Statistics and Computing

Monotonicity Shape Constraints for Binary Classifiers

--Manuscript Draft--

Manuscript Number:	STCO-D-19-00096
Full Title:	Monotonicity Shape Constraints for Binary Classifiers
Article Type:	Manuscript
Keywords:	shape constraints; classification; quadratic constraints

Corresponding Author:	Maya Gupta Google Inc Mountain View, CA UNITED STATES
Corresponding Author Secondary Information:	
Corresponding Author's Institution:	Google Inc
Corresponding Author's Secondary Institution:	
First Author:	Daniel Kraft
First Author Secondary Information:	
Order of Authors:	Daniel Kraft
	Maya Gupta, Ph.D.
Order of Authors Secondary Information:	

Funding Information:	

Noname manuscript No.
(will be inserted by the editor)

Monotonicity Shape Constraints for Binary Classifiers

Daniel Kraft · Maya R. Gupta

Received: March 3, 2019 / Accepted: date

Abstract Constraining a function to respond monotonically to changes in selected inputs is a popular shape constraint to capture domain knowledge and regularize a model. For binary classifiers, a common strategy to produce a monotonic classifier is to first train a monotonic function, and then threshold it. An open question is whether better performance is possible by instead optimizing over the set of all monotonic classifiers. We investigate this for lattice models, which are a state-of-the-art function class for training multi-dimensional monotonic functions. Monotonic lattice functions require satisfying linear inequality constraints to guarantee monotonicity. We show that there exist monotonic lattice classifiers that cannot be produced by thresholding a monotonic lattice function. We give quadratic inequality constraints that form a tighter sufficient condition for a lattice classifier to be monotonic, and show that these constraints are necessary and sufficient for two-dimensional classifiers. However, we also provide theoretical results showing that the same classifier expressability can be achieved by training and thresholding a two-layer lattice function. Our analysis and simulations lead us to hypothesize that training and thresholding more flexible monotonic functions will generally be easier and preferable in practice to optimizing over the set of all monotonic classifiers.

Keywords shape constraints · classification · constrained optimization

1 Introduction

A popular shape constraint is to restrict a nonlinear function to respond only positively (or only negatively) to selected inputs (see, e.g. [3] [13],[7] [15], [4], [5], [28], [22], [6], [19], [27]). For example, in classifying whether an applicant should be approved for a credit card, the classifier decision could be constrained to be monotoni-

D. Kraft · M. R. Gupta
Google AI
Tel.: +1206-431-9410
E-mail: dkraft@google.com, mayagupta@google.com,

cally increasing with respect to the applicant's credit score, monotonically decreasing with respect to the number of prior bankruptcies, and no constraints imposed with respect to the applicant's zipcode. Monotonicity constraints are widely-regarded as making models more trustworthy and interpretable, and can provide useful regularization, particularly when there is domain shift between training and test distributions.

In this paper, we focus on the problem of training monotonic binary classifiers. This is straightforward for simple models such as linear classifiers or boosted stumps. However, for more complex function classes, it is a challenging problem to characterize and find the optimal monotonic binary classifier. For example, the set of all possible monotonic *decision tree* classifiers are those where the two classifier labels for every pair of leaves in the final tree satisfies the monotonicity constraints [10]. This is a difficult set to optimize over; Potharst and Feelders [21] considered training trees on different random samples of the training samples, and then choosing the best tree out of any that satisfied the monotonicity constraints.

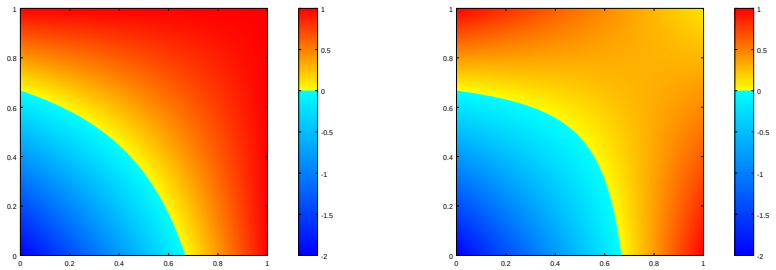
A more common strategy for producing binary classifiers is to train a monotonic discriminant function, and then threshold the monotonic discriminant to produce a monotonic classifier. For example, if one trains a neural net with non-negative weights, then it forms a monotonic function [1] (there are many more examples of this strategy [25], [18], [28]).

However, this strategy is only sufficient and not necessary [15]: one can have a non-monotonic discriminant that after thresholding produces a monotonic classifier. See the top-right classifier in Figure 1 for an example. This leads to the open question, "If we optimize over the set of all possible monotonic classifiers can we obtain monotonic classifiers that work better than thresholding monotonic functions?" And as a stepping stone to answering that, "Can we state the necessary conditions for a monotonic classifier for any nonlinear smooth function classes?"

In this paper, we answer these two open questions for the function class of *lattice models*. Lattice models generalize one-dimensional interpolated look-up tables to multiple dimensions [12] [24] [11]. Because they are parameterized by highly-structured look-up tables, lattice models have been shown to be particularly well-suited for learning multi-dimensional functions with shape constraints [15][5][28][14]. Monotonic lattice models have been shown to perform experimentally as well or better than monotonic DNN's or monotonic min-max networks [28]. Next, we give a brief review of lattice models, then state our main contributions.

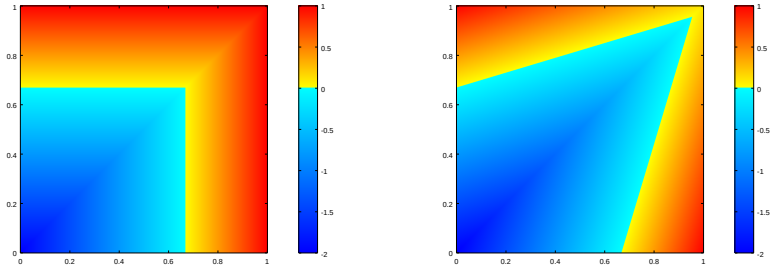
1.1 Background on Lattice Models

Lattices are interpolated look-up tables. A lattice function on one-dimension is simply a piecewise linear function, and one-dimensional lattices been used to approximate and represent one-dimensional functions for centuries, for example tables for logarithms [23] [20] and actuarial tables [9]. Multi-dimensional functions can also be represented and approximated by interpolating multi-dimensional look-up tables, see Figure 1 for examples of such two-dimensional lattices. Approximating low-dimensional functions with a look-up table that is later interpolated is also common, for example four-dimensional two-layer lattice models are a standard way to express



Monotonic function
Monotonic classifier
Bilinear interpolation

Non-monotonic function
Monotonic classifier
Bilinear interpolation



Monotonic function
Monotonic classifier
Simplex interpolation

Non-monotonic function
Non-monotonic classifier
Simplex interpolation

Fig. 1: Four examples of classifiers defined over $D = 2$ inputs bounded to the unit square. Each classifier is formed by thresholding a lattice function at $f(x) = 0$. A lattice function is parametrized by a 2×2 look-up table defined on the four corners of the input space. The left functions have parameters (clockwise from the origin) $(-2, 1, 1, 1)$, and the right functions have parameters $(-2, 1, 0{,}1, 1)$. The top functions interpolate their four look-up table values with bilinear interpolation (the two-dimensional special case of multilinear interpolation), which produces a function that is a bilinear polynomial (in higher dimensions, a multilinear polynomial). The bottom functions interpolate the same look-up table values but with simplex interpolation (aka Lovasz extension [2]), which produces a function that is linear on each of $D!$ simplices; for $D = 2$ the two simplices are the lower-right and upper-left triangles (see Gupta et al. 2016 [15] for a discussion on handling the lack of rotational invariance). Our colormap is discontinuous with a jump from blue to yellow at $f(\cdot) = 0$, so that the binary classifier's two classes are shown as blue vs orange. The functions on the left are monotonically increasing in both inputs. The functions on the right are not monotonic in either direction. However, thresholding the top-right non-monotonic function at $f(x) = 0$ produces a monotonic classifier. Further, we will show that there does not exist a monotonic lattice function that can be thresholded to product the top-right classifier decision boundary.

how a given CMYK color description will be transformed by a particular printer into a CIELab color value [24]. Given enough look-up table values, lattices can approximate any bounded continuous function [11].

The parameters of a lattice function are the underlying look-up table values, and these parameters can be trained using standard empirical risk minimization [12], [11]. Each cell of a D-dimensional lattice function is defined by the 2^D look-up table values at its corners. To evaluate the function at a point inside the cell, the 2^D corner values are linearly interpolated in one of two ways: either with multilinear interpolation (which is the multi-dimensional generalization of bilinear interpolation) at a computational cost of $O(2^D)$, or with *simplex* interpolation (aka Lovasz extension [2]) at a computational cost of $O(D \log D)$. See Figure 1 for examples of the two different interpolations.

A one-dimensional lattice function is monotonically increasing if every look-up table parameter is larger than its left-neighbor. More generally, a multi-dimensional lattice function is *monotonic* with respect to an input dimension, if the look-up table values are always monotonically increasing in that dimension [15]. Thus monotonicity can be ensured by pairwise parameter constraints that force neighboring look-up table values to be increasing, and these pairwise parameter constraints are linear inequality constraints that can be imposed during empirical risk minimization training [15].

Lattices can be summed into ensembles [5] and cascaded with linear embeddings into deep models [28], but even for these *deep lattice networks*, the monotonicity guarantees still only require pairwise linear inequality constraints on the model parameters [28]. The open-source Tensor Flow Lattice package is available for learning monotonic deep lattice networks [17].

1.2 Main Contributions

First, we will show in Section 3 that if one interpolates the look-up table with *simplex interpolation*, learning a monotonic lattice model and then thresholding it is in fact optimal: we show there do not exist monotonic classifiers that cannot be expressed as a thresholded monotonic lattice.

Second, we show in Section 4 that if instead one interpolates the look-up table with *multilinear interpolation*, then there are indeed infinite monotonic classifiers that are not achievable by thresholding a monotonic lattice model.

Third, we show in Section 4 that it is sufficient for the lattice model parameters to satisfy a set of quadratic inequality constraints in order to produce a monotonic classifier, that the given quadratic conditions are looser than the known linear conditions for a monotonic lattice, and that the given quadratic constraints are both necessary and sufficient for $D = 2$ inputs. Thus one can directly train a monotonic classifier by minimizing empirical risk with quadratic constraints on the lattice parameters.

Fourth, and surprisingly, we show in Section 6 that if one expands the function class to *two-layer* lattice models [15], where the first layer calibrates each feature with a piecewise linear function (represented as a one-dimensional lattice) before fusing the D calibrated inputs with a multi-dimensional lattice, then one can express

any monotonic lattice classifier for $D = 2$ using a thresholded monotonic calibrated lattice function. Extension to $D \geq 3$ is discussed.

Overall, our investigation leads us to believe that thresholding monotonic functions to produce monotonic classifiers is likely sufficient in practice, and that there is little to gain in directly trying to learn monotonic classifiers. We discuss our conclusions and hypotheses in Section 8.

2 Preliminaries

Let D be the number of input features, and assume a bounded input domain. Thus, without loss of generality, we train a function $f\colon [0,1]^D \to \mathbb{R}$, which is thresholded at 0 to form the binary classifier $I_{f>0}$. Each f is taken to be a single-cell lattice, i.e. a lattice of size 2^D. Most of our results can be generalized to multi-cell lattices by considering each cell separately. The 2^D lattice is parametrized by the set $\{v_x \in \mathbb{R}\}$ of 2^D parameters, with $x \in V_D = \{0,1\}^D$ corresponding to the set of vertices of the D-unit-hypercube. For an input vector of feature values $\xi \in [0,1]^D$, the model's output is $f(\xi) = \sum_{x \in V_D} v_x \theta_x(\xi)$, where θ_x is the interpolation weight on vertex x for the input ξ. In the following, we denote the dth component of ξ by ξ_d. The exact definition of $\theta_x(\xi)$ depends on whether one uses multilinear interpolation or simplex interpolation, as detailed below. In either case, the interpolation weights are defined such that $f(x) = v_x$ for all vertices $x \in V_D$.

Recall a function g is said to be *monotonically increasing* if $g(\xi)$ never decreases as ξ_d increases for any d. For simplicity, we use the term *monotonic* to mean *monotonically increasing* unless explicitly stated otherwise. A function is called *partially monotonic* if monotonicity holds for some subset of the input dimensions, but not all [8]. For simplicity, we prove many of our results for monotonicity; but most can be easily extended to partial monotonicity.

3 Simplex Interpolation

Let us first consider the case where the look-up table is interpolated using *simplex interpolation*, as defined in Subsection 5.2 of Gupta et al. 2016 [15] and also described, for example, in Weiser and Zarantonello 1998 [26]. Simplex interpolation is equivalent to the Lovasz extension in submodularity [2]. Simplex interpolation implicitly partitions the unit cube $[0,1]^D$ into the $D!$ simplices that touch on the main diagonal spanning from $(0,\ldots,0)$ to $(1,\ldots,1)$, such that for each permutation π of $\{1,\ldots,D\}$, the corresponding simplex is $S_\pi = \left\{\xi \in \mathbb{R}^D \;\middle|\; 0 \leq \xi_{\pi(1)} \leq \cdots \leq \xi_{\pi(D)} \leq 1\right.$. On each such simplex S, the restriction $f|_S$ of f to S is the linear interpolation of the parameters v_x corresponding to the simplex's $D+1$ vertices. Thus, $f|_S$ is the unique hyperplane that passes through the simplex's $D+1$ vertices. Simplex interpolation yields a well-defined, continuous interpolation function $f\colon [0,1]^D \to \mathbb{R}$ that is piecewise linear. In practice, simplex interpolation is very useful because it is fast to evaluate due to its $O(D \log D)$ complexity (rather than

the $O(2^D)$ complexity of multilinear interpolation), and produces good classification results [15][5].

Lemma 3 of Gupta et al. 2016 [15] states that pairwise linear inequality constraints must be satisfied for a simplex-interpolated lattice function to be monotonic. In Theorem 1, we show the same criteria determines whether the corresponding *classifier* is monotonic, and hence for simplex interpolation, there are no monotonic lattice classifiers that are not also monotonic lattice functions.

Theorem 1. *Let f be defined through simplex interpolation on a D-unit-hypercube and assume that the binary classifier $I_{f>0}$ is monotonic and non-degenerate such that $v_{(0,\dots,0)} < 0 < v_{(1,\dots,1)}$. Then f is monotonic on $[0,1]^D$.*

Proof Let S be any simplex S_π as introduced above, and note that it always contains the vertices $a = (0,\dots,0)$ and $b = (1,\dots,1)$. We can also find points $\xi_a, \xi_b \in S^\circ$ in the interior of S, arbitrarily close to a and b. In particular, we can assume $f(\xi_a) < 0 < f(\xi_b)$ by continuity of f. Then, by the intermediate-value theorem, there exists $\xi \in S^\circ$ with $f(\xi) = 0$.

Next, note that $f|_S$ is an affine function. If it is not increasing, then there is a dimension d such that the partial derivative $\partial f/\partial \xi_d$ in this dimension is strictly negative—throughout the whole of S. Thus, in particular, $\partial f(\xi)/\partial \xi_d < 0$. This means that there exists $\epsilon > 0$ small enough such that $\xi \pm \epsilon \hat{d} \in S$, where $\hat{d}$ is the unit vector in direction d, and also $f(\xi + \epsilon\hat{d}) < 0 < f(\xi - \epsilon\hat{d})$. But this contradicts the assumption that $I_{f>0}$ is a monotonic classifier on S. Hence, f must be increasing on S. Since S was arbitrary and f is continuous across the boundaries of simplices, f is increasing on the whole of $[0,1]^D$.

4 Multilinear Interpolation

For the rest of this paper, we focus on *multilinear* interpolation. Multilinear interpolation is the multi-dimensional generalization of the common bilinear interpolation. Of all the possible ways one can linearly interpolate the vertices of a hypercube, multilinear interpolation is the maximum entropy solution [16]. In contrast to simplex interpolation, there exist monotonic classifiers that can only be produced by thresholding a non-monotonic multilinear function, as shown in the top-right picture in Figure 1.

The multilinear interpolation weight on vertex $x \in V_D$ of the D-unit-hypercube is defined for $\xi \in [0,1]^D$ as

$$\theta_x(\xi) = \prod_{d=1}^{D} \xi_d^{x_d} \left(1 - \xi_d\right)^{1-x_d}. \tag{1}$$

Multilinear interpolation then applies these weights to the 2^D lattice vertices:

$$f(\xi) = \sum_{x \in V_D} v_x \theta_x(\xi). \tag{2}$$

4.1 The Subset Criterion for Monotone Classifiers

One of the key analytical properties of a multilinear function f is that its partial derivatives $\partial f/\partial \xi_d$ are independent of ξ_d. From this property, we establish the following *subset criterion* for the monotonicity of a classifier $I_{f>0}$:

Lemma 2. *Let f be a multilinear interpolation function and $d = 1, \ldots, D$ one of the dimensions. Then the classifier $I_{f>0}$ is monotonic in dimension d if and only if*

$$\begin{aligned} &\left\{\xi \in [0,1]^{D-1} \;\middle|\; f(\xi_1, \ldots, \xi_{d-1}, 0, \xi_d, \ldots, \xi_{D-1}) > 0\right\} \\ &\subset \left\{\xi \in [0,1]^{D-1} \;\middle|\; f(\xi_1, \ldots, \xi_{d-1}, 1, \xi_d, \ldots, \xi_{D-1}) > 0\right\}. \end{aligned} \tag{3}$$

Proof Assume first that $I_{f>0}$ is a monotonic classifier, and let ξ be in the left-hand side of (3). Since $f(\xi_1, \ldots, \xi_{d-1}, 0, \xi_d, \ldots, \xi_{D-1}) > 0$ and $I_{f>0}$ is monotonic in dimension d, it follows that $f(\xi_1, \ldots, \xi_{d-1}, 1, \xi_d, \ldots, \xi_{D-1}) > 0$ must be true as well. This shows that ξ is also in the right-hand side of (3), so that the subset relation holds.

Now assume that (3) is true. We have to show that $I_{f>0}$ is monotonic in dimension d. For this, let $\xi, \zeta \in [0,1]^D$ with $\xi_i = \zeta_i$ for all $i \neq d$ and $\xi_d < \zeta_d$. We have to show that $I_{f>0}(\xi) \leq I_{f>0}(\zeta)$. Assume to the contrary that $f(\xi) > 0$ and $f(\zeta) \leq 0$ for some ξ and ζ. Let us define $\underline{\xi}$ and $\overline{\xi}$ such that

$$\begin{aligned} \underline{\xi}_i = \overline{\xi}_i = \xi_i = \zeta_i \text{ for } i \neq d, \\ \underline{\xi}_d = 0 \leq \xi_d < \zeta_d \leq \overline{\xi}_d = 1. \end{aligned}$$

If $f(\underline{\xi}) > 0$, then also $f(\overline{\xi}) > 0$ according to (3). Since f is linear in dimension d, this implies that $f(\zeta) > 0$, contradicting our assumption. A similar argument applies if both $f(\underline{\xi}) \leq 0$ and $f(\overline{\xi}) \leq 0$ are true. Thus, it remains to consider the case $f(\underline{\xi}) \leq 0 < f(\overline{\xi})$. But then, the mean-value theorem and the observation made above about multilinear functions imply that $\partial f/\partial \xi_d > 0$ along the line through ξ and ζ. This means that $f(\zeta) > f(\xi)$, which is also a contradiction.

The left-hand side of (3) is the set of points on the $\xi_d = 0$ "lower" face of the input hypercube that the classifier labels positive. The right-hand side is the set of points on the "upper" face with $\xi_d = 1$. Thus Lemma 2 states that the classifier is monotonic if and only if the set of "lower-face positive points" is a subset of the "upper-face positive points".

4.2 Quadratic Constraints Are Necessary and Sufficient for $D = 2$

For the special case of $D = 2$, we show that the Lemma 2 can be written as quadratic constraints on the look-up table values:

Corollary 3. *Let $D = 2$ and let f be the bilinear interpolation for the parameters v_{00}, v_{01}, v_{10} and v_{11}. Assume $v_{00} < 0 < v_{11}$. Then $I_{f>0}$ is monotonic if and only if*

$$v_{00} v_{11} \leq v_{01} v_{10}. \tag{4}$$

Proof We apply Lemma 2, for which we have to show that (4) is equivalent to (3) for both $d = 1$ and $d = 2$. Assume first that $v_{01}v_{10} \geq 0$. In this case, (4) is always fulfilled since $v_{00}v_{11} < 0$ by assumption. If $v_{01}, v_{10} \geq 0$, then the right-hand side of (3) contains at least $(0, 1]$, so that (3) is fulfilled as well. If $v_{01}, v_{10} \leq 0$, then the left-hand side is empty with the same conclusion.

It remains to consider the case $v_{01} \cdot v_{10} < 0$. Without loss of generality, we can assume that $v_{01} < 0 < v_{10}$ is the case. In this case, (3) is trivially fulfilled for $d = 1$ since the left-hand side is empty and the right hand side is $[0, 1]$. So what we have to show is equivalence of (4) to this simplified version of (3):

$$\{\xi \in [0,1] \mid f(\xi, 0) > 0\} \subset \{\xi \in [0,1] \mid f(\xi, 1) > 0\}.$$

Using the bilinear form of f, this subset relation can be rewritten as

$$\xi v_{10} + (1-\xi)v_{00} > 0 \;\;\Rightarrow\;\; \xi v_{11} + (1-\xi)v_{01} > 0.$$

Noting $v_{10} - v_{00} > 0$ and $v_{11} - v_{01} > 0$, this can be reformulated equivalently to:

$$\xi > \frac{-v_{00}}{v_{10} - v_{00}} \;\;\Rightarrow\;\; \xi > \frac{-v_{01}}{v_{11} - v_{01}}.$$

Requiring this for all $\xi \in [0, 1]$ is equivalent to

$$\frac{-v_{00}}{v_{10} - v_{00}} \geq \frac{-v_{01}}{v_{11} - v_{01}} \;\;\Leftrightarrow\;\; v_{00}v_{11} \leq v_{01}v_{10}.$$

Note that, as above in Theorem 1, the assumption $v_{00} < 0 < v_{11}$ only excludes trivial cases (where all inputs are mapped to a single class) for a monotonic classifier.

4.3 Quadratic Constraints: Sufficient for Monotonic Classifiers

For $D \geq 3$, we use (3) to show that a set of 2^{D-1} quadratic constraints forms *sufficient conditions* for monotonicity of a lattice classifier. Without loss of generality, we consider monotonicity with respect to the dth input. Let $\underline{f}\colon [0,1]^{D-1} \to \mathbb{R}$ and $\overline{f}\colon [0,1]^{D-1} \to \mathbb{R}$ be the multilinear interpolation functions on the lower and upper faces of the full hypercube with respect to d, i.e.

$$\underline{f}(\xi) = f(\xi_1, \ldots, \xi_{d-1}, 0, \xi_d, \ldots, \xi_{D-1}),$$
$$\overline{f}(\xi) = f(\xi_1, \ldots, \xi_{d-1}, 1, \xi_d, \ldots, \xi_{D-1}).$$

For any $x \in V_{D-1}$, let $\underline{v}_x \in \mathbb{R}$ denote a look-up table parameter corresponding to vertices of the lower face such that their dth coordinate is zero and the other $D-1$ coordinates are given by the coordinates of x. Similarly, $\overline{v}_x \in \mathbb{R}$ denotes a look-up table parameter corresponding to vertices on the upper face with dth coordinate equal to one. In other words, $\underline{f}(x) = \underline{v}_x$ and $\overline{f}(x) = \overline{v}_x$ for all $x \in V_{D-1}$.

Theorem 4. *Let f be a multilinear interpolation function of dimension D, and $\underline{f}$, $\overline{f}$ as introduced above. Then a sufficient condition for (3) and thus the monotonicity of the classifier $I_{f>0}$ in dimension d is any of:*

1. $\underline{v}_x \leq 0$ *for all* $x \in V_{D-1}$*, or* $\overline{v}_x > 0$ *for all* x*.*
2. *There exists* $x \in V_{D-1}$ *with* $\underline{v}_x \cdot \overline{v}_x > 0$ *such that for all* $y \in V_{D-1} \setminus \{x\}$*:*

$$\underline{v}_y \left|\overline{v}_x\right| \leq \overline{v}_y \left|\underline{v}_x\right|. \tag{5}$$

Proof If the first condition is true, then (3) follows by the properties of interpolation. Otherwise if the second condition is true for some $x \in V_{D-1}$*, then for all* $y \in V_{D-1} \setminus \{x\}$*:*

$$\underline{v}_y \left|\overline{v}_x\right| \leq \overline{v}_y \left|\underline{v}_x\right| \quad \Leftrightarrow \quad \underline{v}_y \leq \overline{v}_y \left|\frac{\underline{v}_x}{\overline{v}_x}\right| = \underline{v}_y \leq \overline{v}_y \left(\frac{\underline{v}_x}{\overline{v}_x}\right).$$

The right inequality is trivially also satisfied for $y = x$*. Thus with* $\alpha = \underline{v}_x / \overline{v}_x > 0$*, we have* $\underline{f} \leq \alpha \overline{f}$*. This finishes the proof, since* $I_{\overline{f}>0} = I_{\alpha \overline{f}>0}$*.*

For $D = 2$, Theorem 4 actually turns into Corollary 3 and is an equivalent characterization of monotonicity for the classifier. For $D \geq 3$, the condition is only sufficient and not necessary, as quantified in Table 1. Next we verify that if the lattice function is monotonic, it does satisfy these sufficient conditions. In the next section, we will, however, see that the sufficient conditions of Theorem 4 are *strictly looser* than only requiring the entire function to be monotonic (which can be achieved by satisfying linear constraints [15]).

Theorem 5. *Let* f *be a multilinear interpolation function as before, and let* $\underline{f}$ *and* $\overline{f}$ *be as introduced above. Assume that* f *is monotonic with respect to dimension* d*. Then the conditions of Theorem 4 are satisfied for* $\underline{f}$ *and* $\overline{f}$*.*

Proof Note first that monotonicity of f *implies* $\underline{v} \leq \overline{v}$ *for all parameters. If there is no* $x \in V_{D-1}$ *with* $\underline{v}_x \cdot \overline{v}_x > 0$*, then the first condition of Theorem 4 must be true: If* $\underline{v}_x > 0$ *were true for any* x*, then also* $\overline{v}_x \leq 0$ *would have to be true. But this is not possible due to monotonicity, showing that, in fact,* $\underline{v}_x \leq 0$ *must be the case for all* $x \in V_{D-1}$*.*

Next, let us assume that $\underline{v}_x, \overline{v}_x < 0$ *is true for at least one* $x \in V_{D-1}$*. Furthermore, let us assume*

$$\left|\frac{\underline{v}_x}{\overline{v}_x}\right| \leq \left|\frac{\underline{v}_y}{\overline{v}_y}\right|, \tag{6}$$

for all other $y \neq x$ *with* $\underline{v}_y, \overline{v}_y < 0$*. (In other words, we consider that pair of both-negative parameters that has the smallest ratio* $|\underline{v}| \,/\, |\overline{v}|$*.) We can now show that the second condition of Theorem 4 holds for this* x*:*

For this, let $y \in V_{D-1} \setminus \{x\}$ *be fixed. If* $\underline{v}_y \leq 0 \leq \overline{v}_y$*, then (5) holds trivially, since the left-hand side is non-positive and the right-hand side non-negative. If* $0 \leq \underline{v}_y \leq \overline{v}_y$*, then (note that* $|\overline{v}_x| \leq |\underline{v}_x|$*):*

$$\underline{v}_y \leq \overline{v}_y \quad \Rightarrow \quad \underline{v}_y \left|\overline{v}_x\right| \leq \overline{v}_y \left|\overline{v}_x\right| \leq \overline{v}_y \left|\underline{v}_x\right|,$$

so that (5) holds also in this case. The only case that remains to consider now is $\underline{v}_y \leq \overline{v}_y < 0$*. In that case, note that (6) holds for* y*. This implies:*

$$\begin{aligned} \left|\frac{\underline{v}_x}{\overline{v}_x}\right| \leq \left|\frac{\underline{v}_y}{\overline{v}_y}\right| \quad &\Rightarrow \quad \left|\overline{v}_y\right| \left|\underline{v}_x\right| \leq \left|\underline{v}_y\right| \left|\overline{v}_x\right| \\ &\Rightarrow \quad \overline{v}_y \left|\underline{v}_x\right| \geq \underline{v}_y \left|\overline{v}_x\right|. \end{aligned}$$

This, in turn, shows that (5) holds again.

If $\underline{v}_x, \overline{v}_x > 0$ for one $x \in V_{D-1}$, then a similar argument can be applied to show the claim.

5 How Tight Are These Sufficiency Conditions for Monotonic Classifiers?

In this section we give a quantitative sense of how large the set of monotonic classifiers is compared to the set of thresholded monotonic functions for lattices, and thus a quantative sense of how tight our given sufficient conditions in Theorem 4 are.

We draw random lattices for $D \in \{2, 3, 4\}$, where for each random lattice each of its 2^D lattice parameters is chosen independently and uniformly from $[-1, 1]$. We discard any lattices that result in a constant classifier (the parameters are all positive or all negative) to exclude trivial cases, which happens for 12.5% of random $D = 2$ lattices, but only 0.8% of random $D = 3$ lattices. The columns of Table Table 1 show the number of random lattice classifiers that are: (i) monotonic functions on all D inputs (using the necessary and sufficient linear constraints [15]), (ii) satisfy the new sufficient quadratic constraints of Theorem 4, and (iii) form a monotonic classifier. To check if a lattice is a monotonic classifier, for $D = 2$ and $D = 3$, we can check analytically. For $D = 4$ we approximate the subset criterion of Lemma 2 by discretizing the sets on a fine grid, such that the statistic given in the last column for 4D is both an approximation and an exact upper bound.

Table 1: The three right-most columns show the count of random lattices that satisfy the column headings. For the 100 million sample of 4D classifiers, we have marked the last column *N/A*, as we were computationally unable to establish this number.

# Inputs	# Samples	Monotonic Function	Quadratic Sufficient Conditions Satisfied	Monotonic Classifier
2D	1 000 000	83 203	214 300	214 300
3D	1 000 000	1 168	18 650	27 602
4D	1 000 000	1	53	≤ 204
4D	100 000 000	8	5 597	*N/A*

Table 1 shows that the set of lattices that form monotonic functions is about than the set of monotonic classifiers, and the gap becomes bigger as the input dimension D rises. Our quadratic sufficient conditions narrow the gap, but we did find $D = 3$ and $D = 4$ classifiers that did not satisfy the quadratic conditions. As expected, the simulation did not discover any $D = 2$ monotonic classifiers that failed the quadratic conditions, consistent with our Corollary 3 showing the quadratic conditions are also necessary for $D = 2$.

Table 1 also gives a sense of what a strong regularizer monotonicity constraints are, in that only a small fraction of random lattice functions are monotonic.

6 Calibrated Lattices with Multilinear Interpolation

A more flexible model and generally better accuracy can be achieved by using a two-layer *calibrated lattice* model that first passes each feature through its own "calibrator" - a one-dimensional lattice (which is simply a piecewise linear function), before feeding the D calibrated values into a multi-dimensional lattice [24], [15], [5], [28].

A calibrator $c_d(\xi_d)$ can approximate any continuous, bounded, one-dimensional function if the look-up table parameterizing it is given enough values. Taking this to the extreme to simplify our analysis, we will assume that a calibrator c_d, for $d = 1, \dots, D$, is simply a continuous and surjective mapping $c_d \colon [0,1] \to [0,1]$.

Let a *calibrated multilinear lattice* be described by the tuple $(f, c_1, \dots, c_D)$ where f is the multilinear lattice function and c_d is the calibrator for feature d. The output of a calibrated lattice function g is then

$$g \colon [0,1]^D \to \mathbb{R}, \;\; \xi \mapsto f(c_1(\xi_1), \dots, c_D(\xi_D)).$$

By composition of monotonicity, a calibrated lattice function g is monotonic in dimension d if the lattice f itself is monotonic in this dimension and also c_d is monotonic.

Calibration makes a 2^D lattice model significantly more flexible and expressive. In fact, we show its additional expressiveness is enough to completely close the gap between monotonic lattices and monotonic classifiers for special cases in low dimensions. For higher dimensions, we will see in Section **??** that the same is true at least empirically for numerical experiments.

Let f^* denote a multilinear interpolation function that forms a monotonic classifier such that $I_{f^*>0}$. We investigate whether or not there exists a calibrated lattice function g that is *fully monotonic* in the sense that both its multi-dimensional interpolation function f and its calibrators are monotonic, and that when thresholded yields the same classifier as f^* itself:

$$f^*(\xi) > 0 \;\; \Leftrightarrow \;\; f(c_1(\xi_1), \dots, c_D(\xi_D)) > 0.$$

6.1 Two-Input Case

Let $D = 2$ and denote the two inputs as $(\xi, \zeta) \in [0,1]^2$. The monotonic classifier formed by f^* is completely characterized by its *decision boundary* $\{(\xi, \zeta) \in [0,1]^2 \mid f^*(\xi, \zeta) = 0\}$. We restrict our analysis to the non-pathological cases where this decision boundary can be parametrized (for instance, by the implicit-function theorem) as $d^* \colon [0, \overline{\xi}] \to [0,1]$, where $f^*(\xi, d^*(\xi)) = 0$ for all $\xi \in [0, \overline{\xi}]$.

For the upper bound, we have either $\overline{\xi} = 1$ if $f^*(1,0) < 0$ or $\overline{\xi} \in (0,1)$ with $f^*(\overline{\xi}, 0) = 0$. This assumes a monotonic classifier and $f^*(0,0) < 0$, which just excludes trivial cases. Under suitable assumptions of a smooth and non-degenerate decision boundary, the function d^* is actually smooth itself and injective, so that it its inverse d^{*-1} exists. By the increasing monotonicity of the classifier, d^* and d^{*-1} are strictly *decreasing*.

Next, we show that given another function f (which can be monotonic) we can construct calibrators that transform its decision boundary d to replicate the target decision boundary d^*:

Theorem 6. *Let f^*, d^*, f and d be as discussed above, and let c_1 be a given, monotonic calibration function. For simplicity, assume $f^*(0,1) = f^*(1,0) = 0$, so that d^* and ${d^*}^{-1}$ are defined on $[0,1]$. The same shall be true also for f and d. Then the calibrated lattice $g(\xi,\zeta) = f(c_1(\xi), c_2(\zeta))$ yields the same classifier as f^* if and only if the calibration functions satisfy*

$$c_2 = d \circ c_1 \circ {d^*}^{-1}. \tag{7}$$

Proof Fix ζ arbitrarily. Then for all ξ, the following are equivalent:

$$f^*(\xi,\zeta) > 0 \;\;\Leftrightarrow\;\; \zeta > d^*(\xi) \;\;\Leftrightarrow\;\; \xi > {d^*}^{-1}(\zeta).$$

Similarly,

$$f(c_1(\xi), c_2(\zeta)) > 0 \;\;\Leftrightarrow\;\; c_2(\zeta) > d(c_1(\xi)).$$

Thus, the classifiers of f^* and g are the same if and only if

$$\xi > {d^*}^{-1}(\zeta) \;\;\Leftrightarrow\;\; c_2(\zeta) > d(c_1(\xi))$$

holds for all ξ. This is the case if and only if

$$c_2(\zeta) = d(c_1({d^*}^{-1}(\zeta))),$$

which finishes the proof.

Our assumption of $f^*(0,1) = f^*(1,0) = 0$ in Theorem 6 clarifies the argument and leads to the elegant form of (7). More general situations can be reduced to this case by scaling the lattice and considering suitable subdomains.

Corollary 7. *For $D = 2$, any monotonic target classifier f^* as above can be reproduced by a monotonic calibrated lattice.*

Proof Choose the lattice f such that

$$f(0,0) < 0, \;\; f(0,1) = f(1,0) = 0, \;\; f(1,1) > 0$$

and note that this is, in particular, a monotonic lattice.

Choose $c_1(\xi) = \xi$. Then note that $c_2 = d \circ {d^*}^{-1}$ constructed according to (7) is the composition of two decreasing functions, which is itself monotonic. The claim now follows from Theorem 6.

6.2 Higher Dimensions: Theoretically Calibration Is Not Sufficient

A similar (although more complex) construction can be applied for $D = 3$ in special cases. This also implies that when we try to match a given target classifier f^* for $D \geq 3$, then all infinite-dimensional degrees of freedom from the calibration functions are completely removed. The only flexibility that remains is in some of the parameters of the lattice f, which is not enough to fully fit an arbitrary target decision in higher dimensions. This suggests that for $D \geq 3$ monotonic calibrated lattice functions will not be sufficient to express every monotonic lattice classifier.

7 Simulations

We investigated via simulations whether this theoretical gap is likely to be important in practice. In our simulations the target classifier is a monotonic lattice classifier defined on $[0, 1]^4$. For each run of the simulation, we randomly generated a new target classifier by randomly independently drawing each of the 2^4 lattice parameters uniformly from $[-1, 1]$, forming the discriminant function $f(x)$ by multilinear interpolation of the 2^4 lattice parameters, and thresholding $f(x)$ at 0 to form a binary classifier. If the resulting binary classifier was degenerate in that it classified everything as one class, we discarded the run. If the resulting binary classifier did not satsify the sufficient quadratic conditions to be monotonic for all four inputs, we discarded the run. In this way, each run of the simulation we had a random monotonic lattice classifier. We then sampled $N = 100$ training examples $x_i \in [0, 1]^D$ uniformly independently randomly from the unit hypercube, and evaluated the classifier to form the training label $y_i = I_{f(x_i)>0}$. We similarly sampled 1000 IID test samples.

For each target classifier, we compared training (i) monotonic lattice function, (ii) monotonic calibrated lattice function, (iii) monotonic lattice, (iv) monotonic calibrated lattice. All training minimized the empirical risk on the training examples using the hinge loss and projected gradient descent with a fixed step size of 0.1 and 10,000 epochs. For the monotonic functions, after each gradient step we projected onto the necessary linear inequality constraints [15]. For the monotonic classifiers, after each gradient step we attempted to project onto the necessary nonconvex set of quadratic inequality constraints by alternately projecting onto convex subsets of the quadratic constraints for 100 rounds. For the calibrated models, we alternately optimized the calibrator layer and lattice layer: each epoch we took a gradient descent step for the lattice parameters (with fixed calibrator parameters) and then took a gradient descent step for the calibrator parameters (with fixed lattice parameters). Once trained, each model was thresholded at $f(x) > 0$ to form a binary classifier.

Table 2 shows the results for 1,000,000 runs of the simulation, which resulted in 51 monotonic classifiers, so the train and test accuracy numbers are averaged over the 51 fitted models and the 51 corresponding sets of 1,000 test samples.

As expected, the monotonic lattice function was not the best performer, as it was not flexible enough to model the true function in some cases. However, the test accuracy of the three other models was not statistically significantly different on average. This is in-line with our theoretical analysis, which suggests that by calibrating a monotonic lattice function one gains substantial flexibility.

8 CONCLUSIONS

We investigated whether there it is possible, and whether it is useful, to train using constrained empirical risk minimization monotonic classifiers without requiring an underlying monotonic discriminant. We focused on lattice models, because they are a state-of-the-art function class for shape-constrained machine learning, and are amenable to analysis. We gave new quadratic inequality constraints that are necessary and sufficient for a $D = 2$ input lattice classifier to be monotonic, and formed tighter

Table 2: Test accuracy for simulated monotonic binary classifiers on the $[0, 1]^4$ domain.

Function Class	**Mean Test Accuracy**
Monotonic Lattice Function	95.4%
Monotonic Lattice Classifier	97.4%
Calibrated Monotonic Lattice Function	97.4%
Calibrated Monotonic Lattice Classifier	97.2%

sufficient conditions than known linear inequality constraints for higher-dimensional models.

However, we showed that one can also express any $D = 2$ monotonic lattice classifier by thresholding a monotonic two-layer *calibrated* lattice function. That is, a *more flexible* monotonic function gives the same classifier expressiveness. While this result does not strictly generalize to higher dimensions, our simulations for $D = 4$ were consistent with this result, and lead us to hypothesize that in practice the most effective way to achieve a good monotonic classifier is by training a more flexible monotonic discriminant function, and then thresholding the monotonic discriminant to produce a monotonic classifier. Here, we used two-layer monotonic lattice functions, but more flexible deep lattice networks You:2017 and multi-cell lattices Garcia:12 can also can be trained by constrained optimization with easy-to-enforce sparse *linear* inequality constraints.

Further, in practice, thresholding monotonic functions has the practical advantage the decision threshold can be tuned after training for a desired recall or precision without breaking the monotonicity property, whereas a monotonic classifier is only guaranteed to be monotonic for the specific decision threshold it was trained for, reducing the ability to adjust the decision threshold after training and still guarantee monotonicity.

However, if latency or CPU is at a premium, then training a one-layer monotonic classifier can be more efficient and thus preferable to thresholding a two-layer monotonic function.

References

1. Archer, N.P., Wang, S.: Application of the back propagation neural network algorithm with monotonicity constraints for two-group classification problems. Decision Sciences **24**(1), 60–75 (1993)
2. Bach, F.: Learning with submodular functions: A convex optimization perspective. Foundations and Trends in Machine Learning **6**(2) (2013)
3. Barlow, R.E., Bartholomew, D.J., Bremner, J.M.: Statistical inference under order restrictions; the theory and application of isotonic regression. Wiley (1972)
4. Bonakdarpour, M., Chatterjee, S., Barber, R.F., Lafferty, J.: Prediction rule reshaping. ICML (2018)
5. Canini, K., Cotter, A., Fard, M.M., Gupta, M.R., Pfeifer, J.: Fast and flexible monotonic functions with ensembles of lattices. Advances in Neural Information Processing Systems (NIPS) (2016)
6. Chen, Y., Samworth, R.J.: Generalized additive and index models with shape constraints. Journal Royal Statistical Society B (2016)

7. Chetverikov, D., Santos, A., Shaikh, A.M.: The econometrics of shape restrictions. Annual Review of Economics (2018)
8. Daniels, H., Velikova, M.: Monotone and partially monotone neural networks. IEEE Trans. Neural Networks **21**(6), 906–917 (2010)
9. Farr, W.: On the construction of life tables, illustrated by a new life table of the healthy districts of england. Journal of the Institute of Actuaries **9**, 121–141 (1860)
10. Feelders, A.J., Pardoel, M.: Pruning for monotone classification trees. Springer Lecture Notes on Computer Science **2810**, 1–12 (2003)
11. Garcia, E.K., Arora, R., Gupta, M.R.: Optimized regression for efficient function evaluation. IEEE Trans. Image Processing **21**(9), 4128–4140 (2012)
12. Garcia, E.K., Gupta, M.R.: Lattice regression. In: Advances in Neural Information Processing Systems (NIPS) (2009)
13. Groeneboom, P., Jongbloed, G.: Nonparametric estimation under shape constraints. Cambridge Press, New York, USA (2014)
14. Gupta, M.R., Bahri, D., Cotter, A., Canini, K.: Diminishing returns shape constraints for interpretability and regularization. Advances in Neural Information Processing Systems (NeurIPS) (2018)
15. Gupta, M.R., Cotter, A., Pfeifer, J., Voevodski, K., Canini, K., Mangylov, A., Moczydlowski, W., Esbroeck, A.V.: Monotonic calibrated interpolated look-up tables. Journal of Machine Learning Research **17**(109), 1–47 (2016). URL `http://jmlr.org/papers/v17/15-243.html`
16. Gupta, M.R., Gray, R.M., Olshen, R.A.: Nonparametric supervised learning by linear interpolation with maximum entropy. IEEE Trans. on Pattern Analysis and Machine Intelligence **28**(5), 766–781 (2006)
17. Gupta, M.R., Pfeifer, J., You, S.: Tensor flow lattice: Flexibility empowered by prior knowledge (2017). URL `https://ai.googleblog.com/2017/10/tensorflow-lattice-flexibility.html`
18. Kalai, A.T., Sastry, R.: The isotron algorithm: High-dimensional isotonic regression. Conference on Learning Theory (COLT) (2009)
19. Luss, R., Rosset, S.: Bounded isotonic regression. Electronic Journal of Statistics **11**(2), 4488–4514 (2017)
20. Perry, J.: Practical Mathematics. Wiley and Sons (1899)
21. Potharst, R., Feelders, A.J.: Classification trees for problems with monotonicity constraints. ACM SIGKDD Explorations pp. 1–10 (2002)
22. Pya, N., Wood, S.N.: Shape constrained additive models. Statistics and Computing (2015)
23. Sang, E.: On last-place errors in Vlacq's table of logarithms. Proceedings of the Royal Society of Edinburgh **8**, 371–376 (1875)
24. Sharma, G., Bala, R.: Digital Color Imaging Handbook. CRC Press, New York (2002)
25. Sill, J., Abu-Mostafa, Y.S.: Monotonicity hints. Advances in Neural Information Processing Systems (NIPS) pp. 634–640 (1997)
26. Weiser, A., Zarantonello, S.E.: A note on piecewise linear and multilinear table interpolation in many dimensions. Mathematics of Computation **50**(181), 189–196 (1988)
27. Wellner, J.A.: Some theory for estimation with shape constraints (2008). URL `https://www.stat.washington.edu/jaw/RESEARCH/TALKS/niad.pdf`
28. You, S., Canini, K., Ding, D., Pfeifer, J., Gupta, M.R.: Deep lattice networks. Advances in Neural Information Processing Systems (NIPS) (2017)