2019 27th European Signal Processing Conference (EUSIPCO)

Tensor Network Kalman Filter for LTI Systems

Daniel Gedon†, Pieter Piscaer†, Kim Batselier†, Carlas Smith†, Michel Verhaegen†

***Abstract*—An extension of the Tensor Network (TN) Kalman filter [2], [3] for large scale LTI systems is presented in this paper. The TN Kalman filter can handle exponentially large state vectors without constructing them explicitly. In order to have efficient algebraic operations, a low TN rank is required. We exploit the possibility to approximate the covariance matrix as a TN with a low TN rank. This reduces the computational complexity for general SISO and MIMO LTI systems with TN rank greater than one significantly while obtaining an accurate estimation. Improvements of this method in terms of computational complexity compared to the conventional Kalman filter are demonstrated in numerical simulations for large scale systems.**

***Index Terms*—Kalman filter, LTI systems, tensors, tensor train, large scale systems, SISO, MIMO, curse of dimensionality.**

I. Introduction

The Kalman filter [9] is a stochastic optimal filter for dynamic linear systems. Since its introduction, it is successfully applied to a variety of different applications, see e.g. [1], [7]. For systems with exponentially large state size n^d and output size p assuming $p \leq n^d$, the conventional Kalman filter is infeasible. First, because the computational complexity scales with order $\mathcal{O}\left(n^{3d}\right)$. Second, the storage of exponentially large system dynamics is in matrix form prohibitive. A more suitable filter framework has to be developed.

One possibility for a large scale Kalman filter is the Tensor Network (TN) Kalman filter as developed in [2], [3]. Both concern the system identification of Multiple-Input Multiple-Output (MIMO) Volterra systems using Kalman filter where the latter one uses a batch of multiple measurements. The Volterra systems are rewritten in LTI system form and therefore the described implementation can equally be used for dynamic linear systems. Hence, in the remainder of the paper the method in [2] will be denoted as Single-Input Single-Output (SISO) and the method in [3] as MIMO TN Kalman filter. This filter makes use of special TNs [12] in Tensor Train (TT) format [13] without explicitly constructing the underlying exponentially large matrices and vectors. Hence, a reduction of computational complexity from $\mathcal{O}\left(n^d\right)$ to $\mathcal{O}\left(dn\right)$ is achieved. Fig. 1 highlights the computational advantage of the TN Kalman filter for large LTI systems while still obtaining accurate estimation results (see full and dashed line). This holds for the special case where the system dynamics in TT-format have the property that all TN ranks are equal to one, denoted as $\mathrm{ttr}(\cdot) = 1$ and formally introduced in the preliminaries.

† Affiliated with Delft Center for Systems and Control, Delft University of Technology, The Netherlands. E-mail addresses of corresponding authors: d.gedon@student.tudelft.nl (Daniel Gedon), p.j.piscaer@tudelft.nl (Pieter Piscaer).

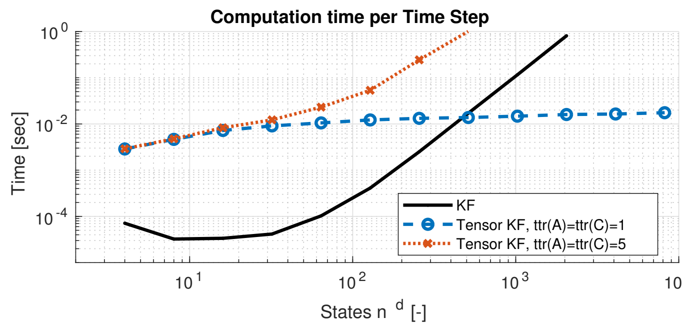

Fig. 1. Computation time per time step of random stable LTI SISO system for conventional and TN Kalman filter using rounding tolerance $\epsilon = 10^{-6}$ and mean out of 15 Monte Carlo simulations (using varying initial conditions and noise) over 25 time steps.

In general, the TN ranks can be high for large unstructured systems and are only bounded by the canonical rank [16] of an exact canonical decomposition of the underlying tensor [4], [11]. The computational complexity of the TN Kalman filter is polynomial dependent on the TN rank of the system dynamics matrices $\mathbf{A}$ and $\mathbf{C}$ in TT-format and the TN rank of the internal variables of the filter (covariance, Kalman gain, state estimate). This results in longer computation times for larger TN ranks. Additionally, the implementation of the MIMO TN Kalman filter in [3] has a complexity of $\mathcal{O}\left(p^3\right)$.

In this paper, a solution is proposed to tackle the problem of large TN ranks in order to reduce computation time while obtaining accurate results. This is especially of interest since general system matrices do not have an optimal TN rank of one. Fig. 1 shows the computation times in a comparison for systems with TN rank at one and higher. The main contributions of this paper can be summarized as:

- A possibility to reduce the effect of the TN ranks on the computational complexity is derived. Therefore, the covariance tensor in the TN Kalman filter is approximated with low TN ranks, yielding a fast and accurate estimation.
- Numeric simulation results are presented as comparison between the conventional and TN Kalman filter. The latter is shown with and without low TN rank approximation of the covariance tensor to demonstrate the power of the novel approach.

The paper is structured as follows. Section 2 introduces the notion of tensors throughout the paper. It gives an overview on TTs with multilinear operations and introduces the use of TTs in the generalized TN Kalman filter. In section 3, the method to identify the computational bottlenecks of the TN Kalman filter is highlighted. Based on this analysis, the approach to approximate the covariance tensor is derived. Using these insights, numerical simulations of the novel approach are presented in section 4. Finally, section 5 concludes with final remarks and open research directions.

II. Preliminaries

In this paper, a tensor is a multidimensional array as a generalization of matrices to higher order. An order-d tensor has d indices and is denoted with capital calligraphic letters $\mathcal{X} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$. Matrices will be denoted by capital bold letters $\mathbf{X} \in \mathbb{R}^{n_1 \times n_2}$ and vectors by bold letters $\mathbf{x} \in \mathbb{R}^{n_1}$. Scalars are given as Roman letters $x \in \mathbb{R}$. The ith tensor element of a set of tensors is indicated by superscript in round brackets $\mathcal{X}^{(i)}$. Introductions to tensors and their decompositions are given in [6], [11], [12].

A. Tensor Train Theory

The Tensor Train (TT) decomposition is mathematically introduced by [13] after previous work in [15], [17]. It decomposes an order-d tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ in a series of d order-3 tensors $\mathcal{X}^{(1)}, \ldots, \mathcal{X}^{(d)}$ with $\mathcal{X}^{(i)} \in \mathbb{R}^{r_{X_{i-1}} \times n_i \times r_{X_i}}$ $\forall i$, called TN cores. The parameter r_{X_i} are the TN ranks connecting the single TN cores in a line network. The border TN cores are defined as $r_{X_0} = r_{X_d} = 1$. Throughout this paper, the maximum TN rank over all TN ranks of a TT $\mathcal{X}$ will be denoted as $\mathrm{ttr}(\mathcal{X}) = \max(r_{X_i}) = r_X$ $\forall i$.

An example for a graphical visualization of the TT decomposition using TN diagrams [6], [12] is given for an order-6 tensor $\mathcal{X}$ in Fig. 2. Each circle depicts a TN core; the number of free lines is the order of the tensor; the size of the interconnecting lines represent the TN ranks r_{X_i}. The example decomposes the order-6 tensor $\mathcal{X}$ in six order-3 tensors $\mathcal{X}^{(i)}$, $i = 1, \ldots, 6$.

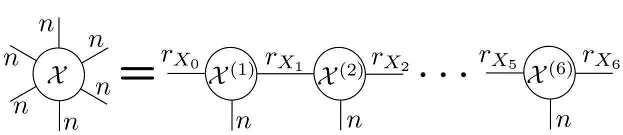

Fig. 2. Graphical visualization of the TT decomposition of an order-6 tensor $\mathcal{X}$ using TN diagrams.

The power of the TT decomposition is twofold. Storing an exponentially large matrix $\mathbf{X} \in \mathbb{R}^{n^d \times n^d}$ requires storage of $\mathcal{O}\left(n^{2d}\right)$ entries. In TT-format storage of $\mathcal{X}$ requires only $\mathcal{O}\left(dn^2 r_X^2\right)$ entries, i.e. linear in the exponent d, which is a reduction if the TN ranks r_X are small. Secondly, multilinear algebra as a generalization of linear algebra to higher spaces can be used to effectively apply basic mathematical operations [8]. This is computationally effective since the operations work on each single TN core, which are small compared to the full matrix. An example of multilinear algebra is the so called mode-n product as a higher order equivalent of matrix multiplications for tensors [6], [11]. Often, the term contraction along the mode n is used for this operation.

A problem with multilinear operations in TT-format is the increase of the TN rank. For example, the contraction along the second mode of the tensor $\mathcal{A}$ with TN cores $\mathcal{A}^{(i)} \in \mathbb{R}^{r_A \times n \times r_A}$ with the tensor $\mathcal{B}$ with TN cores $\mathcal{B}^{(i)} \in \mathbb{R}^{r_B \times n \times r_B}$ has resulting TN ranks $r_A r_B$. In [13], a procedure called rounding is described to decrease the TN ranks towards a given rounding tolerance ϵ. Two main steps are needed. First, an orthogonalization of all TN cores using QR decompositions and second, a δ-truncated SVD.

Consider for example a tensor $\mathcal{X}$ in TT-format with orthogonal TN cores. The procedure computes the δ-truncated SVD of each matricified TN core $\mathcal{X}^{(i)} \in \mathbb{R}^{r_{X_{i-1}} \times n \times r_{X_i}}$. The matricification is done such that the number of columns is equal to the right TN rank $\mathbf{X}^{(i)}_{(r_{X_i})} \in \mathbb{R}^{r_{X_{i-1}} n \times r_{X_i}}$. The truncation threshold is given by

$$\delta = ||\mathbf{X}||_F \frac{\epsilon}{\sqrt{d-1}}, \tag{1}$$

using the rounding tolerance ϵ and the number of TN cores d.

B. Tensor Kalman Filter

The TT theory can be used to define a Kalman filter and lift the curse of dimensionality for large scale systems. Therefore, the following LTI system is considered

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{A}\mathbf{x}_k + \mathbf{w}_k \\ \mathbf{y}_k &= \mathbf{C}\mathbf{x}_k + \mathbf{v}_k \end{aligned} \tag{2}$$

with state $\mathbf{x}_k \in \mathbb{R}^{n^d}$, measurement $\mathbf{y}_k \in \mathbb{R}^p$, $\mathbf{A} \in \mathbb{R}^{n^d \times n^d}$, $\mathbf{C} \in \mathbb{R}^{p \times n^d}$ and covariance matrices $\mathbf{Q} \in \mathbb{R}^{n^d \times n^d}$ and $\mathbf{R} \in \mathbb{R}^{p \times p}$ for the process noise $\mathbf{w}_k$ and measurement noise $\mathbf{v}_k$, respectively. The output is $p = 1$ for SISO and $p = n^d$ for the MIMO case. The algebraic equations of the conventional Kalman filter are split in two parts. The measurement update:

$$\mathbf{S} = \mathbf{C}\mathbf{P}_{k|k-1}\mathbf{C}^\top + \mathbf{R} \tag{3}$$

$$\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{C}^\top\mathbf{S}^{-1} \tag{4}$$

$$\mathbf{v} = \mathbf{y}_k - \mathbf{C}\hat{\mathbf{x}}_{k|k-1} \tag{5}$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k\mathbf{v} \tag{6}$$

$$\mathbf{P}_{k|k} = \left(\mathbf{I}_n - \mathbf{K}_k\mathbf{C}\right)\mathbf{P}_{k|k-1} \tag{7}$$

and the time update:

$$\hat{\mathbf{x}}_{k+1|k} = \mathbf{A}\hat{\mathbf{x}}_{k|k} \tag{8}$$

$$\mathbf{P}_{k+1|k} = \mathbf{A}\mathbf{P}_{k|k}\mathbf{A}^\top + \mathbf{Q} \tag{9}$$

When the system matrices $\mathbf{A}$, $\mathbf{C}$ are large scale matrices, the computation time rises exponentially as seen in Fig. 1. This curse of dimensionality can be lifted by using the TN Kalman filter. This is a generalization of the conventional Kalman filter for higher order dimensions in TT format. It adapts the Kalman filter equations to the required multilinear algebra. The TN Kalman filter is introduced in [2] and [3].

In order to apply the TN Kalman filter to general LTI systems, the system dynamics in Eq. (2) have to be transformed in TTs. Consequently, also the variables in the TN Kalman filter are TTs. Use is made of the computational effective multilinear algebra for TTs with this transformation. As an example, the transformation of the state vector estimate $\hat{\mathbf{x}}$ to TT-format is explained intuitively. First, the vector $\hat{\mathbf{x}} \in \mathbb{R}^{n^d}$ is reshaped into an order-d tensor $\hat{\mathcal{X}} \in \mathbb{R}^{n \times \cdots \times n}$. Second, the order-$d$ tensor is decomposed using the TT decomposition in d order-3 tensors $\hat{\mathcal{X}}^{(i)} \in \mathbb{R}^{r_{X_{i-1}} \times n \times r_{X_i}}$. The set counting index i goes from 1 to d and the border TN ranks are $r_{X_0} = r_{X_d} = 1$. This decomposition step is visualized for $d = 6$ in Fig. 2 using a TN diagram. The TN rank r_{X_i} connects the TN core $\hat{\mathcal{X}}^{(i)}$ with $\hat{\mathcal{X}}^{(i+1)}$. For the state transition matrix $\mathbf{A} \in \mathbb{R}^{n^d \times n^d}$ in TT-format, there is one more mode of size n for each TN core compared to the state vector $\hat{\mathbf{x}}$. These examples illustrate how the following variables for the TN Kalman filter are defined in TT-format

- $\mathcal{A}$ with $\mathcal{A}^{(i)} \in \mathbb{R}^{r_{A_{i-1}} \times n \times n \times r_{A_i}}$
- $\mathcal{C}$ with $\mathcal{C}^{(1)} \in \mathbb{R}^{r_{C_0} \times p \times n \times r_{C_1}}$ and $\mathcal{C}^{(i)} \in \mathbb{R}^{r_{C_{i-1}} \times n \times r_{C_i}}$
- $\mathcal{P}_{(\cdot|\cdot)}$ with $\mathcal{P}^{(i)}_{(\cdot|\cdot)} \in \mathbb{R}^{r_{P_{i-1}} \times n \times n \times r_{P_i}}$
- $\hat{\mathcal{X}}_{(\cdot|\cdot)}$ with $\hat{\mathcal{X}}^{(i)}_{(\cdot|\cdot)} \in \mathbb{R}^{r_{X_{i-1}} \times n \times r_{X_i}}$
- $\mathcal{K}_k$ with $\mathcal{K}^{(1)}_k \in \mathbb{R}^{r_{K_0} \times p \times n \times r_{K_1}}$ and $\mathcal{K}^{(i)}_k \in \mathbb{R}^{r_{K_{i-1}} \times n \times r_{K_i}}$
- $\mathcal{Q}$ with $\mathcal{Q}^{(i)} \in \mathbb{R}^{r_{Q_{i-1}} \times n \times n \times r_{Q_i}}$

For completeness, the definition of the remaining matrices and vectors are given as $\mathbf{S} \in \mathbb{R}^{p \times p}$, $\mathbf{R} \in \mathbb{R}^{p \times p}$ and $\mathbf{v} \in \mathbb{R}^p$. The specific implementation for the SISO and MIMO filter is elaborated in [2] and [3] respectively.

III. Problem Analysis and Improvment

In this part, the problem of increasing computation time for high TN ranks due to their polynomial complexity is tackled. This is done for the SISO and MIMO case since both struggle with this phenomenon. Stable LTI systems without specific structure of the matrices are considered. Converting such system dynamics in TT-format yields in general $\mathrm{ttr}(\mathcal{A}) > 1$, $\mathrm{ttr}(\mathcal{C}) > 1$. This general case is taken into account for the following analysis. Hence, this will reduce the computational speed of the TN Kalman filter significantly as presented in Fig. 1 for the SISO case. Previous results in [2] have shown that a rounding tolerance greater than the machine precision is sufficient for accurate results and can speed up the computation time. This result is applied in the following by choosing $\epsilon = 10^{-6}$.

A. Analysis of State of the Art

To get insights on possible bottlenecks of the algorithm, the computational complexity of the state of the art is analysed. For the SISO tensor Kalman filter this can be found in Table 2 of [2] and summarized with $\mathcal{O}(dn^3 r^x_{(\cdot)})$ where the last factor indicates the polynomial complexity in several TN ranks. This compares to $\mathcal{O}(n^{3d})$ for a comparable conventional SISO Kalman filter. For the state of the art MIMO tensor KF, the complexity analysis is given in Table I.

It can be seen that there are two computational bottlenecks. (1) The computation of the Kalman gain with cubic complexity in the outputs p, since the inverse of the result of the Riccati equation $\mathbf{S}$ is necessary. (2) The polynomial dependency on all TN ranks. This paper addresses point two, while the first point is a topic for future research.

TABLE I
Computational Complexity of MIMO TN Kalman Filter.

Step	Kalman filter	TN Kalman filter
$\mathbf{S}$	$\mathcal{O}\left(pn^{2d} + p^2 n^d\right)$	$\mathcal{O}((d-1)(n^2 r_P^2 r_C^2 + n r_P^2 r_C^4) + $ $+ n^2 p r_P r_C + n p^2 r_P r_C^2)$
$\mathcal{K}$	$\mathcal{O}(pn^{2d} + p^2 n^d +$ $+ p^3)$	$\mathcal{O}((d-1)(n^2 r_P^2 r_C^2) + n^2 p r_P r_C +$ $+ n p^2 r_P r_C + p^3)$
$\mathbf{v}$	$\mathcal{O}\left(pn^d\right)$	$\mathcal{O}\left((d-1) n r_X^2 r_C^2 + n p r_X r_C\right)$
$\hat{\mathcal{X}}_{k\|k}$	$\mathcal{O}\left(pn^d\right)$	$\mathcal{O}\left(n p r_{K_1}\right)$
$\mathcal{P}_{k\|k}$	$\mathcal{O}\left(pn^{2d} + p^2 n^d\right)$	$\mathcal{O}\left((d-1) n^2 r_K^4 + n^2 p r_K^2 + n p^2 r_K\right)$
$\hat{\mathcal{X}}_{k+1\|k}$	$\mathcal{O}\left(n^{2d}\right)$	$\mathcal{O}\left(d n^2 r_X^2 r_A^2\right)$
$\mathcal{P}_{k+1\|k}$	$\mathcal{O}\left(n^{3d}\right)$	$\mathcal{O}\left(d n^3 r_P^2 r_A^4\right)$

The TN ranks of the system dynamics r_A, r_C are inherently given and determine subsequently the TN ranks of the variables in the filter. Fig. 3 shows the converged maximum TN rank of TT variables in the MIMO TN Kalman filter using $p = n^d$ for increasing sizes of random systems with $\mathrm{ttr}(\mathcal{A}) = \mathrm{ttr}(\mathcal{C}) = 5$. The TN rank choice is made to ensure a higher TN rank than unity for the system dynamics while keeping it low enough to obtain feasible computation times. The TN cores are generated randomly using the `tt_rand(·)` command from [14]. Rounding tolerance is set to $\epsilon = 10^{-6}$ using 5 Monte Carlo simulations with 50 time steps to ensure convergence. The figure highlights that the TN ranks of the measurement and time update covariance tensor are driving since they grow exponentially with the system size. This yields a tremendous increase in computation time for large systems if no counter measure is done.

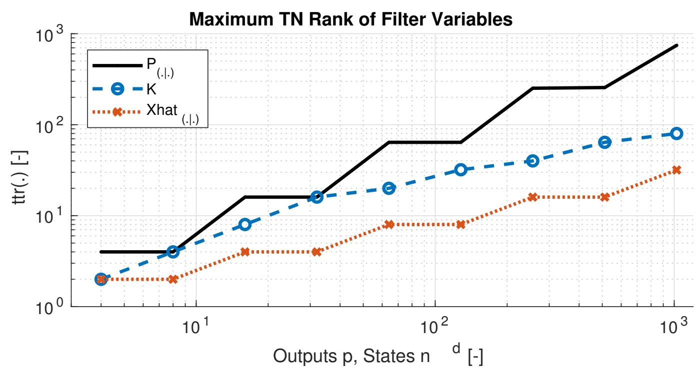

Fig. 3. Maximum TN rank of variables in MIMO TN Kalman filter for increasing system size with $n^d = p$ using $\mathrm{ttr}(\mathcal{A}) = \mathrm{ttr}(\mathcal{C}) = 5$.

B. Extension for Reduction of Complexity

The variables in the analysis of Fig. 3 do have optimal TN ranks according to the selected rounding tolerance ϵ. The main idea of this paper is to obtain low TN rank approximations of the driving factors, the covariance tensors, by choosing lower TN ranks. This decreases the effect of polynomial computational complexity.

The truncation of TN ranks in the covariance tensor to lower values is based on the idea of the rounding procedure [13]. After the orthogonalization step, a δ-truncated SVD is executed which determines the TN rank. The truncation threshold δ is computed with Eq. (1) using the rounding tolerance ϵ. However, the choice for the truncation threshold is dependent on the distribution of the singular values. In practice, it is often difficult to choose the tolerance ϵ for the SVD truncation [5]. One approach is to use a fixed truncation value, see e.g. [10]. If the TN rank is truncated with a fixed $r \leq \delta$, only the r most dominant singular values of each TN core are taken into account. Therefore, the TN rounding function is adapted such that the SVD truncation threshold is chosen as

$$\min(r, \delta)$$

in order to obtain the lowest possible truncation threshold. The consequences of a low TN rank approximation in the TN Kalman filter are in general:

- Reduction in computation time. This holds since a lower TN rank of the covariance tensors will yield lower TN ranks of subsequently computed variables and decreases their influence on the computation time.
- Lower accuracy of the resulting estimation since a low rank approximation is used.

Similar analysis as presented in Sec. IV has shown that in case of truncating the TN rank of the second most driving parameter, the Kalman gain, that the filter even tends to diverge. Also, it is not useful to truncate the TN rank of the estimated state since this is an output of the filter and should be kept with the desired accuracy as chosen by the rounding tolerance. Moreover, its influence is according to Fig. 3 less driving than the covariance or Kalman gain. Note that truncation of the covariances $\mathcal{P}_{k|k}$ and $\mathcal{P}_{k+1|k}$ is done online at each time step.

IV. Simulation Results and Discussion

Both the SISO and the MIMO filter can easily be extended with the proposed approach. The power of the truncation method is shown in simulation. Different random stable LTI systems are generated with the state vector size n^d and output size $p = 1$ for SISO and $p = n^d$ for MIMO systems. For simplicity, a mode size of $n = 2$ is chosen as smallest prime factor equally to [17]. The system dynamics are in TT-format described with $\mathrm{ttr}(\mathcal{A}) = \mathrm{ttr}(\mathcal{C}) = 5$, which is a generalization compared to TN rank unity. The TN rank of the process noise covariance tensor is given by $\mathrm{ttr}(\mathcal{Q}) = 1$. For the assumption of the matrix $\mathbf{Q}$ being diagonal, this holds. The entries of the diagonal process noise covariance are set to 0.1. The measurement noise covariance matrix $\mathbf{R}$ is diagonal with all entries at 0.5. The rounding tolerance is $\epsilon = 10^{-6}$, and a total of 15 Monte Carlo simulations are run over 25 time steps.

The simulation is run on Matlab version 9.3.0.713579 (R2017b) installed on Linux Ubuntu 16.04 LTS making use of the TT toolbox [14], Tensorlab v3.0 [18] and functions provided with the code of [2], [3]. The hardware consists of an Intel core i5-7200 quad-core CPU running at 2.5 GHz with 7.7 GB RAM.

A. SISO Kalman Filter

A comparison between the conventional Kalman filter and the SISO TN Kalman filter with the mentioned properties is presented in Fig. 4. For the TN Kalman filter, three settings are simulated: without truncation of the covariance tensor TN ranks as well as with truncation at $\mathrm{ttr}(\mathcal{P}_{(\cdot|\cdot)}) = \{1, 5\}$. The figure shows the computation time per time step of each filter; the mean computation time over 15 Monte Carlo simulations is depicted. The variance of the computation time is negligible with maximum magnitude of 10^{-1}.

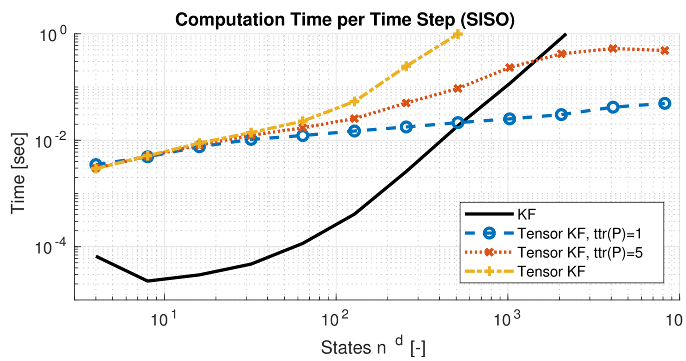

Fig. 4. Computation time per time step of random stable LTI SISO system for conventional and TN Kalman filter with TN rank $\mathrm{ttr}(\mathcal{A}) = \mathrm{ttr}(\mathcal{C}) = 5$ using covariance TN rank truncation.

The results for the SISO filter yield two main points. First, a comparison of the output estimation of the TN Kalman filter with and without covariance TN truncation and the conventional Kalman filter is done. The relative 2-Norm squared of the outputs is used for this comparison defined as

$$\frac{||\mathrm{vec}(\mathbf{y}) - \mathrm{vec}(\hat{\mathbf{y}})||_2^2}{||\mathrm{vec}(\mathbf{y})||_2^2},$$

where $\mathrm{vec}(\mathbf{y})$ is the vectorized output of the simulated LTI system and $\mathrm{vec}(\hat{\mathbf{y}})$ is the vectorized estimated output of the respective Kalman filter. The difference between the relative 2-Norm squared of the conventional Kalman filter and the TN Kalman filter is negligibly small and, therefore, the estimation is accurate even with covariance TN rank truncation. This yields that the approximation of the covariance tensor with lower TN rank is still sufficiently accurate such that the overall error stays small.

Second, the dash-dotted line with '+' marker in Fig. 4 shows the computation time of the state of the art for systems with $\mathrm{ttr}(\mathcal{A}) > 1$, $\mathrm{ttr}(\mathcal{C}) > 1$. The computation time grows exponentially and is even more than two magnitudes higher than for the conventional Kalman filter. Using the developed approach to truncate the TN rank of the covariance tensor yields a great improvement. With truncation at $\mathrm{ttr}(\mathcal{P}) = 1$, the computation time becomes linear - similar to the results for $\mathrm{ttr}(\mathcal{A}) = \mathrm{ttr}(\mathcal{C}) = 1$ in Fig. 1. For large scale systems with $n^d > 600$ states, this approach yields accurate and significantly faster estimation results.

B. MIMO Kalman Filter

For the MIMO Kalman filter, simulations are run with the same setting using $p = n^d$, meaning the number of outputs is equal to the number of states. The resulting computation times per time step are depicted in Fig. 5 using the mean of 15 Monte Carlo simulations.

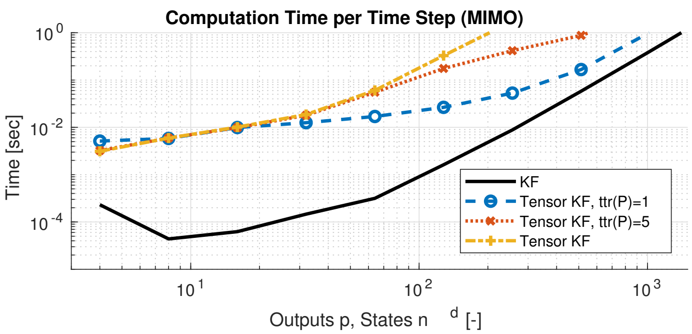

Fig. 5. Computation time per time step of random stable LTI MIMO system for conventional and TN Kalman filter with TN rank $\mathrm{ttr}(\mathcal{A}) = \mathrm{ttr}(\mathcal{C}) = 5$ using covariance TN rank truncation.

Similar to the results of the SISO filter, the MIMO TN Kalman filter has the same relative 2-Norm of the output with and without covariance TN rank truncation and even with the conventional Kalman filter. This confirms the optimality of the TN Kalman filter. Regarding the computation time, the state of the art TN Kalman filter for $\mathrm{ttr}(\mathcal{A}) > 1$, $\mathrm{ttr}(\mathcal{C}) > 1$ is given by the dash-dotted line in Fig. 5. This depicts an exponential increase of computation time being at least two magnitudes slower than the conventional Kalman filter. The TN Kalman filter with $\mathrm{ttr}(\mathcal{P}) = 1$ truncation converges for large n^d to the one of the conventional Kalman filter. This is reasonable since both filters have a complexity of $\mathcal{O}\left(p^3\right)$ which is not tackled in this paper. Hence, for the MIMO case the approach of covariance TN rank truncation yields an improvement in computation time over the TN Kalman filter without it.

Remark: Note that in all simulations the covariance matrix remained symmetric with $\mathcal{P}_{\cdot|\cdot} - \mathcal{P}_{\cdot|\cdot}^{\top}$ at machine precision level. Moreover, $\mathcal{P}_{k+1|k}$ remained postive definite, while $\mathcal{P}_{k|k}$ only had the smallest eigenvalue larger than $-$eps.

V. Conclusion

The paper discusses the need for improvement of TN Kalman filter for general LTI systems with TN rank $\mathrm{ttr}(\mathcal{A}) > 1$, $\mathrm{ttr}(\mathcal{C}) > 1$ due to computation time issues. Therefore, the concept of TNs in TTs and the TN Kalman filter for the SISO and MIMO case is explained in detail. An analysis of the state of the art shows a polynomial dependency on the computational complexity of the TN ranks decreasing the computational time for large TN ranks. The driving variable with the highest TN ranks within the Kalman filter is identified to be the covariance matrix. Truncating its TN ranks by considering only the most dominant singular values is verified in simulation to yield accurate and computationally fast results. This approach is applied to both the SISO and MIMO case, improving the computation time for general LTI systems.

The complexity analysis and results for the MIMO TN Kalman filter highlights the output dependency of $\mathcal{O}\left(p^3\right)$. This limits the MIMO filter to the same speed as the conventional Kalman filter. Future work concentrates on the reduction of complexity in the number of outputs. Moreover, a square-root implementation of the TN Kalman filter is desired for practical and numerical problems. This requires the computation of a Cholesky or QR factorization in TT format, which is to the knowledge of the authors not yet efficiently solved.

References

[1] F. Auger, M. Hilairet, J. M. Guerrero, E. Monmasson, T. Orlowska-Kowalska, and S. Katsura, "Industrial applications of the Kalman filter: A review," *IEEE Transactions on Industrial Electronics*, vol. 60, no. 12, pp. 5458–5471, Dec 2013.

[2] K. Batselier, Z. Chen, and N. Wong, "A Tensor Network Kalman filter with an application in recursive MIMO Volterra system identification," *Automatica*, vol. 84, 10 2016.

[3] K. Batselier and N. Wong, "Matrix output extension of the Tensor Network Kalman filter with an application in MIMO Volterra system identification," *Automatica*, vol. 95, pp. 413–418, 2018.

[4] J. D. Carroll and J.-J. Chang, "Analysis of individual differences in multidimensional scaling via an n-way generalization of "Eckart-Young" decomposition," *Psychometrika*, vol. 35, no. 3, pp. 283–319, Sep 1970. [Online]. Available: https://doi.org/10.1007/BF02310791

[5] D. Chicco and M. Masseroli, "A discrete optimization approach for SVD best truncation choice based on ROC curves," in *13th IEEE International Conference on BioInformatics and BioEngineering*, Nov 2013, pp. 1–4.

[6] A. Cichocki, N. Lee, I. Oseledets, A.-H. Phan, Q. Zhao, and D. P. Mandic, "Tensor Networks for dimensionality reduction and large-scale optimization: Part 1 low-rank tensor decompositions," *Foundations and Trends in Machine Learning*, vol. 9, no. 4-5, pp. 249–429, 2016. [Online]. Available: http://dx.doi.org/10.1561/2200000059

[7] M. S. Grewal and A. P. Andrews, "Applications of Kalman filtering in aerospace 1960 to the present [historical perspectives]," *IEEE Control Systems Magazine*, vol. 30, no. 3, pp. 69–78, Jun 2010.

[8] S. Holtz, T. Rohwedder, and R. Schneider, "The alternating linear scheme for tensor optimization in the Tensor Train format," *SIAM J. Sci. Comput.*, vol. 34, no. 2, pp. 683–713, Mar. 2012. [Online]. Available: http://dx.doi.org/10.1137/100818893

[9] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the ASME–Journal of Basic Engineering*, vol. 82, no. Series D, pp. 35–45, 1960.

[10] P. Khatri, B. Done, A. Rao, A. Done, and S. Draghici, "A semantic analysis of the annotations of the human genome," *Bioinformatics*, vol. 21 16, pp. 3416–21, 2005.

[11] T. Kolda and B. Bader, "Tensor decompositions and applications," *SIAM Review*, vol. 51, no. 3, pp. 455–500, 2009. [Online]. Available: https://doi.org/10.1137/07070111X

[12] R. Orus, "A practical introduction to Tensor Networks: Matrix product states and projected entangled pair states," *Annals of Physics*, vol. 349, pp. 117 – 158, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0003491614001596

[13] I. Oseledets, "Tensor-Train decomposition," *SIAM Journal on Scientific Computing*, vol. 33, no. 5, pp. 2295–2317, 2011. [Online]. Available: https://doi.org/10.1137/090752286

[14] I. Oseledets, S. Dolgov, V. Kazeev, O. Savostyanov, D. Lebedeva, P. Zhlobich, T. Mach, and L. Song, "TT-toolbox," *Available online*, 2012. [Online]. Available: https://github.com/oseledets/TT-Toolbox

[15] I. Oseledets and E. Tyrtyshnikov, "Breaking the curse of dimensionality, or how to use SVD in many dimensions," *SIAM Journal on Scientific Computing*, vol. 31, no. 5, pp. 3744–3759, 2009. [Online]. Available: https://doi.org/10.1137/090748330

[16] ——, "TT-cross approximation for multidimensional arrays," *Linear Algebra Appl.*, vol. 432, no. 1, pp. 70–88, 2010.

[17] I. Oseledets, "Approximation of 2d*2d matrices using tensor decomposition," *SIAM J. Matrix Analysis Applications*, vol. 31, pp. 2130–2145, 2010.

[18] N. Vervliet, O. Debals, L. Sorber, M. Van Barel, and L. De Lathauwer, "Tensorlab v3.0," *Available online*, Mar 2016. [Online]. Available: http://www.tensorlab.net