Neural Networks 94 (2017) 34–45

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

# Piecewise convexity of artificial neural networks

Blaine Rister [a,*], Daniel L. Rubin [b]

[a] Stanford University, Department of Electrical Engineering, 1201 Welch Rd, Stanford, CA, 94305, USA
[b] Stanford University, Department of Radiology (Biomedical Informatics Research), 1201 Welch Rd Stanford, CA, 94305, USA

ARTICLE INFO

*Article history:*
Received 28 December 2016
Received in revised form 21 June 2017
Accepted 22 June 2017
Available online 3 July 2017

*Keywords:*
Convex analysis
Gradient descent
Optimization
Machine learning
Neural networks
Convergence

ABSTRACT

Although artificial neural networks have shown great promise in applications including computer vision and speech recognition, there remains considerable practical and theoretical difficulty in optimizing their parameters. The seemingly unreasonable success of gradient descent methods in minimizing these non-convex functions remains poorly understood. In this work we offer some theoretical guarantees for networks with piecewise affine activation functions, which have in recent years become the norm. We prove three main results. First, that the network is piecewise convex as a function of the input data. Second, that the network, considered as a function of the parameters in a single layer, all others held constant, is again piecewise convex. Third, that the network as a function of all its parameters is piecewise multi-convex, a generalization of biconvexity. From here we characterize the local minima and stationary points of the training objective, showing that they minimize the objective on certain subsets of the parameter space. We then analyze the performance of two optimization algorithms on multi-convex problems: gradient descent, and a method which repeatedly solves a number of convex sub-problems. We prove necessary convergence conditions for the first algorithm and both necessary and sufficient conditions for the second, after introducing regularization to the objective. Finally, we remark on the remaining difficulty of the global optimization problem. Under the squared error objective, we show that by varying the training data, a single rectifier neuron admits local minima arbitrarily far apart, both in objective value and parameter space.



## 1. Introduction

Artificial neural networks are currently considered the state of the art in applications ranging from image classification, to speech recognition and even machine translation. However, little is understood about the process by which they are trained for supervised learning tasks. It has been shown that neural networks and similar models can approximate a wide variety of functions, and various bounds have been derived for the approximation error in terms of the number of computational units (Barron, 1993). Results have been shown for a wide variety of network types, in many different function spaces (Gnecco & Sanguineti, 2008; Kainen & Kůrková, 2009; Makovoz, 1998). However, given a target function and a number of computational units, these theorems show only that there exists a neural network approximating the function within a certain error. Although they may reduce the dimensionality of the search space, these results do not imply that we can efficiently find the specific network.

In practice a network architecture is parameterized by a vector of real numbers, so that training amounts to optimizing the parameters, an active area of both practical and theoretical research. Despite considerable sensitivity to initialization and choice of hyperparameters, neural networks often achieve compelling results after optimization by gradient descent methods. Due to the nonconvexity and massive parameter space of these functions, it is poorly understood how these sub-optimal methods have proven so successful. Indeed, training a certain kind of neural network is known to be NP-Complete, making it difficult to provide any worst-case guarantees (Blum & Rivest, 1992). Much recent work has attempted to reconcile these differences between theory and practice (Kawaguchi, 2016; Soudry & Carmon, 2016).

This article attempts a modest step towards understanding the dynamics of the training procedure. We establish three main convexity results for a certain class of neural networks, which is currently the state of the art. First, that the objective is piecewise convex as a function of the input data, with parameters fixed, which corresponds to the behavior at test time. Second, that the objective is again piecewise convex as a function of the parameters of a single layer, with the input data and all other parameters held constant. Third, that the training objective function, for which all parameters are variable but the input data is fixed, is piecewise

* Corresponding author.
*E-mail addresses:* blaine@stanford.edu (B. Rister), dlrubin@stanford.edu (D.L. Rubin).

http://dx.doi.org/10.1016/j.neunet.2017.06.009
0893-6080/

multi-convex. That is, it is a continuous function which can be represented by a finite number of multi-convex functions, each active on a multi-convex parameter set. This generalizes the notion of biconvexity found in the optimization literature to piecewise functions and arbitrary index sets (Gorski, Pfeuffer, & Klamroth, 2007). To prove these results, we require two main restrictions on the definition of a neural network: that its layers are piecewise affine functions, and that its objective function is convex and continuously differentiable. Our definition includes many contemporary use cases, such as least squares or logistic regression on a convolutional neural network with rectified linear unit (ReLU) activation functions and either max- or mean-pooling. In recent years these networks have mostly supplanted the classic sigmoid type, except in the case of recurrent networks (Glorot, Bordes, & Bengio, 2011). We make no assumptions about the training data, so our results apply to the current state of the art in many practical scenarios.

Piecewise multi-convexity allows us to characterize the extrema of the training objective. As in the case of biconvex functions, stationary points and local minima are guaranteed optimality on larger sets than we would have for general smooth functions. Specifically, these points are partial minima when restricted to the relevant piece. That is, they are points for which the training objective cannot be decreased without simultaneously varying the parameters across multiple layers, or crossing the boundary into a different piece of the function. Unlike global minima, we show that partial minima are reliably found by the optimization algorithms used in current practice.

Finally, we provide some guarantees for solving general multi-convex optimization problems by various algorithms. We first analyze gradient descent, showing that every limit point is a piecewise partial minimum, excepting some boundary conditions. To prove stronger results, we define a different optimization procedure separating each parameter update into a number of convex subproblems. For this procedure, we show both necessary and sufficient conditions for convergence to a piecewise partial minimum. Interestingly, adding regularization to the training objective is all that is needed to prove sufficient conditions. Similar results have been independently established for many kinds of optimization problems, including bilinear and biconvex optimization, and in machine learning the special case of linear autoencoders (Baldi and Lu, 2012; Gorski et al., 2007; Wendell and Hurter, 1976). Our analysis extends existing results on alternating convex optimization to the case of arbitrary index sets, and general multi-convex point sets, which is needed to analyze neural networks. We admit biconvex problems, and therefore linear autoencoders, as a special case.

Despite these results, we find that it is difficult to pass from partial to global optimality. Unlike the encouraging case of linear autoencoders, we show that a single rectifier neuron, under the squared error objective, admits arbitrarily poor local minima. This suggests that much work remains to be done in understanding how sub-optimal methods can succeed with neural networks. Still, piecewise multi-convex functions are in some senses easier to minimize than the general class of smooth functions, for which none of our previous guarantees can be made. We hope that our characterization of neural networks could contribute to a better understanding of these important machine learning systems.

This article is divided into two main parts: Sections 2–4 define piecewise convexity and multi-convexity, and show that neural networks belong to this more general class of functions, while Sections 5–7 discuss minimization of piecewise multi-convex functions. More specifically, Section 2 defines the basic model of a neural network, showing that the testing function is piecewise convex. Section 3 extends these results to the training function, restricted to the parameters of a single layer. Section 4 considers the training function in full generality, showing that it is piecewise multi-convex as a function of all its parameters. Section 5 establishes necessary convergence conditions for gradient descent. Section 6 establishes stronger necessary and sufficient conditions for iterated convex optimization. Finally, Section 7 illustrates the difficulty of the global optimization problem for neural networks.

## 2. Neural networks

The field of artificial neural networks contains such a wide variety of models that it seems impossible for any one theory to apply to all of them. However, closer inspection reveals that most of these models are composed of just a few basic units, stacked in layers of arbitrary width and depth. More specifically, the state of the art consists of a handful of basic units such as rectified linear unit (ReLU), pooling and embedding layers. ReLU layers have the form $g(\boldsymbol{x}) = \max(\boldsymbol{0}, A\boldsymbol{x} + \boldsymbol{b})$ where $\boldsymbol{x} \in \mathbb{R}^n$ is the input data, $A \in \mathbb{R}^{m\times n}$ and $\boldsymbol{b} \in \mathbb{R}^m$ are the parameters, and the maximum is taken pointwise. In this notation, $m$ is the number of neurons in the layer. Pooling layers partition the input variables into subsets, taking either the maximum or mean of each subset. Embedding layers have the form $g(\boldsymbol{x}) = A\boldsymbol{x}$, with $\boldsymbol{x}$ and $A$ the same as before. We show that these layers are all examples of piecewise affine functions, allowing us to establish results applying to any of the wide variety of networks composed of them. We ultimately conclude that any such network, regardless of the number of layers, is a piecewise multiconvex function, a term which will be defined later.

This work models neural networks using continuous piecewise convex functions. These are functions for which the domain can be divided into a finite number of closed convex sets, called pieces, such that the function is convex when restricted to each piece. Continuous piecewise affine functions are defined analogously, where the restriction to each piece is an affine function. The reader is encouraged to review Appendix A, which provides technical definitions for these terms, and proves some basic algebraic results for piecewise convex and piecewise affine functions, which will be used in the remainder of the text. Sections 2–4 define a neural network in terms of training and testing functions, establishing convexity properties of these functions. The remainder of the work discusses the consequences of these properties for parameter optimization. Optimization results are proven for piecewise multiconvex functions, of which neural networks are a special case.

In supervised learning, a neural network is represented by a pair of closely-related functions, one for training and one for inference. For example, consider the familiar equation $f = (ax + b - y)^2$ with parameters $(a, b)$ and data $(x, y)$ in $\mathbb{R}^2$. During testing, we hold $(a, b)$ constant, and consider $f$ as a function of the data $(x, y)$. During training, we hold $(x, y)$ constant and consider $f$ as a function of the parameters $(a, b)$. This section concerns the testing function of a neural network, which is a map from a pair $(\boldsymbol{x}, \boldsymbol{y}) \in \mathbb{R}^{n+m}$ to an objective value in $\mathbb{R}$, where $\boldsymbol{x}$ is the input data, and $\boldsymbol{y}$ is its label. Since we are ultimately interested in training, where the labels are fixed, we will assume $\boldsymbol{y}$ is constant and describe the testing function only in terms of $\boldsymbol{x}$, in which case the function is piecewise convex. Sections 3 and 4 establish similar results for the training function, which holds $\boldsymbol{x}$ constant and instead varies the network parameters. When the function can be inferred from the context, we will simply refer to the testing or training function as the neural network. First we define the testing function explicitly.

In this work, we define a neural network as a function $f : \mathbb{R}^n \to \mathbb{R}$ having the form $f = h \circ g_N \circ g_{N-1} \circ \cdots \circ g_1$, where $h : \mathbb{R}^m \to \mathbb{R}$ is a convex continuously differentiable objective (or loss) function, and $g_1, g_2, \ldots, g_N$ are continuous piecewise affine functions constituting the $N$ layers. This definition is not as restrictive as it may seem upon first glance. For example, it is easily verified that ReLU layers are continuous piecewise affine, as we have

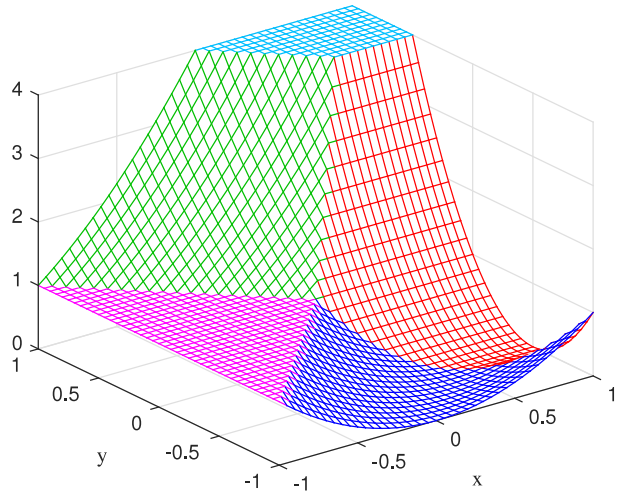


**Fig. 1.** The two-layer neural network of Eq. (1), plotted as a function of the input data, with each piece shaded in a different color. Although $f$ is not convex on $\mathbb{R}^2$, it is convex on each piece, and each piece is a convex set. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

$g(\boldsymbol{x}) = \max(\boldsymbol{0}, A\boldsymbol{x} + \boldsymbol{b})$ where the maximum is taken pointwise, and maxima of affine functions are piecewise affine (Ovchinnikov, 2002). This includes the convolutional variant, in which $A$ is composed of Toeplitz matrices. Similarly, max pooling is continuous piecewise linear, while mean pooling is simply linear. Linear embedding layers are of course linear functions. Furthermore, many of the objective functions commonly seen in machine learning are convex and continuously differentiable, as in least squares and logistic regression. Thus this seemingly restrictive class of neural networks actually encompasses the current state of the art.

By Theorem A.5, the composition of all layers $g = g_N \circ g_{N-1} \circ \cdots \circ g_1$ is continuous piecewise affine. Therefore, the testing function of a neural network is ultimately the composition of a convex function with a single continuous piecewise affine function. Thus by Theorem A.6 the network is continuous piecewise convex. Fig. 1 visualizes this result for the example network

$$f(x, y) = \left(2 - \left[[x - y]_+ - [x + y]_+ + 1\right]_+\right)^2, \tag{1}$$

where $[x]_+ = \max(x, 0)$. We can easily see that each piece is a convex polygon, and $f$ is a convex quadratic when restricted to each polygon. For clarity, this is just the two-layer ReLU network

$$f(x, y, z) = \left(z - \left[a_5[a_1x + a_2y]_+ + a_6[a_3x + a_4y]_+ + b_1\right]_+\right)^2 \tag{2}$$

with the squared error objective and a single data point $((x, y), z)$, setting $z = 2$ and $a_2 = a_6 = -1$, with all other parameters set to 1. In later sections we shall illustrate similar properties for the training function of this simple example network.

## 3. Network parameters of a single layer

In the previous section we defined the testing function of a neural network, and showed it is continuous piecewise convex. Now we extend this result to the training function, for which the data is constant and the parameters are variable. This is what we mean when we say that a network is being "considered as a function of its parameters". This leads us to an additional stipulation on our definition of a neural network. That is, each layer must be piecewise affine *as a function of its parameters* as well. This is easily verified for all of the layer types previously mentioned. For example, with the ReLU neuron has $f(A, \boldsymbol{b}) = [A\boldsymbol{x} + \boldsymbol{b}]_+$, so for $(A\boldsymbol{x} + \boldsymbol{b})_k \geq 0$ the $k$th component of $f$ is linear in $(A, \boldsymbol{b})$, while for $(A\boldsymbol{x} + \boldsymbol{b})_k \leq 0$ it is constant. To see this, we can directly compute $\alpha(A_1\boldsymbol{x} + \boldsymbol{b}_1) + \beta(A_2\boldsymbol{x} + \boldsymbol{b}_2) = (\alpha A_1 + \beta A_2)\boldsymbol{x} + \alpha\boldsymbol{b}_1 + \beta\boldsymbol{b}_2$ for any $\alpha, \beta \in \mathbb{R}$. The same is true of the convolutional variant, as $\alpha(\boldsymbol{a} * \boldsymbol{x}) + \beta(\boldsymbol{a} * \boldsymbol{x}) = (\alpha\boldsymbol{a} + \beta\boldsymbol{b}) * \boldsymbol{x}$, i.e. convolution is a linear operator.

In Section 2 we have said that a neural network, considered as a function of its input data, is piecewise convex. Now, a neural network need *not* be piecewise convex as a function of the entirety of its parameters.[1] However, we can regain piecewise convexity by restricting it to the parameters in a single layer, all others held constant.[2]

**Theorem 3.1.** *Let $f = h \circ g_N \circ g_{N-1} \circ \cdots \circ g_1$ be a neural network, where $h$ is convex, and each of the layers $g_1, g_2, \ldots, g_N$ is continuous piecewise affine as a function of either its parameters or its input. Then $f$ is continuous piecewise convex as a function of the parameters in a single layer.*

**Proof.** For the time being, assume the input data consists of a single point $\boldsymbol{x}$. Let $f_m(\boldsymbol{x})$ denote the network training objective $f$, for data point $\boldsymbol{x}$, viewed as a function of the parameters of layer $g_m$, all others held constant. Now, the layers $g_{m-1} \circ g_{m-2} \circ \cdots \circ g_1$ are constant with respect to the parameters of $g_m$, so we can write $\boldsymbol{y} = g_{m-1} \circ g_{m-2} \circ \cdots \circ g_1(\boldsymbol{x})$. Thus on each piece of $g_m$ we can write $\tilde{g}_m = g_m \circ g_{m-1} \circ \cdots \circ g_1 = A\boldsymbol{y} + \boldsymbol{b}$. Since $\boldsymbol{y}$ is constant, $\tilde{g}_m$ is a continuous piecewise affine function of $(A, \boldsymbol{b})$. Since $g_{m+1}, g_{m+2}, \ldots, g_N$ are continuous piecewise affine functions of their input, $g = g_N \circ g_{N-1} \circ \cdots \circ \tilde{g}_m$ is continuous piecewise affine by Theorem A.5. Thus by Theorem A.6, $f_m$ is continuous piecewise convex.

Having established the theorem for the case of a single data point, consider the case where we have multiple data points $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_M$. By Theorem A.7, the arithmetic mean $(1/M)\sum_{k=1}^{M} f_m(\boldsymbol{x}_k)$ is continuous piecewise convex. □

To illustrate this result, let us return to the simple two-layer neural network from Eq. (2), and plot the objective as a function of parameters $a_2$ and $a_3$ from the first layer. Setting $z = 2$ with all other variables set to 1 as before, we can simplify the expression to

$$f(a_2, a_3) = \left(1 - [a_2 + 1]_+ - [a_3 + 1]_+\right)^2 \tag{3}$$

which is plotted in Fig. 2. This function is divided into four pieces by the lines $a_2 = -1$ and $a_3 = -1$, and on each piece it is a convex quadratic. This example shows that a neural network, taken as a function of the parameters of a single layer, has the same properties as the testing function from the previous section.

We conclude this section with a simple remark which will be useful in later sections. Let $f_m$ be a neural network, considered as a function of the parameters of the $m$th layer, and let $S$ be a piece of $f_m$. Then the optimization problem

$$\begin{aligned} &\text{minimize} && f_m(\boldsymbol{x}) \\ &\text{subject to} && \boldsymbol{x} \in S \end{aligned} \tag{4}$$

is convex.

## 4. Network parameters of multiple layers

In the previous section we analyzed the convexity properties of the training function when restricted to the parameters of a single layer, all others held constant. We now extend these results

[1] To see this, consider the following two-layer network: $h(x) = x$, $g_2(x) = ax$, and $g_1(x) = bx$. For $f = h \circ g_2 \circ g_1$ we have $f(x) = abx$. Now fix the input at $x = 1$. Considered as a function of its parameters, this is $f(a, b) = ab$, which is not convex.

[2] This is made rigorous by taking cross-sections of point sets in Section 4.

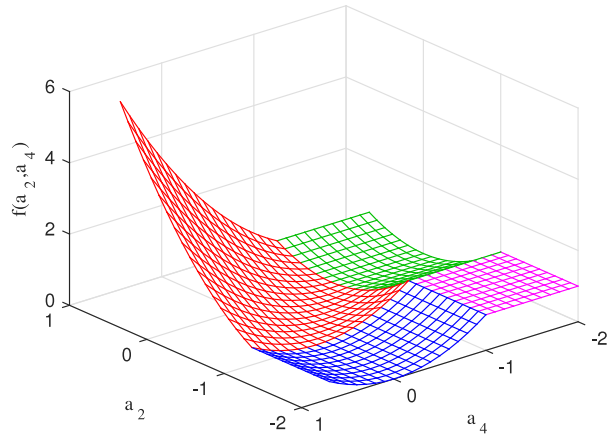


**Fig. 2.** The two-layer neural network of Eq. (3), plotted as a function of two parameters from the first layer. As in Fig. 1, the function is piecewise convex, with each piece shaded in a different color. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

to the ultimate goal of simultaneously optimizing all network parameters. Although not convex, the problem has a special convex substructure which we can exploit in proving future results. We begin by defining this substructure for point sets and functions.

**Definition 4.1.** Let $S \subseteq \mathbb{R}^n$, let $I \subset \{1, 2, \ldots, n\}$, and let $\boldsymbol{x} \in S$. The set

$$S_I(\boldsymbol{x}) = \{\boldsymbol{y} \in S : (\boldsymbol{y}_k = \boldsymbol{x}_k)_{k \notin I}\} \tag{5}$$

is the **cross-section** of $S$ intersecting $\boldsymbol{x}$ with respect to $I$.

In other words, $S_I(\boldsymbol{x})$ is the subset of $S$ for which every point is equal to $\boldsymbol{x}$ in the components not indexed by $I$. Note that this differs from the typical definition, which is the intersection of a set with a hyperplane. For example, $\mathbb{R}^3_{\{1\}}(\boldsymbol{0})$ is the $x$-axis, whereas $\mathbb{R}^3_{\{1,2\}}(\boldsymbol{0})$ is the $xy$-plane. Note also that cross-sections are not unique, for example $\mathbb{R}^3_{\{1,2\}}(0, 0, 0) = \mathbb{R}^3_{\{1,2\}}(1, 2, 0)$. In this case the first two components of the cross section are irrelevant, but we maintain them for notational convenience. We can now apply this concept to functions on $\mathbb{R}^n$.

**Definition 4.2.** Let $S \subseteq \mathbb{R}^n$, let $f : S \to \mathbb{R}$ and let $\mathcal{I}$ be a collection of sets covering $\{1, 2, \ldots, n\}$. We say that $f$ is **multi-convex** with respect to $\mathcal{I}$ if $f$ is convex when restricted to the cross section $S_I(\boldsymbol{x})$, for all $\boldsymbol{x} \in S$ and $I \in \mathcal{I}$. We say that $f$ is **multi-affine** if $f$ is affine when restricted to $S_I(\boldsymbol{x})$.

Multi-convexity formalizes the notion of restricting a non-convex function to a variable subset on which it is convex, as in Section 3 when a neural network was restricted to the parameters of a single layer. For example, let $f(x, y, z) = xy + z$, and let $I_1 = \{1, 3\}$, and $I_2 = \{2, 3\}$. Clearly $f$ is a convex function of $(x, z)$ with $y$ fixed at $y_0$, and of $(y, z)$ with $x$ fixed at $x_0$. Thus $f$ is multi-convex with respect to $\mathcal{I} = \{I_1, I_2\}$. To fully define a multi-convex optimization problem, we introduce a similar concept for point sets.

**Definition 4.3.** Let $S \subseteq \mathbb{R}^n$ and let $\mathcal{I}$ be a collection of sets covering $\{1, 2, \ldots, n\}$. We say that $S$ is **multi-convex** with respect to $\mathcal{I}$ if the cross-section $S_I(\boldsymbol{x})$ is convex for all $\boldsymbol{x} \in S$ and $I \in \mathcal{I}$.

This generalizes the notion of biconvexity found in the optimization literature (Gorski et al., 2007). From here, we can extend Definition A.1 to multi-convex functions.

**Definition 4.4.** Let $f : \mathbb{R}^n \to \mathbb{R}$ be a continuous function. We say that $f$ is **continuous piecewise multi-convex** if there exists a collection of multi-convex functions $g_1, g_2, \ldots, g_N$ defined on $\mathbb{R}^n$ and multi-convex sets $S_1, S_2, \ldots, S_N$ covering $\mathbb{R}^n$ such that for each $k \in \{1, \ldots, N\}$ we have $f(\boldsymbol{x}) = g_k(\boldsymbol{x})$ for all $\boldsymbol{x} \in S_k$. We say that $f$ is **continuous piecewise multi-affine** if $g_1, g_2, \ldots, g_N$ are multi-affine. Next, let $h : \mathbb{R}^m \to \mathbb{R}^n$. Then, $h$ is continuous piecewise multi-affine so long as each component is, as in Definition A.3.

We have assumed continuity in the previous definition so as to avoid issues concerning the topology of the pieces $S_1, S_2, \ldots, S_N$. Note that unlike Definition A.1, the multi-convex pieces need not be closed, nor even connected.

Before we can extend the results of Section 3 to multiple layers, we must add one final constraint on the definition of a neural network. That is, each of the layers must be continuous piecewise multi-affine, considered as functions of both the parameters *and* the input, with the index sets separating the parameters from the input. Again, this is easily verified for the all of the layer types previously mentioned. The only layer which merits consideration is the ReLU, which consists of two pieces for each component: the "dead" or constant region, with $(A\boldsymbol{x})_j + b_j \leq 0$, and the "alive" region $(A\boldsymbol{x})_j + b_j \geq 0$. Similarly, for the convolutional variant the pieces of each component are $(\boldsymbol{a} * \boldsymbol{x})_j + b_j \geq 0$ and $(\boldsymbol{a} * \boldsymbol{x})_j + b_j \leq 0$. With $n$ components the layer has at most $2^n$ pieces, corresponding to binary assignments of "dead" or "alive". These pieces cover the parameter and input spaces, and we know they are multi-convex since their cross-sections are convex polytopes, as in the proof of Theorem A.4.

Having said that each layer is continuous piecewise multi-convex, we can extend these results to the whole network. But first we define a notion of differentiability which will be important for the later sections on multi-convex optimization.

**Definition 4.5.** Let $f$ be piecewise continuous. We say that $f$ is **piecewise continuously differentiable** if each active function $g$ is continuously differentiable.

Finally we state our representation theorem for neural networks.

**Theorem 4.6.** *Let $f = h \circ g_N \circ g_{N-1} \circ \cdots \circ g_1$ be a neural network, where $h$ is convex and continuously differentiable, and $g_1, g_2, \ldots, g_N$ are piecewise multi-affine functions each with respect to $\{I_k, J_k\}$, where $I_k$ indexes the parameters of $g_k$ and $J_k$ indexes the input variables. Then $f$ is continuous piecewise multi-convex with respect to $\mathcal{I} = \{I_1, I_2, \ldots, I_n\}$, and piecewise continuously differentiable.*

**Proof.** The proof is similar to that of Theorem 3.1, substituting affine for multi-affine functions. Due to its length, we have deferred the proof to Appendix B. □

To visualize Theorem 4.6, we return to the simple two-layer ReLU network from Eq. (2), plotting the network as a function of parameters $a_6$ and $a_3$. Setting $z = 2$ and $a_2 = -1$, with all other variables set to 1 as before, we have

$$f(a_6, a_3) = \left(2 - \left[a_6[a_3 + 1]_+ + 1\right]_+\right)^2. \tag{6}$$

From Fig. 3, we can see that the function is divided into three pieces by the line $a_3 = -1$ and the parabola $a_6(a_3 + 1) + 1 = 0$. The red piece is not convex, but biconvex. For $a_3 \leq -1$ or $a_6(a_3+1)+1 < 0$ the function is constant at $f = 1$ or $f = 4$, respectively. On the non-constant piece, the active function is $f(a_6, a_3) = (1 - a_6(a_3 + 1))^2$. This function is not convex, but biconvex, since holding either $a_6$ or $a_3$ constant yields a convex parabola, as shown in Fig. 4. This simple example illustrates that when optimizing over multiple layers, the

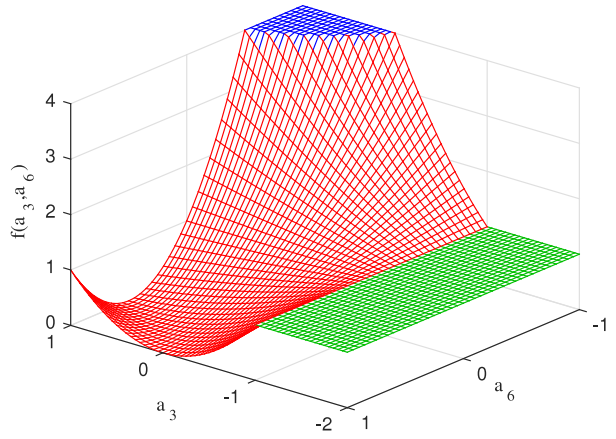


**Fig. 3.** The two-layer neural network of Eq. (6), plotted as a function of parameters from different layers. The function is piecewise biconvex, with each piece shaded in a different color. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

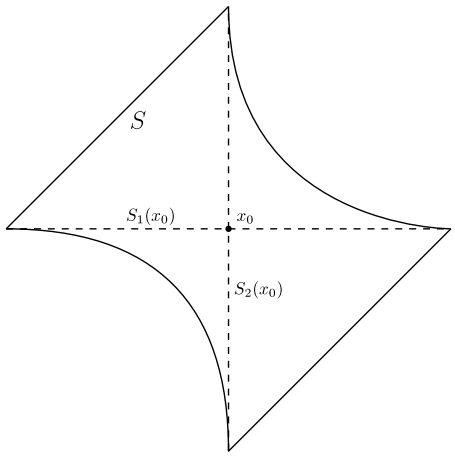


**Fig. 5.** Cross-sections of a biconvex set.

objective need not be piecewise convex, but we are still guaranteed piecewise multiconvexity.

More generally, Theorem 4.6 implies that the network from Eq. (2) is piecewise multi-convex as a function of all its parameters, with respect to the partition $\{(a_1, a_2, a_3, a_4), (a_5, a_6, b_1)\}$ which separates the parameters from each of the two layers. In other words, if we fix $(a_5, a_6, b_1) = (\alpha_1, \alpha_2, \beta)$, then $f(a_1, a_2, a_3, a_4, \alpha_1, \alpha_2, \beta)$ is piecewise convex on the cross-section $T = \{(a_1, a_2, a_3, a_4, \alpha_1, \alpha_2, \beta) : (a_1, a_2, a_3, a_4) \in \mathbb{R}^4\}$. If $S$ is a piece of $f$, then $f$ is convex on the cross-section $S \cap T$, and this is a convex set. (In fact, by the proof of Theorem 3.1 it is a closed convex polytope.) Although we cannot plot these higher-dimensional functions directly, their algebraic properties reveal their convexity.

In the coming sections, we shall see that multi-convexity allows us to establish convergence guarantees for various optimization algorithms. But first, we shall prove some basic results independent of the optimization procedure. These results were summarized by Gorski et al. for the case of biconvex differentiable functions (Gorski et al., 2007). Here we extend them to piecewise functions and arbitrary index sets. First we define a special type of minimum relevant for multi-convex functions.

**Definition 4.7.** Let $f : S \to \mathbb{R}$ and let $\mathcal{I}$ be a collection of sets covering $\{1, 2, \ldots, n\}$. We say that $\boldsymbol{x}_0$ is a **partial minimum** of $f$ with respect to $\mathcal{I}$ if $f(\boldsymbol{x}_0) \leq f(\boldsymbol{x})$ for all $\boldsymbol{x} \in \cup_{I \in \mathcal{I}} S_I(\boldsymbol{x}_0)$.

In other words, $\boldsymbol{x}_0$ is a partial minimum of $f$ with respect to $\mathcal{I}$ if it minimizes $f$ on all of its cross-sections, as shown in Fig. 5. By convexity, these points are intimately related to the stationary points of $f$.

**Theorem 4.8.** *Let $\mathcal{I} = \{I_1, I_2, \ldots, I_m\}$ be a collection of sets covering $\{1, 2, \ldots, n\}$, let $f : \mathbb{R}^n \to \mathbb{R}$ be continuous piecewise multi-convex with respect to $\mathcal{I}$, and let $\nabla f(\boldsymbol{x}_0) = \boldsymbol{0}$. Then $\boldsymbol{x}_0$ is a partial minimum of $f$ on every piece containing $\boldsymbol{x}_0$.*

**Proof.** Let $S$ be a piece of $f$ containing $\boldsymbol{x}_0$, let $I \in \mathcal{I}$, and let $S_I(\boldsymbol{x}_0)$ denote the relevant cross-section of $S$. We know $f$ is convex on $S_I(\boldsymbol{x}_0)$, and since $\nabla f(\boldsymbol{x}_0) = \boldsymbol{0}$, we know $\boldsymbol{x}_0$ minimizes $f$ on this convex set. Since this holds for all $I \in \mathcal{I}$, $\boldsymbol{x}_0$ is a partial minimum of $f$ on $S$. □

It is clear that multi-convexity provides a wealth of results concerning partial minima, while piecewise multi-convexity restricts those results to a subset of the domain. Less obvious is that partial minima of smooth multi-convex functions need not be local minima. An example was pointed out by a reviewer of this work, that the biconvex function $f(x, y) = xy$ has a partial minimum at the origin which is not a local minimum. However, the converse is easily verified, even in the absence of differentiability.

**Theorem 4.9.** *Let $\mathcal{I}$ be a collection of sets covering $\{1, 2, \ldots, n\}$, let $f : \mathbb{R}^n \to \mathbb{R}$ be continuous piecewise multi-convex with respect to $\mathcal{I}$, and let $\boldsymbol{x}_0$ be a local minimum of $f$ when restricted to some piece $S$. Then $\boldsymbol{x}_0$ is a partial minimum on $S$.*

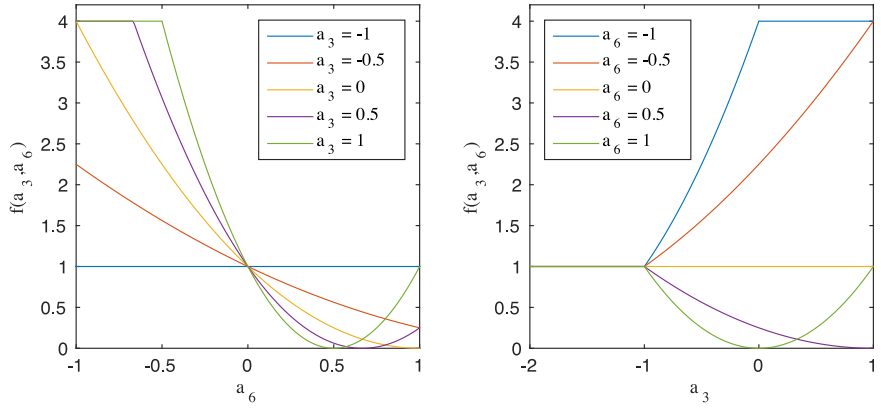


**Fig. 4.** Cross-sections from Fig. 3. The function is piecewise convex when restricted to either $a_3$ or $a_6$.

**Proof.** The proof is essentially the same as that of Theorem 4.8. □

We have seen that for multi-convex functions there is a close relationship between stationary points, local minima and partial minima. For these functions, infinitesimal results concerning derivatives and local minima can be extended to larger sets. However, we make no guarantees about global minima. The good news is that, unlike global minima, we shall see that we can easily solve for partial minima.

## 5. Gradient descent

In the realm of non-convex optimization, also called global optimization, methods can be divided into two groups: those which can certifiably find a global minimum, and those which cannot. In the former group we sacrifice speed, in the latter correctness. This work focuses on algorithms of the latter kind, called local or sub-optimal methods, as only this type is used in practice for deep neural networks. In particular, the most common methods are variants of gradient descent, where the gradient of the network with respect to its parameters is computed by a procedure called backpropagation. Since its explanation is often obscured by jargon, we provide a simple summary.

Backpropagation is nothing but the chain rule applied to the layers of a network. For the moment, assume we have only a single data point $\boldsymbol{x} \in \mathbb{R}^m$. Splitting the network into two functions $f = u \circ v$, where $u : \mathbb{R}^n \to \mathbb{R}$, and $v : \mathbb{R}^m \to \mathbb{R}^n$, we have

$$\nabla f = \nabla u \mathcal{D} v \tag{7}$$

where the row vector $\nabla u$ is multiplied by the Jacobian matrix $\mathcal{D}v$. The special observation is that we can apply this formula to each layer of the neural network, starting with the top layer $g_N$ and ending with the bottom $g_1$. To compute the gradient of $f$ with respect to the parameters of $g_m$, we take $u_m = g_N \circ g_{N-1} \circ \cdots \circ g_{m+1}$, and $v_m = g_m \circ g_{m-1} \circ \cdots \circ g_1$. Then $\nabla u_m$ is the gradient of $g_N \circ g_{N-1} \circ \cdots \circ g_{m+1}$ with respect to the output of $g_m$, while $\mathcal{D}v_m$ is the matrix of partial derivatives of $g_m \circ g_{m-1} \circ \cdots \circ g_1$ with respect to the parameters of $g_m$. Note that these partial derivatives do not depend on the form of $g_{m-1} \circ \cdots \circ g_1$, but only on the value of $g_{m-1} \circ \cdots \circ g_1(\boldsymbol{x})$. Then, for $g_{m-1}$ we can compute $\nabla u_{m-1} = \nabla u_m \mathcal{D} g_m$, where $\mathcal{D}g_m$ is the matrix of partial derivatives of $g_m$ with respect to the output of $g_{m-1}$. This is known as the "backward pass", which efficiently computes the gradient of a neural network with respect to its parameters. A similar algorithm computes $g_{m-1} \circ g_{m-2} \circ \cdots \circ g_1(\boldsymbol{x})$, which is often needed to evaluate $\mathcal{D}v_m$. First we compute and store $g_1(\boldsymbol{x})$, then $g_2 \circ g_1(\boldsymbol{x})$, and so on. This is known as the "forward pass". After one forward and one backward pass, we have computed $\nabla f$ with respect to all the network parameters, for a fixed $\boldsymbol{x}$. To generalize this procedure to multiple data points $\{\boldsymbol{x}_k\}_{k=1}^K$, we compute $\nabla f_k$ for each $\boldsymbol{x}_k$ as before, and take the arithmetic mean $\nabla f = (1/K) \sum_{k=1}^K \nabla f_k$.

Having computed $\nabla f$, we can update the parameters by gradient descent, defined as follows.

**Definition 5.1.** Let $S \subset \mathbb{R}^n$, and $f : S \to \mathbb{R}$ be partially differentiable, with $\boldsymbol{x}_0 \in S$. Then **gradient descent** on $f$ is the sequence $\{\boldsymbol{x}_k\}_{k=0}^\infty$ defined by

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \alpha_k \nabla f(\boldsymbol{x}_k) \tag{8}$$

where $\alpha_k > 0$ is called the **step size** or "learning rate". In this work we shall make the additional assumption that $\sum_{k=0}^\infty \alpha_k = \infty$.

Variants of this basic procedure are preferred in practice because their computational cost scales well with the number of network parameters. There are many different ways to choose the step size, but our assumption that $\sum_{k=0}^\infty \alpha_k = \infty$ covers what is usually done with deep neural networks. Note that we have not defined what happens if $\boldsymbol{x}_k \notin S$. Since we are ultimately interested in neural networks on $\mathbb{R}^n$, we can ignore this case and say that the sequence diverges. Gradient descent is not guaranteed to converge to a global minimum for all differentiable functions. However, it is natural to ask to which points it can converge. This brings us to a basic but important result.

**Theorem 5.2.** *Let $f : \mathbb{R}^n \to \mathbb{R}$, and let $\{\boldsymbol{x}_k\}_{k=0}^\infty$ result from gradient descent on $f$ with $\lim_{k\to\infty} \boldsymbol{x}_k = \boldsymbol{x}^*$, and $f$ continuously differentiable at $\boldsymbol{x}^*$. Then $\nabla f(\boldsymbol{x}^*) = \boldsymbol{0}$.*

**Proof.** First, we have

$$\boldsymbol{x}^* = \boldsymbol{x}_0 - \sum_{k=0}^\infty \alpha_k \nabla f(\boldsymbol{x}_k). \tag{9}$$

Assume for the sake of contradiction that for the $j$th partial derivative we have $|\partial f(\boldsymbol{x}^*)/\partial(\boldsymbol{x})_j| > 0$. Now, pick some $\varepsilon$ such that $0 < \varepsilon < |\partial f(\boldsymbol{x}^*)/\partial(\boldsymbol{x})_j|$, and by continuous differentiability, there is some $\delta > 0$ such that for all $\boldsymbol{x}$, $\|\boldsymbol{x}^* - \boldsymbol{x}\|_2 < \delta$ implies $\|\nabla f(\boldsymbol{x}^*) - \nabla f(\boldsymbol{x})\|_2 < \varepsilon$. There must be some $K$ such that for all $k \geq K$ we have $\|\boldsymbol{x}^* - \boldsymbol{x}_k\|_2 < \delta$, so that $\partial f(\boldsymbol{x}_k)/\partial(\boldsymbol{x})_j$ does not change sign. Then we can write

$$\begin{aligned}\left|\sum_{k=K}^\infty \alpha_k \frac{\partial f(\boldsymbol{x}_k)}{\partial(\boldsymbol{x})_j}\right| &= \sum_{k=K}^\infty \alpha_k \left|\frac{\partial f(\boldsymbol{x}_k)}{\partial(\boldsymbol{x})_j}\right| \\ &\geq \sum_{k=K}^\infty \alpha_k \left(\left|\frac{\partial f(\boldsymbol{x}^*)}{\partial(\boldsymbol{x})_j}\right| - \varepsilon\right) \\ &= \infty.\end{aligned}$$

But this contradicts the fact that $\boldsymbol{x}_k$ converges. Thus $\nabla f(\boldsymbol{x}^*) = \boldsymbol{0}$. □

In the convex optimization literature, this simple result is sometimes stated in connection with Zangwill's much more general convergence theorem (Iusem, 2003; Zangwill, 1969). Note, however, that unlike Zangwill we state necessary, rather than sufficient conditions for convergence. While many similar results are known, it is difficult to strictly weaken the conditions of Theorem 5.2. For example, if we relax the condition that $\alpha_k$ is not summable, and take $f(x) = x$, then $x_k$ will always converge to a non-stationary point. Similarly, if we relax the constraint that $f$ is continuously differentiable, taking $f(x) = |x|$ and $\alpha_k$ decreasing monotonically to zero, we will always converge to the origin, which is not differentiable. Furthermore, if we have $f(x) = |x|$ with $\alpha_k$ constant, then $x_k$ will not converge for almost all $x_0$. It is possible to prove much stronger necessary and sufficient conditions for gradient descent, but these results require additional assumptions about the step size policy as well as the function to be minimized, and possibly even the initialization $\boldsymbol{x}_0$ (Nesterov, 2004 pp. 32).

It is worth discussing $f(x) = |x|$ in greater detail, since this is a piecewise affine function and thus of interest in our investigation of neural networks. While we have said its only convergence point is not differentiable, it remains subdifferentiable, and convergence results are known for subgradient descent (Iusem, 2003). In this work we shall not make use of subgradients, instead considering descent on a piecewise continuously differentiable function, where the pieces are $x \leq 0$ and $x \geq 0$. Although Theorem 5.2 does not apply to a sequence converging to $x = 0$, since the function is not differentiable at this point, it still tells us that $x = 0$ is the only possible convergence point. Too see this, note that if gradient descent converged to any other point, the derivative at that point would be zero. Thus, the conclusion of Theorem 5.2 could be restated as follows: either $\nabla f(\boldsymbol{x}^*) = \boldsymbol{0}$, or $f$ is not continuously differentiable at $\boldsymbol{x}^*$.

Here we should note one way in which this analysis fails in practice. So far we have assumed the gradient $\nabla f$ is precisely known. In practice, it is often prohibitively expensive to compute the average gradient over large datasets. Instead we take random subsamples, in a procedure known as *stochastic* gradient descent. We will not analyze its properties here, as current results on the topic impose additional restrictions on the objective function and step size, or require different definitions of convergence (Bach & Moulines, 2011; Bertsekas, 2010; Ge, Huang, Jin, & Yuan, 2015). Restricting ourselves to the true gradient $\nabla f$ allows for simple proofs applying to an extensive class of neural networks.

We are now ready to generalize these results to neural networks. There is a slight ambiguity in that the boundary points between pieces need not be differentiable, nor even sub-differentiable. Since we are only proving necessary conditions, we say that gradient descent diverges when $\nabla f(\boldsymbol{x}_k)$ does not exist. This releases us from the burden of inventing an arbitrary policy when the usual one is undefined. In practice this is a very weak limitation, as it is highly unlikely that a training sequence would contain infinitely many boundary points. If a sequence $\{\boldsymbol{x}_k\}_{k=0}^{\infty}$ has only a finite number of non-differentiable points, $\boldsymbol{x}_K$ being the last one, then we can apply these theorems to the sequence starting at $\boldsymbol{x}_{K+1}$. Furthermore, our next theorem can handle non-differentiable limit points as well.

**Theorem 5.3.** *Let $\mathcal{I} = \{I_1, I_2, \ldots, I_m\}$ be a collection of sets covering $\{1, 2, \ldots, n\}$, let $f : \mathbb{R}^n \to \mathbb{R}$ be continuous piecewise multi-convex with respect to $\mathcal{I}$, and piecewise continuously differentiable. Then, let $\{\boldsymbol{x}_k\}_{k=0}^{\infty}$ result from gradient descent on $f$, with $\lim_{k\to\infty} \boldsymbol{x}_k = \boldsymbol{x}^*$. Then,*

1. *If $f$ is continuously differentiable at $\boldsymbol{x}^*$, then $\boldsymbol{x}^*$ is a partial minimum of $f$ on every piece containing $\boldsymbol{x}^*$.*
2. *If there is some piece $S$ of $f$ and some $K > 0$ such that $\boldsymbol{x}_k \in S^o$ for all $k \geq K$, then $\boldsymbol{x}^*$ is a partial minimum on $S$.*

**Proof.** If the first condition holds, then Theorem 5.2 gives $\nabla f(\boldsymbol{x}^*) = 0$, so $\boldsymbol{x}^*$ is a partial minimum by Theorem 4.8. If the second condition holds, then $\{\boldsymbol{x}_k\}_{k=K}^{\infty}$ is a convergent gradient descent sequence on $g$, the active function of $f$ on $S$. Since $g$ is continuously differentiable on $\mathbb{R}^n$, the first condition holds for $g$. Since $f|_S = g|_S$, $\boldsymbol{x}^*$ is a partial minimum of $f|_S$ as well. □

The first condition of Theorem 5.3 holds for every point in the interior of a piece, and some boundary points. The second condition extends these results to non-differentiable boundary points so long as gradient descent is eventually confined to a single piece of the function. For example, consider the continuous piecewise convex function $f(x) = \min(x, x^4)$ as shown in Fig. 6. When we converge to $x = 0$ from the piece $[0, 1]$, it is as if we were converging on the smooth function $g(x) = x^4$. This example also illustrates an important caveat regarding boundary points: although $x = 0$ is an extremum of $f$ on $[0, 1]$, it is not an extremum on $\mathbb{R}$.

## 6. Iterated convex optimization

Although the previous section contained some powerful results, Theorem 5.3 suffers from two main weaknesses, that it is a necessary condition and that it requires extra care at non-differentiable points. It is difficult to overcome these limitations with gradient descent. Instead, we shall define a different optimization technique, from which necessary and sufficient convergence results follow, regardless of differentiability.

Iterated convex optimization splits a non-convex optimization problem into a number of convex sub-problems, solving the sub-problems in each iteration. For a neural network, we have shown that the problem of optimizing the parameters of a single layer, all others held constant, is piecewise convex. Thus, restricting ourselves to a given piece yields a convex optimization problem. In this section, we show that these convex sub-problems can be solved repeatedly, converging to a piecewise partial optimum.

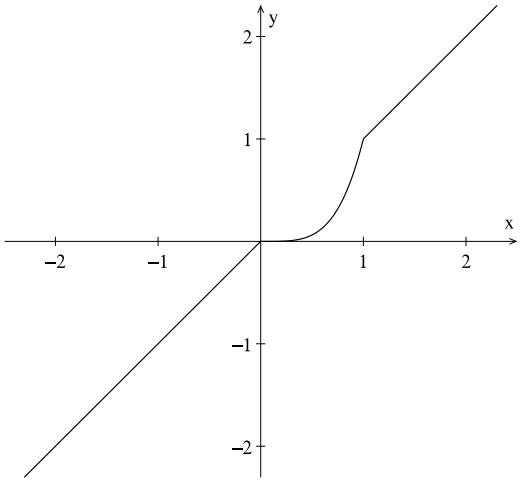


**Fig. 6.** Example of a piecewise convex function. The point $x = 0$ minimizes the function on the piece $[0, 1]$.

**Definition 6.1.** Let $\mathcal{I} = \{I_1, I_2, \ldots, I_m\}$ be a collection of sets covering $\{1, 2, \ldots, n\}$, and let $S \subseteq \mathbb{R}^n$ and $f : S \to \mathbb{R}$ be multi-convex with respect to $\mathcal{I}$. Then **iterated convex optimization** is any sequence where $\boldsymbol{x}_k$ is a solution to the optimization problem

$$\begin{aligned} &\text{minimize} && f(\boldsymbol{y}) \\ &\text{subject to} && \boldsymbol{y} \in \cup_{I\in\mathcal{I}} S_I(\boldsymbol{x}_{k-1}) \end{aligned} \tag{10}$$

with $\boldsymbol{x}_0 \in S$.

We call this iterated convex optimization because problem (10) can be divided into convex sub-problems

$$\begin{aligned} &\text{minimize} && f(\boldsymbol{y}) \\ &\text{subject to} && \boldsymbol{y} \in S_I(\boldsymbol{x}_{k-1}) \end{aligned} \tag{11}$$

for each $I \in \mathcal{I}$. In this work, we assume the convex sub-problems are solvable, without delving into specific solution techniques. Methods for alternating between solvable sub-problems have been studied by many authors, for many different types of sub-problems (Wendell & Hurter, 1976). In the context of machine learning, the same results have been developed for the special case of linear autoencoders (Baldi & Lu, 2012). Still, extra care must be taken in extending these results to arbitrary index sets. The key is that $\boldsymbol{x}_k$ is not updated until all sub-problems have been solved, so that each iteration consists of solving $m$ convex sub-problems. This is equivalent to the usual alternating convex optimization for biconvex functions, where $\mathcal{I}$ consists of two sets, but not for general multi-convex functions.

Some basic convergence results follow immediately from the solvability of problem (10). First, note that $\boldsymbol{x}_{k-1}$ is a feasible point, so we have $f(\boldsymbol{x}_k) \leq f(\boldsymbol{x}_{k-1})$. This implies that $\lim_{k\to\infty} f(\boldsymbol{x}_k)$ exists, so long as $f$ is bounded below. However, this does not imply the existence of $\lim_{k\to\infty} \boldsymbol{x}_k$. See Gorski et al. for an example of a biconvex function on which $\boldsymbol{x}_k$ diverges (Gorski et al., 2007). To prove stronger convergence results, we introduce regularization to the objective.

**Theorem 6.2.** *Let $\mathcal{I}$ be a collection of sets covering $\{1, 2, \ldots, n\}$, and let $S \subseteq \mathbb{R}^n$ and $f : S \to \mathbb{R}$ be multi-convex with respect to $\mathcal{I}$, such that $\inf f > -\infty$. Next, let $g(\boldsymbol{x}) = f(\boldsymbol{x}) + \lambda h(\|\boldsymbol{x}\|)$, where $\lambda > 0$, $h : \mathbb{R} \to \mathbb{R}$ is convex and strictly increasing, and $\|\boldsymbol{x}\|$ is any norm.*

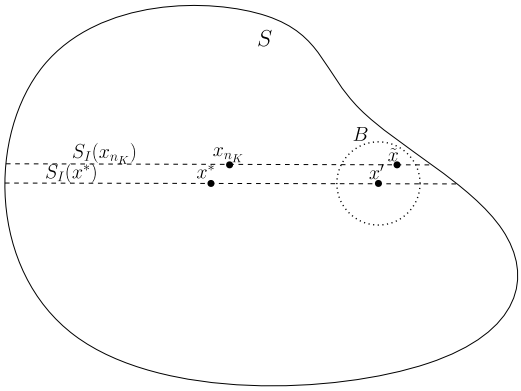


**Fig. 7.** Illustration of the proof of Theorem 6.3. Note the cross-sections of the biconvex set $S$.

*Finally, let $\{\boldsymbol{x}_k\}_{k=0}^{\infty}$ result from iterated convex optimization of $g$. Then $\boldsymbol{x}_k$ has at least one convergent subsequence, in the norm topology.*

**Proof.** First, $h(\|\boldsymbol{x}\|)$ is convex since both $h$ and $\|\boldsymbol{x}\|$ are convex, and $h$ is increasing (Rockafellar, 1970 pp. 32). Then from Lemma A.8, $g$ is multi-convex, so we are allowed to use iterated convex optimization. Furthermore, $h$ is injective so it has a left inverse $h^{-1}$. Now, if $\inf f + \lambda h(\|\boldsymbol{x}\|) > g(\boldsymbol{x}_0)$ we have that $g(\boldsymbol{x}) > g(\boldsymbol{x}_0)$. Let $M = h^{-1}\left((g(\boldsymbol{x}_0) - \inf f)/\lambda\right)$, so that $g(\boldsymbol{x}) > g(\boldsymbol{x}_0)$ whenever $\|\boldsymbol{x}\| > M$. Since $g(\boldsymbol{x}_k)$ is a non-increasing sequence, we have that $\|\boldsymbol{x}_k\| \leq M$. Then, by the Bolzano–Weierstrauss theorem, $\boldsymbol{x}_k$ has at least one convergent subsequence (Johnsonbaugh & Pfaffenberger, 1981 pp. 151). □

In Theorem 6.2, the function $g$ is called the **regularized** version of $f$. Of course the identity function suffices for $h$, but a nontrivial $h$ is often composed with a norm for computational convenience, for example raising an $\ell^p$ norm to the $p$th power, with $p \in [1, \infty)$. In practice, regularization often makes a non-convex optimization problem easier to solve, and can reduce over-fitting. The theorem shows that iterated convex optimization on a regularized function always has at least one convergent subsequence. Next, we shall establish some rather strong properties of the limits of these subsequences.

**Theorem 6.3.** *Let $\mathcal{I}$ be a collection of sets covering $\{1, 2, \ldots, n\}$, and let $S \subseteq \mathbb{R}^n$ and $f : S \to \mathbb{R}$ be multi-convex with respect to $\mathcal{I}$. Next, let $\{\boldsymbol{x}_k\}_{k=0}^{\infty}$ result from iterated convex optimization of $f$. Then the limit of every convergent subsequence is a partial minimum on $S^o$ with respect to $\mathcal{I}$, in the topology induced by some norm $\|\boldsymbol{x}\|$. Furthermore, if $\{\boldsymbol{x}_{m_k}\}_{k=1}^{\infty}$ and $\{\boldsymbol{x}_{n_k}\}_{k=1}^{\infty}$ are convergent subsequences, then $\lim_{k\to\infty} f(\boldsymbol{x}_{m_k}) = \lim_{k\to\infty} f(\boldsymbol{x}_{n_k})$.*

**Proof.** Let $\boldsymbol{x}_{n_k}$ denote a subsequence of $\boldsymbol{x}_k$ with $\boldsymbol{x}^* = \lim_{n\to\infty} \boldsymbol{x}_{n_k}$. Now, assume for the sake of contradiction that $\boldsymbol{x}^*$ is not a partial minimum on $S^o$ with respect to $\mathcal{I}$. Then there is some $I \in \mathcal{I}$ and some $\boldsymbol{x}' \in S_I(\boldsymbol{x}^*) \cap S^o$ such that $f(\boldsymbol{x}') < f(\boldsymbol{x}^*)$. As in the convex case, it can be shown that $f$ is continuous on $S^o$ (Aumann & Hart, 1986). Then there must be some $\delta > 0$ such that for all $\boldsymbol{x} \in S$, $\|\boldsymbol{x} - \boldsymbol{x}'\| < \delta$ implies $|f(\boldsymbol{x}) - f(\boldsymbol{x}')| < f(\boldsymbol{x}^*) - f(\boldsymbol{x}')$. Since $\boldsymbol{x}'$ is an interior point, there must be some open ball $B \subset S$ of radius $r$ centered at $\boldsymbol{x}'$, as shown in Fig. 7. Now, there must be some $K$ such that $\|\boldsymbol{x}_{n_K} - \boldsymbol{x}^*\| < \min(\delta, r)$. Then, let $\tilde{\boldsymbol{x}} = \boldsymbol{x}_{n_K} + \boldsymbol{x}' - \boldsymbol{x}^*$, and since $\|\tilde{\boldsymbol{x}} - \boldsymbol{x}'\| < r$, we know that $\tilde{\boldsymbol{x}} \in B$, and thus $\tilde{\boldsymbol{x}} \in S_I(\boldsymbol{x}_{n_K})$. Finally, $\|\tilde{\boldsymbol{x}} - \boldsymbol{x}'\| < \delta$, so we have $f(\tilde{\boldsymbol{x}}) < f(\boldsymbol{x}^*) \leq f(\boldsymbol{x}_{n_K+1})$, contradicting the fact that $\boldsymbol{x}_{n_K+1}$ minimizes $g$ over a set containing $\tilde{\boldsymbol{x}}$. Thus $\boldsymbol{x}^*$ is a partial minimum on $S^o$ with respect to $\mathcal{I}$.

Finally, let $\{\boldsymbol{x}_{m_k}\}_{k=1}^{\infty}$ and $\{\boldsymbol{x}_{n_k}\}_{k=1}^{\infty}$ be two convergent subsequences of $\boldsymbol{x}_k$, with $\lim_{k\to\infty} \boldsymbol{x}_{m_k} = \boldsymbol{x}_m^*$ and $\lim_{k\to\infty} \boldsymbol{x}_{n_k} = \boldsymbol{x}_n^*$, and assume for the sake of contradiction that $f(\boldsymbol{x}_m^*) > f(\boldsymbol{x}_n^*)$. Then by continuity, there is some $K$ such that $f(\boldsymbol{x}_{n_K}) < f(\boldsymbol{x}_m^*)$. But this contradicts the fact that $f(\boldsymbol{x}_k)$ is non-increasing. Thus $f(\boldsymbol{x}_m^*) = f(\boldsymbol{x}_n^*)$. □

The previous theorem is an extension of results reviewed in Gorski et al. to arbitrary index sets (Gorski et al., 2007). While Gorski et al. explicitly constrain the domain to a compact biconvex set, we show that regularization guarantees $\boldsymbol{x}_k$ cannot escape a compact set, establishing the necessary condition for convergence. In the context of neural networks, $f$ can be interpreted as a regularized version of the training objective, so that a convergent subsequence is guaranteed to exist. Furthermore, our results hold for general multi-convex sets, while the earlier result is restricted to Cartesian products of compact sets. Optimality is restricted to the interior of $S$ so that we can guarantee $\tilde{\boldsymbol{x}} \in S$, as seen in Fig. 7. In case $S$ is not $n$-dimensional, the result could be improved to the interior relative to the multi-affine hull of $S$ (Aumann & Hart, 1986).

These results for iterated convex optimization are considerably stronger than what we have shown for gradient descent. While any bounded sequence in $\mathbb{R}^n$ has a convergent subsequence, and we can guarantee boundedness for some variants of gradient descent, we cannot normally say much about the limits of subsequences. For iterated convex optimization, we have shown that the limit of any subsequence is a partial minimum, and all limits of subsequences are equal in objective value. For all practical purposes, this is just as good as saying that the original sequence converges to a partial minimum.

## 7. Global optimization

Although we have provided necessary and sufficient conditions for convergence of various optimization algorithms on neural networks, the limit points need only minimize the objective on cross-sections of pieces of the domain. Of course we would prefer results relating the limit points to global minima of the training objective. In this section we illustrate the difficulty of establishing such results, even for the simplest of neural networks.

In recent years much work has been devoted to providing theoretical explanations for the empirical success of deep neural networks, a full accounting of which is beyond the scope of this article. In order to simplify the problem, many authors have studied *linear* neural networks, in which the layers have the form $g(\boldsymbol{x}) = A\boldsymbol{x}$, where $A$ is the parameter matrix. With multiple layers this is clearly a linear function of the input, but not of the parameters. As a trivial case of piecewise affine functions, our previous results suffice to show that these networks are multi-convex as functions of their parameters. This was proven for the special case of linear autoencoders by Baldi and Lu (2012).

Many authors have claimed that linear neural networks contain no "bad" local minima, i.e. every local minimum is a global minimum (Kawaguchi, 2016; Soudry & Carmon, 2016). This is especially evident in the study of linear autoencoders, which were shown to admit many points of inflection, but only a single strict minimum (Baldi & Lu, 2012). While powerful, this claim does not apply to the networks seen in practice. To see this, consider the dataset $D = \{(0, 1/2), (-1, \alpha), (1, 2\alpha)\}$ consisting of three $(x, y)$ pairs, parameterized by $\alpha > 1$. Note that the dataset has zero mean and unit variance in the $x$ variable, which is common practice in machine learning. However, we do not take zero mean in the $y$ variable, as the model we shall adopt is non-negative.

Next, consider the simple neural network

$$f(a,b) = \sum_{(x,y)\in D} \left(y - [ax+b]_+\right)^2 \tag{12}$$
$$= \left(\frac{1}{2} - [b]_+\right)^2 + \left(\alpha - [b-a]_+\right)^2 + \left(2\alpha - [b+a]_+\right)^2.$$

This is the squared error of a single ReLU neuron, parameterized by $(a, b) \in \mathbb{R}^2$. We have chosen this simplest of all networks because we can solve for the local minima in closed form, and show they are indeed very bad. First, note that $f$ is a continuous piecewise convex function of six pieces, realized by dividing the plane along the line $ax + b = 0$ for each $x \in D$, as shown in Fig. 8. Now, for all but one of the pieces, the ReLU is "dead" for at least one of the three data points, i.e. $ax + b < 0$. On these pieces, at least one of the three terms of Eq. (12) is constant. The remaining terms are minimized when $y = ax + b$, represented by the three dashed lines in Fig. 8. There are exactly three points where two of these lines intersect, and we can easily show that two of them are strict local minima. Specifically, the point $(a_1, b_1) = (1/2 - \alpha, 1/2)$ minimizes the first two terms of Eq. (12), while $(a_2, b_2) = (2\alpha - 1/2, 1/2)$ minimizes the first and the last term. In each case, the remaining term is constant on the piece containing the point of intersection. Thus these points are strict global minima on their respective pieces, and strict local minima on $\mathbb{R}^2$. Furthermore, we can compute $f(a_1, b_1) = 4\alpha^2$ and $f(a_2, b_2) = \alpha^2$. This gives

$$\lim_{\alpha\to\infty} a_1 = -\infty, \qquad \lim_{\alpha\to\infty} a_2 = \infty, \tag{13}$$

and

$$\lim_{\alpha\to\infty} \left(f(a_1, b_1) - f(a_2, b_2)\right) = \infty. \tag{14}$$

Now, it might be objected that we are not permitted to take $\alpha \to \infty$ if we require that the $y$ variable has unit variance. However, these same limits can be achieved with variance tending to zero by adding $\lfloor \alpha \rfloor$ instances of the point $(1, 2\alpha)$ to our dataset. Thus even under fairly stringent requirements we can construct a dataset yielding arbitrarily bad local minima, both in the parameter space and the objective value. This provides some weak justification for the empirical observation that success in deep learning depends greatly on the data at hand.

We have shown that the results concerning local minima in linear networks do not extend to the nonlinear case. This is not surprising, as with linear networks the problem can be relaxed to linear regression on a convex objective. That is, the composition of all linear layers $g(\boldsymbol{x}) = A_1 A_2 \ldots A_n \boldsymbol{x}$ is equivalent to the function $f(\boldsymbol{x}) = A\boldsymbol{x}$ for some matrix $A$. Consider the relaxed problem of finding the optimal matrix $A$. Under our previous assumptions this is a convex optimization problem. Furthermore, it is easily shown that the number of parameters in the relaxed problem is polynomial in the number of original parameters. Since the relaxed problem fits the data at least as well as the original, it is not surprising that the original problem is computationally tractable.

This simple example was merely meant to illustrate the difficulty of establishing results for *every* local minimum of *every* neural network. Since training a certain kind of network is known to be NP-Complete, it is difficult to give any guarantees about worst-case global behavior (Blum & Rivest, 1992). We have made no claims, however, about probabilistic behavior on the average practical dataset, nor have we ruled out the effects of more specialized networks, such as very deep ones.

## 8. Conclusion

We showed that a common class of neural networks is piecewise convex in each layer, with all other parameters fixed. We extended this to a theory of a piecewise multi-convex functions, showing that the training objective function can be represented by a finite number of multi-convex functions, each active on a multi-convex parameter set. From here we derived various results concerning the extrema and stationary points of piecewise multi-convex functions. We established convergence conditions for both gradient descent and iterated convex optimization on this class of functions, showing they converge to piecewise partial minima. Similar results are likely to hold for a variety of other optimization algorithms, especially those guaranteed to converge at stationary points or local minima.

We have witnessed the utility of multi-convexity in proving convergence results for various optimization algorithms. However, this property may be of practical use as well. Better understanding of the training objective could lead to the development of faster or more reliable optimization methods, heuristic or otherwise. These results may provide some insight into the practical success of sub-optimal algorithms on neural networks. However, we have also seen that local optimality results do not extend to global optimality as they do for linear autoencoders. Clearly there is much left to discover about how, or even if we can optimize deep, nonlinear neural networks.

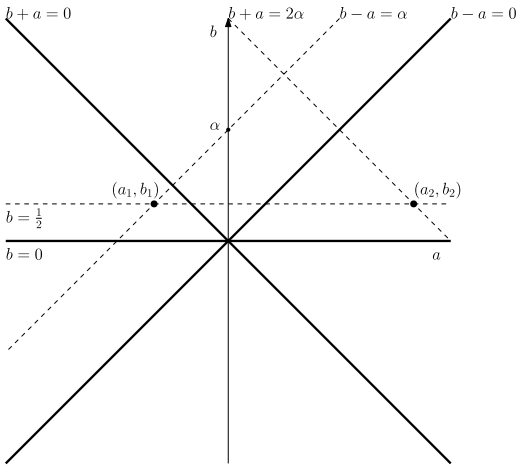


**Fig. 8.** Parameter space of the single neuron from Eq. (12), with pieces divided by the bold black lines. The dashed lines each minimize a term of Eq. (12). The points $(a_1, b_1)$ and $(a_2, b_2)$ are local minima, which can be made arbitrarily far apart by varying the dataset.

## Acknowledgment

The authors would like to thank Mihir Mongia for his helpful comments in preparing this article.

**Funding**

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

## Appendix A. Piecewise continuous functions

This appendix establishes some basic results concerning piecewise affine and piecewise convex functions, which are referred to in the main body of the text.

**Definition A.1.** Let $g_1, g_2, \ldots, g_N$ be continuous functions from $\mathbb{R}^n \to \mathbb{R}$. A **continuous piecewise** function $f$ has a finite number of closed, connected sets $S_1, S_2, \ldots, S_N$ covering $\mathbb{R}^n$ such that for each $k$ we have $f(\boldsymbol{x}) = g_k(\boldsymbol{x})$ for all $\boldsymbol{x} \in S_k$. The set $S_k$ is called a **piece** of $f$, and the function $g_k$ is called **active** on $S_k$.

More specific definitions follow from restricting the functions $g$. A **continuous piecewise affine** function has $g_k(\boldsymbol{x}) = \boldsymbol{a}^T\boldsymbol{x}+b$ where $\boldsymbol{a} \in \mathbb{R}^n$ and $b \in \mathbb{R}$. A **continuous piecewise convex** function has $g_k$ convex, with $S_k$ convex as well.

Note that this definition of piecewise convexity differs from that found in the convex optimization literature, which focuses on *convex* piecewise convex functions, i.e. maxima of convex functions (Tsevendorj, 2001). Note also that we do not claim a unique representation in terms of active functions $g_k$ and pieces $S_k$, only that there exists at least one such representation. In practice we can often choose $S_1, S_2, \ldots, S_n$ as the closures of the connected components on which $f$ is differentiable, in which case the pieces are almost disjoint. However, for simplicity and generality we do not require this choice. An elementary observation is that every continuous piecewise function is continuous, for which we offer a short proof.

**Theorem A.2.** *Every continuous piecewise function is continuous.*

**Proof.** Let $f$ be continuous piecewise with pieces $S_1, S_2, \ldots, S_N$ and active functions $f_1, f_2, \ldots, f_N$. Since each $f_k$ is continuous and $f = f_k$ on $S_k$, we know that $f$ is continuous on the interior of $S_k$. Let $x$ be a point on the boundary between pieces $S_1, \ldots, S_n$, and fix $\varepsilon > 0$. Since the pieces are closed, we know $x$ is a member of each of them. Then for each $k$ there is some $\delta_k$ such that $d(x, y) < \delta_k$ implies $|f_k(x) - f_k(y)| < \varepsilon$, where $d$ is the metric for which $x$ is continuous. Then let $\delta = \min_k \delta_k$, so that $d(x, y) < \delta$ implies $|f(x) - f(y)| < \varepsilon$. Thus $f$ is continuous at $x$. □

Note that this proof actually works for the more general case where the active functions are only defined on their respective pieces, rather than all of $\mathbb{R}^n$.

Before proceeding, we shall extend Definition A.1 to functions of multidimensional codomain for the affine case.

**Definition A.3.** A function $f : \mathbb{R}^m \to \mathbb{R}^n$, and let $f_k : \mathbb{R}^m \to \mathbb{R}$ denote the $k$th component of $f$. Then $f$ is **continuous piecewise affine** if each $f_k$ is. Take a collection of pieces $S_1, S_2, \ldots, S_n$ with non-empty intersection, one from each component $f_1, f_2, \ldots, f_n$. Then $S = \cap_{k=1}^n S_k$ is a piece of $f$, on which we have $f(\boldsymbol{x}) = A\boldsymbol{x} + \boldsymbol{b}$ for some $A \in \mathbb{R}^{n\times m}$ and $\boldsymbol{b} \in \mathbb{R}^n$.

First, we prove an intuitive statement about the geometry of the pieces of continuous piecewise affine functions.

**Theorem A.4.** *Let $f : \mathbb{R}^m \to \mathbb{R}^n$ be continuous piecewise affine. Then $f$ admits a representation in which every piece is a convex polytope.*

**Proof.** Let $f_k : \mathbb{R}^m \to \mathbb{R}$ denote the $k$th component of $f$. Now, $f_k$ can be written in closed form as a max–min polynomial (Ovchinnikov, 2002). That is, there is some partitioning of the active functions $g_1, g_2, \ldots, g_N$ by index sets $I_1, I_2, \ldots, I_m$ such that we can write

$$f(\boldsymbol{x}) = \max_{j\in\{1,\ldots,m\}} \min_{k\in I_j} g_k(\boldsymbol{x}). \tag{A.1}$$

Now, for the minimum of two affine functions we have

$$\min(g_i, g_j) = \min(\boldsymbol{a}_i^T\boldsymbol{x} + b_i, \boldsymbol{a}_j^T\boldsymbol{x} + b_j). \tag{A.2}$$

This function has two pieces divided by the hyperplane $(\boldsymbol{a}_i - \boldsymbol{a}_j)^T\boldsymbol{x} + b_i - b_j = 0$. The same can be said of $\max(g_i, g_j)$. Thus the pieces of $f_k$ are intersections of half-spaces, which are just convex polytopes. Since the pieces of $f$ are intersections of the pieces of $f_1, f_2, \ldots, f_n$, they are convex polytopes as well. □

See Fig. 8 in Section 7 for an example of this result on a specific neural network. Our next result concerns the composition of piecewise functions, which is essential for the later sections.

**Theorem A.5.** *Let $g : \mathbb{R}^m \to \mathbb{R}^n$ and $f : \mathbb{R}^n \to \mathbb{R}$ be continuous piecewise affine. Then so is $f \circ g$.*

**Proof.** To establish continuity, note that the composition of continuous functions is continuous.

By Theorem A.4, we can represent $g$ and $f$ by collections of convex polytopes $S_1, S_2, \ldots, S_p$ and $T_1, T_2, \ldots, T_q$, respectively. Let $S$ be a piece of $g$ and $T$ a piece of $f$ such that $S \cap g^{-1}(T) \neq \emptyset$, where $g^{-1}(T)$ denotes the inverse image of $T$. Since $g$ is affine, $g^{-1}(T)$ is closed and convex (Rockafellar, 1970 pp. 19). Thus $S \cap g^{-1}(T)$ is a closed, convex set on which we can write

$$\begin{aligned} f(\boldsymbol{x}) &= \boldsymbol{a}^T\boldsymbol{x} + b \\ g(\boldsymbol{x}) &= C\boldsymbol{x} + \boldsymbol{d}. \end{aligned} \tag{A.3}$$

Thus

$$f \circ g(\boldsymbol{x}) = \boldsymbol{a}^T C\boldsymbol{x} + \boldsymbol{a}^T\boldsymbol{d} + b \tag{A.4}$$

which is an affine function.

Now, consider the finite set of all such pieces $S \cap g^{-1}(T)$. The union of $g^{-1}(T)$ over all pieces $T_1, T_2, \ldots, T_q$ is just $\mathbb{R}^n$, as is the union of all pieces $S_1, S_2, \ldots, S_p$. Thus we have

$$\begin{aligned} \cup_{i=1}^p \cup_{j=1}^q \left(S_i \cap g^{-1}(T_j)\right) &= \cup_{i=1}^p \left(S_i \cap \cup_{j=1}^q g^{-1}(T_j)\right) \\ &= \cup_{i=1}^p \left(S_i \cap \mathbb{R}^n\right) \\ &= \mathbb{R}^n. \end{aligned}$$

Thus $f \circ g$ is piecewise affine on $\mathbb{R}^n$. □

We now turn to continuous piecewise convex functions, of which continuous piecewise affine functions are a subset.

**Theorem A.6.** *Let $g : \mathbb{R}^m \to \mathbb{R}^n$ be a continuous piecewise affine function, and $h : \mathbb{R}^n \to \mathbb{R}$ a convex function. Then $f = h \circ g$ is continuous piecewise convex.*

**Proof.** By Theorem A.4, we can choose the pieces of $g$ to be convex. On each piece there is some pair $(A, \boldsymbol{b})$ such that $f(\boldsymbol{x}) = h(A\boldsymbol{x} + \boldsymbol{b})$. This function is convex, as it is the composition of a convex and an affine function (Rockafellar, 1970 pp. 38). Thus $f$ is continuous piecewise convex with the same pieces as $g$. □

Our final theorem concerns the arithmetic mean of continuous piecewise convex functions, which is essential for the analysis of neural networks.

**Theorem A.7.** *Let $f_1, f_2, \ldots, f_N$ be continuous piecewise convex functions. Then so is their arithmetic mean $(1/N)\sum_{i=1}^N f_i(\boldsymbol{x})$.*

The proof takes the form of two lemmas.

**Lemma A.8.** *Let $f_1$ and $f_2$ be a pair of continuous piecewise convex functions on $\mathbb{R}^n$. Then so is $f_1 + f_2$.*

**Proof.** Let $S_1$ be a piece of $f_1$, and $S_2$ a piece of $f_2$, with $S_1 \cap S_2 \neq \emptyset$. Note that the sum of convex functions is convex (Rockafellar, 1970 pp. 33). Thus $f_1 + f_2$ is convex on $S_1 \cap S_2$. Furthermore, $S_1 \cap S_2$ is convex because it is an intersection of convex sets (Rockafellar, 1970 pp. 18). Since this holds for all pieces of $f_1$ and $f_2$, we have that $f_1 + f_2$ is continuous piecewise convex on $\mathbb{R}^n$. □

**Lemma A.9.** *Let $\alpha > 0$, and let $f$ be a continuous piecewise convex function. Then so is $\alpha f$.*

**Proof.** The continuous function $\alpha f$ is convex on every piece of $f$. □

Having established that continuous piecewise convexity is closed under addition and positive scalar multiplication, we can see that it is closed under the arithmetic mean, which is just the composition of these two operations. Note that this result holds only for *finite* sums of piecewise convex functions. For example, the sawtooth function on $\mathbb{R}$ can be written as a countably infinite sum of piecewise affine functions, but it is not piecewise convex, since it would require infinitely many pieces. This prevents us from directly extending these results to infinite neural networks, which have been represented in the literature by integral operators (Kainen & Kůrková, 2009). Integral operators could be used directly if the functions were convex, rather than piecewise convex (Gnecco & Sanguineti, 2008). However, we do not require infinite sums to analyze networks with finitely many neurons, which are always seen in practice.

Unfortunately this theorem does not extend to some of the other functions commonly referred to as "means". For example, the geometric mean of convex functions need not be convex. To see this, let $f(x, y) = x$ and $g(x, y) = y$, both defined on the positive orthant, so that $\sqrt{fg}(x, y) = \sqrt{xy}$ is concave. However, the arithmetic mean suffices for our analysis.

## Appendix B. Proof of Theorem 4.6

We begin the proof with a lemma for multi-affine functions serving a similar purpose as Theorem A.5.

**Lemma B.1.** *Let $X = \mathbb{R}^n$, $Y = \mathbb{R}^m$ and $Z = \mathbb{R}^p$, and let $g : X \to Z$ and $f : Z \times Y \to \mathbb{R}^q$ be continuous piecewise multi-affine, $g$ with respect to a collection of index sets $\mathcal{G}$, and $f$ with respect to $\mathcal{F} = \{I_Z, I_Y\}$, where $I_Z$ indexes the variables in $Z$, and $I_Y$ the variables in $Y$. Then $h(\boldsymbol{x}, \boldsymbol{y}) = f(g(\boldsymbol{x}), \boldsymbol{y})$ is continuous piecewise multi-affine with respect to $\mathcal{H} = \mathcal{G} \cup \{I_Y\}$.*

**Proof.** Let $G$ be a piece of $g$, let $F$ be a piece of $f$ and let $H = \{(\boldsymbol{x}, \boldsymbol{y}) : \boldsymbol{x} \in G, (g(\boldsymbol{x}), \boldsymbol{y}) \in F\}$, with $F$ chosen so that $H \neq \emptyset$. Next, let $\pi_X : X \times Y \to X$, $\pi_Y : Z \times Y \to Y$ and $\pi_Z : Z \times Y \to Z$ denote the usual projections, e.g. $\pi_X(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{x}$. Finally, let $(\boldsymbol{x}, \boldsymbol{y}) \in H$ and we shall show that the cross-sections are convex. First, $H_{I_Y}(\boldsymbol{x}, \boldsymbol{y}) = \{\boldsymbol{x}\} \times \pi_Y F_{I_Y}(g(\boldsymbol{x}), \boldsymbol{y})$. Note that $\pi_Y F_{I_Y}(g(\boldsymbol{x}), \boldsymbol{y})$ is convex since $F_{I_Y}(g(\boldsymbol{x}), \boldsymbol{y})$ is. Then $H_{I_Y}(\boldsymbol{x}, \boldsymbol{y})$ is a Cartesian product of convex sets, and thus convex.

Next, for any $I_X \in \mathcal{G}$ and $\boldsymbol{z} \in Z$, we have

$$H_{I_X}(\boldsymbol{x}, \boldsymbol{y}) = \{\tilde{\boldsymbol{x}} : \tilde{\boldsymbol{x}} \in G_{I_X}(\boldsymbol{x}),\ g(\tilde{\boldsymbol{x}}) \in \pi_Z F_{I_Z}(\boldsymbol{z}, \boldsymbol{y})\} \times \{\boldsymbol{y}\}. \tag{B.1}$$

Note that the choice of $\boldsymbol{z}$ does not matter for $F_{I_Z}(\boldsymbol{z}, \boldsymbol{y})$. Next, we can write

$$S = g\left(G_{I_X}(\boldsymbol{x})\right) \cap \pi_Z F_{I_Z}(\boldsymbol{z}, \boldsymbol{y})$$
$$H_{I_X}(\boldsymbol{x}, \boldsymbol{y}) = \left(G_{I_X}(\boldsymbol{x}) \cap g^{-1}(S)\right) \times \{\boldsymbol{y}\}.$$

Now, convexity is preserved under Cartesian products and intersections with the convex sets $\{\boldsymbol{y}\}$, $G_{I_X}(\boldsymbol{x})$ and $\pi_Z F_{I_Z}(\boldsymbol{z}, \boldsymbol{y})$. Since $g$ is affine on $G_{I_X}(\boldsymbol{x})$ and $S \subseteq g(G_{I_X}(\boldsymbol{x}))$, we know $g(G_{I_X}(\boldsymbol{x}))$ and $g^{-1}(S)$ are convex, as they are an image and a preimage of an affine function on a convex set (Rockafellar, 1970 pp. 19). We need to take the intersection with $G_{I_X}(\boldsymbol{x})$ since $g$ might not be injective, for example if $g = \boldsymbol{0}$ then $g^{-1}(S) = X$. Since all these operations preserve convexity, $H_{I_X}(\boldsymbol{x}, \boldsymbol{y})$ is convex.

Finally, $h$ is affine on $H_{I_Y}(\boldsymbol{x}, \boldsymbol{y})$ since $f$ is affine on $F_{I_Y}(g(\boldsymbol{x}), \boldsymbol{y})$, with $g$ constant on $\pi_X H_{I_Y}(\boldsymbol{x}, \boldsymbol{y}) = \{\boldsymbol{x}\}$. Similarly, $h$ is affine on $H_{I_X}(\boldsymbol{x}, \boldsymbol{y})$ since $g$ is affine on $G_{I_X}(\boldsymbol{x})$ and $f$ is affine on $F_{I_Z}(\boldsymbol{z}, \boldsymbol{y})$. Thus $h$ is multi-affine on $H$ with respect to $\mathcal{H}$. Finally, as in the proof of Theorem A.5, we can cover $X \times Y$ with the finite collection of all such pieces $H$, taken over all such $G$ and $F$. □

Our next lemma extends Theorem A.7 to multi-convex functions.

**Lemma B.2.** *Let $\mathcal{I}$ be a collection of sets covering $\{1, 2, \ldots, n\}$, and let $f : \mathbb{R}^n \to \mathbb{R}$ and $g : \mathbb{R}^n \to \mathbb{R}$ be continuous piecewise multi-convex with respect to $\mathcal{I}$. Then so is $f + g$. If $f$ and $g$ are piecewise continuously differentiable, then so is $f + g$.*

**Proof.** Let $F$ be a piece of $f$ and $G$ be a piece of $g$ with $\boldsymbol{x} \in F \cap G$. Then for all $I \in \mathcal{I}$, $(F \cap G)_I(\boldsymbol{x}) = F_I(\boldsymbol{x}) \cap G_I(\boldsymbol{x})$, a convex set on which $f + g$ is convex. Thus $f + g$ is continuous piecewise multi-convex, where the pieces of $f + g$ are the non-empty intersections of pieces of $f$ and $g$.

Let $\tilde{f}$ and $\tilde{g}$ be the active functions of $f$ and $g$ on $F$ and $G$, respectively. Then $\tilde{f} + \tilde{g}$ is active on $F \cap G$, and $\tilde{f}, \tilde{g} \in C^1$ so $\tilde{f} + \tilde{g} \in C^1$. □

We can now prove the theorem.

**Proof.** For the moment, assume we have only a single fixed data point, as in the proof of Theorem 3.1. Let $g_1$ and $g_2$ denote the first two layers of $f$, with parameters $\boldsymbol{\theta}_1 \in \mathbb{R}^m$ and $\boldsymbol{\theta}_2 \in \mathbb{R}^n$, respectively. Consider the two-layer sub-network $g_2 \circ g_1 = g_2(g_1(\boldsymbol{\theta}_1), \boldsymbol{\theta}_2)$. By Lemma B.1, the sub-network is continuous piecewise multi-affine with respect to $\{I_1, I_2\}$. Repeating this argument, the composition of all layers $g = g_N \circ g_{N-1} \circ \cdots \circ g_1$ is continuous piecewise multi-affine with respect to $\mathcal{I}$.

Let $T$ be a piece of $g, \boldsymbol{\theta} \in T$ and $I \in \mathcal{I}$. Then $g$ is affine on $T_I(\boldsymbol{\theta})$, so $f = h \circ g$ is convex when restricted to this set. Thus $f$ is continuous piecewise multi-convex with the same pieces as $g$.

To see that $f$ is piecewise continuously differentiable, note that multi-affine functions are continuously differentiable, since their partial derivatives are the derivatives of affine functions. Thus $g_1, g_2, \ldots, g_N$ are piecewise continuously differentiable, so their composition with $h$ is piecewise continuously differentiable as well.

Now we extend the theorem to the whole dataset, where each data point defines a continuous piecewise multi-convex function $f_k$. By Lemma B.2, the arithmetic mean $(1/N)\sum_{k=1}^{N} f_k$ is continuous piecewise multi-convex and piecewise continuously differentiable. □

## References


Aumann, R. J., & Hart, S. (1986). Bi-convexity and bi-martingales. *Israel Journal of Mathematics*, *54*(2), 159–180. http://dx.doi.org/10.1007/BF02764940.

Bach, F. R. & Moulines, E. (2011). Non-asymptotic analysis of stochastic approximation algorithms for machine learning. In *Proceedings of the 25th annual conference on neural information processing systems*, (pp. 451–459).

Baldi, P., & Lu, Z. (2012). Complex-valued autoencoders. *Neural Networks*, *33*, 136–147. http://dx.doi.org/10.1016/j.neunet.2012.04.011.

Barron, A. R. (1993). Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory*, *39*(3), 930–945. http://dx.doi.org/10.1109/18.256500.

Bertsekas, D. P. (2010). Incremental gradient, subgradient, and proximal methods for convex optimization: A Survey, Tech. rep., Massachusetts Institute of Technology Labratory for Information and Decision Systems. URL http://arxiv.org/abs/1507.01030.

Blum, A. L., & Rivest, R. L. (1992). Training a 3-node neural network is np-complete. *Neural Networks*, *5*(1), 117–127. http://dx.doi.org/10.1016/S0893-6080(05)80010-3.

Ge, R., Huang, F., Jin, C., & Yuan, Y. (2015). Escaping from saddle points - online stochastic gradient for tensor decomposition In *Journal of machine learning research - workshop and conference proceedings, Vol. 1* (pp. 1–46).

Glorot, X., Bordes, A., & Bengio, Y. (2011). Deep sparse rectifier neural networks, In *G.J. Gordon, D.B. Dunson (Eds.), Proceedings of the fourteenth international conference on artificial intelligence and statistics, AISTATS-11, journal of machine learning research - workshop and conference proceedings, Vol. 15* (pp. 315–323).

Gnecco, G., & Sanguineti, M. (2008). Estimates of the approximation error using rademacher complexity: learning vector-valued functions. *Journal of Inequalities and Applications*, *2008*(1), 640758. http://dx.doi.org/10.1155/2008/640758.

Gorski, J., Pfeuffer, F., & Klamroth, K. (2007). Biconvex sets and optimization with biconvex functions: a survey and extensions. *Mathematical Methods of Operations Research*, *66*(3), 373–407. http://dx.doi.org/10.1007/s00186-007-0161-1.

Iusem, A. N. (2003). On the convergence properties of the projected gradient method for convex optimization. *Computational and Applied Mathematics*, *22*, 37–52.

Johnsonbaugh, R., & Pfaffenberger, W. E. (1981). *Foundations of mathematical analysis*. New York, New York, USA: Marcel Dekker, Dover 2002 Edition.

Kainen, P. C., & Kúrková, V. (2009). An integral upper bound for neural network approximation. *Neural Computation*, *21*(10), 2970–2989. http://dx.doi.org/10.1162/neco.2009.04-08-745.

Kawaguchi, K. (2016). Deep learning without poor local minima. In *Advances in Neural Information Processing Systems Vol. 29* (pp. 586–594). Curran Associates, Inc.

Makovoz, Y. (1998). Uniform approximation by neural networks. *Journal of Approximation Theory*, *95*(2), 215–228. http://dx.doi.org/10.1006/jath.1997.3217.

Nesterov, Y. (2004). *Applied optimization. Introductory lectures on convex optimization: a basic course*. Boston, Dordrecht, London: Kluwer Academic Publishers.

Ovchinnikov, S. (2002). Max-Min representation of piecewise linear functions. *Contributions To Algebra and Geometry*, *43*(1), 297–302.

Rockafellar, R. T. (1970). *Convex analysis*. Princeton, NJ, USA: Princeton University Press, 41 William Street.

Soudry, D. & Carmon, Y. (2016). No bad local minima: data independent training error guarantees for multilayer neural networks, arXiv:1605.08361.

Tsevendorj, I. (2001). Piecewise-Convex maximization problems. *Journal of Global Optimization*, *21*(1), 1–14. http://dx.doi.org/10.1023/A:1017979506314.

Wendell, R. E., & Hurter, A. P. (1976). Minimization of a non-separable objective function subject to disjoint constraints. *Operations Research*, *24*(4), 643–657. http://dx.doi.org/10.1287/opre.24.4.643.

Zangwill, W. I. (1969). *Prentice-Hall international series in management. Nonlinear programming : a unified approach*. Englewood Cliffs, N.J.: Prentice-Hall.