Convex Formulations of Finite-Width Neural Networks

Haruka Watanabe

Contents

1 Introduction

The achievements of neural networks are familiar to us in our daily lives [MD23]. These achievements include image recognition [ERH02] [Law+97], search engines [Ser19] or transportation [Dou95] [PK17] [Yu+17]. Recent achievements in generative models, such as image generative models [Alo20] or large language models [Nav+23] [Rai+24], are spectacular: they produce images or texts as if humans produced. These achievements have generated a growing interest in understanding the efficacy of neural networks [Sam+21]. However, mathematical arguments to support the reasons for this good behaviour remain elusive [Gar22] [EP21b].

This essay aims to introduce you to some theories that look at the optimisation problems for neural networks from a viewpoint of convex optimisation. These theories help you understand the inner workings of hidden neurons from two perspectives: high-dimensional feature selection and Clifford algebra.

1.1 Overview

We start with mathematical backgrounds in Section 1.2 to introduce notations and the central theme of this essay. In Section 2, we provide a step-by-step proof for the exact convex formulation of two-layer ReLU neural networks introduced by Pilanci and Ergen [PE20]. We have invented numerical experiments to confirm the global minimality of the convex formulation and assess the performance of minimisation by stochastic gradient descent. Interestingly, we observed that minimisation by stochastic gradient descent was more likely to reach the global minimum suggested by the convex formulation for noisy data than for noiseless data. Furthermore, our results indicated the superiority of minimisation by the convex formulation in a random design as well. These observations might be supported theoretically in future research. In Section 3, we focus on an intuitive understanding of the inner workings of hidden neurons. We also discuss the similarity between the neural network training process and other machine learning techniques.

The numerical results in various settings (Section 2.3) and the visualisations from simple examples (Section 3) will help you gain insight and intuition into the training process of neural networks.

1.2 Mathematical backgrounds

A basic goal of neural networks is to predict the unknown output of new input data by learning the known relationships between the training input data and their outputs. The neural networks usually take in a large amount of data and are equipped with many parameters to improve the prediction. We formalise this process mathematically.

Given a training data matrix $X \in \mathbb{R}^{n\times d}$ and a training output $y \in \mathbb{R}^n$, we want to construct a function $f : \mathbb{R}^d \to \mathbb{R}$ that gives a "good prediction" $f(x)$ for the output of new data x. Here, we consider d-dimensional input and 1-dimensional output. The ith row x_i^T of X and the ith element y_i of y represent the ith input training data and the ith output training data, respectively. An architecture of deep neural network (DNN) (with K-subnetworks, each of which is an L-layer unbiased ReLU network with weights $W_{lk} \in \mathbb{R}^{m_{l-1}\times\ m_l}$ where $l \in [L]$, $k \in [K]$, $m_0 = d$, $m_L = 1$) can be formulated as $f_\theta(X) = \sum_{k=1}^K f_{\theta,k}(X)$ by $f_{\theta,k} : \mathbb{R}^{n\times d} \to \mathbb{R}^n$ where

$$f_{\theta,k}(X) \coloneqq ((XW_{1k})_+ ... W_{(L-1)k})_+ W_{Lk}. \tag{1.1}$$

For notation, X is an $n \times d$ data matrix, θ denotes the weight parameters $\{\{W_{lk}\}_{l=1}^{L}\}_{k=1}^{K}$, and $(.)_+$ denotes the elementwise ReLU function. (For a matrix M, $(M)_+$ is a matrix whose ijth element is $\max\{M_{ij}, 0\}$). We want to find θ such that $\forall\ x \in \mathbb{R}^d$, $\sum_{k=1}^{K}((x^T W_{1k})_+ ... W_{(L-1)k})_+ W_{Lk}$ is a "good prediction" of the corresponding output.

So far, we have used the term "good prediction". We want to quantitatively assess how good our prediction is. To this end, we introduce a loss function $\mathcal{L}$ to evaluate the difference between the prediction $f(x)$ and the true output $y(x)$. There are various loss functions, such as a squared loss (usually for regression) $\mathcal{L}(f(x), y(x)) = (f(x) - y(x))^2$ or 0-1 loss function (usually for classification) $\mathcal{L}(f(x), y(x)) = \mathbb{1}(f(x) \neq y(x))$. The loss of the training data can be denoted by $\mathcal{L}(f_\theta(X), y)$. For example, a squared loss can be written as $\mathcal{L}(f_\theta(X), y) = \| \sum_{k=1}^{K} f_{\theta,k}(X) - y\|_2^2 = \sum_{i=1}^{n}((\sum_{k=1}^{K}(f_{\theta,k}(X)))_i - y_i)^2$. We take the loss function $\mathcal{L}(.,.)$ as an arbitrary convex function.

A naive way to find an optimal θ is to find θ that minimises the loss in the training set. It is to solve

$$\min_{\theta \in \Theta} \mathcal{L}(f_\theta(X), y)$$

where Θ is the parameter space. However, as this optimisation heavily depends on training data, it can lead to overfitting and poor generalisation performance. To avoid this problem, various regularisation methods, such as Ridge regression [E H70] or Lasso [Tib96], can be used. The regularised network training problem can be formulated as

$$\min_{\theta \in \Theta} \mathcal{L}(f_\theta(X), y) + \beta \sum_{k=1}^{K} \mathcal{R}_k(\theta) \tag{1.2}$$

where $\mathcal{R}_k(.)$ is the regularisation function for the layer weights in the kth sub-network, and $\beta > 0$ is a regularisation parameter.

We aim to find an optimal $\theta \in \Theta$ which minimises the above value (1.2). Classical machine learning theory often relies on empirical risk minimisation methods, such as stochastic gradient descent [S A67] with backpropagation [Ama93]. It works when the objective function in (1.2) is convex; however, when the objective function is non-convex, it may converge to a local minimum [SSS17]. Safran and Shamir [SS18] provided a computer-assisted proof that the optimisation problem with squared loss for ReLU neural networks can have spurious local minima when the number of parameters is small, and conducted experiments showing that the probability of stochastic gradient descent hitting such local minima is quite high.

On the other hand, empirical results suggest that neural networks generalise better with overparametrisation [Ney+18]. For example, increased depth of neural networks has been reported to speed up optimisation [ACH18]. A figure illustrating the increase in the number of sub-networks leads to a less non-convex loss landscape can be found in [EP21b].

The research aiming to elucidate the relationship between overparametrised ReLU networks and their generalisation performance has been gaining importance. For example, it is suggested that early stopping of gradient descent or stochastic gradient descent might be working as a form of regularisation [JL18]. This essay delves into characterising the global optima of neural network problems as solutions to high-dimensional convex optimisation problems. This essay also illustrates the relationship between

strong duality and the number of hidden neurons. This perspective can shed light on the connection between overparameterised neural networks and their effectiveness.

1.3 Related work

There have been attempts to find global optima of neural network problems. We present an example here.

Particle Gradient Descent
Consider the problem of training neural networks with a single hidden layer in a more general setting:

Find an element in a Hilbert space $\mathcal{F}$, square-integrable real-valued functions on $\mathbb{R}^d$, that minimises the loss $\mathcal{L}$ and that is a linear combination of a few elements from a large given parametrised set $\{\phi(\theta)\}_{\theta\in\Theta} \subset \mathcal{F}$ where $\phi(x) = \left(\sum_{i=1}^{d} x_i\theta_i + \theta_{d+1}\right)_+$.

This problem can be generalised to describe the linear combination through an unknown signed measure μ on Θ to solve for

$$J^* = \min_{\mu\in\mathcal{M}(\Theta)} J(\mu), \qquad J(\mu) \coloneqq \mathcal{L}\left(\int \phi \, d\mu\right) + G(\mu) \tag{1.3}$$

where $\mathcal{M}(\Theta)$ is the set of signed measures on Θ and G is an optional convex regulariser.
An approach for this problem is to discretise the unknown measure μ as a mixture of m particles parameterised by their positions and weights so that

$$\min_{\substack{\omega\in\mathbb{R}^m \\ \theta\in\Theta^m}} J_m(\omega,\ \theta), \qquad J_m(\omega,\ \theta) \coloneqq J\left(\frac{1}{m}\sum_{i=1}^{m} \omega_i \delta_{\theta_i}\right). \tag{1.4}$$

Chizat and Bach [CB18] studied the many particle limit $m \to \infty$ of the gradient flow of J_m. They showed that, under some assumptions, the gradient flow converges to global minimisers.

The importance of this approach lies in the fact that our training problem can be viewed as a more general optimisation problem, and global optimality can be proven in the asymptotic regime. However, networks with infinite width are not practical. Pilanci and Ergen [PE20] [EP21a] showed that a global minimum for finite-width networks can be found through convex formulations.

2 Convex formulations

In this section, we restrict our attention to two-layer unbiased ReLU neural networks ($K = 1$ and $L = 2$ in (1.1)) with scalar output trained with squared loss and weight decay regularisation. The squared loss is often used in a regression setting.

Our Neural Network (NN) model is

$$f(X) = \sum_{j=1}^{m} (Xu_j)_+ \omega_j. \tag{2.1}$$

($W_{11} = (u_1,\ \dots, u_m) \in \mathbb{R}^{d\times m}$ and $W_{21} = (\omega_1,\ \dots, \omega_m)^T \in \mathbb{R}^{m\times 1}$ in (1.1))

Our objective function is

$$\frac{1}{2}\|\sum_{j=1}^{m}(Xu_j)_+\omega_j - y\|_2^2 + \frac{\beta}{2}\sum_{j=1}^{m}(\|u_j\|_2^2 + \omega_j^2). \tag{2.2}$$

Our goal is to find an optimal solution $\{u_i^*, \omega_i^*\}_{i=1}^m$ to the optimisation problem

$$p^* := \min_{\{u_j,\omega_j\}_{j=1}^m} \frac{1}{2}\|\sum_{j=1}^{m}(Xu_j)_+\omega_j - y\|_2^2 + \frac{\beta}{2}\sum_{j=1}^{m}(\|u_j\|_2^2 + \omega_j^2). \tag{2.3}$$

This problem is a non-convex optimisation problem, so applying classical machine learning theory methods that employ a local heuristic search might get stuck in a local minimum of the objective function before reaching its global minimum.

Pilanci and Ergen [PE20] showed the following novel theorem. The theorem states that when the number of hidden neurons is large enough, this non-convex optimisation problem can be formulated as a convex optimisation problem, which can be globally optimised by standard convex optimisation techniques. The theorem also finds the connection between the optimal minimiser of the convex optimisation problem and the optimal minimiser of the original non-convex optimisation problem.

The importance of this theorem is that it enables the formulation of the non-convex optimisation problem for overparametrised neural networks as a convex problem with finite constraints. The strong duality for overparametrised neural networks might play a key role in explaining why neural networks perform well when they are overparametrised.

We use $\mathbf{1}[Xu \geq 0]$ to denote the element-wise 0-1 value indicator.

Theorem 2.1. *Consider a diagonal matrix $Diag(\mathbf{1}[Xu \geq 0])$ for $u \in \mathbb{R}^d$. Let $D_1, D_2, ..., D_P$ denote all such distinct diagonal matrices that can be obtained for all possible $u \in \mathbb{R}^d$ i.e. $\{D_1, D_2, ..., D_P\} = \{Diag(\mathbf{1}[Xu \geq 0]) \mid \forall u \in \mathbb{R}^d\}$. We formulate a finite-dimensional convex problem as*

$$\min_{\{v_i,t_i\}_{i=1}^P} \frac{1}{2}\|\sum_{i=1}^{P} D_iX(v_i - t_i) - y\|_2^2 + \beta\sum_{i=1}^{P}(\|v_i\|_2 + \|t_i\|_2) \quad s.t. \quad (2D_i - I_n)Xv_i \geq 0, (2D_i - I_n)Xt_i \geq 0, \ \forall\, i. \tag{2.4}$$

The convex problem (2.4) and the non-convex problem (2.3) have identical optimal values when $m \geq m^*$ for some $0 < m^* < n$. Moreover, an optimal solution $\{u_i^*, \omega_i^*\}_{i=1}^{m^*}$ to (2.3) with m^* neurons can be constructed from an optimal solution $\{v_i^*, t_i^*\}_{i=1}^P$ to (2.4) as follows

$$(u_{1i}^*, \omega_{1i}^*) = \left(\frac{v_i^*}{\sqrt{\|v_i^*\|_2}}, \sqrt{\|v_i^*\|_2}\right) \quad if \quad v_i^* \neq 0,$$
$$(u_{2i}^*, \omega_{2i}^*) = \left(\frac{t_i^*}{\sqrt{\|t_i^*\|_2}}, \sqrt{\|t_i^*\|_2}\right) \quad if \quad t_i^* \neq 0.$$

So, $m^* = \sum_{i:v_i^*\neq 0}^{P} 1 + \sum_{i:t_i^*\neq 0}^{P} 1$.

2.1 Why do we want the convex formulation?

The convex formulation has advantages in the following points.

- **Suboptimality**
  The convex problem (2.4) can be globally optimised without tuning hyperparameters such as learning rate or starting point. Hence, the optimised value can be used to assess the suboptimality of other minimisation methods, such as SGD. This is illustrated in Section 2.3.

- **Computational Complexity**
  It can be shown (details are in Section 3.1) that

$$P \leq 2\sum_{k=0}^{r-1}\binom{n}{k} \leq 2r\left(\frac{e(n-1)}{r}\right)^r$$

  where $r \coloneqq rank(X)$ ([Ojh00]; [MRS]; [Win66]; [Cov65]). Two-layer ReLU networks with m hidden neurons can be globally optimised through the second-order cone program† (2.4) with $2dP$ variables and $2nP$ linear inequalities. Hence, the cost per iteration for standard interior-point solvers is at most $\mathcal{O}(d^3r^3(\frac{n}{r})^{3r})$.

  Note that, for fixed rank r, the complexity is polynomial in n and m. This result is a significant improvement compared to previous results [Aro+18] or [BMP23]

2.2 Proof of the exact convex formulation

In this section, we prove Theorem 2.1 and the supporting lemmas. The proof of Theorem 2.1 consists of stating the original problem in an equivalent form and taking advantage of the convex duality. Although the original proofs by Pilanci and Ergen [PE20] do not explicitly use the notation of Lagrangian, we introduce it in our proofs for clarity. The preliminary knowledge of optimisation theory required for the proofs in this section is listed in the Appendix in Section 5.1.

We begin with formulating the original problem (2.3) as a Lasso optimisation problem.

Lemma 2.1. *The non-convex problem (2.2) has an equivalent form*

$$p^* = \min_{\substack{\|u_j\|_2\leq 1\\ \forall j\in[m]}} \min_{\{\omega_j\}_{j=1}^m} \frac{1}{2}\|\sum_{j=1}^m (Xu_j)_+\omega_j - y\|_2^2 + \beta\sum_{j=1}^m |\omega_j|. \tag{2.5}$$

Proof. The key idea of this proof is that the rescalings of parameters

$$\forall j \in [m], \quad \bar{u_j} = \gamma_j u_j, \quad \bar{\omega_j} = \omega_j/\gamma_j \quad \text{for some} \quad \gamma_j > 0 \tag{2.6}$$

do not change the output value $\sum_{j=1}^m (Xu_j)_+\omega_j$, so $\|\sum_{j=1}^m (Xu_j)_+\omega_j - y\|_2^2 = \|\sum_{j=1}^m (X\bar{u_j})_+\bar{\omega_j} - y\|_2^2$. Also, they do not change the sum $\sum_{j=1}^m |\omega_j|\|u_j\|_2 = \sum_{j=1}^m |\bar{\omega_j}|\|\bar{u_j}\|_2$.

Basic inequality $\frac{1}{2}\sum_{j=1}^m(\omega_j^2 + \|u_j\|_2^2) \geq \sum_{j=1}^m |\omega_j|\|u_j\|_2$ implies that

$$p^* = \min_{\{\omega_j,u_j\}_{j=1}^m} \frac{1}{2}\|\sum_{j=1}^m (Xu_j)_+\omega_j - y\|_2^2 + \frac{\beta}{2}\sum_{j=1}^m(\|u_j\|_2^2+\omega_j^2) \geq \min_{\{\omega_j,u_j\}_{j=1}^m} \frac{1}{2}\|\sum_{j=1}^m (Xu_j)_+\omega_j - y\|_2^2 + \beta\sum_{j=1}^m |\omega_j|\|u_j\|_2. \tag{2.7}$$

† See [Lob+98] or [AG03] for more details.

By rescaling with $\gamma_j = \sqrt{\frac{|\omega_j|}{\|u_j\|_2}}$ in (2.6), we may find an optimal solution $\{\omega_j^*, u_j^*\}_{j=1}^m$ to the right-hand side in (2.7) such that they satisfy $\forall j \in [m], |\omega_j^*| = \|u_j^*\|_2$. As $|\omega_j^*| = \|u_j^*\|_2$ holds for all $j \in [m]$, the equality in the basic inequality holds. Hence,

$$\begin{aligned}
&\min_{\{\omega_j,u_j\}_{j=1}^m} \frac{1}{2}\|\sum_{j=1}^m (Xu_j)_+\omega_j - y\|_2^2 + \beta\sum_{j=1}^m |\omega_j|\|u_j\|_2 \\
=&\frac{1}{2}\|\sum_{j=1}^m (Xu_j^*)_+\omega_j^* - y\|_2^2 + \beta\sum_{j=1}^m |\omega_j^*|\|u_j^*\|_2 \\
=&\frac{1}{2}\|\sum_{j=1}^m (Xu_j^*)_+\omega_j^* - y\|_2^2 + \frac{\beta}{2}\sum_{j=1}^m (\|u_j^*\|_2^2 + \omega_j^{*2}) \\
\geq& \min_{\{\omega_j,u_j\}_{j=1}^m} \frac{1}{2}\|\sum_{j=1}^m (Xu_j)_+\omega_j - y\|_2^2 + \frac{\beta}{2}\sum_{j=1}^m (\|u_j\|_2^2 + \omega_j^2).
\end{aligned}$$

Compared with (2.7), the equality holds in the last line, so

$$p^* = \min_{\{\omega_j,u_j\}_{j=1}^m} \frac{1}{2}\|\sum_{j=1}^m (Xu_j)_+\omega_j - y\|_2^2 + \frac{\beta}{2}\sum_{j=1}^m (\|u_j\|_2^2 + \omega_j^2) = \min_{\{\omega_j,u_j\}_{j=1}^m} \frac{1}{2}\|\sum_{j=1}^m (Xu_j)_+\omega_j - y\|_2^2 + \beta\sum_{j=1}^m |\omega_j|\|u_j\|_2.$$

Again, by rescaling with $\gamma_j = \frac{1}{\|u_j\|_2}$ in (2.6), we may find an optimal solution $\{\bar{\omega_j}, \bar{u_j}\}_{j=1}^m$ to the right-hand side in (2.7) such that $\forall j, \|\bar{u}\|_j = 1$. Hence,

$$\begin{aligned}
\min_{\{\omega_j,u_j\}_{j=1}^m} \frac{1}{2}\|\sum_{j=1}^m (Xu_j)_+\omega_j - y\|_2^2 + \beta\sum_{j=1}^m |\omega_j|\|u_j\|_2 &= \frac{1}{2}\|\sum_{j=1}^m (X\bar{u_j})_+\bar{\omega_j} - y\|_2^2 + \beta\sum_{j=1}^m |\bar{\omega_j}|\|\bar{u_j}\|_2 \\
&= \frac{1}{2}\|\sum_{j=1}^m (X\bar{u_j})_+\bar{\omega_j} - y\|_2^2 + \beta\sum_{j=1}^m |\bar{\omega_j}| \\
&= \min_{\substack{\|u_j\|_2=1 \\ \forall j\in[m]}} \min_{\{\omega_j\}_{j=1}^m} \frac{1}{2}\|\sum_{j=1}^m (Xu_j)_+\omega_j - y\|_2^2 + \beta\sum_{j=1}^m |\omega_j|.
\end{aligned}$$

Now, we show that we can replace the condition $\|u_j\|_2 = 1$ with $\|u_j\|_2 \leq 1$.
Let $\{\tilde{\omega_j}, \tilde{u_j}\}_{j=1}^m$ be an optimal solution to

$$\min_{\substack{\|u_j\|_2\leq 1 \\ \forall j\in[m]}} \min_{\{\omega_j\}_{j=1}^m} \frac{1}{2}\|\sum_{j=1}^m (Xu_j)_+\omega_j - y\|_2^2 + \beta\sum_{j=1}^m |\omega_j|.$$

If $\exists\ j$ s.t. $\|\tilde{u_j}\|_2 < 1$ and $\tilde{\omega}_j \neq 0$, we can find $\epsilon > 0$ s.t. $|\tilde{\omega_j}| > (\|\tilde{u_j}\|_2 + \epsilon)|\tilde{\omega_j}|$.
For such $\epsilon > 0$,

$$\begin{cases} (X\tilde{u_j})_+\tilde{\omega_j} = (X\frac{\tilde{u_j}}{\|\tilde{u_j}\|+\epsilon})_+(\|\tilde{u_j}\| + \epsilon)\omega_j \\ |\omega_j| > (\|\tilde{u_j}\|_2 + \epsilon)|\omega_j| \end{cases}$$

so we can make the objective value smaller by replacing $(\tilde{\omega_j}, \tilde{u_j})$ with $((\|\tilde{u_j}\| + \epsilon)\omega_j, \frac{\tilde{u_j}}{(\|\tilde{u_j}\|+\epsilon)})$. This contradicts that $\{\tilde{\omega_j}, \tilde{u_j}\}_{j=1}^m$ is an optimal solution.
Hence, for optimal $\{\tilde{\omega_j}, \tilde{u_j}\}_{j=1}^m$,

$$\tilde{\omega_j} \neq 0 \Rightarrow \|\tilde{u_j}\|_2 = 1.$$

Hence,

$$\min_{\substack{\|u_j\|_2=1 \\ \forall j\in[m]}} \min_{\{\omega_j\}_{j=1}^m} \frac{1}{2}\|\sum_{j=1}^m (Xu_j)_+\omega_j - y\|_2^2 + \frac{\beta}{2}\sum_{j=1}^m |\omega_j| = \min_{\substack{\|u_j\|_2\le 1 \\ \forall j\in[m]}} \min_{\{\omega_j\}_{j=1}^m} \frac{1}{2}\|\sum_{j=1}^m (Xu_j)_+\omega_j - y\|_2^2 + \frac{\beta}{2}\sum_{j=1}^m |\omega_j|.$$

□

Remark 2.1. Solving Lasso optimisation (2.5) encourages some of the values of $|\omega_j|$ ($j \in [m]$) to be zero. Lemma 2.1 motivates us to interpret the neural network training process as a variable selection.

The following Lemma 2.2 finds the convex dual of the optimisation problem (2.3) by finding the dual of the optimisation problem (2.5). By weak duality, it gives a lower bound for our objective value p^*. For notation, we introduce $\mathcal{B}_p^d$ to denote the unit p-norm ball around the origin in $\mathbb{R}^d$, and $\mathbf{B}_2$ to denote the unit 2-norm ball $\mathcal{B}_2^d = \{u \in \mathbb{R}^d |\ \|u\|_2 \le\ 1\}$.

Lemma 2.2. *The non-convex problem (2.3) has a convex dual formulation*

$$p^* \ge d^* := \max_{v\in\mathbb{R}^n} -\frac{1}{2}\|y-v\|_2^2 + \frac{1}{2}\|y\|_2^2 \quad s.t. \quad |v^T(Xu)_+| \le \beta \quad \forall\ u \in \mathbf{B}_2. \tag{2.8}$$

Proof. For now, we treat u_j as a fixed vector in $\mathbf{B}_2$ for all $j \in [m]$.

We first show that the dual of the primal optimisation problem

$$\min_{\{\omega_j\}_{j=1}^m} \frac{1}{2}\|\sum_{j=1}^m (Xu_j)_+\omega_j - y\|_2^2 + \beta\sum_{j=1}^m |\omega_j| \tag{2.9}$$

is the problem

$$\max_{\substack{v\in\mathbb{R}^n\ s.t. \\ |v^T(Xu_j)_+|\le\beta\ \forall j\in[m]}} -\frac{1}{2}\|y-v\|_2^2 + \frac{1}{2}\|y\|_2^2.$$

The primal optimisation problem (2.9) is equivalent to

$$\min_{\{\omega_j\}_{j=1}^m} \frac{1}{2}\|r\|_2^2 + \beta\sum_{j=1}^m |\omega_j| \quad \text{s.t.} \quad r = \sum_{j=1}^m (Xu_j)_+\omega_j - y. \tag{2.10}$$

The Lagrangian is defined as

$$L(v,r,\omega) = \frac{1}{2}\|r\|_2^2 + \beta\sum_{j=1}^m |\omega_j| + v^T(r + y - \sum_{j=1}^m (Xu_j)_+\omega_j)$$

where $\omega = (\omega_1,\ \dots\ ,\omega_m)^T \in \mathbb{R}^m$ and $v \in \mathbb{R}^n$.

Define $\Lambda := \{v \in \mathbb{R}^n\ :\ \min_{r\in\mathbb{R}^n,\ \omega\in\mathbb{R}^m} L(v,r,\omega) > -\infty\}$ and, for $v \in \Lambda$, define
$L(v) := \min_{r\in\mathbb{R}^n,\ \omega\in\mathbb{R}^m} L(v,r,\omega)$.

We are going to find $L(v)$.
We write $\hat{r}, \hat{\omega}$ to denote parameter values of r and ω which minimise $L(v,r,\omega)$ for fixed $v \in \Lambda$.

$$\nabla_r L = r + v = 0 \text{ at } \hat{r}\ \Rightarrow \hat{r} = -v \tag{2.11}$$

$$0 \in \partial_{\omega_j} L = \{\beta W_j - v^T(Xu_j)_+\ :\ \|W_j\|_\infty \le 1,\ W_j = sgn(\omega_j) \text{ if } \omega_j \ne 0\} \text{ at } \hat{\omega_j} \tag{2.12}$$

$$\Rightarrow (\beta\ sgn(\hat{\omega_j}) - v^T(Xu_j)_+)\hat{\omega_j} = 0 \tag{2.13}$$

Using (2.11) and (2.13),

$$\begin{aligned} L(v) &= L(v, \hat{r}, \hat{\omega}) \\ &= L(v, -v, \hat{\omega}) \\ &= -\frac{\|v\|_2^2}{2} + v^T y + \sum_{j=1}^{m} (\beta \ sgn(\hat{\omega_j}) - v^T (Xu_j)_+) \hat{\omega_j} \\ &= -\frac{\|v\|_2^2}{2} + v^T y \\ &= -\frac{1}{2}\|y - v\|_2^2 + \frac{1}{2}\|y\|_2^2. \end{aligned}$$

For $\min_{r \in \mathbb{R}^n, \omega \in \mathbb{R}^m} L(v, r, \omega) > -\infty$ to hold, it is necessary that L has minimum value at some finite ω i.e. $\forall j \in [m],\ \exists\ \hat{\omega_j} < \infty\ s.t.$ (2.12) holds. This holds iff $\forall j \in [m],\ |v^T (Xu_j)_+| \leq \beta$ Hence, $\Lambda \subseteq \{v : \forall j \in [m],\ |v^T (Xu_j)_+| \leq \beta\}$. Note that for any $v \in \{v \in \mathbb{R}^n : \forall j \in [m],\ |v^T (Xu_j)_+| \leq \beta\}$, $\nabla_r L = 0$ is achieved by taking $r = -v$. Hence,

$$\Lambda = \{v \in \mathbb{R}^n : \forall j \in [m],\ |v^T (Xu_j)_+| \leq \beta\}.$$

Therefore, the dual of (2.10) is

$$\max_{v \in \Lambda} L(v)$$

i.e.

$$\max_{\substack{v \in \mathbb{R}^n s.t. \\ |v^T (Xu_j)_+| \leq \beta\ \forall j \in [m]}} -\frac{1}{2}\|y - v\|_2^2 + \frac{1}{2}\|y\|_2^2. \tag{2.14}$$

By the weak duality, (2.9) is no less than (2.14).
By minimising with respect to $\{u_j\}_{j=1}^m$ on both sides, and by Lemma 2.1,

$$\begin{aligned} p^* &\geq \min_{\substack{\|u_j\|_2 \leq 1 \\ \forall j \in [m]}} \max_{\substack{v \in \mathbb{R}^n s.t. \\ |v^T (Xu_j)_+| \leq \beta\ \forall j \in [m]}} -\frac{1}{2}\|y - v\|_2^2 + \frac{1}{2}\|y\|_2^2 \\ \Rightarrow p^* &\geq \max_{v \in \mathbb{R}^n} -\frac{1}{2}\|y - v\|_2^2 + \frac{1}{2}\|y\|_2^2 \quad \text{s.t.} \quad |v^T (Xu)_+| \leq \beta \quad \forall\ u \in \mathbf{B}_2. \end{aligned}$$

□

Remark 2.2. Lemma 2.2 tells us that the dual of the original non-convex optimisation problem is a convex semi-infinite optimisation problem. A convex optimisation problem with infinitely many constraints is generally not polynomial-time tractable. The usefulness of Theorem 2.1 is that it formulates the problem as a tractable polynomial-time problem.

Now, we prove Theorem 2.1 by showing that the convex problem (2.4) is a strong dual of the convex problem (2.8). From the result in Lemma 2.2, (2.4) gives a lower bound for (2.3). Equality is proven by a direct substitution when m is large.

Proof of Theorem 2.1:

Proof. We first prove that the strong dual of the convex problem (2.8) is the convex problem (2.4). The key idea of this proof is rewriting the condition with infinitely many constraints as a condition with finitely many constraints.

The condition $v \in \{v \in \mathbb{R}^n : \forall\ u \in \mathbf{B}_2,\ |v^T(Xu)_+| \leq \beta\}$ can be written as

$$v \in \{v \in \mathbb{R}^n : \max_{u \in \mathbf{B}_2} v^T(Xu)_+ \leq \beta\} \cap \{v \in \mathbb{R}^n : \max_{u \in \mathbf{B}_2} -v^T(Xu)_+ \leq \beta\}.$$

We introduce some notations. For $S \subset \{1, 2, ...n\}$ and $X \in \mathbb{R}^{n \times d}$,

$$\begin{aligned}
D(S) &\coloneqq \text{A diagonal matrix with } D(S)_{ii} \coloneqq \begin{cases} 1 & (i \in S) \\ 0 & (i \notin S) \end{cases} \\
P_S &\coloneqq \{u \in \mathbb{R}^d : x_i^T u \geq 0\ \forall i \in S,\ x_j^T u \leq 0\ \forall j \in S^c\} \\
&= \{u \in \mathbb{R}^d : D(S)Xu \geq 0,\ (I - D(S))Xu \leq 0\} \\
\mathcal{H}_X &\coloneqq \cup\{\{sgn(Xu)\} : u \in \mathbb{R}^d\} \\
\mathcal{S}_X &\coloneqq \{\{\cup_{h_i=1}\{i\}\} : h \in \mathcal{H}_X\}
\end{aligned}$$

$\mathcal{H}_X$ corresponds to the set of all hyperplane arrangements patterns for X. $\mathcal{S}_X = \{S_1, S_2, ..., S_M\}$ for some subsets $S_1, S_2, ..., S_M$ of $\{1, 2, ..., n\}$ where $M = |\mathcal{H}_X|$.

We can rewrite the condition $\max_{u \in \mathbf{B}_2} v^T(Xu)_+ \leq \beta$ by using the above notations.

$$\begin{aligned}
&\max_{u \in \mathbf{B}_2} v^T(Xu)_+ \leq \beta \\
\Leftrightarrow &\max_{u \in \mathbf{B}_2} v^T Diag(\mathbf{1}(Xu \geq 0))Xu \leq \beta \\
\Leftrightarrow &\max_{S \in \mathcal{S}_X} \max_{u \in \mathbf{B}_2 \cap P_S} v^T D(S)Xu \leq \beta \\
\Leftrightarrow &\forall\ i \in [M], \max_{u \in \mathbf{B}_2 \cap P_{S_i}} v^T D(S_i)Xu \leq \beta \\
\Leftrightarrow &\forall\ i \in [M],\ \exists\ (\alpha_i, \beta_i) \in \mathbb{R}^n \text{ s.t. } \alpha_i \geq 0,\ \beta_i \geq 0, \\
&\quad \|X^T D(S_i)(v + \alpha_i + \beta_i) - X^T \beta_i\|_2 \leq \beta.
\end{aligned}$$

The last equivalence holds according to a strong duality result shown in Lemma 4 in [PE20] applied to the optimisation problem $\max_{u \in \mathbf{B}_2 \cap P_{S_i}} v^T D(S_i)Xu$.

This formulation turns the infinitely many constraints over $u \in \mathbf{B}_2$ into a finite number of constraints.

$$\begin{aligned}
\max_{\substack{v \in \mathbb{R}^n \\ (\alpha_i, \beta_i) \in \mathbb{R}^n \\ \alpha_i \geq 0, \beta_i \geq 0, \forall i \in [M] \\ (\alpha_i', \beta_i') \in \mathbb{R}^n \\ \alpha_i' \geq 0, \beta_i' \geq 0, \forall i \in [M]}} \quad & -\frac{1}{2}\|v - y\|_2^2 + \frac{1}{2}\|y\|_2^2 \\
\text{s.t. } \forall i \in [M],\ & \|X^T D(S_i)(v + \alpha_i + \beta_i) - X^T \beta_i\|_2 \leq \beta \\
\forall i \in [M],\ & \|X^T D(S_i)(-v + \alpha_i' + \beta_i') - X^T \beta_i'\|_2 \leq \beta.
\end{aligned}$$

Note that the constraints are of the form $\|X^T D(S_i)(v + \alpha_i + \beta_i) - X^T \beta_i\|_2 - \beta \leq 0$
where $\|X^T D(S_i)(v + \alpha_i + \beta_i) - X^T \beta_i\|_2 - \beta$ is convex in v, α_i, β_i. Also, a particular choice of parameters $v = 0,\ (\alpha_i, \beta_i) = (0, 0),\ (\alpha_i', \beta_i') = (0, 0)\ \forall i \in [M]$ are strictly feasible in the constraints as long as $\beta > 0$. Hence, strong duality holds by Sler's condition [BV04].

We can construct the Lagrangian and then the dual of the optimisation problem (2.8) as

$$\begin{aligned}
&\min_{\substack{\lambda_i,\lambda_i'\in\mathbb{R}\\ \lambda_i,\lambda_i'\geq 0\\ \forall i\in[M]}} \max_{\substack{v\in\mathbb{R}^n\\ (\alpha_i,\beta_i)\in\mathbb{R}^n\\ \alpha_i\geq 0,\beta_i\geq 0,\forall i\in[M]\\ (\alpha_i',\beta_i')\in\mathbb{R}^n\\ \alpha_i'\geq 0,\beta_i'\geq 0,\forall i\in[M]}} -\frac{1}{2}\|v-y\|_2^2+\frac{1}{2}\|y\|_2^2\\
&+\sum_{i=1}^{M}\lambda_i(\beta-\|X^TD(S_i)(v+\alpha_i+\beta_i)-X^T\beta_i\|_2)\\
&+\sum_{i=1}^{M}\lambda_i'(\beta-\|X^TD(S_i)(-v+\alpha_i'+\beta_i')-X^T\beta_i'\|_2).
\end{aligned}$$

Note that

$$-\|X^TD(S_i)(v+\alpha_i+\beta_i)-X^T\beta_i\|_2=\min_{\substack{r_i\in\mathbb{R}^d,\|r_i\|_2\leq 1\\ \forall i\in[M]}} -r_i^TX^TD(S_i)(v+\alpha_i+\beta_i)+r_i^TX^T\beta_i.$$

Hence, the dual is

$$\begin{aligned}
&\min_{\substack{\lambda_i,\lambda_i'\in\mathbb{R}\\ \lambda_i,\lambda_i'\geq 0\\ \forall i\in[M]}} \max_{\substack{v\in\mathbb{R}^n\\ (\alpha_i,\beta_i)\in\mathbb{R}^n\\ \alpha_i\geq 0,\beta_i\geq 0,\forall i\in[M]\\ (\alpha_i',\beta_i')\in\mathbb{R}^n\\ \alpha_i'\geq 0,\beta_i'\geq 0,\forall i\in[M]}} \min_{\substack{r_i\in\mathbb{R}^d,\|r_i\|_2\leq 1\\ r_i'\in\mathbb{R}^d,\|r_i'\|_2\leq 1\\ \forall i\in[M]}} -\frac{1}{2}\|v-y\|_2^2+\frac{1}{2}\|y\|_2^2\\
&+\sum_{i=1}^{M}\lambda_i(\beta-r_i^TX^TD(S_i)(v+\alpha_i+\beta_i)+r_i^TX^T\beta_i)\\
&+\sum_{i=1}^{M}\lambda_i'(\beta-r_i'^TX^TD(S_i)(-v+\alpha_i'+\beta_i')+r_i'^TX^T\beta_i').
\end{aligned}$$

Sion's minimax theorem (Theorem 3.4 in [Sio58]) for the inner max min problem implies the following lemma.

Lemma 2.3. *Let X and Y be convex, compact spaces. Let $f(x,y)$ be a function on $X\times Y$ satisfying:*

For each fixed $y\in Y$, $f(x,y)$ is continuous in x and concave in x.
For each fixed $x\in X$, $f(x,y)$ is continuous in y and convex in y.

Then, $\sup_{x\in X}\inf_{y\in Y}f(x,y)=\inf_{y\in Y}\sup_{x\in X}f(x,y)$.

Proof. This is a corollary of Sion's minimax theorem (Theorem 3.4 in [Sio58]). □

By using Lemma 2.3 to swap maximisation and minimisation, we can write the dual as

$$\min_{\substack{\lambda_i,\lambda_i'\in\mathbb{R}\\ \lambda_i,\lambda_i'\geq 0\\ \forall i\in[M]}} \min_{\substack{r_i\in\mathbb{R}^d,\|r_i\|_2\leq 1\\ r_i'\in\mathbb{R}^d,\|r_i'\|_2\leq 1\\ \forall i\in[M]}} \max_{\substack{v\in\mathbb{R}^n\\ (\alpha_i,\beta_i)\in\mathbb{R}^n\\ \alpha_i\geq 0,\beta_i\geq 0,\forall i\in[M]\\ (\alpha_i',\beta_i')\in\mathbb{R}^n\\ \alpha_i'\geq 0,\beta_i'\geq 0,\forall i\in[M]}} -\frac{1}{2}\|v-y\|_2^2+\frac{1}{2}\|y\|_2^2$$
$$+\sum_{i=1}^{M}\lambda_i(\beta-r_i^TX^TD(S_i)(v+\alpha_i+\beta_i)+r_i^TX^T\beta_i)$$
$$+\sum_{i=1}^{M}\lambda_i'(\beta-r_i'^TX^TD(S_i)(-v+\alpha_i'+\beta_i')+r_i'^TX^T\beta_i').$$

We maximise with respect to v, $\{\alpha_i\}_{i=1}^M$, $\{\beta_i\}_{i=1}^M$, $\{\alpha_i'\}_{i=1}^M$, $\{\beta_i'\}_{i=1}^M$ first.

$$\begin{aligned}
L(\{\lambda_i\},\{\lambda_i'\},\{r_i\},\{r_i'\},v,\{\alpha_i\},\{\beta_i\},\{\alpha_i'\},\{\beta_i'\}) \coloneqq & -\frac{1}{2}\|v-y\|_2^2+\frac{1}{2}\|y\|_2^2\\
&+\sum_{i=1}^{M}\lambda_i(\beta-r_i^TX^TD(S_i)(v+\alpha_i+\beta_i)+r_i^TX^T\beta_i)\\
&+\sum_{i=1}^{M}\lambda_i'(\beta-r_i'^TX^TD(S_i)(-v+\alpha_i'+\beta_i')+r_i'^TX^T\beta_i')\\
\nabla_v L &= -(v-y)+\sum_{i=1}^{M}D(S_i)X(-\lambda_i r_i+\lambda_i' r_i')=0 \text{ at } \hat{v}\\
\Rightarrow \hat{v} &= y+\sum_{i=1}^{M}D(S_i)X(-\lambda_i r_i+\lambda_i' r_i')\\
\nabla_{\alpha_i} L &= -D(S_i)Xr_i\\
\nabla_{\beta_i} L &= (I-D(S_i))Xr_i
\end{aligned}$$

For L to have finite maximum on $(\alpha_i,\ \beta_i)\in\mathbb{R}^n$, $\alpha_i\geq 0, \beta_i\geq 0$,

$$\begin{cases} D(S_i)Xr_i\geq 0\\ (D(S_i)-I)Xr_i\geq 0\end{cases} \Leftrightarrow r_i\in P_{S_i}.$$

Under this condition, the optimal $(\alpha_i,\ \beta_i)$ is (0,0). A similar argument applies for $(\alpha_i',\ \beta_i')$ and the optimal $(\alpha_i',\ \beta_i')$ is (0,0).

Now we can write the dual as

$$\min_{\substack{\lambda_i,\lambda_i'\in\mathbb{R}\\ \lambda_i,\lambda_i'\geq 0\\ \forall i\in[M]}} \min_{\substack{r_i,r_i'\in\mathbb{R}^d\\ \|r_i\|_2\leq 1\\ |r_i'\|_2\leq 1\\ r_i\in P_{S_i}\\ r_i'\in P_{S_i}\\ \forall i\in[M]}} L(\lambda,\lambda',\{r_i\},\{r_i'\},\hat{v},\{0\},\{0\},\{0\},\{0\})$$

$$= \min_{\substack{\lambda_i,\lambda_i'\in\mathbb{R}\\ \lambda_i,\lambda_i'\geq 0\\ \forall i\in[M]}} \min_{\substack{r_i,r_i'\in\mathbb{R}^d\\ \|r_i\|_2\leq 1\\ |r_i'\|_2\leq 1\\ r_i\in P_{S_i}\\ r_i'\in P_{S_i}\\ \forall i\in[M]}} \frac{1}{2}\|\sum_{i=1}^{M}(\lambda_i D(S_i)Xr_i - \lambda_i' D(S_i)Xr_i') - y\|_2^2 + \beta\sum_{i=1}^{M}(\lambda_i+\lambda_i')$$

We can apply a change of variables $v_i = \lambda_i r_i$ and $t_i = \lambda_i' r_i'$ for all $i \in [M]$. Then, without changing the optimal value, we obtain the dual as

$$\min_{\substack{v_i\in P_{S_i}\\ \|v_i\|_2\leq\lambda_i\\ t_i\in P_{S_i}\\ \|t_i\|_2\leq\lambda_i'\\ \lambda_i,\lambda_i'\geq 0\\ \forall i\in[M]}} \frac{1}{2}\|\sum_{i=1}^{M} D(S_i)X(v_i-t_i) - y\|_2^2 + \beta\sum_{i=1}^{M}(\lambda_i+\lambda_i').$$

Note that $\lambda_i = \|v_i\|_2$ and $\lambda_i' = \|t_i\|_2$ are optimal and feasible. So, we can write the dual as

$$\min_{\substack{v_i\in P_{S_i}\\ t_i\in P_{S_i}\\ \forall i\in[M]}} \frac{1}{2}\|\sum_{i=1}^{M} D(S_i)X(v_i-t_i) - y\|_2^2 + \beta\sum_{i=1}^{M}(\|v_i\|_2+\|t_i\|_2)$$

This optimisation problem is in the same form as (2.4). Hence, the strong dual of the convex problem (2.8) is the convex problem (2.4) and the optimal value of (2.4) is d^*.

We then show that the optimal values for $\{u_j,\omega_j\}_{j=1}^m$ are constructed when $m = m^* := \sum_{i:v_i^*\neq 0}^{P} 1 + \sum_{i:t_i^*\neq 0}^{P} 1$, and $p^* = d^*$ holds when $m \geq m^*$ where $\{v_i^*, t_i^*\}_{i=1}^P$ are optimal solution to (2.4).

Define sets of indices S_1 and S_2 as

$$S_1 = \{i\in[P]:\ v_i^*\neq 0\},$$
$$S_2 = \{i\in[P]:\ t_i^*\neq 0\}.$$

Also,

$$\forall i\in S_1,\ \text{define } (u_{1i}^*,\omega_{1i}^*) = (\frac{v_i^*}{\sqrt{\|v_i^*\|_2}}, \sqrt{\|v_i^*\|_2}),$$
$$\forall i\in S_2,\ \text{define } (u_{2i}^*,\omega_{2i}^*) = (\frac{t_i^*}{\sqrt{\|t_i^*\|_2}}, -\sqrt{\|t_i^*\|_2}).$$

Note that $m^* = |S_1| + |S_2|$. The optimal values are constructed as

$$\{u_j^*, \omega_j^*\}_{j=1}^{m^*} = \{u_{1i}^*, \omega_{1i}^*\}_{i\in S_1} \cup \{u_{2i}^*, \omega_{2i}^*\}_{i\in S_2}.$$

We show its optimality by substituting this formulation back in the objective value (2.2). By the minimality of p^*,

$$\begin{aligned}
p^* \leq & \frac{1}{2}\| \sum_{i\in S_1} (Xu_{1i}^*)_+ \omega_{1i}^* + \sum_{i\in S_2} (Xu_{2i}^*)_+ \omega_{2i}^* - y\|_2^2 + \frac{\beta}{2}\left(\sum_{i\in S_1} (\|u_{1i}^*\|_2^2 + {\omega_{1i}^*}^2) + \sum_{i\in S_2} (\|u_{2i}^*\|_2^2 + {\omega_{2i}^*}^2) \right) \\
= & \frac{1}{2}\| \sum_{i\in S_1} (Xv_i^*)_+ - \sum_{i\in S_2} (Xt_i^*)_+ - y\|_2^2 + \beta \left(\sum_{i\in S_1} \|v_i^*\|_2 + \sum_{i\in S_2} \|t_i^*\|_2 \right) \\
= & \frac{1}{2}\| \sum_{i=1}^{P} D(S_i) X(v_i^* - t_i^*) - y\|_2^2 + \beta \sum_{i=1}^{P} (\|v_i^*\|_2 + \|t_i^*\|_2) \\
= & (\text{optimal value for (2.4)}) \\
= & d^*.
\end{aligned}$$

The third last equality holds because $v_i^* \in P_{S_i} \Rightarrow (Xv_i^*)_+ = D(S_i)Xv_i^*$ and similarly for t_i^*.
By Lemma 2.2, we know that $d^* \leq p^*$. Hence, $p^* = d^*$ holds. □

2.3 Numerical experiments

In this section, we present some results from numerical experiments. These experiments aim to compare the performance of the training loss minimisation by the convex formulation and empirical risk minimisation methods in fixed and random designs. In Section 2.3.1, we compare the objective value minimised by the convex formulation from Theorem 2.1 and the objective value minimised by stochastic gradient descent (SGD) or gradient descent (GD) with randomised initial starting points. We train neural networks with different numbers of hidden neurons in a fixed design. In Section 2.3.2, we make the comparison in a random design.

The training loss we want to minimise is the sum of the mean squared error (MSE) and the regularisation term $\frac{\beta}{2}\sum_{j=1}^{m}(\|u_j\|_2^2 + \omega_j^2)$. We consider two-dimensional input data with one-dimensional output. When n data samples are collected, they construct an input training data matrix $X \in \mathbb{R}^{n\times 2}$ and the output training data $Y \in \mathbb{R}^n$. The data samples are independent and identically distributed following a simple distribution with noise: $y_j = 2 \times \mathbf{1}(x_{j,1} > x_{j,2}) - 1 + \epsilon_j$ where $\epsilon_j \overset{\text{iid}}{\sim} \mathcal{N}(0, v^2)$ for known v. Note that x_j is the jth row of X and y_j is the jth element of Y.

We train two-layer ReLU networks. The batch size of SGD is taken to be 1. The regularisation parameter β is taken to be 10^{-3}. The learning rate is taken to be 0.01. For each independent experiment, the initial starting points for the parameters are randomly chosen. The Python library `cvxpy` is used for convex optimisation, and `Pytorch` is used to construct neural networks and implement SGD or GD. The codes used for the numerical experiments are listed on a GitHub page [25].

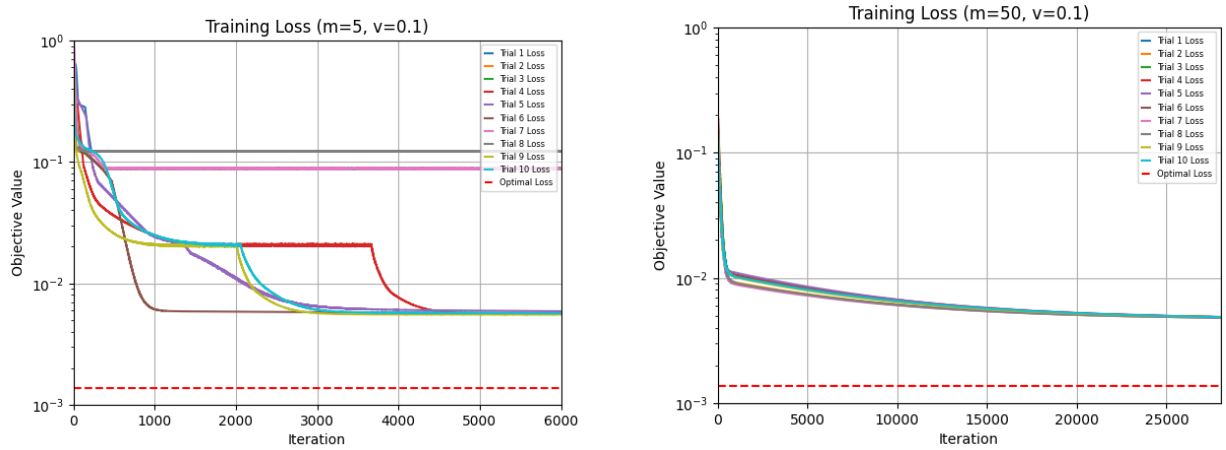
2.3.1 Fixed design setting

Simple training data, SGD

We first consider an input data $X = [(1,-2),(1,-1),(1,0),(1,1),(1,2)]^T$ and an output data $Y = [1.0337, 1.0129, 1.0234, -0.9770, -1.1123]^T$, which is generated with small noise $v = 0.1$. The global minimum proposed by the convex formulation is 0.00136 ("Optimal Loss"). We implement SGD to fit the data with two-layer neural networks with different numbers of hidden neurons; $m = 5$ and $m = 50$. After running the program over 28000 iterations, I observed that the minimised objective values do not change much after 5000 iterations for $m = 5$, and after 24000 iterations for $m = 50$.

The learning process is shown in Figure 1. After each iteration, we plot the objective values minimised by SGD until convergence (6000 iterations for $m = 5$ and 28000 iterations for $m = 50$). Ten independent trials starting from different random starting points were plotted, with the "Optimal Loss" (red dashed line) found by the convex formulation of Theorem 2.1. As demonstrated in Figure 1, the neural networks with a small number of hidden neurons will likely converge to a local minimum of a larger value. In this experiment, $m^* \leq n = 5$, so theoretically, a strong duality holds for both $m = 5$ and $m = 50$. However, the 10 trials after the 28000 iterations did not converge to the global minimum proposed by the convex formulation in both cases. (The difference is over 10^{-3}. See Table 1.)

Networks with 5 hidden neurons showed sudden decreases in their objective values at some points in their learning process. In contrast, networks with 50 hidden neurons showed a rapid decrease until around 500 iterations and a gradual decrease between 500 and 24000 iterations. I expect that this is because networks with fewer neurons have fewer parameters to update, so they adapt to changes more quickly.



((a)) $m = 5$, 6000 iterations ((b)) $m = 50$, 28000 iterations

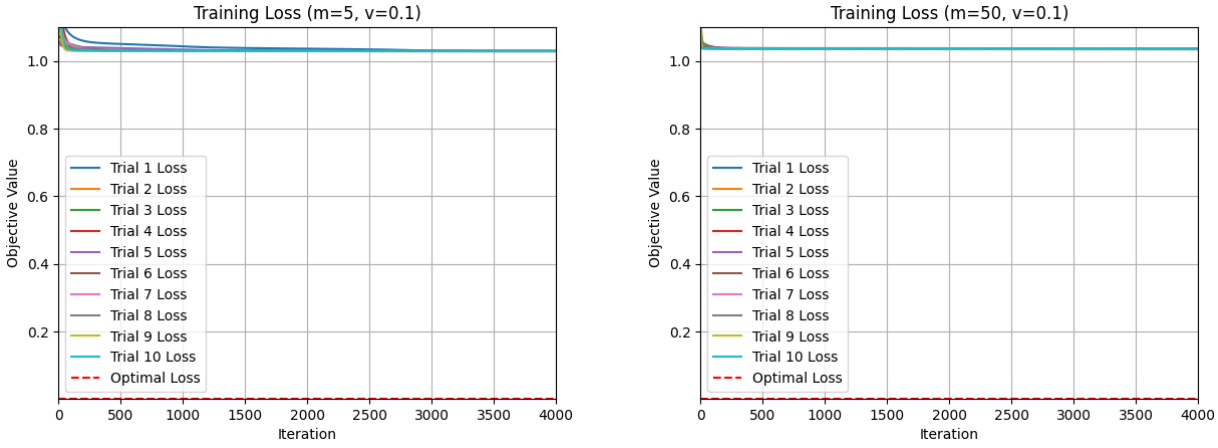
Figure 1: Objective value (MSE+L_2 regularisation) of two-layer ReLU networks trained with SGD on a simple dataset with a small noise ($v = 0.1$). Optimal Loss is proposed by the convex formulation (Theorem 2.1).

Simple training data, GD

We then compare the training process of SGD and GD trained on the same data. We take the same training data $X = [(1,-2),(1,-1),(1,0),(1,1),(1,2)]^T$ and $Y = [1.0337, 1.0129, 1.0234, -0.9770, -1.1123]^T$. Figure 2 shows the training process of networks with the number of hidden neurons $m = 5$ and $m = 50$

trained with GD.

Comparing Figure 1 and Figure 2, it is clear that SGD-trained networks performed much better in minimising the objective value than GD-trained networks after the same number of iterations. This is because SGD does not calculate a precise gradient, which enables the network to escape from local minima and explore the loss landscape to find better solutions. On the other hand, GD makes more precise updates, so the derivative is always close to zero when it is around local minima. Indeed, as shown in Figure 2, the objective values for GD converge to the local minima more quickly than for SGD and stay there. We can see that there is a significant gap of more than 1 between the optimal value proposed by the convex formulation and the objective values minimised by GD even after 4000 iterations. The objective value does not change much after 2500 iterations when $m = 5$ and after 500 iterations when $m = 50$. These observations imply that, unless we know the value of the global minimum beforehand, there is a risk of stopping the training process with GD too early, since it will pass a convergence test.



((a)) $m = 5$, 4000 iterations

((b)) $m = 50$, 4000 iterations

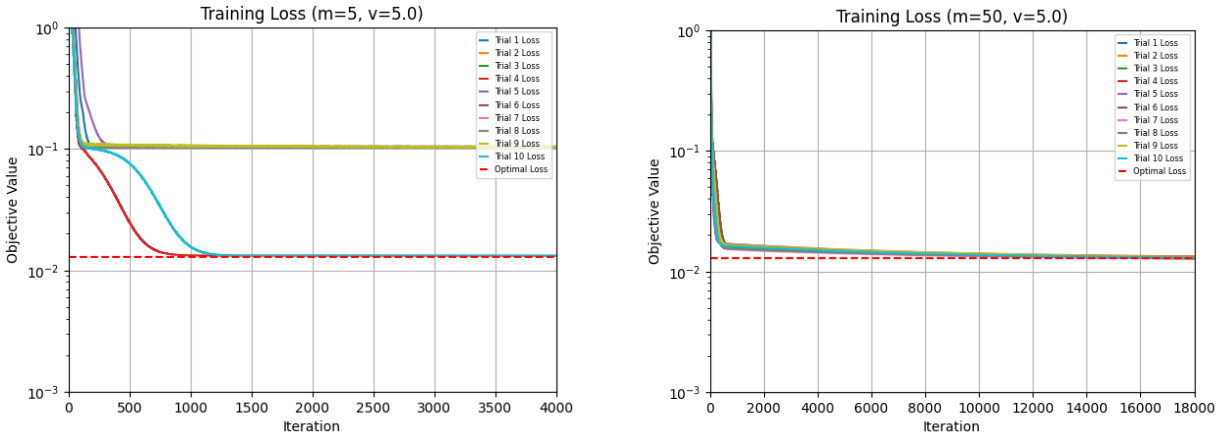
Figure 2: Objective value (MSE+L_2 regularisation) of two-layer ReLU networks trained with GD on a simple dataset with a small noise ($v = 0.1$). Optimal Loss is proposed by the convex formulation (Theorem 2.1).

Noisy training data, SGD

We also consider training processes on noisy data. We take the same input training matrix $X = [(1, -2), (1, -1), (1, 0), (1, 1), (1, 2)]^T$, but change the standard deviation of the noise to be $v = 5.0$. The output training data is now $Y = [2.6835, 1.6440, 2.1723, 0.1517, -6.6143]^T$. Figure 3 shows the SGD-trained network training process for the number of hidden neurons $m = 5$ and $m = 50$. The global minimum proposed by the convex formulation is 0.0127.

We compare the training process on these noisy data and the training process on the same input training data X with the small noise of $v = 0.1$ shown in Figure 1. The behaviour for $m = 50$ on noisy data ($v = 5.0$) is similar to that of less noisy data ($v = 0.1$), showing a rapid decrease until around 500 iterations and a gradual decrease after that. However, Table 1 and Figure 3 show that, unlike the case with $v = 0.1$, all independent training processes avoid local minima, reaching the global minimum suggested by the convex formulation. When $m = 5$, the training process on the noisy data showed fewer frequent updates than on the less noisy data. However, there are 2 realisations showing their objective

values reached very close (within 5×10^{-4} difference) to the global minimum suggested by the convex formulation. The other 8 realisations stuck at local minima around 0.1. A key observation common to both $m = 5$ and $m = 50$ is that the minimum objective value among 10 SGD realisations is closer to its global minimum when the output is noisy ($v = 5$) compared to when the output depends more on the location of the input data ($v = 0.1$). (See Table 1.) I expect that the loss function is smoother when the correlation between the input and the output is weaker.



((a)) $m = 5$, 4000 iterations

((b)) $m = 50$, 18000 iterations

Figure 3: Objective value (MSE+L_2 regularisation) of two-layer ReLU networks trained with GD on a simple dataset with a large noise ($v = 5.0$). Optimal Loss is proposed by the convex formulation (Theorem 2.1).

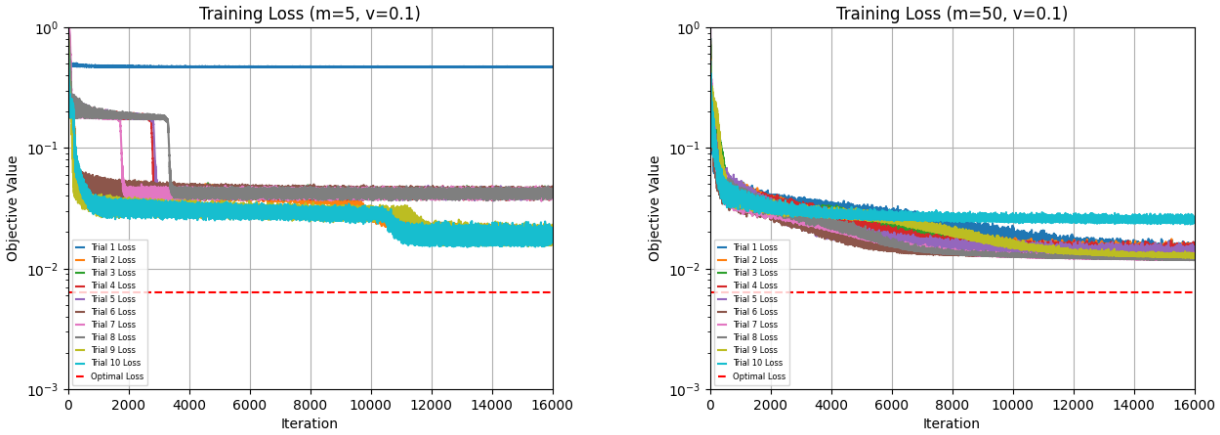
Complex data, SGD

We further consider complex training data $X = [(1.5, 1), (2.5, 2), (2, 2), (1, 1.5), (2, 2.5)]^T$ and $Y = [1.0337, 1.0129, -0.9766, -0.9770, -1.1123]$, which is generated with small noise $v = 0.1$. The data is complex in the sense that positive samples ($y \approx 1$) are placed closer to negative samples ($y \approx -1$) than in the previous training dataset. This training data will create a more complex loss landscape. We train networks with the number of hidden neurons $m = 5$ and $m = 50$ with SGD.

Figure 4 shows that the training process of SGD needs many more iterations to escape the local minima ($m = 5$) or needs many more iterations to show a gradual decrease ($m = 50$) compared to simple training data shown in Figure 1. When I ran the programme over 28000 iterations for the simple training data, the objective values did not fluctuate after 5000 iterations when $m = 5$, and showed only a gradual decrease after 1000 iterations when $m = 50$. On the other hand, for the complex training data, the objective values fluctuated throughout the whole 16000 iterations. The global minimum suggested by the convex formulation is 0.00635. Similarly to the case of simple training data, the objective value minimised by SGD did not reach the global minimum for both $m = 5$ and $m = 50$. (The difference is over 5×10^{-3}. See Table 1.)

2.3.2 Random design setting

In this section, we assess the optimality of the convex formulation in a random setting. We perform 100 independent minimisations in total (50 independent minimisations by using the convex formulation and 50 independent trials for SGD). For each of these minimisations, we collect n samples uniformly



((a)) $m = 5$, 16000 iterations

((b)) $m = 50$, 16000 iterations

Figure 4: Objective value (MSE+L_2 regularisation) of two-layer ReLU networks trained with SGD on a complex dataset with small noise ($v = 0.1$). Optimal Loss is proposed by the convex formulation (Theorem 2.1).

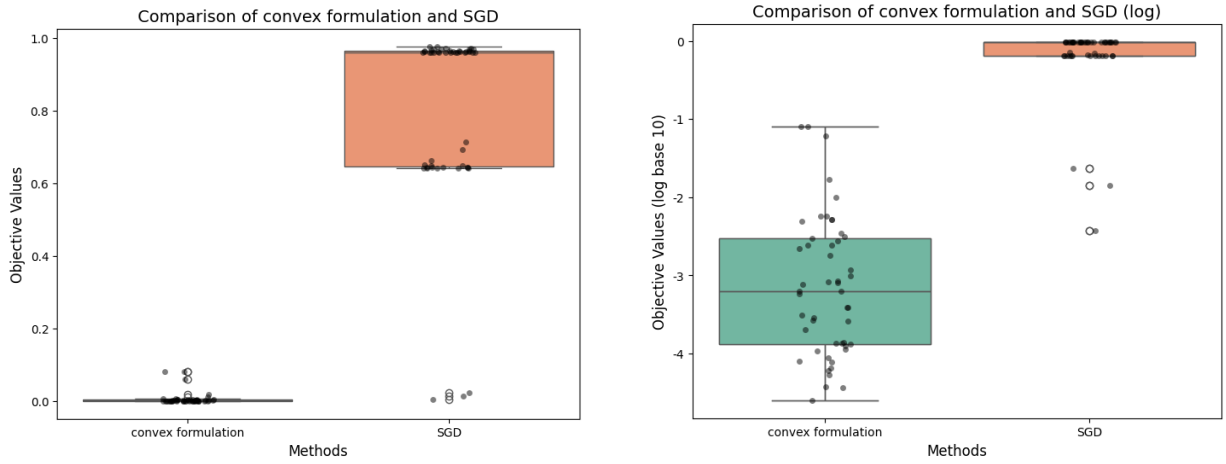
v	m	iteration	training data	SGD	convex formulation	difference
0.1	5	6000	simple	0.0055	0.00136	4.14×10^{-3}
0.1	50	28000	simple	0.0048	0.00136	3.44×10^{-3}
5.0	5	4000	simple	0.0132	0.0127	5.00×10^{-4}
5.0	50	4000	simple	0.0127	0.0127	0
0.1	5	16000	complex	0.0177	0.00635	1.135×10^{-2}
0.1	50	16000	complex	0.0119	0.00635	5.55×10^{-3}

Table 1: Objective values minimised by SGD (minimum value among 10 realisations), objective values minimised by the convex formulation, and their differences.

at random and independently from $[-5, 5) \times [-5, 5)$ to construct a random $n \times 2$ input training matrix. As a result, for the total of 100 minimisations, their input matrices are randomly constructed independently of each other. The output sample y_j corresponding to x_j is generated from the simple distribution without noise: $y_j = 2 \times \mathbf{1}(x_{j,1} > x_{j,2}) - 1$. If the optimality of the convex formulation does not hold in a random design, the gap between the minimised values by the convex formulation and the minimised values by SGD is not larger than what can be explained by the randomness in the training data.

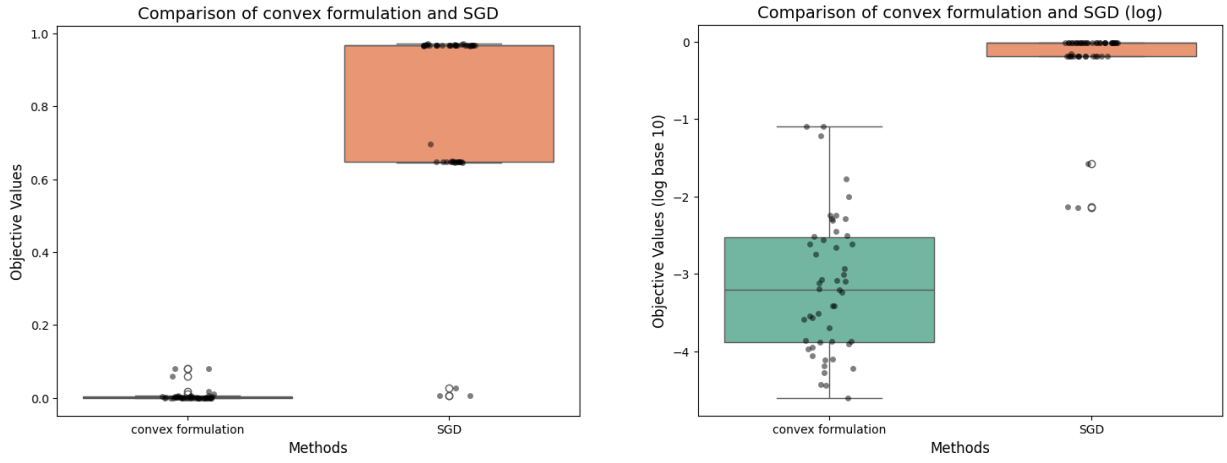
We first consider the case where the number of random samples is $n = 5$ and the number of hidden neurons is $m = 5$. I observed that the minimised values by SGD do not change much after 3000 iterations. So, the minimised value by SGD is set to be the minimised value after 6000 iterations. For each independent trial for SGD, the initial starting point for the parameters is randomly chosen. The minimised values by the convex formulation and by SGD are shown in Figure 5. Although the training data matrices differ, most of the objective values minimised by SGD converged around 0.7 or 1.0. The differences between minimised values by the convex formulation and SGD are larger than the variability that can be explained by the randomness in the training data. This observation suggests that, with high probability, the training loss of SGD starting from a random initialisation point trained on a random input X is bounded below by the training loss minimised by the convex formulation trained

on another random input X'. (It is likely that $X \neq X'$.) Figure 6 suggests that this observation also holds when we increase the number of hidden neurons for networks to be $m = 50$. Figure 7 confirms the optimality of the convex formulation in random design for a larger number of samples $n = 30$ as well.



((a)) (n, m)=(5, 5), scale 1 ((b)) (n, m)=(5, 5), logarithmic scale (base 10)

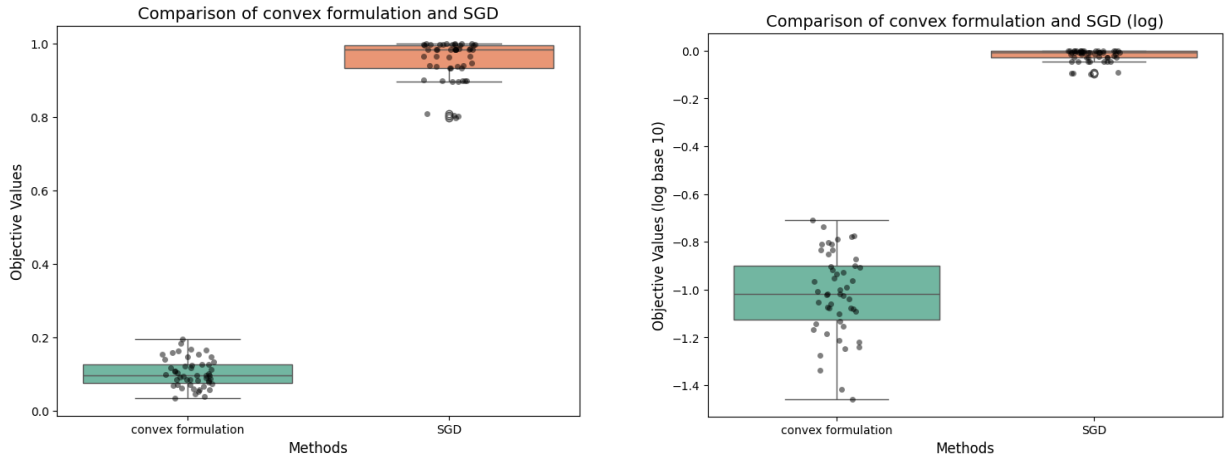
Figure 5: Objective values (MSE+L_2 regularisation) minimised by convex formulation (Theorem 2.1) for 50 independent trials with 5 random samples (left, green). Objective values of two-layer ReLU networks with 5 hidden neurons trained with SGD after training over 6000 iterations for 50 independent trials with 5 random samples (right, orange).



((a)) (n, m)=(5, 50), scale 1 ((b)) (n, m)=(5, 50), logarithmic scale (base 10)

Figure 6: Objective values (MSE+L_2 regularisation) minimised by convex formulation (Theorem 2.1) for 50 independent trials with 5 random samples (left, green). Objective values of two-layer ReLU networks with 50 hidden neurons trained with SGD after training over 6000 iterations for 50 independent trials with 5 random samples (right, orange).

Remark 2.3. In Section 2.3, we focused on minimising training loss. One may argue that, in general, we are often not interested in achieving global minima of training loss. For example, to improve



((a)) (n, m)=(30, 5), scale 1

((b)) (n, m)=(30, 5), logarithmic scale (base 10)

Figure 7: Objective values (MSE+L_2 regularisation) minimised by convex formulation (Theorem 2.1) for 50 independent trials with 30 random samples (left, green), Objective values of two-layer ReLU networks with 5 hidden neurons trained with SGD after training over 6000 iterations for 50 independent trials with 30 random samples (right, orange).

prediction accuracy for squared loss, it can be shown that the training error, Err_{tr}, underestimates the in-sample error, Err_{in}, as $Err_{in} = \mathbb{E}[Err_{tr}] + \frac{2}{n}\sum_{i=1}^{n} \mathrm{Cov}(f_{\hat{\theta}}(x_i), Y_i)$. This result implies that the better the model fits the training data, the more the training error underestimates the in-sample error. However, we included a regularisation term in the loss function in our numerical experiments. Pilanci and Ergen [PE20] reported that the convex formulation also surpasses in test accuracy in a similar setting.

For the rest of Section 2, we will look at a special case, where a closed form for optimal weights can be found, and two generalisations of the convex formulation. The two generalisations help our understanding of the inner workings of neural networks in Section 3.2

2.4 Special case: whitened data

Pilanci and Ergen [EP21a] showed that, when a data matrix is a special form, *whitened data matrix*, a closed-form solution to the optimisation problem (2.3) can be found. We use convex optimisation theories to find this closed form. This result is useful because we can find the optimal weights without training.

Definition 2.1 (Whitening Transformation)**.** *A whitening transformation is a linear transformation that transforms a vector of random variables with a known covariance matrix into a set of new variables whose covariance is the identity matrix.*

Remark 2.4. The whitening transformation for a matrix is not unique. More details about optimal whitening transformations can be found in [KLS18].

After a whitening transformation, the samples in the input data are uncorrelated, and each has a variance of one. Informally speaking, the transformation changes the input matrix into white noise. Whitening transformation is a powerful technique that enhances data processing and analysis. It is used in various fields such as automatic speaker recognition for children's speech [RBS23] or noise reduction in ultrasonic testing [BK19].

Theorem 2.2. *Suppose that the data matrix $X \in \mathbb{R}^{n\times d}$ is whitened such that $XX^T = I_n$. We write the neural network we want to train as $f(X) = (XU)_+\omega$. Then, an optimal weights (U^*, ω^*) for (2.3) can be formulated as follows*

$$(U^*, \omega^*) = \begin{cases} \left(\left[\frac{X^T(y)_+}{\|X^T(y)_+\|_2}, \frac{X^T(-y)_+}{\|X^T(-y)_+\|_2} \right], \begin{bmatrix} \|X^T(y)_+\|_2 - \beta \\ -\|X^T(-y)_+\|_2 + \beta \end{bmatrix} \right) & if\ \beta \le \|(y)_+\|_2,\ \beta \le \|(-y)_+\|_2 \\ \left(\frac{X^T(-y)_+}{\|X^T(-y)_+\|_2}, -\|X^T(-y)_+\|_2 + \beta \right) & if\ \beta > \|(y)_+\|_2,\ \beta \le \|(-y)_+\|_2 \\ \left(\frac{X^T(y)_+}{\|X^T(y)_+\|_2}, -\|X^T(y)_+\|_2 - \beta \right) & if\ \beta \le \|(y)_+\|_2,\ \beta > \|(-y)_+\|_2 \\ (\boldsymbol{0}, 0) & if\ \beta > \|(y)_+\|_2,\ \beta > \|(-y)_+\|_2 \end{cases} \tag{2.15}$$

where $U^* = (u_1^*,\ \dots, u_m^*)$ and $\omega^* = (\omega_1^*,\ \dots, \omega_m^*)$.

Remark 2.5. The regularisation parameter β controls the number of neurons and the analytical solution to the optimisation problem. If β is large enough, that is, if $\beta > \|(y)_+\|_2$ and $\beta > \|(-y)_+\|_2$, the optimal weights are zero. This result agrees with our intuition that the optimal solution shrinks towards zero when the regularisation penalty is large.

First, we prove the following Lemma 2.4 about a property of a whitened matrix. The result of Lemma 2.4 enables us to rewrite the constraint in the dual problem (2.8) as a simple constraint.

Lemma 2.4. *Suppose data matrix $X \in \mathbb{R}^{n\times d}$ is whitened such that $XX^T = I_n$. Then,*

$$\forall v \in \mathbb{R}^n, \quad \max_{u\in\mathcal{B}_2} |v^T(Xu)_+| = \max\{\|(v)_+\|_2, \|(-v)_+\|_2\}.$$

Proof. We begin with properties of a whitened data matrix.
Let $X = UDV^T$ denote the singular value decomposition (SVD) of X. Here, $U \in \mathbb{R}^{n\times n}$ and $V \in \mathbb{R}^{d\times d}$ are orthogonal matrices, and $D \in \mathbb{R}^{n\times d}$ has diagonal elements $D_{11} \ge D_{22} \ge\ \dots\ \ge D_{mm} \ge 0$ where $m \coloneqq \min\{n, d\}$ and other entries of D are zero.
By basic results from linear algebra,

$$n = rank(XX^T) \le rank(X) \le m \le d.$$

Also,

$$\begin{aligned} I_n = XX^T = UDD^TU^T \\ \Rightarrow DD^T = U^TU = I_n. \end{aligned}$$

By the construction of D, $\forall i \in [n],\ D_{ii} \in \{-1, 1\}$.
Hence, $D^TD = \begin{pmatrix} I_n, & \mathbf{0} \\ \mathbf{0}, & \mathbf{0} \end{pmatrix}$.
This implies that $X^TX = V \begin{pmatrix} I_n, & \mathbf{0} \\ \mathbf{0}, & \mathbf{0} \end{pmatrix} V^T$.
For $u \in \mathbf{B}_2$, denote Xu by u'. Then,

$$\|u'\|_2^2 = u'^Tu' = u^TX^TXu = u^TV \begin{pmatrix} I_n, & \mathbf{0} \\ \mathbf{0}, & \mathbf{0} \end{pmatrix} V^Tu \le 1$$

Also, $\forall u' \in \{u' \in \mathbb{R}^n\ :\ \|u'\|_2 \le 1\},\ \exists u \in \mathbf{B}_2\ \ s.t.\ \ Xu = u'$.
Hence, $\max_{u\in\mathbf{B}_2} |v^T(Xu)_+| = \max_{u'\in\mathbb{R}^n:\|u'\|_2\le 1} |v^T(u')_+|$.

$$v^T(u')_+ \le (v)_+^T(u')_+ \le \|(v)_+\|_2\|(u')_+\|_2 \le \|(v)_+\|_2. \text{ (Equality holds when } u' = \frac{(v)_+}{\|(v)_+\|_2}.)$$

$-v^T(u')_+ \leq (-v)_+^T(u')_+ \leq \|(-v)_+\|_2\|(u')_+\|_2 \leq \|(-v)_+\|_2$. (Equality holds when $u' = \dfrac{(-v)_+}{\|(-v)_+\|_2}$.)

Thus, $\max_{u\in\mathcal{B}_2} |v^T(Xu)_+| = \max\{\|(v)_+\|_2, \|(-v)_+\|_2\}$ holds. □

Now, we prove Theorem 2.2 by applying Lemma 2.4 to Lemma 2.2.

Proof of Theorem 2.2

Proof. By applying Lemma 2.4 to Lemma 2.2, the dual problem can be written as

$$\max_{v\in\mathbb{R}^n} -\frac{1}{2}\|v-y\|_2^2 + \frac{1}{2}\|y\|_2^2 \quad \text{s.t.} \quad \max\{\|(v)_+\|_2, \|(-v)_+\|_2\} \leq \beta.$$

$-\frac{1}{2}\|v-y\|_2^2 + \frac{1}{2}\|y\|_2^2$ is maximised when $\|v-y\|_2^2$ is minimised. If $\exists\ i\ \ s.t.\ \ sgn(y_i) \neq sgn(v_i)$, changing v_i to 0 decreases $\|v-y\|_2^2$ while not increasing $\max\{\|(v)_+\|_2, \|(-v)_+\|_2\}$. Hence, we may assume

$$\|v-y\|_2^2 = \|(v)_+ - (y)_+\|_2^2 + \|(-v)_+ - (-y)_+\|_2^2$$

for optimal v.
By the constraint $\max\{\|(v)_+\|_2, \|(-v)_+\|_2\} \leq \beta$, optimal v^* is

$$(v^*)_+ = \begin{cases} (y)_+ & if\ \beta > \|(y)_+\|_2 \\ \beta\frac{(y)_+}{\|(y)_+\|_2} & if\ \beta \leq \|(y)_+\|_2 \end{cases}, \quad (-v^*)_+ = \begin{cases} (-y)_+ & if\ \beta > \|(-y)_+\|_2 \\ \beta\frac{(-y)_+}{\|(-y)_+\|_2} & if\ \beta \leq \|(-y)_+\|_2. \end{cases}$$

Hence,

$$v^* = \begin{cases} \beta\frac{(y)_+}{\|(y)_+\|_2} - \beta\frac{(-y)_+}{\|(-y)_+\|_2} & if\ \beta \leq \|(y)_+\|_2, \beta \leq \|(-y)_+\|_2 \\ \beta\frac{(y)_+}{\|(y)_+\|_2} - (-y)_+ & if\ \beta \leq \|(y)_+\|_2, \beta > \|(-y)_+\|_2 \\ (y)_+ - \beta\frac{(-y)_+}{\|(-y)_+\|_2} & if\ \beta > \|(y)_+\|_2, \beta \leq \|(-y)_+\|_2 \\ y & if\ \beta > \|(y)_+\|_2, \beta > \|(-y)_+\|_2 \end{cases}.$$

Recall conditions (2.11) and (2.13) for optimal solutions U^* and ω^*, and that r is defined in (2.10). They give the relationship between v^*, U^* and ω^* as

$$\begin{cases} \sum_{j=1}^m (Xu_j^*)_+\omega_j^* - y = -v^* \\ (\beta sgn(\omega_j^*) - v^{*T}(Xu_j^*)_+)\omega_j^* = 0 \end{cases}$$

Solving the above equations gives the desired values for U^* and ω^*. □

2.5 General cases

In practice, we often want to use loss functions other than squared loss, or regularisations other than weight decay. For example, we want to use hinge loss or cross-entropy for classification tasks, or use Lasso regularisation when X has redundant information in its columns. Similarly, we often want to use neural network architectures other than the two-layer ReLU network. For instance, we want to use a convolutional neural network when we perform image classification.

A wide range of applications of neural networks motivates us to generalise the convex formulation (Theorem 2.1). Pilanci and Ergen [EP21a] [EP21b] showed convex formulations for general cases. These generalisations are the central topics in this section.

2.5.1 General case: general loss function

We present the convex formulation for general loss functions and regularisations. The significance of the result is that it enables us to use the convex formulation for a wide range of tasks, such as classification.

We consider a general convex loss function $\mathcal{L}(.,.)$ and l_p-norms in the reguralisation term $\mathcal{R}(\theta)$ in (1.2). We take the same two-layer ReLU neural networks defined as (2.1). So, the optimisation problem we work on is

$$p^* \coloneqq \min_{\{u_j,\omega_j\}_{j=1}^m} \mathcal{L}(\sum_{j=1}^m (Xu_j)_+\omega_j,\ y) + \beta \sum_{j=1}^m (\|u_j\|_p^2 + \|\omega_j\|_p^2). \tag{2.16}$$

The convex formulation is stated as follows.

Theorem 2.3. *The non-convex problem (2.16) has a convex dual formulation*

$$p^* \geq d^* \coloneqq \max_{v\in\mathbb{R}^n} -\mathcal{L}^*(v,y) \qquad s.t. \qquad |v^T(Xu)_+| \leq \beta \qquad \forall u \in \mathcal{B}_p^d \tag{2.17}$$

where $\mathcal{L}^*$ is convex conjugate of $\mathcal{L}$ defined as

$$\mathcal{L}^*(v,y) \coloneqq \sup_{z\in\mathbb{R}^n} z^T v - \mathcal{L}(z,y) \tag{2.18}$$

When m exceeds a critical threshold m^*, $p^* = d^*$ holds. The value of the threshold m^* can be determined from an optimal solution of the dual problem in (2.16).

Proof. See [EP21a]. □

Remark 2.6. Unlike the result in Theorem 2.1, optimal weights are not explicitly constructed.

2.5.2 General case: deep neural networks with multiple sub-networks

We present the convex formulation for a more general neural network architecture. It is important because there has been a significant interest in understanding the effectiveness of overparametrised neural networks, including neural networks with multiple sub-networks. In fact, empirical studies proved that increasing the number of subnetworks leads to better training performance and remarkable generalisation performance [EP21b].

We consider a general form of neural networks consisting of K subnetworks, each of which is an L-layer ReLU network defined as in (1.1). This architecture represents many neural network architectures in practice. For instance, various networks with multiple subnetworks are used in practice, especially for image classification tasks. ResNets [He+16] is an example. We take squared loss and weight decay regularisation for our objective function. So, the optimal problem we work on is

$$p^* \coloneqq \min_{\theta\in\Theta} \mathcal{L}(\sum_{k=1}^K f_{\theta,k}(X), y) + \frac{\beta}{2} \sum_{k=1}^K \sum_{l=L-1}^L \|W_{lk}\|_F^2 \tag{2.19}$$

where $\theta \coloneqq \{\{W_{lk}\}_{l=1}^L\}_{k=1}^K$ takes a value in a parameter space Θ.

The convex formulation is stated as follows.

Theorem 2.4. *The non-convex problem (2.19) has a convex dual formulation*

$$p^* \geq d^* \coloneqq \max_{v \in \mathbb{R}^n} -\mathcal{L}^*(v, y) \qquad s.t. \qquad |v^T((XW_1)_+ ... W_{(L-1)})_+| \leq \beta \qquad \forall \theta \in \Theta_p \tag{2.20}$$

where $\Theta_p \coloneqq \{\theta \in \Theta \mid \forall k \in [K],\ \forall l \in [L-2],\ \|W_{1k}\|_F \leq 1, \|W_{(L-1),k}\|_2 \leq 1\}$ *and* $\mathcal{L}^*$ *is convex conjugate of* $\mathcal{L}$ *defined as*

$$\mathcal{L}^*(v, y) \coloneqq \sup_{z \in \mathbb{R}^n} z^T v - \mathcal{L}(z, y) \tag{2.21}$$

The strong duality $p^* = d^*$ *holds when hidden layer weights are optimised.*

Proof. See [EP21b]. □

Remark 2.7. The exact convex problem equivalent to the non-convex training problem for the case where $\mathcal{L}(v, y) = \|v - y\|_2^2$, general K and $L = 3$ is shown in [EP21b]. Similarly to the two-layer case, the convex formulation considers an equivalence formulation with group norm operator regularisation and feature selection of data samples.

3 Interpretations of the inner workings of hidden neurons

3.1 Feature selection in a high-dimensional space

The convex formulation in Theorem 2.1 can be interpreted as a high-dimensional feature selection. Specifically, multiplying the data matrix X by a single hyperplane arrangement matrix $D(S_i)$ works as a selection of data points. With all such selections, a transformation of an input matrix X into an effective high-dimensional matrix $[D(S_1),\ ...\ ,\ D(S_M)]$ is performed. The l_1-regularisation term in the convex formulation induces sparsity, which encourages a parsimonious model.

Here is an example to illustrate this idea. Our optimisation problem is for two-layer ReLU networks with squared loss and weight decay regularisation (2.3).

Example 1. We have $n = 5$ samples in $\mathbb{R}^2$ $(d = 2)$ to form the input data matrix X and the corresponding label vector y.

$$X = [(2, 1), (1, 2), (2, 4), (4, 2), (-1, 1)]^T$$
$$y = [1, -1, -1, 1, -1]^T$$

They are sampled from a simple distribution $y_j = 2 \times \mathbf{1}(x_{j,1} > x_{j,2}) - 1$.

We want to find the minimum objective value for the neural network (2.3) by formulating it as a convex optimisation problem (2.4) in Theorem 2.1. We use the same notation as in Thm 2.1. Figure 8 shows the points corresponding to the rows of $D_1X, D_2X, ..., D_6X$ as large circles, where $\{D_1, D_2, ..., D_6\} = \{\text{Diag}(\mathbf{1}[Xu \geq 0]) \mid \forall u \in \mathbb{R}^2\}$.

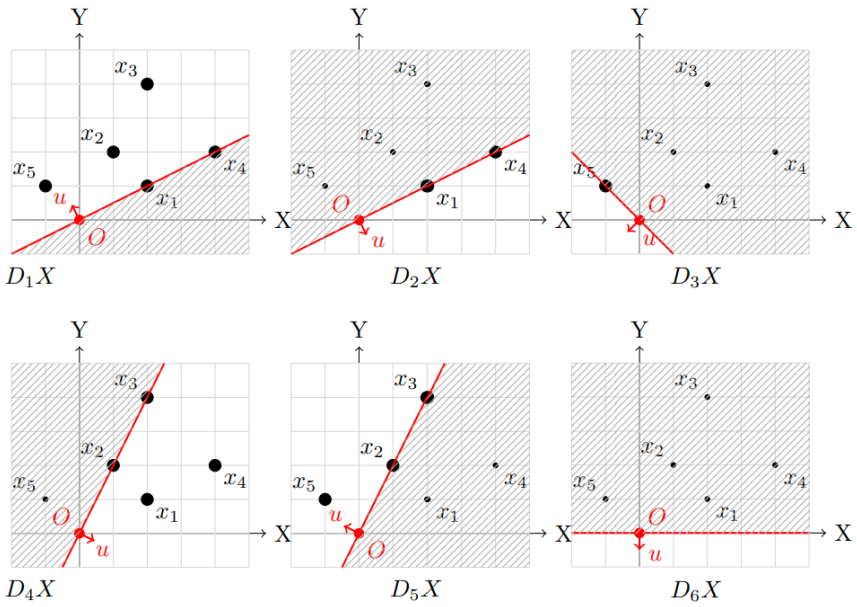


Figure 8: Data points filtered by $\{\text{Diag}(\mathbf{1}[Xu \geq 0]) \mid \forall u \in \mathbb{R}^2\}$.

The inside of the square loss for the convex formulation in (2.4) is

$$\begin{bmatrix} (2,1) \\ (1,2) \\ (2,4) \\ (4,2) \\ (-1,1) \end{bmatrix} (v_1 - t_1) + \begin{bmatrix} (2,1) \\ (0,0) \\ (0,0) \\ (4,2) \\ (0,0) \end{bmatrix} (v_2 - t_2) + \begin{bmatrix} (0,0) \\ (0,0) \\ (0,0) \\ (0,0) \\ (-1,1) \end{bmatrix} (v_3 - t_3)$$
$$+ \begin{bmatrix} (2,1) \\ (1,2) \\ (2,4) \\ (4,2) \\ (0,0) \end{bmatrix} (v_4 - t_4) + \begin{bmatrix} (0,0) \\ (1,2) \\ (2,4) \\ (0,0) \\ (-1,1) \end{bmatrix} (v_5 - t_5) - \begin{bmatrix} 1 \\ -1 \\ -1 \\ 1 \\ -1 \end{bmatrix}.$$

We can see that X is transformed into an effective high-dimensional data matrix $\tilde{X} = [D_1X, ...D_5X]$ corresponding to the set of all the hyperplane arrangements passing through the origin and data points in X. (See Figure 8.) This transformation illustrates the idea that non-linear ReLU layers can be replaced with a linear transformation that maps X to $\tilde{X}$. The regularisation term $\beta \sum_{i=1}^{6} (\|v_1\|_2 + \|t_i\|_2)$ in (2.4) reminds us of group Lasso [YL06], which encourages either an entire v_i (or t_i) to be zero or all the components of v_i (or t_i) to be non-zero. These ideas give us an interesting view of the inner workings of neural networks: the non-convex ReLU network has an implicit regularisation structure, the group Lasso regularisation, and learns an optimal feature selection of high-dimensional data $[D_1X, ...D_6X]$.

It is also clear that $1[Xu_1 \geq 0]$ and $1[Xu_2 \geq 0]$ have different sign patterns if and only if a hyperplane that passes through the origin and is orthogonal to some of the rows of X separates u_1 and u_2 to different sides. Hence, the number of sign patterns $|\{\mathbf{1}[Xu \geq 0]) \mid \forall u \in \mathbb{R}^2\}|$ is equal to the number of regions separated by hyperplanes passing through originis and orthogonal to some of the rows of X.

It can be shown [Ojh00] [Cov65] that the number of such regieans $P_{n,r}$ satisfies

$$P_{n,r} = 2\sum_{i=0}^{r-1}(\binom{n}{i})$$

where $X \in \mathbb{R}^{n\times d}$ and $r \coloneqq rank(X) \leq n$.
[EOS86] further presented an algorithm that presents all possible hyperplane arrangements in $\mathcal{O}(n^r)$ time. This ensures the polynomial-time trainability of the 2-layer ReLU network with weight decay.

Remark 3.1. Ergen and Pilanci [EP21b] further investigated models beyond the two-layer ReLU networks. They revealed that deep ReLU networks are group Lasso models with additional linear constraints. They showed that polynomial-time trainability is also possible for regularised ReLU networks with multiple non-linear layers.

3.2 Optimisation via Hodge dual

Another interesting insight into the inner workings of hidden neurons is given via Clifford Algebra [‡] [Pil23]. This is reminiscent of the hyperplane arguments discussed in Section 3.1. In the hyperplane arguments, it is enough to consider feature selections by hyperplanes separating linearly independent data points. Similarly, for the Clifford algebra arguments explained in this section, it is enough to consider the signed volumes of parallelotopes formulated by linearly independent data points. In addition, for both cases, only data points lying on one particular side of a hyperplane matter. For the Clifford algebra argument, instead of considering an optimisation problem by using the feature selections as in Section 3.1, we construct a matrix whose elements are calculated exactly from the spatial relations of data points and solve a minimisation problem using this matrix.

Clifford algebra is an algebraic system generated from an inner product vector space. It generalises the dot product and cross product to higher dimensions, and treats objects in a coordinate-free setting. This property is useful when we analyse the spatial relations between data points. A novel result shown by Pilanci [Pil23] is that optimal weights for deep neural networks can be obtained by using the Hodge dual, an important operator in Clifford algebra. In this essay, we consider two-layer networks and will not delve into details of Clifford algebra beyond basic concepts, such as the wedge product. However, it is good to keep in mind that the results we present in this section can be generalised by using the language of Clifford algebra.

We consider the optimisation problem for two-layer ReLU networks with a general loss function (2.16). The main idea is to analyse the optimal weights for (2.16) by examining the extreme points in its convex dual (2.17).

Firstly, we set $p = 1$, so our optimisation problem is

$$p^* \coloneqq \min_{\{u_j,\omega_j\}_{j=1}^m} \mathcal{L}(\sum_{j=1}^m (Xu_j)_+\omega_j,\ y) + \beta\sum_{j=1}^m(\|u_j\|_1^2 + \|\omega_j\|_1^2). \tag{3.1}$$

The following Theorem 3.1 formulates this non-convex optimisation problem as a convex problem using the spatial relations between data points.

[‡] A concise explanation of Clifford algebra can be found in [Hit12], and more details can be found in [Art16]. An interesting perspective on Clifford algebra is provided in [DDL12]

Theorem 3.1. *The non-convex problem (3.1) is equivalent to the following convex Lasso problem*

$$\min_{z\in\mathbb{R}^{P(n+d,d-1)}} \mathcal{L}(Kz,y)+\lambda\|z\|_1. \tag{3.2}$$

The matrix K is defined as $K_{ij}=k(x_i,x_{j_1},...,x_{j_{d-1}})$ for $i\in[n]$ and a multi-index $j=(j_1,...,j_{d-1})$, where

$$k(x_i,x_{j_1},...,x_{j_{d-1}})=\frac{(x_i\wedge x_{j_1}\wedge...\wedge x_{j_{d-1}})_+}{\|x_{j_1}\wedge...\wedge x_{j_{d-1}}\|_1}=\frac{\mathbf{Vol}_+(\mathcal{P}(x_i,x_{j_1},...,x_{j_{d-1}}))}{\|x_{j_1}\wedge...\wedge x_{j_{d-1}}\|_1}. \tag{3.3}$$

The multi-index $j=(j_1,...,j_{d-1})$ indexes over all combinations of $d-1$ rows $x_{j_1},...,x_{j_{d-1}}\in\mathbb{R}^d$ of $\tilde{X}\in\mathbb{R}^{(n+d)\times d}$, where ith row of $\tilde{X}$ is x_i if $i\le n$, and is e_{i-n} for the standard basis $\{e_1,...,e_d\}$ if $i>n$. $\mathbf{Vol}_+(\mathcal{C})$ denotes the positive part of the signed d-volume of a subset $\mathcal{C}\subset\mathbb{R}^d$. $v\wedge u$ denotes the wedge product [1] of vectors v and u. $\mathcal{P}(x_i,x_{j_1},...,x_{j_{d-1}})$ denotes the parallelotope [2] formulated by vectors $x_i,x_{j_1},...,x_{j_{d-1}}$.

An optimal network can be constructed as:

$$f(x)=\sum_{j=(j_1,...,j_{d-1})} z_j^* k(x,x_{j_1},...,x_{j_{d-1}})$$

where z^* is an optimal solution to (3.2). The optimal hidden neurons are given by a scalar multiple of the generalised cross product $\times(x_{j_1},...,x_{j_{d-1}})=\star(x_{j_1},...,x_{j_{d-1}})$ (Hodge star operator) with breaklines $x\wedge x_i\wedge x_{j_1}\wedge...\wedge x_{j_{d-1}}=0$ corresponding to non-zero z_j^* for $j=(j_1,...,j_{d-1})$.

Proof. See [Pil23]. □

Remark 3.2. We recall that $\mathbf{Vol}(\mathcal{P}(x_i,x_{j_1},...,x_{j_{d-1}}))=0$ if the set $(x_i,x_{j_1},...,x_{j_{d-1}})$ is linearly dependent, and the permutations of indices $x_i,x_{j_1},...,x_{j_{d-1}}$ only change the sign of the volume $\mathbf{Vol}(\mathcal{P}(x_i,x_{j_1},...,x_{j_{d-1}}))$. Hence, it is enough to consider subsets $(x_{j_1},...,x_{j_{d-1}})$ consisting of $d-1$ linearly independent data points and compute $\mathbf{Vol}_+(\mathcal{P}(x_i,x_{j_1},...,x_{j_{d-1}}))$ for each of such subset and a point x_i.

Next, we set $p=2$, so our optimisation problem is

$$p^*\coloneqq\min_{\{u_j,\omega_j\}_{j=1}^m}\mathcal{L}(\sum_{j=1}^m(Xu_j)_+\omega_j,\ y)+\beta\sum_{j=1}^m(\|u_j\|_2^2+\|\omega_j\|_2^2). \tag{3.4}$$

Theorem 3.2. *Consider the following convex Lasso problem*

$$\hat{p}_\lambda\coloneqq\min_{z\in\mathbb{R}^{P(n,d-1)}}\mathcal{L}(Kz,y)+\lambda\|z\|_2. \tag{3.5}$$

The matrix K is defined as $K_{ij}=k(x_i,x_{j_1},...,x_{j_{d-1}})$ for $i\in[n]$ and a multi-index $j=(j_1,...,j_{d-1})$, where

$$k(x_i,x_{j_1},...,x_{j_{d-1}})=\frac{(x_i\wedge x_{j_1}\wedge...\wedge x_{j_{d-1}})_+}{\|x_{j_1}\wedge...\wedge x_{j_{d-1}}\|_2}=\mathbf{dist}_+(x_i,\mathbf{Span}(x_{j_1},...,x_{j_{d-1}})). \tag{3.6}$$

The multi-index $j=(j_1,...,j_{d-1})$ indexes over all combinations of $d-1$ rows $x_{j_1},...,x_{j_{d-1}}\in\mathbb{R}^d$ of $X\in\mathbb{R}^{n\times d}$. $\mathbf{dist}_+(x,S)$ denotes the positive part of the Euclidean distance between x and S.

[1] A generalisation of the cross product to higher dimensions with properties such as anti-symmetry

[2] A generalisation of a parallelogram or a parallelepiped to higher dimensions. It is defined by a set of vectors.

We define the maximum chamber diameter $\mathcal{D}(X)$ *as*

$$\mathcal{D}(X) \coloneqq \max_{\substack{\omega,v\in\mathbb{R}^d,\|\omega\|_2=\|v\|_2=1 \\ sign(X\omega)=sign(Xv)}} \|\omega - v\|_2$$

.

When the maximum chamber diameter satisfies $\mathcal{D}(X) \leq \epsilon$ *for some* $\epsilon \in (0,1)$*, we have the following approximation bounds*

$$p^* \leq \hat{p}_\lambda \leq \frac{1}{1-\epsilon} p^* \tag{3.7}$$

where p^* *is the optimal value of the objective in (3.4).*

Networks that achieve the cost $\hat{p}_\lambda$ *for (3.4) can be constructed as:*

$$f(x) = \sum_{j=(j_1,\ldots,j_{d-1})} z_j^* k(x, x_{j_1}, ..., x_{j_{d-1}})$$

where z^* *is an optimal solution to (3.5).*

Proof. See [Pil23]. □

Remark 3.3. Although the convex formulation (3.5) is not exactly an equivalence of (3.4), it produces a nearly optimal solution if the maximum chamber diameter is small, which is expected for large n.

Here is an example to illustrate the case where $p = 2$. For comparison, we take the same data as for Example 1, and our optimisation problem is (2.3). The optimisation problem (2.3) is the case where $\mathcal{L}(v_1, v_2) = \frac{1}{2}\|v_1 - v_2\|_2^2$ and the regularisation parameter is $\frac{\beta}{2}$ in (3.1).

Example 2. We have $n = 5$ samples in $\mathbb{R}^2$ $(d = 2)$ to form the input data matrix X and the corresponding label vector y.

$$X = [(2,1),(1,2),(2,4),(4,2),(-1,1)]^T$$
$$y = [1,-1,-1,1,-1]^T$$

They are sampled from a simple distribution $y_j = 2 \times \mathbf{1}(x_{j,1} > x_{j,2}) - 1$.

We want to find the minimum objective value for the neural network (2.3). We will find a near-optimal value using Theorem 3.2. Figure 9 illustrates the values of the elements in K. As $d = 2$, multi-index $j = (j_1, ..., j_{d-1})$ indexes just one number j. So, each element $K_{i,j}$ in K is indexed by

$$k_{ij} \coloneqq k(x_i, x_j) = \mathbf{dist}_+(x_i, \mathbf{Span}(x_j))$$

for $i, j \in [n]$.

The figure shows that $k_{i,j} > 0$ only when x_j is on one specific side of the hyperplane passing through the origin and x_i, and $k_{i,j} = 0$ if x_j is on the other side. We can see that the geometric relationship between the data points plays a role in the learning process.

Remark 3.4. In Figure 9, the data points in the shadowed areas do not affect the elements in K. In Figure 8, the data points in the shadowed areas are discarded in the data point selection. For both $p = 1$ and $p = 2$, only one of the half spaces, which are separated by a hyperplane passing through the origin and a data point, matters. Although the convex formulation and the Clifford algebra approach are different formulations of neural networks, they showed the inner workings of hidden neurons in a similar way.

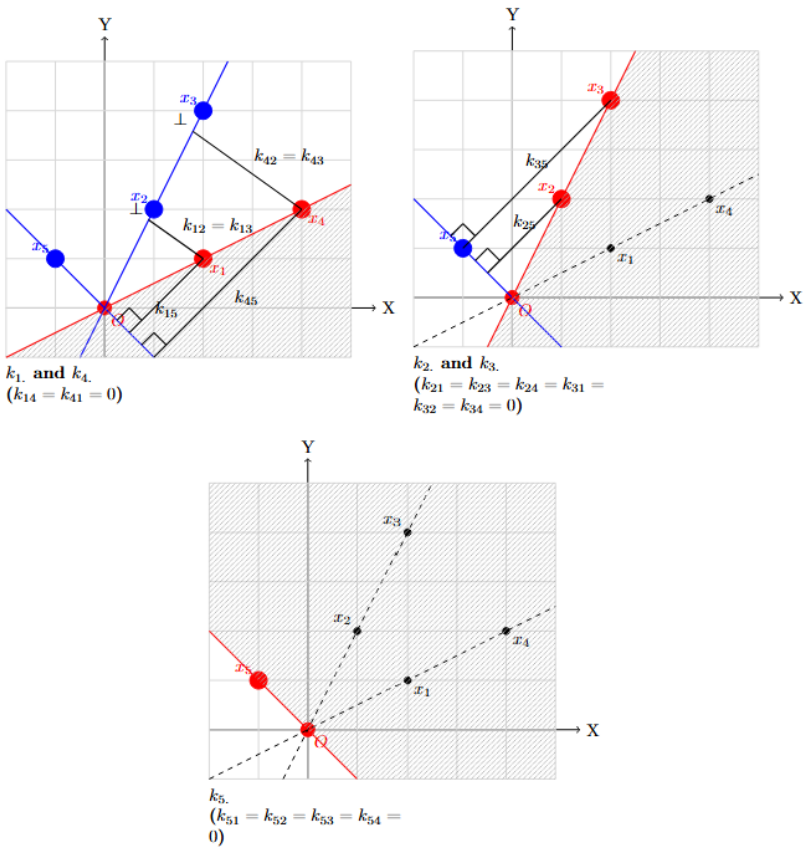


Figure 9: The elements in K.

3.2.1 Comparison with other machine learning methods

We note that the Clifford algebra approach to neural networks is reminiscent of the kernel method and the representer theorem [SHS01], where we reduce a problem of finding an optimal function in a (possibly infinite-dimensional) reproducing kernel Hilbert space $\mathcal{H}$ to a problem of finding optimal coefficients $\hat{\alpha}$ for a kernel matrix K. In Theorem 3.1, we reduce the problem of finding an optimal function in a space $\{f_\theta(X) = \sum_{j=1}^{m}(Xu_j)_+\omega_j \mid \theta \in \Theta\}$ for the non-convex problem (3.1) to the problem (3.2) of finding optimal coefficients z^* for the matrix K defined as (3.3). The convex problem (3.2) is more manageable. However, there is a significant distinction between the kernel method and the Clifford algebra approach to neural networks. For example, the kernel in the kernel method is constructed using a feature map function, which we can choose as long as some conditions are satisfied. On the other hand, the matrix K in the Clifford algebra approach is determined from spatial relations of data points.

Moreover, the Clifford algebra approach to neural networks is reminiscent of support vector machines [CV95], where the final classifier depends only on some of the data points (support vectors) and their geometric locations. For the Clifford algebra approach to neural networks, the exact distance between the data points and the hyperplanes matters as long as the data points are on one particular side. This is different from support vector machines, where we only care about the minimal distance between data points and the hyperplane that separates data points for classification.

4 Conclusion and outlook

Despite the recent success of neural networks, the theoretical explanation of their inner workings is not yet fully developed. This essay introduced some theories about formulations of finite-width neural networks via convex formulations and via the Clifford algebra. These theories gave us interesting interpretations of the inner workings of hidden neurons.

We presented that the convex formulation introduced by [PE20] and [EP21a] gives a lower bound for the non-convex optimisation problem of minimising the sum of a loss function and a regularisation term for two-layer ReLU neural networks. For networks with a large enough number of hidden neurons, strong duality holds. This theoretical result may be related to the empirical success of neural networks with many hidden neurons.

Our simple numerical experiments confirmed the optimality of the convex formulation. We obtained two original observations. Firstly, we observed that the minimised training loss reached its global minimum when networks are trained on noisy data, but does not reach it when networks are trained on noiseless data. One possible explanation for this observation is that the loss function is smoother when the training data is noisy. Secondly, we investigated the optimality of the convex formulation in a random design. Our results suggest that the minimisation by convex formulation is better than SGD with high probability. Theoretical explanations behind these outcomes can be explored in future research.

We then showed that the optimal weights can be calculated explicitly for whitened data. Given the wide range of applications of whitened data, this result will produce useful applications. We also presented the generalised versions of the convex formulation. We considered general loss functions and more complex but widely used neural network architectures. These results ensure that the convex formulation works not only in a regression setting but also in other settings, such as a classification setting. Furthermore, convex formulations are applied to deep neural networks with multiple layers as well [EP21b].

In the last section, we gave an insight into the inner workings of hidden neurons in neural networks through simple examples. We can interpret it as a feature selection in a high-dimensional space (convex formulation) or as a Hodge dual (Clifford algebra). In addition, we can analyse the optimal weights by examining the extreme points in a dual set. This analysis leads to another interesting interpretation of the convex formulation given by convex geometry [Pil23]. Both the convex formulation and the Clifford algebra method consider the construction of optimal weights based on the spatial relationships of data points. We saw that only the data points on one side of a hyperplane, which passes through the origin and a data point, are grouped together in the training process. In future research, it would be interesting to combine these geometric interpretations in a more general inner product space, which may not necessarily be a Euclidean space. We can do it by constructing a feature map that maps data points to a general feature space where the distance between data points is defined.

5 Appendix

5.1 Optimization Theory

Definition 5.1 (Lagrangian)**.** *For an optimisation problem*

$$P : \min_{x \in X} f(x) \quad s.t. \quad g(x) = b.$$

The Lagrangian is defined as $L(x, \lambda) = f(x) + \lambda^T (b - g(x))$.

Theorem 5.1. *(Lagrangian Sufficient Theorem) Let* x^* *be feasible. Suppose* $\exists \lambda^*$ *s.t.* $L(x^*, \lambda^*) \leq L(x, \lambda^*)$ *for all* $x \in X$*. Then,* x^* *is optimal for* P*.*

We define the following notations.

$$\Lambda \coloneqq \{\lambda : \min_{x \in X} L(x, \lambda) > -\infty\}.$$
$$\forall \lambda \in \Lambda, \; L(\lambda) \coloneqq \min_{x \in X} L(x, \lambda).$$
$$X_b \coloneqq \{x : g(x) \leq b\}.$$

Theorem 5.2 (Weak duality)**.** *For any* $x \in X_b$ *and* $\lambda \in \Lambda$*,* $f(x) \geq L(\lambda)$*.*

Definition 5.2 (Dual)**.** *The dual of the optimisation problem*

$$P : \min_{x \in X} f(x) \quad s.t. \quad g(x) = b$$

is

$$D : \max_{\lambda \in \Lambda} \min_{x \in X} L(x, \lambda).$$

Alternatively, it can be written as

$$D : \max_{\lambda \in \Lambda} L(\lambda).$$

Remark 5.1. The weak duality theorem tells us that the solution to the original problem P is no less than the solution to the dual problem D.

References

[Sio58] Maurice Sion. "On general minimax theorems." In: *Pacific Journal of Mathematics* 8.1 (Jan. 1958). Publisher: Pacific Journal of Mathematics, A Non-profit Corporation, pp. 171–176. ISSN: 0030-8730.

[Cov65] Thomas M. Cover. "Geometrical and Statistical Properties of Systems of Linear Inequalities with Applications in Pattern Recognition". In: *IEEE Transactions on Electronic Computers* EC-14.3 (June 1965), pp. 326–334. ISSN: 0367-7508. DOI: 10.1109/PGEC.1965.264137.

[Win66] R. O. Winder. "Partitions of N-Space by Hyperplanes". In: *SIAM Journal on Applied Mathematics* 14.4 (July 1966). Publisher: Society for Industrial and Applied Mathematics, pp. 811–818. ISSN: 0036-1399. DOI: 10.1137/0114068.

[S A67] S. Amari. "Theory of adaptive pattern classifiers". In: *IEEE Trans. Elect. Comput.* EC.16 (1967), pp. 299–307.

[E H70] E. Hoerl and Robert W. Kennard and. "Ridge Regression: Biased Estimation for Nonorthogonal Problems". In: *Technometrics*. ASA Website 12.1 (1970), pp. 55–67. DOI: 10.1080/00401706.1970.10488634.

[EOS86] H. Edelsbrunner, J. O'Rourke, and R. Seidel. "Constructing Arrangements of Lines and Hyperplanes with Applications". en. In: *SIAM Journal on Computing* 15.2 (May 1986), pp. 341–363. ISSN: 0097-5397, 1095-7111. DOI: 10.1137/0215024.

[Ama93] Shun-ichi Amari. "Backpropagation and stochastic gradient descent method". In: *Neurocomputing* 5.4 (June 1993), pp. 185–196. ISSN: 0925-2312. DOI: 10.1016/0925-2312(93)90006-O.

[CV95] Corinna Cortes and Vladimir Vapnik. "Support-vector networks". en. In: *Machine Learning* 20.3 (Sept. 1995), pp. 273–297. ISSN: 1573-0565. DOI: 10.1007/BF00994018.

[Dou95] Mark Dougherty. "A review of neural networks applied to transport". In: *Transportation Research Part C: Emerging Technologies* 3.4 (Aug. 1995), pp. 247–260. ISSN: 0968-090X. DOI: 10.1016/0968-090X(95)00009-8.

[Tib96] Robert Tibshirani. "Regression Shrinkage and Selection Via the Lasso". In: *Journal of the Royal Statistical Society: Series B (Methodological)* 58.1 (Jan. 1996), pp. 267–288. ISSN: 0035-9246. DOI: 10.1111/j.2517-6161.1996.tb02080.x.

[Law+97] S. Lawrence et al. "Face recognition: a convolutional neural-network approach". In: *IEEE Transactions on Neural Networks* 8.1 (Jan. 1997), pp. 98–113. ISSN: 1941-0093. DOI: 10.1109/72.554195.

[Lob+98] Miguel Sousa Lobo et al. "Applications of second-order cone programming". In: *Linear Algebra and its Applications*. International Linear Algebra Society (ILAS) Symposium on Fast Algorithms for Control, Signals and Image Processing 284.1 (Nov. 1998), pp. 193–228. ISSN: 0024-3795. DOI: 10.1016/S0024-3795(98)10032-0.

[Ojh00] P.C. Ojha. "Enumeration of linear threshold functions from the lattice of hyperplane intersections". In: *IEEE Transactions on Neural Networks* 11.4 (July 2000), pp. 839–850. ISSN: 1941-0093. DOI: 10.1109/72.857765.

[SHS01] Bernhard Schölkopf, Ralf Herbrich, and Alex J. Smola. "A Generalized Representer Theorem". en. In: *Computational Learning Theory*. Ed. by David Helmbold and Bob Williamson. Berlin, Heidelberg: Springer, 2001, pp. 416–426. ISBN: 978-3-540-44581-4. DOI: 10.1007/3-540-44581-1_27.

[ERH02] M. Egmont-Petersen, D. de Ridder, and H. Handels. "Image processing with neural networks—a review". In: *Pattern Recognition* 35.10 (Oct. 2002), pp. 2279–2301. ISSN: 0031-3203. DOI: 10.1016/S0031-3203(01)00178-9.

[AG03] F. Alizadeh and D. Goldfarb. "Second-order cone programming". en. In: *Mathematical Programming* 95.1 (Jan. 2003), pp. 3–51. ISSN: 1436-4646. DOI: 10.1007/s10107-002-0339-5.

[BV04] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. en. ISBN: 9780511804441 Publisher: Cambridge University Press. Mar. 2004. DOI: 10.1017/CBO9780511804441.

[YL06] Ming Yuan and Yi Lin. "Model Selection and Estimation in Regression with Grouped Variables". In: *Journal of the Royal Statistical Society Series B: Statistical Methodology* 68.1 (Feb. 2006), pp. 49–67. ISSN: 1369-7412. DOI: 10.1111/j.1467-9868.2005.00532.x.

[DDL12] Leo Dorst, Chris Doran, and Joan Lasenby. *Applications of Geometric Algebra in Computer Science and Engineering*. en. Google-Books-ID: z8_TBwAAQBAJ. Springer Science & Business Media, Dec. 2012. ISBN: 978-1-4612-0089-5.

[Hit12] Eckhard Hitzer. "Introduction to Clifford's Geometric Algebra". In: 51.4 (2012), pp. 338–350. DOI: 10.11499/sicejl.51.338.

[Art16] Emil Artin. *Geometric Algebra*. en. Google-Books-ID: zD1xCwAAQBAJ. Courier Dover Publications, Jan. 2016. ISBN: 978-0-486-80920-5.

[He+16] Kaiming He et al. "Deep Residual Learning for Image Recognition". In: 2016, pp. 770–778.

[PK17] Engin Pekel and Selin Soner Kara. "A COMPREHENSIVE REVIEW FOR ARTIFICAL NEURAL NETWORK APPLICATION TO PUBLIC TRANSPORTATION". en. In: *Sigma Journal of Engineering and Natural Sciences* 35.1 (Mar. 2017). Number: 1 Publisher: Yildiz Technical University, pp. 157–179. ISSN: 1304-7191, 1304-7205.

[SSS17] Shai Shalev-Shwartz, Ohad Shamir, and Shaked Shammah. "Failures of Gradient-Based Deep Learning". en. In: *Proceedings of the 34th International Conference on Machine Learning*. ISSN: 2640-3498. PMLR, July 2017, pp. 3067–3075.

[Yu+17] Haiyang Yu et al. "Spatiotemporal Recurrent Convolutional Networks for Traffic Prediction in Transportation Networks". en. In: *Sensors* 17.7 (July 2017). Number: 7 Publisher: Multidisciplinary Digital Publishing Institute, p. 1501. ISSN: 1424-8220. DOI: 10.3390/s17071501.

[Aro+18] Raman Arora et al. *Understanding Deep Neural Networks with Rectified Linear Units*. arXiv:1611.01491 [cs]. Feb. 2018. DOI: 10.48550/arXiv.1611.01491.

[ACH18] Sanjeev Arora, Nadav Cohen, and Elad Hazan. "On the Optimization of Deep Networks: Implicit Acceleration by Overparameterization". en. In: *Proceedings of the 35th International Conference on Machine Learning*. ISSN: 2640-3498. PMLR, July 2018, pp. 244–253.

[CB18] Lénaïc Chizat and Francis Bach. "On the Global Convergence of Gradient Descent for Over-parameterized Models using Optimal Transport". In: *Advances in Neural Information Processing Systems*. Vol. 31. Curran Associates, Inc., 2018.

[JL18] Bangti Jin and Xiliang Lu. "On the regularizing property of stochastic gradient descent". en. In: *Inverse Problems* 35.1 (Jan. 2018), p. 015004. ISSN: 0266-5611. DOI: 10.1088/1361-6420/aaea2a.

[KLS18] Agnan Kessy, Alex Lewin, and Korbinian Strimmer. "Optimal Whitening and Decorrelation". en. In: *The American Statistician* 72.4 (Oct. 2018), pp. 309–314. ISSN: 0003-1305, 1537-2731. DOI: 10.1080/00031305.2016.1277159.

[Ney+18] Behnam Neyshabur et al. *Towards Understanding the Role of Over-Parametrization in Generalization of Neural Networks*. arXiv:1805.12076 [cs]. May 2018. DOI: 10.48550/arXiv.1805.12076.

[SS18] Itay Safran and Ohad Shamir. "Spurious Local Minima are Common in Two-Layer ReLU Neural Networks". en. In: *Proceedings of the 35th International Conference on Machine Learning*. ISSN: 2640-3498. PMLR, July 2018, pp. 4433–4441.

[BK19] E. G. Bazulin and D. A. Konovalov. "Applying the Whitening Transformation to Echo Signals for Reducing Pattern Noise in Ultrasonic Testing". en. In: *Russian Journal of Nondestructive Testing* 55.11 (Nov. 2019), pp. 791–802. ISSN: 1608-3385. DOI: 10.1134/S1061830919110020.

[Ser19] Will Serrano. "Neural Networks in Big Data and Web Search". en. In: *Data* 4.1 (Mar. 2019). Number: 1 Publisher: Multidisciplinary Digital Publishing Institute, p. 7. ISSN: 2306-5729. DOI: 10.3390/data4010007.

[Alo20] Aziz Alotaibi. "Deep Generative Adversarial Networks for Image-to-Image Translation: A Review". en. In: *Symmetry* 12.10 (Oct. 2020). Number: 10 Publisher: Multidisciplinary Digital Publishing Institute, p. 1705. ISSN: 2073-8994. DOI: 10.3390/sym12101705.

[PE20] Mert Pilanci and Tolga Ergen. "Neural Networks are Convex Regularizers: Exact Polynomial-time Convex Optimization Formulations for Two-layer Networks". en. In: *Proceedings of the 37th International Conference on Machine Learning*. ISSN: 2640-3498. PMLR, Nov. 2020, pp. 7695–7705.

[EP21a] Tolga Ergen and Mert Pilanci. "Convex Geometry and Duality of Over-parameterized Neural Networks". In: *Journal of Machine Learning Research* 22.212 (2021), pp. 1–63. ISSN: 1533-7928.

[EP21b] Tolga Ergen and Mert Pilanci. "Global Optimality Beyond Two Layers: Training Deep ReLU Networks via Convex Programs". en. In: *Proceedings of the 38th International Conference on Machine Learning*. ISSN: 2640-3498. PMLR, July 2021, pp. 2993–3003.

[Sam+21] Wojciech Samek et al. "Explaining Deep Neural Networks and Beyond: A Review of Methods and Applications". In: *Proceedings of the IEEE* 109.3 (Mar. 2021), pp. 247–278. ISSN: 1558-2256. DOI: 10.1109/JPROC.2021.3060483.

[Gar22] Julia García Cabello. "Mathematical Neural Networks". en. In: *Axioms* 11.2 (Feb. 2022). Number: 2 Publisher: Multidisciplinary Digital Publishing Institute, p. 80. ISSN: 2075-1680. DOI: 10.3390/axioms11020080.

[BMP23] Daniel Bienstock, Gonzalo Muñoz, and Sebastian Pokutta. "Principled deep neural network training through linear programming". In: *Discrete Optimization* 49 (Aug. 2023), p. 100795. ISSN: 1572-5286. DOI: 10.1016/j.disopt.2023.100795.

[MD23] Ragini Mokkapati and Venkata Lakshmi Dasari. "A Comprehensive Review on Areas and Applications of Artificial Intelligence, Machine Learning, Deep Learning, and Data Science". In: *2023 3rd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA)*. Feb. 2023, pp. 427–435. DOI: 10.1109/ICIMIA60377.2023.10426237.

[Nav+23] Humza Naveed et al. *A Comprehensive Overview of Large Language Models*. Version Number: 10. 2023. DOI: 10.48550/ARXIV.2307.06435.

[Pil23] Mert Pilanci. "From Complexity to Clarity: Analytical Expressions of Deep Neural Network Weights via Clifford Algebra and Convexity". en. In: *Transactions on machine learning research* (Oct. 2023). Publisher: Transactions on Machine Learning Research.

[RBS23] Kodali Radha, Mohan Bansal, and Rajeev Sharma. "Whitening Transformation of i-vectors in Closed-Set Speaker Verification of Children". In: *2023 10th International Conference on Signal Processing and Integrated Networks (SPIN)*. ISSN: 2688-769X. Mar. 2023, pp. 243–248. DOI: 10.1109/SPIN57001.2023.10116604.

[Rai+24] Mohaimenul Azam Khan Raiaan et al. "A Review on Large Language Models: Architectures, Applications, Taxonomies, Open Issues and Challenges". In: *IEEE Access* 12 (2024), pp. 26839–26874. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2024.3365742.

[25] *Partlll_Essay_codes*. 2025. DOI: https://github.com/hw581/Partlll_Essay.git.

[MRS] Ezra Miller, Victor Reiner, and Bernd Sturmfels. *Geometric Combinatorics*. en. Google-Books-ID: W_SPdwfPTw8C. American Mathematical Soc. ISBN: 978-0-8218-8695-3.