Equivalences Between Sparse Models and Neural Networks

Ryan J. Tibshirani

April 15, 2021

Abstract

We present some observations about neural networks that are, on the one hand, the result of fairly trivial algebraic manipulations, and on the other hand, potentially noteworthy and deserving of further study. A summary is as follows.

- The lasso is equivalent to a two-layer neural network fit with weight decay (i.e., with a ridge penalty placed on all of the parameters), linear activation functions, no bias terms, and a very simple connectivity structure.
- A k-layer neural network that has otherwise the same structure is in turn equivalent to an ℓ_p-penalized regression problem, where $p = 2/k < 1$.
- Similar equivalences hold for regression problems in which we seek group sparsity (the group lasso, and an ℓ_p variant of the group lasso for $p < 1$) and neural networks with richer connectivity structures.
- All of these equivalences extend to any loss function (not just squared loss, as is traditional in regression).

Lastly, we present equivalent representations for fully-connected neural networks that use rectified linear unit (ReLU) activation functions, and have two or three layers. These representations may help shed light on how weight decay can be sparsity-inducing in such network structures.

1 Introduction

In this note, we make some basic observations about reparametrizing certain neural network optimization problems. On the one hand, the arguments that we present are elementary, and in many cases, quite trivial. On the other, they lead to interesting and potentially fruitful directions for further study, and we have not seen the same observations presented in the literature (though to be fair, several related ideas have appeared before, and we draw connections to them).

2 Sparse models

2.1 Two-layer case

For responses $y_i \in \mathbb{R}$, $i = 1, \ldots, n$, feature vectors $x_i \in \mathbb{R}^d$, $i = 1, \ldots, n$, and a tuning parameter $\lambda \geq 0$, we consider the standard lasso problem

$$\underset{\beta}{\text{minimize}} \quad \sum_{i=1}^n (y_i - x_i^\mathsf{T}\beta)^2 + 2\lambda \sum_{j=1}^d |\beta_j|. \tag{1}$$

We note the following simple observation: for any c,

$$\min_{ab=c} \, (a^2 + b^2) = 2|c|,$$

the minimum being achieved at $a = b = \sqrt{|c|}$. In other words, this is just saying that the arithmetic mean-geometric mean (AM-GM) inequality $\frac{1}{2}(a^2 + b^2) \geq ab$ is tight when $a = b$. We can apply this observation to (1): by introducing $u, v \in \mathbb{R}^d$ such that $\beta_j = u_j v_j$, $j = 1, \ldots, d$, we can rewrite problem (1) as

$$\underset{u,v}{\text{minimize}} \quad \sum_{i=1}^n \bigg(y_i - \sum_{j=1}^d x_{ij} u_j v_j \bigg)^2 + \lambda \sum_{j=1}^d (u_j^2 + v_j^2). \tag{2}$$

The above problem is an optimization over a two-layer neural network[1] that we call *simply-connected*: each input unit j is connected to hidden unit j (and no others). We have all linear activation functions, with no bias parameters; we also have a ridge penalty on all parameters, which is equivalent to training via gradient descent with weight decay.

Although the equivalence of (1), (2) is a result of trivial algebra, it is a bit surprising, at least at first glance. Due to the sparsity-inducing nature of the ℓ_1 norm in (1), we see that solutions in (2) must also be sparse. But this would not be immediately obvious by simply looking at (2) on its own. That sparse solutions arise in this problem–where a ridge penalty is combined with a product parametrization (in the loss function)—is somewhat counterintuitive, and suggests some interesting geometry may be at play in the nonconvex, higher-dimensional problem (2) ("higher-dimensional" in the sense that it has $2d$ variables, compared to d variables in (1)).

Another perspective. There is a different way to see the equivalence of (1), (2); it is a bit circuitous compared to the above reasoning, but we tend to like it because we find that it helps to see how to derive other equivalences, presented after this section. First, note that, without a loss of generality, we may constrain $v \geq 0$ in problem (2) (as the signs can always be absorbed into the components of u), yielding

$$\underset{u,\, v \geq 0}{\text{minimize}} \quad \sum_{i=1}^n \bigg(y_i - \sum_{j=1}^d x_{ij} u_j v_j \bigg)^2 + \lambda \sum_{j=1}^d (u_j^2 + v_j^2).$$

Now reparametrize, using

$$v_j = \sqrt{\alpha_j},\ u_j = \beta_j / \sqrt{\alpha_j}, =\ j = 1, \ldots, d,$$

to give the equivalent problem

$$\underset{\alpha \geq 0,\, \beta}{\text{minimize}} \quad \sum_{i=1}^n (y_i - x_i^\mathsf{T} \beta)^2 + \lambda \sum_{j=1}^d (\beta_j^2 / \alpha_j + \alpha_j). \tag{3}$$

(Some care needs to be taken to handle division by zero: we interpret the jth summand be 0 if $\alpha_j = \beta_j = 0$, and ∞ if $\alpha_j = 0$ but $\beta_j \neq 0$.) Lastly, that (3) is equivalent to the lasso problem (1) is given by the following simple lemma.

Lemma 1. *The function $g : \mathbb{R}_+ \times \mathbb{R} \to \mathbb{R}$ defined by*

$$g(x,y) = \begin{cases} \frac{1}{2}(y^2/x + x) & x \neq 0 \\ 0 & x = y = 0 \\ \infty & x = 0,\ y \neq 0 \end{cases}$$

has

$$\min_{x \geq 0}\ g(x,y) = |y|,$$

with the minimum achieved at $x = |y|$.

To reiterate, we have argued that (2) $\iff$ (3) $\iff$ (1) (where we write $P \iff Q$ to denote that problems P, Q are equivalent). Introducing (3) as a middleman to equate (2) and (1) is a strategy that we will mimick both in the next subsection (on the multi-layer case), and in the coming sections (on group sparsity and fully-connected networks).

Before moving to the next subsection, we make a few remarks.

General loss functions. There is nothing special here about squared loss, and the same equivalence actually holds for any loss function L. The exact same arguments as above show that for any L (convex or not), the problem

$$\underset{u,v}{\text{minimize}} \quad \sum_{i=1}^n L\bigg(y_i, \sum_{j=1}^d x_{ij} u_j v_j \bigg) + \lambda \sum_{j=1}^d (u_j^2 + v_j^2). \tag{4}$$

is equivalent to

$$\underset{\beta}{\text{minimize}} \quad \sum_{i=1}^n L(y_i, x_i^\mathsf{T} \beta) + 2\lambda \sum_{j=1}^d |\beta_j|. \tag{5}$$

[1]The structure in (2) hardly deserves to be referred to as a neural network, because it is so simple, and there is a symmetry between layers—we can interpret u as the first-layer weights and v as the second-layer weights, or vice versa. (In a standard neural network, the aspect of composition with nonlinear activation functions would of course result in asymmetry between the layers.) That said, we will still refer to it as a neural network; and the networks we study in this short note will become progressively less simple and more structured.

Matrix factorization. The equivalence between (4), (5) is quite related to what are now well-known results in the matrix factorization literature. From Srebro et al. (2004), we know that for any matrix $Y \in \mathbb{R}^{n \times d}$ and loss L,

$$\underset{M}{\text{minimize}} \quad L(Y, M) + 2\lambda \|M\|_* \tag{6}$$

(with $\|\cdot\|_F$ denoting the Frobenius norm, and $\|\cdot\|_*$ denoting the trace norm) is equivalent to

$$\underset{U,V}{\text{minimize}} \quad L(Y, UV^\mathsf{T}) + \lambda(\|U\|_F^2 + \|V\|_F^2). \tag{7}$$

This is in fact, at its core, the same equivalence as that between (4), (5). For diagonal matrices, the trace norm is the ℓ_1 norm of the diagonal, and the Frobenius norm is the ℓ_2 norm of the diagonal.

Further, for problems like low-rank matrix completion, the equivalent representation given by problem (7), though nonconvex, can often present significant computational advantages, see, e.g., Hastie et al. (2015). This motivates the next remark.

Large-scale optimization. To solve a large-scale ℓ_1-penalized problem (5), it might be worth pursuing optimization in the formulation (4). While latter is nonconvex (even for convex L), the modern deep learning optimization toolkit can be incredibly efficient (in large part, it seems, thanks to advances in hardware). It can also be surprisingly effective at delivering practically reasonable (local) solutions to continuous, nonconvex problems. In this sense, it seems worth applying this toolkit to (4) at scale, to see how it fares (especially for smooth L beyond squared loss, e.g., for logistic or multinomial regression loss, in which case (5) becomes more challenging than in the squared loss case).

Early stopping. It is well-known that early-stopped gradient descent (as well as stochastic gradient descent) admits practical and formal connections to ridge regularization; see, e.g., Ali et al. (2019, 2020) and references therein. In this light, we could apply the deep learning optimization toolkit to problem (4), but instead of using ridge penalties (instead of weight decay), we could use early stopping. It would be reasonable to expect that the early-stopped iterates would approximate solutions in (5) (where the stopping time would be in inverse correspondence with the tuning parameter λ, i.e., earlier stopping would mean heavier regularization).

2.2 Multi-layer case

Consider now a k-layer neural network, again with simple connections, and linear activations (with no bias terms),

$$\underset{w^{(\ell)},\, \ell=1,\ldots,k}{\text{minimize}} \quad \sum_{i=1}^n L\bigg(y_i, \sum_{j=1}^d x_{ij} \prod_{\ell=1}^k w_j^{(\ell)}\bigg) + \lambda \sum_{j=1}^d \sum_{\ell=1}^k (w_j^{(\ell)})^2. \tag{8}$$

Similar to the two-layer case, we note the following simple observation: for any c,

$$\min_{\prod_{\ell=1}^k a_\ell = c} \bigg(\sum_{\ell=1}^k a_\ell^2\bigg) = k|c|^{2/k},$$

the minimum being achieved at $a_\ell = |c|^{1/k}$, $\ell = 1, \ldots, k$. As before, we can interpret this as saying that the AM-GM inequality $\frac{1}{k}\sum_{\ell=1}^k a_\ell^2 \geq \prod_{\ell=1}^k a_\ell^{2/k}$ is tight when $a_1 = \cdots = a_k$. Applying this fact to (8), we can rewrite it as

$$\underset{\beta}{\text{minimize}} \quad \sum_{i=1}^n L\bigg(y_i, \sum_{j=1}^d x_{ij}\beta_j\bigg) + k\lambda \sum_{j=1}^d |\beta_j|^{2/k}. \tag{9}$$

The above problem uses an ℓ_p penalty, where $p = 2/k$, as its regularizer. Note that when we have $k > 2$ layers in (8), we have $p = 2/k < 1$, which makes the penalty in (9) nonconvex. Note also that in the infinite depth limit ($k \to \infty$), we get an ℓ_0 penalty in (9) (we have to shrink λ appropriately as we grow k, in order to end up with a finite effective regularization parameter in (9)).

The algebra equating (8), (9) is again quite trivial. Nonetheless, the connection seems surprising and interesting. From what we know about how ℓ_p regularization acts in the latter problem (9), we see that the former problem (8) admits sparse solutions, even more so for greater depth (larger k).

Another perspective. As before, there is a different way to see the equivalence between (8), (9). The next lemma generalizes Lemma 1, though it is still very simple.

Lemma 2. *For any $r > 0$, the function $g_r : \mathbb{R}_+ \times \mathbb{R} \to \mathbb{R}$ defined by*

$$g_r(x,y) = \begin{cases} \frac{1}{r+1}(y^2/x^r + rx) & x \neq 0 \\ 0 & x = y = 0 \\ \infty & x = 0,\ y \neq 0 \end{cases}$$

has

$$\min_{x \geq 0}\ g_r(x,y) = |y|^{\frac{2}{r+1}},$$

with the minimum achieved at $x = |y|^{\frac{2}{r+1}}$.

Returning to (8), without a loss of generality, we can introduce the constraint $w^{(k)} \geq 0$, then reparametrize via

$$w_j^{(k)} = \sqrt{\alpha_j},\ w_j^{(k-1)} = \beta_j/\sqrt{\alpha_j},\ j = 1, \ldots, d.$$

Applying Lemma 2 (with $r = 1$), we can collapse the last two layers into one by minimizing over $\alpha \geq 0$, and then for simplicity we can change notation back to $w^{(k-1)} = \beta$. This yields

$$\underset{w^{(\ell)},\, \ell=1,\ldots,k-1}{\text{minimize}} \quad \sum_{i=1}^n L\bigg(y_i, \sum_{j=1}^d x_{ij} \prod_{\ell=1}^{k-1} w_j^{(\ell)}\bigg) + \lambda \sum_{j=1}^d \bigg(\sum_{\ell=1}^{k-2} (w_j^{(\ell)})^2 + 2|w_j^{(k-1)}|\bigg).$$

Proceeding along the same lines, we introduce the constraint $w^{(k-1)} \geq 0$, reparametrize via

$$w_j^{(k-1)} = \alpha_j,\ w_j^{(k-2)} = \beta_j/\alpha_j,\ j = 1, \ldots, d,$$

and collapse the last two layers into one, by applying Lemma 2 (with $r = 2$), yielding

$$\underset{w^{(\ell)},\, \ell=1,\ldots,k-2}{\text{minimize}} \quad \sum_{i=1}^n L\bigg(y_i, \sum_{j=1}^d x_{ij} \prod_{\ell=1}^{k-2} w_j^{(\ell)}\bigg) + \lambda \sum_{j=1}^d \bigg(\sum_{\ell=1}^{k-3} (w_j^{(\ell)})^2 + 3|w_j^{(k-2)}|^{2/3}\bigg).$$

Carrying on collapsing layers in this manner, we arrive at the ℓ_p-penalized problem (9).

The remarks about optimization at the end of Section 2.1 in the discussion of the equivalence between (1), (2) all carry over to the current setting as well. We reiterate that it would be interesting to approximately solve (9) by attacking (8) with the modern deep learning optimization toolkit. This could be potentially even more fruitful than solving (1) via (2), since problem (9) for $k > 2$ is itself nonconvex to begin with, and generally much harder to solve than (1).

We make one more remark before moving on to discuss group sparsity.

Hardness. For any $p < 1$, it is known that ℓ_p-penalized problems of the form (9) are strongly NP-hard (Chen et al., 2015), for a wide class of loss functions L (this includes common ones like squared loss and logistic loss). The same result therefore applies to problem (8), for each $k > 2$. This gives a sense of how difficult it must be to train a neural network to global optimality, in general: even an extremely simple neural network, with all simple connections and linear activations, is strongly NP-hard once we have $k > 2$ layers!

On the other hand, this result may have little practical relevance, because as already stated, current deep learning optimization toolkits seem to be able to produce useful local solutions at huge scale.

3 Group-sparse models

3.1 Two-layer case

We now extend the results to the case of a more interesting connectivity structure, and as we will see, the group lasso arises as an equivalence in the two-layer case (whose groups are in correspondence with the connectivity structure). Consider

$$\underset{w^{(1)}, w^{(2)}}{\text{minimize}} \quad \sum_{i=1}^n L\bigg(y_i, \sum_{j=1}^{d_2} x_{i,G_j}^T w_j^{(1)} w_j^{(2)}\bigg) + \lambda \bigg(\sum_{j=1}^{d_2} \|w_j^{(1)}\|_2^2 + \|w^{(2)}\|_2^2\bigg). \tag{10}$$

In this network, we have all linear activation functions with no bias terms. There are d_2 units in the output layer, and each output unit j is connected to all input units in some set G_j. (Take $G_j = \{1, \ldots, d\}$ and we get a fully-connected network; take $G_j = \{j\}$ and we get the simply-connected networks considered previously.)

Now take $w^{(2)} \geq 0$ without a loss of generality, reparametrize via

$$w_j^{(2)} = \sqrt{\alpha_j},\ w_j^{(1)} = \beta_j / \sqrt{\alpha_j},\ j = 1, \ldots, d_2,$$

and minimize over $\alpha \geq 0$ (applying Lemma 1 with $x = \sqrt{\alpha_j}$ and $y = \|\beta_j\|_2$, for each $j = 1, \ldots, d_2$) to yield

$$\underset{\beta}{\text{minimize}} \quad \sum_{i=1}^n L\Big(y_i, \sum_{j=1}^{d_2} x_{i,G_j}^T \beta_j\Big) + 2\lambda \sum_{j=1}^{d_2} \|\beta_j\|_2. \tag{11}$$

Thus the two-layer neural network (10) is evidently equivalent to (11), which is a kind of group lasso problem.

3.2 Multi-layer case

In the multi-layer case, it seems the most interesting extension of the previous two-layer result comes from a network structure with group connections between the first and second layers, and simple connections between all other layers. That is, consider

$$\underset{w^{(\ell)},\, \ell=1,\ldots,k}{\text{minimize}} \quad \sum_{i=1}^n L\Big(y_i, \sum_{j=1}^{d_2} x_{i,G_j}^T w_j^{(1)} \prod_{\ell=2}^k w_j^{(\ell)}\Big) + \Big(\sum_{j=1}^{d_2} \|w_j^{(1)}\|_2^2 + \sum_{\ell=2}^k \|w^{(\ell)}\|_2^2\Big). \tag{12}$$

Then virtually the same arguments as in the previous multi-layer result (when we have all simple connections) produce the equivalent group sparse problem:

$$\underset{\beta}{\text{minimize}} \quad \sum_{i=1}^n L\Big(y_i, \sum_{j=1}^{d_2} x_{i,G_j}^T \beta_j\Big) + k\lambda \sum_{j=1}^{d_2} \|\beta_j\|_2^{2/k}. \tag{13}$$

The penalty in problem (13) is a kind of ℓ_p variant of the group lasso penalty, with $p = 2/k$. Note that when we have $k > 2$ layers in (12), this gives $p = 2/k < 1$, and the penalty in (13) is nonconvex.

The remarks about optimization made at the end of Section 2.1 in reference to problems (1), (2) all carry over to the connection between (12), (13) (and (10), (11) in the case $k = 2$) as well. Even just the group lasso (11), at large problem sizes, is considerably more challenging computationally than the lasso (1). Of course, the ℓ_p variant (13) only makes matters worse, computationally. Therefore, it could be useful to explore optimization in (10) or (12), with a deep learning toolkit.

4 Fully-connected networks

4.1 Two-layer case

We apply the same ideas in order to derive equivalences for fully-connected neural networks with rectified linear unit (ReLU) activation functions, and weight decay. Starting with the two-layer case, consider

$$\underset{b^{(1)}, w^{(1)}, w^{(2)}}{\text{minimize}} \quad \sum_{i=1}^n L\Big(y_i, \sum_{j=1}^{d_2} \phi(x_i^T w_j^{(1)} + b_j^{(1)}) w_j^{(2)}\Big) + \lambda\Big(\sum_{j=1}^{d_2} \|w_j^{(1)}\|_2^2 + \|w^{(2)}\|_2^2\Big). \tag{14}$$

Here $\phi(x) = \max\{x, 0\}$ is the ReLU function. Note that this has the important positive homogeneity property:

$$\phi(ax) = \text{sign}(a)\phi(|a|x).$$

This allows us to rewrite (14) as

$$\underset{\substack{b^{(1)}, w^{(1)}, w^{(2)} \geq 0, \\ s^{(2)} \in \{-1,1\}^{d_2}}}{\text{minimize}} \quad \sum_{i=1}^n L\Big(y_i, \sum_{j=1}^{d_2} \phi(x_i^T w_j^{(1)} w_j^{(2)} + b_j^{(1)} w_j^{(2)}) s_j^{(2)}\Big) + \lambda\Big(\sum_{j=1}^{d_2} \|w_j^{(1)}\|_2^2 + \|w^{(2)}\|_2^2\Big).$$

Reparametrize via

$$w_j^{(2)} = \sqrt{\alpha_j},\ w_j^{(1)} = \beta_j/\sqrt{\alpha_j},\ b_j^{(1)} = \gamma_j/\sqrt{\alpha_j},\ j = 1, \ldots, d_2,$$

then minimize over $\alpha \geq 0$ (applying Lemma 1 with $x = \sqrt{\alpha_j}$ and $y = \|\beta_j\|_2$, for each $j = 1, \ldots, d_2$) to obtain

$$\underset{\alpha,\, \beta,\, s^{(2)} \in \{-1,1\}^{d_2}}{\text{minimize}} \quad \sum_{i=1}^n L\bigg(y_i, \sum_{j=1}^{d_2} \phi(x_i^\mathsf{T} \beta_j + \gamma_j) s_j^{(2)}\bigg) + 2\lambda \sum_{j=1}^{d_2} \|\beta_j\|_2. \tag{15}$$

The equivalence between (14) and (15) is quite interesting. Due to what we know about the group lasso penalty, many of the parameters β_j, $j = 1, \ldots, d_2$ will be sparse at the solution in (15), which means that the same will be true of the feature maps

$$x \mapsto \phi(x_i^\mathsf{T} w_j^{(1)} + b_j^{(1)}) w_j^{(2)},\ j = 1, \ldots, d_2.$$

at the solution in (14). (That is, many of these functions will be either the zero map, or a constant map.) Therefore, a standard two-layer neural network, trained with weight decay, is actually "hunting" for a sparse representation in terms of its learned features!

We should be clear that this is not a new result: the equivalence between (14), (15) already appears in Theorem 1 by Neyshabur et al. (2015). It has been followed up on by several authors, in similar contexts (fully-connected ReLU networks, usually with two layers, sometimes infinitely wide), including Savarese et al. (2019); Ongie et al. (2020); Parhi and Nowak (2021). We are only stating it here, in the current note, because it fits the general theme of equating neural networks with sparse models and it follows from the same line of arguments.

4.2 Three-layer case

We now move to the three-layer case. We omit bias terms only for simplicity of exposition. Consider

$$\underset{w^{(1)}, w^{(2)}, w^{(3)}}{\text{minimize}} \quad \sum_{i=1}^n L\bigg(y_i, \sum_{j=1}^{d_3} \phi\Big(h^{(2)}(x_i)^\mathsf{T} w_j^{(2)}\Big) w_j^{(3)}\bigg) + \lambda\bigg(\sum_{j=1}^{d_2} \|w_j^{(1)}\|_2^2 + \sum_{j=1}^{d_3} \|w_j^{(2)}\|_2^2 + \|w^{(3)}\|_2^2\bigg), \tag{16}$$

where $h^{(2)}$ has component functions

$$h_j^{(2)}(x) = \phi(x^\mathsf{T} w_j^{(1)}),\ j = 1, \ldots, d_2.$$

By the arguments of the last subsection, we can reduce (16) to

$$\underset{\substack{w^{(1)}, w^{(2)} \\ s^{(3)} \in \{-1,1\}^{d_3}}}{\text{minimize}} \quad \sum_{i=1}^n L\bigg(y_i, \sum_{j=1}^{d_3} \phi\Big(h^{(2)}(x_i)^\mathsf{T} w_j^{(2)}\Big) s_j^{(3)}\bigg) + \lambda\bigg(\sum_{j=1}^{d_2} \|w_j^{(1)}\|_2^2 + \sum_{j=1}^{d_3} 2\|w_j^{(2)}\|_2\bigg),$$

Constraining $w^{(2)} \geq 0$ and introducing a separate sign variable, then using the positive homogeneity property of the ReLU function, we can rewrite the above as

$$\underset{\substack{w^{(1)},\, w^{(2)} \geq 0, \\ s^{(2)} \in \{-1,1\}^{d_3 \times d_2}, \\ s^{(3)} \in \{-1,1\}^{d_3}}}{\text{minimize}} \quad \sum_{i=1}^n L\bigg(y_i, \sum_{j=1}^{d_3} \phi\bigg(\sum_{\ell=1}^{d_2} \phi(x_i^\mathsf{T} w_\ell^{(1)} w_{j\ell}^{(2)}) s_{j\ell}^{(2)}\bigg) s_j^{(3)}\bigg) + \lambda\bigg(\sum_{j=1}^{d_2} \|w_j^{(1)}\|_2^2 + 2\sum_{j=1}^{d_3} \|w_j^{(2)}\|_2\bigg),$$

Now the reparametrization and partial minimization is a bit more tricky than it was before. First define

$$w_{j\ell}^{(2)} = \alpha_{j\ell},\ w_\ell^{(1)} = \beta_{j\ell}/\alpha_{j\ell},\ \ell = 1, \ldots, d_2,\ j = 1, \ldots, d_3.$$

Then the above problem becomes

$$\underset{\substack{\alpha \geq 0,\, \beta, \\ \beta_{j\ell} \propto \beta_{k\ell} \text{ for all } j,k,\ell \\ s^{(2)} \in \{-1,1\}^{d_3 \times d_2}, \\ s^{(3)} \in \{-1,1\}^{d_3}}}{\text{minimize}} \quad \sum_{i=1}^n L\bigg(y_i, \sum_{j=1}^{d_3} \phi\bigg(\sum_{\ell=1}^{d_2} \phi(x_i^\mathsf{T} \beta_{j\ell}) s_{j\ell}^{(2)}\bigg) s_j^{(3)}\bigg) + \lambda\bigg(\frac{1}{d_3} \sum_{\ell=1}^{d_2} \sum_{j=1}^{d_3} \frac{\|\beta_{j\ell}\|_2^2}{\alpha_{j\ell}} + 2\sum_{j=1}^{d_3} \sqrt{\sum_{\ell=1}^{d_2} \alpha_{j\ell}^2}\bigg).$$

There are two important points to note about the above problem. First, we placed additional constraints $\beta_{j\ell} \propto \beta_{k\ell}$ for all j, k, ℓ (where for vectors u, v, we use $u \propto u$ to denote that $u = cv$ for a scalar c) in the optimization problem, which are necessary to preserve the structure in the original parametrization. Second, while there was a degree of flexibility in expressing each $\|w_\ell^{(1)}\|_2^2$ in the new parametrization, we purposely chose

$$\|w_\ell^{(1)}\|_2^2 = \frac{1}{d_3} \sum_{j=1}^{d_2} \frac{\|\beta_{j\ell}\|_2^2}{\alpha_{j\ell}}$$

in order to make the partial optimization over $\alpha \geq 0$ analytically tractable. To that end, the following lemma is useful.

Lemma 3. *Define a function $g : \mathbb{R}_+^d \times \mathbb{R}^d \to \mathbb{R}$ by*

$$g(x, y) = \frac{1}{d} \sum_{\ell=1}^{d} \frac{y_\ell^2}{x_\ell^2} + 2\|x\|_2$$

(where we interpret the each summand as per the rules of Lemmas 1 and 2: the ℓth summand is 0 if $x_\ell = y_\ell = 0$, and ∞ if $x_\ell = 0$ but $y_\ell \neq 0$). Then

$$\min_{x \geq 0} \; g(x, y) = \frac{5}{d} \left(\frac{d}{2}\right)^{2/3} \left(\sum_{\ell=1}^{d} y_\ell\right)^{2/3},$$

with the minimum achieved at

$$x_\ell = \left(\frac{2}{d}\right)^{1/3} y_\ell^{1/2} \left(\sum_{m=1}^{d} y_m\right)^{1/6}, \; \ell = 1, \ldots, d.$$

Applying this lemma to the problem in the display that immediately precedes the lemma (i.e., applying it separately for each $j = 1, \ldots, d_3$, with $x = (\sigma_{j1}, \ldots, \sigma_{jd_2})$ and $y = (\|\beta_{j1}\|_2, \ldots, \|\beta_{jd_2}\|_2)$), yields

$$\underset{\substack{\beta_{j\ell} \propto \beta_{k\ell} \text{ for all } j,k,\ell \\ s^{(2)} \in \{-1,1\}^{d_3 \times d_2}, \\ s^{(3)} \in \{-1,1\}^{d_3}}}{\text{minimize}} \quad \sum_{i=1}^{n} L\left(y_i, \sum_{j=1}^{d_3} \phi\left(\sum_{\ell=1}^{d_2} \phi(x_i^T \beta_{j\ell}) s_{j\ell}^{(2)}\right) s_j^{(3)}\right) + \frac{5\lambda}{d_3} \left(\frac{d_3}{2}\right)^{2/3} \sum_{j=1}^{d_3} \left(\sum_{\ell=1}^{d_2} \|\beta_{j\ell}\|_2\right)^{2/3}.$$

Finally, we can eliminate the proportionality constraints by (re)introducing variables $\beta_{j\ell} = w_\ell^{(1)} w_{j\ell}^{(2)}$ for each j, ℓ, and further simplify by eliminating the sign variable $s^{(2)}$, yielding

$$\underset{w^{(1)},\, w^{(2)},\, s^{(3)} \in \{-1,1\}^{d_3}}{\text{minimize}} \quad \sum_{i=1}^{n} L\left(y_i, \sum_{j=1}^{d_3} \phi\left(\sum_{\ell=1}^{d_2} \phi(x_i^T w_\ell^{(1)}) w_{j\ell}^{(2)}\right) s_j^{(3)}\right) + \frac{5\lambda}{d_3} \left(\frac{d_3}{2}\right)^{2/3} \sum_{j=1}^{d_3} \left(\sum_{\ell=1}^{d_2} |w_{j\ell}^{(2)}| \|w_\ell^{(1)}\|_2\right)^{2/3}. \tag{17}$$

The equivalence between (16), (17) is similar to that in the two-layer case. However, similar to what happens in other multi-layer equivalences derived in this note, the penalty in (17) has become nonconvex. It therefore promotes sparsity to an even greater degree, in the learned feature maps.

Acknowledgements

This note was inspired by a conversation with Rob Nowak. We would also like to thank Rob for his insights in this and ensuing conversations, and for pointing us to the paper by Neyshabur et al. (2015) and follow up work.

References

Alnur Ali, J. Zico Kolter, and Ryan J. Tibshirani. A continuous-time view of early stopping for least squares. In *International Conference on Artificial Intelligence and Statistics*, 2019.

Alnur Ali, Edgar Dobriban, and Ryan J. Tibshirani. The implicit regularization of stochastic gradient flow for least squares. In *International Conference on Machine Learning*, 2020.

Yichen Chen, Dongdong Ge, Mengdi Wang, Zizhuo Wang, Yinyu Ye, and Hao Yin. Strong NP-hardness for sparse optimization with concave penalty functions. In *International Conference on Machine Learning*, 2015.

Trevor Hastie, Rahul Mazumder, Jason D. Lee, and Reza Zadeh. Matrix completion and low-rank SVD via fast alternating least squares. *Journal of Machine Learning Research*, 16(104):3367–3402, 2015.

Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. In search of the real inductive bias: on the role of implicit regularization in deep learning. In *International Conference on Learning Representations (Workshop)*, 2015.

Greg Ongie, Rebecca Willett, Daniel Soudry, and Nathan Srebro. A function space view of bounded norm infinite width ReLU nets: The multivariate case. In *International Conference on Learning Representations*, 2020.

Rahul Parhi and Robert D. Nowak. Banach space representer theorems for neural networks and ridge splines. *Journal of Machine Learning Research*, 22(43):1–40, 2021.

Pedro Savarese, Itay Evron, Daniel Soudry, and Nathan Srebro. How do infinite width bounded norm networks look in function space? In *Conference on Learning Theory*, 2019.

Nathan Srebro, Jason Rennie, and Tommi S. Jaakkola. Maximum-margin matrix factorization. In *Advances in Neural Information Processing Systems*, 2004.