Journal of Machine Learning Research 23 (2022) 1-56 Submitted 1/22; Revised 07/22; Published 10/22

Existence, Stability and Scalability of Orthogonal Convolutional Neural Networks

El Mehdi Achour EL_MEHDI.ACHOUR@MATH.UNIV-TOULOUSE.FR
Institut de Mathématiques de Toulouse ; UMR 5219
Université de Toulouse ; CNRS
UPS IMT F-31062 Toulouse Cedex 9, France

François Malgouyres FRANCOIS.MALGOUYRES@MATH.UNIV-TOULOUSE.FR
Institut de Mathématiques de Toulouse ; UMR 5219
Université de Toulouse ; CNRS
UPS IMT F-31062 Toulouse Cedex 9, France

Franck Mamalet FRANCK.MAMALET@IRT-SAINTEXUPERY.COM
Institut de Recherche Technologique Saint Exupéry, Toulouse, France

Editor: Kilian Weinberger

Abstract

Imposing orthogonality on the layers of neural networks is known to facilitate the learning by limiting the exploding/vanishing of the gradient; decorrelate the features; improve the robustness. This paper studies the theoretical properties of orthogonal convolutional layers.

We establish necessary and sufficient conditions on the layer architecture guaranteeing the existence of an orthogonal convolutional transform. The conditions prove that orthogonal convolutional transforms exist for almost all architectures used in practice for 'circular' padding. We also exhibit limitations with 'valid' boundary conditions and 'same' boundary conditions with zero-padding.

Recently, a regularization term imposing the orthogonality of convolutional layers has been proposed, and impressive empirical results have been obtained in different applications (Wang et al., 2020). The second motivation of the present paper is to specify the theory behind this. We make the link between this regularization term and orthogonality measures. In doing so, we show that this regularization strategy is stable with respect to numerical and optimization errors and that, in the presence of small errors and when the size of the signal/image is large, the convolutional layers remain close to isometric.

The theoretical results are confirmed with experiments and the landscape of the regularization term is studied. Experiments on real data sets show that when orthogonality is used to enforce robustness, the parameter multiplying the regularization term can be used to tune a tradeoff between accuracy and orthogonality, for the benefit of both accuracy and robustness.

Altogether, the study guarantees that the regularization proposed in Wang et al. (2020) is an efficient, flexible and stable numerical strategy to learn orthogonal convolutional layers.

Keywords: Convolutional layers, orthogonality, deep learning theory, vanishing/exploding gradient, robustness

1. Introduction

We first start by introducing the problem, related work and the context of this paper.

1.1 On Orthogonal Convolutional Neural Networks

Orthogonality constraint has first been considered for fully connected neural networks (Arjovsky et al., 2016). For Convolutional Neural Networks (LeCun and Bengio, 1995; Krizhevsky et al., 2012; Zhang et al., 2015), the introduction of the orthogonality constraint is a way to improve the neural network in several regards. First, despite well-established solutions (He et al., 2016; Ioffe and Szegedy, 2015), the training of very deep convolutional networks remains difficult. This is in particular due to vanishing/exploding gradient problems (Hochreiter, 1991; Bengio et al., 1994). As a result, the expressive capacity of convolutional layers is not fully exploited (Ioffe and Szegedy, 2015). This can lead to lower performances on machine learning tasks. Also, the absence of constraint on the convolutional layer often leads to irregular predictions that are prone to adversarial attacks (Szegedy et al., 2014; Nguyen et al., 2015). Gradient vanishing/exploding avoidance, built-in robustness and better generalization capabilities are the main aims of the introduction of Lipschitz (Szegedy et al., 2014; Qian and Wegman, 2018; Gouk et al., 2021; Tsuzuku et al., 2018; Sedghi et al., 2018) and orthogonality constraints to convolutional layers (Xie et al., 2017; Cisse et al., 2017; Huang et al., 2018; Zhang et al., 2019; Li et al., 2019b; Guo and Ye, 2020; Qi et al., 2020; Wang et al., 2020; Trockman and Kolter, 2021; Jia et al., 2019; Li et al., 2019a; Huang et al., 2020; Jia et al., 2019; Bansal et al., 2018; Xiao et al., 2018). Orthogonal convolutional networks have been applied successfully in diverse applications, such as classification, segmentation, inpainting (Wang et al., 2020; Zhang et al., 2020; Larrazabal et al., 2021), or recently in few-shot learning (Osahor and Nasrabadi, 2022). Orthogonality is also proposed for Generative Adversarial Networks (GAN)(Miyato et al., 2018), or even required for Wasserstein distance estimation, such as in Wasserstein-GAN (Arjovsky and Bottou, 2017; Gulrajani et al., 2017), and Optimal Transport based classifier (Serrurier et al., 2021).

Orthogonal convolutional networks are made of several orthogonal convolutional layers. This means that, when expressing the computation performed by the layer as a matrix, the matrix is orthogonal. The term 'orthogonal' applies both to square and non-square matrices[1]. In the latter case, it includes two commonly distinguished but related notions: row-orthogonality and column-orthogonality. This article focuses on the theoretical properties of orthogonal convolutional layers. Furthermore, since deconvolution (also called transposed convolution) layers are defined using convolution layers, the results can also be applied to orthogonal deconvolution layers. We will consider the architecture of a convolutional layer as characterized by (M, C, k, S), where M is the number of output channels, C of input channels, convolution kernels are of size $k \times k$ and the stride parameter is S. Unless we specify otherwise, we consider convolutions with circular boundary conditions[2]. Thus, applied on input channels of size $SN \times SN$, the M output channels are of size $N \times N$. We denote by $\mathbf{K} \in \mathbb{R}^{M \times C \times k \times k}$ the kernel tensor and by $\mathcal{K} \in \mathbb{R}^{MN^2 \times CS^2N^2}$ the matrix that applies the convolutional layer of architecture (M, C, k, S) to C vectorized channels of size $SN \times SN$.

We will first answer the important questions:

- **Existence:** What is a necessary and sufficient condition on (M, C, k, S) and N such that there exists an orthogonal convolutional layer (i.e. $\mathcal{K}$ orthogonal) for this architecture? How do the 'valid' and 'same' boundary conditions restrict the orthogonality existence?

Answers to these questions are respectively in Section 2.1, Theorem 4 and Section 2.2, Proposition 5 and Proposition 6.

1. The same property is sometimes called 'semi-orthogonal'.
2. Before computing a convolution the input channels are made periodic outside their genuine support.

Besides, we will rely on recently published papers (Wang et al., 2020; Qi et al., 2020) which characterize orthogonal convolutional layers as the zero level set of a particular function that is called L_{orth} in Wang et al. (2020)[3] (see Section 1.3.2 for details). Formally, $\mathcal{K}$ is orthogonal if and only if $L_{orth}(\mathbf{K}) = 0$. They use L_{orth} as a regularization term and obtain impressive performances on several machine learning tasks (see Wang et al., 2020). The regularization is later successfully applied for medical image segmentation (Zhang et al., 2020), inpainting (Larrazabal et al., 2021) and few-shot learning (Osahor and Nasrabadi, 2022).

In the present paper, we investigate the following theoretical questions:

- **Stability with regard to minimization errors:** Does $\mathcal{K}$ still have good 'approximate orthogonality properties' when $L_{orth}(\mathbf{K})$ is small but non zero? Without this guarantee, it could happen that $L_{orth}(\mathbf{K}) = 10^{-9}$ and $\|\mathcal{K}\mathcal{K}^T - Id\|_2 = 10^9$. This would make the regularization with L_{orth} useless, unless the algorithm reaches $L_{orth}(\mathbf{K}) = 0$.

- **Scalability and stability with regard to N:** Remarking that, for a given kernel tensor $\mathbf{K}$, $L_{orth}(\mathbf{K})$ is independent of N but the layer transform matrix $\mathcal{K}$ depends on N: When $L_{orth}(\mathbf{K})$ is small, does $\mathcal{K}$ remain approximately orthogonal and isometric when N grows? If so, the regularization with L_{orth} remains efficient even for very large N.

- **Optimization:** Does the landscape of L_{orth} lend itself to global optimization?

We give a positive answer to these questions, thus showing theoretical bounds proving that the regularization with L_{orth} is stable (see Theorem 7, Theorem 8 and Section 3.1.3), and can be used in most cases to ensure quasi-orthogonality of the convolutional layers (see Section 3.1.1 and Section 3.1.2).

We describe the related works in Section 1.2 and give the main elements of context in Section 1.3. The theorems constituting the main contributions of the article are in Section 2. Experiments illustrating the theorems, on the landscape of L_{orth}, as well as experiments showing the benefits of approximate orthogonality on image classification problems are in Section 3. In particular, the latter shows that when orthogonality is used to enforce robustness, the regularization parameter λ multiplying $L_{orth}(\mathbf{K})$ can be used to tune a tradeoff between accuracy and orthogonality, for the benefit of both accuracy and robustness. The code will be made available in the *DEEL.LIP* [4] library.

For clarity, we only consider convolutional layers applied to images (2D) in the introduction and the experiments. But we emphasize that the theorems in Section 2 and their proofs are provided for both signals (1D) and images (2D).

1.2 Related Work and Contributions

Orthogonal matrices form the Stiefel Manifold and were studied in Edelman et al. (1998). In particular, the Stiefel Manifold is compact, smooth and of known dimension. It is made of several connected components. This can be a numerical issue since most algorithms have difficulty changing connected components during optimization. The Stiefel Manifold has many other nice properties that make it suitable for (local) Riemannian optimization (Lezcano-Casado and Martınez-Rubio, 2019; Li et al., 2019a). Orthogonal convolutional layers are a subpart of this Stiefel Manifold. To the best

3. The situation is more complex in (Wang et al., 2020; Qi et al., 2020). One of the contributions of the present paper is to clarify the situation. We describe here the clarified statement.
4. `https://github.com/deel-ai/deel-lip`

of our knowledge, the understanding of orthogonal convolutional layers is weak. There is no paper focusing on the theoretical properties of orthogonal convolutional layers.

Many articles (Xu and Mannor, 2012; Cisse et al., 2017; Sokolić et al., 2017; Jia et al., 2019; Virmaux and Scaman, 2018; Gouk et al., 2021; Farnia et al., 2018) focus on Lipschitz and orthogonality constraints of the neural network layers from a statistical point of view, in particular in the context of adversarial attacks.

Many recent papers have investigated the numerical problem of optimizing a kernel tensor $\mathbf{K}$ under the constraint that $\mathcal{K}$ is orthogonal or approximately orthogonal. They also provide modeling arguments and experiments in favor of this constraint. We can distinguish two main strategies: **kernel orthogonality** (Xie et al., 2017; Cisse et al., 2017; Huang et al., 2018; Zhang et al., 2019; Guo and Ye, 2020; Jia et al., 2019; Li et al., 2019a; Huang et al., 2020; Jia et al., 2019; Bansal et al., 2018; Serrurier et al., 2021) and **convolutional layer orthogonality** (Li et al., 2019b; Qi et al., 2020; Wang et al., 2020; Trockman and Kolter, 2021; Kiani et al., 2022). The latter has been introduced more recently.

We denote the input of the layer by $X \in \mathbb{R}^{C\times SN\times SN}$ and its output by $Y = \mathbf{conv}(\mathbf{K}, X) \in \mathbb{R}^{M\times N\times N}$.

- **Kernel Orthogonality:** This class of methods views the convolution as a multiplication between a matrix $\overline{\mathbf{K}} \in \mathbb{R}^{M\times Ck^2}$ formed by reshaping the kernel tensor $\mathbf{K}$ (see, for instance, Cisse et al., 2017; Wang et al., 2020, for more details) and the matrix $U(X) \in \mathbb{R}^{Ck^2\times N^2}$ whose columns contain the concatenation of the C vectorized patches of X needed to compute the M output channels at a given spatial position (see Heide et al., 2015; Yanai et al., 2016). We therefore have, $\mathrm{Vect}\,(Y) = \mathrm{Vect}\,\left(\overline{\mathbf{K}}U(X)\right)$. The kernel orthogonality strategy enforces the orthogonality of the matrix $\overline{\mathbf{K}}$.

- **Convolutional Layer Orthogonality:** This class of methods connects the input and the output of the layer directly by writing $\mathrm{Vect}\,(Y) = \mathcal{K}\,\mathrm{Vect}\,(X)$ and enforces the orthogonality of $\mathcal{K}$. The difficulty of this method is that the size of the matrix $\mathcal{K} \in \mathbb{R}^{MN^2\times CS^2N^2}$ depends on N and can be very large.

Kernel orthogonality provides a numerical strategy whose complexity is independent of N. However, kernel orthogonality does not imply that $\mathcal{K}$ is orthogonal. In a nutshell, the problem is that the composition of an orthogonal embedding[5] and an orthogonal dimensionality reduction has no reason to be orthogonal. This phenomenon has been observed empirically in Li et al. (2019b) and Jia et al. (2019). The authors of Wang et al. (2020) and Qi et al. (2020) also argue that, when $\mathcal{K}$ has more columns than rows (row orthogonality), the orthogonality of $\overline{\mathbf{K}}$ is necessary but not sufficient to guarantee $\mathcal{K}$ orthogonal. Kernel orthogonality and convolutional layer orthogonality are different, the latter better avoids gradient vanishing and feature correlation.

We can distinguish between two numerical ways of enforcing orthogonality during training:

- **Hard Orthogonality:** This method consists in keeping the matrix of interest orthogonal during the whole training process. This can be done either by optimizing on the Stiefel Manifold, or by considering a parameterization of a subset of orthogonal matrices (e.g., Li et al., 2019a,b; Trockman and Kolter, 2021; Singla and Feizi, 2021; Huang et al., 2018; Zhang et al., 2019; Kiani et al., 2022). Note that some hard convolutional layer orthogonality methods consider mappings of $\mathcal{K}$, therefore resulting in convolutions with kernels of size larger than $k \times k$.

5. Up to a re-scaling, when considering circular boundary conditions, the mapping U is orthogonal.

- **Soft Orthogonality:** Another method to impose orthogonality of matrices during the optimization is to add a regularization of the type $\|WW^T - I\|^2$ to the loss of the specific task. This regularization penalizes the matrices far from orthogonal (e.g., Bansal et al., 2018; Cisse et al., 2017; Qi et al., 2020; Wang et al., 2020; Xie et al., 2017; Guo and Ye, 2020; Jia et al., 2019; Huang et al., 2020).

Note that, unlike Kernel Orthogonality, Convolutional Layer Orthogonality deals directly with $\mathcal{K}$, and thus has a complexity that generally depends on N. However, in the context of Soft Convolutional Layer Orthogonality, the authors of Qi et al. (2020); Wang et al. (2020) introduce the regularizer L_{orth} which is independent of N (see Section 1.3.2 for details), as a surrogate to $\|\mathcal{K}\mathcal{K}^T - \mathrm{Id}_{MN^2}\|_F^2$ and $\|\mathcal{K}^T\mathcal{K} - \mathrm{Id}_{CS^2N^2}\|_F^2$. In Wang et al. (2020), orthogonal convolutional layers involving a stride are considered for the first time.

The present paper specifies the theory supporting the regularization with L_{orth} and the construction of orthogonal convolutional layers. We give necessary and sufficient conditions on the architecture for the orthogonal convolutional layers to exist (see Theorem 4); we unify the L_{orth} formulation for both Row-Orthogonality and Column-Orthogonality cases (see Definition 1); and prove that the regularization with L_{orth}: 1/ is stable, i.e. $L_{orth}(\mathbf{K})$ is small $\Longrightarrow$ $\mathcal{K}\mathcal{K}^T - \mathrm{Id}_{MN^2}$ is small in various senses (see Theorem 7 and Theorem 8); 2/ leads to an orthogonality error that scales favorably when input signal size N grows (see Theorem 8 and Section 3.1.3). We empirically show that, in most cases, the landscape of L_{orth} is such that its minimization can be achieved by *Adam* (Kingma and Ba, 2015), a standard first-order optimizer (see Section 3.1.1). We also identify and analyse the problematic cases (see Section 3.1.2). We show numerically that approximate orthogonality is preserved when N increases (see Section 3.1.3). Finally, we illustrate on Cifar10 and Imagenette data sets how the regularization parameter can be chosen to control the tradeoff between accuracy and orthogonality, for the benefit of both accuracy and robustness (see Section 3.2).

1.3 Context

In this section, we describe the context of the article by defining orthogonality, the regularization function L_{orth} and the Frobenius and spectral norms of the orthogonality residuals, which are two measures of approximate orthogonality. We relate the latter to an approximate isometry property whose benefits are listed in Table 1. The main notations defined in this section are reminded in Table 2, in Appendix A.1.

1.3.1 ORTHOGONALITY

Given a kernel tensor $\mathbf{K}$, the convolutional layer transform matrix $\mathcal{K}$ can be written as:

$$\mathcal{K} = \begin{pmatrix} \mathcal{M}(\mathbf{K}_{1,1}) & \dots & \mathcal{M}(\mathbf{K}_{1,C}) \\ \vdots & \vdots & \vdots \\ \mathcal{M}(\mathbf{K}_{M,1}) & \dots & \mathcal{M}(\mathbf{K}_{M,C}) \end{pmatrix} \in \mathbb{R}^{MN^2 \times CS^2N^2} ,$$

where $\mathcal{M}(\mathbf{K}_{i,j})$ is a matrix that computes a strided convolution for the kernel $\mathbf{K}_{i,j} = \mathbf{K}_{i,j,:,:}$, from the input channel j, to the output channel i (See Appendix B for details). Notice that we use the 'matlab-colon-notation', such that $\mathbf{K}_{i,j,:,:} = (\mathbf{K}_{i,j,m,n})_{0 \le m,n \le k-1} \in \mathbb{R}^{k \times k}$.

In order to define orthogonal matrices, we need to distinguish two cases:

- **Row case (RO case).** When the size of the input space of $\mathcal{K} \in \mathbb{R}^{MN^2 \times CS^2N^2}$ is larger than the size of its output space, i.e. $M \leq CS^2$, $\mathcal{K}$ is orthogonal if and only if its rows are normalized and mutually orthogonal. Denoting the identity matrix $\mathrm{Id}_{MN^2} \in \mathbb{R}^{MN^2 \times MN^2}$, this is written

$$\mathcal{K}\mathcal{K}^T = \mathrm{Id}_{MN^2} \,. \tag{1}$$

  In this case, the mapping $\mathcal{K}$ performs a dimensionality reduction.

- **Column case (CO case).** When $M \geq CS^2$, $\mathcal{K}$ is orthogonal if and only if its columns are normalized and mutually orthogonal:

$$\mathcal{K}^T\mathcal{K} = \mathrm{Id}_{CS^2N^2} \,. \tag{2}$$

  In this case, the mapping $\mathcal{K}$ is an embedding.

Both the RO case and CO case are encountered in practice. When $M = CS^2$, the matrix $\mathcal{K}$ is square and if it is orthogonal then both (1) and (2) hold. The matrix $\mathcal{K}$ is then orthogonal in the usual sense and both $\mathcal{K}$ and $\mathcal{K}^T$ are isometric.

1.3.2 THE FUNCTION $L_{orth}(\mathbf{K})$

In this section, we define a variant of the function $L_{orth} : \mathbb{R}^{M\times C\times k\times k} \longrightarrow \mathbb{R}$ defined in Wang et al. (2020); Qi et al. (2020). The purpose of the proposed variant is to unify the properties of L_{orth} in the RO case and CO case.

Reminding that $k \times k$ is the size of the convolution kernel, for any $h, g \in \mathbb{R}^{k\times k}$ and any $P \in \mathbb{N}$, we define $\mathrm{conv}(h, g, \text{padding zero} = P, \text{stride} = 1) \in \mathbb{R}^{(2P+1)\times(2P+1)}$ as the convolution[6] between h and the zero-padding of g (see Figure 1). Formally, for all $i, j \in [\![0, 2P]\!]$,

$$[\mathrm{conv}(h, g, \text{padding zero} = P, \text{stride} = 1)]_{i,j} = \sum_{i'=0}^{k-1}\sum_{j'=0}^{k-1} h_{i',j'}\bar{g}_{i+i',j+j'},$$

where $\bar{g} \in \mathbb{R}^{(k+2P)\times(k+2P)}$ is defined, for all $(i, j) \in [\![0, k + 2P - 1]\!]^2$, by

$$\bar{g}_{i,j} = \begin{cases} g_{i-P,j-P} & \text{if } (i,j) \in [\![P, P+k-1]\!]^2, \\ 0 & \text{otherwise.} \end{cases}$$

We define $\mathrm{conv}(h, g, \text{padding zero} = P, \text{stride} = S) \in \mathbb{R}^{(\lfloor 2P/S\rfloor+1)\times(\lfloor 2P/S\rfloor+1)}$, for all integer $S \geq 1$ and all $i, j \in [\![0, \lfloor 2P/S\rfloor]\!]$, by

$$[\mathrm{conv}(h, g, \text{padding zero} = P, \text{stride} = S)]_{i,j} = [\mathrm{conv}(h, g, \text{padding zero} = P, \text{stride} = 1)]_{Si,Sj}.$$

We denote (in bold) $\mathbf{conv}(\mathbf{K}, \mathbf{K}, \text{padding zero} = P, \text{stride} = S) \in \mathbb{R}^{M\times M\times(\lfloor 2P/S\rfloor+1)\times(\lfloor 2P/S\rfloor+1)}$ the fourth-order tensor such that, for all $m, l \in [\![1, M]\!]$,

$$\mathbf{conv}(\mathbf{K}, \mathbf{K}, \text{padding zero} = P, \text{stride} = S)_{m,l,:,:} \\ = \sum_{c=1}^{C} \mathrm{conv}(\mathbf{K}_{m,c}, \mathbf{K}_{l,c}, \text{padding zero} = P, \text{stride} = S),$$

6. As is common in machine learning, we do not flip h.

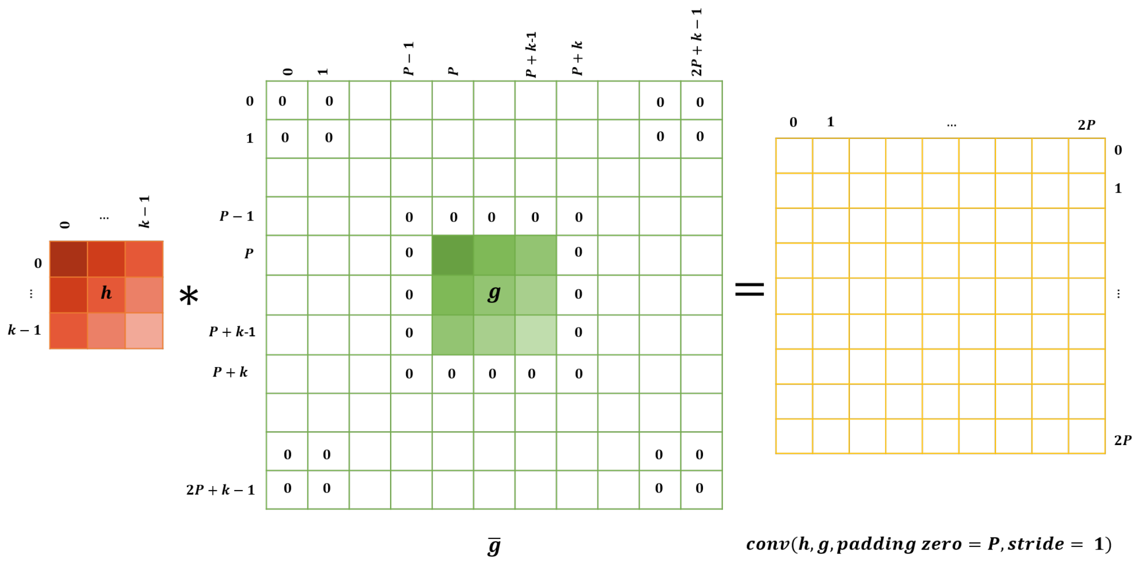

Figure 1: Illustration of $\text{conv}(h, g, \text{padding zero} = P, \text{stride} = 1)$, in the 2D case.

where, for all $m \in [\![1, M]\!]$ and $c \in [\![1, C]\!]$, $\mathbf{K}_{m,c} = \mathbf{K}_{m,c,:,:} \in \mathbb{R}^{k \times k}$.

It has been noted in Wang et al. (2020) that, in the RO case, when $P = \left\lfloor \frac{k-1}{S} \right\rfloor S$,

$$\mathcal{K} \quad \text{orthogonal} \qquad \Longleftrightarrow \qquad \mathbf{conv}(\mathbf{K}, \mathbf{K}, \text{padding zero} = P, \text{stride} = S) = \mathbf{I}_{r0}, \tag{3}$$

where $\mathbf{I}_{r0} \in \mathbb{R}^{M \times M \times (2P/S+1) \times (2P/S+1)}$ is the tensor whose entries are all zero except its central $M \times M$ entry which is equal to an identity matrix: $[\mathbf{I}_{r0}]_{:,:,P/S,P/S} = Id_M$.

Therefore, denoting by $\|.\|_F$ the Euclidean norm in high-order tensor spaces, it is natural to define the following regularization penalty (we justify the CO case right after the definition).

Definition 1 ($\mathbf{L_{orth}}$) *We denote by* $P = \left\lfloor \frac{k-1}{S} \right\rfloor S$. *We define* $L_{orth} : \mathbb{R}^{M \times C \times k \times k} \longrightarrow \mathbb{R}_+$ *as follows*

- *In the RO case,* $M \leq CS^2$*:*

$$L_{orth}(\mathbf{K}) = \| \mathbf{conv}(\mathbf{K}, \mathbf{K}, \textit{padding zero} = P, \textit{stride} = S) - \mathbf{I}_{r0} \|_F^2 \, . \tag{4}$$

- *In the CO case,* $M \geq CS^2$*:*

$$L_{orth}(\mathbf{K}) = \| \mathbf{conv}(\mathbf{K}, \mathbf{K}, \textit{padding zero} = P, \textit{stride} = S) - \mathbf{I}_{r0} \|_F^2 - (M - CS^2) \, .$$

When $M = CS^2$, the two definitions trivially coincide. In the definition, the padding parameter P is the largest multiple of S strictly smaller than k. The difference with the definitions of L_{orth} in Wang et al. (2020); Qi et al. (2020) is in the CO case. In this case with $S = 1$, Qi et al. (2020); Wang et al. (2020) use (4) with $\mathbf{K}^T$ instead of $\mathbf{K}$. For $S \geq 2$ in the CO case, we can not derive a simple equality as in (3). In Wang et al. (2020), remarking that $\|\mathcal{K}^T\mathcal{K} - \mathrm{Id}_{CS^2N^2}\|_F^2 - \|\mathcal{K}\mathcal{K}^T - \mathrm{Id}_{MN^2}\|_F^2$ is a constant which only depends on the size of $\mathcal{K}$, the authors also argue that, whatever S, one can also use (4) in the CO case. We alter this in the CO case as in Definition 1 to obtain both in the RO case and the CO case:

$$\mathcal{K} \text{ orthogonal} \qquad \Longleftrightarrow \qquad L_{orth}(\mathbf{K}) = 0.$$

Once adapted to our notations, the authors in Wang et al. (2020); Qi et al. (2020) propose to regularize convolutional layers parameterized by $(\mathbf{K}_l)_l$ by optimizing

$$L_{task} + \lambda \sum_l L_{orth}(\mathbf{K}_l) \tag{5}$$

where L_{task} is the original objective function of a machine learning task. The function $L_{orth}(\mathbf{K})$ does not depend on N and can be implemented in a few lines of code with Neural Network frameworks. Its gradient is then computed using automatic differentiation.

Of course, when doing so, even if the optimization is efficient, we expect $L_{orth}(\mathbf{K}_l)$ to be different from 0 but less than ε, for a small ε. We investigate, in this article, whether, in this case, the transformation matrix $\mathcal{K}$, still satisfies useful orthogonality properties. To quantify how much $\mathcal{K}$ deviates from being orthogonal, we define the approximate orthogonality criteria and approximate isometry property in the next section. These notions allow to state the stability and scalability theorems (Sections 2.3 and 2.4) and guarantee that the singular values remain close to 1 when L_{orth} is small, even when N is large. This proves that the benefits related to the orthogonality of the layers, which are presented in Table 1, still hold.

1.3.3 APPROXIMATE ORTHOGONALITY AND APPROXIMATE ISOMETRY PROPERTY

Perfect orthogonality is an idealization that never happens, due to floating-point arithmetic, and numerical and optimization errors. In order to measure how $\mathcal{K}$ deviates from being orthogonal, we define the **orthogonality residual** by $\mathcal{K}\mathcal{K}^T - \mathrm{Id}_{MN^2}$, in the RO case, and $\mathcal{K}^T\mathcal{K} - \mathrm{Id}_{CS^2N^2}$, in the CO case. Considering both the Frobenius norm $\|.\|_F$ of the orthogonality residual and its spectral norm $\|.\|_2$, we have two criteria:

$$\mathrm{err}_N^F(\mathbf{K}) = \begin{cases} \|\mathcal{K}\mathcal{K}^T - \mathrm{Id}_{MN^2}\|_F & \text{, in the RO case,} \\ \|\mathcal{K}^T\mathcal{K} - \mathrm{Id}_{CS^2N^2}\|_F & \text{, in the CO case,} \end{cases} \tag{6}$$

and

$$\mathrm{err}_N^s(\mathbf{K}) = \begin{cases} \|\mathcal{K}\mathcal{K}^T - \mathrm{Id}_{MN^2}\|_2 & \text{, in the RO case,} \\ \|\mathcal{K}^T\mathcal{K} - \mathrm{Id}_{CS^2N^2}\|_2 & \text{, in the CO case.} \end{cases} \tag{7}$$

When $M = CS^2$, the definitions in the RO case and the CO case coincide. The two criteria are of course related since for any matrix $A \in \mathbb{R}^{a\times b}$, the Frobenius and spectral norms are such that

$$\|A\|_F \leq \sqrt{\min(a,b)}\|A\|_2 \qquad \text{and} \qquad \|A\|_2 \leq \|A\|_F \,. \tag{8}$$

However, the link is weak, when $\min(a,b)$ is large.

The regularization with $(\mathrm{err}_N^F(\mathbf{K}))^2$ is a natural way to enforce soft-orthogonality of $\mathcal{K}$. However, as mentioned in the introduction, it is not practical because the sizes of $\mathcal{K}$ are too large. We will see in Theorem 7 that $(\mathrm{err}_N^F(\mathbf{K}))^2$ and $L_{orth}(\mathbf{K})$ differ by a multiplicative constant and it will make a clear connection between the regularization with $L_{orth}(\mathbf{K})$ and the regularization with $(\mathrm{err}_N^F(\mathbf{K}))^2$.

However, $\text{err}_N^F(\mathbf{K})$ is difficult to interpret, this is why we consider $\text{err}_N^s(\mathbf{K})$ which relates to the *approximate isometry property* of $\mathcal{K}$ as we explain below. The latter has direct consequences on the properties of the layer (see Table 1).

Indeed, in the applications, one key property of orthogonal operators is their connection to isometries. It is the property that prevents the gradient from exploding and vanishing (Cisse et al., 2017; Xiao et al., 2018; Li et al., 2019a; Huang et al., 2020). This property also enables to keep the examples well separated, which has an effect similar to the batch normalization (Qi et al., 2020; Chai et al., 2020), and to have a 1-Lipschitz forward pass and therefore improves robustness (Wang et al., 2020; Cisse et al., 2017; Li et al., 2019b; Trockman and Kolter, 2021; Jia et al., 2019).

We denote the Euclidean norm of a vector by $\|.\|$. To clarify the connection between orthogonality and isometry, we define the 'ε-Approximate Isometry Property' (ε-AIP).

Definition 2 *A layer transform matrix $\mathcal{K} \in \mathbb{R}^{MN^2 \times CS^2N^2}$ satisfies the ε-Approximate Isometry Property if and only if*

- *RO case, $M \leq CS^2$:*

$$\begin{cases} \forall x \in \mathbb{R}^{CS^2N^2} & \|\mathcal{K}x\|^2 \leq (1+\varepsilon)\|x\|^2 \\ \forall y \in \mathbb{R}^{MN^2} & (1-\varepsilon)\|y\|^2 \leq \|\mathcal{K}^Ty\|^2 \leq (1+\varepsilon)\|y\|^2 \end{cases}$$

- *CO case, $M \geq CS^2$:*

$$\begin{cases} \forall x \in \mathbb{R}^{CS^2N^2} & (1-\varepsilon)\|x\|^2 \leq \|\mathcal{K}x\|^2 \leq (1+\varepsilon)\|x\|^2 \\ \forall y \in \mathbb{R}^{MN^2} & \|\mathcal{K}^Ty\|^2 \leq (1+\varepsilon)\|y\|^2 \end{cases}$$

The following proposition makes the link between $\text{err}_N^s(\mathbf{K})$ and AIP. It shows that minimizing $\text{err}_N^s(\mathbf{K})$ enhances the AIP property.

Proposition 3 *Let N be such that $SN \geq k$. We have, both in the RO case and CO case,*

$$\mathcal{K} \textit{ is } \text{err}_N^s(\mathbf{K})\textit{-AIP}.$$

This statement actually holds for any matrix (not only layer transform matrix) and is already stated in Bansal et al. (2018); Guo and Ye (2020). For completeness, we provide proof, in Appendix G.

In Proposition 3 and in Theorem 4 (see the next section), the condition $SN \geq k$ only states that the input width and height are larger than the size of the kernels. This is always the case in practice.

We summarize in Table 1 the properties of ε-AIP layers when ε is small, in the different possible scenarios. We remind that a kernel tensor $\mathbf{K}$ can define a convolutional layer or a deconvolution layer. Deconvolution layers are, for instance, used to define layers of the decoder of an auto-encoder or variational auto-encoder (Kingma and Welling, 2014). In the convolutional case, $\mathcal{K}$ is applied during the forward pass and $\mathcal{K}^T$ is applied during the backward pass. In a deconvolution layer, $\mathcal{K}^T$ is applied during the forward pass and $\mathcal{K}$ during the backward pass. Depending on whether we have $M < CS^2$, $M > CS^2$ or $M = CS^2$, when $\mathcal{K}$ is ε-AIP with $\varepsilon << 1$, either $\mathcal{K}^T$, $\mathcal{K}$ or both preserve distances (see Table 1).

To complement Table 1, notice that in the RO case, if $\text{err}_N^F(\mathbf{K}) \leq \varepsilon$, then for any i, j with $i \neq j$, we have $|\mathcal{K}_{i,:}\mathcal{K}_{j,:}^T| \leq \varepsilon$, where $\mathcal{K}_{i,:}$ is the i^{th} line of $\mathcal{K}$. In other words, when ε is small, the features computed by $\mathcal{K}$ are mostly uncorrelated (Wang et al., 2020).

		Forward pass		Backward pass	
		Lipschitz Forward pass	Keep examples separated	Prevent grad. expl.	Prevent grad. vanish.
Convolutional layer	$M < CS^2$	✓	✗	✓	✓
	$M > CS^2$	✓	✓	✓	✗
Deconvolution layer	$M < CS^2$	✓	✓	✓	✗
	$M > CS^2$	✓	✗	✓	✓
Conv. & Deconv.	$M = CS^2$	✓	✓	✓	✓

Table 1: Properties of a ε-AIP layers (when $\varepsilon \ll 1$), depending on whether $\mathbf{K}$ defines a convolutional or deconvolutional layer. The red crosses indicate when the forward or backward pass performs a dimensionality reduction.

2. Theoretical Analysis of Orthogonal Convolutional Layers

This section contains the theoretical contributions of the article. In all the theorems in this section, the considered convolutional layers are either applied to a signal, when $d = 1$, or an image, when $d = 2$.

We remind that the architecture of the layer is characterized by (M, C, k, S) where: M is the number of output channels; C is the number of input channels; $k \geq 1$ is an odd positive integer and the convolution kernels are of size k, when $d = 1$, and $k \times k$, when $d = 2$; the stride parameter is S.

We want to highlight that the theorems of Sections 2.1, 2.3 and 2.4 are for convolution operators defined with circular boundary conditions (see Appendix B for details). We point out in Section 2.2 restrictions for the ‘valid’ and ‘same’ zero-padding boundary conditions (see Appendix D.1 and Appendix D.2 for details).

With circular boundary conditions, all input channels are of size SN, when $d = 1$, $SN \times SN$, when $d = 2$. The output channels are of size N and $N \times N$, respectively when $d = 1$ and 2. When $d = 1$, the definitions of L_{orth}, err_N^F and err_N^s are in Appendix A.2.

In Section 2.1, we state a theorem that provides the necessary and sufficient conditions on the architecture for an orthogonal convolutional layer to exist. In Section 2.2, we describe restrictions for the ’valid’ and ’same’ boundary conditions. In Section 2.3, we state a theorem that provides a relation between the Frobenius norm of the orthogonality residual and the regularization penalty L_{orth}. Finally, in Section 2.4, we state a theorem that provides an upper bound of the spectral norm of the orthogonality residual using the regularization penalty L_{orth}.

2.1 Existence of Orthogonal Convolutional Layers

The next theorem gives a necessary and sufficient condition on the architecture (M, C, k, S) and N for an orthogonal convolutional layer transform to exist. To simplify notations, we denote, for $d = 1$ or 2, the space of all the kernel tensors by

$$\mathbb{K}_d = \begin{cases} \mathbb{R}^{M\times C\times k} & \text{when } d = 1, \\ \mathbb{R}^{M\times C\times k\times k} & \text{when } d = 2. \end{cases}$$

We also denote, for $d = 1$ or 2,

$$\mathbb{K}_d^\perp = \{\mathbf{K} \in \mathbb{K}_d | \ \mathcal{K} \text{ is orthogonal}\}.$$

Theorem 4 *Let N be such that $SN \geq k$ and $d = 1$ or 2.*

- *RO case, i.e. $M \leq CS^d$: $\mathbb{K}_d^\perp \neq \emptyset$ if and only if $M \leq Ck^d$.*
- *CO case, i.e. $M \geq CS^d$: $\mathbb{K}_d^\perp \neq \emptyset$ if and only if $S \leq k$.*

Theorem 4 is proved in Appendix C. Again, the conditions coincide when $M = CS^d$.

When $S \leq k$, which is by far the most common situation, there exist orthogonal convolutional layers in both the CO case and the RO case. Indeed, in the RO case, when $S \leq k$, we have $M \leq CS^d \leq Ck^d$.

However, skip-connection layers (also called shortcut connection) with stride in Resnet (He et al., 2016) for instance, usually have an architecture (M, C, k, S) = $(2C, C, 1, 2)$, where C is the number of input channels. The kernels are of size 1×1. In that case, $M \leq CS^d$ and $M > Ck^d$. Theorem 4 says that there is no orthogonal convolutional layer for this type of layer.

To conclude, the main consequence of Theorem 4 is that, with circular boundary conditions and for most of the architecture used in practice (with an exception for the skip-connections with stride), there exist orthogonal convolutional layers.

2.2 Restrictions due to Boundary Conditions

In Sections 2.1, 2.3 and 2.4, we consider convolutions defined with circular boundary conditions. This choice is neither for technical reasons nor to enable the use of the Fourier basis. We illustrate in the next two propositions that, for convolutions defined with the 'valid' condition, or the 'same' condition with zero-padding, hard-orthogonality is in many situations too restrictive. We consider in this section an unstrided convolution.

Before stating the next proposition, we remind that with the 'valid' boundary conditions, only the entries of the output such that the support of the translated kernel is entirely included in the input's domain are computed. The size of each output channel is smaller than the size of the input channels. The formal definition of the 'valid' boundary conditions is at the beginning of Appendix D.1.

Proposition 5 *Let $N \geq 2k - 1$. With the 'valid' condition, there exists no orthogonal convolutional layer in the CO case.*

This proposition holds in the 1D and 2D cases. We give its proof only in the 1D case in Appendix D.1.

Before stating the next proposition, we remind that to compute a convolution with the zero-padding 'same' boundary conditions, we first extend each input channel with zeros and then compute the convolutions such that each output channel has the same support as the input channels, before padding/extension. A formal definition of the zero-padding 'same' boundary condition is at the beginning of Appendix D.2.

Let $(e_{i,j})_{i=0..k-1,j=0..k-1}$ be the canonical basis of $\mathbb{R}^{k\times k}$. For the zero-padding 'same', we have the following proposition.

Proposition 6 *Let $N \geq k$. For $\mathbf{K} \in \mathbb{R}^{M\times C\times k\times k}$, with the zero-padding 'same' and $S = 1$, both in the RO case and CO case, if $\mathcal{K}$ is orthogonal then there exist $(\alpha_{m,c})_{m=1..M,c=1..C} \in \mathbb{R}^{M\times C}$ such that for all $(m, c) \in [\![1, M]\!] \times [\![1, C]\!]$, $\mathbf{K}_{m,c} = \alpha_{m,c}e_{r,r}$, where r satisfies $k = 2r + 1$. As a consequence*

$$\mathcal{K} = \begin{pmatrix} \alpha_{1,1}Id_{N^2} & \dots & \alpha_{1,C}Id_{N^2} \\ \vdots & \vdots & \vdots \\ \alpha_{M,1}Id_{N^2} & \dots & \alpha_{M,C}Id_{N^2} \end{pmatrix} \in \mathbb{R}^{MN^2\times CN^2} .$$

This proposition holds in the 1D and 2D cases. We give its proof only in the 1D case in Appendix D.2.

To recapitulate, the results state that with padding 'valid', no orthogonal convolution can be built in the CO case and that for zero-padding 'same', the orthogonal convolution layers are trivial transformations.

Note that, the propositions do not exclude the existence of sufficiently expressive sets of 'approximately orthogonal' convolutional layers with these boundary conditions. A strategy based on soft-orthogonality may still enjoy most of the benefits of orthogonality for these boundary conditions.

2.3 Frobenius Norm Stability

We recall that the motivation behind this is the following : The authors of Wang et al. (2020); Qi et al. (2020) argue that $L_{orth}(\mathbf{K}) = 0$ is equivalent to $\mathcal{K}$ being orthogonal. However, they do not provide stability guarantees. Without this guarantee, it could happen that $L_{orth}(\mathbf{K}) = 10^{-9}$ and $\|\mathcal{K}\mathcal{K}^T - Id\|_F = 10^9$. This would make the regularization with L_{orth} useless, unless the algorithm reaches $L_{orth}(\mathbf{K}) = 0$.

The following theorem proves that this cannot occur. Therefore, if $L_{orth}(\mathbf{K})$ is small, $\mathrm{err}_N^F(\mathbf{K})$ is small at least for moderate signal sizes. Also, a corollary is that adding L_{orth} as a penalty regularization is equivalent to adding the Frobenius norm of the orthogonality residual.

Theorem 7 *Let N be such that $SN \geq 2k - 1$ and $d = 1$ or 2. For a convolutional layer defined using circular boundary conditions, we have, both in the RO case and CO case,*

$$(\mathrm{err}_N^F(\mathbf{K}))^2 = N^d L_{orth}(\mathbf{K}) ,$$

where $\mathrm{err}_N^F(\mathbf{K})$ is defined in (6).

Theorem 7 is proved, in Appendix E. We remind that $L_{orth}(\mathbf{K})$ is independent of N. The theorem formalizes for circular boundary conditions and for both the CO case and the RO case, the reasoning leading to the regularization with L_{orth} in Wang et al. (2020).

Using Theorem 7, we find that (5) becomes

$$L_{task} + \sum_l \frac{\lambda}{N_l^d}(\mathrm{err}_{N_l}^F(\mathbf{K}_l))^2.$$

Once the parameter λ is made dependent on the input size of layer l, the regularization term λL_{orth} is equal to the Frobenius norm of the orthogonality residual. This justifies the use of L_{orth} as a regularizer.

We can also see from Theorem 7 that, for both the RO case and the CO case, when $L_{orth}(\mathbf{K}) = 0$, $\mathcal{K}$ is orthogonal, independently of N. This recovers the result stated in Qi et al. (2020) for $S = 1$, and the result stated in Wang et al. (2020) in the RO case for any S .

Considering another signal size N' and applying Theorem 7 with the sizes N and N', we find

$$(\mathrm{err}^F_{N'}(\mathbf{K}))^2 = \frac{(N')^d}{N^d}(\mathrm{err}^F_N(\mathbf{K}))^2.$$

To the best of our knowledge, this equality is new. This could be of importance in situations when N varies. For instance when the neural network is learned on a data set containing signals/images of a given size, but the inference is done for signals/images of varying size (Ren et al., 2015; Shelhamer et al., 2017; Ji et al., 2019).

Finally, using (8) and Proposition 3, $\mathcal{K}$ is ϵ-AIP with ϵ scaling like the square root of the signal/image size. This might not be satisfactory. We exhibit in the next section a tighter bound on ϵ, independent of the input size N.

2.4 Spectral Norm Stability and Scalability

We prove in Theorem 8 that $\mathrm{err}^s_N(\mathbf{K})^2$ is sandwiched between two quantities proportional to $L_{orth}(\mathbf{K})$. The multiplicative factors do not depend on N. Hence, when $L_{orth}(\mathbf{K})$ is small, $\mathrm{err}^s_N(\mathbf{K})^2$ is also small for all N. As a consequence, as long as $L_{orth}(\mathbf{K}) \ll 1$ even if the algorithm does not reach $L_{orth}(\mathbf{K}) = 0$, regularizing with $L_{orth}(\mathbf{K})$ permits to construct nearly orthogonal and isometric convolutional layers independently of N.

Moreover, combined with Proposition 3 this ensures that, if $L_{orth}(\mathbf{K})$ is small, $\mathcal{K}$ is ε-AIP with ε small. Using Table 1, we see that this property leads to more robustness and avoids gradient vanishing/exploding. This is in line with the empirical results observed in Wang et al. (2020); Qi et al. (2020).

Theorem 8 *Let N be such that $SN \geq 2k - 1$ and $d = 1$ or 2. For a convolutional layer defined using circular boundary conditions, we have*

$$\alpha'\, L_{orth}(\mathbf{K}) \;\leq\; (\mathrm{err}^s_N(\mathbf{K}))^2 \;\leq\; \alpha\, L_{orth}(\mathbf{K})$$

where $\mathrm{err}^s_N(\mathbf{K})$ is defined in (7), for $\alpha' = \frac{1}{\min(M, CS^2)}$ and

$$\alpha = \begin{cases} \left(2\left\lfloor \frac{k-1}{S} \right\rfloor + 1\right)^d M & \textit{in the RO case } (M \leq CS^d), \\ (2k-1)^d C & \textit{in the CO case } (M \geq CS^d). \end{cases}$$

Theorem 8 is proved, in Appendix F. When $M = CS^d$, the two inequalities hold and it is possible to take the minimum of the two α values.

As we can see from Theorem 8, unlike with the Frobenius norm, the spectral norm of the orthogonality residual is upper-bounded by a quantity that does not depend on N. The lower-bound of Theorem 8 guarantees that the upper-bound is tight up to a multiplicative constant. However, we cannot expect much improvement in this regard since the multiplicative constant $\sqrt{\alpha}$ is usually moderately large[7]. For instance, with $(M, C, k, S) = (128, 128, 3, 2)$, for images, $\sqrt{\alpha} \leq 34$. If as is

7. For usual architectures, $\sqrt{\alpha}$ is always smaller than 200.

common in practice the optimization algorithm reaches $L_{orth}(\mathbf{K}) \leq 10^{-6}$, Theorem 8 guarantees that, independently of N,

$$\text{err}_N^s(\mathbf{K}) \leq \sqrt{\alpha L_{orth}(\mathbf{K})} \leq 0.034.$$

Using Proposition 3, independently of N, a convolutional layer defined with $\mathbf{K}$ is ϵ-AIP, for $\epsilon \leq 0.034$, and we have, using Definition 2,

$$\begin{cases} \forall x \in \mathbb{R}^{CS^2N^2} & \|\mathcal{K}x\| \leq \sqrt{1+\varepsilon}\|x\| \leq 1.017\|x\| \\ \forall y \in \mathbb{R}^{MN^2} & 0.982\|y\| \leq \sqrt{1-\varepsilon}\|y\| \leq \|\mathcal{K}^T y\| \leq 1.017\|y\| \end{cases}$$

The layer benefits from the properties described in Table 1.

The development done for the above example can be repeated as soon as $L_{orth}(\mathbf{K}) \ll 1$, both in the RO case and CO case. Experiments that confirm this behavior are in Section 3.

3. Experiments

Before illustrating the benefits of approximate orthogonality to robustly classify images in Section 3.2, we conduct several synthetic experiments in Section 3.1. The synthetic experiments empirically evaluate the landscape of L_{orth} in Section 3.1.1, 3.1.2 and illustrate the theorems of Section 2, in Section 3.1.3.

Both in Section 3.1 and Section 3.2, to evaluate how close $\mathcal{K}$ is to being orthogonal, we compute some of its singular values σ for different input sizes $SN \times SN$. When $S = 1$, we compute all the singular values of $\mathcal{K}$ with the Algorithm 1, Appendix I, from Sedghi et al. (2018). For convolutions with stride, $S > 1$, there is no known practical algorithm to compute all the singular values and we simply apply the well-known power iteration algorithm associated with a spectral shift, to retrieve the smallest and largest singular values $(\sigma_{min}, \sigma_{max})$ of $\mathcal{K}$ (see Algorithm 2 in Appendix I). We remind that $\mathcal{K}$ orthogonal is equivalent to $\sigma_{min} = \sigma_{max} = 1$.

3.1 Synthetic Experiments

Section 3.1.1, 3.1.2 and 3.1.3 report on results of the massive experiment that is described below.

In order to avoid interaction with other objectives, we train a single 2D convolutional layer with circular padding. We explore all the architectures such that $\mathbb{K}_2^\perp \neq \emptyset$, for $C \in [\![1, 64]\!]$, $M \in [\![1, 64]\!]$, $S \in \{1, 2, 4\}$, and $k \in \{1, 3, 5, 7\}$. This leads to 44924 architectures for which an orthogonal convolutional layer exists (among 49152 architectures in total).

For each architecture, the model is trained using a *Glorot uniform* initializer and an *Adam* optimizer (Kingma and Ba, 2015) with fixed learning rate[8] 0.01 on a null loss ($L_{task}(X, Y, \mathbf{K}) = 0$, for all input X, target Y, and kernel tensor $\mathbf{K}$) and the $L_{orth}(\mathbf{K})$ regularization (see Definition 1) during 3000 steps[9].

We report below implementation details that have no influence on the results, since $L_{task} = 0$. No data are involved in the synthetic experiments and the input of the layer contains a null input of size $(C, 64, 64)$. Other input sizes from 8 to 256 were tested but not reported, leading to the same conclusions. Batch size (which thus has no influence on the results) is set to one.

8. We do not report experiments for other tested learning rates $10^{-1}, 10^{-3}, 10^{-4}, 10^{-5}$ because they lead to the same conclusions.

9. Increasing the number of steps leads to the same conclusions.

3.1.1 OPTIMIZATION LANDSCAPE

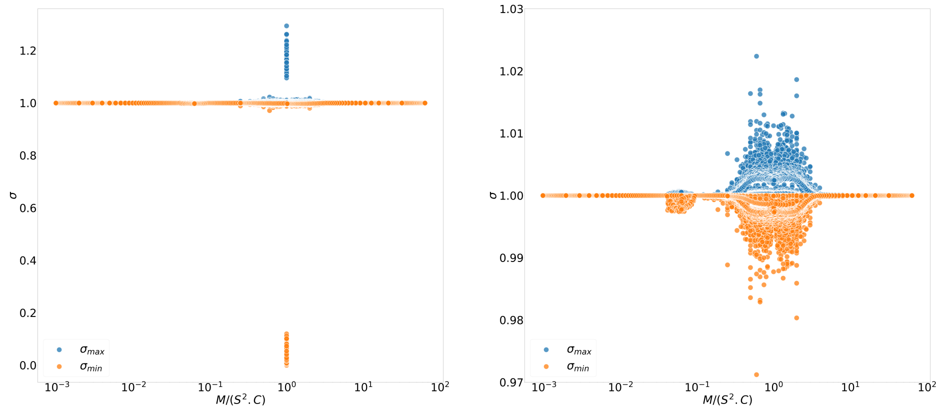

Figure 2: **Optimization of $\mathbf{L_{orth}}$.** Minimization of $L_{orth}(\mathbf{K})$ for a kernel tensor of architecture (M, C, k, S) among the 44924 possible configurations satisfying $\mathbb{K}_2^{\perp} \neq \emptyset$. For each architecture the resulting trained kernel is represented by a blue dot and an orange dot corresponding respectively to its largest and smallest singular values σ_{max} and σ_{min}. The x-axis represents M/CS^2 in log scale. (left) All configurations; (right) All configurations for which $M \neq CS^2$. On the right final convolutions are nearly orthogonal ($\sigma_{max} = \sigma_{min} \approx 1$), but some configurations on the left (where $M = CS^2$) have σ_{max} larger that one, and σ_{min} close to zero.

For each architecture, we plot on Figure 2 the values of σ_{min} and σ_{max} for the obtained $\mathcal{K}$ and $SN \times SN = 64 \times 64$. The experiment for a given architecture (M, C, k, S) is represented by two points: σ_{max}, in blue, and σ_{min}, in orange. For each point (x, y) in Figure 2, the first coordinate x corresponds to the ratio $\frac{M}{CS^2}$ of the considered architecture, and the second coordinate y equals the singular value (σ_{min} or σ_{max}) of the obtained $\mathcal{K}$. The points with $x \leq 1$ correspond to the artchitecture in the RO case ($\mathcal{K}$ is a fat matrix), and the others correspond to the architectures in the CO case ($\mathcal{K}$ is a tall matrix).

The right plot of Figure 2 shows that all configurations where $M \neq CS^2$ are trained very accurately to near-perfect orthogonal convolutions. These configurations represent the vast majority of cases found in practice. However, the left plot of Figure 2 points out that some architectures, with $M = CS^2$, might not fully benefit of the regularization with L_{orth}. These architectures, corresponding to a square $\mathcal{K}$, can mostly be found when $M = C$ and $S = 1$, for instance in VGG (Simonyan and Zisserman, 2015) and Resnet (He et al., 2016). We have conducted experiments that we do not report here in detail, and it seems that this is specific to the convolutional case. Fully-connected layers optimized to be orthogonal do not suffer from this phenomenon.

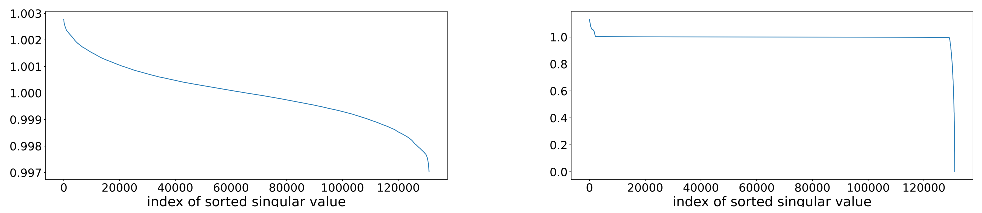

Figure 3: **Singular values of** $\mathcal{K}$, when $C = M$ and $S = 1$. Optimization is (Left) successful, $L_{orth}(\mathbf{K}) \ll 1$, (Right) Unsuccessful, $L_{orth}(\mathbf{K}) \geq 0.1$. On the left, we see that orthogonal convolutions can be reached even when $C = M$ and $S = 1$. On the right we see that, even for unsuccessful optimization, most of the singular values are very close to one.

3.1.2 ANALYSIS OF THE $M = CS^2$ CASES

Since we know that $\mathbb{K}_2^\perp \neq \emptyset$, the explanation for the failure cases (when σ_{max} or σ_{min} significantly differ from 1) is that the optimization was not successful. We tried many learning rate schemes and iteration numbers but obtained similar results[10].

To evaluate the proportion of successful optimizations when $M = CS^2$, we run 100 training experiments, with independent initialization, for each configuration when $M = CS^2$. In average, after convergence, we found $\sigma_{min} \sim 1 \sim \sigma_{max}$ for 14% of runs, proving that the minimizer can be reached. The explanation of this phenomenon and the evaluation of its impact on applications are open questions that we keep for future research. A contribution of the article is to empirically identify these problematic cases.

We display on Figure 3 the singular values of $\mathcal{K}$ defined for $S = 1$ and $N \times N = 64 \times 64$ for two experiments where $M = C$. In the experiment on the left, the optimization is successful and the singular values are very accurately concentrated around 1. On the right, we see that only a few of the singular values significantly differ from 1.

Figure 3 shows that even if σ_{min} and σ_{max} are not close to 1, as shown in Figure 2, most of the singular values are close to 1. This probably explains why the landscape problem does not alter the performance on real data sets in Wang et al. (2020) and Qi et al. (2020). Notice that Wang et al. (2020) display a curve similar to Figure 3 when used for a real data set.

3.1.3 STABILITY OF $(\sigma_{min}, \sigma_{max})$ WHEN N VARIES

In this experiment, we evaluate how the singular values σ_{min} and σ_{max} of $\mathcal{K}$ vary when the parameter N defining the size $SN \times SN$ of the input channels varies, for $\mathbf{K}$ fixed. This is important for applications (Shelhamer et al., 2017; Ji et al., 2019; Ren et al., 2015) using fully convolutional networks, or for transfer learning using pre-learnt convolutional feature extractor.

To do so, we randomly select 50 experiments for which the optimization was successful ($L_{orth}(\mathbf{K}) \leq 0.001$) and 50 experiments for which it was unsuccessful ($L_{orth}(\mathbf{K}) \geq 0.02$). They are respectively used to construct the figures on the left and the right side of Figure 4. For a given $\mathbf{K}$, we display the singular values σ_{min} and σ_{max} of $\mathcal{K}$ for $N \in \{5, 12, 15, 32, 64, 128, 256, 512, 1024\}$, as orange and blue dots. The dots corresponding to the same $\mathbf{K}$ are linked by a line.

10. See the description of the experiments at the beginning of Section 3.1 and the related footnotes.

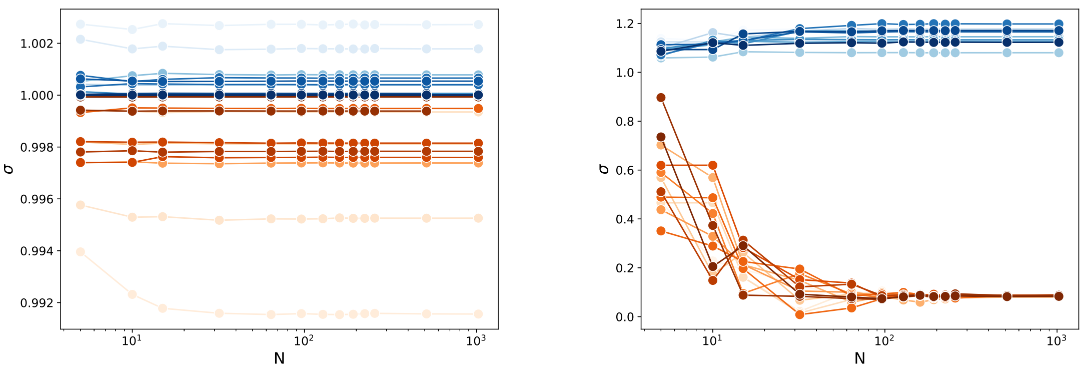

Figure 4: **Evolution of $\sigma_{\mathbf{min}}$ and $\sigma_{\mathbf{max}}$ according to input image size** (x-axis: N in log-scale). Each line (transparency) represents the singular values σ_{max} (in blue) and σ_{min} (in orange) of $\mathcal{K}$ for different N and a fixed $\mathbf{K}$. (Left) $\mathbf{K}$ is such that $L_{orth}(\mathbf{K}) \ll 1$, singular values remain close to one whatever N. (Right) $\mathbf{K}$ is such that $L_{orth}(\mathbf{K}) \not\ll 1$, the largest (resp. smallest) singular values increase (resp. decrease) when N grows.

We see, on the left of Figure 4, that for successful experiments ($L_{orth}(\mathbf{K}) \ll 1$), the singular values are very stable when N varies. This corresponds to the behavior described in Theorem 8 and Proposition 3. We also point out, on the right of Figure 4, that for unsuccessful optimization ($L_{orth} \not\ll 1$), σ_{min} (resp. σ_{max}) values decrease (resp. increase) rapidly when N increases.

3.2 Data sets Experiments

In this section we compare, on Cifar10 and Imagenette data sets, the performance, robustness, spectral properties and processing time of three networks: standard convolutional neural networks with unconstrained convolutions called *Conv2D*, the same network architectures with convolutions regularized with L_{orth} and the same network architectures with convolutions constrained with a method that we call *Cayley*, a hard convolutional layer orthogonality method[11] based on the Cayley transform (Trockman and Kolter, 2021). The latter builds convolutions parameterized by $k \times k$ parameters but, because a mapping is applied to obtain orthogonality, the convolution kernels are of size $N \times N$. In comparison, L_{orth} regularization provides convolutions kernels of size $k \times k$, as is standard. The methods are therefore not expected to provide the same results which makes the comparison a bit complicated. This comparison is also somewhat unfair since the regularization with L_{orth} enjoys a parameter λ. We show results for a wide range of λ but assume, when interpreting the results, that an optimal λ is chosen, for instance using cross-validation.

The design of the experiments aims at simultaneously obtaining good accuracy and robustness. Therefore, for the purpose of robustness, we only use isometric activations and nearly orthogonal convolutional layers. We cannot expect, with this robustness constraint, to obtain clean accuracies as good as those reported in Wang et al. (2020).

11. Our experiments complement the comparison of L_{orth} with kernel orthogonality methods in Wang et al. (2020).

On Cifar10, we use a VGG-like architecture (Simonyan and Zisserman, 2015) with nine convolutional layers and a single dense output layer with ten 1-Lipschitz neurons. In all experiments, for a fair comparison, we use invertible downsampling emulation as in Trockman and Kolter (2021). In order to avoid problematic configurations described in Section 3.1.2, we alternate channels numbers $C, C+2, C$ within each VGG block (see Table H.1).

The network is trained during 400 epochs with a batch size of 128, using cross-entropy loss with temperature, Adam optimizer (Kingma and Ba, 2015) with a decreasing learning rate, and standard data augmentation. A full description of hyperparameters is given in Appendix H.1. The optimized network achieves 91% accuracy on the Cifar10 test set, for the $Conv2D$ classical network.

As already mentioned, three configurations are compared: $Conv2D$: classical convolutions (i.e. no regularization), $Cayley$: convolutions constrained by Cayley method (Trockman and Kolter, 2021), and L_{orth}: the regularization with L_{orth}. For the L_{orth} regularization, we investigate the properties of the solution obtained when minimizing (5) for $\lambda \in \{10, 1, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}\}$.

After training, σ_{max} and σ_{min} values are computed for each convolutional layer using the method described in Appendix I. Each configuration is learnt 10 times to provide mean and standard deviation for the following metrics:

- *Acc. clean*: Classical accuracy on a clean test set
- $\Sigma_{max} = \max_l(\sigma_{max}(\mathcal{K}_l))$: the largest singular value among all the convolutional layers's singular values.
- $\Sigma_{min} = \min_l(\sigma_{min}(\mathcal{K}_l))$: the smallest singular value among all the convolutional layers's singular values.
- E_{lip} : Empirical local Lipschitz constants of the network computed using the PGD-like method proposed by Yang et al. (2020).
- E_{rob} : The empirical robustness accuracy, i.e. the proportion of test samples on which a vanilla Projected Gradient Descent (PGD) attack (Madry et al., 2018) failed (for a robustness radius $\epsilon = 36/255$). PGD attack is applied with 10 iterations and a factor $\alpha = \epsilon/4.0$.
- T_{epoch}: the average epoch processing time.

Figure 5 shows that the regularization parameter λ, in (5), provides a way to tune a tradeoff between robustness (E_{rob}) and clean accuracy ($Acc.clean$), by controlling the singular values of the layers (Σ_{max} and Σ_{min}). On the contrary $Cayley$ or $Conv2D$ each provide a single tradeoff (shown with constant value in figures). The configurations $\lambda = 10^{-1}$ and 10^{-2} achieve better clean accuracy and similar empirical robustness performances as the $Cayley$ method. Furthermore, their empirical Lipschitz constants are very close to one. Finally, error bands for $Cayley$ and L_{orth} methods are very narrow.

Processing time T_{epoch} for regularizing with L_{orth} is only 5% slower than the reference network $Conv2D$, but 2.2 times faster than the one for the $Cayley$ method. It is not reported here in detail but the convergence speeds, in number of epochs, are similar. Moreover, L_{orth} provides classical convolution at inference. On the contrary, the $Cayley$ method provides orthogonal convolutions of size $N \times N$ obtained using a mapping that involves Fourier transforms, which leads to higher computational complexity even at inference. The change of support can also explain the slight

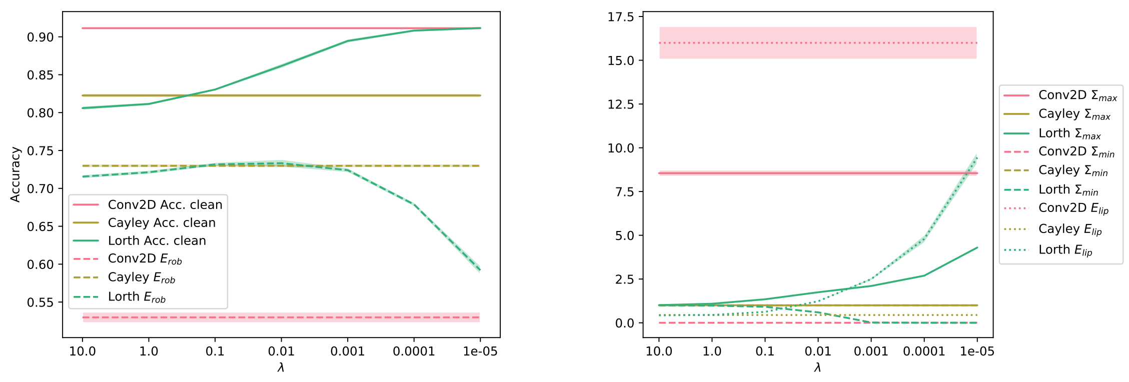

Figure 5: **Cifar10**: Mean evolution of metrics (and error band in shadow) according to λ parameter for L_{orth} method, and comparison with $Conv2D$ and $Cayley$ configurations (constant values): (Left) Clean accuracy and empirical robustness for $\epsilon = 36/255$, (Right) Σ_{max}, Σ_{min} and E_{lip} metrics. The parameter λ permits tune a tradeoff between accuracy and orthogonality, for the benefit of a better accuracy and a better robustness.

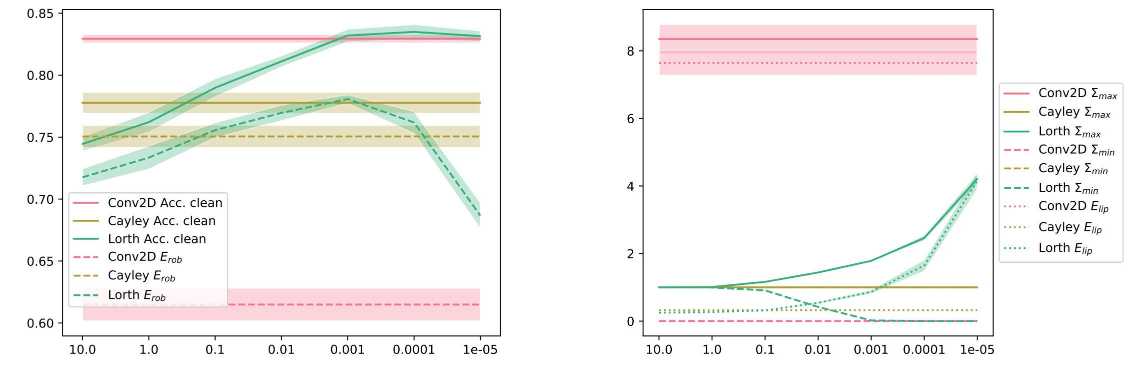

Figure 6: **Imagenette**: Mean evolution of metrics (and error band in shadow) according to λ parameter for L_{orth} method, and comparison with $Conv2D$ and $Cayley$ configurations (constant values): (Left) Clean accuracy and empirical robustness for $\epsilon = 36/255$, (Right) Σ_{max}, Σ_{min} and E_{lip} metrics. The parameter λ permits to tune a tradeoff between accuracy and orthogonality, for the benefit of a better accuracy and a better robustness.

difference in $Acc.clean$ between the $Cayley$ method and the strong regularization $\lambda = 10$ for L_{orth} method.

Figure 6 presents the same experiments on the Imagenette data set (Howard, 2020). The latter is a 10-class subset of Imagenet data set (Deng et al., 2009) with 160×160 images. The architecture is also a VGG-like one but with 15 convolutional layers. We trained 10 times during 400 epochs with a batch size of 64, using the same loss and optimizer as for the Cifar10 experiments. The architecture and hyperparameters are described in Section H.2. The average performance of the unconstrained $Conv2D$ networks is about 83%.

Interestingly, even if the image size is larger on the Imagenette data set, L_{orth} regularization shows the same profile as for Cifar10, when λ decreases, ranging from strong orthogonality to high clean accuracy (equivalent to Conv2D configuration), with the best compromise for $\lambda = 10^{-3}$. Notice that for λ decreasing from 10 to 0.01 the loss of orthogonality permits to obtain a significantly better accuracy but does not significantly favor attacks. Altogether, this leads to an increase in the empirical robustness accuracy. The phenomenon is present but less visible on Cifar10.

Besides, because L_{orth} does not depend on the size parameter N of the input channels, the processing time for the L_{orth} regularization is only 1.1 times slower than for the non-constrained convolution $Conv2D$. In comparison, the $Cayley$ method is 6.5 slower than $Conv2D$.

4. Conclusion

This paper provides a necessary and sufficient condition on the architecture for the existence of an orthogonal convolutional layer with circular padding. The conditions prove that orthogonal convolutional layers exist for most relevant architectures. We show that the situation is less favorable with 'valid' and 'same' zero-paddings. We also prove that the minimization of the surrogate L_{orth} enables constructing orthogonal convolutional layers in a stable manner, that also scales well with the input size parameter N. The experiments confirm that this is practically the case for most of the configurations, except when $M = CS^2$ for which interrogations remain.

Altogether, the study guarantees that the regularization with L_{orth} is an efficient, stable numerical strategy to learn orthogonal convolutional layers. It can safely be used even when the signal/image size is very large. The regularization parameter λ is chosen depending on the tradeoff we want between accuracy and orthogonality, for the benefit of both accuracy and robustness.

Let us mention three open questions related to this article. First, a better understanding of the landscape problem as well as solutions to this problem when $M = CS^2$ could be useful. Also, as initiated in Kim et al. (2021); Fei et al. (2022), the extension of Lipschitz and orthogonal constraints and regularization to the attention-based networks is a natural and relevant open question. Finally, a clean adaptation of the regularization with L_{orth} for the 'valid' and 'same' boundary conditions is needed. As shown in Section 2.2, approximate orthogonality seems to be key with these boundary conditions.

Acknowledgments

Our work has benefited from the AI Interdisciplinary Institute ANITI. ANITI is funded by the French "Investing for the Future – PIA3" program under the Grant agreement n°ANR-19-PI3A-0004. The authors gratefully acknowledge the support of the DEEL project.[12]

Appendix A. Notation and Definitions

In this section we specify some notation and definitions.

12. https://www.deel.ai/

A.1 Notation

We summarize the notations specific to our problem in Table 2. We then describe mathematical notations, their adaptation to our context and notations for the canonical bases of matrix spaces that appear in the proofs.

notation	domain/type	description
M	$\mathbb{N}$	number of output channels
C	$\mathbb{N}$	number of input channels
k	$\mathbb{N}$, odd	the convolution kernel is of support k for signals, $k \times k$ for images
d	{1,2}	1 when the layer applies to signals; 2 for images
S	$\mathbb{N}$	stride/sampling parameter
N	$\mathbb{N}$	input channels are of size SN for signals, $SN \times SN$ for images output channels are of size N for signals, $N \times N$ for images
$\mathbb{K}_d$	vector space	equal to $\mathbb{R}^{M\times C\times k\times k}$ for images or $\mathbb{R}^{M\times C\times k}$ for signals
$\mathbf{K}$	$\mathbb{K}_d$	kernel tensor that contains all weights defining the layer
$\mathcal{K}$	$\mathbb{R}^{MN^2\times CS^2N^2}$ or $\mathbb{R}^{MN\times CSN}$	the matrix that applies the convolutional layer defined by $\mathbf{K}$ to inputs of size defined by N
$\mathbb{K}_d^\perp$	subset of $\mathbb{K}_d$	kernel tensors $\mathbf{K}$ such that $\mathcal{K}$ is orthogonal
$\mathbf{K}_{i,j}$	$\mathbb{R}^{k\times k}$ or $\mathbb{R}^k$	weights of the convolution from input channel j to output channel i
$\mathcal{M}(\mathbf{K}_{i,j})$	$\mathbb{R}^{N^2\times S^2N^2}$ or $\mathbb{R}^{N\times SN}$	matrix that applies the strided convolution defined by $\mathbf{K}_{i,j}$ to inputs of size defined by N
$L_{orth}(\mathbf{K})$	$\mathbb{R}_+$	regularization applied to $\mathbf{K}$ and enforcing orthogonality of $\mathcal{K}$, see Definition 1
$\mathrm{err}_N^F(\mathbf{K})$	$\mathbb{R}_+$	measures, in Frobenius norm, how matrix $\mathcal{K}$ for the signal size N deviates from being orthogonal,
$\mathrm{err}_N^s(\mathbf{K})$	$\mathbb{R}_+$	same as above for the spectral norm
$\mathbf{I}_{r0}$	tensor	tensor appearing in the definition of L_{orth}

Table 2: Summary of the main notations.

The floor of a real number will be denoted by $\lfloor . \rfloor$. For two integers a and b, $[\![a, b]\!]$ denotes the set of integers n such that $a \leq n \leq b$. We also denote by $a\%b$ the rest of the euclidean division of a by b, and $[\![a, b]\!]\%n = \{x\%n | x \in [\![a, b]\!]\}$. We denote by $\delta_{i=j}$, the Kronecker symbol, which is equal to 1 if $i = j$, and 0 if $i \neq j$.

We denote by 0_s the null vector of $\mathbb{R}^s$. For a matrix $A \in \mathbb{R}^{m\times n}$, $\sigma_{max}(A)$ denotes the largest singular value of A and $\|A\|_2 = \sigma_{max}(A)$ is its spectral norm. We also have $\|A\|_1 = \max_{0\leq j\leq n-1} \sum_{i=0}^{m-1} |A_{i,j}|$ and $\|A\|_\infty = \max_{0\leq i\leq m-1} \sum_{j=0}^{n-1} |A_{i,j}|$. We denote by $\mathrm{Id}_n \in \mathbb{R}^{n\times n}$ the identity matrix of size n. We use 'Matlab colon notation' as index of matrices and tensors. For instance, $A_{i,:}$ is the i^{th} line of A.

Recall that $\|.\|_F$ denotes the norm which, to any tensor of order larger than or equal to 2, associates the square root of the sum of the squares of all its elements (e.g., for a matrix it corresponds to the Frobenius norm).

Recall that S is the stride parameter, $k = 2r + 1$ is the size of the 1D kernels. SN is the size of the input channels and N is the size of the output channels.

For a vector space $\mathcal{E}$, we denote by $\mathcal{B}(\mathcal{E})$ its canonical basis. We set

$$\begin{cases} (e_i)_{i=0..k-1} = \mathcal{B}(\mathbb{R}^k) \\ (f_i)_{i=0..SN-1} = \mathcal{B}(\mathbb{R}^{SN}) \\ (E_{a,b})_{a=0..N-1,b=0..SN-1} = \mathcal{B}(\mathbb{R}^{N\times SN}) \\ (\overline{E}_{a,b})_{a=0..SN-1,b=0..N-1} = \mathcal{B}(\mathbb{R}^{SN\times N}) \\ (F_{a,b})_{a=0..SN-1,b=0..SN-1} = \mathcal{B}(\mathbb{R}^{SN\times SN}) \\ (G_{a,b})_{a=0..N-1,b=0..N-1} = \mathcal{B}(\mathbb{R}^{N\times N}) \ . \end{cases} \tag{9}$$

Note that the indices start at 0, thus we have for example $e_0 = \begin{bmatrix} 1 \\ 0_{k-1} \end{bmatrix}$, $e_{k-1} = \begin{bmatrix} 0_{k-1} \\ 1 \end{bmatrix}$, and for all $i \in [\![1, k-2]\!]$, $e_i = \begin{bmatrix} 0_i \\ 1 \\ 0_{k-i-1} \end{bmatrix}$.

To simplify the calculations, the definitions are extended for a, b outside the usual intervals, it is done by periodization. Hence, for all $a, b \in \mathbb{Z}$, denoting by $\hat{a} = a\%SN$, $\tilde{a} = a\%N$, and similarly $\hat{b} = b\%SN$, $\tilde{b} = b\%N$, we set

$$\begin{cases} e_a = e_{a\%k}, & f_a = f_{\hat{a}} \\ E_{a,b} = E_{\tilde{a},\hat{b}}, & \overline{E}_{a,b} = \overline{E}_{\hat{a},\tilde{b}}, \qquad F_{a,b} = F_{\hat{a},\hat{b}}, \qquad G_{a,b} = G_{\tilde{a},\tilde{b}}. \end{cases} \tag{10}$$

Therefore, for all $a, b, c, d \in \mathbb{Z}$, we have

$$\begin{cases} E_{a,b}F_{c,d} = \delta_{\hat{b}=\hat{c}}E_{a,d}, & E_{a,b}\overline{E}_{c,d} = \delta_{\hat{b}=\hat{c}}G_{a,d} \\ \overline{E}_{a,b}E_{c,d} = \delta_{\tilde{b}=\tilde{c}}F_{a,d}, & F_{a,b}\overline{E}_{c,d} = \delta_{\hat{b}=\hat{c}}\overline{E}_{a,d}. \end{cases} \tag{11}$$

Note also that

$$E_{a,b}^T = \overline{E}_{b,a} \ . \tag{12}$$

A.2 Corresponding 1D Definitions

In this section, we give the definitions for signals (1D case), of the objects defined in the introduction for images (2D case).

A.2.1 ORTHOGONALITY

As in Section 1.3.1, we denote by $\mathbf{K} \in \mathbb{R}^{M\times C\times k}$ the kernel tensor and $\mathcal{K} \in \mathbb{R}^{MN\times CSN}$ the matrix that applies the convolutional layer of architecture (M, C, k, S) to C vectorized channels of size SN. Note that, in the 1D case, we need to compare M with CS instead of CS^2.
RO case: When $M \le CS$, $\mathcal{K}$ is orthogonal if and only if $\mathcal{K}\mathcal{K}^T = Id_{MN}$.
CO case: When $M \ge CS$, $\mathcal{K}$ is orthogonal if and only if $\mathcal{K}^T\mathcal{K} = Id_{CSN}$.

A.2.2 THE FUNCTION L_{orth}

We define L_{orth} similarly to the 2D case (see Section 1.3.2 and Figure 1). Formally, for $P \in \mathbb{N}$, and $h, g \in \mathbb{R}^k$, we define

$$\text{conv}(h, g, \text{padding zero} = P, \text{stride} = 1) \in \mathbb{R}^{2P+1} \tag{13}$$

such that for all $i \in [\![0, 2P]\!]$,

$$[\mathrm{conv}(h, g, \text{padding zero} = P, \text{stride} = 1)]_i = \sum_{i'=0}^{k-1} h_{i'} \bar{g}_{i'+i} \ , \tag{14}$$

where $\bar{g}$ is defined for $i \in [\![0, 2P + k - 1]\!]$ as follows

$$\bar{g}_i = \begin{cases} g_{i-P} & \text{if } i \in [\![P, P + k - 1]\!], \\ 0 & \text{otherwise.} \end{cases} \tag{15}$$

Note that, for $P' \leq P$, we have, for all $i \in [\![0, 2P']\!]$,

$$\begin{aligned}&[\mathrm{conv}(h, g, \text{padding zero} = P', \text{stride} = 1)]_i \\ &\qquad = [\mathrm{conv}(h, g, \text{padding zero} = P, \text{stride} = 1)]_{i+P-P'}.\end{aligned} \tag{16}$$

The strided version will be denoted by $\mathrm{conv}(h, g, \text{padding zero} = P, \text{stride} = S) \in \mathbb{R}^{\lfloor 2P/S \rfloor + 1}$ and is defined as follows: For all $i \in [\![0, \lfloor 2P/S \rfloor]\!]$

$$[\mathrm{conv}(h, g, \text{padding zero} = P, \text{stride} = S)]_i = [\mathrm{conv}(h, g, \text{padding zero} = P, \text{stride} = 1)]_{Si}. \tag{17}$$

Finally, reminding that for all $m \in [\![1, M]\!]$ and $c \in [\![1, C]\!]$, $\mathbf{K}_{m,c} \in \mathbb{R}^k$, we denote by

$$\mathbf{conv}(\mathbf{K}, \mathbf{K}, \text{padding zero} = P, \text{stride} = S) \in \mathbb{R}^{M \times M \times (\lfloor 2P/S \rfloor + 1)}$$

the third-order tensor such that, for all $m, l \in [\![1, M]\!]$,

$$\begin{aligned}&\mathbf{conv}(\mathbf{K}, \mathbf{K}, \text{padding zero} = P, \text{stride} = S)_{m,l,:} \\ &\qquad = \sum_{c=1}^{C} \mathrm{conv}(\mathbf{K}_{m,c}, \mathbf{K}_{l,c}, \text{padding zero} = P, \text{stride} = S).\end{aligned} \tag{18}$$

From now on, we take $P = \left\lfloor \frac{k-1}{S} \right\rfloor S$ and $\mathbf{I}_{r0} \in \mathbb{R}^{M \times M \times (2P/S+1)}$ the tensor whose entries are all zero except its central $M \times M$ entry which is equal to an identity matrix: $[\mathbf{I}_{r0}]_{:,:,P/S} = Id_M$. Put differently, we have for all $m, l \in [\![1, M]\!]$,

$$[\mathbf{I}_{r0}]_{m,l,:} = \delta_{m=l} \begin{bmatrix} 0_{P/S} \\ 1 \\ 0_{P/S} \end{bmatrix} \ . \tag{19}$$

And L_{orth} for 1D convolutions is defined as follows:

- In the RO case:

$$L_{orth}(\mathbf{K}) = \| \mathbf{conv}(\mathbf{K}, \mathbf{K}, \text{padding zero} = P, \text{stride} = S) - \mathbf{I}_{r0} \|_F^2 \ .$$

- In the CO case:

$$L_{orth}(\mathbf{K}) = \| \mathbf{conv}(\mathbf{K}, \mathbf{K}, \text{padding zero} = P, \text{stride} = S) - \mathbf{I}_{r0} \|_F^2 - (M - CS) \ .$$

A.2.3 MEASURES OF DEVIATION FROM ORTHOGONALITY

The orthogonality errors are defined by

$$\mathrm{err}_N^F(\mathbf{K}) = \begin{cases} \|\mathcal{K}\mathcal{K}^T - \mathrm{Id}_{MN}\|_F & \text{, in the RO case,} \\ \|\mathcal{K}^T\mathcal{K} - \mathrm{Id}_{CSN}\|_F & \text{, in the CO case,} \end{cases}$$

and

$$\mathrm{err}_N^s(\mathbf{K}) = \begin{cases} \|\mathcal{K}\mathcal{K}^T - \mathrm{Id}_{MN}\|_2 & \text{, in the RO case,} \\ \|\mathcal{K}^T\mathcal{K} - \mathrm{Id}_{CSN}\|_2 & \text{, in the CO case.} \end{cases}$$

Appendix B. The Convolutional Layer as a Matrix-Vector Product

In this section, we write the convolutional layer as a matrix-vector product. In other words, we explicit $\mathcal{K}$ and the ingredients composing it. The notation and preliminary results are useful in the proofs. Note that the results are already known and can be found for example in Sedghi et al. (2018).

B.1 1D Case

We denote by $S_N \in \mathbb{R}^{N\times SN}$ the sampling matrix (i.e., for $x = (x_0, \dots, x_{SN-1})^T \in \mathbb{R}^{SN}$, we have for all $m \in [\![0, N-1]\!]$, $(S_N x)_m = x_{Sm}$).
Put differently, we have

$$S_N = \sum_{i=0}^{N-1} E_{i,Si} \, . \tag{20}$$

Also, note that, using (11) and (12), we have $S_N S_N^T = Id_N$ and

$$S_N^T S_N = \sum_{i=0}^{N-1} F_{Si,Si} \, . \tag{21}$$

For a vector $x = (x_0, \dots, x_{n-1})^T \in \mathbb{R}^n$, we denote by $C(x) \in \mathbb{R}^{n\times n}$ the circulant matrix defined by

$$C(x) = \begin{pmatrix} x_0 & x_{n-1} & \cdots & x_2 & x_1 \\ x_1 & x_0 & x_{n-1} & & x_2 \\ \vdots & x_1 & x_0 & \ddots & \vdots \\ x_{n-2} & & \ddots & \ddots & x_{n-1} \\ x_{n-1} & x_{n-2} & \cdots & x_1 & x_0 \end{pmatrix} . \tag{22}$$

In other words, for $x \in \mathbb{R}^n$ and $X \in \mathbb{R}^{n\times n}$, we have

$$X = C(x) \iff \forall m, l \in [\![0, n-1]\!],\ X_{m,l} = x_{(m-l)\%n} \, . \tag{23}$$

The notation for the circulant matrix $C(.)$ should not be confused with the number of the input channels C. We also denote by $\tilde{x} \in \mathbb{R}^n$ the vector such that for all $i \in [\![0, n-1]\!]$, $\tilde{x}_i = x_{(-i)\%n}$. Again, the notation $\tilde{x}$, for $x \in \mathbb{R}^n$, should not be confused with $\tilde{a}$, for $a \in \mathbb{Z}$. We have

$$C(x)^T = C(\tilde{x}) \, . \tag{24}$$

Also, for $x, y \in \mathbb{R}^n$, we have

$$C(x)C(y) = C(x * y), \tag{25}$$

where $x * y \in \mathbb{R}^n$, is such that for all $j \in [\![0, n-1]\!]$,

$$[x * y]_j = \sum_{i=0}^{n-1} x_i y_{(j-i)\%n}. \tag{26}$$

$x * y$ is extended by n-periodicity. Note that here $x * y$ denotes the classical convolution as defined in math (i.e. by flipping the second argument). Note also that $x * y = y * x$ and therefore

$$C(x)C(y) = C(y)C(x)\ . \tag{27}$$

Throughout the article, the size of a filter is smaller than the size of the signal ($k = 2r + 1 \leq SN$). For $n \geq k$, we introduce an embedding P_n which associates to each $h = (h_0, \dots, h_{2r})^T \in \mathbb{R}^k$ the corresponding vector

$$P_n(h) = (h_r, \dots, h_1, h_0, 0, \dots, 0, h_{2r}, \dots, h_{r+1})^T \in \mathbb{R}^n\ .$$

Setting $[P_n(h)]_i = [P_n(h)]_{i\%n}$ for all $i \in \mathbb{Z}$, we have the following formula for P_n: for $i \in [\![-r, -r+n-1]\!]$,

$$[P_n(h)]_i = \begin{cases} h_{r-i} & \text{if } i \in [\![-r, r]\!] \\ 0 & \text{otherwise.} \end{cases} \tag{28}$$

Single-channel case: Let $x = (x_0, \dots, x_{SN-1})^T \in \mathbb{R}^{SN}$ be a 1D signal. We denote by Circular_Conv$(h, x, \text{stride} = 1)$ the result of the circular convolution[13] of x with the kernel $h = (h_0, \dots, h_{2r})^T \in \mathbb{R}^k$. We have

$$\text{Circular\_Conv}(h, x, \text{stride} = 1) = \left(\sum_{i'=0}^{k-1} h_{i'} x_{(i'+i-r)\%SN} \right)_{i=0..SN-1} .$$

13. as defined in machine learning (we do not flip h).

Written as a matrix-vector product, this becomes

$$\begin{pmatrix} h_0 & \cdots & h_{2r} & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & h_0 & \cdots & h_{2r} \end{pmatrix} \begin{pmatrix} x_{SN-r} \\ \vdots \\ x_{SN-1} \\ x_0 \\ \vdots \\ x_{SN-1} \\ x_0 \\ \vdots \\ x_{r-1} \end{pmatrix} \in \mathbb{R}^{SN}$$

$$= \begin{pmatrix} h_r & h_{r+1} & \cdots & h_{2r} & 0 & \cdots & 0 & h_0 & \cdots & h_{r-1} \\ h_{r-1} & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & h_0 \\ h_0 & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & h_{2r} \\ h_{2r} & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & h_{r+1} \\ h_{r+1} & \cdots & h_{2r} & 0 & \cdots & 0 & h_0 & \cdots & h_{r-1} & h_r \end{pmatrix} x$$

$$= C(P_{SN}(h))x \ .$$

The strided convolution is

$$\text{Circular\_Conv}(h, x, \text{stride} = S) = S_N C(P_{SN}(h))x \in \mathbb{R}^N \ . \tag{29}$$

Notice that $S_N C(P_{SN}(h)) \in \mathbb{R}^{N \times SN}$.

Multi-channel convolution: Let $X \in \mathbb{R}^{C \times SN}$ be a multi-channel 1D signal. We denote by Circular_Conv$(\mathbf{K}, X, \text{stride} = S)$ the result of the strided circular convolutional layer of kernel $\mathbf{K} \in \mathbb{R}^{M \times C \times k}$ applied to X. Using (29) for all the input-output channel correspondences, we have $Y = \text{Circular\_Conv}(\mathbf{K}, X, \text{stride} = S) \in \mathbb{R}^{M \times N}$ if and only if

$$\text{Vect}(Y) = \begin{pmatrix} S_N C(P_{SN}(\mathbf{K}_{1,1})) & \dots & S_N C(P_{SN}(\mathbf{K}_{1,C})) \\ \vdots & \vdots & \vdots \\ S_N C(P_{SN}(\mathbf{K}_{M,1})) & \dots & S_N C(P_{SN}(\mathbf{K}_{M,C})) \end{pmatrix} \text{Vect}(X) \ ,$$

where $\mathbf{K}_{i,j} = \mathbf{K}_{i,j,:} \in \mathbb{R}^k$. Therefore,

$$\mathcal{K} = \begin{pmatrix} S_N C(P_{SN}(\mathbf{K}_{1,1})) & \dots & S_N C(P_{SN}(\mathbf{K}_{1,C})) \\ \vdots & \vdots & \vdots \\ S_N C(P_{SN}(\mathbf{K}_{M,1})) & \dots & S_N C(P_{SN}(\mathbf{K}_{M,C})) \end{pmatrix} \in \mathbb{R}^{MN \times CSN} \tag{30}$$

is the layer transform matrix associated to kernel $\mathbf{K}$.

B.2 2D Case

Notice that, since they are very similar, the proofs and notation are detailed in the 1D case, but we only provide a sketch of the proof and the main equations in 2D. In order to distinguish between the 1D and 2D versions of $C(.)$, P_n and S_N, we use calligraphic symbols in the 2D case. We denote by $\mathcal{S}_N \in \mathbb{R}^{N^2 \times S^2N^2}$ the sampling matrix in the 2D case (i.e., for a matrix $x \in \mathbb{R}^{SN \times SN}$, if we denote by $z \in \mathbb{R}^{N \times N}$, such that for all $i, j \in [\![0, N-1]\!]$, $z_{i,j} = x_{Si,Sj}$, then $\mathrm{Vect}(z) = \mathcal{S}_N \mathrm{Vect}(x)$).
For a matrix $x \in \mathbb{R}^{n \times n}$, we denote by $\mathcal{C}(x) \in \mathbb{R}^{n^2 \times n^2}$ the doubly-block circulant matrix defined by

$$\mathcal{C}(x) = \begin{pmatrix} C(x_{0,:}) & C(x_{n-1,:}) & \cdots & C(x_{2,:}) & C(x_{1,:}) \\ C(x_{1,:}) & C(x_{0,:}) & C(x_{n-1,:}) & & C(x_{2,:}) \\ \vdots & C(x_{1,:}) & C(x_{0,:}) & \ddots & \vdots \\ C(x_{n-2,:}) & & \ddots & \ddots & C(x_{n-1,:}) \\ C(x_{n-1,:}) & C(x_{n-2,:}) & \cdots & C(x_{1,:}) & C(x_{0,:}) \end{pmatrix} .$$

For $n \geq k = 2r+1$, we introduce the operator $\mathcal{P}_n$ which associates to a matrix $h \in \mathbb{R}^{k \times k}$ the corresponding matrix

$$\mathcal{P}_n(h) = \begin{pmatrix} h_{r,r} & \cdots & h_{r,0} & 0 & \cdots & 0 & h_{r,2r} & \cdots & h_{r,r+1} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ h_{0,r} & \cdots & h_{0,0} & 0 & \cdots & 0 & h_{0,2r} & \cdots & h_{0,r+1} \\ 0 & \ldots & 0 & 0 & \ldots & 0 & 0 & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \ldots & 0 & 0 & \ldots & 0 & 0 & \ldots & 0 \\ h_{2r,r} & \cdots & h_{2r,0} & 0 & \cdots & 0 & h_{2r,2r} & \cdots & h_{2r,r+1} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ h_{r+1,r} & \cdots & h_{r+1,0} & 0 & \cdots & 0 & h_{r+1,2r} & \cdots & h_{r+1,r+1} \end{pmatrix} \in \mathbb{R}^{n \times n} .$$

Setting $[\mathcal{P}_n(h)]_{i,j} = [\mathcal{P}_n(h)]_{i\%n, j\%n}$ for all $i, j \in \mathbb{Z}$, we have the following formula for $\mathcal{P}_n$: for $(i,j) \in [\![-r, -r+n-1]\!]^2$,

$$[\mathcal{P}_n(h)]_{i,j} = \begin{cases} h_{r-i,r-j} & \text{if } (i,j) \in [\![-r, r]\!]^2 \\ 0 & \text{otherwise.} \end{cases}$$

Single-channel case: Let $x \in \mathbb{R}^{SN \times SN}$ be a 2D image. We denote by Circular_Conv$(h, x, \text{stride} = 1)$ the result of the circular convolution of x with the kernel $h \in \mathbb{R}^{k \times k}$. As in the 1D case, we have

$$y = \text{Circular\_Conv}(h, x, \text{stride} = 1) \iff \mathrm{Vect}(y) = \mathcal{C}(\mathcal{P}_{SN}(h)) \, \mathrm{Vect}(x)$$

and the strided circular convolution

$$y = \text{Circular\_Conv}(h, x, \text{stride} = S) \iff \mathrm{Vect}(y) = \mathcal{S}_N \mathcal{C}(\mathcal{P}_{SN}(h)) \, \mathrm{Vect}(x) .$$

Notice that $\mathcal{S}_N \mathcal{C}(\mathcal{P}_{SN}(h)) \in \mathbb{R}^{N^2 \times S^2N^2}$.

Multi-channel convolution : Let $X \in \mathbb{R}^{C \times SN \times SN}$ be a multi-channel 2D image. We denote by Circular_Conv$(\mathbf{K}, X, \text{stride} = S)$ the result of the strided circular convolutional layer of kernel

$\mathbf{K} \in \mathbb{R}^{M\times C\times k\times k}$ applied to X. We have $Y = \text{Circular\_Conv}(\mathbf{K}, X, \text{stride} = S) \in \mathbb{R}^{M\times N\times N}$ if and only if

$$\text{Vect}(Y) = \begin{pmatrix} \mathcal{S}_N\mathcal{C}(\mathcal{P}_{SN}(\mathbf{K}_{1,1})) & \dots & \mathcal{S}_N\mathcal{C}(\mathcal{P}_{SN}(\mathbf{K}_{1,C})) \\ \vdots & \vdots & \vdots \\ \mathcal{S}_N\mathcal{C}(\mathcal{P}_{SN}(\mathbf{K}_{M,1})) & \dots & \mathcal{S}_N\mathcal{C}(\mathcal{P}_{SN}(\mathbf{K}_{M,C})) \end{pmatrix} \text{Vect}(X) ,$$

where $\mathbf{K}_{i,j} = \mathbf{K}_{i,j,:,:} \in \mathbb{R}^{k\times k}$. Therefore,

$$\mathcal{K} = \begin{pmatrix} \mathcal{S}_N\mathcal{C}(\mathcal{P}_{SN}(\mathbf{K}_{1,1})) & \dots & \mathcal{S}_N\mathcal{C}(\mathcal{P}_{SN}(\mathbf{K}_{1,C})) \\ \vdots & \vdots & \vdots \\ \mathcal{S}_N\mathcal{C}(\mathcal{P}_{SN}(\mathbf{K}_{M,1})) & \dots & \mathcal{S}_N\mathcal{C}(\mathcal{P}_{SN}(\mathbf{K}_{M,C})) \end{pmatrix} \in \mathbb{R}^{MN^2\times CS^2N^2}$$

is the layer transform matrix associated to kernel $\mathbf{K}$.

Appendix C. Proof of Theorem 4

As the proofs are very similar in the 1D and 2D cases, we give the full proof in the 1D case, in Section C.1, and we only give a sketch of the proof in the 2D case, in Section C.2.

We first prove the result in the RO case, then in the CO case. In each case, we prove separately the statement when an orthogonal convolutional layer exists and when no orthogonal convolutional layer exists. When the architecture places us in the former case, to prove an orthogonal convolutional layer exists, we exhibit an explicit kernel tensor $\mathbf{K}$ and do the calculations to prove that $\mathcal{K}$ is orthogonal. The calculations are based on Lemma 10, in the RO case, and Lemma 11, in the CO case. Lemma 9 synthesize the result of a calculation that is used to prove both Lemma 10 and Lemma 11. When on the contrary the architecture is such that there does not exist any orthogonal convolutional layer, we prove that for all $\mathbf{K}$ the architecture condition implies $\text{rk}(\mathcal{K}\mathcal{K}^T) < \text{rk}(Id_{MN})$, in the RO case, and $\text{rk}(\mathcal{K}^T\mathcal{K}) < \text{rk}(Id_{CSN})$, in the CO case. This proves that no orthogonal convolutional layer exists.

C.1 Proof of Theorem 4, for 1D Convolutional Layers

We start by stating and proving three intermediate lemmas. Recall that $k = 2r + 1$ and from (9), that $(e_i)_{i=0..k-1} = \mathcal{B}(\mathbb{R}^k)$ and $(E_{a,b})_{a=0..N-1,b=0..SN-1} = \mathcal{B}(\mathbb{R}^{N\times SN})$.

Lemma 9 *Let $j \in [\![0, k-1]\!]$. We have*

$$S_N C\left(P_{SN}\left(e_j\right)\right) = \sum_{i=0}^{N-1} E_{i,Si+j-r} \ .$$

Proof Let $j \in [\![0, k-1]\!]$. Using (28), (9), (10) and (22), we have

$$C(P_{SN}(e_j)) = C(f_{r-j}) = \sum_{i=0}^{SN-1} F_{i,i-(r-j)} = \sum_{i=0}^{SN-1} F_{i,i+j-r} \ .$$

Using (20) and (11), we have

$$S_N C\left(P_{SN}\left(e_j\right)\right) = \left(\sum_{i=0}^{N-1} E_{i,Si}\right)\left(\sum_{i'=0}^{SN-1} F_{i',i'+j-r}\right) = \sum_{i=0}^{N-1} E_{i,Si+j-r} \ .$$

■

Lemma 10 *Let* $k_S = \min(k, S)$ *and* $j, l \in [\![0, k_S - 1]\!]$. *We have*

$$S_N C(P_{SN}(e_j))C(P_{SN}(e_l))^T S_N^T = \delta_{j=l} Id_N \ .$$

Proof Let $j, l \in [\![0, k_S - 1]\!]$. Since $k_S \leq k$, using Lemma 9 and (12),

$$\begin{aligned} S_N C\left(P_{SN}\left(e_j\right)\right) C\left(P_{SN}\left(e_l\right)\right)^T S_N^T &= \left(\sum_{i=0}^{N-1} E_{i,Si+j-r}\right)\left(\sum_{i'=0}^{N-1} E_{i',Si'+l-r}\right)^T \\ &= \left(\sum_{i=0}^{N-1} E_{i,Si+j-r}\right)\left(\sum_{i'=0}^{N-1} \overline{E}_{Si'+l-r,i'}\right) \ . \end{aligned} \tag{31}$$

We know from (11) that $E_{i,Si+j-r}\overline{E}_{Si'+l-r,i'} = \delta_{\widehat{Si+j-r}=\widehat{Si'+l-r}} G_{i,i'}$. But for $i, i' \in [\![0, N-1]\!]$ and $j, l \in [\![0, k_S - 1]\!]$, since $k_S \leq S$, we have

$$-r \leq Si + j - r \leq S(N-1) + k_S - 1 - r \leq SN - 1 - r.$$

Similarly, $Si' + l - r \in [\![-r, SN - 1 - r]\!]$. Therefore, $Si + j - r$ and $Si' + l - r$ lie in the same interval of size SN, hence

$$\widehat{Si + j - r} = \widehat{Si' + l - r} \iff Si + j - r = Si' + l - r \iff Si + j = Si' + l \ .$$

If $Si + j = Si' + l$, then

$$|S(i - i')| = |j - l| < k_S \leq S.$$

Since $|i - i'| \in \mathbb{N}$, the latter inequality implies $i = i'$ and, as a consequence, $j = l$. Finally,

$$\widehat{Si + j - r} = \widehat{Si' + l - r} \iff i = i' \text{ and } j = l \ .$$

Hence, using (11), the equality (31) becomes

$$S_N C(P_{SN}(e_j))C(P_{SN}(e_l))^T S_N^T = \delta_{j=l} \sum_{i=0}^{N-1} G_{i,i} = \delta_{j=l} Id_N \ .$$

■

Lemma 11 *Let* $S \leq k$. *We have*

$$\sum_{z=0}^{S-1} C(P_{SN}(e_z))^T S_N^T S_N C(P_{SN}(e_z)) = Id_{SN} \ .$$

Proof Let $z \in [\![0, S-1]\!]$. Since $S \leq k$, we have $z \in [\![0, k-1]\!]$. Hence using Lemma 9, then (12) and (11), we have

$$\begin{aligned} C\left(P_{SN}\left(e_z\right)\right)^T S_N^T S_N C\left(P_{SN}\left(e_z\right)\right) &= \left(\sum_{i=0}^{N-1} E_{i,Si+z-r}\right)^T \left(\sum_{i'=0}^{N-1} E_{i',Si'+z-r}\right) \\ &= \left(\sum_{i=0}^{N-1} \overline{E}_{Si+z-r,i}\right) \left(\sum_{i'=0}^{N-1} E_{i',Si'+z-r}\right) \\ &= \sum_{i=0}^{N-1} F_{Si+z-r,Si+z-r} \ . \end{aligned}$$

Hence

$$\sum_{z=0}^{S-1} C(P_{SN}(e_z))^T S_N^T S_N C(P_{SN}(e_z)) = \sum_{z=0}^{S-1} \sum_{i=0}^{N-1} F_{Si+z-r,Si+z-r} \ .$$

But, for $z \in [\![0, S-1]\!]$ and $i \in [\![0, N-1]\!]$, $Si+z-r$ traverses $[\![-r, SN-1-r]\!]$. Therefore, using (10)

$$\sum_{z=0}^{S-1} C(P_{SN}(e_z))^T S_N^T S_N C(P_{SN}(e_z)) = \sum_{i=-r}^{SN-1-r} F_{i,i} = \sum_{i=0}^{SN-1} F_{i,i} = Id_{SN} \ .$$

■

Proof [Proof of Theorem 4] Let N be a positive integer such that $SN \geq k$.
We start by proving the theorem in the RO case.
Suppose $CS \geq M$ **and** $M \leq Ck$**:**
Let us exhibit $\mathbf{K} \in \mathbb{R}^{M\times C\times k}$ such that $\mathcal{K}\mathcal{K}^T = Id_{MN}$.
Let $k_S = \min(k, S)$. Since $M \leq CS$ and $M \leq Ck$, we have $1 \leq M \leq Ck_S$. Therefore, there exist a unique couple $(i_{max}, j_{max}) \in [\![0, k_S-1]\!] \times [\![1, C]\!]$ such that $M = i_{max}C + j_{max}$. We define the kernel tensor $\mathbf{K} \in \mathbb{R}^{M\times C\times k}$ as follows: For all $(i,j) \in [\![0, k_S-1]\!] \times [\![1, C]\!]$ such that $iC + j \leq M$, we set $\mathbf{K}_{iC+j,j} = e_i$, and $\mathbf{K}_{u,v} = 0$ for all the other indices. Put differently, if we write $\mathbf{K}$ as a 3rd order tensor (where the rows represent the first dimension, the columns the second one, and the $\mathbf{K}_{i,j} \in \mathbb{R}^k$ are in the third dimension) we have :

$$\mathbf{K} = \begin{bmatrix} \mathbf{K}_{1,1} & \cdots & \mathbf{K}_{1,C} \\ \vdots & \ddots & \vdots \\ \mathbf{K}_{C,1} & \cdots & \mathbf{K}_{C,C} \\ \mathbf{K}_{C+1,1} & \cdots & \mathbf{K}_{C+1,C} \\ \vdots & \ddots & \vdots \\ \mathbf{K}_{2C,1} & \cdots & \mathbf{K}_{2C,C} \\ & \vdots & \\ \mathbf{K}_{i_{max}C+1,1} & \cdots & \mathbf{K}_{i_{max}C+1,C} \\ \vdots & \ddots & \vdots \end{bmatrix} = \begin{bmatrix} e_0 & & \\ 0 & \ddots & 0 \\ & & e_0 \\ e_1 & & \\ 0 & \ddots & 0 \\ & & e_1 \\ & \vdots & \\ e_{i_{max}} & & \\ 0 & \ddots & 0 \end{bmatrix} \in \mathbb{R}^{M\times C\times k} ,$$

where $e_{i_{max}}$ appears j_{max} times. Therefore, using (30), we have

$$\mathcal{K} = \begin{bmatrix} S_N C(P_{SN}(e_0)) & & \\ 0 & \ddots & 0 \\ & & S_N C(P_{SN}(e_0)) \\ S_N C(P_{SN}(e_1)) & & \\ 0 & \ddots & 0 \\ & & S_N C(P_{SN}(e_1)) \\ & \vdots & \\ S_N C(P_{SN}(e_{i_{max}})) & & \\ 0 & \ddots & 0 \end{bmatrix} \in \mathbb{R}^{MN \times CSN} ,$$

where $S_N C(P_{SN}(e_{i_{max}}))$ appears j_{max} times. We have $\mathcal{K} = D_{1:MN,:}$, where we set

$$D = \begin{bmatrix} S_N C(P_{SN}(e_0)) & & \\ 0 & \ddots & 0 \\ & & S_N C(P_{SN}(e_0)) \\ S_N C(P_{SN}(e_1)) & & \\ 0 & \ddots & 0 \\ & & S_N C(P_{SN}(e_1)) \\ & \vdots & \\ S_N C(P_{SN}(e_{k_S-1})) & & \\ 0 & \ddots & 0 \\ & & S_N C(P_{SN}(e_{k_S-1})) \end{bmatrix} \in \mathbb{R}^{k_S CN \times CSN} .$$

But, for $j, l \in [\![0, k_S - 1]\!]$, the (j, l)-th block of size (CN, CN) of DD^T is :

$$\begin{bmatrix} S_N C(P_{SN}(e_j)) & & \\ 0 & \ddots & 0 \\ & & S_N C(P_{SN}(e_j)) \end{bmatrix} \begin{bmatrix} C(P_{SN}(e_l))^T S_N^T & & \\ 0 & \ddots & 0 \\ & & C(P_{SN}(e_l))^T S_N^T \end{bmatrix} ,$$

which is equal to

$$\begin{bmatrix} S_N C(P_{SN}(e_j)) C(P_{SN}(e_l))^T S_N^T & & \\ 0 & \ddots & 0 \\ & & S_N C(P_{SN}(e_j)) C(P_{SN}(e_l))^T S_N^T \end{bmatrix} .$$

Using Lemma 10, this is equal to $\delta_{j=l} Id_{CN}$. Hence, $DD^T = Id_{k_S CN}$, and therefore,

$$\mathcal{K}\mathcal{K}^T = D_{1:MN,:}(D_{1:MN,:})^T = (DD^T)_{1:MN,1:MN} = Id_{MN} .$$

This proves the first implication in the RO case, i.e., if $M \leq Ck$, then $\mathbb{K}_1^{\perp} \neq \emptyset$.

Suppose $CS \geq M$ and $M > Ck$:
We need to prove that for all $\mathbf{K} \in \mathbb{R}^{M \times C \times k}$, we have $\mathcal{K}\mathcal{K}^T \neq Id_{MN}$.
Since for all $(i,j) \in [\![1,M]\!] \times [\![1,C]\!]$, each of the N rows of $S_N C(P_{SN}(\mathbf{K}_{i,j}))$ has at most k non-zero elements, the number of non-zero columns of $S_N C(P_{SN}(\mathbf{K}_{i,j}))$ is less than or equal to kN. Also, for all $i, i' \in [\![1,M]\!]$, the columns of $S_N C(P_{SN}(\mathbf{K}_{i,j}))$ which can be non-zero are the same as those of $S_N C(P_{SN}(\mathbf{K}_{i',j}))$. Hence, we have for all j, the number of non-zero columns of $\begin{bmatrix} S_N C(P_{SN}(\mathbf{K}_{1,j})) \\ \vdots \\ S_N C(P_{SN}(\mathbf{K}_{M,j})) \end{bmatrix}$ is less than or equal to kN. Therefore, the number of non-zero columns of $\mathcal{K}$ is less than or equal to CkN. Hence, since $Ck < M$, we have $\text{rk}(\mathcal{K}\mathcal{K}^T) \leq \text{rk}(\mathcal{K}) \leq CkN < MN = \text{rk}(Id_{MN})$. Therefore, for all $\mathbf{K} \in \mathbb{R}^{M \times C \times k}$, we have $\mathcal{K}\mathcal{K}^T \neq Id_{MN}$.
This proves that if $CS \geq M$ and $M > Ck$, then $\mathbb{K}_1^\perp = \emptyset$. This concludes the proof in the RO case.

Suppose $M \geq CS$ and $S \leq k$:
Let us exhibit $\mathbf{K} \in \mathbb{R}^{M \times C \times k}$ such that $\mathcal{K}^T\mathcal{K} = Id_{CSN}$.
For all $(i,j) \in [\![0, S-1]\!] \times [\![1,C]\!]$, we set $\mathbf{K}_{iC+j,j} = e_i$, and $\mathbf{K}_{u,v} = 0$ for all the other indices. Put differently, if we write $\mathbf{K}$ as a 3rd order tensor, we have

$$\mathbf{K} = \begin{bmatrix} \mathbf{K}_{1,1} & \cdots & \mathbf{K}_{1,C} \\ \vdots & \ddots & \vdots \\ \mathbf{K}_{C,1} & \cdots & \mathbf{K}_{C,C} \\ \mathbf{K}_{C+1,1} & \cdots & \mathbf{K}_{C+1,C} \\ \vdots & \ddots & \vdots \\ \mathbf{K}_{2C,1} & \cdots & \mathbf{K}_{2C,C} \\ & \vdots & \\ \mathbf{K}_{(S-1)C+1,1} & \cdots & \mathbf{K}_{(S-1)C+1,C} \\ \vdots & \ddots & \vdots \\ \mathbf{K}_{CS,1} & \cdots & \mathbf{K}_{CS,C} \\ \mathbf{K}_{CS+1,1} & \cdots & \mathbf{K}_{CS+1,C} \\ \vdots & \vdots & \vdots \\ \mathbf{K}_{M,1} & \cdots & \mathbf{K}_{M,C} \end{bmatrix} = \begin{bmatrix} e_0 & & \\ 0 & \ddots & 0 \\ & & e_0 \\ e_1 & & \\ 0 & \ddots & 0 \\ & & e_1 \\ & \vdots & \\ e_{S-1} & & \\ 0 & \ddots & 0 \\ & & e_{S-1} \\ & & \\ & O & \\ & & \end{bmatrix} \in \mathbb{R}^{M \times C \times k} ,$$

where $O = 0_{(M-CS)\times C\times k}$ denotes the null tensor. Therefore, using (30), we have

$$\mathcal{K} = \begin{bmatrix} S_N C(P_{SN}(e_0)) & & \\ 0 & \ddots & 0 \\ & & S_N C(P_{SN}(e_0)) \\ S_N C(P_{SN}(e_1)) & & \\ 0 & \ddots & 0 \\ & & S_N C(P_{SN}(e_1)) \\ & \vdots & \\ S_N C(P_{SN}(e_{S-1})) & & \\ 0 & \ddots & 0 \\ & & S_N C(P_{SN}(e_{S-1})) \\ & \mathcal{O} & \end{bmatrix} \in \mathbb{R}^{MN\times CSN} ,$$

where $\mathcal{O} = 0_{(MN-CSN)\times CSN}$ denotes the null matrix. Hence, $\mathcal{K}^T\mathcal{K}$ equals

$$\begin{bmatrix} \sum_{z=0}^{S-1} C(P_{SN}(e_z))^T S_N^T S_N C(P_{SN}(e_z)) & & 0 \\ & \ddots & \\ 0 & & \sum_{z=0}^{S-1} C(P_{SN}(e_z))^T S_N^T S_N C(P_{SN}(e_z)) \end{bmatrix} .$$

Using Lemma 11, we obtain $\mathcal{K}^T\mathcal{K} = Id_{CSN}$.
This proves that in the CO case, if $S \leq k$, then $\mathbb{K}_1^\perp \neq \emptyset$.

Suppose $M \geq CS$ and $S > k$:
We need to prove that for all $\mathbf{K} \in \mathbb{R}^{M\times C\times k}$, we have $\mathcal{K}^T\mathcal{K} \neq Id_{CSN}$.
Following the same reasoning as in the case $CS \geq M$ and $M > Ck$, we have that the number of non-zero columns of $\mathcal{K}$ is less than or equal to CkN. So, since $k < S$, we have rk$(\mathcal{K}^T\mathcal{K}) \leq$ rk$(\mathcal{K}) \leq CkN < CSN =$ rk(Id_{CSN}). Therefore, for all $\mathbf{K} \in \mathbb{R}^{M\times C\times k}$, we have $\mathcal{K}^T\mathcal{K} \neq Id_{CSN}$.
This proves that in the CO case, if $k < S$, then $\mathbb{K}_1^\perp = \emptyset$. This concludes the proof. ■

C.2 Sketch of the Proof of Theorem 4, for 2D Convolutional Layers

We first set $(e_{i,j})_{i=0..k-1,j=0..k-1} = \mathcal{B}(\mathbb{R}^{k\times k})$. As in the 1D case, we have the following two lemmas

Lemma 12 *Let $k_S = \min(k, S)$ and $j, j', l, l' \in [\![0, k_S - 1]\!]$. We have*

$$\mathcal{S}_N \mathcal{C}(\mathcal{P}_{SN}(e_{j,j'}))\mathcal{C}(\mathcal{P}_{SN}(e_{l,l'}))^T \mathcal{S}_N^T = \delta_{j=l}\delta_{j'=l'} Id_{N^2} .$$

Lemma 13 *Let $S \leq k$. We have*

$$\sum_{z=0}^{S-1}\sum_{z'=0}^{S-1} \mathcal{C}(\mathcal{P}_{SN}(e_{z,z'}))^T \mathcal{S}_N^T \mathcal{S}_N \mathcal{C}(\mathcal{P}_{SN}(e_{z,z'})) = Id_{S^2N^2} .$$

For $CS^2 \geq M$ and $M \leq Ck^2$:
We set $\overline{e}_{i+kj} = e_{i,j}$ for $i, j \in [\![0, k-1]\!]$.
Let $i_{max}, j_{max} \in [\![0, k_S^2 - 1]\!] \times [\![1, C]\!]$ such that $i_{max}C + j_{max} = M$. We set

$$\mathbf{K} = \begin{bmatrix} \mathbf{K}_{1,1} & \cdots & \mathbf{K}_{1,C} \\ \vdots & \ddots & \vdots \\ \mathbf{K}_{C,1} & \cdots & \mathbf{K}_{C,C} \\ \mathbf{K}_{C+1,1} & \cdots & \mathbf{K}_{C+1,C} \\ \vdots & \ddots & \vdots \\ \mathbf{K}_{2C,1} & \cdots & \mathbf{K}_{2C,C} \\ & \vdots & \\ \mathbf{K}_{i_{max}C+1,1} & \cdots & \mathbf{K}_{i_{max}C+1,C} \\ \vdots & \ddots & \vdots \end{bmatrix} = \begin{bmatrix} \overline{e}_0 & & \\ 0 & \ddots & 0 \\ & & \overline{e}_0 \\ \overline{e}_1 & & \\ 0 & \ddots & 0 \\ & & \overline{e}_1 \\ & \vdots & \\ \overline{e}_{i_{max}} & & \\ 0 & \ddots & 0 \end{bmatrix} \in \mathbb{R}^{M \times C \times k \times k} ,$$

where $\overline{e}_{i_{max}}$ appears j_{max} times. Then we proceed as in the 1D case.

For $CS^2 \geq M$ and $M > Ck^2$:
Using the same argument as in 1D, we can conclude that the number of non-zero columns of $\mathcal{K}$ is less than or equal to Ck^2N^2. Hence, rk$(\mathcal{K}) \leq Ck^2N^2 < MN^2$. Therefore, for all $\mathbf{K} \in \mathbb{R}^{M \times C \times k \times k}$, we have $\mathcal{K}\mathcal{K}^T \neq Id_{MN^2}$.

For $M \geq CS^2$ and $S \leq k$:
Denoting by $O \in \mathbb{R}^{(M-CS^2) \times C \times k \times k}$ the null 4th order tensor of size $(M - CS^2) \times C \times k \times k$, we set

$$\mathbf{K} = \begin{bmatrix} \mathbf{K}_{1,1} & \cdots & \mathbf{K}_{1,C} \\ \vdots & \ddots & \vdots \\ \mathbf{K}_{C,1} & \cdots & \mathbf{K}_{C,C} \\ \mathbf{K}_{C+1,1} & \cdots & \mathbf{K}_{C+1,C} \\ \vdots & \ddots & \vdots \\ \mathbf{K}_{2C,1} & \cdots & \mathbf{K}_{2C,C} \\ & \vdots & \\ \mathbf{K}_{C(S^2-1)+1,1} & \cdots & \mathbf{K}_{C(S^2-1)+1,C} \\ \vdots & \ddots & \vdots \\ \mathbf{K}_{CS^2,1} & \cdots & \mathbf{K}_{CS^2,C} \\ \mathbf{K}_{CS^2+1,1} & \cdots & \mathbf{K}_{CS^2+1,C} \\ \vdots & \vdots & \vdots \\ \mathbf{K}_{M,1} & \cdots & \mathbf{K}_{M,C} \end{bmatrix} = \begin{bmatrix} e_{0,0} & & \\ 0 & \ddots & 0 \\ & & e_{0,0} \\ e_{1,0} & & \\ 0 & \ddots & 0 \\ & & e_{1,0} \\ & \vdots & \\ e_{S-1,S-1} & & \\ 0 & \ddots & 0 \\ & & e_{S-1,S-1} \\ & O & \end{bmatrix} \in \mathbb{R}^{M \times C \times k \times k} .$$

Then we proceed as in the 1D case.

For $M \geq CS^2$ and $S > k$:
By the same reasoning as in the 1D case, we have that the number of non-zero columns of $\mathcal{K}$ is less than or equal to Ck^2N^2. So, since $k < S$, we have rk$(\mathcal{K}) \leq Ck^2N^2 < CS^2N^2$. Therefore, for all $\mathbf{K} \in \mathbb{R}^{M \times C \times k \times k}$, we have $\mathcal{K}^T\mathcal{K} \neq Id_{CS^2N^2}$.

Appendix D. Restrictions due to Boundary Conditions

In this section, we prove the theorems related to 'valid' and 'same' boundary conditions.

D.1 Proof of Proposition 5

Proof For a single-channel convolution of kernel $h \in \mathbb{R}^k$ with 'valid' padding, the matrix applying the transformation on a signal $x \in \mathbb{R}^N$ has the following form:

$$A_N(h) := \begin{pmatrix} h_0 & \cdots & h_{2r} & & & 0 \\ & \ddots & \ddots & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ 0 & & & h_0 & \cdots & h_{2r} \end{pmatrix} \in \mathbb{R}^{(N-k+1)\times N} .$$

Hence, for $\mathbf{K} \in \mathbb{R}^{M\times C\times k}$, the layer transform matrix is:

$$\mathcal{K} = \begin{pmatrix} A_N(\mathbf{K}_{1,1}) & \dots & A_N(\mathbf{K}_{1,C}) \\ \vdots & \vdots & \vdots \\ A_N(\mathbf{K}_{M,1}) & \dots & A_N(\mathbf{K}_{M,C}) \end{pmatrix} \in \mathbb{R}^{M(N-k+1)\times CN} .$$

Let us focus on the columns corresponding to the first input channel. To simplify the notation, for $m \in [\![1, M]\!]$ we denote by $a^{(m)} := \mathbf{K}_{m,1} \in \mathbb{R}^k$. By contradiction, suppose that $\mathcal{K}^T\mathcal{K} = Id_{CN}$. In particular, for the first block matrix of size $M(N-k+1) \times N$ of $\mathcal{K}$ (i.e., corresponding to the first input channel), its first column, last column and column of index $2r$ are of norm 1. Since $N \geq 2k-1$, we have

$$\sum_{m=1}^{M} \left(a_0^{(m)}\right)^2 = 1, \qquad \sum_{m=1}^{M} \left(a_{2r}^{(m)}\right)^2 = 1 \qquad \text{and} \qquad \sum_{i=0}^{2r}\sum_{m=1}^{M} \left(a_i^{(m)}\right)^2 = 1 .$$

This is impossible. Therefore, for all $\mathbf{K} \in \mathbb{R}^{M\times C\times k}$, we have $\mathcal{K}^T\mathcal{K} \neq Id_{CN}$.
■

D.2 Proof of Proposition 6

Proof For a single-channel convolution of kernel $h \in \mathbb{R}^k$ with zero-padding 'same', the matrix applying the transformation on a signal $x \in \mathbb{R}^N$ has the following form:

$$A_N(h) := \begin{pmatrix} h_r & \cdots & h_{2r} & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ h_0 & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & h_{2r} \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & h_0 & \cdots & h_r \end{pmatrix} \in \mathbb{R}^{N\times N} .$$

Hence, for $\mathbf{K} \in \mathbb{R}^{M\times C\times k}$, the matrix that applies the convolutional layer is :

$$\mathcal{K} = \begin{pmatrix} A_N(\mathbf{K}_{1,1}) & \dots & A_N(\mathbf{K}_{1,C}) \\ \vdots & \vdots & \vdots \\ A_N(\mathbf{K}_{M,1}) & \dots & A_N(\mathbf{K}_{M,C}) \end{pmatrix} \in \mathbb{R}^{MN\times CN} .$$

Suppose $M \leq C$ **(RO case):** If $\mathcal{K}$ is orthogonal, then $\mathcal{K}\mathcal{K}^T = Id_{MN}$. Let us fix $m \in [\![1, M]\!]$. Since $\mathcal{K}\mathcal{K}^T = Id_{MN}$, the first row, the last row and the row of index r of the m-th block matrix of size $N \times CN$ of $\mathcal{K}$ are of norm equal to 1, i.e.

$$\|\mathcal{K}_{(m-1)N,:}\|_2^2 = 1, \qquad \|\mathcal{K}_{mN-1,:}\|_2^2 = 1 \qquad \text{and} \qquad \|\mathcal{K}_{(m-1)N+r,:}\|_2^2 = 1 .$$

To simplify the notation, for $c \in [\![1, C]\!]$, we denote by $a^{(c)} := \mathbf{K}_{m,c} \in \mathbb{R}^k$. Since $N \geq k$, the previous equations are equivalent to

$$\sum_{i=r}^{2r}\sum_{c=1}^{C}\left(a_i^{(c)}\right)^2 = 1, \qquad \sum_{i=0}^{r}\sum_{c=1}^{C}\left(a_i^{(c)}\right)^2 = 1 \qquad \text{and} \qquad \sum_{i=0}^{2r}\sum_{c=1}^{C}\left(a_i^{(c)}\right)^2 = 1 .$$

Substracting the first equality from the third one, and the second equality from the third one, we obtain

$$\sum_{i=0}^{r-1}\sum_{c=1}^{C}\left(a_i^{(c)}\right)^2 = 0, \qquad \sum_{i=r+1}^{2r}\sum_{c=1}^{C}\left(a_i^{(c)}\right)^2 = 0 \qquad \text{and} \qquad \sum_{i=0}^{2r}\sum_{c=1}^{C}\left(a_i^{(c)}\right)^2 = 1 .$$

This implies that for all $c \in [\![1, C]\!]$, for all $i \in [\![0, 2r]\!] \setminus \{r\}$, $a_i^{(c)} = 0$.
As a conclusion, for any $m \in [\![1, M]\!]$, any $c \in [\![1, C]\!]$, and any $i \in [\![0, 2r]\!] \setminus \{r\}$,

$$\mathbf{K}_{m,c,i} = 0.$$

This proves the result in the RO case.
The proof of the CO case is similar, and we have the same conclusion. ■

Appendix E. Proof of Theorem 7

As in Section C, we give the full proof in the 1D case and a sketch of proof in the 2D case.

In the RO case, the proof is based on calculations in which we carefully detail the structure of the matrix $\mathcal{K}\mathcal{K}^T - Id_{MN}$ and identify its constituent with those of $L_{orth}(\mathbf{K})$. The main lemma describing the structure of $\mathcal{K}\mathcal{K}^T - Id_{MN}$ is Lemma 16. It is deduced from Lemma 14 and Lemma 15 which focus on submatrices of $\mathcal{K}\mathcal{K}^T - Id_{MN}$.

The result in the CO case is obtained from the result in the RO case and a known relation between $\|\mathcal{K}\mathcal{K}^T - Id_{MN}\|_F^2$ and $\|\mathcal{K}^T\mathcal{K} - Id_{CSN}\|_F^2$, see for instance Lemma 1 in Wang et al. (2020).

E.1 Proof of Theorem 7, in the 1D Case

Before proving Theorem 7, we first present three intermediate lemmas.

Lemma 14 *Let $x \in \mathbb{R}^{SN}$. We have*

$$S_N C(x) S_N^T = C(S_N x)\ .$$

Proof Let $x \in \mathbb{R}^{SN}$, $X = C(x)$ and $Y = S_N X S_N^T \in \mathbb{R}^{N\times N}$. The matrix Y is formed by sampling X, i.e., for all $m, n \in [\![0, N-1]\!]$,

$$Y_{m,n} = X_{Sm,Sn}.$$

Hence, using (23), $Y_{m,n} = x_{(Sm-Sn)\%SN} = x_{S((m-n)\%N)}$. Setting $y = S_N x$, we have $y_l = x_{Sl}$ for all $l \in [\![0, N-1]\!]$. Therefore, $Y_{m,n} = y_{(m-n)\%N}$, and using (23), we obtain $Y = C(y)$. Hence, from the definitions of Y, X and y we conclude that

$$S_N C(x) S_N^T = C(S_N x)\ .$$

This completes the proof of the lemma. ■

For N such that $SN \geq 2k-1$, and $P = \left\lfloor \frac{k-1}{S} \right\rfloor S$, we introduce the operator $Q_{S,N}$ which associates to a vector $x = (x_0, \ldots, x_{2\frac{P}{S}})^T \in \mathbb{R}^{2\frac{P}{S}+1}$, the vector

$$Q_{S,N}(x) = (x_{\frac{P}{S}}, \ldots, x_{2\frac{P}{S}}, 0, \ldots, 0, x_0, x_1, \ldots, x_{\frac{P}{S}-1})^T \in \mathbb{R}^N. \tag{32}$$

Lemma 15 *Let S, $k = 2r+1$ and N be positive integers such that $SN \geq 2k-1$. Let $h, g \in \mathbb{R}^k$ and $P = \left\lfloor \frac{k-1}{S} \right\rfloor S$, we have*

$$S_N C(P_{SN}(h)) C(P_{SN}(g))^T S_N^T = C(Q_{S,N}(\mathrm{conv}(h, g, \textit{padding zero} = P, \textit{stride} = S)))\ . \tag{33}$$

Proof Let N be such that $SN \geq 2k-1$, and $P = \left\lfloor \frac{k-1}{S} \right\rfloor S$. Let us first detail and analyse the left-hand side of (33). Recall that by definition $P_{SN}(h)$ is SN-periodic: $[P_{SN}(h)]_i = [P_{SN}(h)]_{i\%SN}$ for all $i \in \mathbb{Z}$. Using (24), (25), and (26), we have

$$\begin{aligned} C(P_{SN}(h))C(P_{SN}(g))^T &= C(P_{SN}(h))C(\widetilde{P_{SN}(g)}) \\ &= C\left(\left(\sum_{i=0}^{SN-1} [P_{SN}(h)]_i \left[\widetilde{P_{SN}(g)}\right]_{j-i}\right)_{j=0..SN-1}\right) \\ &= C\left(\left(\sum_{i=0}^{SN-1} [P_{SN}(h)]_i [P_{SN}(g)]_{i-j}\right)_{j=0..SN-1}\right) . \end{aligned}$$

Setting $b^{(SN)}[h,g] = \left(\sum_{i=0}^{SN-1} [P_{SN}(h)]_i [P_{SN}(g)]_{i-j}\right)_{j=0..SN-1}$, we have

$$C(P_{SN}(h))C(P_{SN}(g))^T = C(b^{(SN)}[h,g])\ . \tag{34}$$

To simplify the forthcoming notation, we temporarily denote by

$$b := b^{(SN)}[h,g]. \tag{35}$$

Notice that by definition, b is SN-periodic. Therefore, we can restrict its study to an interval of size SN. We consider $j \in [\![-2r, SN-2r-1]\!]$. From the definition of P_{SN} in (28), we have, for $i \in [\![-r, -r+SN-1]\!]$,

$$[P_{SN}(h)]_i = \begin{cases} h_{r-i} & \text{if} \quad i \in [\![-r, r]\!] \\ 0 & \text{if} \quad i \in [\![r+1, -r+SN-1]\!] \ . \end{cases} \tag{36}$$

Hence, since $P_{SN}(h)$ and $P_{SN}(g)$ are periodic, we have

$$\begin{aligned} b_j &= \sum_{i=0}^{SN-1} [P_{SN}(h)]_i [P_{SN}(g)]_{i-j} \\ &= \sum_{i=-r}^{SN-1-r} [P_{SN}(h)]_i [P_{SN}(g)]_{i-j} \\ &= \sum_{i=-r}^{r} [P_{SN}(h)]_i [P_{SN}(g)]_{i-j}. \end{aligned} \tag{37}$$

The set of indices $i \in [\![-r, r]\!]$ such that $[P_{SN}(h)]_i [P_{SN}(g)]_{i-j} \neq 0$ is included in $[\![-r, r]\!] \cap \{i | (i-j)\%SN \in [\![-r, r]\!]\%SN\}$.
Since $j \in [\![-2r, SN-2r-1]\!]$: We have $-r \leq i \leq r$ and $-2r \leq j \leq SN-2r-1$, then $-SN+r+1 \leq i-j \leq 3r$, but by hypothesis, $SN \geq 2k-1 = 4r+1$, hence $3r < SN-r$ and so $-SN+r < i-j < SN-r$. Therefore, for $i \in [\![-r, r]\!]$ and $j \in [\![-2r, SN-2r-1]\!]$

$$(i-j)\%SN \in ([\![-r, r]\!]\%SN) \iff i-j \in [\![-r, r]\!] \iff i \in [\![-r+j, r+j]\!].$$

As a conclusion, for $j \in [\![-2r, SN-2r-1]\!]$,

$$\Big\{ i \in [\![-r, r]\!] \quad | \quad [P_{SN}(h)]_i [P_{SN}(g)]_{i-j} \neq 0 \Big\} \subset [\![-r, r]\!] \cap [\![-r+j, r+j]\!]. \tag{38}$$

Let us now analyse the right-side of (33). We start by considering zero-padding $= k-1$ and stride $= 1$, and we will arrive to the formula with zero-padding $= P$ and stride $= S$ later. Using (13), we denote by

$$a = \text{conv}(h, g, \text{padding zero} = k-1, \text{stride} = 1) \in \mathbb{R}^{2k-1}. \tag{39}$$

We have from (14), for $j \in [\![0, 2k-2]\!]$,

$$a_j = \sum_{i=0}^{k-1} h_i \bar{g}_{i+j} \ .$$

Using (15) and keeping the indices $i \in [\![0, k-1]\!]$ for which $\bar{g}_{i+j} \neq 0$, i.e. such that $i+j \in [\![k-1, 2k-2]\!]$, we obtain

$$\begin{cases} a_j = \sum_{i=k-1-j}^{k-1} h_i g_{i+j-(k-1)} & \text{if} \quad j \in [\![0, k-2]\!] \ , \\ a_j = \sum_{i=0}^{2k-2-j} h_i g_{i+j-(k-1)} & \text{if} \quad j \in [\![k-1, 2k-2]\!] \ . \end{cases} \tag{40}$$

In the following, we will connect b with a by distinguishing several cases depending on the value of j.
We distinguish $j \in [\![0, 2r]\!]$, $j \in [\![-2r, -1]\!]$ and $j \in [\![2r+1, -2r+SN-1]\!]$. Recall that $k = 2r+1$.

If $j \in [\![0, 2r]\!]$**:** then $[\![-r, r]\!] \cap [\![-r+j, r+j]\!] = [\![-r+j, r]\!]$. Using (38) and (36), the equality (37) becomes

$$b_j = \sum_{i=-r+j}^{r} [P_{SN}(h)]_i [P_{SN}(g)]_{i-j} = \sum_{i=-r+j}^{r} h_{r-i} g_{r-i+j} \ .$$

By changing the variable $l = r - i$, and using $k = 2r+1$, we find

$$b_j = \sum_{l=0}^{2r-j} h_l g_{l+j} = \sum_{l=0}^{k-1-j} h_l g_{l+j} = \sum_{l=0}^{2k-2-(k-1+j)} h_l g_{l+(k-1+j)-(k-1)} \ .$$

When $j \in [\![0, 2r]\!] = [\![0, k-1]\!]$, we have $k-1+j \in [\![k-1, 2k-2]\!]$, therefore using (40), we obtain

$$b_j = a_{k-1+j} \ . \tag{41}$$

If $j \in [\![-2r, -1]\!]$**:** then $[\![-r, r]\!] \cap [\![-r+j, r+j]\!] = [\![-r, r+j]\!]$. Using (38) and (36), the equality (37) becomes

$$b_j = \sum_{i=-r}^{r+j} [P_{SN}(h)]_i [P_{SN}(g)]_{i-j} = \sum_{i=-r}^{r+j} h_{r-i} g_{r-i+j} \ .$$

By changing the variable $l = r - i$, and using $k = 2r+1$, we find

$$b_j = \sum_{l=-j}^{2r} h_l g_{l+j} = \sum_{l=-j}^{k-1} h_l g_{l+j} = \sum_{l=k-1-(k-1+j)}^{k-1} h_l g_{l+(k-1+j)-(k-1)} \ .$$

When $j \in [\![-2r, -1]\!] = [\![-(k-1), -1]\!]$, we have $k-1+j \in [\![0, k-2]\!]$, and using (40), we obtain

$$b_j = a_{k-1+j} \ . \tag{42}$$

If $j \in [\![2r+1, SN-2r-1]\!]$**:** then $[\![-r, r]\!] \cap [\![-r+j, r+j]\!] = \emptyset$. The equality (37) becomes

$$b_j = 0. \tag{43}$$

Therefore, we summarize (41), (42) and (43): For all $j \in [\![-(k-1), -(k-1)+SN-1]\!]$,

$$b_j = \begin{cases} a_{k-1+j} & \text{if } j \in [\![-(k-1), k-1]\!], \\ 0 & \text{if } j \in [\![k, SN-k]\!]. \end{cases} \tag{44}$$

Let us now introduce 'padding zero $= P$' and 'stride $= S$'. We will prove the equality between matrices in (33) using the equality between vectors in (44).

Recall that $P = \left\lfloor \frac{k-1}{S} \right\rfloor S \leq k-1$, and let $i \in [\![0, 2P]\!]$. Therefore $i - P \in [\![-P, P]\!] \subset [\![-(k-1), k-1]\!]$, hence using (39), (16) and (44), we have

$$[\text{conv}(h, g, \text{padding zero} = P, \text{stride} = 1)]_i = a_{k-1+i-P} = b_{i-P} \ .$$

Therefore, using (17) and $\lfloor 2P/S \rfloor + 1 = 2P/S + 1$

$$\begin{aligned}&\text{conv}(h, g, \text{padding zero} = P, \text{stride} = S)\\ &= \left(b_{-\lfloor\frac{k-1}{S}\rfloor S}, \ldots, b_{-2S}, b_{-S}, b_0, b_S, b_{2S}, \ldots, b_{\lfloor\frac{k-1}{S}\rfloor S}\right)^T \in \mathbb{R}^{2P/S+1} .\end{aligned}$$

Using the definition of $Q_{S,N}$ in (32), we obtain

$$\begin{aligned}&Q_{S,N}(\text{conv}(h, g, \text{padding zero} = P, \text{stride} = S))\\ &= \left(b_0, b_S, b_{2S}, \ldots, b_{\lfloor\frac{k-1}{S}\rfloor S}, 0, \ldots, 0, b_{-\lfloor\frac{k-1}{S}\rfloor S}, \ldots, b_{-2S}, b_{-S}\right)^T \in \mathbb{R}^N .\end{aligned}$$

But, using (43), and since $\lfloor\frac{k-1}{S}\rfloor S$ is the largest multiple of S less than or equal to $k-1$ and b is SN-periodic, we have

$$\begin{aligned}S_N b &= \left(b_0, b_S, b_{2S}, \ldots, b_{\lfloor\frac{k-1}{S}\rfloor S}, 0, \ldots, 0, b_{SN-\lfloor\frac{k-1}{S}\rfloor S}, \ldots, b_{SN-2S}, b_{SN-S}\right)^T\\ &= \left(b_0, b_S, b_{2S}, \ldots, b_{\lfloor\frac{k-1}{S}\rfloor S}, 0, \ldots, 0, b_{-\lfloor\frac{k-1}{S}\rfloor S}, \ldots, b_{-2S}, b_{-S}\right)^T \in \mathbb{R}^N .\end{aligned}$$

Finally, we have

$$S_N b = Q_{S,N}(\text{conv}(h, g, \text{padding zero} = P, \text{stride} = S)) .$$

Using (35), (34) and Lemma 14, we conclude that

$$\begin{aligned}S_N C(P_{SN}(h)) C(P_{SN}(g))^T S_N^T &= S_N C(b^{(SN)}[h, g]) S_N^T\\ &= C(S_N b^{(SN)}[h, g])\\ &= C(Q_{S,N}(\text{conv}(h, g, \text{padding zero} = P, \text{stride} = S))) .\end{aligned}$$

■

Lemma 16 *Let M, C, S, $k = 2r+1$ be positive integers, and let $\mathbf{K} \in \mathbb{R}^{M\times C\times k}$. Let N be such that $SN \geq 2k-1$, and $P = \lfloor\frac{k-1}{S}\rfloor S$. We denote by $z_{P/S} = \begin{bmatrix} 0_{P/S} \\ 1 \\ 0_{P/S}\end{bmatrix} \in \mathbb{R}^{2P/S+1}$. We have*

$$\mathcal{K}\mathcal{K}^T - Id_{MN} = \begin{pmatrix} C(Q_{S,N}(x_{1,1})) & \ldots & C(Q_{S,N}(x_{1,M})) \\ \vdots & \ddots & \vdots \\ C(Q_{S,N}(x_{M,1})) & \ldots & C(Q_{S,N}(x_{M,M})) \end{pmatrix} ,$$

where for all $m, l \in [\![1, M]\!]$,

$$x_{m,l} = \sum_{c=1}^{C} \text{conv}(\mathbf{K}_{m,c}, \mathbf{K}_{l,c}, \textit{padding zero} = P, \textit{stride} = S) - \delta_{m=l} z_{P/S} \in \mathbb{R}^{2P/S+1} . \quad (45)$$

Proof We have from (30),

$$\mathcal{K} = \begin{pmatrix} S_N C(P_{SN}(\mathbf{K}_{1,1})) & \dots & S_N C(P_{SN}(\mathbf{K}_{1,C})) \\ \vdots & \vdots & \vdots \\ S_N C(P_{SN}(\mathbf{K}_{M,1})) & \dots & S_N C(P_{SN}(\mathbf{K}_{M,C})) \end{pmatrix} \in \mathbb{R}^{MN \times CSN} .$$

Hence, we have that the block $(m, l) \in [\![1, M]\!]^2$ of size (N, N) of $\mathcal{K}\mathcal{K}^T$ is equal to :

$$\begin{aligned} &\begin{pmatrix} S_N C(P_{SN}(\mathbf{K}_{m,1})) & \dots & S_N C(P_{SN}(\mathbf{K}_{m,C})) \end{pmatrix} \begin{pmatrix} C(P_{SN}(\mathbf{K}_{l,1}))^T S_N^T \\ \vdots \\ C(P_{SN}(\mathbf{K}_{l,C}))^T S_N^T \end{pmatrix} \\ &= \sum_{c=1}^{C} S_N C(P_{SN}(\mathbf{K}_{m,c})) C(P_{SN}(\mathbf{K}_{l,c}))^T S_N^T . \end{aligned}$$

We denote by $A_{m,l} \in \mathbb{R}^{N \times N}$ the block $(m, l) \in [\![1, M]\!]^2$ of size (N, N) of $\mathcal{K}\mathcal{K}^T - Id_{MN}$. We want to prove that $A_{m,l} = C(Q_{S,N}(x_{m,l}))$ where $x_{m,l}$ is defined in (45). Using (9), (22), and (32), we have $Id_N = C\left(\begin{bmatrix} 1 \\ 0_{N-1} \end{bmatrix}\right) = C(Q_{S,N}(z_{P/S}))$, and therefore,

$$A_{m,l} = \sum_{c=1}^{C} S_N C(P_{SN}(\mathbf{K}_{m,c})) C(P_{SN}(\mathbf{K}_{l,c}))^T S_N^T - \delta_{m=l} C(Q_{S,N}(z_{P/S})) .$$

Using Lemma 15, this becomes

$$A_{m,l} = \sum_{c=1}^{C} C(Q_{S,N}(\text{conv}(\mathbf{K}_{m,c}, \mathbf{K}_{l,c}, \text{padding zero} = P, \text{stride} = S))) - \delta_{m=l} C(Q_{S,N}(z_{P/S})) .$$

By linearity of C and $Q_{S,N}$, we obtain

$$\begin{aligned} A_{m,l} &= C\left(Q_{S,N}\left(\sum_{c=1}^{C} \text{conv}(\mathbf{K}_{m,c}, \mathbf{K}_{l,c}, \text{padding zero} = P, \text{stride} = S) - \delta_{m=l} z_{P/S}\right)\right) \\ &= C\left(Q_{S,N}(x_{m,l})\right) . \end{aligned}$$

■

Proof [Proof of Theorem 7] Let M, C, S, $k = 2r + 1$ be positive integers, and let $\mathbf{K} \in \mathbb{R}^{M \times C \times k}$. Let N be such that $SN \geq 2k - 1$, and $P = \left\lfloor \frac{k-1}{S} \right\rfloor S$. For all $m, l \in [\![1, M]\!]$, we denote by $A_{m,l} \in \mathbb{R}^{N \times N}$ the block (m, l) of size (N, N) of $\mathcal{K}\mathcal{K}^T - Id_{MN}$. Using Lemma 16, we have

$$A_{m,l} = C\left(Q_{S,N}\left(\sum_{c=1}^{C} \text{conv}(\mathbf{K}_{m,c}, \mathbf{K}_{l,c}, \text{padding zero} = P, \text{stride} = S) - \delta_{m=l} z_{P/S}\right)\right) .$$

Hence, from (22) and (32), using the fact that for all $x \in \mathbb{R}^N$, $\|C(x)\|_F^2 = N\|x\|_2^2$, and for all $x \in \mathbb{R}^{2P/S+1}$, $\|Q_{S,N}(x)\|_2^2 = \|x\|_2^2$, we have

$$\begin{aligned}
&\|\mathcal{K}\mathcal{K}^T - Id_{MN}\|_F^2 \\
&= \sum_{m=1}^{M}\sum_{l=1}^{M} \|A_{m,l}\|_F^2 \\
&= \sum_{m=1}^{M}\sum_{l=1}^{M} \left\| C\left(Q_{S,N}\left(\sum_{c=1}^{C} \text{conv}(\mathbf{K}_{m,c}, \mathbf{K}_{l,c}, \text{padding zero} = P, \text{stride} = S) - \delta_{m=l} z_{P/S} \right)\right)\right\|_F^2 \\
&= \sum_{m=1}^{M}\sum_{l=1}^{M} N \left\| Q_{S,N}\left(\sum_{c=1}^{C} \text{conv}(\mathbf{K}_{m,c}, \mathbf{K}_{l,c}, \text{padding zero} = P, \text{stride} = S) - \delta_{m=l} z_{P/S} \right)\right\|_2^2 \\
&= N\sum_{m=1}^{M}\sum_{l=1}^{M} \left\| \sum_{c=1}^{C} \text{conv}(\mathbf{K}_{m,c}, \mathbf{K}_{l,c}, \text{padding zero} = P, \text{stride} = S) - \delta_{m=l} z_{P/S} \right\|_2^2 .
\end{aligned}$$

Therefore, using (19) and (18), we obtain for any M, C, S, $k = 2r+1$ and $\mathbf{K} \in \mathbb{R}^{M\times C\times k}$,

$$\|\mathcal{K}\mathcal{K}^T - Id_{MN}\|_F^2 = N\|\,\mathbf{conv}(\mathbf{K}, \mathbf{K}, \text{padding zero} = P, \text{stride} = S) - \mathbf{I}_{r0}\|_F^2 . \tag{46}$$

This concludes the proof in the RO case.
In order to prove the theorem in the CO case we use Lemma 1 in Wang et al. (2020). This lemma states that

$$\|\mathcal{K}^T\mathcal{K} - Id_{CSN}\|_F^2 = \|\mathcal{K}\mathcal{K}^T - Id_{MN}\|_F^2 + CSN - MN.$$

Therefore, using that (46) holds for all M, C and S, we have

$$\|\mathcal{K}^T\mathcal{K} - Id_{CSN}\|_F^2 = N\left(\|\,\mathbf{conv}(\mathbf{K}, \mathbf{K}, \text{padding zero} = P, \text{stride} = S) - \mathbf{I}_{r0}\|_F^2 - (M - CS)\right) \tag{47}$$

Hence, using the definitions of err_N^F and L_{orth} in Sections A.2.2 and A.2.3, (46) and (47) lead to

$$\left(\text{err}_N^F(\mathbf{K})\right)^2 = N L_{orth}(\mathbf{K}).$$

This concludes the proof of Theorem 7 in the 1D case. ■

E.2 Sketch of the Proof of Theorem 7, in the 2D Case

We start by stating intermediate lemmas. First we introduce a slight abuse of notation, for a vector $x \in \mathbb{R}^{N^2}$, we denote by $\mathcal{C}(x) = \mathcal{C}(X)$, where $X \in \mathbb{R}^{N\times N}$ such that $\text{Vect}(X) = x$. The main steps of the proof in the 2D case follow those in the 1D case and are given below.

Lemma 17 *Let $X \in \mathbb{R}^{SN\times SN}$. We have*

$$\mathcal{S}_N \mathcal{C}(X) \mathcal{S}_N^T = \mathcal{C}(\mathcal{S}_N\,\text{Vect}(X)).$$

Let $\mathcal{Q}_{S,N}$ be the operator which associates to a matrix $x \in \mathbb{R}^{(2P/S+1)\times(2P/S+1)}$ the matrix

$$\begin{pmatrix}
x_{P/S,P/S} & \cdots & x_{P/S,2P/S} & 0 & \cdots & 0 & x_{P/S,0} & \cdots & x_{P/S,P/S-1} \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
x_{2P/S,P/S} & \cdots & x_{2P/S,2P/S} & 0 & \cdots & 0 & x_{2P/S,0} & \cdots & x_{2P/S,P/S-1} \\
0 & \dots & 0 & 0 & \dots & 0 & 0 & \dots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
0 & \dots & 0 & 0 & \dots & 0 & 0 & \dots & 0 \\
x_{0,P/S} & \cdots & x_{0,2P/S} & 0 & \cdots & 0 & x_{0,0} & \cdots & x_{0,P/S-1} \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
x_{P/S-1,P/S} & \cdots & x_{P/S-1,2P/S} & 0 & \cdots & 0 & x_{P/S-1,0} & \cdots & x_{P/S-1,P/S-1}
\end{pmatrix} \in \mathbb{R}^{N\times N} .$$

Lemma 18 *Let N be such that $SN \geq 2k-1$, $h, g \in \mathbb{R}^{k\times k}$ and $P = \left\lfloor \frac{k-1}{S} \right\rfloor S$, we have*

$$\mathcal{S}_N \mathcal{C}(\mathcal{P}_{SN}(h))\mathcal{C}(\mathcal{P}_{SN}(g))^T \mathcal{S}_N^T = \mathcal{C}(\mathcal{Q}_{S,N}(\text{conv}(h, g, \textit{padding zero} = P, \textit{stride} = S))) .$$

Lemma 19 *Let M, C, S, $k = 2r+1$ be positive integers, and let $\mathbf{K} \in \mathbb{R}^{M\times C\times k\times k}$. Let N be such that $SN \geq 2k-1$, and $P = \left\lfloor \frac{k-1}{S} \right\rfloor S$. We set $z_{P/S,P/S} \in \mathbb{R}^{(2P/S+1)\times(2P/S+1)}$ such that for all $i, j \in [\![0, 2P/S]\!]$, $\left[z_{P/S,P/S}\right]_{i,j} = \delta_{i=P/S}\delta_{j=P/S}$. We have*

$$\mathcal{K}\mathcal{K}^T - Id_{MN^2} = \begin{pmatrix} \mathcal{C}(\mathcal{Q}_{S,N}(x_{1,1})) & \dots & \mathcal{C}(\mathcal{Q}_{S,N}(x_{1,M})) \\ \vdots & \ddots & \vdots \\ \mathcal{C}(\mathcal{Q}_{S,N}(x_{M,1})) & \dots & \mathcal{C}(\mathcal{Q}_{S,N}(x_{M,M})) \end{pmatrix} ,$$

where for all $m, l \in [\![1, M]\!]$,

$$x_{m,l} = \sum_{c=1}^{C} \text{conv}(\mathbf{K}_{m,c}, \mathbf{K}_{l,c}, \textit{padding zero} = P, \textit{stride} = S) - \delta_{m=l} z_{P/S,P/S}.$$

Then we proceed as in the 1D case.

Appendix F. Proof of Theorem 8

As in Section C and Section E, we give the full proof in the 1D case and a sketch of proof in the 2D case.

The lower-bound is a consequence of Theorem 7.

The proof of the upper-bound is different in the RO case and the CO case. In the RO case, we first express the orthogonality residual using Lemma 16. Then we conclude with calculations based on matrix norm inequalities, properties of circulant matrices and the definition of $L_{orth}(\mathbf{K})$.

The CO case is more difficult since, to use in place of Lemma 16, we first need to establish Lemma 20. We then proceed with calculations to express that $\|\mathcal{K}^T\mathcal{K} - Id_{CSN}\|_2$ is upper-bounded by a quantity independent of N, as long as $N \geq 2k-1$. Then, after calculations, a key argument is to apply Theorem 7 to the matrix $\mathcal{K}$ obtained for the signal size $N' = 2k-1$.

F.1 Proof of Theorem 8, in the 1D Case

The lower-bound of Theorem 8 is an immediate consequence of (8) and Theorem 7. We have indeed both in the RO and CO case:

$$(\mathrm{err}_N^s(\mathbf{K}))^2 \geq \frac{1}{\min(M, CS^2)N^2}(\mathrm{err}_N^F(\mathbf{K}))^2 = \frac{1}{\min(M, CS^2)} L_{orth}(\mathbf{K}).$$

We focus from now on on the upper-bound. Let M, C, S, $k = 2r+1$ be positive integers, and let $\mathbf{K} \in \mathbb{R}^{M\times C\times k}$. Let N be such that $SN \geq 2k-1$, and $P = \left\lfloor \frac{k-1}{S} \right\rfloor S$. We denote by $z_{P/S} = \begin{bmatrix} 0_{P/S} \\ 1 \\ 0_{P/S} \end{bmatrix} \in \mathbb{R}^{2P/S+1}$.

RO case ($M \leq CS$): From Lemma 16, we have

$$\mathcal{K}\mathcal{K}^T - Id_{MN} = \begin{pmatrix} C(Q_{S,N}(x_{1,1})) & \dots & C(Q_{S,N}(x_{1,M})) \\ \vdots & \ddots & \vdots \\ C(Q_{S,N}(x_{M,1})) & \dots & C(Q_{S,N}(x_{M,M})) \end{pmatrix}, \tag{48}$$

where for all $m, l \in [\![1, M]\!]$,

$$x_{m,l} = \sum_{c=1}^{C} \mathrm{conv}(\mathbf{K}_{m,c}, \mathbf{K}_{l,c}, \text{padding zero} = P, \text{stride} = S) - \delta_{m=l} z_{P/S} \in \mathbb{R}^{2P/S+1}. \tag{49}$$

We set

$$B = \mathcal{K}\mathcal{K}^T - Id_{MN}.$$

Since B is symmetric and due to the well-known properties of matrix norms, we have $\|B\|_1 = \|B\|_\infty$ and $\|B\|_2^2 \leq \|B\|_1\|B\|_\infty$. Hence, using the definition of $\|B\|_1$, we have

$$\|B\|_2^2 \leq \|B\|_1\|B\|_\infty = \|B\|_1^2 = \left(\max_{1\leq l\leq MN} \sum_{m=1}^{MN} |B_{m,l}|\right)^2.$$

Using (48), and (22), we obtain

$$\|B\|_2^2 \leq \max_{1\leq l\leq M} \left(\sum_{m=1}^{M} \|Q_{S,N}(x_{m,l})\|_1\right)^2.$$

Given the definition of $Q_{S,N}$ in (32), we have for all $x \in \mathbb{R}^{2P/S+1}$, $\|Q_{S,N}(x)\|_1 = \|x\|_1$, therefore,

$$\|B\|_2^2 \leq \max_{1\leq l\leq M} \left(\sum_{m=1}^{M} \|x_{m,l}\|_1\right)^2.$$

We set $l_0 \in \arg\max_{1\leq l\leq M} \left(\sum_{m=1}^{M} \|x_{m,l}\|_1\right)^2$. Using that for all $x \in \mathbb{R}^n$, $\|x\|_1 \leq \sqrt{n}\|x\|_2$, we have

$$\|B\|_2^2 \leq \left(\sum_{m=1}^{M} \|x_{m,l_0}\|_1\right)^2 \leq (2P/S+1)\left(\sum_{m=1}^{M} \|x_{m,l_0}\|_2\right)^2.$$

Using Cauchy-Schwarz inequality, we obtain

$$\|B\|_2^2 \leq (2P/S+1)M \sum_{m=1}^{M} \|x_{m,l_0}\|_2^2 \leq (2P/S+1)M \sum_{m=1}^{M}\sum_{l=1}^{M} \|x_{m,l}\|_2^2 \ .$$

Using (49), then (19) and (18), we obtain

$$\begin{aligned}
&\|B\|_2^2 \\
&\leq (2P/S+1)M \sum_{m=1}^{M}\sum_{l=1}^{M} \left\| \sum_{c=1}^{C} \text{conv}(\mathbf{K}_{m,c}, \mathbf{K}_{l,c}, \text{padding zero} = P, \text{stride} = S) - \delta_{m=l} z_{P/S} \right\|_2^2 \\
&= (2P/S+1)M \sum_{m=1}^{M}\sum_{l=1}^{M} \left\| [\mathbf{conv}(\mathbf{K}, \mathbf{K}, \text{padding zero} = P, \text{stride} = S) - \mathbf{I}_{r0}]_{m,l,:} \right\|_2^2 \\
&= (2P/S+1)M \left\| \mathbf{conv}(\mathbf{K}, \mathbf{K}, \text{padding zero} = P, \text{stride} = S) - \mathbf{I}_{r0} \right\|_F^2 \\
&= (2P/S+1)M L_{orth}(\mathbf{K}) \ .
\end{aligned}$$

This proves the inequality in the RO case.

CO case ($M \geq CS$): First, for $n \geq 2k-1$, let R_n be the operator that associates to $x \in \mathbb{R}^{2k-1}$, the vector

$$R_n(x) = (x_{k-1}, \ldots, x_{2k-2}, 0, \ldots, 0, x_0, \ldots, x_{k-2})^T \in \mathbb{R}^n \ . \tag{50}$$

Note that, when $S' = 1$, $N' = SN$, we have in (32), $P' = k-1$ and

$$Q_{1,SN} = R_{SN}. \tag{51}$$

Recall from (9) that $(f_i)_{i=0..SN-1}$ is the canonical basis of $\mathbb{R}^{SN}$. Let $\Lambda_j = C(f_j) \in \mathbb{R}^{SN\times SN}$ be the permutation matrix which shifts down (cyclically) any vector by $j \in [\![0, SN-1]\!]$: for all $x \in \mathbb{R}^{SN}$, for $i \in [\![0, SN-1]\!]$, $(\Lambda_j x)_i = x_{(i-j)\%SN}$. Note that, using (22), we have for all $x \in \mathbb{R}^{SN}$,

$$[C(x)]_{:,j} = \Lambda_j x. \tag{52}$$

Recall that $k = 2r+1$, and for all $h \in \mathbb{R}^k$,

$$P_{SN}(x) = (h_r, \ldots, h_0, 0, \ldots, 0, h_{2r}, \ldots, h_{r+1})^T \in \mathbb{R}^{SN}.$$

For $j \in [\![0, SN-1]\!]$, for $x \in \mathbb{R}^k$, we denote by

$$P_{SN}^{(j)}(x) = \Lambda_j P_{SN}(x) \tag{53}$$

and for $x \in \mathbb{R}^{2k-1}$, we denote by

$$R_{SN}^{(j)}(x) = \Lambda_j R_{SN}(x). \tag{54}$$

By assumption $SN \geq 2k-1$, hence $R_{SN}(x)$ is well-defined and we have for all $j \in [\![0, SN-1]\!]$, for all $x \in \mathbb{R}^{2k-1}$,

$$\begin{cases} \|R_{SN}^{(j)}(x)\|_1 = \|x\|_1, \\ \|R_{SN}^{(j)}(x)\|_2 = \|x\|_2. \end{cases} \tag{55}$$

We first start by introducing the following Lemma.

Lemma 20 *Let $h, g \in \mathbb{R}^k$. There exist S vectors $x_0, \ldots, x_{S-1} \in \mathbb{R}^{2k-1}$ such that for all N satisfying $SN \geq 2k-1$, we have for all $j \in [\![0, SN-1]\!]$,*

$$\left[C(P_{SN}(h))^T S_N^T S_N C(P_{SN}(g))\right]_{:,j} = R_{SN}^{(j)}(x_{j\%S}) \ .$$

Proof Recall that from (20) and (21), we have $S_N = \sum_{i=0}^{N-1} E_{i,Si}$ and $A_N := S_N^T S_N = \sum_{i=0}^{N-1} F_{Si,Si}$. When applied to a vector $x \in \mathbb{R}^{SN}$, A_N keeps unchanged the components of x whose indices are multiples of S, while the other components of $A_N x$ are equal to zero. We know from (52) and (53) that, for $j \in [\![0, SN-1]\!]$, the j-th column of $C(P_{SN}(g))$ is equal to $P_{SN}^{(j)}(g)$. Therefore, when applying A_N, this becomes $A_N P_{SN}^{(j)}(g) = P_{SN}^{(j)}\left(g^j\right)$, where $g^j \in \mathbb{R}^k$ is formed from g by putting zeroes in the place of the elements that have been replaced by 0 when applying A_N. But since A_N preserves the component whose index is a multiple of S, we have that the j-th column of $A_N C(P_{SN}(g))$ has the same elements as its $j\%S$-th column, shifted down by $(j - j\%S)$ indices. More precisely, $A_N P_{SN}^{(j)}(g) = \Lambda_{j-j\%S} A_N P_{SN}^{(j\%S)}(g)$, hence $P_{SN}^{(j)}\left(g^j\right) = \Lambda_{j-j\%S} P_{SN}^{(j\%S)}\left(g^{j\%S}\right) = P_{SN}^{(j)}\left(g^{j\%S}\right)$. This implies that $g^j = g^{j\%S}$. Note that, using (28), we can also derive the exact formula of g^j, in fact for all $i \in [\![0, 2r]\!]$,

$$\left[g^j\right]_i = \begin{cases} g_i & \text{if } (i-r-j)\%S = 0, \\ 0 & \text{otherwise.} \end{cases}$$

We again can see that $g^j = g^{j\%S}$. Therefore, using (52) and (53), we have

$$A_N[C(P_{SN}(g))]_{:,j} = A_N P_{SN}^{(j)}(g) = P_{SN}^{(j)}\left(g^j\right) = P_{SN}^{(j)}\left(g^{j\%S}\right) = \left[C\left(P_{SN}\left(g^{j\%S}\right)\right)\right]_{:,j} .$$

Therefore, we have, for all $j \in [\![0, SN-1]\!]$,

$$[C(P_{SN}(h))^T A_N C(P_{SN}(g))]_{:,j} = \left[C(P_{SN}(h))^T C(P_{SN}(g^{j\%S}))\right]_{:,j} .$$

Using the fact that the transpose of a circulant matrix is a circulant matrix and that two circulant matrices commute with each other (see Equation 24 and Equation 27), we conclude that the transpose of any circulant matrix commutes with any circulant matrix, therefore

$$[C(P_{SN}(h))^T A_N C(P_{SN}(g))]_{:,j} = \left[C(P_{SN}(g^{j\%S}))C(P_{SN}(h))^T\right]_{:,j} .$$

Using Lemma 15 with $S' = 1$ and $N' = SN$, and noting that,when $S' = 1$, the sampling matrix $S_{N'}$ is equal to the identity, we have

$$\begin{aligned} & C(P_{SN}(g^{j\%S}))C(P_{SN}(h))^T \\ & = Id_{N'} C(P_{N'}(g^{j\%S}))C(P_{N'}(h))^T Id_{N'}^T \\ & = C(Q_{S',N'}(\text{conv}(g^{j\%S}, h, \text{padding zero} = \left\lfloor \frac{k-1}{S'} \right\rfloor S', \text{stride} = S'))) \\ & = C(Q_{1,SN}(\text{conv}(g^{j\%S}, h, \text{padding zero} = k-1, \text{stride} = 1))) \end{aligned}$$

To simplify, we denote by $x_{j\%S} = \text{conv}(g^{j\%S}, h, \text{padding zero} = k-1, \text{stride} = 1) \in \mathbb{R}^{2k-1}$. Using (51), we obtain

$$C(P_{SN}(g^{j\%S}))C(P_{SN}(h))^T = C(Q_{1,SN}(x_{j\%S})) = C(R_{SN}(x_{j\%S})) \ .$$

Using (52) and (54), we obtain

$$[C(P_{SN}(h))^T A_N C(P_{SN}(g))]_{:,j} = [C(R_{SN}(x_{j\%S}))]_{:,j} = \Lambda_j R_{SN}(x_{j\%S}) = R_{SN}^{(j)}(x_{j\%S}).$$

Therefore, we have for all $j \in [\![0, SN-1]\!]$,

$$\left[C(P_{SN}(h))^T S_N^T S_N C(P_{SN}(g))\right]_{:,j} = R_{SN}^{(j)}(x_{j\%S}) \ .$$

This concludes the proof of the lemma. ■

Let us go back to the main proof.

Using (30), we have that the block $(c, c') \in [\![1, C]\!]^2$ of size (SN, SN) of $\mathcal{K}^T\mathcal{K}$ is equal to :

$$\begin{pmatrix} C(P_{SN}(\mathbf{K}_{1,c}))^T S_N^T & \dots & C(P_{SN}(\mathbf{K}_{M,c}))^T S_N^T \end{pmatrix} \begin{pmatrix} S_N C(P_{SN}(\mathbf{K}_{1,c'})) \\ \vdots \\ S_N C(P_{SN}(\mathbf{K}_{M,c'})) \end{pmatrix}$$
$$= \sum_{m=1}^{M} C(P_{SN}(\mathbf{K}_{m,c}))^T S_N^T S_N C(P_{SN}(\mathbf{K}_{m,c'})) \ . \tag{56}$$

For any $(m, c, c') \in [\![1, M]\!] \times [\![1, C]\!]^2$, we denote by $(x_{m,c,c',s})_{s=0..S-1}$ the S vectors of $\mathbb{R}^{2k-1}$ obtained when applying Lemma 20 with $h = \mathbf{K}_{m,c}$, and $g = \mathbf{K}_{m,c'}$. Hence, we have, for all $j \in [\![0, SN-1]\!]$,

$$[C(P_{SN}(\mathbf{K}_{m,c}))^T S_N^T S_N C(P_{SN}(\mathbf{K}_{m,c'}))]_{:,j} = R_{SN}^{(j)}(x_{m,c,c',j\%S}) \ . \tag{57}$$

Let $\overline{f}_{k-1} = \begin{bmatrix} 0_{k-1} \\ 1 \\ 0_{k-1} \end{bmatrix} \in \mathbb{R}^{2k-1}$. For all $s \in [\![0, S-1]\!]$, we denote by

$$x_{c,c',s} = \sum_{m=1}^{M} x_{m,c,c',s} - \delta_{c=c'} \overline{f}_{k-1} \in \mathbb{R}^{2k-1}. \tag{58}$$

Note that, from (9), (50), and (54), we have for all $j \in [\![0, SN-1]\!]$, $f_j = R_{SN}^{(j)}(\overline{f}_{k-1})$. Therefore, $Id_{SN} = (f_0, \dots, f_{SN-1}) = \left(R_{SN}^{(0)}(\overline{f}_{k-1}), \dots, R_{SN}^{(SN-1)}(\overline{f}_{k-1})\right)$. We set

$$B_N = \mathcal{K}^T\mathcal{K} - Id_{CSN} \ .$$

We denote by $A_{c,c'}^N \in \mathbb{R}^{SN \times SN}$ the block $(c, c') \in [\![1, C]\!]^2$ of size (SN, SN) of B_N. Using (56), (57), and (58), we have, for all $j \in [\![0, SN-1]\!]$,

$$\begin{aligned} \left[A_{c,c'}^N\right]_{:,j} &= \left[\sum_{m=1}^{M} C(P_{SN}(\mathbf{K}_{m,c}))^T S_N^T S_N C(P_{SN}(\mathbf{K}_{m,c'})) - \delta_{c=c'} Id_{SN}\right]_{:,j} \\ &= \sum_{m=1}^{M} R_{SN}^{(j)}(x_{m,c,c',j\%S}) - \delta_{c=c'} R_{SN}^{(j)}(\overline{f}_{k-1}) \\ &= R_{SN}^{(j)}(x_{c,c',j\%S}) \ . \end{aligned} \tag{59}$$

We then proceed in the same way as in the RO case. Since B_N is clearly symmetric, we have

$$\|B_N\|_2^2 \leq \|B_N\|_1 \|B_N\|_\infty = \|B_N\|_1^2 = \left(\max_{1\leq j \leq CSN} \sum_{i=1}^{CSN} |(B_N)_{i,j}| \right)^2$$
$$= \max_{1\leq c' \leq C\ ,\ 0\leq j \leq SN-1} \left(\sum_{c=1}^{C} \| \left[A_{c,c'}^N \right]_{:,j} \|_1 \right)^2 .$$

Using (59) and (55), this becomes

$$\|B_N\|_2^2 \leq \max_{\substack{1\leq c' \leq C \\ 0\leq j \leq SN-1}} \left(\sum_{c=1}^{C} \|R_{SN}^{(j)} \left(x_{c,c',j\%S}\right)\|_1 \right)^2 = \max_{\substack{1\leq c' \leq C \\ 0\leq s \leq S-1}} \left(\sum_{c=1}^{C} \|x_{c,c',s}\|_1 \right)^2 .$$

We set $(c_0', s_0) \in \arg\max_{\substack{1\leq c' \leq C \\ 0\leq s \leq S-1}} \left(\sum_{c=1}^{C} \|x_{c,c',s}\|_1 \right)^2$. Using that for all $x \in \mathbb{R}^n$, $\|x\|_1 \leq \sqrt{n}\|x\|_2$, we have

$$\|B_N\|_2^2 \leq \left(\sum_{c=1}^{C} \|x_{c,c_0',s_0}\|_1 \right)^2 \leq (2k-1) \left(\sum_{c=1}^{C} \|x_{c,c_0',s_0}\|_2 \right)^2 .$$

Using Cauchy-Schwarz inequality, we obtain

$$\|B_N\|_2^2 \leq (2k-1)C \sum_{c=1}^{C} \|x_{c,c_0',s_0}\|_2^2 \leq (2k-1)C \sum_{c=1}^{C} \sum_{c'=1}^{C} \sum_{s=0}^{S-1} \|x_{c,c',s}\|_2^2 .$$

Using (55) in the particular case of $N' = 2k-1$, we obtain

$$\|B_N\|_2^2 \leq (2k-1)C \sum_{c=1}^{C} \sum_{c'=1}^{C} \sum_{s=0}^{S-1} \|R_{S(2k-1)} \left(x_{c,c',s}\right)\|_2^2$$
$$= C \sum_{c=1}^{C} \sum_{c'=1}^{C} \sum_{s=0}^{S-1} (2k-1) \|R_{S(2k-1)} \left(x_{c,c',s}\right)\|_2^2$$
$$= C \sum_{c=1}^{C} \sum_{c'=1}^{C} \sum_{j=0}^{S(2k-1)-1} \left\| R_{S(2k-1)}^{(j)} \left(x_{c,c',j\%S}\right) \right\|_2^2 .$$

Using (59) for $N' = 2k-1$, we obtain

$$\|B_N\|_2^2 \leq C \sum_{c=1}^{C} \sum_{c'=1}^{C} \sum_{j=0}^{S(2k-1)-1} \left\| \left[A_{c,c'}^{2k-1} \right]_{:,j} \right\|_2^2 = C \|B_{2k-1}\|_F^2 .$$

Using Theorem 7 for $N = 2k-1$, we have $\|B_{2k-1}\|_F^2 = (2k-1)L_{orth}(\mathbf{K})$ and we obtain

$$\|B_N\|_2^2 \leq (2k-1)CL_{orth}(\mathbf{K}) .$$

Therefore, we conclude that, in the CO case

$$(\mathrm{err}_N^s(\mathbf{K}))^2 \leq (2k-1)CL_{orth}(\mathbf{K}) .$$

This concludes the proof in the 1D case.

F.2 Sketch of the Proof of Theorem 8, for 2D Convolutional Layers

In the RO case, we proceed as in the 1D case.
In the CO case, we first prove a lemma similar to Lemma 20, then we proceed as in the 1D case.

Appendix G. Proof of Proposition 3

Below, we prove Proposition 3 for a general matrix $A \in \mathbb{R}^{a\times b}$ with $a \geq b$. In order to obtain the statement for a convolutional layer $\mathcal{K} \in \mathbb{R}^{MN\times CSN}$:
In the RO case ($M \leq CS$): we take $A = \mathcal{K}^T$, $a = CSN$, $b = MN$.
In the CO case ($M \geq CS$): we take $A = \mathcal{K}$, $a = MN$, $b = CSN$.

Let $A \in \mathbb{R}^{a\times b}$ such that $a \geq b$. We denote by $\varepsilon = \|A^TA - Id_b\|_2$. Let $x \in \mathbb{R}^b$, we have

$$\begin{aligned}\left|\|Ax\|^2 - \|x\|^2\right| = \left|x^TA^TAx - x^Tx\right| = \left|x^T(A^TA - Id_b)x\right| &\leq \|x^T\|\|A^TA - Id_b\|_2\|x\| \\ &\leq \varepsilon\|x\|^2 \ .\end{aligned}$$

Hence, for all $x \in \mathbb{R}^b$,

$$(1-\varepsilon)\|x\|^2 \leq \|Ax\|^2 \leq (1+\varepsilon)\|x\|^2 \ .$$

This also implies $\sigma_{max}(A)^2 \leq 1+\varepsilon$. But we know that $\sigma_{max}(A^T) = \sigma_{max}(A)$, hence $\sigma_{max}(A^T)^2 \leq 1+\varepsilon$ and therefore, for all $x \in \mathbb{R}^a$,

$$\|A^Tx\|^2 \leq (1+\varepsilon)\|x\|^2 \ .$$

Finally:

- In the RO case, for $\varepsilon = \text{err}_N^s(\mathbf{K}) = \|\mathcal{K}\mathcal{K}^T - Id_{CSN}\|_2$, $\mathcal{K}$ is ε-AIP.
- In the CO case, for $\varepsilon = \text{err}_N^s(\mathbf{K}) = \|\mathcal{K}^T\mathcal{K} - Id_{MN}\|_2$, $\mathcal{K}$ is ε-AIP.

Appendix H. Experiment Configurations

In this section, we describe the details of the experiments conducted on Cifar10 an Imagenette data sets.

H.1 Cifar10 Experiments

The network architecture used for Cifar10 data set is described in Table H.1 (1.1 million parameters). Conv2D layer will depend on the configuration: classical $Conv2D$ for unconstrained reference configuration, $CayleyConv$ for $Cayley$ configuration (we use Trockman and Kolter, 2021, implementation), and L_{orth} regularization for L_{orth} configuration (we use our implementation according to Definition 1). Weight initialization is done according to *Glorot uniform.*

Task loss is the classical cross-entropy (CE). As described in Béthune et al. (2022), 1-lipschitz property of orthogonal neural network may prevent learning with CE and require introducing a temperature, i.e. multiply the network predictions/logits by a factor τ. Experiments for $Cayley$ and L_{orth} are done with $\tau = 20$ (and $\tau = 1$ for classical $Conv2D$).

Layer	Parameters (M, C, k, k)	Output size (M, H, W)
Input		$32 \times 32 \times 3$
Conv2D, GS2	$(64, 3, 3, 3)$	$64 \times 32 \times 32$
Conv2D, GS2	$(66, 64, 3, 3)$	$66 \times 32 \times 32$
InvDown		$264 \times 16 \times 16$
Conv2D, GS2	$(64, 264, 3, 3)$	$64 \times 16 \times 16$
Conv2D, GS2	$(128, 64, 3, 3)$	$128 \times 16 \times 16$
Conv2D, GS2	$(130, 128, 3, 3)$	$130 \times 16 \times 16$
InvDown		$520 \times 8 \times 8$
Conv2D, GS2	$(128, 520, 3, 3)$	$128 \times 8 \times 8$
Conv2D, GS2	$(192, 128, 3, 3)$	$192 \times 8 \times 8$
Conv2D, GS2	$(194, 192, 3, 3)$	$194 \times 8 \times 8$
InvDown		$776 \times 4 \times 4$
Conv2D, GS2	$(192, 776, 3, 3)$	$192 \times 4 \times 4$
Flatten, Dense	$(10, 3072)$	10

Table 3: Cifar10 Neural network architectures: Conv2D, GS2 is GroupSort2, InvDown is Invertible-Downsampling (Trockman and Kolter, 2021)

We use classical data augmentation: random translation ($\pm 10\%$), random rotation (± 15 degree), random horizontal flipping (0.5 probability), random contrast modification ($[0.8, 1.2]$). For affine transformation zero-padding is used when required. The initial learning rate is set to 0.03, and linearly decreased down to 3×10^{-4}.

The E_{rob} and E_{lip} metrics are computed using the code provided by Trockman and Kolter (2021).

H.2 Imagenette Experiments

The network architecture used for Imagenette data set is described in Table 4 (1.2 million parameters). Conv2D layer will depend on the configuration: classical $Conv2D$ for unconstrained reference configuration, $CayleyConv$ for $Cayley$ configuration (we use Trockman and Kolter, 2021, implementation), and L_{orth} regularization for L_{orth} configuration (we use our implementation according to Definition 1). Weight initialization is done according to *Glorot uniform*.

Task loss is the classical cross-entropy (CE). As described in Béthune et al. (2022), 1-lipschitz property of orthogonal neural network may prevent learning with CE and require introducing a temperature, i.e. multiply the network predictions/logits by a factor τ. Experiments for $Cayley$ and L_{orth} are done with $\tau = 20$ (and $\tau = 1$ for classical $Conv2D$).The initial learning rate is set to 5×10^{-4}, and linearly decreased down to 5×10^{-6}.

Input images are normalized per channel using the recommended mean and std ($[0.485, 0.456, 0.406]$, $[0.229, 0.224, 0.225]$). The only data augmentation used is random horizontal flipping (0.5 probability).

Table 4: Imagenette Neural network architectures: Conv2D, GS2 is GroupSort2, InvDown is InvertibleDownsampling (Trockman and Kolter, 2021)

Layer	Parameters (M, C, k, k)	Output size (M, H, W)
Input		$3 \times 160 \times 160$
Conv2D, GS2	$(32, 3, 3, 3)$	$32 \times 160 \times 160$
Conv2D, GS2	$(34, 32, 3, 3)$	$34 \times 160 \times 160$
InvDown		$136 \times 80 \times 80$
Conv2D, GS2	$(32, 136, 3, 3)$	$32 \times 80 \times 80$
Conv2D, GS2	$(64, 32, 3, 3)$	$64 \times 80 \times 80$
Conv2D, GS2	$(66, 64, 3, 3)$	$66 \times 80 \times 80$
InvDown		$264 \times 40 \times 40$
Conv2D, GS2	$(64, 264, 3, 3)$	$64 \times 40 \times 40$
Conv2D, GS2	$(96, 64, 3, 3)$	$96 \times 40 \times 40$
Conv2D, GS2	$(98, 96, 3, 3)$	$98 \times 40 \times 40$
InvDown		$392 \times 20 \times 20$
Conv2D, GS2	$(96, 392, 3, 3)$	$96 \times 20 \times 20$
Conv2D, GS2	$(128, 96, 3, 3)$	$128 \times 20 \times 20$
Conv2D, GS2	$(130, 128, 3, 3)$	$130 \times 20 \times 20$
InvDown		$520 \times 10 \times 10$
Conv2D, GS2	$(128, 520, 3, 3)$	$128 \times 10 \times 10$
Conv2D, GS2	$(160, 128, 3, 3)$	$160 \times 10 \times 10$
Conv2D, GS2	$(162, 160, 3, 3)$	$162 \times 10 \times 10$
InvDown		$648 \times 5 \times 5$
Conv2D, GS2	$(160, 648, 3, 3)$	$160 \times 5 \times 5$
Flatten, Dense	$(10, 4000)$	10

Appendix I. Computing the Singular Values of $\mathcal{K}$

In this appendix, we describe methods for computing singular values of a 2D layer transform matrix, with or without stride. The codes are provided in the *DEEL.LIP* [14] library.

I.1 Computing the Singular Values of $\mathcal{K}$ when $S = 1$

For convolutional layers without stride, $S = 1$, we use the algorithm described in Sedghi et al. (2018). We describe the algorithm for 2D convolutional layers in Algorithm 1. The algorithm provides the full list of singular values.

I.2 Computing the Smallest and the Largest Singular Value of $\mathcal{K}$ for any Stride S

For convolutions with stride, $S > 1$, there is no known practical algorithm to compute the list of singular values σ. In this configuration, we use the well-known power iteration algorithm associated with a spectral shift to compute the smallest and the largest singular value $(\sigma_{min}, \sigma_{max})$ of $\mathcal{K}$.

14. `https://github.com/deel-ai/deel-lip`

Algorithm 1 Computing the list of singular values of $\mathcal{K}$, when $S = 1$, (Sedghi et al., 2018).

```
Input: kernel tensor: $\mathbf{K} \in \mathbb{R}^{M\times C\times k\times k}$, channel size: $N \geq k$
Output: list of the singular values of $\mathcal{K}$: $\sigma$
1: procedure COMPUTESINGULARVALUES($\mathbf{K}$,$N$)
2:     transforms = np.fft.fft2($\mathbf{K}$, ($N$,$N$), axes=[0, 1])        ▷ np stands for numpy
3:     sigma = np.linalg.svd(transforms, compute_uv=False)
4:     return sigma
5: end procedure
```

We give the principle of the algorithm in Algorithm 2. For clarity, we assume a function 'λ = power_iteration(M,u_{init})', that applies the power iteration algorithm to a symetric matrix M starting from a random vector u_{init}, and returns its largest eigenvalue λ. In practice, of course, we cannot construct M and the implementation must use the usual functions that apply $\mathcal{K}$ and $\mathcal{K}^T$. A detailed python implementation is provided in the *DEEL.LIP* library.

Algorithm 2 Computing $(\sigma_{min}, \sigma_{max})$, for any $S \geq 1$.

```
Input: kernel tensor: $\mathbf{K} \in \mathbb{R}^{M\times C\times k\times k}$, channel size: $N \geq k$, stride parameter: $S \geq 1$
Output: the smallest and the largest singular value of $\mathcal{K}$: $(\sigma_{min}, \sigma_{max})$
  procedure COMPUTEMINANDMAXSINGULARVALUES($\mathbf{K}$,$N$,$S$)

    if $CS^2 \geq M$ then                                                  ▷ RO case
        u = np.random.randn($M$,$N$,$N$)
        lambda_1 = power_iteration($\mathcal{K}\mathcal{K}^T$, u)
        bigCste = 1.1* lambda_1
        u = np.random.randn($M$,$N$,$N$)
        lambda_2 = power_iteration(bigCste. $\mathrm{Id}_{MN^2} - \mathcal{K}\mathcal{K}^T$, u)
    else                                                                   ▷ CO case
        u = np.random.randn($C$,$SN$,$SN$)
        lambda_1 = power_iteration( $\mathcal{K}^T\mathcal{K}$, u )
        bigCste = 1.1* lambda_1
        u = np.random.randn($C$,$SN$,$SN$)
        lambda_2 = power_iteration(bigCste. $\mathrm{Id}_{CS^2N^2} - \mathcal{K}^T\mathcal{K}$, u)

    end if
    sigma_max = np.sqrt(lambda_1)
    sigma_min = np.sqrt(bigCste-lambda_2)
    return (sigma_min,sigma_max)
  end procedure
```

References

Martin Arjovsky and Léon Bottou. Towards principled methods for training generative adversarial networks. In International Conference on Learning Representation, ICLR'17, 2017.

Martin Arjovsky, Amar Shah, and Yoshua Bengio. Unitary evolution recurrent neural networks. In International Conference on Machine Learning, ICML'16, 2016.

Nitin Bansal, Xiaohan Chen, and Zhangyang Wang. Can we gain more from orthogonality regularizations in training deep networks? Advances in Neural Information Processing Systems, NeurIPS'18, 2018.

Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. IEEE Transactions on Neural Networks, 5(2):157–166, 1994.

Louis Béthune, Alberto González-Sanz, Franck Mamalet, and Mathieu Serrurier. Pay attention to your loss: understanding misconceptions about 1-lipschitz neural networks. arXiv preprint arXiv:2104.05097, 2022.

Elaina Chai, Mert Pilanci, and Boris Murmann. Separating the effects of batch normalization on cnn training speed and stability using classical adaptive filter theory. In Asilomar Conference on Signals, Systems, and Computers, pages 1214–1221. IEEE, 2020.

Moustapha Cisse, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. Parseval networks: improving robustness to adversarial examples. In International Conference on Machine Learning, ICML'17, 2017.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In IEEE Conference on Computer Vision and Pattern Recognition, CVPR'09, 2009.

Alan Edelman, Tomás A Arias, and Steven T Smith. The geometry of algorithms with orthogonality constraints. SIAM journal on Matrix Analysis and Applications, 20(2):303–353, 1998.

Farzan Farnia, Jesse Zhang, and David Tse. Generalizable adversarial training via spectral normalization. In International Conference on Learning Representations, ICLR'18, 2018.

Yanhong Fei, Yingjie Liu, Xian Wei, and Mingsong Chen. O-vit: Orthogonal vision transformer. arXiv preprint arXiv:2201.12133, 2022.

Henry Gouk, Eibe Frank, Bernhard Pfahringer, and Michael J Cree. Regularisation of neural networks by enforcing lipschitz continuity. Machine Learning, 110(2):393–416, 2021.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. Advances in Neural Information Processing Systems, NeurIPS'17, 2017.

Pei-Chang Guo and Qiang Ye. On the regularization of convolutional kernel tensors in neural networks. Linear and Multilinear Algebra, 0(0):1–13, 2020.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In IEEE conference on Computer Vision and Pattern Recognition, CVPR'16, 2016.

Felix Heide, Wolfgang Heidrich, and Gordon Wetzstein. Fast and flexible convolutional sparse coding. In IEEE Conference on Computer Vision and Pattern Recognition, CVPR'15, 2015.

Sepp Hochreiter. Untersuchungen zu dynamischen neuronalen netzen. Diploma, Technische Universität München, 91(1), 1991.

Jeremy Howard. Imagenette, 2020. URL `https://github.com/fastai/imagenette/`.

Lei Huang, Xianglong Liu, Bo Lang, Adams Yu, Yongliang Wang, and Bo Li. Orthogonal weight normalization: Solution to optimization over multiple dependent stiefel manifolds in deep neural networks. In Conference on Artificial Intelligence, AAAI'18, 2018.

Lei Huang, Li Liu, Fan Zhu, Diwen Wan, Zehuan Yuan, Bo Li, and Ling Shao. Controllable orthogonalization in training dnns. In IEEE Conference on Computer Vision and Pattern Recognition, CVPR'20, 2020.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In International Conference on Machine Learning, ICML'15, 2015.

Xu Ji, João F Henriques, and Andrea Vedaldi. Invariant information clustering for unsupervised image classification and segmentation. In IEEE/CVF International Conference on Computer Vision, ICCV'19, 2019.

Kui Jia, Shuai Li, Yuxin Wen, Tongliang Liu, and Dacheng Tao. Orthogonal deep neural networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, TPAMI'19, 2019.

Bobak Kiani, Randall Balestriero, Yann LeCun, and Seth Lloyd. Projunn: efficient method for training deep networks with unitary matrices. In Advances in Neural Information Processing Systems, NeurIPS'22, 2022.

Hyunjik Kim, George Papamakarios, and Andriy Mnih. The lipschitz constant of self-attention. In International Conference on Machine Learning, ICML'21, 2021.

D. P. Kingma and M. Welling. Auto-encoding variational bayes. In International Conference on Learning Representations, ICLR'14, 2014.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In International Conference on Learning Representations, ICLR'15, 2015.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. Advances in Neural Information Processing Systems, NeurIPS'12, 2012.

Agostina J Larrazabal, César Martínez, Jose Dolz, and Enzo Ferrante. Orthogonal ensemble networks for biomedical image segmentation. In International Conference on Medical Image Computing and Computer-Assisted Intervention, pages 594–603. Springer, 2021.

Yann LeCun and Yoshua Bengio. Convolutional networks for images, speech, and time series. The handbook of brain theory and neural networks, 3361(10), 1995.

Mario Lezcano-Casado and David Martınez-Rubio. Cheap orthogonal constraints in neural networks: A simple parametrization of the orthogonal and unitary group. In International Conference on Machine Learning, ICML'19, 2019.

Jun Li, Fuxin Li, and Sinisa Todorovic. Efficient riemannian optimization on the stiefel manifold via the cayley transform. In International Conference on Learning Representations, ICLR'19, 2019a.

Qiyang Li, Saminul Haque, Cem Anil, James Lucas, Roger B Grosse, and Jörn-Henrik Jacobsen. Preventing gradient attenuation in lipschitz constrained convolutional networks. In Advances in Neural Information Processing Systems, NeurIPS'19, 2019b.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In International Conference on Learning Representations, ICLR'18, 2018.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for Generative Adversarial Networks. In International Conference on Learning Representations, ICLR'18, 2018.

Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In IEEE Conference on Computer Vision and Pattern Recognition, CVPR'15, 2015.

Uche Osahor and Nasser M Nasrabadi. Ortho-shot: Low displacement rank regularization with data augmentation for few-shot learning. In IEEE/CVF Winter Conference on Applications of Computer Vision, 2022.

Haozhi Qi, Chong You, Xiaolong Wang, Yi Ma, and Jitendra Malik. Deep isometric learning for visual recognition. In International Conference on Machine Learning, ICML'20, 2020.

Haifeng Qian and Mark N Wegman. L2-nonexpansive neural networks. In International Conference on Learning Representations, ICLR'18, 2018.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. Advances in Neural Information Processing Systems, NeurIPS'15, 2015.

Hanie Sedghi, Vineet Gupta, and Philip M Long. The singular values of convolutional layers. In International Conference on Learning Representations, ICLR'18, 2018.

Mathieu Serrurier, Franck Mamalet, Alberto González-Sanz, Thibaut Boissin, Jean-Michel Loubes, and Eustasio Del Barrio. Achieving robustness in classification using optimal transport with hinge regularization. In IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR'21, 2021.

Evan Shelhamer, Jonathan Long, and Trevor Darrell. Fully convolutional networks for semantic segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence, TPAMI'17, 2017.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In International Conference on Learning Representations, ICLR'15, 2015.

Sahil Singla and Soheil Feizi. Skew orthogonal convolutions. In International Conference on Machine Learning, ICML'21, 2021.

Jure Sokolić, Raja Giryes, Guillermo Sapiro, and Miguel RD Rodrigues. Robust large margin deep neural networks. IEEE Transactions on Signal Processing, 2017.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In International Conference on Learning Representations, ICLR'14, 2014.

Asher Trockman and J Zico Kolter. Orthogonalizing convolutional layers with the Cayley Transform. In International Conference on Learning Representations, ICLR'21, 2021.

Yusuke Tsuzuku, Issei Sato, and Masashi Sugiyama. Lipschitz-margin training: Scalable certification of perturbation invariance for deep neural networks. Advances in Neural Information Processing Systems, NeurIPS'18, 2018.

Aladin Virmaux and Kevin Scaman. Lipschitz regularity of deep neural networks: analysis and efficient estimation. Advances in Neural Information Processing Systems, NeurIPS'18, 2018.

Jiayun Wang, Yubei Chen, Rudrasis Chakraborty, and Stella X Yu. Orthogonal convolutional neural networks. In IEEE Conference on Computer Vision and Pattern Recognition, CVPR'20, 2020.

Lechao Xiao, Yasaman Bahri, Jascha Sohl-Dickstein, Samuel Schoenholz, and Jeffrey Pennington. Dynamical isometry and a mean field theory of CNNs: How to train 10,000-layer vanilla convolutional neural networks. In International Conference on Machine Learning, ICML'18, 2018.

Di Xie, Jiang Xiong, and Shiliang Pu. All you need is beyond a good init: Exploring better solution for training extremely deep convolutional neural networks with orthonormality and modulation. In IEEE Conference on Computer Vision and Pattern Recognition, CVPR'17, 2017.

Huan Xu and Shie Mannor. Robustness and generalization. Machine learning, 86(3):391–423, 2012.

Keiji Yanai, Ryosuke Tanno, and Koichi Okamoto. Efficient mobile implementation of a CNN-based object recognition system. In ACM International Conference on Multimedia, ACMMM'16, 2016.

Yao-Yuan Yang, Cyrus Rashtchian, Hongyang Zhang, Russ R Salakhutdinov, and Kamalika Chaudhuri. A closer look at accuracy vs. robustness. Advances in Neural Information Processing Systems, NeurIPS'20, 2020.

Guoqiang Zhang, Kenta Niwa, and W Bastiaan Kleijn. Approximated orthonormal normalisation in training neural networks. arXiv preprint arXiv:1911.09445, 2019.

Junming Zhang, Ruxian Yao, Wengeng Ge, and Jinfeng Gao. Orthogonal convolutional neural networks for automatic sleep stage classification based on single-channel EEG. Computer Methods and Programs in Biomedicine, 2020.

Xiang Zhang, Junbo Zhao, and Yann Lecun. Character-level convolutional networks for text classification. Advances in Neural Information Processing Systems, NIPS'15, 2015.