Existence, Stability and Scalability of Orthogonal Convolutional Neural Networks

El Mehdi Achour, François Malgouyres, Franck Mamalet

Existence, Stability and Scalability of Orthogonal Convolutional Neural Networks

El Mehdi Achour[1], François Malgouyres[1], and Franck Mamalet[2]

[1]Institut de Mathématiques de Toulouse ; UMR 5219, Université de Toulouse ; CNRS , UPS IMT F-31062 Toulouse Cedex 9, France
[2]Institut de Recherche Technologique Saint Exupéry, Toulouse, France

February 8, 2022

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 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 condition and 'same' boundary condition 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 [44]. 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, the landscape of the regularization term is studied and the regularization strategy is validated on real datasets.

Altogether, the study guarantees that the regularization with L_{orth} [44] 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 [2]. For Convolutional Neural Networks (CNN) [22, 21, 52], the introduction of the orthogonality constraint is a way to improve the neural network in several regards. First, despite well established solutions [11, 17], the training of very deep convolutional networks remains difficult. This is in particular due to vanishing/exploding gradients problems [13, 4]. As a result, the expressive capacity of convolutional layers is not fully exploited [17]. 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 [41, 29]. For these reasons, some authors have introduced Lipschitz [41, 31, 8, 43, 35] and orthogonality constraints to convolutional layers [47, 5, 16, 51, 25, 10, 30, 44, 42, 19, 24, 15, 19, 3]. Beside the above motivations for considering Lipschitz and orthogonality constraints, these constraints are commonly used : - in Generative Adversarial Networks (GAN) [28] and Wasserstein-GAN [1, 9]; - in Recurrent Neural Networks [2, 15].

Orthogonal convolutional networks are made of several orthogonal convolutional layers. This article focuses on theoretical properties of orthogonal convolutional 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. 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?

Besides, we will rely on recently published papers [44, 30] which characterize orthogonal convolutional layers as the zero level set of a particular function that is called L_{orth} in [44][1] (see Sect. 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 [44]).

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} 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 interrogations, thus showing theoretical bounds proving that the regularization with L_{orth} is stable, and can be used in most cases to ensure quasi-orthogonality of the convolutional layers.

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 are in Section 3. The code will be made available in *DEEL.LIP*[2] 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 [6]. 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 component during optimization. The Stiefel Manifold has many other nice properties that make it suitable to (local) Riemannian optimization [23, 24]. Orthogonal convolutional layers are a subpart of this Stiefel Manifold. To the best of our knowledge, the understanding of orthogonal convolutional layers is weak. There is no paper focusing on the theoretical properties of orthogonal convolutional layers.

[1] The situation is more complex in [44, 30]. One of the contributions of the present paper is to clarify the situation. We describe here the clarified statement.

[2] https://github.com/deel-ai/deel-lip

Many articles [48, 5, 40, 19, 34, 8, 7] focus on Lipschitz and orthogonality constraints of the neural networks 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** [47, 5, 16, 51, 10, 19, 24, 15, 19, 3, 36] and **convolutional layer orthogonality** [25, 30, 44, 42]. 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, [5, 44] for more details), and the im2col matrix $U(X)$ where the columns of $U(X) \in \mathbb{R}^{Ck^2 \times N^2}$ contain the vectorized patches of X required to compute the M output channels at a given spatial position (see [12, 49]). We therefore have, $\mathrm{Vect}\,(Y) = \mathrm{Vect}\,\big(\overline{\mathbf{K}}U(X)\big)$. 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[3] and an orthogonal dimensionality reduction has no reason to be orthogonal. This phenomenon has been observed empirically in [25] and [19]. The authors of [44] and [30] 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 exploding, 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., [24, 25, 42, 39, 16, 51]). Note that some convolutional layer orthogonality methods in this case consider iterations of $\mathcal{K}$, therefore resulting in a convolution where the kernel is of size larger than $k \times k$.

- **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., [3, 5, 30, 44, 47, 10, 19, 15]).

Note that unlike Kernel Orthogonality, Convolutional Layer Orthogonality deals directly with $\mathcal{K}$, and thus has a complexity which generally depends on N. However, in the context of Soft Convolutional Layer Orthogonality, the authors of [30, 44] introduce the regularizer L_{orth} which is independent of N (see Sect. 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 [44], orthogonal convolutional layers involving a stride are considered for the first time. The authors also provide very impressive classification experiments using L_{orth} on CIFAR100 and ImageNet, including in a semi-supervised setting, on image inpainting, image generation and robustness.

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; we unify the L_{orth} formulation for both Row-Orthogonality and Column-Orthogonality cases; and prove that the regularization with L_{orth}: 1/ is stable ($L_{orth}(\mathbf{K})$ small $\Longrightarrow \mathcal{K}\mathcal{K}^T - Id$ small in various senses); 2/ leads to an orthogonality error that scales favorably when input signal size N grows. We empirically show that, in most cases, the landscape of L_{orth} is benign and we identify the problematic cases. Finally, we also illustrate how the regularization parameter can be chosen to control the tradeoff between accuracy and orthogonality.

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

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. We also derive the notions of approximate orthogonality and relate it to the approximate isometry property whose benefits are listed in Table 1.

1.3.1 Orthogonality

Given a kernel tensor $\mathbf{K}$, the 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).

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 [44, 30]. 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[4] between h and the zero padding of g (see Figure 1). Formally, for all $i, j \in [\![0, 2P]\!]$,

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

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

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

[4] As is common in machine learning, we do not flip h.

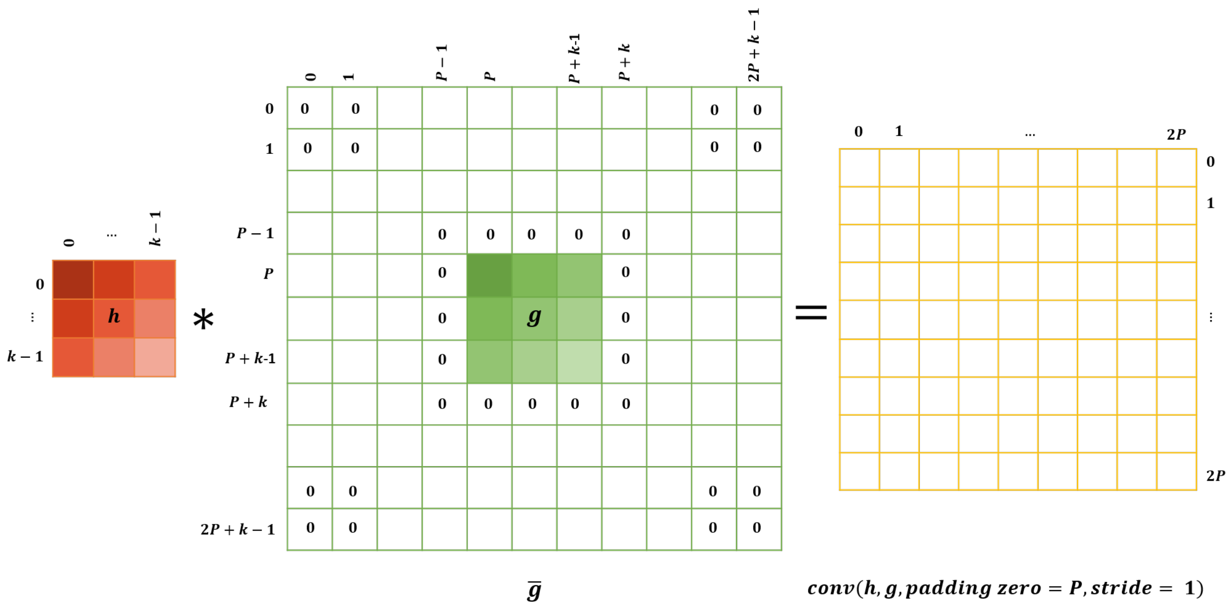

Figure 1: Illustration of $\mathrm{conv}(h, g, \text{padding zero} = P, \text{stride} = 1)$, in the 2D case.

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 $m, l \in [\![1, M]\!]$,

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

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 [44] that, in the RO case, when $P = \left\lfloor \frac{k-1}{S} \right\rfloor S$,

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

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

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,* $M \leq CS^2$*:*

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

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

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

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 [44, 30] is in the CO case. In this case with $S = 1$, [30, 44] 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 [44], 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:

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

Once adapted to our notations, the authors in [44, 30] 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 the sequel, 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 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, 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:

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

and

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

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 Froebenius and spectral norms are such that

$$\|A\|_F \leq \sqrt{\min(a,b)}\|A\|_2 \qquad \text{and} \qquad \|A\|_2 \leq \|A\|_F \; . \tag{6}$$

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

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 [5, 46, 24, 15]. This property also enables to keep the examples well separated [30], like the batch normalization does, and to have a Lipschitz forward pass and therefore improve robustness [44, 5, 25, 42, 19].

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).

Table 1: Properties of a ε-AIP layer (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.

		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.	$M = CS^2$	✓	✓	✓	✓

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, $M \leq CS^2$:*

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

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

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

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

Proposition 1. *Let N be such that $SN \geq k$. We have, both in the RO case and CO case,*

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

This statement actually holds for any matrix (not only layer transform matrix) and is already stated in [3, 10]. For completeness, we provide a proof, in Appendix G.

In Proposition 1 and in Theorem 1 (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 the layer satisfying the AIP, 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 [20]. 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 $\mathrm{err}_N^F(\mathbf{K}) \leq \varepsilon$, 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 [44].

2 Theoretical analysis of orthogonal convolutional layers

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.

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.

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. We highlight in Section 2.2 restrictions for the 'valid' and 'same' zero padding 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 of a convolutional layer (M, C, k, S) and N for an orthogonal convolutional layers 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 1. *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 1 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 (also called shortcut connection) with stride in Resnet [11] 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 1 says that there is no orthogonal convolutional layer for this type of layers.

To conclude, the main consequence of Theorem 1 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 not for technical reasons neither to enable the use of Fourier basis. We illustrate in the next two propositions that, for convolutions defined with the 'valid' condition, or the 'same' condition with zero padding, the orthogonality is too restrictive.

We consider in this section an unstrided convolution and we state results about other paddings and some of their limitations.

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

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

Let $k = 2r + 1$, and 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 3. *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 $(m, c) \in [\![1, M]\!] \times [\![1, C]\!]$, $\mathbf{K}_{m,c} = \alpha_{m,c}e_{r,r}$. As a consequence*

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

This proposition holds in the 1D and 2D case. 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 convolutions layers are trivial transformations.

2.3 Frobenius norm stability

We recall that the motivation behind this is the following : The authors of [44, 30] 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 it can not occur. Therefore, if $L_{orth}(\mathbf{K})$ is small, $\text{err}_N^F(\mathbf{K})$ 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 2. *Let N be such that $SN \geq 2k - 1$ and $d = 1$ or 2. We have, both in the RO case and CO case,*

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

Theorem 2 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 [44].

Using Theorem 2, we find that (5) becomes

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

Once the parameter λ is made dependent of 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 2 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 [30] for $S = 1$, and the result stated in [44] in the RO case for any S .

Considering another signal size N' and applying Theorem 2 with the sizes N and N', we find

$$(\text{err}_{N'}^F(\mathbf{K}))^2 = \frac{(N')^d}{N^d}(\text{err}_N^F(\mathbf{K}))^2.$$

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 dataset containing signals/images of a given size, but the inference is done for signals/images of varying size [32, 26, 18].

Finally, using (6) and Proposition 1, $\mathcal{K}$ is ϵ-AIP with ϵ scaling like the square root of the signal/image size. This might not be satisfactory. We prove in the next section that it is actually not the case.

2.4 Spectral norm stability and scalability

We prove in Theorem 3 that $\mathrm{err}_N^s(\mathbf{K})^2$ is bounded by a quantity which is proportional to $L_{orth}(\mathbf{K})$ and the multiplicative factor does not depend on N. Hence, when $L_{orth}(\mathbf{K})$ is small, $\mathrm{err}_N^s(\mathbf{K})^2$ is also small for all N. As a consequence, regularizing with $L_{orth}(\mathbf{K})$ is efficient for all N, even if the algorithm does not reach $L_{orth}(\mathbf{K}) = 0$.

Moreover, combined with Proposition 1 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 [44, 30].

Theorem 3. *Let N be such that $SN \geq 2k-1$ and $d = 1$ or 2. We have,*

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

with:

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

Theorem 3 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 3, unlike with the Frobenius norm, the spectral norm of the orthogonality residual is bounded by a quantity which does not depend on N. Moreover, $\sqrt{\alpha}$ is usually moderately large. For instance, with $(M, C, k, S) = (128, 128, 3, 2)$, for images, $\sqrt{\alpha} \leq 34$. For usual architectures, $\sqrt{\alpha}$ is smaller than 200. This ensures that, independently of N, we have a tight control of the AIP, when $L_{orth}(\mathbf{K}) \ll 1$, both in the RO case and CO case. We recall that this is what explains the benefits of the approach, as stated in Table 1. Experiments which confirm this statement are in Section 3. This explains some of the impressive results obtained empirically on real datasets by [44, 30].

3 Experiments

Before illustrating the benefits of approximate orthogonality in section 3.4, we conduct several synthetic experiments to test and illustrate the theorems of Section 2. 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 $k \in \{1, 3, 5, 7\}$, leading to 44924 (among 49152) architectures for which an orthogonal convolutional layer exists. The model is trained using a *Glorot uniform* initializer and a *Adam* optimizer with learning rate 0.01 on a null loss ($L_{task} = 0$) and the L_{orth} regularization (see Definition 1) during 3000 steps.

After training, we evaluate the singular values (σ) of $\mathcal{K}$ for different input sizes $SN \times SN$. When $S = 1$, we can compute all the singular values of $\mathcal{K}$ with the algorithm in [35]. For convolutions with stride, $S > 1$, there is no practical algorithm to compute the singular values and we simply apply the well known power iteration algorithm, to retrieve the smallest and largest singular values ($\sigma_{min}, \sigma_{max}$) of $\mathcal{K}$ (see Appendix H). We remind that, when $\mathcal{K}$ is orthogonal, we have $\sigma_{min} = \sigma_{max} = 1$.

3.1 Optimization landscape

We plot $(\sigma_{min}, \sigma_{max})$ for $\mathcal{K}$ such that $SN \times SN = 64 \times 64$, on Figure 2. Each experiment is represented by two points: σ_{max}, in blue, and σ_{min}, in orange. For each point (x, y), the first coordinate x corresponds to $\frac{M}{CS^2}$, and the second coordinate y denotes the singular value of the corresponding $\mathcal{K}$. 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 Fig. 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 Fig. 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 [38] and Resnet [11]. We have conducted experiments that we do not report here in details, and it seems that this

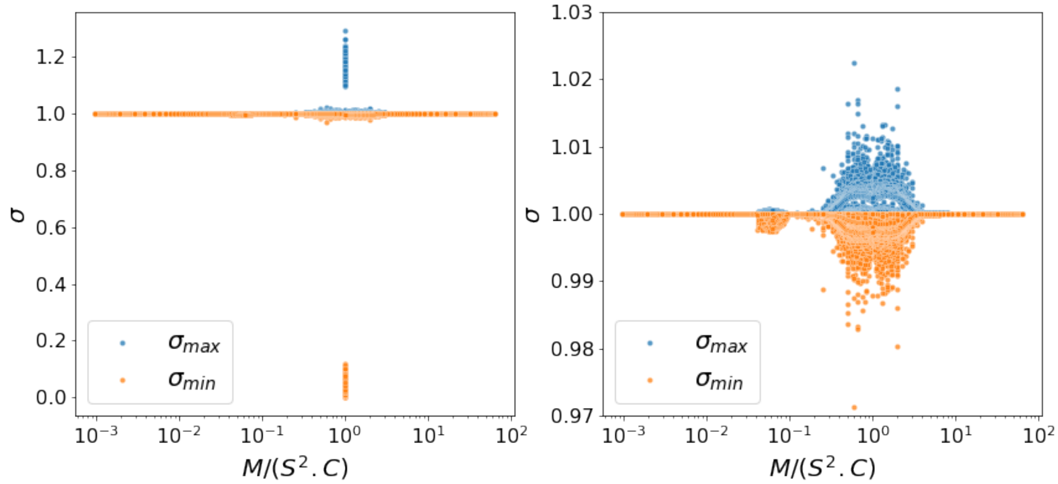

Figure 2: **Optimization of $\mathbf{L_{orth}}$.** Each experiment corresponds to two dots: a blue dot for σ_{max} and an orange dot for σ_{min}. The x-axis is M/CS^2 in log scale. (left) All experiments for which $\mathbb{K}_2^\perp \neq \emptyset$; (right) All experiments for which $\mathbb{K}_2^\perp \neq \emptyset$ and $M \neq CS^2$.

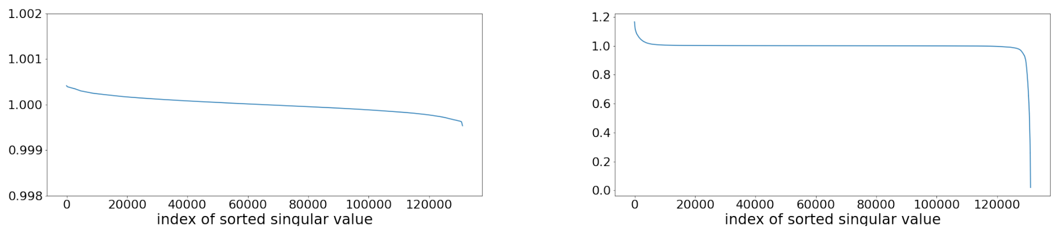

Figure 3: **Singular values of** $\mathcal{K}$, when $C = M$ and $S = 1$ and optimization is (Left) successful, L_{orth} small (Right) Unsuccessful, L_{orth} large.

is specific to the convolutional layer case. Fully-connected layers do not suffer from this phenomenon when optimized to be orthogonal.

3.2 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, number of iterations and obtained similar results. This suggests that, in the failure cases, the landscape of L_{orth} does not lend itself to global optimization. The explanation of this phenomenon and the evaluation of its impact on applications are open questions that we keep for future research. The contributions of the article is to empirically identify these problematic cases. We also ran 100 training experiments, with independent initialization, for each configuration when $M = CS^2$ ($M \in [\![1, 64]\!]$ and $k \in \{1, 3, 5, 7\}$). In average, at convergence, we found $\sigma_{min} \sim 1 \sim \sigma_{max}$ in 14% of runs, proving that the minimizer can be reached.

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 datasets in [44] and [30]. Notice that [44] contains a curve similar to Figure 3 when used for a real dataset.

3.3 Stability of $(\sigma_{min}, \sigma_{max})$ when N varies

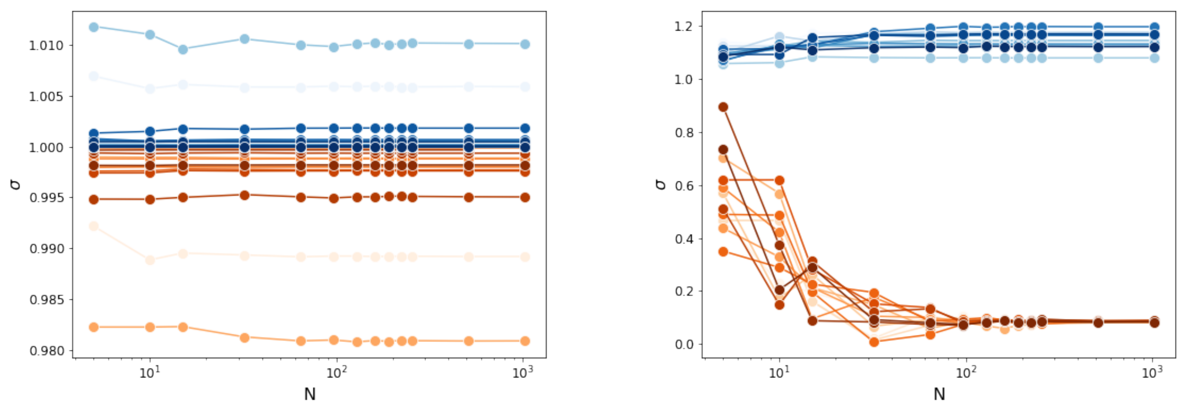

Figure 4: **Evolution of $\sigma_{\mathbf{min}}$ and $\sigma_{\mathbf{max}}$ according to input image size** (x-axis: N in log-scale) (Left) successful training, L_{orth} small, (Right) unsuccessful training, L_{orth} large

In this experiment, we evaluate how the singular values $(\sigma_{min}, \sigma_{max})$ 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 [37, 18, 32] 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 and 50 for which it was unsuccessful. They are respectively used to construct the figures on the left and the right of Figure 4. We display the $(\sigma_{min}, \sigma_{max})$ values of $\mathcal{K}$ as orange and blue dots, for $N \in \{5, 12, 15, 32, 64, 128, 256, 512, 1024\}$. The dots corresponding to the same $\mathbf{K}$ are linked by a line.

We see, on the left of Figure 4, that for successful experiments (L_{orth} small), the singular values are very stable when N varies. This corresponds to the behavior described in Theorem 3 and Proposition 1. We also point out, on the right of Figure 4, that for unsuccessful optimization (L_{orth} large), σ_{min} (resp. σ_{max}) values decrease (resp. increase) rapidly when N increases.

3.4 Datasets experiments

In this section we compare performance, robustness, and processing time, on several datasets, for the L_{orth} regularization and a method that we call Cayley [42], a hard convolutional layer orthogonality method[5]. As a reminder, [42] method is based on the Cayley transform. It builds convolutions parameterized by $k \times k$ parameters but, because a mapping is applied to obtain the 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.

On Cifar10, we use the same configuration as in [42]: a standard data augmentation (i.e., random cropping and flipping), a $KWLarge$ architecture [25, 45], a piecewise triangular learning rate, a multiclass hinge loss with a $\sqrt{2}\epsilon$ margin, where $\epsilon = 36/255$, and an Adam optimizer. In all experiments, for fair comparison, we use $CayleyLinear$ dense layers, and invertible downsampling emulation as in [42].

For L_{orth} regularization and for $\lambda \in \{10, 1, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}\}$, we investigate the properties of the solution of (5). Since small λ can lead to poor regularization, after training σ_{max} and σ_{min} values are computed for each convolutional layer.

[5]comparison with kernel orthogonality was done in [44]

In the following tables, the non-lipschitz-constraint convolution (Conv2D) performance is given as a reference[6]. We report the following metrics for each experiments:

- *Acc. clean*: Classical accuracy on test set
- $\Sigma_{max} = \max_l(\sigma_{max}(\mathcal{K}_l))$: the largest singular value over all the convolutional layers
- L_{up} : An upper bound of the Lipschitz constant computed as the product $L_{up} = \prod_l \sigma_{max}(\mathcal{K}_l)$ of convolution layers largest singular values.
- $\Sigma_{min} = \min_l(\sigma_{min}(\mathcal{K}_l))$: the smallest singular value over all the convolutional layers
- L_{low}: the product $L_{low} = \prod_l \sigma_{min}(\mathcal{K}_l)$ of convolution layers lowest singular values.
- E_{lip} : Empirical local Lipschitz constants computed using the PGD-like method proposed by [50].
- E_{rob} : The empirical robustness accuracy, i.e. the proportion of test samples on which a vanilla Projected Gradient Descent (PGD) attack [27] failed (for a coefficient $\alpha = \epsilon/4.0$). The (*Drop in Acc.*) represents the difference between the test clean accuracy and this value.
- T_{epoch}: the average epoch processing time

Table 2: **Cifar10**: Influence of λ for L_{orth} regularization and comparison with Cayley method.

Method	Conv2D	Cayley	L_{orth} 10^1	L_{orth} 1.00	L_{orth} 10^{-1}	L_{orth} 10^{-2}	L_{orth} 10^{-3}	L_{orth} 10^{-4}	L_{orth} 10^{-5}
Acc. clean	0.83	0.75	0.72	0.73	0.74	0.75	0.78	0.81	0.82
Σ_{max}	8.85	1.00	1.00	1.03	1.17	1.56	2.18	2.80	3.77
L_{up}	260.30	1.00	1.01	1.05	1.34	2.56	6.32	18.74	52.41
Σ_{min}	0.41	1.00	1.00	0.99	0.96	0.84	0.56	0.47	0.30
L_{low}	0.13	1.00	1.00	0.99	0.94	0.75	0.42	0.26	0.24
E_{lip}	16.41	0.76	0.72	0.73	0.79	1.08	2.01	3.48	7.39
E_{rob} (Drop in acc.)	0.51 (0.32)	0.68 (0.07)	0.65 (0.07)	0.66 (0.07)	0.68 (0.06)	0.67 (0.08)	0.67 (0.11)	0.67 (0.14)	0.62 (0.19)
T_{epoch}	4.00	5.80	4.20	4.20	4.20	4.20	4.20	4.20	4.20

Table 2 shows that the regularization parameter λ, in (5), provides a way to tune a tradeoff between robustness (Drop in acc.) and clean accuracy, by controlling the Lipschitz constant of the layers. The L_{low} line shows that λ allows to control the importance of the vanishing gradient. The configurations $\lambda = 10^{-1}$ and 10^{-2} achieve similar empirical performances as the Cayley method. Furthermore their empirical Lipschitz constants are very close to one. The processing time for the regularizing with L_{orth} is 1.4 times faster than the one of the Cayley method. It is not reported here in details but the convergence speed 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 which involves Fourier transforms. This leads to a higher computational complexity even at inference. The change of support also explains the difference of the accuracies between the Cayley method and the regularization with $\lambda = 10$.

Table 3 presents the same experiments on Imagenette dataset [14]. The latter is a 10 classes subset of Imagenet dataset [33] with 160×160 images. We train a $KWLarge$-like architecture with five convolutional blocks and pooling, and use the same parameters as for Cifar10 experiments (data augmentation, optimizer, loss, epsilon). Interestingly, although the drop in accuracy is increased for $\lambda = 10^{-4}$ and 10^{-5}, the clean performance increases sufficiently to

[6]It gives a reference performance for neural networks with orthogonal dense layers

Table 3: **Imagenette**: Influence of λ for L_{orth} regularization and comparison with Cayley method.

Method	Conv2D	Cayley	L_{orth} 1.00	L_{orth} 10^{-1}	L_{orth} 10^{-2}	L_{orth} 10^{-3}	L_{orth} 10^{-4}	L_{orth} 10^{-5}
Acc. clean	0.79	0.75	0.68	0.70	0.70	0.75	0.79	0.79
Σ_{max}	9.82	1.00	1.00	1.00	1.05	1.31	1.71	2.29
L_{up}	25909	1.00	1.00	1.01	1.21	3.06	14.70	217.77
Σ_{min}	0.14	1.00	1.00	1.00	0.99	0.73	0.53	0.15
L_{low}	$< 3.10^{-5}$	1.00	1.00	1.00	0.94	0.44	0.15	$< 8.10^{-4}$
E_{lip}	23.86	0.48	0.42	0.42	0.41	0.49	0.91	2.14
E_{rob} (Drop in acc.)	0.57 (0.22)	0.73 (0.02)	0.66 (0.02)	0.69 (0.01)	0.68 (0.02)	0.72 (0.02)	0.75 (0.04)	0.75 (0.05)
T_{epoch}	16.90	87.30	18.80	18.80	18.80	18.80	18.80	18.80

obtain better robustness measures than with the Cayley method. Besides, because L_{orth} does not depend on the size N of the input feature map, the processing time for the L_{orth} regularization is only 1.1 times slower than for the non-constrained convolution (Conv2D). In comparizon, the Cayley method is 5.2 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 to construct orthogonal convolutional layers in a stable manner, that also scales well with the input size 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.

Acknowledgements

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.[7]

References

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

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

[3] 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, 31:4261–4271, 2018.

[7] https://www.deel.ai/

[4] 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.

[5] Moustapha Cisse, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. Parseval networks: improving robustness to adversarial examples. In Proceedings of the 34th International Conference on Machine Learning, ICML'17, pages 854–863, 2017.

[6] 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.

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

[8] 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.

[9] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of wasserstein gans. In Proceedings of the 31st International Conference on Neural Information Processing Systems, NeurIPS'17, pages 5769–5779, 2017.

[10] 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.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, CVPR'16, pages 770–778, 2016.

[12] 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.

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

[14] Jeremy Howard. imagenette.

[15] Lei Huang, Li Liu, Fan Zhu, Diwen Wan, Zehuan Yuan, Bo Li, and Ling Shao. Controllable orthogonalization in training dnns. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR'20, pages 6429–6438, 2020.

[16] 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 Proceedings of the Conference on Artificial Intelligence, AAAI'18, volume 32, 2018.

[17] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In International Conference on Machine Learning, ICML'15, pages 448–456. PMLR, 2015.

[18] Xu Ji, João F Henriques, and Andrea Vedaldi. Invariant information clustering for unsupervised image classification and segmentation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, ICCV'19, pages 9865–9874, 2019.

[19] 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.

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

[21] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems, NIPS'12, 25:1097–1105, 2012.

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

[23] 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, pages 3794–3803. PMLR, 2019.

[24] 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, 2019.

[25] 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, 2019.

[26] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In IEEE conference on Computer Vision and Pattern Recognition, CVPR'15, 2015.

[27] 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.

[28] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In International Conference on Learning Representations, ICLR'18, 2018.

[29] Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, CVPR'15, pages 427–436, 2015.

[30] 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.

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

[32] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: towards real-time object detection with region proposal networks. In Proceedings of the 28th International Conference on Neural Information Processing Systems, NeurIPS'15, pages 91–99, 2015.

[33] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision, IJCV'15, 115(3):211–252, 2015.

[34] Kevin Scaman and Aladin Virmaux. Lipschitz regularity of deep neural networks: analysis and efficient estimation. In Proceedings of the 32nd International Conference on Neural Information Processing Systems, NeurIPS'18, pages 3839–3848, 2018.

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

[36] 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 Conference on Computer Vision and Pattern Recognition (CVPR'21), 2021.

[37] Evan Shelhamer, Jonathan Long, and Trevor Darrell. Fully convolutional networks for semantic segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence, TPAMI'17, 39(4):640–651, 2017.

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

[39] Sahil Singla and Soheil Feizi. Skew orthogonal convolutions. In Marina Meila and Tong Zhang, editors, Proceedings of the 38th International Conference on Machine Learning, ICML'21, volume 139, pages 9756–9766, 2021.

[40] Jure Sokolić, Raja Giryes, Guillermo Sapiro, and Miguel RD Rodrigues. Robust large margin deep neural networks. IEEE Transactions on Signal Processing, 65(16):4265–4280, 2017.

[41] 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.

[42] Asher Trockman and J Zico Kolter. Orthogonalizing convolutional layers with the cayley transform. In International Conference on Learning Representations, ICLR'21, 2021.

[43] Yusuke Tsuzuku, Issei Sato, and Masashi Sugiyama. Lipschitz-margin training: scalable certification of perturbation invariance for deep neural networks. In Proceedings of the 32nd International Conference on Neural Information Processing Systems, NeurIPS'18, pages 6542–6551, 2018.

[44] Jiayun Wang, Yubei Chen, Rudrasis Chakraborty, and Stella X Yu. Orthogonal convolutional neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR'20, pages 11505–11515, 2020.

[45] Eric Wong, Frank Schmidt, Jan Hendrik Metzen, and J. Zico Kolter. Scaling provable adversarial defenses. In Advances in Neural Information Processing Systems, NeurIPS'18, 2018.

[46] 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.

[47] 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 Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR'17, pages 6176–6185, 2017.

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

[49] Keiji Yanai, Ryosuke Tanno, and Koichi Okamoto. Efficient mobile implementation of a cnn-based object recognition system. In Proceedings of the 24th ACM International Conference on Multimedia, ACMMM'16, page 362–366, 2016.

[50] Yao-Yuan Yang, Cyrus Rashtchian, Hongyang Zhang, Ruslan Salakhutdinov, and Kamalika Chaudhuri. A Closer Look at Accuracy vs. Robustness. arXiv:2003.02460 [cs, stat], 2020.

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

[52] Xiang Zhang, Junbo Zhao, and Yann Lecun. Character-level convolutional networks for text classification. Advances in Neural Information Processing Systems, NIPS'15, 2015:649–657, 2015.

A Notation

First, we set notation.

A.1 Standard math definition

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 $i \neq j$.

For a vector $x = (x_0, \ldots, x_{n-1})^T \in \mathbb{R}^n$, we recall the classic norm definitions, $\|x\|_1 = \sum_{i=0}^{n-1} |x_i|$, and $\|x\|_2 = \sqrt{\sum_{i=0}^{n-1} x_i^2}$. For $x, y \in \mathbb{R}^n$, $\langle x, y \rangle = x^T y$ denotes the scalar product between x and y. We denote by 0_s the null vector of $\mathbb{R}^s$.

For a matrix $A \in \mathbb{R}^{m \times n}$, $\|.\|_2$ denotes the spectral norm defined by $\|A\|_2 = \sigma_{max}(A)$, where $\sigma_{max}(A)$ denotes the largest singular value of A. 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 $Id_n \in \mathbb{R}^{n \times n}$ the identity matrix of size n.

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

$$\begin{cases} (e_i)_{i=0..k-1} = \mathcal{B}(\mathbb{R}^k) \\ (f_i)_{i=0..SN-1} = \mathcal{B}(\mathbb{R}^{SN}) \\ (E_{a,b})_{a=0..N-1, b=0..SN-1} = \mathcal{B}(\mathbb{R}^{N \times SN}) \\ (\overline{E}_{a,b})_{a=0..SN-1, b=0..N-1} = \mathcal{B}(\mathbb{R}^{SN \times N}) \\ (F_{a,b})_{a=0..SN-1, b=0..SN-1} = \mathcal{B}(\mathbb{R}^{SN \times SN}) \\ (G_{a,b})_{a=0..N-1, b=0..N-1} = \mathcal{B}(\mathbb{R}^{N \times N}) \ . \end{cases} \tag{7}$$

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

$$\begin{cases} e_a = e_{a\%k}, \qquad f_a = f_{\hat{a}} \\ E_{a,b} = E_{\tilde{a},\hat{b}}, \qquad \overline{E}_{a,b} = \overline{E}_{\hat{a},\tilde{b}}, \qquad F_{a,b} = F_{\hat{a},\hat{b}}, \qquad G_{a,b} = G_{\tilde{a},\tilde{b}}. \end{cases} \tag{8}$$

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

$$\begin{cases} E_{a,b} F_{c,d} = \delta_{\hat{b}=\hat{c}} E_{a,d}, \qquad E_{a,b} \overline{E}_{c,d} = \delta_{\hat{b}=\hat{c}} G_{a,d} \\ \overline{E}_{a,b} E_{c,d} = \delta_{\tilde{b}=\tilde{c}} F_{a,d}, \qquad F_{a,b} \overline{E}_{c,d} = \delta_{\hat{b}=\hat{c}} \overline{E}_{a,d}. \end{cases} \tag{9}$$

Note also that

$$E_{a,b}^T = \overline{E}_{b,a} \ . \tag{10}$$

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 $M \leq CS$, $\mathcal{K}$ is orthogonal if and only if $\mathcal{K}\mathcal{K}^T = Id_{MN}$.
CO case: When $M \geq CS$, $\mathcal{K}$ 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

$$\mathrm{conv}(h, g, \text{padding zero} = P, \text{stride} = 1) \in \mathbb{R}^{2P+1} \tag{11}$$

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

$$[\mathrm{conv}(h, g, \text{padding zero} = P, \text{stride} = 1)]_i = \sum_{i'=0}^{k-1} h_{i'}\bar{g}_{i'+i} \ , \tag{12}$$

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

$$\bar{g}_i = \begin{cases} g_{i-P} & \text{if } i \in [\![P, P + k - 1]\!], \\ 0 & \text{otherwise.} \end{cases} \tag{13}$$

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

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

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 $i \in [\![0, \lfloor 2P/S\rfloor]\!]$

$$[\mathrm{conv}(h, g, \text{padding zero} = P, \text{stride} = S)]_i = [\mathrm{conv}(h, g, \text{padding zero} = P, \text{stride} = 1)]_{Si}. \tag{15}$$

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 $m, l \in [\![1, M]\!]$,

$$\mathbf{conv}(\mathbf{K}, \mathbf{K}, \text{padding zero} = P, \text{stride} = S)_{m,l,:}$$
$$= \sum_{c=1}^{C} \mathrm{conv}(\mathbf{K}_{m,c}, \mathbf{K}_{l,c}, \text{padding zero} = P, \text{stride} = S). \tag{16}$$

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

$$[I_{r0}]_{m,l,:} = \delta_{m=l} \begin{bmatrix} 0_{P/S} \\ 1 \\ 0_{P/S} \end{bmatrix} \ . \tag{17}$$

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

- In the RO case:
$$L_{orth}(\mathbf{K}) = \| \mathbf{conv}(\mathbf{K}, \mathbf{K}, \text{padding zero} = P, \text{stride} = S) - I_{r0}\|_F^2 \ .$$
- In the CO case:
$$L_{orth}(\mathbf{K}) = \| \mathbf{conv}(\mathbf{K}, \mathbf{K}, \text{padding zero} = P, \text{stride} = S) - I_{r0}\|_F^2 - (M - CS) \ .$$

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

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

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 [35].

B.1 1D case

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

$$S_N = \sum_{i=0}^{N-1} E_{i,Si} \ . \tag{18}$$

Also, note that, using (9) and (10), we have $S_N S_N^T = Id_N$ and

$$S_N^T S_N = \sum_{i=0}^{N-1} F_{Si,Si} \ . \tag{19}$$

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

$$C(x) = \begin{pmatrix} x_0 & x_{n-1} & \cdots & x_2 & x_1 \\ x_1 & x_0 & x_{n-1} & & x_2 \\ \vdots & x_1 & x_0 & \ddots & \vdots \\ x_{n-2} & & \ddots & \ddots & x_{n-1} \\ x_{n-1} & x_{n-2} & \cdots & x_1 & x_0 \end{pmatrix} . \tag{20}$$

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

$$X = C(x) \iff \forall m, l \in [\![0, n-1]\!],\ X_{m,l} = x_{(m-l)\%n} \ . \tag{21}$$

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

$$C(x)^T = C(\tilde{x}) \ . \tag{22}$$

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

$$C(x)C(y) = C(x * y), \tag{23}$$

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

$$[x * y]_j = \sum_{i=0}^{n-1} x_i y_{(j-i)\%n}. \tag{24}$$

$x * y$ 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

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

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 $h = (h_0, \ldots, h_{2r})^T \in \mathbb{R}^k$ the corresponding vector

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

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 $i \in [\![-r, -r+n-1]\!]$,

$$[P_n(h)]_i = \begin{cases} h_{r-i} & \text{if } i \in [\![-r, r]\!] \\ 0 & \text{otherwise.} \end{cases} \tag{26}$$

Single-channel case: Let $x = (x_0, \ldots, x_{SN-1})^T \in \mathbb{R}^{SN}$ be a 1D signal. We denote by Circular_Conv$(h, x, \text{stride} = 1)$ the result of the circular convolution[8] of x with the kernel $h = (h_0, \ldots, h_{2r})^T \in \mathbb{R}^k$. We have

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

Written as a matrix-vector product, this becomes

$$\begin{aligned}
&\begin{pmatrix} h_0 & \cdots & h_{2r} & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & h_0 & \cdots & h_{2r} \end{pmatrix} \begin{pmatrix} x_{SN-r} \\ \vdots \\ x_{SN-1} \\ x_0 \\ \vdots \\ x_{SN-1} \\ x_0 \\ \vdots \\ x_{r-1} \end{pmatrix} \in \mathbb{R}^{SN} \\
&= \begin{pmatrix}
h_r & h_{r+1} & \cdots & h_{2r} & 0 & \cdots & 0 & h_0 & \cdots & h_{r-1} \\
h_{r-1} & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\
\vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & h_0 \\
h_0 & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\
0 & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\
\vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\
0 & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & h_{2r} \\
h_{2r} & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\
\vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & h_{r+1} \\
h_{r+1} & \cdots & h_{2r} & 0 & \cdots & 0 & h_0 & \cdots & h_{r-1} & h_r
\end{pmatrix} x \\
&= C(P_{SN}(h))x \ .
\end{aligned}$$

[8]as defined in machine learning (we do not flip h).

The strided convolution is

$$\text{Circular\_Conv}(h, x, \text{stride} = S) = S_N C(P_{SN}(h))x \in \mathbb{R}^N \; . \tag{27}$$

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 (27) for all the input-output channel correspondances, 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,

$$\mathcal{K} = \begin{pmatrix} S_N C(P_{SN}(\mathbf{K}_{1,1})) & \dots & S_N C(P_{SN}(\mathbf{K}_{1,C})) \\ \vdots & \vdots & \vdots \\ S_N C(P_{SN}(\mathbf{K}_{M,1})) & \dots & S_N C(P_{SN}(\mathbf{K}_{M,C})) \end{pmatrix} \in \mathbb{R}^{MN \times CSN} \tag{28}$$

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 $\mathcal{S}_N \in \mathbb{R}^{N^2 \times S^2 N^2}$ 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 $\text{Vect}(z) = \mathcal{S}_N \, \text{Vect}(x)$).
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

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

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

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

and the strided circular convolution

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

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

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

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

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

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

C Proof of Theorem 1

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

C.1 Proof of Theorem 1, for 1D convolutional layers

We start by stating and proving three intermediate lemmas. Recall that $k = 2r+1$ and from (7), 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 1. *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 (26), (7), (8) and (20), 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 (18) and (9), 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 2. *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 $k_S \le k$, using Lemma 1 and (10),

$$\begin{aligned} S_N C\left(P_{SN}\left(e_j\right)\right) C\left(P_{SN}\left(e_l\right)\right)^T S_N^T &= \left(\sum_{i=0}^{N-1} E_{i,Si+j-r}\right)\left(\sum_{i'=0}^{N-1} E_{i',Si'+l-r}\right)^T \\ &= \left(\sum_{i=0}^{N-1} E_{i,Si+j-r}\right)\left(\sum_{i'=0}^{N-1} \overline{E}_{Si'+l-r,i'}\right)\ . \end{aligned} \tag{29}$$

We know from (9) 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 $k_S \le S$, we have

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

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

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

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 (9), the equality (29) 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 3. *Let $S \le k$. 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 $S \le k$, we have $z \in [\![0, k - 1]\!]$. Hence using Lemma 1, then (10) and (9), 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 (8)

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

□

Proof of Theorem 1. 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 (28), 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 2, 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 (28), 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 3, 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 $\text{rk}(\mathcal{K}^T\mathcal{K}) \leq \text{rk}(\mathcal{K}) \leq CkN < CSN = \text{rk}(Id_{CSN})$. 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. $\square$

C.2 Sketch of the proof of Theorem 1, 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 4. *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 5. *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, $\text{rk}(\mathcal{K}) \leq Ck^2N^2 < MN^2$. 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 $\text{rk}(\mathcal{K}) \leq Ck^2N^2 < CS^2N^2$. Therefore, for all $\mathbf{K} \in \mathbb{R}^{M \times C \times k \times k}$, we have $\mathcal{K}^T\mathcal{K} \neq Id_{CS^2N^2}$.

D Restrictions due to boundary conditions

D.1 Proof of Proposition 2

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 3

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. □

E Proof of Theorem 2

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

E.1 Proof of Theorem 2, in the 1D case

Before proving Theorem 2, we first present three intermediate lemmas.

Lemma 6. *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 (21), $Y_{m,n} = x_{(Sm-Sn)\%SN} = x_{S((m-n)\%N)}$. 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 (21), 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

$$Q_{S,N}(x) = (x_{\frac{P}{S}}, \ldots, x_{2\frac{P}{S}}, 0, \ldots, 0, x_0, x_1, \ldots, x_{\frac{P}{S}-1})^T \in \mathbb{R}^N. \tag{30}$$

Lemma 7. *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*

$$S_N C(P_{SN}(h)) C(P_{SN}(g))^T S_N^T = C(Q_{S,N}(\text{conv}(h, g, \textit{padding zero} = P, \textit{stride} = S))) \ . \tag{31}$$

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 (31). 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 (22), (23), and (24), 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

$$C(P_{SN}(h))C(P_{SN}(g))^T = C(b^{(SN)}[h,g]) . \tag{32}$$

To simplify the forthcoming notation, we temporarily denote by

$$b := b^{(SN)}[h,g]. \tag{33}$$

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 (26), we have, for $i \in [\![-r, -r+SN-1]\!]$,

$$[P_{SN}(h)]_i = \begin{cases} h_{r-i} & \text{if} \quad i \in [\![-r, r]\!] \\ 0 & \text{if} \quad i \in [\![r+1, -r+SN-1]\!] . \end{cases} \tag{34}$$

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

$$\begin{aligned}
b_j &= \sum_{i=0}^{SN-1} [P_{SN}(h)]_i [P_{SN}(g)]_{i-j} \\
&= \sum_{i=-r}^{SN-1-r} [P_{SN}(h)]_i [P_{SN}(g)]_{i-j} \\
&= \sum_{i=-r}^{r} [P_{SN}(h)]_i [P_{SN}(g)]_{i-j}.
\end{aligned} \tag{35}$$

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 $j \in [\![-2r, SN-2r-1]\!]$,

$$\left\{ i \in [\![-r, r]\!] \quad | \quad [P_{SN}(h)]_i [P_{SN}(g)]_{i-j} \neq 0 \right\} \subset [\![-r, r]\!] \cap [\![-r+j, r+j]\!]. \tag{36}$$

Let us now analyse the right-side of (31). We start by considering padding zero $= k-1$ and stride $= 1$, and we will arrive to the formula with padding zero $= P$ and stride $= S$ later. Using (11), we denote by

$$a = \text{conv}(h, g, \text{padding zero} = k-1, \text{stride} = 1) \in \mathbb{R}^{2k-1}. \tag{37}$$

We have from (12), for $j \in [\![0, 2k-2]\!]$,

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

Using (13) 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

$$\begin{cases} a_j = \sum_{i=k-1-j}^{k-1} h_i g_{i+j-(k-1)} & \text{if} \quad j \in [\![0, k-2]\!] \ , \\ a_j = \sum_{i=0}^{2k-2-j} h_i g_{i+j-(k-1)} & \text{if} \quad j \in [\![k-1, 2k-2]\!] \ . \end{cases} \tag{38}$$

In the sequel, 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 (36) and (34), the equality (35) 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 (38), we obtain

$$b_j = a_{k-1+j} \ . \tag{39}$$

If $j \in [\![-2r, -1]\!]$**:** then $[\![-r, r]\!] \cap [\![-r+j, r+j]\!] = [\![-r, r+j]\!]$. Using (36) and (34), the equality (35) 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 (38), we obtain

$$b_j = a_{k-1+j} \ . \tag{40}$$

If $j \in [\![2r+1, SN-2r-1]\!]$**:** then $[\![-r, r]\!] \cap [\![-r+j, r+j]\!] = \emptyset$. The equality (35) becomes

$$b_j = 0. \tag{41}$$

Therefore, we summarize (39), (40) and (41): For all $j \in [\![-(k-1), -(k-1)+SN-1]\!]$,

$$b_j = \begin{cases} a_{k-1+j} & \text{if } j \in [\![-(k-1), k-1]\!], \\ 0 & \text{if } j \in [\![k, SN-k]\!]. \end{cases} \tag{42}$$

Let us now introduce 'padding zero $= P$' and 'stride $= S$'. We will prove the equality between matrices in (31) using the equality between vectors in (42).

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 (37), (14) and (42), we have

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

Therefore, using (15) 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 (30), 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 (41), and since $\left\lfloor \frac{k-1}{S} \right\rfloor S$ 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 (33), (32) and Lemma 6, we conclude that

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

$\square$

Lemma 8. *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 $z_{P/S} = \begin{bmatrix} 0_{P/S} \\ 1 \\ 0_{P/S} \end{bmatrix} \in \mathbb{R}^{2P/S+1}$. We have*

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

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

$$x_{m,l} = \sum_{c=1}^{C} \text{conv}(\mathbf{K}_{m,c}, \mathbf{K}_{l,c}, \textit{padding zero} = P, \textit{stride} = S) - \delta_{m=l} z_{P/S} \in \mathbb{R}^{2P/S+1} \ . \tag{43}$$

Proof. We have from (28),

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

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 :

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

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 (43). Using (7), (20), and (30), 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 7, 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

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

$\square$

Proof of Theorem 2. 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 8, 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 (20) and (30), 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 (17) and (16), we obtain for any M, C, S, $k = 2r+1$ and $\mathbf{K} \in \mathbb{R}^{M\times C\times k}$,

$$\|\mathcal{K}\mathcal{K}^T - Id_{MN}\|_F^2 = N\| \,\mathbf{conv}(\mathbf{K},\mathbf{K}, \text{padding zero} = P, \text{stride} = S) - I_{r0}\|_F^2 \ . \tag{44}$$

This concludes the proof in the RO case.

In order to prove the theorem in the CO case we use Lemma 1 in [44]. 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 (44) holds for all M, C and S, we have

$$\|\mathcal{K}^T\mathcal{K} - Id_{CSN}\|_F^2 = N\left(\| \,\mathbf{conv}(\mathbf{K},\mathbf{K}, \text{padding zero} = P, \text{stride} = S) - I_{r0}\|_F^2 - (M - CS)\right) \tag{45}$$

Hence, using the definitions of err_N^F and L_{orth} in Sections A.2.2 and A.2.3, (44) and (45) lead to

$$\left(\mathrm{err}_N^F(\mathbf{K})\right)^2 = NL_{orth}(\mathbf{K}).$$

This concludes the proof of Theorem 2 in the 1D case. $\square$

E.2 Sketch of the proof of Theorem 2, 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 $\mathrm{Vect}(X) = x$. The main steps of the proof in the 2D case follow those in the 1D case and are given below.

Lemma 9. *Let $X \in \mathbb{R}^{SN\times SN}$. We have*

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

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 10. *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*

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

Lemma 11. *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 $m, l \in [\![1, M]\!]$,

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

Then we proceed as in the 1D case.

F Proof of Theorem 3

F.1 Proof of Theorem 3, in the 1D case

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 8, we have

$$\mathcal{K}\mathcal{K}^T - Id_{MN} = \begin{pmatrix} C(Q_{S,N}(x_{1,1})) & \dots & C(Q_{S,N}(x_{1,M})) \\ \vdots & \ddots & \vdots \\ C(Q_{S,N}(x_{M,1})) & \dots & C(Q_{S,N}(x_{M,M})) \end{pmatrix} , \tag{46}$$

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

$$x_{m,l} = \sum_{c=1}^{C} \mathrm{conv}(\mathbf{K}_{m,c}, \mathbf{K}_{l,c}, \text{padding zero} = P, \text{stride} = S) - \delta_{m=l} z_{P/S} \in \mathbb{R}^{2P/S+1} . \tag{47}$$

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 (46), and (20), 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 (30), 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 (47), then (17) and (16), we obtain

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

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

$$R_n(x) = (x_{k-1}, \dots, x_{2k-2}, 0, \dots, 0, x_0, \dots, x_{k-2})^T \in \mathbb{R}^n \ . \tag{48}$$

Note that, when $S' = 1$, $N' = SN$, we have in (30), $P' = k-1$ and

$$Q_{1,SN} = R_{SN}. \tag{49}$$

Recall from (7) 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 (20), we have for all $x \in \mathbb{R}^{SN}$,

$$[C(x)]_{:,j} = \Lambda_j x. \tag{50}$$

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

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

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

$$P_{SN}^{(j)}(x) = \Lambda_j P_{SN}(x) \tag{51}$$

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

$$R_{SN}^{(j)}(x) = \Lambda_j R_{SN}(x). \tag{52}$$

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 $x \in \mathbb{R}^{2k-1}$,

$$\begin{cases} \|R_{SN}^{(j)}(x)\|_1 = \|x\|_1, \\ \|R_{SN}^{(j)}(x)\|_2 = \|x\|_2. \end{cases} \tag{53}$$

We first start by introducing the following Lemma.

Lemma 12. *Let $h, g \in \mathbb{R}^k$. There exist S vectors $x_0, \dots, x_{S-1} \in \mathbb{R}^{2k-1}$ 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 (18) and (19), 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 (50) and (51) 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 (26), 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 (50) and (51), 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 (22) and (25)), 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 7 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 (49), 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 (50) and (52), 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 (28), we have that the block $(c,c') \in [\![1,C]\!]^2$ of size (SN,SN) of $\mathcal{K}^T\mathcal{K}$ is equal to :

$$\begin{pmatrix} C(P_{SN}(\mathbf{K}_{1,c}))^T S_N^T & \dots & C(P_{SN}(\mathbf{K}_{M,c}))^T S_N^T \end{pmatrix} \begin{pmatrix} S_N C(P_{SN}(\mathbf{K}_{1,c'})) \\ \vdots \\ S_N C(P_{SN}(\mathbf{K}_{M,c'})) \end{pmatrix}$$
$$= \sum_{m=1}^{M} C(P_{SN}(\mathbf{K}_{m,c}))^T S_N^T S_N C(P_{SN}(\mathbf{K}_{m,c'})) \ . \tag{54}$$

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 12 with $h = \mathbf{K}_{m,c}$, and $g = \mathbf{K}_{m,c'}$. Hence, we have, for all $j \in [\![0, SN-1]\!]$,

$$[C(P_{SN}(\mathbf{K}_{m,c}))^T S_N^T S_N C(P_{SN}(\mathbf{K}_{m,c'}))]_{:,j} = R_{SN}^{(j)}(x_{m,c,c',j\%S}) \ . \tag{55}$$

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

$$x_{c,c',s} = \sum_{m=1}^{M} x_{m,c,c',s} - \delta_{c=c'} \overline{f}_{k-1} \in \mathbb{R}^{2k-1}. \tag{56}$$

Note that, from (7), (48), and (52), 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 (54), (55), and (56), we have, for all $j \in [\![0, SN-1]\!]$,

$$\begin{aligned} \left[A_{c,c'}^N\right]_{:,j} &= \left[\sum_{m=1}^{M} C(P_{SN}(\mathbf{K}_{m,c}))^T S_N^T S_N C(P_{SN}(\mathbf{K}_{m,c'})) - \delta_{c=c'} Id_{SN}\right]_{:,j} \\ &= \sum_{m=1}^{M} R_{SN}^{(j)}(x_{m,c,c',j\%S}) - \delta_{c=c'} R_{SN}^{(j)}(\overline{f}_{k-1}) \\ &= R_{SN}^{(j)}(x_{c,c',j\%S}) \ . \end{aligned} \tag{57}$$

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

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

Using (57) and (53), this becomes

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

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

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

Using (57) 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 2 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

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

This concludes the proof in the 1D case.

F.2 Sketch of the proof of Theorem 3, 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 12, then we proceed as in the 1D case.

G Proof of Proposition 1

Below, we prove Proposition 1 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.

H 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 *DEEL.LIP*[9] library.

H.1 Computing the singular values of $\mathcal{K}$ when $S = 1$

For convolutional layers without stride, $S = 1$, we use the algorithm described in [35]. We describe the algorithm for 2D convolutional layers in Algorithm 1. The algorithm provides the full list of singular values.

Algorithm 1 Computing the list of singular values of $\mathcal{K}$, when $S = 1$, [35].

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 $\mathcal{K}$: σ

transforms = FFT2(**K**, (N,N), axes=[0, 1])
σ = linalg.svd(transforms, compute_uv=False)

H.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 and a spectral shift to compute the smallest and the largest singular value ($\sigma_{min}, \sigma_{max}$) of $\mathcal{K}$. We give the principle of the algorithm in Algorithm 2. For clarity, in Algorithm 2, we assume a function 'σ = power_iteration(M)', that applies the power iteration algorithm to a square matrix M and returns its largest singular value $\sigma \geq 0$. 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 *DEEL.LIP*[10] library.

[9] https://github.com/deel-ai/deel-lip
[10] https://github.com/deel-ai/deel-lip

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 $\mathcal{K}$: $(\sigma_{min}, \sigma_{max})$

if $CS^2 \geq M$ **then**
  # RO case
  σ_{max} = sqrt(power_iteration($\mathcal{K}\mathcal{K}^T$))
  $\lambda = 1.1 * \sigma_{max} * \sigma_{max}$
  σ_{min} = sqrt(λ - power_iteration($\lambda \,\mathrm{Id}_{MN^2} - \mathcal{K}\mathcal{K}^T$))

else
  # CO case
  σ_{max} = sqrt(power_iteration($\mathcal{K}^T\mathcal{K}$))
  $\lambda = 1.1 * \sigma_{max} * \sigma_{max}$
  σ_{min} = sqrt(λ - power_iteration($\lambda \,\mathrm{Id}_{CS^2N^2} - \mathcal{K}^T\mathcal{K}$))

end if