Parseval Proximal Neural Networks

Marzieh Hasannasab*   Johannes Hertrich*
Sebastian Neumayer*   Gerlind Plonka†   Simon Setzer‡
Gabriele Steidl*

December 24, 2019

Abstract

The aim of this paper is twofold. First, we show that a certain concatenation of a proximity operator with an affine operator is again a proximity operator on a suitable Hilbert space. Second, we use our findings to establish so-called proximal neural networks (PNNs) and stable Parseval (frame) proximal neural networks (PPNNs).

Let $\mathcal{H}$ and $\mathcal{K}$ be real Hilbert spaces, $b \in \mathcal{K}$ and $T \in \mathcal{B}(\mathcal{H}, \mathcal{K})$ a linear operator with closed range and Moore-Penrose inverse $T^\dagger$. Based on the well-known characterization of proximity operators by Moreau, we prove that for any proximity operator $\mathrm{Prox} \colon \mathcal{K} \to \mathcal{K}$ the operator $T^\dagger \mathrm{Prox}(T \cdot + b)$ is a proximity operator on $\mathcal{H}$ equipped with a suitable norm. In particular, it follows for the frequently applied soft shrinkage operator $\mathrm{Prox} = S_\lambda \colon \ell_2 \to \ell_2$ and any frame analysis operator $T \colon \mathcal{H} \to \ell_2$, that the frame shrinkage operator $T^\dagger S_\lambda T$ is a proximity operator in a suitable Hilbert space.

Further, the concatenation of proximity operators on $\mathbb{R}^d$ equipped with different norms establishes a PNN. If the network arises from Parseval frame analysis or synthesis operators, it forms an averaged operator, called PPNN. The involved linear operators, respectively their transposed operators, are in a Stiefel manifold, so that minimization methods on Stiefel manifolds can be applied for training such networks. Finally, some proof-of-the concept examples demonstrate the performance of PPNNs.

1. Introduction

Wavelet and frame shrinkage operators became very popular in recent years. A certain starting point was the iterative shrinkage-thresholding algorithm (ISTA) in [15] which was interpreted as a special case of the forward-backward algorithm in [13]. For relations

[1]Department of Mathematics, Technische Universität Kaiserslautern, Paul-Ehrlich-Str. 31, D-67663 Kaiserslautern, Germany, {name}@mathematik.uni-kl.de.
[2]Institute for Numerical and Applied Mathematics, Göttingen University, Lotzestr. 16-18, 37083 Göttingen, Germany, {name}@math.uni-goettingen.de.
[3]simon.setzer@gmail.com.

with other algorithms see also [7, 33]. Let $T \in \mathbb{R}^{n \times d}$, $n \geq d$, have full column rank. Then, the problem

$$\operatorname*{argmin}_{y \in \mathbb{R}^d} \left\{ \tfrac{1}{2} \|x - y\|_2^2 + \lambda \|Ty\|_1 \right\}, \quad \lambda > 0 \tag{1}$$

is known as analysis point of view. For orthogonal $T \in \mathbb{R}^{d \times d}$, the solution of (1) is given by the frame soft shrinkage operator $T^\dagger S_\lambda T = T^* S_\lambda T$. If $T \in \mathbb{R}^{n \times d}$ with $n \leq d$ and $TT^* = I_n$, the solution of problem (1) is given by $I_d - T^*T + T^*S_\lambda T$, see [5, Theorem 6.15]. For arbitrary $T \in \mathbb{R}^{n \times d}$, $n \geq d$, there are no analytic expressions for the solution of (1) in the literature.

The question, whether the frame shrinkage operator can itself be seen as a proximity operator, has been recently studied in [19]. They showed that the set-valued operator $(T^\dagger S_\lambda T)^{-1} - I_d$ is maximally cyclically monotone, which implies that it is a proximity operator with respect to some norm in $\mathbb{R}^d$. In this paper, we prove that for any operator $T \in \mathcal{B}(\mathcal{H}, \mathcal{K})$ with closed range, $b \in \mathcal{K}$ and any proximity operator $\operatorname{Prox} \colon \mathcal{K} \to \mathcal{K}$, the new operator $T^\dagger \operatorname{Prox}(T \cdot + b) \colon \mathcal{H} \to \mathcal{H}$ is also a proximity operator on the linear space $\mathcal{H}$, but equipped with another inner product. The above mentioned finite dimensional setting is included as a special case. In contrast to [19], we directly approach the problem using a classical result of Moreau [28]. Moreover, we provide the function for the definition of the proximity operator. Here, we like to mention that this function can be also deduced from Proposition 3.9 in [11]. However, since this deduction appears to be more space consuming than the direct proof of our Theorem 3.4, we prefer to give a direct approach.

There are several fields where our results may be of interest. Different norms in the definition of the proximity operator were successfully used in variable metric algorithms, see [9]. Our findings may be also of interest in connection with so-called Plug-and-Play algorithms [8, 34, 36].

Recently, it was shown that many activation functions appearing in neural networks are indeed proximity functions [12]. Based on this observations and our previous findings, we consider neural networks which are concatenations of proximity operators and call them proximal neural networks. Due to stability reasons, PNNs related to linear operators from Stiefel manifolds are of special interest. They form so-called averaged operators and are consequently nonexpansive. We refer to these networks as Parseval (frame) proximal neural networks (PPNNs). Orthogonal matrices have already shown advantages in training recurrent neural networks (RNNs) [2, 3, 24, 26, 37, 39]. Using orthogonal matrices, the vanishing or explosion of gradient in learning RNNs can be avoided [16]. The more general setting of learning rectangular matrices from a Stiefel manifold was proposed, e.g., in [22], but with a different focus than in this paper. The most relevant paper with respect to our setting is [23], where the authors considered the so-called optimization over multiple dependent Stiefel manifolds (OMDSM). We will see that the NNs in [23] are special cases of our PNNs so that our analysis ensures that they are averaged operators.

Our paper is organized as follows: We begin with preliminaries on convex analysis in Hilbert spaces in Section 2. In Section 3, we prove our general results on the interplay between proximity and certain affine operators. As a special case we emphasize that the frame soft shrinkage operator is itself a proximity operator in Section 4. In Section 5,

we use our findings to set up neural networks as a concatenation of proximity operators on $\mathbb{R}^d$ equipped with different norms related to linear operators. If these operators or their transposed operators are in a Stiefel manifold, which is equivalent to the property that their columns or rows form a Parseval frame in the Euclidean space, then our proposed network is actually an averaged operator. Section 6 deals with the training of PPNNs by stochastic gradient descent on Stiefel manifolds. In Section 7, we provide first numerical examples. Finally, Section 8 contains conclusions and addresses further research questions.

2. Preliminaries

Let $\mathcal{H}$ be a real Hilbert space with inner product $\langle \cdot, \cdot \rangle$ and norm $\| \cdot \|$. By $\Gamma_0(\mathcal{H})$ we denote the set of proper, convex, lower semi-continuous functions on $\mathcal{H}$ mapping into $(-\infty, \infty]$. For $f \in \Gamma_0(\mathcal{H})$ and $\lambda > 0$, the *proximity operator* $\mathrm{prox}_{\lambda f} \colon \mathcal{H} \to \mathcal{H}$ and its *Moreau envelope* $M_{\lambda f} \colon \mathcal{H} \to \mathbb{R}$ are defined by

$$\mathrm{prox}_{\lambda f}(x) := \operatorname*{argmin}_{y \in \mathcal{H}} \big\{ \tfrac{1}{2} \|x - y\|^2 + \lambda f(y) \big\},$$

$$M_{\lambda f}(x) := \min_{y \in \mathcal{H}} \big\{ \tfrac{1}{2} \|x - y\|^2 + \lambda f(y) \big\}.$$

Clearly, the proximity operator and its Moreau envelope depend on the underlying space $\mathcal{H}$, in particular on the chosen inner product. Recall that an operator $A \colon \mathcal{H} \to \mathcal{H}$ is called *firmly nonexpansive* if for all $x, y \in \mathcal{H}$ the following relation is fulfilled

$$\|Ax - Ay\|^2 \le \langle x - y, Ax - Ay \rangle.$$

Obviously, firmly nonexpansive operators are nonexpansive.

For a Fréchet differentiable function $\Phi \colon \mathcal{H} \to \mathbb{R}$, the gradient $\nabla \Phi(x)$ at $x \in \mathcal{H}$ is defined as the vector satisfying for all $h \in \mathcal{H}$,

$$\langle \nabla \Phi(x), h \rangle = D\Phi(x)h,$$

where $D\Phi \colon \mathcal{H} \to \mathcal{B}(\mathcal{H}, \mathbb{R})$ denotes the Fréchet derivative of Φ, i.e., for all $x, h \in \mathcal{H}$,

$$\Phi(x + h) - \Phi(x) = D\Phi(x)h + o(\|h\|).$$

Note that the gradient crucially depends on the chosen inner product in $\mathcal{H}$. The following results can be found, e.g., in [4, Props. 12.27, 12.29].

Theorem 2.1. *Let $f \in \Gamma_0(\mathcal{H})$. Then the following relations hold true:*
i) The operator $\mathrm{prox}_{\lambda f} \colon \mathcal{H} \to \mathcal{H}$ is firmly nonexpansive.
ii) The function $M_{\lambda f}$ is (Fréchet) differentiable with Lipschitz-continuous gradient given by

$$\nabla M_{\lambda f}(x) = x - \mathrm{prox}_{\lambda f}(x).$$

Clearly, ii) implies that

$$\text{prox}_{\lambda f}(x) = \nabla\left(\tfrac{1}{2}\|x\|^2 - M_{\lambda f}(x)\right) = \nabla\Phi(x), \tag{2}$$

where $\Phi := \tfrac{1}{2}\|\cdot\|^2 - M_{\lambda f}$ is convex since $\text{prox}_{\lambda f}$ is nonexpansive [4, Prop. 17.10]. Further, it was shown by Moreau that also the following (reverse) statement holds true [28, Cor. 10c].

Theorem 2.2. *The operator* $\text{Prox}\colon \mathcal{H} \to \mathcal{H}$ *is a proximity operator if and only if it is nonexpansive and there exists a function* $\Psi \in \Gamma_0(\mathcal{H})$, *such that for any* $x \in \mathcal{H}$ *we have* $\text{Prox}(x) \in \partial\Psi(x)$, *where* $\partial\Psi$ *denotes the subdifferential of* Ψ.

Thanks to (2), we conclude that $\text{Prox}\colon \mathcal{H} \to \mathcal{H}$ is a proximity operator if and only if it is nonexpansive and the gradient of a convex, differentiable function $\Phi\colon \mathcal{H} \to \mathbb{R}$. Note that recently, the characterization of Bregman proximity operators in a more general setting was discussed in [21]. In the following example, we recall the Moreau envelope and the proximity operator related to the soft thresholding operator.

Example 2.3. *Let* $\mathcal{H} = \mathbb{R}$ *with usual norm* $|\cdot|$ *and* $f(x) := |x|$. *Then,* $\text{prox}_{\lambda f}$ *is the soft shrinkage operator* S_λ *defined by*

$$S_\lambda(x) := \begin{cases} x - \lambda & \text{for } x > \lambda, \\ 0 & \text{for } x \in [-\lambda, \lambda], \\ x + \lambda & \text{for } x < -\lambda, \end{cases}$$

and the Moreau envelope is the Huber function

$$m_{\lambda|\cdot|}(x) = \begin{cases} \lambda x - \frac{\lambda^2}{2} & \text{for } x > \lambda, \\ \frac{1}{2}x^2 & \text{for } x \in [-\lambda, \lambda], \\ -\lambda x - \frac{\lambda^2}{2} & \text{for } x < -\lambda. \end{cases}$$

Hence, $\text{prox}_{\lambda f} = \nabla\varphi$, *where* $\varphi(x) = \frac{x^2}{2} - m_{\lambda|\cdot|}(x)$, *i.e.,*

$$\varphi(x) = \begin{cases} \frac{1}{2}(x - \lambda)^2 & \text{for } x > \lambda, \\ 0 & \text{for } x \in [-\lambda, \lambda], \\ \frac{1}{2}(x + \lambda)^2 & \text{for } x < -\lambda. \end{cases}$$

For $\mathcal{H} = \mathbb{R}^d$ *and* $f(x) := \|x\|_1$, *we can just use a componentwise approach. Then* S_λ *is defined componentwise, the Moreau envelope reads as* $M_{\lambda\|\cdot\|_1}(x) = \sum_{i=1}^d m_{\lambda|\cdot|}(x_i)$ *and the potential of* $\text{prox}_{\lambda\|\cdot\|_1}$ *is* $\Phi(x) = \sum_{i=1}^d \varphi(x_i)$.

3. Interplay Between Proximity and Linear Operators

Let $\mathcal{H}$ and $\mathcal{K}$ be real Hilbert spaces with inner products $\langle\cdot,\cdot\rangle_{\mathcal{H}}$ and $\langle\cdot,\cdot\rangle_{\mathcal{K}}$ and corresponding norms $\|\cdot\|_{\mathcal{H}}$ and $\|\cdot\|_{\mathcal{K}}$, respectively. By $\mathcal{B}(\mathcal{H}, \mathcal{K})$ we denote the space of bounded,

linear operators from $\mathcal{H}$ to $\mathcal{K}$. The kernel and the range of T are denoted by $\mathcal{N}(T)$ and $\mathcal{R}(T)$, respectively. In this section, we show that for any operator $T \in \mathcal{B}(\mathcal{H}, \mathcal{K})$ with closed range $\mathcal{R}(T)$, $b \in \mathcal{K}$ and proximity operator $\text{Prox} \colon \mathcal{K} \to \mathcal{K}$, the operator $T^\dagger \text{Prox}(T \cdot + b) \colon \mathcal{H} \to \mathcal{H}$ is itself a proximity operator on the linear space $\mathcal{H}$ equipped with a suitable (equivalent) norm $\| \cdot \|_{\mathcal{H}_T}$, i.e., there exits a function $f \in \Gamma_0(\mathcal{H})$ such that

$$T^\dagger \text{Prox}(Tx + b) = \operatorname*{argmin}_{y \in \mathcal{H}} \big\{ \tfrac{1}{2} \|x - y\|_{\mathcal{H}_T}^2 + f(y) \big\}.$$

Throughout this section, let $T \in \mathcal{B}(\mathcal{H}, \mathcal{K})$ have closed range. Then, the same holds true for its adjoint T^* and the following (orthogonal) decompositions hold

$$\mathcal{K} = \mathcal{R}(T) \oplus \mathcal{N}(T^*), \qquad \mathcal{H} = \mathcal{R}(T^*) \oplus \mathcal{N}(T). \tag{3}$$

The Moore-Penrose inverse (generalized inverse, pseudo-inverse) $T^\dagger \in \mathcal{B}(\mathcal{K}, \mathcal{H})$ is given point-wise by

$$\{T^\dagger y\} = \{x \in \mathcal{H} : T^* T x = T^* y\} \cap \mathcal{R}(T^*),$$

see [4]. Further, it satisfies $\mathcal{R}(T^\dagger) = \mathcal{R}(T^*)$ and

$$T^\dagger T = P_{\mathcal{R}(T^*)}, \quad T T^\dagger = P_{\mathcal{R}(T)}, \tag{4}$$

where P_C is the orthogonal projection onto the closed, convex set C, see [4, Prop. 3.28]. Then, it follows

$$T^\dagger T T^* = P_{\mathcal{R}(T^*)} T^* = T^* \quad \text{and} \quad T^\dagger P_{\mathcal{R}(T)} = T^\dagger T T^\dagger = T^\dagger. \tag{5}$$

If T is injective, then $T^\dagger = (T^* T)^{-1} T^*$ and if T is surjective, we have $T^\dagger = T^* (T T^*)^{-1}$.

Every $T \in \mathcal{B}(\mathcal{H}, \mathcal{K})$ gives rise to an inner product in $\mathcal{H}$ via

$$\langle x, y \rangle_{\mathcal{H}_T} = \langle Tx, Ty \rangle_{\mathcal{K}} + \langle P_{\mathcal{N}(T)} x, P_{\mathcal{N}(T)} y \rangle_{\mathcal{H}} \tag{6}$$

with corresponding norm

$$\|x\|_{\mathcal{H}_T} = \big(\|Tx\|_{\mathcal{K}}^2 + \|P_{\mathcal{N}(T)} x\|_{\mathcal{H}}^2 \big)^{\frac{1}{2}}.$$

If T is injective, the second summand vanishes. In general, this norm only induces a pre-Hilbert structure. Since $T \in \mathcal{B}(\mathcal{H}, \mathcal{K})$ has closed range, the norms $\| \cdot \|_{\mathcal{H}}$ and $\| \cdot \|_{\mathcal{H}_T}$ are equivalent on $\mathcal{H}$ due to

$$\|x\|_{\mathcal{H}_T}^2 = \|Tx\|_{\mathcal{K}}^2 + \|P_{\mathcal{N}(T)} x\|_{\mathcal{H}}^2 \le \big(\|T\|_{\mathcal{B}(\mathcal{H}, \mathcal{K})}^2 + 1 \big) \|x\|_{\mathcal{H}}^2 \tag{7}$$

and

$$\|x\|_{\mathcal{H}}^2 = \|T^\dagger Tx\|_{\mathcal{H}}^2 + \|P_{\mathcal{N}(T)} x\|_{\mathcal{H}}^2 \le \big(\|T^\dagger\|_{\mathcal{B}(\mathcal{K}, \mathcal{H})}^2 + 1 \big) \|x\|_{\mathcal{H}_T}^2$$

for all $x \in \mathcal{H}$. The norm equivalence also ensures the completeness of $\mathcal{H}$ equipped with the new norm. To emphasize that we consider the linear space $\mathcal{H}$ with this norm, we write $\mathcal{H}_T$. For special $T \in \mathcal{B}(\mathcal{H}, \mathcal{K})$, the inner product (7) coincides with the one in $\mathcal{H}$.

Lemma 3.1. *Let $T \in \mathcal{B}(\mathcal{H}, \mathcal{K})$ fulfill $T^*T = \mathrm{Id}_{\mathcal{H}}$ or $TT^* = \mathrm{Id}_{\mathcal{K}}$, where $\mathrm{Id}_{\mathcal{H}}$ and $\mathrm{Id}_{\mathcal{K}}$ denote the identity operator on $\mathcal{H}$ and $\mathcal{K}$, respectively. Then, the inner product (6) coincides with the one in $\mathcal{H}$ and consequently $\mathcal{H} = \mathcal{H}_T$.*

Proof. If $T^*T = \mathrm{Id}_{\mathcal{H}}$, then T is injective such that (6) implies

$$\langle x, y \rangle_{\mathcal{H}_T} = \langle Tx, Ty \rangle_{\mathcal{K}} = \langle x, T^*Ty \rangle_{\mathcal{H}} = \langle x, y \rangle_{\mathcal{H}}.$$

If $TT^* = \mathrm{Id}_{\mathcal{K}}$, then $P_{\mathcal{N}(T)} = \mathrm{Id}_{\mathcal{H}} - T^{\dagger}T = \mathrm{Id}_{\mathcal{H}} - T^*T$ and

$$
\begin{aligned}
\langle x, y \rangle_{\mathcal{H}_T} &= \langle Tx, Ty \rangle_{\mathcal{K}} + \langle x - T^*Tx, y - T^*Ty \rangle_{\mathcal{H}} \\
&= \langle x, T^*Ty \rangle_{\mathcal{H}} + \langle x, y \rangle_{\mathcal{H}} - \langle x, T^*Ty \rangle_{\mathcal{H}} - \langle T^*Tx, y \rangle_{\mathcal{H}} + \langle T^*Tx, T^*Ty \rangle_{\mathcal{H}} \\
&= \langle x, y \rangle_{\mathcal{H}}.
\end{aligned}
$$

$\square$

To apply the characterization of proximal mappings in $\mathcal{H}_T$ by Moreau, see Theorem 2.2, we have to compute gradients in $\mathcal{H}_T$. Here, the following result is crucial.

Lemma 3.2. *Let $\mathcal{H}$ and $\mathcal{K}$ be real Hilbert spaces with inner products $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ and $\langle \cdot, \cdot \rangle_{\mathcal{K}}$, respectively. For an operator $T \in \mathcal{B}(\mathcal{H}, \mathcal{K})$ with closed range, let $\mathcal{H}_T$ be the Hilbert space with inner product (6). For (Fréchet) differentiable $\Phi \colon \mathcal{H} \to \mathbb{R}$, the gradients $\nabla_{\mathcal{H}}\Phi$ and $\nabla_{\mathcal{H}_T}\Phi$ with respect to the different inner products are related by*

$$(T^*T + P_{\mathcal{N}(T)})\nabla_{\mathcal{H}_T}\Phi(x) = \nabla_{\mathcal{H}}\Phi(x).$$

Proof. The gradient $\nabla_{\mathcal{H}_T}\Phi(x)$ at $x \in \mathcal{H}$ in the space $\mathcal{H}_T$ is given by the vector satisfying

$$\langle \nabla_{\mathcal{H}_T}\Phi(x), h \rangle_{\mathcal{H}_T} = D\Phi(x)h = \langle \nabla_{\mathcal{H}}\Phi(x), h \rangle_{\mathcal{H}}$$

for all $h \in \mathcal{H}$. Since

$$
\begin{aligned}
\langle \nabla_{\mathcal{H}_T}\Phi(x), h \rangle_{\mathcal{H}_T} &= \langle T\nabla_{\mathcal{H}_T}\Phi(x), Th \rangle_{\mathcal{K}} + \langle P_{\mathcal{N}(T)}\nabla_{\mathcal{H}_T}\Phi(x), P_{\mathcal{N}(T)}h \rangle_{\mathcal{H}} \\
&= \langle T^*T\nabla_{\mathcal{H}_T}\Phi(x), h \rangle_{\mathcal{H}} + \langle P_{\mathcal{N}(T)}\nabla_{\mathcal{H}_T}\Phi(x), h \rangle_{\mathcal{H}},
\end{aligned}
$$

the gradient depends on the chosen inner product through

$$(T^*T + P_{\mathcal{N}(T)})\nabla_{\mathcal{H}_T}\Phi(x) = \nabla_{\mathcal{H}}\Phi(x).$$

$\square$

Now, the desired result follows from the next theorem.

Theorem 3.3. *Let $b \in \mathcal{K}$, $T \in \mathcal{B}(\mathcal{H}, \mathcal{K})$ have closed range and $\mathrm{Prox} \colon \mathcal{K} \to \mathcal{K}$ be a proximity operator on $\mathcal{K}$. Then, the operator $A := T^{\dagger}\,\mathrm{Prox}\,(T \cdot + b) \colon \mathcal{H}_T \to \mathcal{H}_T$ is a proximity operator.*

Proof. In view of Theorems 2.1 and 2.2, it suffices to show that A is nonexpansive and that there exists a convex function $\Psi\colon \mathcal{H}_T \to \mathbb{R}$ with $A = \nabla_{\mathcal{H}_T}\Psi$.

1. First, we show that A is firmly nonexpansive, and thus nonexpansive. By (3), we see that

$$P_{\mathcal{N}(T)}T^\dagger = 0. \tag{8}$$

Using this and (4), it follows

$$\|Ax - Ay\|_{\mathcal{H}_T}^2 = \|TT^\dagger \left(\operatorname{Prox}(Tx+b) - \operatorname{Prox}(Ty+b)\right)\|_{\mathcal{K}}^2 + \|P_{\mathcal{N}(T)}(Ax - Ay)\|_{\mathcal{H}}^2$$
$$\leq \|\operatorname{Prox}(Tx+b) - \operatorname{Prox}(Ty+b)\|_{\mathcal{K}}^2. \tag{9}$$

By (8) and (4), we obtain

$$\langle Ax - Ay, x - y\rangle_{\mathcal{H}_T} = \langle TT^\dagger\left(\operatorname{Prox}(Tx+b) - \operatorname{Prox}(Ty+b)\right), Tx - Ty\rangle_{\mathcal{K}}$$
$$= \langle P_{\mathcal{R}(T)}\left(\operatorname{Prox}(Tx+b) - \operatorname{Prox}(Ty+b)\right), Tx - Ty\rangle_{\mathcal{K}}$$
$$= \langle \operatorname{Prox}(Tx+b) - \operatorname{Prox}(Ty+b), Tx - Ty\rangle_{\mathcal{K}},$$
$$= \langle \operatorname{Prox}(Tx+b) - \operatorname{Prox}(Ty+b), Tx + b - (Ty+b)\rangle_{\mathcal{K}},$$

and since Prox is firmly nonexpansive with respect to $\|\cdot\|_{\mathcal{K}}$, the estimate (9) further implies that A is firmly nonexpansive

$$\langle Ax - Ay, x - y\rangle_{\mathcal{H}_T} \geq \|\operatorname{Prox}(Tx+b) - \operatorname{Prox}(Ty+b)\|_{\mathcal{K}}^2 \geq \|Ax - Ay\|_{\mathcal{H}_T}^2.$$

2. It remains to prove that there exists a convex function $\Psi\colon \mathcal{H}_T \to \mathbb{R}$ with $\nabla_{\mathcal{H}_T}\Psi = A$. Since Prox is a proximity operator, there exists $\Phi\colon \mathcal{H} \to \mathbb{R}$ with $\operatorname{Prox} = \nabla_{\mathcal{K}}\Phi$. Then, a natural candidate is given by $\Psi = \Phi(T \cdot + b)$. Using the definition of the gradient and the chain rule, it holds for all $x, h \in \mathcal{H}$ that

$$\langle \nabla_{\mathcal{H}}\Psi(x), h\rangle_{\mathcal{H}} = D\Psi(x)h = D\Phi(Tx+b)\,Th = \langle \nabla_{\mathcal{K}}\Phi(Tx+b), Th\rangle_{\mathcal{K}}$$
$$= \langle T^*\operatorname{Prox}(Tx+b), h\rangle_{\mathcal{H}}.$$

Incorporating Lemma 3.2, we conclude

$$(T^*T + P_{\mathcal{N}(T)})\nabla_{\mathcal{H}_T}\Psi = \nabla_{\mathcal{H}}\Psi(x) = T^*\operatorname{Prox}(Tx+b)$$

which implies $T^*T\,\nabla_{\mathcal{H}_T}\Psi = T^*\operatorname{Prox}(Tx+b)$ and $\nabla_{\mathcal{H}_T}\Psi \in \mathcal{R}(T^*)$. By definition of $T^\dagger$, we obtain $\nabla_{\mathcal{H}_T}\Psi = A$. Finally, Ψ is convex since it is the concatenation of a convex function with a linear function. $\square$

Let

$$(f \square g)(x) := \inf_{y \in \mathcal{H}} f(y) + g(x - y)$$

denote the infimal convolution of $f, g \in \Gamma_0(\mathcal{H})$ and $x \mapsto \iota_S(x)$ the indicator function of the set S taking the value 0 if $x \in S$ and $+\infty$ otherwise.

For $\operatorname{Prox} := \operatorname{prox}_g$ with $g \in \Gamma_0(\mathcal{H})$, we are actually able to explicitly compute $f \in \Gamma_0(\mathcal{H})$ such that $T^\dagger \operatorname{Prox}(T \cdot + b) = \operatorname{prox}_f$ on $\mathcal{H}_T$. Clearly, this also gives an alternative proof for Theorem 3.3.

Theorem 3.4. *Let $b \in \mathcal{K}$, $T \in \mathcal{B}(\mathcal{H}, \mathcal{K})$ with closed range and $\mathrm{Prox} := \mathrm{prox}_g$ for some $g \in \Gamma_0(\mathcal{K})$. Then $T^\dagger \mathrm{prox}_g (T \cdot + b) \colon \mathcal{H}_T \to \mathcal{H}_T$ is the proximity operator on $\mathcal{H}_T$ of $f \in \Gamma_0(\mathcal{H})$ given by*

$$f(x) := g \square \big(\tfrac{1}{2} \| \cdot \|_{\mathcal{K}}^2 + \iota_{\mathcal{N}(T^*)} \big)(Tx + b) + \iota_{\mathcal{R}(T^*)}(x). \tag{10}$$

This expression simplifies to

$$f(x) = g \square \big(\tfrac{1}{2} \| \cdot \|_{\mathcal{K}}^2 + \iota_{\mathcal{N}(T^*)} \big)(Tx + b) \quad \text{if } T \text{ is injective,}$$
$$f(x) = g(Tx + b) + \iota_{\mathcal{R}(T^*)}(x) \quad \text{if } T \text{ is surjective,}$$
$$f(x) = g(Tx + b) \quad \text{if } T \text{ is bijective.}$$

Proof. By (5) and (3), we obtain

$$T^\dagger \mathrm{prox}_g (Tx + b)$$
$$= T^\dagger \operatorname*{argmin}_{z \in \mathcal{K}} \big\{ \tfrac{1}{2} \| z - Tx - b \|_{\mathcal{K}}^2 + g(z) \big\}$$
$$= T^\dagger P_{\mathcal{R}(T)} \operatorname*{argmin}_{z_1 \in \mathcal{R}(T), z_2 \in \mathcal{N}(T^*)} \big\{ \tfrac{1}{2} \| z_1 + z_2 - Tx \|_{\mathcal{K}}^2 + g(z_1 + z_2 + b) \big\}$$
$$= T^\dagger \operatorname*{argmin}_{z_1 \in \mathcal{R}(T)} \inf_{z_2 \in \mathcal{N}(T^*)} \big\{ \tfrac{1}{2} \| z_1 - Tx \|_{\mathcal{K}}^2 + \tfrac{1}{2} \| z_2 \|_{\mathcal{K}}^2 + g(z_1 + z_2 + b) \big\}$$
$$= T^\dagger \operatorname*{argmin}_{z_1 \in \mathcal{R}(T)} \big\{ \tfrac{1}{2} \| z_1 - Tx \|_{\mathcal{K}}^2 + \inf_{z_2 \in \mathcal{N}(T^*)} \big\{ \tfrac{1}{2} \| z_2 \|_{\mathcal{K}}^2 + g(z_1 + z_2 + b) \big\} \big\}$$
$$= T^\dagger T \operatorname*{argmin}_{y \in \mathcal{R}(T^*)} \big\{ \tfrac{1}{2} \| Ty - Tx \|_{\mathcal{K}}^2 + \inf_{z_2 \in \mathcal{N}(T^*)} \big\{ \tfrac{1}{2} \| z_2 \|_{\mathcal{K}}^2 + g(Ty + z_2 + b) \big\} \big\}$$

and by (4) further

$$T^\dagger \mathrm{prox}_g (Tx + b)$$
$$= \operatorname*{argmin}_{y \in \mathcal{R}(T^*)} \big\{ \tfrac{1}{2} \| Ty - Tx \|_{\mathcal{K}}^2 + \inf_{z_2 \in \mathcal{N}(T^*)} \big\{ \tfrac{1}{2} \| z_2 \|_{\mathcal{K}}^2 + g(Ty + z_2 + b) \big\} \big\}$$
$$= \operatorname*{argmin}_{y \in \mathcal{R}(T^*)} \big\{ \tfrac{1}{2} \| y - x \|_{\mathcal{H}_T}^2 + \inf_{z_2 \in \mathcal{N}(T^*)} \big\{ \tfrac{1}{2} \| z_2 \|_{\mathcal{K}}^2 + g(Ty + z_2 + b) \big\} \big\} \tag{11}$$
$$= \operatorname*{argmin}_{y \in \mathcal{H}} \big\{ \tfrac{1}{2} \| y - x \|_{\mathcal{H}_T}^2 + g \square \big(\tfrac{1}{2} \| \cdot \|_{\mathcal{K}}^2 + \iota_{\mathcal{N}(T^*)} \big)(Ty + b) + \iota_{\mathcal{R}(T^*)}(y) \big\}.$$

Hence, we conclude that $T^\dagger \mathrm{prox}_g (T \cdot + b)$ is the proximity operator on $\mathcal{H}_T$ of f in (10). $\quad\square$

Note that for injective T and $b = 0$, the function f is in general a weaker regularizer than g, i.e., $f \leq g$. This is necessary since for the latter (11) would lead to

$$\operatorname*{argmin}_{y \in \mathcal{R}(T^*)} \big\{ \tfrac{1}{2} \| x - y \|_{\mathcal{H}_T}^2 + g(Ty) \big\} = T^\dagger \operatorname*{argmin}_{z \in \mathcal{K}} \big\{ \tfrac{1}{2} \| z - Tx \|_{\mathcal{K}}^2 + g(z) + \iota_{\mathcal{R}(T)}(z) \big\} \neq T^\dagger \mathrm{prox}_g(Tx).$$

4. Frame Soft Shrinkage as Proximity Operator

In this section, we investigate the frame soft shrinkage as a special proximity operator. Let $\mathcal{K} = \ell_2$ be the Hilbert space of square summable sequences $c = \{c_k\}_{k \in \mathbb{N}}$ with norm $\|c\|_{\ell_2} := (\sum_{k \in \mathbb{N}} |c_k|^2)^{\frac{1}{2}}$ and assume that $\mathcal{H}$ is separable. A sequence $\{x_k\}_{k \in \mathbb{N}}$, $x_k \in \mathcal{H}$, is called a frame of $\mathcal{H}$, if constants $0 < A \leq B < \infty$ exist such that for all $x \in \mathcal{H}$,

$$A\|x\|_{\mathcal{H}}^2 \leq \sum_{k \in \mathbb{N}} |\langle x, x_k \rangle_{\mathcal{H}}|^2 \leq B\|x\|_{\mathcal{H}}^2.$$

Given a frame $\{x_k\}_{k \in \mathbb{N}}$ of $\mathcal{H}$, the corresponding analysis operator $T \colon \mathcal{H} \to \ell_2$ is defined as

$$Tx = \big\{\langle x, x_k \rangle_{\mathcal{H}}\big\}_{k \in \mathbb{N}}, \quad x \in \mathcal{H}.$$

Its adjoint $T^* \colon \ell_2 \to \mathcal{H}$ is the synthesis operator given by

$$T^*\{c_k\}_{k \in \mathbb{N}} = \sum_{k \in \mathbb{N}} c_k x_k, \quad \{c_k\}_{k \in \mathbb{N}} \in \ell_2.$$

By composing T and T^*, we obtain the frame operator

$$T^*Tx = \sum_{k \in \mathbb{N}} \langle x, x_k \rangle_{\mathcal{H}} x_k, \quad x \in \mathcal{H}.$$

The frame operator T^*T is invertible on $\mathcal{H}$, see [10], such that

$$x = \sum_{k \in \mathbb{N}} \langle x, x_k \rangle_{\mathcal{H}} (T^*T)^{-1} x_k, \quad x \in \mathcal{H}.$$

The sequence $\{(T^*T)^{-1}x_k\}_{k \in \mathbb{N}}$ is called the canonical dual frame of $\{x_k\}_{k \in \mathbb{N}}$. If $T^*T = \alpha \mathrm{Id}_{\mathcal{H}}$, $\alpha > 0$, then $\{x_k\}_{k \in \mathbb{N}}$ is called a tight frame, and for $\alpha = 1$ a Parseval frame. Here, Lemma 3.1 comes into the play. Note that $T^\dagger$ is indeed the synthesis operator for the canonical dual frame of $\{f_k\}_{k \in \mathbb{N}}$. The relation between linear, bounded, injective operators of closed range and frame analysis operators reads as follows:

Proposition 4.1. *i) An operator $T \in \mathcal{B}(\mathcal{H}, \ell_2)$ is injective and has closed range if and only if it is the analysis operator of some frame of $\mathcal{H}$.*

ii) An operator $T \in \mathcal{B}(\ell_2, \mathcal{H})$ is surjective if and only if it is the synthesis operator of some frame of $\mathcal{H}$.

Proof. *i*) If T is the analysis operator for a frame $\{x_k\}_{k \in \mathbb{N}}$, then T is bounded, injective and has closed range, see [10]. Conversely, assume that $T \in \mathcal{B}(\mathcal{H}, \ell_2)$ is injective and that $\mathcal{R}(T)$ is closed. By (3), it holds $\mathcal{R}(T^*) = \mathcal{H}$. Let $\{\delta_k\}_{k \in \mathbb{N}}$ be the canonical basis of ℓ_2 and set $\{x_k\}_{k \in \mathbb{N}} := \{T^*\delta_k\}_{k \in \mathbb{N}}$. Since $\sum_{k \in \mathbb{N}} |\langle x, x_k \rangle_{\mathcal{H}}|^2 = \|Tx\|_{\ell_2}^2$, we conclude that $\{x_k\}_{k \in \mathbb{N}}$ is a frame of $\mathcal{H}$ and that T is the corresponding analysis operator.
ii) Let $\{x_k\}_{k \in \mathbb{N}} = \{T\delta_k\}_{k \in \mathbb{N}}$. Then, the result follows from [10, Thm. 5.5.1]. $\square$

The soft shrinkage operator S_λ on ℓ_2 (applied componentwise) is the proximity operator corresponding to the function $g := \lambda \| \cdot \|_1$, $\lambda > 0$. As immediate consequence of Theorem 3.4 we obtain the following corollary.

Corollary 4.2. *Assume that $T : \mathcal{H} \to \ell_2$ is an analysis operator for some frame of $\mathcal{H}$ and $\mathrm{Prox} : \ell_2 \to \ell_2$ is an arbitrary proximity operator. Then $T^\dagger \mathrm{Prox}\, T$ is itself a proximity operator on $\mathcal{H}$ equipped with the norm $\| \cdot \|_{\mathcal{H}_T}$. In particular, if $\mathrm{Prox} := S_\lambda$, $\lambda > 0$, then*

$$T^\dagger S_\lambda (Tx) = \operatorname*{argmin}_{y \in \mathcal{H}} \left\{ \|x - y\|_{\mathcal{H}_T}^2 + f(y) \right\}, \quad f(y) := \lambda \| \cdot \|_1 \square \big(\tfrac{1}{2} \| \cdot \|_{\ell_2}^2 + \iota_{\mathcal{N}(T^*)} \big)(Ty).$$

Finally, let us have a look at the finite dimensional setting with $\mathcal{H} := \mathbb{R}^d$, $\mathcal{K} := \mathbb{R}^n$, $n \geq d$. Then, we have for any $T \in \mathbb{R}^{n,d}$ with full rank d and the proximity operator S_λ, $\lambda > 0$, on $\mathbb{R}^n$ that

$$T^\dagger S_\lambda (T(x)) = \operatorname*{argmin}_{y \in \mathbb{R}^d} \left\{ \tfrac{1}{2} \|x - y\|_{\mathcal{H}_T}^2 + f(y) \right\}, \quad f(y) := \lambda \| \cdot \|_1 \square \big(\tfrac{1}{2} \| \cdot \|_2^2 + \iota_{\mathcal{N}(T^{\mathrm{T}})} \big)(Ty). \tag{12}$$

Example 4.3. *We want to compute f for the matrix $T : \mathbb{R}^1 \to \mathbb{R}^2$ given by $T = (1,2)^{\mathrm{T}}$ and the soft shrinkage operator S_λ on $\mathbb{R}^2$ with $\lambda > 0$. Note that this example was also considered in [19]. By (12) and since $x = (x_1, x_2)^{\mathrm{T}} \in \mathcal{N}(T^*)$ if and only if $x_1 = -2x_2$, we obtain*

$$
\begin{aligned}
f(y) &= \lambda \| \cdot \|_1 \square \big(\tfrac{1}{2} \| \cdot \|_2^2 + \iota_{\mathcal{N}(T^*)}(\cdot) \big)(Ty) \\
&= \min_{Ty = z + x} \left\{ \lambda \|z\|_1 + \tfrac{1}{2} \|x\|_2^2 + \iota_{\mathcal{N}(T^*)}(x) \right\} \\
&= \min_{x \in \mathbb{R}^2} \left\{ \lambda \|Ty - x\|_1 + \tfrac{1}{2} \|x\|_2^2 + \iota_{\mathcal{N}(T^*)}(x) \right\} \\
&= \min_{x \in \mathbb{R}^2} \left\{ \lambda \big\| (y, 2y)^{\mathrm{T}} - (x_1, x_2)^{\mathrm{T}} \big\|_1 + \tfrac{1}{2} \|x\|_2^2 + \iota_{\mathcal{N}(T^*)}(x) \right\} \\
&= \min_{x_2 \in \mathbb{R}} \left\{ \lambda |y + 2x_2| + \lambda |2y - x_2| + \tfrac{5}{2} x_2^2 \right\}.
\end{aligned}
$$

Consider the strictly convex function $g_y(x_2) = \lambda |y + 2x_2| + \lambda |2y - x_2| + \tfrac{5}{2} x_2^2$. For $|y| \leq \tfrac{2}{5} \lambda$, it holds

$$0 \in \partial_{x_2} g_y \left(-\tfrac{y}{2} \right) = [-2\lambda, 2\lambda] - \lambda \mathrm{sgn}(y) - \tfrac{5}{2} y.$$

Hence, by Fermat's theorem, the unique minimizer of $g_y(x_2)$ is given by $-\tfrac{y}{2}$. Consequently, we have for $|y| \leq \tfrac{2}{5} \lambda$ that

$$f(y) = \tfrac{5\lambda}{2} |y| + \tfrac{5}{8} y^2.$$

For $|y| > \tfrac{2\lambda}{5}$, the function g_y is differentiable in $-\tfrac{\lambda}{5} \mathrm{sgn}(y)$ and it holds

$$\partial_{x_2} g_y \left(-\tfrac{\lambda}{5} \mathrm{sgn}(y) \right) = 2\lambda \mathrm{sgn}(y) - \lambda \mathrm{sgn}(y) - \lambda \mathrm{sgn}(y) = 0.$$

Therefore, for $|y| > \tfrac{2\lambda}{5}$, the minimizer of g_y is $-\tfrac{\lambda}{5} \mathrm{sgn}(y)$ and

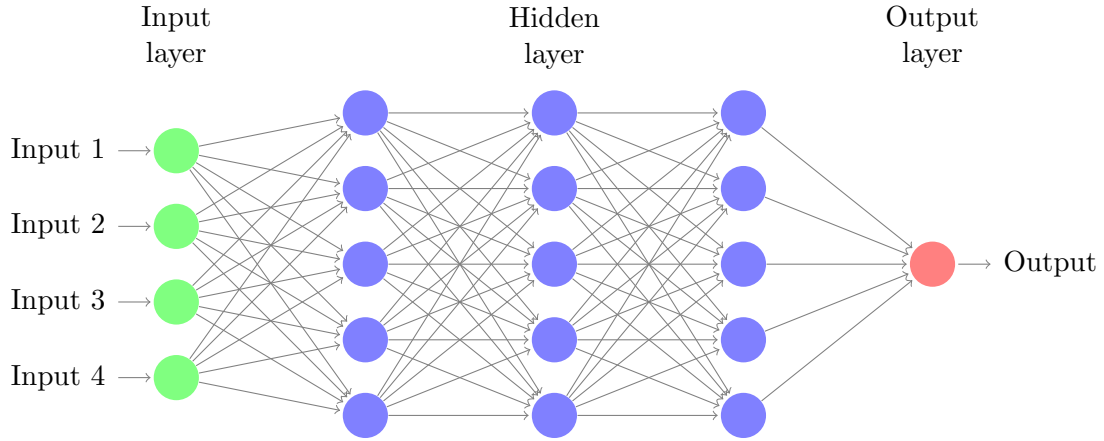$$f(y) = 3\lambda |y| - \tfrac{\lambda^2}{10}.$$

Figure 1: Model of a NN with three hidden layers, i.e., $d = 4$, $K = 4$, $n_1 = n_2 = n_3 = 5$, $n_4 = 1$.

Choosing, e.g., $\lambda = \frac{1}{3}$ we obtain

$$
f(y) = \begin{cases} \frac{5}{6}|y| + \frac{5}{8}y^2 & |y| \leq \frac{2}{15} \\ |y| - \frac{1}{90} & |y| > \frac{2}{15} \end{cases},
$$

which is a good approximation of $|y|$.

5. Proximal Neural Networks

In this section, we consider neural networks (NNs) consisting of $K \in \mathbb{N}$ layers with dimensions $n_1, \ldots, n_K$, defined by mappings $\Phi = \Phi(\cdot; u) \colon \mathbb{R}^d \to \mathbb{R}^{n_K}$ of the form

$$
\Phi(x; u) := A_K \sigma \circ A_{K-1} \sigma \circ \ldots \sigma \circ A_1(x). \tag{13}
$$

Such NNs are composed of affine functions $A_k \colon \mathbb{R}^{n_{k-1}} \to \mathbb{R}^{n_k}$ given by

$$
A_k(x) := L_k x + b_k, \qquad k = 1, \ldots, K, \tag{14}
$$

with weight matrices $L_k \in \mathbb{R}^{n_k, n_{k-1}}$, $n_0 = d$, bias vectors $b_k \in \mathbb{R}^{n_k}$ as well as a non-linear activation $\sigma \colon \mathbb{R} \to \mathbb{R}$ acting at each component, i.e., for $x = (x_j)_{j=1}^n$ we have $\sigma(x) = (\sigma(x_j))_{j=1}^n$. The parameter set $u := (L_k, b_k)_{k=1}^K$ of such a NN has the overall dimension $D := n_0 n_1 + n_1 n_2 + \cdots + n_{K-1} n_K + n_1 + \cdots + n_K$. For an illustration see Fig. 1.

In [12], the notation of stable activation functions was introduced. An activation function $\sigma \colon \mathbb{R} \to \mathbb{R}$ is called *stable*, if it is monotone increasing, 1-Lipschitz continuous and satisfies $\sigma(0) = 0$. The following result was proved in [12].

Lemma 5.1. *A function $\sigma \colon \mathbb{R} \to \mathbb{R}$ is a stable activation function if and only if there exists $g \in \Gamma_0(\mathbb{R})$ having 0 as a minimizer such that $\sigma = \text{prox}_g$.*

11

Various common activation functions σ and corresponding functions $g \in \Gamma_0(\mathbb{R})$ are listed in Tab. 3 in the appendix. For $T_k \in \mathbb{R}^{n_k,d}$, we consider the norm (7) and denote it by

$$\|x\|_{T_k} := \left(\|T_k x\|_2^2 + \|(I - T_k^\dagger T_k)x\|_2^2\right)^{\frac{1}{2}}, \qquad x \in \mathbb{R}^d. \tag{15}$$

In the previous sections, we have considered two different kinds of proximity operators, namely prox_g with respect to the Euclidean norm

$$\text{prox}_g = \underset{y \in \mathbb{R}^d}{\text{argmin}}\left\{\tfrac{1}{2}\|x - y\|_2^2 + g(y)\right\}, \tag{16}$$

and $\text{prox}_{T_k,g}$ with respect to the norm (15)

$$\text{prox}_{T_k,g} = \underset{y \in \mathbb{R}^d}{\text{argmin}}\left\{\tfrac{1}{2}\|x - y\|_{T_k}^2 + g(y)\right\}.$$

Further, we derived a function f_k depending on g, T_k and b_k, see Theorem 3.4, such that

$$\text{prox}_{T_k,f_k}(x) = \underset{y \in \mathbb{R}^d}{\text{argmin}}\left\{\tfrac{1}{2}\|x - y\|_{T_k}^2 + f_k(y)\right\} = T_k^\dagger \text{prox}_g(T_k x + b_k).$$

Based on our observations in the previous sections, we consider the following special NNs. We choose a stable activation function $\sigma = \text{prox}_g$ for some $g \in \Gamma_0(\mathbb{R})$ and matrices $T_k \in \mathbb{R}^{n_k,d}$, as well as bias vectors $b_k \in \mathbb{R}^{n_k}$, $k = 1, \dots, K$ and construct according to (14) the affine mappings

$$A_k(x) := \underbrace{T_k T_{k-1}^\dagger}_{L_k}(x) + b_k, \qquad k = 1, \dots, K. \tag{17}$$

Then, the NN $\Phi \colon \mathbb{R}^d \to \mathbb{R}^{n_K}$ in (13) with A_k, b_k in (17) can be rewritten as

$$\begin{aligned}\Phi(x; u) &= T_K T_{K-1}^\dagger \sigma\left(T_{K-1} \dots T_2^\dagger \sigma\left(T_2 T_1^\dagger \sigma(T_1 x + b_1) + b_2\right) \dots\right) + b_K \\ &= T_K \text{prox}_{T_{K-1},f_{K-1}} \circ \dots \circ \text{prox}_{T_1,f_1}(x) + b_K.\end{aligned} \tag{18}$$

We call Φ a *proximal neural network* (PNN) with network parameters $u := (T_k, b_k)_{k=1}^K$.

Next, we investigate stability properties of such networks. Recall that an operator $\Psi \colon \mathcal{H} \to \mathcal{H}$ on a Hilbert space $\mathcal{H}$ is α-averaged, $\alpha \in (0, 1)$, if there exists a nonexpansive operator $R \colon \mathcal{H} \to \mathcal{H}$ such that

$$\Psi = \alpha R + (1 - \alpha)I_{\mathcal{H}}.$$

The following theorem summarizes properties of α-averaged operators, c.f. [4] and [32] for the third statement.

Theorem 5.2. *Let $\mathcal{H}$ be a separable real Hilbert space. Then the following holds true:*

i) An operator on $\mathcal{H}$ is firmly nonexpansive if and only if it is $\tfrac{1}{2}$-averaged.

ii) *The concatenation of K operators which are α_k-averaged with respect to the same norm is α-averaged with $\alpha = \frac{K}{K-1+1/\max_k \alpha_k}$.*

iii) *For an α-averaged operator $\Psi \colon \mathcal{H} \to \mathcal{H}$ with a nonempty fixed point set, the sequence generated by the iteration*

$$x^{(r+1)} = \Psi\big(x^{(r)}\big)$$

converges weakly for every starting point $x^{(0)} \in \mathcal{H}$ to a fixed point of Ψ.

In the following, we study special PNNs, which are α-averaged operators such that $x^{(r+1)} = \Phi(x^{(r)}; u)$ converges to a fixed point of Φ.

Lemma 5.3.   i) *Let $T_k \in \mathbb{R}^{n_k,d}$ fulfill $T_k^* T_k = I_d$ or $T_k T_k^* = I_{n_k}$, $k = 1, \ldots, K-1$, and let $T_K = I_d$. Then Φ in (18) is α-averaged with $\alpha = \frac{K-1}{K}$.*

ii) *Let $T_1 \in \mathbb{R}^{n_k,d}$ have full column rank and let $T_k T_k^* = d_k T_1 T_1^*$, $d_k > 0$, $k = 1, \ldots, K-1$ and $T_K = I_d$. Then, Φ in (18) is α-averaged with $\alpha = \frac{K-1}{K}$. The setting is in particular fulfilled if the rows of T_k, $k = 1, \ldots, K-1$ form a tight frame.*

Proof. i) By Lemma 3.1 we know that $\| \cdot \|_{T_k} = \| \cdot \|_2$ so that Φ is the concatenation of $K-1$ proximity operators on $\mathbb{R}^d$ with respect to the Euclidean norm

$$\Phi = T_K \operatorname{prox}_{f_{K-1}} \circ \ldots \circ \operatorname{prox}_{f_1}(x) + b_K$$

with f_k as in Theorem 3.4. Now, the assertion follows from Theorem 5.2.

ii) By assumption, we obtain $\|x\|_{T_k} = x^* T_k^* T_k x = d_k x^* T_1^* T_1 x = d_k \|x\|_{T_1}$ so that

$$
\begin{aligned}
\operatorname{prox}_{T_k, f_k} &= \operatorname*{argmin}_{y \in \mathbb{R}^d} \big\{ \tfrac{1}{2} \|x - y\|_{T_k}^2 + f_k(y) \big\} \\
&= \operatorname*{argmin}_{y \in \mathbb{R}^d} \big\{ \tfrac{d_k}{2} \|x - y\|_{T_1}^2 + f_k(y) \big\} \\
&= \operatorname*{argmin}_{y \in \mathbb{R}^d} \big\{ \tfrac{1}{2} \|x - y\|_{T_1}^2 + d_k^{-1} f_k(y) \big\} = \operatorname{prox}_{T_1, d_k^{-1} f_k}.
\end{aligned}
$$

Clearly, if $f_k \in \Gamma_0(\mathbb{R}^d)$ then the same holds true for $d_k^{-1} f_k$. Hence, Φ becomes the concatenation of $K-1$ proximity operators on $\mathbb{R}^d$ all with respect to the T_1 norm. Then, the assertion follows from Theorem 5.2. $\qquad\square$

Remark 5.4. *Lemma 5.3 i) can be generalized to the case where $T_K \in \mathbb{R}^{d,d}$ is a symmetric positive semi-definite matrix with norm not larger than 1. In this case, T_K can be written in the form $T_K = Q^* Q$ for some $Q \in \mathbb{R}^{d,d}$ with $\|Q\|^2 = \|Q^* Q\| = \|T_K\| \le 1$. Thus, for every $x, y \in \mathbb{R}^d$,*

$$
\begin{aligned}
\|T_K x + b_K - (T_K y + b_K)\|^2 = \|Q^* Q(x - y)\|^2 &\le \|Q(x - y)\|^2 \\
&= \langle Q(x - y), Q(x - y) \rangle \\
&= \langle x - y, T_K x + b_K - (T_K y + b_K) \rangle.
\end{aligned}
$$

This shows that $T_K \cdot + b_K$ is firmly nonexpansive and therefore $\frac{1}{2}$-averaged. Consequently, Φ in (18) is the concatenation of K $\frac{1}{2}$-averaged operators with respect to the Euclidean norm. Hence Φ is itself α-averaged with $\alpha = \frac{K}{K+1}$.

Remark 5.5. *In [12], the following NN structure was studied: Let $\mathcal{H}_0, \ldots, \mathcal{H}_K$ be a sequence of real Hilbert spaces and $\mathcal{H}_0 = \mathcal{H}_K = \mathcal{H}$. Further, let $W_k \in \mathcal{B}(\mathcal{H}_{k-1}, \mathcal{H}_k)$ and $P_k \colon \mathcal{H}_k \to \mathcal{H}_k$, $k = 1, \ldots, K$ be firmly nonexpansive operators. For this case, Combettes and Pesquet [12] have posed conditions on W_k such that*

$$\Psi := W_K \circ P_{K-1} \circ W_{K-1} \circ \ldots \circ W_2 \circ P_1 \circ W_1 \qquad (19)$$

is α-averaged for some $\alpha \in (1/2, 1)$. For $\mathcal{H} = \mathbb{R}^d$ equipped with the Euclidean norm, $\mathcal{H}_k = \mathbb{R}^d$ equipped with the norm (15) and $T_K = I_d$, our PNN Φ has exactly the form (19) with $P_k := \mathrm{prox}_{T_k, f_k} \colon \mathcal{H}_k \to \mathcal{H}_k$ and the embedding operators $W_k \colon \mathcal{H}_{k-1} \hookrightarrow \mathcal{H}_k$, $k = 1, \ldots, K$. For the special PNNs in Lemma 5.3 it holds $W_k = I_d$, such that the conditions in [12] are fulfilled.

In the rest of this paper, we restrict our attention to matrices $T_k \colon \mathbb{R}^d \to \mathbb{R}^{n_k}$ fulfilling

$$T_k^* T_k = I_d \quad \text{or} \quad T_k T_k^* = I_{n_k}, \quad k = 1, \ldots, K-1,$$

and $T_K = I_d$, i.e., the rows, resp. columns of T_k form a Parseval frame. Then, the PNN in (18) has the form

(PPNN) $\qquad \Phi(x; u) = T_K \circ \mathrm{prox}_{f_{K-1}} \circ \ldots \circ \mathrm{prox}_{f_1}(x) + b_K,$

with the "usual" proximity operator, cf. (16), and

$$f_k(x) = g \square \big(\tfrac{1}{2} \| \cdot \|_2^2 + \iota_{\mathcal{N}(T_k^*)} \big)(T_k x + b_k) \quad \text{if } d \le n_k,$$
$$f_k(x) = g(T_k x + b_k) + \iota_{\mathcal{R}(T_k^*)}(x) \quad \text{if } d \ge n_k.$$

Due to the use of Parseval frames we call these networks *Parseval (frame) proximal neural networks* (PPNNs). By our previous considerations, see Lemma 5.3, PPNNs are averaged operators.

Remark 5.6. *An interesting result follows from convergence considerations of the cyclic proximal point algorithm, see [6]. Let $\{\lambda_r\}_{r \in \mathbb{N}} \in \ell_2 \setminus \ell_1$. Then, for every $x^{(0)} \in \mathbb{R}^d$, the sequence generated by*

$$x^{(r+1)} := \mathrm{prox}_{\lambda_r f_{K-1}} \circ \ldots \circ \mathrm{prox}_{\lambda_r f_1}\big(x^{(r)}\big) \qquad (20)$$

converges to a minimizer of $f_1 + \ldots + f_{K-1}$. In particular, Theorem 3.4 implies that for orthogonal matrices T_k, $k = 1, \ldots, K-1$ and $T_K = I_d$, $b_K = 0$, the sequence $\{x^{(r)}\}_{r \in \mathbb{N}}$ in (20) converges to

$$\hat{x} \in \underset{x}{\arg\min} \sum_{k=1}^{K-1} g(T_k x - b_k).$$

6. Training PPNNs on Stiefel Manifolds

In PPNNs, we assume that either T_k or T_k^*, $k = 1, \ldots, K-1$, is an element of the Stiefel manifold

$$\mathrm{St}\big(\min(n_k, d), \max(n_k, d)\big), \quad k = 1, \ldots, K - 1.$$

The following facts on Stiefel manifolds can be found, e.g., in [1]. For $d \leq n$, the (compact) Stiefel manifold is defined as

$$\mathrm{St}(d, n) \coloneqq \big\{T \in \mathbb{R}^{n,d} : T^*T = I_d\big\}.$$

For $d = 1$, this reduces to the sphere $\mathbb{S}^{n-1}$, and for $d = n$ we obtain the special orthogonal group $\mathrm{SO}(n)$. In general, $\mathrm{St}(d, n)$ is a manifold of dimension $nd - \frac{1}{2}d(d+1)$ with tangential space at $T \in \mathrm{St}(d, n)$ given by

$$\mathcal{T}_T\mathrm{St}(d, n) = \big\{TU + T_\perp V : U^* = -U, V \in \mathbb{R}^{n-d,d}\big\},$$

where the columns of $T_\perp \in \mathbb{R}^{n,n-d}$ are the basis of an orthonormal complement of T fulfilling $T_\perp^*T_\perp = I_{n-d}$ and $T^*T_\perp = 0$. The Riemannian gradient of a function on $\mathrm{St}(d, n)$ can be obtained by the orthogonal projection of the gradient in $\mathbb{R}^{n,d}$ onto $\mathrm{St}(d, n)$. The orthogonal projection of $X \in \mathbb{R}^{n,d}$ onto $\mathcal{T}_T\mathrm{St}(d, n)$ is given by

$$P_T X = (I_n - TT^*)X + \tfrac{1}{2}T(T^*X - X^*T), \tag{21}$$

$$= WT, \qquad W \coloneqq \hat{W} - \hat{W}^*, \quad \hat{W} \coloneqq XT^* - \tfrac{1}{2}T(T^*XT^*). \tag{22}$$

To emphasize that for fixed T the matrix W depends on X we will also write W_X. A *retraction* $\mathcal{R}$ on the manifold $\mathrm{St}(d, n)$ is a smooth mapping from the tangent bundle of $\mathrm{St}(d, n)$ to the manifold fulfilling $\mathcal{R}_T(0) = T$, where 0 is the zero element in $\mathcal{T}_T\mathrm{St}(d, n)$, and with the identification $\mathcal{T}_0(\mathcal{T}_T\mathrm{St}(d, n)) \cong \mathcal{T}_T\mathrm{St}(d, n)$ the local rigidity condition $D\mathcal{R}_T(0) = \mathrm{Id}_{\mathcal{T}_T\mathrm{St}(d,n)}$ holds true. A well-known retraction on $\mathrm{St}(d, n)$ is

$$\tilde{\mathcal{R}}_T(X) = \mathrm{qf}(T + X), \quad X \in \mathcal{T}_T\mathrm{St}(d, n), \tag{23}$$

where $\mathrm{qf}(A)$ denotes the Q factor of the decomposition of a matrix $A \in \mathbb{R}^{n,d}$ with linearly independent columns as $A = QR$ with $Q \in \mathrm{St}(d, n)$ and R an upper triangular matrix of size $d \times d$ with strictly positive diagonal elements. The complexity of the QR decomposition using the Householder algorithm is $2d^2(n - d/3)$, see [20]. Since the computation of the QR decomposition appears to be time consuming on a GPU, we prefer to apply another retraction, based on the Cayley transform of skew-symmetric matrices W in (22),

$$\mathcal{R}_T(X) = (I_n - \tfrac{1}{2}W)^{-1}(I_n + \tfrac{1}{2}W)T, \quad X \in \mathcal{T}_T\mathrm{St}(d, n), \tag{24}$$

see [30, 38]. By straightforward computation it can be seen that W_X and $W_{P_T X}$ coincide, so that the retraction (24) enlarged to the whole $\mathbb{R}^{n,d}$ fulfills

$$\mathcal{R}_T(X) = \mathcal{R}_T(P_T X), \qquad X \in \mathbb{R}^{n,d}. \tag{25}$$

Remark 6.1. *The retraction (24) has the drawback that it contains a matrix inversion. In our numerical algorithm, the following simple fixed point iteration is used for computing the matrix $R = \mathcal{R}_T(X)$ for fixed T and X. By definition, R fulfills the fixed point equation*

$$R = \tfrac{1}{2}WR + (I_n + \tfrac{1}{2}W)T. \tag{26}$$

Starting with an arbitrary $R^{(0)} \in \mathrm{St}(d,n)$, we apply the iteration

$$R^{(r+1)} := \tfrac{1}{2}WR^{(r)} + (I_n + \tfrac{1}{2}W)T,$$

which converges by Banach's fixed point theorem to the fixed point of (26) if $\tfrac{1}{2}\rho(W) < 1$, where $\rho(W)$ denotes the spectral radius of W.

We want to train a PPNN by minimizing

$$\mathcal{J}(u) := \sum_{i=1}^{N} \ell\big(\Phi(x_i; u); y_i\big), \tag{27}$$

where $\ell \colon \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is a differentiable loss function on the first d variables.

Example 6.2. *Let us specify two special cases of PPNNs with one layer.*

i) *For one layer without bias and componentwise soft shrinkage σ as activation function, i.e., summands*

$$\sum_{i=1}^{N} \ell\big(T_1^*\sigma(T_1 x_i); y_i\big), \quad T_1 \in \mathrm{St}(d, n_1),$$

we learn Parseval frames, e.g., for denoising tasks with y_i as a noisy version of x_i. Here, we want to mention the significant amount of work on dictionary learning, see [17], which starts with the same goal.

ii) *For $x_i = y_i$, $i = 1, \ldots, N$, the above network could be used as so-called autoencoder. Again, for one layer without activation function, $b_1 = 0$ and $\ell = h(\|x - y\|)$ with some norm $\|\cdot\|$ on $\mathbb{R}^d$ we get*

$$\sum_{i=1}^{N} \ell\left(T_1^*T_1 x_i; x_i\right) = \sum_{i=1}^{N} h\big(\|I_d - T_1^*T_1)x_i\|\big), \quad T_1^* \in \mathrm{St}(d, n_1).$$

For the Euclidean norm and $h(x) = x^2$ we get the classical PCA approach and for $h(x) = x$ the robust rotationally invariant L_1-norm PCA, recently discussed in [25, 29].

The following remark points out that special cases of our PPNNs were already considered in the literature.

Remark 6.3. *In [23], NNs with weight matrices $L_k \in \mathbb{R}^{n_k, n_{k-1}}$, $k \in \{1, \ldots, K-1\}$ such that the matrices or their transpose lie in a Stiefel manifold, were examined. The authors called this approach optimization over multiple dependent Stiefel manifolds (OMDSM). Indeed, by the following reasons, these NNs are special cases of our PPNNs if $n_k \leq d$ for all $k = 1, \ldots, K-1$. In particular, this implies that the NNs considered in [23] (with appropriately chosen last layer) are averaged operators.*

i) Case $n_k \leq n_{k-1}$: Let $L_k^* \in \mathrm{St}(n_k, n_{k-1})$, i.e., $L_k L_k^* = I_{n_k}$. Choosing an arbitrary fixed $T_{k-1} \in \mathbb{R}^{n_{k-1}, d}$ with $T_{k-1} T_{k-1}^* = I_{n_{k-1}}$, we want to find $T_k \in \mathbb{R}^{n_k, d}$ such that

$$T_k T_k^* = I_{n_k} \quad \text{and} \quad L_k = T_k T_{k-1}^*. \tag{28}$$

It is straightforward to verify that $T_k := L_k T_{k-1}$ has the desired properties.

Note that if the transposes of T_k and T_{k-1} are in a Stiefel manifold, this does not necessarily hold for the transpose of $T_k T_{k-1}^*$. Therefore, our PPNNs are more general.

ii) Case $n_{k-1} < n_k$: Let $L_k \in \mathrm{St}(n_{k-1}, n_k)$, i.e., $L_k^* L_k = I_{n_{k-1}}$. For an arbitrary fixed $T_{k-1} \in \mathbb{R}^{n_{k-1}, d}$ with $T_{k-1} T_{k-1}^* = I_{n_{k-1}}$, we want to find $T_k \in \mathbb{R}^{n_k, d}$ fulfilling (28). To this end, we complete L_k to an orthogonal matrix $\tilde{L}_k \in \mathbb{R}^{n_k, n_k}$ and T_{k-1} to a matrix $\tilde{T}_{k-1} \in \mathbb{R}^{n_k, d}$ with orthogonal rows. By straightforward computation we verify that $T_k := \tilde{L}_k \tilde{T}_{k-1}$ satisfies (28) such that this case also fits into our PPNN framework.

We apply a stochastic gradient descent algorithm on the Stiefel manifold to find a minimizer of (27). To this end, we compute the Euclidean gradient with respect to one layer and apply the usual backpropagation for multiple layers.

Lemma 6.4. *Let*

$$J(T, b) := \ell\big(T^* \sigma(Tx + b); y\big),$$

where T or T^* are in $\mathrm{St}(d, n)$. Set

$$r := \sigma(Tx+b), \quad s := T^* \sigma(Tx+b), \quad t := \nabla \ell\big(T^* \sigma(Tx+b); y\big), \quad \Sigma := \mathrm{diag}\big(\sigma'(Tx + b)\big),$$

where the gradient of ℓ is taken with respect to the first d variables. Then it holds for the Euclidean gradient

$$\nabla_T J(T, b) = -T(ts^* + st^*) + rt^* + \Sigma T t x^*, \qquad \nabla_b J(T, b) = \Sigma T t.$$

The proof follows by straightforward computations which are carried out in the appendix.

Now, we can formulate stochastic gradient descent (SGD) for $\mathcal{J}$ as in Algorithm 1. This algorithm works for an arbitrary retraction, in particular the retraction in (23). In our numerical computations we use the special retraction (24) in connection with the iteration scheme (6.1). Then, by (25), the projection step 3 of the algorithm can be skipped and the retraction can be directly applied to the Euclidean gradient.

Algorithm 1 Stochastic Gradient Descent Algorithm for minimizing (27)

Input: Training data $(x_i, y_i)_{i=1}^N$, batch size $B \in \mathbb{N}$, learning rate $(\lambda^{(r)})_{r \in \mathbb{N}}$.

Initialization: $(T^{(0)}, b^{(0)})$.

for $r = 0, 1, ...$ **do**

    1. Choose a mini batch $I \subset \{1, \dots, N\}$ of size $|I| = B$.

    2. Compute the Euclidean gradients of $\mathcal{J}_I(T, b) = \sum_{i \in I} \ell(\Phi(x_i; u); y_i)$
    using Lemma 6.4 and backpropagation

$$\nabla_b \mathcal{J}_I\big(T^{(r)}, b^{(r)}\big) = \sum_{i \in I} \nabla_b \ell\big(\Phi(x_i; u); y_i\big),$$

$$\nabla_T \mathcal{J}_I\big(T^{(r)}, b^{(r)}\big) = \sum_{i \in I} \nabla_T \ell\big(\Phi(x_i; u); y_i\big).$$

    3. Compute the Riemannian gradient on the Stiefel manifold with respect to T
    by projection (21). Skip this step if retraction (24) is used

$$\nabla_{\mathrm{St}, T} \mathcal{J}_I\big(T^{(r)}, b^{(r)}\big) = P_{T^{(r)}}\big(\nabla_T \mathcal{J}_I\big(T^{(r)}, b^{(r)}\big)\big).$$

    4. Update T and b by a gradient descent step with retraction, see (23)

$$\tilde{T}^{(r+1)} = \tilde{\mathcal{R}}_{T^{(r)}}\big(-\tfrac{\lambda^{(r)}}{B} \nabla_{\mathrm{St}, T} \mathcal{J}_I\big(T^{(r)}, b^{(r)}\big)\big),$$

$$b^{(r+1)} = b^{(r)} - \tfrac{\lambda^{(r)}}{B} \nabla_b \mathcal{J}_I\big(T^{(r)}, b^{(r)}\big).$$

7. Some Numerical Results

In this section, we present simple numerical results to get a first impression on the performance of PPNNs for denoising and classification. More sophisticated examples which include the full repertoire of fine tuning of NNs will follow in an experimental paper, see also our conclusions. Throughout this section, we use the quadratic loss function $\ell(x; y) := \|x - y\|_2^2$. For training we apply a stochastic gradient descent algorithm. The batch size and learning rate are given for all examples separately.

Denoising. In this experiment, we compare PPNNs with Haar wavelet thresholding both for discrete Haar bases and Haar frames arising from the undecimated (translation invariant) version of the Haar transform. In particular, the experiment is linked to the starting point of our considerations, namely wavelet and frame shrinkage. For further details on the corresponding filter banks we refer to [14, 31]. As quality measure for our experiments we choose the average peak signal to noise ratio (PSNR) over the test set. Recall that for a prediction $x \in \mathbb{R}^n$ and ground truth $y \in \mathbb{R}^n$ the PSNR is defined by

$$\mathrm{PSNR}(x, y) = 10 \log_{10}\Big(\frac{(\max y - \min y)^2}{\sum_{i=1}^n (x_i - y_i)^2}\Big).$$

Since we focus on the Haar filter, we restrict our attention to piecewise constant signals

with mean 0. By $(x_i, y_i) \in \mathbb{R}^d \times \mathbb{R}^d$, $i = 1, \ldots, N$, we denote pairs of piecewise constant signals y_i of length $d = 2^7 = 128$ and their noisy versions by $x_i = y_i + \epsilon_i$, where ϵ_i is white noise with standard deviation $\sigma = 0.1$. For the *signal generation*, we choose

- the number of constant parts of y_i as $\max\{2, t_i\}$, where t_i is the realization of a random variable following the Poisson distribution with mean 5;

- the discontinuities of y_i as realization of a uniform distribution;

- the signal intensity of y_i for every constant part as realization of the standard normal distribution, where we subtract the mean of the signal finally.

Using this procedure, we generate training data $(x_i, y_i)_{i=1}^{N}$ and test data $(x_i, y_i)_{i=N+1}^{N+N_{\text{test}}}$ with $N = 500000$ and $N_{\text{test}} = 1000$. The average PSNR of the noisy signals in the test set is 25.22. We use PPNNs with $K - 1 \in \{1, 2, 3\}$ layers and set $T_K = I_d$ and $b_K = 0$. In all examples a batch size of 32 and a learning rate of 0.5 is used.

We are interested in two different settings:

1. *Learned orthogonal matrices versus Haar basis.* First, we consider PPNNs with 128 neurons in each hidden layer and componentwise soft-shrinkage S_λ as activation function. In particular, all matrices T_k have to be orthogonal. The denoising results of our learned PPNN are compared with the soft wavelet shrinkage with respect to the discrete orthogonal Haar basis in $\mathbb{R}^{128}$, i.e., the signal on all 6 scales is decomposed by

$$\Psi(x) = H^* S_\lambda(Hx), \tag{29}$$

where $H := H_2 \cdots H_7$ with matrices

$$H_j := \begin{pmatrix} \tilde{H}_j & 0 \\ 0 & I_{2^7 - 2^j} \end{pmatrix}, \quad \tilde{H}_j := \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 & 0 & 0 & \ldots & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & \ldots & 0 & 0 & 0 & 0 \\ \vdots & & & & & & & & \\ 0 & 0 & 0 & 0 & \ldots & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & \ldots & 0 & 0 & 1 & 1 \\ 1 & -1 & 0 & 0 & \ldots & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & \ldots & 0 & 0 & 0 & 0 \\ \vdots & & & & & & & & \\ 0 & 0 & 0 & 0 & \ldots & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & \ldots & 0 & 0 & 1 & -1 \end{pmatrix} \in \mathbb{R}^{2^j, 2^j}.$$

The average PSNRs on the test data are given in Tab. 1, where the optimal threshold in S_λ was determined by 5-fold cross validation. More precisely, the training data is divided into 5 subsets and each is used once as a test set with the remaining samples as training set. The test loss for given λ is averaged over all 5 trials for judging the quality of the model. The tested parameters λ are chosen in $[0.05, 0.3]$ with steps of 0.05 for a NN with one layer. For NNs with two and three layers, the tested parameters are divided by 2 and 3, respectively. It appears that for only one hidden layer, the Haar wavelet shrinkage is still better than the learned orthogonal matrix. If we increase the number of layers, then PPNNs lead to a better average PSNR.

Method	PSNR	Loss	Optimal λ
Haar basis	29.89	0.00359	0.1
One layer	29.73	0.00373	0.1
Two layers	30.55	0.00313	0.05
Three layers	30.83	0.00296	0.033

Table 1: PSNRs (average on test data) for denoising piecewise constant signals. For only one layer, the Haar wavelet shrinkage is still better than the learned orthogonal matrix. This changes if we increase the number of layers.

Two exemplary noisy signals and their denoised versions are shown in Fig. 2. Since we have learned the orthogonal matrices T_k with respect to the quadratic loss function, the visual quality of the PPNN denoised signals is clearly not satisfactory even with an improved PSNR. The visual impression of signals denoised by Haar wavelet shrinkage can by improved (smoother signal) by increasing the threshold to $\lambda = 0.3$, resulting in a worse PSNR. To achieve a similar behavior with orthogonal matrices learned by PPNNs, we have to choose a different loss function.

2. *Learned Stiefel matrices versus Haar frame.* Haar wavelet shrinkage can be improved by using Haar wavelet frames within a so-called "algorithm á trous", see [27]. We apply a similar method as in (29), but with a rectangular matrix H whose rows form a Haar frame. More precisely, the Haar filter is used without subsampling. This results, in contrast to the original Haar transform, in a translational invariant multiscale transform. Instead of the matrices H_7, the nonsubsampled (convolution) matrix

$$
\frac{1}{\sqrt{2}}
\begin{pmatrix}
1 & 1 & 0 & 0 & \ldots & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 & \ldots & 0 & 0 & 0 & 0 \\
\vdots & & & & & & & & \\
0 & 0 & 0 & 0 & \ldots & & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & \ldots & 0 & 0 & 1 & 1 \\
1 & 0 & 0 & 0 & \ldots & 0 & 0 & 0 & 1 \\
1 & -1 & 0 & 0 & \ldots & 0 & 0 & 0 & 0 \\
0 & 1 & -1 & 0 & \ldots & 0 & 0 & 0 & 0 \\
\vdots & & & & & & & & \\
0 & 0 & 0 & 0 & \ldots & 0 & 1 & -1 & 0 \\
0 & 0 & 0 & 0 & \ldots & 0 & 0 & 1 & -1 \\
-1 & 0 & 0 & 0 & \ldots & 0 & 0 & 0 & 1
\end{pmatrix}
\in \mathbb{R}^{256,128}.
$$

is applied to the original signal. Note that we assume a periodic continuation of the signal. In each of the following j steps, $j = 1, \ldots, 6$, we keep the lower part and transform again the upper smoothed part by essentially the same matrix, where $2^j - 1$ zeros are inserted between the filter coefficients. The output signal has size $8 \cdot 128$, where the last part of the signal is just the averaged signal, which is equal to zero. Overall, the original signal is multiplied by $H \in \mathbb{R}^{1024,128}$, where for $j \in \{0, \ldots, 6\}$ and $i \in \{0, \ldots, 128 - 2j\}$

(a) Original I

(b) Noisy (PSNR 29.11)

(c) Haar basis, $\lambda = 0.1$ (PSNR 33.20)

(d) 1 layer, $\lambda = 0.1$ (PSNR 35.04)

(e) 3 layers, $\lambda = 0.033$ (PSNR 34.37)

(f) Haar basis, $\lambda = 0.3$ (PSNR 27.43)

(g) Original II

(h) Noisy (PSNR 17.45)

(i) Haar basis, $\lambda = 0.1$ (PSNR 23.81)

(j) 1 layer, $\lambda = 0.1$ (PSNR 23.32)

(k) 3 layers, $\lambda = 0.033$ (PSNR 24.87)

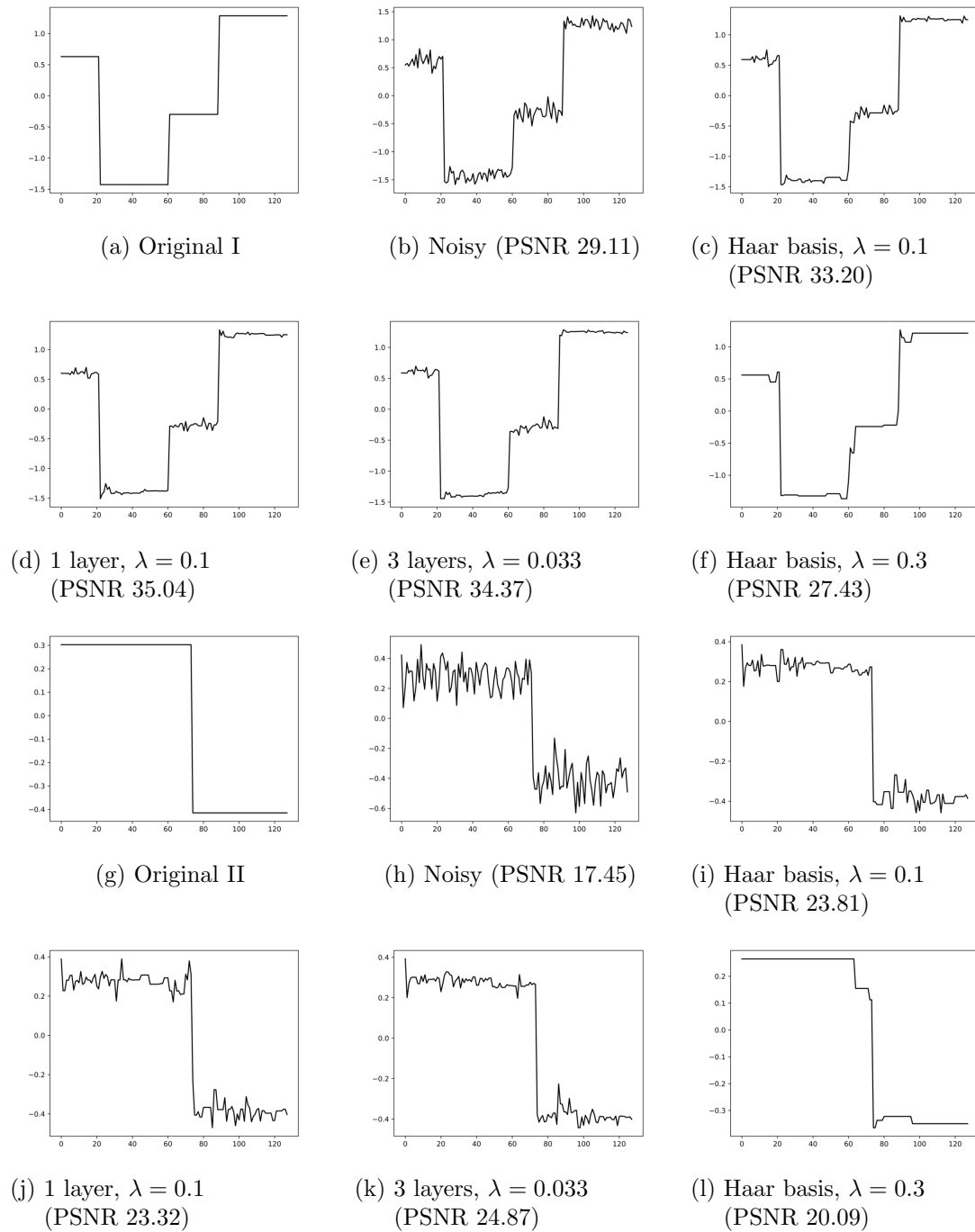(l) Haar basis, $\lambda = 0.3$ (PSNR 20.09)

Figure 2: Two denoising examples using the Haar basis and learned orthogonal matrices. The signals denoised by PPNNs look better for an increasing number of layers. For Haar wavelet shrinkage, a smoother denoised signal can be attained by increasing the threshold to $\lambda = 0.3$, although this signal has a smaller PSNR.

Method	PSNR	Loss	Optimal λ
Haar frame with S_λ	26.99	0.00669	0.02
Haar frame with $\tilde{S}_\lambda$	30.58	0.00307	0.08
One layer PPNN	32.46	0.00212	0.06
Two layer PPNN	32.68	0.00206	0.04
Three layer PPNN	32.91	0.00195	0.02

Table 2: PSNRs (average on test data) for denoising piecewise constant signals. The first row with non-scale-adapted thresholds are worse than those with the adapted ones in the second row. Learned Stiefel matrices with the same componentwise activation function lead to better results when the corresponding PPNNs are applied for denoising.

the $(i + 128j)$-th row is given by one element of a Haar wavelet frame

$$(\ \underbrace{0 \ \ ... \ \ 0}_{i \text{ times } 0} \ \ \underbrace{1 \ \ ... \ \ 1}_{j \text{ times } 1} \ \ \underbrace{-1 \ \ ... \ \ -1}_{j \text{ times } -1} \ \ \underbrace{0 \ \ ... \ \ 0}_{128-i-2j \text{ times } 0} \ )$$

and the last 128 rows of H are given by $(\ 1 \ \ ... \ \ 1 \ )$. It is well known that for the above translation invariant Haar frame transform, a scale dependent shrinkage has to be applied to each scale, namely starting with threshold λ the next scales should be thresholded by

$$\frac{1}{\sqrt{2}^j}\lambda, \quad j = 0, \ldots, 6. \tag{30}$$

For an explanation of this statement we refer to [35]. In summary, we obtain

$$\hat{\Psi}(x) = H^* \tilde{S}_\lambda(Hx),$$

where $\tilde{S}_\lambda$ denotes the scale-wise adapted thresholding.

Now, we compare this scale-dependent Haar frame soft thresholding method with a learned PPNN with 1024 neurons in the hidden layers and componentwise soft-shrinkage S_λ as activation function. The optimal threshold in S_λ is determined by 3-fold cross validation, where the tested parameters are chosen in $[0.01, 0.1]$ with steps of 0.01. We emphasize that in contrast to the Haar frame shrinkage procedure with (30), the same threshold for each component is used in the activation function of our PPNN. The resulting PSNRs are given in Tab. 2. As expected, using the same threshold in the classical Haar frame shrinkage is worse than the scale-adapted Haar frame shrinkage. PPNNs with learned Stiefel matrices perform better for an increasing number of layers.

Finally, Fig. 3 contains the denoised signals from Fig. 2. The results are visually better than in the previous figure, although still not satisfactory due to the used loss function.

Classification: In this example, we train a PPNN for classifying the MNIST data set[1]. The length of the input signals is $d = 28^2$. We consider a PPNN with $K - 1 = 5$

---

[1]http://yann.lecun.com/exdb/mnist

(a) Haar frame with $\tilde{S}_\lambda$, $\lambda = 0.08$, (PSNR 34.26)

(b) 1 layer, $\lambda = 0.06$, (PSNR 36.03)

(c) 3 layers, $\lambda = 0.02$, (PSNR 36.75)

(d) Haar frame with $\tilde{S}_\lambda$, $\lambda = 0.08$, (PSNR 23.45)

(e) 1 layer, $\lambda = 0.06$, (PSNR 26.74)
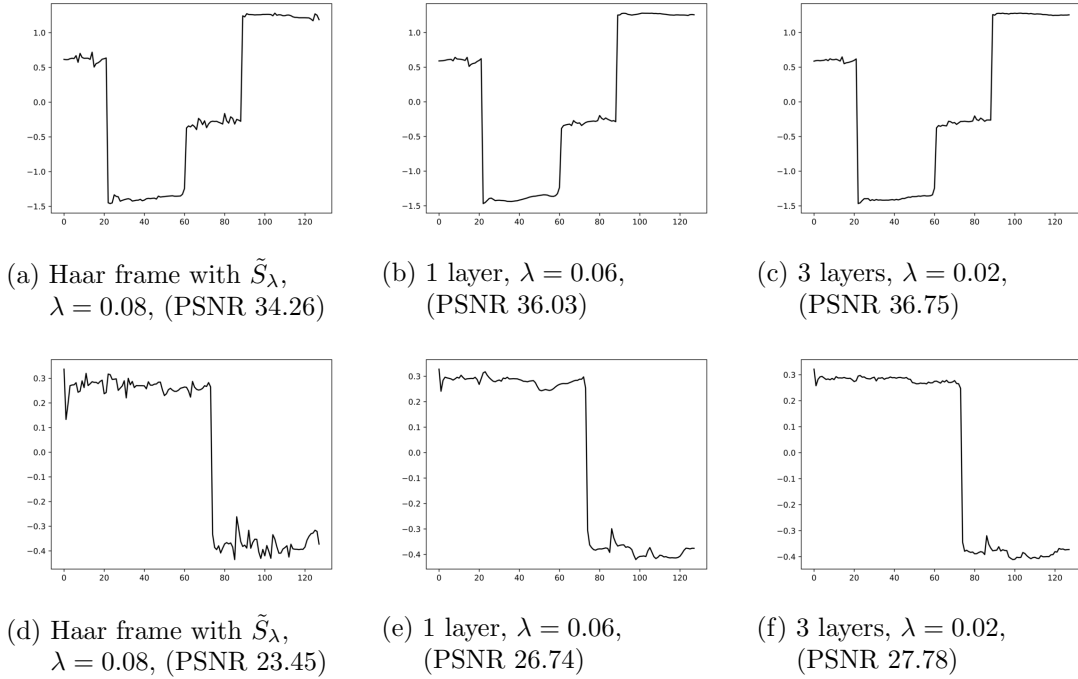
(f) 3 layers, $\lambda = 0.02$, (PSNR 27.78)

Figure 3: Denoising for the signals in Fig. 2. The undecimated Haar frame with scale-adapted shrinkage and learned Stiefel matrices of the same size as the Haar frame are compared for λ from Tab. 2. The PPNN denoised signals are visually nicer than those with scale-adapted Haar frame shrinkage.

layers and $n_1 = n_2 = 784$, $n_3 = n_4 = 400$ and $n_5 = 200$ neurons in the layers and componentwise applied ReLu activation function $\sigma(x) = \max(0, x)$. To get 10 output elements (probabilities) in $(0, 1)$, we use an additional sixth layer

$$g(T_K x + b_K), \quad T_K \in \mathbb{R}^{10,d}, \ b_K \in \mathbb{R}^{10}$$

with another activation function $g(x) \coloneqq \frac{1}{1+\exp(-x)}$. For training, we use a batch size of 1024 and a learning rate of 5, ending up with an accuracy of 0.9855 on the test set. In Fig. 4 the training and test loss of our PPNN during training are plotted.

Remark 7.1. *As already mentioned in Remark 6.3, NNs with Stiefel matrices were also applied in [23]. The authors of [23] reported that the training process using Riemannian optimization on the Stiefel manifold could be unstable or divergent. We do not observe such instabilities in our setting.*

8. Conclusions

In this paper, we have shown that for real Hilbert spaces $\mathcal{H}$ and $\mathcal{K}$, a proximity operator $\mathrm{Prox}\colon \mathcal{K} \to \mathcal{K}$ and a linear bounded operator $T\colon \mathcal{H} \to \mathcal{K}$ the operator $T^\dagger \mathrm{Prox}(T \cdot + b)$
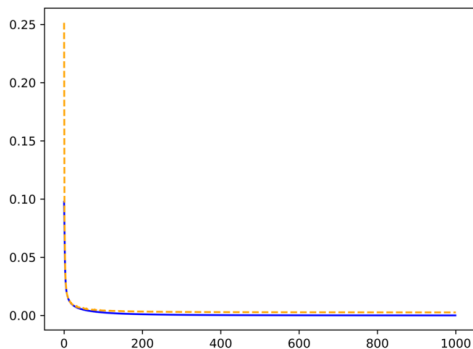
Figure 4: Training loss (blue) and test loss (orange) of a PPNN on the MNIST data set. The x-axis corresponds to the number of epochs and the y-axis to the associated loss value.

with $b \in \mathcal{K}$ is a proximity operator in $\mathcal{H}$. As a consequence, the famous frame soft shrinkage operator can be seen as a proximity operator. Using this new relations, we have discussed special neural networks arising from Parseval frames and stable activation functions. These networks include recently proposed ones containing matrices whose transposes are in a Stiefel manifold and interpret them from another, more general point of view. In our future work we want to explore for which learning tasks the higher flexibility of our PPNNs is advantageous. Taking more general operators T into account, may be also useful.

Another question we want to address is to constrain our Stiefel matrices further, e.g., towards convolutional networks and to sparsity constraints. Depending on the application, we have to design appropriate loss functions as well as to incorporate regularizing terms.

For our experiments the stochastic gradient algorithm on Stiefel manifolds worked well. However, other minimization methods could be taken into account. In [23] for example, the authors proposed an orthogonal weight normalization algorithm which was inspired by the fact that eigenvalue decomposition is differentiable. Finally, we like to mention that a proximal backpropagation algorithm taking implicit instead of explicit gradient steps to update the network parameters during neural network training was proposed in [18].

A better understanding of the convergence of the cyclic proximal point algorithm, see Remark 5.6, and suitable early stopping criteria if the network Φ is iteratively used may help to design NNs and to understand their success.

A. Gradient computation

Proof of Lemma 6.4: For potential further applications, we compute the gradient of the more general functional

$$J(T, b) := \ell\big(T^\dagger \sigma(Tx + b); y\big) = \ell\big(f(T, b); y\big),$$

where $T^\dagger = (T^*T)^{-1}T^*$. Note that the case $T^\dagger = T^*(TT^*)^{-1}$ can be treated in an analog way. Using $f_1(T) = (T^*T)^{-1}$, $f_2(T) = T^*$, $f_3(T, b) = \sigma(Tx + b)$, the function $f(T, b) := T^\dagger \sigma(Tx + b)$ is decomposed as

$$f(T, b) = f_1(T) \, f_2(T) \, f_3(T, b).$$

By the product rule it holds

$$\begin{aligned}
&D_T f(T, b)[H] \\
&= Df_1(T)[H] \, f_2(T) \, f_3(T, b) + f_1(T) \, Df_2(T)[H] \, f_3(T, b) + f_1(T) \, f_2(T) \, D_T f_3(T, b)[H].
\end{aligned}$$

The involved differentials are given by

$$\begin{aligned}
Df_1(T)[H] &= -(T^*T)^{-1}(T^*H + H^*T)(T^*T)^{-1}, \\
Df_2(T)[H] &= H^*, \\
D_T f_3(T, b)[H] &= \mathrm{diag}\big(\sigma'(Tx + b)\big) \, Hx.
\end{aligned}$$

Consequently, we obtain

$$D_T f(T, b)[H] = -(T^*T)^{-1}\big((T^*H + H^*T)(T^*T)^{-1}s + H^* \, r + T^*\Sigma Hx\big).$$

Using the chain rule, we conclude

$$\begin{aligned}
D_T J(T, b)[H] &= \langle \nabla_T J(T, b), H \rangle = \langle t, D_T f(T, b)[H] \rangle \\
&= -\langle t, (T^*T)^{-1}(T^*H + H^*T)(T^*T)^{-1}s \rangle + \langle t, (T^*T)^{-1}H^*r \rangle + \langle t, (T^*T)^{-1}T^*\Sigma Hx \rangle \\
&= -\langle T(T^*T)^{-1}ts^*(T^*T)^{-1}, H \rangle - \langle (T^*T)^{-1}ts^*(T^*T)^{-1}T^*, H^* \rangle \\
&\quad + \langle (T^*T)^{-1}tr^*, H^* \rangle + \langle \Sigma T(T^*T)^{-1}tx^*, H \rangle \\
&= \langle -T(T^*T)^{-1}(ts^* + st^*)(T^*T)^{-1} + rt^*(T^*T)^{-1} + \Sigma T(T^*T)^{-1}tx^*, H \rangle.
\end{aligned}$$

Thus,

$$\nabla_T J(T, b) = -T(T^*T)^{-1}(ts^* + st^*)(T^*T)^{-1} + rt^*(T^*T)^{-1} + \Sigma T(T^*T)^{-1}tx^*.$$

For the gradient with respect to b, we obtain by the chain rule

$$\nabla_b J(T, b)^* = t^*T^\dagger \nabla_b\big(\sigma(Tx + b)\big)(b) = t^*T^\dagger \Sigma.$$

$\square$

Name	activation function $\sigma(x)$	$f(x)$ with $\sigma = \text{prox}_f$
Linear activation	x	0
Saturated linear activation (SaLU)	$\begin{cases} 1 & \text{if } x > 1 \\ x & \text{if } -1 \leq x \leq 1 \\ -1 & \text{if } x < -1 \end{cases}$	$\iota_{[-1,1]} = \begin{cases} 0 & \text{if } x \in [-1,1] \\ \infty & \text{if } x \notin [-1,1] \end{cases}$
Soft Thresholding	$\begin{cases} x - \lambda & \text{if } x > \lambda \\ 0 & \text{if } x \in [-\lambda, \lambda] \\ x + \lambda & \text{if } x < -\lambda \end{cases}$	$\lambda |x|$
Saturated linear activation (SaLU)	$\begin{cases} 1 & \text{if } x > 1 \\ x & \text{if } -1 \leq x \leq 1 \\ -1 & \text{if } x < -1 \end{cases}$	$\iota_{[-1,1]} = \begin{cases} 0 & \text{if } x \in [-1,1] \\ \infty & \text{if } x \notin [-1,1] \end{cases}$
Rectified linear unit (ReLU)	$\begin{cases} x & \text{if } x > 0 \\ 0 & \text{if } x \leq 0 \end{cases}$	$\iota_{[0,\infty)} = \begin{cases} 0 & \text{if } x \in [0,\infty) \\ \infty & \text{if } x \notin [0,\infty) \end{cases}$
Parametric rectified linear unit (PReLU)	$\begin{cases} x & \text{if } x > 0 \\ \alpha x & \text{if } x \leq 0 \end{cases}, \alpha \in (0,1]$	$\begin{cases} 0 & \text{if } x > 0 \\ (\frac{1}{\alpha} - 1)\frac{x^2}{2} & \text{if } x \leq 0 \end{cases}$
Bent identity activation	$\dfrac{x + \sqrt{x^2 + 1}}{2}$	$\begin{cases} x/2 - \ln(x + \frac{1}{2})/4 & \text{if } x > -\frac{1}{2} \\ \infty & \text{if } x \leq -\frac{1}{2} \end{cases}$
Inverse square root linear unit	$\begin{cases} x & \text{if } x \geq 0 \\ \dfrac{x}{\sqrt{x^2 + 1}} & \text{if } x < 0 \end{cases}$	$\begin{cases} 0 & \text{if } x \geq 0 \\ 1 - x^2/2 - \sqrt{1 - x^2} & \text{if } -1 \leq x < 0 \\ \infty & \text{if } x < -1 \end{cases}$
Inverse square root unit	$\dfrac{x}{\sqrt{x^2 + 1}}$	$\begin{cases} -x^2/2 - \sqrt{1 - x^2} & \text{if } |x| \leq 1 \\ \infty & \text{if } |x| > 1 \end{cases}$
Arctangent activation	$\dfrac{2}{\pi}\arctan(x)$	$\begin{cases} -\dfrac{2}{\pi}\ln(\cos(\frac{\pi x}{2})) - \dfrac{x^2}{2} & \text{if } |x| < 1 \\ \infty & \text{if } |x| \geq 1 \end{cases}$
Hyperbolic tangent activation	$\tanh(x)$	$\begin{cases} x\,\text{arctanh}(x) + \dfrac{\ln(1 - x^2) - x^2}{2} & \text{if } |x| < 1 \\ \infty & \text{if } |x| \geq 1 \end{cases}$
Elliot activation	$\dfrac{x}{|x| + 1}$	$\begin{cases} -|x| - \ln(1 - |x|) - \dfrac{x^2}{2} & \text{if } |x| < 1 \\ \infty & \text{if } |x| \geq 1 \end{cases}$
Inverse hyperbolic sine	$\text{arcsinh}(x)$	$\cosh(x) - \dfrac{|x|^2}{2}$
Logarithmic activation	$\text{sgn}(x)\ln(|x| + 1)$	$\exp(|x|) - |x| - 1 - \dfrac{|x|^2}{2}$

Table 3: Stable activation functions and their corresponding proximal mappings, see [12].

B. Activation functions

The following table lists many functions f having a proximity σ which is a common activation function in NNs from [12].

References

[1] P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization Algorithms on Matrix Manifolds.* Princeton and Oxford, Princeton University Press, 2008.

[2] M. Arjovsky, A. Shah, and Y. Bengio. Unitary evolution recurrent neural networks. In *International Conference on Machine Learning*, pages 1120–1128, 2016.

[3] N. Bansal, X. Chen, and Z. Wang. Can we gain more from orthogonality regularizations in training deep networks? In *Advances in Neural Information Processing Systems*, pages 4261–4271, 2018.

[4] H. H. Bauschke and P. L. Combettes. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces.* Springer, New York, 2011.

[5] A. Beck. *First-Order Methods in Optimization*, volume 25 of *MOS-SIAM Series on Optimization.* SIAM, 2017.

[6] D. P. Bertsekas. Incremental proximal methods for large scale convex optimization. *Mathematical programming*, 129:163–195, 2011.

[7] M. Burger, A. Sawatzky, and G. Steidl. First order algorithms in variational image processing. In *Operator Splittings and Alternating Direction Methods*. Springer, 2017.

[8] S. H. Chan, X. Wang, and O. A. Elgendy. Plug-and-play ADMM for image restoration: Fixed-point convergence and applications. *IEEE Transactions on Computational Imaging*, 3:84–98, 2016.

[9] E. Chouzenoux, J.-C. Pesquet, and A. Repetti. Variable metric forward-backward algorithm for minimizing the sum of a differentiable function and a convex function. *Journal of Optimization Theory and Applications*, 162:107–132, 2014.

[10] O. Christensen. *An Introduction to Frames and Riesz Bases.* Springer, 2016.

[11] P. L. Combettes. Monotone operator theory in convex optimization. *Mathematical Programming*, 170(1):177–206, 2018.

[12] P. L. Combettes and J.-C. Pesquet. Deep neural network structures solving variational inequalities. *ArXiv Preprint*, 2018.

[13] P. L. Combettes and V. R. Wajs. Signal recovery by proximal forward-backward splitting. *Multiscale Modeling & Simulation*, 4:1168–1200, 2005.

[14] Z. Cvetković and M. Vetterli. Oversampled filter banks. *IEEE Transactions on Signal Processing*, 46:1245–1255, 1998.

[15] I. Daubechies, M. Defrise, and C. De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, 57(11):1413–1457, 2004.

[16] V. Dorobantu, P. A. Stromhaug, and J. Renteria. DIZZYRNN: Reparameterizing recurrent neural networks for norm-preserving backpropagation. In *CoRR abs/1612.04035*, 2016.

[17] M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image processing*, 15(12):3736–3745, 2006.

[18] T. Frerix, T. Möllenhoff, M. Moeller, and D. Cremers. Proximal backpropagation. Technical report, ArXiv Preprint, 2017.

[19] J. A. Geppert and G. Plonka. Frame soft shrinkage operators are proximity operators. Technical report, ArXiv Preprint, 2019.

[20] G. H. Golub and C. F. V. Loan. *Matrix Computations*. The Johns Hopkins University Press, 2013.

[21] R. Gribonval and M. Nikolova. A characterization of proximity operators. *arXiv preprint arXiv:1807.04014*, 2018.

[22] M. Harandi and B. Fernando. Generalized backpropagation, etude de cas: Orthogonality. In *CoRR abs/1611.05927*, 2016.

[23] L. Huang, X. Liu, B. Lang, A. W. Yu, Y. Wang, and B. Li. Orthogonal weight normalization: Solution to optimization over multiple dependent stiefel manifolds in deep neural networks. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[24] L. Jing, Y. Shen, T. Dubcek, J. Peurifoy, S. Skirlo, Y. LeCun, M. Tegmark, and M. Soljačić. Tunable efficient unitary neural networks (EUNN) and their application to RNNs. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1733–1741. JMLR. org, 2017.

[25] G. Lerman and T. Maunu. An overview of robust subspace recovery. *Proceedings of the IEEE*, 106(8):1380–1410, 2018.

[26] M. Lezcano-Casado and D. Martínez-Rubio. Cheap orthogonal constraints in neural networks: A simple parametrization of the orthogonal and unitary group. *arXiv preprint arXiv:1901.08428*, 2019.

[27] S. Mallat. *A wavelet tour of signal processing: the sparse way.* Access Online via Elsevier, 2008.

[28] J.-J. Moreau. Proximité et dualité dans un espace Hilbertien. *Bulletin de la Société Mathématique de France*, 93:273–299, 1965.

[29] S. Neumayer, M. Nimmer, S. Setzer, and G. Steidl. On the rotational invariant l_1-norm PCA. *Linear Algebra and its Applications*, 587:243–270, 2019.

[30] Y. Nishimori and S. Akaho. Learning algorithms utilizing quasi-geodesic flows on the Stiefel manifold. *Neurocomputing*, 67:106–135, 2005.

[31] G. Plonka and G. Steidl. A multiscale wavelet-inspired scheme for nonlinear diffusion. *International Journal of Wavelets, Multiresolution and Information Processing*, (1):1–21, 2006.

[32] S. Reich. Weak convergence theorems for nonexpansive mappings in Banach spaces. *Journal of Mathematical Analysis and Applications*, 67:274–276, 1979.

[33] S. Setzer. Operator splittings, Bregman methods and frame shrinkage in image processing. *International Journal of Computer Vision*, 92(3):265–280, 2011.

[34] S. Sreehariand, S. V. Venkatakrishnan, and B. Wohlberg. Plug-and-play priors for bright field electron tomography and sparse interpolation. *IEEE Transactions on Computational Imaging*, 2:408–423, 2016.

[35] G. Steidl, J. Weickert, T. Brox, P. Mrázek, and M. Welk. On the equivalence of soft wavelet shrinkage, total variation diffusion, total variation regularization, and sides. *SIAM Journal on Numerical Analysis*, (2):686–713, 2004.

[36] A. M. Teodoro, J. M. Bioucas-Dias, and M. A. Figueiredo. A convergent image fusion algorithm using scene-adapted Gaussian-mixture-based denoising. *IEEE Transactions on Image Processing*, 28(1):451–463, 2018.

[37] E. Vorontsov, C. Trabelsi, S. Kadoury, and C. Pal. On orthogonality and learning recurrent networks with long term dependencies. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3570–3578. JMLR. org, 2017.

[38] Z. Wen and W. Yin. A feasible method for optimization with orthogonality constraints. *Mathematical Programming*, 142(1–2):397–434, 2013.

[39] S. Wisdom, T. Powers, J. Hershey, J. Le Roux, and L. Atlas. Full-capacity unitary recurrent neural networks. In *NIPS*, 2016.