
TENSOR NETWORK-CONSTRAINED KERNEL MACHINES AS GAUSSIAN PROCESSES

A PREPRINT

Frederiek Wesel

Delft Center for Systems and Control
Delft University of Technology
The Netherlands
f.wesel@tudelft.nl

Kim Batselier

Delft Center for Systems and Control
Delft University of Technology
The Netherlands
k.batselier@tudelft.nl

March 29, 2024

ABSTRACT

Tensor Networks (TNS) have recently been used to speed up kernel machines by constraining the model weights, yielding exponential computational and storage savings. In this paper we prove that the outputs of *Canonical Polyadic Decomposition* (CPD) and *Tensor Train* (TT)-constrained kernel machines recover a *Gaussian Process* (GP), which we fully characterize, when placing i.i.d. priors over their parameters. We analyze the convergence of both CPD and TT-constrained models, and show how TT yields models exhibiting more GP behavior compared to CPD, for the same number of model parameters. We empirically observe this behavior in two numerical experiments where we respectively analyze the convergence to the GP and the performance at prediction. We thereby establish a connection between TN-constrained kernel machines and GPs.

1 Introduction

Tensor Networks [TNS, Cichocki, 2014; Cichocki et al., 2016, 2017], a tool from multilinear algebra, extend the concept of rank from matrices to tensors allowing to represent an exponentially large object with a linear number of parameters. As such, TNS have been used to reduce the storage and computational complexities by compressing the model parameters of a range of models such as *Deep Neural Networks* (DNNs) [Novikov et al., 2015], *Convolutional Neural Networks* (CNNs) [Jaderberg et al., 2014; Lebedev et al., 2015], *Recurrent Neural Networks* (RNNs) [Ye et al., 2018], *Graph Neural Networks* (GNNs) [Hua et al., 2022] and transformers [Ma et al., 2019].

Similarly, TNS have also found application in the context of kernel machines [Stoudenmire and Schwab, 2016; Novikov et al., 2018; Wesel and Batselier, 2021]. Such models learn a multilinear (i.e. nonlinear) data-dependent representation from an exponentially large number of fixed features by means of a linear number of parameters, and are as such characterized by an implicit source of regularization. Furthermore, storage and the evaluation of the model and its gradient require a linear complexity in the number of parameters, rendering these methods promising candidates for applications requiring both good generalization and scalability. However their multilinearity precludes closed-form Bayesian inference, and has restricted the training of these models to the *maximum likelihood* (ML) and *maximum a posteriori* (MAP) framework.

In contrast, *Gaussian Processes* [GPs, Rasmussen and Williams, 2006] are an established framework for modeling functions which naturally allow the practitioner to incorporate prior knowledge. Importantly, when considering i.i.d. observations and Gaussian likelihoods, GPs allow for the determination of the posterior in *closed-form*, which considerably facilitates tasks such as inference, sampling and the construction of sparse approximations among many others. The main drawback of having a closed-form posterior lies however in their inability to autonomously learn features, which has arguably favored the use of deep learning models.

In this paper we establish a connection between TN-constrained kernel machines and GPs, thus solving an open problem considered by Wesel and Batselier [2021, 2023]. We prove that the outputs of *Canonical Polyadic Decomposition*

(CPD) and *Tensor Train* (TT)-constrained kernel machines converge to a fully characterized GP when specifying i.i.d. priors across their components. This result allows us to derive that for the same number of model parameters, TT-constrained models achieve faster convergence to the GP compared to their CPD counterparts and thus are more prone to exhibit GP behavior. We analyze the consequences of these findings in the context of MAP estimation and finally empirically observe GP convergence and GP behavior in two numerical experiments.

The rest of this paper is organized as follows. In section 2 we provide a brief introduction to GPs and their approximations, TNs and TN-constrained kernel machines. In section 3 we present our main result, i.e. the equivalence in the limit between TN-constrained kernel machines and GPs. In section 4 we discuss our numerical results which illustrate the found results. We then provide a review of related work (section 5) and a conclusion (section 6). We discuss the notation used throughout the paper in appendix A.

2 Background

GPs are a collection of random variables such that any finite subset has a joint Gaussian distribution [Rasmussen and Williams, 2006]. They provide a flexible formalism for modeling functions which inherently allows for the incorporation of prior knowledge and the production of uncertainty estimates in the form of a predictive distribution. More specifically, a GP is fully specified by a mean function $\mu \in \mathbb{R}$, typically chosen as zero, and a covariance or kernel function $k(\cdot, \cdot) : \mathbb{R}^D \times \mathbb{R}^D \rightarrow \mathbb{R}$:

$$f(\mathbf{x}) \sim \mathcal{GP}(\mu, k(\mathbf{x}, \cdot)).$$

Given a labeled dataset $\{(\mathbf{x}_n, y_n)\}_{n=1}^N$ consisting of inputs $\mathbf{x}_n \in \mathbb{R}^D$ and i.i.d. noisy observations $y_n \in \mathbb{R}$, GPs can be used for modeling the underlying function f in classification or regression tasks by specifying a likelihood function. For example the likelihood

$$p(y_n | f(\mathbf{x}_n)) = \mathcal{N}(f(\mathbf{x}_n), \sigma^2), \quad (1)$$

yields a GP posterior which can be obtained in closed-form by conditioning the prior GP on the noisy observations. Calculating the mean and covariance of such a posterior crucially requires instantiating and formally inverting the kernel matrix $\mathbf{K}$ such that $k_{n,m} := k(\mathbf{x}_n, \mathbf{x}_m)$. These operations respectively incur a computational cost of $\mathcal{O}(N^2)$ and $\mathcal{O}(N^3)$ and therefore prohibit the processing of large-sampled datasets.

2.1 Basis Function Approximation

The prevailing approach in literature to circumvent the $\mathcal{O}(N^3)$ computational bottleneck is to project the GP onto a finite number of *Basis Functions* (BFs) [e.g., Rasmussen and Williams, 2006; Quiñero-Candela and Rasmussen, 2005]. This is achieved by approximating the kernel as

$$k(\mathbf{x}, \mathbf{x}') \approx \boldsymbol{\varphi}(\mathbf{x})^T \boldsymbol{\Lambda} \boldsymbol{\varphi}(\mathbf{x}'), \quad (2)$$

where here $\boldsymbol{\varphi}(\mathbf{x}) : \mathbb{R}^D \rightarrow \mathbb{R}^M$ are (nonlinear) basis functions and $\boldsymbol{\Lambda} \in \mathbb{R}^{M \times M}$ are the BF weights. This finite-dimensional kernel approximation ensures a *degenerate* kernel [Rasmussen and Williams, 2006], as it characterized by a finite number of non-zero eigenvalues. Its associated GP can be characterized equivalently as

$$f(\mathbf{x}) = \langle \boldsymbol{\varphi}(\mathbf{x}), \mathbf{w} \rangle, \quad \mathbf{w} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Lambda}), \quad (3)$$

wherein $\mathbf{w} \in \mathbb{R}^M$ are the model weights and $\boldsymbol{\Lambda}$ is the associated prior covariance. Once more considering a Gaussian likelihood (equation (1)) yields a closed-form posterior GP whose mean and covariance require only the inversion of the matrix $\sum_{n=1}^N \boldsymbol{\varphi}(\mathbf{x}_n) \boldsymbol{\varphi}(\mathbf{x}_n)^T$. This yields a computational complexity of $\mathcal{O}(NM^2 + M^3)$, which allows to tackle large-sampled data when $N \gg M$.

2.2 Product Kernels

In the remainder of this paper we consider GPs with product kernels.

Definition 2.1 (Product kernel [Rasmussen and Williams, 2006]). A kernel $k(\mathbf{x}, \mathbf{x}')$ is a product kernel if

$$k(\mathbf{x}, \mathbf{x}') = \prod_{d=1}^D k^{(d)}(x_d, x'_d), \quad (4)$$

where each $k^{(d)}(\cdot, \cdot) : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$ is a valid kernel.

While many commonly used kernels are product kernels e.g. the Gaussian kernel and the polynomials kernel, product kernels provide a straightforward strategy to extend one-dimensional kernels to the higher-dimensional case [Rasmussen and Williams, 2006; Hensman et al., 2017]. The basis functions and prior covariance of product kernels can then be determined based on the basis function expansion of their constituents as follows.

Lemma 2.2 (Basis functions and prior covariances of product kernels). *Consider the product kernel of definition 2.1. Denote the basis functions and prior covariance of each factor $k^{(d)}(x_d, x'_d)$ as $\varphi^{(d)}(x_d) \in \mathbb{R}^{M_d}$ and $\Lambda^{(d)} \in \mathbb{R}^{M_d \times M_d}$ respectively, then the basis functions and prior covariance of $k(\mathbf{x}, \mathbf{x}')$ are*

$$\varphi(\mathbf{x}) = \otimes_{d=1}^D \varphi^{(d)}(x_d), \quad (5)$$

and

$$\Lambda = \otimes_{d=1}^D \Lambda^{(d)}, \quad (6)$$

The inherent challenge in this approach stems from the exponential increase of the number of basis functions M and thus of model parameters as a function of the dimensionality of the input data, thereby restricting their utility to low-dimensional datasets.

Such structure arises for instance when dealing with Mercer expansions of product kernels, in the structured kernel interpolation framework [Wilson and Nickisch, 2015; Yadav et al., 2021] variational Fourier features framework [Hensman et al., 2017] and Hilbert-GP framework [Solin and Särkkä, 2020]. Alternative important approximation strategies which avoid this exponential scaling are random features [Rahimi and Recht, 2007; Lázaro-Gredilla et al., 2010], inducing features [Csató and Opper, 2002; Seeger et al., 2003; Quiñero-Candela and Rasmussen, 2005; Snelson and Ghahramani, 2006; Hensman et al., 2013, 2015] and additive GPs [Duvenaud et al., 2011; Lu et al., 2022] which circumvent the outlined computational issue. All those approaches can be interpreted as projecting the GP on a set of BFs.

The performance of these methods however tends to deteriorate in higher dimensions, as they need to cover an exponentially large domain with a linear number of random samples or inducing points. These issues are some of the computational aspects of the *curse of dimensionality*, which renders it difficult to operate in high-dimensional feature spaces [Hastie et al., 2001].

2.3 Tensor Networks

A recent alternative approach to remedy said curse of dimensionality affecting the exponentially increasing weights of the linear model in equation (3) consists in constraining the models weights $\mathbf{w}$ to be a low-rank tensor network. TNS express a D -dimensional tensor $\mathcal{W}$ as a multi-linear function of a number of *core* tensors. Two commonly used TNS are the CPD and TT, defined as follows.

Definition 2.3 (CPD [Hitchcock, 1927]). A D -dimensional tensor $\mathcal{W} \in \mathbb{R}^{M_1 \times M_2 \times \dots \times M_D}$ has a rank- R CPD if

$$w_{m_1, m_2, \dots, m_D} = \sum_{r=1}^R \prod_{d=1}^D w^{(d)}_{m_d, r}. \quad (7)$$

The cores of a CPD are the matrices $\mathbf{W}^{(d)} \in \mathbb{R}^{M_d \times R}$. Since a CPD tensor can be expressed solely in terms of its cores, its storage requires $P_{\text{CPD}} = R \sum_{d=1}^D M_d$ parameters as opposed to $\prod_{d=1}^D M_d$.

Definition 2.4 (TT [Oseledets, 2011]). A D -dimensional tensor $\mathcal{W} \in \mathbb{R}^{M_1 \times M_2 \times \dots \times M_D}$ admits a rank- $(R_0 := 1, R_1, \dots, R_D := 1)$ tensor train if

$$w_{m_1, m_2, \dots, m_D} = \sum_{r_0=1}^{R_0} \sum_{r_1=1}^{R_1} \dots \sum_{r_D=1}^{R_D} \prod_{d=1}^D w^{(d)}_{r_{d-1}, m_d, r_d}. \quad (8)$$

The cores of a tensor train are D 3-dimensional tensors $\mathcal{W}^{(d)} \in \mathbb{R}^{R_{d-1} \times M_d \times R_d}$ which yield $P_{\text{TT}} = \sum_{d=1}^D R_{d-1} M_d R_d$ parameters.

In the following we denote by $\text{TN}(\mathcal{W})$ a tensor which admits a general TN format, by $\text{CPD}(\mathcal{W})$ a tensor which admits a rank- R CP form and by $\text{TT}(\mathcal{W})$ a tensor in rank- $(R_0 := 1, R_1, \dots, R_D := 1)$ TT form. Lastly, we denote by $R_1(\mathcal{W})$ a tensor which is in rank-1 CP form or rank- $(1, 1, \dots, 1)$ TT, as both are equivalent.

Importantly, we refer to a tensor in general TN format $\text{TN}(\mathcal{W}) \in \mathbb{R}^{M_1 \times M_2 \times \dots \times M_D}$ as *underparametrized* if its rank hyperparameters, e.g. R in case of CPD, are chosen such that its storage cost is less than $\prod_{d=1}^D M_d$. This is crucial in order to obtain storage, and as we will see, computational benefits.

2.4 Tensor Network-Constrained Kernel Machines

TNs have been used to reduce the number of model parameters in kernel machines (equation (3)) by tensorizing the BFs $\varphi(\mathbf{x})$ and model weights $\mathbf{w}$ and by constraining both to be underparameterized TNs. This approach lays its foundations on the fact that the Frobenius inner product of a tensorized vector is isometric with respect to the Euclidean inner product, i.e.

$$f(\mathbf{x}) = \langle \varphi(\mathbf{x}), \mathbf{w} \rangle = \langle \text{ten}(\varphi(\mathbf{x})), \text{ten}(\mathbf{w}) \rangle_{\text{F}}. \quad (9)$$

This isometry allows then to constrain the BFs and the model weights to be an underparameterized TN. Since product kernels yield an expansion in terms of Kronecker-product BFs (equation (5)), they are a rank-1 TN by definition after tensorization. Embedding these relations yields an approximate model

$$f(\mathbf{x}) \approx f_{\text{TN}}(\mathbf{x}) := \langle \mathbf{R}_1(\text{ten}(\varphi(\mathbf{x}))), \text{TN}(\text{ten}(\mathbf{w})) \rangle_{\text{F}}, \quad (10)$$

characterized by lower storage and computational complexities. This approach has been proposed mostly for weights modeled as CPD [Kargas and Sidiropoulos, 2021; Wesel and Batselier, 2021, 2023] or TT [Wahls et al., 2014; Stoudenmire and Schwab, 2016; Batselier et al., 2017; Novikov et al., 2018; Chen et al., 2018] as they arguably introduce fewer hyperparameters (only one in case of CPD) and thus are in practice easier to work with compared to other TNs such as the *Multi-Scale Entanglement Renormalization Ansatz* (MERA) [Reyes and Stoudenmire, 2021].

We define such models as we will need them in detail in the next section, where we present our main contribution.

Definition 2.5 (CPD-constrained kernel machine). The CPD-constrained kernel machine is defined as

$$f_{\text{CPD}}(\mathbf{x}) := \langle \mathbf{R}_1(\text{ten}(\varphi(\mathbf{x}))), \text{CPD}(\text{ten}(\mathbf{w})) \rangle_{\text{F}} \quad (11)$$

$$= \sum_{r=1}^R h_r(\mathbf{x}), \quad (12)$$

where the intermediate variables $h_r \in \mathbb{R}$ are defined as

$$h_r(\mathbf{x}) := \prod_{d=1}^D \varphi^{(d)}(x_d)^{\text{T}} \mathbf{w}^{(d)}_{:,r}. \quad (13)$$

Similarly, we provide a definition for the TT-constrained kernel machine.

Definition 2.6 (TT-constrained kernel machine). The TT-constrained kernel machine is defined as

$$f_{\text{TT}}(\mathbf{x}) := \langle \mathbf{R}_1(\text{ten}(\varphi(\mathbf{x}))), \text{TT}(\text{ten}(\mathbf{w})) \rangle_{\text{F}} \quad (14)$$

$$= \sum_{r_D=1}^{R_D} \sum_{r_{D-1}=1}^{R_{D-1}} \cdots \sum_{r_0=1}^{R_0} \prod_{d=1}^D z_{r_{d-1}, r_d}^{(d)}(x_d), \quad (15)$$

where the intermediate variables $\mathbf{Z}^{(d)} \in \mathbb{R}^{R_{d-1} \times R_d}$ are defined element-wise as

$$z_{r_{d-1}, r_d}^{(d)}(x_d) := \sum_{m_d=1}^{M_d} \varphi_{m_d}^{(d)}(x_d) w_{r_{d-1}, m_d, r_d}^{(d)}. \quad (16)$$

Evaluating CPD and TT-constrained kernel machines (equation (11), equation (14)) and their gradients can be accomplished with $\mathcal{O}(P_{\text{CPD}})$ and $\mathcal{O}(P_{\text{TT}})$ computations, respectively. This allows the practitioner to tune the rank hyperparameter in order to achieve a model that fits in the computational budget at hand and that learns from the specified BFs.

From an optimization point-of-view, models in the form of equation (10) have been trained both in the ML [Stoudenmire and Schwab, 2016; Batselier et al., 2017] and in the MAP setting [Wahls et al., 2014; Novikov et al., 2018; Chen et al., 2018; Kargas and Sidiropoulos, 2021; Wesel and Batselier, 2021, 2023] and in the context of GP variational inference [Izmailov et al., 2018] where TTs are used to parameterize the variational distribution. It is however not clear if and how these models relate to the weight-space GP equation (3).

In the following section we present the main contribution of our work: we show how when placing i.i.d. priors on the cores of these approximate models, they converge to a GP which we fully characterize.

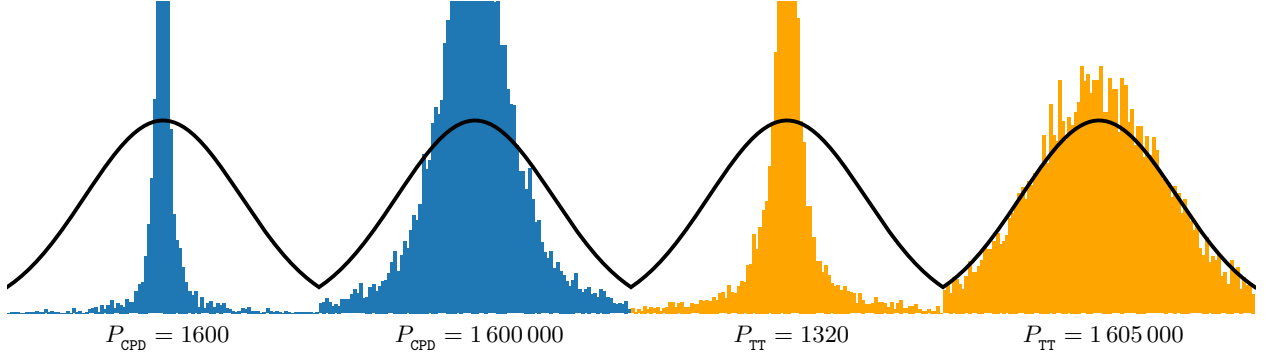


Figure 1: Histograms of the empirical PDF of CPD (blue) and TT (orange) models specified in theorem 3.1 and 3.2 evaluated at a random point as a function of model parameters P for $D = 16$. The black line is the PDF of the GP. Notice how TT converges faster to the GP for the same number of model parameters P .

3 TN-Constrained Kernel Machines as GPs

We commence to outline the correspondence between TN-constrained kernel machine and GPs, which makes use of the *Central Limit Theorem* (CLT). We begin by elucidating the simplest case, i.e. the CPD.

Theorem 3.1 (GP limit of CPD-constrained kernel machine). *Consider the CPD-constrained kernel machine*

$$f_{\text{CPD}}(\mathbf{x}) := \langle \mathbf{R}_1(\text{ten}(\varphi(\mathbf{x}))), \text{CPD}(\text{ten}(\mathbf{w})) \rangle_{\text{F}}.$$

If each of the R columns $\mathbf{w}_{:,r}^{(d)} \in \mathbb{R}^{M_d}$ of each CPD core is an i.i.d. random variable such that

$$\begin{aligned} \mathbb{E}[\mathbf{w}_{:,r}^{(d)}] &= \mathbf{0}, \\ \mathbb{E}[\mathbf{w}_{:,r}^{(d)} \mathbf{w}_{:,r}^{(d)\top}] &= R^{-\frac{1}{D}} \mathbf{\Lambda}^{(d)}, \end{aligned}$$

then $f_{\text{CPD}}(\mathbf{x})$ converges in distribution as $R \rightarrow \infty$ to the GP

$$f_{\text{CPD}}(\mathbf{x}) \sim \mathcal{GP}\left(0, \prod_{d=1}^D \varphi^{(d)}(x_d)^{\top} \mathbf{\Lambda}^{(d)} \varphi^{(d)}(\cdot)\right).$$

Proof. See appendix B.1. □

A similar result can be constructed for the TT case.

Theorem 3.2 (GP limit of TT-constrained kernel machine). *Consider the TT-constrained kernel machine*

$$f_{\text{TT}}(\mathbf{x}) := \langle \mathbf{R}_1(\text{ten}(\varphi(\mathbf{x}))), \text{TT}(\text{ten}(\mathbf{w})) \rangle_{\text{F}}$$

If each of the $R_{d-1}R_d$ fibers $\mathbf{W}_{r_{d-1},:,r_d}^{(d)} \in \mathbb{R}^{M_d}$ of each TT core is an i.i.d. random variable such that

$$\begin{aligned} \mathbb{E}[\mathbf{W}_{r_{d-1},:,r_d}^{(d)}] &= \mathbf{0}, \\ \mathbb{E}[\mathbf{W}_{r_{d-1},:,r_d}^{(d)} \mathbf{W}_{r_{d-1},:,r_d}^{(d)\top}] &= \frac{1}{\sqrt{R_{d-1}R_d}} \mathbf{\Lambda}^{(d)}, \end{aligned}$$

then $f_{\text{TT}}(\mathbf{x})$ converges in distribution as $R_1 \rightarrow \infty, R_2 \rightarrow \infty, \dots, R_{D-1} \rightarrow \infty$ to the Gaussian process

$$f_{\text{TT}}(\mathbf{x}) \sim \mathcal{GP}\left(0, \prod_{d=1}^D \varphi^{(d)}(x_d)^{\top} \mathbf{\Lambda}^{(d)} \varphi^{(d)}(\cdot)\right).$$

Proof. See appendix B.2. □

Theorem 3.2 guarantees the convergence in distribution of $f_{\text{TT}}(\mathbf{x})$ to the GP of equation (3) by taking successive limits of each TT rank. Importantly, the same convergence results also holds true if the TT ranks grow simultaneously, see appendix B.3.

Both theorem 3.1 and 3.2 are remarkable, as they imply that a degenerate GP which can be defined with an exponential number of $\prod_{d=1}^D M_d$ weights $\mathbf{w}$ can be also obtained with an *infinite* number of model parameters P using the CPD-constrained model of definition 2.5 or the TT-constrained model of definition 2.6. Further, theorem 3.1 and 3.2 suggest that CPD and TT-based models exhibit GP behavior in the overparameterized regime. GP behavior is characterized by a fixed learning representation, which in case of the kernel in theorem 3.1 and 3.2 is fully defined by the BFs and is hence data-independent. On the contrary, in the finite rank regime, both CPD and TT models are able to craft nonlinear features from the provided BFs, potentially learning more latent patterns in the data.

3.1 Convergence Rates to the GP

While both theorem 3.1 and theorem 3.2 guarantee convergence in distribution to the GP of equation (3), they do so at rates that differ in terms of the number of model parameters. Let us assume, for simplicity, that the number of basis functions is the same along each dimension, i.e., M , and that the $D-1$ TT ranks equal R . It follows then that the number of CPD model parameters $P_{\text{CPD}} = MDR_{\text{CPD}}$ and the number of TT model parameters $P_{\text{TT}} = M(D-2)R_{\text{TT}}^2 + 2MR_{\text{TT}} = \mathcal{O}(MDR_{\text{TT}}^2)$. Given the convergence rate of the CLT for the expression in equation (11) to the GP in equation (3), denoted as $\mathcal{O}(1/\sqrt{R_{\text{CPD}}})$ with respect to the variable P_{CPD} , we can establish the following corollary by substituting R_{CPD} as a function of P_{CPD} .

Corollary 3.3 (Convergence rate for CPD). *Under the conditions of theorem 3.1, the function $f_{\text{CPD}}(\mathbf{x})$ converges in distribution to the GP defined by equation (3). The convergence rate is given by:*

$$f_{\text{CPD}}(\mathbf{x}) \rightarrow \mathcal{O}\left(\left(\frac{MD}{P_{\text{CPD}}}\right)^{\frac{1}{2}}\right).$$

Due to their hierarchical structure, TT models are a composition of R_{TT}^{D-1} variables, but can be represented in a quadratic number of model parameters in R_{TT} , since $P_{\text{TT}} = \mathcal{O}(MDR_{\text{TT}}^2)$. Expressing then the CLT convergence rate of $\mathcal{O}(1/\sqrt{R_{\text{TT}}^{D-1}})$ as a function of P_{TT} yields the following corollary.

Corollary 3.4 (Convergence rate for TT). *Under the conditions of theorem 3.2, the function $f_{\text{TT}}(\mathbf{x})$ converges in distribution to the GP defined by equation (3). The convergence rate is given by:*

$$f_{\text{TT}}(\mathbf{x}) \rightarrow \mathcal{O}\left(\left(\frac{MD}{P_{\text{TT}}}\right)^{\frac{D-1}{4}}\right).$$

Therefore, when dealing with identical models in terms of the number of basis functions (M), dimensionality of the inputs (D), and the number of model parameters ($P_{\text{CPD}} = P_{\text{TT}}$), $f_{\text{TT}}(\mathbf{x})$ will converge at a polynomially faster rate than $f_{\text{CPD}}(\mathbf{x})$, thus exhibiting GP behavior with a reduced number of model parameters. In particular, based on corollaries 3.3 and 3.4 we expect the GP convergence rate of TT models to be faster for $D \geq 3$.

These insights are relevant for practitioners engaged with TN-constrained kernel machines, as they shed light on the balance between the GP and (deep) neural network behavior inherent in these models. Notably, CPD and TT-constrained models, akin to shallow and DNNs respectively, have the capacity to craft additional nonlinearities beyond the provided basis functions. This characteristic can result in superior generalization when dealing with a limited number of parameters. However, as the parameter count increases, we expect these models to transition towards GP behavior, characterized by a fixed feature representation and static in comparison.

3.2 Consequences for MAP Estimation

As discussed in section 2.4, TN-constrained kernel machines are typically trained in the ML or MAP framework by constraining the weights $\mathbf{w}$ in the log-likelihood or log-posterior to be a TN. In said MAP context, and e.g. when specifying a normal prior on the model weights $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Lambda})$, the resulting regularization term Ω is approximated by Ω_{TN} as

$$\Omega := \left\| \mathbf{\Lambda}^{-\frac{1}{2}} \mathbf{w} \right\|_{\text{F}}^2 \approx \Omega_{\text{TN}} := \left\| \text{TN}(\text{ten}(\mathbf{\Lambda}^{-\frac{1}{2}} \mathbf{w})) \right\|_{\text{F}}^2,$$

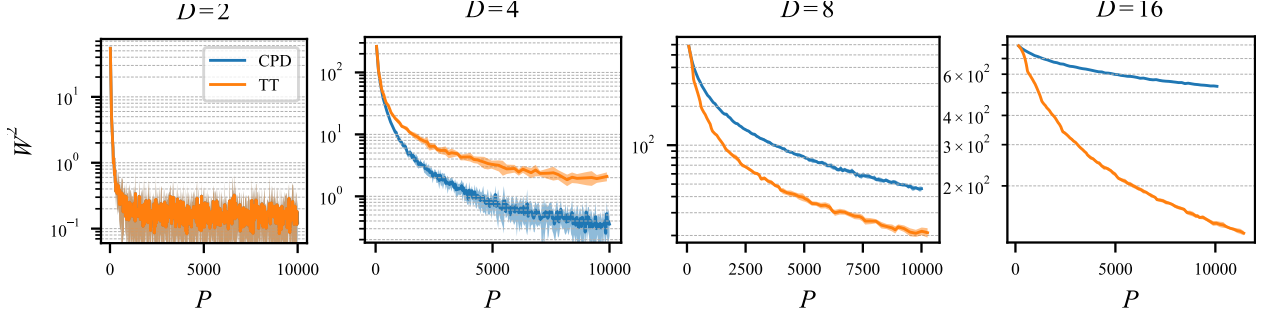


Figure 2: Mean and standard deviation of the Cramér-von Mises statistic W^2 evaluated between the empirical CDF of CPD and TT models specified in theorem 3.1 and 3.2 evaluated at $N = 10$ random points as a function of model parameters P for $D = 2, 4, 8, 16$. The two models are equivalent for $D = 2$. Notice how TT converges faster to the GP as the dimensionality of the inputs D increases.

where $\Lambda = \otimes_{d=1}^D \Lambda^{(d)}$. For example, in case of CPD-constrained models we have

$$\Omega_{\text{CPD}} = \left\| \odot_{d=1}^D \left(\mathbf{W}^{(d)\top} \Lambda^{(d)-1} \mathbf{W}^{(d)} \right) \right\|_{\text{F}}^2. \quad (17)$$

This form of regularization is considered for TT by Wahls et al. [2014]; Novikov et al. [2018]; Chen et al. [2018] and for CPD by Wesel and Batselier [2021, 2023]. It provides a Frobenius norm approximation of the regularization term which recovers the original MAP estimate as the hyperparameters of $\text{TN}(\text{ten}(\Lambda \mathbf{w}))$ are chosen such that $\text{TN}(\text{ten}(\Lambda \mathbf{w})) = \text{ten}(\Lambda \mathbf{w})$. If we now consider the regularization term in the log-posterior of theorem 3.1 we end up with

$$\Omega_{\text{CPD}} := R^{\frac{1}{D}} \sum_{d=1}^D \left\| \Lambda^{(d)-\frac{1}{2}} \mathbf{W}^{(d)} \right\|_{\text{F}}^2. \quad (18)$$

This regularization has been applied without the scaling factor $R^{\frac{1}{D}}$ and with $\Lambda^{(d)} = \mathbf{I}_{M_d}$, as observed in the work of Kargas and Sidiropoulos [2021], who may not have been aware of the underlying connection at that time. It does not account for the model interactions across the dimensionality due to the i.i.d. assumptions on the cores. Contrary to the regularization Ω_{TN} in equation (17), it provides an approximation which recovers the log-prior Ω and thus the MAP *in the limit*. These considerations point to the fact that if the practitioner is interested only in a MAP estimate which gives weight to the full prior $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \Lambda)$ as well as possible, he might be more interested in the regularization of equation (17). Furthermore, the priors in theorem 3.1 and 3.2 provide a sensible initial guess for gradient-based optimization which is invariant w.r.t. the dimensionality of the inputs and the choice of rank hyperparameters. We hence directly address the initialization issues affecting TN-constrained kernel machines [Barratt et al., 2021] by providing a sensible initialization strategy and regularization which does not suffer from vanishing or exploding gradients.

4 Numerical Experiments

We setup two numerical experiments in order to respectively empirically observe the claims in theorem 3.1 and 3.2 by evaluating the convergence to the prior GP in equation (3), and to evaluate the GP behavior of such models at prediction in the finite rank case. In all experiments we made use of the Hilbert-GP Solin and Särkkä [2020] to provide a BF expansion for the Gaussian kernel and opt for $M = 10$ basis functions per dimension. When sampling the cores of the models in theorem 3.1 and 3.2 we resorted to normally distributed random variables. We implemented both models in PyMC [Abril-Pla et al., 2023]. The fully anonymized Python implementation is available at github.com/fwesel/tensorGP.

4.1 GP Convergence

In order to empirically verify the convergence to the GP of equation (3) we sample 10 000 instances of the CPD and TT models specified in theorem 3.1 and 3.2 for increasing CPD and TT ranks yielding up to $P = 10\,000$ model parameters. Since the target distribution is Gaussian with known moments, we record the Cramér-von Mises statistic W^2 [D’Agostino and Stephens, 1986] which gives a metric of closeness between the target and our sampled empirical CDF. We repeat this for $N = 10$ randomly sampled data points and for $D = 2, 4, 8, 16$ and report the mean and standard deviation of the results in figure 2. Therein it can be observed that for the same number of model parameters,

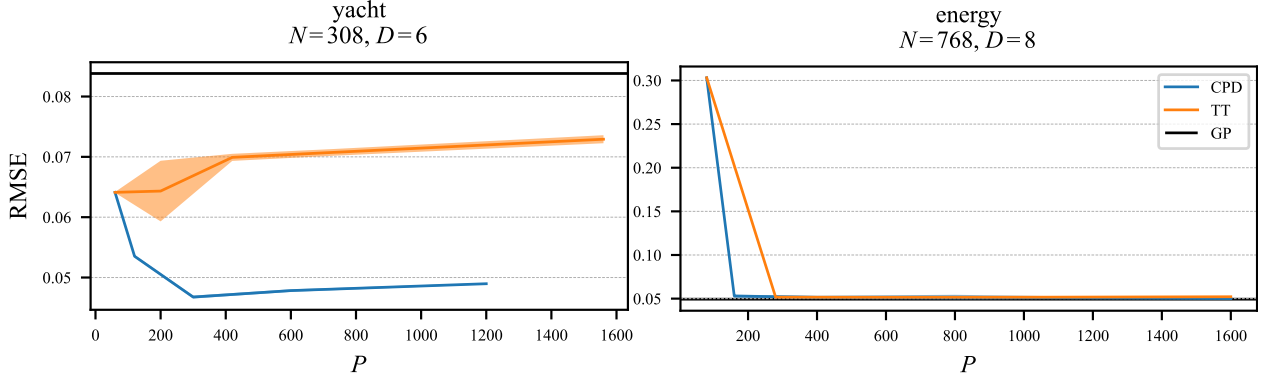


Figure 3: Mean and standard deviation of the test RMSE evaluated of CPD and TT models as a function of model parameters P as well as their target GP. Notice how in the *yacht* dataset TT exhibits more GP behavior compared to CPD as P increases. On the *energy* dataset both methods exhibit GP behavior already for ranks different than one, which explains why TT appears to be slower.

TT converges more rapidly than CPD as the dimensionality of the inputs grows. Both approaches however need exponentially more parameters to converge at the same rate for increasing dimensionality of the inputs. Note that for $D = 4$ CPD, contrary to what stated in section 3.1 still converges faster due to the approximation made when considering $P_{\text{TT}} = DMR^2$. Histograms of the empirical CDF for one datapoint are shown in figure 1. This behavior stems from the fact that for a fixed combination of D , M and P , TT captures an exponential R^{D-1} range of model interactions in contrast to the R linear interactions exhibited by CPD. This fact renders the choice of TT more suitable with respect to CPD if one wishes the model to exhibit more GP behavior, which is characterized by a weight-independent feature representation and is less likely to overfit.

4.2 GP Behavior at Prediction

To investigate whether CPD and TT-constrained kernel machines exhibit GP behavior as the number of parameters increases we tackle two small *UCI* regression problems, *yacht* and *energy* [Dua and Graff, 2017]. We consider 70 % of the data for training and the remaining for test and model our observations as having i.i.d. Gaussian likelihood (equation (1)). We then consider the GP in theorem 3.1 and 3.2 which we train by maximizing the marginalized likelihood (10 random initializations). In order to compare models with the target GP, we fix the obtained hyperparameters $\{\sigma, \Lambda\}$ and sample 4 chains of 2000 instances from the posteriors $p(\text{CPD}(\text{ten}(w)) | y)$ and $p(\text{TT}(\text{ten}(w)) | y)$ for a range of model parameters P using the *No U-Turn Sampling* Hamiltonian Monte Carlo scheme with default parameters. After discarding the first 1000 samples (burn-in), we obtain posterior predictive distributions $p(f_{\text{CPD}}(x) | y)$ and $p(f_{\text{TT}}(x) | y)$. We then compare the two models in terms of the *Root Mean Squared Error* (RMSE) of the posterior mean on the test data. We plot the mean and standard deviation of the RMSE over the chains in figure 3.

In figure 3 one can observe that the prediction of both CPD and TT models tends towards the GP as the number of model parameters increases. In case of the *yacht* dataset this happens with smaller errors, i.e. both models generalize better. As expected, the TT-constrained kernel machine exhibits more GP behavior and in this case worse generalization. In case of the *energy* dataset both methods have very similar performance compared to the GP and converge with larger test errors as soon as their ranks are different than one. While the behavior on both datasets can be explained in term of the GP underfitting on the *yacht* dataset, it is worth noting that both TT and CPD-based models perform at least as well as the GP for ranks higher than one, rendering them computationally advantageous alternatives to the GP in this context.

5 Related Work

Our contribution is closely tied to the links between Bayesian neural networks and GPs, first established for single-layer single-output neural networks [Neal, 1996a,b] having sigmoidal [Williams, 1996], Gaussian [Williams, 1997] and rectified linear unit [Cho and Saul, 2009] as activation function.

This idea was extended to DNNs by Lee et al. [2018] and Matthews et al. [2018] for the all the most common activation functions. It is important to note that the derivation of Lee et al. [2018] makes use of recursive application of the CLT

to each layer of the network, while [Matthews et al. \[2018\]](#) considers the case where the width of each layer grows simultaneously to the others, which is arguably more valuable in practice.

Further extensions have been proposed to CNNs where the number of channels tends to infinity [\[Novak et al., 2018; Garriga-Alonso et al., 2018\]](#), to RNNs [\[Sun et al., 2022\]](#) and to DNNs having low-rank constraints on the weight matrices [\[Nait-Saada et al., 2023\]](#).

In particular theorem 3.1 resembles the results of [Neal \[1996a,b\]](#); [Williams \[1997\]](#) which relate infinite-width single layer neural networks to GPs. The CPD rank corresponds exactly to the width of the neural network. The crucial difference lies however in the product structure, which is not present in neural networks and introduces a nonlinearity of different kind than the one induced by the activation function, which is linear in the CPD and more in general for any TN. TTs on the other hand resemble DNNs as they map the output of each core to the next one. However, in contrast to DNNs, the inputs are processed over the depth of the network. For a more in depth discussion we refer the reader to [\[Cohen et al., 2016\]](#).

Likewise theorem 3.2 is the TN counterpart to the works of [Lee et al. \[2018\]](#); [Matthews et al. \[2018\]](#) which relate finite depth neural networks to GPs. The TT ranks are then equivalent to the width of each layer and the activation function is linear. In contrast to the DNNs case, the induced GPs are degenerate.

6 Conclusion

In this paper we proved that CPD and TT-constrained kernel machines are GPs in the limit or large TN ranks. We characterized the target GP, analyzed the convergence behavior of both models and showed that compared to CPD, TT-based models converge faster to the GP when dealing with higher-dimensional inputs. We empirically demonstrated these properties by means of numerical experiments.

While the GP convergence is too slow to warrant these models as GP prior approximations, the insights we derived are useful in practice as they shed light on the effects of the choice of TN and regularization in these models.

References

- O. Abril-Pla, V. Andreani, C. Carroll, L. Dong, C. J. Fannesbeck, M. Kochurov, R. Kumar, J. Lao, C. C. Luhmann, O. A. Martin, M. Osthege, R. Vieira, T. Wiecki, and R. Zinkov. PyMC: A modern, and comprehensive probabilistic programming framework in Python. *PeerJ Computer Science*, 9:e1516, Sept. 2023. ISSN 2376-5992. doi: 10.7717/peerj-cs.1516.
- F. Barratt, J. Dborin, and L. Wright. Improvements to Gradient Descent Methods for Quantum Tensor Network Machine Learning. In *Second Workshop on Quantum Tensor Networks in Machine Learning*, May 2021.
- K. Batselier, Z. Chen, and N. Wong. Tensor Network alternating linear scheme for MIMO Volterra system identification. *Automatica*, 84:26–35, Oct. 2017. ISSN 0005-1098. doi: 10.1016/j.automatica.2017.06.033.
- Z. Chen, K. Batselier, J. A. K. Suykens, and N. Wong. Parallelized Tensor Train Learning of Polynomial Classifiers. *IEEE Transactions on Neural Networks and Learning Systems*, 29(10):4621–4632, Oct. 2018. ISSN 2162-2388. doi: 10.1109/TNNLS.2017.2771264.
- Y. Cho and L. Saul. Kernel Methods for Deep Learning. In *Advances in Neural Information Processing Systems*, volume 22. Curran Associates, Inc., 2009.
- A. Cichocki. Era of Big Data Processing: A New Approach via Tensor Networks and Tensor Decompositions. *arXiv:1403.2048 [cs]*, Aug. 2014.
- 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*, 9(4-5):249–429, 2016. ISSN 1935-8237, 1935-8245. doi: 10.1561/22000000059.
- A. Cichocki, A.-H. Phan, Q. Zhao, N. Lee, I. V. Oseledets, M. Sugiyama, and D. Mandic. Tensor Networks for Dimensionality Reduction and Large-Scale Optimizations. Part 2 Applications and Future Perspectives. *Foundations and Trends® in Machine Learning*, 9(6):249–429, 2017. ISSN 1935-8237, 1935-8245. doi: 10.1561/22000000067.
- N. Cohen, O. Sharir, and A. Shashua. On the Expressive Power of Deep Learning: A Tensor Analysis. In *Conference on Learning Theory*, pages 698–728. PMLR, June 2016.
- L. Csató and M. Oppor. Sparse On-Line Gaussian Processes. *Neural Computation*, 14(3):641–668, Mar. 2002. ISSN 0899-7667. doi: 10.1162/089976602317250933.
- R. B. D’Agostino and M. A. Stephens. *Goodness-of-Fit Techniques*. CRC Press, Jan. 1986.

- D. Dua and C. Graff. UCI Machine Learning Repository, 2017.
- D. K. Duvenaud, H. Nickisch, and C. Rasmussen. Additive Gaussian Processes. *Advances in Neural Information Processing Systems*, 24:226–234, 2011.
- A. Garriga-Alonso, C. E. Rasmussen, and L. Aitchison. Deep Convolutional Networks as shallow Gaussian Processes. In *International Conference on Learning Representations*, Sept. 2018.
- T. Hastie, J. Friedman, and R. Tibshirani. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer, New York, NY, 2001. ISBN 978-1-4899-0519-2 978-0-387-21606-5. doi: 10.1007/978-0-387-21606-5.
- J. Hensman, N. Fusi, and N. D. Lawrence. Gaussian processes for Big data. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence*, UAI’13, pages 282–290, Arlington, Virginia, USA, Aug. 2013. AUAI Press.
- J. Hensman, A. Matthews, and Z. Ghahramani. Scalable Variational Gaussian Process Classification. In *Artificial Intelligence and Statistics*, pages 351–360. PMLR, Feb. 2015.
- J. Hensman, N. Durrande, and A. Solin. Variational Fourier features for Gaussian processes. *The Journal of Machine Learning Research*, 18(1):5537–5588, Jan. 2017. ISSN 1532-4435.
- F. L. Hitchcock. The Expression of a Tensor or a Polyadic as a Sum of Products. *Journal of Mathematics and Physics*, 6(1-4):164–189, 1927. ISSN 1467-9590. doi: 10.1002/sapm192761164.
- C. Hua, G. Rabusseau, and J. Tang. High-Order Pooling for Graph Neural Networks with Tensor Decomposition. *Advances in Neural Information Processing Systems*, 35:6021–6033, Dec. 2022.
- P. Izmailov, A. Novikov, and D. Kropotov. Scalable Gaussian Processes with Billions of Inducing Inputs via Tensor Train Decomposition. In *International Conference on Artificial Intelligence and Statistics*, pages 726–735. PMLR, Mar. 2018.
- M. Jaderberg, A. Vedaldi, and A. Zisserman. Speeding up Convolutional Neural Networks with Low Rank Expansions. *Proceedings of the British Machine Vision Conference 2014*, 2014. doi: 10.5244/c.28.88.
- N. Kargas and N. D. Sidiropoulos. Supervised Learning and Canonical Decomposition of Multivariate Functions. *IEEE Transactions on Signal Processing*, pages 1–1, 2021. ISSN 1941-0476. doi: 10.1109/TSP.2021.3055000.
- M. Lázaro-Gredilla, J. Quiñero-Candela, C. E. Rasmussen, and b. R. Figueiras-Vidal. Sparse Spectrum Gaussian Process Regression. *Journal of Machine Learning Research*, 11(63):1865–1881, 2010. ISSN 1533-7928.
- V. Lebedev, Y. Ganin, M. Rakhuba, I. V. Oseledets, and V. S. Lempitsky. Speeding-up Convolutional Neural Networks Using Fine-tuned CP-Decomposition. In *International Conference on Learning Representations*, Jan. 2015.
- J. Lee, Y. Bahri, R. Novak, S. S. Schoenholz, J. Pennington, and J. Sohl-Dickstein. Deep Neural Networks as Gaussian Processes. In *International Conference on Learning Representations*, Feb. 2018.
- X. Lu, A. Boukouvalas, and J. Hensman. Additive Gaussian Processes Revisited. In *Proceedings of the 39th International Conference on Machine Learning*, pages 14358–14383. PMLR, June 2022.
- X. Ma, P. Zhang, S. Zhang, N. Duan, Y. Hou, M. Zhou, and D. Song. A Tensorized Transformer for Language Modeling. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.
- A. G. d. G. Matthews, M. Rowland, J. Hron, R. E. Turner, and Z. Ghahramani. Gaussian Process Behaviour in Wide Deep Neural Networks, Aug. 2018.
- T. Nait-Saada, A. Naderi, and J. Tanner. Beyond IID weights: Sparse and low-rank deep Neural Networks are also Gaussian Processes. In *The Twelfth International Conference on Learning Representations*, Oct. 2023. doi: 10.48550/arXiv.2310.16597.
- R. M. Neal. *Bayesian Learning for Neural Networks*. Springer Science & Business Media, Jan. 1996a. ISBN 978-1-4612-0745-0.
- R. M. Neal. Priors for Infinite Networks. In R. M. Neal, editor, *Bayesian Learning for Neural Networks*, Lecture Notes in Statistics, pages 29–53. Springer, New York, NY, 1996b. ISBN 978-1-4612-0745-0. doi: 10.1007/978-1-4612-0745-0.2.
- R. Novak, L. Xiao, Y. Bahri, J. Lee, G. Yang, J. Hron, D. A. Abolafia, J. Pennington, and J. Sohl-dickstein. Bayesian Deep Convolutional Networks with Many Channels are Gaussian Processes. In *International Conference on Learning Representations*, Sept. 2018.
- A. Novikov, D. Podoprikin, A. Osokin, and D. P. Vetrov. Tensorizing neural networks. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

- A. Novikov, I. Oseledets, and M. Trofimov. Exponential machines. *Bulletin of the Polish Academy of Sciences: Technical Sciences*; 2018; 66; No 6 (Special Section on Deep Learning: Theory and Practice); 789-797, 2018. ISSN 2300-1917.
- I. V. Oseledets. Tensor-Train Decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–2317, Jan. 2011. ISSN 1064-8275, 1095-7197. doi: 10.1137/090752286.
- J. Quiñonero-Candela and C. E. Rasmussen. A Unifying View of Sparse Approximate Gaussian Process Regression. *Journal of Machine Learning Research*, 6(65):1939–1959, 2005. ISSN 1533-7928.
- A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *Proceedings of the 20th International Conference on Neural Information Processing Systems, NIPS’07*, pages 1177–1184, Red Hook, NY, USA, Dec. 2007. Curran Associates Inc. ISBN 978-1-60560-352-0.
- C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, Mass, 2006. ISBN 978-0-262-18253-9.
- J. A. Reyes and E. M. Stoudenmire. Multi-scale tensor network architecture for machine learning. *Machine Learning: Science and Technology*, 2(3):035036, July 2021. ISSN 2632-2153. doi: 10.1088/2632-2153/abffe8.
- M. W. Seeger, C. K. I. Williams, and N. D. Lawrence. Fast Forward Selection to Speed Up Sparse Gaussian Process Regression. In *International Workshop on Artificial Intelligence and Statistics*, pages 254–261. PMLR, Jan. 2003.
- E. Snelson and Z. Ghahramani. Sparse Gaussian Processes using Pseudo-inputs. In *Advances in Neural Information Processing Systems*, volume 18. MIT Press, 2006.
- A. Solin and S. Särkkä. Hilbert space methods for reduced-rank Gaussian process regression. *Statistics and Computing*, 30(2):419–446, Mar. 2020. ISSN 1573-1375. doi: 10.1007/s11222-019-09886-w.
- E. M. Stoudenmire and D. J. Schwab. Supervised learning with tensor networks. In *Proceedings of the 30th International Conference on Neural Information Processing Systems, NIPS’16*, pages 4806–4814, Red Hook, NY, USA, Dec. 2016. Curran Associates Inc. ISBN 978-1-5108-3881-9.
- X. Sun, S. Kim, and J.-I. Choi. Recurrent neural network-induced Gaussian process. *Neurocomputing*, 509:75–84, Oct. 2022. ISSN 0925-2312. doi: 10.1016/j.neucom.2022.07.066.
- S. Wahls, V. Koivunen, H. V. Poor, and M. Verhaegen. Learning multidimensional Fourier series with tensor trains. In *2014 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 394–398, Dec. 2014. doi: 10.1109/GlobalSIP.2014.7032146.
- F. Wesel and K. Batselier. Large-Scale Learning with Fourier Features and Tensor Decompositions. In *Advances in Neural Information Processing Systems*, May 2021.
- F. Wesel and K. Batselier. Tensor-based Kernel Machines with Structured Inducing Points for Large and High-Dimensional Data. In *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, pages 8308–8320. PMLR, Apr. 2023.
- C. Williams. Computing with Infinite Networks. In *Advances in Neural Information Processing Systems*, volume 9. MIT Press, 1996.
- C. Williams. Computing with Infinite Networks. *Advances in Neural Information Processing Systems* 9, pages 295–301, 1997.
- A. Wilson and H. Nickisch. Kernel Interpolation for Scalable Structured Gaussian Processes (KISS-GP). In *Proceedings of the 32nd International Conference on Machine Learning*, pages 1775–1784. PMLR, June 2015.
- M. Yadav, D. Sheldon, and C. Musco. Faster Kernel Interpolation for Gaussian Processes. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, pages 2971–2979. PMLR, Mar. 2021.
- J. Ye, L. Wang, G. Li, D. Chen, S. Zhe, X. Chu, and Z. Xu. Learning Compact Recurrent Neural Networks With Block-Term Tensor Decomposition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9378–9387, 2018.

A Notation

Throughout this paper we denote scalars in both capital and non-capital italics w, W , vectors in non-capital bold $\mathbf{w}$, matrices in capital bold $\mathbf{W}$ and tensors in capital italic bold font $\mathcal{W}$. The m -th entry of a vector $\mathbf{w} \in \mathbb{R}^M$ is indicated as w_m and the $m_1 m_2 \dots m_D$ -th entry of a D -dimensional tensor $\mathcal{W} \in \mathbb{R}^{M_1 \times M_2 \times \dots \times M_D}$ as $w_{m_1, m_2, \dots, m_D}$. We employ the column notation to indicate a set of elements of tensor given a set of indices, e.g. $\mathbf{W}_{m_1, :, m_2}$ and $\mathbf{W}_{m_1, 1:3, m_2}$ represent respectively all elements and the first three elements along the second dimension of tensor $\mathcal{W}$ with fixed indices m_1 and m_2 . The Kronecker product is denoted by $\otimes$ and the Hadamard (elementwise) by $\odot$. We employ one-based indexing for all tensors. The Frobenius inner product between two D -dimensional tensors $\mathcal{V}, \mathcal{W} \in \mathbb{R}^{M_1 \times M_2 \times \dots \times M_D}$ is

$$\langle \mathcal{V}, \mathcal{W} \rangle_{\text{F}} := \sum_{m_1=1}^{M_1} \sum_{m_2=1}^{M_2} \dots \sum_{m_D=1}^{M_D} v_{m_1, m_2, \dots, m_D} w_{m_1, m_2, \dots, m_D},$$

and the Frobenius norm of $\mathcal{W} \in \mathbb{R}^{M_1 \times M_2 \times \dots \times M_D}$ is denoted and defined as

$$\|\mathcal{W}\|_{\text{F}}^2 := \langle \mathcal{W}, \mathcal{W} \rangle_{\text{F}}.$$

We define the vectorization operator as $\text{vec}(\cdot) : \mathbb{R}^{M_1 \times M_2 \times \dots \times M_D} \rightarrow \mathbb{R}^{M_1 M_2 \dots M_D}$ such that

$$\text{vec}(\mathcal{W})_m = w_{m_1, m_2, \dots, m_D},$$

with $m = m_1 + \sum_{d=2}^D (m_d - 1) \prod_{k=1}^{d-1} M_k$. Likewise, its inverse, the tensorization operator $\text{ten}(\cdot) : \mathbb{R}^{M_1 M_2 \dots M_D} \rightarrow \mathbb{R}^{M_1 \times M_2 \times \dots \times M_D}$ is defined such that

$$\text{ten}(\mathbf{w})_{m_1, m_2, \dots, m_D} = w_m.$$

B Proofs

B.1 GP of CPD-Constrained Kernel Machine

Theorem B.1 (GP limit of CPD-constrained kernel machine). *Consider the CPD-constrained kernel machine*

$$f_{\text{CPD}}(\mathbf{x}) := \langle \mathbf{R}_1(\text{ten}(\boldsymbol{\varphi}(\mathbf{x}))), \text{CPD}(\text{ten}(\mathbf{w})) \rangle_{\text{F}}.$$

If each of the R columns of each CPD core $\mathbf{w}^{(d)}_{:,r} \in \mathbb{R}^{M_d}$ is an i.i.d. random variable such that

$$\begin{aligned} \mathbb{E}[\mathbf{w}^{(d)}_{:,r}] &= 0, \\ \mathbb{E}[\mathbf{w}^{(d)}_{:,r} \mathbf{w}^{(d)\text{T}}_{:,r}] &= \frac{1}{R^{\frac{1}{D}}} \boldsymbol{\Lambda}^{(d)}, \end{aligned}$$

then $f_{\text{CPD}}(\mathbf{x})$ converges in distribution as $R \rightarrow \infty$ to the Gaussian process

$$f_{\text{CPD}}(\mathbf{x}) \sim \mathcal{GP} \left(0, \prod_{d=1}^D \boldsymbol{\varphi}^{(d)}(x_d)^{\text{T}} \boldsymbol{\Lambda}^{(d)} \boldsymbol{\varphi}^{(d)}(\cdot) \right).$$

Proof. Consider the R intermediate functions h_r of equation (13) which constitute the CPD-constrained model of equation (11). Due to the i.i.d. assumption on $\mathbf{w}^{(d)}_{:,r}$ each addend is the same function of i.i.d. random variables and thus is itself i.i.d.. The mean of each addend is

$$\mathbb{E}[h_r(\mathbf{x})] = \mathbb{E} \left[\prod_{d=1}^D \boldsymbol{\varphi}^{(d)}(x_d)^{\text{T}} \mathbf{w}^{(d)}_{:,r} \right] = 0, \quad (19)$$

due to the i.i.d assumption and the linearity of expectation. Its covariance is

$$\mathbb{E} [h_r(\mathbf{x})h_r(\mathbf{x}')] \quad (20a)$$

$$= \mathbb{E} \left[\prod_{d=1}^D \boldsymbol{\varphi}^{(d)}(x_d)^T \mathbf{w}^{(d)}_{:,r} \prod_{d=1}^D \boldsymbol{\varphi}^{(d)}(x'_d)^T \mathbf{w}^{(d)}_{:,r} \right] \quad (20b)$$

$$= \mathbb{E} \left[\prod_{d=1}^D \boldsymbol{\varphi}^{(d)}(x_d)^T \mathbf{w}^{(d)}_{:,r} \mathbf{w}^{(d)}_{:,r}^T \boldsymbol{\varphi}^{(d)}(x'_d) \right] \quad (20c)$$

$$= \prod_{d=1}^D \boldsymbol{\varphi}^{(d)}(x_d)^T \mathbb{E} \left[\mathbf{w}^{(d)}_{:,r} \mathbf{w}^{(d)}_{:,r}^T \right] \boldsymbol{\varphi}^{(d)}(x'_d) \quad (20d)$$

$$= \frac{1}{R} \prod_{d=1}^D \boldsymbol{\varphi}^{(d)}(x_d)^T \boldsymbol{\Lambda}^{(d)} \boldsymbol{\varphi}^{(d)}(x'_d).$$

Here the step from equation (20b) to equation (20c) exploits the fact that the transpose of a scalar is equal to itself, the step from equation (20c) to equation (20d) is due to the linearity of expectation. As the variances of each intermediate function h_r are appropriately scaled, by the multivariate central limit theorem $f_{\text{CPD}}(\mathbf{x})$ converges in distribution to a multivariate normal distribution, which is fully specified by its first two moments

$$\begin{aligned} \mathbb{E} [f_{\text{CPD}}(\mathbf{x})] &= 0, \\ \mathbb{E} [f_{\text{CPD}}(\mathbf{x})f_{\text{CPD}}(\mathbf{x}')] &= \prod_{d=1}^D \boldsymbol{\varphi}(x_d)^T \boldsymbol{\Lambda}^{(d)} \boldsymbol{\varphi}(x'_d). \end{aligned}$$

Since any finite collection of $\{f_{\text{CPD}}(\mathbf{x}), \dots, f_{\text{CPD}}(\mathbf{x}')\}$ will have a joint multivariate normal distribution with the aforementioned first two moments, we conclude that $f_{\text{CPD}}(\mathbf{x})$ is the Gaussian process

$$f_{\text{CPD}}(\mathbf{x}) \sim \mathcal{GP} \left(0, \prod_{d=1}^D \boldsymbol{\varphi}^{(d)}(x_d)^T \boldsymbol{\Lambda}^{(d)} \boldsymbol{\varphi}^{(d)}(\cdot) \right).$$

□

B.2 GP of TT-Constrained Kernel Machine in the Sequential Limit of the TT Ranks

Theorem B.2 (GP in the limit of TT-constrained kernel machine). *Consider the TT-constrained kernel machine*

$$f_{\text{TT}}(\mathbf{x}) := \langle \mathbf{R}_1(\text{ten}(\boldsymbol{\varphi}(\mathbf{x}))), \text{TT}(\text{ten}(\mathbf{w})) \rangle_{\text{F}}$$

If each of the $R_{d-1}R_d$ fibers of each TT core $\mathbf{W}^{(d)}_{r_{d-1},:,r_d} \in \mathbb{R}^{M_d}$ is an i.i.d. random variable such that

$$\begin{aligned} \mathbb{E} [\mathbf{W}^{(d)}_{r_{d-1},:,r_d}] &= \mathbf{0}, \\ \mathbb{E} [\mathbf{W}^{(d)}_{r_{d-1},:,r_d} \mathbf{W}^{(d)}_{r_{d-1},:,r_d}^T] &= \frac{1}{\sqrt{R_{d-1}R_d}} \boldsymbol{\Lambda}^{(d)}, \end{aligned}$$

then $f_{\text{TT}}(\mathbf{x})$ converges in distribution as $R_1 \rightarrow \infty, R_2 \rightarrow \infty, \dots, R_{D-1} \rightarrow \infty$ to the Gaussian process

$$f_{\text{TT}}(\mathbf{x}) \sim \mathcal{GP} \left(0, \prod_{d=1}^D \boldsymbol{\varphi}^{(d)}(x_d)^T \boldsymbol{\Lambda}^{(d)} \boldsymbol{\varphi}^{(d)}(\cdot) \right).$$

Proof. Define the vector of intermediate function $\mathbf{h}^{(d+1)} \in \mathbb{R}^{R_{d+1}}$ recursively as

$$h_{r_{d+1}}^{(d+1)} := \sum_{r_d=1}^{R_d} z_{r_d,r_{d+1}}^{(d+1)}(x_{d+1}) h_{r_d}^{(d)},$$

with $h^{(0)} := 1$. Note that the first two moments of intermediate variable $z_{r_d, r_{d+1}}^{(d+1)}(x_{d+1})$ are

$$\begin{aligned}\mathbb{E} \left[z_{r_d, r_{d+1}}^{(d+1)}(x_{d+1}) \right] &= 0, \\ \mathbb{E} \left[z_{r_d, r_{d+1}}^{(d+1)}(x_{d+1}) z_{r_d, r_{d+1}}^{(d+1)}(x'_{d+1}) \right] \\ &= \frac{1}{\sqrt{R_d R_{d+1}}} \boldsymbol{\varphi}^{(d)}(x_d)^\top \boldsymbol{\Lambda}^{(d)} \boldsymbol{\varphi}^{(d)}(x'_d)\end{aligned}$$

We proceed by induction. For the induction step suppose that $h_{r_d}^{(d)}$ is a GP, identical and independent for every r_d such that

$$h_{r_d}^{(d)} \sim \mathcal{GP} \left(0, \frac{1}{\sqrt{R_d}} \prod_{p=1}^d \boldsymbol{\varphi}^{(p)}(x_p)^\top \boldsymbol{\Lambda}^{(p)} \boldsymbol{\varphi}^{(p)}(\cdot) \right).$$

The scalar $h_{r_{d+1}}^{(d+1)}$ is the sum of R_d i.i.d. terms having mean

$$\mathbb{E} \left[h_{r_{d+1}}^{(d+1)} \right] = \mathbb{E} \left[z_{r_d, r_{d+1}}^{(d+1)}(x_{d+1}) h_{r_d}^{(d)} \right] = 0,$$

and covariance

$$\begin{aligned}\mathbb{E} \left[h_{r_{d+1}}^{(d+1)} h_{r_{d+1}}^{(d+1)} \right] \\ &= \mathbb{E} \left[z_{r_d, r_{d+1}}^{(d+1)}(x_{d+1}) h_{r_d}^{(d)} z_{r_d, r_{d+1}}^{(d+1)}(x'_{d+1}) h_{r_d}^{(d)} \right] \\ &= \mathbb{E} \left[z_{r_d, r_{d+1}}^{(d+1)}(x_{d+1}) z_{r_d, r_{d+1}}^{(d+1)}(x'_{d+1}) \right] \mathbb{E} \left[h_{r_d}^{(d)} h_{r_d}^{(d)} \right] \\ &= \frac{1}{\sqrt{R_{d+1}}} \prod_{p=1}^{d+1} \boldsymbol{\varphi}^{(p)}(x_p)^\top \boldsymbol{\Lambda}^{(p)} \boldsymbol{\varphi}^{(p)}(x'_p).\end{aligned}$$

Since the assumptions of the CLT are satisfied $h_{r_{d+1}}^{(d+1)}$ converges in distribution to the normal distribution, fully specified by the above mentioned first two moments. Since any finite collection of $\{h_{r_{d+1}}^{(d+1)}(\mathbf{x}_{1:d+1}), \dots, h_{r_{d+1}}^{(d+1)}(\mathbf{x}'_{1:d+1})\}$ will have a joint multivariate normal distribution with the aforementioned first two moments, we conclude that $h_{r_{d+1}}^{(d+1)}(\mathbf{x}_{1:d+1})$ is the GP

$$h_{r_{d+1}}^{(d+1)} \sim \mathcal{GP} \left(0, \frac{1}{\sqrt{R_{d+1}}} \prod_{p=1}^{d+1} \boldsymbol{\varphi}^{(p)}(x_p)^\top \boldsymbol{\Lambda}^{(p)} \boldsymbol{\varphi}^{(p)}(\cdot) \right).$$

For the base case, consider the R_1 outputs of the first hidden function $h_{r_1}^{(1)}$. They are i.i.d. with mean

$$\mathbb{E} \left[h_{r_1}^{(1)}(x_1) \right] = 0.$$

and covariance

$$\mathbb{E} \left[h_{r_1}^{(1)}(x_1) h_{r_1}^{(1)}(x'_1) \right] = \frac{1}{\sqrt{R_1}} \boldsymbol{\varphi}^{(1)}(x_1)^\top \boldsymbol{\Lambda}^{(1)} \boldsymbol{\varphi}^{(1)}(x_1).$$

We now consider the R_2 outputs of the second hidden function $h_{r_2}^{(2)}$

$$h_{r_2}^{(2)} = \sum_{r_1=1}^{R_1} z_{r_1, r_2}^{(2)}(x_2) h_{r_1}^{(1)},$$

which are i.i.d. as they are the same function of the R_1 i.i.d. outputs of $h_{r_1}^{(1)}(x_1)$. More specifically, their mean and covariance are

$$\begin{aligned}\mathbb{E} \left[h_{r_2}^{(2)} \right] &= 0, \\ \mathbb{E} \left[h_{r_2}^{(2)}(x_2) h_{r_2}^{(2)}(x'_2) \right] \\ &= \frac{1}{\sqrt{R_2}} \prod_{d=1}^2 \boldsymbol{\varphi}^{(d)}(x_d)^\top \boldsymbol{\Lambda}^{(d)} \boldsymbol{\varphi}^{(d)}(x_d).\end{aligned}$$

Once more by the CLT $h_{r_2}^{(2)}$ converges in distribution to the normal distribution with the above first two moments. Since any finite collection of $\{h_{r_2}^{(2)}(\mathbf{x}_{1:2}), \dots, h_{r_2}^{(2)}(\mathbf{x}'_{1:2})\}$ will have a joint multivariate normal distribution with the aforementioned first two moments, we conclude that $h_2^{(2)}(\mathbf{x}_{1:2})$ is the GP

$$h_{r_2}^{(2)} \sim \mathcal{GP} \left(0, \frac{1}{\sqrt{R_2}} \prod_{d=1}^2 \varphi^{(d)}(x_d)^\top \mathbf{\Lambda}^{(d)} \varphi^{(d)}(\cdot) \right),$$

which is our base case. Hence by induction $f_{\text{TT}}(\mathbf{x}) = h^{(D)}$ converges in distribution as $R_1 \rightarrow \infty, R_2 \rightarrow \infty, \dots, R_{D-1} \rightarrow \infty$ to the GP

$$f_{\text{TT}}(\mathbf{x}) \sim \mathcal{GP} \left(0, \prod_{d=1}^D \varphi^{(d)}(x_d)^\top \mathbf{\Lambda}^{(d)} \varphi^{(d)}(\cdot) \right).$$

□

B.3 GP of TT-Constrained Kernel Machine in the Simultaneous Limit of the TT Ranks

In theorem 3.2 we prove by induction that the TT-constrained kernel machine converges to a GP by taking successive limits of the TT ranks. This result is analogous to the work of [Lee et al. \[2018\]](#), who prove that for the DNNs, taking sequentially the limit of each layer. A more practically useful result consists in the convergence in the *simultaneous* limit of TT ranks.

In deep learning [Matthews et al. \[2018, theorem 4\]](#) prove convergence in the context of DNNs over the widths of all layers simultaneously. Said theorem has been employed to prove GP convergence in the context of convolutional neural networks [[Garriga-Alonso et al., 2018](#)] and in the context of DNNs where each weight matrix is of low rank [[Nait-Saada et al., 2023](#)].

Seeing the similarity between TT-constrained kernel machines (equation (14)) and DNNs and the technicality of the proof, similarly to [[Garriga-Alonso et al., 2018; Nait-Saada et al., 2023](#)] we draw a one-to-one map between the TT-constrained kernel machines and the DNNs considered in [Matthews et al. \[2018, theorem 4\]](#). Convergence in the simultaneous limit is then guaranteed by [Matthews et al. \[2018, theorem 4\]](#).

We begin by restating the definitions of linear envelope property, DNNs, linear envelope property and normal recursion as found in [Matthews et al. \[2018\]](#). To make the comparison easier for the reader, we change the indexing notation to match the one in this paper.

Definition B.3 (Linear envelope property for nonlinearities [[Matthews et al., 2018](#)]). A nonlinearity $t : \mathbb{R} \rightarrow \mathbb{R}$ is said to obey the linear envelope property if there exist $c, l \geq 0$ such that the following inequality holds

$$|t(u)| < c + l|u| \quad \forall u \in \mathbb{R}. \quad (25)$$

Definition B.4 (Fully connected DNN [[Matthews et al., 2018](#)]). A fully connected deep neural with one-dimensional output and inputs $\mathbf{x} \in \mathbb{R}^{R_0}$ is defined recursively such that the initial step is

$$h_{r_1}^{(1)}(\mathbf{x}) = \sum_{r_0=1}^{R_0} z_{r_1, r_0}^{(1)} x_{r_0} + b_{r_1}^{(1)}, \quad (26)$$

the activation step by nonlinear activation function t is given by

$$g_{r_d}^{(d)} = t(f_{r_d}^{(d)}), \quad (27)$$

and the subsequent layers are defined by the recursion

$$h_{r_{d+1}}^{(d+1)} = \sum_{r_d=1}^{R_d} z_{r_{d+1}, r_d}^{(d+1)} g_{r_d}^{(d)} + b_{r_{d+1}}^{d+1}, \quad (28)$$

so that $h^{(D)}$ is the output of the network. In the above, $\mathbf{Z}^{(d)} \in \mathbb{R}^{R_{d-1} \times R_d}$ and $\mathbf{b}^{(d)} \in \mathbb{R}^{R_d}$ are respectively the weights and biases of the d -th layer.

Definition B.5 (Width function [Matthews et al. \[2018\]](#)). For a given fixed input $n \in \mathbb{N}$, a width function $v^{(d)} : \mathbb{N} \rightarrow \mathbb{N}$ at depth d specifies the number of hidden units R_d at depth d .

Lemma B.6 (Normal recursion [Matthews et al., 2018]). *Consider $z_{r_{d-1}, r_d}^{(d)} \sim \mathcal{N}(0, C_w^{(d)})$ and $b_{r_d}^{(d)} \sim \mathcal{N}(0, C_b^{(d)})$. If the activations of the d -th layer are normally distributed with moments*

$$\mathbb{E} [h_{r_d}^{(d)}] = 0 \quad (29)$$

$$\mathbb{E} [h_{r_d}^{(d)} h_{r_d}^{(d)}] = K(x, x'), \quad (30)$$

then under recursion equations (27) and (28), as $R_{d-1} \rightarrow \infty$, the activations of the next layer converge in distribution to a normal distribution with moments

$$\mathbb{E} [h_{r_{d+1}}^{(d+1)}] = 0 \quad (31)$$

$$\mathbb{E} [h_{r_{d+1}}^{(d+1)} h_{r_{d+1}}^{(d+1)}] = C_w^{(d+1)} \mathbb{E}_{(\epsilon_1, \epsilon_2) \sim \mathcal{N}(0, K)} [t(\epsilon_1) t(\epsilon_2)] + C_b^{(d+1)}. \quad (32)$$

We can now state the major result in Matthews et al. [2018].

Theorem B.7 (GP in the simultaneous limit of fully connected DNNs [Matthews et al., 2018]). *Consider a random DNN of the form of definition B.4 obeying the linear envelope condition of definition B.3. Then for all sets of strictly increasing width functions $v^{(d)}$ and for any countable input set $\{\mathbf{x}, \dots, \mathbf{x}'\}$, the distribution of the output of the network converges in distribution to a GP as $n \rightarrow \infty$. The GP has mean and covariance functions given by the recursion in lemma B.6.*

Corollary B.8 (GP in the simultaneous limit of TT-constrained kernel machines). *Consider a random TT-constrained kernel machine of the form of definition 2.6 obeying the linear envelope condition of definition B.3. Then for all sets of strictly increasing width functions $v^{(d)}$ and for any countable input set $\{\mathbf{x}, \dots, \mathbf{x}'\}$, the distribution of the output of the network converges in distribution to a GP as $P \rightarrow \infty$. The GP has mean and covariance functions given by the recursion in lemma B.6 and stated in theorem 3.2.*

Proof. When examining definition B.4 and comparing it with definition 2.6 it becomes clear that both models are similar. In the special case of involving linear activation function and zero biases, the models are structurally identical if one considers unit inputs $x = 1$ in equation (26). The normal recursion in lemma B.6 is satisfied by TT-constrained kernel machines, as we have that

$$\begin{aligned} t(u) &:= u \quad \forall u \in \mathbb{R}, \\ C_b^{(d+1)} &:= 0, \\ C^{(d+1)} &:= \frac{1}{\sqrt{R_d R_{d+1}}} \boldsymbol{\varphi}^{(d)}(x_d)^T \boldsymbol{\Lambda}^{(d)} \boldsymbol{\varphi}^{(d)}(x'_d), \\ K &:= \frac{1}{\sqrt{R_d}} \prod_{p=1}^d \boldsymbol{\varphi}^{(p)}(x_p)^T \boldsymbol{\Lambda}^{(p)} \boldsymbol{\varphi}^{(p)}(x'_p) \\ \mathbb{E}_{(\epsilon_1, \epsilon_2) \sim \mathcal{N}(0, K)} [t(\epsilon_1) t(\epsilon_2)] &:= K. \end{aligned}$$

Hence by theorem B.7, for all sets of strictly increasing width functions $v^{(d)}$ and for any countable input set $\{\mathbf{x}, \dots, \mathbf{x}'\}$, the distribution of the output of the network converges in distribution to a GP, fully specified by the output of the normal recursion in lemma B.6, which equals the GP in theorem 3.2. $\square$