Supplement to: Bayesian Latent Factor Model for Higher-order Data

Zerui Tao ZERUI.TAO@RIKEN.JP
Xuyang Zhao ZHAO@SIP.TUAT.AC.JP
Toshihisa Tanaka TANAKAT@CC.TUAT.AC.JP
Department of Electrical and Electronic Engineering
Tokyo University of Agriculture and Technology, Tokyo, Japan
RIKEN Center for Advanced Intelligence Project (AIP), Tokyo, Japan

Qibin Zhao* QIBIN.ZHAO@RIKEN.JP
RIKEN Center for Advanced Intelligence Project (AIP), Tokyo, Japan

Editors: Vineeth N Balasubramanian and Ivor Tsang

Appendix A. Gibbs Sampler

In this section, we firstly derive the Gibbs sampler of our model. Since the distributions are all conditionally conjugate, the derivation is straightforward. For the parameters, we denote $\boldsymbol{\mathcal{Q}} = \{\boldsymbol{\mathcal{Q}}^{(d)}\}_{d=1}^{D}$, $\boldsymbol{\eta} = \{\boldsymbol{\eta}^{(n)}\}_{n=1}^{N}$, $\boldsymbol{\phi} = \{\boldsymbol{\phi}^{(d)}\}_{d=1}^{D}$, $\boldsymbol{\delta} = \{\boldsymbol{\delta}^{(d)}\}_{d=1}^{D+1}$ and $\boldsymbol{\Theta} = \{\boldsymbol{\mathcal{Q}}, \boldsymbol{\eta}, \boldsymbol{\phi}, \boldsymbol{\delta}, \tau\}$. The joint distribution of the TRLF model is

$$\log p(\boldsymbol{\mathcal{Y}}, \boldsymbol{\Theta}) = \log p(\boldsymbol{\mathcal{Y}} \mid \boldsymbol{\Theta}) + \log p(\boldsymbol{\Theta}), \tag{1}$$

where

$$\log p(\boldsymbol{\mathcal{Y}} \mid \boldsymbol{\Theta}) = \log p(\boldsymbol{\mathcal{Y}} \mid \boldsymbol{\mathcal{Q}}, \boldsymbol{\eta}, \boldsymbol{\tau}) = \sum_{n=1}^{N} \log p(\boldsymbol{\mathcal{Y}}^{(n)} \mid \boldsymbol{\mathcal{Q}}, \boldsymbol{\eta}^{(n)}, \tau).$$

The prior distribution is

$$\log p(\boldsymbol{\Theta}) = \log p(\boldsymbol{\mathcal{Q}} \mid \boldsymbol{\phi}, \boldsymbol{u}) + \log p(\boldsymbol{\phi}) + \log p(\boldsymbol{\delta}) + \log p(\tau) + \log p(\boldsymbol{\eta}),$$

where

$$\log p(\boldsymbol{\mathcal{Q}} \mid \boldsymbol{\phi}, \boldsymbol{u}) = \sum_{d=1}^{D} \log p(\boldsymbol{\mathcal{Q}}^{(d)} \mid \boldsymbol{\phi}^{(d)}, \boldsymbol{u}^{(d)}, \boldsymbol{u}^{(d+1)}), \quad u_l^{(d)} = \prod_{h=1}^{l} \delta_h^{(d)},$$

$$\log p(\boldsymbol{\delta}) = \sum_{d=1}^{D+1} \log p(\boldsymbol{\delta}^{(d)}), \quad \log p(\boldsymbol{\phi}) = \sum_{d=1}^{D} \log p(\boldsymbol{\phi}^{(d)}), \quad \log p(\boldsymbol{\eta}) = \sum_{n=1}^{N} \log p(\boldsymbol{\eta}^{(n)}).$$

Given the joint distribution Eq. (1), we can establish the Gibbs sampler algorithm by sampling from the conditional distributions sequentially.

* Corresponding author

Sample $\mathcal{Q}$ The conditional distributions of the core tensors $\mathcal{Q}$ involve subchains of $\mathcal{Q}$ and $\boldsymbol{\eta}$. Hence, to make the notations consistent, we denote $\mathcal{Q}^{(D+1)} = \boldsymbol{\eta}$. For $d = 1, \ldots, D$ and $i = 1, \ldots, P_d$, the conditional distribution is

$$p(\boldsymbol{Q}^{(d)}[i] \mid -) \sim \mathcal{N}(\mathrm{vec}(\boldsymbol{Q}^{(d)}[i]) \mid \boldsymbol{\mu}^{(i)}_{\boldsymbol{Q}^{(d)}}, \boldsymbol{\Lambda}^{(i)}_{\boldsymbol{Q}^{(d)}}), \tag{2}$$

where

$$\boldsymbol{\mu}^{(i)}_{\boldsymbol{Q}^{(d)}} = \boldsymbol{\Lambda}^{(i)}_{\boldsymbol{Q}^{(d)}} \left(\tau \sum_{n=1}^{N} \sum_{\mathbf{p}_{-d}} \boldsymbol{\mathcal{Y}}^{(n)}_{\mathbf{p}} \mathrm{vec}(\boldsymbol{Q}^{\neq d, \intercal}[\overline{\mathbf{p}_{-d}}]) \right),$$

$$\boldsymbol{\Lambda}^{(i)}_{\boldsymbol{Q}^{(d)}} = \left(\tau \sum_{n=1}^{N} \sum_{\mathbf{p}_{-d}} \mathrm{vec}(\boldsymbol{Q}^{\neq d, \intercal}[\overline{\mathbf{p}_{-d}}]) \mathrm{vec}(\boldsymbol{Q}^{\neq d, \intercal}[\overline{\mathbf{p}_{-d}}])^{\intercal} + \mathrm{diag}(\mathrm{vec}(\boldsymbol{\phi}^{(d)}_{:i:})) * (\boldsymbol{U}^{(d+1)} \otimes \boldsymbol{U}^{(d)}) \right)^{-1}.$$

where $\boldsymbol{U}^{(d)} = \mathrm{diag}(\boldsymbol{u}^{(d)})$.

Sample $\boldsymbol{\phi}$ For $d = 1, \ldots, D$, $j = 1, \ldots, R_d$ and $h = 1, \ldots, R_{d+1}$, the conditional distribution of $\boldsymbol{\phi}$ is

$$p(\phi^{(d)}_{jih} \mid -) \sim Ga(a_\phi, b_\phi),$$

where

$$a_\phi = \frac{1}{2} + \nu, \quad b_\phi = \nu + (\boldsymbol{Q}^{(d)}_{jh}[i])^2 u^{(d)}_j u^{(d+1)}_h.$$

Sample $\boldsymbol{\delta}$ We denote $u^{(d)}_{l,-h} = \prod_{j=1, j\neq h}^{l} \delta^{(d)}_j$ and $\boldsymbol{U}^{(d)} = \mathrm{diag}(\boldsymbol{u}^{(d)})$. Hence we have $u^{(d)}_l = \prod_{j=1}^{l} \delta^{(d)}_j = u^{(d)}_{l,-h} \delta^{(d)}_h$. For $\boldsymbol{\delta}^{(1)}$, the conditional distribution is

$$\log p(\delta^{(1)}_h \mid -) \propto \log p(\mathcal{Q}^{(1)} \mid \boldsymbol{\phi}^{(1)}, \boldsymbol{u}^{(1)}, \boldsymbol{u}^{(2)}) + \log p(\boldsymbol{\delta}^{(1)}), \quad \forall h = 1, \ldots, R_1.$$

Hence, we have

$$p(\delta^{(1)}_h \mid -) \sim Ga(a^h_\delta, b^h_\delta),$$

where

$$a^h_\delta = \alpha_\delta + \frac{P_1 R_2 (R_1 - h + 1)}{2}, \quad b^h_\delta = 1 + \frac{1}{2} \sum_{l=h}^{R_1} \left[\sum_{i=1}^{P_1} \boldsymbol{u}^{(2), \intercal} (\boldsymbol{Q}^{(1)}_{l\cdot}[i] * \boldsymbol{Q}^{(1)}_{l\cdot}[i]) \right] u^{(1)}_{l,-h}.$$

For $d = 2, \ldots, D$, we have

$$p(\delta^{(d)}_h \mid -) \propto \log p(\mathcal{Q}^{(d-1)} \mid \boldsymbol{\phi}^{(d-1)}, \boldsymbol{u}^{(d-1)}, \boldsymbol{u}^{(d)}) + \log p(\mathcal{Q}^{(d)} \mid \boldsymbol{\phi}^{(d)}, \boldsymbol{u}^{(d)}, \boldsymbol{u}^{(d+1)}) + \log p(\boldsymbol{\delta}^{(d)})$$

Then the conditional distribution is also a Gamma,

$$p(\delta^{(d)}_h \mid -) \sim Ga(a^\delta_h, b^\delta_h), \quad \forall h = 1, \ldots, R_d,$$

where

$$a_\delta^h = \alpha_\delta + \frac{(P_d R_{d+1} + P_{d-1} R_d)(R_d - h + 1)}{2},$$
$$b_\delta^h = 1 + \frac{1}{2} \sum_{l=h}^{R_d} \left[\sum_{i=1}^{P_d} \boldsymbol{u}^{(d+1),\intercal} (\boldsymbol{Q}_{l\cdot}^{(d)}[i] * \boldsymbol{Q}_{l\cdot}^{(d)}[i]) + \sum_{i=1}^{P_{d-1}} (\boldsymbol{Q}_{\cdot l}^{(d-1)}[i] * \boldsymbol{Q}_{\cdot l}^{(d-1)}[i]) \boldsymbol{u}^{(d-1)} \right] u_{l,-h}^{(d)}.$$

Similarly, for $\boldsymbol{\delta}^{(D+1)}$, we have

$$p(\delta_h^{(D+1)} \mid -) \sim Ga(a_\delta^h, b_\delta^h), \quad \forall h = 1, \ldots, R_{D+1},$$

where

$$a_\delta^h = \alpha_\delta + \frac{P_D R_{D+1}(R_{D+1} - h + 1)}{2}, \quad b_\delta^h = 1 + \frac{1}{2} \sum_{l=h}^{R_{D+1}} \left[\sum_{i=1}^{P_D} (\boldsymbol{Q}_{\cdot l}^{(D)}[i] * \boldsymbol{Q}_{\cdot l}^{(D)}[i]) \boldsymbol{u}^{(d+1)} \right] u_{l,-h}^{(D+1)}.$$

Sample $\boldsymbol{\eta}$ The conditional distribution is,

$$p(\boldsymbol{\eta}^{(n)} \mid -) \sim \mathcal{N}(\mathrm{vec}(\boldsymbol{\eta}^{(n)}) \mid \boldsymbol{\mu}_{\boldsymbol{\eta}}^{(n)}, \boldsymbol{\Lambda}_{\boldsymbol{\eta}}^{(n)}), \tag{3}$$

where

$$\boldsymbol{\mu}_{\boldsymbol{\eta}}^{(n)} = \boldsymbol{\Lambda}_{\boldsymbol{\eta}}^{(n)} \left(\tau \sum_{\mathbf{p}} \boldsymbol{\mathcal{Y}}_{\mathbf{p}}^{(n)} \mathrm{vec}(\boldsymbol{Q}^{\leq D,\intercal}[\overline{\mathbf{p}}]) \right),$$
$$\boldsymbol{\Lambda}_{\boldsymbol{\eta}}^{(n)} = \left(\tau \sum_{\mathbf{p}} \mathrm{vec}(\boldsymbol{Q}^{\leq D,\intercal}[\overline{\mathbf{p}}]) \mathrm{vec}(\boldsymbol{Q}^{\leq D,\intercal}[\overline{\mathbf{p}}])^\intercal + \boldsymbol{I} \right)^{-1}.$$

Sample τ The conditional distribution is $p(\tau \mid -) \propto Ga(a_\tau, b_\tau)$, where

$$a_\tau = \frac{N \prod_{d=1}^D P_d}{2} + \alpha_\tau, \quad b_\tau = \beta_\tau + \frac{1}{2} \sum_{n=1}^N \left\| \boldsymbol{\mathcal{Y}}^{(n)} - \hat{\boldsymbol{\mathcal{Y}}}^{(n)} \right\|_F^2.$$

Appendix B. PX-EM Algorithm

In this section, we firstly derive the original EM algorithm. Then we describe the PX-EM algorithm by adding an additional double-rotation step. In the EM algorithm, we treat $\boldsymbol{\eta}$ as latent variables and update them in the expectation step. Other parameters are updated in the maximization step. In this section, we denote the expectation w.r.t. $\boldsymbol{\eta}$ as $\langle \cdot \rangle$.

B.1. Expectation step

Based on Eq. (3), we can compute the expectation of the latent variables $\boldsymbol{\eta}$, as

$$\langle \tilde{\boldsymbol{\eta}}^{(n)} \rangle = \boldsymbol{\Lambda}_{\boldsymbol{\eta}}^{(n)} \left(\tau \sum_{\mathbf{p}} \boldsymbol{\mathcal{Y}}_{\mathbf{p}}^{(n)} \mathrm{vec}(\boldsymbol{Q}^{\leq D,\intercal}[\overline{\mathbf{p}}]) \right), \tag{4}$$
$$\langle \tilde{\boldsymbol{\eta}}^{(n)} \cdot (\tilde{\boldsymbol{\eta}}^{(n)})^\intercal \rangle = \langle \tilde{\boldsymbol{\eta}}^{(n)} \rangle \cdot \langle \tilde{\boldsymbol{\eta}}^{(n)} \rangle^\intercal + \boldsymbol{\Lambda}_{\boldsymbol{\eta}}^{(n)}, \tag{5}$$

for $n = 1, \ldots, N$, where $\tilde{\boldsymbol{\eta}}^{(n)} = \mathrm{vec}(\boldsymbol{\eta}^{(n)})$ and

$$\boldsymbol{\Lambda}_{\boldsymbol{\eta}}^{(n)} = \left(\tau \sum_{\mathbf{p}} \mathrm{vec}(\boldsymbol{Q}^{\leq D,\intercal}[\overline{\mathbf{p}}]) \mathrm{vec}(\boldsymbol{Q}^{\leq D,\intercal}[\overline{\mathbf{p}}])^{\intercal} + \boldsymbol{I} \right)^{-1}.$$

The computational complexity is $\mathcal{O}(K^3 + \sum_{d=1}^{D} P_d R^4 + NK^2 + NPDR^3)$, where K is the factor number, D is the data order, N is the sample size, $P = \prod_{d=1}^{D} P_d$ is the feature length and $R_1 = \cdots = R_D = R$ is the TR ranks. However, if $\boldsymbol{\mathcal{Y}}$ admits a TR format of rank r, the inner product in Eq. (4) can be computed polynomially with D and the computational complexity reduces to $\mathcal{O}(K^3 + \sum_{d=1}^{D} P_d R^4 + NK^2 + N \sum_{d=1}^{D} P_d R^2 r^2)$.

B.2. Maximization step

In the maximization step we give MAP estimate of the rest paramters. These estimates can be derived from the conditional distributions by replacing $\boldsymbol{\eta}$ by its expectations in Eq. (4) and (5).

Update $\mathcal{Q}$ For the core tensors $\boldsymbol{\mathcal{Q}}^{(d)}, \forall d = 1, \ldots, D$, according the conditional posterior Eq. (2), we have the update rule

$$\mathrm{vec}(\boldsymbol{Q}^{(d)}[i]) = \langle \boldsymbol{\Lambda}_Q^{-1} \rangle \left(\tau \sum_{n=1}^{N} \sum_{\mathbf{p}_{-d}} \boldsymbol{\mathcal{Y}}_{\mathbf{p}_d}^{(n)} \langle \mathrm{vec}(\boldsymbol{Q}^{\neq d,\intercal}[\overline{\mathbf{p}_{-d}}]) \rangle \right) \tilde{\boldsymbol{q}}^{(d)}[i], \quad \forall i = 1, \ldots, P_d, \tag{6}$$

where

$$\boldsymbol{\Lambda}_Q = \left(\tau \sum_{n=1}^{N} \sum_{\mathbf{p}_{-d}} \mathrm{vec}(\boldsymbol{Q}^{\neq d,\intercal}[\overline{\mathbf{p}_{-d}}]) \mathrm{vec}(\boldsymbol{Q}^{\neq d,\intercal}[\overline{\mathbf{p}_{-d}}])^{\intercal} + \mathrm{diag}(\mathrm{vec}(\boldsymbol{\phi}_{:i:}^{(d)})) * (\mathrm{diag}(\boldsymbol{u}^{(d+1)}) \otimes \mathrm{diag}(\boldsymbol{u}^{(d)})) \right)^{-1}.$$

The computation of $\langle \mathrm{vec}(\boldsymbol{Q}^{\neq d,\intercal}[\overline{\mathbf{p}_{-d}}]) \rangle$ is straightforward. Then, we have

$$\begin{aligned} &\langle \sum_{\mathbf{p}} \mathrm{vec}(\boldsymbol{Q}^{\neq d,\intercal}[\overline{\mathbf{p}}]) \mathrm{vec}(\boldsymbol{Q}^{\neq d,\intercal}[\overline{\mathbf{p}}])^{\intercal} \rangle \\ &= \sum_{p_{d-1}} \boldsymbol{Q}_{r_{d-2} m}^{(d_1)}[p_{d-1}] \boldsymbol{Q}_{r'_{d-2} n}^{(d-1)}[p_{d-1}] \cdots \sum_{n=1}^{N} \left\langle \mathrm{vec}(\boldsymbol{\eta}^{(n),\intercal}) \mathrm{vec}(\boldsymbol{\eta}^{(n),\intercal})^{\intercal} \right\rangle \cdots \sum_{p_1} \boldsymbol{Q}_{r_1 i}^{(1)}[p_1] \boldsymbol{Q}_{r'_1 j}^{(1)}[p_1], \end{aligned}$$

where the expectation term is given by Eq. (5).

The computational complexity for all $\{\boldsymbol{\mathcal{Q}}^{(d)}\}_{d=1}^{D}$ is $\mathcal{O}(\sum_{d=1}^{D} P_d R^6 + N \sum_{d=1}^{D} P_d R^4 + NPD^2 R^3)$. Similarly, if $\boldsymbol{\mathcal{Y}}$ admits TR format of rank r, the complexity cab be reduced to $\mathcal{O}(\sum_{d=1}^{D} P_d R^6 + N \sum_{d=1}^{D} P_d R^4 + ND \sum_{d=1}^{D} P_d R^2 r^2)$.

Update $\boldsymbol{\phi}$ The update rule for $\boldsymbol{\phi}$ is

$$\phi_{jih}^{(d)} = \frac{\frac{1}{2} + \nu}{\nu + (\boldsymbol{Q}_{jh}^{(d)}[i])^2 u_j^{(d)} u_h^{(d+1)}}. \tag{7}$$

Update $\boldsymbol{\delta}$ For $\boldsymbol{\delta}^{(1)}$, the update rule is

$$\delta_h^{(1)} = \frac{\alpha_\delta + P_1 R_2 (R_1 - h + 1)/2}{1 + \frac{1}{2}\sum_{l=h}^{R_1} \left[\sum_{i=1}^{P_1} \boldsymbol{u}^{(2),\intercal} (\boldsymbol{Q}_{l\cdot}^{(1)}[i] * \boldsymbol{Q}_{l\cdot}^{(1)}[i])\right] u_{l,-h}^{(1)}}. \tag{8}$$

For $d = 2, \ldots, D$, the update rule is

$$\delta_h^{(d)} = \frac{\alpha_\delta + (P_d R_{d+1} + P_{d-1} R_d)(R_d - h + 1)/2}{1 + \frac{1}{2}\sum_{l=h}^{R_d} \left[\sum_{i=1}^{P_d} \boldsymbol{u}^{(d+1),\intercal} (\boldsymbol{Q}_{l\cdot}^{(d)}[i] * \boldsymbol{Q}_{l\cdot}^{(d)}[i]) + \sum_{i=1}^{P_{d-1}} (\boldsymbol{Q}_{\cdot l}^{(d-1)}[i] * \boldsymbol{Q}_{\cdot l}^{(d-1)}[i]) \boldsymbol{u}^{(d-1)}\right] u_{l,-h}^{(d)}}. \tag{9}$$

For $\boldsymbol{\delta}^{(D+1)}$, we have

$$\delta_h^{(D+1)} = \frac{\alpha_\delta + P_D R_{D+1} (R_{D+1} - h + 1)/2}{1 + \frac{1}{2}\sum_{l=h}^{R_{D+1}} \left[\sum_{i=1}^{P_D} (\boldsymbol{Q}_{\cdot l}^{(D)}[i] * \boldsymbol{Q}_{\cdot l}^{(D)}[i]) \boldsymbol{u}^{(d+1)}\right] u_{l,-h}^{(D+1)}}. \tag{10}$$

Update τ For the noise precision, the update rule is

$$\tau_{\mathbf{p}} = \frac{NP_1 \cdots P_D/2 + \alpha_\tau}{\beta_\tau + \sum_{n=1}^{N} \langle \left\| \boldsymbol{\mathcal{Y}}_{\mathbf{p}}^{(n)} - \hat{\boldsymbol{\mathcal{Y}}}_{\mathbf{p}}^{(n)} \right\|_F^2 \rangle / 2}. \tag{11}$$

The expectation term can be computed using the similar strategy with updating $\boldsymbol{\mathcal{Q}}$.

B.3. Rotation step

For the parameter-expanded version of the TRLF model, we have

$$\begin{aligned} \boldsymbol{\mathcal{Y}} &= \ll \tilde{\boldsymbol{\mathcal{Q}}}^{(1)}, \boldsymbol{\mathcal{Q}}^{(2)}, \ldots, \boldsymbol{\mathcal{Q}}^{(D-1)}, \tilde{\boldsymbol{\mathcal{Q}}}^{(D)}, \boldsymbol{\eta} \gg + \boldsymbol{\mathcal{E}}, \\ \boldsymbol{\eta}^{(n)} &\sim \mathcal{MN}(\mathbf{0}, \boldsymbol{A}, \boldsymbol{B}), \end{aligned}$$

where

$$\begin{aligned} \tilde{\boldsymbol{Q}}^{(1)}[i] &= \boldsymbol{R}_1^{-1} \boldsymbol{Q}^{(1)}[i], \quad \forall i = 1, \ldots, P_1, \\ \tilde{\boldsymbol{Q}}^{(D)}[i] &= \boldsymbol{Q}^{(D)}[i] \boldsymbol{R}_2^{-1}, \quad \forall i = 1, \ldots, P_D. \end{aligned}$$

In the PX-EM algorithm, we add MGP priors on the loading core tensors $\tilde{\boldsymbol{\mathcal{Q}}}^{(1)}$ and $\tilde{\boldsymbol{\mathcal{Q}}}^{(D)}$, instead of $\boldsymbol{\mathcal{Q}}^{(1)}$ and $\boldsymbol{\mathcal{Q}}^{(D)}$. Moreover, by setting $\boldsymbol{R}_1$ as the upper triangular part of the Cholesky decomposition of $\boldsymbol{B}$ and $\boldsymbol{R}_2$ as the lower triangular part of the Cholesky decomposition of $\boldsymbol{A}$, it is equivalent to the original TRLF model. The key part is to find the optimal rotation matrix $\boldsymbol{R}_1$ and $\boldsymbol{R}_2$, based on $\boldsymbol{A}$ and $\boldsymbol{B}$. Like other parameters, the optimal rotation matrices are also obtained by maximizing expectation of the log-likelihood function. In specific, we have

$$\boldsymbol{A}_*, \boldsymbol{B}_* = \arg\max_{\boldsymbol{A},\boldsymbol{B}} \sum_{n=1}^{N} -\frac{R_1}{2} \log|\boldsymbol{A}| - \frac{R_D}{2} \log|\boldsymbol{B}| - \frac{1}{2} \mathrm{tr}(\boldsymbol{A}^{-1} \boldsymbol{\eta}^{(n)} \boldsymbol{B}^{-1} \boldsymbol{\eta}^{(n),\intercal}). \tag{12}$$

Though optimization problem Eq. (12) is non-convex for $\boldsymbol{A}$ and $\boldsymbol{B}$, we can update them in an alternating manner. Specifically, we have

$$\begin{aligned}
\boldsymbol{A}_* &= \arg\max_{\boldsymbol{A}} \sum_{n=1}^{N} -\frac{R_1}{2}\log|\boldsymbol{A}| - \frac{1}{2}\langle \mathrm{tr}(\boldsymbol{A}^{-1}\boldsymbol{\eta}^{(n)}\boldsymbol{B}^{-1}\boldsymbol{\eta}^{(n),\intercal})\rangle \\
&= \langle \sum_{n=1}^{N} \boldsymbol{\eta}^{(n),\intercal}\boldsymbol{B}^{-1}\boldsymbol{\eta}^{(n)}\rangle/(NR_1), \\
\boldsymbol{B}_* &= \arg\max_{\boldsymbol{B}} \sum_{n=1}^{N} -\frac{R_D}{2}\log|\boldsymbol{B}| - \frac{1}{2}\langle \mathrm{tr}(\boldsymbol{A}^{-1}\boldsymbol{\eta}^{(n)}\boldsymbol{B}^{-1}\boldsymbol{\eta}^{(n),\intercal})\rangle \\
&= \langle \sum_{n=1}^{N} \boldsymbol{\eta}^{(n)}\boldsymbol{A}^{-1}\boldsymbol{\eta}^{(n),\intercal}\rangle/(NR_D).
\end{aligned} \tag{13}$$

In each iteration of the PX-EM algorithm, we only update $\boldsymbol{A}$ and $\boldsymbol{B}$ once to get a local solution. It should be noted that if we use the TT format (i.e., TTLF), $\boldsymbol{A}$ or $\boldsymbol{B}$ is fixed as an identity matrix. Hence, the update rule for TTLF is optimal. The rotation step is used to get sparse latent factors and is optional. In practice, we can add this step for the first several iterations. The PX-EM algorithm is summarized in Algorithm 1.

Appendix C. Experiments

C.1. Synthetic Analysis

For the POET model[1], we use factor number 50. For the InfLF model[2], we initialize the factor number as 30 and adaptively tune the factor number in the Gibbs sampling process. For the HOLQ model[3], we use the default settings. For our model, we initialize with TT-rank 30 and prune small factors during training. The TTLF is a special case of TRLF in that the first rank is 1. In the simulation study, the TRLF has similar log-Euclidean distance (LED) with TTLF.

C.2. Real Data Covariance Estimation

For high dimensional data, if the covariance matrix has no special structure, it is hard to compute the log-likelihood function. Hence, we do not compare with the LW method in this experiment. Moreover, we find that the POET model is hard to fit these data. So we compare our model with InfLF and HOLQ in this subsection. For InfLF and our model, we compute the log-likelihood by using the Woodbury identity. For HOLQ, since it has Kronecker structure, it is also very efficient to compute the matrix determinant and matrix inverse. For all the data, we initialize the InfLF model with factor number 40 and our model with TT-rank 40.

1. https://cran.r-project.org/web/packages/POET/
2. https://cran.r-project.org/web/packages/infinitefactor/
3. https://cran.r-project.org/web/packages/tensr/index.html

Algorithm 1 PX-EM algorithm for TRLF.

Input: Data $\mathcal{Y}$, hyper-parameters $\alpha_\delta, \beta_\delta, \alpha_\tau, \beta_\tau, \nu$.
Initialize: Estimator $\hat{\mathcal{Y}} = \ll \mathcal{Q}^{(1)}, \cdots, \mathcal{Q}^{(D)}, \boldsymbol{\eta} \gg$.
Output: $\hat{\mathcal{Y}}$.
repeat
  // E-step
  for $n = 1 : N$ **do**
    Update $\boldsymbol{\eta}^{(n)}$ by Eq. (4) and (5).
  end for
  // M-step
  for $d = 1 : D$ **do**
    Update $\mathcal{Q}^{(d)}$ by Eq. (6).
  end for
  for $d = 1 : D$ **do**
    Update $\boldsymbol{\phi}^{(d)}$ by Eq. (7).
  end for
  for $d = 1 : D + 1$ **do**
    Update $\boldsymbol{\delta}^{(d)}$ by Eq. (8), (9) and (10) respectively.
  end for
  Update $\boldsymbol{\tau}$ by Eq. (11).
  // Rotation (optional)
  Compute $\boldsymbol{A}$ and $\boldsymbol{B}$ by Eq. (13) and the corresponding Cholesky decompositions.
  for $i = 1 : P_1$ **do**
    Set $\boldsymbol{Q}^{(1)}[i] := \boldsymbol{R}_1 \boldsymbol{Q}^{(1)}[i]$.
  end for
  for $i = 1 : P_D$ **do**
    Set $\boldsymbol{Q}^{(D)}[i] := \boldsymbol{Q}^{(D)}[i] \boldsymbol{R}_2$.
  end for
until Maximum iteration or convergence.

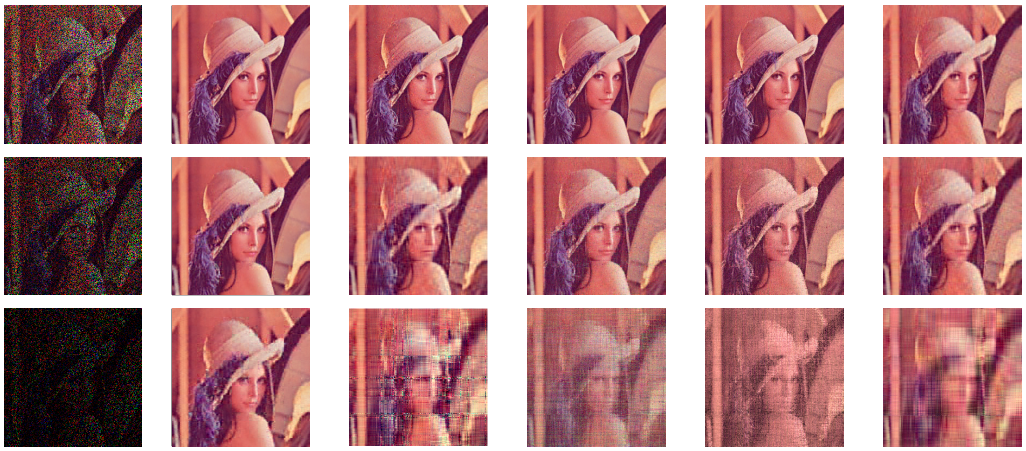
C.3. Image Inpainting

The BCPF model also uses sparse Bayesian priors to automatically choose latent factors and requires no hyperparameters. However, for the rest of models, we have to carefully tune the hyperparameters, e.g., the TT/TR-ranks. Here we compute those models under several TT/TR-rank settings and select the best performance. However, it should be noted that this is not realistic in many applications where the true signals are unknown. For our model, we simply initialize with TR format of rank 15, and we truncate the redundant factors while training, in order to reduce computational cost. For the image inpainting task, if we use the tensor train decomposition, it may require much larger TT-ranks to get comparable performance. So we only use tensor ring decomposition in this experiment.

We choose 8 standard pictures as benchmark, as shown in Figure 1. We randomly generate masks of missing rate 0.5, 0.7 and 0.9. Figure 2 shows the visualization results of the Lena image completion. Our model has better recovery results than the competitors, especially when the missing rate is high.

(a) (b) (c) (d) (e) (f) (g) (h)

Figure 1: Benchmark images.



(a) Missing (b) TRLF (c) TRALS (d) TRLRF (e) TTWOPT (f) BCPF

Figure 2: Illustration of the Lena image completion. Each row from the above to the bottom according to missing rate 0.5, 0.7 and 0.9 respectively.