Dynamic Matrix Factorization: A State Space Approach

John Z. Sun, Kush R. Varshney, and Karthik Subbian

Abstract

Matrix factorization from a small number of observed entries has recently garnered much attention as the key ingredient of successful recommendation systems. One unresolved problem in this area is how to adapt current methods to handle changing user preferences over time. Recent proposals to address this issue are heuristic in nature and do not fully exploit the time-dependent structure of the problem. As a principled and general temporal formulation, we propose a dynamical state space model of matrix factorization. Our proposal builds upon probabilistic matrix factorization, a Bayesian model with Gaussian priors. We utilize results in state tracking, i.e. the Kalman filter, to provide accurate recommendations in the presence of both process and measurement noise. We show how system parameters can be learned via expectation-maximization and provide comparisons to current published techniques.

Index Terms

collaborative filtering, Kalman filtering, recommendation systems, expectation-maximization, learning

I. Introduction

Matrix factorization (MF), the decomposition of a matrix into a product of two simpler matrices, has a long and storied history in statistics, signal processing, and machine learning for high-dimensional data analysis [1]. The approach garnered much attention for its successful application to recommendation systems based on collaborative filtering, including the Netflix prize problem [2]. Recommendation is of interest in a variety of domains. The most common examples are recommending movies, television shows, or songs that a particular individual will rate highly, but there are many other examples. Business analytics examples from marketing and salesforce management include recommending products to a salesperson to cross-sell and upsell that a particular customer is likely to purchase, and recommending a sales team to a sales manager that will be able to successfully sell to a particular business customer.

In these domains, customer preferences often follow a trajectory over time. Customers may be interested in basic products at first and then higher-end products later, or products for toddlers first and for adolescents later; customers may need a sales team with greater relationship-building expertise at first and technical expertise later. Additionally, we can distinguish recommendation for discovery and recommendation for consumption; new items are recommended in the former whereas the same item may be repeatedly recommended in the latter.

The MF approach to collaborative filtering usually includes Frobenius-norm regularization [2], which is supported by a linear-Gaussian probabilistic model known as *probabilistic matrix factorization* or PMF [3]. Due to its linear-Gaussian nature, PMF lends itself to incorporating temporal trajectories through the state space representation of linear dynamical systems [4] and algorithms for estimation based on the Kalman filter [5], [6]. We propose a general recommendation model of this form and develop an expectation-maximization (EM) algorithm to learn the model parameters from data. The Kalman filter and Rauch-Tung-Striebel (RTS) smoother [7] appear in the expectation step of the EM.

Several recent works also address dynamic and temporal issues in recommendation. TimeSVD, a component of the Netflix prize-winning algorithm, addresses temporal dynamics through a specific parameterization with factors drifting from a central time, but unlike our general formulation can only handle limited temporal structure [2]. The probabilistic tensor factorization approach does not take temporal causality into account as we do [8]. The formulation of [9] is based on nearest neighbor collaborative filtering rather than MF, and is known to have scaling difficulties. The spatiotemporal Kalman filter of [10] has a limited state evolution and convergence issues, target tracking in recommendation space has no element of collaboration and requires prior knowledge of the 'recommendation space' [11], and the hidden Markov model for collaborative filtering only captures evolution of a known attribute over time among users [12]. The contribution of our paper is the development of a principled and general MF-based approach to recommendation and predictive analytics, including recommendation for consumption, in which the given data samples arrive over time and contain significant time dynamics.

J. Z. Sun is with the Department of Electrical Engineering and Computer Science and the Research Laboratory of Electronics, Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: johnsun@mit.edu).

K. R. Varshney and K. Subbian are with the Business Analytics and Mathematical Sciences Department, IBM Thomas J. Watson Research Center, Yorktown Heights, NY 10598 USA (e-mail: krvarshn@us.ibm.com,mailtokarthik@in.ibm.com).

II. Probabilistic Matrix Factorization

Recommendation systems comprise N users and M items, along with some measure of preference represented by a matrix $O \in \mathbb{R}^{N \times M}$. For most practical applications, only a small fraction of the entries of O are observed and are usually corrupted by noise, quantization, and different interpretations of the scaling of preferences. In MF, each user and item is represented by a row vector of length K denoted u_i and v_j respectively, corresponding to weights of K latent factors. We concatenate the factors into matrices $U \in \mathbb{R}^{N \times K}$ and $V \in \mathbb{R}^{M \times K}$. The preference matrix is then $O = UV^T$, meaning the preference of user i for item j is $o_{ij} = \langle u_i, v_j \rangle$, a common assumption in recommendation systems [2].

Under MF, latent factors are learned from past responses of users rather than formulated from known attributes. Factors are not necessarily easy to interpret and change dramatically depending on the choice of K. The value of K is an engineering decision, balancing the tradeoff of forming a rich model to capture user behavior and being simple enough to prevent overfitting.

Given K, a common way to learn factor matrices and consequently the complete preference matrix O from limited observations is the following program:

$$\min_{U,V} \sum_{(i,j) \in \mathcal{O}} (o_{ij} - u_i v_j^T)^2 + \lambda_1 \|U\|_F^2 + \lambda_2 \|V\|_F^2, \tag{1}$$

where the set $\mathcal{O}$ contains observed preference entries, and (λ_1, λ_2) are regularization parameters. This program can be solved efficiently using stochastic gradient descent and has been experimentally shown to have excellent root mean-square error (RMSE) performance [2].

More recently, the regularization of the above program was motivated by assigning Gaussian priors to the factor matrices U and V respectively [3]. Coined PMF, this Bayesian formulation means (1) is justified as producing the *maximum a posteriori* (MAP) estimate for this prior. In this case, the regularization parameters λ_1 and λ_2 are effectively signal-to-noise ratios (SNR). Since O is not a linear function of latent factors, the MAP estimate does not in general produce the best RMSE performance, which is the measure commonly desired in recommendation systems. However, wisdom gained from the Netflix Challenge and experimental validation from [3] show that the MAP estimate provides very competitive RMSE performance compared to other approximation methods.

III. State Space Model

Given the success of MAP estimation in linear-Gaussian PMF models and our interest in capturing time dynamics, we propose a linear-Gaussian dynamical state space model of MF whose MAP estimates can be obtained using Kalman filtering. We assume that user factors $\mathbf{u}_i(t)$ are functions of time and hence states in the state space model, with bold font indicating the vector being random. In our proposed model, we have coupled dynamical systems, and to adhere to typical Kalman filter notation, we use $\mathbf{x}_{i,t} = \mathbf{u}_i(t)$ to denote the state of user i at time t.

For each user, the initial state $\mathbf{x}_{i,0}$ is distributed according to $\mathcal{N}(\mu_i, \Sigma_i)$, the multivariate Gaussian distribution with mean vector μ_i and covariance matrix Σ_i. The user-factor evolution is linear according to the generally non-stationary transition process $A_{i,t}$ and contains transition process noise $\mathbf{w}_{i,t} \sim \mathcal{N}(0, Q_{i,t})$ to capture variability of individuals. Taken together, the state evolution is described by the set of equations:

$$\mathbf{x}_{i,t} = A_{i,t}\mathbf{x}_{i,t-1} + \mathbf{w}_{i,t} \qquad i = 1, \ldots, N. \tag{2}$$

We assume that item factors evolve very slowly and can be considered constant over the time frame that preferences are collected. Also, due to the sparsity of user preference observations, a particular user-item pair at a given time t may not be known. Thus, we incorporate the item factors through a non-stationary linear measurement process $H_{i,t}$ which is composed of subsets of rows of the item factor matrix V based on item preferences observed at time t by user i. Note that all $H_{i,t}$ are subsets of the same fixed V and are coupled in this way. We also include measurement noise $\mathbf{z}_{i,t} \sim \mathcal{N}(0, R_{i,t})$ in the model. The overall observation model is:

$$y_{i,t} = H_{i,t}\mathbf{x}_{i,t} + \mathbf{z}_{i,t} \qquad i = 1, \ldots, N. \tag{3}$$

The product $H_{i,t}\mathbf{x}_{i,t}$ in (3) parallels the $\langle u_i, v_j \rangle$ product in Sec. II. Again adhering to Kalman filter notation, we use $y_{i,t}$ to denote the observations, corresponding to the observed entries of O, now a tensor in $\mathbb{R}^{M \times N \times T}$.

The state space model can be generalized in many different ways that may be relevant to recommendation systems, including non-Gaussian priors, nonlinear process transformation and measurement models, and continuous-time dynamics. We focus on the linear-Gaussian assumption and defer discussion on extensions to Sec. VI.

IV. Factorization via Learning

In the regularized formulation (1), the MF task is learning user and item factors given sparse observations. In our dynamical model, MF is composed of two dependent tasks: learning model parameters that govern the motion of states, and performing MAP estimation of user factors.

Given the model parameters and access to observations for all past times $t = 1, \ldots, T$, the MAP estimate of $U(t)$ can be found using a noncausal Kalman filter called the RTS smoother. In the PMF setting, N of these RTS smoothers run in parallel, all of which share the same item factor matrix V in the measurement process. We call this architecture *collaborative Kalman filtering* (CKF). Let the Kalman estimates and covariances be defined as:

$$\hat{\mathbf{x}}_{i,t|s} = \mathbf{E}\left[\mathbf{x}_{i,t} \mid y_{i,1}, \ldots, y_{i,s}\right] \tag{4}$$

$$P_{i,t|s} = \mathrm{Var}\left(\mathbf{x}_{i,t} \mid y_{i,1}, \ldots, y_{i,s}\right). \tag{5}$$

Then, the Kalman filtering equations are as follows:

$$\hat{\mathbf{x}}_{i,t+1|t} = A_{i,t+1}\,\hat{\mathbf{x}}_{i,t|t}; \tag{6}$$

$$P_{i,t+1|t} = A_{i,t+1}\,P_{i,t|t}\,A_{i,t+1}^T + Q_{i,t}; \tag{7}$$

$$\hat{\mathbf{x}}_{i,t|t} = \hat{\mathbf{x}}_{i,t|t-1} + K_{i,t}\left(y_{i,t} - H_{i,t}\,\hat{\mathbf{x}}_{i,t|t-1}\right); \tag{8}$$

$$P_{i,t|t} = P_{i,t|t-1} - K_{i,t}\,H_{i,t}\,P_{i,t|t-1}. \tag{9}$$

Moreover, the RTS smoothing equations are:

$$\hat{\mathbf{x}}_{i,t|T} = \hat{\mathbf{x}}_{i,t|t} + J_{i,t}\left(\hat{\mathbf{x}}_{i,t+1|T} - \hat{\mathbf{x}}_{i,t+1|t}\right); \tag{10}$$

$$P_{i,t|T} = P_{i,t|t} + J_{i,t}\left(P_{i,t+1|T} - P_{i,t+1|t}\right)J_{i,t}^T; \tag{11}$$

$$P_{i,T,T-1|T} = \left(I - K_{i,T}\,H_{i,T}\right)A_{i,T}\,P_{i,T-1|T-1}; \tag{12}$$

$$\begin{aligned} P_{i,t,t-1|T} = {} & P_{i,t|t}\,J_{i,t-1}^T \\ & + J_{i,t}\left(P_{i,t+1,t|T} - A_{i,t+1}\,P_{i,t|t}\right)J_{i,t-1}^T, \end{aligned} \tag{13}$$

where

$$K_{i,t} = P_{i,t|t-1}\,H_{i,t}^T\left(H_{i,t}\,P_{i,t|t-1}\,H_{i,t}^T + R_{i,t}\right)^{-1}; \tag{14}$$

$$J_{i,t} = P_{i,t|t}\,A_{i,t+1}^T\,P_{i,t|t-1}^{-1}. \tag{15}$$

The CKF steps above fit naturally in the expectation step of the EM algorithm used to learn model parameters such as mean and covariance of the initial states, the transition process matrices, the process noise covariances, the measurement process matrices, and the measurement noise covariances. In learning the measurement process matrices, we also get an estimate for the item factor matrix V, which is the other ingredient in the MF problem. The EM algorithm proceeds by alternating between the expectation step in which the expectation of the likelihood of the observed data is evaluated for fixed parameters, and the maximization step in which the expected likelihood is maximized with respect to the parameters.

The model proposed in Sec. III is a fully general Gaussian state space model whose parameters could be learned, but would require many observation samples. In typical applications however, the observations are sparse and the parameters are heavily correlated in time; thus we simplify the model to reduce the number of parameters to learn. First, we make the approximation that parameters are independent of time, meaning they do not change during the observation period. Second, we take the initial states to be iid $\mathcal{N}(0, \sigma_U^2)$ for all users, meaning they are homogeneous enough to share similar scalings of preferences. Last, we assume both process and measurement noise are iid, reducing the learning to variances σ_Q^2 and σ_R^2 respectively. With these simplifications, the variance parameters can be chosen to maximize the log-likelihood as follows:

$$\hat{\sigma}_U^2 = \frac{1}{NK}\sum_{i=1}^{N}\mathrm{tr}\left(P_{i,0|T} + \hat{\mathbf{x}}_{i,0|T}\,\hat{\mathbf{x}}_{i,0|T}^T\right) \tag{16}$$

$$\begin{aligned} \hat{\sigma}_Q^2 = \frac{1}{NKT}\sum_{i=1}^{N}\sum_{t=1}^{T}\mathrm{tr}\Big\{ & \left(P_{i,t|T} + \hat{\mathbf{x}}_{i,t|T}\,\hat{\mathbf{x}}_{i,t|T}^T\right) \\ & - 2\left(P_{i,t,t-1|T} + \hat{\mathbf{x}}_{i,t|T}\,\hat{\mathbf{x}}_{i,t-1|T}^T\right)A^T \\ & + A\left(P_{i,t-1|T} + \hat{\mathbf{x}}_{i,t-1|T}\;\hat{\mathbf{x}}_{i,t-1|T}^T\right)A^T\Big\} \end{aligned} \tag{17}$$

$$\begin{aligned} \hat{\sigma}_R^2 = \frac{1}{|\mathcal{O}|}\Bigg[& \sum_{i=1}^{N}\sum_{t=1}^{T}\mathrm{tr}\left(y_{i,t}\,y_{i,t}^T\right) \\ & - 2\sum_{i=1}^{N}\sum_{t=1}^{T}\mathrm{tr}\left(y_{i,t}\;\hat{\mathbf{x}}_{i,t|T}^T\,H_{i,t}^T\right) \\ & + \sum_{i=1}^{N}\sum_{t=1}^{T}\mathrm{tr}\left(H_{i,t}\left(P_{i,t|T} + \hat{\mathbf{x}}_{i,t|T}\,\hat{\mathbf{x}}_{i,t|T}^T\right)H_{i,t}^T\right)\Bigg], \end{aligned} \tag{18}$$

where $\mathrm{tr}(\cdot)$ denotes the trace operator. Learning of general Σ, Q and R is discussed in Appendix B.

Expressions for the transition and measurement process matrices that maximize log-likelihood are derived to be:

$$\widehat{A} = A_2 \,\mathrm{inv}(A_1) \quad \text{where} \tag{19}$$
$$A_1 = \sum_{i=1}^{N}\sum_{t=1}^{T}(P_{i,t-1|T} + \hat{\mathbf{x}}_{i,t-1|T}\,\hat{\mathbf{x}}_{i,t-1|T}^T)$$
$$A_2 = \sum_{i=1}^{N}\sum_{t=1}^{T}(P_{i,t,t-1|T} + \hat{\mathbf{x}}_{i,t|T}\,\hat{\mathbf{x}}_{i,t-1|T}^T)$$
$$\widehat{V} = V_2 \,\mathrm{inv}(V_1) \quad \text{where} \tag{20}$$
$$V_1 = \sum_{i=1}^{N}\sum_{t=1}^{T}\left(P_{i,t|T} + \hat{\mathbf{x}}_{i,t|T}\,\hat{\mathbf{x}}_{i,t|T}^T\right)$$
$$V_2 = \sum_{i=1}^{N}\sum_{t=1}^{T}\left(\mathrm{fill}(y_{i,t})\,\hat{\mathbf{x}}_{i,t|T}^T\right).$$

Remembering that each $y_{i,t}$ is a subvector of $\mathbb{R}^M$ corresponding to items observed at time t, the fill operator expands its argument back to $\mathbb{R}^M$, with the observations in its appropriate positions and zeros elsewhere. Derivations for these parameters are discussed in Appendix B.

V. Empirical Results

To validate the effectiveness of Kalman learning compared to existing methods, we present results tested on generative data that follow a state space model. For this work, two main reasons led to our decision to use generative data rather than common datasets such as Netflix. First, a goal of the work is to understand how algorithms perform on preferences that evolved following a state space model. It is not clear that common datasets used in the recommendation systems literature match this model, and results would be too data-specific and not illuminating to the goal at hand. Second, a generative dataset gives insight on how the algorithms discussed perform in different parameter regimes, which is impossible in collected data.

We generate the item factor matrix V iid $\mathcal{N}(0, \sigma_V^2)$ and the initial user factor matrix $U(0)$ iid $\mathcal{N}(0, \sigma_U^2)$. Under the assumption that user factors do not change much with time, the stationary transition process matrix A is the weighted sum of the identity matrix and a random matrix, normalized so that the expected power of the state $\mathbf{x}_{i,t}$ is constant in time. We note that A can be more general with similar results, but the normalization is important so that preference observations do not change scales over time. Finally, iid noise is added to both the transition and measurement processes as described in (2) and (3). The observation triplets (i, j, t) are uniformly drawn iid from all possibilities from the preference tensor.

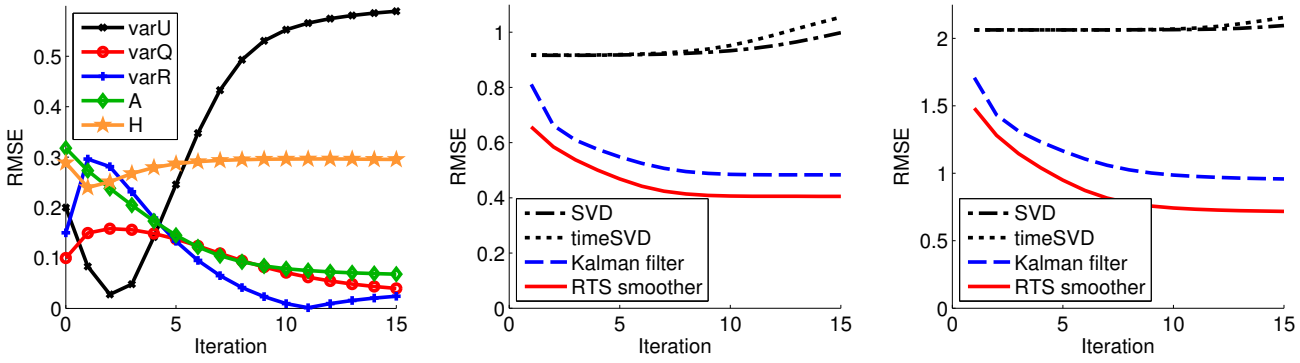
We present performance results for a particular choice of parameters in Fig. 1, expressed in RMSE. Space limitations prevent us from presenting results for other parameter choices, but they are similar when the SNR is reasonable. For arbitrary initial guesses of the parameters, we find learning of variances and process matrices to converge and stabilize after about 10-20 EM iterations. As a result, state tracking is reliable and approaches the lower bound specified by the Kalman smoother output when the parameters, including the item factor matrix V, are known a priori. The estimate for the entire preference tensor O also performs well, meaning that CKF is a valid approach for recommendation systems with data following a state space model.

In contrast, current algorithms such as SVD and timeSVD perform poorly on this dataset because they cannot handle general dynamics in user factors. Thus, the algorithm becomes confused and the estimates for the factor matrices tend to be close to zero, which is the best estimate when no data is observed. As shown in Fig. 2, the true trajectory of users may be that of an arc in factor space with additive perturbations. While CKF is able to track this evolution using smoothed and stable estimates, both SVD and timeSVD fail to capture this motion and hence have poor RMSE. SVD does not have temporal considerations and would give a stationary dot in the factor space. Meanwhile, timeSVD can only account for drift, meaning it can move in a linear fashion from a central point. In fact, this constraint leads to worse RMSE for most parameter choices than SVD because timeSVD overfits an incorrect model.

VI. Conclusion

In this paper, motivated by recommendation systems for consumption that arise in business analytics, we have proposed an extension to Gaussian PMF to take into account trajectories of user behavior. This has been done using a dynamical state space model from which predictions were made using the Kalman filter. We have derived an expectation-maximization algorithm to learn the parameters of the model from previously collected observations. We have validated the proposed CKF and shown its advantages over SVD and timeSVD on generated data. Future work underway includes testing and comparison on real-world collected data.

In contrast to heuristic and limited prior methods that incorporate time dynamics in recommendation, the approach proposed in this paper is a principled formulation that can take advantage of decades of developments in tracking and algorithms for



(a) (b) (c)

Fig. 1. For this testbench, we set model dimensions to be $(M, N, T, K) = (500, 500, 20, 5)$ and variances to be $(\sigma_U^2, \sigma_V^2, \sigma_Q^2, \sigma_R^2) = (1, 1, 0.05, 0.1)$. The item factor dynamics are controlled by A, which is a weighted average of identity and a random dense matrix. The sampling factor is 0.005, meaning only 0.5% of the entries of the preference matrix are observed. For the generated data and crude initial guesses of the parameters, RMSE performance is given for estimation of (a) Kalman parameters learned via EM; (b) user factors/states; and (c) the preference matrix. We observe that EM learning is effective in estimating parameters through noisy data, and this translates to better state tracking and estimation of the preference matrix. Convergence is fast and robust to initialization of parameters.

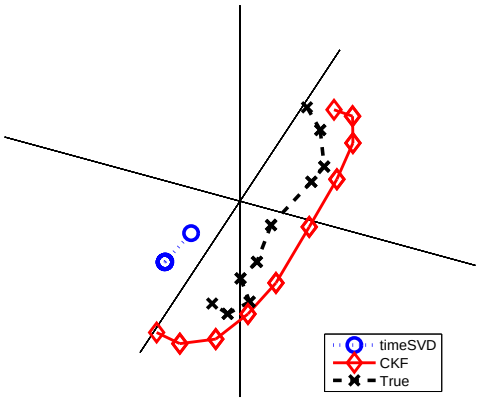


Fig. 2. State-tracking ability of CKF and timeSVD in three factor dimensions. The true user factors are well-tracked using CKF after parameters have been learned. However, timeSVD does not have flexibility to track general state evolutions and gives poor RMSE.

estimation. To break away from linearity assumptions, the extended or unscented Kalman filter can be used. Particle filtering can be used for non-Gaussian distributions, analogous to sampling-based inference in Bayesian PMF [13].

Based on the state space model, we can also include a control signal in future work to control what items are recommended to users. There are a variety of reasons to include a control signal: certain items may have corporate sponsorship or are high revenue items, or certain media files may be cached at a wireless base station and it is inexpensive to serve those items. The proposed dynamic formulation can also be extended in a control-theoretic way to address the cold start problem: recommending items to users with no or little previous expressed preferences.

Appendix A
Useful Facts From Matrix Theory

We present some useful facts for the below derivations [14]:

Fact 1. *For* $\mathbf{x} \sim \mathcal{N}(\mu, \Sigma)$,

$$\mathrm{E}[(\mathbf{x}-\mu)^T \Sigma^{-1} (\mathbf{x}-\mu)] = \mathrm{tr}\left(\Sigma^{-1}\, \mathrm{E}[(\mathbf{x}-\mu)(\mathbf{x}-\mu)^T]\right).$$

Fact 2.

$$\frac{d \log |X|}{dX} = \left(X^T\right)^{-1} = \left(X^{-1}\right)^T.$$

Fact 3.

$$\frac{d\, \mathrm{tr}\left(X^{-1}A\right)}{dX} = -\left(X^{-1}AX^{-1}\right)^T.$$

Fact 4. *For square matrices A and B,*

$$\frac{d\,\mathrm{tr}\left(AX^T\right)}{dX} = A.$$

Fact 5. *For square matrices A and B,*

$$\frac{d\,\mathrm{tr}\left(AXBX^T\right)}{dX} = A^T X B^T + AXB.$$

APPENDIX B
DETERMINING EM PARAMETERS

We now derive the EM-parameter equations given in (16)–(20). In the maximization step of the EM algorithm, we solve for parameters that maximize the expected joint likelihood:

$$\hat{\theta}^{(n+1)} = \underset{\theta}{\mathrm{argmax}}\, \mathrm{E}\left[p_{\mathbf{x},\mathbf{y}}(\mathbf{x}, y; \theta) \mid \mathbf{y} = y;\, \hat{\theta}^{(n)}\right], \tag{21}$$

where $\hat{\theta}^{(n)}$ is the guess of the true parameter set on the nth iteration. It is common to consider the log-likelihood to change the products in the joint likelihood to summations; the maximizing parameters are the same for either optimization. The below derivations reference proofs in [6, Chap. 13] and [15].

A. Simplification of log-likelihood

For a collaborative Kalman filter, the log-likelihood simplifies to

$$\log L = \log p(x, y; \theta) = \underbrace{\sum_{i=1}^{N} \log p(x_{i,0})}_{L_1} + \underbrace{\sum_{i=1}^{N}\sum_{t=1}^{T} \log p(x_{i,t}|x_{i,t-1})}_{L_2} + \underbrace{\sum_{i=1}^{N}\sum_{t=1}^{T} \log p(y_{i,t}|x_{i,t})}_{L_3}, \tag{22}$$

with

$$\begin{aligned} p(x_{i,0}) &\sim \mathcal{N}(x_{i,0};\, \mu_i, \Sigma_i), \\ p(x_{i,t}|x_{i,t-1}) &\sim \mathcal{N}(x_{i,t};\, A_{i,t}\, x_{i,t-1}, Q_i), \\ p(y_{i,t}|x_{i,t}) &\sim \mathcal{N}(y_{i,t};\, H_{i,t}\, x_{i,t}, R_i). \end{aligned} \tag{23}$$

Using Fact 1, the first term L_1 becomes

$$\mathrm{E}[L_1] = -\frac{NK}{2}\log(2\pi) - \frac{1}{2}\sum_{i=1}^{N}\log|\Sigma_i| - \frac{1}{2}\sum_{i=1}^{N}\mathrm{tr}\left(\Sigma_i^{-1}\,\mathrm{E}[(\mathbf{x}_{i,0}-\mu_i)(\mathbf{x}_{i,0}-\mu_i)^T]\right). \tag{24}$$

We then use the identity $\mathbf{x}_{i,0} - \mu_i = (\mathbf{x}_{i,0} - \hat{\mathbf{x}}_{i,0|T}) + (\hat{\mathbf{x}}_{i,0|T} - \mu_i)$ and note that estimation error and innovation of a Kalman filter are uncorrelated to rewrite the expectation of L_1 to be

$$\mathrm{E}[L_1] = c_1 - \frac{1}{2}\sum_{i=1}^{N}\log|\Sigma_i| - \frac{1}{2}\sum_{i=1}^{N}\mathrm{tr}\left(\Sigma_i^{-1}\left(P_{i,0|T} + (\hat{\mathbf{x}}_{i,0|T}-\mu_i)(\hat{\mathbf{x}}_{i,0|T}-\mu_i)^T\right)\right). \tag{25}$$

We repeat a similar procedure for L_2, this time using the identity

$$\begin{aligned} \mathbf{x}_{i,t} - A_{i,t}\,\mathbf{x}_{i,t-1} &= \mathbf{x}_{i,t} - A_{i,t}\,\mathbf{x}_{i,t-1} + \hat{\mathbf{x}}_{i,t|T} - \hat{\mathbf{x}}_{i,t|T} + A_{i,t}\,\hat{\mathbf{x}}_{i,t-1|T} - A_{i,t}\,\hat{\mathbf{x}}_{i,t-1|T} \\ &= \left(\hat{\mathbf{x}}_{i,t|T} - A_{i,t}\hat{\mathbf{x}}_{i,t-1|T}\right) - \left(\hat{\mathbf{x}}_{i,t|T} - \mathbf{x}_{i,t}\right) + A_{i,t}\left(\hat{\mathbf{x}}_{i,t-1|T} - \mathbf{x}_{i,t-1}\right). \end{aligned} \tag{26}$$

We then rewrite the expectation of L_2 as

$$\begin{aligned} \mathrm{E}[L_2] &= -\frac{NTK}{2}\log(2\pi) - \frac{T}{2}\sum_{i=1}^{n}\log|Q_i| - \frac{1}{2}\sum_{i=1}^{N}\sum_{t=1}^{T}\mathrm{tr}\left(Q_i^{-1}\,\mathrm{E}[(\mathbf{x}_{i,t}-A_{i,t})(\mathbf{x}_{i,t}-A_{i,t})^T]\right) \\ &= c_2 - \sum_{i=1}^{n}\frac{T}{2}\log|Q_i| - \frac{1}{2}\sum_{i=1}^{N}\sum_{t=1}^{T}\mathrm{tr}\left(Q_i^{-1}\,\mathrm{E}\left[\left\{\left(\hat{\mathbf{x}}_{i,t|T} - A_{i,t}\hat{\mathbf{x}}_{i,t-1|T}\right) - \left(\hat{\mathbf{x}}_{i,t|T} - \mathbf{x}_{i,t}\right) + A_{i,t}\left(\hat{\mathbf{x}}_{i,t-1|T} - \mathbf{x}_{i,t-1}\right)\right\}\right.\right. \\ &\qquad \left.\left. \times \left\{\left(\hat{\mathbf{x}}_{i,t|T} - A_{i,t}\hat{\mathbf{x}}_{i,t-1|T}\right) - \left(\hat{\mathbf{x}}_{i,t|T} - \mathbf{x}_{i,t}\right) + A_{i,t}\left(\hat{\mathbf{x}}_{i,t-1|T} - \mathbf{x}_{i,t-1}\right)\right\}^T\right]\right). \end{aligned} \tag{27}$$

Expanding everything and again noting that the Kalman estimation error and innovation are uncorrelated, (27) simplifies to

$$\mathrm{E}[L_2] = c_2 - \frac{T}{2}\sum_{i=1}^{n}\log|Q_i| - \frac{1}{2}\sum_{i=1}^{N}\sum_{t=1}^{T}\mathrm{tr}\left(Q_i^{-1}\left\{\left(P_{i,t|T} + \hat{\mathbf{x}}_{i,t|T}\,\hat{\mathbf{x}}_{i,t|T}^T\right) - 2\left(P_{i,t,t-1|T} + \hat{\mathbf{x}}_{i,t|T}\,\hat{\mathbf{x}}_{i,t-1|T}^T\right)A_{i,t}^T + A_{i,t}\left(P_{i,t-1|T} + \hat{\mathbf{x}}_{i,t-1|T}\,\hat{\mathbf{x}}_{i,t-1|T}^T\right)A_{i,t}^T\right\}\right). \tag{28}$$

A similar derivation is employed for L_3 utilizing

$$y_{i,t} - H_{i,t}\,\mathbf{x}_{i,t} = (y_{i,t} - H_{i,t}\,\hat{\mathbf{x}}_{i,t|T}) + H_{i,t}(\hat{\mathbf{x}}_{i,t|T} - \mathbf{x}_{i,t}). \tag{29}$$

Some care is needed in writing out R_i in L_3 since $y_{i,t}$ can be of different lengths depending on the observation matrix O and hence only a subset of the noise covariance matrix is required at each time step. To circumvent this issue, we define a fill function that expands the observation vector back to $\mathbb{R}^M$.

Currently, the formulation is extremely general and parameters may change with users and in time. We can maximize with respect to the log-likelihood but the resulting estimation would be poor and does not exploit the possible similarities between a population of users. To fully realize the benefits of CKF, we make simplifying assumptions that $\mu_i = 0$, $\Sigma_i = \Sigma$, $A_i = A$, $Q_i = Q$, and $R_i = \sigma_R^2 I_M$. We now move summations into the trace operator and the log-likelihood simplifies to

$$\mathrm{E}[L_1] = c_1 - \frac{N}{2}\log|\Sigma| - \frac{1}{2}\mathrm{tr}\left(\Sigma^{-1}\Gamma_1\right), \tag{30}$$

$$\mathrm{E}[L_2] = c_2 - \frac{NT}{2}\log|Q| - \frac{1}{2}\mathrm{tr}(Q^{-1}\Gamma_2), \tag{31}$$

$$\mathrm{E}[L_3] = c_3 - \frac{|\mathcal{O}|}{2}\log\sigma_R^2 - \frac{1}{2\sigma_R^2}\mathrm{tr}(\Gamma_3), \tag{32}$$

where

$$\Gamma_1 = \sum_{i=1}^{N}\left(P_{i,0|T} + \hat{\mathbf{x}}_{i,0|T}\,\hat{\mathbf{x}}_{i,0|T}^T\right), \tag{33}$$

$$\Gamma_2 = \sum_{i=1}^{N}\sum_{t=1}^{T}\left\{\left(P_{i,t|T} + \hat{\mathbf{x}}_{i,t|T}\,\hat{\mathbf{x}}_{i,t|T}^T\right) - 2\left(P_{i,t,t-1|T} + \hat{\mathbf{x}}_{i,t|T}\,\hat{\mathbf{x}}_{i,t-1|T}^T\right)A_t^T + A_t\left(P_{i,t-1|T} + \hat{\mathbf{x}}_{i,t-1|T}\,\hat{\mathbf{x}}_{i,t-1|T}^T\right)A_t^T\right\}, \tag{34}$$

$$\Gamma_3 = \sum_{i=1}^{N}\sum_{t=1}^{T}\left\{\mathrm{fill}(y_{i,t})\;\mathrm{fill}(y_{i,t})^T - 2\,\mathrm{fill}(y_{i,t})\,\hat{\mathbf{x}}_{i,t|T}^T\,V^T + V\left(P_{i,t|T} + \hat{\mathbf{x}}_{i,t|T}\,\hat{\mathbf{x}}_{i,t|T}^T\right)V^T\right\}. \tag{35}$$

B. Determining Σ, Q and R

To maximize with respect to Σ, we can differentiate (30), set to zero, and solve. Using Facts 2 and 3,

$$\frac{\partial\,\mathrm{E}[L_1]}{\partial\Sigma} = -\frac{N}{2}\left(\Sigma^{-1}\right)^T + \frac{1}{2}\left(\Sigma^{-1}\Gamma_1\Sigma^{-1}\right)^T, \tag{36}$$

and maximization gives

$$\widehat{\Sigma} = \frac{1}{N}\Gamma_1. \tag{37}$$

If we had further assumed that $\Sigma = \sigma_U^2 I_K$, then (30) would simplify to

$$\mathrm{E}[L_1] = c_1 - \frac{NK}{2}\log\sigma_U^2 - \frac{1}{2\sigma_U^2}\mathrm{tr}\left(\Gamma_1\right),$$

and maximization yields (16).

The derivations for Q and R follow similarly and lead to (17) and (18) respectively.

C. Determining A and V

Rewriting (31) as

$$\mathrm{E}[L_2] = c_A + \mathrm{tr}(Q^{-1}A_2A^T) - \frac{1}{2}\mathrm{tr}(Q^{-1}AA_1A^T), \tag{38}$$

where c_A is the collection of terms that do not depend on A, we maximize using the same procedure as for Σ. We utilize Facts 4 and 5 while noting that Q and $\Gamma_{2,i}$ are symmetric and invertible, and the maximization yields (19).

Following a similar procedure for optimization of V, we express (32) as

$$\mathrm{E}[L_3] = c_V + \mathrm{tr}\left(V_2V^T\right) - \frac{1}{2}\mathrm{tr}\left(R^{-1}VV_1V^T\right). \tag{39}$$

Maximization then gives (20).

REFERENCES

[1] N. Srebro, "Learning with matrix factorizations," Ph.D. Thesis, Mass. Inst. Technol., Cambridge, MA, 2004.

[2] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *IEEE Computer*, vol. 42, no. 8, pp. 30–37, Aug. 2009.

[3] R. Salakhutdinov and A. Mnih, "Probabilistic matrix factorization," in *Adv. Neural Inf. Process. Syst. 20*. Cambridge, MA: MIT Press, 2008, pp. 1257–1264.

[4] A. E. Bryson, Jr. and Y.-C. Ho, *Applied Optimal Control*. Waltham, MA: Ginn and Company, 1969.

[5] R. E. Kalman, "A new approach to linear filtering and prediction problems," *J. Basic Eng.–T. ASME*, vol. 82, no. 1, pp. 35–45, Mar. 1960.

[6] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York, NY: Springer, 2006.

[7] H. E. Rauch, F. Tung, and C. T. Striebel, "Maximum likelihood estimates of linear dynamic systems," *AIAA J.*, vol. 3, no. 8, pp. 1445–1450, Aug. 1965.

[8] L. Xiong, X. Chen, T.-K. Huang, J. Schneider, and J. G. Carbonell, "Temporal collaborative filtering with Bayesian probabilistic tensor factorization," in *Proc. SIAM Int. Conf. Data Mining*, Columbus, OH, Apr.–May 2010, pp. 211–222.

[9] N. Lathia, S. Hailes, and L. Capra, "Temporal collaborative filtering with adaptive neighbourhoods," in *Proc. ACM SIGIR Conf. Res. Dev. Inf. Retrieval*, Boston, MA, Jul. 2009, pp. 796–797.

[10] Z. Lu, D. Agarwal, and I. S. Dhillon, "A spatio-temporal approach to collaborative filtering," in *Proc. ACM Conf. Recommender Syst.*, New York, NY, Oct. 2009, pp. 13–20.

[11] S. Nowakowski, C. Bernier, and A. Boyer, "Target tracking in the recommender space: Toward a new recommender system based on Kalman filtering," http://arxiv.org/pdf/1011.2304, Nov. 2010.

[12] N. Sahoo, P. V. Singh, and T. Mukhopadhyay, "A hidden Markov model for collaborative filtering," in *Proc. Winter Conf. Business Intell.*, Salt Lake City, UT, Mar. 2011.

[13] R. Salakhutdinov and A. Mnih, "Bayesian probabilistic matrix factorization using Markov chain Monte Carlo," in *Proc. Int. Conf. Mach. Learn.*, Helsinki, Finland, Jul. 2008, pp. 880–887.

[14] K. B. Petersen and M. S. Pedersen, "The matrix cookbook," 2008.

[15] T. Rosenbaum and A. Zetlin-Jones, "The Kalman filter and the EM algorithm," 2006.