A Library of Mirrors: Deep Neural Nets in Low Dimensions are Convex Lasso Models with Reflection Features

Emi Zeger*, Yifei Wang*, Aaron Mishkin†, Tolga Ergen*
Emmanuel Candès‡, and Mert Pilanci*
Stanford University

Key words. deep neural networks, convex optimization, LASSO, sparsity

MSC codes. 62M45, 46N10

Abstract. We prove that training neural networks on 1-D data is equivalent to solving convex Lasso problems with discrete, explicitly defined dictionary matrices. We consider neural networks with piecewise linear activations and depths ranging from 2 to an arbitrary but finite number of layers. We first show that two-layer networks with piecewise linear activations are equivalent to Lasso models using a discrete dictionary of ramp functions, with breakpoints corresponding to the training data points. In certain general architectures with absolute value or ReLU activations, a third layer surprisingly creates features that reflect the training data about themselves. Additional layers progressively generate reflections of these reflections. The Lasso representation provides valuable insights into the analysis of globally optimal networks, elucidating their solution landscapes and enabling closed-form solutions in certain special cases. Numerical results show that reflections also occur when optimizing standard deep networks using standard non-convex optimizers. Additionally, we demonstrate our theory with autoregressive time series models.

1. Introduction.

Training deep neural networks is an important optimization problem. However, the non-convexity of neural nets makes their training challenging. In this paper, we show that for low-dimensional data, e.g., 1-D or 2-D, training a deep neural network can be simplified to solving a convex Lasso problem with an easily constructable dictionary matrix.

Neural networks are used as predictive models for low-dimensional data in acoustic signal processing (5; 20; 21; 27; 32; 34), physics-informed machine learning problems, uncertainty quantification (9; 10; 36; 40; 41; 43), and predicting financial data (Section 5). In (33; 15; 16), the problem of learning 1-D data is studied for two-layer ReLU networks, and it is proved that an optimal two-layer ReLU neural network precisely interpolates the training data as a piecewise linear function for which the breakpoints are at the data points. Recent work in (22; 23; 26) also studied 2-layer ReLU neural networks and their behavior on 1-D data.

However, even for low-dimensional data, the current literature still lacks analysis on the expressive power and learning capabilities of deeper neural networks with generic activations. This motivates us to study the optimization of 2 and 3-layer networks with piecewise linear activations and deeper neural networks with sign and ReLU activations. For 1-D data, we simplify the training problem by recasting it as a convex Lasso problem, which has been extensively studied (12; 37; 38).

Convex analysis of neural networks was developed in several prior works. As an example,

*Department of Electrical Engineering
†Department of Computer Science
‡Departments of Statistics and Mathematics

infinite-width neural networks enable the convexification of the overall model (2; 4; 17). However, due to the infinite-width assumption, these results do not reflect finite-width neural networks in practice. Recently, a series of papers (14; 15; 31) developed a convex analytic framework for the training problem of two-layer neural networks with ReLU activation. As a follow-up work, a similar approach is used to formulate the training problem for threshold activations with data in general d-dimensions as a Lasso problem (13). However, the dictionary matrix is described implicitly and requires high computational complexity to create (13). By focusing on 1-D data, we can provide simple, explicit Lasso dictionaries and consider additional activations. For example, we analyze networks with sign activation, which is useful in contexts such as saving memory to meet hardware constraints (8; 25).

Throughout the paper, all scalar functions extend to vector and matrix inputs component-wise. We write vectors as $\mathbf{v} = (v_1, \cdots, v_n)$ and denote the set of n-dimensional, real-valued column and row vectors by $\mathbb{R}^n$ and $\mathbb{R}^{1\times n}$, respectively. For $L \geq 2$, an L-layer *neural network* for d-dimensional data is denoted by $f_L(\mathbf{x};\theta) : \mathbb{R}^{1\times d} \to \mathbb{R}$, where $\mathbf{x} \in \mathbb{R}^{1\times d}$ is an input row vector and θ is the *parameter set*. The set θ may contain matrices, vectors, and scalars representing weights and biases. We let $\theta \in \Theta$, where Θ is the *parameter space*. Let $\mathbf{X} \in \mathbb{R}^{N\times d}$ be a *data matrix* consisting of N *training samples* $\mathbf{x}_1, \cdots \mathbf{x}_N \in \mathbb{R}^{1\times d}$. We consider regression tasks and call $\mathbf{y} \in \mathbb{R}^N$ the *target vector*. The (non-convex) neural net *training problem* is

$$\min_{\theta\in\Theta} \frac{1}{2}\|f_L(\mathbf{X};\theta) - \mathbf{y}\|_2^2 + \frac{\beta}{\tilde{L}}\|\theta_w\|^{\tilde{L}} \tag{1.1}$$

where $\beta > 0$ is a regularization coefficient for a subset of parameters $\theta_w \subset \theta$ that incur a weight penalty when training. We denote $\|\theta_w\|^{\tilde{L}} = \sum_{q\in\theta_w} \|q\|_2^{\tilde{L}}$, which penalizes the total network weight. The results can be generalized to other p-norm regularizations as well. $\tilde{L}$ is the *effective regularized depth*, defined to be the usual depth L for ReLU, leaky ReLU, and absolute value activations, but defined to be just 2 for threshold and sign activations. This is motivated by the property that neurons with sign or threshold activations are invariant to the magnitude of their neuron weights, so it does not make sense to penalize the inner weights (Remark H.1).

A central element of this paper is the *Lasso problem*

$$\min_{\mathbf{z},\xi} \frac{1}{2}\|\mathbf{A}\mathbf{z} + \xi\mathbf{1} - \mathbf{y}\|_2^2 + \tilde{\beta}\|\mathbf{z}\|_1 \tag{1.2}$$

where $\mathbf{z}$ is a vector, $\xi \in \mathbb{R}$, $\mathbf{1}$ is a vector of ones, and $\tilde{\beta} > 0$ (whose relation to β in (1.1) is defined in Subsection 3.1). $\mathbf{A}$ is called the *dictionary matrix*, with columns $\mathbf{A}_i \in \mathbb{R}^N$.

A neural net is *trained* by searching for θ that optimizes (1.1). A neural net $f_L(\mathbf{x};\theta)$ is called *optimal* if θ is a global minimizer in (1.1). Unfortunately, training is complicated by the non-convexity of the optimization problem. However, for data of dimension d=1, we reformulate the training problem (1.1) into the equivalent but simpler Lasso problem (1.2), where $\mathbf{A}$ is a fixed matrix that is constructed based on the training data $\mathbf{X}$ and neural net architecture. Moreover, the dictionary matrix columns $\mathbf{A}_i$ correspond to piecewise linear functions we call *features* that satisfy the following: 1) the value of the feature when evaluated on x_n is $\mathbf{A}_{i,n}$, and 2) the function consisting of their affine combination, where the coefficients are elements of a Lasso solution, is equal to an optimal neural net. The set of features for a given network is a *dictionary*. We call a collection of dictionaries a *library* when referring to the features from multiple depths.

We explicitly provide the elements of $\mathbf{A}$, making it straightforward to build and solve the convex Lasso problem instead of solving the non-convex training problem. This reformulation allows for exploiting fast Lasso solvers based on the proximal gradient method and Least Angle Regression (LARS) (12).

Whereas in the training problem (1.1), the quality of the neural net fit to the data is measured by the l_2 loss as $\frac{1}{2}||f_L(\mathbf{X};\theta)-\mathbf{y}||_2^2$, our results generalize to a wide class of convex loss functions $\mathcal{L}_{\mathbf{y}}:\mathbb{R}^N\to\mathbb{R}$. With a general loss function, (1.1) becomes $\min_{\theta\in\Theta}\mathcal{L}_{\mathbf{y}}(f_L(\mathbf{X};\theta))+\frac{\beta}{\tilde{L}}\|\theta_w\|^{\tilde{L}}$. This is shown to be equivalent to the generalization of (1.2), namely $\min_{\mathbf{z},\xi}\mathcal{L}_{\mathbf{y}}(\mathbf{Az}+\xi\mathbf{1}-\mathbf{y})+\tilde{\beta}\|\mathbf{z}\|_1$.

The Lasso problem selects solutions $\mathbf{z}$ that generalize well by penalizing their total weight in l_1 norm (37). The l_1 norm typically selects a small number of elements in $\mathbf{z}$ to be nonzero. The Lasso equivalence demonstrates that neural networks can learn a sparse representation of the data by selecting certain features to fit $\mathbf{y}$.

The Lasso representation also elucidates the solution path of neural networks. The *solution path* for the Lasso or training problem is the map from $\beta\in(0,\infty)$ to the solution set. The Lasso solution path is well understood (37; 38; 12), providing insight into the solution path of the ReLU training problem (29).

This paper is organized as follows. Section 2 defines various neural network architectures. Section 3 describes our main theoretical result: neural networks are solutions to Lasso problems. One important consequence is that deep networks with ReLU and absolute value activations learn geometric reflections in the data. The next sections describe applications of the Lasso equivalence. Section 4 uses our theory to find explicit optimal neural networks for examples of binary data, and Section 5 uses the Lasso formulation to improve training of neural networks that predict financial time-series. Appendix F examines the relationship between the entire set of optimal neural nets given by the training problem versus the Lasso problem, while Appendix G applies the Lasso model to examine neural net behavior under minimum regularization, finding closed-form solutions. Appendix H presents experiments that support our theoretical results, and shows examples where neural networks trained with Adam naturally exhibit Lasso features.

1.1. Contributions.

Our contributions can be summarized as follows:

- Training various neural network architectures on 1-D data is equivalent to solving Lasso problems with finite, explicit and fixed dictionaries of basis signals that grow richer with depth (Theorems 3.12, 3.7 and 3.17). We identify dictionaries for various architectures in closed form (Lemma B.2).
- Features with reflections of training data appear in libraries for simple 3-layer and deeper architectures with ReLU (Theorem 3.5) or absolute value activation (Theorem 3.2). Experimentally, training these networks using the Adam optimizer leads to the same reflection features that we prove and matches our theoretical results on the global optima (Appendix H). In contrast, no reflection features are generated for the sign activation for any depth.
- Although the sign activation does not produce reflection features, its features become

richer for depth 3 networks compared to depth 2. Accordingly, for certain binary classification tasks, we analytically observe that optimal 3-layer sign activation networks generalize better than their 2-layer counterparts in the sense that their predictions are more uniform. Moreover, the Lasso problem yields closed-form expressions for these neural networks (Corollaries D.2 and D.4)

- After depth 3, the libraries freeze for networks with ReLU activation that have the same number of biased neurons in the middle and final layers (Lemma 3.8), and for networks with sign activation that have a constant number of neurons per layer (Theorem 3.17). But, the libraries grow when the width expands to twice as many neurons in the middle layer as the final layer for networks with ReLU activation (Theorem 3.12), and a tree structure for networks with sign activation (Theorem 3.17).
- A similar Lasso equivalence extends to neural networks trained on 2-D data in the upper half plane (Theorem C.3).

1.2. Notation.

Assume 1-D training data is ordered as $x_1 > x_2 > \cdots > x_N$. The indicator function of a logical statement z is $\mathbf{1}\{z\}$. Let $[n] = \{1, 2, \ldots, n\}$. For a matrix $\mathbf{Z}$, let $\mathbf{Z}_S$ be the submatrix of $\mathbf{Z}$ corresponding to indices in S. The number of nonzero elements in a vector $\mathbf{z}$ is $||\mathbf{z}||_0$. Let $\mathbf{1}, \mathbf{0} \in \mathbb{R}^N$ be the all-ones and all-zeros vectors, respectively.

A network that uses ReLU activation is a "ReLU network" or "ReLU-activated network," and a feature in a Lasso problem that is equivalent to a ReLU network is a "ReLU feature." Similar terminology holds for other activations and architectures.

2. Neural net architectures.

This section is devoted to defining neural net terminology and notation to be used throughout the rest of the paper. Let $L \geq 2$ be the depth of a neural network (which has $L-1$ hidden layers). We assume the activation $\sigma : \mathbb{R} \to \mathbb{R}$ is *piecewise linear around* 0 , i.e., of the form

$$\sigma(x) = \begin{cases} a^- x + b^- & \text{if } x < 0 \\ a^+ x + b^+ & \text{else} \end{cases}$$

for some $a^-, a^+, b^-, b^+ \in \mathbb{R}$. As shorthand, "piecewise linear" will mean "piecewise linear around 0." We focus on the piecewise linear activations of ReLU, leaky ReLU, absolute value, sign, and threshold functions. The ReLU activation is $\sigma(x) = (x)_+ := \max\{x, 0\}$, and absolute value activation is $\sigma(x) = |x|$. The leaky ReLU generalizes ReLU and absolute value as $\sigma(x) = (a^+\mathbf{1}\{x > 0\} + a^-\mathbf{1}\{x < 0\})x$ where $a^+ \neq a^-$. ReLU, leaky ReLU and absolute value activations will be referred to as "continuous piecewise linear." The threshold activation is $\sigma(x) = \mathbf{1}\{x \geq 0\}$, and the sign activation is $\sigma(x) = \text{sign}(x)$, where $\text{sign}(x)$ is -1 if $x < 0$, and 1 if $x \geq 0$. Note $\text{sign}(0) = 1$.

For $\mathbf{Z} \in \mathbb{R}^{n \times m}, \mathbf{s} \in \mathbb{R}^m$, let $\sigma_{\mathbf{s}}(\mathbf{Z}) = \sigma(\mathbf{Z})\text{Diag}(\mathbf{s})$. When the columns of $\sigma(\mathbf{Z})$ are neuron outputs, the i^{th} column of $\sigma_{\mathbf{s}}(\mathbf{Z}) \in \mathbb{R}^{N \times m}$ represents a neuron scaled by an *amplitude parameter* $s_i \in \mathbb{R}$. Amplitude parameters are (trainable) parameters for only the sign and threshold activations, and are to be ignored by interpreting them as 1 for ReLU, leaky ReLU, and absolute value activations.

Next, we define some neural net architectures. The parameter set is partitioned into $\theta = \theta_w \cup \theta_b \cup \{\xi\}$, where θ_b is a set of *internal bias* terms, and ξ is an *external* bias term. We define the elements of each parameter set below. We define neural nets by their output on row vectors $\mathbf{x} \in \mathbb{R}^{1\times d}$. Their outputs then extend to matrix inputs $\mathbf{X} \in \mathbb{R}^{N\times d}$ row-wise.

2.1. Standard networks. The following is a commonly studied neural net architecture. Let $L \geq 2$, the number of layers. Let $m_0 = d, m_{L-1} = 1$ and $m_l \in \mathbb{N}$ for $l \in [L-2]$, which are the number of neurons in each layer. For $l \in [L-1]$, let $\mathbf{W}^{(l)} \in \mathbb{R}^{m_{l-1}\times m_l}, \mathbf{s}^{(l)} \in \mathbb{R}^{m_l}, \mathbf{b}^{(l)} \in \mathbb{R}^{1\times m_l}, \xi \in \mathbb{R}$, which denote weights, amplitude parameters, internal biases, and external bias, respectively. Let $\mathbf{X}^{(1)} = \mathbf{x} \in \mathbb{R}^{1\times d}$ be the input to the neural net and $\mathbf{X}^{(l+1)} \in \mathbb{R}^{1\times m_l}$ be viewed as the inputs to layer $l+1$, defined by

$$\mathbf{X}^{(l+1)} = \sigma_{\mathbf{s}^{(l)}}\left(\mathbf{X}^{(l)}\mathbf{W}^{(l)} + \mathbf{b}^{(l)}\right). \tag{2.1}$$

Let $\boldsymbol{\alpha} \in \mathbb{R}^{m_L}$, which is the vector of final layer coefficients. A *standard neural network* is $f_L(\mathbf{x};\theta) = \xi + \mathbf{X}^{(L)}\boldsymbol{\alpha}$. The regularized and bias parameter sets are $\theta_w = \left\{\boldsymbol{\alpha}, \mathbf{W}^{(l)}, \mathbf{s}^{(l)} : l \in [L-1]\right\}$ and $\theta_b = \left\{\mathbf{b}^{(l)} : l \in [L-1]\right\}$, respectively.

The ultimate goal is analyzing the training problem for standard networks, but this appears to be challenging. However, by changing the architecture to a parallel structure defined next, we show that the training problem simplifies to the Lasso problem. These alternative architectures allow neural nets to be reconstructed more tractably from a Lasso solution than with a standard network. In the parallel architecture, m_L is the number of neurons in the final layer and for $i \in [m_L]$, we define the disjoint unions $\theta_w = \bigcup_{i\in[m_L]} \theta_w^{(i)}$ and $\theta_b = \bigcup_{i\in[m_L]} \theta_b^{(i)}$.

2.2. Parallel networks. A parallel network is a linear combination of standard networks in parallel, as we now define. Each standard network is called a *parallel unit*. Let $L \geq 2, m_0 = d, m_{L-1} = 1$ and $m_l \in \mathbb{N}$ for $l \in [L] - \{L-1\}$. For $i \in [m_L], l \in [L-1]$, let $\mathbf{W}^{(i,l)} \in \mathbb{R}^{m_{l-1}\times m_l}$, $\mathbf{s}^{(i,l)} \in \mathbb{R}^{m_l}$, $\mathbf{b}^{(i,l)} \in \mathbb{R}^{1\times m_l}, \xi \in \mathbb{R}$, which are the weights, amplitude parameters, and biases of the i^{th} parallel unit. Let $\hat{\mathbf{X}}^{(i,1)} = \mathbf{x} \in \mathbb{R}^{1\times d}$ be the input to the neural net and $\hat{\mathbf{X}}^{(i,l+1)} \in \mathbb{R}^{1\times m_l}$ be viewed as the input to layer $l+1$ in unit i, defined by

$$\hat{\mathbf{X}}^{(i,l+1)} = \sigma_{\mathbf{s}^{(i,l)}}\left(\hat{\mathbf{X}}^{(i,l)}\mathbf{W}^{(i,l)} + \mathbf{b}^{(i,l)}\right). \tag{2.2}$$

Let $\boldsymbol{\alpha} \in \mathbb{R}^{m_L}$. A *parallel neural network* is $f_L(\mathbf{x};\theta) = \xi + \sum_{i=1}^{m_L} \hat{\mathbf{X}}^{(i,L)}\alpha_i$. The regularized and bias parameter sets are $\theta_w^{(i)} = \left\{\alpha_i, \mathbf{s}^{(i,l)}, \mathbf{W}^{(i,l)} : l \in [L-1]\right\}, \theta_b^{(i)} = \left\{\mathbf{b}^{(i,l)} : l \in [L-1]\right\}$, for $i \in [m_L]$. We view parallel units as functions $\hat{\mathbf{X}}^{(i,L)} : \mathbb{R}^{1\times d} \to \mathbb{R}$ and with abuse of notation write $\hat{\mathbf{X}}^{(i,L)}(\mathbf{x}) \in \mathbb{R}$ as the output of $\hat{\mathbf{X}}^{(i,L)}$ evaluated on a sample $\mathbf{x}$. For a training dataset $\mathbf{X} \in \mathbb{R}^{N\times d}$, we denote the evaluations of the functions $\hat{\mathbf{X}}^{(i,L)}$ on the training data as $\hat{\mathbf{X}}^{(i,L)}(\mathbf{X})$. This paper primarily focuses on parallel architectures. However, a parallel network can be converted into a standard network (Remark A.1).

3. Main results. In this section, we show that non-convex deep neural net training problems are *equivalent* to Lasso problems, that is, their optimal values are the same, and given a Lasso solution, we can reconstruct a neural net that is optimal in the training problem. We say that a neural network *model* is equivalent to a Lasso model to mean that the optimal models are convertible to each other. We note that the optimal solutions may not be unique. Discussion of the relation between the solution sets of the Lasso and non-convex training problems with respect to a specified straightforward reconstruction is given in Appendix F. Further analysis of all possible reconstruction maps between the models is an area of future work. Analysis of the span or uniqueness and the generalizing abilities of different optimal or stationary solutions to the training problem is also an area for future work.

Unless otherwise stated, for the rest of this paper assume the data is 1-D. Define $\tilde{\beta} = \frac{\beta}{2}$ in (1.1),(1.2) for 3-layer symmetrized networks (defined below). For all other networks, let $\tilde{\beta} = \beta$.

3.1. 2-layer networks with piecewise linear activations and deep networks with continuous piecewise linear activations. A *deep narrow network* is a parallel network where the number of neurons in each parallel path is equal to 1, i.e., $m_1 = \cdots = m_{L-1} = 1$. In other words, a deep narrow network has m_L parallel units, each of which has one neuron in every layer, i.e., m_L neurons across units in each layer, adding up to Lm_L total neurons. For a 2-layer network, the parallel and deep narrow networks are the same as the standard network. A *symmetrized 3-layer network* is a parallel network with continuous piecewise linear activation where $m_1=2$ (which means each i^{th} unit out of m_3 parallel units has 2-D weight vectors $\mathbf{W}^{(i,1)}$ and $\mathbf{W}^{(i,2)}$ for the first and second layer, respectively) and the parameter space Θ enforces the constraint $\left|\mathbf{W}_1^{(i,l)}\right| = \left|\mathbf{W}_2^{(i,l)}\right|$ for $l \in [2], i \in [m_3]$. Therefore a symmetrized 3-layer network has $2m_3$ neurons in the "middle" layer. A symmetrized network extends the expressibility of a deep narrow network network by expanding its width. For other architectures, see Subsection 3.2.

In this section, we focus on the architectures discussed above to derive explicit, simple features that provide some of the first steps towards intuitively understanding the representation power of neural networks. We reformulate the training problem for 2-layer networks and for deeper depths with absolute value, ReLU and leaky ReLU activations into a convex Lasso problem. In general, our convexification approach and proofs provide a framework to analyze neural networks with more arbitrary widths. Additional results are deferred to Appendix B due to space, and proofs are deferred to Appendix H.3.

We first discuss networks with absolute value activation, as the symmetry of $|x|$ significantly simplifies the features. However, absolute value activation models are equivalent to ReLU activation models, as long as a skip connection is present. A 2-layer network with a *skip connection* is $f_L^{\text{skip}}(x;\theta) = f_L(x;\theta) + \omega x$ if $x \in \mathbb{R}$ (or more generally, $f_L^{\text{skip}}(\mathbf{x};\theta) = f_L(\mathbf{x};\theta) + \mathbf{x}\omega$) where $\omega \in \mathbb{R}^d$ is a trainable parameter in θ.

Lemma 3.1. *The training problem for a 2-layer network with skip connection and ReLU activation remains equivalent if the activation is changed to absolute value, and there is a map between the solutions for either activation.*

By Lemma 3.1, the absolute value activation is of interest to analyze as it can map to ReLU when skip connections are incorporated. Next we define some terms used to state our results.

For a piecewise linear function $f : \mathbb{R} \to \mathbb{R}$, x *is a breakpoint* of f (alternatively, f *has a breakpoint at* x) if f changes slope or is discontinuous at x. A breakpoint is a "kink" in the graph of f. For $a, b, c \in \mathbb{R}$, the *reflection of* a *about* b is $R_{(a,b)} = 2b - a$. A *double reflection* is a reflection of a reflection, i.e., is of the form $R_{\left(R_{(a,c)},b\right)}$ or $R_{\left(b,R_{(a,c)}\right)}$. The *generalized reflection*

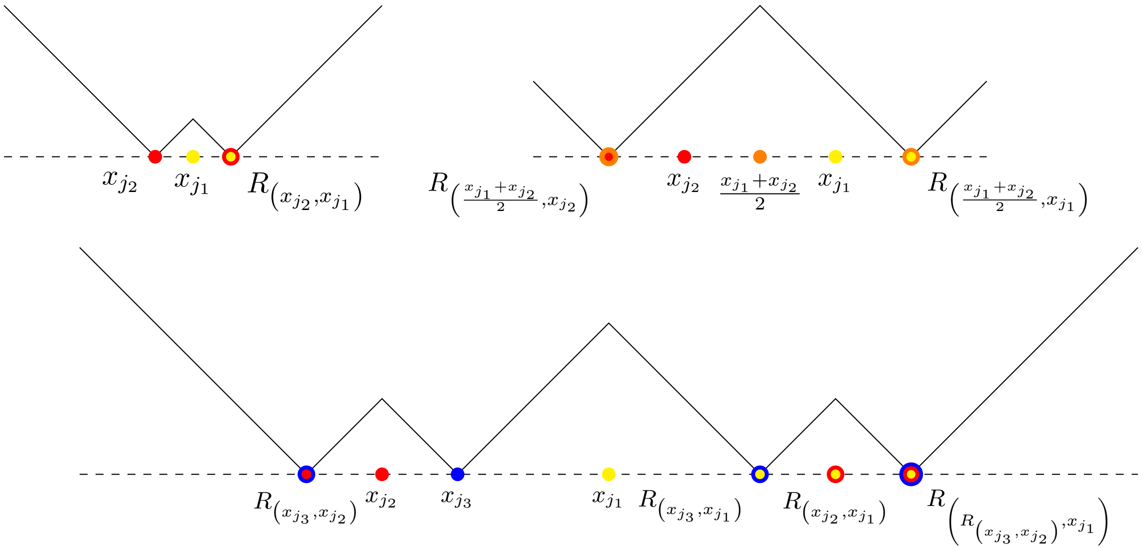


Figure 3.1: Example features, not including reversed directions, for deep narrow networks with absolute value activation. Top row: 3-layer features. The top left feature contains a breakpoint at the reflection of x_{j_2} (red) across x_{j_1} (yellow), which is denoted as $R_{(x_{j_2},x_{j_1})}$ (red encircling yellow). Other breakpoints are colored similarly. Bottom row: an example of a 4-layer feature, which contains a double reflection of x_{j_3} (blue) reflected across x_{j_2} (red), then reflected across x_{j_1} (yellow), which is denoted as $R_{\left(R_{(x_{j_3},x_{j_2})},x_{j_1}\right)}$ (blue encircling red, encircling yellow). All lines have slopes ± 1, and $x_{j_1}, x_{j_2}, x_{j_3}$ are training data.

of a and c about b is $a+c-b = R_{\left(b,\frac{a+c}{2}\right)}$, the reflection of b about the average of a and c. When $a = c$, the generalized reflection of a and c about b is the reflection of a about b. A breakpoint that is of the form $R_{(x_{j_1},x_{j_2})}$ for training data x_{j_1}, x_{j_2} is called a *reflection breakpoint* and a feature with a reflection breakpoint is called a *reflection feature*; and similarly for double reflections. Networks with absolute value activation can be modeled as Lasso problems with reflection features, as stated next.

Theorem 3.2 (Lasso equivalent of deep absolute value networks). *A deep narrow network of arbitrary depth with $\sigma(x) = |x|$ is equivalent*[1] *to a Lasso model with a finite set of features. Its dictionary matrix for 2 layers is $\mathbf{A}_{i,j} = |x_i - x_j|$. For 3 and 4 layers, its library includes features whose i^{th} element is $\big||x_i - x_{j_1}| - |x_{j_2} - x_{j_1}|\big|$ and $\Big|\big||x_i - x_{j_1}| - |x_{j_2} - x_{j_1}|\big| - \big||x_{j_3} - x_{j_1}| - |x_{j_2} - x_{j_1}|\big|\Big|$, respectively, over all training samples $x_i, x_{j_1}, x_{j_2}, x_{j_3}$. A similar pattern holds for deeper networks. The 2-layer features have breakpoints exactly at training data. The libraries for 3 and 4 layers additionally include reflection and double reflection features, respectively.*

Theorem 3.2 implies that an absolute value network learns to model data with a discrete and fixed dictionary of features. Figure 3.1 plots these features for L=3 and L=4. It illustrates

[1] See Section 3 for the definition of model equivalence.

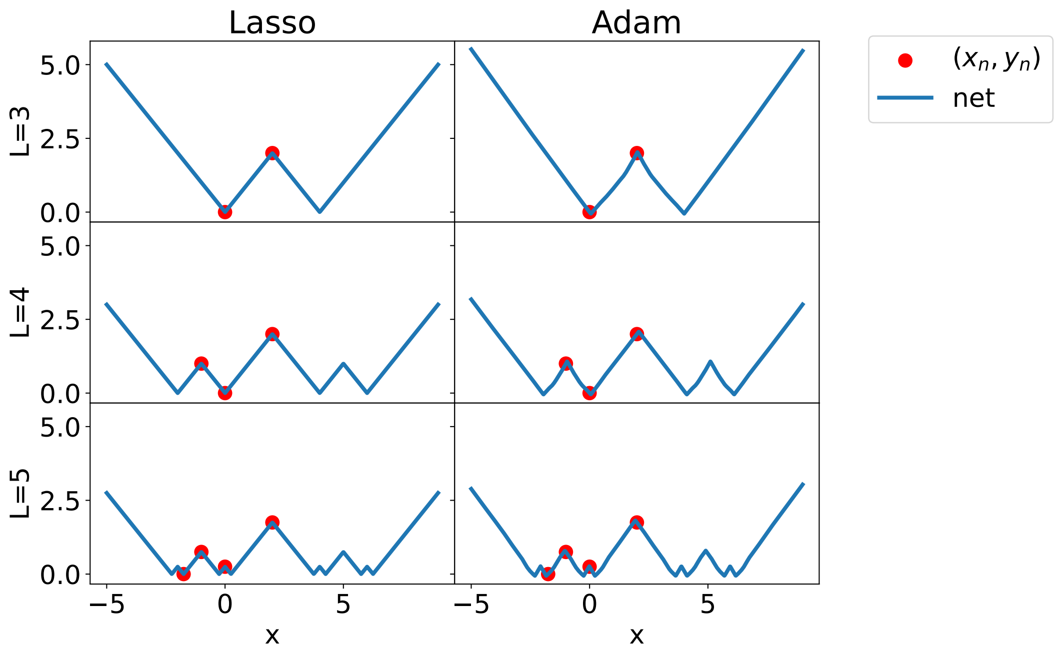


Figure 3.2: Lasso and Adam-trained deep narrow networks with absolute value activation. For L=3, the breakpoint at 2 is not a training point; it is the reflection of x_2=0 across x_1=1. For L=4, the breakpoint at 6 is not a training point; it is x_2=0 reflected about x_3=−1 to −2 (which not a training point) and then reflected across x_1=2. Similarly the 5-layer network contains more complex reflections.

that the breakpoints of the L=3 features occur at training data reflections, averages, and reflections about averages. The bottom row plots a possible feature for 4 layers, which shows breakpoints at reflections and double reflections. Even with just one neuron per layer (per path for parallel networks), adding a third layer in neural networks with absolute value activation adds features to the dictionary, and adds new locations of breakpoint to those features, namely at reflections $R_{(x_i,x_j)}$ of data points about themselves. Adding a fourth layer creates double reflections. In Figure 3.2, standard 3, 4, and 5-layer networks are trained with Adam. We make β=10^{-7} close to zero, and also solve the Lasso problem as $\beta \to 0$ (Appendix G). The Adam-trained networks closely match the Lasso solutions. Moreover, the 5-layer networks suggest that features continue to gain more complex reflections with depth. Simulations details are given in Appendix H.

Theorem 3.2 describe a subset of the library for $L \geq 3$ layers. The full library for 3 layers is defined in Theorem 3.12 and the features are explicitly described in Lemma B.2. Our approach lays a foundation to enumerate the full library for $L \geq 4$ layers as an area of future work.

In contrast, as will be shown in Theorem 3.17, the sign activation dictionary has no reflection features at any depth, which may limit its expressibility (28). A similar argument applies to the threshold activation. Reflection features allow neural networks to fit functions with breakpoints

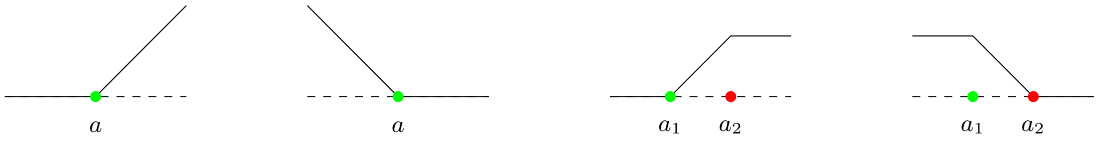


Figure 3.3: Examples of capped ramp functions in Definition 3.3. If $a, a_1, a_2 \in \{x_1, \cdots, x_N\}$, these functions are Lasso features for 3-layer deep narrow networks with ReLU activation.

at locations in between data points. The reflection breakpoints for absolute value networks suggest that they can learn geometric structures or symmetries from the data. Moreover, as the depth increases, this dictionary expands, which deepens its representation power.

With intuition from absolute value networks, we next discuss ReLU networks. Since reflection features appear in 3-layer networks with absolute value activation and $|x| = \frac{(x)_+ + (-x)_+}{2}$, we might expect reflections to also appear in 3-layer ReLU networks with twice as many neurons in the middle layer as the absolute value network. This is indeed the case, as shown below. First, we define parameterized families of functions from $\mathbb{R}$ to $\mathbb{R}$ that will be used to describe features for ReLU networks.

Definition 3.3. *Let $a_1 \in [-\infty, \infty), a_2 \in (-\infty, \infty]$. The* capped ramp *functions are*

$$Ramp^+_{a_1,a_2}(x) = \begin{cases} 0 & \text{if } x \le a_1 \\ x - a_1 & \text{if } a_1 \le x \le a_2 \\ a_2 - a_1 & \text{if } x \ge a_2 \end{cases}, \quad Ramp^-_{a_1,a_2}(x) = \begin{cases} a_2 - a_1 & \text{if } x \le a_1 \\ a_2 - x & \text{if } a_1 \le x \le a_2 \\ 0 & \text{if } x \ge a_2 \end{cases}$$

provided that $a_1 \le a_2$, and otherwise $Ramp^+_{a_1,a_2} = Ramp^-_{a_1,a_2} = 0$. In particular, the ramp *functions are $ReLU^+_a(x) = Ramp^+_{a,\infty} = (x-a)_+$ and $ReLU^-_a(x) = Ramp^-_{-\infty,a} = (a-x)_+$.*

In Definition 3.3, the parameters a, a_1, a_2 are the breakpoints of ramp and capped ramp functions. This is illustrated in Figure 3.3. Using these features, we state our results on Lasso equivalence for ReLU networks.

Theorem 3.4 (deep narrow ReLU network representation capability stagnates). *A deep narrow network of arbitrary depth with ReLU activation is equivalent to a Lasso model with a finite set of features. Its library contains only ramps and capped ramps, and beyond 3 layers, ReLU features do not change as the network deepens.*

In contrast to absolute value activation, for deep narrow networks, the ReLU library never gains reflection features. However, a symmetrized ReLU architecture creates reflections.

Theorem 3.5 (wider ReLU networks do not stagnate and generate reflections). *A three-layer symmetrized network with ReLU activation is equivalent to a Lasso model with a finite set of features, including those with breakpoints at reflections.*

Figure 3.5 plots ReLU features. A 2-layer ReLU network learns ramp features. Extending

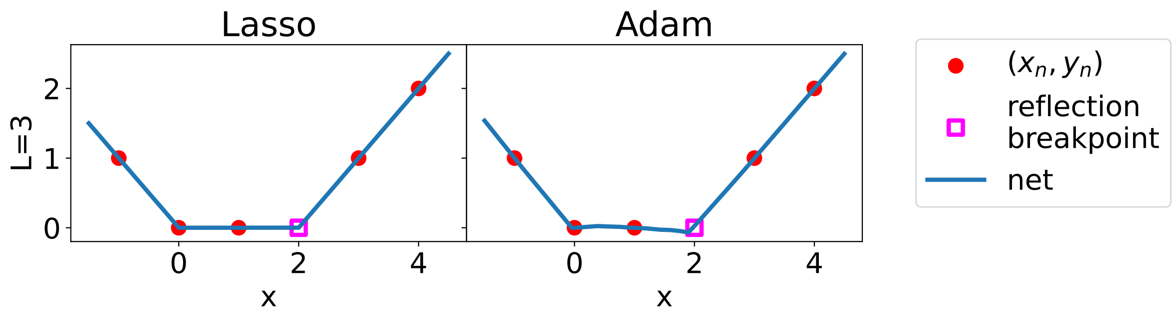


Figure 3.4: Lasso and Adam-trained 3-layer symmetrized ReLU networks. Most crucially, the breakpoint at 2 is not a training point; it is the reflection of $x_4 = 0$ across $x_3 = 1$.

the depth to one more layer without changing the width creates a deep narrow 3-layer network, which additionally learns capped ramp features. Extending both the depth and width enables creating a 3-layer symmetrized network, which learns an additional host of features, including generalized reflection features shown in the bottom row of the figure. The generalized reflections $x_j + x_k - x_i$ are reflections when $j = k$. Reflection features appear in Adam-trained ReLU networks as predicted by the Lasso formulation, as shown in Figure 3.4. Details of this simulation are given in Appendix H. Next, we formally define a feature function and its properties.

Remark 3.6. *Theorem 3.12 and Definition B.3 below show that the training problem is equivalent to a Lasso problem with solution* $(\mathbf{z}^*, \xi^*)$ *and a dictionary matrix whose* i^{th} *column* $\mathbf{A}_i$ *maps to a parallel unit function* $\hat{\mathbf{X}}^{(i,L)} : \mathbb{R} \to \mathbb{R}$ *such that* $\hat{\mathbf{X}}^{(i,L)}(\mathbf{X}) = \mathbf{A}_i$ *and* $\sum_i z_i^* \hat{\mathbf{X}}^{(i,L)}(x) + \xi^*$ *is the same function as an optimal neural net. In this section we define a* feature *as described in Section 1 to be a parallel unit function* $\hat{\mathbf{X}}^{(i,L)}$ *corresponding to* $\mathbf{A}_i$ *such that* $z_i^* \neq 0$.

In addition to ReLU and absolute value networks, leaky ReLU networks are also equivalent to Lasso models. Moreover, we can upper bound the size of the libraries.

Theorem 3.7. *A deep narrow network of any depth* $L \geq 2$ *and piecewise linear activation is equivalent to a Lasso problem with a finite set of features. The number of features is* $O(N^2)$ *for ReLU activation and* $O(N^{L-1} 2^L L!)$ *for leaky ReLU and absolute value activations.*

In particular, the number of features is finite and at most polynomial in the number of training points and exponential in the depth. However, the number of features is an overestimate and in fact saturates for certain activations and architectures, as seen in the next result.

Lemma 3.8. *Training a deep narrow ReLU network with an arbitrary number of layers* $(L \geq 2)$ *is a Lasso problem where features are ReLU or capped ramp functions with breakpoints at data points. The number of features is* $O(N^2)$.

Adding only one neuron per layer limits the expressibility of ReLU networks. In contrast to absolute value, ReLU networks rely more heavily on wider layers for expressibility.

Definition 3.9. *For $\alpha, \beta, a^+, a^- \in \mathbb{R}$ such that $a^+ \neq a^-$, let the* normalized midpoint *be*

$$m_{\alpha,\beta} = -\frac{a^+ \max\{\alpha,\beta\} - a^- \min\{\alpha,\beta\}}{a^+ - a^-}. \tag{3.1}$$

Note $m_{\alpha,\beta}$ is the midpoint $\frac{\alpha+\beta}{2}$ between α and β if $\sigma(x)=|x|$. If $\sigma(x)=(x)_+$, or if $\sigma(x)=|x|$ and $\alpha=\beta$, then $m_{\alpha,\beta}=\alpha$. We use the normalized midpoint to define bias parameters for features. Recall that $\hat{\mathbf{X}}^{(1,L)}$ denotes a parallel unit, which is a standard network (Subsection 2.2). Let $\hat{\mathbf{X}}^{(l)}=\hat{\mathbf{X}}^{(1,l)}$, $\mathbf{W}^{(l)}=\mathbf{W}^{(1,l)}$, and $\mathbf{b}^{(l)}=\mathbf{b}^{(1,l)}$. The output $\hat{\mathbf{X}}^{(l)}$ of each layer can be interpreted as a feature extracted by that layer. This motivates the following definition.

Definition 3.10. *A bias parameter $\mathbf{b}^{(l)}$ is a* data feature bias *if $\mathbf{b}^{(l)}=-\hat{\mathbf{X}}^{(l)}(\mathbf{x}_0)\mathbf{W}^{(l)}$ for some column vector $\mathbf{x}_0 \in \{x_1, \cdots, x_N\}^{m_l}$. The bias parameter $\mathbf{b}^{(1)}$ is a first-layer* midpoint feature bias *if $\mathbf{b}^{(1)} = -m_{x_{j_1}, x_{j_1'}}\mathbf{W}^{(1)}$ for some $j_1, j_1' \in [N]$ and it is in a 3-layer deep narrow network with a non-monotone, continuous piecewise linear activation (such as $\sigma(x) = |x|$.)*

In the simplest case when $L=2$, a data feature bias is of the form $b = -wx_n$. These bias parameters are used to define a deep library.

Definition 3.11 (Deep Library). *Consider a 3-layer symmetrized or L-layer deep narrow network. The* deep library *is the set of all network outputs $\hat{\mathbf{X}}^{(L)}(\mathbf{X})$ defined in (2.2) with data or midpoint feature biases and all elements of $\mathbf{W}^{(1)}$ being ± 1.*

The deep library contains a finite number of standard networks evaluated on the training data. The next result shows that a neural network learns features in the deep library.

Theorem 3.12 (complete Lasso libraries for general activations). *Consider a deep narrow L-layer network where $L \in \{2, 3\}$ and with activation σ which is ReLU, leaky ReLU, absolute value, sign, or threshold if $L=2$, and ReLU, leaky ReLU or absolute value if $L=3$. Consider a Lasso problem whose dictionary is the deep library and where $\xi=0$ if σ is sign or threshold. Suppose $(\mathbf{z}^*, \xi^*)$ is a solution, and let $m^*=\|\mathbf{z}^*\|_0$. This Lasso problem is equivalent to the training problem for the network, provided $m_L \geq m^*$.*

The notion of equivalence between optimization problems is defined in the beginning of Section 3. Definition B.3 defines a map to reconstruct an optimal neural network from a Lasso solution (Lemma B.4). The map is especially straightforward for 2-layer networks (Definition B.5). These results are given in Appendix B due to space. The next theorem generalizes Theorem 3.5 to leaky ReLU activations.

Theorem 3.13. *The training problem for a 3-layer symmetrized network with monotone activations such as ReLU is equivalent to a Lasso problem with solution $(\mathbf{z}^*, \xi^*)$ and whose dictionary contains the deep library, provided $m_L \geq m^*$, where $m^* = \|\mathbf{z}^*\|_0$.*

Theorem 3.13 states that the deep library is a sub-dictionary for symmetrized networks. Finding the full dictionary for a symmetrized network is an area of future work. Theorem 3.12 shows that instead of training a neural network with a non-convex problem and reaching a possibly local optimum, we can simply solve a straightforward Lasso problem whose convexity guarantees that gradient descent approaches global optimality. Figure H.3 shows an example where a network trained with Lasso achieves a better function fit than training with the non-convex problem. In previous work (13), a similar Lasso formulation is developed for networks

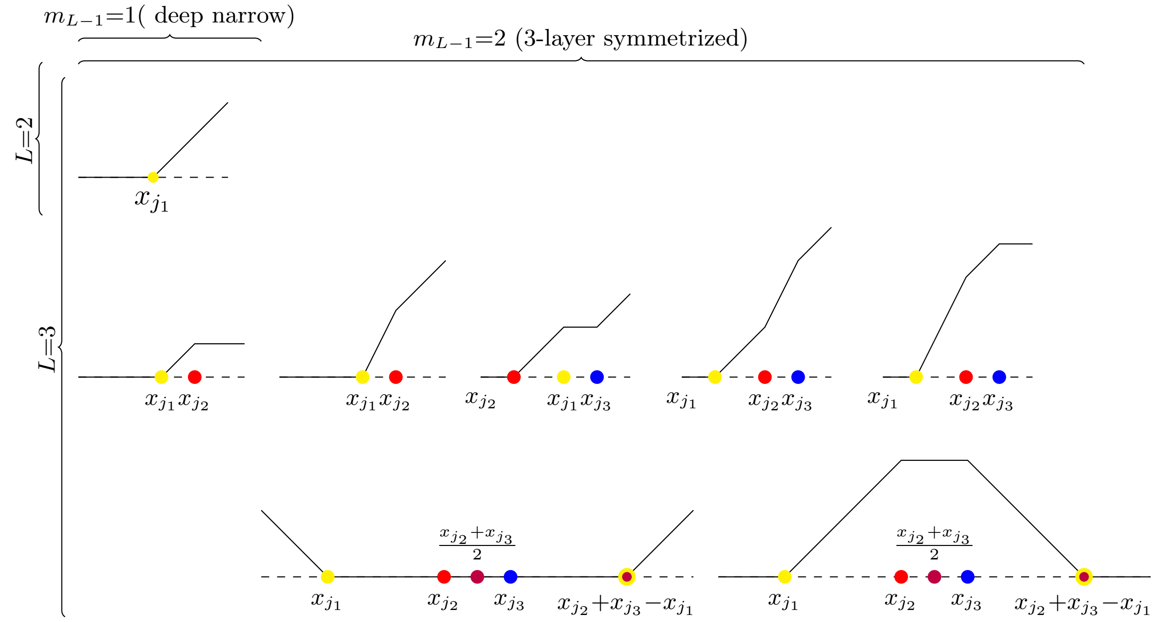


Figure 3.5: Example ReLU features, excluding reverse directions. Bottom row: 3-layer symmetrized ReLU features with breakpoints at generalized reflections of x_{j_1} (yellow) across x_{j_2} (red) and x_{j_3} (blue) $x_{j_2}+x_{j_3}-x_{j_1}$ (yellow encircling purple). Lines have slopes $\pm 2, \pm 1$ or 0.

with threshold activation but requires up to 2^N features of length N in the dictionary for a 2-layer network. In contrast, with 1-D data, Theorem 3.12 shows that at most $2N^2$ features are needed for a 2-layer network.

Remark 3.14. *Note that when the network is 2 layers, the equivalent Lasso dictionary only contains features with breakpoints at training data, leading to a prediction with breakpoints only at data locations. In contrast, when the network has 3 layers there can be breakpoints at* ***reflections*** *of data points with respect to other data points due to the reflection features. As a result, for activations such as absolute value and ReLU with symmetrized networks, the sequence of dictionaries as the network gets deeper converges to a richer library that includes reflections.*

Our approach lays the foundation to further analyze the evolution of feature libraries over expanding depth and widths as an area of future work. For 2-layer networks, the dictionary (Theorem 3.12) is simple, as described next.

Corollary 3.15 (2-layer libraries). *Let $\mathbf{A}_+, \mathbf{A}_- \in \mathbb{R}^{N\times N}$ with $(\mathbf{A}_+)_{i,n}=\sigma(x_i-x_n), (\mathbf{A}_-)_{i,n}=\sigma(x_n-x_i)$. We can write the dictionary matrix for 2-layer networks as $\mathbf{A}=\mathbf{A}_+$ for absolute value and sign activations, and $\mathbf{A}=[\mathbf{A}_+, \mathbf{A}_-] \in \mathbb{R}^{N\times 2N}$ for ReLU, leaky ReLU, and threshold activations.*

In Corollary 3.15, $\mathbf{A}_+$ and $\mathbf{A}_-$ contain features $\hat{\mathbf{X}}^{(2)}(\mathbf{X})$ where $\mathbf{W}^{(1)} = 1$ and $\mathbf{W}^{(1)} = -1$, respectively (Definition 3.11). Figure 3.6 illustrates $\mathbf{A}_+$ for the ReLU and sign activations. Using the notation of Definition 3.11, the 3-layer deep narrow absolute value features are

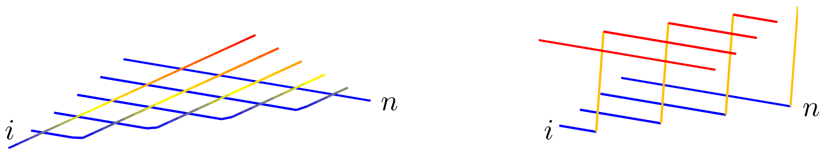


Figure 3.6: Generic shape of $\mathbf{A}_+ \in \mathbb{R}^{N\times N}$ defined by $\mathbf{A}_{+i,n} = \sigma(x_i - x_n)$, where σ is ReLU (left) and sign activation (right). Each i^{th} curve represents a feature. The points $\left(i, n, \mathbf{A}_{+i,n}\right)$ are plotted in 3-D, with $\mathbf{A}_{+i,n}$ represented by the curve height and color. Here, $n \in [N]$ but each curve interpolates between integer values of n.

of the form $\hat{\mathbf{X}}^{(3)}(x){=}\big||x{-}x_{j_1}|{-}|x_{j_2}{-}x_{j_1}|\big|$, and for 4 layers, include features of the form $\hat{\mathbf{X}}^{(4)}(x){=}\Big|\big||x{-}x_{j_1}|{-}|x_{j_2}{-}x_{j_1}|\big|{-}\big||x_{j_3}{-}x_{j_1}|{-}|x_{j_2}{-}x_{j_1}|\big|\Big|$. These features are plotted in Figure 3.1, which highlights their reflection and double reflection breakpoints. Figure 3.5 shows a subset of the ReLU library, with generalized reflection features for 3-layer symmetrized networks. Figure 3.5 illustrates general feature shapes not including mirrored directions. In Figure B.1, we choose distinct training samples $x_i, x_j, x_k{\in}\{-1, 0, 2\}$ and numerically compute all possible deep library features for 3-layer, symmetrized ReLU networks using Definition 3.11. Since the features $\hat{\mathbf{X}}^{(L)}(x)$ are continuous with respect to x_i, x_j, x_k, the features for non-distinct x_i, x_j, x_k can be extrapolated by merging adjacent training points. The numerically plotted features are consistent with Figure 3.5. In addition to graphical representations, Lemma B.2 in Appendix B gives examples of simple, explicit expressions for features defined in Definition 3.11.

So far, we have emphasized deep ReLU and absolute value networks as having reflection features. Next, we show that in contrast, deep networks with sign activation do not have reflection features, even if they have many neurons per layer, suggesting the importance of choice of activation. The results for sign activation can be similarly extended to threshold activation (13).

3.2. Deep neural networks with sign activation. In this section, we analyze the training problem of an L-layer deep network with sign activation, which need not be a deep narrow network. The formal statements of a few results are deferred to Appendix C due to space. Proofs in this section are deferred to Appendix H.4.

We say the vector $\mathbf{h}{\in}\{-1, 1\}^N$ *switches* at $n > 1$ if $h_n \neq h_{n-1}$. For $n \in \mathbb{N}$, let the *switching set* $\mathbf{H}^{(n)}$ be the set of all vectors in $\{-1, 1\}^N$ that start with 1 and switch at most n times. For a set of vectors $\mathcal{Z}$, let $[\mathcal{Z}]$ be a matrix whose set of columns is $\mathcal{Z}$. The next result relates the switching set to the Lasso dictionary matrix in Theorem 3.12.

Lemma 3.16. *The dictionary matrix for a 2-layer network with sign activation is* $\left[\mathbf{H}^{(1)}\right]$.

We will show in Theorem 3.17 that the training problem (1.1) for deeper networks with sign activation is also equivalent to a Lasso problem (1.2) whose dictionary is a switching set.

We now consider another architecture. While each unit of the parallel neural network is a standard network, every branch of a *tree network* is a parallel network. A detailed definition is given in Appendix C.1 due to space. Parallel and tree nets have the same architecture for $L = 3$ layers. For $L \geq 3$, a *rectangular network* is a parallel network (Subsection 2.2) with $m_1 = \cdots = m_{L-2}$. A deep narrow network is a special case of a rectangular network. Now

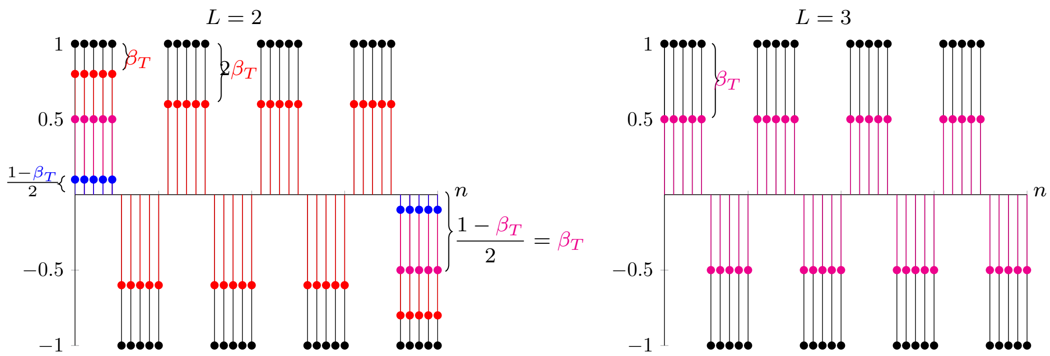


Figure 3.7: Each of the two figures depicts (n, y_n) with black dots, where $\mathbf{y} = \mathbf{h}^{(T)}$ with $T = 10, N = 40$. Sign-activated neural net predictions are depicted as $(n, f_L(x_n; \theta))$ with blue, magenta, and red dots for $\beta_T = \frac{4}{5} \in \left[\frac{1}{2}, 1\right], \beta_T = \frac{1}{2}$, and $\beta_T = \frac{1}{5} \leq \frac{1}{2}$, respectively.

we state the main result of this section: networks with sign activation are equivalent to Lasso models with libraries that increase until depth 3 and then freeze for rectangular networks but continue multiplying geometrically for tree networks.

Theorem 3.17. *Consider a Lasso problem whose dictionary is the switching set $\mathbf{H}^{(K)}$, $\xi = 0$, and with solution $\mathbf{z}^*$. Let $m^* = \|\mathbf{z}^*\|_0$. This Lasso problem is equivalent to the training problem for a neural network with sign activation, $m_L \geq m^*$, and $m_{L-2} = K$ when it is a rectangular network, and $\prod_{l=1}^{L-1} m_l = K$ when it is a tree network.*

Similar to Remark 3.6, we can formally define features for rectangular and tree networks as parallel units and subtrees, respectively.

Corollary 3.18. *The features defined and described in Remark 3.6 also apply to arbitrarily deep parallel networks with sign activation, and analogously for tree networks.*

Features for sign activation are step functions that take on the value $\mathbf{A}_{i,n}{=}\pm 1$ at x_n (where $\mathbf{A}_i$ is the corresponding column of the dictionary matrix) and remain at that value until the next data point x_{n+1}. They only switch value at data points, and do not have breakpoints at reflections. The green graph in Figure B.2 illustrates an example of a sign activation feature.

Theorem 3.17 generalizes Theorem 3.12 for sign networks. By Lemma 3.16, for $L{=}2$, $1{=}d{=}m_0{=}L{-}2$, so the sign activation dictionary is also $\mathbf{H}^{(m_{L-2})}{=}\mathbf{H}^{(1)}$. Adding a third layer to a parallel network with sign activation expands the library to encompass features with up to m_1 switches. But no new features enter the library if the depth increases beyond 3. This limited library suggests that the representation power of sign activation networks may stagnate after three layers, unless the architecture is changed. Indeed, for a tree architecture, the library continues to expand, and the features have as many breakpoints as the product of the number of neurons in each layer.

An explicit and efficient reconstruction of an optimal 3-layer neural net with sign activation is described in Lemma C.1 in Appendix C. It is drawn in Figure B.2. Reconstructions for other architectures are given in Appendix H.4. The Lasso dictionary for deep neural nets in previous

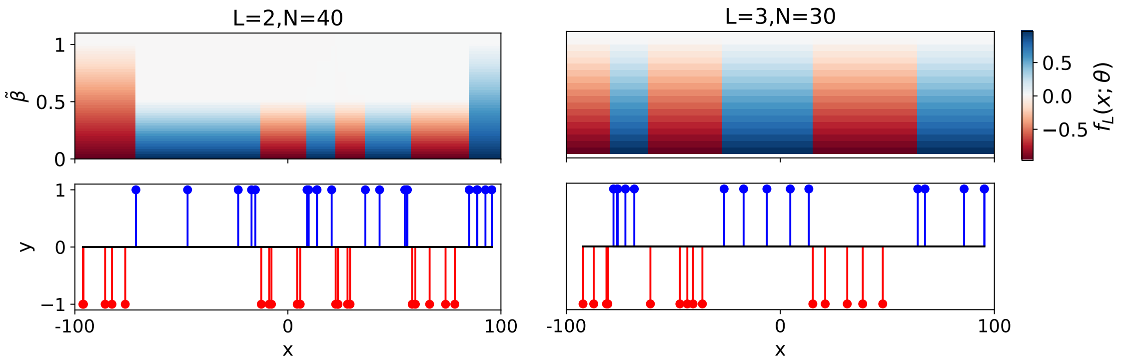


Figure 3.8: The bottom figures plot training data (x_n, y_n). The top figures plot sign-activated neural net predictions by color for each x, as parameterized by β on the vertical axis. With abuse of notation, in this figure (only), $\tilde{\beta} = \beta/T$.

work (13) uses a dictionary that depends on the training data. However, Lemma 3.16 and Theorem 3.17 show that networks with sign activation have dictionaries that are invariant to the training data. So to train multiple neural nets on different data, the dictionary matrix $\mathbf{A}$ only needs to be constructed once.

Our theory for 1-D data provides a lens to analyze higher dimensional data. Using a similar approach, Theorem C.3 in Appendix C gives an example of 2-D data for which a neural net can be recast as a Lasso model. Extending this to more general data and higher dimensions is an area for future work. The next remark summarizes the expansion of libraries over depth.

Remark 3.19. *The dictionary for an architecture discussed in Theorem 3.12, Theorem 3.17 or Theorem C.3 is a superset of any dictionary with the same architecture but shallower depth.*

This section analyzed the equivalence of neural networks with sign activation and Lasso models. While their features do not contain reflections, they are straightforward binary features that elucidate how depth increases the representation power of 2 versus 3-layer networks. For example, Corollary C.2 in Appendix C uses the Lasso problem to bound the training loss for 2-layer networks with sign activation between those of depth 3. The next section leverages the sign activation features to show that in a case of binary, periodic data, 3-layer networks have a solution path indicating better generalization properties than 2-layer networks.

4. Solution path for sign activation and binary, periodic labels.

This section examines solution paths of Lasso problems for 2 and 3-layer neural nets with sign activation and 1-D data where the target vector $\mathbf{y}$ is binary. Such data appears in temporal sequences such as binary encodings of messages communicated digitally (24), neuron firings in the brain (18), and other applications, where x_n represents time. These real-world sequences are in general aperiodic. However, in the special case that the target vector is periodic and binary, the Lasso problem gives tractable solutions for optimal neural networks. This offers a step towards analyzing neural network behavior for more general, aperiodic data, which is an area for future work. We call the binary, periodic sequence a square wave, defined as follows. For a positive even

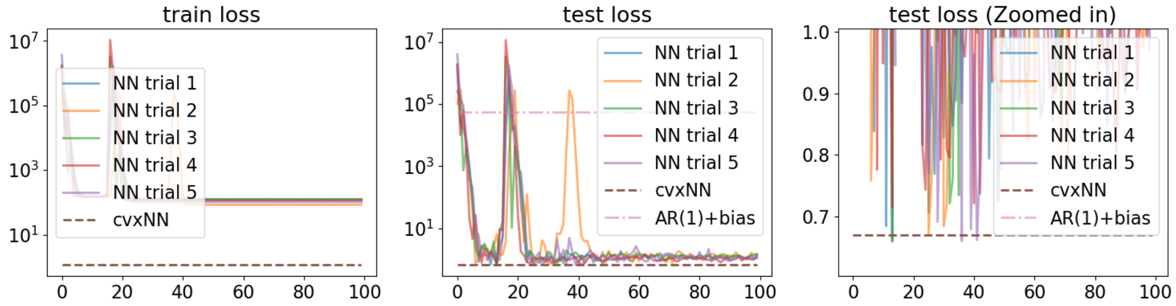


Figure 5.1: Comparison of neural autoregressive models of the form $x_t = f(x_{t-1};\theta) + \epsilon_t$ using convex and non-convex optimizers and the classical linear model AR(1) for time series forecasting. The horizontal axis is the training epoch. The dataset is BTC-2017min from Kaggle, which contains all 1-minute Bitcoin prices in 2017 (1). The non-linear models outperform the linear AR(1) model. Moreover, SGD underperforms in training and test loss compared to the convex model which is guaranteed to find a global optimum of the NN objective.

integer T that divides N, define a *square wave* to be $\mathbf{h}^{(T)} \in \{-1,1\}^N$ that starts with 1 and is periodic with period T. If the real line is split into a finite number of regions by binary labels, the square wave represents the labels of a monotone sequence of points, with the same number of samples in each region. The black dots in Figure 3.7 plot an example of a square wave.

Consider training a 2-layer neural net with sign activation when $\mathbf{y} = \mathbf{h}^T$. When $\beta > T$, an optimal neural net is the constant zero function. When $\beta \leq \frac{T}{2}$, we can find an optimal neural net $f_2(\mathbf{X};\theta)$ which is is periodic over $\left[\frac{T}{2}, N - \frac{T}{2}\right]$ with period T, and has amplitude $2\frac{\beta}{T}$ less than that of $\mathbf{y}$. Theorem D.1 gives the entire Lasso solution path and Corollary D.2 gives a closed form expression for the resulting optimal neural net. Theorem D.3 and Corollary D.4 give the solution path and an optimal neural net when $L = 3$. Only one parallel unit is active in this network, and so it is also a standard neural net. The neural net has output $f_3(x;\theta)(\mathbf{X}) = (1 - \beta_T)_+ \mathbf{y}$. If $\beta > N$, then the optimal neural net is the constant zero function. These results are in Appendix D due to space.

Figure 3.7 illustrates the solution path of $f_L(\mathbf{X};\theta)$ when $\mathbf{y} = \mathbf{h}^{(T)}$. It suggests that as the regularization increases, the 2-layer network focuses on preserving the boundary points of the data (first and last T points) to closely match the target vector. Therefore the network will generalize well if noise occurs in the middle of the data. In contrast, if noise occurs uniformly over the data, the 3-layer network will generalize well.

We verify our theoretical predictions for optimal neural nets in Theorem D.1 and Theorem D.3 by solving the Lasso problem on sample training data with target vector $\mathbf{h}^{(T)}$. Figure 3.8 illustrates the training data in the bottom plot and neural net predictions for each β in the rows of the top plot. The 2-layer network is biased towards predicting strongly in the first and last intervals, suggesting worse generalizability than the 3-layer network. In addition, the 3-layer net changes more uniformly with β than the 2-layer net, making it easier to tune β.

5. Application: Time-series modeling.

In this section, we apply the Lasso problem for neural networks to an autoregression problem. Suppose at times $1, \cdots, T+1$ we observe data points $x_1, \cdots, x_{T+1} \in \mathbb{R}$ that follow the time-series model

$$x_t = f(x_{t-1}; \theta) + \epsilon_t, \tag{5.1}$$

where $f : \mathbb{R} \to \mathbb{R}$ is parameterized by some parameter θ and $\epsilon_t \sim \mathcal{N}(0, \sigma^2)$ represents observation noise. The parameter θ is unknown, and the goal is find θ that best fits the model (5.1) to the data $x_1, \cdots, x_{T+1}$. For example, the *auto-regressive model with lag* 1 (AR(1)) is a linear model $f(x; \theta) = ax$ where $\theta = \{a\}$ is chosen as a solution to $\min_{\theta \in \Theta} \sum_{t=1}^{T} \left(f(x_t; \theta) - x_{t+1}\right)^2$. For a more expressive model, instead of $f(x; \theta) = ax$ suppose we use a 2-layer neural network

$$f_2^{\text{NN}}(x; \theta) = \sum_{i=1}^{m} |x w_i + b_i| \, \alpha_i, \tag{5.2}$$

which has m neurons and absolute value activation. The parameter set is $\theta = \{w_i, b_i, \alpha_i\}_{i=1}^m$.

Here, we show how to find a neural network model $f_2^{\text{NN}}(x_t; \theta)$ (5.2) that represents the τ-quantile of the distribution of x_{t+1} given the observation x_t, where $\tau \in [0, 1]$, by using the *quantile regression loss* $L_\tau(z) = 0.5|z| + (\tau - 0.5)z$ and choosing θ that solves the *neural net (NN) quantile regression (QR) training problem*

$$\min_{\theta \in \Theta} \frac{1}{T} \sum_{t=1}^{T} L_\tau\left(f_2^{\text{NN}}(x_t; \theta) - x_{t+1}\right) + \frac{\beta}{2} \|\theta_w\|_2^2. \tag{5.3}$$

Problem (5.3) can be solved by converting it to an equivalent Lasso problem. Figure 5.1 shows that using the Lasso model to predict Bitcoin price reaches a lower loss than training with the non-convex model. More details are found in Appendix E.

6. Conclusion.

Our results show that deep neural networks with a variety of activation functions trained on 1-D data with weight regularization can be recast as convex Lasso models with simple dictionary matrices. This provides critical insight into their solution path as the weight regularization changes. The Lasso problem also provides a fast way to train neural networks for 1-D data. Moreover, the understanding of the neural networks through Lasso models could also be used to explore designing better neural network architectures.

We proved that reflection features can emerge in the Lasso dictionary when the depth is 3 or deeper. This leads to predictions that have breakpoints at reflections of data points about other data points. In contrast, for networks of depth 2, the breakpoints are only located at a subset of training data. We believe that this mechanism enables deep neural networks to generalize to the unseen by encoding a geometric regularity prior.

Our analysis of various architectures provides a foundation for studying more complex network topologies. The 1-D results can extend to sufficiently structured or low rank data in higher dimensions. Generalizing to higher dimensions is also an area of future work. Building on a similar theme, (30) showed that the structure of hidden neurons can be expressed through convex optimization and Clifford's Geometric Algebra. The techniques developed in this paper can be combined with the Clifford Algebra to develop higher-dimensional analogues of the results.

References.

[1] *Kaggle*, https://www.kaggle.com/datasets/prasoonkottarathil/btcinusd.

[2] F. Bach, *Breaking the curse of dimensionality with convex neural networks*, JMLR, 18 (2017), pp. 629–681.

[3] Y. Bengio, N. Léonard, and A. C. Courville, *Estimating or propagating gradients through stochastic neurons for conditional computation*, ArXiv:1308.3432, (2013).

[4] Y. Bengio, N. Roux, P. Vincent, O. Delalleau, and P. Marcotte, *Convex neural networks*, in Advances in Neural Information Processing Systems, vol. 18, MIT Press, 2005.

[5] M. J. Bianco, P. Gerstoft, J. Traer, E. Ozanich, M. A. Roch, S. Gannot, and C.-A. Deledalle, *Machine learning in acoustics: Theory and applications*, The Journal of the Acoustical Society of America, 146 (2019), pp. 3590–3628.

[6] J. M. Borwein and A. S. Lewis, *Convex Analysis and Nonlinear Optimization: Theory and Examples*, Springer, 2000.

[7] S. Boyd and L. Vandenberghe, *Convex optimization*, Cambridge university press, 2004.

[8] A. Bulat and G. Tzimiropoulos, *XNOR-Net++: Improved binary neural networks*, ArXiv, abs/1909.13863 (2019).

[9] P. Chen and O. Ghattas, *Projected Stein variational gradient descent*, Advances in Neural Information Processing Systems, 33 (2020), pp. 1947–1958.

[10] P. Chen, K. Wu, J. Chen, T. O'Leary-Roseberry, and O. Ghattas, *Projected Stein variational newton: A fast and scalable Bayesian inference method in high dimensions*, Advances in Neural Information Processing Systems, 32 (2019).

[11] T. M. Cover, *Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition*, IEEE Transactions on Electronic Computers, (1965), pp. 326–334.

[12] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, *Least angle regression*, The Annals of statistics, 32 (2004), pp. 407–499.

[13] T. Ergen, H. I. Gulluk, J. Lacotte, and M. Pilanci, *Globally optimal training of neural networks with threshold activation functions*, arXiv:2303.03382, (2023).

[14] T. Ergen and M. Pilanci, *Convex geometry of two-layer ReLU networks: Implicit autoencoding and interpretable models*, PMLR, 26–28 Aug. 2020, pp. 4024–4033.

[15] T. Ergen and M. Pilanci, *Convex geometry and duality of over-parameterized neural networks*, The Journal of Machine Learning Research, 22 (2021), pp. 9646–9708.

[16] T. Ergen and M. Pilanci, *Revealing the structure of deep neural networks via convex duality*, in ICML, PMLR, 2021, pp. 3004–3014.

[17] C. Fang, Y. Gu, W. Zhang, and T. Zhang, *Convex formulation of overparameterized deep neural networks*, arXiv:1911.07626, (2019).

[18] H. Fang, Y. Wang, and J. He, *Spiking neural networks for cortical neuronal spike train decoding*, Neural Computation, 22 (2010), pp. 1060–1085.

[19] S. Feizi, H. Javadi, J. M. Zhang, and D. Tse, *Porcupine neural networks: (almost) all local optima are global*, ArXiv, abs/1710.02196 (2017).

[20] M. Freitag, S. Amiriparian, S. Pugachevskiy, N. Cummins, and B. Schuller, *audeep: Unsupervised learning of representations from audio with deep recurrent neural*

networks, JMLR, 18 (2017), pp. 6340–6344.

[21] C.-L. Hsu and J.-S. R. Jang, *On the improvement of singing voice separation for monaural recordings using the MIR-1K dataset*, IEEE Transactions on Audio, Speech, and Language Processing, 18 (2009), pp. 310–319.

[22] N. Joshi, G. Vardi, and N. Srebro, *Noisy interpolation learning with shallow univariate relu networks*, arXiv.2307.15396, (2023).

[23] K. Karhadkar, M. Murray, H. Tseran, and G. Montúfar, *Mildly overparameterized relu networks have a favorable loss landscape*, arXiv:2305.19510, (2023), https://arxiv.org/abs/2305.19510.

[24] H. Kim, Y. Jiang, R. Rana, S. Kannan, S. Oh, and P. Viswanath, *Communication algorithms via deep learning*, arXiv:1805.09317, (2018).

[25] M. Kim and P. Smaragdis, *Bitwise neural networks for efficient single-channel source separation*, in 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2018, pp. 701–705.

[26] G. Kornowski, G. Yehudai, and O. Shamir, *From tempered to benign overfitting in ReLU neural networks*, arXiv:2305.15141, (2023), https://arxiv.org/abs/2305.15141.

[27] S. Mavaddati, *A novel singing voice separation method based on a learnable decomposition technique*, Circuits, Systems, and Signal Processing, 39 (2020), pp. 3652–3681.

[28] M. Minsky and S. A. Papert, *Perceptrons: An Introduction to Computational Geometry*, The MIT Press, 09 2017, https://doi.org/10.7551/mitpress/11301.001.0001, https://doi.org/10.7551/mitpress/11301.001.0001.

[29] A. Mishkin and M. Pilanci, *Optimal sets and solution paths of ReLU networks*, in International Conference on Machine Learning, ICML 2023, PMLR, 2023.

[30] M. Pilanci, *From complexity to clarity: Analytical expressions of deep neural network weights via Clifford's geometric algebra and convexity*, arXiv:2309.16512, (2023).

[31] M. Pilanci and T. Ergen, *Neural networks are convex regularizers: Exact polynomial-time convex optimization formulations for two-layer networks*, in Proceedings of the 37th International Conference on Machine Learning, vol. 119, 13–18 July 2020, pp. 7695–7705.

[32] H. Purwins, B. Li, T. Virtanen, J. Schlüter, S.-Y. Chang, and T. Sainath, *Deep learning for audio signal processing*, IEEE Journal of Selected Topics in Signal Processing, 13 (2019), pp. 206–219.

[33] P. Savarese, I. Evron, D. Soudry, and N. Srebro, *How do infinite width bounded norm networks look in function space?*, Annual Conference on Learning Theory, (2019), pp. 2667–2690.

[34] J. Serrà, S. Pascual, and C. S. Perales, *Blow: a single-scale hyperconditioned flow for non-parallel raw-audio voice conversion*, Advances in Neural Information Processing Systems, 32 (2019).

[35] R. P. Stanley et al., *An introduction to hyperplane arrangements*, Geometric Combinatorics, 13 (2004), p. 24.

[36] A. M. Stuart, *Uncertainty quantification in bayesian inversion*, ICM2014. Invited Lecture, 1279 (2014).

[37] R. Tibshirani, *Regression shrinkage and selection via the Lasso*, Journal of the Royal Statistical Society: Series B (Methodological), 58 (1996), pp. 267–288, https://arxiv.org/abs/https://rss.onlinelibrary.wiley.com/doi/pdf/10.1111/j.2517-6161.1996.tb02080.x.

[38] R. J. Tibshirani, *The lasso problem and uniqueness*, Electronic Journal of Statistics, 7 (2013), pp. 1456–1490.
[39] S. Vaiter, C. Deledalle, G. Peyré, J. Fadili, and C. Dossal, *The degrees of freedom of the group lasso for a general design*, CoRR, abs/1212.6478 (2012).
[40] Y. Wang, P. Chen, and W. Li, *Projected Wasserstein gradient descent for high-dimensional Bayesian inference*, SIAM/ASA Journal on Uncertainty Quantification, 10 (2022), pp. 1513–1532.
[41] Y. Wang, P. Chen, M. Pilanci, and W. Li, *Optimal neural network approximation of Wasserstein gradient direction via convex optimization*, arXiv:2205.13098, (2022).
[42] Y. Wang, J. Lacotte, and M. Pilanci, *The hidden convex optimization landscape of regularized two-layer relu networks: an exact characterization of optimal solutions*, in International Conference on Learning Representations, 2021.
[43] O. Zahm, T. Cui, K. Law, A. Spantini, and Y. Marzouk, *Certified dimension reduction in nonlinear Bayesian inverse problems*, Mathematics of Computation, 91 (2022), pp. 1789–1835.

Appendix A. Detailed results for Section 2.

Remark A.1 (Parallel to standard architecture conversion). *Let* $\mathbf{W}^{(1)} = \left[\mathbf{W}^{(1,1)} \cdots \mathbf{W}^{(m_1,1)}\right]$*. For* $l \geq 1$*, let* $\mathbf{b}^{(l)} = \left(\mathbf{b}^{(1,l)} \cdots \mathbf{b}^{(m_l,l)}\right)$*. For* $l > 1$*, let* $\mathbf{W}^{(l)} = \text{blockdiag}\left(\mathbf{W}^{(1,l)} \cdots \mathbf{W}^{(m_l,l)}\right)$*. And let* $\boldsymbol{\alpha}, \xi$ *be the same in the standard network as the parallel one.*

Appendix B. Detailed results for Subsection 3.1.

Proofs are defered to Appendix H.3.

Definition B.1. *Define the function* $\mathcal{W}_{\alpha,\beta} : \mathbb{R} \to \mathbb{R}$ *parameterized by* $\alpha, \beta \in \mathbb{R}$ *as*

$$\mathcal{W}_{\alpha,\beta}(x) = \begin{cases} \alpha - x & \text{if } x \leq \alpha \\ x - \alpha & \text{if } \alpha \leq x \leq \beta \\ R_{(\alpha,\beta)} - x & \text{if } \beta \leq x \leq R_{(\alpha,\beta)} \\ x - R_{(\alpha,\beta)} & \text{if } x \geq R_{(\alpha,\beta)}. \end{cases}$$

Lemma B.2 (Examples of the Deep Library). *ReLU features* $\hat{\mathbf{X}}^{(L)}(x)$ *are those plotted in Figure 3.5. As shown in the figure, symmetrized ReLU network features contain generalized reflections of the form* $x_i + x_j - x_k$*. We now describe features for other activations. Let* $L \in \{2, 3\}$*.*
If $\mathbf{L = 2}$*: for* $\mathbf{b}^{(1)} = -x_j \mathbf{W}^{(1)}$*,*

$$\hat{\mathbf{X}}^{(L)}(x) = \begin{cases} \sigma(x_{j_1} - x) & \text{if } \mathbf{W}^{(1)} = -1 \\ \sigma(x - x_{j_1}) & \text{if } \mathbf{W}^{(1)} = 1 \end{cases}$$

If $\mathbf{L = 3}$*:*

if $\sigma(x) = ReLU(x)$ *and* $m_1 = 1$*: for* $\mathbf{b}^{(1)} = -x_{j_1} \mathbf{W}^{(1)}$*,*

if $\mathbf{W}^{(2)} = 1$*:*

$$\hat{\mathbf{X}}^{(2)}(x) = \begin{cases} ReLU^{-}_{\min\{x_{j_1}, x_{j_2}\}}(x) & \text{if } \mathbf{W}^{(1)} = -1 \\ ReLU^{+}_{\max\{x_{j_1}, x_{j_2}\}}(x) & \text{if } \mathbf{W}^{(1)} = 1 \end{cases}$$

if $\mathbf{W}^{(2)} = -1$*:*

$$\hat{\mathbf{X}}^{(L)}(x) = \begin{cases} Ramp^{+}_{x_{j_2}, x_{j_1}}(x) & \text{if } \mathbf{W}^{(1)} = -1 \\ Ramp^{-}_{x_{j_1}, x_{j_2}}(x) & \text{if } \mathbf{W}^{(1)} = 1. \end{cases}$$

if $\sigma(x) = |x|$ *and* $m_1 = 1$*: for* $a \in \left\{x_{j_1}, \frac{x_{j_1} + x_{j_2}}{2}\right\}$*,*

$$\hat{\mathbf{X}}^{(L)}(x) = \begin{cases} \mathcal{W}_{\min\left\{x_{j_2}, R_{(x_{j_2}, a)}\right\}, a}(x) & \text{if } \mathbf{b}^{(1)} = -a\mathbf{W}^{(1)} \\ \mathcal{W}_{\min\left\{R_{(a, x_{j_1})}, R_{(a, x_{j_2})}\right\}, a}(x) & \text{if } \mathbf{b}^{(1)} = -a\mathbf{W}^{(1)}. \end{cases}$$

B.1. Reconstruction results.

In this section, let $(\mathbf{z}^*, \xi^*)$ be a solution to the Lasso problem. We give a map to efficiently and explicitly reconstruct an optimal neural net from $(\mathbf{z}^*, \xi^*)$

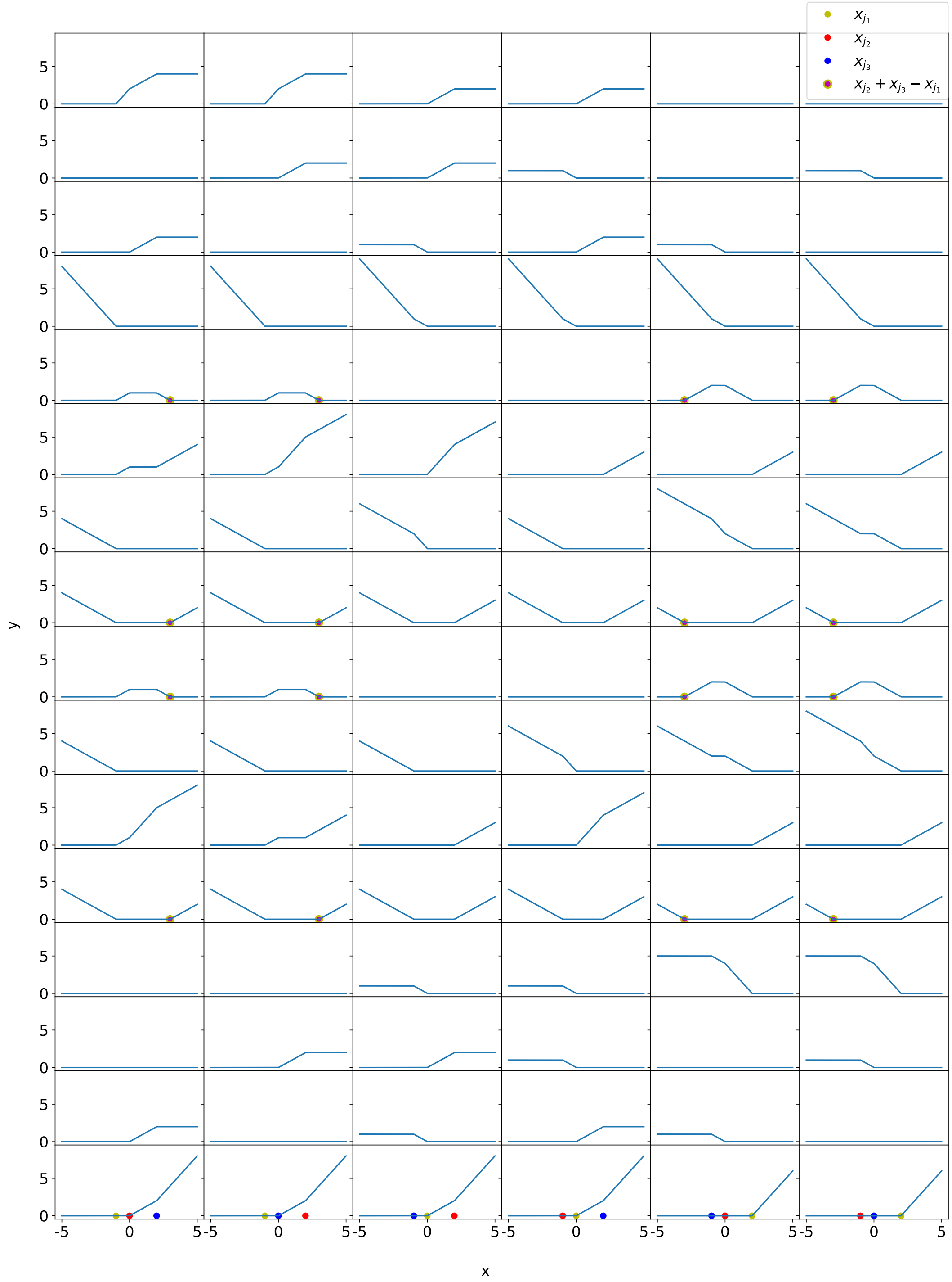


Figure B.1: Figure for Appendix B. Deep library features for a 3-layer symmetrized ReLU network. Each row corresponds to a different set of weights $\mathbf{W}^{(1)} \in \{-1, 1\}^{1\times 2}, \mathbf{W}^{(2)} \in \{-1, 1\}^{2\times 1}$. Each column corresponds to a different ordering of x_i, x_j, x_k. The generalized reflections of x_{j_1} (yellow) across x_{j_2} (red) and x_{j_3} (blue) are depicted by yellow encircling purple.

by leveraging the structure of the deep library (Definition 3.11). Proofs are deferred to Appendix H.3.

Definition B.3 (Reconstructed parameters). *The* reconstructed parameters *for a parallel network are constructed as follows. For each i^{th} column $\mathbf{A}_i$ of the dictionary matrix such that $z_i^* \neq 0$, let $\hat{\mathbf{X}}^{(i,L)}$ be the parallel unit $\hat{\mathbf{X}}^{(L)}$ corresponding to that column in the deep library. Let $\boldsymbol{\alpha}{=}\mathbf{z}^*$ and $\xi{=}\xi^*$. For sign and threshold activation, let all amplitude parameters be 1. Finally, unscale parameters (Definition H.17).*

A *reconstructed network* is a network with reconstructed parameters.

Lemma B.4. *A reconstructed parallel network is optimal in the training problem.*

Recall that each 2-layer feature corresponds to $\mathbf{W}^{(1)}{\in}\{-1,1\}, \mathbf{b}^{(1)}{=}{-}\mathbf{W}^{(1)}x_n$ for some $n{\in}[N]$.

Definition B.5. *Consider a 2-layer ReLU, absolute value, or leaky ReLU network. Let $R^{z\to\alpha}(z){=}\mathrm{sign}(z)\sqrt{|z|}$. For $w{\in}\{-1,1\}, b{\in}\{-x_n : n{\in}[N]\}$, let $R^{\alpha,w,b\to\theta}(\alpha){=}\,(\alpha,\alpha w,-\alpha w b)$. Let $R^{w,b}(z){=}R^{\alpha,w,b\to\theta}\left(R^{z\to\alpha}(z)\right)$. Define the 2-layer* reconstruction function *$R(\mathbf{z})$ which outputs a vector whose i^{th} element is $R^{w,b}(z_i)$, where $(w,b) = \left(\mathbf{W}^{(1)},\mathbf{b}^{(1)}\right)$ corresponds to the i^{th} feature.*

For a 2-layer network, let $w_i{=}\mathbf{W}^{(i,1)}, b_i{=}\mathbf{b}^{(i,1)}{\in}\mathbb{R}$ and $\mathbf{w},\mathbf{b},\boldsymbol{\alpha}$ be vectors stacking together all w_i, b_i, α_i respectively. The reconstructed parameters are $(\boldsymbol{\alpha},\mathbf{w},\mathbf{b}) = R(\mathbf{z}^*)$ and $\xi = \xi^*$.

Appendix C. Detailed results for Subsection 3.2.

Proofs are deferred to Appendix H.4.

C.1. Tree network definition.

Let $L \geq 3, m_2, \cdots, m_L \in \mathbb{N}$. Given $l \in \{0,\cdots,L-2\}$, let $\mathbf{u}$ be an l-tuple where if $l=0$, we denote $\mathbf{u}=\emptyset$ and otherwise, $\mathbf{u}=(u_1,\cdots,u_l)$ such that $u_i \in [m_{L-i}]$ for $i \in [l]$. For an integer a, denote $\mathbf{u}\oplus a$ as the concatenation $(u_1,\cdots,u_l,a)$. For $l \in [L-1]$, and $\mathbf{u}$ of length l, let $\alpha^{(\mathbf{u})}, s^{(\mathbf{u})}, b^{(\mathbf{u})}, \mathbf{w}^{(\mathbf{u})} \in \mathbb{R}$, except let $\mathbf{w}^{(u_1,\cdots,u_{L-1})} \in \mathbb{R}^d$. For all $\mathbf{u}$ of length $L-1$, let $\mathbf{X}^{(u_1,\cdots,u_{L-1})} = \mathbf{x} \in \mathbb{R}^{1\times d}$. For $\mathbf{u}$ of length $l \in \{0,\cdots,L-2\}$, let $\mathbf{X}^{(\mathbf{u})} \in \mathbb{R}$ be defined by

$$\mathbf{X}^{(\mathbf{u})} = \sum_{i=1}^{m_{L-l}} \sigma_{s^{(\mathbf{u}\oplus i)}}\left(\mathbf{X}^{(\mathbf{u}\oplus i)}\mathbf{w}^{(\mathbf{u}\oplus i)} + b^{(\mathbf{u}\oplus i)}\right)\alpha^{(\mathbf{u}\oplus i)}. \tag{C.1}$$

A *tree neural network* is $f_L(\mathbf{x};\theta) = \xi + \mathbf{X}^{(\emptyset)}$. Visualizing the neural network as a tree, $\mathbf{X}^{(\emptyset)}$ is the "root," $\mathbf{u}=(u_1,\cdots u_l)$ specifies the path from the root at level 0 to the $u_l{}^{\mathrm{th}}$ node (or neuron) at level l, $\mathbf{X}^{(\mathbf{u})}$ represents a subtree at this node, and (C.1) specifies how this subtree is built from its child nodes $\mathbf{X}^{(\mathbf{u}\oplus i)}$. The leaves of the tree are all copies of $\mathbf{X}^{(u_1,\cdots,u_{L-1})} = \mathbf{X}$. Let $\mathcal{U} = \prod_{l=0}^{L-2}[m_{L-l}]$. The regularized and bias parameter sets (Subsection 2.1) are $\theta_w^{(i)} = \left\{\alpha^{(\mathbf{u})}, s^{(\mathbf{u})}, \mathbf{w}^{(\mathbf{u})} : \mathbf{u}\in\mathcal{U}, u_1 = i\right\}, \theta_b^{(i)} = \left\{b^{(\mathbf{u})} : \mathbf{u}\in\mathcal{U}, u_1=i\right\}$. For tree networks, let $\boldsymbol{\alpha} = \left(\alpha^{(1)},\cdots,\alpha^{(m_L)}\right) \in \mathbb{R}^{m_L}$.

The rest of this section includes additional results. The reconstruction defined below uses the unscaling operation (Definition H.17).

Lemma C.1. *Consider a 3-layer sign-activated network. Suppose $\mathbf{z}^*$ is optimal in the Lasso problem, and $m_L \geq ||\mathbf{z}^*||_0$. Let $\xi{=}0$. Let $\boldsymbol{\alpha}{=}\mathbf{z}^*$. Suppose $\mathbf{A}_i$ switches at $I_1^{(i)} < I_2^{(i)} < \cdots < I_{m^{(i)}}^{(i)}$. Let $\mathbf{W}_n^{(i,1)}{=}1, \mathbf{b}_n^{(i,1)}{=}-x_{I_n^{(i)}-1}, \mathbf{W}_n^{(i,2)}{=}(-1)^{n+1}$ and $\mathbf{b}^{(i,2)}{=}-\mathbf{1}\left\{m^{(i)}\ \mathrm{odd}\right\}$. Let all amplitude*

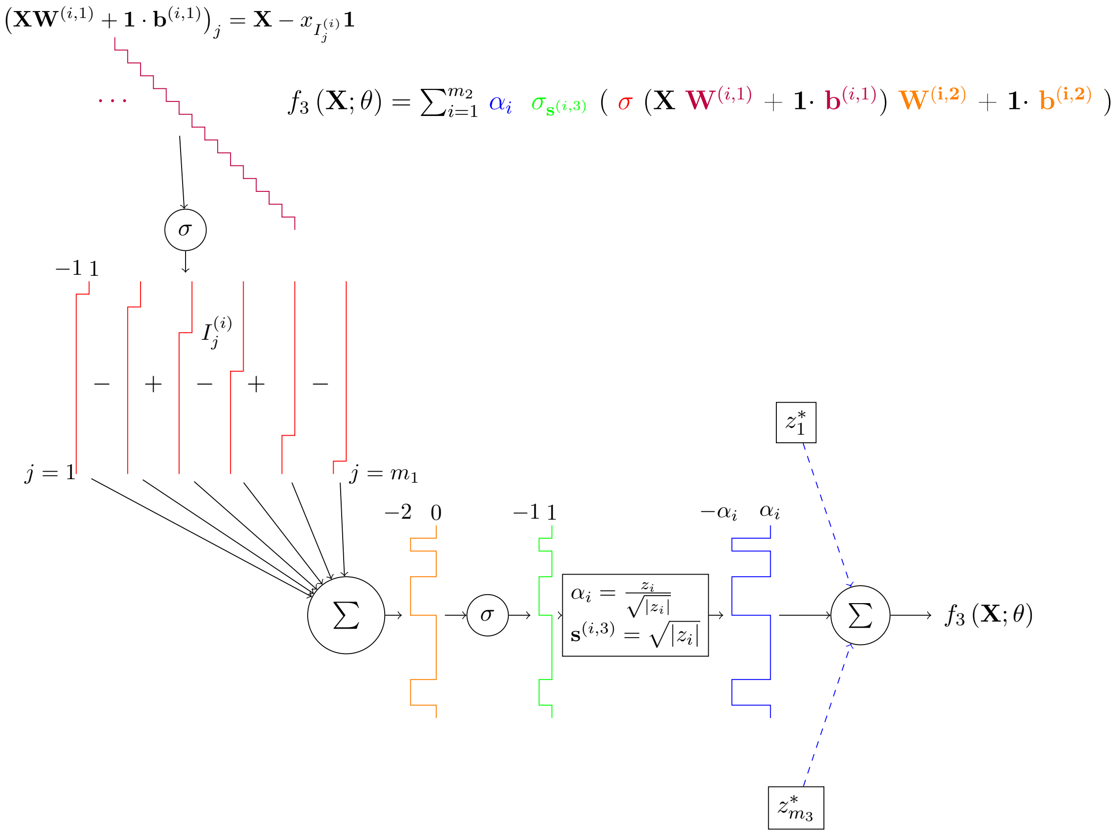


Figure B.2: Figure for Appendix B. Output of an optimal 3-layer neural net with sign activation reconstructed from a Lasso solution $\mathbf{z}^*$ using Lemma C.1 . The pulse colors correspond to network operations. The alternating $+,-$ represent $\mathbf{W}^{(i,2)} = (1,-1,1,-1,\cdots)$. The red and green pulses illustrate 2 and 3-layer dictionary features, respectively (Theorem 3.12, Theorem 3.17), while the other colors represent multiplication by weights and amplitudes.

parameters be 1. *Let* $\mathcal{I}=\{i : z_i \neq 0\}$. *If* $i \notin \mathcal{I}$, *set* $s^{(i,l)}, \alpha_i, \mathbf{W}^{(i,l)}, \mathbf{b}^{(i,l)}$ *to zero. These parameters are optimal when unscaled (Definition H.17).*

In the next result, we only consider networks with sign activation. The number of neurons in each layer l of a 2 and 3-layer network is denoted by m_l' and m_l, respectively.

Corollary C.2. *Consider a* 3*-layer sign-activated network. There exists an equivalent* 2*-layer network with* $m_2' = m_1 m_3$ *neurons. Let* $p^*_{L,\beta}$ *be the optimal value of the training problem* (1.1) *for* L *layers, regularization* β *and sign activation. Then, for* 2*-layer nets trained with* $m_2' \geq m_1 m_2$ *neurons,* $p^*_{L=3,\beta} \leq p^*_{L=2,\beta} \leq p^*_{L=3,m_1\beta}$.

Corollary C.2 states that a 3-layer net can achieve lower training loss than a 2-layer net, but only while its regularization β is at most m_1 times stronger.

C.2. Example of 2-D data. The next result extends Theorem 3.17 to 2-D data on the upper half plane. We consider parallel neural nets without internal bias parameters, that is, $\mathbf{b}^{(i,l)} \notin \theta$ (they can be thought of as set to 0). Proofs are located in Appendix H.6.

Theorem C.3. *Consider a Lasso problem whose dictionary is the switching set $\mathbf{H}^{(K)}$, $\xi = 0$, and with solution $\mathbf{z}^*$. Let $m^* = \|\mathbf{z}^*\|_0$. This Lasso problem is equivalent to the training problem for a sign-activated network without internal biases that is 2-layer or rectangular, satisfies $m_L \geq m^*$, $m_{L-2} = K$, and is trained on 2-D data with unique angles in $(0, \pi)$.*

The next result reconstructs an optimal neural net from the Lasso problem in Theorem C.3. The unscaling operation is used (Definition H.17).

Lemma C.4. *Let $\mathbf{R}_{\frac{\pi}{2}} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$ be the counterclockwise rotation matrix by $\frac{\pi}{2}$. An optimal parameter set for the training problem in Theorem C.3 when $L = 2$ is the unscaled version of $\theta = \left\{ \alpha_i = z_i^*, \mathbf{s}^{(i,1)} = 1, \mathbf{W}^{(i,1)} = \mathbf{R}_{\frac{\pi}{2}} \left(\mathbf{x}^{(i)}\right)^T, \xi = 0 : z_i^* \neq 0 \right\}$, where $\mathbf{z}^*$ is optimal in the Lasso problem.*

Appendix D. Detailed results for Section 4.

Proofs in this section are deferred to Appendix H.8. Given a square wave of period T, let $k = \frac{N}{T}$ be the number of cycles it has. There is a *critical value* $\beta_c = \max_{n \in [N]} |\mathbf{A}_n^T \mathbf{y}|$ such that when $\beta > \beta_c$, $\mathbf{z}^* = \mathbf{0}$ is optimal in the Lasso problem (12). Let $\beta_T = \frac{\beta}{\beta_c}$. Theorem 3.12 specifies the Lasso problem for a 2-layer network with sign activation. We will use the $N \times N$ dictionary matrix $\mathbf{A}$ with $\mathbf{A}_{i,n} = \sigma(x_i - x_n)$, as defined in Corollary 3.15. The Lasso solution $\mathbf{z}^* \in \mathbb{R}^N$ is unique, by Proposition G.3.

Theorem D.1. *Consider the Lasso problem for a 2-layer net with sign activation and square wave target vector of period T. The critical value is $\beta_c = T$. And the solution is*

$$
z^*_{\frac{T}{2} i} = \begin{cases} \begin{cases} \frac{1}{2}(1 - \beta_T)_+ & \text{if } i \in \{1, 2k-1\} \\ 0 & \text{else} \end{cases} & \text{if } \beta_T \geq \frac{1}{2} \\ \begin{cases} 1 - \frac{3}{2}\beta_T & \text{if } i \in \{1, 2k-1\} \\ (-1)^{i+1}(1 - 2\beta_T) & \text{else} \end{cases} & \text{if } \beta_T \leq \frac{1}{2}. \end{cases}, \tag{D.1}
$$

for $i \in [2k-1]$ and $z_n^* = 0$ at all other $n \in [N]$.

Corollary D.2. *For a square wave target vector with period T, there is an optimal 2-layer neural network with sign activation specified by*

$$
\begin{aligned}
f_2(x; \theta) &= 0, && \text{if } \beta_T \geq 1 \\
f_2(x; \theta) &= \begin{cases} -(1 - \beta_T) & \text{if } x < x_{N - \frac{T}{2}} \\ 0 & \text{if } x_{N - \frac{T}{2}} \leq x < x_{\frac{T}{2}} \\ 1 - \beta_T & \text{if } x \geq x_{\frac{T}{2}} \end{cases} && \text{if } \frac{1}{2} \leq \beta_T \leq 1 \\
f_2(x; \theta) &= \begin{cases} -(1 - \beta_T) & \text{if } x < x_{N - \frac{T}{2}} \\ (-1)^i (1 - 2\beta_T) & \text{if } x_{\frac{T}{2}(i+1)} \leq x < x_{\frac{T}{2} i}, \quad i \in [2k-2] \\ 1 - \beta_T & \text{if } x \geq x_{\frac{T}{2}} \end{cases} && \text{if } \beta_T \leq \frac{1}{2}
\end{aligned}
$$

Theorem D.3. *Consider the Lasso problem for a 3-layer network with sign activation and target vector a square wave of period T and $m_3 \geq 2\frac{T}{N} - 1$. Then $\beta_c = N$ and $\mathbf{A}_i = -\mathbf{h}^{(T)}$ for some i. The solution to the Lasso problem is $z_i^* = -(1-\beta_T)_+$ and $z_n^* = 0$ at all other n.*

Corollary D.4. *Let $x_0 = \infty$. For a square wave target vector with period T, there is an optimal 3-layer neural net with sign activation specified by $f_3(x;\theta) = (1-\beta_T)_+(-1)^{(i-1)}$ if $x_{\frac{T}{2}i} \leq x < x_{\frac{T}{2}(i-1)}$ for $i \in [2k-1]$, and $f_3(x;\theta) = -(1-\beta_T)_+$ if $x < x_{N-\frac{T}{2}}$.*

Consider the 2-layer network in the left plot of Figure 3.7. When $\beta \to 0$, the network interpolates the target vector perfectly. As β_T increases to $\frac{1}{2}$ (red dots) from 0, the magnitude of the middle segments decrease at a faster rate than the outer segments until at $\beta_T = \frac{1}{2}$, the net consists of just the outer segments (magenta dots). As β_T increases to 1 from $\frac{1}{2}$ (blue dots), these outer segments decrease until the neural net is the zero function.

In Figure 3.8, the training data is chosen randomly from a uniform distribution on $[-100, 100]$. Suppose we use the neural net as a binary classifier whose output is the sign of $f_L(x;\theta)$, where the network is "undecided" if $f_L(x;\theta) = 0$. The red, blue and white indicate classifications of -1, 1, and "undecided," respectively. For $\beta < \beta_c$, the 3-layer net always classifies the training data accurately, but the 2-layer net is undecided on all but the first and last interval if $\beta > \beta_c/2$. When used as a regressor, for each β, the magnitude of the 3-layer net's prediction is the same over all samples, while the 2-layer net is biased toward a stronger prediction on the first and last intervals. In this sense, the 3-layer network generalizes better.

Appendix E. Detailed results of Section 5.

The *neural net (NN) autoregression training problem* is

$$\min_{\theta \in \Theta} \frac{1}{T} \sum_{t=1}^{T} \left(f_2^{\mathrm{NN}}(x_t;\theta) - x_{t+1}\right)^2 + \frac{\beta}{2}\|\theta_w\|_2^2. \tag{E.1}$$

This model represents predictors of x_{t+1} from x_t. By Theorem 3.12, this non-convex problem is equivalent to the convex Lasso problem (1.2) where $\mathbf{A}_{i,j} = |x_i - x_j|$ and $y_i = x_{i+1}$.

We now compare solving the autoregression (E.1) and quantile regression (5.3) problems directly with 5 trials of stochastic gradient descent (SGD) initializations versus using the Lasso problem (1.2). We also compare against the baseline linear method $f(x;\theta) = ax + b$ (AR1+bias), where we include an additional bias term b. We test upon real financial data for bitcoin price, including minutely bitcoin (BTC) price (BTC-2017min) and hourly BTC price (BTC-hourly). We consider the training problem on τ-quantile regression (Appendix E) with $\tau = 0.3$ and $\tau = 0.7$. For each dataset, we first choose T data points as a training set and the consecutive T data points as a test set. The numerical results are presented in Figure 5.1. We observe that cvxNN provides a consistent lower bound on the training loss and demonstrates strong generalization properties, compared to large fluctuation in the loss curves of NN. More results can be found in Appendix H.12.

We first build a network $f_2^{\mathrm{NN}}(x;\theta)$ (5.2) with m known, or *planted* neurons. We use this network to generate training samples $x_1, \ldots, x_{T+1}$ based on (5.1) with $f(x;\theta) = f_2^{\mathrm{NN}}(x;\theta)$ where $x_1 \sim \mathcal{N}(0, \sigma^2)$. Using the same model $f_2^{\mathrm{NN}}(x;\theta)$, we also generate test samples $x_1^{\mathrm{test}}, \ldots, x_{T+1}^{\mathrm{test}}$ in an analogous way. We use $T = 1000$ time samples. Then, we try to recover the planted neurons based on only the training samples by solving the NN AR/QR training problems.

In Figure H.6, we present experiments based on the selection of m planted neurons and noise level σ^2. More results can be found in Appendix H.12. The neural net trained with Lasso is labeled cvxNN, which we observe has lower training loss. This appears to occur because different trials of NN (neural net trained directly without Lasso) get stuck into local minima. The global optimum that cvxNN reaches also enjoys effective generalization properties, as seen by the test loss. The *regularization path* is the optimal neural net's performance loss as a function of the regularization coefficient β. Figure H.7 plots the regularization path for $\sigma^2 = 1$ and $m = 5$. The regularization path taken by cvxNN is smoother than NN, and can therefore be found more precisely and robustly by using the Lasso problem.

Appendix F. The solution sets of Lasso and the training problem.

We have shown that training neural networks on 1-D data is equivalent to fitting a Lasso model. Now we develop analytical expressions for all minima of the Lasso problem and its relationship to the set of all minima of the training problem. These results, which build on the existing literature for convex reformulations (29) as well as characterizations of the Lasso (12), illustrate that the Lasso model provides insight into non-convex networks. We focus on 2-layer models with ReLU, leaky ReLU and absolute value activations, although our results can be extended to other architectures by considering the corresponding neural net reconstruction. Proofs are deferred to Appendix H.11.

We start by characterizing the set of global optima to the Lasso problem (1.2). Suppose $(\mathbf{z}^*, \xi^*)$ is a solution to the convex training problem. In this notation, the optimal model fit $\hat{\mathbf{y}}$ and equicorrelation set $\mathcal{E}_\beta$ are given by

$$\hat{\mathbf{y}} = \mathbf{A}\mathbf{z}^* + \xi^*\mathbf{1}, \qquad \mathcal{E}_\beta = \left\{ i : |\mathbf{A}_i^\top (\hat{\mathbf{y}} - \mathbf{y})| = \beta \right\},$$

where $\hat{\mathbf{y}}$ is unique over the optimal set (39; 38). The equicorrelation set contains the features maximally correlated with the residual $\hat{\mathbf{y}} - \mathbf{y}$ and plays a critical role in the solution set.

Proposition F.1. *Suppose $\beta > 0$. Then the set of global optima of the Lasso problem (1.2) is*

$$\text{(F.1)} \quad \Phi^*(\beta) = \left\{ (\mathbf{z}, \xi) : z_i \neq 0 \Rightarrow \operatorname{sign}(z_i) = \operatorname{sign}\left(\mathbf{A}_i^\top (\hat{\mathbf{y}} - \mathbf{y}) \right),\ z_i = 0\, \forall i \notin \mathcal{E}_\beta,\ \mathbf{A}\mathbf{z} + \xi\mathbf{1} = \hat{\mathbf{y}}. \right\}$$

The solution set $\Phi^*(\beta)$ is polyhedral and its vertices correspond exactly to minimal models, i.e. models with the fewest non-zero elements of $\mathbf{z}$ (29). Let R be the reconstruction function described in Definition B.5. All networks generated from applying R to a Lasso solution are globally optimal in the training problem. The next result gives a full description of such networks. The 2-layer parameter notation defined in Subsection 3.1 is used.

Proposition F.2. *Suppose $\beta > 0$ and the activation is ReLU, leaky ReLU or absolute value. The set of all 2-layer Lasso-reconstructed networks is*

$$\text{(F.2)} \quad \begin{aligned} R(\Phi(\beta)) = \Bigg\{ & (\mathbf{w}, \mathbf{b}, \alpha, \xi) : \alpha_i \neq 0 \Rightarrow \operatorname{sign}(\alpha_i) = \operatorname{sign}\left(\mathbf{A}_i^\top (\mathbf{y} - \hat{\mathbf{y}}) \right), b_i = -x_i \frac{\tilde{\alpha}_i}{\sqrt{|\alpha_i|}}, \\ & w_i = \frac{\tilde{\alpha}_i}{\sqrt{|\alpha_i|}};\ \alpha_i = 0\, \forall i \notin \mathcal{E}_\beta,\ f_2(\mathbf{X}; \theta) = \hat{\mathbf{y}} \Bigg\}. \end{aligned}$$

Proposition F.2 shows that all neural nets trained using our Lasso and reconstruction share the same model fit whose set of active neurons is at most the equicorrelation set. By finding just *one* optimal neural net that solves Lasso, we can form $R(\Phi^{\beta})$ and compute all others.

The min-norm solution path is continuous for the Lasso problem (38). Since the solution mapping in Definition B.3, Appendix H.3 is continuous, the corresponding reconstructed neural net path is also continuous as long as the network is sufficiently wide. Moreover, we can compute this path efficiently using the LARS algorithm (12). This is in contrast to the under-parameterized setting, where the regularization path is discontinuous (29).

What subset of optimal, or more generally, stationary, points of the non-convex training problem (1.1) consist of Lasso-generated networks $R(\Phi(\beta))$? First, $R(\Phi(\beta))$ can generate additional optimal networks through neuron splitting, described as follows. Consider a single neuron $\alpha\sigma_s(wx+b)$ (where $\alpha, w, b \in \mathbb{R}$), and let $\{\gamma_i\}_{i=1}^n \subset [0,1]^n$ be such that $\sum_{i=1}^n \gamma_i = 1$. The neuron can be *split* into n neurons $\left\{\sqrt{\gamma_i}\alpha\sigma\left(\sqrt{\gamma_i}wx + \sqrt{\gamma_i}b\right)\right\}_{i=1}^n$ (42). For any collection Θ of parameter sets θ, let $P(\Theta)$ be the collection of parameter sets generated by all possible neuron splits and permutations of each $\theta \in \Theta$. Next, let $\mathcal{C}(\beta)$ and $\tilde{C}(\beta)$ be the sets of Clarke stationary points and solutions to the non-convex training problem (1.1), respectively.

Proposition F.3. *Suppose $L = 2, \beta > 0$, the activation is ReLU, leaky ReLU or absolute value and $m^* \leq m \leq 2N$. Let $\Theta^P = \{\theta : \forall i \in [m], \exists j \in [N] \text{ s.t. } b_i = -x_j w_i\}$. Then*

$$P(R(\Phi(\beta))) = \tilde{\mathcal{C}}(\beta) \cap \Theta^P = \mathcal{C}(\beta) \cap \Theta^P. \tag{F.3}$$

Proposition F.3 states that up to neuron splitting and permutation, our Lasso method gives all stationary points in the training problem satisfying $b_i = -x_i w_i$. Moreover, all such points are optimal in the training problem, similar to (19).

Since optimal solutions are stationary, a neural net reconstructed from the Lasso model is in $\tilde{C}(\beta) \subset \mathcal{C}(\beta)$. However, $C(\beta) \not\subset \Theta^P$. This is because there may be other neural nets with the same output on $\mathbf{X}$ as the reconstructed net so that they are all in $\mathcal{C}(\beta)$, but that differ in the the unregularized parameters $\mathbf{b}$ and ξ, so that they are not in Θ^P. For example, if β is large enough, the Lasso solution is $\mathbf{z} = \mathbf{0}$ (12), so the reconstructed net will have $\boldsymbol{\alpha} = \mathbf{0}$, which makes the neural net invariant to $\mathbf{b}$. In this section, we analyzed the general structure of the Lasso solution set when $\beta > 0$. Next, we analyze the Lasso solution set for specific activations and training data when $\beta \to 0$.

Appendix G. Solution sets of Lasso under minimal regularization. One of the insights that the Lasso formulation provides is that under minimal regularization, certain neural nets perfectly interpolate the data.

Corollary G.1. *For the ReLU, absolute value, sign, and threshold networks with $L = 2$ layers, and sign-activated deeper networks, if $m_L \geq m^*$, then $f_L(\mathbf{X}; \theta) \to \mathbf{y}$ as $\beta \to 0$.*

Proofs in this section are deferred to Appendix H.7. In Corollary G.1, m^* depends on L and the activation and is defined in Theorem 3.12 and Theorem 3.17. The Lasso equivalence and reconstruction also shed light on optimal neural network structure as regularization decreases. The *minimum (l_1) norm subject to interpolation* version of the Lasso problem is

$$\min_{\mathbf{z},\xi} \|\mathbf{z}\|_1 \text{ s.t. } \mathbf{A}\mathbf{z} + \xi\mathbf{1} = \mathbf{y}. \tag{G.1}$$

Loosely speaking, as $\beta \to 0$, if $\mathbf{A}$ has full column rank, the Lasso problem (1.2) "approaches" the minimum norm problem (G.1), where $\xi = 0$ for sign and threshold activations. The rest of this section describes the solution sets of (G.1) for certain networks.

Proposition G.2. *Let $L = 2$. Suppose σ is the absolute value activation. Let $\mathbf{z}^*$ be a solution to* (G.1). *Then, we have $z_1^* z_n^* \leq 0$. Moreover, the entire solution set of* (G.1) *for $\mathbf{z}^*$ is*

$$\left\{ \mathbf{z}^* + t \, \mathrm{sign}(z_1^*)(1, 0, \cdots, 0, 1)^T \middle| -|z_1^*| \leq t \leq |z_N^*| \right\}. \tag{G.2}$$

Proposition G.3. *For a 2-layer network with sign activation and $\beta \geq 0$, the Lasso problem* (1.2) *has a unique solution. Furthermore, the minimum norm solution in* (G.1) *is $\mathbf{z}^* = \mathbf{A}^{-1}\mathbf{y}$.*

Given an optimal bias term ξ^*, if $\mathbf{A}$ is invertible, then $\mathbf{z}^* = \mathbf{A}^{-1}(\mathbf{y} - \xi^* \mathbf{1})$ is optimal in (G.1). Appendix H.8 explicitly finds $\mathbf{A}^{-1}$ for some activation functions. The structure of $\mathbf{A}^{-1}$ suggests the behavior of neural networks under minimal regularization: sign-activated neural networks act as difference detectors, while neural networks with absolute value activation, whose subgradient is the sign activation, act as a second-order difference detectors (see Remark H.41). The next result shows that threshold-activated neural networks are also difference detectors, but for the special case of positive, nonincreasing y_n. An example of such data is cumulative revenue, e.g. $y_n = \sum_{i=1}^n r_i$ where r_i is the revenue in dollars earned on day i.

Proposition G.4. *Let $L = 2$. Suppose σ is threshold activation and $y_1 \geq \cdots \geq y_N \geq 0$. Then*

$$z_n^* = \begin{cases} y_n - y_{n-1} & \text{if } n \leq N-1 \\ y_N & \text{if } n = N \\ 0 & \text{else} \end{cases}$$

is the unique solution to the minimum norm problem (G.1).

The next result gives a lower bound on the optimal value of the minimum weight problem for ReLU networks. If we can find $\mathbf{z}$ with a l_1-norm that meets the lower bound and a ξ such that $\mathbf{A}\mathbf{z} + \xi \mathbf{1} = \mathbf{y}$, then we know $\mathbf{z}, \xi$ is optimal. In this section, for $n \in [N-1]$, let $\mu_n = \frac{y_n - y_{n+1}}{x_n - x_{n+1}}$ be the slope between the n^{th} and $n + 1^{th}$ data points. Let $\mu_N = 0$.

Lemma G.5. *The optimal value $\|\mathbf{z}^*\|_1$ of the minimum norm problem* (G.1) *for deep narrow networks with ReLU or absolute value activation is at least $\max_{n \in [N-1]} |\mu_n|$.*

When L=2, the next result gives a solution to the min-norm problem. For $i \in [N]$, let $(z_+)_i$ and $(z_-)_i$ be the Lasso variable corresponding to the features $\mathrm{ReLU}_{x_i}^+$ and $\mathrm{ReLU}_{x_i}^-$, respectively. In other words, $\mathbf{z}_+$ corresponds to $\mathbf{A}_+$, and $\mathbf{z}_-$ corresponds to $\mathbf{A}_-$ as defined in Corollary 3.15.

Lemma G.6. *The min-norm problem* (G.1) *for L=2, σ=ReLU has optimal value $\|\mathbf{z}^*\|_1 = \sum_{n=1}^{N-1} |\mu_n - \mu_{n+1}|$. An optimal solution is $(z_+)_{n+1} = \mu_n - \mu_{n+1}$ for $n \in [N-1]$, $\mathbf{z}_- = \mathbf{0}$, and $\xi = y_N$.*

Lemma G.7. *For a 3-layer symmetrized ReLU network and training data as shown in Figure 3.2, the optimal value $\|\mathbf{z}^*\|_1$ of the minimum norm problem* (G.1) *is at least* 1.

Lemma G.7 can be generalized to more complex sets of training data.

Appendix H. Numerical results. The following simulations support our theoretical results.

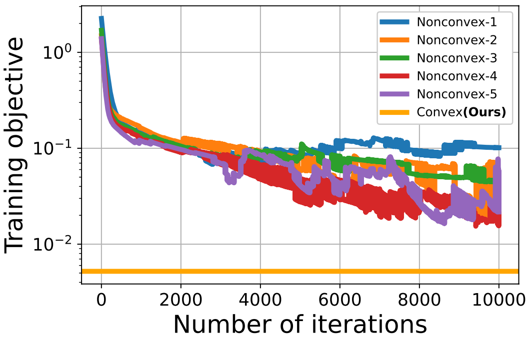


Figure H.1: Training objective

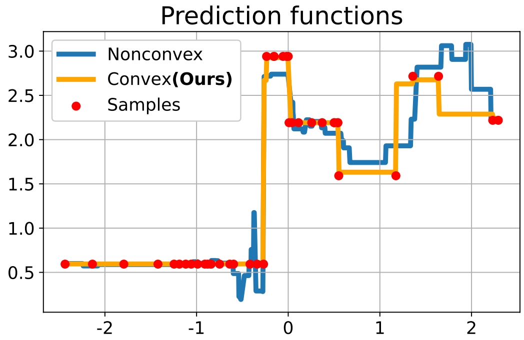


Figure H.2: Function fit

Figure H.3: Figure for Appendix H. Training objective (left) and function fit (right) for a neural net using the convex Lasso problem versus the non-convex training problem using STE.

In Figure 3.2, a deep narrow, absolute value network is trained. In Figure 3.4, a 3-layer symmetrized ReLU network is trained. Both neural nets have standard architecture, m_L=100, and are trained with Adam with the non-convex problem. The learning rate is $5(10^{-3})$, weight decay is 10^{-4}, and β=10^{-7}≈0. When β→0, the Lasso problem approaches the minimum norm problem (G.1). For each L∈3, 4, 5, the neural net reconstructed from the single feature in the left plot with corresponding Lasso parameter $z_i^* = 1$ and $\xi^* = 0$ is optimal in the minimum norm problem (G.1) by Lemma G.5 and Lemma G.7. This network is used to initialize a subset of the neurons in the non-convex training. All other weights are initialized randomly according to Pytorch defaults. The figures show that the networks trained with the non-convex problem closely match the Lasso solutions. The networks exhibit breakpoints at data points and their reflections not in the training data. The standard architecture in the non-convex model shows the applicability of the Lasso formulation. SGD gives similar results as Adam.

In addition to training ReLU networks under minimal β, we train neural networks with threshold activations and larger β. We label $N = 40$ 1-D data samples from an i.i.d. distribution $x \in \mathcal{N}(0, 1)$ with a Bernoulli random variable. We use β=10^{-3} to train a 2-layer neural network with threshold activation using the Lasso problem (1.2) and a non-convex training approach based on the Straight Through Estimator (STE) (3). As illustrated in Figure H.3, the convex training approach achieves significantly lower objective value than all of the non-convex trials with different seeds for initialization. Figure H.3 also plots the predictions of the models. We observe that the non-convex training approach fits the data samples exactly on certain intervals but provides a poor overall function fitting, whereas our convex models yields a more reasonable piecewise linear fit. In particular, the neural net trained with the non-convex problem fits the data in Figure H.3 poorly compared to Figure 3.4 and Figure 3.2. This may occur because in Figure H.3, the data set is larger and more complex, and STE training is used because of the threshold activation, and β is larger instead of being close to zero.

Supplementary Material.

Definitions and preliminaries.

H.1. Activation function. A function f is *bounded* if there is $M \geq 0$ with $|f(x)| \leq M$ for all x. If $\sigma(x)$ is piecewise linear around zero, $\sigma(x)$ is bounded if and only if $a^- = a^+ = 0$, e.g. $\sigma(x)$ is a threshold or sign activation. We call f *symmetric* if it is an even or odd function, for example absolute value. The activation $\sigma(x)$ is defined to be *homogeneous* if for any $a \geq 0$, $\sigma(ax) = a\sigma(x)$. Homogeneous activations include ReLU, leaky ReLU, and absolute value, and don't have amplitude parameters. We say $\sigma(x)$ is *sign-determined* if its value depends only on the sign of its input and not its magnitude. Threshold and sign activations are sign-determined.

H.2. Effective depth.

Remark H.1. *Plugging* (2.2) *into itself shows that in parallel network, for* $l \in [L-2]$,

$$\hat{\mathbf{X}}^{(i,l+2)} = \sigma_{\mathbf{s}^{(i,l+1)}}\left(\sigma\left(\hat{\mathbf{X}}^{(i,l)}\mathbf{W}^{(i,l+1)} + \mathbf{b}^{(i,l)}\right)\mathbf{s}^{(i,l)}\mathbf{W}^{(i,l+1)} + \mathbf{b}^{(i,l+1)}\right).$$

Similarly, plugging in (C.1) *into itself shows that in a tree network, for* $0 \leq l \leq L-3$,

$$\mathbf{X}^{(\mathbf{u})} = \sum_{i=1}^{m_{L-l}} \alpha^{(\mathbf{u}\oplus i)} \sigma_{s^{(\mathbf{u}\oplus i)}}\left(b^{(\mathbf{u}\oplus i)} + \sum_{j=1}^{m_{L-l-1}} \alpha^{(\mathbf{u}\oplus i \oplus j)} \sigma\left(\mathbf{X}^{(\mathbf{u}\oplus i\oplus j)}\mathbf{w}^{(\mathbf{u}\oplus i\oplus j)} + b^{(\mathbf{u}\oplus i\oplus j)}\right) s^{(\mathbf{u}\oplus i\oplus j)}\mathbf{w}^{(\mathbf{u}\oplus i)}\right).$$

The inner parameters *of a parallel network are* $\mathbf{s}^{(i,l)}$ *for* $l \leq L-2$ *and* $\mathbf{W}^{(i,l)}$ *for* $l \leq L-1$. *In a tree network, they are* $\left(\mathbf{s}^{\mathbf{u}\oplus 1}, \cdots \mathbf{s}^{\mathbf{u}\oplus m_{L-l}}\right), \left(\alpha^{\mathbf{u}\oplus 1}, \cdots \alpha^{\mathbf{u}\oplus m_{L-l}}\right)$ *for* $\mathbf{u}$ *of positive length, and* $\left(\mathbf{w}^{\mathbf{u}\oplus 1}, \cdots \mathbf{w}^{\mathbf{u}\oplus m_{L-l}}\right)$ *for* $\mathbf{u}$ *of any length. Suppose* σ *is sign-determined. Then* $f_L(\mathbf{X};\theta)$ *is invariant to the value of the inner parameters, so they would be driven to* 0 *by weight regularization. We define the minimum value in* (1.1) *as an infimum which is approached as the norms of the inner parameters approach* 0. *Therefore the inner parameters are not regularized (the effective depth is* 2*), and we optimize for their directions rather than their magnitudes.*

Parallel and tree networks with data dimension $d \geq 1$.

For convenience, we will omit ξ when writing $f_L(\mathbf{X};\theta)$. This does not change the training problem because we can write (1.1) as $\min_{\theta - \{\xi\}} \frac{\tilde{\beta}}{\tilde{L}} \|\theta_w\|_2^2 + \min_{\xi} \{\mathcal{L}_{\mathbf{y}}(f_L(X) - \xi\mathbf{1} + \xi\mathbf{1})\}$ and apply the change of variables/functions $\theta' = \theta - \{\xi\}$, $f_L(\mathbf{X};\theta)' = f_L(\mathbf{X};\theta) - \xi\mathbf{1}$ and $\mathcal{L}_{\mathbf{y}}(\mathbf{z})' = \min_{\xi} L_y(\mathbf{z} + \xi\mathbf{1})$. The loss function absorbs ξ while preserving its convexity. We begin by assuming the data is d-dimensional and consider the general *training problem*

$$\min_{\theta \in \Theta} \mathcal{L}_{\mathbf{y}}(f_L(\mathbf{X};\theta)) + \frac{\beta}{\tilde{L}} r(\theta)$$

with a general regularization of the form $r(\theta) = \sum_{i=1}^{m_L} \sum_{\theta^{(i,l)} \in \theta_w^{(i)}} \left(r^{(i,l)}\left(\theta^{(i,l)}\right)\right)^{\tilde{L}}$, where $\theta^{(i,l)}$ is a parameter such as $\mathbf{W}^{(i,l)}$ and $r^{(i,l)}$ is a regularization function such as $r^{(i,l)}\left(\mathbf{W}^{(i,l)}\right) = \left\|\mathbf{W}^{(i,l)}\right\|_p$.

Assume $r^{(i,l)}$ is nonnegative and positively homogeneous, i.e., for any $\alpha \geq 0$, $r^{(i,l)}(\alpha \mathbf{W}) = \alpha r^{(i,l)}(\mathbf{W}) \geq 0$. The training problem (1.1) is a special case of the general training problem.

Definition H.2. *A simplified neural network with sign-determined activation is*

$$f_L(\mathbf{x};\theta) = \xi + \sum_{i=1}^{m_L} \hat{\mathbf{X}}^{(i,L)} \alpha_i s^{(i,L-1)}$$
$$\hat{\mathbf{X}}^{(i,l+1)} = \sigma\left(\hat{\mathbf{X}}^{(i,l)} \mathbf{W}^{(i,l)} + \mathbf{b}^{(i,l)}\right), l \in [L-1] \tag{H.1}$$
$$\theta_w^{(i)} = \left\{\alpha_i, s^{(i,L-1)}\right\}, \theta_b^{(i)} = \left\{\mathbf{b}^{(i,l)}, \mathbf{W}^{(i,l)} : l \in [L-1]\right\} \text{ for } i \in [m_L]$$

for a parallel network, and

$$f_L(\mathbf{x};\theta) = \xi + \sum_{i=1}^{m_L} \sigma\left(\mathbf{X}^{(i)} + b^{(i)}\mathbf{1}\right) \alpha_i s^{(i)}$$
$$\mathbf{X}^{(\mathbf{u})} = \begin{cases} \sum_{i=1}^{m_{L-l}} \alpha^{(\mathbf{u}\oplus i)} \sigma\left(\mathbf{X}^{(\mathbf{u}\oplus i)} + b^{(\mathbf{u}\oplus i)}\right) & \text{if } 1 \leq l \leq L-3 \\ \sum_{i=1}^{m_{L-l}} \alpha^{(\mathbf{u}\oplus i)} \sigma\left(\mathbf{X}\mathbf{w}^{(\mathbf{u}\oplus i)} + b^{(\mathbf{u}\oplus i)}\right) & \text{if } l = L-2 \end{cases} \tag{H.2}$$
$$\theta_w^{(i)} = \left\{\alpha^{(i)}, \mathbf{s}^{(i)}\right\} \theta_b^{(i)} = \left\{\alpha^{(\mathbf{u})}, b^{(\mathbf{u})} : \mathbf{u} \in \mathcal{U}, u_1 = i\right\} \text{ for } i \in [m_L]$$

where $\mathbf{u}$ *has positive length for* $\alpha^{(\mathbf{u})} \in \theta_b^{(i)}$, *for a tree network.*

In other words, a simplified network has amplitude parameters only in the last layer.

Lemma H.3. *The simplified neural network is equivalent to the original neural network.*

Proof. By Remark H.1, it suffices to only regularize the outermost $\tilde{L}=2$ layers. For parallel networks, apply a change of variables $\mathbf{W}^{(i,l)'} = \mathbf{s}^{(i,l-1)} \mathbf{W}^{(i,l)}$ for $2 \leq l \leq L-1$. This removes $\mathbf{s}^{(i,l)}$ from θ for $l \leq L-2$. Similarly for tree networks, for $\mathbf{u}$ of length $1 \leq l \leq L-2$ and $i \in [m_{L-l-1}]$, let $\alpha^{(\mathbf{u}\oplus i)'} = \alpha^{(\mathbf{u}\oplus i)} s^{(\mathbf{u}\oplus i)} \mathbf{w}^{(\mathbf{u})}$ (note $\mathbf{w}^{(\mathbf{u})} \in \mathbb{R}$). This removes $\mathbf{w}^{(\mathbf{u})}$ and $s^{(\mathbf{u}\oplus i)}$ from θ. ■

Henceforth, tree networks are assumed to have sign-determined activation and sign-determined networks are assumed to be simplified.

Let $\mathcal{D} = \{L - \tilde{L} + 1, \cdots, L\}$. By Lemma H.3, the training problem's regularization is

$$r(\theta) = \sum_{i=1}^{m_L} \sum_{l \in \mathcal{D}} \left(r^{(i,l)}\left(\theta_w^{(i,l)}\right)\right)^{\tilde{L}} \tag{H.3}$$

where $\theta^{(i,L)} = \alpha_i$ and $\theta^{(i,l)} = \mathbf{W}^{(i,l)}$ for $l < L$ in parallel networks with homogeneous σ, $\theta^{(i,L)} = \alpha_i$ and $\theta^{(i,L-1)} = s^{(i,L-1)}$ in parallel networks with sign-determined σ, and $\theta^{(i,L)} = \alpha^{(i)}$ and $\theta^{(i,L-1)} = \mathbf{s}^{(i)}$ in tree networks. Similar to the parallel network in Subsection 2.2, extend the standard (2.1) and tree (C.1) network definitions row-wise to the cases where the input is $\mathbf{X} \in \mathbb{R}^{N \times d}$.

Lemma H.4. *Let* $\tilde{\mathbf{X}}^{(i)} \in \mathbb{R}^N$ *be* $\hat{\mathbf{X}}^{(i,L)}$ *for a parallel network or* $\sigma\left(\mathbf{X}^{(i)} + b^{(i)}\mathbf{1}\right)$ *for a tree network, where the input is* $\mathbf{X} \in \mathbb{R}^{N \times d}$. *The training problem is equivalent to*

$$\min_{\theta \in \Theta : r_{(i,l)}\left(\theta^{(i,l)}\right) = 1, l \in \mathcal{D} - \{L\}} \mathcal{L}_{\mathbf{y}}\left(\sum_{i=1}^{m_L} \alpha_i \tilde{\mathbf{X}}^{(i)}\right) + \beta \sum_{i=1}^{m_L} r_{(i,L)}(\alpha_i) \tag{H.4}$$

Proof. By the AM-GM inequality on (H.3), a lower bound on the training problem is

$$\min_{\theta\in\Theta} \mathcal{L}_y\left(f_L\left(\mathbf{X};\theta\right)\right)+\beta\sum_{i=1}^{m_L}\prod_{l\in\mathcal{D}} r_{(i,l)}\left(\theta^{(i,l)}\right). \tag{H.5}$$

Consider the minimization problem

$$\min_{\theta\in\Theta: r_{(i,l)}\left(\theta^{(i,l)}\right)=1, l\in\mathcal{D}-\{L\}} \mathcal{L}_y\left(f_L\left(\mathbf{X};\theta\right)\right)+\beta\sum_{i=1}^{m_L}\prod_{l\in\mathcal{D}} r_{(i,l)}\left(\theta^{(i,l)}\right) \tag{H.6}$$

Problem (H.6) is an upper bound on (H.5). Given optimal $\left\{\theta^{(i,l)}\right\}$ in (H.5), the rescaled parameters $\theta^{(i,l)\prime}=\theta^{(i,l)}/r_{(i,l)}\left(\theta^{(i,l)}\right)$ for $l\in\mathcal{D}-\{L\}$ and $\theta^{(i,L)\prime}=\theta^{(i,L)}\prod_{l\in\mathcal{D}} r^{(i,l)}\left(\theta^{(i,l)}\right)$ (and rescaled bias parameters) achieve the same objective in (H.6). Hence (H.6) and (H.5) are equivalent. Given optimal $\left\{\theta^{(i,l)}\right\}$ in (H.6), the rescaled parameters $\theta^{(i,l)\prime}=|\theta^{(i,L)}|^{\frac{1}{L}}\theta^{(i,l)}$ (and rescaled bias parameters) achieve the same objective in the training problem, which is therefore equivalent to (H.6). Simplifying (H.6) gives (H.4). ∎

Lemma H.4 applies to networks without any weight constraints, i.e., it excludes 3-layer symmetrized networks. A L-layer *symmetrized network* is a parallel network with homogeneous activation such that $m_{L-3}=1$ and the elements of $\mathbf{W}^{(i,l)}$ have the same magnitude for $l\in\{L-2,L-1\}$. In a symmetrized network, the last two layer's weights are vectors: $\mathbf{W}^{(i,L-2)}\in\mathbb{R}^{1\times m_{L-2}}$ and $\mathbf{W}^{(i,L-1)}\in\mathbb{R}^{m_{L-2}}$. The constraint on the weight magnitudes is encoded in Θ. A 3-layer symmetrized network and a deep narrow network are special cases of a symmetrized network.

Lemma H.5. *Let* $r^{(i,l)}\left(\mathbf{W}^{(i,l)}\right)=\left\|\mathbf{W}^{(i,l)}\right\|_2$ *for* $l\in\{L-2,L-1\}$. *The rescaled problem for a symmetrized network is equivalent to*

$$\min_{\theta\in\Theta: r^{(i,l)}\left(\mathbf{W}^{(i,l)}\right)=1 \text{ for } l\in[L-3]; \left|\mathbf{W}_j^{(i,l)}\right|=1 \text{ for } l\in\{L-2,L-1\}, j\in[m_{L-3}]} \mathcal{L}_{\mathbf{y}}\left(f_L\left(\mathbf{X};\theta\right)\right)+\tilde{\beta}\sum_{i=1}^{m_L} r_{(i,L)}\left(\alpha_i\right) \tag{H.7}$$

where $\tilde{\beta}=\frac{\beta}{m_{L-2}}$.

Proof. Since $m_{L-3}=1$, we have $\mathbf{W}^{(i,L-2)}\in\mathbb{R}^{1\times m_{L-2}}$ and $\mathbf{W}^{(i,L-1)}\in\mathbb{R}^{m_{L-2}}$. The constraint states that for $l\in\{L-2,L-1\}$, $\left|\mathbf{W}_1^{(i,l)}\right|=\cdots=\left|\mathbf{W}_{m_{L-2}}^{(i,l)}\right|$. For $l\in\{L-2,L-1\}$, given $\mathbf{W}^{(i,l)},\mathbf{b}^{(i,l)}$ and α_i in apply a change of variables $\mathbf{W}^{(i,l)\prime}=\sqrt{m_{L-2}}\mathbf{W}^{(i,l)}$ and $\mathbf{b}^{(i,l)\prime}=\sqrt{m_{L-2}}^{\,l+3-L}\mathbf{b}^{(i,l)}$ and $\alpha_i'=\frac{1}{m_{L-2}}\alpha_i$ to the parameters in (H.4) to arrive at (H.7). ∎

Henceforth, assume $r_{(i,L)}\left(\alpha_i\right)=|\alpha_i|$.

Definition H.6. *Define the* rescaled training problem *as*

$$\min_{\theta\in\Theta}\mathcal{L}_{\mathbf{y}}\left(\sum_{i=1}^{m_L}\alpha_i\tilde{\mathbf{X}}^{(i)}\right)+\beta\sum_{i=1}^{m_L}|\alpha_i|. \tag{H.8}$$

If the network is symmetrized, $\tilde{\beta} = \frac{\beta}{m_{L-2}}$ *and* Θ *includes the constraints that* $r^{(i,l)}\left(\mathbf{W}^{(i,l)}\right) = 1$ *for* $l \in [L-3]$ *and* $\left|\mathbf{W}_j^{(i,l)}\right| = 1$ *for* $l \in \{L-2, L-1\}$*. Otherwise,* $\tilde{\beta} = \beta$ *and* $r_{(i,l)}\left(\theta^{(i,l)}\right) = 1$ *for* $l \in \mathcal{D} - \{L\}$*.* $\tilde{\mathbf{X}}^{(i)}$ *is defined as in Lemma H.4.*

For 3-layer networks with l_2 regularization ($r^{(i,l)}\left(\mathbf{W}^{(i,l)}\right) = \|\mathbf{W}^{(i,l)}\|_2$), Θ constrains the absolute value of all elements of all inner layer weights to be 1 in the rescaled training problem. The rescaled training problem is equivalent to both symmetrized and non-symmetrized networks.

Lemma H.7. *The rescaled training problem (H.8) is equivalent to the training problem for all the architectures and activations discussed above.*

Proof. Follows from Lemma H.4 and Lemma H.5. ∎

Lemma H.8. *A lower bound on the rescaled training problem is the dual problem*

$$\max_{\lambda \in \mathbb{R}^N} -\mathcal{L}_{\mathbf{y}}^*(\lambda) \quad s.t. \quad \max_{\theta \in \Theta} \left|\lambda^T \tilde{\mathbf{X}}\right| \leq \tilde{\beta}, \tag{H.9}$$

where $\tilde{\mathbf{X}} = \tilde{\mathbf{X}}^{(1)}$ *and* $f^*(\mathbf{x}) := \max_{\mathbf{x}} \left\{\mathbf{z}^T\mathbf{x} - f(\mathbf{x})\right\}$ *is the convex conjugate of* f*.*

Proof. Find the dual of (H.4), by rewriting (H.4) as

$$\min_{\theta \in \Theta} \mathcal{L}_{\mathbf{y}}(\mathbf{z}) + \tilde{\beta}||\boldsymbol{\alpha}||_1, \quad \text{s.t.} \quad \mathbf{z} = \sum_{i=1}^{m_L} \alpha_i \tilde{\mathbf{X}}^{(i)}. \tag{H.10}$$

The Lagrangian of problem (H.10) is $L(\lambda, \theta) = \mathcal{L}_{\mathbf{y}}(\mathbf{z}) + \tilde{\beta}||\boldsymbol{\alpha}||_1 - \lambda^T\mathbf{z} + \sum_{i=1}^{m_L} \lambda^T \tilde{\mathbf{X}}^{(i)} \alpha_i$. Minimize the Lagrangian over $\mathbf{z}$ and $\boldsymbol{\alpha}$ and use Fenchel duality (7). The dual of (H.10) is

$$\max_{\lambda \in \mathbb{R}^N} -\mathcal{L}_{\mathbf{y}}^*(\lambda) \quad \text{s.t.} \quad \max_{\theta \in \Theta} \left|\lambda^T \tilde{\mathbf{X}}^{(i)}\right| \leq \tilde{\beta}, i \in [m_L]. \tag{H.11}$$

In the tree and parallel nets, $\tilde{\mathbf{X}}^{(i)}$ is of the same form for all $i \in [m_L]$. So the m_L constraints in (H.11) collapse to a single constraint. Then we can write (H.11) as (H.9). ∎

For a parallel network, the dual problem (H.9) is

$$\max_{\lambda \in \mathbb{R}^N} -\mathcal{L}_{\mathbf{y}}^*(\lambda) \quad \text{s.t.} \quad \max_{\theta \in \Theta} \left|\lambda^T \hat{\mathbf{X}}^{(L)}\right| \leq \tilde{\beta}, \tag{H.12}$$

where $\hat{\mathbf{X}}^{(1)} = \mathbf{X}$. Henceforth, all regularizations are l_2-norm: e.g., for a parallel network, $r^{(i,l)}\left(\mathbf{W}^{(i,l)}\right) = \left\|\mathbf{W}^{(i,l)}\right\|_2$, denoting the square root of sum of squares of $\mathbf{W}^{(i,l)}$'s elements.

Deep narrow and symmetrized networks with $d=1$. In this section, we assume the data is 1-D and find the maximizers of $\left|\lambda^T \hat{\mathbf{X}}^{(L)}\right|$ in the dual constraint (H.12). To this end, assume the elements of $\mathbf{W}^{(l)}$ are ± 1. Note that $\mathbf{b}^{(L-1)}$ is a scalar and $\mathbf{X}^{(l+1)} = \sigma\left(\mathbf{X}^{(l)}\mathbf{W}^{(l)} + \mathbf{b}^{(l)}\mathbf{1}\right) \in \mathbb{R}^N$. The next remark refers to the leaky ReLU slopes a^+ and a^- defined in Section 2. Let $\mathcal{K}(f)$ and $\mathcal{Z}(f)$ be the sets of breakpoints and zeros of a function f, respectively.

Remark H.9. *Let* $b{=}\mathbf{b}^{(L-1)}$, $a_n{=}\hat{\mathbf{X}}_n^{(L-1)}\mathbf{W}^{(L-1)}$, $g_n(b){=}\sigma\left(a_n{+}b\right)$, *and* $g(b){=}\sum_{n=1}^N \lambda_n g_n(b){=}\lambda^T\mathbf{X}^{(L)}$. *Let* $\mathcal{I}{=}\bigcup_{n=1}^N \mathcal{K}\left(g_n\right) \supset \mathcal{K}(g)$. *Then* $g(b){=}\sum_{n=1}^N \lambda_n a(a_n{+}b){=}ab\sum_{n=1}^N \lambda_n{+}\sum_{n=1}^N \lambda_n a_n a$ *for* b *large enough (with* $a{=}a^+$*) and for* b *small enough (with* $a{=}a^-$*). So* $a^-{=}a^+{=}0$ *or* $\sum_{n=1}^N \lambda_n{=}0$ *if and only if* g *is bounded, if and only if* g *has a (finite) maximizer and minimizer. In this case, assuming* g *is not a constant function,* $\mathcal{I}$ *contains a maximizer and minimizer of* g.

Henceforth, assume $\lambda^T\mathbf{1}{=}0$ if $a^-{\neq}0$ or $a^+{\neq}0$. Suppose $\mathbf{b}^{(l)}, \cdots, \mathbf{b}^{(L-1)}$ are scalar-valued. We call $\mathbf{b}^{(l)}{=}\mathbf{b}^{(l)^*}$ *optimal* if $\mathbf{b}^{(l)^*}{\in}\arg\max_{\mathbf{b}^{(l)}}\max_{\mathbf{b}^{(l+1)}}\cdots\max_{\mathbf{b}^{(L-1)}}\left|\sum_{n=1}^N \lambda_n \hat{\mathbf{X}}_n^{(L)}\right|$. The next result refers to the normalized midpoint defined in (3.1).

Lemma H.10. *Let* σ *be leaky ReLU. Let* $\alpha, \beta \in \mathbb{R}$ *with* $\alpha{\neq}\beta$. *Let* $f(x){=}\sigma\left(x{+}\alpha\right){-}\sigma\left(x{+}\beta\right)$. *For* $s \in \{-,+\}$, *let* $g_s(x){=}\sigma(sx)$. *Then, if* σ *is monotone,*

$$\mathcal{K}\left(g_s(f)\right) \subset \{-\alpha,-\beta\}. \tag{H.13}$$

Otherwise if σ *is not monotone,*

$$\mathcal{K}(g_s(f)) \subset \{-\alpha,-\beta, m_{\alpha,\beta}\}. \tag{H.14}$$

Proof. Since $f_{\alpha,\beta}$ is piecewise linear,

$$\mathcal{K}\left(g_s(f)\right) \subset \mathcal{K}(f) \cup \mathcal{Z}(f). \tag{H.15}$$

Moreover if $f(x)$ has the same sign for all x, then observe that

$$\mathcal{K}\left(g_s(f)\right){=}\mathcal{K}(f). \tag{H.16}$$

Let $\mathcal{G}(f) = \{(x, f(x)) : x \in \mathcal{K}\left(f\right)\}$. We find that

$$\mathcal{G}(f) = \left\{\left(-\max\{\alpha,\beta\}, a^-(\alpha{-}\beta)\right), \left(-\min\{\alpha,\beta\}, a^+(\alpha{-}\beta)\right)\right\}. \tag{H.17}$$

In particular, $\mathcal{K}(f){=}\{-\alpha,-\beta\}$. Observe that f has constant value (an in particular, sign) beyond its breakpoints. If σ is monotone, then $\mathrm{sign}(a^+){=}\mathrm{sign}(a^-)$, so (H.17) implies $\mathrm{sign}\left(f(x)\right){=}\mathrm{sign}\left(a^+(\alpha{-}\beta)\right)$ for all x, so (H.16) implies (H.13). If σ is not monotone, linearly interpolating between the points in $\mathcal{G}\left(f\right)$ (H.17) shows that f changes sign exactly when $x{=}m_{\alpha,\beta}$, so (H.15) implies (H.14). ■

Lemma H.11. *Let* σ *be piecewise linear if* $L{=}2$ *and leaky ReLU otherwise. Let* $l{\in}\{L{-}1, L{-}2\}$*. Consider the bias* $\mathbf{b}^{(l)}$. *Let* $m_{L-2}{=}1$. *If* σ *is monotone there is either a data feature bias that is optimal. If* σ *is not monotone there is either a data feature bias or a midpoint feature bias* $b^{(L-2)}{=}m_{\mathbf{X}^{(L-2)}_{n(L-2)},\mathbf{X}^{(L-2)}_{n(L-1)}}\mathbf{W}^{(L-2)}$ *(if* $l{=}L{-}2$*) that is optimal.*

Proof. Remark H.9 implies that $\bigcup_{n=1}^N \mathcal{K}\left(\hat{\mathbf{X}}_n^{(L)}\right)$ contains an optimal $\mathbf{b}^{(L-1)}$. The breakpoint of $\hat{\mathbf{X}}_n^{(L)}{=}\sigma\left(\mathbf{X}_n^{(L-1)}\mathbf{W}^{(L-1)}{+}\mathbf{b}^{(L-1)}\right)$ as a function of $\mathbf{b}^{(L-1)}$ is $\mathbf{b}^{(L-1)}{=}{-}\mathbf{X}_n^{(L-1)}\mathbf{W}^{(L-1)}$.

Therefore for some $n^{(L-1)}{\in}[N]$, the data feature bias $\mathbf{b}^{(L-1)}{=}{-}\mathbf{X}^{(L-1)}_{n^{(L-1)}}\mathbf{W}^{(L-1)}$ is optimal. Plugging in this optimal $\mathbf{b}^{(L-1)}$ back into $\hat{\mathbf{X}}^{(L)}_n$ gives

$$\lambda^T\mathbf{X}^{(L)}{=}\sum_{n=1}^{N}\lambda_n\sigma\left(\left(\hat{\mathbf{X}}^{(L-1)}_n{-}\hat{\mathbf{X}}^{(L-1)}_{n^{(L-1)}}\right)\mathbf{W}^{(L-1)}\right).$$

For $L>2$, expanding $\hat{\mathbf{X}}^{(L-1)}$ gives

(H.18)
$$\lambda^T\mathbf{X}^{(L)}{=}\sum_{n=1}^{N}\lambda_n\sigma\left(\left(\sigma\left(\hat{\mathbf{X}}^{(L-2)}_n\mathbf{W}^{(L-2)}{+}\mathbf{b}^{(L-2)}\right)-\sigma\left(\hat{\mathbf{X}}^{(L-2)}_{n^{(L-1)}}\mathbf{W}^{(L-2)}{+}\mathbf{b}^{(L-2)}\right)\right)\mathbf{W}^{(L-1)}\right).$$

Since $m_{L-2}{=}1$, we have $\mathbf{b}^{(L-2)}{\in}\mathbb{R}$. Applying Lemma H.10 with $x{=}\mathbf{b}^{(L-2)}, \alpha{=}\hat{\mathbf{X}}^{(L-2)}_n\mathbf{W}^{(L-2)}$, $\beta{=}\hat{\mathbf{X}}^{(L-1)}_n\mathbf{W}^{(L-2)}, s{=}\mathbf{W}^{(L-1)}$ and a similar argument as Remark H.9 gives the result. ∎

Lemma H.12. *Let $L\geq 2$. Consider a deep narrow ReLU network with L layers. For every $l\in\{2,\cdots,L\}$, a data feature bias is optimal for layer $L{-}l{+}1$, and with this optimal bias, if $l{<}L$, there exists $N_1,N_2{\in}[N]$ such that for all $n{\in}[N]$, either $\hat{\mathbf{X}}^{(L)}_n{=}0$ or*

(H.19)
$$\hat{\mathbf{X}}^{(L)}_n=\pm\left(\sigma\left(\hat{\mathbf{X}}^{(L-l)}_{N_1}\mathbf{W}^{(L-l)}{+}\mathbf{b}^{(L-l)}\right){-}\sigma\left(\hat{\mathbf{X}}^{(L-l)}_{N_2}{+}\mathbf{b}^{(L-l)}\right)\right).$$

Proof. We prove by induction on l.

Base case: suppose $l=2$. Then the claim holds by (H.18) in the proof of Lemma H.11.

Now, suppose the claim holds for $l{=}k{\in}\{2,\cdots,L{-}1\}$. Applying argument similar to the proof of Lemma H.11 to (H.19) shows that a data feature bias $\mathbf{b}^{(L-k)}{=}{-}\hat{\mathbf{X}}^{(L-k)}_{n^{(L-k)}}\mathbf{W}^{(L-k)}$ is optimal for $\mathbf{b}^{(L-l+1)}$ when $l{=}k{+}1$. Plugging in this optimal $\mathbf{b}^{(L-k)}$ into $\hat{\mathbf{X}}^{(L)}$ in (H.19) for $l{=}k$ and expanding $\hat{\mathbf{X}}^{(L-k)}{=}\sigma\left(\hat{\mathbf{X}}^{(L-k-1)}\mathbf{W}^{(L-k-1)}{+}\mathbf{b}^{(L-k-1)}\right)$ shows that for some $s^{(1)},s^{(2)}{\in}\{-1,1\}$, either $\hat{\mathbf{X}}^{(L)}_n{=}0$ or

$$\begin{aligned}\hat{\mathbf{X}}^{(L)}_n &{=}\sigma\Bigg(s^{(1)}\Bigg(\sigma\left(s^{(2)}\left(\sigma\left(\hat{\mathbf{X}}^{(L-k-1)}_{n'}\mathbf{W}^{(L-k-1)}{+}\mathbf{b}^{(L-k-1)}\right){-}\sigma\left(\hat{\mathbf{X}}^{(L-k-1)}_{n^{(L-2)}}\mathbf{W}^{(L-3)}{+}\mathbf{b}^{(L-k-1)}\right)\right)\right)\\ &\quad{-}\sigma\left(s^{(2)}\left(\sigma\left(\hat{\mathbf{X}}^{(L-k-1)}_{n^{(L-1)}}\mathbf{W}^{(L-k-1)}{+}\mathbf{b}^{(L-k-1)}\right){-}\sigma\left(\hat{\mathbf{X}}^{(L-k-1)}_{n^{(L-2)}}\mathbf{W}^{(L-k-1)}{+}\mathbf{b}^{(L-k-1)}\right)\right)\right)\Bigg)\Bigg)\\ &\in\Bigg\{s^{(2)}\left(\sigma\left(\hat{\mathbf{X}}^{(L-k-1)}_{n'}\mathbf{W}^{(L-k-1)}+\mathbf{b}^{(L-k-1)}\right)-\sigma\left(\hat{\mathbf{X}}^{(L-k-1)}_{n^{(L-2)}}+\mathbf{b}^{(L-k-1)}\right)\right),\\ &\quad -s^{(2)}\left(\sigma\left(\hat{\mathbf{X}}^{(L-k-1)}_{n^{(L-1)}}\mathbf{W}^{(L-k-1)}+\mathbf{b}^{(L-k-1)}\right)-\sigma\left(\hat{\mathbf{X}}^{(L-k-1)}_{n^{(L-2)}}+\mathbf{b}^{(L-k-1)}\right)\right),\\ &\quad s^{(2)}\left(\sigma\left(\hat{\mathbf{X}}^{(L-k-1)}_{n'}\mathbf{W}^{(L-k-1)}+\mathbf{b}^{(L-k-1)}\right)-\sigma\left(\hat{\mathbf{X}}^{(L-k-1)}_{n^{(L-1)}}+\mathbf{b}^{(L-k-1)}\right)\right),0\Bigg\}.\end{aligned}$$

So (H.19) holds for $l{=}k{+}1$. Finally if (H.19) holds for $l{=}L{-}1$ then by a similar argument as the proof of Lemma H.11, a data feature bias for $\mathbf{b}^{(L-l+1)}$ is optimal when $l{=}k{+}1{=}L$. By induction, the result holds. ∎

Lemma H.13. *Let $L\geq 2$. Consider a deep narrow ReLU network with L layers. For $l\in[L-1]$, suppose $\mathbf{b}^{(l)}$ is a data feature. For every $l{\in}\{2,\cdots,L{-}1\}$, there exist $N_1,N_2{\in}[N]$ such that for all x,*

$$\hat{\mathbf{X}}^{(l)}(x) \in \left\{ ReLU^{\pm}_{x_{N_1}}(x), Ramp^{\pm}_{x_{N_1},x_{N_2}}(x) \right\}. \tag{H.20}$$

Proof. We prove by induction on l. Base case: for $l=2$, $\hat{\mathbf{X}}^{(2)}(x) = \sigma\left(x\mathbf{W}^{(1)} + \mathbf{b}^{(1)}\right) = \sigma\left(\left(x - x_{n^{(1)}}\right)\mathbf{W}^{(1)}\right) = \text{ReLU}^{\mathbf{W}^{(1)}}_{x_{n^{(1)}}}(x)$. Now, suppose the claim holds for $l = 1, \cdots, k < L-1$. Then $\hat{\mathbf{X}}^{(k+1)}(x) = \sigma\left(\hat{\mathbf{X}}^{(k)}\mathbf{W}^{(k)} + \mathbf{b}^{(k)}\right) = \sigma\left(\left(\hat{\mathbf{X}}^{(k)}_n - \hat{\mathbf{X}}^{(k)}_{n^{(k)}}\right)\mathbf{W}^{(k)}\right)$ so either there exist $N_1', N_2' \in \left\{N_1, n^{(k)}\right\}, s \in \{+,-\}$ such that for all x, $\hat{\mathbf{X}}^{(k+1)}(x) = \sigma\left(\left(\text{ReLU}^{\pm}_{x_{N_1}}(x) - \text{ReLU}^{\pm}_{x_{N_1}}\left(x_{n^{(k)}}\right)\right)\mathbf{W}^{(k)}\right) \in \left\{\text{ReLU}^{s}_{x_{N_1'}}(x), \text{Ramp}^{s}_{x_{N_1'}, x_{N_1'}}(x)\right\}$, or there exist $N_1', N_2' \in \left\{N_1, N_2, n^{(k)}\right\}, s \in \{+,-\}$ such that for all x, $\hat{\mathbf{X}}^{(k+1)}(x) = \sigma\left(\left(\text{Ramp}^{\pm}_{x_{N_1},x_{N_2}}(x) - \text{Ramp}^{\pm}_{x_{N_1},x_{N_2}}\left(x_{n^{(k)}}\right)\right)\mathbf{W}^{(k)}\right) = \text{Ramp}^{s}_{x_{N_1'},x_{N_2'}}(x)$. By induction, the result holds. ∎

Lemma H.14. *Consider a deep narrow network. Let $L \geq 2$. For all $l \in [L-1]$, there exist $n^{(L-l)} \in [N]$ and a l-tuple $(a_0, a_1, \cdots, a_l)$ such that for all $l \in [L-1]$,*

$$\mathbf{b}^{(L-l)^*} = -\sum_{i=1}^{l} a_i \hat{\mathbf{X}}^{(L-l)}_{n^{(L-i)}} \mathbf{W}^{(L-l)} \tag{H.21}$$

is optimal. Note that optimal $\mathbf{b}^{(1)}, \cdots, \mathbf{b}^{(L-1)}$ can be found sequentially, by computing $\mathbf{b}^{(l)^*}$ as a function of $\mathbf{b}^{(l-1)^*}$ and so on. Moreover, there are $O\left(2^L L!\right)$ options for $\mathbf{b}^{(1)^*}$. Additionally, for all $l \in [L-1]$, under such optimal $\mathbf{b}^{(1)}, \cdots, \mathbf{b}^{(l-1)}$, for all $n \in [N]$,

$$\hat{\mathbf{X}}^{(L-l+1)}_n = \sigma\left(\hat{\mathbf{X}}^{(L-l)'}_n \mathbf{W}^{(L-l)}\right) \tag{H.22}$$

where

$$\hat{\mathbf{X}}^{(L-l)'}_n = a_0 \hat{\mathbf{X}}^{(L-l)}_n + \sum_{i=1}^{l} a_i \hat{\mathbf{X}}^{(L-l)}_{n^{(L-i)}}. \tag{H.23}$$

Proof. We prove (H.21), (H.22) and (H.23) by strong induction on l.

Base case: suppose $l=1$. Then (H.21), (H.22), (H.23) hold by Lemma H.11 and its proof.

Now, suppose (H.21), (H.22) and (H.23) hold for $l = 1, \cdots, k < L-1$. By Equation (H.22) for $l=k$, $\hat{\mathbf{X}}^{(L-k+1)}_n(x)$ is locally a linear combination of the $k+1$ terms $\hat{\mathbf{X}}^{(L-k+1)}_{n^{(L-k-1)}}, \hat{\mathbf{X}}^{(L-k+1)}_{n^{(L-1)}}, \cdots, \hat{\mathbf{X}}^{(L-k+1)}_{n^{(L-k)}}$, and hence so is $\hat{\mathbf{X}}^{(L)}$. So the breakpoints of $\hat{\mathbf{X}}^{(L)}$ as a function of $\mathbf{b}^{(L-(k+1))}$ are linear combinations of the $k+1$ terms, which proves (H.21) for $l=k+1$. Plugging in this breakpoint $\mathbf{b}^{(L-k-1)}$ into $\hat{\mathbf{X}}^{(L-k)}_n$ proves (H.22) and (H.23) for $l=k+1$. Therefore (H.21), (H.22) and (H.23) hold for all $l \in [L-1]$.

Now we prove that the number of options for $\mathbf{b}^{(1)}$ is $O\left(2^L L!\right)$. By Equation (H.23) for $l=L-1$, as a function of $\mathbf{b}^{(1)}$, $\hat{\mathbf{X}}^{(L-1)'}_j$ has breakpoints at $-\hat{\mathbf{X}}^{(L-l-1)}_i \mathbf{W}^{(L-l-1)}$ for $i \in \{n, n^{(L-1)},$

$\cdots, n^{(L-l)}\}$, which totals to $l+1=L$ breakpoints. A piecewise linear function f has up to plus or minus one as many zeros as breakpoints, and the zeros become breakpoints in $\sigma(f)$. So by Equation (H.22), $\hat{\mathbf{X}}_n^{(L-l+1)}$ as a function of $\mathbf{b}^{(1)}$ has breakpoints at $O(2L)$ places. Then by Equation (H.23) for $l=L-2$, $\hat{\mathbf{X}}_n^{(L-l+1)'}$ as a function of $\mathbf{b}^{(1)}$ has breakpoints at $O(2(L-1)(L))$ places. And by Equation (H.22) for $l=L-2$, $\hat{\mathbf{X}}_n^{(L-k+2)}$ as a function of $\mathbf{b}^{(1)}$ has breakpoints at $O\left(2^2 L(L-1)\right)$ places. By repeating this argument for $l=L-1,\cdots,1$, $\hat{\mathbf{X}}_n^{(L)}$ as a function of $\mathbf{b}^{(1)}$ has breakpoints at $O\left(2^L L!\right)$ places. ■

Recall the deep library defined in Definition 3.11. Note that any $\mathbf{a}$ in the deep library is of the form $\mathbf{a} = \hat{\mathbf{X}}^{(L)}(\mathbf{X})$, with $\mathbf{a}_n = \hat{\mathbf{X}}^{(L)}(x_n)$.

Lemma H.15. *For $L \in \{2,3\}$ the maximization constraint in (H.12) is equivalent to: for all vectors $\mathbf{a}$ in the deep library,*

$$\begin{aligned} \left|\lambda^T \mathbf{a}\right| &\leq \tilde{\beta} \\ \mathbf{1}^T\lambda &= 0 \text{ if } a^+ \neq 0 \text{ or } a^- \neq 0. \end{aligned} \tag{H.24}$$

Proof. Lemma H.11 gives (H.24). Now, if the activation is symmetric, then $\hat{\mathbf{X}}^{(L)}$ is invariant under the sign of the components of $\mathbf{W}^{(1)}$. Next, recall $x_1 > \cdots > x_N$ and $\operatorname{sign}(0)=1$. If the activation is sign, then for all $n \in [N-1]$, with $\mathbf{W}^{(1)} = 1$ and $\mathbf{b}^{(1)} = -x_n$ we have $\hat{\mathbf{X}}^{(L=2)}(\mathbf{X}) = \sigma(\mathbf{X} - x_n) = \left(\mathbf{1}_{1:n}^T, -\mathbf{1}_{n+1:N}^T\right) = -\sigma(x_{n+1} - \mathbf{X})$; while for $\mathbf{W}^{(1)} = -1$ and $\mathbf{b}^{(1)} = x_{n+1}$ we have $\hat{\mathbf{X}}^{(L=2)}(\mathbf{X}) = \sigma(x_{n+1} - \mathbf{X})$. And for $\mathbf{W}^{(1)} = 1$ and $\mathbf{b}^{(1)} = -x_N$, we have $\hat{\mathbf{X}}^{(2)}(\mathbf{X}) = \mathbf{1}$ while for $\mathbf{W}^{(1)} = -1$ and $\mathbf{b}^{(1)} = x_1$ we have $\hat{\mathbf{X}}^{(2)}(\mathbf{X}) = \mathbf{1}$. So for $L=2$ with symmetric or sign activation, (H.24) is unchanged if $\mathbf{W}^{(1)} \in \{-1,1\}$ is restricted to be 1. ■

Lemma H.16. *Let $\mathbf{A}$ be a matrix whose set of columns is the deep library. Replace the maximization constraint in (H.12) with (H.24). The dual of (H.12) then is*

$$\min_{\mathbf{z},\xi\in\mathbb{R}} \mathcal{L}_{\mathbf{y}}(\mathbf{A}\mathbf{z} + \xi\mathbf{1}) + \tilde{\beta}\|\mathbf{z}\|_1, \quad \textit{where} \quad \xi = 0 \textit{ if } c_1 = c_2 = 0. \tag{H.25}$$

Proof. Problem (H.12) can be written as

$$-\min_{\lambda\in\mathbb{R}^N} \mathcal{L}_{\mathbf{y}}^*(\lambda) \quad \text{s.t.} \quad \lambda^T\mathbf{1} = 0 \text{ if } a^- \neq 0 \text{ or } a^+ \neq 0, \text{ and } |\lambda^T \mathbf{A}| \leq \tilde{\beta}\mathbf{1}^T. \tag{H.26}$$

The Lagrangian of the negation of (H.26) with bidual variables $\mathbf{z}, \xi$ is

$$L(\lambda, \mathbf{z}, \xi) = \mathcal{L}_{\mathbf{y}}^*(\lambda) - \lambda^T(\mathbf{A}\mathbf{z} + \xi\mathbf{1}) - \tilde{\beta}\|\mathbf{z}\|_1, \text{ where } \xi = 0 \text{ if } a^- = a^+ = 0. \tag{H.27}$$

Equation (H.27) holds because the constraint $|\lambda^T\mathbf{A}| \leq \tilde{\beta}\mathbf{1}^T$ i.e., $\lambda^T\mathbf{A} - \tilde{\beta}\mathbf{1}^T \leq \mathbf{0}^T, -\lambda^T\mathbf{A} - \tilde{\beta}\mathbf{1}^T \leq \mathbf{0}^T$, appears in the Lagrangian as $\lambda^T\mathbf{A}\left(\mathbf{z}^{(1)} - \mathbf{z}^{(2)}\right) - \tilde{\beta}\mathbf{1}^T\left(\mathbf{z}^{(1)} + \mathbf{z}^{(2)}\right)$ with bidual variables $\mathbf{z}^{(1)}, \mathbf{z}^{(2)}$, which are combined into one bidual variable $\mathbf{z} = \mathbf{z}^{(1)} - \mathbf{z}^{(2)}$. This makes $\mathbf{z}^{(1)} + \mathbf{z}^{(2)} = \|\mathbf{z}\|_1$. Changing variables $\mathbf{z}' = -\mathbf{z}, \xi' = -\xi$ gives (H.27). Since $\mathcal{L}^{**} = \mathcal{L}$ (6), $\inf_\lambda L(\lambda, \mathbf{z}, \xi) = -\mathcal{L}_y - \tilde{\beta}\|\mathbf{z}\|_1$ and negating its maximization gives (H.25). ■

Definition H.17 (Parameter unscaling). *Let $\gamma_i = |\alpha_i|^{\frac{1}{L}}$. For ReLU, absolute value, or leaky ReLU, parameter unscaling is the following transformation. For symmetrized networks, first change variables as $\mathbf{W}^{(i,l)\prime} = \frac{1}{\sqrt{2}}\mathbf{W}^{(i,l)}$ and $\mathbf{b}^{(i,l)\prime} = \frac{1}{\sqrt{2}^l}\mathbf{b}^{(i,l)}$ for $l \in [2]$, $\alpha_i' = 2\alpha_i$. Then, for all architectures, change variables as $q' = \mathrm{sign}(q)\gamma_i$ for $q \in \theta_w^{(i)}$. For parallel networks, change variables as $\mathbf{b}^{(i,l)\prime} = \mathbf{b}^{(i,l)}(\gamma_i)^l$. For tree networks, change variables as $\mathbf{b}^{\mathbf{u}\prime} = \mathbf{b}^{\mathbf{u}}(\gamma_i)^l$ for $\mathbf{u} \in \mathcal{U}$ with length l such that $u_1 = i$. For sign and threshold activations, parameter unscaling is the transformation $\alpha_i' = \mathrm{sign}(\alpha_i)\sqrt{|\alpha_i|}$ and $s^{(i,L-1)\prime} = \sqrt{|\alpha_i|}$ for parallel nets, and $s^{(i)\prime} = \sqrt{|\alpha_i|}$ for tree nets.*

H.3. Proofs of results in Subsection 3.1 .

Proof of Lemma 3.8. The Lasso equivalence follows from a similar argument as the proof of Theorem 3.12. The Lasso features follow from Lemma H.12 and Lemma H.13 with $l = L$. ■

Proof of Theorem 3.7. By the proof of Lemma H.14 and a similar argument as Theorem 3.12, the training problem is equivalent to a Lasso problem. Moreover Lemma H.14 and choice of $n^{(1)}, \cdots, n^{(L-1)} \in [N]$ give the number of possible $\mathbf{b}^{(1)}$ for given $\mathbf{W}^{(1)}, \cdots, \mathbf{W}^{(L-1)}$ as $O\left(N^{L-1}2^L L!\right)$. Note that by (H.22) and (H.23), $\mathbf{b}^{(1)}$ determines all other $\mathbf{b}^{(l)}$. Finally, there are 2^L possibilities of $\mathbf{W}^{(l)} \in \{-1, 1\}$. The ReLU result follows from Lemma 3.8. ■

Proof of Theorem 3.12. By Lemma H.16, problem (H.25) is a lower bound on the training problem (1.1), where the Lasso features are all vectors in the deep library. Let $(\mathbf{z}^*, \xi^*)$ be a Lasso solution. From Definition 3.11, it can be seen that $\mathbf{A}\mathbf{z}^* + \xi^* = \sum_i z_i^* \mathbf{A}_i + \xi^* = \sum_i \alpha_i \hat{\mathbf{X}}^{(i,L)}(\mathbf{X}) + \xi^* = f_L(\mathbf{X}; \theta)$ and $\|\mathbf{z}^*\|_1 = \|\boldsymbol{\alpha}\|_1$. Therefore a reconstructed neural net achieves the same objective in the rescaled training problem, as $(\mathbf{z}^*, \xi^*)$ does in the Lasso objective. Parameter unscaling (Definition H.17) makes them achieve the same objective in the training problem (see Remark H.20). Therefore the Lasso problem in Theorem 3.12 is equivalent to the training problem. ■

Proof of Lemma B.4. By Theorem 3.12 and its proof, the reconstructed neural net achieves the same objective as the Lasso problem, which is equivalent to the training problem. So the reconstructed neural net is optimal. ■

Proof of Theorem 3.13. By considering each component of $\mathbf{b}^{(l)}$ separately in the proof of Lemma H.11, we see that optimal bias parameters for a symmetrized network can include data features. The Lasso equivalence follows from a similar argument as the proof of Theorem 3.12. ■

Proof of Theorem 3.5. Follows from Theorem 3.13 and evaluating the features. See proof of Figure 3.5 below. ■

Proof of Theorem 3.2. For $L \in \{2, 3\}$, apply Theorem 3.7, Theorem 3.12 where $\sigma(x) = |x|$ and Lemma B.2. For $L=4$, continue the same line of argument as Lemma H.11 by plugging in $\mathbf{b}^{(L-2)} = -\hat{\mathbf{X}}^{(L-2)}_{n^{(L-2)}}\mathbf{W}^{(L-2)}$ into (H.18), expanding $\hat{\mathbf{X}}^{(L-2)} = \sigma\left(\hat{\mathbf{X}}^{(L-3)}\mathbf{W}^{(L-3)} + \mathbf{b}^{(L-3)}\right)$, and observing that the breakpoints of $\lambda^T\hat{\mathbf{X}}^{(L)}$ as a function of $\mathbf{b}^{(L-3)}$ include data features, and hence these can be optimal bias parameters. Plugging in these data feature biases in to $\hat{\mathbf{X}}^{(4)}$ gives the features and reflections. Continue repeating a similar argument for $L > 4$. The reconstruction and Lasso equivalence follows from a similar argument as the proof of Theorem 3.12. ■

Remark H.18. *In the proof of Theorem 3.2 for $L \geq 4$, we only used one of the possible forms of an optimal $\mathbf{b}^{(L-2)}$ specified by Lemma H.11 for absolute value activation, and we only specified one of the possible forms of a breakpoint in $\lambda^T \hat{\mathbf{X}}^{(L)}$ as a function of $\mathbf{b}^{(L-3)}$. So a 4-layer network may have additional features not specified.*

Proof of Theorem 3.4. Follows from Lemma 3.8 and Theorem 3.12 where $\sigma(x) = (x)_+$ is monotone. ■

Proof of Corollary 3.15. Follows from Theorem 3.12 for $L = 2$. ■

Remark H.19. *For $b_1, b_2 \geq 0$,*

$$||x - b_1| - b_2| = \begin{cases} b_1 - b_2 - x & \text{if } x \leq b_1 - b_2 \\ x - (b_1 - b_2) & \text{if } b_1 - b_2 \leq x \leq b_1 \\ b_1 + b_2 - x & \text{if } b_1 \leq x \leq b_1 + b_2 \\ x - (b_1 + b_2) & \text{if } x \geq s_1 b_1 + b_2 \end{cases}$$
$$= \mathcal{W}_{b_1 - b_2, b_1}.$$

Proof of Lemma B.2, Figure 3.1, Figure 3.5. Follows from direct computation of Equation (2.1) and application of Remark H.19. In particular, for ReLU symmetrized networks, it can be verified that for $\mathbf{W}^{(1)} = (-1, 1), \mathbf{W}^{(2)} = (-1, -1)$; $\mathbf{W}^{(1)} = (-1, 1), \mathbf{W}^{(2)} = (1, 1)$; $\mathbf{W}^{(1)} = (1, -1), \mathbf{W}^{(2)} = (-1, -1)$; and $\mathbf{W}^{(1)} = (1, -1), \mathbf{W}^{(2)} = (1, 1)$, the deep library features can contain reflections. Note ReLU symmetrized features are consistent with Figure B.1. ■

Remark H.20. *For sign-determined activations, by Remark H.1, the inner weights can be unregularized. So, reconstructed parameters (as defined in Definition B.3) that are unscaled according to Definition H.17 achieve the same objective in the training problem as the optimal value of the rescaled problem.*

Proof of Lemma 3.1. Since $|x| = 2(x)_+ + x$, we have $\sum_{i=1}^{m_2} \left|\mathbf{X}\mathbf{W}^{(i,1)} + \mathbf{b}^{(i,1)}\right| \alpha_i + \mathbf{X}\omega + \xi$ $= \sum_{i=1}^{m_2} 2\left(\mathbf{X}\mathbf{W}^{(i,1)} + \mathbf{b}^{(i,1)}\right)_+ \alpha_i + \mathbf{X}\left(\omega - \sum_{i=1}^{m_2} \mathbf{W}^{(i,1)}\alpha_i\right) - \sum_{i=1}^{m_2} \mathbf{b}^{(i,1)}\alpha_i + \xi$. Given a solution ${\mathbf{W}^{(i,l)}}^*, {\mathbf{b}^{(i,l)}}^*, {\alpha_i}^*, \xi^*, \omega^*$ to the training problem for absolute value activation with skip connection, the parameters $\mathbf{W}^{(i,l)} = {\mathbf{W}^{(i,l)}}^*, {\mathbf{b}^{(i,l)}}^* = {\mathbf{b}^{(i,l)}}^*, \alpha_i = 2{\alpha_i}^*, \xi = \xi^* - \sum_{i=1}^{m_2} {\mathbf{b}^{(i,l)}}^* {\alpha_i}^*, \omega = \omega^*$ $- \sum_{i=1}^{m_2} {\mathbf{W}^{(i,1)}}^* {\alpha_i}^*$ achieve the same objective in the training problem for ReLU activation with skip connection. Conversely, since $(x)_+ = \frac{|x|+x}{2}$, we have $\sum_{i=1}^{m_2} \left(\mathbf{X}\mathbf{W}^{(i,1)} + \mathbf{b}^{(i,1)}\right)_+ \alpha_i + \mathbf{X}\omega + \xi$ $= \frac{1}{2}\sum_{i=1}^{m_2} \left|\mathbf{X}\mathbf{W}^{(i,1)} + \mathbf{b}^{(i,1)}\right| \alpha_i + \mathbf{X}\left(\omega + \frac{1}{2}\sum_{i=1}^{m_2} \mathbf{W}^{(i,1)}\alpha_i\right) + \frac{1}{2}\sum_{i=1}^{m_2} \mathbf{b}^{(i,1)}\alpha_i + \xi$. Given a solution ${\mathbf{W}^{(i,l)}}^*, {\mathbf{b}^{(i,l)}}^*, {\alpha_i}^*, \xi^*, \omega^*$ to the training problem for ReLU activation with skip connection, the parameters $\mathbf{W}^{(i,l)} = {\mathbf{W}^{(i,l)}}^*, {\mathbf{b}^{(i,l)}}^* = {\mathbf{b}^{(i,l)}}^*, \alpha_i = \frac{1}{2}{\alpha_i}^*, \xi = \xi^* + \frac{1}{2}\sum_{i=1}^{m_2} {\mathbf{b}^{(i,l)}}^* {\alpha_i}^*,$ $\omega = \omega^* + \frac{1}{2}\sum_{i=1}^{m_2} {\mathbf{W}^{(i,1)}}^* {\alpha_i}^*$ achieve the same objective in the training problem for absolute value activation with skip connection. Therefore the problems achieve the same optimal value and we have given a map between the solutions. ■

Deep neural networks with sign activation.

In this section, we assume $\sigma(x) = \text{sign}(x)$. First we discuss parallel architectures.

Definition H.21. *Define the* hyperplane arrangement set *for a matrix* $\mathbf{Z} \in \mathbb{R}^{N\times m}$ *as*

$$\mathcal{H}(\mathbf{Z}) := \{\sigma(\mathbf{Z}\mathbf{w} + b\mathbf{1}) : \mathbf{w} \in \mathbb{R}^m, b \in \mathbb{R}\} \subset \{-1, 1\}^N. \tag{H.28}$$

Let S_0 be the set of columns of $\mathbf{X}$. Let $\{S_l\}_{l=1}^{L-1}$ be a tuple of sets satisfying $S_l \subset \mathcal{H}([S_{l-1}])$ and $|S_l| = m_l$. Let $A_{Lpar}(\mathbf{X})$ be the union of all possible sets S_{L-1}.

In Definition H.21, since $m_{L-1} = 1$ in a parallel network, S_{L-1} contains one vector. The set $\mathcal{H}(\mathbf{Z})$ denotes all possible $\{1, -1\}$ labelings of the samples $\{\mathbf{z}_i\}_{i=1}^N$ by a linear classifier. Its size is upper bounded by $|\mathcal{H}(\mathbf{Z})| \le 2\sum_{k=0}^{r-1}\binom{N-1}{k} \le 2r\left(\frac{e(n-1)}{r}\right)^r \le 2^N$, where $r := \text{rank}(\mathbf{Z}) \le \min(N, m)$ (11; 35).

Lemma H.22. *The lower bound problem* (H.12) *is equivalent to*

$$\max_{\lambda} -\mathcal{L}^*_{\mathbf{y}}(\lambda), \quad \text{s.t.} \max_{\mathbf{h}\in A_{Lpar}(\mathbf{X})} |\lambda^T\mathbf{h}| \le \beta. \tag{H.29}$$

Proof. For $l \in [L]$, there is $S_{l-1} \subset \mathcal{H}\left(\mathbf{X}^{(l-1)}\right)$ with $\mathbf{X}^{(l)} = [S_{l-1}]$. Recursing over $l \in [L]$ gives $\left\{\mathbf{X}^{(L)} : \theta \in \Theta\right\} = A_{Lpar}(\mathbf{X})$. So, the constraints of (H.12) and (H.29) are the same. ■

Remark H.23. *Without loss of generality (by scaling by -1), assume that the vectors in $\mathcal{H}(\mathbf{Z})$ start with* 1. *Under this assumption, Lemma* H.22 *still holds.*

Lemma H.24. *Let $\mathbf{A} = [A_{Lpar}(\mathbf{X})]$. The lower bound problem* (H.29) *is equivalent to*

$$\min_{\mathbf{z}} \quad \mathcal{L}_{\mathbf{y}}(\mathbf{A}\mathbf{z}) + \beta||\mathbf{z}||_1. \tag{H.30}$$

Proof. Problem (H.29) is the dual of (H.30), and since the problems are convex with feasible regions that have nonempty interior, by Slater's condition, strong duality holds (7). ■

Remark H.25. *The set $A_{L,par}$ consists of all possible sign patterns at the final layer of a parallel neural net, up to multiplying by -1.*

Lemma H.26. *Let $\mathbf{A}$ be defined as in Lemma* H.24. *Let $\mathbf{z}$ be a solution to* (H.30). *Suppose $m_L \ge \|\mathbf{z}\|_0$. There is a parallel neural network satisfying $f_L(\mathbf{X}; \theta) = \mathbf{A}\mathbf{z}$ which achieves the same objective in the rescaled training problem as $\mathbf{z}$ does in* (H.30).

Proof. By definition of $A_{L,par}$ (Definition H.21), for every $\mathbf{A}_i \in A_L(\mathbf{X})$, there are tuples $\left\{\mathbf{W}^{(i,l)}\right\}_{l=1}^{L-1}, \left\{\mathbf{b}^{(i,l)}\right\}_{l=1}^{L-1}$ of parameters such that $\mathbf{A}_i = \hat{\mathbf{X}}^{(i,L)}$. Let $\mathcal{I} = \{i : z_i \ne 0\}$. For $i \in \mathcal{I}$, set $\alpha_i = z_i$. This gives a neural net $f_L(\mathbf{X}; \theta) = \sum_{i\in\mathcal{I}} \alpha_i \hat{\mathbf{X}}^{(i,L)} = \mathbf{A}\mathbf{z}$ with $|\mathcal{I}| \le m_L$. ■

Remark H.27. *Lemma* H.26 *analogously holds for the tree network by a similar argument: by construction of A_{Ltree}, there is a neural net satisfying $f_L(\mathbf{X}; \theta) = \mathbf{A}\mathbf{z}$.*

Proposition H.28. *For L-layer parallel networks with sign activation, the Lasso problem* (H.30) *problem and the original training problem are equivalent.*

Proof. By Lemma H.24, the Lasso problem is a lower bound for the training problem. By the reconstruction in Lemma H.26 (see Remark H.20), the lower bound is met with equality. ■

Definition H.29. *Recall the set $\mathcal{H}$ defined in (H.28). Define a matrix-to-matrix operator*

$$J^{(m)}(\mathbf{Z}) := \left[\bigcup_{|S|=m} \mathcal{H}(\mathbf{Z}_S)\right]. \tag{H.31}$$

For $L = 2$, let $A_{L\,tree}(\mathbf{X}) = \mathcal{H}(\mathbf{X})$ and for $L \geq 2$, let $A_{L\,tree}(\mathbf{X})$ be the set of columns in $J^{(m_{L-1})} \circ \cdots \circ J^{(m_2)}(\mathcal{H}(\mathbf{X}))$.

The columns of $J^{(m)}(\mathbf{Z})$ are all hyperplane arrangement patterns of m columns of $\mathbf{Z}$.

Lemma H.30. *For $L \geq 3$, the lower bound problem (H.9) for tree networks is equivalent to*

$$\max_{z\in\mathbb{R}^N} -\mathcal{L}^*_{\mathbf{y}}(\lambda), \quad \text{s.t.} \max_{\mathbf{h}\in A_{L\,tree}(\mathbf{X})} |\lambda^T \mathbf{h}| \leq \beta. \tag{H.32}$$

Proof. Let $\mathbf{u}$ be a tuple such that $u_1 = 1$. First suppose $\mathbf{u}$ has length $L-2$. For all nodes i, $\left\{\sigma\left(\mathbf{X}\mathbf{w}^{(\mathbf{u}+i)} + b^{(\mathbf{u}+i)}\mathbf{1}\right) : \mathbf{w}^{(\mathbf{u}+i)} \in \mathbb{R}^d, b^{(\mathbf{u}+i)} \in \mathbb{R}\right\} = \mathcal{H}(\mathbf{X})$ independently of any other sibling nodes $j \neq i$. So every $\mathbf{X}^{(\mathbf{u})}$ is the linear combination of m_2 columns in $\mathcal{H}(\mathbf{X})$, with the choice of columns independent of other $\mathbf{u}$ of the same length. Next, for all $\mathbf{u}$ of length $L-3$, the set of all possible $\sigma\left(\mathbf{X}^{(\mathbf{u}+i)} + b^{(\mathbf{u}+i)}\mathbf{1}\right)$ is $J^{(m_2)}\left(\mathcal{H}(\mathbf{X})\right)$. Repeating this for decreasing lengths of $\mathbf{u}$ until $\mathbf{u}$ has length 1 gives $\tilde{\mathbf{X}} = \sigma\left(\mathbf{X}^{(i)} + b^{(i)}\mathbf{1}\right) = A_{L\text{tree}}(\mathbf{X})$. ■

H.4. 1-D data. In this section, we assume the data is 1-D. We will refer to a switching set and a rectangular network defined in Subsection 3.2 and Subsection 2.2.

Lemma H.31. *Let $m_1, m_2 \in \mathbb{N}, k \in [m_1 m_2]$. A sequence $\{h_i\}$ in $\{-1, 1\}$ that starts with 1 and switches k times is the sum of at most m_2 sequences in $\{-1, 1\}$ that switch at most m_1 times, and the all-ones sequence.*

Proof. Suppose h_i switches at $i_1 < \cdots < i_k$. Let $Q = \left\lceil \frac{k}{m_1} \right\rceil \leq m_2$. For $q \in [Q]$, let $\left\{h_i^{(q)}\right\}$ be a sequence in $\{-1, 1\}$ that starts with $(-1)^{q+1}$ and switches precisely at $i \in \{I_1, \cdots, I_k\}$ satisfying $i = j \mod m_2$, which occurs at most m_1 times. Let $s_i = \sum_j h_i^{(q)}$. Then $s_1 = \mathbf{1}\{Q \text{ odd}\} \in \{h_1, h_1 - 1\}$. For $i > 1$,

$$s_i = \begin{cases} s_{i-1} + 2 & \text{if } h_{i-1}^{(q)} = -1, h_i^{(q)} = 1 \text{ for some } q \\ s_{i-1} - 2 & \text{if } h_{i-1}^{(q)} = 1, h_i^{(q)} = -1 \text{ for some } q \\ s_i & \text{else.} \end{cases}$$

So $\{s_i\}$ is a sequence in $\{0, -2\}$ or $\{-1, 1\}$ that changes value precisely at $i_1, \ldots, i_k$. Therefore $\{s_i\}$ is either $\{h_i\}$ or $\{h_i - 1\}$. ■

Lemma H.32. *Let $p, m \in \mathbf{N}$. Let $\mathbf{z} \in \{-1, 1\}^N$ with at most pm switches. There is an integer $n \leq m$, $\mathbf{w} \in \{-1, 1\}^n$, and a $N \times n$ matrix $\mathbf{H}$ with columns in $\mathbf{H}^{(p)}$ such that $\mathbf{z} = \sigma(\mathbf{H}\mathbf{w})$.*

Proof. For $x \in \{-1, 1\}, \sigma(x) = \sigma(x - 1)$. Apply Lemma H.31 with $m_2 = p, m_1 = m$. ■

Lemma H.33. *$\mathcal{H}(\mathbf{X})$ consists of all columns in $\mathbf{H}^{(1)}$.*

Proof. First, $\mathbf{1} = \sigma(\mathbf{0}) = \sigma(X \cdot \mathbf{0}) \in \mathcal{H}(\mathbf{X})$. Next, let $\mathbf{h} \in \mathcal{H}(\mathbf{X}) - \{\mathbf{1}\} \in \{-1,1\}^N$. By definition of $\mathcal{H}(\mathbf{X})$, there exists $w, b \in \mathbb{R}$ such that $\mathbf{h} = \mathbf{X}w + b\mathbf{1}$. Note $h_1 = 1$ (Remark H.23). Let i be the first index at which $\mathbf{h}$ switches. So $x_i w + b < 0 \leq x_{i-1}w + b$, which implies $x_i w < x_{i-1} w$. For all $j > i$, $x_j < x_i$ so $h_j = \sigma(x_j w + b) \leq \sigma(x_i w + b) = \sigma(h_i) = -1$, so $h_j = -1$. So $\mathbf{h}$ switches at most once.

Now, let $\mathbf{h} \in \{-1,1\}^N$ with $h_1 = 1$. Suppose $\mathbf{h}$ switches once, at index $i \in \{2, \cdots, N\}$. In particular, $h_i = -1$. Let $w = 1, b = -x_{i-1}$. Then at $j < i$, $x_j w + b = x_j - x_{i-1} \geq 0$ so $h_j = \sigma(x_j w + b) = 1$. And for $j \geq i$, $x_j w + b = x_j - x_{i-1} < 0$ so $h_j = -1$. So $\mathbf{h} \in \mathcal{H}(\mathbf{X})$. ■

Proposition H.34. *The set $A_{L=3,par}$ contains all columns of $\mathbf{H}^{(m_1)}$.*

Proof. Note $A_{L=3,par} = \bigcup_{\mathbf{X^{(1)}}} \mathcal{H}\left(\mathbf{X^{(1)}}\right)$. From Lemma H.33, any possible column in $\mathbf{X^{(1)}}$ switches at most once. Apply Lemma H.32 with $p = 1, m = m_1$ (so that $mp = m_1$ switches), to any column $\mathbf{z}$ of $\mathbf{X^{(1)}}$. ■

Proposition H.35. *For a rectangular network, $A_{L,par}(\mathbf{X})$ is contained in the columns of $\mathbf{H}^{(m_1)}$.*

Proof. Let $\mathbf{W}^{(1)} \in \mathbb{R}^{1 \times m_1}$. For any $w, b \in \mathbb{R}$, since the data is ordered, $\sigma\left(\mathbf{X}w + b\mathbf{1}\right) \in \{-1,1\}^N$ has at most 1 switch. Then $\mathbf{X}^{(1)} = \sigma\left(\mathbf{X}\mathbf{W}^{(1)} + \mathbf{1} \cdot \mathbf{b}^{(1)}\right)$ has m_1 columns each with at most one switch. Therefore $\mathbf{X}^{(1)}$ has at most $m_1 + 1$ unique rows. Let R be the set of the smallest index of each unique row. We claim that for all layers $l \in [L]$, the rows of $\mathbf{X}^{(l)}$ are constant at indices in $[N] - R$, that is, for all $i \in [N] - R$, the i^{th} and $(i-1)^{\text{th}}$ rows of $\mathbf{X^{(l)}}$ are the same. We prove our claim by induction. The base case for $l = 1$ already holds.

Suppose our claim holds for $l \in \{1, \cdots, L-1\}$. Let $\mathbf{W}^{(l+1)} \in \mathbb{R}^{(m_l \times m_{l+1})}$. The rows of $\mathbf{X}^{(l)}$ are constant at indices in $[N] - R$, so for any $\mathbf{w} \in \mathbb{R}^{m_l}, b \in \mathbb{R}$, the elements of the vector $\mathbf{X}^{(l)}\mathbf{w} + b\mathbf{1}$ are constant at indices in $[N] - R$. This held for any $\mathbf{w} \in \mathbb{R}^{m_l}$, so the rows of $\mathbf{X}^{(l+1)} = \mathbf{X}^{(l)}\mathbf{W}^{(l+1)} + \mathbf{1} \cdot \mathbf{b}^{(l+1)}$ are again constant at indices in $[N] - R$. By induction, our claim holds for all $l \in [L]$. So $\mathbf{X}^{(L)}$ has at most $|R| \leq m_1 + 1$ unique rows and hence has columns that each switch at most m_1 times. ■

Proposition H.36. *For a rectangular network, $A_{L,par}(\mathbf{X})$ contains all columns of $\mathbf{H}^{(m_1)}$.*

Proof. Let $\mathbf{z} \in \{-1,1\}^N$ with at most $\min\{N-1, m_1\}$ switches. By Proposition H.34, there exists a feasible $\mathbf{X^{(1)}} = \sigma\left(\mathbf{X}\mathbf{W}^{(1)} + \mathbf{1} \cdot \mathbf{b}^{(1)}\right) \in \mathbb{R}^{N \times m}$ and $\mathbf{W^{(2)}} \in \mathbb{R}^{m \times m}$ such that $\mathbf{z}$ is a column of $\mathbf{X^{(2)}} = \sigma\left(\mathbf{X^{(1)}}\mathbf{W^{(2)}} + \mathbf{1} \cdot \mathbf{b}^{(2)}\right)$. Now for every $l \in \{3, \ldots, L-1\}$ we can set $\mathbf{W^{(l)}} = \mathbf{I}_m$ to be the $m \times m$ identity matrix and $\mathbf{b}^{(l)} = \mathbf{0}$ so that $\mathbf{X^{(l)}} = \sigma\left(\mathbf{X^{(l-1)}}\mathbf{W^{(l)}}\right) = \mathbf{X^{(l-1)}}$. Then $\mathbf{X}^{(L)} = \mathbf{X}^{(2)}$ contains $\mathbf{z}$ as a column. Therefore $\mathbf{z} \in A_{L,par}(\mathbf{X})$. ■

Lemma H.37. *Let $K = \prod_{l=1}^{L-1} m_l$. The set $A_{L\,tree}(\mathbf{X})$ consists of all columns in $\mathbf{H}^{(K)}$.*

Proof. For $l \in [L]$, let $A_l = A_{l\,tree}(\mathbf{X})$. Let $p_k = \prod_{l=1}^{k-1} m_l$. We claim for all $l \in \{2, \cdots, L\}$, A_l consists of all columns in $\mathbf{H}^{(p_l)}$. We prove our claim by induction on l. The base case when $l = 2$ holds by Lemma H.33. Now suppose Lemma H.37 holds when $l = k \in \{2, \cdots, L-1\}$. Observe $A_k \subset A_{k+1}$, so if $p_k \geq N-1$, then Lemma H.37 holds for $l = k+1$. So suppose $p_k < N-1$. Then A_k contains all vectors in $\{-1,1\}^N$ with at most p_k switches.

Let $\mathbf{h} \in A_{k+1}$. The set A_{k+1} contains all columns in $J^{(m_k)}([A_k])$. So, there exist $\mathbf{w} \in \mathbb{R}^{m_k}, b \in \mathbb{R}$ and a submatrix $\mathbf{Z} = [A_k]_S$ where $|S| = m_k$ and $\mathbf{h} = \sigma(\mathbf{Z}\mathbf{w} + \mathbf{b})$. Each of the at most m_k columns of $\mathbf{Z}$ has at most p_k switches, so the N rows of $\mathbf{Z}$ change at most $m_k p_k = p_{k+1}$ times. So $\mathbf{Zw} + \mathbf{b}$ changes value, and hence $\mathbf{h} = \sigma(\mathbf{Zw} + \mathbf{b})$ switches, at most p_{k+1} times. Conversely, by Lemma H.32 with $p = p_k, m = m_k$, the set A_{k+1} of all columns in $J^{(m_k)}([A_k])$ contains all vectors with at most $p_k m_k = p_{k+1}$ switches. So our claim holds for $l = k+1$. By induction, it holds for $l = L$ layers. ∎

H.5. Proofs of results in Subsection 3.2.

Proof of Lemma 3.16. The result follows from the definition of $\mathbf{A}$, the assumption that the data is ordered and that the output of sign activation is σ is ± 1 for sign activation. ∎

Proof of Theorem 3.17. For the parallel network, apply Proposition H.28 and then apply Proposition H.34 for $L = 3$, and Proposition H.35 and Proposition H.36 for $L \geq 3$. For tree networks, by Remark H.27 the training problem is equivalent to Lasso lower bound with dictionary $A_{L,tree}$ given by Lemma H.37. ∎

Proof of Corollary 3.18. Follows from Theorem 3.17, Lemma H.26 and Remark H.27. ∎

Proof of Lemma C.1. By Theorem 3.17 and Lemma H.4, it suffices to show the parameters without unscaling achieve the same objective in the rescaled problem (H.4) as Lasso. First, $|\mathcal{I}| \leq m_2$ and by Theorem 3.17, $m^{(i)} \leq m_1$ so the weight matrices are the correct size. Let $S_n^{(i)}$ be the number of times $\mathbf{A}_i$ switches until index n. Since $x_1 > \cdots > x_N$, we get $\hat{\mathbf{X}}^{(i,2)}_{n,j} = \sigma\left(\mathbf{X}\mathbf{W}^{(i,1)} + \mathbf{1} \cdot \mathbf{b}^{(i,1)}\right)_{n,j} = \sigma\left(x_n - x_{I_j^{(i)}-1}\right) = -\sigma\left(S_n^{(i)} - j\right)$. So $\hat{\mathbf{X}}^{(i,3)}_n = \sigma\left(\hat{\mathbf{X}}^{(i,2)}\mathbf{W}^{(i,2)} + \mathbf{b}^{(i,2)}\mathbf{1}\right)_n = \sigma\left(\sum_{j=1}^{S_n^{(i)}}(-1)(-1)^{j+1} + \sum_{j=S_n^{(i)}+1}^{m^{(i)}}(1)(-1)^{j+1} - \mathbf{1}\left\{m^{(i)} \text{ odd}\right\}\right) = \sigma\left(-2 \cdot \mathbf{1}\left\{S_n^{(i)} \text{ odd}\right\}\right) = (-1)^{S_n^{(i)}} = \mathbf{A}_{n,i}$. So, $f_3(\mathbf{X};\theta) = \xi + \sum_{i \in \mathcal{I}} \alpha_i \hat{\mathbf{X}}^{(i,3)} = \mathbf{Az}$. And, $||\boldsymbol{\alpha}||_1 = ||\mathbf{z}^*_{\mathcal{I}}||_1 = ||\mathbf{z}^*||_1$. So the rescaled problem and Lasso achieve the same objective. ∎

Remark H.38. *For a rectangular network, a reconstruction similar to Lemma C.1 holds by setting additional layer weight matrices to the identity.*

Proof of Corollary C.2 . By Remark 3.19, $p^*_{L=3,\beta} \leq p^*_{L=2,\beta}$. Let $\theta^{(L)}$ and $\boldsymbol{\alpha}^{(L)}$ denote θ and $\boldsymbol{\alpha}$ for a L-layer net. Since the training and rescaled problems have the same optimal value, to show $p^*_{L=2,\beta} \leq p^*_{L=3,m_1\beta}$, it suffices to show for any optimal $\theta^{(3)}$, there is $\theta^{(2)}$ with $f_2\left(\mathbf{X};\theta^{(2)}\right) = f_3\left(\mathbf{X};\theta^{(3)}\right)$ and $\left\|\boldsymbol{\alpha}^{(2)}\right\|_1 \leq m_1 \left\|\boldsymbol{\alpha}^{(3)}\right\|_1$. Let $\mathbf{z}^*$ be optimal in the 3-layer Lasso problem (1.2). Let $m_3^* = ||\mathbf{z}^*||_0$ and let $\mathbf{z} \in \mathbb{R}^{m_3^*}$ be the subvector of nonzero elements of $\mathbf{z}^*$. Let $m = m_1 m_3^*$. By Lemma C.1 and its proof, there are $\mathbf{W}^{(i,1)} \in \mathbb{R}^{1 \times m_1}, \mathbf{b}^{(i,1)} \in \mathbb{R}^{1 \times m_1}, \mathbf{W}^{(i,2)} \in \{1,-1,0\}^{m_1}, \mathbf{b}^{(i,2)} \in \mathbb{R}$ such that $\hat{\mathbf{X}}^{(i,2)}\mathbf{W}^{(i,2)} + \mathbf{b}^{(i,2)}\mathbf{1} \in \{-2,0\}^N$ and $f_3\left(\mathbf{X};\theta\right) =$

$$\sum_{n=1}^{m_3^*} z_i \sigma\left(\hat{\mathbf{X}}^{(i,2)}\mathbf{W}^{(i,2)} + \mathbf{1}\cdot\mathbf{b}^{(i,2)}\right) = \sum_{i=1}^{m_2^*} z_i^*\left(\hat{\mathbf{X}}^{(i,2)}\mathbf{W}^{(i,2)} + \mathbf{1}\cdot\mathbf{b}^{(i,2)} + \mathbf{1}\right) =$$
$$\sum_{i=1}^{m_3^*} z_i^*\left(\sigma\left(\mathbf{X}\mathbf{W}^{(i,1)} + \mathbf{1}\cdot\mathbf{b}^{(i,1)}\right)\mathbf{W}^{(i,2)} + \mathbf{1}\cdot\mathbf{b}^{(i,2)} + \mathbf{1}\right) =$$
$$\sigma\left(\mathbf{X}\underbrace{\left[\mathbf{W}^{(1,1)}\cdots\mathbf{W}^{(m_3^*,1)}\right]}_{\left(\mathbf{W}^{(1,1)},\cdots,\mathbf{W}^{(m,1)}\right)\in\mathbb{R}^{1\times m}} + \mathbf{1}\cdot\underbrace{\left[\mathbf{b}^{(1,1)},\cdots,\mathbf{b}^{(m_3^*,1)}\right]}_{\left(\mathbf{b}^{(1,1)},\cdots,\mathbf{b}^{(m,1)}\right)\in\mathbb{R}^{1\times m}}\right)\underbrace{\begin{bmatrix} z_1\mathbf{W}^{(1,2)} \\ \vdots \\ z_{m_3^*}\mathbf{W}^{(m_3^*,2)}\end{bmatrix}}_{\boldsymbol{\alpha}^{(2)}} + \mathbf{1}\underbrace{\sum_{i=1}^{m_3^*} z_i\left(1+\mathbf{b}^{(i,2)}\right)}_{\xi},$$

which is $f_2\left(\mathbf{X};\theta^{(2)}\right)$ with m neurons. And $\|\boldsymbol{\alpha}^{(2)}\|_1 \leq m_1\|\mathbf{z}^*\|_1 = m_1\|\boldsymbol{\alpha}^{(3)}\|_1$. ■

H.6. Proofs of results for 2-D data.

Proof of Theorem C.3. Lemma H.24 holds for any dimension d, so the Lasso formulation in Theorem 3.12 and Theorem 3.17 similarly hold for $d > 1$ but with a different dictionary $A_{L,par}$ and matrix $\mathbf{A}$.

Let $x'_n = \angle\mathbf{x}^{(n)}$ and order the data so that $x'_1 > x'_2 > \ldots x'_N$. Let $\mathbf{X}' = (x'_1 \cdots x'_N) \in \mathbb{R}^N$. Let $\mathbf{w} \in \mathbb{R}^2$ with $\angle\mathbf{w} \in \left[\frac{\pi}{2}, \frac{3\pi}{2}\right]$. Let $w' = \angle\mathbf{w}$. Note $\mathbf{w}\mathbf{x}^{(n)} \geq 0$ if and only if $x'_n \in \left[w' - \frac{\pi}{2}, w' + \frac{\pi}{2}\right]$. Since $x'_n < \pi$ for all $n \in [N]$, this condition is equivalent to $x'_n \geq w' - \frac{\pi}{2}$, ie $n \leq \max\left\{n \in [N] : x'_n \geq w' - \frac{\pi}{2}\right\}$. So $\left\{\sigma(\mathbf{X}'\mathbf{w}) : w' \in \left[\frac{\pi}{2}, \frac{3\pi}{2}\right]\right\} = \mathbf{H}^{(1)} \cup \{-1\}$. Similarly $\left\{\sigma(\mathbf{X}'\mathbf{w}) : w' \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]\right\}$ is the "negation" of this set. Therefore, the $L = 2$ training problem is equivalent to the Lasso problem with dictionary $\mathbf{H}^{(1)}$. Proposition H.36 holds analogously for this training data, so for $L > 2$, the dictionary is $\mathbf{H}^{(m_{L-1})}$. ■

Proof of Lemma C.4. Observe $n \leq i$ if and only if $\mathbf{x}^{(n)}\mathbf{W}^{(i,1)} \geq 0$ and so $\sigma\left(\mathbf{X}\mathbf{W}^{(i,1)}\right)\left[\mathbf{1}_i^T, -\mathbf{1}_{N-i}^T\right]$ $= \mathbf{A}_i$. Thus $f_{L=2}(\mathbf{X};\theta) = \sum_i \alpha_i\sigma\left(\mathbf{X}\mathbf{W}^{(i,1)}\right) = \mathbf{A}\mathbf{z}^*$ so θ achieves the same objective in the rescaled training problem (H.8), as the optimal value of Lasso (1.2). Unscaling (Definition H.17) optimal parameters of the rescaled problem makes them optimal in the training problem. ■

Solution sets of Lasso under minimal regularization.

Remark H.39. *For each optimal* $\mathbf{z}^*$ *of the Lasso problem, minimizing the objective over* ξ *gives the optimal bias term as* $\xi^* = \left(\mathbf{y} - \mathbf{1}\mathbf{1}^T\mathbf{y}\right) - \left(\mathbf{A} - \mathbf{1}\mathbf{1}^T\mathbf{A}\right)\mathbf{z}^*$.

Remark H.40. *For a neural net with* $L = 2$ *layers and sign activation, by Theorem* 3.12 *the Lasso problem has an objective function* $f(\mathbf{z}) = \frac{1}{2}\|\mathbf{A}\mathbf{z} - \mathbf{y}\|_2^2 + \beta\|\mathbf{z}\|_1$ *where* $\mathbf{A} \in \mathbb{R}^{N\times N}$. *By Lemma* H.43, $\mathbf{A}$ *is full rank, which makes* f *strongly convex. Therefore the Lasso problem has a unique solution* $\mathbf{z}^*$ *(7). Moreover, for any Lasso problem,* $\mathbf{z}^*$ *satisfies the subgradient condition* $\mathbf{0} \in \delta f(\mathbf{z}) = \mathbf{A}^T(\mathbf{A}\mathbf{z}^* - \mathbf{y}) + \beta\partial\|\mathbf{z}^*\|_1$. *Equivalently,*

$$\frac{1}{\beta}\mathbf{A}_n^T(\mathbf{A}\mathbf{z}^* - \mathbf{y}) \in \begin{cases}\{-\text{sign}(\mathbf{z}_n^*)\} & \text{if } \mathbf{z}_n^* \neq 0 \\ [-1,1] & \text{if } \mathbf{z}_n^* = 0\end{cases}, \quad n \in [N].$$

H.7. Proofs of results in Appendix G. Let $\mathbf{e}^{(n)} \in \mathbb{R}^N$ be the n^{th} canonical basis vector, that is $\mathbf{e}_i^{(n)} = \mathbf{1}\{i = n\}$.

Proof of Proposition G.2. We analyze the solution set of $\mathbf{Az} + \xi\mathbf{1} = \mathbf{y}$. We note that $\left(I - \mathbf{1}\mathbf{1}^T/N\right)\mathbf{Az} = \left(I - \mathbf{1}\mathbf{1}^T/N\right)\mathbf{y}$.. As $\mathbf{z}^*$ is optimal in (G.1) , this implies that $(I - \mathbf{1}\mathbf{1}^T/N)\mathbf{Az}^* = (I - \mathbf{1}\mathbf{1}^T/N)\mathbf{y}$. This implies that $(I - \mathbf{1}\mathbf{1}^T/N)\mathbf{A}(\mathbf{z} - \mathbf{z}^*) = 0$. As $x_1 > x_2 > \cdots > x_N$, we have $\mathbf{A}\left(\mathbf{e}^{(1)} + \mathbf{e}^{(N)}\right) \propto \mathbf{1}$. As $\mathbf{A}$ is invertible by Lemma H.44 in Appendix H.8, this implies that there exists $t \in \mathbb{R}$ such that $\mathbf{z} - \mathbf{z}^* = t\left(\mathbf{e}^{(1)} + \mathbf{e}^{(N)}\right)$. It is impossible to have $z_1^* z_N^* > 0$ from the optimality of $\mathbf{z}^*$. Otherwise, by taking $t = -\text{sign}(z_1^*)\min\{|z_1^*|, |z_n^*|\}$, we have $\|\mathbf{z}\|_1 = \|\mathbf{z}^*\|_1 - 2\min\{|z_1^*|, |z_n^*|\} < \|\mathbf{z}^*\|_1$. Therefore, we have $z_1^* z_n^* \leq 0$. We can reparameterize $\mathbf{z} = \mathbf{z}^* + t\text{sign}(z_1^*)(\mathbf{e}^{(1)} + \mathbf{e}^{(N)})$. It is easy to verify that for t such that $-|z_1^*| \leq t \leq |z_n^*|$, we have $\|\mathbf{z}\|_1 = \|\mathbf{z}_n\|_1$, while for other choice of t, we have $\|\mathbf{z}\|_1 > \|\mathbf{z}_n\|_1$. Therefore, the solution set of (G.1) is given by (G.2). ■

Proof of Proposition G.3. Follows from Remark H.40 describing the Lasso objective. ■

Proof of Proposition G.4. By Lemma H.45, for $n \in [N-1]$, $z_{+n}^* = y_n - y_{n-1} \geq 0$ and $z_{+N}^* = y_N \geq 0$. So $\mathbf{z}^*$ achieves an objective value of $\|\mathbf{z}^*\|_1 = y_1$ in (G.1). Now let $\mathbf{z}$ be any solution to (G.1). Then $\mathbf{Az} = \mathbf{y}$. Since the first row of $\mathbf{A}$ is $[\mathbf{1}^T, \mathbf{0}^T]$, we have $y_1 = (\mathbf{Az})_1 = \mathbf{1}^T\mathbf{z}_+ \leq \|\mathbf{z}_+\| \leq \|\mathbf{z}\|_1 \leq \|\mathbf{z}^*\| = y_1$. So $\|\mathbf{z}_+\|_1 = \|\mathbf{z}\|_1 = y_1$, leaving $\mathbf{z}_- = \mathbf{0} = \mathbf{z}_-^*$. Therefore $\mathbf{z}_+ = \mathbf{A}^{-1}\mathbf{y} = \mathbf{z}_+^*$. Applying Lemma H.45 gives the result. ■

Proof of Corollary G.1 . By Lemma H.43, Lemma H.44, Lemma H.45 and Lemma H.46, the dictionary matrix for the 2-layer net is full rank for sign, absolute value, threshold and ReLU activations. The dictionary matrices for deeper nets with sign activation are also full rank by Remark 3.19. Let $\mathbf{u} = \mathbf{A}^T(\mathbf{Az}^* - \mathbf{y})$. By Remark H.40, as $\beta \to 0$, we have $\mathbf{u} \to \mathbf{0}$, so $\mathbf{Az} - \mathbf{y} = (\mathbf{A}\mathbf{A}^T)^{-1}\mathbf{Au} \to \mathbf{0}$. So as $\beta \to 0$, the optimal Lasso objective approaches 0, and by Theorem 3.17 and Theorem 3.12, so does the training problem. So $f_L(\mathbf{X};\theta) \xrightarrow{\beta \to 0} \mathbf{y}$. ■

Proof of Lemma G.7. It can be verified that as shown in Figure 3.5, the Lasso features for a symmetrized network all have slope magnitude $0, 1$, or 2. However, only monotone features contain a segment with slope magnitude 2, and the training data (x_n, y_n) in Figure 3.4 is not monotone. There is a "left branch" consisting of $\{(x_5, y_5) = (-1, 1), (x_4, y_4) = (0, 0)\}$ and a "right branch" consisting of $\{(x_2, y_2) = (3, 1), (x_1, y_1) = (4, 2)\}$. Let $\mathbf{z}^*, \xi^*$ be a Lasso solution that fits the data exactly: $\mathbf{Az}^* + \xi^* = \mathbf{y}$. Let $\mathcal{I}^-$ ($\mathcal{I}^+$) be the set of indices i where the i^{th} feature is monotone and has negative (positive) slope over the left (right) branch. Let $\mathcal{I}^0$ be the set of indices corresponding to features that are not monotone. Let m^- and m^+ be the magnitude of the slopes of the left and right branch, respectively. Then $2\sum_{i \in \mathcal{I}^-}|z_i^*| + \sum_{i \in \mathcal{I}^0}|z_i^*| \geq m^-$ and $2\sum_{i \in \mathcal{I}^+}|z_i^*| + \sum_{i \in \mathcal{I}^0}|z_i^*| \geq m^+$. Note $\mathcal{I}^-, \mathcal{I}^+, \mathcal{I}^0$ are all pairwise disjoint. Therefore $\|\mathbf{z}^*\|_1 \geq \frac{m^+ + m^-}{2} = 1$. ■

Proof of Lemma G.5. Since ReLU and absolute value activations have slopes ± 1 or 0, and the weights of deep narrow networks are ± 1, the features $\hat{\mathbf{X}}^{(L)}(x)$ have slopes ± 1 or 0. Observe $\mathbf{y} = f_L(\mathbf{X};\theta) = \xi^*\mathbf{1} + \mathbf{Az}^* = \xi^*\mathbf{1} + \sum_i z_i^*\mathbf{A}_i$. For any $n \in [N-1]$, $|\mu_n| = \left|\frac{f_L(\mathbf{X};\theta)_{n+1} - f_L(\mathbf{X};\theta)_n}{x_{n+1} - x_n}\right| = \left|\frac{\sum_i z_i^*(\mathbf{A}_{n+1,i} - \mathbf{A}_{n,i})}{x_{n+1} - x_n}\right| = \left|\sum_i z_i^* \frac{\hat{\mathbf{X}}^{(L)}(x_{n+1}) - \hat{\mathbf{X}}^{(L)}(x_n)}{x_{n+1} - x_n}\right|$

$\leq \sum_i |z_i^*| \left|\frac{\hat{\mathbf{X}}^{(L)}(x_{n+1}) - \hat{\mathbf{X}}^{(L)}(x_n)}{x_{n+1} - x_n}\right| \leq \sum_i |z_i^*| = \|\mathbf{z}^*\|$. ■

Proof of Lemma G.6. Let $n \in [N-1]$. Let $\mathcal{S}_n^+ = \{i \in [N] : i > n, (z_+)_i \neq 0\}, \mathcal{S}_n^- = \{i \in [N] : i \leq n, (z_-)_i \neq 0\}$. Observe $\mathcal{S}_{n+1}^+ = \mathcal{S}_n^+ - \{n+1\}$ if $(z_+)_{n+1} \neq 0$ and $\mathcal{S}_{n+1}^+ = \mathcal{S}_n^+$ otherwise. Similarly, $\mathcal{S}_{n+1}^- = \mathcal{S}_n^+ \cup \{n+1\}$ if $(z_-)_{n+1} \neq 0$ and $\mathcal{S}_{n+1}^- = \mathcal{S}_n^-$ otherwise. Now, $\text{ReLU}_{x_i}^+$ has slope 1 after x_i and $\text{ReLU}_{x_i}^-$ has slope 1 before x_i, so $\mu_n = \sum_{i \in \mathcal{S}_n^+} (z_+)_i + \sum_{i \in \mathcal{S}_n^-} (z_-)_i$. Therefore, $|\mu_n - \mu_{n+1}| = \left|-(z_+)_{n+1} + (z_-)_{n+1}\right| \leq \left|(z_+)_{n+1}\right| + \left|(z_-)_{n+1}\right|$. So $\sum_{n=1}^{N-1} |\mu_n - \mu_{n+1}| \leq \|\mathbf{z}^*\|_1 - \left|(z_+)_1\right| - \left|(z_-)_1\right| \leq \|\mathbf{z}^*\|_1$.

Now, for any $n \in [N-1]$, $(\mathbf{A}\mathbf{z}+\xi\mathbf{1})_n - (\mathbf{A}\mathbf{z}+\xi\mathbf{1})_{n+1} = \sum_{i=1}^N (z_+)_i \left(\mathbf{A}_{+n,i} - \mathbf{A}_{+n+1,i}\right) = \sum_{i=1}^N (z_+)_i \left((x_n - x_i)_+ - (x_{n+1} - x_i)_+\right) = \sum_{i=n+1}^N (\mu_{i-1} - \mu_i)(x_n - x_{n+1}) = (x_n - x_{n+1})\mu_n = y_n - y_{n+1}$. And $(\mathbf{A}\mathbf{z} + \xi\mathbf{1})_N = \xi + \sum_{i=1}^N (z_+)_i (x_N - x_i)_+ = y_N + \sum_{i=1}^N (z_+)_i (0) = y_N$. So $\mathbf{A}\mathbf{z} + \xi\mathbf{1} = \mathbf{y}$. And $\|\mathbf{z}\|_1 = \sum_{n=1}^{N-1} |\mu_n - \mu_{n+1}|$, which exactly hits the lower bound on $\|\mathbf{z}^*\|_1$. Therefore $\mathbf{z}, \xi$ is optimal. ■

Remark H.41. *By Lemma H.44, the absolute value network "de-biases" the target vector, normalizes is by the interval lengths $x_i - x_{i+1}$, and applies $\mathbf{E}$ (which contains the difference matrix Δ) twice, acting as a second-order difference detector. By Lemma H.43, the sign network's dictionary inverse contains Δ just once, acting as a first-order difference detector.*

Inverses of 2-layer dictionary matrices.

In this section, we consider the 2-layer dictionary matrix $\mathbf{A}$ as defined in Corollary 3.15. Define the *finite difference matrix*

$$\Delta = \begin{pmatrix} 1 & -1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & -1 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & -1 \\ 0 & 0 & 0 & \cdots & 0 & 1 \end{pmatrix} \in \mathbb{R}^{N-1 \times N-1}. \tag{H.33}$$

Multiplying a matrix on its right by Δ subtracts its consecutive rows. Define the diagonal matrix $\mathbf{D} \in \mathbb{R}^{N \times N}$ by $\mathbf{D}_{i,i} = \frac{1}{x_i - x_{i+1}}$ for $i \in [N-1]$ and $\mathbf{D}_{N,N} = \frac{1}{x_1 - x_N}$. For $n \in \mathbb{N}$, let

$$\mathbf{A}_n^{(s)} = \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ -1 & 1 & 1 & \cdots & 1 \\ -1 & -1 & 1 & \cdots & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -1 & -1 & -1 & \cdots & 1 \end{pmatrix}, \mathbf{A}_n^{(t)} = \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 0 & 1 & 1 & \cdots & 1 \\ 0 & 0 & 1 & \cdots & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix} \in \mathbb{R}^{n \times n}. \tag{H.34}$$

Remark H.42. *The dictionary matrices for sign and threshold activation satisfy $\mathbf{A} = \mathbf{A}_N^{(s)}$ and $\mathbf{A}_+ = \mathbf{A}_N^{(t)}$, respectively.*

H.8. results about 2-layer dictionary matrices.

Lemma H.43. *The dictionary matrix for $\sigma(x) = \text{sign}(x)$ has inverse* $\mathbf{A}^{-1} = \frac{1}{2}\left(\begin{array}{ccc|c} & & & 0 \\ & \Delta & & \vdots \\ & & & 0 \\ & & & -1 \\ \hline 1 & 0 \cdots & 0 & 1 \end{array}\right)$.

Proof. Multiplying the two matrices (see Remark H.42) gives the identity. ■

Lemma H.44. *The dictionary matrix* $\mathbf{A}$ *for absolute value activation has inverse* $\mathbf{A}^{-1} = \frac{1}{2}\mathbf{PEDE}$*, where* $\mathbf{P} = \left(\begin{array}{cccc|c} 0 & 0 & \cdots & 0 & 1 \\ \hline & & & & 0 \\ & & I_{N-1} & & \vdots \\ & & & & 0 \\ & & & & 0 \end{array}\right)$, $\mathbf{E} = \left(\begin{array}{cccc|c} & & & & 0 \\ & & \Delta & & \vdots \\ & & & & 0 \\ & & & & -1 \\ \hline -1 & 0 & \cdots & 0 & -1 \end{array}\right)$.

Proof. For $i \in [N-1], j \in [N]$, $\mathbf{A}_{i,j} - \mathbf{A}_{i+1,j} = \begin{cases} (x_j - x_i) - (x_j - x_{i+1}) = x_{i+1} - x_i & \text{if } i > j \\ (x_i - x_j) - (x_{i+1} - x_j) = x_i - x_{i+1} & \text{if } i \leq j. \end{cases}$
And for all $j \in [N]$, $\mathbf{A}_{1,j} + \mathbf{A}_{N,j} = (x_1 - x_j) + (x_j - x_N) = x_1 + x_N$. Therefore,

$$\mathbf{DEA} = \left(\begin{array}{c|ccc} -1 & & & \\ \vdots & & \mathbf{A}^{(s)}_{N-1} & \\ -1 & & & \\ \hline -1 & -1 & \cdots & -1 \end{array}\right), \quad \frac{1}{2}\mathbf{EDEA} = \left(\begin{array}{c|ccc} 0 & & & \\ \vdots & & I_{N-1} & \\ 0 & & & \\ \hline 1 & 0 & \cdots & 0 \end{array}\right).$$

Applying the permutation $\mathbf{P}$ makes $\frac{1}{2}\mathbf{PEDEA} = \mathbf{I}$, so $\mathbf{A^{-1}} = \frac{1}{2}\mathbf{PEDE}$. ■

Lemma H.45. *The inverse of* $\mathbf{A}_+$ *for threshold activation is* $\mathbf{A}_+^{-1} = \left(\begin{array}{cccc|c} & & & & 0 \\ & & \Delta & & \vdots \\ & & & & 0 \\ & & & & -1 \\ \hline 0 & 0 & \cdots & 0 & 1 \end{array}\right)$.

Proof. Multiplying the two matrices (see Remark H.42) gives the identity. ■

Lemma H.46. *The submatrix* $[(\mathbf{A}_+)_{1:N,2:N}, (\mathbf{A}_-)_{1:N,1}] \in \mathbb{R}^{N \times N}$ *of the dictionary matrix for ReLU activation has inverse* $\mathbf{E}_+\mathbf{DE}_-$*, where*

$$\mathbf{E}_+ = \left(\begin{array}{cccc|c} & & & & 0 \\ & & \Delta & & \vdots \\ & & & & 0 \\ & & & & 1 \\ \hline 0 & 0 & \cdots & 0 & 1 \end{array}\right), \quad \mathbf{E}_- = \left(\begin{array}{cccc|c} & & & & 0 \\ & & \Delta & & \vdots \\ & & & & 0 \\ & & & & -1 \\ \hline 0 & 0 & \cdots & 0 & -1 \end{array}\right).$$

Proof. For $i \in [N-1], j \in [N]$,
$(\mathbf{A}_+)_{i,j} - (\mathbf{A}_+)_{i+1,j} = \begin{cases} 0 & \text{if } i \geq j \\ (x_i - x_j) - (x_{i+1} - x_j) = x_i - x_{i+1} & \text{if } i < j \end{cases}$
and $(\mathbf{A}_-)_{i,1} - (\mathbf{A}_-)_{i+1,1} = (x_1 - x_i) - (x_1 - x_{i+1}) = x_{i+1} - x_i$. Observe that $\mathbf{DE}_-\mathbf{A} = \left(\begin{array}{cccc|c} & & & & -1 \\ & \mathbf{A}^{(t)}_{N-1} & & & \vdots \\ & & & & -1 \\ & & & & -1 \\ \hline 0 & 0 & \cdots & 0 & 1 \end{array}\right)$, and applying $\mathbf{E}_+$ gives $\mathbf{I}$. ■

Solution path for sign activation and binary, periodic labels.

In this section we assume the neural net uses sign activation, and $d = 1$. Recall $\mathbf{e}^{(n)}$ as the n^{th} canonical basis vector (Appendix H.7). Note that in Figures 3.7 , H.4, nd H.5, $\mathbf{y} = \mathbf{h}^{(T)}$, $N = 40, T = 10$, and vectors $\mathbf{v} = (v_1, \cdots, v_N)$ are depicted by plotting (n, v_n) as a dot.

Remark H.47. *A neural net with all weights being* 0 *achieves the same objective in the training problem as the optimal Lasso value and is therefore optimal.*

In this section, we will find the critical value β_c defined in Appendix D. Then for $\beta < \beta_c$, we use the subgradient condition from Remark H.40 to solve the Lasso problem (1.2). Note when $L = 2$, $(\mathbf{A}_n)^T = (\mathbf{1}_{1:n}, -\mathbf{1}_{n+1:N})$ switches at $n+1$.

H.9. $L = 2$. Assume the network has 2 layers.

Lemma H.48. *The elements of* $\mathbf{A}^T\mathbf{A} \in \mathbb{R}^{N \times N}$ *are* $(\mathbf{A}^T\mathbf{A})_{i,j} = N - 2|i-j|$.

Proof. If $1 \le i \le j \le N$ then $(\mathbf{A}^T\mathbf{A})_{i,j} = \sum_{k=1}^{i-1} \mathbf{A}_{i,k}\mathbf{A}_{j,k} + \sum_{k=i}^{j-1} \mathbf{A}_{i,k}\mathbf{A}_{j,k} + \sum_{k=j}^{N} \mathbf{A}_{i,k}\mathbf{A}_{j,k}$ $= \sum_{k=1}^{i-1}(1)(1) + \sum_{k=i}^{j-1}(-1)(1) + \sum_{k=j}^{N}(-1)(-1) = (i-1) - (j-1-i+1) + (N-j+1) = N + 2(i-j)$ $= N - 2|i-j|$. Since $\mathbf{A}^T\mathbf{A}$ is symmetric, if $1 \le j \le i \le N$ then $(\mathbf{A}^T\mathbf{A})_{i,j} = (\mathbf{A}^T\mathbf{A})_{j,i} = N + 2(j-i) = N - 2|i-j|$. So for any $i, j \in [N]$, $(\mathbf{A}^T\mathbf{A})_{i,j} = N - 2|i-j|$. ∎

Remark H.49. *By Lemma H.48,* $\mathbf{A}^T\mathbf{A}$ *is of the form*

(H.35)
$$\mathbf{A}^T\mathbf{A} = \begin{pmatrix} N & N-2 & N-4 & \cdots & 2 & 0 & -2 & \cdots & 6-N & 4-N & 2-N \\ N-2 & N & N-2 & \cdots & 4 & 2 & 0 & \cdots & 8-N & 6-N & 4-N \\ N-4 & N-2 & N & \cdots & 6 & 4 & 2 & \cdots & 10-N & 8-N & 6-N \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 6-N & 8-N & 10-N & \cdots & 2 & 4 & 6 & \cdots & 10-N & 8-N & 6-N \\ 4-N & 6-N & 8-N & \cdots & 0 & 2 & 4 & \cdots & 8-N & 6-N & 4-N \\ 2-N & 4-N & 6-N & \cdots & -2 & 0 & 2 & \cdots & N-4 & N-2 & N \end{pmatrix}.$$

An example of a column of $\mathbf{A}^T\mathbf{A}$ is plotted in the left plot of Figure H.4.

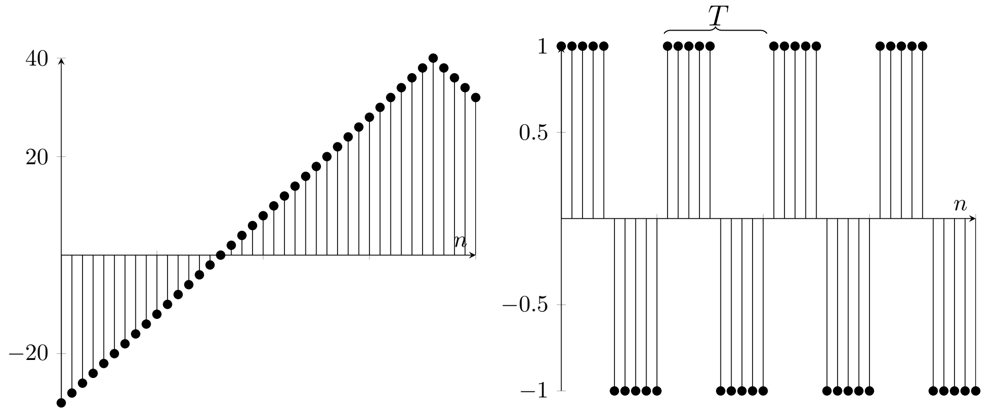


Figure H.4: Left: Column 10 of $\mathbf{A}^T\mathbf{A}$ for $N = 40$. Right: Vector $\mathbf{h}^{(T)}$ for $T = 10$.

Remark H.50. *For* $n \in [N]$, $(\mathbf{A}^T\mathbf{y})_n = \sum_{i=1}^{n} \mathbf{y}_i - \sum_{i=n+1}^{N} \mathbf{y}_i$.

Definition H.51. *For* $a, b \in \mathbb{Z}$, *let* $\mathrm{Quot}(a,b) \in \mathbb{Z}$ *and* $\mathrm{Rem}(a,b) \in \{0, \cdots, b-1\}$ *be the* *quotient and remainder, respectively, when a is divided by b. The* modified remainder *is*

$$\mathrm{rem}(a,b) = \begin{cases} \mathrm{Rem}(a,b) & \text{if } \mathrm{Rem}(a,b) > 0 \\ b & \text{if } \mathrm{Rem}(a,b) = 0 \end{cases} \in [b].$$

The modified quotient *is*

$$\mathrm{quot}(a,b) = \frac{a - \mathrm{rem}(a,b)}{b} = \begin{cases} \mathrm{Quot}(a,b) & \text{if } \mathrm{Rem}(a,b) > 0 \\ \mathrm{Quot}(a,b) - 1 & \text{if } \mathrm{Rem}(a,b) = 0. \end{cases}$$

The quotient and remainder are modified to handle vector indices starting at 1 instead of being zero-indexed.

Remark H.52. *The square wave has elements* $\mathbf{h}^{(T)}_n = \begin{cases} -1 & \text{if} \quad \mathrm{rem}(T,n) \le T/2 \\ 1 & \text{else.} \end{cases}$

Remark H.53. *Since $\mathbf{h}^{(T)}$ is periodic and zero mean, for $i, n \ge 0$, $\sum_{j=iT+1}^{nT} \mathbf{h}^{(T)}{}_j = 0$.*

Lemma H.54. *The vector $\mathbf{A}^T\mathbf{h}^{(T)}$ is periodic with period T. For $n \in [T]$,*

$$\left(\mathbf{A}^T\mathbf{h}^{(T)}\right)_n = 2\begin{cases} n & \text{if } n \le \frac{T}{2} \\ T - n & \text{else} \end{cases} \in [0,T].$$

Proof. By Remark H.50, Remark H.53, and periodicity of $\mathbf{h}^{(T)}$, for $n \in [T], j \in [2k-1]$,

$$\left(\mathbf{A}^T\mathbf{h}^{(T)}\right)_{n+jT} = \sum_{i=jT+1}^{jT+n} \mathbf{h}^{(T)}{}_i - \sum_{i=n+jT+1}^{(j+1)T} \mathbf{h}^{(T)}{}_i = \begin{cases} \sum_{i=1}^{n} 1 - \sum_{i=n+1}^{T/2} 1 + \sum_{i=1+T/2}^{T} 1 & \text{if } n \le \frac{T}{2} \\ \sum_{i=1}^{T/2} 1 - \sum_{i=\frac{T}{2}+1}^{n} 1 + \sum_{i=n+1}^{T} 1 & \text{else.} \end{cases}$$

Simplifying gives the result. ∎

Lemma H.55. *Let $q_n = \mathrm{quot}\left(n, \frac{T}{2}\right) \in \{0, \cdots, 2k-1\}$. Then*

$$\left(\mathbf{A}^T\mathbf{h}^{(T)}\right)_n = 2(-1)^{q_n+1}\mathrm{rem}\left(n, \frac{T}{2}\right) - \mathbf{1}\{q_n \text{ odd}\}T. \tag{H.36}$$

Proof. Follows from Lemma H.54. ∎

Corollary H.56. *Suppose $\mathbf{z} = \mathbf{e}^{\left(\frac{T}{2}\right)} + \mathbf{e}^{\left(N-\frac{T}{2}\right)}$. Then for $n \le \frac{T}{2}$ and $n \ge N - \frac{T}{2}$, $\frac{1}{2}(\mathbf{A}^T\mathbf{A}\mathbf{z})_n = \left(\mathbf{A}^T\mathbf{h}^{(T)}\right)_n$. And if $\frac{T}{2} \le n \le N - \frac{T}{2}$, then $(\mathbf{A}^T\mathbf{A}\mathbf{z})_n = 2T$.*

Proof. By Lemma H.48, for $n \in [N]$, $(\mathbf{A}^T\mathbf{A}\mathbf{z})_n = 2\left(N - \left|n - \frac{T}{2}\right| - \left|n - N + \frac{T}{2}\right|\right)$. Simplifying and applying Lemma H.54 gives the result. ∎

Lemma H.57. *If $\mathbf{y} = \mathbf{h}^{(T)}$, then the critical β (defined in Section 4) is $\beta_c = T$.*

Proof. By Remark H.47, $\beta_c = \max_{n\in[N]} |\mathbf{A}_n^T\mathbf{y}| = \max_{n\in[N]} \left|\left(\mathbf{A}^T\mathbf{h}^{(T)}\right)_n\right| = T$. ■

Lemma H.58. *Let $\mathbf{y} = \mathbf{h}^{(T)}$. If $\beta_T \geq \frac{1}{2}$ then the solution to the Lasso problem (1.2) is $\mathbf{z}^* = \frac{1}{2}(1-\beta_T)_+ \left(\mathbf{e}^{(\frac{T}{2})} + \mathbf{e}^{(N-\frac{T}{2})}\right)$.*

Proof. By Lemma H.57, $\beta_c = T$. By Lemma H.57, if $\beta_T \geq 1$ then $\mathbf{z}^* = \mathbf{0}$ as desired. Now suppose $\frac{1}{2} \leq \beta_T \leq 1$. Let $\delta = 1 - \beta_T, \mathbf{g} = \mathbf{A}_n^T(\mathbf{A}\mathbf{z}^* - \mathbf{y})$. By Corollary H.56 and Lemma H.54,

$$\mathbf{g} = \begin{cases} (\delta-1)\left(\mathbf{A}^T\mathbf{h}^{(T)}\right)_n = -2\beta_T n & \in [-\beta, 0] \qquad \text{if } n \leq \dfrac{T}{2} \\ \left(\delta T - (\mathbf{A}^T\mathbf{h}_{\mathbf{k},\mathbf{T}})_n\right) = \left(\beta_c - \beta - (\mathbf{A}^T\mathbf{h}^{(T)})_n\right) & \in [-\beta, \beta] \quad \text{if } \dfrac{T}{2} \leq n \leq N - \dfrac{T}{2} \\ (\delta-1)\left(\mathbf{A}^T\mathbf{h}^{(T)}\right)_n = -2\beta_T(N-n) & \in [-\beta, 0] \qquad \text{if } N - \dfrac{T}{2} \leq n \end{cases},$$

where the second set inclusion follows from $\left(\mathbf{A}^T\mathbf{h}^{(T)}\right)_n \in [0, \beta_c]$ by Lemma H.54 so that $-\beta \leq \mathbf{g} \leq \beta_c - \beta \leq \beta$. Therefore, $|\mathbf{A}_n^T(\mathbf{A}\mathbf{z}^* - \mathbf{y})| \leq \beta$, and at $n \in \{n : \mathbf{z}_n^* \neq 0\} = \left\{\frac{T}{2}, N - \frac{T}{2}\right\}$, we have $\mathbf{A}_n^T(\mathbf{A}\mathbf{z}^* - \mathbf{y}) = -\beta_T\frac{T}{2} = -\beta = -\beta\text{sign}(\mathbf{z}_n^*)$. By Remark H.40, $\mathbf{z}^*$ is optimal. ■

Lemma H.59. *Let $a, b, c, d \in \mathbf{Z}_+, d \in \mathbb{R}, r = 1 - \text{rem}(b-a, 2)$. Then*

(H.37)
$$\sum_{j=a}^{b}(-1)^j(c-jd) = (-1)^a\left((c-ad)r + (-1)^r\frac{(b-(a+r)+1)d}{2}\right) = (-1)^a\begin{cases}\frac{(b-a+1)d}{2} & \textit{if } r=0 \\ c-ad-\frac{(b-a)d}{2} & \textit{else.}\end{cases}$$

Proof. We have

$$\begin{aligned}\sum_{j=a}^{b}(-1)^j(c-jd) &= (-1)^a(c-ad)r + \sum_{j=a+r}^{b}(-1)^j(c-jd) \\ &= (-1)^a(c-ad)r + (-1)^{a+r}\sum_{\substack{a+r\leq j\leq b-1 \\ j-(a+r) \text{ is even}}}(c-jd)-(c-(j+1)d)\end{aligned}$$

Simplifying gives (H.37). ■

Lemma H.60. *Let $L{=}2$, $\mathbf{y}{=}\mathbf{h}^{(T)}$, $0{<}\beta{<}\frac{\beta_c}{2}$. Let $w_{\text{bdry}}{=}1{-}\frac{3}{2}\beta_T, w_{\text{cycle}}{=}2\beta_T{-}1$. Let $\mathbf{z}_{\text{bdry}}{=}w_{\text{bdry}}\left(\mathbf{e}^{(\frac{T}{2})}{+}\mathbf{e}^{(N-\frac{T}{2})}\right), \mathbf{z}_{\text{cycle}}{=}w_{\text{cycle}}\sum_{i=2}^{2k-2}(-1)^i\mathbf{e}^{(\frac{T}{2}i)}$. Then $\mathbf{z}^*{=}\mathbf{z}_{bdry}{+}\mathbf{z}_{cycle}$ solves the Lasso problem (1.2).*

Proof. We show $\mathbf{z}^*$ is optimal using the subgradient condition in Remark H.40. By Corollary H.56,

(H.38)
$$\frac{1}{2w_{\text{bdry}}}(\mathbf{A}^T\mathbf{A}\mathbf{z}_{\text{bdry}})_n = \begin{cases}\left(\mathbf{A}^T\mathbf{h}^{(T)}\right)_n & \text{if } n \leq \frac{T}{2} \text{ or } n \geq N - \frac{T}{2} \\ T & \text{if } \frac{T}{2} \leq n \leq N - \frac{T}{2}\end{cases}, \quad n \in [N].$$

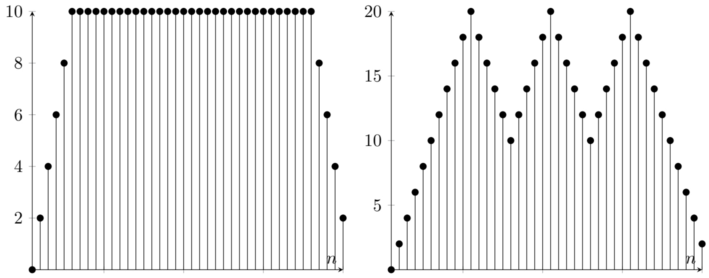


Figure H.5: Examples of $\frac{1}{2w_{bdry}}\mathbf{A}^T\mathbf{A}\mathbf{z}_{\mathbf{bdry}}$ (left) and $\frac{1}{w_{cycle}}\mathbf{A}^T\mathbf{A}\mathbf{z}_{\mathbf{cycle}}$ (right).

Next,

$$\frac{1}{w_{\text{cycle}}}(\mathbf{A}^T\mathbf{A}\mathbf{z}_{\text{cycle}})_n = \sum_{j=2}^{2k-2}(-1)^j\left(N-2\left|n-j\frac{T}{2}\right|\right). \tag{H.39}$$

See Figure H.5. If $n\le\frac{T}{2}$ or $n\ge N-\frac{T}{2}$ then there is $s\in\{-1,1\}$ such that for all $2\le j\le 2k-2$, $n-j\frac{T}{2}=s(n-j\frac{T}{2})$. Applying Lemma H.59 to (H.39) and simplifying gives $(\mathbf{A}^T\mathbf{A}\mathbf{z}_{\mathbf{cycle}})_n=$ $w_{\text{cycle}}\,(N-skT+2sn)$. Comparing with Lemma H.54 gives

$$\frac{1}{w_{cycle}}(\mathbf{A}^T\mathbf{A}\mathbf{z}_{\text{cycle}})_n = 2\begin{cases} n & \text{if } n\le\frac{T}{2} \\ N-n & \text{if } n\ge N-\frac{T}{2}\end{cases} = (\mathbf{A}^T\mathbf{h}^{(T)})_n. \tag{H.40}$$

Next suppose $\frac{T}{2}\le n\le N-\frac{T}{2}$. Let $q_n=\text{quot}\left(n,\frac{T}{2}\right), r_n=\text{rem}\left(n,\frac{T}{2}\right)$. Then

$$\frac{1}{w_{cycle}}(\mathbf{A}^T\mathbf{A}\mathbf{z}_{\mathbf{cycle}})_n = \sum_{j=2}^{q_n}(-1)^j\left(N-2\left(n-j\frac{T}{2}\right)\right)+\sum_{j=q_n+1}^{2k-2}(-1)^j\left(N+2\left(n-j\frac{T}{2}\right)\right). \tag{H.41}$$

Applying Lemma H.59 to (H.41) and simplifying gives

$$\begin{aligned}\frac{1}{w_{cycle}}\left(\mathbf{A}^T\mathbf{A}\mathbf{z}_{\text{cycle}}\right)_n&=\begin{cases} N-2n+2T+\frac{q_n-2}{2}T-\frac{2k-2-q_n}{2}T=2\left(T-\text{rem}\left(n,\frac{T}{2}\right)\right) & \text{if } q_n \text{ is even}\\ \frac{1-q_n}{2}T+N+2n-(1+q_n)T-\frac{2k-3-q_n}{2}T=T+2\text{rem}\left(n,\frac{T}{2}\right) & \text{if } q_n \text{ is odd}\end{cases}\\ &=(2-\mathbf{1}\{q_n\text{ odd}\})T+2(-1)^{q_n+1}\text{rem}\left(n,\frac{T}{2}\right)=2T-\left(\mathbf{A}^T\mathbf{h}_{\mathbf{k},\mathbf{T}}\right)_n,\end{aligned}$$

where the last equality follows from Lemma H.55. Combining with (H.40) gives

$$(\mathbf{A}^T\mathbf{A}\mathbf{z}_{\mathbf{cycle}})_n = w_{cycle}\cdot\begin{cases}\left(\mathbf{A}^T\mathbf{h}^{(T)}\right)_n & \text{if } n\le\frac{T}{2}\text{ or } n\ge N-\frac{T}{2}\\ 2T-\left(\mathbf{A}^T\mathbf{h}^{(T)}\right)_n & \text{if } \frac{T}{2}\le n\le N-\frac{N}{2}.\end{cases} \tag{H.42}$$

Add (H.38) and (H.42) and plug in $\mathbf{y}=\mathbf{h}^{(T)}$ to get

(H.43)

$$(\mathbf{A}^T\mathbf{A}\mathbf{z})_n = \begin{cases} (w_{cycle}+2w_{bdry})(\mathbf{A}^T\mathbf{y})_n = (1-\beta_T)\,(\mathbf{A}^T\mathbf{y})_n & \text{if } n \le \frac{T}{2} \text{ or } n \ge N-\frac{T}{2} \\ (w_{bdry}+w_{cycle})2T - w_{cycle}(\mathbf{A}^T\mathbf{y})_n = \beta + (1-2\beta_T)\,(\mathbf{A}^T\mathbf{y})_n & \text{if } \frac{T}{2} \le n \le N-\frac{N}{2}. \end{cases}$$

Therefore,

$$\text{(H.44)} \qquad -\frac{1}{\beta}\mathbf{A}^T(\mathbf{A}\mathbf{z}-\mathbf{y})_n = \begin{cases} \frac{1}{\beta_c}(\mathbf{A}^T\mathbf{y})_n & \text{if } n \le \frac{T}{2} \text{ or } n \ge N-\frac{T}{2} \\ 1 + \frac{2}{\beta_c}(\mathbf{A}^T\mathbf{y})_n & \text{if } \frac{T}{2} \le n \le N-\frac{N}{2}. \end{cases}$$

By Lemma H.57, $\beta_c = T$. By Lemma H.54, $0 \le \frac{1}{\beta_c}(\mathbf{A}^T\mathbf{h}^{(T)})_n = \frac{1}{\beta_c}(\mathbf{A}^T\mathbf{y})_n \le 1$, and $\frac{1}{\beta_c}(\mathbf{A}^T\mathbf{y})_n = 1$ when n is an odd multiple of $\frac{T}{2}$. Since $0 < \beta < \frac{\beta_c}{2}$, we have $w_{bdry} > 0$ and $w_{cycle} < 0$. Therefore $\frac{1}{\beta}\mathbf{A}^T(\mathbf{A}\mathbf{z}-\mathbf{y})_n = -\text{sign}(\mathbf{z}^*_n)$ when $\mathbf{z}^*_n \neq 0$, ie n is an integer multiple of $\frac{T}{2}$. And for all $n \in [N]$, $\left|\frac{1}{\beta}\mathbf{A}^T(\mathbf{A}\mathbf{z}-\mathbf{y})_n\right| \le 1$. By Remark H.40, $\mathbf{z}^*$ is optimal. ■

The nonzero indices of $\mathbf{z_{bdry}}$ and $\mathbf{z_{cycle}}$ partition those that are multiples of $\frac{T}{2}$.

Proof of Theorem D.1. By Lemma H.58 and Lemma H.60,

$$\mathbf{z}^* = \begin{cases} \frac{1}{2}(1-\beta_T)_+ \left(\mathbf{e}^{\left(\frac{T}{2}\right)} + \mathbf{e}^{\left(N-\frac{T}{2}\right)}\right) & \text{if } \beta_T \ge \frac{1}{2} \\ \mathbf{z_{bdry}} + \mathbf{z_{cycle}} & \text{if } 0 < \beta_T \le \frac{1}{2}. \end{cases}$$ ■

Proof of Corollary D.2. Note that unscaling (defined in Section 2) does not change the neural network as a function. The reconstructed neural net (Definition B.3) before unscaling is $f_2(x;\theta) = \sum_{i=1}^N z_i^* \sigma(x-x_i)$. For $\frac{1}{2} \le \beta_T \le 1$, $f_2(x;\theta) = \frac{1}{2}(1-\beta_T)_+ \left(\sigma\left(x - x_{\frac{T}{2}}\right) + \sigma\left(x - x_{N-\frac{T}{2}}\right)\right)$. We can compute $f_2(x;\theta)$ similarly for $\beta_T \le \frac{1}{2}$. ■

H.10. $L = 3$ layer nets.

Theorem D.3. Since $\mathbf{y} = \mathbf{h}^{(T)}$ switches $2k-1$ times, by Theorem 3.17, $\mathbf{A}_i = \mathbf{h}^{(T)}$ for some i. Since $\mathbf{y} = -\mathbf{A}_i$, and for all $n \in [N]$, $\|\mathbf{A}_n\|_2 = N$, we have $i \in \text{argmax}_{n\in N}|\mathbf{A}_n^T\mathbf{y}|$. By Remark H.47, $\beta_c = \max_{n\in N}|\mathbf{A}_n^T\mathbf{y}| = \mathbf{y}^T\mathbf{A}_i = N$. So if $\beta > \beta_c$ then $\mathbf{z}^* = 0$, consistent with $z_i = -(1-\beta_T)_+$.

Next, $\mathbf{z}^*$ satisfies the subgradient condition in Remark H.40, since for $n \in [N]$, $\left|\mathbf{A}_n^T(\mathbf{A}\mathbf{z}^*-\mathbf{y})\right| = \left|\mathbf{A}_n^T(z_i\mathbf{A}_i - \mathbf{A}_i)\right| = (z_i - 1)\left|\mathbf{A}_n^T\mathbf{A}_i\right| = \left|\frac{\beta}{\beta_c}\mathbf{A}_n^T\mathbf{y}\right| \le \frac{\beta}{\beta_c}\text{argmax}_{n\in N}\left|\mathbf{A}_n^T\mathbf{y}\right| = \le \beta$. Since $\mathbf{z}_i^* < 0$, when $i = n$, $\mathbf{A}_i^T(\mathbf{A}\mathbf{z}^*-\mathbf{y}) = \beta = -\beta\text{sign}(\mathbf{z}_i^*)$. By Remark H.40, $\mathbf{z}^*$ is optimal. ■

Corollary D.4. Follows from the reconstruction in Lemma C.1. ■

The solution sets of Lasso and the training problem.

H.11. Proofs for results in Appendix F.

Proposition F.1. The result is almost a sub-case of that given by Mishkin and Pilanci (29) with the exception that the bias parameter ξ, is not regularized. Therefore optimality conditions do not impose a sign constraint and it is sufficient that $\mathbf{1}^\top(\mathbf{A}\mathbf{z}+\xi\mathbf{1}-\mathbf{y}) = 0$ for ξ to be optimal. This stationarity condition is guaranteed by $\mathbf{A}\mathbf{z}+\xi\mathbf{1} = \hat{\mathbf{y}}$. Now let us look at the parameters z_i. If $i \notin \mathcal{E}(\beta)$, then $z_i = 0$ is necessary and sufficient from standard results on the Lasso (38). If $i \in \mathcal{E}(\beta)$ and $z_i \neq 0$, then $\mathbf{A}_i^\top(\hat{\mathbf{y}} - \mathbf{y}) = \beta \text{sign}(z_i)$, which shows that $\mathbf{z}_i$ satisfies first-order conditions. If $z_i = 0$, then first-order optimality is immediate since $|\mathbf{A}_i^\top(\hat{\mathbf{y}} - \mathbf{y})| \leq \beta$, holds. Putting these cases together completes the proof. ■

Proposition F.2. This result follows from applying the reconstruction in Definition B.3 to each optimal point in Φ. The reconstution sets $\alpha_i = \text{sign}(z_i)\sqrt{|z_i|}$. From this we deduce $\text{sign}(\alpha_i) = \text{sign}\left(\mathbf{A}_i^\top(\mathbf{y} - \hat{\mathbf{y}})\right)$. The solution mapping determines the values of w_i and b_i and in terms of α_i. Finally, the constraint $f_2(\mathbf{X}; \theta) = \hat{\mathbf{y}}$ follows immediately by equality of the convex and non-convex prediction functions on the training set. ■

Lemma H.61. *Suppose $L = 2$, and the activation is ReLU, leaky ReLU or absolute value. Suppose $m^* \leq m \leq 2N$. Since $m \leq 2N$, we can let $\Theta^{\text{Lasso,stat}} = \{\theta : \exists j \in [N] \text{ s.t. } b_i = -x_j w_i\} \subset \Theta$. Let $\theta^* \in \Theta^{\text{Lasso,stat}} \cap C(\beta)$. Then θ^* is a minima of the Lasso problem.*

Proof. We can ignore ξ since its derivative and reconstruction are straightforward, and it does not interact with any other parameters. We denote vector-vector operations as being performed elementwise.

Since $m^* \leq m$, a neural net reconstructed from Lasso is optimal in the training problem. Let $F(\theta)$ and $F^{\text{Lasso}}(\mathbf{z})$ be the objectives of the non-convex training problem (1.1) and Lasso (1.2), respectively. The parameters θ are stationary if $\theta \in C(\beta)$, i.e., $0 \in \partial F(\theta)$.

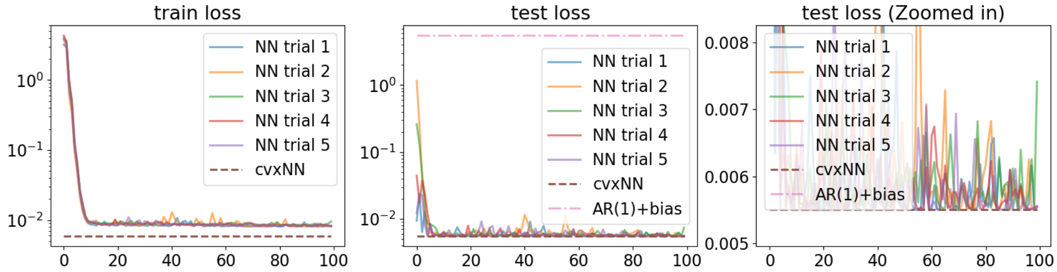
Let $\Theta^{\text{Lasso}} = \{\theta : \exists j \in [N] \text{ s.t. } |w_i| = |\alpha_i|, b_i = -x_j w_i\} \subset \Theta^{\text{Lasso,stat}}$. By a similar argument as the proof of Theorem 3 in (42), since $0 \in \partial F(\theta^*)$, we have $|w_i| = |\alpha_i|$ for all neurons i. Therefore $\theta^* \in \Theta^{\text{Lasso}}$. Thus for $\alpha^* \in \theta^*$, observe that $\theta^* = R^{\alpha,w,b \to \theta}(\boldsymbol{\alpha}^*)$. Let $\tilde{F}(\boldsymbol{\alpha}) = F\left(R^{\alpha,w,b \to \theta}(\boldsymbol{\alpha})\right)$.

Since $0 \in \partial F(\theta^*)$ at $(\boldsymbol{\alpha}^*) = \left(R^{\alpha,w,b \to \theta}\right)^{-1}(\theta^*)$, we have $\tilde{F}(\boldsymbol{\alpha}^*) = F(\theta^*)$. Perform the following operations elementwise. The chain rule gives $\partial \tilde{F}(\boldsymbol{\alpha}^*) = \partial F(\theta^*) \partial R^{\alpha,w,b \to \theta}(\boldsymbol{\alpha}^*) \ni \mathbf{0}$. Let $R^{\alpha toz}(\boldsymbol{\alpha}) = \text{sign}(\boldsymbol{\alpha})\boldsymbol{\alpha}^2$. At $\mathbf{z}^* = R^{\alpha \to z}(\boldsymbol{\alpha}^*)$, we have $F^{\text{Lasso}}(\mathbf{z}^*) = \tilde{F}(\boldsymbol{\alpha}^*)$. The chain rule gives $\partial F^{\text{Lasso}}(\mathbf{z}^*) = \partial \tilde{F}(\boldsymbol{\alpha}^*) \partial R(\mathbf{z}^*) \ni \mathbf{0}$. Since the Lasso problem (1.2) is convex, the result holds. ■

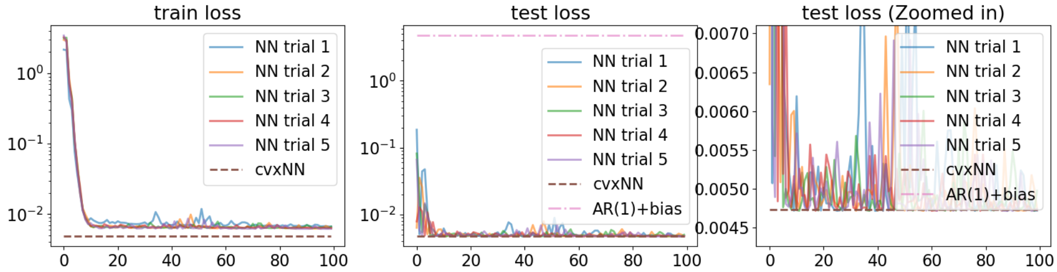
Proof of Proposition F.3. Observe that $R(\Phi(\beta)) \subseteq \tilde{\mathcal{C}}(\beta) \cap \Theta^{\text{Lasso,stat}} \subseteq \mathcal{C}(\beta) \cap \Theta^{\text{Lasso,stat}} \subseteq R(\Phi(\beta))$, where the first and last subset inequality follow from Theorem 3.12 and Lemma H.61, respectively. Therefore all subsets in the above expression are equal. Observe that $P\left(\Theta^{\text{Lasso,stat}}\right) = \Theta^P$ and so $P\left(C(\beta) \cap \Theta^{\text{Lasso,stat}}\right) = C(\beta) \cap P\left(\Theta^{\text{Lasso,stat}}\right) = C(\beta) \cap \Theta^P$ and similarly $P\left(\tilde{C}(\beta) \cap \Theta^{\text{Lasso,stat}}\right) = \tilde{C}(\beta) \cap \Theta^P$. Now apply P to all subsets above. ■

Numerical results.

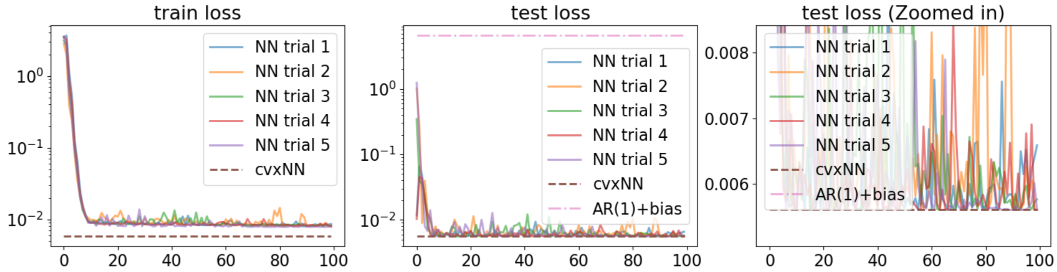
H.12. Autoregression figures. In all figures except for the regularization path, the horizontal axis is the training epoch.



$m = 2$



$m = 5.$



$m = 10.$

Figure H.6: Planted data. $\sigma^2 = 1$.

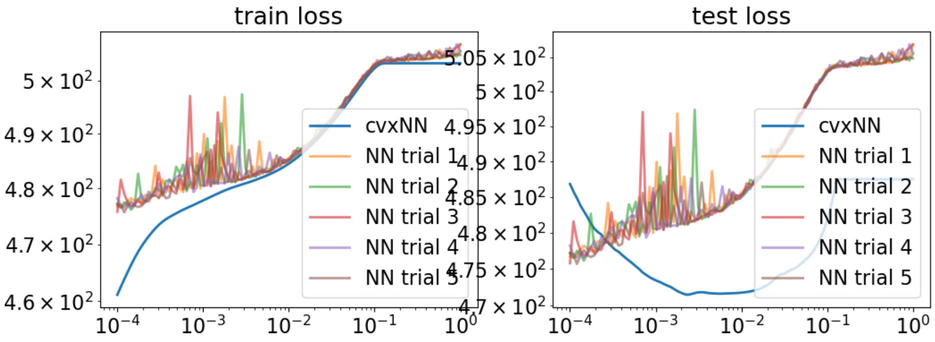
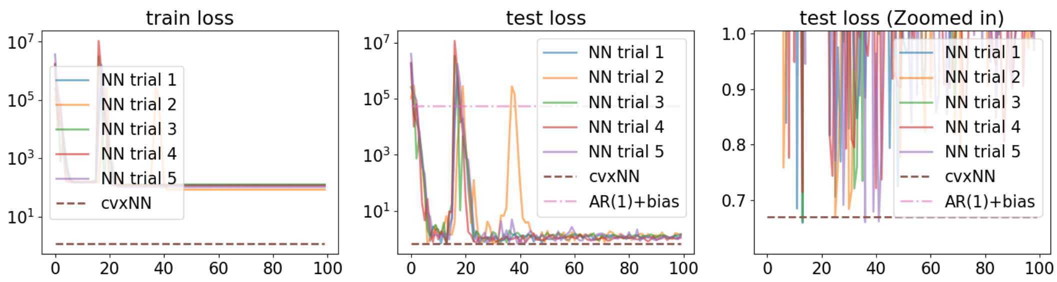
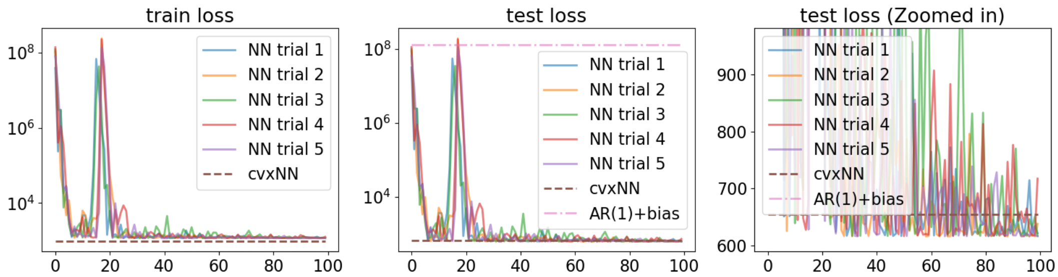


Figure H.7: The regularization path. Here, $\sigma^2 = 1$, $m = 5$.

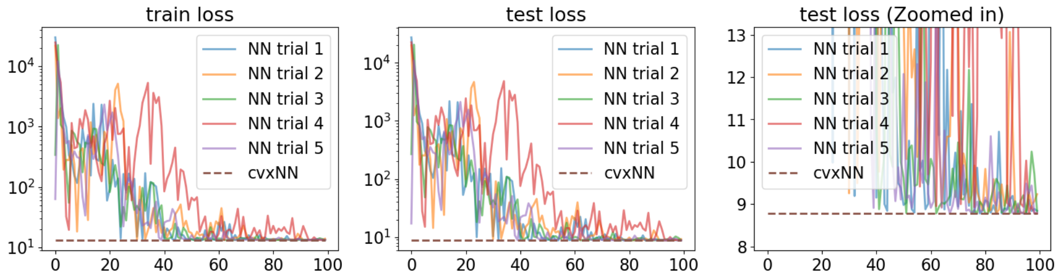


BTC-2017min.



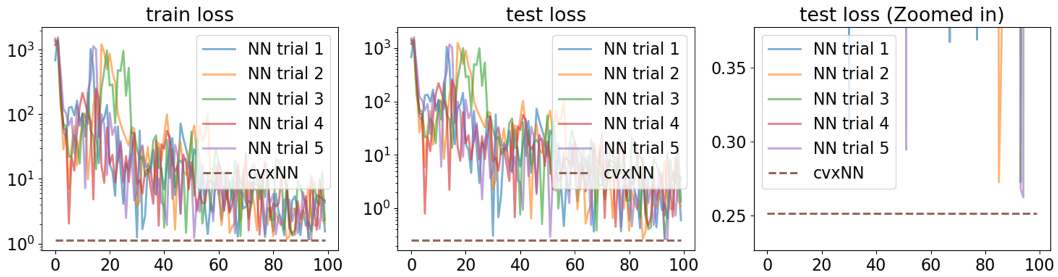
BTC-hourly.

Figure H.8: Regression with L2 loss.



BTC-hourly.

Figure H.9: Regression with quantile loss. $\tau = 0.3$



BTC-2017min.

Figure H.10: Regression with quantile loss. $\tau = 0.7$