Machine Learning (2022) 111:2663–2713
https://doi.org/10.1007/s10994-021-06067-7

Large scale tensor regression using kernels and variational inference

Robert Hu[1] · Geoff K. Nicholls[1] · Dino Sejdinovic[1]

Received: 9 April 2020 / Revised: 23 June 2021 / Accepted: 6 September 2021 /
Published online: 9 November 2021

Abstract
We outline an inherent flaw of tensor factorization models when latent factors are expressed as a function of side information and propose a novel method to mitigate this. We coin our methodology *kernel fried tensor* (KFT) and present it as a large-scale prediction and forecasting tool for high dimensional data. Our results show superior performance against *LightGBM* and *Field aware factorization machines* (FFM), two algorithms with proven track records, widely used in large-scale prediction. We also develop a variational inference framework for KFT which enables associating the predictions and forecasts with calibrated uncertainty estimates on several datasets.

Keywords Large scale prediction · Tensor · RKHS · Kernel methods · Variational inference · Bayesian · Uncertainty quantification

1 Introduction and related work

In recent times, industrial prediction problems (Caro and Gallien 2010; Seeger et al. 2016) are not only large scale but also high dimensional (Zhai et al. 2014). Problems of this nature are ubiquitous and the most common setting is, but not limited to, the recommendation systems (Bobadilla et al. 2013). Here we are tasked with predicting user preference (i.e. ratings, buying propensity, etc.) for a product (movies, clothing, etc). More often than not, the number of users and products far exceeds the practicality of data matrix formalism, which characterizes the perils of high dimensionality in modern prediction problems. The choice of models is often limited to boosting models such as *LightGBM* (Ke et al. 2017),

Editor: Shai Shalev-Shwartz.

Robert Hu
robert.hu@stats.ox.ac.uk

Geoff K. Nicholls
nicholls@stats.ox.ac.uk

Dino Sejdinovic
dino.sejdinovic@stats.ox.ac.uk

[1] Department of Statistics, University of Oxford, Oxford, UK

factorization machines (Juan et al. 2016; Rendle 2010) and matrix (Bobadilla et al. 2013) and tensor factorization models (Kolda and Bader 2009; Oseledets 2011). In particular, matrix/tensor factorization models are often used due to their memory-efficient representation of high dimensional data (Kuang et al. 2014). An additional benefit of factorization machines and matrix/tensor factorization models is their relative ease of extension to a Bayesian formulation, making uncertainty quantification straightforward.

A recurring problem in recommendation systems is the *cold start* problem (Bobadilla et al. 2012), where new users yield inaccurate predictions due to an absence of historical purchasing behavior. A common technique to overcome the cold start problem is to incorporate *side information* (Agarwal and Chen 2009; Xu et al. 2015; Zhang et al. 2014), which means adding descriptive covariates about each individual user (or product) to the model. While side information is not immediately applicable to factorization machines, there is extensive literature (Kim et al. 2016; Liu et al. 2019; Narita et al. 2012) on utilizing side information with matrix/tensor factorization models. There are limited choices of models when simultaneously considering scalability, uncertainty quantification, the cold start problem, and, ideally, interpretability (Rudin 2018). We contextualize our contribution by reviewing related works.

Tensor factorization is a generalization of matrix factorization to n-dimensions, where any technique that applies to tensors is directly applicable to matrices but not vice versa. We focus on tensor factorization as it offers the most flexibility and generality. In the frequentist setting, Canonical Polyadic (CP) and Tucker decomposition (Kolda and Bader 2009) are the most common factorization methods for tensors while Tensor-Train (TT) (Oseledets 2011) and Tensor-Ring (TR) (Zhao et al. 2016) decomposition are newer additions focusing on scalability. While Tucker decomposition has admirable analytical properties, the $\mathcal{O}(r^d)$ memory storage of the core tensor is infeasible in any large-scale application. CP decomposition is superseded by TT-decomposition, which in turn is extended by TR-decomposition. Due to certain pathologies (Batselier 2018) exhibited by TR, only TT remains plausible for a large-scale model. The existing methods *LightGBM*(Ke et al. 2017) and *Field-Aware Factorization Machines*(FFM) (Juan et al. 2016) are considered the gold standard[1,2] of large scale prediction, where an overall objective of this paper is to challenge this duopoly. To enhance the performance of tensor models, Du et al. (2016), Wu et al. (2019) and Zhang et al. (2019) applied neural methods for matrix and tensor factorization, where (Du et al. 2016; Zhang et al. 2019) is considered the state-of-the-art in performance. In the Bayesian domain, only FFM carries over with (Saha et al. 2017), which introduces a variational coordinate ascent method for factorization machines. There is rich literature in Bayesian matrix/tensor factorization ranging from Monte Carlo methods in the pioneering works of Salakhutdinov and Mnih (2007) to variational matrix factorization (Gönen et al. 2013; Kim and Choi 2014) and tensor factorization in Hawkins and Zhang (2018). Kim and Choi (2014) is of particular interest as they present a large-scale variational model incorporating side information. While coordinate ascent approaches proposed in Hawkins and Zhang (2018), Kim and Choi (2014) and Saha et al. (2017) are useful, we believe that the cyclical update scheme can be constraining for large scale scenarios that rely on parallelism. Further, the updating rules are hard to maintain with changes in model architecture. To mitigate maintenance of complicated gradient updates, we consider

[1] https://www.kaggle.com/c/avazu-ctr-prediction/discussion/12608.

[2] https://bit.ly/3bGGzpf.

Table 1 Where our contribution places

	Interpretable	Scalable	Tensor	Bayesian	Side information	Performant
CF-NADE (Du et al. 2016)		✓				✓
FFM (Juan et al. 2016)		✓		✓		✓
LightGBM (Ke et al. 2017)		✓			✓	✓
Tensor Factorization with Auxiliary side information (Narita et al. 2012)			✓		✓	
NTF (Wu et al. 2019)			✓			✓
KPMF (Pal and Jenamani 2018)					✓	✓
VBMF (Kim and Choi 2014)	✓	✓		✓	✓	
Kernel fried tensor	✓	✓	✓	✓	✓	✓

automatic differentiation (Paszke et al. 2017b) instead. In the context of maintainable variational inference, it would then suffice to find an analytical expression of the Evidence Lower BOund (ELBO) that is general for a family of models (Hoffman et al. 2013).

Side information is applied in two ways for matrix/tensor factorization models: implicitly through regularization schemes in He et al. (2017), Narita et al. (2012) and Pal and Jenamani (2018) Zhao et al. (2013) and directly as covariates in Agarwal and Chen (2009), Kim et al. (2016), Kim and Choi (2014) and Zhang et al. (2014). In terms of interpretability, the latter is more desirable as predictions are now an explicit expression of covariates, allowing for direct attribution analysis. A concerning observation of the results in Agarwal and Chen (2009), Kim et al. (2016) and Kim and Choi (2014) suggests that using side information in a covariate format barely improves performance and even worsens it in Agarwal and Chen (2009) using the "features only" model.

In this paper we develop a novel all-purpose large-scale prediction model that strives for a new level of versatility existing models lack. Our contribution *Kernel Fried Tensor*(KFT) aims to bridge the gap in the literature and answer the following questions:

1. Is there an interpretable tensor model that avoids constraints and complex global dependencies arising from the addition of side information but still makes full use of side information?
2. Can we formulate and characterize this model class in both primal and dual (Reproducing Kernel Hilbert Space) space?
3. Do models in this class compare favourably, for large scale prediction, to state-of-the-art models such as *LightGBM* (Ke et al. 2017), FFM (Juan et al. 2016) and existing factorization models?
4. Can we work with these new models in a scalable Bayesian context with calibrated uncertainty estimates?
5. What are the potential pitfalls of using these models? When are they appropriate to use?

Following our introduction, we were tasked with the multifaceted problem of developing a tensor model that scales, is interpretable, is Bayesian, handles side information, and provides on-par performance with the existing gold-standard.

The rest of the paper is organized as follows: Sect. 2 illustrates an important limitation of side information applied as covariates to tensors, Sect. 3 introduces and characterises KFT in both the frequentist and Bayesian setting, Sect. 4 are experiments and Sect. 5 provides an ablation study related to Question 5 (Table 1).

2 Background and problem

Tensor models are used in large-scale prediction tasks ranging from bioinformatics to industrial prediction; we work in the latter setting. While existing tensor models are versatile and scalable, they have a flaw: When we build a model of the latent factors in tensor factorization as a function of covariates (Agarwal and Chen 2009; Kim et al. 2016; Kim and Choi 2014; Zhang et al. 2014), the model may be restricted by global parameter couplings that are generated. These couplings lead to a reduction of tractability at scale. We provide a new family of tensor models which admit side information without model restriction or loss of tractability.

Tensor Train decomposition Before we explain how side information restricts tensor model expressiveness, we set out the background. Consider the task of reconstructing the tensor $\mathbf{Y} \in \mathbb{R}^{n_1 \times n_2 \cdots \times n_P}$. Many existing decomposition techniques (Kolda and Bader 2009) treat this problem. We focus on the Tensor Train (TT) decomposition (Oseledets 2011) as this generalises more readily than existing alternatives.

The n-mode (matrix) product of a tensor $\mathbf{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ with a matrix $\mathbf{U} \in \mathbb{R}^{J \times I_n}$ is $\mathbf{X} \times_n \mathbf{U}$. This product is of size $I_1 \times \dots I_{n-1} \times J \times I_{n+1} \times \dots \times I_N$. Elementwise, the product is

$$\left(\mathbf{X} \times_n \mathbf{U}\right)_{i_1 \dots i_{n-1}\, j\, i_{n+1}\, \dots\, i_N} = \sum_{i_n=1}^{I_n} x_{i_1 i_2 \dots i_N}\, u_{j i_n}.$$

We will consider the following notion of a mode product between tensor $\mathbf{X} \in \mathbb{R}^{I_1 \times \cdots I_{N-1} \times I_N}$, applying the N-th mode product $\times_N$ with a tensor $\mathbf{U} \in \mathbb{R}^{I_N \times K_1 \times K_2}$ gives $\mathbf{X} \times_N \mathbf{U} \in \mathbb{R}^{I_1 \times \cdots I_{N-1} \times K_1 \times K_2}$.

In TT, $\mathbf{Y}$ is decomposed into P latent tensors $\mathbf{V}_p \in \mathbb{R}^{R_p \times n_p \times R_{p-1}}, p = 1, \dots P$, with $\mathbf{V}_1 \in \mathbb{R}^{R_1 \times n_1 \times 1}$ and $\mathbf{V}_P \in \mathbb{R}^{1 \times n_P \times R_{P-1}}$. Here R_p is the latent dimensionality for each factor in the decomposition, with $R_1 = R_P = 1$. Let $\times_{-1} \mathbf{V}_p$ be the operation of applying the mode product to the last dimension of $\mathbf{V}_p$. We seek $\mathbf{V}_1 \dots \mathbf{V}_P$ so that

$$\begin{aligned} \mathbf{Y} &\approx \prod_{p=1}^{P} \times_{-1} \mathbf{V}_p \\ y_{i_1 \dots i_P} &\approx \sum_{r_0 \dots r_P} \prod_{p=1}^{P} v_{r_p i_p r_{p-1}}. \end{aligned} \tag{1}$$

Suppose that, associated with dimension p, we have c_p-dimensional side information denoted $\mathbf{D}_p \in \mathbb{R}^{n_p \times c_p}$. For example, if p is the dimension representing $n_p = 10{,}000$ different books, then the columns of $\mathbf{D}_p \in \mathbb{R}^{10{,}000 \times c_p}$ might contain the author of the book, page count etc. Similar to Kim et al. (2016) and Kim and Choi (2014), side information is built into the second dimension of the latent tensor $\mathbf{V}_p \in \mathbb{R}^{R_p \times c_p \times R_{p-1}}$ using the mode product $\mathbf{V}_p \times_2 \mathbf{D}_p$. It should be noted that the middle dimension of $\mathbf{V}_p$ changed from n_p to c_p to accommodate the dimensionality of the side information. For TT decomposition our approximation becomes

$$\mathbf{Y} \approx \prod_{p=1}^{P} \times_{-1} (\mathbf{V}_p \times_2 \mathbf{D}_p)$$
$$y_{i_1 \dots i_P} \approx \sum_{\substack{r_0 \dots r_P \\ i'_1 \dots i'_P}} \prod_{p=1}^{P} v_{r_p i'_p r_{p-1}} d_{i_p, i'_p} \tag{2}$$

The above example illustrates the *primal* setting where side information is applied directly. Similarly to Kim et al. (2016), we also consider kernelized side information in the reproducing kernel hilbert space (RKHS) which we will refer to as the *dual* setting.

2.1 The problem with adding side information to tensor factorization

Consider a matrix factorization problem for $\mathbf{Y} \in \mathbb{R}^{n_1 \times n_2}$ with unknown latent factors $\mathbf{U} \in \mathbb{R}^{n_1 \times R}, \mathbf{V} \in \mathbb{R}^{n_2 \times R}$. We are approximating

$$\mathbf{Y} = \begin{bmatrix} y_{11} & \dots & y_{1n_2} \\ \vdots & \ddots & \\ y_{n_1 1} & & y_{n_1 n_2} \end{bmatrix} \approx \mathbf{U} \cdot \mathbf{V}^\top = \begin{bmatrix} \sum_{r=1}^{R} u_{1r} v_{1r} & \dots & \sum_{r=1}^{R} u_{1r} v_{n_2 r} \\ \vdots & \ddots & \\ \sum_{r=1}^{R} u_{n_1 r} v_{1r} & & \sum_{r=1}^{R} u_{n_1 r} v_{n_2 r} \end{bmatrix}. \tag{3}$$

If we update u_{1r} in the approximation $y_{11} \approx \sum_{r=1}^{R} u_{1r} v_{1r}$, we change the approximation $y_{1,2} \approx \sum_{r=1}^{R} u_{1r} v_{2r}$ since they share the parameter u_{1r}. However, $y_{2,1}$ and $y_{2,2}$ remain unchanged. Parameters are coupled across rows and columns but not globally. This is the standard setup in latent factorization.

Now consider the case where we have $\mathbf{D}_1 = \mathbf{D}_2 = \mathbf{1}_{n_1 \times n_1}$. We take our latent factors to be a linear function of available side information which leads $\mathbf{U}, \mathbf{V}$ to form $\mathbf{D}_1 \mathbf{U} = \begin{bmatrix} \sum_{i=1}^{n_1} u_{i1} & \dots & \sum_{i=1}^{n_1} u_{iR} \\ \vdots & \ddots & \\ \sum_{i=1}^{n_1} u_{i1} & & \sum_{i=1}^{n_1} u_{iR} \end{bmatrix}$ and $\mathbf{D}_2 \mathbf{V}$ (similar form). It follows that

$$(\mathbf{D}_1 \mathbf{U}) \cdot (\mathbf{D}_2 \mathbf{V})^\top = \begin{bmatrix} \sum_{rij} u_{ir} v_{jr} & \dots & \sum_{rij} u_{ir} v_{jr} \\ \vdots & \ddots & \\ \sum_{rij} u_{ir} v_{jr} & & \sum_{rij} u_{ir} v_{jr} \end{bmatrix}, \tag{4}$$

is a constant matrix! We have lost all model flexibility as we are approximating $\mathbf{Y}$ with a constant. Now consider a more realistic example with $\mathbf{D}_1 = \begin{bmatrix} d_{11} & \dots & d_{1n_1} \\ \vdots & \ddots & \\ d_{n_1 1} & & d_{n_1 n_1} \end{bmatrix}$ and $\mathbf{D}_2 = \begin{bmatrix} z_{11} & \dots & z_{1n_2} \\ \vdots & \ddots & \\ z_{n_2 1} & & z_{n_2 n_2} \end{bmatrix}$. In this case

$$(\mathbf{D}_1 \mathbf{U}) \cdot (\mathbf{D}_2 \mathbf{V})^\top = \begin{bmatrix} \sum_{rij} d_{1i} z_{1j} u_{ir} v_{jr} & \dots & \sum_{rij} d_{1i} z_{n_2 j} u_{ir} v_{jr} \\ \vdots & \ddots & \\ \sum_{rij} d_{n_1 i} z_{1j} u_{ir} v_{jr} & & \sum_{rij} d_{n_1 i} z_{n_2 j} u_{ir} v_{jr} \end{bmatrix}. \tag{5}$$

Again, u_{ir} appears in all entries in our matrix approximation for $\mathbf{Y}$. However, this time changing u_{ir} will not change all entries by the same amount but rather differently across *all* entries depending on the entries of $\mathbf{D}_1$ and $\mathbf{D}_2$. This connects all entries in the approximating matrix, introduces complex global variable dependence, and makes fitting infeasible at a large scale as the optimization updates globally. The observation applies in both primal and dual representations. In the primal representation, there is a restriction in expressiveness as the rank of our approximation falls off with the rank of the side information. In this setting, near-colinearity is also a problem as it leads to unstable optimization and factorizations which are very sensitive to noise.

We see that when we add side information we may inadvertently restrict the expressiveness of our model. We formulate a new tensor model with a range and dependence structure unaffected by the addition of side information in either the primal or dual setting.

3 Proposed approach

3.1 Tensors and side information

We seek a tensor model that benefits from additional side information while not forfeiting model flexibility. We introduce two strategies, which we call *weighted latent regression* (WLR) and *latent scaling* (LS).

3.1.1 Weighted latent regression

We now return to the previous setting but with additional latent tensors $\mathbf{U}' \in \mathbb{R}^{n_1 \times R}$ and $\mathbf{V}' \in \mathbb{R}^{n_2 \times R}$. We approximate $\mathbf{Y}$ as:

$$\begin{aligned}\mathbf{Y} &\approx ((\mathbf{D}_1\mathbf{U})\circ\mathbf{U}') \cdot ((\mathbf{D}_2\mathbf{V})\circ\mathbf{V}')^\top \\ &= \begin{bmatrix} \sum_{rij} u'_{1r}v'_{1r}d_{1i}z_{1j}u_{ir}v_{jr} & \cdots & \sum_{rij} u'_{1r}v'_{n_2r}d_{1i}z_{n_2j}u_{ir}v_{jr} \\ \vdots & \ddots & \\ \sum_{rij} u'_{n_1r}v'_{1r}d_{1i}z_{1j}u_{ir}v_{jr} & & \sum_{rij} u'_{n_1r}v'_{n_2r}d_{1i}z_{1j}u_{ir}v_{jr} \end{bmatrix}\end{aligned} \tag{6}$$

By taking the Hadamard product ($\circ$) with additional tensors $\mathbf{U}'$, $\mathbf{V}'$ we recover the model flexibility and dependence structure of vanilla matrix factorization, as $\mathbf{U}'$ and $\mathbf{V}'$ are independent of $\mathbf{U}$ and $\mathbf{V}$. Here, changing u_{ir} would still imply a change for all entries by magnitudes defined by the side information, however we can calibrate these changes on a latent entrywise level by scaling each entry with $u'_{ir}v'_{jr}$. For any TT decomposition with an additional tensor $\mathbf{V}'_p$, the factorization becomes

$$\begin{aligned}\mathbf{Y} &\approx \prod_{p=1}^{P} \times_{-1}(\mathbf{V}'_p \circ (\mathbf{V}_p \times_2 \mathbf{D}_p)) \\ y_{i_1 \dots i_P} &\approx \sum_{\substack{r_1 \dots r_P \\ i'_1 \dots i'_P \\ i''_1 \dots i''_P}} \prod_{p=1}^{P} v'_{r_p i''_p r_{p-1}} v_{r_p i'_p r_{p-1}} d_{i_p, i'_p} \delta_{i_p}(i''_p)\end{aligned} \tag{7}$$

where $\delta(\cdot)$ denotes Kronecker delta. We interpret this as weighting the regression terms $\sum_{i,j} d_{pi}z_{qj}u_{ir}v_{jr}$ over indices p, q with $u_{pr}v_{qr}$ and then summing over latent indices r.

Interpretability We can decompose the estimate into weights of side information illustrated in Fig. 1. A guiding example on analyzing the latent factors is provided in Appendix C

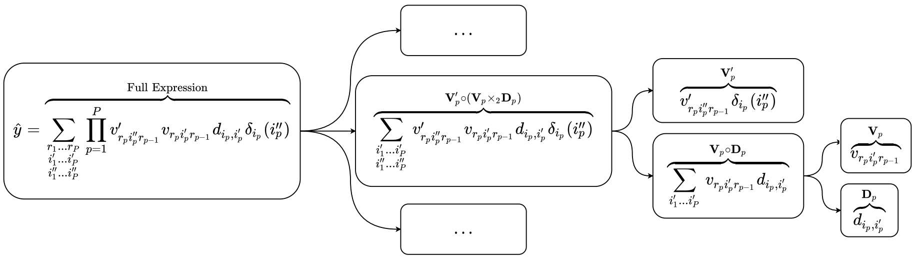

Full Expression
$\hat{y} = \sum_{\substack{r_1 \dots r_P \\ i'_1 \dots i'_P \\ i''_1 \dots i''_P}} \prod_{p=1}^{P} v'_{r_p i''_p r_{p-1}} v_{r_p i'_p r_{p-1}} d_{i_p, i'_p} \delta_{i_p}(i''_p)$
$\mathbf{V}'_p \circ (\mathbf{V}_p \times_2 \mathbf{D}_p)$
$\sum_{\substack{i'_1 \dots i'_P \\ i''_1 \dots i''_P}} v'_{r_p i''_p r_{p-1}} v_{r_p i'_p r_{p-1}} d_{i_p, i'_p} \delta_{i_p}(i''_p)$
...
...
$\mathbf{V}'_p$
$v'_{r_p i''_p r_{p-1}} \delta_{i_p}(i''_p)$
$\mathbf{V}_p \circ \mathbf{D}_p$
$\sum_{i'_1 \dots i'_P} v_{r_p i'_p r_{p-1}} d_{i_p, i'_p}$
$\mathbf{V}_p$
$v_{r_p i'_p r_{p-1}}$
$\mathbf{D}_p$
d_{i_p, i'_p}

Fig. 1 Decomposing the estimate into interpretable weights

3.1.2 Latent scaling

An alternative computationally cheaper procedure would be to consider additional latent tensors $\mathbf{U}_1 \in \mathbb{R}^{n_1 \times K}, \mathbf{U}_2 \in \mathbb{R}^{n_1 \times L}, \mathbf{V}_1 \in \mathbb{R}^{n_2 \times K}, \mathbf{V}_2 \in \mathbb{R}^{n_2 \times L}$. We would approximate

$$\begin{aligned} \mathbf{Y} &\approx (\mathbf{U}_1 \mathbf{V}_1^\top) \circ ((\mathbf{D}_1 \mathbf{U}) \cdot (\mathbf{D}_2 \mathbf{V})^\top) + (\mathbf{U}_2 \mathbf{V}_2^\top) \\ &= \begin{bmatrix} \sum_q u^1_{1k} v^1_{1k} \sum_{rij} d_{1i} z_{1j} u_{ir} v_{jr} + \sum_l u^2_{1l} v^2_{1l} & \cdots \\ \vdots & \ddots \\ \sum_q u^1_{n_1 k} v^1_{1k} \sum_{rij} d_{1i} z_{1j} u_{ir} v_{jr} + \sum_l u^2_{n_1 l} v^2_{1l} & \end{bmatrix} \end{aligned} \tag{8}$$

We have similarly regained our original model flexibility where have introduced back independence for each term by scaling ($\mathbf{V}^s$) and adding a constant ($\mathbf{V}^b$) for each regression term with a 'latent scale and bias' term. We generalize this to

$$\begin{aligned} \mathbf{Y} &\approx \left(\prod_{p=1}^{P} \times_{-1} \mathbf{V}^s_p \right) \circ \left(\prod_{p=1}^{P} \times_{-1} (\mathbf{V}_p \times_2 \mathbf{D}_p) \right) + \left(\prod_{p=1}^{P} \times_{-1} \mathbf{V}^b_p \right) \\ y_{i_1 \dots i_P} &\approx \sum_{r^s_1 \dots r^s_P} \prod_{p=1}^{P} v^s_{r_p i_p r_{p-1}} \sum_{\substack{r_1 \dots r_P \\ i'_1 \dots i'_P}} \prod_{p=1}^{P} v_{r_p i'_p r_{p-1}} d_{i_p, i'_p} + \sum_{r^b_1 \dots r^b_P} \prod_{p=1}^{P} v^b_{r_p i_p r_{p-1}} \end{aligned} \tag{9}$$

One may conjecture that adding side information to tensors through a linear operation is counterproductive due to the restrictions it imposes on the approximation, and dispute that our proposal of introducing additional tensors to increase model flexibility is futile when side information is likely to be marginally informative or potentially uninformative. As an example, return to the case of completely non-informative constant side information, $\mathbf{D}_1 = \mathbf{1}, \mathbf{D}_2 = \mathbf{1}$. In this corner case, both our proposed models reduce to regular matrix factorization: the side information regression term collapses to a constant, which in conjunction with the added terms reduces to regular tensor factorization without side information.

Interpretability We can decompose the estimate into weights of side information illustrated in Fig. 2.

A comment on identifiability It should be noted that the proposed models need not be indentifiable.

To see this, return to the scenario where side information is constant. The term $\mathbf{V}_p \times_2 \mathbf{D}_p$ has constant rows equal to row sums of $\mathbf{V}_p$. Any transformation of $\mathbf{V}_p$ which preserves these row sums leaves the fit unchanged. However, in large-scale industrial

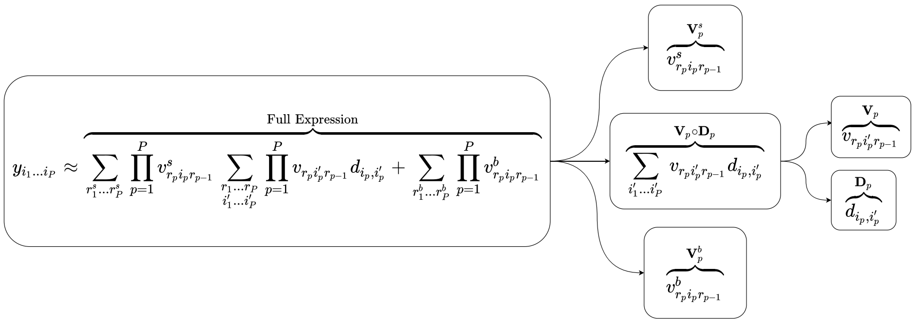

Full Expression
$y_{i_1\dots i_P} \approx \sum_{r_1^s\dots r_P^s}\prod_{p=1}^P v^s_{r_p i_p r_{p-1}} \sum_{\substack{r_1\dots r_P\\ i'_1\dots i'_P}}\prod_{p=1}^P v_{r_p i'_p r_{p-1}} d_{i_p,i'_p} + \sum_{r_1^b\dots r_P^b}\prod_{p=1}^P v^b_{r_p i_p r_{p-1}}$
$\mathbf{V}^s_p$
$v^s_{r_p i_p r_{p-1}}$
$\mathbf{V}_p \circ \mathbf{D}_p$
$\sum_{i'_1\dots i'_P} v_{r_p i'_p r_{p-1}} d_{i_p,i'_p}$
$\mathbf{V}_p$
$v_{r_p i'_p r_{p-1}}$
$\mathbf{D}_p$
d_{i_p,i'_p}
$\mathbf{V}^b_p$
$v^b_{r_p i_p r_{p-1}}$

Fig. 2 Decomposing the estimate into interpretable weights

prediction, we draw utility from good generalization performance in prediction, and parameter identifiability is secondary.

3.2 Primal regularization terms

In weight space regression, the regularization terms are given by the squared Frobenius norm. The total regularization term would be written as

$$\Lambda = \sum_p \lambda_p \|\mathbf{V}_p\|_F^2 + \lambda'_p \|\mathbf{V}'_p\|_F^2 \tag{10}$$

3.3 RKHS and the representer theorem

We extend our framework to the RKHS dual space formalism, where our extension can intuitively be viewed as a tensorized version of kernel ridge regression (Vovk 2013). The merit of this is to enhance the performance by providing an implicit non-linear feature map of side information using kernels.

Firstly consider side information $\mathbf{D}_p = \{\mathbf{x}_i^p\}_{i=1}^{n_p}, \mathbf{x}_i^p \in \mathbb{R}^{c_p}$ which are kernelized using a kernel function $k : \mathbb{R}^{c_p} \times \mathbb{R}^{c_p} \rightarrow \mathbb{R}$. Denote $k_{ij}^p = k(\mathbf{x}_i^p, \mathbf{x}_j^p) \in \mathbb{R}$ and $\mathbf{K}_p = k(\mathbf{D}_p, \mathbf{D}_p) \in \mathbb{R}^{n_p \times n_p}$. Consider a $\mathbf{V} \in \mathbb{R}^{R_1 \times n_1 \cdots \times n_Q \times R_2}$, where $Q < P$. Using the Representer theorem (Schölkopf et al. 2001) we can express $\prod_{q=1}^Q \mathbf{V} \times_{q+1} \mathbf{K}_q$ as a function in RKHS

$$\mathsf{v}_{r_1,r_2} = \sum_{n_1\dots n_q} v_{r_1 n_1 \dots n_q r_2} \prod_{q=1}^Q k_q(\cdot, \mathbf{x}_{n_q}^q). \tag{11}$$

where $\mathsf{v}_{r_1,r_2} : \mathbb{R}^{n_1} \times \cdots \times \mathbb{R}^{n_Q} \rightarrow \mathbb{R}$ and $\mathsf{v}_{r_1,r_2} \in \mathcal{H}$, which denotes the RKHS with respect to the kernels $\prod_q^Q \times k_q$. We use mode dot notation $\times_{q+1}$ here to apply kernelized side information $\mathbf{K}_q$ to each dimension of size $n_1 \dots n_Q$, where $q+1$ is used to account for the first dimension consisting of R_1.

3.4 Dual space regularization term for WLR

Consider applying another tensor with the same shape as $\mathbf{V}'$ through element wise product to robustify $\mathbf{V}$, i.e. $\mathbf{V}' \circ \prod_{q=1}^{Q} \mathbf{V} \times_{q+1} \mathbf{K}_q$. Then we have

$$\begin{aligned}\mathsf{v}_{r_1 r_2} &= \left(\sum_{n_1' \dots n_q'} v'_{r_1 n_1' \dots n_q' r_2} \prod_{q=1}^{Q} \delta_{\mathbf{x}^q_{n_q'}}(\cdot) \right) \cdot \left(\sum_{n_1 \dots n_q} v_{r_1 n_1 \dots n_q r_2} \prod_{q=1}^{Q} k(\cdot, \mathbf{x}^q_{n_q}) \right) \\ &= \sum_{\substack{n_1 \dots n_q \\ n_1' \dots n_q}} v_{r_1 n_1 \dots n_q r_2} v'_{r_1 n_1' \dots n_q' r_2} \prod_{q=1}^{Q} \delta_{\mathbf{x}^Q_{n_q'}}(\cdot) \prod_{q=1}^{Q} k(\cdot, \mathbf{x}^q_{n_q})\end{aligned} \tag{12}$$

where the regularization term for v_{r_1,r_2} is given by:

$$\begin{aligned}\Lambda &= \sum_p \lambda_p \sum_{r_1,r_2} \langle \mathsf{v}_{r_1,r_2}, \mathsf{v}_{r_1,r_2} \rangle_{\mathcal{H}} \\ &= \sum_p \lambda_p \left(\left(\prod_{q=1}^{Q} \mathbf{V} \times_{q+1} \mathbf{K}_q \right) \circ \left((\prod_{q=1}^{Q} (\mathbf{V}' \circ \mathbf{V}') \times_{q+1} \mathbf{1}_{n_q \times n_q}) \circ \mathbf{V} \right) \right)_{++}\end{aligned} \tag{13}$$

and $(\cdot)_{++}$ means summing all elements.

3.5 Dual space regularization term for LS

For LS models, the regularization term is calculated as

$$\Lambda = \sum_p \lambda_p \left[\|\mathbf{V}_p^s\|_F^2 + \sum_r \langle \mathsf{v}_r, \mathsf{v}_r \rangle_{\mathcal{H}} + \|\mathbf{V}_p^b\|_F^2 \right] \tag{14}$$

where $\mathsf{v}_{r_1,r_2} = \sum_{n_1 \dots n_q} v_{r_p n_1 \dots n_q r_{p-1}} \prod_{q=1}^{Q} k(\cdot, \mathbf{x}^Q_{n_q})$ and $\sum_r \langle \mathsf{v}_r, \mathsf{v}_r \rangle_{\mathcal{H}} = \left(\left(\prod_{q=1}^{Q} \mathbf{V} \times_{q+1} \mathbf{K}_q \right) \circ \mathbf{V} \right)_{++}$.

3.6 Scaling with random Fourier features

To make tensors with kernelized side information scalable, we rely on a random fourier feature (Rahimi and Recht 2007) (RFFs) approximation of the true kernels. RFFs approximate a translation-invariant kernel function k using Monte Carlo:

$$\hat{k}(\mathbf{x}, \mathbf{y}) = \frac{2}{M} \sum_{i=1}^{M/2} \left[\cos(\omega_i^T \mathbf{x}) \cos(\omega_i^T \mathbf{y}) + \sin(\omega_i^T \mathbf{x}) \sin(\omega_i^T \mathbf{y}) \right] \tag{15}$$

where ω_i are frequencies drawn from a normalized non-negative spectral measure Λ of kernel k. Our primary goal in using RFFs is to create a memory efficient, yet expressive method. Thus, we write

$$\hat{k}\left(\mathbf{x}_i^p, \mathbf{x}_j^p\right) \approx \phi\left(\mathbf{x}_i^p\right)^\top \phi\left(\mathbf{x}_j^p\right) \tag{16}$$

with explicit feature map $\phi : \mathbb{R}^{D_p} \to \mathbb{R}^M$, and $M \ll N_p$. In the case of RFFs,

$$\phi(\cdot)^\top = [\cos(\omega_1^T \cdot), \dots, \cos(\omega_{M/2}^T \cdot), \sin(\omega_1^T \cdot), \dots, \sin(\omega_{M/2}^T \cdot)].$$

This feature map can be applied in the primal space setting as a computationally cheap alternative to the RKHS dual setting.

A drawback of tensors with kernelized side information is the $\mathcal{O}(N_p^2)$ memory growth of kernel matrices. If one of the dimensions has a large N_p in the dual space setting, we approximate large kernels $\mathbf{K}_p$ with

$$\mathbf{V} \times_{p+1} \boldsymbol{\Phi} \times_{p+1} \boldsymbol{\Phi}^\top \approx \mathbf{V} \times_{p+1} \mathbf{K}_p, \tag{17}$$

where $\boldsymbol{\Phi} = \phi(\mathbf{D}_p) \in \mathbb{R}^{N_p \times M}$. To see that this is a valid approximation, an element $\bar{v}_{i_1 \dots i_p}$ in $\mathbf{V} \times_p \mathbf{K}_p$ is given by $\bar{v}_{i_1 \dots i_p} = \sum_{i'_p=1}^{N_p} v_{i_1 \dots i'_p} k(\mathbf{x}^p_{i_p}, \mathbf{x}^p_{i'_p})$. Using RFFs we have

$$\bar{v}_{i_1 \dots i_p} = \sum_{i'_p=1}^{N_p} \sum_{c=1}^{M} v_{i_1 \dots i'_p} \boldsymbol{\Phi}_{i'_p c} \boldsymbol{\Phi}_{i_p c} = \sum_{i'_p=1}^{N_p} v_{i_1 \dots i'_p} \phi(\mathbf{x}^p_{i'_p})^\top \phi(\mathbf{x}^p_{i_p}). \tag{18}$$

KFT with RFFs now becomes:

$$\mathsf{v}_{r_p} = \sum_{\substack{n_1 \dots n_q \\ n'_1 \dots n_q}} v_{r_p n_1 \dots n_q r_{p-1}} v'_{r_p n'_1 \dots n'_q r_{p-1}} \prod_{q=1}^{Q} \delta_{\mathbf{x}^Q_{n'_q}}(\cdot) \prod_{q=1}^{Q} \phi(\mathbf{x}^Q_{n_q}) \phi(\cdot) \tag{19}$$

with the regularization term

$$\sum_r \langle \mathsf{v}_r, \mathsf{v}_r \rangle_{\mathcal{H}} = \left(\left(\prod_{q=1}^{Q} \mathbf{V} \times_{q+1} \boldsymbol{\Phi}_q \times_q \boldsymbol{\Phi}_q^\top \right) \circ \left(\left(\prod_{q=1}^{Q} (\mathbf{V}' \circ \mathbf{V}') \times_q \mathbf{1}_{n_q \times n_q} \right) \circ \mathbf{V} \right) \right)_{++}. \tag{20}$$

For a derivation, please refer to the Appendix B.

3.7 Kernel fried tensor

Having established our new model, we coin it kernel fried tensor (KFT). Given some loss $\mathcal{L}(\mathbf{Y}, \tilde{\mathbf{Y}})$ for predictions $\tilde{\mathbf{Y}}$ the full objective is

$$\min_{\text{wrt } \mathbf{V}_p, \mathbf{V}'_p, \Theta_p} \mathcal{L}(\mathbf{Y}, \tilde{\mathbf{Y}}) + \Lambda(\mathbf{V}_p, \mathbf{V}'_p, \Theta_p) \tag{21}$$

where $\mathbf{V}_p, \mathbf{V}'_p, \Theta_p$ are parameters of the model and Θ_p are kernel parameters if we use the RKHS dual formulation. As our proposed model involves mutually dependent components with non-zero mixed partial derivatives, optimizing them jointly with a first order solver is inappropriate as mixed partial derivatives will not be considered during each gradient step. Inspired by the EM-algorithm (Dempster et al. 1977), we summarize our training procedure in Algorithm 1. By updating each parameter group sequentially and independently, we eliminate the effects of mixed partials leading to accurate gradient updates. For further details, we refer to the Appendix E.1.

Algorithm 1 Tensor training procedure

```
Require: $\{\mathbf{V}_p\}_{p=1}^P, \{\mathbf{V}'_p\}_{p=1}^P, \{\Theta_p\}_{p=1}^P$, Data, learning rate $\alpha_{\mathrm{lr}}$
  for epochs do
    for $p = 1, \ldots, P$ do
      Fix all parameters except $\mathcal{G}_p = \{\mathbf{V}_p, \mathbf{V}'_p, \Theta_p\}$
      for batch in data do
        Calculate $\mathcal{L}(\mathbf{Y}, \tilde{\mathbf{Y}}) + \Lambda$
        Update $\mathcal{G}_p$ as $\mathcal{G}_p = \mathcal{G}_p - \alpha_{\mathrm{lr}} \frac{\partial(\mathcal{L}+\Lambda)}{\partial \mathcal{G}_p}$
      end for
    end for
  end for
```

3.7.1 Joint features

From a statistical point of view, we are assuming that each of our latent tensors $\mathbf{V}_p$ factorizes $\mathbf{Y}$ into P independent components with prior distribution corresponding to $\mathcal{N}(\mathsf{v}^p_{r_{p-1}r_p}|0, \mathbf{K}_p)$, where $\mathsf{v}^p_{r_{p-1}r_p} \in \mathbb{R}^{n_p}$ is the r_{p-1}, r_p cell selected from $\mathbf{V}_p$. We can enrich our approximation by jointly modelling some dimensions p by choosing some $\mathbf{V}_p \in \mathbb{R}^{R_{p+1}\times n_{p+1}\times n_p\times R_{p-1}}$. If we denote this dimension p by p' we have that

$$\mathbf{Y} \approx \prod_{p=1}^{p'-1} \times_{-1}(\mathbf{V}_p \times_2 \mathbf{K}_p) \times_{-1} (\mathbf{V}_{p'} \times_3 \mathbf{K}_{p'} \times_2 \mathbf{K}_{p'+1}) \prod_{p=p'+2}^{P} \times_{-1}(\mathbf{V}_p \times_2 \mathbf{K}_p)$$

$$y_{i_1\ldots i_P} \approx \sum_{\substack{r_1 \ldots r_P \\ i'_1 \ldots i'_P}} \prod_{p=1}^{p'-1} v_{r_p i'_p r_{p-1}} k(\mathbf{x}^p_{i_p}, \mathbf{x}^p_{i'_p}) \cdot v_{r_p i'_{p'} i'_{p'+1} r_{p-1}}$$
$$k(\mathbf{x}^{p'}_{i_{p'}}, \mathbf{x}^{p'}_{i'_{p'}}) k\left(\mathbf{x}^{p'+1}_{i_{p'+1}}, \mathbf{x}^{p'+1}_{i'_{p'+1}}\right) \prod_{p=1}^{p'+2} v_{r_p i'_p r_{p-1}} k(\mathbf{x}^p_{i_p}, \mathbf{x}^p_{i'_p}) \tag{22}$$

and the prior would instead be given as $\mathcal{N}(\mathrm{vec}(\mathsf{v}^{p'}_{r_{p'-1}r_{p'+1}})|0, \mathbf{K}_{p'} \otimes \mathbf{K}_{p'+1})$. Here $\mathrm{vec}(\mathsf{v}^{p'}_{r_{p'-1}r_{p'+1}}) \in \mathbb{R}^{n_{p'}n_{p'+1}}$ and $\mathrm{vec}(\cdot)$ means flattening a tensor to a vector. The cell selected from $\mathbf{V}_{p'}$ now has a dependency between dimensions p' and $p'+1$. We refer to a one dimensional factorization component of TT as a *TT-core* and a multi dimensional factorization component as a *joint TT-core*.

3.8 Bayesian inference

We turn to Bayesian inference for uncertainty quantification with KFT. Assume a Gaussian conditional likelihood for an observation $y_{i_1\ldots i_P}$ with inspiration from Gönen et al. (2013) Kim and Choi (2014). For KFT-WLR we have that

$$
\begin{aligned}
&p(y_{i_1\dots i_P} | \mathbf{V}_1 \dots \mathbf{V}_P, \mathbf{V}'_1 \dots \mathbf{V}'_P) \\
&= \mathcal{N}\left(y_{i_1\dots i_P} \,\middle|\, \sum_{\substack{r_1 \dots r_P \\ i'_1 \dots i'_P \\ i''_1 \dots i''_P}} \prod_{p=1}^{P} v'_{r_p i''_p r_{p-1}} v_{r_p i'_p r_{p-1}} k(\mathbf{x}^p_{i_p}, \mathbf{x}^p_{i'_p}) \delta_{\mathbf{x}^p_{i_p}}(\mathbf{x}^p_{i''_p}), \sigma_y^2 \right).
\end{aligned}
\tag{23}
$$

The corresponding objective for KFT-LS is

$$
\begin{aligned}
&p(y_{i_1\dots i_P} | \mathbf{V}_1 \dots \mathbf{V}_P, \mathbf{V}'_1 \dots \mathbf{V}'_P) \\
&= \mathcal{N}\left(y_{i_1\dots i_P} \,\middle|\, \sum_{r^s_1 \dots r^s_P} \prod_{p=1}^{P} v^s_{r_p i_p r_{p-1}} \sum_{\substack{r_1 \dots r_P \\ i'_1 \dots i'_P}} \prod_{p=1}^{P} v_{r_p i'_p r_{p-1}} k(\mathbf{x}^p_{i_p}, \mathbf{x}^p_{i'_p}) + \sum_{r^b_1 \dots r^b_P} \prod_{p=1}^{P} v^b_{r_p i_p r_{p-1}}, \sigma_y^2 \right)
\end{aligned}
\tag{24}
$$

where σ_y^2 is a scalar hyperparameter.

3.9 Variational approximation

Our goal is to maximize the posterior distribution $p(\mathbf{V}_p, \mathbf{V}'_p | \mathbf{Y})$ which is intractable as the likelihood $p(\mathbf{Y}) = \int p(\mathbf{Y} | \mathbf{V}_1 \dots \mathbf{V}_P, \mathbf{V}'_1 \dots \mathbf{V}'_P) p(\mathbf{V}_1) ... p(\mathbf{V}_P) p(\mathbf{V}'_1) ... p(\mathbf{V}'_P) d\mathbf{V}_{1\dots P} d\mathbf{V}'_{1\dots P}$ does not have a closed form solution due to the product of Gaussians. Instead we use variational approximations for $\mathbf{V}_p, \mathbf{V}'_p$ by parametrizing distributions of the Gaussian family and optimize the evidence lower bound (ELBO)

$$
\begin{aligned}
\mathcal{L}(y_{i_1\dots i_P}, \mathbf{V}_1 \dots \mathbf{V}_P, \mathbf{V}'_1 \dots \mathbf{V}'_P) &= \mathbb{E}_q[\log p(y_{i_1\dots i_P} | \mathbf{V}_1 \dots \mathbf{V}_P, \mathbf{V}'_1 \dots \mathbf{V}'_P)] \\
&- \left(\sum_{p=1}^{P} D_{\mathrm{KL}}(q(\mathbf{V}_p) \| p(\mathbf{V}_p)) + D_{\mathrm{KL}}(q(\mathbf{V}'_p) \| p(\mathbf{V}'_p)) \right)
\end{aligned}
\tag{25}
$$

In our framework, we consider the univariate Gaussian and multivariate Gaussian as variational approximations with corresponding priors where σ_y^2 is interpreted to control the weight of the reconstruction term against the KL-term.

3.9.1 Univariate VI

Univariate KL For the case of univariate normal priors, we calculate the KL divergence as

$$D_{\mathrm{KL}}(\mathcal{N}_q \,\|\, \mathcal{N}_p) = \frac{(\mu_q - \mu_p)^2}{2\sigma_p^2} + \frac{1}{2}\left(\frac{\sigma_q^2}{\sigma_p^2} - 1 - \ln\frac{\sigma_q^2}{\sigma_p^2}\right) \tag{26}$$

where μ_q, μ_p and σ_q^2, σ_p^2 are the mean and variance for the variational approximation and prior respectively, where μ_p, σ_p^2 are chosen a priori.

Model For a univariate Gaussian variational approximation we assume the following prior structure

$$p(\mathbf{V}_p') = \prod_{r_p, i_p} \mathcal{N}(v'_{r_p i_p r_{p-1}} | \mu_p', \sigma_p'^2), \quad p(\mathbf{V}_p) = \prod_{r_p, i_p} \mathcal{N}(v_{r_p i_p r_{p-1}} | \mu_p, \sigma_p^2) \tag{27}$$

with corresponding univariate meanfield approximation

$$\begin{aligned} q(\mathbf{V}_p') &= \prod_{r_p, i_p} \mathcal{N}(v'_{r_p i_p r_{p-1}} | \mu'_{r_p i_p r_{p-1}}, \sigma'^2_{r_p i_p r'_{p-1}}), \\ q(\mathbf{V}_p) &= \prod_{r_p, i_p} \mathcal{N}(v_{r_p i_p r_{p-1}} | \mu_{r_p i_p r_{p-1}}, \sigma^2_{r_p i_p r_{p-1}}). \end{aligned} \tag{28}$$

We take $\mu_p', \sigma_p'^2, \mu_p, \sigma_p^2$ to be hyperparameters.

Weighted latent regression reconstruction term For Weighted latent regression, we express the reconstruction term as

$$\begin{aligned} &\mathbb{E}_q[\log p(y_{i_1 \dots i_P} | \mathbf{V}_1 \dots \mathbf{V}_P, \mathbf{V}_1' \dots \mathbf{V}_P')] \\ &\propto \frac{1}{\sigma_y^2}\left[\mathbf{Y}^2 - 2\mathbf{Y} \circ \left(\prod_{p=1}^{P} \times_{-1} (\mathbf{M}_p' \circ (\mathbf{M}_p \times_2 \mathbf{K}_p))\right) + \left(\prod_{p}^{P} \times_{-1} \mathbf{M}_p' \circ (\mathbf{M}_p \times_2 \mathbf{K}_p)\right)^2\right. \\ &\quad + \left(\prod_{p}^{P} \times_{-1} \Sigma_p' \circ (\Sigma_p \times_2 (\mathbf{K}_p)^2)\right) + \left(\prod_{p}^{P} \times_{-1} (\Sigma_p' \circ (\mathbf{M}_p \times_2 \mathbf{K}_p)^2)\right) \\ &\quad \left. + \left(\prod_{p}^{P} \times_{-1} \mathbf{M}_p'^2 \circ (\Sigma_p \times_2 (\mathbf{K}_p)^2)\right)\right] \end{aligned} \tag{29}$$

where $\mathbf{M}_p', \mathbf{M}_p$ and Σ_p', Σ_p correspond to the tensors containing the variational parameters $\mu'_{r_p i_p r_{p-1}}, \mu_{r_p i_p r_{p-1}}$ and $\sigma'^2_{r_p i_p r_{p-1}}, \sigma^2_{r_p i_p r_{p-1}}$ respectively. For the case of RFF's, we approximate $\Sigma_p \times_2 (\mathbf{K}_p)^2 \approx \Sigma_p \times_2 (\boldsymbol{\Phi}_p \bullet \boldsymbol{\Phi}_p)^\top \times_2 (\boldsymbol{\Phi}_p \bullet \boldsymbol{\Phi}_p)$, where $\bullet$ is the transposed Khatri–Rao product. It should further be noted that any square term means element wise squaring. We provide a derivation in the Appendix A.1.

Latent scaling reconstruction term For Latent Scaling, we express the reconstruction term as

$$\begin{aligned}
&\mathbb{E}_q[\log p(y_{i_1\dots i_P}|\mathbf{V}_1 \dots \mathbf{V}_P, \mathbf{V}'_1 \dots \mathbf{V}'_P)] \\
&\propto \frac{1}{\sigma_y^2}\Bigg[\mathbf{Y}^2 - 2\mathbf{Y}\circ\left(\prod_{p=1}^{P}(\mathbf{M}_p^s\circ(\mathbf{M}_p \times_2 \mathbf{K}_p) + \mathbf{M}_p^b)\right) \\
&\quad + \left(\left(\prod_{p=1}^{P}\times_{-1}\mathbf{M}_p^s\right)^2 + \left(\prod_{p=1}^{P}\times_{-1}\Sigma_p^s\right)\right) \\
&\quad \circ\left(\left(\prod_{p}^{P}\times_{-1}\mathbf{M}_p \times_2 \mathbf{K}_p\right)^2 + \prod_{p=1}^{P}\times_{-1}\Sigma_p \times_2 (\mathbf{K}_p)^2\right) \\
&\quad + 2\left(\prod_{p=1}^{P}\times_{-1}\mathbf{M}_p^s\right)\circ\left(\prod_{p=1}^{P}\times_{-1}\mathbf{M}_p^b\right)\circ\left(\prod_{p=1}^{P}\times_{-1}\mathbf{M}_p \times_2 \mathbf{K}_p\right) \\
&\quad + \left(\left(\prod_{p=1}^{P}\times_{-1}\mathbf{M}_p^b\right)^2 + \left(\prod_{p=1}^{P}\times_{-1}\Sigma_p^b\right)\right)\Bigg].
\end{aligned} \tag{30}$$

For details, see the Appendix A.2.

3.9.2 Multivariate VI

Multivariate KL The KL divergence between a multivariate normal prior p and a variational approximation given by q:

$$D_{\mathrm{KL}}(\mathcal{N}_q \parallel \mathcal{N}_p) = \frac{1}{2}\left[\mathrm{tr}\left(\Sigma_p^{-1}\Sigma_q\right) + (\mu_p - \mu_q)^{\top}\Sigma_p^{-1}(\mu_p - \mu_q) - k + \ln\left(\frac{\det \Sigma_p}{\det \Sigma_q}\right)\right] \tag{31}$$

Where μ_p, μ_q and Σ_p, Σ_q are the mean and covariance for the prior and variational respectively. Inspired by g-prior (Zellner 1986), we take $\Sigma_p = \mathbf{K}_p^{-1}$, where $\mathbf{K}_p^{-1}$ is the inverse kernel covariance matrix of side information for mode p. When side information is absent, we take $\Sigma_p = \mathbf{I}$. Another benefit of using the inverse is that is simplifies calculations, since we now avoid inverting a dense square matrix in the KL-term. Similar to the univariate case we choose μ_p a priori, although here it becomes a constant tensor rather than a constant scalar.

Model For the multivariate case, we consider the following priors

$$\begin{aligned}
p(\mathbf{V}'_p) &= \prod_{r_p, i_p} \mathcal{N}(v'_{r_p i_p r_{p-1}}|\mu'_p, \sigma'^2_p) \\
p(\mathbf{V}_p) &= \prod_{r_p}\mathcal{N}(\mathrm{vec}(\mathrm{v}_{r_p})|\mu_p, \prod_{q=1}^{Q_p}\otimes(\mathbf{K}_q)^{-1})
\end{aligned} \tag{32}$$

Where Q_p is the number of dimensions jointly modeled in each TT-core. For the variational approximations, we have

$$
\begin{aligned}
q(\mathbf{V}'_p) &= \prod_{r_p, i_p} \mathcal{N}(v'_{r_p i_p r_{p-1}} | \mu'_{r_p i_p r_{p-1}}, \sigma'^2_{r_p i_p r_{p-1}}) \\
q(\mathbf{V}_p) &= \prod_{r_p, i_p} \mathcal{N}(\mathrm{vec}(\mathrm{v}_{r_p}) | \mu_{r_p i_p r_{p-1}}, \prod_{q=1}^{Q_p} \otimes (\mathbf{B}_q \mathbf{B}_q^\top))
\end{aligned}
\tag{33}
$$

We take $\mu'_p, \sigma'^2_p, \mu_p$ to be hyperparameters and $\Sigma_q = \mathbf{B}_q \mathbf{B}_q^\top$.

Sampling and parametrization Calculating $\prod_{q=1}^{Q_i} \otimes \mathbf{B}_q \mathbf{B}_q^\top$ directly will yield a covariance matrix that is prohibitively large. To sample from $q(\mathbf{V}_p)$ we exploit that positive definite matrices A and B with their Cholesky decompositions L_A and L_B have the following property

$$
A \otimes B = (L_A L_A^\top) \otimes (L_B L_B^\top) = (L_A \otimes L_B)(L_A \otimes L_B)^\top. \tag{34}
$$

together with the fact that

$$
\prod_{i=1}^{N} \mathbf{X} \times_{i+1} A_i = \left(\prod_{i=1}^{N} \otimes A_i \right) \cdot \mathrm{vec}(\mathbf{X}), \tag{35}
$$

where $\mathrm{vec}(\mathbf{X}) \in \mathbb{R}^{\prod_i^N I_i \times R}$. We would then draw a sample $\mathbf{b} \sim q(\mathbf{V})$ as

$$
\mathbf{b} = \mu_{r_p} + \prod_{q=1}^{Q_i} \tilde{\mathbf{z}} \times_{q+1} \mathbf{B}_q \tag{36}
$$

where $\tilde{\mathbf{z}} \sim \mathcal{N}(0, \mathbf{I}_{\prod_{p=1}^{P} n_p})$ is reshaped into $\tilde{\mathbf{z}} \in \mathbb{R}^{\prod_{q=1} \times n_q}$. We take $\mathbf{B}_q = \mathbf{ltri}(B_q B_q^\top) + D_q$ (Ong et al. 2017), where $B_q \in \mathbb{R}^{n_q \times r}$, D_q to be a diagonal matrix and **ltri** denotes taking the lower triangular component of a square matrix including the diagonal. We choose this parametrization for a linear time-complexity calculation of the determinant in the KL-term by exploiting that $\det\left(\Sigma_q\right) = \det\left(\mathbf{B}_q \mathbf{B}_q^\top\right) = (\det\left(\mathbf{B}_q\right))^2$. In the RFF case, we take $\mathbf{B}_q = B_q$ and estimate the covariance as $\mathbf{B}_q \mathbf{B}_q^\top + D_q^2$

Weighted latent regression reconstruction term We similarly to the univariate case express the reconstruction term as

$$
\begin{aligned}
&\mathbb{E}_q[\log p(y_{i_1 \dots i_P} | \mathbf{V}_1 \dots \mathbf{V}_P, \mathbf{V}'_1 \dots \mathbf{V}'_P)] \\
&\propto \frac{1}{\sigma_y^2} \Bigg[\mathbf{Y}^2 - 2\mathbf{Y} \circ \left(\prod_{p=1}^{P} \times_{-1} (\mathbf{M}'_p \circ (\mathbf{M}_p \times_2 \mathbf{K}_p)) \right) + \left(\prod_{p}^{P} \times_{-1} \mathbf{M}'_p \circ (\mathbf{M}_p \times_2 \mathbf{K}_p) \right)^2 \\
&\quad + \left(\prod_{p}^{P} \times_{-1} \Sigma'_p \circ (\mathbf{1} \times_2 \left(\mathrm{diag}((\mathbf{K}_p \cdot \mathbf{B}_p)^2 \cdot \bar{1}) \right) \right) + \left(\prod_{p}^{P} \times_{-1} (\Sigma'_p \circ (\mathbf{M}_p \times_2 \mathbf{K}_p)^2) \right) \\
&\quad + \left(\prod_{p}^{P} \times_{-1} \mathbf{M}'^2_p \circ (\mathbf{1} \times_2 \left(\mathrm{diag}((\mathbf{K}_p \cdot \mathbf{B}_p)^2 \cdot \bar{1}) \right) \right) \Bigg]
\end{aligned}
\tag{37}
$$

where $\Sigma_p = \mathbf{B}_p \mathbf{B}_p^T$, $\mathbf{1}$ denotes a constant one tensor with the same dimensions as Σ'_p, $\bar{1} \in \mathbb{R}^{R \times 1}$ where R is the column dimension of $\mathbf{B}_p$ and Σ'_p is the same as in the univariate case. For RFF's we have that

$$
\begin{aligned}
(\mathbf{K}_p \cdot \mathbf{B}_p)^2 \cdot \bar{1} \approx& ((\boldsymbol{\Phi}_p \cdot \boldsymbol{\Phi}_p^\top) \cdot \mathbf{B}_p)^2 \cdot \bar{1} + \mathrm{vec}(D_p^2) \\
=& (\boldsymbol{\Phi}_p \cdot (\boldsymbol{\Phi}_p^\top \cdot \mathbf{B}_p))^2 \cdot \bar{1} + \mathrm{vec}(D_p^2).
\end{aligned} \tag{38}
$$

Latent scaling reconstruction term The latent scaling version has the following expression

$$
\begin{aligned}
&\mathbb{E}_q[\log p(y_{i_1 \dots i_P} | \mathbf{V}_1 \dots \mathbf{V}_P, \mathbf{V}'_1 \dots \mathbf{V}'_P)] \\
&\propto \frac{1}{\sigma_y^2} \Bigg[\mathbf{Y}^2 - 2\mathbf{Y} \circ \Bigg(\prod_{p=1}^{P} (\mathbf{M}_p^s \circ (\mathbf{M}_p \times_2 \mathbf{K}_p) + \mathbf{M}_p^b) \Bigg) \\
&+ \Bigg(\Bigg(\prod_{p=1}^{P} \times_{-1} \mathbf{M}_p^s \Bigg)^2 + \Bigg(\prod_{p=1}^{P} \times_{-1} \Sigma_p^s \Bigg) \Bigg) \\
&\circ \Bigg(\Bigg(\prod_{p}^{P} \times_{-1} \mathbf{M}_p \times_2 \mathbf{K}_p \Bigg)^2 + \prod_{p=1}^{P} \times_{-1} (\mathbf{1} \times_2 (\mathrm{diag}((\mathbf{K}_p \cdot \mathbf{B}_p)^2 \cdot \bar{1}))) \Bigg) \\
&+ 2 \Bigg(\prod_{p=1}^{P} \times_{-1} \mathbf{M}_p^s \Bigg) \circ \Bigg(\prod_{p=1}^{P} \times_{-1} \mathbf{M}_p^b \Bigg) \circ \Bigg(\prod_{p=1}^{P} \times_{-1} \mathbf{M}_p \times_2 \mathbf{K}_p \Bigg) \\
&+ \Bigg(\Bigg(\prod_{p=1}^{P} \times_{-1} \mathbf{M}_p^b \Bigg)^2 + \Bigg(\prod_{p=1}^{P} \times_{-1} \Sigma_p^b \Bigg) \Bigg) \Bigg].
\end{aligned} \tag{39}
$$

For details, see the Appendix A.2.

RFFs and KL divergence Using $(\boldsymbol{\Phi}_p \boldsymbol{\Phi}_p^\top)^{-1} \approx (\mathbf{K}_p)^{-1}$ as our prior covariance, we observe that the KL-term presents computational difficulties as a naive approach would require storing $(\mathbf{K}_p)^{-1} \in \mathbb{R}^{n_p \times n_p}$ in memory. Assuming we take $\Sigma_p = BB^\top, B \in \mathbb{R}^{n_p \times R}$, we can manage the first term by using the equivalence

$$
(A \bullet B) \cdot (A \bullet B)^\top = (AA^\top) \circ (BB^\top). \tag{40}
$$

Consequently, we have that

$$
\mathrm{tr}(\Sigma_p^{-1} \Sigma_q) = \Bigg(\underbrace{(\boldsymbol{\Phi}\boldsymbol{\Phi}^\top)}_{\text{Using } (\mathbf{K}_p)^{-1}} \circ BB^\top \Bigg)_{++} = \Big((\boldsymbol{\Phi} \bullet B)^\top \cdot (\boldsymbol{\Phi} \bullet B) \Big)_{++}. \tag{41}
$$

We can calculate the second term using (34) and (35). For the third term, we remember Weinstein–Aronszajn's identity

$$
\det(I_m + AB) = \det(I_n + BA) \tag{42}
$$

where $A \in \mathbb{R}^{m \times n}, B \in \mathbb{R}^{n \times m}$ and AB is trace class. If we were to take our prior covariance matrix to be $\Sigma_p = (\boldsymbol{\Phi}_p \boldsymbol{\Phi}_p^\top + I_{n_p})^{-1} \approx (\mathbf{K}_p + I_{n_p})^{-1}$ and our posterior covariance matrix to be approximated as $\Sigma_q = BB^\top + I_{n_p}$, we could use Weinstein–Aronszajn's identity to calculate the third log term in a computationally efficient manner.

From a statistical perspective, adding a diagonal to the covariance matrix implies regularizing it by increasing the diagonal variance terms. Taking inspiration from Kim and Teh (2017), we can further choose the magnitude σ of the regularization

$$
\begin{aligned}
\det(\sigma^2 I_n + \boldsymbol{\Phi}\boldsymbol{\Phi}^\top) &= (\sigma^2)^n \det(I_n + \sigma^{-2}\boldsymbol{\Phi}\boldsymbol{\Phi}^\top) \\
&= (\sigma^2)^n \det(I_m + \sigma^{-2}\boldsymbol{\Phi}^\top\boldsymbol{\Phi}) \\
&= (\sigma^2)^{n-m} \det(\sigma^2 I_m + \boldsymbol{\Phi}^\top\boldsymbol{\Phi})
\end{aligned}
$$

The KL expression then becomes

$$
\begin{aligned}
D_{\mathrm{KL}}(\mathcal{N}_q \parallel \mathcal{N}_p) = \frac{1}{2}\Bigg[&\left((\boldsymbol{\Phi} \bullet B)^\top \cdot (\boldsymbol{\Phi} \bullet B)\right)_{++} + \sigma_p^2 (B \circ B)_{++} \\
&+ \sigma_q^2 (\boldsymbol{\Phi} \circ \boldsymbol{\Phi})_{++} + \sigma_q^2 \sigma_p^2 N + (\mu_p - \mu_q)^\top (\boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \sigma_p^2 I)(\mu_p - \mu_q) \\
&- k + \ln\left(\frac{|(\sigma_p^2)^{N-I} \det(\boldsymbol{\Phi}^\top \boldsymbol{\Phi} + \sigma_p^2 I)|^{-1}}{|(\sigma_q^2)^{N-I} \det(B^\top B + \sigma_q^2 I)|} \right) \Bigg].
\end{aligned} \tag{43}
$$

3.9.3 Calibration metric

We evaluate the overall calibration of our variational model using the sum

$$
\Xi = \sum_{\alpha \in \Omega} |\xi_{1-2\alpha} - (1 - 2\alpha)| \tag{44}
$$

of the *calibration rate* $\xi_{1-2\alpha}$, which we define as

$$
\xi_{1-2\alpha} = \frac{\text{number of } y_{i_1 \dots i_P} \text{ within } 1 - 2\alpha \text{ confidence level}}{\text{total number of } y_{i_1 \dots i_P}} \tag{45}
$$

where we consider α to take values in $\{0.05, 0.15, 0.25, 0.35, 0.45\}$. This calibration rate can be understood as the true (frequentist) coverage probability and $1 - 2\alpha$ as the nominal coverage probability. The model is calibrated when the true coverage probability is close to the nominal coverage probability, that is when Ξ is small. To ensure that our model finds a meaningful variational approximation, we take our hyperparameter selection criteria to be:

$$
\eta_{\text{criteria}} = \Xi - R^2 \tag{46}
$$

where $R^2 := 1 - \frac{\sum_{i_1 \dots i_n} (y_{i_1 \dots i_n} - \hat{y}_{i_1 \dots i_n})^2}{\mathrm{Var}(\mathbf{Y})}$ is the coefficient of determination (Draper and Smith 1966) calculated using the "mean terms" $\left(\prod_{p=1}^P \times_{-1} (\mathbf{M}'_p \circ (\mathbf{M}_p \times_2 \mathbf{K}_p)) \right)$ or $\left(\prod_{p=1}^P (\mathbf{M}_p^s \circ (\mathbf{M}_p \times_2 \mathbf{K}_p) + \mathbf{M}_p^b) \right)$ as predictions $\hat{y}_{i_1 \dots i_n}$. If we only use $\eta_{\text{criteria}} = \Xi$, we argue that there is an inductive bias in choosing α's which may lead to an approximation that is calibrated per se, but not meaningful as the modes are incorrect (low R^2 value). Similarly to the frequentist case, we use an EM-inspired optimization strategy in Algorithm 2. The main idea is to find the mode and variance parameters of our variational approximation in a mutually exclusive sequential order, starting with the modes. Similar to the frequentist case, the reconstruction term of the ELBO has terms that both contain Σ_p and $\mathbf{M}'_p$ which motivates the EM-inspired approach. For further details, please refer to the Appendix E.2.

Algorithm 2 Bayesian tensor training procedure

```
Require: $\{\mathbf{M}_p\}_{p=1}^P, \{\mathbf{M}'_p\}_{p=1}^P, \{\Sigma_p\}_{p=1}^P, \{\Sigma'_p\}_{p=1}^P, \{\Theta_p\}_{p=1}^P$, Data, learning rate $\alpha_{\mathrm{lr}}$
  for epochs do
      for parameter group $\mathcal{Z}$ in $\left\{ \left(\{\mathbf{M}_p\}_{p=1}^P, \{\mathbf{M}'_p\}_{p=1}^P, \{\Theta_p\}_{p=1}^P\right), \left(\{\Sigma_p\}_{p=1}^P, \{\Sigma'_p\}_{p=1}^P\right) \right\}$ do
          for sub-parameter group $\mathcal{G}$ in $\mathcal{Z}$ do
              for $p = 1, \dots, P$ do
                  Fix all parameters except $\mathcal{G}_p \subset \mathcal{G}$
                  for batch in data do
                      Calculate ELBO $\mathcal{L}_{\mathrm{ELBO}}$
                      Update $\mathcal{G}_p$ as $\mathcal{G}_p = \mathcal{G}_p - \alpha_{\mathrm{lr}} \frac{\partial \mathcal{L}_{\mathrm{ELBO}}}{\partial \mathcal{G}_p}$
                  end for
              end for
          end for
      end for
  end for
```

3.10 Extension to forecasting

KFT in its current form is fundamentally unable to accommodate forecasting problems. To see this, we first consider the forecasting problem of predicting observation y_T from previous observations $\mathbf{y}_t = [y_0, \dots, y_t], \quad 0 < t < T$ using model $f(\mathbf{y}_t)$. The model is then optimized by minimizing

$$\mathcal{L}(y_T, \mathbf{y}_t) = \|y_T - f(\mathbf{y}_t)\|^2 \tag{47}$$

for all T. Forecasting problems assume that the model does not have access to future y_{T+1} outside the training set and assumes it to learn y_{T+1} through an autoregressive assumption. Imposing this assumption on KFT would imply that latent factorizations for time-indexed $T+1$ would remain untrained with the current training procedure, as we do not access to these indices during training. With untrained latent factorizations for $T+1$, any forecast would at best be random.

However, we can easily extend KFT to be autoregressive by directly applying (Yu et al. 2016).

3.10.1 Frequentist setting

We consider the Temporal Regularized Matrix Factorization (TRMF) framework presented in Yu et al. (2016)

$$\mathcal{T}_{\mathrm{AR}}(\mathbf{X} \mid \mathbb{L}, \mathcal{W}, \eta) := \frac{1}{2} \sum_{t=m}^{T} \left\| \boldsymbol{x}_t - \sum_{l \in \mathbb{L}} W^{(l)} \boldsymbol{x}_{t-l} \right\|^2 + \frac{\eta}{2} \sum_t \|\boldsymbol{x}_t\|^2 \tag{48}$$

where $\mathbf{X} \in \mathbb{R}^{T \times k}$ is the temporal factorization component in the matrix factorization $\mathbf{U} \cdot \mathbf{X}^\top$ and $\mathbf{x}_t$ denotes $\mathbf{X}$ sliced at time index t. Further we take $\mathcal{W} = \left\{ W^{(l)} \in \mathrm{diag}(\mathbb{R}^k) \mid l \in \mathbb{L} \right\}$(i.e. set of diagonal matrices) and $\mathbb{L} = \{l_i < T \mid i = 1, \dots, I\}$ as the set of time indices to lag.

Additionally, the regularization weight η is needed to ensure that $\mathbf{X}$ varies smoothly. However in KFT, such regularization already exists and it suffices to consider

$$\mathcal{T}_{\mathrm{AR}}(\mathbf{X} \mid \mathbb{L}, \mathcal{W}) = \frac{1}{2}\sum_{t=m}^{T}\left\| \boldsymbol{x}_t - \sum_{l\in\mathbb{L}} W^{(l)}\boldsymbol{x}_{t-l}\right\|^2 .$$

Forecasting for WLR TRMF can be extended to WLR by simply taking $\mathbf{X} = \mathbf{V}'_t \circ (\mathbf{V}_t \times_2 \mathbf{D}_t) \in \mathbb{R}^{r_{T+1}\times T \times r_{T-1}}$ and $\mathcal{W} = \left\{ W^{(l)} \in \mathbb{R}^{r_{T+1}\times r_{T-1}} \mid l \in \mathbb{L}\right\}$, which then yields

$$\mathcal{T}_{\mathrm{AR}}(\mathbf{X} \mid \mathbb{L}, \mathcal{W}) = \frac{1}{2}\sum_{t=m}^{T}\left\| \boldsymbol{x}_t - \sum_{l\in\mathbb{L}} W^{(l)}\circ\boldsymbol{x}_{t-l}\right\|^2 , \tag{49}$$

which we coin KFTRegularizer (KFTR).[3] We follow the same training strategy proposed in Yu et al. (2016) by sequentially updating $\mathcal{F} = \{\mathbf{V}_p, \mathbf{V}'_p \mid p = 1\dots, P\}\backslash\mathbf{X}$, $\mathbf{X}$ and $\mathcal{W}$.

Forecasting for LS KFTR can also be applied to the LS variant by applying the temporal regularization to all three components $\mathbf{X}^s = \mathbf{V}^s_t$, $\mathbf{X}^b = \mathbf{V}^b_t$ and $\mathbf{X} = \mathbf{V}_t \times \mathbf{D}_t$. We then apply Eq. (49) to each term.

3.10.2 Bayesian setting

KFTR is extended to the Bayesian setting by optimizing the quantity

$$\log p(\mathbf{X}_T, \dots, \mathbf{X}_m) = \sum_{t=m}^{T} \log p(\mathbf{X}_t) \tag{50}$$

in addition to the ELBO. The probability distribution $p(\cdot)$ is assumed to be a univariate normal with fixed variance σ^2_{KFTR}. We define

$$\mathbf{X}_t = \left[\mathbf{V}'_{\mathrm{time}} \circ (\mathbf{V}_{\mathrm{time}} \times_2 \mathbf{D}_{\mathrm{time}})\right]_t \tag{51}$$

as functions of variational variables $\mathbf{V}, \mathbf{V}'$. Here $[\cdot]_t$ means slicing the tensor at index t. As an autoregressive dependency on $\mathbf{X}_t$ is required, we take

$$p(\mathbf{X}_t) = \int p(\mathbf{X}_t \mid \mathbf{X}_{l_1}, \dots, \mathbf{X}_{l_K}) p(\mathbf{X}_{l_1}) \dots p(\mathbf{X}_{l_K}) d\mathbf{X}_{l_1} \dots \mathbf{X}_{l_K}. \tag{52}$$

However $\mathbf{X}_t$ is composed of variational variables $\mathbf{V}, \mathbf{V}'$ and thus we can write

$$\begin{aligned} \log p(\mathbf{X}_t) &= \log \int p(\mathbf{X}_t \mid \mathbf{X}_{l_1}, \dots, \mathbf{X}_{l_K}) q(\mathbf{V}) q(\mathbf{V}') d\mathbf{V} d\mathbf{V}' \\ &= \log \mathbb{E}_q[p(\mathbf{X}_T \mid \mathbf{X}_{l_1}, \dots, \mathbf{X}_{l_K})] \geq \mathbb{E}_q[\log p(\mathbf{X}_T \mid \mathbf{X}_{l_1}, \dots, \mathbf{X}_{l_K})]. \end{aligned} \tag{53}$$

The log-expectation is intractable, so we use Jensen's inequality and optimize a lower bound instead. We then arrive at the following expression

[3] The recursive abbreviation reflects the recursive nature of the autoregressive regularization!

$$\begin{aligned}
\mathbb{E}_q[\log p(\mathbf{X}_T \mid \mathbf{X}_{l_1}, \dots, \mathbf{X}_{l_K})] &\propto \frac{1}{\sigma^2_{\mathrm{TRMF}}} \mathbb{E}_q\left[\left(x_{r_p,i_t=T,r_{p+1}} - \sum_{k=1}^{K} w_k x_{r_p,i_t=l_k,r_{p+1}}\right)^2\right] \\
&= \frac{1}{\sigma^2_{\mathrm{TRMF}}} \mathbb{E}_q\left[\left(x^2_{r_p,i_t=T,r_{p+1}} - 2x_{r_p,i_t=T,r_{p+1}} \sum_{k=1}^{K} w_k x_{r_p,i_t=l_k,r_{p+1}}\right.\right. \\
&\quad \left.\left. +(\sum_{k=1}^{K} w_k x_{r_p,i_t=l_k,r_{p+1}})^2\right)\right].
\end{aligned} \tag{54}$$

We calculate the expression by taking expectations of the x-terms with respect to v, v' and arrive at

$$\begin{aligned}
\mathbb{E}_q[\log p(\mathbf{X}_T \mid \mathbf{X}_{l_1}, \dots, \mathbf{X}_{l_K})] &\propto \left[(\mathbf{M}'^2 + \Sigma')\circ\left[(\mathbf{M} \times_2 \mathbf{K})^2 + \Sigma \times_2 \mathbf{K}^2\right]\right]_T \\
&\quad - 2\left[\mathbf{M}'\circ(\mathbf{M} \times_2 \mathbf{K})\right]_T \circ\left(\sum_{k=1}^{K} \mathbf{W}_k \circ\left[\mathbf{M}'\circ(\mathbf{M} \times_2 \mathbf{K})\right]_{l_k}\right) \\
&\quad + \sum_{k=1}^{K} \mathbf{W}_k^2 \circ\left[(\mathbf{M}'^2 + \Sigma')\circ\left[(\mathbf{M} \times_2 \mathbf{K})^2 + \Sigma \times_2 \mathbf{K}^2\right]\right]_{l_k} \\
&\quad + \left(\sum_{k=1}^{K} \mathbf{W}_k \circ\left[\mathbf{M}'\circ(\mathbf{M} \times_2 \mathbf{K})\right]_{l_k}\right)^2 - \sum_{k=1}^{K} \mathbf{W}_k^2 \circ\left[\mathbf{M}'\circ(\mathbf{M} \times_2 \mathbf{K})\right]^2_{l_k}.
\end{aligned} \tag{55}$$

It should be noted that the above expression considers the WLR case for a univariate mean-field model. For a complete derivation and an expression for the multivariate meanfield model and the LS version, we refer to Appendix A.2.2.

3.11 Complexity analysis

We give a complexity analysis for all variations of KFT in the frequentist setting and Bayesian setting.

Theorem 1 *KFT has computational complexity*

$$\mathcal{O}\left(\max_p \left(n_p c_p r_p r_{p-1}\right) + \left(\max_p r_p\right)^P\right)$$

and memory footprint $\mathcal{O}\left(P \cdot \left(\max_p \left(n_p r_p r_{p-1}\right) + \max_p \left(n_p c_p\right)\right)\right)$ *for a gradient update on a batch of data. In the dual case, we take* $c_p = n_p$.

We provide proof in the appendix. It should be noted that one can reduce complexity and memory footprint by permuting the modes of the tensor such that the larger modes are on the edges, i.e. when $r_p = 1$ or $r_{p-1} = 1$. Then n_p will only scale with r_{p-1} or r_p.

KFTR complexity The additional complexity associated with adding an autoregressive regularization term is at worst $\mathcal{O}(r_p r_{p-1} KT)$, where K is the number of lags and T the size of the temporal mode. As this term scales linearly with K, it does not have an overall impact on the complexity of KFT.

4 Experiments

The experiments are divided into analyzing the frequentist version and bayesian version of KFT. Frequentist KFT is compared against competing methods on prediction and forecasting on various high dimensional datasets. Datasets we use are summarized in Table 2. For Bayesian KFT, we investigate the performance of the Bayesian version of KFT with a focus on the calibration of the obtained posterior distributions, by comparing nominal coverage probabilities to the true coverage probabilities.

4.1 Predictive performance of KFT

We compare KFT to the established FFM and LightGBM on the task of prediction on three different datasets, Retail Sales, Movielens-20M and Alcohol Sales (cf. Table 2). We compare KFT using squared loss. LightGBM is a challenging benchmark as it has continuously received development, engineering, and performance optimization since its inception in 2017. We execute our experiments by running 20 iterations of hyperopt

Table 2 Summary of datasets

Dataset	Mean	SD	N	Dimensionality
Alcohol Sales	11.567	37.678	2,816,946	$3476 \times 4542 \times 24$
Movielens-20M	3.529	1.051	19,800,443	$138493 \times 10,370 \times 24$
Fashion Retail Sales	2.430	3.968	24,291,539	$80 \times 400,000 \times 24$
CCDS	−0.000	0.997	331,500	$17 \times 125 \times 156$
Traffic	0.053	0.045	10,167,809	$10,560 \times 963$

(Bergstra et al. 2013) for all methods to find the optimal hyperparameter configuration constrained with a memory budget of 16GB (which is the memory limit of a high-end GPU) for 5 different seeds, where the seed controls how the data is split. We split our data into 60% training, 20% validation, and 20% testing. We report scores in R^2, since it provides a normalized goodness-of-fit score and measure the performance in terms of R^2-value on test data. For further details on hyperparameter range, data, and pre-processing, see Appendix F. The results are reported in Table 3, where the best results are boldfaced. We observe that KFT has configurations that can outperform the benchmarks with a good margin. Furthermore, the dual space models are generally doing better than their primal counterparts. We hypothesize that the enhanced expressiveness of kernelized side information is the reason for this.

The next experiment makes a direct comparison of KFT to recent Matrix Factorization methods tailored for the Movielens-1M and Movielens-10M datasets. The purpose of this experiment is to further demonstrate the competitiveness of KFT on recommendation tasks. Table 4 gives comparison of KFT RMSE on Movielens-1M and Movielens-10M to RMSEs of existing methods which are reported in respective paper. KFT outperforms existing non-neural models and marginally underperforms compared to neural-based state-of-the-art models for matrix factorization. In comparison to tensor factorization, KFT outperforms NTF despite NTF being a neural model. The RMSE is generally higher for tensor factorization, as data becomes sparser with each additional dimension.

Table 3 KFT results

Method/dataset	Retail sales	Movielens-20M	Alcohol sales
P-way, WLR, Primal	0.108 ± 0.435	0.38 ± 0.012	0.69 ± 0.015
P-way, WLR, Dual	0.571 ± 0.007	$\mathbf{0.390} \pm \mathbf{0.005}$	$\mathbf{0.704} \pm \mathbf{0.013}$
2-way, time only , WLR, Primal	0.466 ± 0.034	0.168 ± 0.013	0.472 ± 0.025
2-way, time only , WLR, Dual	$\mathbf{0.594} \pm \mathbf{0.005}$	0.349 ± 0.01	0.692 ± 0.013
P-way, LS, Primal	0.152 ± 0.033	0.272 ± 0.032	0.542 ± 0.026
P-way, LS, Dual	0.027 ± 0.006	0.281 ± 0.027	0.562 ± 0.022
2-way, time only, LS, Primal	0.505 ± 0.008	0.065 ± 0.042	0.466 ± 0.022
2-way, time only,LS, Dual	0.458 ± 0.015	0.326 ± 0.024	0.699 ± 0.012
Naive methods			
P-way Vanilla, Primal	-0.374 ± 0.007	-2.197 ± 0.007	0.025 ± 0.012
P-way Vanilla, Dual	-0.087 $\pm$ 0.017	-4.837 ± 0.028	0.196 ± 0.008
P-way Vanilla, No side information	0.385 ± 0.002	0.147 ± 0.038	0.477 ± 0.015
Benchmark models			
LightGBM	0.581 ± 0.009	0.303 ± 0.003	0.646 ± 0.015
FFM	0.48 ± 0.032	0.293 ± 0.021	0.64 ± 0.019
Linear	0.058 ± 0.001	0.185 ± 0	0.03 ± 0.002

Performance measured in R^2

Table 4 Frequentist results in RMSE compared to reported results

Method	Movielens-1M (64/16/20%)	Movielens-1M (85.5/4.5/10%)	Movielens-10M (64/16/20%)	Movielens-10M (85.5/4.5/10%)	Neural	Claims SOTA
CF-NADE (Du et al. 2016)		**0.829**		0.771	✓	✓
NTF (Wu et al. 2019)	0.8829[a]				✓	✓
KPMF (Pal and Jenamani 2018)	0.8554					✓
STAR-GCN (Zhang et al. 2019)		0.832 ± 0.0016		$\mathbf{0.770 \pm 0.0001}$	✓	✓
HIRE (Liu et al. 2019)	0.8607					✓
KFT, P-way, WLR, Dual, movie-genome side info	$\mathbf{0.85 \pm 0.002/0.872}^{a} \pm \mathbf{0.003}$	0.835 ± 0.002	0.796 ± 0.002	0.784 ± 0.002		

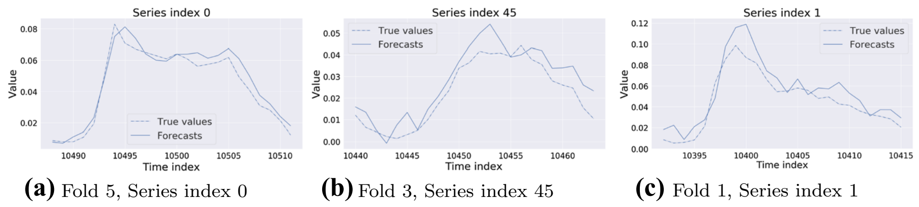
[a]Indicates a three dimensional tensor factorization for exact comparison with NTF

Table 5 Traffic results

	NRSME	ND
KFT, P-way, WLR, dual	**0.405**	0.181
TRMF (Yu et al. 2016)	0.423	0.187
DeepAR (Salinas et al. 2019)	0.420	**0.17**

Table 6 CCDS results

	RSME
KFT, P-way, WLR, dual	**0.8143**
Orthogonal (Yu et al. 2014)	0.8325



(a) Fold 5, Series index 0 **(b)** Fold 3, Series index 45 **(c)** Fold 1, Series index 1

Fig. 3 Forecasts on the Traffic dataset using KFTR.

For forecasting problems, we replicate the experiments in Yu et al. (2016) and Salinas et al. (2019) for the traffic dataset and Yu et al. (2014) for the CCDS data and compare against KFT in Tables 5 and 6.

We plot forecasts for different plots and series in Fig. 3.

4.2 Calibration study of Bayesian KFT

We use the same setup as in the frequentist case, modulo the hyperparameter evaluation objective which instead is (46). For details on hyperparameter choices and data preparation, please refer to the Appendix F. We run a regular tensor factorization without side information as a benchmark for performance, which is intended to mimic (Hawkins and Zhang 2018) as a comparison. We summarize the results in Table 7. We obtain calibrated variational approximations and observe that models using side information yield better predictive performance but that their calibration becomes slightly worse. By using a Bayesian framework we seem to generally lose some predictive performance compared to the corresponding frequentist methods, except in the case of Movielens-20M. We provide a visualization of the calibration ratios for all datasets in Fig. 4.

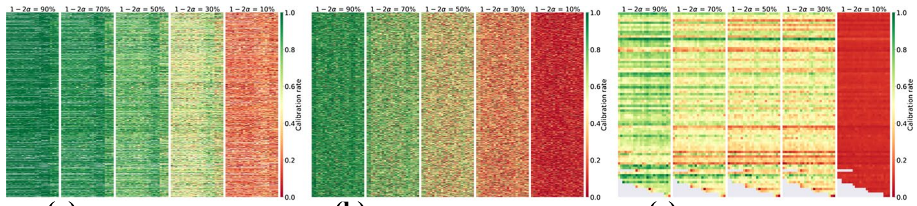
Table 7 Table of results for Bayesian KFT. Here we denote $\Xi_\alpha = |\xi_{1-2\alpha} - (1 - 2\alpha)|$, see Sect. 3.9.3 for more details

Model	Dataset	$\Xi_{0.05}$	$\Xi_{0.15}$	$\Xi_{0.25}$	$\Xi_{0.35}$	$\Xi_{0.45}$	Ξ	R^2	η_{criteria}	ELBO	log-likelihood
P-way, WLR Dual, Univ.	Alcohol	0.08 ±0.011	0.236 ±0.029	0.346 ±0.054	0.349 ±0.072	0.161 ±0.04	1.171 ±0.206	**0**.677 ±**0**.013	0.494 ±0.219	3219.538 ±776.927	14403.153 ±6827.637
P-way, WLR Dual, Multv.	Alcohol	0.065 ±0.045	0.217 ±0.101	0.345 ±0.171	0.411 ±0.246	0.245 ±0.18	1.282 ±0.739	0.559 ±0.024	0.724 ±0.719	517363.686 ±768293.97	628416.438 ±941682.766
P-way, LS Dual, Univ.	Alcohol	0.1 ±0.0	0.299 ±0.001	0.496 ±0.002	0.676 ±0.006	0.509 ±0.013	2.079 ±0.021	0.643 ±0.009	1.436 ±0.011	305418.812 ±111428.387	7469370.333 ±2863617.218
P-way, LS Dual, Multv.	Alcohol	0.1 ±0.0	0.299 ±0.001	0.496 ±0.003	0.678 ±0.013	0.519 ±0.033	2.093 ±0.05	0.668 ±0.004	1.425 ±0.053	1975127.667 ±1804481.827	33060096.333 ±39891533.692
P-way No Side Info	Alcohol	0.051 ±0.005	0.188 ±0.013	0.282 ±0.027	0.279 ±0.038	0.123 ±0.021	**0**.924 ±**0**.104	0.671 ±0.01	**0**.253 ±**0**.106	1334.163 ±35.865	5890.084 ±985.002
P-way, WLR Dual, Univ.	Movielens-20m	0.085 ±0.006	0.231 ±0.017	0.305 ±0.035	0.261 ±0.042	0.102 ±0.02	0.983 ±0.119	**0**.374 ±**0**.003	0.609 ±0.115	468.426 ±44.554	499.725 ±43.162
P-way, WLR Dual, Multv.	Movielens-20m	0.1 ±0.0	0.3 ±0.0	0.5 ±0.0	0.69 ±0.006	0.518 ±0.038	2.108 ±0.044	0.371 ±0.003	1.737 ±0.041	24548.967 ±333.533	29745.238 ±2509.768
P-way, LS Dual, Univ.	Movielens-20m	0.031 ±0.013	0.063 ±0.055	0.075 ±0.055	0.058 ±0.04	0.022 ±0.014	**0**.248 ±**0**.174	0.316 ±0.029	−0.068 ±**0**.196	355.079 ±98.181	361.622 ±100.985
P-way, LS Dual, Multv.	Movielens-20m	0.097 ±0.005	0.277 ±0.035	0.42 ±0.1	0.466 ±0.184	0.244 ±0.134	1.503 ±0.454	0.326 ±0.012	1.177 ±0.454	−56059.229 ±7918.906	−55766.211 ±7650.988

Table 7 (continued)

Model	Dataset	$\varXi_{0.05}$	$\varXi_{0.15}$	$\varXi_{0.25}$	$\varXi_{0.35}$	$\varXi_{0.45}$	$\varXi$	R^2	η_{criteria}	ELBO	log-likelihood
P-way No Side Info	Movielens-20m	0.089 ±0.005	0.234 ±0.012	0.301 ±0.018	0.245 ±0.015	0.092 ±0.005	0.961 ±0.053	0.285 ±0.004	0.676 ±0.053	99.511 ±21.923	128.062 ±20.528
2-way, WLR Dual, Univ.	Retail	0.006 ±0.006	0.053 ±0.009	0.083 ±0.01	0.072 ±0.008	0.029 ±0.003	0.242 ±0.026	**0**.547 ±**0**.004	−0.304 ±0.029	1007.522 ±33.546	1072.142 ±33.192
2-way, WLR Dual, Multv.	Retail	0.087 ±0.014	0.064 ±0.018	0.037 ±0.017	0.017 ±0.013	0.004 ±0.004	**0**.209 ±**0**.067	0.546 ±0.004	−0.337 ±**0**.065	−68984229.333 ±635903.157	−68984202.667 ±635883.04
2-way, LS Dual, Univ.	Retail	0.086 ±0.004	0.247 ±0.012	0.366 ±0.022	0.386 ±0.031	0.188 ±0.02	1.273 ±0.088	0.4 ±0.014	0.873 ±0.078	372.6 ±118.855	469.483 ±121.855
2-way, LS Dual, Multv.	Retail	0.09 ±0.008	0.259 ±0.026	0.391 ±0.05	0.431 ±0.073	0.225 ±0.047	1.396 ±0.203	0.413 ±0.01	0.983 ±0.194	434415541.333 ±37518140.035	434420117.333 ±37518610.316
2-way No Side Info	Retail	0.033 ±0.016	0.037 ±0.021	0.076 ±0.021	0.075 ±0.016	0.032 ±0.006	0.252 ±0.048	0.536 ±0.003	−0.284 ±0.05	422.562 ±66.308	451.334 ±63.625

We ideally want $\varXi_\alpha$ and $\varXi$ to be as close to zero as possible coupled with a high R^2



(a) Alcohol Sales **(b)** Movielens-20m **(c)** Retail Sales

Fig. 4 Heatmap of $\xi_{1-2\alpha}$ for all datasets. Here the y-axis is location/userID and x-axis is time, where targets have been aggregated on items. We see that the calibration rate over all aggregates consistently adjusts with changes in $1 - 2\alpha$

We compare KFTR to existing Bayesian models with reported results in Table 8, where the model is optimized for η_{criteria}, in contrast to RMSE used in Kim and Choi (2014). KFT performs well in a Bayesian setting compared to Kim and Choi (2014) and also yield calibrated estimates. For forecast problems, we apply KFTR to the Traffic and CCDS dataset and report the results in Table 9. We find that the WLR version of KFT does well and yields calibrated forecasts.

We plot forecasts for the Traffic dataset with uncertainty quantification in Fig. 5.

5 Analysis

We demonstrated the practical utility of KFT in both a frequentist and Bayesian context. We now scrutinize the robustness and effectiveness of KFT as a remedy for constraint-imposing side information.

5.1 Does KFT really amend the constraints of directly apply side information?

To validate this, we train KFT-WLR and a naive model on the Alcohol Sales dataset using kernelized side information for 5 hyperparameter searches. We plot the mean training, validation, and test error of the 5 searches (and the standard errors) against epochs in Fig. 6.

5.2 How does KFT perform when applying constant side information?

To answer this question, we replace all side information with a constant **1** and kernelize it. The results in the first row of Table 10 indicate that KFT indeed is robust towards constant side information, as the performance does not degrade dramatically.

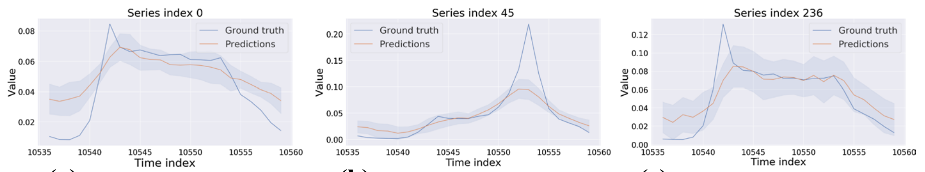
Table 8 Results with RMSE and (Ξ)

Method	Movielens-1M (64/16/20%)	Movielens-1M (85.5/4.5/10%)	Movielens-10M (64/16/20%)	Movielens-10M (85.5/4.5/10%)
VBMF (Kim and Choi 2014)				0.8444[a]
KFT, Univariate, WLR, Dual	0.92 ± 0.008 (0.48 ± 0.023)	0.891 ± 0.006 (0.555 ± 0.046)	0.848 ± 0.012 (0.425 ± 0.018)	0.834 ± 0.007 (0.45 ± 0.021)
KFT, Multivariate, WLR, Dual	0.893 ± 0.002 (0.231 ± 0.02)	0.876 ± 0.013 (0.356 ± 0.181)	0.877 ± 0.009 (0.304 ± 0.084)	0.864 ± 0.004 (0.201 ± 0.092)

[a]As no explicit train/valid/test ratio were provided, the comparison might not be objective

Table 9 Table of results for Bayesian forecast experiments

Model	Dataset	$\Xi_{0.05}$	$\Xi_{0.15}$	$\Xi_{0.25}$	$\Xi_{0.35}$	$\Xi_{0.45}$	Ξ	R^2	η_{criteria}	ELBO	log-likelihood
P-way, WLR Dual, Univ.	Traffic	0.035 ±0.013	0.133 ±0.055	0.147 ±0.078	0.097 ±0.059	0.032 ±0.019	0.444 ±0.221	0.671 ±0.11	−0.227 ±0.28	629.04 ±6.378	629.861 ±6.176
P-way, LS Dual, Univ.	Traffic	0.1 ±0.0	0.3 ±0.0	0.499 ±0.001	0.68 ±0.01	0.489 ±0.046	2.068 ±0.057	0.213 ±0.081	1.855 ±0.096	52845.821 ±47567.564	1031271.526 ±904233.765
P-way, WLR Dual, Univ.	CCDS	0.042 ±0.021	0.109 ±0.05	0.125 ±0.061	0.094 ±0.05	0.036 ±0.021	0.406 ±0.203	0.196 ±0.011	0.21 ±0.196	20591.652 ±6209.691	20614.386 ±6229.005
P-way, LS Dual, Univ.	CCDS	0.1 ±0.0	0.3 ±0.0	0.498 ±0.003	0.649 ±0.079	0.431 ±0.181	1.978 ±0.263	0.103 ±0.005	1.875 ±0.263	14691.796 ±3708.965	66322.241 ±46105.914



(a) Fold 3, Series index 0 **(b)** Fold 3, Series index 45 **(c)** Fold 3, Series index 236

Fig. 5 Forecasts on the Traffic dataset using Bayesian KFTR.

5.3 How does KFT perform when applying noise as side information?

Similar to the previous question, we now replace the side information with standard Gaussian noise instead. The results in the last row of Table 10 indicate that KFT also is robust against noise and surprisingly performant as well. A possible explanation for this is that adding Gaussian noise serves as an implicit regularizer or that the original side information is similarly distributed as standard Gaussian noise. We conclude that KFT is stable against uninformative side information in the form of Gaussian noise.

6 Conclusion

We identified an inherent limitation of side information based tensor regression and gave a method that removes this limitation. Our proposed KFT method yields competitive performance against state-of-the-art large-scale prediction models on a fixed computational budget. Specifically, as the experiments in Table 3 demonstrate, for at least some cases of real practical interest, Weighted Latent Regression is the most performant configuration. Further, KFT offers extended versatility in terms of calibrated Bayesian variational estimates. Our analysis shows that KFT solves the problems we described in Sect. 2 and is robust for adversarial side information in the form of Gaussian noise. A direction for further development would be to characterize identifiability conditions for KFT and extend the Bayesian framework beyond mean-field variational inference.

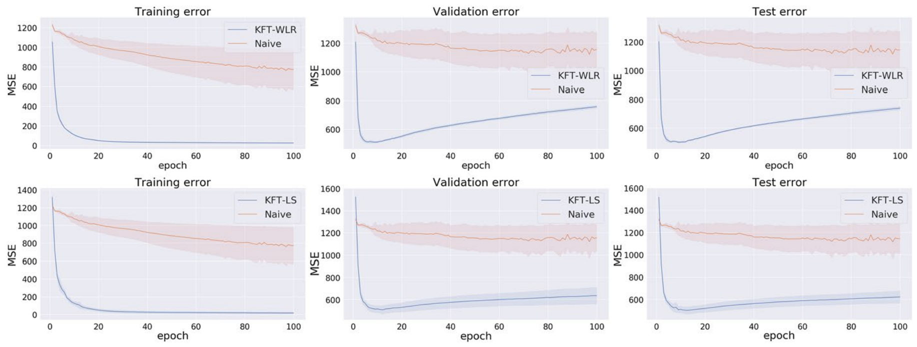


Fig. 6 Training, validation and test error against epoch for KFT-WLR/KFT-LS and a naive model (with "Dual" side information) on the Alcohol Sales dataset

Table 10 Results from additional experiments for P-way dual models

Method/dataset	Retail sales	Movielens-20M	Alcohol sales
WLR, Constant side information, Dual	0.56 ±0.013	0.33 ± 0.039	0.605 ± 0.041
WLR, Noise side information, Dual	0.553 ± 0.006	0.388 ± 0.003	0.704 ± 0.01

Appendix A: $\mathbb{E}_q[\log p(y_{i_1,\dots,i_P}|\mathbf{V}_1,\dots,\mathbf{V}_P,\mathbf{V}'_1,\dots,\mathbf{V}'_P)]$ derivations

A.1 Weighted latent regression

Assuming that σ_y is a constant hyperparameter. We first have that

$$
\begin{aligned}
&\mathbb{E}_q[\log p(y_{i_1,\dots,i_P}|\mathbf{V}_1,\dots,\mathbf{V}_P,\mathbf{V}'_1,\dots,\mathbf{V}'_P)] \\
&\propto \frac{1}{\sigma_y^2}\mathbb{E}_q\left[(y_{i_1,\dots,i_P} - \sum_{\substack{r_1,\dots,r_P\\ i'_1,\dots,i'_P\\ i''_1,\dots,i''_P}} \prod_{p=1}^{P} v'_{r_p i''_p r_{p-1}} v_{r_p i'_p r_{p-1}} \underbrace{k(\mathbf{x}^p_{i_p},\mathbf{x}^p_{i'_p})\delta_{\mathbf{x}^p_{i_p}}(\mathbf{x}^p_{i''_p})}_{f(k,\delta)})^2\right] \\
&= \frac{1}{\sigma_y^2}\Bigg[y^2_{i_1,\dots,i_P} - 2y_{i_1,\dots,i_P}\cdot \sum_{\substack{r_1,\dots,r_P\\ i'_1,\dots,i'_P\\ i''_1,\dots,i''_P}} \prod_{p=1}^{P}\mathbb{E}_q[v'_{r_p i''_p r_{p-1}}]\mathbb{E}_q[v_{r_p i'_p r_{p-1}}]f(k,\delta) \\
&+ \sum_{\substack{r_1,\dots,r_P\\ i'_1,\dots,i'_P\\ i''_1,\dots,i''_P\\ q_1,\dots,q_P\\ j'_1,\dots,j'_P\\ j''_1,\dots,j''_P}} \prod_{p=1}^{P}\mathbb{E}_q[v'_{q_p j''_p q_{p-1}} v'_{r_p i''_p r_{p-1}}]\mathbb{E}_q[v_{r_p i'_p r_{p-1}} v_{q_p j'_p q_{p-1}}]k(\mathbf{x}^p_{i_p},\mathbf{x}^p_{i'_p}) \\
&\delta_{\mathbf{x}^p_{i_p}}(\mathbf{x}^p_{i''_p})k(\mathbf{x}^p_{i_p},\mathbf{x}^p_{j'_p})\delta_{\mathbf{x}^p_{i_p}}(\mathbf{x}^p_{j''_p})\Bigg]
\end{aligned}
\tag{56}
$$

Our goal is to find the corresponding tensor operations of the sum terms.

A.1.1 Univariate meanfield

We first have that

$$
\mathbb{E}_q[v_{r_p i'_p r_{p-1}} v_{q_p j'_p q_{p-1}}] = \mu_{r_p i'_p r_{p-1}}\mu_{q_p j'_p q_{p-1}} + \delta_{r_p}(q_p)\delta_{r_{p-1}}(q_{p-1})\delta_{i'_p}(j'_p)\sigma^2_{r_p i'_p r_{p-1}} \tag{57}
$$

Then the last term is can be written as

$$\begin{aligned}
\Longrightarrow \sum_{\substack{r_1,\dots,r_P\\ i'_1,\dots,i'_P\\ q_1,\dots,q_P\\ j'_1,\dots,j'_P}} \prod_{p=1}^{P} \mathbb{E}_q[v'_{q_p i_p q_{p-1}} v'_{r_p i_p r_{p-1}}]\mathbb{E}_q[v_{r_p i'_p r_{p-1}} v_{q_p j'_p q_{p-1}}]k(\mathbf{x}^p_{i_p},\mathbf{x}^p_{i'_p})k(\mathbf{x}^p_{i_p},\mathbf{x}^p_{j'_p})\\
&= \sum\prod_{p=1}^{P}(\mu'_{q_p i_p q_{p-1}}\mu'_{r_p i_p r_{p-1}} + \delta_{r_p}(q_p)\delta_{r_{p-1}}(q_{p-1})\sigma'^2_{r_p i_p r_{p-1}})\\
&\quad (\mu_{q_p j'_p q_{p-1}}\mu_{r_p i'_p r_{p-1}} + \delta_{r_p}(q_p)\delta_{r_{p-1}}(q_{p-1})\delta_{i'_p}(j'_p)\sigma^2_{r_p i'_p r_{p-1}})k(\mathbf{x}^p_{i_p},\mathbf{x}^p_{i'_p})k(\mathbf{x}^p_{i_p},\mathbf{x}^p_{j'_p})\\
&= \sum\prod_{p=1}^{P}\mu'_{q_p i_p q_{p-1}}\mu'_{r_p i_p r_{p-1}}\mu_{q_p j'_p q_{p-1}}\mu_{r_p i'_p r_{p-1}}k(\mathbf{x}^p_{i_p},\mathbf{x}^p_{i'_p})k(\mathbf{x}^p_{i_p},\mathbf{x}^p_{j'_p})\\
&\quad + \sum_{\substack{r_1,\dots,r_P\\ i'_1,\dots,i'_P}}\prod_{p=1}^{P}\sigma'^2_{r_p i_p r_{p-1}}\sigma^2_{r_p i'_p r_{p-1}}k(\mathbf{x}^p_{i_p},\mathbf{x}^p_{i'_p})^2\\
&\quad + \sum_{\substack{r_1,\dots,r_P\\ i'_1,\dots,i'_P\\ j'_1,\dots,j'_P}}\prod\sigma'^2_{r_p i_p r_{p-1}}\mu_{r_p i'_p r_{p-1}}\mu_{r_p j'_p r_{p-1}}k(\mathbf{x}^p_{i_p},\mathbf{x}^p_{i'_p})k(\mathbf{x}^p_{i_p},\mathbf{x}^p_{j'_p})\\
&\quad + \sum_{\substack{r_1,\dots,r_P\\ i'_1,\dots,i'_P}}\prod\sigma^2_{r_p i'_p r_{p-1}}\mu'^2_{r_p i_p r_{p-1}}k(\mathbf{x}^p_{i_p},\mathbf{x}^p_{i'_p})^2
\end{aligned} \tag{58}$$

$$\begin{aligned}
\Longleftrightarrow \left(\prod_p^P \times_{-1}\mathbf{M}'_p\circ(\mathbf{M}_p\times_2\mathbf{K}_p)\right)^2 + \left(\prod_p^P\times_{-1}\Sigma'_p\circ(\Sigma_p\times_2(\mathbf{K}_p)^2)\right)\\
+\left(\prod_p^P\times_{-1}(\Sigma'_p\circ(\mathbf{M}_p\times_2\mathbf{K}_p)^2)\right) + \left(\prod_p^P\times_{-1}\mathbf{M}'^2_p\circ(\Sigma_p\times_2(\mathbf{K}_p)^2)\right)
\end{aligned} \tag{59}$$

where $\mathbf{M}'_p, \mathbf{M}$ and Σ', Σ correspond to the tensors containing the variational parameters $\mu'_{r_p i_p r_{p-1}}$, $\mu_{r_p i_p r_{p-1}}$ and $\sigma'^2_{r_p i_p r_{p-1}}$, $\sigma^2_{r_p i_p r_{p-1}}$ respectively. For the case of RFF's in dual space, we approximate $\Sigma_p\times_2(\mathbf{K}_p)^2 \approx \Sigma_p\times_2(\Phi_p\bullet\Phi_p)^\top\times_2(\Phi_p\bullet\Phi_p)$, where $\bullet$ is the transposed Khatri-Rao product. It should further be noted that any square term, means elementwise squaring. With the middle term given by

$$2\mathbf{Y}\circ\left(\prod_{p=1}^{P}\times_{-1}(\mathbf{M}'_p\circ(\mathbf{M}_p\times_2\mathbf{K}_p))\right) \tag{60}$$

the full reconstruction term is expressed as:

$$
\begin{aligned}
&\mathbb{E}_q[\log p(y_{i_1,\dots,i_P}|\mathbf{V}_1,\dots,\mathbf{V}_p,\mathbf{V}'_1,\dots,\mathbf{V}'_p)] \\
&\propto \frac{1}{\sigma_y^2}\Bigg[\mathbf{Y}^2 - 2\mathbf{Y}\circ\Bigg(\prod_{p=1}^{P}\times_{-1}(\mathbf{M}'_p\circ(\mathbf{M}_p\times_2\mathbf{K}_p))\Bigg) \\
&+\Bigg(\prod_{p}^{P}\times_{-1}\mathbf{M}'_p\circ(\mathbf{M}_p\times_2\mathbf{K}_p)\Bigg)^2 + \Bigg(\prod_{p}^{P}\times_{-1}\Sigma'_p\circ(\Sigma_p\times_2(\mathbf{K}_p)^2)\Bigg) \\
&+\Bigg(\prod_{p}^{P}\times_{-1}(\Sigma'_p\circ((\mathbf{M}_p\times_2\mathbf{K}_p)^2)\Bigg) + \Bigg(\prod_{p}^{P}\times_{-1}\mathbf{M}'^2_p\circ(\Sigma_p\times_2(\mathbf{K}_p)^2)\Bigg)\Bigg]
\end{aligned}
\tag{61}
$$

A.1.2 Multivariate meanfield

Similarly first observe that

$$
\mathbb{E}_q[v_{r_p i'_p r_{p-1}} v_{q_p j'_p q_{p-1}}] = \mu_{r_p i'_p r_{p-1}}\mu_{q_p j'_p q_{p-1}} + \delta_{r_p}(q_p)\delta_{r_{p-1}}(q_{p-1})\sigma_{i'_p}\sigma_{j'_p}. \tag{62}
$$

It then follows that the third term is calculated as

$$
\begin{aligned}
&\sum\prod_{p=1}^{P}(\mu'_{q_p i_p q_{p-1}}\mu'_{r_p i_p r_{p-1}} + \delta_{r_p}(q_p)\delta_{r_{p-1}}(q_{p-1})\sigma'^2_{r_p i_p r_{p-1}}) \\
&\quad(\mu_{q_p j'_p q_{p-1}}\mu_{r_p i'_p r_{p-1}} + \delta_{r_p}(q_p)\delta_{r_{p-1}}(q_{p-1})\sigma_{i'_p}\sigma_{j'_p})k(\mathbf{x}^p_{i_p},\mathbf{x}^p_{i'_p})k(\mathbf{x}^p_{i_p},\mathbf{x}^p_{j'_p}) \\
&= \sum\prod_{p=1}^{P}\mu'_{q_p i_p q_{p-1}}\mu'_{r_p i_p r_{p-1}}\mu_{q_p j'_p q_{p-1}}\mu_{r_p i'_p r_{p-1}}k(\mathbf{x}^p_{i_p},\mathbf{x}^p_{i'_p})k(\mathbf{x}^p_{i_p},\mathbf{x}^p_{j'_p}) \\
&\quad+\sum_{\substack{r_1,\dots,r_P\\ i'_1,\dots,i'_P\\ j'_1,\dots,j'_P}}\prod_{p=1}^{P}\sigma'^2_{r_p i_p r_{p-1}}\sigma_{i'_p}\sigma_{j'_p}k(\mathbf{x}^p_{i_p},\mathbf{x}^p_{i'_p})k(\mathbf{x}^p_{i_p},\mathbf{x}^p_{j'_p}) \\
&\quad+\sum_{\substack{r_1,\dots,r_P\\ i'_1,\dots,i'_P\\ j'_1,\dots,j'_P}}\prod_{p=1}^{P}\sigma'^2_{r_p i_p r_{p-1}}\mu_{r_p i'_p r_{p-1}}\mu_{r_p j'_p r_{p-1}}k(\mathbf{x}^p_{i_p},\mathbf{x}^p_{i'_p})k(\mathbf{x}^p_{i_p},\mathbf{x}^p_{j'_p}) \\
&\quad+\sum_{\substack{r_1,\dots,r_P\\ i'_1,\dots,i'_P\\ j'_1,\dots,j'_P}}\prod_{p=1}^{P}\sigma_{i'_p}\sigma_{j'_p}\mu'^2_{r_p i_p r_{p-1}}k(\mathbf{x}^p_{i_p},\mathbf{x}^p_{i'_p})k(\mathbf{x}^p_{i_p},\mathbf{x}^p_{j'_p})
\end{aligned}
\tag{63}
$$

$$\iff \left(\prod_p^P \times_{-1} \mathbf{M}'_p \circ (\mathbf{M}_p \times_2 \mathbf{K}_p)\right)^2 \left(\prod_p^P \times_{-1} \Sigma'_p \circ (\mathbf{1} \times_2 \left(\mathrm{diag}((\mathbf{K}_p \cdot \mathbf{B}_p)^2 \cdot \bar{1})\right)\right)$$
$$+ \left(\prod_p^P \times_{-1} (\Sigma'_p \circ (\mathbf{M}_p \times_2 \mathbf{K}_p)^2)\right) + \left(\prod_p^P \times_{-1} \mathbf{M}'^2_p \circ (\mathbf{1} \times_2 \left(\mathrm{diag}((\mathbf{K}_p \cdot \mathbf{B}_p)^2 \cdot \bar{1})\right)\right) \tag{64}$$

where $\Sigma_p = \mathbf{B}_p \mathbf{B}_p^T$, $\mathbf{1}$ denotes a constant one tensor with the same dimensions as Σ'_p, $\bar{1} \in \mathbb{R}^{R\times 1}$ where R is the column dimension of $\mathbf{B}_p$ and Σ'_p is the same as in the univariate case. For RFF's we have that

$$\begin{aligned}(\mathbf{K}_p \cdot \mathbf{B}_p)^2 \cdot \bar{1} \approx& ((\boldsymbol{\Phi}_p \cdot \boldsymbol{\Phi}_p^\top) \cdot \mathbf{B}_p)^2 \cdot \bar{1} + \mathrm{vec}(D_p^2) \\ =& (\boldsymbol{\Phi}_p \cdot (\boldsymbol{\Phi}_p^\top \cdot \mathbf{B}_p))^2 \cdot \bar{1} + \mathrm{vec}(D_p^2).\end{aligned} \tag{65}$$

As the middle term remains the same as in the univariate case our full reconstruction term is

$$\begin{aligned}&\mathbb{E}_q[\log p(y_{i_1,\dots,i_P} | \mathbf{V}_1, \dots, \mathbf{V}_p, \mathbf{V}'_1, \dots, \mathbf{V}'_p)] \\ &\propto \frac{1}{\sigma_y^2}\Bigg[\mathbf{Y}^2 - 2\mathbf{Y}\circ\left(\prod_{p=1}^P \times_{-1} (\mathbf{M}'_p \circ (\mathbf{M}_p \times_2 \mathbf{K}_p))\right) \\ &+ \left(\prod_p^P \times_{-1} \mathbf{M}'_p \circ (\mathbf{M}_p \times_2 \mathbf{K}_p)\right)^2 + \left(\prod_p^P \times_{-1} \Sigma'_p \circ (\mathbf{1} \times_2 \left(\mathrm{diag}((\mathbf{K}_p \cdot \mathbf{B}_p)^2 \cdot \bar{1})\right)\right) \\ &+ \left(\prod_p^P \times_{-1} (\Sigma'_p \circ (\mathbf{M}_p \times_2 \mathbf{K}_p)^2)\right) + \left(\prod_p^P \times_{-1} \mathbf{M}'^2_p \circ (\mathbf{1} \times_2 \left(\mathrm{diag}((\mathbf{K}_p \cdot \mathbf{B}_p)^2 \cdot \bar{1})\right)\right)\Bigg]\end{aligned} \tag{66}$$

A.2 Latent scaling

Assuming that σ_y is a constant hyperparameter. We first have that

$$
\begin{aligned}
&\mathbb{E}_q[\log p(y_{i_1,\ldots,i_P}|\mathbf{V}_1,\ldots,\mathbf{V}_P,\mathbf{V}'_1,\ldots,\mathbf{V}'_P)] \\
&\propto \frac{1}{\sigma_y^2}\mathbb{E}_q\left[\left(y_{i_1,\ldots,i_P} - \left(\sum_{s_1,\ldots,s_P}\prod_{p=1}^{P} v^s_{s_p i_p s_{p-1}} \sum_{\substack{r_1,\ldots,r_P\\ i'_1,\ldots,i'_P}} \prod_{p=1}^{P} v_{r_p i'_p r_{p-1}} k(\mathbf{x}^p_{i_p},\mathbf{x}^p_{i'_p}) \right.\right.\right.\\
&\quad \left.\left.\left. + \sum_{b_1,\ldots,b_P}\prod_{p=1}^{P} v^b_{b_p i_p b_{p-1}} \right)\right)^2\right] \\
&= \frac{1}{\sigma_y^2}\left[y^2_{i_1,\ldots,i_P} - 2y_{i_1,\ldots,i_P} \right.\\
&\quad \cdot \left(\underbrace{\sum_{s_1,\ldots,s_P}\prod_{p=1}^{P}\mathbb{E}_q[v^s_{s_p i_p s_{p-1}}]}_{f_{v^s_{s_p i_p s_{p-1}}}} \underbrace{\sum_{\substack{r_1,\ldots,r_P\\ i'_1,\ldots,i'_P}}\prod_{p=1}^{P}\mathbb{E}_q[v_{r_p i'_p r_{p-1}}]k(\mathbf{x}^p_{i_p},\mathbf{x}^p_{i'_p})}_{f_{v_{r_p i'_p r_{p-1}}}} \right.\\
&\quad \left. + \underbrace{\sum_{b_1,\ldots,b_P}\prod_{p=1}^{P}\mathbb{E}_q[v^b_{b_p i_p b_{p-1}}]}_{f_{v^b_{b_p i_p b_{p-1}}}} \right)\\
&\quad + \sum_{\substack{s_1,\ldots,s_P\\ s'_1,\ldots,s'_P}}\prod_{p=1}^{P}\mathbb{E}_q[v^s_{s_p i_p s_{p-1}} v^s_{s'_p i_p s'_{p-1}}] \sum_{\substack{r_1,\ldots,r_P\\ q_1,\ldots,q_P\\ i'_1,\ldots,i'_P\\ j'_1,\ldots,j'_P}}\prod_{p=1}^{P}\mathbb{E}_q[v_{r_p i'_p r_{p-1}} v_{q_p j'_p q_{p-1}}]k(\mathbf{x}^p_{i_p},\mathbf{x}^p_{i'_p})k(\mathbf{x}^p_{i_p},\mathbf{x}^p_{j'_p})\\
&\quad \left. + 2f_{v^s_{s_p i_p s_{p-1}}} f_{v_{r_p i'_p r_{p-1}}} f_{v^b_{b_p i_p b_{p-1}}} + \sum_{\substack{b_1,\ldots,b_P\\ b'_1,\ldots,b'_P}}\prod_{p=1}^{P}\mathbb{E}_q[v^b_{b_p i_p b_{p-1}} v^b_{b'_p i_p b'_{p-1}}] \right].
\end{aligned}
\tag{67}
$$

A.2.1 Univariate case

For the univariate case, the squared component of the reconstruction term in the ELBO becomes

$$\left(\left(\prod_{p=1}^{P}\times_{-1}\mathbf{M}_p^s\right)^2+\left(\prod_{p=1}^{P}\times_{-1}\Sigma_p^s\right)\right)\circ\left(\left(\prod_{p}^{P}\times_{-1}\mathbf{M}_p\times_2\mathbf{K}_p\right)^2+\prod_{p=1}^{P}\times_{-1}\Sigma_p\times_2(\mathbf{K}_p)^2\right)$$
$$+2\left(\prod_{p=1}^{P}\times_{-1}\mathbf{M}_p^s\right)\circ\left(\prod_{p=1}^{P}\times_{-1}\mathbf{M}_p^b\right)\circ\left(\prod_{p=1}^{P}\times_{-1}\mathbf{M}_p\times_2\mathbf{K}_p\right)$$
$$+\left(\left(\prod_{p=1}^{P}\times_{-1}\mathbf{M}_p^b\right)^2+\left(\prod_{p=1}^{P}\times_{-1}\Sigma_p^b\right)\right). \tag{68}$$

Too see this, we notice that all of the sums from the weighted latent regression case reappears here as well. The entire expression is given by

$$\begin{aligned}
&\mathbb{E}_q[\log p(y_{i_1,\dots,i_P}|\mathbf{V}_1,\dots,\mathbf{V}_P,\mathbf{V}'_1,\dots,\mathbf{V}'_P)]\\
&\propto\frac{1}{\sigma_y^2}\Bigg[\mathbf{Y}^2-2\mathbf{Y}\circ\left(\prod_{p=1}^{P}(\mathbf{M}_p^s\circ(\mathbf{M}_p\times_2\mathbf{K}_p)+\mathbf{M}_p^b)\right)\\
&+\left(\left(\prod_{p=1}^{P}\times_{-1}\mathbf{M}_p^s\right)^2+\left(\prod_{p=1}^{P}\times_{-1}\Sigma_p^s\right)\right)\\
&\circ\left(\left(\prod_{p}^{P}\times_{-1}\mathbf{M}_p\times_2\mathbf{K}_p\right)^2+\prod_{p=1}^{P}\times_{-1}\Sigma_p\times_2(\mathbf{K}_p)^2\right)\\
&+2\left(\prod_{p=1}^{P}\times_{-1}\mathbf{M}_p^s\right)\circ\left(\prod_{p=1}^{P}\times_{-1}\mathbf{M}_p^b\right)\\
&\circ\left(\prod_{p=1}^{P}\times_{-1}\mathbf{M}_p\times_2\mathbf{K}_p\right)+\left(\left(\prod_{p=1}^{P}\times_{-1}\mathbf{M}_p^b\right)^2+\left(\prod_{p=1}^{P}\times_{-1}\Sigma_p^b\right)\right)\Bigg].
\end{aligned}\tag{69}$$

A.2.2 Multivariate case

For multivariate case the last term becomes

$$
\begin{aligned}
&\left(\left(\prod_{p=1}^{P}\times_{-1}\mathbf{M}_p^s\right)^2+\left(\prod_{p=1}^{P}\times_{-1}\Sigma_p^s\right)\right)\\
&\quad\circ\left(\left(\prod_{p}^{P}\times_{-1}\mathbf{M}_p\times_2\mathbf{K}_p\right)^2+\prod_{p=1}^{P}\times_{-1}(\mathbf{1}\times_2\left(\mathrm{diag}((\mathbf{K}_p\cdot\mathbf{B}_p)^2\cdot\bar{1})\right))\right)\\
&\quad+2\left(\prod_{p=1}^{P}\times_{-1}\mathbf{M}_p^s\right)\circ\left(\prod_{p=1}^{P}\times_{-1}\mathbf{M}_p^b\right)\\
&\quad\circ\left(\prod_{p=1}^{P}\times_{-1}\mathbf{M}_p\times_2\mathbf{K}_p\right)+\left(\left(\prod_{p=1}^{P}\times_{-1}\mathbf{M}_p^b\right)^2+\left(\prod_{p=1}^{P}\times_{-1}\Sigma_p^b\right)\right)
\end{aligned}
\tag{70}
$$

Again, we notice that all of the sums from the weighted latent regression case reappears here as well. The full expression is given by

$$
\begin{aligned}
&\mathbb{E}_q[\log p(y_{i_1,\dots,i_P}|\mathbf{V}_1,\dots,\mathbf{V}_p,\mathbf{V}'_1,\dots,\mathbf{V}'_p)]\\
&\propto\frac{1}{\sigma_y^2}\left[\mathbf{Y}^2-2\mathbf{Y}\circ\left(\prod_{p=1}^{P}(\mathbf{M}_p^s\circ(\mathbf{M}_p\times_2\mathbf{K}_p)+\mathbf{M}_p^b)\right)\right.\\
&\quad+\left(\left(\prod_{p=1}^{P}\times_{-1}\mathbf{M}_p^s\right)^2+\left(\prod_{p=1}^{P}\times_{-1}\Sigma_p^s\right)\right)\\
&\quad\circ\left(\left(\prod_{p}^{P}\times_{-1}\mathbf{M}_p\times_2\mathbf{K}_p\right)^2+\prod_{p=1}^{P}\times_{-1}(\mathbf{1}\times_2\left(\mathrm{diag}((\mathbf{K}_p\cdot\mathbf{B}_p)^2\cdot\bar{1})\right))\right)\\
&\quad+2\left(\prod_{p=1}^{P}\times_{-1}\mathbf{M}_p^s\right)\circ\left(\prod_{p=1}^{P}\times_{-1}\mathbf{M}_p^b\right)\circ\left(\prod_{p=1}^{P}\times_{-1}\mathbf{M}_p\times_2\mathbf{K}_p\right)\\
&\quad\left.+\left(\left(\prod_{p=1}^{P}\times_{-1}\mathbf{M}_p^b\right)^2+\left(\prod_{p=1}^{P}\times_{-1}\Sigma_p^b\right)\right)\right]
\end{aligned}
\tag{71}
$$

A.3 Forecasting

We derive an expression for the forecasting term for the WLR case

$$\begin{aligned}
&\mathbb{E}_q[\log p(\mathbf{X}_T \mid \mathbf{X}_{l_1}, \dots, \mathbf{X}_{l_K})] \\
&\quad \propto \frac{1}{\sigma^2_{\text{TRMF}}}\mathbb{E}_q\left[\left(x_{r_p,i_t=T,r_{p+1}} - \sum_{k=1}^{K} w_k x_{r_p,i_t=l_k,r_{p+1}}\right)^2\right] \\
&\quad = \frac{1}{\sigma^2_{\text{TRMF}}}\mathbb{E}_q\left[\left(x^2_{r_p,i_t=T,r_{p+1}} - 2x_{r_p,i_t=T,r_{p+1}}\sum_{k=1}^{K} w_k x_{r_p,i_t=l_k,r_{p+1}}\right.\right. \\
&\qquad \left.\left. +(\sum_{k=1}^{K} w_k x_{r_p,i_t=l_k,r_{p+1}})^2\right)\right].
\end{aligned} \tag{72}$$

We first establish that

$$x_{r_p,i_t=\tau,r_{p+1}} = \sum_{\substack{t'=1\dots T\\ t''=\tau}} v'_{r_p t'' r_{p-1}} v_{r_p t' r_{p-1}} k(\mathbf{x}_t, \mathbf{x}_{t'})\delta_{\mathbf{x}_t}(\mathbf{x}_{t''}). \tag{73}$$

Consequently,

$$\begin{aligned}
\mathbb{E}_q[x_{r_p,i_t=\tau,r_{p+1}}] &= \sum_{\substack{t'=1\dots T\\ t''=\tau}} \mathbb{E}_q[v'_{r_p t'' r_{p-1}}]\mathbb{E}_q[v_{r_p t' r_{p-1}}]k(\mathbf{x}_t, \mathbf{x}_{t'})\delta_{\mathbf{x}_t}(\mathbf{x}_{t''}) \\
&= \sum_{\substack{t'=1\dots T\\ t''=\tau}} \mu'_{r_p t'' r_{p-1}}\mu_{r_p t' r_{p-1}} k(\mathbf{x}_t, \mathbf{x}_{t'})\delta_{\mathbf{x}_t}(\mathbf{x}_{t''}) \\
&= \left[\mathbf{M}' \circ (\mathbf{M} \times_2 \mathbf{K})\right]_\tau.
\end{aligned} \tag{74}$$

where $[\cdot]_\tau$ denotes slicing in the temporal dimension at index τ. We then calculate the first term

$$\begin{aligned}
x^2_{r_p,i_t=\tau,r_{p+1}} &= \sum_{\substack{t'=1\dots T\\ t''=\tau\\ z'=1\dots T\\ z''=\tau}} v'_{r_p t'' r_{p-1}} v'_{r_p z'' r_{p-1}} v_{r_p t' r_{p-1}} k(\mathbf{x}_t, \mathbf{x}_{t'})\delta_{\mathbf{x}_t}(\mathbf{x}_{t''})k(\mathbf{x}_t, \mathbf{x}_{z'})\delta_{\mathbf{x}_t}(\mathbf{x}_{z''}) \\
&= v'^2_{r_p \tau r_{p-1}} \sum_{\substack{t'=1\dots T\\ z'=1\dots T}} v_{r_p z' r_{p-1}} v_{r_p t' r_{p-1}} k(\mathbf{x}_t, \mathbf{x}_{t'})k(\mathbf{x}_t, \mathbf{x}_{z'}).
\end{aligned} \tag{75}$$

For univariate meanfield $\mathbf{V}, \mathbf{V}'$, taking the expectation we arrive at

$$\mathbb{E}_q[x^2_{r_p,i_t=\tau,r_{p+1}}] = \mathbb{E}_q[v'^2_{r_p\tau r_{p-1}}] \sum_{\substack{t'=1\dots T\\ z'=1\dots T}} \mathbb{E}_q[v_{r_p z' r_{p-1}} v_{r_p t' r_{p-1}}] k(\mathbf{x}_t,\mathbf{x}_{t'})k(\mathbf{x}_t,\mathbf{x}_{z'})$$
$$= (\mu'^2_{r_p\tau r_{p-1}} + \sigma'^2_{r_p\tau r_{p-1}}) \left[\sum_{\substack{t'=1\dots T\\ z'=1\dots T}} \mu_{r_p z' r_{p-1}} \mu_{r_p t' r_{p-1}} k(\mathbf{x}_t,\mathbf{x}_{t'})k(\mathbf{x}_t,\mathbf{x}_{z'}) + \sum_{t'=1} \sigma^2_{r_p t' r_{p-1}} k(\mathbf{x}_t,\mathbf{x}_{t'})^2 \right] \tag{76}$$
$$= \left[(\mathbf{M}'^2 + \Sigma')\circ\left[(\mathbf{M}\times_2\mathbf{K})^2 + \Sigma\times_2\mathbf{K}^2\right]\right]_\tau$$

If we instead consider a multivariate meanfield $\mathbf{V},\mathbf{V}'$, the expression becomes

$$\mathbb{E}_q[x^2_{r_p,i_t=\tau,r_{p+1}}]$$
$$= (\mu'^2_{r_p\tau r_{p-1}} + \sigma'^2_{r_p\tau r_{p-1}}) \left[\sum_{\substack{t'=1\dots T\\ z'=1\dots T}} (\mu_{r_p z' r_{p-1}} \mu_{r_p t' r_{p-1}} + \sigma_{z'}\sigma_{t'}) k(\mathbf{x}_t,\mathbf{x}_{t'})k(\mathbf{x}_t,\mathbf{x}_{z'}) \right] \tag{77}$$
$$= \left[(\mathbf{M}'^2 + \Sigma')\circ\left[(\mathbf{M}\times_2\mathbf{K})^2 + (\mathbf{1}\times_2\left(\mathrm{diag}((\mathbf{K}\cdot\mathbf{B})^2\cdot\bar{1})\right)\right]\right]_\tau$$

For the third expression, we take

$$\mathbb{E}_q[(\sum_{k=1}^K w_k x_{r_p,i_t=l_k,r_{p+1}})^2] = \sum_{f=1}^K \sum_{k=1}^K w_k w_f \mathbb{E}_q[x_{r_p,i_t=l_k,r_{p+1}} x_{r_p,i_t=l_f,r_{p+1}}]$$
$$= \underbrace{\sum_{k=1}^K w_k^2 \mathbb{E}_q[x^2_{r_p,i_t=l_k,r_{p+1}}]}_{:=\text{Expression 1}} + \underbrace{\sum_{k\neq f}^K w_k w_f \mathbb{E}_q[x_{r_p,i_t=l_k,r_{p+1}} x_{r_p,i_t=l_f,r_{p+1}}]}_{:=\text{Expression 2}}. \tag{78}$$

The tensor expression for Expression 1 is given by

$$\text{Expression 1} = \sum_{k=1}^K \mathbf{W}_k^2\circ\left[(\mathbf{M}'^2 + \Sigma')\circ\left[(\mathbf{M}\times_2\mathbf{K})^2 + \Sigma\times_2\mathbf{K}^2\right]\right]_{l_k}. \tag{79}$$

This follows directly from equation 76. Expression 2 can be written as

$$\text{Expression 2} = \left(\sum_{k=1}^K \mathbf{W}_k\circ\left[\mathbf{M}'\circ(\mathbf{M}\times_2\mathbf{K})\right]_{l_k}\right)^2 - \sum_{k=1}^K \mathbf{W}_k^2\circ\left[\mathbf{M}'\circ(\mathbf{M}\times_2\mathbf{K})\right]^2_{l_k}. \tag{80}$$

Here the idea is to reduce the memory footprint by calculating a slightly erroneous expression and removing the error rather than calculate the exact expression due to the difficulty of expressing the cross terms $w_k w_f$. The above derivation extends directly to the LS case by simply removing the $\mathbf{V}'$ variables and calculating the forecasting term for each component $\mathbf{V}_s, \mathbf{V}, \mathbf{V}_b$.

Appendix B: Derivation for WLR regularization term

$$\begin{aligned}
\sum_{r_1 r_2} \|\mathsf{v}_r\|^2_{\mathcal{H}} &= \sum_{r_1 r_2} \langle \mathsf{v}_r, \mathsf{v}_r \rangle_{\mathcal{H}} \\
&= \sum_{r_1 r_2} \sum_{\substack{n_1,\dots,n_q \\ n'_1,\dots,n'_q}} v_{r_1 n_1 \dots n_q r_1} v'_{r_1 n'_1 \dots n'_q r_2} \prod_{q=1}^{Q} \delta_{\mathbf{x}^Q_{n'_q}}(\cdot) \prod_{q=1}^{Q} k(\cdot, \mathbf{x}^Q_{n_q}) \\
&\quad \sum_{\substack{\bar{n},\dots,\bar{n}_q \\ \bar{n}'_1,\dots,\bar{n}'_q}} v_{r_1 \bar{n}_1 \dots \bar{n}_q r_2} v'_{r_1 \bar{n}'_1 \dots \bar{n}'_q r_2} \prod_{q=1}^{Q} \delta_{\mathbf{x}^Q_{\bar{n}'_q}}(\cdot) \prod_{q=1}^{Q} k(\cdot, \mathbf{x}^Q_{\bar{n}_q}) \\
&= \sum_{r_1 r_2} \sum_{\substack{n_1,\dots,n_q \\ n'_1,\dots,n'_q \\ \bar{n}_1,\dots,\bar{n}_q}} {v'_{r_1 n'_1,\dots,n'_q r_2}}^2 v_{r_1 n_1 \dots n_q r_2} v_{r_1 \bar{n}_1 \dots \bar{n}_q r_2} \prod_{q=1}^{Q} k(\mathbf{x}^Q_{n_q}, \mathbf{x}^Q_{\bar{n}_q}) \\
&= \operatorname{tr}\left(\left(\prod_{q=1}^{Q} \otimes \mathbf{K}_q \right) \cdot \mathbf{V}^\top_{R_1 \cdot R_2 \times \prod_{q=1}^{Q} n_q} \cdot \left(\left(\prod_{q=1}^{Q} (\mathbf{V}' \circ \mathbf{V}') \times_q \mathbf{1}_{n_q \times n_q} \right) \circ \mathbf{V} \right)_{R_1 \cdot R_2 \times \prod_{q=1}^{Q} n_q} \right) \\
&= \operatorname{tr}\left(\left(\prod_{q=1}^{Q} \mathbf{V} \times_{q+1} \mathbf{K}_q \right)^\top_{R_1 \cdot R_2 \times \prod_{q=1}^{Q} n_q} \cdot \left(\left(\prod_{q=1}^{Q} (\mathbf{V}' \circ \mathbf{V}') \times_q \mathbf{1}_{n_q \times n_q} \right) \circ \mathbf{V} \right)_{R_1 \cdot R_2 \times \prod_{q=1}^{Q} n_q} \right) \\
&= \left(\left(\prod_{q=1}^{Q} \mathbf{V} \times_{q+1} \mathbf{K}_q \right) \circ \left(\left(\prod_{q=1}^{Q} (\mathbf{V}' \circ \mathbf{V}') \times_q \mathbf{1}_{n_q \times n_q} \right) \circ \mathbf{V} \right) \right)_{++}
\end{aligned} \tag{81}$$

B.1 RFF regularization term

The regularization term can similarly be generalized to

$$\sum_{r_1r_2} \|\mathsf{v}_r\|^2_{\mathcal{H}} = \sum_{r_1r_2} \sum_{\substack{n_1,\dots,n_q \\ n'_1,\dots,n'_q \\ \bar{n}_1,\dots,\bar{n}_q}} {v'_{r_1n'_1,\dots,n'_qr_2}}^2 v_{r_1n_1\dots n_qr_2} v_{r_1\bar{n}_1\dots\bar{n}_qr_2} \prod_{q=1}^{Q} k(\mathbf{x}^Q_{n_q}, \mathbf{x}^Q_{\bar{n}_q})$$

$$\approx \sum_{r_1r_2} \sum_{\substack{n_1,\dots,n_q \\ n'_1,\dots,n'_q \\ \bar{n}_1,\dots,\bar{n}_q}} {v'_{r_1n'_1,\dots,n'_qr_2}}^2 \prod_{q=1}^{Q} (v_{r_1n_1\dots n_qr_2}\phi(\mathbf{x}^Q_{n_q}))^\top (v_{r_1\bar{n}_1\dots\bar{n}_qr_2}\phi(\mathbf{x}^Q_{\bar{n}_q})) \tag{82}$$

$$= \left(\left(\prod_{q=1}^{Q} \mathbf{V} \times_{q+1} \boldsymbol{\Phi}_q \times_q \boldsymbol{\Phi}_q^\top\right) \circ \left(\left(\prod_{q=1}^{Q} (\mathbf{V}' \circ \mathbf{V}') \times_q \mathbf{1}_{n_q \times n_q}\right) \circ \mathbf{V}\right)\right)_{++}$$

Appendix C: Complexity proofs

We recall the theorem presented.

Theorem 2 *KFT has computational complexity*

$$\mathcal{O}\left(\max_p \left(n_p c_p r_p r_{p-1}\right) + \left(\max_p r_p\right)^P\right)$$

and memory footprint $\mathcal{O}\big(P \cdot \big(\max_p \left(n_p r_p r_{p-1}\right) + \max_p \left(n_p c_p\right)\big)\big)$ *for a gradient update on a batch of data. In the dual case, we take* $c_p = n_p$.

Proof When training on batches of data, the reconstruction for each datapoint is calculated by sequentially multiplying slices $\mathbf{V}_p(i_p), \mathbf{V}'_p(i_p) \in \mathbb{R}^{r_{p-1} \times r_p}$ from each TT-core. We start with the frequentist model for WLR and LS. We list the operations needed:

1. Any $\mathbf{V} \times_{p+1} \mathbf{D}_p$ operation. Complexity: $\mathcal{O}\big(\max_p \left(n_p c_p r_p r_{p-1}\right)\big)$. Memory: $\mathcal{O}\big(\max_p \left(n_p r_p r_{p-1}\right)\big) + \mathcal{O}\big(\max_p \left(n_p c_p\right)\big)$.
2. Any $\mathbf{V}_p \circ \mathbf{V}'_p$ operation. Complexity: $\mathcal{O}\big(\max_p \left(r_p r_{p-1}\right)\big)$. Memory: $\mathcal{O}\big(\max_p \left(r_p r_{p-1}\right)\big)$.
3. Reconstructing the tensor $\prod_{p=1}^{P} \times_{-1} \mathbf{V}_p$. Complexity: $\mathcal{O}\Big(\left(\max_p r_p\right)^P\Big)$. Memory: $\mathcal{O}\big(P \cdot \max_p \left(r_p r_{p-1}\right)\big)$.

For the Bayesian case we have the additional operations for the multivariate case:

1. $(\mathbf{K}_p \cdot \mathbf{B}_p)^2 \cdot \bar{1}$ component. Complexity: $\mathcal{O}\Big(\max_p \left(n_p^2 k_p\right)\Big)$. Here we take $\mathbf{B}_p \in \mathbb{R}^{n_p \times k_p}$ with $k_p \ll n_p$. Memory: $\mathcal{O}\Big(\max_p \left(n_p^2\right)\Big)$.
2. Prior determinant $\det\left(\Sigma_P\right)$. Complexity: $\mathcal{O}(\max_p(M_p^2))$, assuming we use RFFs when n_p is large. When using RFFs, we expect $M_p \ll N_p$. Memory: $\mathcal{O}(\max_p(n_p \cdot M_p))$.
3. Variational determinant $\det\left(\Sigma_Q\right)$. Complexity: $\mathcal{O}(\max_p(n_p))$, Memory: $\mathcal{O}(\max_p(n_p^2))$.

The dominating terms for complexity and memory are $\mathcal{O}\Big(\max_p \left(n_p c_p r_p r_{p-1}\right) + \left(\max_p r_p\right)^P\Big)$ and $\mathcal{O}\big(P \cdot \left(\max_p \left(n_p r_p r_{p-1}\right) + \max_p \left(n_p c_p\right)\right)\big)$ respectively. $\square$

Appendix D: Interpretability example analysis

As a guiding example of utilizing the interpretability of KFT, we analyse the temporal component of WLR applied to the alcohol dataset in the primal setting. We visualize the weights for the time component and in particular the "year" covariate.

Applying an LS model instead, we can directly decompose the predictions into scaling contribution, regression contribution and bias contribution (Fig. 7; Table 11).

Appendix E: Training procedure

E.1 Frequentist

To motivate the sequential updating scheme, consider a data matrix $\frac{\mathbf{X}}{\sigma} \in \mathbb{R}^{N \times d}$ where σ is a hyperparameter that controls the scaling of $\mathbf{X}$ and a target $\mathbf{Y} \in \mathbb{R}^N$. Assume N is too large and we have to resort to stochastic first order gradient methods to approximate $\mu \in \mathbb{R}^d$ in our regression $\left(\frac{\mathbf{X}}{\sigma}\right)\mu \approx \mathbf{Y}$. Using autograd (Paszke et al. 2017a), together with ADAM (Kingma and Ba 2014) we can in practice optimize $\{\sigma, \mu\}$ simultaneously for each iteration. However doing this, we commit a fallacy as when updating $\mu_t = \mu_{t-1} - \frac{\partial \mathcal{L}(\sigma_{t-1}, \mu_{t-1})}{\partial \mu_{t-1}}$ and $\sigma_t = \sigma_{t-1} - \frac{\partial \mathcal{L}(\sigma_{t-1}, \mu_{t-1})}{\partial \sigma_{t-1}}$, we do not account for the mixed partial $\frac{\partial}{\partial \sigma_{t-1}} \frac{\partial \mathcal{L}(\sigma_{t-1}, \mu_{t-1})}{\partial \mu_{t-1}} \propto \frac{-1}{\sigma_{t-1}^3}$ assuming $\mathcal{L}$ is mean square error loss. Thus updating $\{\sigma, \mu\}$ simultaneously using first order derivatives would yield an update error for σ as the updating gradient does not adjust for the mixed partial $\frac{\partial}{\partial \sigma_{t-1}} \frac{\partial \mathcal{L}(\sigma_{t-1}, \mu_{t-1})}{\partial \mu_{t-1}}$ when μ is being updated at the same time. This scenario extends one-to-one for the variables of KFT, as we would commit a similar fallacy by updating all parameters at once. Hence we take an EM inspired approach when updating $l_p, \mathbf{V}_p, \mathbf{V}_p'$.

E.2 Bayesian: variational inference

As the practical utility of calibrated uncertainty estimates rely on a good fit of the model, we motivate our sequential update scheme as a method to encourage good predictive performance by first finding the optimal modes determined by $\mathbf{M}, \mathbf{M}'$ and then the associated variance determined by Σ, Σ'. As we are considering the Gaussian meanfield family of models, this strategy is well motivated as Gaussian distribution is symmetric.

Appendix F: Data and data processing

F.1 Data processing

Our strategy in this paper is to limit data processing to simple operations that does not require excessive engineering for a fair comparison in both utility and input data. For all the models, we carry out the exact same preprocessing modulo the models requirements of data format. We do the following general processing steps:

1. Extract relevant features and parse them to be continuous or categorical.
2. Scale all features using a z-transformation.

For each model specifically we do the following:

1. KFT: tensorize all data by expressing all main modes (i.e. person, movie, time etc) as a tensor with side information associated with each mode.
2. LightGBM: here, we don't scale the features as boosting trees generally performs better with unscaled data. In some cases we have applied PCA to some of the side information that was joined on the data matrix to decrease the memory footprint of the data matrix to contain it to a practical size.
3. FFM: here we bin all continuous features, as FFM requires all data to be categorical.
4. Linear regression: for large categorical features, we using feature hashing to avoid data matrices of infeasible sizes. All other categorical features get one-hot encoded.

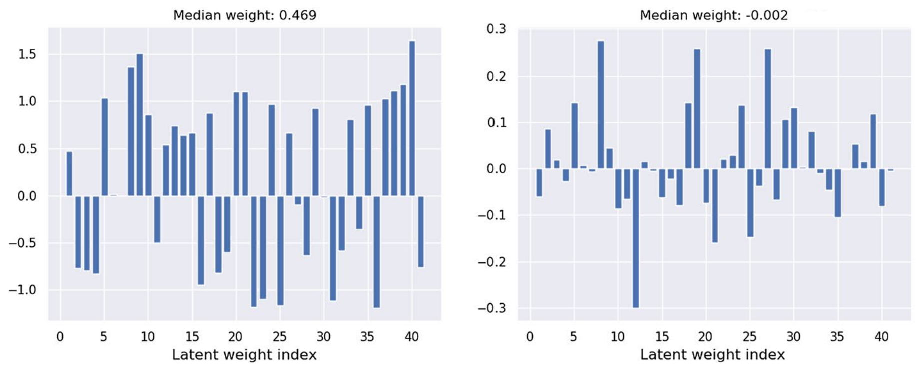


Fig. 7 The first plot shows the magnitude of the latent weights $\mathbf{V}$ for the "year" covariate. The second plot show the weights of the latent weights $\mathbf{V}'$ for year = 0

Table 11 5 decomposed example predictions for the LS model

Prediction nr	$\mathbf{V}_s$	$\mathbf{V}$	$\mathbf{V}_b$
0	−0.032064	46.948650	15.560461
1	−0.001596	333.260956	0.218784
2	−0.110225	29.397287	13.096196
3	0.028649	−45.733292	10.711301
4	−0.067812	−34.203766	3.496001

F.2 Retail sales data

We detail the features of the Retail Sales data in Table 12. Here we choose our modes to be store, articles and time.

F.3 Movielens-20M

We detail the features of the Retail Sales data in Table 13. Here we choose our modes to be users, movies and time of rating given. For the Movielens-20M data, it should be noted that we filter the movies on existing entries in the side information. This is why we only have roughly 11 million observations rather than 20 million.

F.4 Alcohol sales

We detail the features of the Alcohol Sales data in Table 14. Here we choose our modes to be location, item and time.

Appendix G: Hyperparameter configuration

G.1 KFT hyperparameters

We run all KFT experiments for 10 epochs with 20 hyperparameter search iterations. We consider two decomposition types:

1. *P*-way latent factorization, where each dimension has a latent component. In principle, this can be thought of as each dimension being independently factorized. Further we utilize all possible side information.
2. 2-way latent factorization, where time is grouped with another dimension as one latent component and with the other dimensions grouped in a second latent component. Here we only consider time as side information, for the purpose of only modelling temporal changes.

Our models are generally searched over the configurations described in Table 15. For exact details we refer to the code base.

Table 12 Dataset description for retail sales data

Property	Unit	Quantity
Monthly sales	Count	42,490,633
Unique location_id's	Count	80
Unique article_id's	Count	463,763
Article_id side info	**Description**	**Example**
Appearance_id	Article style	26
Colour_id	Article color	1337
Product_id	Article group	15
Size_id	Article size	L
Department_id	Article group theme	Teen
Product_season_id	Article group season	Spring
Product_type_id	Product group type	Shirt
Product_group_no	Product group no	Accessories
Price	Article price	5usd
Location_id side info	**Description**	**Example**
City_id	City of store	56
Longitude	Longitude	34.23
Latitude	Latitude	34.24
Brand_id	Brand	12
Opening_year_id	Opening year	123
Opening_month_id	Opening month	92
Opening_day_id	Opening day	97
Opening_hours_mo-su	Hours open mo-su	12,...,10
No_of_floors	Store no. of floors	10
Total_sales_area	Store area size	1231 sqm
Special_sales_area	Special sales area	789 sqm

Table 12 (continued)

Time side info	Description	Example
Year	Year	2019
Month	Month	1

Note that we only present the ID of each feature to preserve anonymity and that the examples are not real

Table 13 Dataset description of Movielens-20M

Property	Unit	Quantity
Ratings	Count	11,880,265
Users	Count	138,493
Movie_id	Count	10,370
Movie_id side info	Description	Example
Genome_score_1	Genome score dimension 1	0.345
⋮	⋮	⋮
Genome_score_1000	Genome score dimension 1000	0.1337
Time side info	Description	Example
Hour	Hour ratings was given	12

Table 14 Dataset description of Iowa alcohol sales

Property	Unit	Quantity
Bottles sold	Count	3,036,063
Unique store_location_id's	Count	3476
Unique item_id's	Count	4542
Item_id side info	Description	Example
Category	Type of alcohol	Irish whiskey
Pack	Size of package	6
Bottle volume (ml)	Bottle volume	750
State bottle cost	Cost for retailer to buy	4 usd
State bottle retail	Retail price	5 usd
Store_location_id side info	Description	Example
City_id	City of store	56
Longitude	Longitude	34.23
Latitude	Latitude	34.24
County	County	Shelby
Store_number	Store number	12
Zip code	Zip code	157
Time side info	Description	Example
Year	Year	2019
Month	Month	1

Table 15 Hyperparameter search space for KFT.

Property	Range/choices
Batch size[a]	[0.01, 0.1]
Learning rate	{1e−3, 1e−2, 1e−1}
R[b]	[5, 70]
λ_p	[0, 1]
λ'_p	[0, 1]
Kernel choice	{rbf, matern 0.5, matern 1.5, matern 2.5}

[a]The upper bound varies for each dataset, as memory limitation changes with dataset.

[b]Batch size expressed as a proportion of total training data examples. As an example our smallest batch size use 1% of the training data as the batch size

G.1.1 LightGBM hyperparameters

We provide hyperparameters for LightGBM in Table 16

Table 16 Hyperparameter search space for LightGBM.

Property	Range/choices
Num leaves	[7, 4095]
Learning rate	[exp(−5), exp(−2.3)]
Min data in leaf	[10, 30]
Min sum hessian in leaf	[exp(0), exp(2.3)]
Bagging freq	[1, 5]
λ_1	[0, 10]
λ_2	[0, 10]

G.1.2 FFM hyperparameters

We provide hyperparameters for FFM in Table 17.

Table 17 Hyperparameter search space for FFM.

Property	Range/choices
R	[2, 20]
Batch size	[1e−6, 0.1]
Learning rate	[0.05, 1.0]
λ	[0, 0.005]

G.1.3 Linear regresison hyperparameters

We provide hyperparameters for linear regression in Table 18.

Table 18 Hyperparameter search space for linear regression.

Property	Range/choices
Batch size	[1e−6, 0.1]
Learning rate	[0.05, 1.0]
λ	[0, 1.0]

G.2 Bayesian hyperparameters

We run all experiments for 25 epochs with 5 hyperparameter search iterations. For exact details we refer to the code base.

References

Agarwal, D., & Chen, B. C. (2009). *Regression-based latent factor models* (Vol. '09, pp. 19–28). Association for Computing Machinery. https://doi.org/10.1145/1557019.1557029

Batselier, K. (2018). The trouble with tensor ring decompositions. Preprint

Bergstra, J., Yamins, D., Cox, D. D. (2013) Making a Science of Model Search: Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures. Proc. of the 30th International Conference on Machine Learning (ICML 2013).

Bobadilla, J., Ortega, F., Hernando, A., & Bernal, J. (2012). A collaborative filtering approach to mitigate the new user cold start problem. *Knowledge-Based Systems, 26*, 225–238. https://doi.org/10.1016/j.knosys.2011.07.021

Bobadilla, J., Ortega, F., Hernando, A., & Gutiérrez, A. (2013). Recommender systems survey. *Knowledge-Based Systems, 46*, 109–132. https://doi.org/10.1016/j.knosys.2013.03.012

Caro, F., & Gallien, J. (2010). Inventory management of a fast-fashion retail network. *Operations Research.* https://doi.org/10.1287/opre.1090.0698

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)* **39**(1), 1–38. http://www.jstor.org/stable/2984875

Draper, N., & Smith, H. (1966). *Applied regression analysis. Wiley series in probability and mathematical statistics.* Wiley. http://gso.gbv.de/DB=2.1/CMD?ACT=SRCHA&SRT=YOP&IKT=1016&TRM=ppn+022791892&sourceid=fbw_bibsonomy

Du, Chao, Chongxuan Li, Yin Zheng, Jun Zhu, and Bo Zhang. (2018). Collaborative Filtering With User-Item Co-Autoregressive Models. *Proceedings of the AAAI Conference on Artificial Intelligence* **32**(1). https://ojs.aaai.org/index.php/AAAI/article/view/11884.

Gönen, M., Khan, S., & Kaski, S. (2013). In S. Dasgupta & D. McAllester (Eds.), *Kernelized Bayesian matrix factorization* (Vol. 28, pp. 864–872). PMLR. http://proceedings.mlr.press/v28/gonen13a.html

Hawkins, C., & Zhang, Z. (2018). Variational Bayesian Inference for Robust Streaming Tensor Factorization and Completion. 2018 IEEE International Conference on Data Mining (ICDM), 1446–1451.

He, L., Lu, C. T., Ma, G., Wang, S., Shen, L., Yu, P. S., & Ragin, A. B. (2017). In D. Precup & Y. W. Teh (Eds.) *Kernelized support tensor machines* (Vol. 70, pp. 1442–1451). PMLR, International Convention Centre. http://proceedings.mlr.press/v70/he17a.html

Hoffman, M. D., Blei, D. M., Wang, C., & Paisley, J. (2013). Stochastic variational inference. *Journal of Machine Learning Research,* **14**(1), 1303–1347. http://dl.acm.org/citation.cfm?id=2502581.2502622

Juan, Y., Zhuang, Y., Chin, W. S., & Lin, C. J. (2016). Field-aware factorization machines for ctr prediction. In *Proceedings of the 10th ACM Conference on Recommender Systems, RecSys '16* (pp. 43–50). ACM. https://doi.org/10.1145/2959100.2959134

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T. Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. In *NeurIPS*.

Kim, H., Lu, X., Flaxman, S., & Teh, Y. W. (2016). Collaborative filtering with side information: A Gaussian process perspective. Preprint

Kim, H., & Teh, Y.W. (2018). Scaling up the Automatic Statistician: Scalable Structure Discovery using Gaussian Processes. AISTATS.

Kim, Y. D., & Choi, S. (2014). Scalable variational Bayesian matrix factorization with side information. In *AISTATS*.

Kingma, Diederik & Ba, Jimmy. (2014). Adam: A Method for Stochastic Optimization. International Conference on Learning Representations.

Kolda, T. G., & Bader, B. W. (2009). Tensor decompositions and applications. *SIAM Review, 51*(3), 455–500.

Kuang, L., Hao, F., Yang, L. T., Lin, M., Luo, C., & Min, G. (2014). A tensor-based approach for big data representation and dimensionality reduction. *IEEE Transactions on Emerging Topics in Computing, 2*(03), 280–291. https://doi.org/10.1109/TETC.2014.2330516

Liu, T., Wang, Z., Tang, J., Yang, S., Huang, G. Y., & Liu, Z. (2019). *Recommender systems with heterogeneous side information* (Vol. '19, pp. 3027–3033). Association for Computing Machinery. https://doi.org/10.1145/3308558.3313580

Narita, A., Hayashi, K., Tomioka, R., & Kashima, H. (2012). Tensor factorization using auxiliary information. *Data Mining and Knowledge Discovery, 25*(2), 298–324. https://doi.org/10.1007/s10618-012-0280-z

Ong, Victor & Nott, David & Smith, Michael. (2017). Gaussian Variational Approximation With a Factor Covariance Structure. Journal of Computational and Graphical Statistics. 27. https://doi.org/10.1080/10618600.2017.1390472.

Oseledets, I. V. (2011). Tensor-train decomposition. *SIAM Journal on Scientific Computing, 33*(5), 2295–2317. https://doi.org/10.1137/090752286

Pal, B., & Jenamani, M. (2018). *Kernelized probabilistic matrix factorization for collaborative filtering: Exploiting projected user and item graph* (pp. 437–440). Association for Computing Machinery. https://doi.org/10.1145/3240323.3240402

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L. & Lerer, A. (2017). Automatic Differentiation in PyTorch. NIPS 2017 Workshop on Autodiff.

Rahimi, A., & Recht, B. (2007). *Random features for large-scale kernel machines* (Vol. NIPS'07, pp. 1177–1184). Curran Associates Inc.

Rendle, S. (2010). Factorization machines. In G.I. Webb, B. Liu, C. Zhang, D. Gunopulos, & X. Wu (Eds.) *ICDM 2010, The 10th IEEE international conference on data mining, Sydney, Australia, 14–17 December 2010* (pp. 995–1000). IEEE Computer Society. https://doi.org/10.1109/ICDM.2010.127

Rudin, Cynthia. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence.* 1, 206–215. https://doi.org/10.1038/s42256-019-0048-x.

Saha, A., Misra, R., Acharya, A., & Ravindran, B. (2017). Scalable variational Bayesian factorization machine. Preprint

Salakhutdinov, R., & Mnih, A. (2007). Probabilistic matrix factorization. In *Proceedings of the 20th international conference on neural information processing systems, NeurIPS* (pp. 1257–1264). Curran Associates Inc. http://dl.acm.org/citation.cfm?id=2981562.2981720

Flunkert, Valentin & Salinas, David & Gasthaus, Jan. (2017). DeepAR: Probabilistic Forecasting with Autoregressive Recurrent Networks. International Journal of Forecasting. 36. https://doi.org/10.1016/j.ijforecast.2019.07.001.

Schölkopf, B., Herbrich, R., & Smola, A. J. (2001). A generalized representer theorem. In *COLT/EuroCOLT*.

Seeger, M., Salinas, D., & Flunkert, V. (2016). Bayesian intermittent demand forecasting for large inventories. In *NeurIPS*.

Vovk, V. (2013). Kernel Ridge Regression. Empirical Inference.

Wu, X., Shi, B., Dong, Y., Huang, C., & Chawla, N. V. (2019). *Neural tensor factorization for temporal interaction learning* (Vol. WSDM '19, pp. 537–545). Association for Computing Machinery. https://doi.org/10.1145/3289600.3290998

Xu, J., Yao, Y., Tong, H., Tao, X., & Lu, J. (2015). Ice-breaking: Mitigating cold-start recommendation problem by rating comparison. In *IJCAI international joint conference on artificial intelligence. 24th international joint conference on artificial intelligence, IJCAI 2015; conference date: 25-07-2015 through 31-07-2015* (Vol. 2015, pp. 3981–3987).

Yu, H., Rao, N.S., & Dhillon, I.S. (2016). Temporal Regularized Matrix Factorization for High-dimensional Time Series Prediction. NIPS.

Yu, R., Bahadori, M. T., & Liu, Y. (2014). Fast multivariate spatio-temporal analysis via low rank tensor learning. In *Advances in neural information processing systems* (pp. 3491–3499).

Zellner, A. (1986). On Assessing Prior Distributions and Bayesian Regression Analysis With g-Prior Distributions. Basic Bayesian Inference and Decision Techniques: Essays in Honor of Bruno de Finetti

Zhai, Y., Ong, Y., & Tsang, I. W. (2014). The emerging "big dimensionality". *IEEE Computational Intelligence Magazine, 9*(3), 14–26. https://doi.org/10.1109/MCI.2014.2326099

Zhang, J., Shi, X., Zhao, S., & King, I. (2019). STAR-GCN: Stacked and Reconstructed Graph Convolutional Networks for Recommender Systems. IJCAI.

Zhang, M., Tang, J., Zhang, X., & Xue, X. (2014). Addressing cold start in recommender systems: A semi-supervised co-training algorithm. *Association for Computing Machinery.* **14**, 73–82 https://doi.org/10.1145/2600428.2609599

Zhao, Q., Zhou, G., Adali, T., Zhang, L., & Cichocki, A. (2013). Kernelization of tensor-based models for multiway data analysis: Processing of multidimensional structured data. *IEEE Signal Processing Magazine, 30*(4), 137–148. https://doi.org/10.1109/MSP.2013.2255334

Zhao, Q., Zhou, G., Xie, S., Zhang, L., & Cichocki, A. (2016). Tensor ring decomposition. Preprint