Application of a Gaussian, Missing-Data Model to Product Recommendation

William J. J. Roberts, *Senior Member, IEEE*

***Abstract*—A Gaussian, missing-data model is applied to predict product ratings. Vectors of product ratings from users are assumed to be independent and identically distributed. Two approaches for parameter estimation in this model are studied: Little and Rubin's expectation-maximization algorithm and McMichael's modified stochastic gradient descent approach. The resulting estimates are used in minimum mean squared error prediction of product ratings using the conditional mean. On a large dataset, performance using McMichael's approach is better than reported performance of the popular matrix factorization approach.**

***Index Terms*—Conditional mean, expectation maximization, gradient descent, maximum likelihood.**

I. Introduction

A RECOMMENDER system predicts preferences of users for products. Consider a recommender system involving n users and k products. An *observed rating* is a rating given by one of the n users to one of the k products. Any rating not observed is a *missing rating*. The total number of observed and missing ratings is nk. The product recommendation problem is to predict missing ratings. Applications for recommender systems include social networking, dating sites, and movie recommendation [7].

In this letter, we assume the ratings from each user are k-dimensional Gaussian random vectors. The k-dimensional vectors from different users are assumed to be independent and identically distributed (*iid*). The common mean and covariance are estimated from the observed ratings. Due to desirable asymptotic properties—large datasets with large n and k are common in real applications—we focus here on maximum likelihood (ML) estimation. An explicit ML estimate of the mean is known which is not, in general, the arithmetic average of the observed ratings. No explicit ML estimate of the covariance is known and we study two iterative algorithms: an expectation-maximization (EM) algorithm studied by Little and Rubin [9] and a modified stochastic gradient descent algorithm due to McMichael [10].

Given estimates of the mean and covariance, minimum mean squared error (MMSE) prediction of the missing ratings is performed here using the conditional mean. The conditional mean has a well-known, explicit linear form in terms of subvectors and submatrices of, respectively, the mean and covariance [1, Th. 2.5.1]. Conditional mean prediction of missing ratings using the estimated mean and covariance achieves a root mean squared error (RMSE) of 0.8907 on the well-known Netflix data-set (see, e.g., [3]).

A popular approach for large-scale, real-world recommender systems is *matrix factorization*, see e.g., [7] and references therein. In this approach, a sparse $k \times n$ matrix containing all observed ratings is approximated by the product of two rectangular matrices. Missing ratings are predicted by the inner product of appropriate columns and rows from the two rectangular matrices. In [7, Fig. 4], plain matrix factorization is reported to achieve an RMSE of above 0.902 on the Netflix data.

Other studies have performed MMSE prediction of missing ratings using an estimated mean and covariance. The estimation steps, however, are different from those studied here. Schwaighofer *et al.* [11] derive an EM approach for a Bayesian formulation in additive noise. Using a similar model to [11], Bohnert and Schmidt [4] estimate a structured covariance matrix using a Gibbs sampling approach. Lawrence and Urtasun [8] treat the mean of a Gaussian distribution as the product of two matrices and estimate these matrices using stochastic gradient descent.

The remainder of this paper is organized as follows. In Section II, we specify the model, discuss ML estimation of its parameter, and MMSE prediction of missing ratings. In Section III, we detail the implementation of the techniques and the results obtained. Section IV concludes with comments and suggestions for further work.

II. A Gaussian, Missing-Data Model

A. Model Specification

Let Z_t denote a k-dimensional Gaussian random vector representing both observed and missing ratings of the tth user, $t = 1, 2, \ldots, n$. Let $Z^n = \{Z_1, \ldots, Z_n\}$ represent ratings from all n users and assume these ratings are *iid*. Let μ be the $k \times 1$ mean vector of Z_t and let R be the $k \times k$ covariance matrix of Z_t. Let Y_t denote a k_t-dimensional subvector of Z_t representing the $0 < k_t \leqslant k$, observed ratings of the tth user. Let I be the $k \times k$ identity matrix. Let H_{y_t} be a $k_t \times k$ submatrix of I where the rows of I corresponding to the indices of the missing ratings of the tth user have been deleted. The integer k_t and the matrix H_{y_t} are assumed to be deterministic. Thus $Y_t = H_{y_t} Z_t$ is a Gaussian random vector with k_t-dimensional mean $\mu_{y_t} = H_{y_t}\mu$ and $k_t \times k_t$ covariance $R_{y_t} = H_{y_t} R H'_{y_t}$, where $'$ denotes matrix or vector transpose. Note μ_{y_t} and R_{y_t} are a subvector and a principal submatrix of μ and R respectively. If R is a symmetric and positive semi-definite (psd) matrix then all of its principal submatrices are also symmetric and psd [6, Cor. 4.2.2]. Thus all such R_{y_t} are valid covariance matrices.

Manuscript received November 17, 2009; revised February 13, 2010. First published March 15, 2010; current version published April 09, 2010. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Magnus Jansson

The author is with the Department of Engineering Management and Systems Engineering, George Washington University, Washington, DC 20052 USA (e-mail: wroberts@gwu.edu).

Digital Object Identifier 10.1109/LSP.2010.2045543

Let $Y^n = \{Y_1, \ldots, Y_n\}$ denote observed ratings from all n users. Let y_t denote a realization of Y_t and y^n denote a realization of Y^n. Independence of Y^n follows from the independence of Z^n. The Y^n, however, are not *iid*. The probability density function of Y^n is thus

$$p(y^n;\mu,R) = \prod_{t=1}^{n} \frac{\exp\left(-(y_t-\mu_{y_t})' R_{y_t}^{-1}(y_t-\mu_{y_t})/2\right)}{(2\pi)^{k_t/2}|R_{y_t}|^{1/2}}. \quad (1)$$

B. Parameter Estimation

We aim for the μ and R that maximize $p(y^n;\mu,R)$. Joint maximization over μ and R is not possible in closed form. Let $\hat{\mu}$ denote the ML estimate of μ. Assuming a known R, McMichael [10] obtained the following explicit $\hat{\mu}$ by equating to zero and solving the derivative of $\log p(y^n;\mu,R)$ with respect to μ

$$\hat{\mu} = \left(\sum_{t=1}^{n} H'_{y_t} R_{y_t}^{-1} H_{y_t}\right)^{-1} \sum_{t=1}^{n} H'_{y_t} R_{y_t}^{-1} y_t. \quad (2)$$

With $R = I$ the elements of $\hat{\mu}$ are the arithmetic means of the observed ratings.

No explicit ML estimate of R is known. We resort to two iterative techniques that aim to increase the likelihood with each iteration. The first is the EM algorithm studied by Little and Rubin [9, Sec. 11.2.1]. Iterations of the EM algorithm are guaranteed to not decrease the likelihood [9]. Let X_t denote a $(k-k_t)$-dimensional random vector representing all the missing ratings of the tth user. Let H_{x_t} be a $(k-k_t)\times k$ submatrix of I where the rows of I corresponding to the indices of the observed ratings have been deleted. Then

$$Z_t = H'_{y_t} Y_t + H'_{x_t} X_t. \quad (3)$$

Let $\hat{R}^i$ denote an existing estimate of R. Assuming a known μ, an EM iteration that provides an updated estimate of R is given by

$$\hat{R}^{i+1} = \arg\max_R E\left\{\log p(Z^n;\mu,R) | Y^n = y^n; \mu, \hat{R}^i\right\}. \quad (4)$$

As $Z_t \sim \mathcal{N}(\mu, R)$ and applying the independence of Z^n

$$\hat{R}^{i+1} = \frac{1}{n}\sum_{t=1}^{n} E\left\{(Z_t-\mu)(Z_t-\mu)' | Y_t = y_t; \mu, \hat{R}^i\right\}. \quad (5)$$

After substituting (3) into (5) we require conditional moments of X_t. The required conditional distribution is Gaussian [1, Th. 2.5.1] with mean

$$\hat{X}_t = E\{X_t|Y_t = y_t; \mu, R\} = R_{x_t y_t} R_{y_t}^{-1}(y_t - \mu_{y_t}) + \mu_{x_t} \quad (6)$$

and covariance

$$E\left\{(X_t-\hat{X}_t)(X_t-\hat{X}_t)'|Y_t = y_t;\mu,R\right\} = R_{x_t} - R_{x_t y_t} R_{yt}^{-1} R'_{x_t y_t} \quad (7)$$

where $R_{x_t} = H_{x_t} R H_{x_t}{}'$, $R_{x_t y_t} = H_{x_t} R H_{y_t}{}'$, $\mu_{x_t} = H_{x_t}\mu$ and $\mu_{y_t} = H_{y_t}\mu$. The resulting EM iteration is thus

$$\hat{R}^{i+1} = \frac{1}{n}\sum_{t=1}^{n}(\hat{Z}_t-\mu)(\hat{Z}_t-\mu)' + H_{x_t}\left(\hat{R}^i_{x_t} - \hat{R}^i_{x_t y_t}\left(\hat{R}^i_{y_t}\right)^{-1}\left(\hat{R}^i_{x_t y_t}\right)'\right) H_{x_t}{}' \quad (8)$$

where $\hat{Z}_t = H'_{y_t} y_t + H'_{x_t}\hat{X}_t$. Note that the EM approach can be applied to the estimation of μ resulting in the estimate $\sum_t \hat{Z}_t/n$ [9, eq. (11.6)]. We prefer the ML μ estimate (2) as, for a given R, this estimate maximizes the likelihood whereas the EM estimate guarantees only to not decrease it.

The second algorithm that we consider for R estimation is McMichael's modified gradient descent algorithm [10]. Assuming a known μ, an updated estimate of R is given by

$$\hat{R}^{i+1} = \hat{R}^i + \gamma \hat{R}^i \left(\frac{d}{dR}\log p(y^n;\mu,R)|_{R=\hat{R}^i}\right)\hat{R}^i \quad (9)$$

for $\gamma > 0$. The required derivative is given by

$$\frac{d}{dR}\log p(y^n;\mu,R)|_{R=\hat{R}^i} = -\frac{1}{2}\sum_{t=1}^{n} H'_{y_t} \times \left((\hat{R}^i_{y_t})^{-1} - (\hat{R}^i_{y_t})^{-1}(y_t-\mu_{y_t})(y_t-\mu_{y_t})'(\hat{R}^i_{y_t})^{-1}\right) H_{y_t}. \quad (10)$$

Henceforth we will refer to the alternate estimation of μ using (2) and R using (8) as the EM algorithm. The alternate estimation of μ using (2) and R using (9) and (10) will be referred to as McMichael's algorithm. The estimates obtained at the ith iteration of either algorithm are denoted $\hat{\mu}^i$ and $\hat{R}^i$.

C. Initialization

Both McMichael's and the EM algorithm require initialization. Let $\hat{\mu}^0$ denote an initial estimate for μ. Little and Rubin [9, Sec. 11.2.1] suggest using $\hat{\mu}^0$ equal to the arithmetic mean of the observations. Let N be the $k \times k$ diagonal matrix given by $N = \sum_{t=1}^{n} H'_{y_t} H_{y_t}$. The ith diagonal element of N thus equals the total number of ratings of the ith product. The arithmetic mean of the observations is given by $\hat{\mu}^0 = N^{-1}\sum_{t=1}^{n} H'_{y_t} y_t$ which is equivalent to (2) with $R = I$.

Let $\hat{R}^0_j$ denote an initial estimate for R, where $j = 1, \ldots, 4$, represents one of the four different initial estimates we study here. We restrict ourselves to $\hat{R}^0_j$ that are symmetric and psd as such matrices are valid covariances. A simple such initializer is $\hat{R}^0_1 = I$. Define the psd matrix

$$S = \sum_{t=1}^{n} H'_{y_t}(y_t - H_{y_t}\hat{\mu}^0)(y_t - H_{y_t}\hat{\mu}^0)' H_{y_t} \quad (11)$$

constituting an un-normalized sample covariance matrix. Little and Rubin [9, Sec. 11.2.1] suggest using the diagonal matrix whose elements correspond to the sample variances of the observed data. This matrix is given by $\hat{R}^0_2 = N^{-1}\mathrm{diag}(S)$ where $\mathrm{diag}(S)$ is the diagonal matrix consisting of the diagonal elements of S. The third initial estimate of R we study is the psd correlation matrix $\hat{R}^0_3 = \mathrm{diag}(S)^{-1/2} S\, \mathrm{diag}(S)^{-1/2}$. Off-diagonal elements of $\hat{R}^0_3$ are nonzero. As the diagonal elements of $\hat{R}_3^0$ are equal to one, this matrix may not be a good initializer when rating variances are far from one. An initial estimate of R with nonzero off-diagonal elements and diagonal elements equal to the sample variances is $\hat{R}_4^0 = N^{-1/2} S N^{-1/2}$.

D. Prediction of Missing Ratings

Performance of recommender systems is often measured using RMSE. Our criterion of choice in predicting missing ratings is thus MMSE. As is well known, MMSE prediction is accomplished by the conditional mean. Redefine X_t to be only those missing ratings that we wish to predict. Given a realization y_t from the tth user, the conditional mean of the desired missing ratings is the linear estimator $\hat{X}_t = E\{X_t | Y_t = y_t; \mu, R\}$ given by (6). This is implemented using appropriate submatrices and subvectors of the estimates of R and μ obtained from either McMichael's or the EM algorithm.

The constant covariance matrix specified by (7) is referred to as a partial covariance matrix [1, p. 36] and represents the covariance of the predictions. In applications it could be used, for example, to calculate confidence regions. A $(1-\alpha)$ two-sided confidence interval for the ith element of X_t, denoted $X_t(i)$, is given by $\hat{X}_t(i) \pm \sigma_{ii} z_{1-\alpha/2}$ where σ_{ii}^2 is the ith diagonal element of the appropriate partial covariance matrix and $z_{1-\alpha/2}$ is the quantile of the standard Gaussian distribution evaluated at $(1-\alpha/2)$. Alternatively, a confidence ellipsoid for X_t can be constructed.

III. Numerical Setup and Results

A. Implementation

The techniques of Section II were implemented in Matlab and augmented where appropriate with explicit calls to basic linear algebra subprograms (BLAS), see e.g., [5]. To conserve memory—R_{y_t} can be very large in real applications—the BLAS generally operated on a designated $k \times k$ memory location which held a $k_t \times k_t$ matrix when processing observations from the tth user. To illustrate the use of BLAS consider, for example, the ith iteration of McMichael's algorithm consisting of (2), (9), and (10) together with (1). For the tth observation, the matrix $R_{y_t} = H_{y_t} \hat{R}^i H'_{y_t}$ is formed by copying the relevant elements of the current estimate of R into the designated memory location. In this and other similar operations matrix multiplications are not required. Using the BLAS *spotrf*, the matrix R_{y_t} is overwritten by its triangular Cholesky decomposition L_{y_t}, where $R_{y_t} = L_{y_t} L'_{y_t}$. The BLAS *strsm* is used to calculate $L_{y_t}^{-1}(y_t - \mu_{y_t})$, followed by another call to *strsm* to calculate $R_{y_t}^{-1}(y_t - \mu_{y_t})$. The determinant in (1) is calculated using the identity $\log|R_{y_t}| = 2\sum_j \log((L_{y_t})_{jj})$ where $(L_{y_t})_{jj}$ is the jjth element of L_{y_t}. The matrix $R_{y_t}^{-1}(y_t - \mu_{y_t})(y_t - \mu_{y_t})' R_{y_t}^{-1}$, required in (10) is calculated using the BLAS *ssyrk*. The matrix $R_{y_t}^{-1}$ required in (2) is calculated using the BLAS *spotri* with the result overwritten into the designated memory location.

B. Results

The approaches described here were tested using data from the Netflix movie recommendation contest that ended in September 2009. In this dataset each of $n = 480\,189$ users rated some of $k = 17\,770$ movies. Ratings consisted of integers from 1 to 5. The majority of the data set constituted the *training-set*. The *probe-set* is a subset of the training-set. The *quiz-set* is a set of additional ratings not in the training-set used to measure performance. Users rated on average less than 2% of the k movies that they could have rated. Detailed specification and descriptions of the contest and the datasets are available in [3].

Performance in this competition was measured by RMSE. Recall that X_t is a random vector representing the subset of the missing ratings of the tth user that we wish to estimate and $\hat{X}_t$ is an estimate of X_t. Let x_t denote a particular realization of X_t used to measure performance. The RMSE of the estimates $\hat{X}^n = \{\hat{X}_1, \ldots, \hat{X}_n\}$ is denoted by ϵ and obtained using

$$\epsilon^2 = \frac{\sum_{t=1}^n (x_t - \hat{X}_t)'(x_t - \hat{X}_t)}{\sum_{t=1}^n l_t} \tag{12}$$

where $(k - k_t) \leqslant l_t \leqslant 0$ denotes the length of X_t, $\hat{X}_t$ and x_t.

Initial experiments were conducted using the Netflix dataset limited to the 100 movies with the greatest number of observed ratings. Here $k = 100$ and $n = 137\,328$. Users rated on average over 24% of the 100 movies that they could have rated. The portion of this reduced dataset corresponding to the probe-set was removed from the training-set and constituted the variables X^n to be predicted. Conditional mean prediction (6) of X^n was performed using R and μ estimates obtained using the EM algorithm. The EM algorithm was initialized with $\hat{\mu}^0$ and R_4^0. With each EM iteration, the likelihood increased, and the RMSE decreased. Iterations were ceased once the convergence criterion

$$n^{-1} \log p(y^n; \hat{R}^i, \hat{\mu}^i) - n^{-1} \log p(y^n; \hat{R}^{i-1}, \hat{\mu}^{i-1}) < \delta \tag{13}$$

with $\delta = .0005$ was satisfied. The number of iterations required for convergence was 27. The final log-likelihood normalized by n was -31.41. Ratings estimates above 5 were clipped to 5, and those below 1 were clipped to 1. The resulting RMSE was 0.9170.

The experiment was repeated using R and μ estimates obtained by McMichael's algorithm. This algorithm required the specification of an addition parameter, γ in (9) and we used $\gamma = 1 \times 10^{-5}$. The log likelihood and RMSE values were identical, to four significant figures, to those obtained by the EM algorithm. Similar monotonicity of the log-likelihood and RMSE with iteration count was observed. Convergence was slower, however, and required 35 iterations. The behavior and performance of neither algorithm varied greatly when different initial estimates of R were used.

Both algorithms aim for the ML parameter estimate. McMichael's algorithm, however, lacks the guarantee of the EM algorithm to never decrease the likelihood. With an appropriate choice of γ, our results here suggest that in practice the two algorithms can be equally stable. If γ is too large McMichael's algorithm can decrease the likelihood and produce a non-psd covariance estimate. Our choice of γ was the largest we found that avoided this.

The computational requirements of the two algorithms, however, are different. In the EM algorithm, the primary computation burdens are k-dimensional vector outer-products and multiplications involving a $(k - k_t) \times k_t$ matrix required in (8). In McMichael's algorithm the primary computational burdens are $k_t \times k_t$ matrix inversions and multiplications required in (10). For the full Netflix data where $k = 17\,770$ and k_t is on average < 210, the computational burden of the EM algorithm was too

TABLE I
RESULTS FOR MCMICHAEL'S ALGORITHM INITIALIZED WITH FOUR DIFFERENT INITIAL R ESTIMATES

	Initial estimate of R			
	$\hat{R}_1^0$	$\hat{R}_2^0$	$\hat{R}_3^0$	$\hat{R}_4^0$
RMSE using the initial estimates $\hat{R}_j^0$ & $\hat{\mu}^0$	1.053	1.053	.9207	.9203
Iterations i required for convergence	29	29	19	19
RMSE using final estimates $\hat{R}^i$ & $\hat{\mu}^i$	.8959	.8967	.8907	.8910
Final log likelihood $(\times 10^2)$ $n^{-1}\log p(y^n; \hat{R}^i, \hat{\mu}^i)$	-2.481	-2.473	-2.459	-2.458

large and only McMichael's algorithm could be implemented on the available hardware.

McMichael's algorithm was thus applied to estimate a mean and covariance using the much-larger, full Netflix training-set. The setup was similar to that described above except we used $\delta = 0.2$ to speed-up McMichael's algorithm by reducing the number of iterations. Each iteration required approximately 2 h of processing time on a 2.6 GHz Intel Quad 4 processor. The resulting mean and covariance estimates were used in MMSE prediction of the full quiz-set. As before, the likelihood and RMSE changed monotonically with each iteration. The changes in RMSE from iteration to iteration were very small immediately prior to convergence. In this experiment, performance was sensitive to the initial estimates of R. In Table I, we provide the performance of initial estimates, iterations required for convergence, performance of final estimates, and log likelihood of final estimates for McMichael's algorithm initialized with the four different initial R estimates from Section II-C.

The best RMSE in Table I is better than the RMSE for plain matrix factorization mentioned earlier. It is, however, much higher than the best RMSE of 0.8553 obtained during the Netflix contest on identical data. The significantly improved performance was obtained by a team using linear regression to combine the predictions here with predictions from over 100 different systems, including, e.g., systems using matrix factorization. Let $\hat{X}_t^j$ be an estimate of X_t obtained by the jth system, $j = 1, \ldots, J$. Let $(\hat{X}^n)^j = \{\hat{X}_1^j, \hat{X}_2^j, \ldots, \hat{X}_n^j\}$ denote the set of estimates from the jth system. Let $\mathbf{X_t}$ be a $l_t \times J$ matrix whose jth column is $\hat{X}_t^j$. The estimate of X_t obtained by linear regression is given by [2]

$$\hat{X}_t = \mathbf{\hat{X}_t}\left(\sum_{t=1}^{n}\mathbf{\hat{X}_t'}\mathbf{\hat{X}_t}\right)^{-1}\left(\sum_{t=1}^{n}\mathbf{\hat{X}_t'}x_t\right). \qquad (14)$$

Generally x^n is not available, and thus the last bracketed term in (14) can not be explicitly evaluated. In the Netflix competition, implementation of (14) could be accomplished by assigning the probe-set to be x^n. The product of the two bracketed terms in (14) can then be evaluated using systems trained without the probe-set. An alternative approach [2] relied on knowing the RMSE of $(\hat{X}^n)^j, \forall j$ obtained during the Netflix contest by submitting $(\hat{X}^n)^j$ to the contest administrators. If $\sum_t x_t' x_t$ is known, and ϵ^2 in (12) is known for each $(\hat{X}^n)^j$, then the last bracketed term in (14) can be found. The term $\sum_t x_t' x_t$ can be elicited from the RMSE of constant estimates, e.g., $\hat{X}_t = 1, \forall t$.

In another experiment, confidence intervals as described in Section II.D were calculated. Using the mean and covariance estimated by McMichael's algorithm from the training-set, a confidence interval was calculated for each MMSE rating prediction in the quiz set. We then checked how frequently the confidence interval contained the true rating. With $(1-\alpha) = .95$, 94.2% of the intervals contained the true rating. With $(1-\alpha) = .99$, 98.1% of the intervals contained the true rating. Thus the practical performance of these confidence intervals is consistent with that predicted by theory.

IV. CONCLUSION AND FURTHER WORK

Even though it lacks the guarantee of the EM algorithm to never decrease likelihood, we demonstrate that McMichael's algorithm performs similarly to the EM algorithm on a small dataset. McMichael's algorithm has the advantage that it can be scaled up to the large Netflix dataset. On this dataset McMichael's algorithm together with MMSE prediction demonstrate better performance than that reported for plain matrix factorization.

Performance of matrix factorization has been improved by accounting for so-called user and product biases. Further improvements have been obtained by allowing these biases to vary with time, see [7] and references therein. One potential avenue for further work is to investigate applying similar normalizations to the approaches described here.

ACKNOWLEDGMENT

The author thanks C. J. Willy, J. Sill, L. W. Mackey, members of "The Ensemble," and the anonymous reviewers for their careful reading of the manuscript and thoughtful comments.

REFERENCES

[1] T. W. Anderson, *An Introduction to Multivariate Statistical Analysis*, 2nd ed. New York: Wiley, 1984.
[2] R. Bell, Y. Koren, and C. Volinsky, The BellKor Solution to the Netflix Prize AT&T Labs-Res., 2007, Tech. Rep.
[3] J. Bennett and S. Lanning, "The Netflix prize," in *Proc. KDD-Cup and Workshop at the 13th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, San Jose, CA, 2007, pp. 3–6.
[4] F. Bohnert, D. F. Schmidt, and I. Zukerman, "Spatial processes for recommender systems," in *Proc. of the 21st Int. Joint Conf. on Artificial Intelligence*, Pasadena, CA, 2009.
[5] J. Dongarra, "Basic linear algebra subprograms technical forum standard," *Int. J. High Perf. Applicat. Supercomput.*, vol. 16, no. 1, pp. 1–111, 2002.
[6] G. G. Golub and C. F. Van Loan, *Matrix Computations*, 2nd ed. Baltimore, MD: Johns Hopkins Univ. Press, 1994.
[7] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *IEEE Computer*, vol. 42, no. 8, pp. 42–49, Aug. 2009.
[8] N. D. Lawrence and R. Urtasun, "Non-linear matrix factorization with Gaussian processes," in *Proc. 26th Annu. Int. Conf. Machine Learning*, Montreal, QC, Canada, 2009, pp. 601–608.
[9] R. J. A. Little and D. B. Rubin, *Statistical Analysis With Missing Data*, 2nd ed. Hoboken, NJ: Wiley-Interscience, 2002.
[10] D. W. McMichael, "Estimating Gaussian mixture models from data with missing features," in *Proc. 4th Int. Symp. Sig. Proc. and its Apps.*, Gold Coast, Australia, Aug. 1996, pp. 377–378.
[11] A. Schwaighofer, V. Tresp, and K. Yu, "Learning Gaussian process kernels via hierarchical Bayes," *Adv. Neural Inform. Process. Syst.*, vol. 17, pp. 1209–1216, 2005.