Properties and optimization of compositional tensor networks

Charles Miranda [1] Martin Eigel [2] Anthony Nouy [1] David Sommer [2]

[1]Centrale Nantes, Nantes Université, Laboratoire de Mathématiques Jean Leray UMR CNRS 6629, France

[2]Weierstraß Institute for Applied Analysis and Stochastics, Berlin, Germany

April 1, 2026

Context

We aim to *approximate a multivariate function* $f \in L^2_\mu(\mathcal{X})$, which is equivalent to solving the least-squares problem

$$\min_{u \in \mathcal{M}} \mathcal{L}(u) := \frac{1}{2} \|u - f\|^2_{L^2_\mu} \tag{1}$$

Quest for a flexible approximation class $\mathcal{M}$ similar to neural networks, and with a *robust learning algorithm.*

Table of contents

What are tensors?

Let $(V_1, \ldots, V_d)$ be finite dimensional $\mathbb{R}$-vector spaces with bases $(\phi_{i_\nu}^{\nu})_{i_\nu=1}^{n_\nu}$, $\nu = 1 \ldots d$.
A function f in $V_1 \otimes \cdots \otimes V_d$ admits a representation

$$f(x_1, \ldots, x_d) = \sum_{j_1, \ldots, j_d} C_{j_1, \ldots, j_d} (\phi_{j_1}^1 \otimes \cdots \otimes \phi_{j_d}^d)(x_1, \ldots, x_d).$$

f can be identified with $C \in \mathbb{R}^{n_1 \times \cdots \times n_d}$.

Canonical format

All linear combinations of r elementary tensors

$$\mathcal{CP}_r := \left\{ \sum_{\nu=1}^{r} v_\nu^{(1)} \otimes \cdots \otimes v_\nu^{(d)} : v_\nu^{(j)} \in V_j \right\}.$$

(Canonical) tensor rank

The *(canonical) tensor rank* of $v \in \bigotimes_{j=1}^{d} V_j$ is defined by

$$\operatorname{rank}(v) := \min\{r : v \in \mathcal{CP}_r\} \in \mathbb{N}_0.$$

- In general, the computation of rank $\boldsymbol{v}$ is NP-hard[1]
- $\mathcal{CP}_{\leq r}$ is not closed.

[1] Håstad: "Tensor rank is NP-complete", 1990.

Tensor-Train format

For a tuple $\boldsymbol{r} = (r_0, r_1, \ldots, r_d) \in \mathbb{N}_0^{d+1}$, with $r_0 = r_d = 1$ by convention, an element $v \in \mathcal{TT}_{\leq \boldsymbol{r}}$ admits a representation

$$v(\boldsymbol{x}) = \sum_{\alpha_1=1,\ldots,\alpha_{d-1}=1}^{r_1,\ldots,r_d} G_1(1, x_1, \alpha_1) G_2(\alpha_1, x_2, \alpha_2) \ldots G_d(\alpha_{d-1}, x_d, 1)$$

Tensor-Train ranks

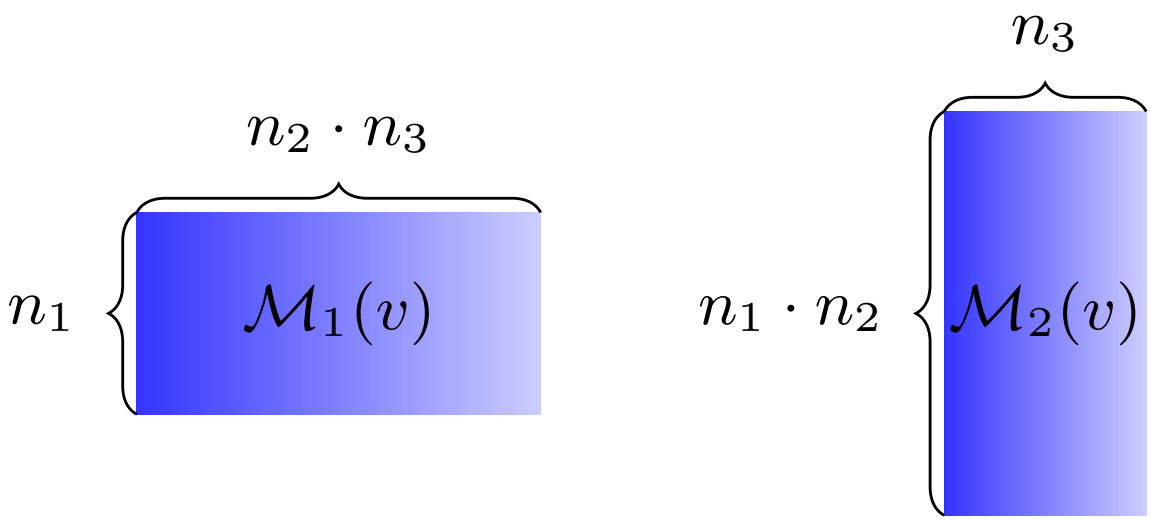
The *tensor-train ranks*[2] of $v \in \bigotimes_{j=1}^{d} V_j$ are defined by

$$\mathrm{rank}_k(v) = \mathrm{rank}(\mathcal{M}_k(v)),$$

with $\mathcal{M}_k : \bigotimes_{j=1}^{d} V_j \to \left(\bigotimes_{j=1}^{k} V_j\right) \otimes \left(\bigotimes_{j=k+1}^{d} V_j\right)$ a so-called *matricisation* operator.

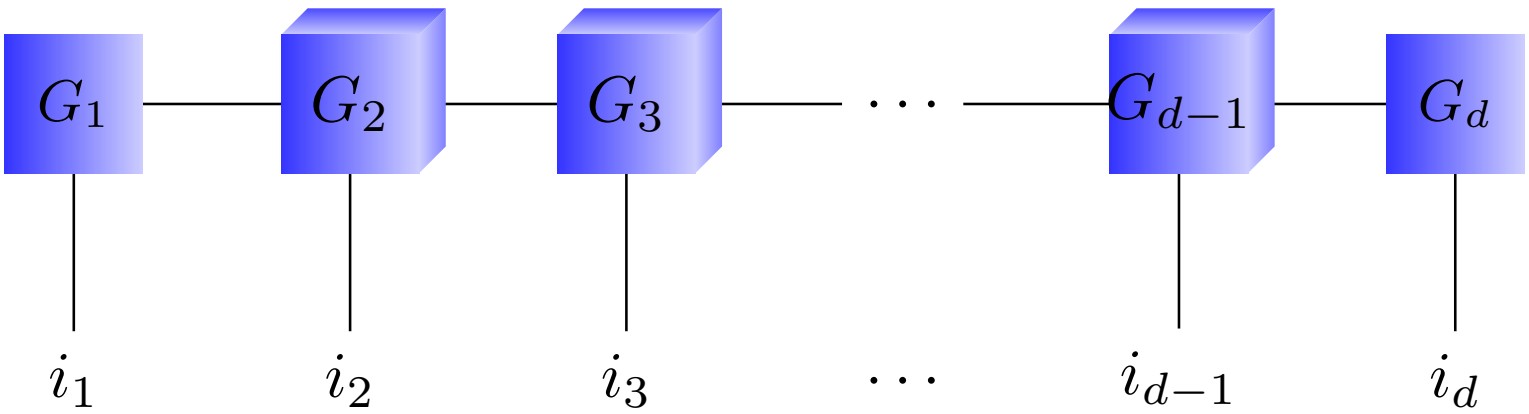
[2] Oseledets: "Tensor-Train Decomposition", 2011.

Tensor-Train format



Matricisation of a 3D tensor

Tensor-Train format



Graphical representation of a Tensor Train

Tensor-Train format

- $\mathcal{TT}_{\leq \boldsymbol{r}}$ is *closed*,
- There exists a polynomial time algorithm that computes a quasi-best approximation of a tensor $\boldsymbol{v}$ by an element of $\mathcal{TT}_{\leq \boldsymbol{r}}$: $\|v - v_{\boldsymbol{r}}\| \leq C \min_{w \in \mathcal{TT}_{\leq \boldsymbol{r}}} \|v - w\|$,
- The storage complexity is in $\mathcal{O}(dnr^2)$.

Table of contents

Composition of tensors in the litterature

- Tree-based tensors can be viewed as a composition of multilinear functions[2],
- Constructing a transport map as a composition of TTs using inverse Rosenblatt transport[3],
- Relation to deep neural networks, potential role they can have in representing solutions to PDEs[4].

---

[2] Michel and Nouy: "Learning with tree tensor networks: Complexity estimates and model selection", 2022.

[3] Cui and Dolgov: "Deep Composition of Tensor-Trains Using Squared Inverse Rosenblatt Transports", 2022.

[4] Schneider and Oster: "Some Thoughts on Compositional Tensor Networks", 2024.

Motivation

Consider a discrete time Markov process X whose density is given by

$$f(x) = f_{5|4}(x_5|x_4)f_{4|3}(x_4|x_3)f_{3|2}(x_3|x_2)f_{2|1}(x_2|x_1)f_1(x_1)$$

with ranks $\boldsymbol{r} = (1, m, m, m, m, 1)$. Let $\sigma = (1, 3, 5, 2, 4)$ be a permutation and $\tilde{f} = f \circ \sigma^{-1}$. Then $\tilde{f}$ has TT ranks

$$\tilde{\boldsymbol{r}} = (1, m^3, m^4, m^3, 1).$$

Towards composition of tensors

Compositional tensors

Given linear operators $\mathfrak{L} : \mathbb{R}^{d_i} \to \mathbb{R}^d$ and $\mathfrak{R} : \mathbb{R}^d \to \mathbb{R}^{d_o}$, called respectively *lift* and *retraction*, we call a function $v : \mathbb{R}^{d_i} \to \mathbb{R}^{d_o}$ a *compositional tensor* with L layers and basis $\Phi = (\phi_j)_{j=1}^n$, if there exist tensors $\psi_1, \ldots, \psi_L : \mathbb{R}^d \to \mathbb{R}^d$, each with the same univariate basis Φ, such that

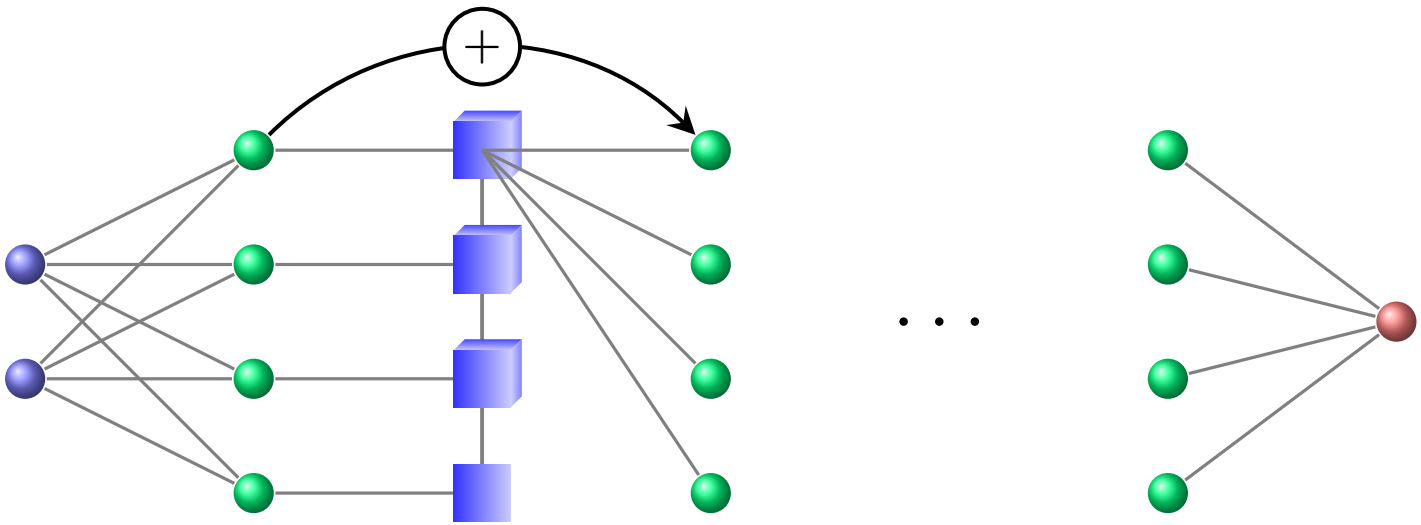
$$v(x) = \mathfrak{R} \circ (\mathrm{Id} + \psi_L) \circ \cdots \circ (\mathrm{Id} + \psi_1) \circ \mathfrak{L}(x).$$

The set of such functions with be denoted $\mathcal{CT}^L(\Phi)$.

One can assume low-rank format for the tensors ψ_ℓ e.g. tensor-train format, and in this case we will denote the corresponding set by $\mathcal{CTT}^L_{\leq \boldsymbol{r}}(\Phi)$.

Connections with deep neural networks

- Related to DNNs with low-rank decomposition[5][6][7],
- CTT is similar to a DNN but with *latent spaces which are tensorized spaces*, and with *low-rank layers*.



Graphical representation of a CTT

[5] Lebedev et al.: "Speeding-up Convolutional Neural Networks Using Fine-tuned CP-Decomposition", 2014.

[6] Zniyed et al.: "A tensor-based approach for training flexible neural networks", 2021.

[7] Hu et al.: "LoRA: Low-Rank Adaptation of Large Language Models", 2021.

Encoding of classical function classes

Theorem (Affine transformation)

Any affine transformation $h \mapsto Ah + b$ can be represented by a single CTT layer with TT ranks at most $\text{rank}(A)$.

- Can encode all the *permutations*,
- Can represent *functions of the form* $u(Ax)$, with u in TT format, with 2 CTT layers,
- The density f of the discrete time Markov process can be exactly represented by *2 CTT layers with non-polynomially growing ranks*.

Encoding of classical function classes

Theorem (Multivariate polynomials)

Let $P(x) = \sum_{\alpha \in \Lambda} C_\alpha x^\alpha$ with maximal degree N, and $q \geq 2$. Then P can be represented by a CTT with basis $\Phi = \{1, \mathrm{Id}\}$, $L = \mathcal{O}(d|\Lambda|\lfloor \log_q(N+1) \rfloor)$ layers and $\mathcal{O}(L(d+q))$ parameters. Moreover the TT ranks are bounded by $(d+2+q, 2, \ldots, 2)$.

Theorem (Neural networks)

Let σ be representable by a CTT with L_σ layers and m_σ parameters, and $f = T_L \circ \sigma \circ T_{L-1} \circ \sigma \circ \cdots \circ \sigma \circ T_1$ be a neural network of width p_f. Then f can be represented by a CTT with basis containing $\{1, \mathrm{Id}\}$, $L + (L-1)L_\sigma$ layers and $\mathrm{size}(f) + (L-1)p_f m_\sigma$ non-zero parameters.

Universal approximation

Corollary (Universality in $C(\Omega)$ and $L^q_\mu(\mathbb{R}^d)$)

The set of CTTs, with $\{1, \mathrm{Id}\} \subseteq \Phi$ and ranks $\boldsymbol{r} \geq (d+1, 2, \ldots, 2)$, is dense in $(C(\Omega), \|\cdot\|_\infty)$, Ω compact, and in $(L^q_\mu(\mathbb{R}^d), \|\cdot\|_{L^q_\mu})$, with $1 \leq q < \infty$ and μ a regular Borel finite measure.

Corollary (Approximation spaces)

Consider the set of DNNs and CTTs with fixed width and variable number of layers with corresponding approximation tools $(\Sigma_n^{DNN})_n$ and $(\Sigma_n^{CTT})_n$. Then

$$A^\alpha_\infty(X, \Sigma^{DNN}) \subseteq A^\alpha_\infty(X, \Sigma^{CTT})$$

where $A^\alpha_\infty(X, \Sigma) = \{f \in X : \sup_{n \geq 1} n^\alpha E(f, \Sigma_{n-1})_X < \infty\}$.

Table of contents

Gradient descent

Let's go back to our minimization problem

$$\min_{u \in \mathcal{M}} \mathcal{L}(u).$$

Assume that $\mathcal{M}$ is parametrized by a function $P \in C^1(\Theta, \mathcal{H})$, with $\Theta \subseteq \mathbb{R}^p$. Let $L := \mathcal{L} \circ P : \Theta \to \mathcal{H}$. It is well-known that the *steepest descent* is given locally by,

$$\theta_{k+1} = \arg\min_{h \in \Theta} L(\theta_k) + \langle \nabla L(\theta_k), h \rangle + \frac{1}{2\alpha_k} \|h - \theta_k\|_2^2$$

Gradient descent intialized randomly converges *almost surely* to local minimizers[8].

[8] Lee et al.: "Gradient descent only converges to minimizers", 2016.

Taking into account the geometry of the model class

The idea behind *AdaGrad*[9], *AdaDelta*[10] and *Adam*[11] is

- dynamically incorporates *knowledge of the geometry of the model class* observed in earlier iterations,
- determines automatically *a learning rate for each parameter* using first-order information

For example, for AdaGrad, the updates are given by,

$$\theta_{t+1} = \theta_t - \alpha \,\mathrm{diag}(\widehat{G}_t)^{-1/2}\widehat{g}_t, \quad \widehat{G}_t = \sum_{\tau=1}^{t} \widehat{g}_\tau \widehat{g}_\tau{}^\top,$$

where $\widehat{g}_\tau$ is an unbiased estimator of the true gradient $\nabla_\theta L(\theta_\tau)$.

---

[9] Duchi, Hazan, and Singer: "Adaptive subgradient methods for online learning and stochastic optimization.", 2011.

[10] Zeiler: "Adadelta: an adaptive learning rate method", 2012.

[11] Kingma and Ba: "Adam: A method for stochastic optimization", 2014.

Natural gradient descent

We consider the gradient descent in the Hilbert space $\mathcal{H} = L^2_\mu$,

$$\tilde{u}_{k+1} = u_k + \alpha_k d_k, \quad d_k = -\nabla \mathcal{L}(u_k).$$

Natural gradient descent

We aim to design an algorithm in the parameter space,

$$\theta_{k+1} = \theta_k + \alpha_k w_k$$

such that $u_{k+1} := P(\theta_{k+1}) \approx \tilde{u}_{k+1}$. Therefore, by Taylor's theorem on $P(\theta_{k+1})$, we would like that $DP(\theta_k)[w_k] \approx d_k$.

$$w_k \in \arg\min_{w \in \Theta} \frac{1}{2} \| DP(\theta_k)[w] - d_k \|^2_{L^2_\mu},$$

The *natural gradient* is given by,

$$w_k = -G(\theta_k)^\dagger \nabla L(\theta_k), \quad G(\theta)_{ij} = \langle \partial_{\theta_i} P(\theta), \partial_{\theta_j} P(\theta) \rangle.$$

Natural gradient is *reparametrization invariant*[12].

[12] Oostrum, Müller, and Ay: "Invariance properties of the natural gradient in overparametrised systems", 2022.

Natural gradient descent for CTT

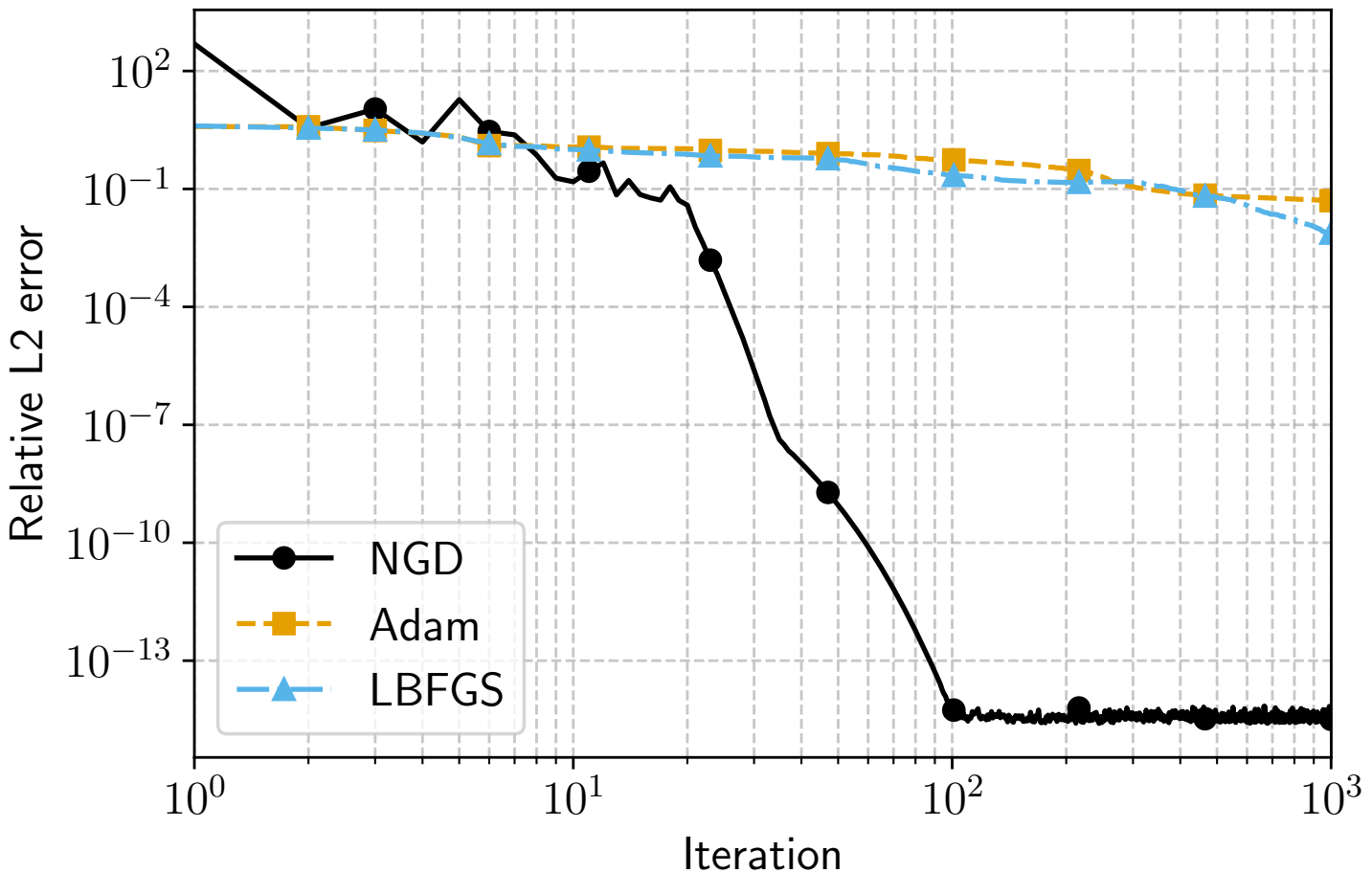
We want to exploit low-rank tensor format in the update of the layers, therefore we consider an *update scheme on the layers* ψ_k.
One can show that for each $1 \leq k \leq K$, we have,

$$\partial_{\psi_k} v(x) = \frac{\partial v}{\partial w_k}(w_{k-1}(x)) \otimes \Phi^{\otimes d}(w_{k-1}(x)) \in \mathbb{R}^{d_o \times d} \otimes (\mathbb{R}^n)^{\otimes d},$$

where $w_k = (\mathrm{Id} + \psi_k) \circ \cdots \circ (\mathrm{Id} + \psi_1) \circ L$.
The inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ is approximated by Monte-Carlo.
The *natural gradient can be computed efficiently* by exploiting tensor algebra, using for example ALS with conditioning!

Toy example

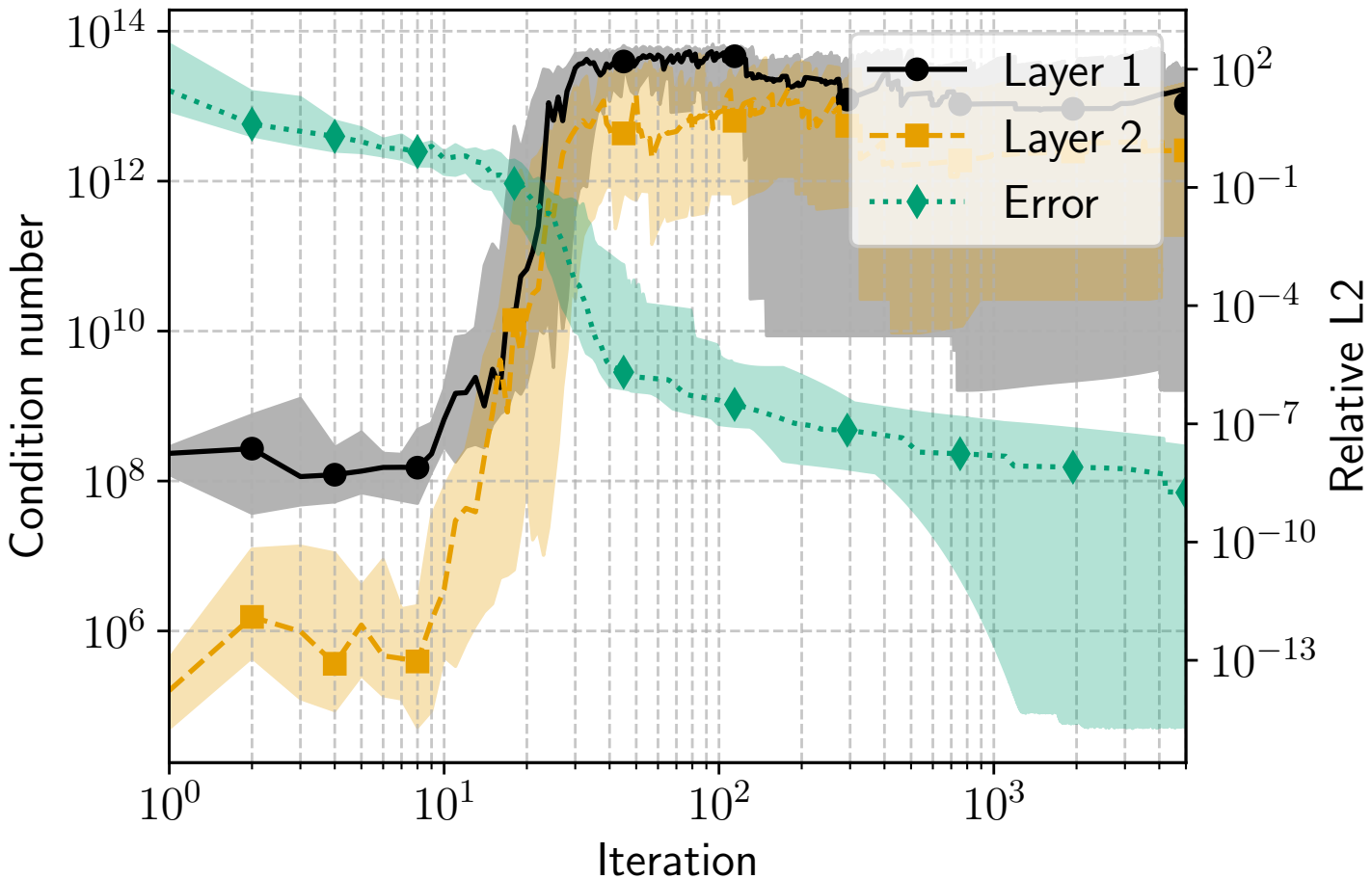
$$u(x) = (x_1 - x_3)^2(x_2 - x_4)^2, \quad \mathcal{X} = [0,1]^d, \quad d = 4, \quad K = 2$$



Evolution of the condition number

We track the *effective condition number*

$$\kappa_\ell(\theta) := \|G_\ell(\theta)\|_2 \|G_\ell(\theta)^\dagger\|_2$$



Regularization

We propose to project the Gram matrix G_ℓ onto a *randomly generated low-dimensional subspace*. We introduce a tensor-structured sketch $S \in \mathbb{R}^{s \times d \times n \times \cdots \times n}$, and compute the projections $G_\ell S_j \in \mathbb{R}^{d \times n \times \cdots \times n}$.

- These *projections can be computed efficiently exploiting tensor algebra, and without forming explicitly* G_ℓ.
- With high probability, the span of the *sketch captures the dominant eigenspace of* G_ℓ,
- Dominant eigendirections are not necessarily important directions in the sense of natural gradient.

Effect of the random sketching on the convergence

The behavior of the random sketching on the convergence is shown for various sketching sizes s.

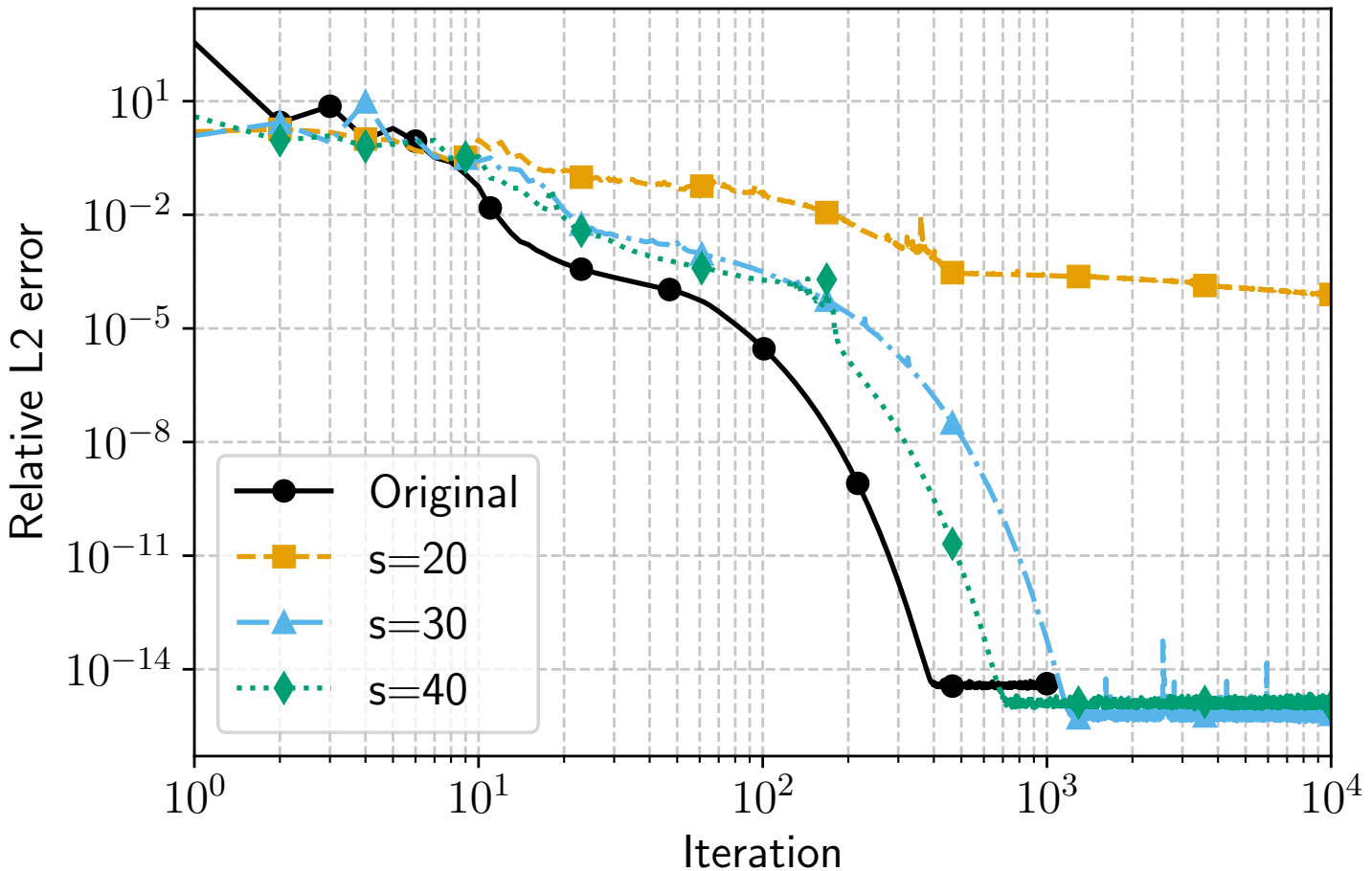


Table of contents

Open questions

- Can we design a *random sketching that captures the important directions in the sense of natural gradient*?
- Can we include *optimal sampling* (e.g. Christoffel sampling or DPP) in an active learning setting?

References I

[CD22] Tiangang Cui and Sergey Dolgov. "Deep Composition of Tensor-Trains Using Squared Inverse Rosenblatt Transports". In: *Foundations of Computational Mathematics* 22.6 (Dec. 2022), pp. 1863–1922.

[DHS11] John Duchi, Elad Hazan, and Yoram Singer. "Adaptive subgradient methods for online learning and stochastic optimization.". In: *Journal of machine learning research* 12.7 (2011).

[Hås90] Johan Håstad. "Tensor rank is NP-complete". In: *Journal of Algorithms* 11.4 (Dec. 1990), pp. 644–654.

[Hu+21] J. Edward Hu et al. "LoRA: Low-Rank Adaptation of Large Language Models". In: *ArXiv* abs/2106.09685 (2021).

References II

[KB14] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (2014).

[Leb+14] Vadim Lebedev et al. "Speeding-up Convolutional Neural Networks Using Fine-tuned CP-Decomposition". In: *CoRR* abs/1412.6553 (2014).

[Lee+16] Jason D Lee et al. "Gradient descent only converges to minimizers". In: *Conference on learning theory*. PMLR. 2016, pp. 1246–1257.

[MN22] Bertrand Michel and Anthony Nouy. "Learning with tree tensor networks: Complexity estimates and model selection". In: *Bernoulli* 28.2 (May 2022).

References III

[OMA22] Jesse van Oostrum, Johannes Müller, and Nihat Ay. "Invariance properties of the natural gradient in overparametrised systems". In: *Information Geometry* 6.1 (June 2022), pp. 51–67.

[Ose11] I. V. Oseledets. "Tensor-Train Decomposition". In: *SIAM Journal on Scientific Computing* 33.5 (Jan. 2011), pp. 2295–2317.

[SO24] Reinhold Schneider and Mathias Oster. "Some Thoughts on Compositional Tensor Networks". In: *Multiscale, Nonlinear and Adaptive Approximation II*. Springer Nature Switzerland, 2024, pp. 419–447.

[Zei12] Matthew D Zeiler. "Adadelta: an adaptive learning rate method". In: *arXiv preprint arXiv:1212.5701* (2012).

References IV

[Zni+21] Yassine Zniyed et al. "A tensor-based approach for training flexible neural networks". In: *2021 55th Asilomar Conference on Signals, Systems, and Computers*. IEEE, Oct. 2021, pp. 1673–1677.