Probabilistic Sequential Matrix Factorization

Ö. Deniz Akyildiz[1,⋆], Gerrit J.J. van den Burg[1,⋆]
Theodoros Damoulas[1,2], Mark F. J. Steel[2]

The Alan Turing Institute[1]
University of Warwick[2]

⋆Equal contribution.

Problem definition

Matrix factorization

We are interested in the problem factorizing a data matrix $Y \in \mathbb{R}^{m \times n}$ as

$$Y \approx CX$$

with $C \in \mathbb{R}^{m \times r}$, *the dictionary*, and $X \in \mathbb{R}^{r \times n}$ the *coefficients*.

Problem definition

Matrix factorization

We are interested in the problem factorizing a data matrix $Y \in \mathbb{R}^{m \times n}$ as

$$Y \approx CX$$

with $C \in \mathbb{R}^{m \times r}$, *the dictionary*, and $X \in \mathbb{R}^{r \times n}$ the *coefficients*.

- Probabilistic: We want to obtain approximate probability measures over C and X

Problem definition

Matrix factorization

We are interested in the problem factorizing a data matrix $Y \in \mathbb{R}^{m \times n}$ as

$$Y \approx CX$$

with $C \in \mathbb{R}^{m \times r}$, *the dictionary*, and $X \in \mathbb{R}^{r \times n}$ the *coefficients*.

- Probabilistic: We want to obtain approximate probability measures over C and X
- Dynamic: We are interested in the case where Y is a Markovian process (e.g. a time-series).

Problem definition

Matrix factorization

We are interested in the problem factorizing a data matrix $Y \in \mathbb{R}^{m \times n}$ as

$$Y \approx CX$$

with $C \in \mathbb{R}^{m \times r}$, *the dictionary*, and $X \in \mathbb{R}^{r \times n}$ the *coefficients*.

- Probabilistic: We want to obtain approximate probability measures over C and X
- Dynamic: We are interested in the case where Y is a Markovian process (e.g. a time-series).
- Sequential: We want to process the columns of Y sequentially in time in a scalable way.

The Probabilistic Model

A state-space formulation

We aim at solving the matrix factorization problem by solving the inference problem for the following probabilistic state-space model:

$$
\begin{aligned}
p(C) &= \mathcal{MN}(C; C_0, I_d, V_0) \\
p(x_0) &= \mathcal{N}(x_0; \mu_0, P_0) \\
p_\theta(x_t|x_{t-1}) &= \mathcal{N}(x_t; f_\theta(x_{t-1}), Q_t) \\
p(y_t|x_t, C) &= \mathcal{N}(y_t; Cx_t, R_t),
\end{aligned}
$$

where $\mathcal{MN}$ denotes the matrix-normal distribution.

The Probabilistic Model

A state-space formulation

We aim at solving the matrix factorization problem by solving the inference problem for the following probabilistic state-space model:

$$\begin{aligned} p(C) &= \mathcal{MN}(C; C_0, I_d, V_0) \\ p(x_0) &= \mathcal{N}(x_0; \mu_0, P_0) \\ p_\theta(x_t|x_{t-1}) &= \mathcal{N}(x_t; f_\theta(x_{t-1}), Q_t) \\ p(y_t|x_t, C) &= \mathcal{N}(y_t; Cx_t, R_t), \end{aligned}$$

where $\mathcal{MN}$ denotes the matrix-normal distribution.

The advantages

- Ensures $y_t \approx Cx_t$ (which implies $Y \approx CX$),

The Probabilistic Model

A state-space formulation

We aim at solving the matrix factorization problem by solving the inference problem for the following probabilistic state-space model:

$$
\begin{aligned}
p(C) &= \mathcal{MN}(C; C_0, I_d, V_0) \\
p(x_0) &= \mathcal{N}(x_0; \mu_0, P_0) \\
p_\theta(x_t | x_{t-1}) &= \mathcal{N}(x_t; f_\theta(x_{t-1}), Q_t) \\
p(y_t | x_t, C) &= \mathcal{N}(y_t; C x_t, R_t),
\end{aligned}
$$

where $\mathcal{MN}$ denotes the matrix-normal distribution.

The advantages

- Ensures $y_t \approx C x_t$ (which implies $Y \approx CX$),
- Encodes f_θ: A flexible nonlinearity that can be customized,

The Probabilistic Model

A state-space formulation

We aim at solving the matrix factorization problem by solving the inference problem for the following probabilistic state-space model:

$$
\begin{aligned}
p(C) &= \mathcal{MN}(C; C_0, I_d, V_0) \\
p(x_0) &= \mathcal{N}(x_0; \mu_0, P_0) \\
p_\theta(x_t | x_{t-1}) &= \mathcal{N}(x_t; f_\theta(x_{t-1}), Q_t) \\
p(y_t | x_t, C) &= \mathcal{N}(y_t; C x_t, R_t),
\end{aligned}
$$

where $\mathcal{MN}$ denotes the matrix-normal distribution.

The advantages

- Ensures $y_t \approx C x_t$ (which implies $Y \approx CX$),
- Encodes f_θ: A flexible nonlinearity that can be customized,
- Returns probability measures over C and X (i.e. $(x_t)_{t \geq 1}$).

Inference

Scalable and efficient inference with matrix updates

The special structure of our prior on C and the dynamic model enables us to obtain an efficient algorithm that

Inference

Scalable and efficient inference with matrix updates

The special structure of our prior on C and the dynamic model enables us to obtain an efficient algorithm that

- approximates $p(C|y_{1:t})$ and $p(x_t|y_{1:t})$ recursively,

Inference

Scalable and efficient inference with matrix updates

The special structure of our prior on C and the dynamic model enables us to obtain an efficient algorithm that

- approximates $p(C|y_{1:t})$ and $p(x_t|y_{1:t})$ recursively,
- avoids costly sampling schemes (e.g. Gibbs sampling),

Inference

Scalable and efficient inference with matrix updates

The special structure of our prior on C and the dynamic model enables us to obtain an efficient algorithm that

- approximates $p(C|y_{1:t})$ and $p(x_t|y_{1:t})$ recursively,
- avoids costly sampling schemes (e.g. Gibbs sampling),
- based on **cheap** and **stable** matrix-valued and vector updates,

Inference

Scalable and efficient inference with matrix updates

The special structure of our prior on C and the dynamic model enables us to obtain an efficient algorithm that

- approximates $p(C|y_{1:t})$ and $p(x_t|y_{1:t})$ recursively,
- avoids costly sampling schemes (e.g. Gibbs sampling),
- based on **cheap** and **stable** matrix-valued and vector updates,
- enables the encoding of an **interpretable structure** into the model using the nonlinearity f_θ,

Inference

Scalable and efficient inference with matrix updates

The special structure of our prior on C and the dynamic model enables us to obtain an efficient algorithm that

- approximates $p(C|y_{1:t})$ and $p(x_t|y_{1:t})$ recursively,
- avoids costly sampling schemes (e.g. Gibbs sampling),
- based on **cheap** and **stable** matrix-valued and vector updates,
- enables the encoding of an **interpretable structure** into the model using the nonlinearity f_θ,
- scales **linearly** with the number of observations.

Inference

Scalable and efficient inference with matrix updates

The special structure of our prior on C and the dynamic model enables us to obtain an efficient algorithm that

- approximates $p(C|y_{1:t})$ and $p(x_t|y_{1:t})$ recursively,
- avoids costly sampling schemes (e.g. Gibbs sampling),
- based on **cheap** and **stable** matrix-valued and vector updates,
- enables the encoding of an **interpretable structure** into the model using the nonlinearity f_θ,
- scales **linearly** with the number of observations.

We also provide a further **robustified** model (and an inference scheme) for datasets with heavy-tailed noise.

Inference

Scalable and efficient inference with matrix updates

We achieve these by using

Inference

Scalable and efficient inference with matrix updates

We achieve these by using

- the assumed Kronecker structure on the covariance of the prior on C which results in **tractable** matrix updates,

Inference

Scalable and efficient inference with matrix updates

We achieve these by using

- the assumed Kronecker structure on the covariance of the prior on C which results in **tractable** matrix updates,
- the extended Kalman updates and automatic differentiation to obtain the Jacobian of the coefficient dynamics f_θ

Inference

Scalable and efficient inference with matrix updates

We achieve these by using

- the assumed Kronecker structure on the covariance of the prior on C which results in **tractable** matrix updates,
- the extended Kalman updates and automatic differentiation to obtain the Jacobian of the coefficient dynamics f_θ
- gradient descent on the approximate (and tractable) marginal likelihood $\tilde{p}_\theta(y_t|y_{1:t-1})$ to optimise the parameters of f_θ

Inference

Scalable and efficient inference with matrix updates

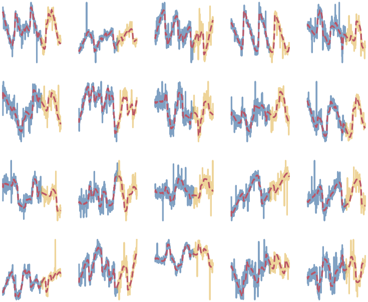
We achieve these by using

- the assumed Kronecker structure on the covariance of the prior on C which results in **tractable** matrix updates,
- the extended Kalman updates and automatic differentiation to obtain the Jacobian of the coefficient dynamics f_θ
- gradient descent on the approximate (and tractable) marginal likelihood $\tilde{p}_\theta(y_t|y_{1:t-1})$ to optimise the parameters of f_θ
  - leverage (again) automatic differentiation
  - take advantage of modern non-convex optimisers, such as Adam.

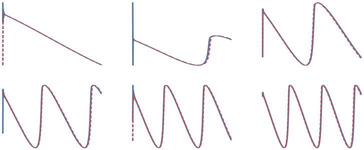
Experimental results

Synthetic dataset

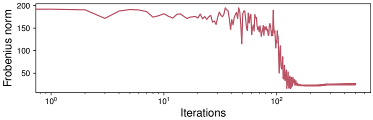
When the subspace model is well-calibrated, we can perform high-dimensional time-series prediction.



(a) Observed time series (blue) with unobserved future data (yellow) and the reconstruction (red).



(b) True (blue) and predicted (red) subspace.



(c) Reconstruction error

Experimental results

Missing data imputation

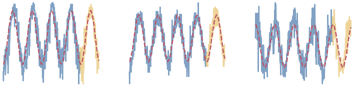
More applications in the paper:

Experimental results

Missing data imputation

More applications in the paper:

- Time-series forecasting (on air quality data),

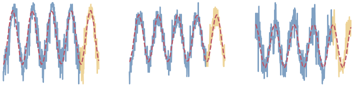


Experimental results

Missing data imputation

More applications in the paper:

- Time-series forecasting (on air quality data),

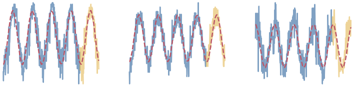


- Missing data imputation,

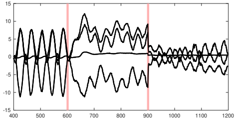
Experimental results
Missing data imputation

More applications in the paper:

- Time-series forecasting (on air quality data),



- Missing data imputation,
- Changepoint detection.



Thanks! See you at the conference!