey) in the projection step. For example, using the sequential quadratic programming approach (ibid), the step direction d is found by solving the following quadratic program: Let $x = U^{(r)}_{i\cdot}$, $x^+ = \max(0, x)$, $x^- = -\min(0, x)$, so $x = x^+ - x^-$:

$$\begin{aligned}
&\min_d \, (\nabla_{r,i}\mathcal{L}_u + \lambda \mathbf{1}) \circ d + \frac{1}{2} d^T \cdot \nabla^2_{r,i}\mathcal{L}_u \cdot d \\
&\text{s.t. } \forall i \;\; x_i^+ + d_i^+ \geq 0 \\
&\qquad \forall i \;\; x_i^- + d_i^- \geq 0
\end{aligned}$$

5.4 Clustering and Other Equality Constraints

Inequality constraints turn the projection into a constrained optimization; but equality constraints can be incorporated into an unconstrained optimization, such as our Newton projection. Equality constraints can be used to place matrix co-clustering into our framework. With no constraints on the factors, each matrix factorization can be viewed as dimensionality reduction or factor analysis: an increase in the influence of one latent variable does not require a decrease in the influence of other latents. In clustering the stochastic constraint, $\forall i \ \sum_j U_{ij}^{(r)} = 1, U_{ij}^{(r)} \geq 0$, implies that entities of $\mathcal{E}_i$ must belong to one of k latent clusters, and that $U_{i\cdot}^{(r)}$ is a distribution over cluster membership. In matrix co-clustering stochastic constraints are placed on both factors. Since the Newton step is based on a quadratic approximation to the objective, a null space argument ([34] Chap. 10) can be used to show that with a stochastic constraint the step direction d for row $U_{i\cdot}^{(r)}$ is the solution to

$$\begin{bmatrix} \nabla^2_{r,i}\mathcal{L} & 1 \\ 1^T & 0 \end{bmatrix} \begin{bmatrix} d \\ \nu \end{bmatrix} = \begin{bmatrix} -\nabla_{r,i}\mathcal{L} \\ 1 - \mathbf{1}^T U_{i\cdot}^{(r)} \end{bmatrix} \qquad (2)$$

where ν is the Lagrange multiplier for the stochastic constraint. The above technique is easily generalized to $p > 1$ linear constraints, yielding a $k + p$ Hessian.

5.5 Bias Terms

Under our assumption of decomposable $\mathcal{L}$ we have that $X_{ij}^{(rs)} \approx f(\Theta_{ij}^{(rs)})$, but matrix exchangeability suggests there may be an advantage to modeling per-row and per-column behaviour. For example, in collaborative filtering, bias terms can calibrate for a user's mean rating. A straightforward way to account for bias is to append an extra column of parameters to U paired with a constant column in V: $\tilde{U} = [U \ u_{k+1}]$ and $\tilde{V} = [V \ \mathbf{1}]$. We do not regularize the bias. It is equally straightforward to allow for bias terms on both rows and columns: $\tilde{U} = [U \ u_{k+1} \ \mathbf{1}]$ and $\tilde{V} = [V \ \mathbf{1} \ v_{k+1}]$, and so $\tilde{U}\tilde{V}^T = (UV^T) + u_{k+1}\mathbf{1}^T + \mathbf{1}v_{k+1}^T$. Note that these are biases in the space of natural parameters, a special case being a margin in the hinge or logistic loss—*e.g.*, the per-row (per-user, per-rating) margins in MMMF are just row biases. The above biases maintain the decomposability of $\mathcal{L}$, but there are cases where this is not true. For example, a version of MMMF that shares the same bias for all users for a given rating—all the elements in u_{k+1} must share the same value.

5.6 Stochastic Newton

The cost of a full Hessian update for $U_{i\cdot}^{(r)}$ is essentially k times more expensive than the gradient update, which is independent of the size of the data. However, the cost of computing the gradient depends linearly on the number of observations in any row or column whose reconstruction depends on $U_{i\cdot}^{(r)}$. If the data matrices are dense, the computational concern is the cost of the gradient. We refer readers to a parallel contribution [3], which describes a provably convergent stochastic approximation to the Newton step.

6 Related Work

The three-factor schema $\mathcal{E}_1 \sim \mathcal{E}_2 \sim \mathcal{E}_3$ includes supervised and semi-supervised matrix factorization, where $X^{(12)}$ contains one or more types of labellings of the rows of $X^{(23)}$. An example of supervised matrix factorization is SVDM [35], where $X^{(23)}$ is factored under squared loss and $X^{(12)}$ is factored under hinge loss. A similar model was proposed by Zhu et al. [36], using a smooth variant of the hinge loss. Supervised matrix factorization has been recently generalized to regular Bregman divergences [4]. Another example is supervised LSI [37], which factors both the data and label matrices under squared loss, with an orthogonality constraint on the shared factors. Principal components analysis, which factors a doubly centered matrix under squared loss, has also been extended to the three-factor schema [38]. An extension of pLSI to two related matrices, pLSI-pHITS, consists of two pLSI models that share latent variables [39].

While our choice of a bilinear form UV^T is common, it is not the only option. Matrix co-clustering often uses a trilinear form $X \approx C_1 A C_2^T$ where $C_1 \in \{0,1\}^{n_1 \times k}$ and $C_2 \in \{0,1\}^{n_2 \times k}$ are cluster indicator matrices, and $A \in \mathbb{R}^{k \times k}$ contains the predicted output for each combination of clusters. This trilinear form is used in k-partite clustering [40], an alternative to collective matrix factorization which assumes that rows and columns are hard-clustered under a Bregman divergence, minimized under alternating projections. The projection step requires solving a clustering problem, for example, using k-means. Under squared loss, the trilinear form can also be approximately minimized using a spectral clustering relaxation [41]. Or, for general Bregman divergences, the trilinear form can be minimized by alternating projections with iterative majorization for the projection [42]. A similar formulation in terms of log-likelihoods uses EM [43]. Banerjee et al. propose a model for Bregman clustering in matrices and tensors [12,44], which is based on Bregman information instead of divergence. While the above approaches generalize matrix co-clustering to the collective case, they make a clustering assumption. We show that both dimensionality reduction and clustering can be placed into the same framework. Additionally, we show that the difference in optimizing a dimensionality reduction and soft co-clustering model is small, an equality constraint in the Newton projection.

7 Experiments

We have argued that our alternating Newton-projection algorithm is a viable approach for training a wide variety of matrix factorization models. Two questions naturally arise: is it worthwhile to compute and invert the Hessian, or is gradient descent sufficient for projection? And, how does alternating Newton-projection compare to techniques currently used for specific factorization models?

While the Newton step is more expensive than the gradient step, our experiments indicate that it is definitely beneficial. To illustrate the point, we use an example of a three-factor model: $X^{(12)}$ corresponds to a user-movie matrix containing ratings, on a scale of 1–5 stars, from the Netflix competition [45]. There are $n_1 = 500$ users and $n_2 = 3000$ movies. Zeros in the ratings matrix correspond to unobserved entries, and are assigned zero weight. $X^{(23)}$ contains movie-genre information from IMDB [46], with $n_3 = 22$ genres. We reconstruct $X^{(12)}$ under I-divergence, and use its matching link—*i.e.*, $X^{(12)}_{ij}$ is Poisson distributed. We reconstruct $X^{(23)}$ under log-loss, and use its matching link—*i.e.*, $X^{(23)}_{js}$ is Bernoulli, a logistic model. From the same starting point, we measure the training loss of alternating projections using either a Newton step or a gradient step for each projection. The results in Fig. 2 are averaged over five trials, and clearly favour the Newton step.

The optimization over $\mathcal{L}$ is a non-convex problem, and the inherent complexity of the objective can vary dramatically from problem to problem. In some problems, our alternating Newton-projection approach appears to perform better than standard alternatives; however, we have found other problems where existing algorithms typically lead to better scores.

Logistic PCA is an example of where alternating projections can outperform an EM algorithm specifically designed for this model [29]. We use a binarized version of the rating matrix described above, whose entries indicate whether a user rated a movie. For the same settings, $k = 25$ and no regularization, we compare the test set error of the model learned using EM[4] vs. the same model learned using alternating projections.[5] Each method is run to convergence, a change of less than one percent in the objective between iterations, and the experiment is repeated ten times. The test error metric is balanced error rate, the average of the error rates on held-out positive and negative entries, so lower is better. Using the EM optimizer, the test error is 0.1813 ± 0.020; using alternating projections, the test error is 0.1253 ± 0.0061 (errors bars are 1-standard deviation).

Logistic Fast-MMMF is a variant of Fast-MMMF which uses log-loss and its matching link instead of smoothed Hinge loss, following [47]. Alternating Newton-projection does not outperform the recommended optimizer, conjugate gradients[6]. To compare the behaviour on multiple trials run to convergence, we use a small sample of the Netflix ratings data (250 users and 700 movies). Our evaluation metric is prediction accuracy on the held-out ratings under mean absolute error. On five repetitions with a rank $k = 20$ factorization, moderate ℓ_2-regularization ($\lambda = 10^5$), and for the Newton step the Armijo procedure described above, the conjugate gradient solver yielded a model with zero error; the alternating-Newton method converged to models with test error > 0.015.

The performance of alternating Newton-projections suffers when k is large. On a larger Netflix instance (30000 users, 2000 movies, 1.1M ratings) an iteration

[4] We use Schein et al.'s implementation of EM for Logistic PCA.

[5] We use an Armijo line search in the Newton projection, considering step lengths as small as $\eta = 2^{-4}$.

[6] We use Rennie et al.'s conjugate gradient code for Logistic Fast-MMMF.

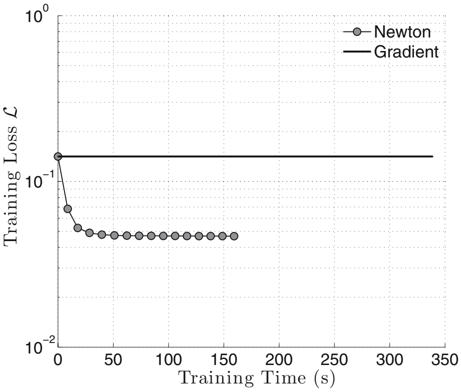


Fig. 2. Gradient vs. Newton steps in alternating projection

of our approach takes almost 40 minutes when $k = 100$; an iteration of the conjugate gradient implementation takes 80–120 seconds.

8 Conclusion

The vast majority of matrix factorization algorithms differ only in a small number of modeling choices: the prediction link, loss, constraints, data weights, and regularization. We have shown that a wide variety of popular matrix factorization approaches, such as weighted SVD, NMF, and MMMF, and pLSI can be distinguished by these modeling choices. We note that this unified view subsumes both dimensionality reduction and clustering in matrices using the bilinear model $X \approx f(UV^T)$, and that there is no conceptual difference between single- and multiple-matrix factorizations.

Exploiting a common property in matrix factorizations, decomposability of the loss, we extended a well-understood alternating projection algorithm to handle weights, bias/margin terms, ℓ_1-regularization, and clustering constraints. In each projection, we recommended using Newton's method: while the Hessian is large, it is also block diagonal, which allows the update for a factor to be performed independently on each of its rows. We tested the relative merits of alternating Newton-projections against plain gradient descent, an existing EM approach for logistic PCA, and a conjugate gradient solver for MMMF.

Acknowledgments

The authors thank Jon Ostlund for his assistance in merging the Netflix and IMDB data. This research was funded in part by a grant from DARPA's RADAR program. The opinions and conclusions are the authors' alone.

References

1. Golub, G.H., Loan, C.F.V.: Matrix Computions, 3rd edn. John Hopkins University Press (1996)
2. Hofmann, T.: Probabilistic latent semantic indexing. In: SIGIR, pp. 50–57 (1999)
3. Singh, A.P., Gordon, G.J.: Relational learning via collective matrix factorization. In: KDD (2008)
4. Rish, I., Grabarnik, G., Cecchi, G., Pereira, F., Gordon, G.: Closed-form supervised dimensionality reduction with generalized linear models. In: ICML (2008)
5. Lee, D.D., Seung, H.S.: Algorithms for non-negative matrix factorization. In: NIPS (2001)
6. Collins, M., Dasgupta, S., Schapire, R.E.: A generalization of principal component analysis to the exponential family. In: NIPS (2001)
7. Gordon, G.J.: Approximate Solutions to Markov Decision Processes. PhD thesis. Carnegie Mellon University (1999)
8. Gordon, G.J.: Generalized^2 linear^2 models. In: NIPS (2002)
9. Bregman, L.: The relaxation method of finding the common points of convex sets and its application to the solution of problems in convex programming. USSR Comp. Math and Math. Phys. 7, 200–217 (1967)
10. Censor, Y., Zenios, S.A.: Parallel Optimization: Theory, Algorithms, and Applications. Oxford University Press, Oxford (1997)
11. Azoury, K.S., Warmuth, M.K.: Relative loss bounds for on-line density estimation with the exponential family of distributions. Mach. Learn. 43, 211–246 (2001)
12. Banerjee, A., Merugu, S., Dhillon, I.S., Ghosh, J.: Clustering with Bregman divergences. J. Mach. Learn. Res. 6, 1705–1749 (2005)
13. Forster, J., Warmuth, M.K.: Relative expected instantaneous loss bounds. In: COLT, pp. 90–99 (2000)
14. Aldous, D.J.: Representations for partially exchangeable arrays of random variables. J. Multivariate Analysis 11(4), 581–598 (1981)
15. Aldous, D.J.: 1. In: Exchangeability and related topics, pp. 1–198. Springer, Heidelberg (1985)
16. Welling, M., Rosen-Zvi, M., Hinton, G.: Exponential family harmoniums with an application to information retrieval. In: NIPS (2005)
17. Welling, M., Chemudugunta, C., Sutter, N.: Deterministic latent variable models and their pitfalls. In: SDM (2008)
18. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet allocation. J. Mach. Learn. Res. 3, 993–1022 (2003)
19. Koenker, R., Bassett, G.J.: Regression quantiles. Econometrica 46(1), 33–50 (1978)
20. Tibshirani, R.: Regression shrinkage and selection via the lasso. J. Royal. Statist. Soc. B. 58(1), 267–288 (1996)
21. Ding, C.H.Q., Li, T., Peng, W.: Nonnegative matrix factorization and probabilistic latent semantic indexing: Equivalence chi-square statistic, and a hybrid method. In: AAAI (2006)
22. Ding, C.H.Q., He, X., Simon, H.D.: Nonnegative Lagrangian relaxation of -means and spectral clustering. In: Gama, J., Camacho, R., Brazdil, P.B., Jorge, A.M., Torgo, L. (eds.) ECML 2005. LNCS (LNAI), vol. 3720, pp. 530–538. Springer, Heidelberg (2005)
23. Buntine, W.L., Jakulin, A.: Discrete component analysis. In: Saunders, C., Grobelnik, M., Gunn, S., Shawe-Taylor, J. (eds.) SLSFS 2005. LNCS, vol. 3940, pp. 1–33. Springer, Heidelberg (2006)

24. Gabriel, K.R., Zamir, S.: Lower rank approximation of matrices by least squares with any choice of weights. Technometrics 21(4), 489–498 (1979)
25. Srebro, N., Jaakola, T.: Weighted low-rank approximations. In: ICML (2003)
26. Hartigan, J.: Clustering Algorithms. Wiley, Chichester (1975)
27. Ke, Q., Kanade, T.: Robust l_1 norm factorization in the presence of outliers and missing data by alternative convex programming. In: CVPR, pp. 739–746 (2005)
28. Paatero, P., Tapper, U.: Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. Environmetrics 5, 111–126 (1994)
29. Schein, A.I., Saul, L.K., Ungar, L.H.: A generalized linear model for principal component analysis of binary data. In: AISTATS (2003)
30. Srebro, N., Rennie, J.D.M., Jaakkola, T.S.: Maximum-margin matrix factorization. In: NIPS (2004)
31. Rennie, J.D.M., Srebro, N.: Fast maximum margin matrix factorization for collaborative prediction. In: ICML, pp. 713–719. ACM Press, New York (2005)
32. Nocedal, J., Wright, S.J.: Numerical Optimization. Series in Operations Research. Springer, Heidelberg (1999)
33. Schmidt, M., Fung, G., Rosales, R.: Fast optimization methods for L1 regularization: A comparative study and two new approaches. In: Kok, J.N., Koronacki, J., Lopez de Mantaras, R., Matwin, S., Mladenič, D., Skowron, A. (eds.) ECML 2007. LNCS (LNAI), vol. 4701, pp. 286–297. Springer, Heidelberg (2007)
34. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press, Cambridge (2004)
35. Pereira, F., Gordon, G.: The support vector decomposition machine. In: ICML, pp. 689–696. ACM Press, New York (2006)
36. Zhu, S., Yu, K., Chi, Y., Gong, Y.: Combining content and link for classification using matrix factorization. In: SIGIR, pp. 487–494. ACM Press, New York (2007)
37. Yu, K., Yu, S., Tresp, V.: Multi-label informed latent semantic indexing. In: SIGIR, pp. 258–265. ACM Press, New York (2005)
38. Yu, S., Yu, K., Tresp, V., Kriegel, H.P., Wu, M.: Supervised probabilistic principal component analysis. In: KDD, pp. 464–473 (2006)
39. Cohn, D., Hofmann, T.: The missing link–a probabilistic model of document content and hypertext connectivity. In: NIPS (2000)
40. Long, B., Wu, X., Zhang, Z.M., Yu, P.S.: Unsupervised learning on k-partite graphs. In: KDD, pp. 317–326. ACM Press, New York (2006)
41. Long, B., Zhang, Z.M., Wú, X., Yu, P.S.: Spectral clustering for multi-type relational data. In: ICML, pp. 585–592. ACM Press, New York (2006)
42. Long, B., Zhang, Z.M., Wu, X., Yu, P.S.: Relational clustering by symmetric convex coding. In: ICML, pp. 569–576. ACM Press, New York (2007)
43. Long, B., Zhang, Z.M., Yu, P.S.: A probabilistic framework for relational clustering. In: KDD, pp. 470–479. ACM Press, New York (2007)
44. Banerjee, A., Basu, S., Merugu, S.: Multi-way clustering on relation graphs. In: SDM (2007)
45. Netflix: Netflix prize dataset (January 2007), `http://www.netflixprize.com`
46. Internet Movie Database Inc.: IMDB alternate interfaces (January 2007), `http://www.imdb.com/interfaces`
47. Rennie, J.D.: Extracting Information from Informal Communication. PhD thesis, Massachusetts Institute of Technology (2007)