Efficient Proximal Mapping of the 1-path-norm of Shallow Networks

Fabian Latore [1] **Paul Rolland** [1] **Nadav Hallak** [1] **Volkan Cevher** [1]

Abstract

We demonstrate two new important properties of the 1-path-norm of shallow neural networks. First, despite its non-smoothness and non-convexity it allows a closed form proximal operator which can be efficiently computed, allowing the use of stochastic proximal-gradient-type methods for regularized empirical risk minimization. Second, when the activation functions is differentiable, it provides an upper bound on the Lipschitz constant of the network. Such bound is tighter than the trivial layer-wise product of Lipschitz constants, motivating its use for training networks robust to adversarial perturbations. In practical experiments we illustrate the advantages of using the proximal mapping and we compare the robustness-accuracy trade-off induced by the 1-path-norm, L1-norm and layer-wise constraints on the Lipschitz constant (Parseval networks).

1. Introduction

Neural networks are the backbone of contemporary applications in machine learning and related fields, having huge influence and significance both in theory and practice. Among the most important and desirable attributes of a trained network are robustness and sparsity. Robustness, is often defined as stability to adversarial perturbations, such as in supervised classification methods. The apparent brittleness of neural networks to adversarial attacks in this context has been considered in the literature for some time, see e.g., (Biggio et al., 2013; Szegedy et al., 2013; Madry et al., 2018) and references therein.

A fundamental question in this regard is how to measure robustness, or more importantly, how to encourage it. One prominent approach supported by theory and practice (Raghunathan et al., 2018; Cisse et al., 2017), is to use the Lipschitz constant of the network function to quantize robustness, and regularization to encourage it.

This approach is also supported theoretically with generalization bounds in terms of the layer-wise product of spectral norms (Bartlett et al., 2017; Miyato et al., 2018), which particularly upper-bounds the Lipschitz constant. However, a recent empirical study (Jiang* et al., 2020) has found in practice a negative *correlation* of this measure with generalization. This casts doubts on its usefulness and signals the fact that it is a rather loose upper bound for the Lipschitz constant (Latorre et al., 2020).

Current methods that compute upper bounds on the Lipschitz constant of neural networks can be roughly classified into two classes: (i) the class of *product bounds*, comprising all upper bounds obtained by the multiplication of layer-wise matrix norms; and, (ii) the class of convex-optimization-based bounds, which addresses the network as a whole entity (Raghunathan et al., 2018; Fazlyab et al., 2019; Latorre et al., 2020).

A trade-off between computational complexity and quality of the upper bound seems apparent. An ideal bound would achieve a balance between both properties: it should provide a good estimate of the constant while being fast and easy to minimize with iterative first-order algorithms.

Recently, the *path-norm* of the network (Neyshabur et al., 2015) has emerged as a complexity measure that is highly-correlated with generalization (Jiang* et al., 2020). Thus, its use as a regularizer holds an increasing interest for researchers in the field.

Despite existing generalization bounds (Neyshabur et al., 2015), our understanding of the optimization aspects of the path-norm-regularized objective is lacking. Jiang* et al. (2020) refrained from using automatic-differentiation methods in this case because, as they argue, the optimization could fail, thus providing no conclusion about its qualities.

It is then natural to ask: *how do we properly optimize the path-norm-regularized objective with theoretical guarantees? What conclusions can we draw about the robustness and sparsity of path-norm-regularized networks?* We focus on the *1-path-norm* and provide partial answers to those questions, further advancing our understanding of this measure. Let us summarize our main contributions:

[*]Equal contribution [1]Learning, information and optimization systems laboratory (LIONS), EPFL, Switzerland. Correspondence to: Fabian Latorre <fabian.latorre@epfl.ch>.

Proceedings of the 37^{th} International Conference on Machine Learning, Vienna, Austria, PMLR 119, 2020.

Optimization. We show a striking property of the 1-path-norm, that makes it a strong candidate for explicit regularization: despite its non-convexity, it admits an efficient *proximal mapping* (Algorithm 3). This allows the use of proximal-gradient type methods which are, as of now, the only first-order optimization algorithms to provide guarantees of convergence for composite non-smooth and non-convex problems (Bolte et al., 2013).

Indeed, automatic differentiation modules of popular deep learning frameworks like PyTorch (Paszke et al., 2019) or TensorFlow (Abadi et al., 2015) may not compute the correct gradient for compositions of non-smooth functions, at points where these are differentiable (Kakade & Lee, 2018; Bolte & Pauwels, 2019). Our proposed optimization algorithm avoids such issue altogether by using differentiable activation functions like ELU (Clevert et al., 2015) and our novel proximal mapping of the 1-path-norm.

Upper bounds. We show that the *1-path-norm* (Neyshabur et al., 2015) achieves a sweet spot in the computation-quality trade-off observed among upper bounds of the Lipschitz constant: it has a simple closed formula in terms of the weights of the network, and it provides an upper bound on the (ℓ_∞, ℓ_1)-Lipschitz constant (cf., Theorem 1), which is always better than the product bound.

Sparsity. Neural network regularization schemes promoting sparsity in a principled way are of great interest in the growing field of *compression* in Deep Learning (Han et al., 2016; Cheng et al., 2017).

Our analysis provides a formula (cf. Lemma 4) for choosing the *strength* of the regularization, which enforces a desired bound on the sparsity level of the iterates generated by the proximal gradient method. This is a suprising, yet intuitive, result, as the sparsity-inducing properties of non-smooth regularizers have been observed before in convex optimization and signal processing literature, see e.g., (Bach et al., 2012; Eldar & Kutyniok, 2012).

Experiments. In section 7, we present numerical evidence that our approach (i) converges faster and to lower values of the objective function, compared to plain SGD; (ii) generates sparse iterates; and, (iii) the magnitude of the regularization parameter of the 1-path-norm allows a better accuracy-robustness trade-off than the common ℓ_1 regularization or constraints on layer-wise matrix norms.

2. Problem Setup

We consider the so-called shallow neural networks with n hidden neurons and p outputs $h : \mathbb{R}^m \to \mathbb{R}^p$ given by

$$h_{V,W}(x) = V^T\sigma(Wx), \tag{1}$$

where $V \in \mathbb{R}^{n\times p}, W \in \mathbb{R}^{n\times m}$ and $\sigma : \mathbb{R} \to \mathbb{R}$ is some differentiable activation function with derivative globally bounded between zero and one. This condition is satisfied, for example, by the ELU or softplus activation functions. To control the robustness of the network to perturbations of its input x, we want to regularize training using its Lipschitz constant as a function of the weights V and W.

To properly define this constant, we utilize the ℓ_∞-norm for the input space, and the ℓ_1-norm for the output space. Exact computation of such constant is a hard task. A simple and easily computable upper bound can be derived by the product of the layer-wise Lipschitz constants, however, it can be quite loose.

We derive an improved upper bound which is still easy to compute. In the following, we denote with $\|W\|_\infty$ the operator norm of a matrix W with respect to the ℓ_∞ norm for both input and output space; it is equal to the maximum ℓ_1-norm of its rows. We denote with $\|V\|_{\infty,1}$ the operator norm of the matrix V with respect to the ℓ_∞ norm in input space and ℓ_1-norm in output space; it is equal to the sum of the ℓ_1 norm of its columns.

Theorem 1. *Let $h_{V,W}(x) = V^T\sigma(Wx)$ be a network such that the derivative of the activation σ is globally bounded between zero and one. Choose the ℓ_∞- and ℓ_1-norm for input and output space, respectively. The Lipschitz constant of the network, denoted by $L_{V,W}$ is bounded as follows:*

$$L_{V,W} \leq \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{p} |W_{ij}V_{ik}| \leq \|V^T\|_{\infty,1}\|W\|_\infty \tag{2}$$

The proof is provided in appendix A. The term in the middle of inequality (2) belongs to the family of *path-norms*, introduced in Neyshabur et al. (2015, Eq. (7)). Throughout, we refer to it as the *1-path-norm*.

Notice that although the path-norm and layer wise product bounds can be equal, this only happens in the following worst case: For the weight matrix in the first layer, the 1-norms of every row are equal. Thus, in practice the bounds can differ drastically.

Remark 1. *In practice, one might want to regularize each ouput of the network in a different way according to some weighting scheme (Raghunathan et al., 2018). Precisely, the 1-path-norm of the network is equal to the sum (with equal weight) of the 1-path-norm of each output. A weighted version of the 1-path-norm can be defined to account for such a weighting scheme. All our results can be adapted to this scenerio, with minor changes.*

We now turn to the task of minimizing an empirical risk functional regularized by the improved upper bound on the Lipschitz constant given in (2):

$$\min_{V,W} \mathbb{E}_{(x,y)}[\ell(h_{V,W}(x), y)] + \lambda \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{p} |W_{ij}V_{ik}| \tag{3}$$

The objective function in problem (3) is composed of an expectation of a nonconvex smooth loss, and a nonconvex nonsmooth regularizer, meaning that it is essentially a composite problem (cf. (Beck, 2017, Ch. 10)). That is, the objective function (3) can be cast as use these notation hereafter)

$$\min_{V,W} \mathcal{F}(V,W) \equiv f(V,W) + \lambda g(V,W), \tag{4}$$

where f is a nonconvex continuously differentiable function, and $g(V,W) = \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{p} |W_{ij}V_{ik}|$ is a continuous, nonconvex, nonsmooth, function. We assume that the objective function is bounded below, i.e., $\inf \mathcal{F} := \mathcal{F}_* > 0$.

A natural choice for a scheme to obtain critical points for (4) is the proximal-gradient framework. However, for a nonconvex g, solving the proximal gradient problem is a hard problem in general. In Section 4 we develop a method that computes the proximal gradient with respect to g efficiently.

To streamline our approach and techniques in a compact and user-friendly manner, we will illustrate the majority of our results and proofs via the particular single-output scenario in which h and g are reduced to

$$h_{v,W}(x) = v^T\sigma(Wx), \; g(v,W) \equiv \lambda \| \operatorname{vec}(\operatorname{Diag}(v)W)\|_1.$$

The multi-output case follows from the same techniques and insights, however, requires more tedious computations and arguments, on which we elaborate in Section 5, and detail in the appendix.

3. The Prox-Grad Method

Assume that f has a Lipschitz continuous gradient with Lipschitz constant $L > 0$, that is

$$\|\nabla f(z) - \nabla f(u)\| \leq L\|z-u\|, \qquad \forall z,u \in \mathbb{R}^n.$$

The prox-grad method is described by Algorithm 1; since g is nonconvex, the prox in (5) can be a set of solutions.

Algorithm 1 Prox-Grad Method

Input: $z^0 \equiv \operatorname{vec}(V^0, W^0) \in \mathbb{R}^{p\cdot n + n\cdot m}$, $\{\eta^k\}_{k\geq 0}$.

```
for k = 0, 1, ... do
    Compute G^k = ∇f(z^k)
    z^{k+1} ← prox_{η^k g}(z^k − η^k G^k)
end for
```

Theoretical guarantees for the prox-grad method with respect to a nonconvex regularizer were established by (Bolte et al., 2013) (for a more general prox-grad type scheme).

Theorem 2 (Convergence guarantees)**.** *Let $\{z^k\}_{k\geq 0}$ be a sequence generated by Algorithm 1 with $\{\eta^k\}_{k\geq 0} \subseteq (0, 1/L)$. Then*

1. *Any accumulation point of $\{z^k\}_{k\geq 0}$ is a critical point of (4).*
2. *If f satisfies the Kurdyka-Lojasiewicz (KL) property, then $\{z^k\}_{k\geq 0}$ converges to a critical point.*
3. *Suppose that η_k is chosen such that there exists $c > 0$ such that $\sum_{k=0}^{K} \frac{1}{\eta_k} \geq cK$ for any $K \geq 0$. Then*

$$\min_{k=0,\ldots,K} \|z^{k+1} - z^k\|_2 \leq \sqrt{\frac{2(\mathcal{F}(z^0) - \mathcal{F}_*)}{(c-L)K}}.$$

Proof. See Section B in the appendix. □

Remark 2 (On KL related convergence rate)**.** *A convergence rate result under the KL property can be derived with respect to the desingularizing function; see (Bolte et al., 2013) for additional details.*

Remark 3 (On the stochastic prox-grad method)**.** *The literature does not provide any theoretical guarantees for a prox-grad type method that uses stochastic gradients (i.e., replacing G^k with an approximation of $\nabla f(z^k)$) under our setting. Recently, (Metel & Takeda, 2019) studied stochastic prox-grad methods, however, their results rely on the assumption that the regularizer is Lipschitz continuous, which is not satisfied by our robust-sparsity regularizer.*

4. Computing the Proximal Mapping

Throughout this section we assume the single-output setting. The path-norm regularizer we propose is a nonconvex nonsmooth function, suggesting that the prox-grad scheme in Algorithm 1 is intractable.

In this section we will not only prove that in fact *it is tractable* in the single output case, but that it can also be *implemented efficiently* with complexity of $O(m\log(m))$; we prove the stated in detail in Section C, and provide here a concise version.

Denote the given pair (x, Y) by z. The proximal mapping with respect to λg at z is defined as

$$\operatorname{prox}_{\lambda g}(z) = \operatorname*{argmin}_{u} \lambda g(u) + \frac{1}{2}\|\operatorname{vec}(u-z)\|_2^2. \tag{5}$$

By the choice of g, the objective function in (5) is coercive and lower bounded, implying that there exists an optimal solution (cf. (Beck, 2014, Thm. 2.32)).

Remark 4. *The derivations in this section can be easily adapted and used with adaptive gradient methods like Adagrad (Duchi et al., 2011), by a careful handling of the per-coordinate scaling coefficients.*

Lemma 1 (Well-posedness of (5))**.** *For any $\lambda \geq 0$ and any (u, z), the problem (5) has a global optimal solution.*

Additionally, we have that (5) is separable with respect to the i-th entry of the vector v and the i-th row of the matrix W, meaning that problem (5) can be solved in a distributed manner by applying the same solution procedure coordinate-wise for v and row-wise for W. In light of this, let us consider the i-th row related problem

$$\min_{v,w\in\mathbb{R}\times\mathbb{R}^m} \frac{1}{2}(v-x)^2+\frac{1}{2}\sum_{j=1}^{m}(w_j-y_j)^2+\lambda|v|\sum_{j=1}^{m}|w_j|. \tag{6}$$

The signs of the elements of the decision variables in (6) are determined by the signs of (x,y), and consequently, the problem in (6) is equivalent to

$$\min_{v,w\in\mathbb{R}_+\times\mathbb{R}_+^m} \frac{1}{2}(v-|x|)^2+\frac{1}{2}\sum_{j=1}^{m}(w_j-|y_j|)^2+\lambda v\sum_{j=1}^{m}w_j. \tag{7}$$

Lemma 2. *Let $(v^*,w^*)\in\mathbb{R}_+\times\mathbb{R}_+^n$ be an optimal solution of (7). Then $(sign(x)\cdot v^*, sign(y)\circ w^*)$ is an optimal solution of problem (6).*

Denote

$$h_\lambda(v,w;x,y)=\frac{1}{2}(v-|x|)^2+\frac{1}{2}\sum_{j=1}^{m}(w_j-|y_j|)^2+\lambda v\sum_{j=1}^{m}w_j.$$

Although h_λ is nonconvex, we will show that a global optimum to (7) can be obtained efficiently by utilizing several tools, the first being the first-order optimality conditions of (7) (cf. (Beck, 2014, Ch. 9)) given below.

Lemma 3 (Stationarity conditions)**.** *Let $(v^*,w^*)\in\mathbb{R}_+\times\mathbb{R}_+^m$ be an optimal solution of (7) for a given $(x,y)\in\mathbb{R}\times\mathbb{R}^m$. Then*

$$w_j^*=\max\{0,|y_j|-\lambda v^*\} \text{ for any } j=1,2,\dots,m,$$
$$v^*=\max\left\{0,|x|-\lambda\sum_{j=1}^{m}w_j^*\right\}.$$

A key insight following Lemma 3 is that: the elements of any solution to (7), satisfy a monotonic relation in magnitude, correlated with the magnitude of the elements of y; this is formulated by the next corollary.

Corollary 1. *Let $(v^*,w^*)\in\mathbb{R}_+\times\mathbb{R}_+^m$ be an optimal solution of (7) for a given $(x,y)\in\mathbb{R}\times\mathbb{R}^m$. Then*

1. *The vector w^* satisfies that for any $j,l\in\{1,2,\dots,m\}$ it holds that $w_j^*\geq w_l^*$ only if $|y_j|\geq|y_l|$.*
2. *Let $\bar{y}$ be the sorted vector of y in descending magnitude order. Suppose that $v^*>0$ and let $s=|\{j: sw_j^*>0\}|$. Then,*

$$v^*=\frac{1}{1-s\lambda^2}\left(|x|-\lambda\sum_{j=1}^{s}|\bar{y}_j|\right), \tag{8}$$

where we use the convention that $\sum_{j=1}^{0}|\bar{y}_j|=0$.

Proof. The first part follows trivially from the stationarity conditions on w^* given in Lemma 3.

From the first part and the conditions in Lemma 3 we have that $\sum_{j=1}^{m}w_j^*=\sum_{j=1}^{s}|\bar{y}_j|-\lambda s v^*$. Plugging the latter to the stationarity condition on v^* (given in Lemma 3) then implies the required. □

Remark 5. *Corollary 1 implies that the solution vector w^* is ordered in the same way as $|y|$. Thus, the s non-zero entries of w^* are precisely the ones corresponding with the s largest entries of $|y|$.*

Without loss of generality, we assume hereafter that the input y is already sorted in decreasing order, such that the s non-zero entries of w^* are always the first s entries.

To supplement the results above, we now show that we can actually upper-bound the sparsity level of the prox-grad output by adjusting the value of λ.

Lemma 4 (Sparsity bound)**.** *Let $(v^*,w^*)\in\mathbb{R}_+\times\mathbb{R}_+^m$ be an optimal solution of (7) for a given $(x,y)\in\mathbb{R}\times\mathbb{R}^m$. Suppose that $v^*>0$ (i.e., non-trivial),[1] and denote $S=\{j: w_j^*>0\}$. Then $|S|\leq\lambda^{-2}$.*

Proof. Since (v^*,w^*) is an optimal solution of (7) and the objective function in (7) is twice continuously differentiable, (v^*,w^*) satisfies the second order necessary optimality conditions (Bertsekas, 1999, Ex. 2.1.10). That is, for any $d\in\mathbb{R}\times\mathbb{R}^m$ satisfying that $(v^*,w^*)+d\in\mathbb{R}_+\times\mathbb{R}_+^m$ and $d^T\nabla h_\lambda(v^*,w^*;x,y)=0$ it holds that

$$d^T\nabla^2 h_\lambda(v^*,w^*;x,y)d=d^T\begin{pmatrix}1&\lambda&\cdots&\lambda\\ \lambda&1&0&0\\ \vdots&0&\ddots&0\\ \lambda&0&0&1\end{pmatrix}d\geq 0,$$

where the first row/column corresponds to v and the others correspond to w. Noting that for any $j\in S$ it holds that $\frac{\partial h_\lambda}{\partial w_j}(v^*,w^*;x,y)=0$, we have that the submatrix of $\nabla^2 h_\lambda(v^*,w^*;x,y)$ containing the rows and columns corresponding to the positive coordinates in (v^*,w^*) must be positive semidefinite.

Since the the minimal eigenvalue of this submatrix equals $1-\lambda\sqrt{|S|}$, we have that $\lambda^{-2}\geq|S|$. □

Moreover, the function h_λ is monotonically decreasing in the sparsity level, which implies that instead of exhaustively checking the value of h_λ for any sparsity level, we can

[1] We will call an optimal solution trivial if $v^*=0$.

employ a binary search. Denote for any $s \in \{0, \dots, m\}$ the $m+1$ possible solutions:

$$v^{(s)} = \frac{1}{1 - s\lambda^2} \left(|x| - \lambda \sum_{j=1}^{s} |y_j| \right)$$
$$w_j^{(s)} = |y_j| - \lambda v^{(s)} \;\text{ for } j \in [s], \;\text{ and } w_j^{(s)} = 0 \text{ otherwise.}$$

Lemma 5. *Let $\bar{s} = \lfloor \lambda^{-2} \rfloor$. For all integer $s \in \{2, 3, \dots, \bar{s}\}$, we have that*

$$h_\lambda(v^{(s)}, w^{(s)}; x, y) < h_\lambda(v^{(s-1)}, w^{(s-1)}; x, y). \tag{9}$$

Lemma 5 follows from algebraic considerations, and thus its proof is deferred to Section C. Its substantial implication is the following.

Corollary 2. *Suppose that there exists a non-trivial optimal solution of (7). Denote $\bar{s} = \min(\lfloor \lambda^{-2} \rfloor, m)$ and let*

$$s^* = \max \left\{ s \in \{0, \dots, \bar{s}\} : v^{(s)}, w_s^{(s)} > 0 \right\}.$$

Then $(v^{(s^*)}, w^{(s^*)})$ is an optimal solution of (7).

Note that since, by definition, the s first entries of the vector $w^{(s)}$ are ordered in decreasing order, the constrained $w_s^{(s)} > 0$ ensures that the full vector $w^{(s)}$ has exactly s nonzero entries, which are all strictly positive.

The final ingredient required for designing an efficient algorithm is the following monotone property of the feasibility criterion in problem (2):

Lemma 6. *For any $k \in [\bar{s}]$, we have*

$$v^{(k)} > 0, w^{(k)} > 0 \Rightarrow v^{(i)} > 0, w^{(i)} > 0, \;\; \forall i < k.$$

This property, whose proof is also deferred to Section C, implies that the optimal sparsity parameter s^* can be efficiently found using a binary search approach.

We conclude this section by combining all the ingredients above to develop Algorithm 2, and to prove that it yields a solution to (5).

Theorem 3 (Prox computation)**.** *Let $(v_i^*, W_{i,:}^*)$ be the output of Algorithm 2 with input $x_i, Y_{i,:}, \lambda$, assuming that each $Y_{i,:}$ is sorted in decreasing magnitude order. Then (v^*, W^*) is a solution to (5).*

Proof. For any $i = 1, 2, \dots, n$, let $(v_i^*, W_{i,:}^*)$ be the output of Algorithm 2 with input $x_i, Y_{i,:}, \lambda$. We will show that (v^*, W^*) is an optimal solution to (5) by arguing that Algorithm 2 chooses the point with the smallest h_λ value out of a feasible set of solutions containing an optimal solution of (5).

Algorithm 2 Single-output robust-sparse proximal mapping

Input: $x \in \mathbb{R}$, $y \in \mathbb{R}^m$ sorted in decreasing magnitude order, $\lambda > 0$.

$v^* = 0, w^* = |y|$
$s_{\text{lb}} \leftarrow 0$, $s_{\text{ub}} \leftarrow \min(\lfloor \lambda^{-2} \rfloor, m)$, $s \leftarrow \lceil (s_{\text{lb}} + s_{\text{ub}})/2 \rceil$
while $s_{\text{lb}} \neq s_{\text{ub}}$ **do**
  $v^{(s)} = \frac{1}{1 - s\lambda^2} \left(|x| - \lambda \sum_{j=1}^{s} |y_j| \right)$
  $w_j^{(s)} = |y_j| - \lambda v^{(s)}, j \in [s]$ and $w_j^{(s)} = 0$ otherwise
  if $v > 0, w_s > 0$ **then**
    $s_{\text{lb}} \leftarrow s$, $s \leftarrow \lceil (s_{\text{lb}} + s_{\text{ub}})/2 \rceil$
    $(v^*, w^*) \leftarrow (v, w)$
  else if $v < 0$ **then** $s_{\text{ub}} \leftarrow s$, $s \leftarrow \lceil (s_{\text{lb}} + s_{\text{ub}})/2 \rceil$
  else $s_{\text{lb}} \leftarrow s$, $s \leftarrow \lceil (s_{\text{lb}} + s_{\text{ub}})/2 \rceil$
  end if
end while
return $(\text{sign}(x) \cdot v^*, \text{sign}(y) \circ w^*)$

For simplicity, and without loss of generality, let us consider the one-coordinate-one-row case, that is, $(v_i^*, W_{i,:}^*) \equiv (v^*, w^*)$, $(x_i, Y_{i,:}) \equiv (x, y)$; the proof for the general case is a trivial replication.

By Lemma 2 it is sufficient to prove that $(|v^*|, |w^*|)$ is an optimal solution of (7), as this will imply the optimality of (v^*, w^*); Recall that Lemma 1 establishes that there exists an optimal solution to (7).

If the trivial solution is the only optimal solution to (7), then obviously it will be the output of Algorithm 2. Otherwise, the point described in Corollary 2 is an optimal solution. Assume that Algorithm 2 returned the point $(v^{(s_{\text{out}})}, w^{(s_{\text{out}})})$ for some $s_{\text{out}} \in [\bar{s}]$, meaning in particular that $(v^{(s_{\text{out}})}, w^{(s_{\text{out}})}) > 0$. By definition, $s^* \geq s_{\text{out}}$. If $s_{\text{out}} < s^*$, then at some $s < s^*$ we had that $v^{(s)} < 0$. Since the value of $v^{(i)}$ is monotonic decreasing in the sparsity level, this implies that $v^{(s^*)} < 0$, which is a contradiction.

Hence, if Algorithm 2 did not return the trivial solution, then $(v^*, w^*) = (v^{(s^*)}, w^{(s^*)})$, meaning that $(\text{sign}(x) \cdot v^*, \text{sign}(y) \circ w^*)$ is a solution to (5). □

Time complexity of Algorithm 2 In the worst case where $m \leq \lambda^{-2}$, the number of searches for finding s^* is at most $\log_2(m)$. Each search requires to compute $v^{(s)}$, and in particular $\sum_{j=1}^{s} |y_j|$, as well as $w_j^{(s)}$, $j = 1, \dots, s$, each taking $\mathcal{O}(s)$ steps. Thus, the overall loop complexity is $\mathcal{O}(m)$.

Moreover, this algorithm assumes that the input vector y is already sorted in decreasing magnitude order. This can easily be achieved by a sorting procedure in time $\mathcal{O}(m \log m)$.

5. Multi-Output

The efficient computation of the robust-sparse proximal mapping we derived for the single-output scenario will now be generalized to the multi-output case. Although we use similar arguments and insights, the analysis is much more complicated and requires more delicate and advanced treatment. Due to the tedious computations that accompany the analysis, the proofs are deferred to appendix D.

When the network has multiple-output, the proximal operator $\text{prox}_{\lambda g}(X, Y)$ can be written as the solution set of

$$\min_{V,W} \|V - X\|_F + \|W - Y\|_F + 2\lambda \sum_{i=1}^{n} \sum_{j=1}^{m} \sum_{k=1}^{p} |W_{ij} V_{ik}|, \tag{10}$$

where $V \in \mathbb{R}^{n\times p}$ and $W \in \mathbb{R}^{n\times m}$. As in the single-output case, we observe that the proximal mapping (10) is separable with respect to the i-th rows of the matrices V and W, and that the signs of the decision variables are determined by the signs of (X, Y). Therefore, it is enough to consider the problem related to the i-th row of V, denoted as x, and i-th row of W, denoted as y, i.e.,

$$\min_{v,w\in\mathbb{R}_+^p\times\mathbb{R}_+^m} h_\lambda(v, w; x, y), \tag{11}$$

where we redefine $h_\lambda(v, w; x, y)$ to include the multi-output case: $h_\lambda(v, w; x, y) = \frac{1}{2}\sum_{k=1}^{p}(v_k - |x_k|)^2 + \frac{1}{2}\sum_{j=1}^{m}(w_j - |y_j|)^2 + \lambda \sum_{k=1}^{p} v_k \sum_{j=1}^{m} w_j$. To improve readability, we will abuse notation and just write $h_\lambda(v, w)$, assuming that (x, y) are understood from context.

Using the same observations we exploited to enumerated all stationary points of the proximal mapping in the single-output setup, we can identify the stationary points depending on the number of non zero elements of v and w.

Lemma 7. *Let $(v^*, w^*) \in \mathbb{R}_+^p \times \mathbb{R}_+^m$ be an optimal solution of (19) for a given $(x, y) \in \mathbb{R} \times \mathbb{R}^m$. Then*

1. *The vector w^* satisfies that for any $j, l \in [m]$ it holds that $w_j^* \geq w_l^*$ only if $|y_j| \geq |y_l|$.*
2. *The vector v^* satisfies that for any $k, l \in [p]$ it holds that $v_k^* \geq v_l^*$ only if $|x_k| \geq |x_l|$.*
3. *Let $\bar{x}, \bar{y}$ be the sorted vectors in descending magnitude order of x and y respectively. Let $s_v = |\{k : v_k^* > 0\}|$ and $s_w = |\{j : w_j^* > 0\}|$. If $v^*, w^* \neq 0$, then we have that for any $k \in \{k : v_k^* > 0\}$ and $j \in \{j : w_j^* > 0\}$, it holds that $v^* = v^{(s_v,s_w)}$ and $w^* = w^{(s_v,s_w)}$ where*

$$v_k^{(s_v,s_w)} = |x_k| + \mu \left(\lambda^2 s_w \sum_{l=1}^{s_v} |\bar{x}_l| - \lambda \sum_{j=1}^{s_w} |\bar{y}_j| \right) \tag{12}$$

$$w_j^{(s_v,s_w)} = |y_j| + \mu \left(\lambda^2 s_v \sum_{l=1}^{s_w} |\bar{y}_l| - \lambda \sum_{k=1}^{s_v} |\bar{x}_k| \right) \tag{13}$$

and $\mu = (1 - s_v s_w \lambda^2)^{-1}$.

From the two first points in Lemma 7, the argument in Remark 5 is also valid in the multi-output case, and so we assume hereafter that the input vectors x, y are sorted in decreasing magnitude order.

Using the second order stationary conditions, we can generalize our sparsity bound in the single-output scenario, given in Lemma 4, to an upper bound on the product of the sparsities of the solutions based on the value of λ; indeed, $s_v = 1$ yields the bound in Lemma 4.

Lemma 8 (Sparsity bound)**.** *Let $(v^*, w^*) \in \mathbb{R}_+^p \times \mathbb{R}_+^m$ be an optimal solution of (11) for a given $(x, y) \in \mathbb{R}^p \times \mathbb{R}^m$. Denote $s_v = |\{j : w_j^* > 0\}|$ and $s_w = |\{j : w_j^* > 0\}|$. Then $s_v s_w \leq \lambda^{-2}$.*

A possible algorithm for computing this proximal mapping would thus be to compute the value of $h_\lambda\left(v^{(s_v,s_w)}, w^{(s_v,s_w)}\right)$ for each pair of sparsities $(s_v, s_w) \in \{0, \ldots, p\} \times \{0, \ldots, m\}$ satisfying $s_v s_w \leq \lambda^{-2}$ and return the pair achieving the smallest value.

However, such an approach would be computationally inefficient. In order to avoid computing the value of h_λ at each pair, we show the following monotonicity property of h_λ in the sparsity levels, which generalizes the same property in the single-output case.

Lemma 9. *Given $(x, y) \in \mathbb{R}^p \times \mathbb{R}^m$, for all $s_v, s_w \in \{0, \ldots, p\} \times \{0, \ldots, m\}$ satisfying $s_v s_w < \lambda^{-2}$, we have*

$$h_\lambda(v^{(s_v,s_w)}, w^{(s_v,s_w)}) < h_\lambda(v^{(s_v,s_w-1)}, w^{(s_v,s_w-1)}),$$
$$h_\lambda(v^{(s_v,s_w)}, w^{(s_v,s_w)}) < h_\lambda(v^{(s_v-1,s_w)}, w^{(s_v-1,s_w)}).$$

Moreover, the feasibility criterion $v \geq 0, w \geq 0$ also has a monotonic property:

Lemma 10. *Let $(k, l) \in [p] \times [m]$ be such that $kl \leq \lambda^{-2}$. If $v^{(k,l)} \geq 0$ and $w^{(k,l)} \geq 0$, then, $v^{(i,j)} \geq 0$ and $w^{(i,j)} \geq 0$ $\forall i = 1, \ldots, k$ and $\forall j = 1, \ldots, l$.*

To properly address the complications arising from handling two intertwining sparsity levels at the same time, we introduce the notion of *maximal feasibility boundary (MFB)* which acts a frontier of possible sparsity levels.

Definition 1 (Maximal feasibility boundary)**.** *We say that a sparsity pair $(s_v, s_w) \in \{0, \ldots, p\} \times \{0, \ldots, m\}$ is on the maximal feasibility boundary (MFB) if incrementing either*

s_v or s_w results with a non-stationary point. That is, if both of the following conditions hold:

- $v^{(s_v+1,s_w)}_{s_v+1} < 0$ *or* $w^{(s_v+1,s_w)}_{s_w} < 0$ *or* $(s_v+1)s_w > \lambda^{-2}$,
- $v^{(s_v,s_w+1)}_{s_v} < 0$ *or* $w^{(s_v,s_w+1)}_{s_w+1} < 0$ *or* $s_v(s_w+1) > \lambda^{-2}$.

The efficient computation of the multi-output robust-sparse proximal mapping is based on the fact that we only need to compute the value of h_λ for sparsity levels that are at the frontier of the MFB. This allows us to find the optimal sparsity in time $\mathcal{O}(p+m)$, improving upon the $\mathcal{O}(pm)$ complexity of the exhaustive search. Algorithm 3 implements the above by employing a binary search type procedure defined in Algorithm 5 to calculate the MFB.

Algorithm 3 Multi-output robust-sparse proximal mapping

Input: $x \in \mathbb{R}^p$, $y \in \mathbb{R}^m$ ordered in decreasing magnitude order, $\lambda > 0$.

```
Employ Algorithm 5: Find the set of sparsity pairs S = {(s_v, s_w)} that are on the MFB
h_opt ← ∞
for (s_v, s_w) ∈ S do
    Compute v^(s_v,s_w) and w^(s_v,s_w) as given in equations (12), (13)
    if h_λ(v^(s_v,s_w), v^(s_v,s_w); |x|, |y|) < h_opt then
        h_opt = h_λ(v^(s_v,s_w), v^(s_v,s_w); |x|, |y|)
        v* ← v^(s_v,s_w), w* ← w^(s_v,s_w)
    end if
end for
return (sign(x) ∘ v*, sign(y) ∘ w*)
```

Theorem 4 (Multi-output prox computation)**.** *Let $(V^*_{:,i}, W^*_{i,:})$ be the output of Algorithm 3 with input $X_{:,i}, Y_{i,:}, \lambda$, where each $X_{:,i}, Y_{i,:}$ are sorted in decreasing magnitude order. Then (V^*, W^*) is a solution to* (5)*.*

Time complexity of Algorithm 3 It is easy to see that the maximal feasibility boundary contains at most $\min(m,p)$ pairs, and Algorithm 5 finds them all in time $\mathcal{O}(m+p)$. Then, for each such pair (s_v, s_w), we must compute $v^{(s_v,s_w)}$ and $w^{(s_v,s_w)}$ and $h_\lambda(v^{(s_v,s_w)}, w^{(s_v,s_w)})$, which takes time $\mathcal{O}(m+p)$. The total complexity of Algorithm 3 is thus $\mathcal{O}(\min(m,p)(m+p))$. In most practical application, the output layer size p can be considered $\mathcal{O}(1)$, so that the complexity of computing this proximal mapping is comparable to the complexity of computing one stochastic gradient.

6. Related Work

The path regularization approach to train neural networks can be traced back to the seminal paper by Neyshabur et al. (2015), who introduced the p-path-norm as a heuristic proxy to control the *capacity* of the network.

In this paper (cf. Theorem 1), we took a step forward by moving from heuristic explanations to rigorous arguments by establishing a new connection between the 1-path-norm and the Lipschitz constant of the network. This result also reads as a relation between the 1-path-norm and the product bound, of which variants have been found to be useful in deriving generalization bounds (Bartlett et al., 2017).

Generalization bounds in terms of the p-path-norm were also derived in (Neyshabur et al., 2015), but the question of how to methodologically exploit these as regularizers remains open; our algorithmic contribution is a first step in this direction. Additionally, issues regarding optimization with path-norm regularization were reported by Ravi et al. (2019), which examined a conditional gradient method in the context of path-norm regularization.

A growing collection of works have focused on the task of network compression, doing so via sparsity-inducing regularizers (Alvarez & Salzmann, 2016; Yoon & Hwang, 2017; Scardapane et al., 2017; Lemhadri et al., 2019). They have achieved a great level of success by setting the regularization term in an *ad-hoc* manner. In contrast, we follow a principled regularization approach with theoretical properties of generalization and robustness, and as a consequence, we are able to quantify the sparsity of the resulting networks.

Moreover, the aforementioned works only use convex regularizers for which efficient proximal mappings are available (see Bach et al. (2012) and references therein). We tackle the much harder non-convex regularization task, and derive a new method to compute the proximal mapping in this case. The merits of non-convex non-smooth regularization, and difficulties regarding their optimization, have been extensively studied in the imaging and signal sciences, see e.g., (Ochs et al., 2015) and the recent survey (Wen et al., 2018).

Layer-wise constraints or regularization with matrix-norms, which are also motivated by the product bound, have been used for robust training (Cisse et al., 2017; Tsuzuku et al., 2018) and generative models (Miyato et al., 2018). These focus on robustness with respect to the ℓ_2-norm, which requires a careful handling of operations on the singular values of the weight matrices, and does not have the extra benefit of inducing sparsity.

In section 7 we compare to this class of methods for the ℓ_∞-norm case, in which a simple rescaling of the rows in the weight matrices yields a numerically stable procedure (Duchi et al., 2008; Condat, 2016).

7. Experiments

We empirically evaluate shallow neural networks trained by regularized empirical risk minimization (4) using cross-entropy loss. In terms of the weight matrices V and W of

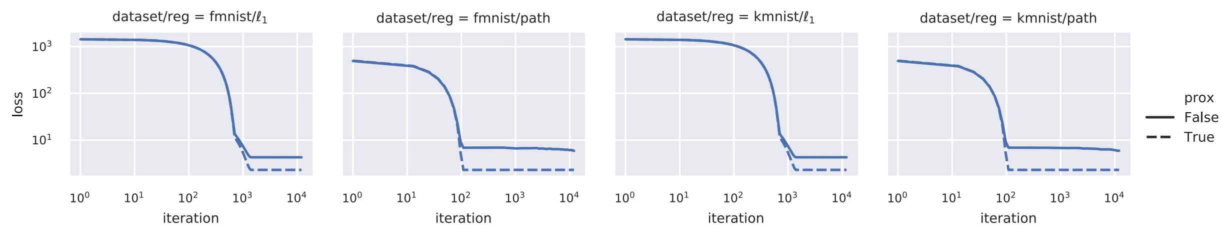

Figure 1. value of regularized cross-entropy loss across iterations.

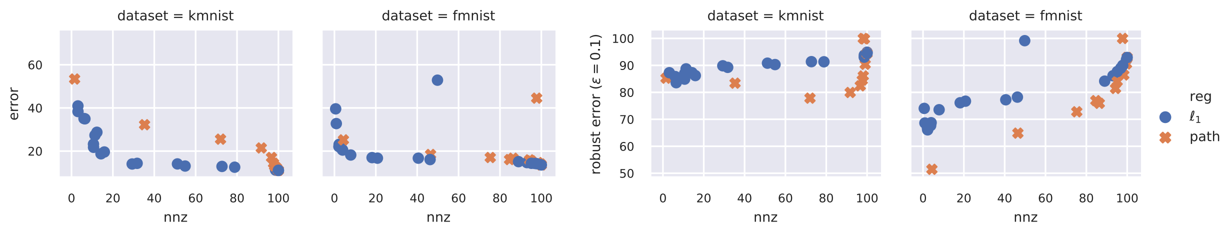

Figure 2. Misclassification test error (left) and robust test error (right) as a function of the percentage of nonzero weights.

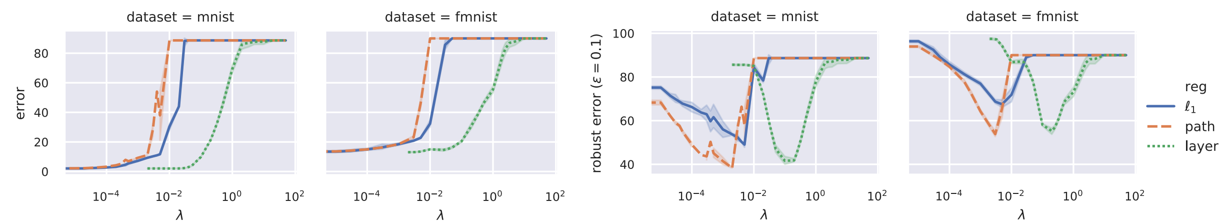

Figure 3. Misclassification test error (left) and robust test error (right) on the test set, as a function of the regularization parameter λ.

the network (1), the following regularizers are considered:

ℓ_1 regularization. We penalize the ℓ_1-norm of the parameters of the network, i.e., $g(V, W) = \|\operatorname{vec}(V)\|_1 + \|\operatorname{vec}(W)\|_1$ in the objective function (4).

1-path-norm regularization. We set $g(V, W)$ as $\sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{p}|W_{ij}V_{ki}|$ in the objective function (4).

Layer-wise regularization (Parseval Networks). we minimize the cross-entropy loss with a hard constrain on the ℓ_∞-operator-norm of the weight matrices i.e., $\|W\|_\infty \leq \lambda^{-1}$ and $\|V\|_\infty \leq \lambda^{-1}$, as described by Cisse et al. (2017). The projection on such set is achieved by projecting each row of the matrices onto an ℓ_1-ball using efficient algorithms (Duchi et al., 2008; Condat, 2016).

Remark. We will refer (incorrectly) to the training loop defined by PyTorch's SGD optimizer as *Stochastic gradient descent (SGD)* (see the discussion in section 1).

Experimental setup. Our benchmarks are the MNIST (LeCun & Cortes, 2010), Fashion-MNIST (Xiao et al., 2017) and Kuzushiji-MNIST (Clanuwat et al., 2018). For a wide range of learning rates, number of hidden neurons and regularization parameters λ, we train networks with SGD and Proximal-SGD (with constant learning rate). We do so for 20 epochs and with batch size set to 100. For each combination of parameters we train 6 networks with the default random initialization. Details and further experiments are reported in appendix E.

7.1. Convergence of SGD vs Proximal-SGD

Due to the non-differentiability of the ℓ_1- and path-norm regularizers, we expect Proximal-SGD to converge faster, and to lower values of the regularized loss, when compared to SGD. This is examined in Figure 1, where we plot the value of the loss function across iterations. For both SGD and Proximal-SGD, the loss function decays rapidly in the first few epochs. We then enter a second regime where SGD suffers from slow convergence, whereas Proximal-SGD continues to reduce the loss at a fast rate. At the end of the 20-epochs, Proximal-SGD consistently achieves a lower value of the loss.

An advantage of Proximal-SGD over plain SGD is that the proximal mappings of both the ℓ_1- and path-norm regularizers can set many weights to *exactly* zero. In Figure 2 we plot the average error and robust test error obtained, as a fuction of the sparsity of the network. Compared to ℓ_1 regularization, the sparsity pattern induced by the 1-path-norm correlates with the robustness to a higher degree. As a drawback, it appears that in more difficult datasets like KMNIST, the 1-path-norm struggles to obtain good accuracy and sparsity simultaneously.

7.2. The robustness-accuracy trade-off

The relation between the Lipschitz constant of a network and its robustness to adversarial perturbations has been extensively studied in the literature. In Theorem 1 we have shown that the 1-path-norm of a single-output network is a tighter upper bound of its Lipschitz constant, compared to the corresponding product bound.

To the best of our knowledge, the ℓ_1-norm regularizer only provides an upper boud on the already loose product bound (Neyshabur et al., 2015, Eq. (4)), which makes it less attractive as a regularizer, despite its sparsity-inducing properties. Hence, the 1-path-norm regularizer is, in theory, a better proxy for robustness than the other regularization schemes.

subsection E.2 shows the misclassification error on clean and adversarial examples as a function of λ, and corresponds to the learning rate minimizing the error on clean samples. The adversarial perturbations were obtained by PGD (Madry et al., 2018).

Any training procedure which promotes robustness of a classifier may decrease its accuracy, and this effect is consistently observed in practice (Tsipras et al., 2019). Hence, the merits of a regularizer should be measured by how efficiently it can trade-off accuracy for robustness. We observe that for all three regularization schemes, there exists choices of λ that attain the best possible error on clean samples.

On the other hand, the error obtained by the ℓ_1 regularization degrades significantly. The layer-wise and 1-path-norm regularization achieve a noticeably low error on adversarial examples. Comparing the latter schemes, the 1-path-norm regularization shows only a slight advantage over the layer-wise methods, which merits further investigation.

8. Future Work: Multilayer Extension

A natural extension of our approach is to apply path regularization to multi-layered networks. Since the number of paths is potentially huge, this scenario requires more sophisticated treatment, and hence left for future research. Nonetheless, a trivial extension of our approach is to divide a multi-layered network into pairs of consecutive layers, and apply our method in a sensible manner. We now describe this approach, to complement the theory.

Precisely, assume that the network contains an even number of layers. For some lists of matrices $\mathbf{V} = (V^1, \dots, V^k)$ and $\mathbf{W} = (W^1, \dots, W^k)$ of appropriate sizes, the network can be written as a composition of activation functions and shallow networks

$$h_{\mathbf{V},\mathbf{W}} := h_{W^l,V^l} \circ \sigma \circ h_{W^{l-1},V^{l-1}} \circ \dots \circ \sigma \circ h_{V^1,W^1} \quad (14)$$

We build the regularized objective as

$$\min_{\mathbf{V},\mathbf{W}} \mathbb{E}_{(x,y)}[\ell(h_{\mathbf{V},\mathbf{W}}(x), y)] + \lambda \sum_{l=1}^{L} P_1(V^l, W^l). \quad (15)$$

where we have introduced the shorthand

$$P_1(V^l, W^l) := \sum_{i=1}^{n_l} \sum_{j=1}^{m_l} \sum_{k=1}^{p_l} |W^l_{ij} V^l_{ik}|$$

For the 1-path-norm of the l-th subnetwork h_{W^l,V^l}.

Because the nonsmooth nonconvex regularizer in (15) is separable in the variables $\{(V_i, W_i) : i = 1, \dots, k\}$, its proximal mapping is indeed nothing but our multioutput prox algorithm (section 5), applied independently to each shallow subnetwork component. Thus, the proximal gradient methods can be applied efficiently to optimize (15).

Acknowledgements

This work is funded (in part) through a PhD fellowship of the Swiss Data Science Center, a joint venture between EPFL and ETH Zurich. This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement n. 725594 - time-data). This work was supported by the Swiss National Science Foundation (SNSF) under grant number 200021_178865 / 1.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

Alvarez, J. M. and Salzmann, M. Learning the number of neurons in deep networks. In Lee, D. D., Sugiyama, M., Luxburg, U. V., Guyon, I., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 29*, pp. 2270–2278. Curran Associates, Inc., 2016.

Attouch, H., Bolte, J., Redont, P., and Soubeyran, A. Proximal alternating minimization and projection methods for nonconvex problems: An approach based on the kurdyka-lojasiewicz inequality. *Mathematics of Operations Research*, 35(2):438–457, may 2010. doi: 10.1287/moor.1100.0449.

Attouch, H., Bolte, J., and Svaiter, B. F. Convergence of descent methods for semi-algebraic and tame problems: proximal algorithms, forward–backward splitting, and regularized gauss–seidel methods. *Mathematical Programming*, 137(1-2):91–129, aug 2011. doi: 10.1007/s10107-011-0484-9.

Bach, F., Jenatton, R., Mairal, J., and Obozinski, G. Optimization with sparsity-inducing penalties. *Found. Trends Mach. Learn.*, 4(1):1–106, January 2012. ISSN 1935-8237. doi: 10.1561/2200000015.

Bartlett, P. L., Foster, D. J., and Telgarsky, M. J. Spectrally-normalized margin bounds for neural networks. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 30*, pp. 6240–6249. Curran Associates, Inc., 2017.

Beck, A. *Introduction to nonlinear optimization*, volume 19 of *MOS-SIAM Series on Optimization*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2014. ISBN 978-1-611973-64-8.

Beck, A. *First-Order Methods in Optimization*, volume 25. SIAM, 2017.

Bertsekas, D. P. *Nonlinear programming*. Athena scientific Belmont, 1999.

Biggio, B., Corona, I., Maiorca, D., Nelson, B., Srndic, N., Laskov, P., Giacinto, G., and Roli, F. Evasion attacks against machine learning at test time. In *ECML/PKDD*, 2013.

Bolte, J. and Pauwels, E. Conservative set valued fields, automatic differentiation, stochastic gradient method and deep learning, 2019.

Bolte, J., Sabach, S., and Teboulle, M. Proximal alternating linearized minimization for nonconvex and nonsmooth problems. *Mathematical Programming*, 146(1-2):459–494, jul 2013. doi: 10.1007/s10107-013-0701-9.

Cheng, Y., Wang, D., Zhou, P., and Zhang, T. A survey of model compression and acceleration for deep neural networks, 2017.

Cisse, M., Bojanowski, P., Grave, E., Dauphin, Y., and Usunier, N. Parseval networks: Improving robustness to adversarial examples. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 854–863, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

Clanuwat, T., Bober-Irizar, M., Kitamoto, A., Lamb, A., Yamamoto, K., and Ha, D. Deep learning for classical japanese literature. 2018.

Clevert, D.-A., Unterthiner, T., and Hochreiter, S. Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs). *arXiv e-prints*, art. arXiv:1511.07289, Nov 2015.

Condat, L. Fast projection onto the simplex and the l1 ball. *Math. Program.*, 158(1–2):575–585, July 2016. ISSN 0025-5610. doi: 10.1007/s10107-015-0946-6.

Duchi, J., Shalev-Shwartz, S., Singer, Y., and Chandra, T. Efficient projections onto the l1-ball for learning in high dimensions. In *Proceedings of the 25th International Conference on Machine Learning*, ICML 2008, pp. 272–279, New York, NY, USA, 2008. Association for Computing Machinery. ISBN 9781605582054. doi: 10.1145/1390156.1390191.

Duchi, J., Hazan, E., and Singer, Y. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

Eldar, Y. C. and Kutyniok, G. *Compressed sensing: theory and applications*. Cambridge university press, 2012.

Fazlyab, M., Robey, A., Hassani, H., Morari, M., and Pappas, G. Efficient and accurate estimation of lipschitz constants for deep neural networks. In *Advances in Neural Information Processing Systems 32*, pp. 11423–11434. Curran Associates, Inc., 2019.

Han, S., Mao, H., and Dally, W. J. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding. *International Conference on Learning Representations*, abs/1510.00149, 2016.

Jiang*, Y., Neyshabur*, B., Krishnan, D., Mobahi, H., and Bengio, S. Fantastic generalization measures and where to find them. In *International Conference on Learning Representations*, 2020.

Kakade, S. M. and Lee, J. D. Provably correct automatic sub-differentiation for qualified programs. In *Advances in Neural Information Processing Systems 31*, pp. 7125–7135. Curran Associates, Inc., 2018.

Latorre, F., Rolland, P., and Cevher, V. Lipschitz constant estimation of neural networks via sparse polynomial optimization. In *International Conference on Learning Representations*, 2020.

LeCun, Y. and Cortes, C. MNIST handwritten digit database. 2010.

Lemhadri, I., Ruan, F., and Tibshirani, R. A neural network with feature sparsity. *arXiv e-prints*, art. arXiv:1907.12207, Jul 2019.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

Metel, M. and Takeda, A. Simple stochastic gradient methods for non-smooth non-convex regularized optimization. In *International Conference on Machine Learning*, pp. 4537–4545, 2019.

Miyato, T., Kataoka, T., Koyama, M., and Yoshida, Y. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018.

Neyshabur, B., Tomioka, R., and Srebro, N. Norm-based capacity control in neural networks. In *Proceedings of The 28th Conference on Learning Theory*, volume 40 of *Proceedings of Machine Learning Research*, pp. 1376–1401, Paris, France, 03–06 Jul 2015. PMLR.

Ochs, P., Dosovitskiy, A., Brox, T., and Pock, T. On iteratively reweighted algorithms for nonsmooth nonconvex optimization in computer vision. *SIAM Journal on Imaging Sciences*, 8(1):331–372, 2015.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019.

Raghunathan, A., Steinhardt, J., and Liang, P. Certified defenses against adversarial examples. In *International Conference on Learning Representations*, 2018.

Ravi, S. N., Dinh, T., Lokhande, V. S., and Singh, V. Explicitly imposing constraints in deep networks via conditional gradients gives improved generalization and faster convergence. In *AAAI*, 2019.

Scardapane, S., Comminiello, D., Hussain, A., and Uncini, A. Group sparse regularization for deep neural networks. *Neurocomputing*, 241:81 – 89, 2017. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2017.02.029.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks, 2013.

Tsipras, D., Santurkar, S., Engstrom, L., Turner, A., and Madry, A. Robustness may be at odds with accuracy. In *International Conference on Learning Representations*, 2019.

Tsuzuku, Y., Sato, I., and Sugiyama, M. Lipschitz-margin training: Scalable certification of perturbation invariance for deep neural networks. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 31*, pp. 6541–6550. Curran Associates, Inc., 2018.

Wen, F., Chu, L., Liu, P., and Qiu, R. C. A survey on nonconvex regularization-based sparse and low-rank recovery in signal processing, statistics, and machine learning. *IEEE Access*, 6:69883–69906, 2018. ISSN 2169-3536. doi: 10.1109/ACCESS.2018.2880454.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *CoRR*, abs/1708.07747, 2017.

Yoon, J. and Hwang, S. J. Combined group and exclusive sparsity for deep neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 3958–3966, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

A. Proof of Theorem 1

We will first prove a particular case of Theorem 1, the single-output case ($p = 1$).

Proposition 1. *Let $h_{V,W}(x) = V^T\sigma(Wx) : \mathbb{R}^m \to \mathbb{R}$ be a neural network where $V \in \mathbb{R}^{n\times 1}$ and $W \in \mathbb{R}^{n\times m}$. Suppose that that the derivative of the activation is globally bounded between zero and one. Its Lipschitz constant with respect to the ℓ_∞ norm (for the input space) and the ℓ_1-norm (for the output space) is bounded as follows:*

$$L_{V,W} \leq \sum_{i=1}^{n}\sum_{j=1}^{m} |W_{i,j}V_{i,1}| \leq \|V\|_1\|W\|_\infty \tag{16}$$

First, note that because the output space is $\mathbb{R}$, the ℓ_1-norm is just the absolute value of the output. In this case the Lipschitz constant of the single-output function h is equal to the supremum of the ℓ_1-norm of its gradient, over its domain (c.f., Latorre et al. (2020, Theorem 1)).

Proof.

$$\begin{aligned}
L_{V,W} &= \sup_x \|\nabla h_{V,W}(x)\|_1 \\
&= \sup_x \sup_{\|t\|_\infty\leq 1} t^T\nabla h_{V,W}(x) \\
&= \sup_x \sup_{\|t\|_\infty\leq 1} t^T W^T \sigma'(Wx)V \\
&\leq \sup_{0\leq s\leq 1} \sup_{\|t\|_\infty\leq 1} t^T W^T \operatorname{Diag}(s)V \\
&= \sup_{0\leq s\leq 1} \sup_{\|t\|_\infty\leq 1} \sum_{i=1}^{n}\sum_{j=1}^{m} t_i(W^T\operatorname{Diag}(V))_{i,j}s_j \\
&\leq \sum_{i=1}^{n}\sum_{j=1}^{m} \sup_{0\leq s_j\leq 1} \sup_{-1\leq t_i\leq 1} t_i(W^T\operatorname{Diag}(V))_{i,j}s_j \\
&= \sum_{i=1}^{n}\sum_{j=1}^{m} |W^T\operatorname{Diag}(V)|_{i,j} = \sum_{i=1}^{n}\sum_{j=1}^{m} |W_{i,j}V_{i,1}|
\end{aligned}$$

This shows the first inequality in (16). We now show the second inequality. Denote the i-th row of the matrix W as w_i:

$$\begin{aligned}
\sum_{i=1}^{n}\sum_{j=1}^{m} |W_{i,j}V_{i,1}| &= \sum_{i=1}^{n} |V_{i,1}| \sum_{j=1}^{m} |W_{i,j}| \\
&= \sum_{i=1}^{n} |V_{i,1}| \|w_i\|_1 \\
&\leq \sum_{i=1}^{n} |V_{i,1}| \max_{j=1,\dots,m} \|w_j\|_1 \\
&= \sum_{i=1}^{n} |V_{i,1}| \|W\|_\infty \\
&= \|V\|_1\|W\|_\infty
\end{aligned}$$

In the fourth line we have used the fact that the ℓ_∞ operator norm of a matrix is equal to the maximum ℓ_1-norm of the rows.

□

Proof of Theorem 1. We now proceed with the general case where $V \in \mathbb{R}^{n\times p}$, $W \in \mathbb{R}^{n\times m}$ and $h_{V,W}(x) = V^T\sigma(Wx)$.

Proof. Denote the columns of V, in order, as $V_1, \dots, V_p$. Using Proposition 1 we have

$$\begin{aligned}
\|V^T\sigma(Wx) - V^T\sigma(Wy)\|_1 &= \sum_{k=1}^{p} |V_k^T\sigma(Wx) - V_k^T\sigma(Wy)| \\
&\leq \sum_{k=1}^{p}\sum_{i=1}^{n}\sum_{j=1}^{m} |W_{i,j}V_{i,k}|\|x-u\|_\infty \\
&\leq \sum_{k=1}^{p} \|V_k\|_1\|W\|_\infty\|x-y\|_\infty \\
&= \|V^T\|_{\infty,1}\|W\|_\infty\|x-y\|_\infty
\end{aligned}$$

where in the fourth line we have used the fact that the (ℓ_∞, ℓ_1) operator norm of a matrix V^T is equal to the sum of the ℓ_1 norm of its rows i.e., the columns of V. This shows that $L_{V,W} \leq \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{p} |W_{i,j}V_{i,k}| \leq \|V^T\|_{\infty,1}\|W\|_\infty$

□

B. Proof of Theorem 2

In this section we prove the theoretical guarantees stated in Theorem 2 of the prox-grad method described by Algorithm 1. The first and second parts of Theorem 2 follow immediately from the results establish by (Bolte et al., 2013). Part two in Theorem 2 states that Algorithm 1 is globally convergent under the celebrated Kurdyka–Lojasiewicz (KL) property (Attouch et al., 2010). The broad classes of semi-algebraic and subanalytic functions, widely used in optimization, satisfy the KL property (see e.g. (Bolte et al., 2013, Section 5)), and in particular, most convex functions encountered in finite dimensional applications satisfy it (see (Bolte et al., 2013, Section 5.1)). We refer the reader to the works (Attouch et al., 2010; 2011; Bolte et al., 2013), in particular to (Bolte et al., 2013, Sections 3.2-3.5) for additional information and results.

For Part three we require the sufficient decrease property stated next.

Lemma 11 (Sufficient decrease property (Bolte et al., 2013, Lemma 2))**.** *Let $\Psi : \mathbb{R}^n \to \mathbb{R}$ be a continuously differentiable function with gradient assumed L_Ψ-Lipschitz continuous, and let $\sigma : \mathbb{R}^n \to (-\infty, \infty]$ be a proper l.s.c function satisfying that* $\inf \sigma > -\infty$*. Fix any $t \in (0, 1/L_\Psi)$. Then, for any $\mathbf{u} \in \operatorname{dom}\sigma$ and any $\mathbf{u}^+ \in \mathbb{R}^n$ defined by*

$$\mathbf{u}^+ \in \operatorname{prox}_{\sigma t}(\mathbf{u} - t\nabla\Psi(\mathbf{u}))$$

we have

$$\Psi(\mathbf{u}) + \sigma(\mathbf{u}) - \Psi(\mathbf{u}^+) - \sigma(\mathbf{u}) \geq \frac{1 - tL_\Psi}{2t}\|\mathbf{u}^+ - \mathbf{u}\|^2.$$

Proof of Theorem 2. The first and second parts follow from the results established by (Bolte et al., 2013). We will now prove the third part. By Lemma 11 we have that

$$\mathcal{F}(z^k) - \mathcal{F}(z^{k+1}) = f(z^k) + \lambda g(z^k) - f(z^{k+1}) - \lambda g(z^{k+1})) \geq \frac{1 - L\eta_k}{2\eta_k}\|z^{k+1} - z^k\|^2. \tag{17}$$

Hence $\{f(z^k) + \lambda g(z^k)\}_{k\geq 0}$ is a non-increasing sequence that strictly decreasing unless a critical point is obtained in a finite number of steps. By summing (17) over $k = 0, 1, \dots, K$ and using the fact that $\{f(z^k) + \lambda g(z^k)\}_{k\geq 0}$ is non-increasing and is bounded below by $\mathcal{F}_*$, we obtain that

$$\begin{aligned}
\mathcal{F}(z^0) - \mathcal{F}_* &\geq \sum_{k=0}^{K} \frac{1 - L\eta_k}{2\eta_k}\|z^{k+1} - z^k\|^2 \\
&\geq \frac{1}{2}(c - L)K \min_{k=0,\dots,K} \|z^{k+1} - z^k\|_2^2.
\end{aligned}$$

Consequently,

$$\min_{k=0,\dots,K} \|z^{k+1} - z^k\|_2 \leq \sqrt{\frac{2(\mathcal{F}(z^0) - \mathcal{F}_*)}{(c - L)K}}.$$

□

C. Single output proximal map computation

This section provides the theoretical background and the required intermediate results to prove Theorem 3.

C.1. Moving to an Equivalent Easier Problem

We are interested in minimizing the nonconvex twice continuously differentiable function

$$\min_{v,w\in\mathbb{R}\times\mathbb{R}^m} \frac{1}{2}(v-x)^2 + \frac{1}{2}\sum_{j=1}^{m}(w_j - y_j)^2 + \lambda|v|\sum_{j=1}^{m}|w_j|. \tag{18}$$

The signs of the elements of the decision variables in (18) are determined by the signs of (x, y), and consequently, the problem in (18) is equivalent to problem (19); this is (partly) formulated by Lemma 12.

$$\min_{v,w\in\mathbb{R}_+\times\mathbb{R}_+^m} h_\lambda(v,w;x,y) \equiv \frac{1}{2}(v-|x|)^2 + \frac{1}{2}\sum_{j=1}^{m}(w_j - |y_j|)^2 + \lambda v\sum_{j=1}^{m}w_j. \tag{19}$$

Lemma 12. *Let $(v^*, w^*) \in \mathbb{R}_+ \times \mathbb{R}_+^n$ be an optimal solution of problem* (19)*. Then $(sign(x)\cdot v^*, sign(y)\circ w^*)$ is an optimal solution of problem* (18)*.*

Proof. We have that

$$\begin{aligned}
\tilde{h}_\lambda(v,w;x,y) &\equiv \frac{1}{2}(v-x)^2 + \frac{1}{2}\sum_{j=1}^{m}(w_j - y_j)^2 + \lambda|v|\sum_{j=1}^{m}|w_j| \\
&= \frac{1}{2}(\text{sign}(x)v - |x|)^2 + \frac{1}{2}\sum_{j=1}^{m}(\text{sign}(y_j)w_j - |y_j|)^2 + \lambda|v|\sum_{j=1}^{m}|w_j| \\
&\geq \frac{1}{2}(|v| - |x|)^2 + \frac{1}{2}\sum_{j=1}^{m}(|w_j| - |y_j|)^2 + \lambda v\sum_{j=1}^{m}w_j \\
&\geq h_\lambda(v^*, w^*; |x|, |y|),
\end{aligned}$$

where the last inequality follows from the fact that (v^*, w^*) is an optimal solution of (19). Since equality with the lower bound is attained by setting $(v, w) = (\text{sign}(x)\cdot v^*, \text{sign}(y)\circ w^*)$, we conclude that $(\text{sign}(x)\cdot v^*, \text{sign}(y)\circ w^*)$ is an optimal solution of (18). □

To summarize, we have established that, finding an optimal solution to (19) and then changing signs accordingly, yields an optimal solution to (18). We will now focus on obtaining an optimal solution for (18).

C.2. Solving the Prox Problem

First we note that problem (19) is well-posed.

Lemma 13 (Well-posedness of (19))**.** *For any $\lambda \geq 0$ and any $(x, y) \in \mathbb{R}\times\mathbb{R}^m$, the problem* (19) *has a global optimal solution.*

Proof. The claim follows from the fact that the objective function is coercive, cf. (Beck, 2014, Thm. 2.32). □

In light of Lemma 13, and due to the fact that in (19) we minimize a continuously differentiable function over a closed convex set, the set of optimal solutions of (19) is a nonempty subset of the set of stationary points of (19). These satisfy the following conditions (cf. (Beck, 2014, Ch. 9.1)).

Lemma 14 (Stationarity conditions). *Let $(v^*, w^*) \in \mathbb{R}_+ \times \mathbb{R}_+^m$ be an optimal solution of (19) for a given $(x, y) \in \mathbb{R} \times \mathbb{R}^m$. Then*

$$w_j^* = \max\{0, |y_j| - \lambda v^*\} \;\; \textit{for any } j = 1, 2, \ldots, m,$$
$$v^* = \max\left\{0, |x| - \lambda \sum_{j=1}^{m} w_j^*\right\}.$$

Proof. The stationarity (first-order) conditions of (19) (cf. (Beck, 2014, Ch. 9.1)) state that

$$\frac{\partial h_\lambda}{\partial v}(v^*, w^*; x, y) \begin{cases} = 0, & v^* > 0, \\ \geq 0, & v^* = 0, \end{cases} \quad \frac{\partial h_\lambda}{\partial w_j}(v^*, w^*; x, y) \begin{cases} = 0, & w_j^* > 0, \\ \geq 0, & w_j^* = 0, \end{cases}$$

which translates to

$$v^* - |x| + \lambda \sum_{j=1}^{m} w_j^* \begin{cases} = 0, & v^* > 0, \\ \geq 0, & v^* = 0, \end{cases} \quad w_j^* - |y_j| + \lambda v^* \begin{cases} = 0, & w_j^* > 0, \\ \geq 0, & w_j^* = 0, \end{cases}$$

and the required follows. □

The stationarity conditions given in Lemma 14 imply a solution form that we exploit in Algorithm 2; this is described by Corollary 3, where we use the convention that $\sum_{j=1}^{0} a_j \equiv 0$ for any $\{a_j\} \subseteq \mathbb{R}$.

Corollary 3. *Let $(v^*, w^*) \in \mathbb{R}_+ \times \mathbb{R}_+^m$ be an optimal solution of (19) for a given $(x, y) \in \mathbb{R} \times \mathbb{R}^m$.*

1. *The vector w^* satisfies that for any $j, l \in \{1, 2, \ldots, m\}$ it holds that $w_j^* \geq w_l^*$ only if $|y_j| \geq |y_l|$.*
2. *If $v^* = 0$, then $w^* = y$.*
3. *If $v^* > 0$, and $s = |\{j : w_j^* > 0\}|$, then we have that*

$$v^* = \frac{1}{1 - s\lambda^2}\left(|x| - \lambda \sum_{j=1}^{s} |\bar{y}_j|\right), \tag{20}$$

where $\bar{y}$ is the sorted vector of y in descending magnitude order.

Proof. The first part follows trivially from the stationarity conditions on w^* given in Lemma 14. The second part also follows trivially from the problem definition.

From the first part and the conditions in Lemma 14 we have that $\sum_{j=1}^{m} w_j^* = \sum_{j=1}^{s} |\bar{y}_j| - \lambda s v^*$. Plugging the latter to the stationarity condition on v^* (given in Lemma 14) then implies the required. □

In our analysis, we strictly distinguish between the trivial solution $(v^*, w^*) = (0, y)$, and the non-trivial solution in which $v^* > 0$. A practical point-of-view suggests that if $v^* = 0$, then the corresponding succeeding weights should also be zero, even though the optimality conditions imply otherwise. However, to avoid hindering the training process, this observation is considered only in the end of the training.

Recall that our analysis so-far implies that the magnitude order of y determines the order magnitude of w, effectively implying on set of non-zero entries in w (cf. Remark 5). For clarity of indices, and without loss of generality, we assume throughout this section that the vector y is already sorted in decreasing order of magnitude, that is $y \equiv \bar{y}$. We will use, without confusion, both notation to describe the same entity in order to maintain coherence with our procedures and results.

Denote

$$\begin{aligned} v^{(s)} &= \frac{1}{1 - s\lambda^2}\left(|x| - \lambda \sum_{j=1}^{s} |y_j|\right) \\ w_j^{(s)} &= |y_j| - \lambda v^{(s)} \;\; \text{for } j = 1, 2, \ldots, s, \;\; \text{and } w_j^{(s)} = 0 \text{ otherwise.} \end{aligned} \tag{21}$$

Lemma 5 which states the monotonicity property

$$h_\lambda(v^{(s)}, w^{(s)}; x, y) < h_\lambda(v^{(s-1)}, w^{(s-1)}; x, y).$$

is proved next.

Proof of Lemma 5. Recall that $h_\lambda(v, w; x, y) := \frac{1}{2}(v - |x|)^2 + \frac{1}{2}\sum_{j=1}^m (w_j - |y_j|)^2 + \lambda v \sum_{j=1}^m w_j$. By plugging $w^{(s)}$ defined in (21) to h_λ we obtain that

$$\begin{aligned} h_\lambda(v^{(s)}, w^{(s)}; x, y) &= \frac{1}{2}(v^{(s)} - |x|)^2 + \frac{1}{2}\sum_{i=1}^{s}(|\bar{y}_i| - (|\bar{y}_i| - \lambda v^{(s)}))^2 + \frac{1}{2}\sum_{i=s+1}^{m}|\bar{y}_i|^2 + \lambda v^{(s)}\sum_{i=1}^{s}(|\bar{y}_i| - \lambda v^{(s)}) \\ &= \frac{1}{2}(v^{(s)} - |x|)^2 + \frac{\lambda^2}{2}s(v^{(s)})^2 + \frac{1}{2}\|y\|_2^2 - \frac{1}{2}\sum_{i=1}^{s}|\bar{y}_i|^2 + \lambda v^{(s)}\sum_{i=1}^{s}|\bar{y}_i| - \lambda^2 s(v^{(s)})^2. \end{aligned}$$

Consequently, plugging $v^{(s)}$, defined in (21), yields

$$\begin{aligned} h_\lambda(v^{(s)}, w^{(s)}; x, y) &= \frac{1}{2}\left(\frac{\lambda^2 s}{1-\lambda^2 s}|x| - \frac{\lambda}{1-\lambda^2 s}\sum_{i=1}^{s}|\bar{y}_i|\right)^2 - \frac{\lambda^2 s}{2(1-\lambda^2 s)^2}\left(|x| - \lambda\sum_{i=1}^{s}|\bar{y}_i|\right)^2 \\ &+ \frac{\lambda}{1-\lambda^2 s}\sum_{i=1}^{s}|\bar{y}_i|\left(|x| - \lambda\sum_{i=1}^{s}|\bar{y}_i|\right) - \frac{1}{2}\sum_{i=1}^{s}|\bar{y}_i|^2 + \frac{1}{2}\|y\|_2^2 \\ &= \frac{\lambda^2 s}{2(1-\lambda^2 s)^2}x^2(\lambda^2 s - 1) + \frac{\lambda^2}{2(1-\lambda^2 s)^2}\left(\sum_{i=1}^{s}|\bar{y}_i|\right)^2(1 - \lambda^2 s - 2(1-\lambda^2 s)) \\ &+ |x|\sum_{i=1}^{s}|\bar{y}_i|\left(-\frac{\lambda^3 s}{(1-\lambda^2 s)^2} + \frac{\lambda^3 s}{(1-\lambda^2 s)^2} + \frac{\lambda}{1-\lambda^2 s}\right) - \frac{1}{2}\sum_{i=1}^{s}|\bar{y}_i|^2 + \frac{1}{2}\|y\|_2^2 \\ &= \frac{1}{2(1-\lambda^2 s)}\left(-\lambda^2 s x^2 - \left(|x| - \lambda\sum_{i=1}^{s}|\bar{y}_i|\right)^2 + x^2\right) - \frac{1}{2}\sum_{i=1}^{s}|\bar{y}_i|^2 + \frac{1}{2}\|y\|_2^2 \\ &= -\frac{1}{2(1-\lambda^2 s)}\left(|x| - \lambda\sum_{i=1}^{s}|\bar{y}_i|\right)^2 + \frac{1}{2}\|x\|_2^2 - \frac{1}{2}\sum_{i=1}^{s}|\bar{y}_i|^2 + \frac{1}{2}\|y\|_2^2 \\ &= -\left(1 + \frac{\lambda^2}{1-\lambda^2 s}\right)\frac{1}{2(1-\lambda^2(s-1))}\left(|x| - \lambda\sum_{i=1}^{s-1}|\bar{y}_i| - \lambda|\bar{y}_s|\right)^2 + \frac{1}{2}\|x\|_2^2 - \frac{1}{2}\sum_{i=1}^{s-1}|\bar{y}_i|^2 - \frac{1}{2}|\bar{y}_s|^2 + \frac{1}{2}\|y\|_2^2 \\ &= h_\lambda(v^{(s-1)}, w^{(s-1)}; x, y) - \frac{1}{2(1-\lambda^2 s + \lambda^2)}\left(-2\lambda|\bar{y}_s|\left(|x| - \lambda\sum_{i=1}^{s-1}|\bar{y}_i|\right) + \lambda^2|\bar{y}_s|^2\right) \\ &- \frac{\lambda^2}{2(1-\lambda^2 s)(1-\lambda^2 s + \lambda^2)}\left(|x| - \lambda\sum_{i=1}^{s}|\bar{y}_i|\right)^2 - \frac{1}{2}|\bar{y}_s|^2. \end{aligned}$$

Therefore,

$$\begin{aligned}
&h_\lambda(v^{(s)}, w^{(s)}; x, y) - h_\lambda(v^{(s-1)}, w^{(s-1)}; x, y)\\
&= -\frac{1}{2(1-\lambda^2 s+\lambda^2)}\left(-2\lambda|\bar{y}_s|\left(|x| - \lambda\sum_{i=1}^{s}|\bar{y}_i|\right) - \lambda^2|\bar{y}_s|^2 + \frac{\lambda^2}{1-\lambda^2 s}\left(|x| - \lambda\sum_{i=1}^{s}|\bar{y}_i|\right)^2 + (1-\lambda^2 s+\lambda^2)|\bar{y}_s|^2\right)\\
&= -\frac{1}{2(1-\lambda^2 s+\lambda^2)}\left((1-\lambda^2 s)|\bar{y}_s|^2 - 2\lambda|\bar{y}_s|\left(|x| - \lambda\sum_{i=1}^{s}|\bar{y}_i|\right) + \frac{\lambda^2}{1-\lambda^2 s}\left(|x| - \lambda\sum_{i=1}^{s}|\bar{y}_i|\right)^2\right)\\
&= -\frac{1-\lambda^2 s}{2(1-\lambda^2 s+\lambda^2)}\left(|\bar{y}_s|^2 - 2\lambda|\bar{y}_s|v^{(s)} + \lambda^2(v^{(s)})^2\right)\\
&= -\frac{1-\lambda^2 s}{2(1-\lambda^2 s+\lambda^2)}\left(|\bar{y}_s| - \lambda v^{(s)}\right)^2 \leq 0,
\end{aligned}$$

meaning that

$$h_\lambda(v^{(s)}, w^{(s)}; x, y) \leq h_\lambda(v^{(s-1)}, w^{(s-1)}; x, y).$$

□

We can now prove our key result formulated in Corollary 2, that states that $(v^{(s^*)}, w^{(s^*)})$ is an optimal solution of (7) for

$$s^* = \max\left\{s \in [\bar{s}] : v^{(s)}, w^{(s)} > 0\right\}, \;\text{ where }\; \bar{s} = \min(\lfloor\lambda^{-2}\rfloor, m).$$

Proof of Corollary 2. By Lemma 3, $(v^{(s^*)}, w^{(s^*)})$ is a stationary point of (7). Moreover, according to Corollary 1 and Lemma 4, $(v^{(s^*)}, w^{(s^*)})$ belongs to the set of $\bar{s}$ stationary points that are candidates to be optimal solutions of (7). Invoking Lemma 5, we have that

$$h_\lambda(v^{(s^*)}, w^{(s^*)}; x, y) < h_\lambda(v^{(j)}, w^{(j)}; x, y), \quad \forall s^* > j. \tag{22}$$

Hence, $(v^{(j)}, w^{(j)})$ is not an optimal solution for any $j < s^*$.

Let us now consider the complementary case. By Lemma 4, for any $i > \bar{s}$ the pair $(v^{(i)}, w^{(i)})$ does not satisfy the second-order optimality conditions, and therefore is not an optimal solution. On the other hand, by the definition of s^*, for any $\bar{s} > i > s^*$ the pair $(v^{(i)}, w^{(i)})$ is not a feasible solution , and subsequently not a stationary point. To conclude, $h_\lambda(v^{(s^*)}, w^{(s^*)}; x, y) < h_\lambda(v^{(j)}, w^{(j)}; x, y)$ holds for any $j \neq s^*$ such that $(v^{(j)}, w^{(j)})$ is a stationary point, meaning that $(v^{(s^*)}, w^{(s^*)})$ is an optimal solution of (7).

□

Finally, we will show that the problem of finding s^* can be easily solved using binary search. To this end, we show that the feasibility criterion (i.e., $v^{(s)} > 0$ and $w^{(s)} > 0$) satisfies that

$$(v^{(k)}, w^{(k)}) \text{ is feasible } \Rightarrow (v^{(i)}, w^{(i)}) \text{ is feasible } \forall i < k$$

Proof of Lemma 6. Suppose that $(v^{(k)}, w^{(k)})$ is feasible for some $k \in \{2, \ldots, \bar{s}\}$. By induction principle, it is sufficient to show that $(v^{(k-1)}, w^{(k-1)})$ is feasible in order to prove the result.

By (21), we have:

$$(1 - k\lambda^2)v^{(k)} = |x| - \lambda\sum_{j=1}^{k}|y_j| = (1 - k\lambda^2 + \lambda^2)v^{(k-1)} - |y_k|.$$

which implies

$$v^{(k-1)} = \frac{1}{(1 - k\lambda^2 + \lambda^2)}((1 - k\lambda^2)v^{(k)} + |y_k|) \geq 0.$$

For $w^{(k)}$, it is easy to see from (21) that since the vector y is sorted in decreasing order of magnitude, the vector $w^{(k)}$ is also sorted in decreasing order, and thus $w^{(k)}$ is feasible if and only if $w_k^{(k)} > 0$.

$$\begin{aligned}(1-k\lambda^2)w_k^{(k)} &= (1-k\lambda^2)|y_k| - \lambda|x| + \lambda^2\sum_{j=1}^{k}|y_j| \\ &= -\lambda|x| + (1-(k-1)\lambda^2)|y_{k-1}| + \lambda^2\sum_{j=1}^{k-1}|y_j| + \lambda^2|y_k| + (1-k\lambda^2)|y_k| - (1-(k-1)\lambda^2)|y_{k-1}| \\ &= (1-(k-1)\lambda^2)w_{k-1}^{(k-1)} + (1-k\lambda^2+\lambda^2)(|y_k|-|y_{k-1}|),\end{aligned}$$

where the last line uses the identity of the first line for $k-1$. We thus have:

$$w_{k-1}^{(k-1)} = \frac{1}{(1-(k-1)\lambda^2)}(1-k\lambda^2)w_k^{(k)} + |y_{k-1}| - |y_k| \geq 0,$$

since $|y_{k-1}| \geq |y_k|$ and $k \leq \lambda^{-2}$.

Therefore, there exists a value s^* such that $v^{(k)} > 0$ and $w^{(k)} > 0\ \forall k \geq s^*$ and $v^{(k)} \geq 0$ or $w^{(k)} \geq 0\ \forall k > s^*$. This value of s^* can thus efficiently be found using binary search.

□

D. Multi-output proximal map computation

D.1. Solving the prox problem

Returning to the multi-output setting, recall that $h_{V,W}(x) = V^T\sigma(Wx)$ with $V \in \mathbb{R}^{p\times n}, W \in \mathbb{R}^{n\times m}$ and

$$g(V,W) = \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{p} W_{ij}V_{ki}.$$

The proximal mapping can then be written as:

$$\begin{aligned}\text{prox}_{\lambda g}(\bar{V},\bar{W}) &= \underset{V,W}{\text{argmin}}\ \frac{1}{2}\|V-\bar{V}\|_F + \frac{1}{2}\|W-\bar{W}\|_F + \lambda\sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{p}W_{ij}V_{ki} \\ &= \underset{V,W}{\text{argmin}}\sum_{i=1}^{n}\left(\frac{1}{2}\sum_{k=1}^{p}(V_{ki}-\bar{V}_{ki})^2 + \sum_{j=1}^{p}(W_{ij}-\bar{W}_{ij})^2 + \sum_{j=1}^{m}\sum_{k=1}^{p}W_{ij}V_{ki}\right).\end{aligned}$$

Noting that the proximal mapping is separable with respect to the columns of W and the rows of V, and using the same sign trick applied in the single-output case, it is enough to solve for any $i = 1,\ldots,n$,

$$\min_{v,w\in\mathbb{R}_+^p\times\mathbb{R}_+^m} h_\lambda(v,w;x,y) \equiv \frac{1}{2}\sum_{k=1}^{p}(v_k-|x_k|)^2 + \frac{1}{2}\sum_{j=1}^{m}(w_j-|y_j|)^2 + \lambda\sum_{j=1}^{m}\sum_{k=1}^{p}v_kw_j, \tag{23}$$

where x denotes the i-th row of V and y the i-th column of W, in order to compute the prox operator.

The stationarity conditions for (23) are stated next; the arguments are the same as in the single-output case.

Lemma 15 (Stationarity conditions)**.** *Let $(v^*,w^*) \in \mathbb{R}_+^p \times \mathbb{R}_+^m$ be an optimal solution of (23) for a given $(x,y) \in \mathbb{R}^p \times \mathbb{R}^m$. Then*

$$w_j^* = \max\left\{0, |y_j| - \lambda\sum_{k=1}^{p}v_k^*\right\}\ \textit{for any } j = 1,2,\ldots,m,$$

$$v_k^* = \max\left\{0, |x_k| - \lambda\sum_{j=1}^{m}w_j^*\right\}\ \textit{for any } k = 1,2,\ldots,p.$$

The next lemma restates the result in Lemma 7 which expands on the monotonic relation in magnitude originally established for single-output networks in Corollary 1.

Lemma 16. *Let $(v^*, w^*) \in \mathbb{R}^p_+ \times \mathbb{R}^m_+$ be an optimal solution of (19) for a given $(x, y) \in \mathbb{R}^p \times \mathbb{R}^m$.*

1. *The vector w^* satisfies that for any $j, l \in \{1, 2, \ldots, m\}$ it holds that $w^*_j \geq w^*_l$ only if $|y_j| \geq |y_l|$.*
2. *The vector v^* satisfies that for any $k, l \in \{1, 2, \ldots, p\}$ it holds that $v^*_k \geq v^*_l$ only if $|x_k| \geq |x_l|$.*
3. *Let $\bar{x}, \bar{y}$ be the sorted vector of x and y respectively in descending magnitude order. Let $s_v = |\{k : v^*_k > 0\}|$ and $s_w = |\{j : w^*_j > 0\}|$. If $v^*, w^* \neq 0$, then*

$$v^*_k = |x_k| + \frac{1}{1 - s_v s_w \lambda^2} \left(\lambda^2 s_w \sum_{l=1}^{s_v} |\bar{x}_l| - \lambda \sum_{j=1}^{s_w} |\bar{y}_j| \right), \tag{24}$$

$$w^*_j = |y_j| + \frac{1}{1 - s_v s_w \lambda^2} \left(\lambda^2 s_v \sum_{l=1}^{s_w} |\bar{y}_l| - \lambda \sum_{k=1}^{s_v} |\bar{x}_k| \right). \tag{25}$$

Proof. The two first points are direct applications of the stationary conditions of Lemma 15.

From the conditions in Lemma 15 we have that

$$\begin{aligned}
\sum_{j=1}^{m} w^*_j &= \sum_{j=1}^{s_w} |\bar{y}_j| - \lambda s_w \sum_{k=1}^{p} v^*_k \\
\sum_{k=1}^{p} v^*_k &= \sum_{k=1}^{s_v} |\bar{x}_k| - \lambda s_v \sum_{j=1}^{m} w^*_j \\
&= \sum_{k=1}^{s_v} |\bar{x}_k| - \lambda s_v \sum_{j=1}^{s_w} |\bar{y}_j| + \lambda^2 s_v s_w \sum_{k=1}^{p} v^*_k \\
&= \frac{1}{1 - \lambda^2 s_v s_w} \left(\sum_{k=1}^{s_v} |\bar{x}_k| - \lambda s_v \sum_{j=1}^{s_w} |\bar{y}_j| \right).
\end{aligned}$$

Thus,

$$\begin{aligned}
\sum_{j=1}^{m} w^*_j &= \sum_{j=1}^{s_w} |\bar{y}_j| - \frac{\lambda s_w}{1 - \lambda^2 s_v s_w} \left(\sum_{k=1}^{s_v} |\bar{x}_k| - \lambda s_v \sum_{j=1}^{s_w} |\bar{y}_j| \right) \\
&= \frac{1}{1 - \lambda^2 s_v s_w} \left(-\lambda s_w \sum_{k=1}^{s_v} |\bar{x}_k| + \sum_{j=1}^{s_w} |\bar{y}_j| \right).
\end{aligned}$$

Plugging the latter to the stationarity condition on v^* (given in Lemma 15) then implies the result. □

Finally, we show, as in the single-output case, that second order stationarity condition constraints the ranges of sparsities of v^* and w^*; this relation is given by Lemma 8, and is proved next.

Proof of Lemma 8. Since (v^*, w^*) is an optimal solution of (23) and the objective function in (23) is twice continuously differentiable, (v^*, w^*) satisfies the second order necessary optimality conditions. That is, for any $d \in \mathbb{R}^p \times \mathbb{R}^m$ satisfying that $(v^*, w^*) + d \in \mathbb{R}^p_+ \times \mathbb{R}^m_+$ and $d^T \nabla h_\lambda(v^*, w^*; x, y) = 0$ it holds that

$$d^T \nabla^2 h_\lambda(v^*, w^*; x, y) d = d^T \begin{pmatrix} I_{p \times p} & \Lambda_{p \times m} \\ \Lambda_{m \times p} & I_{m \times m} \end{pmatrix} d \geq 0,$$

where the first row/column corresponds to v and the others correspond to w, I denotes the identity matrix and Λ denotes a matrix completely filled with λ. Similarly as in the single output case, we require that the submatrix of $\nabla^2 h_\lambda(v^*, w^*; x, y)$ containing the rows and columns corresponding to the positive coordinates in (v^*, w^*) is positive semidefinite. Since the minimal eigenvalue of this submatrix equals $1 - \lambda\sqrt{|S_v||S_w|}$, we have that

$$\lambda^{-2} \geq |S_v||S_w|.$$

□

A possible way of solving this proximal problem is thus to exhaustively compute the value of h_λ at each stationary point associated with sparsities $s_v = 1, \ldots, p$, $s_w = 1, \ldots, m$ such that $s_v s_w \leq \lambda^{-2}$. However, trying all possible pairs of sparsities (s_v, s_w) is computationally costly. Similarly as is the single output case, we can exploit some structure of the objective function h_λ in order to reduce the possible candidate pairs of sparsities.

Without loss of generality, we assume hereafter that the vectors x, y are already sorted in decreasing order of magnitude.

Lemma 16 shows that for each pair (s_v, s_w), $s_v = 0, 1, \ldots, p$, $s_w = 0, 1, \ldots, m$, there exists a stationary point $(v^{(s_v,s_w)}, w^{(s_v,s_w)})$ of $h_\lambda(\cdot, \cdot; x, y)$ such that $|\{k : v_k^{(s_v,s_w)} > 0\}| = s_v$, $|\{j : w_j^{(s_v,s_w)} > 0\}| = s_w$, given by

$$\begin{aligned} v_k^{(s_v,s_w)} &= |x_k| + \frac{1}{1 - s_v s_w \lambda^2}\left(\lambda^2 s_w \sum_{l=1}^{s_v} |x_l| - \lambda \sum_{j=1}^{s_w} |y_j|\right) \text{ for } k = 1, 2, \ldots, s_v, \text{ and } v_k^{(s_v,s_w)} = 0 \text{ otherwise} \\ w_j^{(s_v,s_w)} &= |y_j| + \frac{1}{1 - s_v s_w \lambda^2}\left(\lambda^2 s_v \sum_{l=1}^{s_w} |y_l| - \lambda \sum_{k=1}^{s_v} |x_k|\right) \text{ for } j = 1, 2, \ldots, s_w, \text{ and } w_j^{(s_v,s_w)} = 0 \text{ otherwise.} \end{aligned} \tag{26}$$

We now move to prove the monotonicity property stated in Lemma 9.

Proof of Lemma 9. The proof follows exactly the same lines as in the single output case. We recall the definition of the objective function:

$$h_\lambda(v, w; x, y) \equiv \frac{1}{2}\sum_{k=1}^{p}(v_k - |x_k|)^2 + \frac{1}{2}\sum_{j=1}^{m}(w_j - |y_j|)^2 + \lambda\left(\sum_{k=1}^{p} v_k\right)\left(\sum_{j=1}^{m} w_j\right).$$

Plugging the definitions from equation (26), we have

$$
\begin{aligned}
h_\lambda\left(v^{(s_v,s_w)}, w^{(s_v,s_w)}; x, y\right) &= \frac{s_v}{2}\left(\frac{1}{1-\lambda^2 s_v s_w}\left(\lambda^2 s_w \sum_{k=1}^{s_v}|x_k| - \lambda\sum_{j=1}^{s_w}|y_j|\right)\right)^2 + \frac{1}{2}\sum_{k=s_v+1}^{p} x_k^2 \\
&+ \frac{s_w}{2}\left(\frac{1}{1-\lambda^2 s_v s_w}\left(\lambda^2 s_v \sum_{j=1}^{s_w}|y_j| - \lambda\sum_{k=1}^{s_v}|x_k|\right)\right)^2 + \frac{1}{2}\sum_{j=s_w+1}^{m} y_j^2 \\
&+ \frac{\lambda}{(1-\lambda^2 s_v s_w)^2}\left(\sum_{k=1}^{s_v}|x_k| - \lambda s_v\sum_{j=1}^{s_w}|y_j|\right)\left(-\lambda s_w\sum_{k=1}^{s_v}|x_k| + \sum_{j=1}^{s_w}|y_j|\right) \\
&= \frac{1}{2(1-\lambda^2 s_v s_w)^2}\left(\left(\sum_{k=1}^{s_v}|x_k|\right)^2(\lambda^4 s_v s_w^2 + \lambda^2 s_w - 2\lambda^2 s_w) + \left(\sum_{j=1}^{s_w}|y_j|\right)^2(\lambda^2 s_v + \lambda^4 s_v^2 s_w - 2\lambda^2 s_v)\right. \\
&\left.\left(\sum_{k=1}^{s_v}|x_k|\right)\left(\sum_{j=1}^{s_w}|y_j|\right)(-2\lambda^3 s_v s_w - 2\lambda^3 s_v s_w + 2\lambda + 2\lambda^3 s_v s_w)\right) + \frac{1}{2}\sum_{k=s_v+1}^{p} x_k^2 + \frac{1}{2}\sum_{j=s_w+1}^{m} y_j^2 \\
&= \frac{1}{2(1-\lambda^2 s_v s_w)}\left(-\lambda^2 s_w\left(\sum_{k=1}^{s_v}|x_k|\right)^2 - \lambda^2 s_v\left(\sum_{j=1}^{s_w}|y_j|\right)^2 + 2\lambda\left(\sum_{k=1}^{s_v}|x_k|\right)\left(\sum_{j=1}^{s_w}|y_j|\right)\right) + \frac{1}{2}\sum_{k=s_v+1}^{p} x_k^2 + \frac{1}{2}\sum_{j=s_w+1}^{m} y_j^2
\end{aligned} \tag{27}
$$

$$
\begin{aligned}
&= \left(1 + \frac{\lambda^2 s_v}{1-\lambda^2 s_v s_w}\right)\frac{1}{2(1-\lambda^2 s_v(s_w-1))}\left(-\lambda^2(s_w-1)\left(\sum_{k=1}^{s_v}|x_k|\right)^2 - \lambda^2\left(\sum_{k=1}^{s_v}|x_k|\right)^2\right. \\
&\left.-\lambda^2 s_v\left(\left(\sum_{j=1}^{s_w-1}|y_j|\right)^2 + 2\lambda|y_{s_w}|\sum_{j=1}^{s_w-1}|y_j| + y_{s_w}^2\right) + 2\lambda\sum_{k=1}^{s_v}|x_k|\left(\sum_{j=1}^{s_w-1}|y_j| + |y_{s_w}|\right)\right) + \frac{1}{2}\sum_{k=s_v+1}^{p} x_k^2 + \frac{1}{2}\sum_{j=s_w-1+1}^{m} y_j^2 - \frac{1}{2}y_{s_w}^2.
\end{aligned} \tag{28}
$$

By applying equation (27) at s_v, $s_w - 1$, we can express the right hand side of equation (28) in terms of $h_\lambda\left(v^{(s_v,s_w-1)}, w^{(s_v,s_w-1)}; x, y\right)$ as:

$$
\begin{aligned}
h_\lambda\left(v^{(s_v,s_w)}, w^{(s_v,s_w)}; x, y\right) &= h_\lambda\left(v^{(s_v,s_w-1)}, w^{(s_v,s_w-1)}; x, y\right) + \frac{1}{2(1-\lambda^2 s_v(s_w-1))}\left(-\lambda^2\left(\sum_{k=1}^{s_v}|x_k|\right)^2\right. \\
&\left.-\lambda^2 s_v|y_{s_w}|\left(2\sum_{j=1}^{s_w-1}|y_j| + |y_{s_w}|\right) + 2\lambda|y_{s_w}|\sum_{k=1}^{s_v}|x_k|\right) + \frac{\lambda^2 s_v}{2(1-\lambda^2 s_v s_w)(1-\lambda^2 s_v(s_w-1))}\left(-\lambda^2 s_w\left(\sum_{k=1}^{s_v}|x_k|\right)^2\right. \\
&\left.-\lambda^2 s_v\left(\sum_{j=1}^{s_w}|y_j|\right)^2 - 2\lambda\left(\sum_{k=1}^{s_v}|x_k|\right)\left(\sum_{j=1}^{s_w}|y_j|\right)\right) - \frac{1}{2}y_{s_w}^2.
\end{aligned}
$$

Therefore:

$$
\begin{aligned}
&h_\lambda\left(v^{(s_v,s_w)},w^{(s_v,s_w)};x,y\right)-h_\lambda\left(v^{(s_v,s_w-1)},w^{(s_v,s_w-1)};x,y\right)\\
&\quad=-\frac{1}{2(1-\lambda^2 s_v(s_w-1))}\left(-2\lambda|y_{s_w}|\left(\sum_{k=1}^{s_v}|x_k|-\lambda s_v\sum_{j=1}^{s_w}|y_j|\right)-\lambda^2 s_v|y_{s_w}|^2+\lambda^2\left(\sum_{k=1}^{s_v}|x_k|\right)^2\right.\\
&\quad\left.+\frac{\lambda^2 s_v}{1-\lambda^2 s_v s_w}\left(\lambda^2 s_w\left(\sum_{k=1}^{s_v}|x_k|\right)^2+\lambda^2 s_v\left(\sum_{j=1}^{s_w}|y_j|\right)^2-2\lambda\left(\sum_{k=1}^{s_v}|x_k|\right)\left(\sum_{j=1}^{s_w}|y_j|\right)\right)+(1-\lambda^2 s_v s_w+\lambda^2 s_v)|y_{s_w}|\right)\\
&\quad=-\frac{1}{2(1-\lambda^2 s_v(s_w-1))}\left((1-\lambda^2 s_v s_w)y_{s_w}^2-2\lambda|y_{s_w}|(1-\lambda^2 s_v s_w)\sum_{k=1}^{s_v}v_k^{(s_v,s_w)}+\frac{\lambda^2}{1-\lambda^2 s_v s_w}\left(\sum_{k=1}^{s_v}|x_k|-\lambda s_v\sum_{j=1}^{s_w}|y_j|\right)^2\right)\\
&\quad=-\frac{1-\lambda^2 s_v s_w}{2(1-\lambda^2 s_v(s_w-1))}\left(|y_{s_w}|-\lambda\sum_{k=1}^{s_v}v_k^{(s_v,s_w)}\right)^2.
\end{aligned}
$$

The second result is obtain directly by symmetry between v and w. □

In order to derive an efficient algorithm , we will again exploit the monotone property of the feasibility criterion $v^{(s_v,s_w)}>0$, $w^{(s_v,s_w)}>0$ restated from Lemma 10:

Lemma 17 (Restatement of Lemma 10)**.** *Let $(k,l)\in[p]\times[m]$ be such that $kl\le\lambda^{-2}$. Suppose that*

$$v^{(k,l)}\ge 0 \textit{ and } w^{(k,l)}\ge 0.$$

Then for any $i=1,\dots,k$ and any $j=1,\dots,l$, it holds that

$$v^{(i,j)}\ge 0 \textit{ and } w^{(i,j)}\ge 0.$$

Proof of Lemma 10. Since the first k entries of $v^{(k,l)}$ are ordered in decreasing order, we have that $v^{(k,l)}\ge 0$ if and only if $v_k^{(k,l)}\ge 0$. Similarly, $w^{(k,l)}\ge 0$ if and only if $w_l^{(k,l)}\ge 0$.

Suppose that $v^{(k,l)}\ge 0$ and $w^{(k,l)}\ge 0$. By induction, in order to prove the result, it is sufficient to prove that $v_{k-1}^{(k-1,l)}\ge 0$, $v_k^{(k,l-1)}\ge 0$, $w_l^{(k-1,l)}\ge 0$ and $w_{l-1}^{(k,l-1)}\ge 0$. We only prove the result for v, as the proof for w is identical.

Using equation (26), we have:

$$
\begin{aligned}
(1-kl\lambda^2)v_k^{(k,l)}&=(1-kl\lambda^2)|x_k|+\lambda^2 l\sum_{i=1}^{k}|x_i|-\lambda\sum_{j=1}^{l}|y_j| \qquad (29)\\
&=(1-kl\lambda^2)|x_k|+(1-(k-1)l\lambda^2)|x_{k-1}|-(1-(k-1)l\lambda^2)|x_{k-1}|+\lambda^2 l\sum_{i=1}^{k-1}|x_i|+\lambda^2 l|x_k|\lambda\sum_{j=1}^{l}|y_j|\\
&=(1-(k-1)l\lambda^2)v_{k-1}^{(k-1,l)}+(1-(k-1)l\lambda^2)(|x_k|-|x_{k-1}|).
\end{aligned}
$$

Therefore:

$$v_{k-1}^{(k-1,l)}=\frac{1-(k-1)l\lambda^2}{1-kl\lambda^2}v_k^{(k,l)}+|x_{k-1}|-|x_k|\ge 0,$$

since the vector x is ordered in decreasing order of magnitude, and thus $|x_{k-1}|-|x_k|\ge 0$.

Using again equation (29), we have:

$$\begin{aligned}(1-kl\lambda^2)v_k^{(k,l)} &= (1-kl\lambda^2)|x_k| + (1-k(l-1)\lambda^2)|x_k| - (1-k(l-1)\lambda^2)|x_k| \\ &+ \lambda^2(l-1)\sum_{i=1}^{k}|x_i| + \lambda^2\sum_{i=1}^{k}|x_i| - \lambda\sum_{j=1}^{l-1}|y_j| - \lambda|y_l| \\ &= (1-k(l-1)\lambda^2)v_k^{(k,l-1)} - k\lambda^2|x_k| + \lambda^2\sum_{i=1}^{k}|x_i| - \lambda|y_l|,\end{aligned}$$

where the last equality follows (again) from equation (29) for $v_k^{(k,l-1)}$. Thus,

$$(1-k(l-1)\lambda^2)v_k^{(k,l-1)} = (1-kl\lambda^2)v_k^{(k,l)} + k\lambda^2|x_k| - \lambda^2\sum_{i=1}^{k}|x_i| + \lambda|y_l|. \tag{30}$$

From the definition of $v_k^{(k,l)}$ (equation (26)), we have that $v_k^{(k,l)} \geq 0$ is equivalent to the condition:

$$|x_k| \geq \frac{\lambda\sum_{j=1}^{l}|y_j| - l\lambda^2\sum_{i=1}^{k}|x_i|}{1-kl\lambda^2}.$$

Plugging this inequality in equation (30), we obtain:

$$\begin{aligned}(1-k(l-1)\lambda^2)v_k^{(k,l-1)} &\geq (1-kl\lambda^2)v_k^{(k,l)} + \frac{k\lambda^2}{1-kl\lambda^2}\left(\lambda\sum_{j=1}^{l}|y_j| - l\lambda^2\sum_{i=1}^{k}|x_i|\right) + \lambda|y_l| - \lambda^2\sum_{i=1}^{k}|x_i| \\ &= (1-kl\lambda^2)v_k^{(k,l)} + \frac{\lambda}{1-kl\lambda^2}\left(k\lambda^2\sum_{j=1}^{l}|y_j| - kl\lambda^3\sum_{i=1}^{k}|x_i| + (1-kl\lambda^2)|y_l| - \lambda(1-kl\lambda^2)\sum_{i=1}^{k}|x_i|\right) \\ &= (1-kl\lambda^2)v_k^{(k,l)} + \frac{\lambda}{1-kl\lambda^2}\left(k\lambda^2\sum_{j=1}^{l}|y_j| + (1-kl\lambda^2)|y_l| - \lambda\sum_{i=1}^{k}|x_i|\right).\end{aligned} \tag{31}$$

From the definition of $w_l^{(k,l)}$ (equation (26)), we have that $w_l^{(k,l)} \geq 0$ is equivalent to the condition:

$$(1-kl\lambda^2)|y_l| + k\lambda^2\sum_{j=1}^{l}|y_j| - \lambda\sum_{i=1}^{k}|x_i| \geq 0. \tag{32}$$

Since the expression of equation (32) is exactly the same as the one inside the parentheses of equation (31), plugging this relation to (30) thus shows that $(1-k(l-1)\lambda^2)v_k^{(k,l-1)} \geq 0$, i.e. $v_k^{(k,l-1)} \geq 0$. □

We now introduce the efficient procedure to compute the maximal feasibility boundary (MFB), and prove that it indeed delivers, as promised, all sparsity pairs in the MFB set.

Lemma 18. *The set S returned by Algorithm 5 contains all, and only, the sparsity pairs that are on the maximal feasibility boundary.*

Proof. First recall that the MFB is defined as all pairs $(s_v, s_w) \in \{0,\dots,p\} \times \{0,\dots,m\}$ satisfying the conditions:

1. $v_{s_v}^{(s_v,s_w)} > 0$ and $w_{s_w}^{(s_v,s_w)} > 0$ and $s_v s_w \leq \lambda^{-2}$,
2. $v_{s_v+1}^{(s_v+1,s_w)} \leq 0$ or $w_{s_w}^{(s_v+1,s_w)} \leq 0$ or $(s_v+1)s_w > \lambda^{-2}$ or $s_v = p$,

Algorithm 5 Finding sparsity pairs on the maximal feasibility boundary

Input: $x \in \mathbb{R}^p$, $y \in \mathbb{R}^m$ ordered in decreasing magnitude order, $\lambda > 0$.

1: $s_v \leftarrow 0$, $s_w \leftarrow m$
2: $S \leftarrow \emptyset$
3: $maximal \leftarrow True$
4: **while** $s_v \leq p$ and $s_w \geq 0$ **do**
5:     Compute $v_{s_v}^{(s_v,s_w)}$ and $w_{s_w}^{(s_v,s_w)}$ as shown in equation (26)
6:     **if** $v_{s_v}^{(s_v,s_w)} < 0$ or $w_{s_w}^{(s_v,s_w)} < 0$ or $s_v s_w \geq \lambda^{-2}$ **then**
7:         **if** $maximal$ **then**
8:             $S \leftarrow S \cup \{(s_v - 1, s_w)\}$
9:             $maximal \leftarrow False$
10:         **end if**
11:         $s_w \leftarrow s_w - 1$
12:     **else**
13:         $s_v \leftarrow s_v + 1$
14:         $maximal \leftarrow True$
15:     **end if**
16: **end while**
17: **if** $s_v == p + 1$ **then**
18:     $S \leftarrow S \cup \{(s_v - 1, s_w)\}$
19: **end if**
20: **return** S

3. $v_{s_v}^{(s_v,s_w+1)} \leq 0$ or $w_{s_w+1}^{(s_v,s_w+1)} \leq 0$ or $s_v(s_w + 1) > \lambda^{-2}$ or $s_w = m$.

Algorithm 5 plays on the properties of *feasibility-infeasibility* of the sparsity levels to build the MFB. We say that a pair of the sparsity levels of v and w (s_v, s_w) is *feasible* if $v_{s_v}^{(s_v,s_w)} \geq 0$, $w_{s_w}^{(s_v,s_w)} \geq 0$ and $s_v s_w < \lambda^{-2}$, and denote this by the property $P(i, j)$, i.e.

$$(i, j) \text{ is feasible } \Leftrightarrow P(i, j).$$

Our claim can be read as: Let $(i, j) \in \{0, \dots, p\} \times \{0, \dots, m\}$, then (i, j) is added to S by Algorithm 5 if and only if (i, j) belongs to the MFB, i.e.,

$$(i, j) \in \text{MFB} \Leftrightarrow (i, j) \in S.$$

Obviously, only feasible sparsity pairs belong to the MFB, and it is quite easy to see that only feasible sparsity pairs will belong to an output S of Algorithm 5. Indeed, Algorithm 5 monotonically decrements s_w starting from $s_w = m$ and increments s_v starting from $s_v = 0$. For each value of s_w, it increases s_v while the current pair (s_v, s_w) is feasible (lines $12 - 15$). Once it reaches an infeasible point (i, s_w), and in the case where s_v has been increased at least once for this particular value of s_w, it adds to S the pair encountered just before, i.e., $(i - 1, s_w)$, and then decrements s_w (lines $6 - 11$).

We first prove the $\Rightarrow$ statement. Suppose that some pair (i, j) belongs to the MFB. Let us first leave aside the corner cases, and assume that $i < p$ and $j < m$.

Suppose first that s_w reaches j before s_v reaches i, i.e., $s_v < i$. Since the pair (i, j) is feasible, and due to the monotonicity property of the feasibility condition (Lemma 9), all pairs (k, s_w) with $k \leq i$ must be feasible. Therefore, s_v will be increased until reaching $i + 1$. By definition of the MFB, the pair $(i + 1, j)$ must be infeasible. Since s_v has necessarily been increased at least once for this value of $s_w = j$, and so the pair $(i + 1 - 1, j) = (i, j)$ will be added to S before decrementing s_w.

In the special case where $i = p$, no infeasible point will be found. The loop will thus finish with $s_w = j$ and $s_v = p + 1$. The condition at line 17 will thus hold, and the pair (p, j) will be added to S.

Suppose now that s_v reaches i before s_w reaches j, i.e., $s_w > j$. Since (i, j) is in the MFB, then the pair $(i, j + 1)$ must be infeasible. Thanks to the monotonicity property of the feasibility condition (Lemma 9), all pairs (s_v, k) with $k \geq i$ must also be infeasible. Therefore, s_w will be decreased until reaching $s_w = j$. Then, similarly as in the previous case, since (i, j) is feasible, s_v will be increased, and the pair (i, j) added to S.

We now prove the $\Leftarrow$ statement. We show that if (i, j) is added to S, then it must belong to the MFB, i.e., it satisfies all three properties recalled in the beginning of the proof.

Let us first show that for each pair (s_v, s_w) encountered during the algorithm, the pair $(s_v - 1, s_w)$ is always feasible (or $s_v = 0$). We can show that this property is conserved each time the algorithm either increases s_v or decreases s_w. First note that the pair $(0, m)$ is always feasible. The algorithm will then necessarily first goes to the pair $(1, m)$ and $P(1, m)$ is true. Then suppose that $P(s_v, s_w)$ is true for some pair (s_v, s_w) encountered during the algorithm. Then, if s_v is increases, it means that the pair (s_v, s_w) is feasible. The next encountered pair is then $(s_v + 1, s_w)$ and $P(s_v + 1, s_w)$ is true. On the other hand, suppose that s_w is decreased. The next encountered pair is thus $(s_v, s_w - 1)$. Since $P(s_v, s_w)$ is true, it means that $(s_v - 1, s_w)$ is feasible. By Lemma 9, it implies that $(s_v - 1, s_w - 1)$ is also feasible, and thus $P(s_v, s_w - 1)$ is true. We thus proved that $P(s_v, s_w)$ is true for any pair (s_v, s_w) encountered during the algorithm. Therefore, since any pair added to S is of the form $(s_v - 1, s_w)$ for some pair (s_v, s_w) encountered during the algorithm, then any pair added to S must be feasible.

The second property of the MFB is straightforward to show. Indeed, if $(i - 1, j)$ is added to S, it means that the pair (i, j) is infeasible due to condition on line 6.

Finally, the third property follows from the fact that, when reaching $s_w = j$, s_v must be increased at least once for adding a pair of the form (i, j) to S. Let $s_v^{(j)}$ be the value of s_v when the algorithm reaches $s_w = j$. We necessarily have $s_v^{(j)} \leq i$. This implies that the pair $(s_v^{(j)}, j + 1)$ is infeasible, otherwise s_v would have been increased to a greater value at the previous value $s_w = j + 1$. By Lemma 9, and since $s_v^{(j)} \leq i$ this implies that the pair $(i, j + 1)$ is also infeasible, hence the result.

□

Time complexity of Algorithm 5 At each iteration of the loop, either s_v is incremented by 1 or s_w is decremented by 1. Since s_v starts from 0 and s_w from m, and that the stopping criterion is $s_v > p$ or $s_w < 0$, it follows that the maximal number of iterations inside the loop is $m + p$. At each iteration, we must compute $v_{s_v}^{(s_v, s_w)}$ and $w_{s_w}^{(s_v, s_w)}$, which requires in particular to compute $\sum_{k=1}^{s_v} |x_k|$ and $\sum_{j=1}^{s_w} |y_j|$. However, these cumulative sums can be efficiently computed before the loop in time $\mathcal{O}(m + p)$, so that computing $v_{s_v}^{(s_v, s_w)}$ and $w_{s_w}^{(s_v, s_w)}$ inside the loop can be done in constant time. The overall complexity of this algorithm is thus $\mathcal{O}(m + p)$.

Moreover, we can see that each time we add a pair to S, we must both decrement s_w by 1 (just after adding the element in the algorithm), and increment s_v by 1 (in order for the boolean $maximal$ to become true again). Therefore, there can be at most $\min(m, p)$ pairs in the final set s at the end of the algorithm.

Merging all previous results, we can finally prove Theorem 4.

Proof of Theorem 4. Thanks to the separability argument, it is sufficient to prove that Algorithm 3 returns a solution of problem (11).

Lemma 7 states that given the number of nonzero elements $s_v = |\{k : v_k^* > 0\}|$, $s_w = |\{j : w_j^* > 0\}|$, the optimal solution (v^*, w^*) can be obtained in close form (equations (12), (13)).

Due the monotonicity property of the objective function h_λ (Lemma 9), it follows that the sparsity pair (s_v, s_w) of the optimal solution must lie on the MFB. Indeed, if it does not lie on the MFB, then it means that the candidate solution associated with either the sparsity pair $(s_v + 1, s_w)$ or $(s_v, s_w + 1)$ must be feasible. According to Lemma 9, this pair would then yield a lower value of h_λ, and would then be a better solution.

Algorithm 3 computes the candidate solution associated with all sparsity pair lying on the MFB, and returns the one achieving the lowest value of h_λ. Therefore, the returned solution must necessarily be the optimal solution. □

E. Experimental details and other plots

We consider the following values for the parameters that determine the training loop:

- ▷ batch size: `100`
- ▷ epochs: `20`

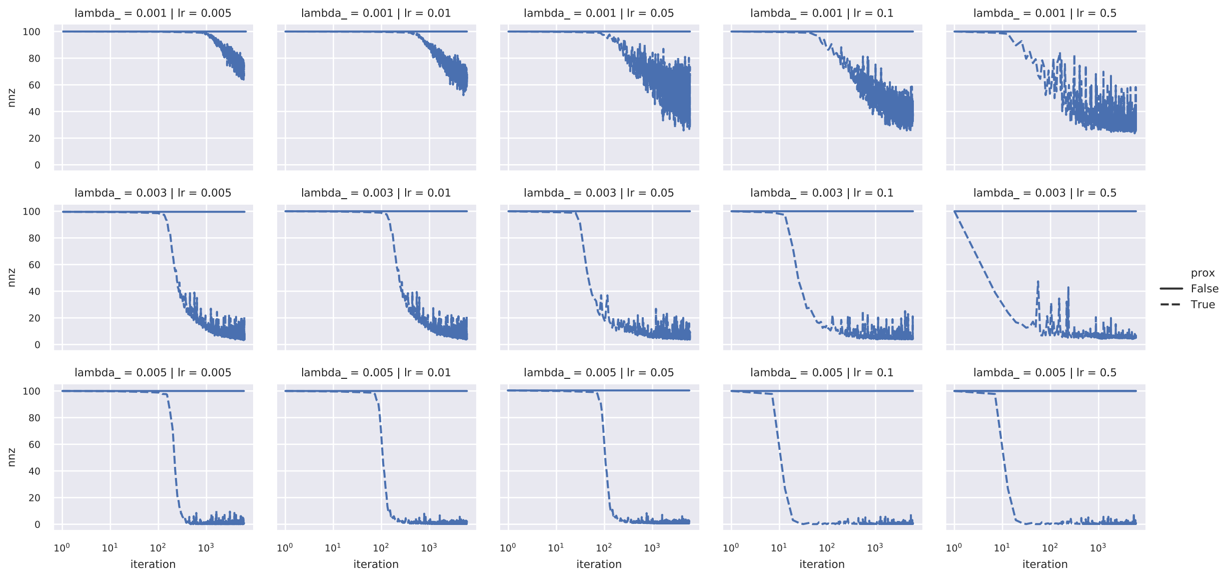

Figure 4. percentage of nonzero weights in the network, as a function of iteration count (path regularization - fmnist dataset).

- ▷ learning rate: `1e-1, 1e-2, 1e-3, 1e-4, 5e-1, 5e-2, 5e-3, 5e-4`
- ▷ dataset: `mnist, fmnist, kmnist`
- ▷ hidden neurons: `200`
- ▷ lambda (λ): `0., 1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1e0, 1e1, 1e2, 2e-5, 2e-4, 2e-3, 2e-2, 2e-1, 2e0, 2e1, 2e2, 3e-5, 3e-4, 3e-3, 3e-2, 3e-1, 3e0, 3e1, 3e2, 4e-5, 4e-4, 4e-3, 4e-2, 4e-1, 4e0, 4e1, 4e2, 5e-5, 5e-4, 5e-3, 5e-2, 5e-1, 5e0, 5e1, 5e2`

The ℓ_∞-bounded adversarial examples used to evaluate the robustness of the networks were generated using the PGD method described in (Madry et al., 2018) and implemented in the *advertorch* toolbox (`https://github.com/BorealisAI/advertorch`) using the following parameters:

- ▷ epsilon: `0.05, 0.1, 0.15, 0.2, 0.25, 0.3`
- ▷ iterations: 40
- ▷ step size: epsilon / 20
- ▷ random initialization: `True`

E.1. sparsity per iteration

One advantage of the proximal mapping of the 1-path-norm and the ℓ_1-norm is that they can set many weights to exactly zero. This has the effect of providing sparse networks from early iterations. This is in contrast to SGD with a constant stepsize which does not generate sparse iterates. In Figures 4, 5, 6 and 7 we plot the percentage of nonzero weights as a function of the iteration count, for both plain SGD and proximal SGD. We observe that in fact this is the case, and that the sparsity of the ℓ_1 and 1-path-norm regularized network can be controlled with the regularization parameter λ.

E.2. Robustness vs accuracy tradeoff

For all possible values of λ, in Figure 8 we plot the data corresponding to the lerning rate with least error. We plot the value of the error on clean samples and the error on adversarial examples. This allows us to understand the tradeoff between accuracy and robustness that is controlled by the regularization paramter λ.

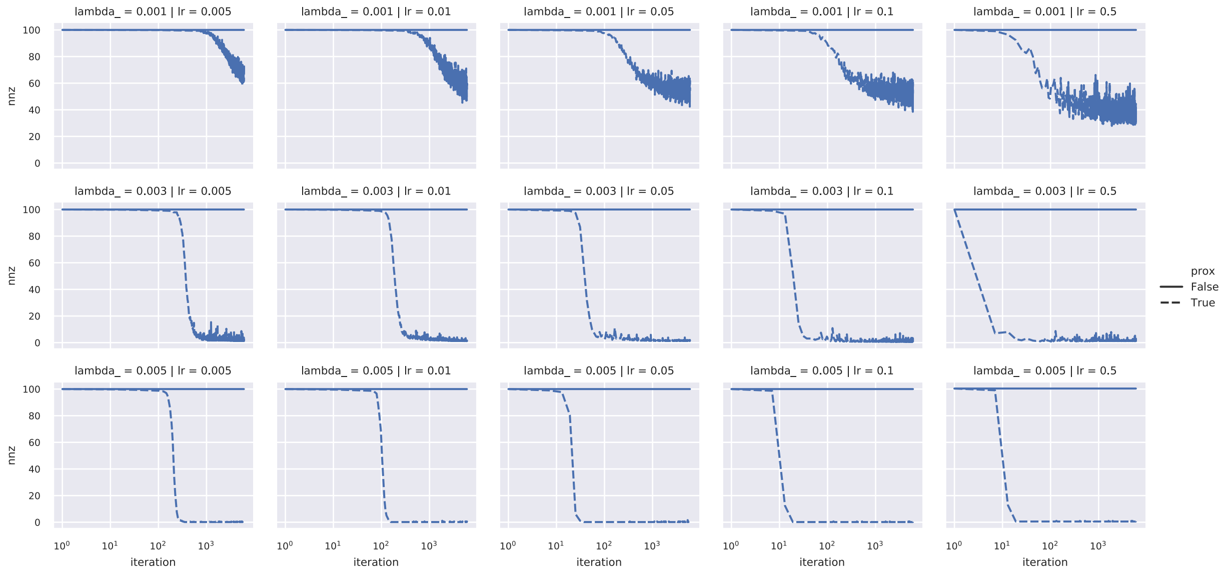

Figure 5. percentage of nonzero weights in the network, as a function of iteration count (path regularization - kmnist dataset).

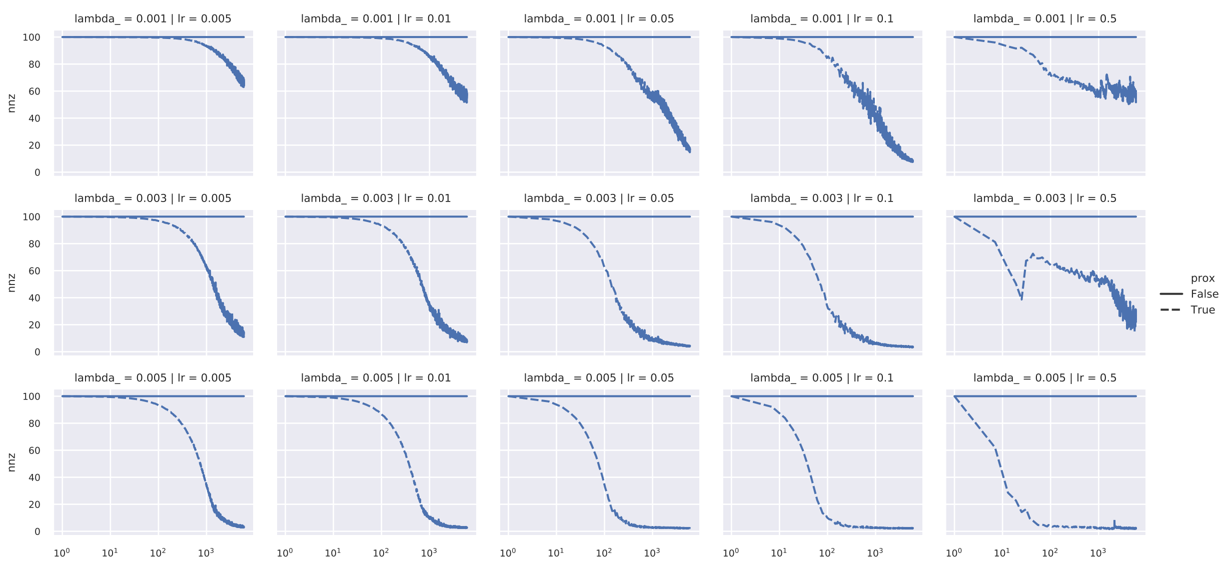

Figure 6. percentage of nonzero weights in the network, as a function of iteration count (ℓ_1 regularization - fmnist dataset).

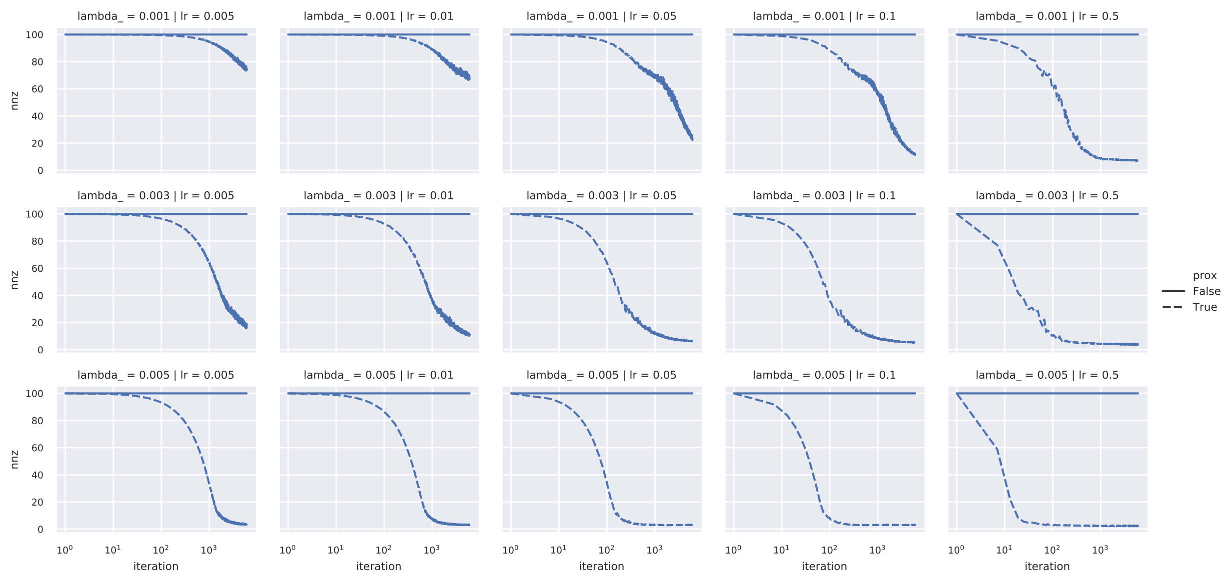

Figure 7. percentage of nonzero weights in the network, as a function of iteration count (ℓ_1 regularization - kmnist dataset).

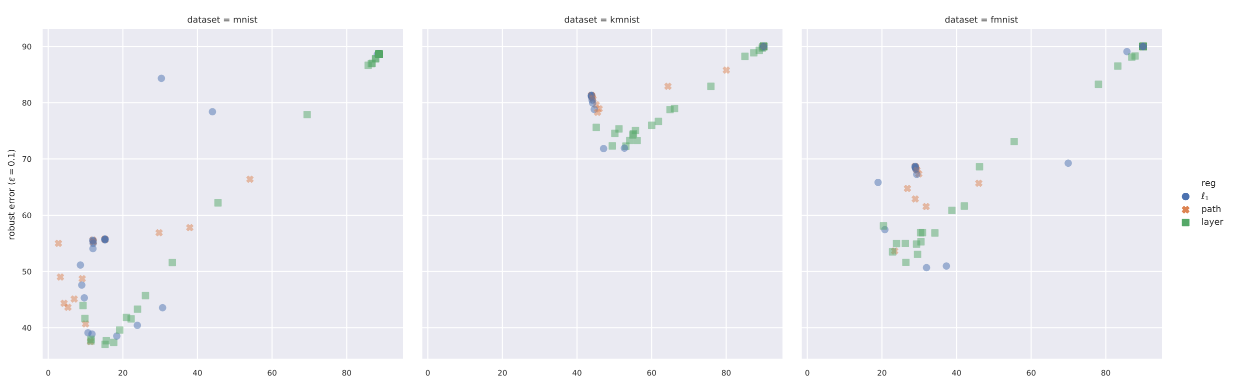

Figure 8. Robustness vs accuracy tradeoff for the different regularizers studied.