Convex Optimization Formulation of Neural Networks: Theories, Applications and Beyond

Yifei Wang

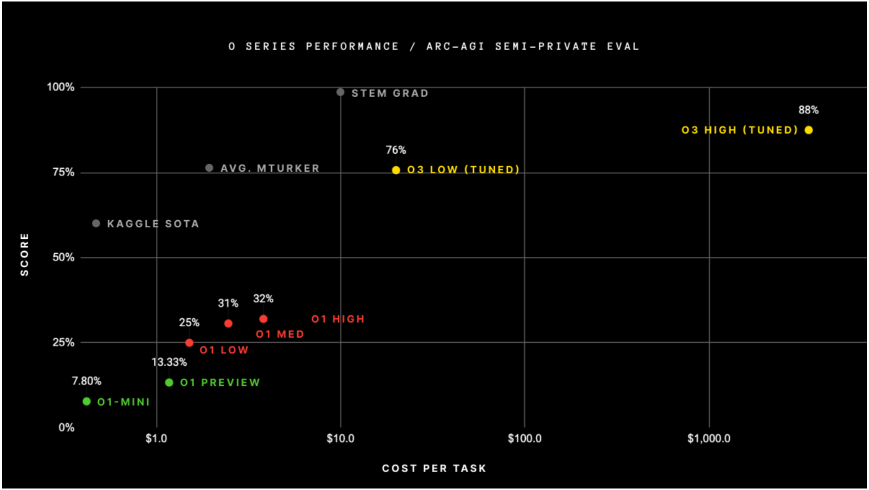
Department of Electrical Engineering, Stanford University

Feb. 5th, 2025

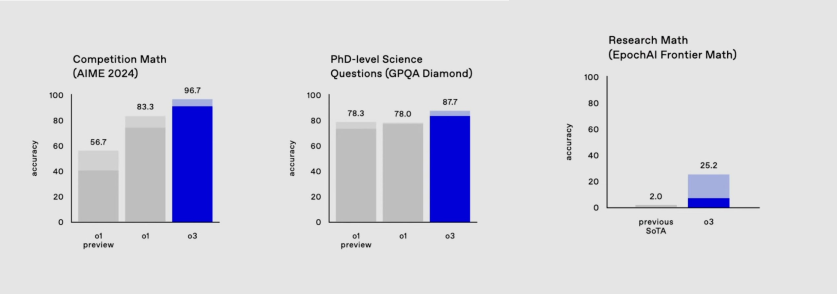
Outline

- understanding neural network loss landscapes via convex optimization
- recovery of planted models via neural networks
- deep parallel networks with strong duality
- geometric algebra perspective of convex neural networks
- complexity characterization: hardness of approximation

Recent Developments in LLMs



LLMs Benchmarks



Motivation

- Large-language models (LLMs) have achieved remarkable success in various tasks, but their training is computationally expensive and requires significant amounts of data and computing resources.
- Finetuning pretrained LLMs on specific tasks is also computationally expensive.
- Convex optimization provides a new perspective to analyze neural networks and design more efficient training and fine-tuning strategies

Theoretical Frameworks to Analyze Over-parametrized Neural Network Training

- Neural Tangent Kernel (Jacot et al. 2018)[1]
- Mean-field theory (Mei et al. 2018)[2]
- Convex optimization formulations

[1] Jacot, Gabriel, Hongler, Neural Tangent Kernel: Convergence and Generalization in Neural Networks. NeurIPS 2018

[2] Mei, Montanari, Nguyen. A mean field view of the landscape of two-layer neural networks. PNAS, 2018

The Simplest Neural Network Architecture

- Data: $X \in \mathbb{R}^{n\times d}$ label: $y \in \mathbb{R}^n$.
- Two-layer ReLU NN:

$$f^{\text{ReLU}}(x;\Theta) = (x^T W_1)_+ w_2 = \sum_{i=1}^{m} (x^T w_{1,i})_+ w_{2,i},$$

where $\Theta = (W_1, w_2)$, $W_1 \in \mathbb{R}^{d\times m}, w_2 \in \mathbb{R}^m$.

Regularized Training Problem

- Consider the ReLU NN architecture.

$$\min_{\Theta} \ell(f^{\mathsf{ReLU}}(X;\Theta), y) + \beta \mathcal{R}_2(\Theta),$$

where $\mathcal{R}_p(\Theta) = \frac{1}{2}(\|W_1\|_p^2 + \|w_2\|_p^2)$.

- $l(\cdot, y)$ is a convex loss function, e.g., square or logistic loss

Convex Optimization Formulation

- An optimal neural network can be constructed based on a solution of the convex program[1]

$$\min_{\{(u_i,u_i')\}_{i=1}^p} \ell\left(\sum_{i=1}^p D_i X(u_i-u_i'), y\right) + \beta \sum_{i=1}^p (\|u_i\|_2 + \|u_i'\|_2)$$
$$\text{s.t. } (2D_i - I)Xu_i \geq 0, (2D_i - I)Xu_i' \geq 0.$$

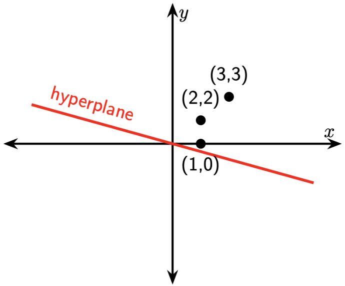
where $D_1, \dots, D_p$ are the enumeration of all possible hyperplane arrangements

$$\{\text{diag}(\mathbf{1}(Xu \geqslant 0)) | u \in \mathbb{R}^d\}.$$

[1] M. Pilanci, T. Ergen. Neural Networks are Convex Regularizers: Exact Polynomial-time Convex Optimization Formulations for Two-Layer Networks. ICML 2020.

Hyperplane arrangements

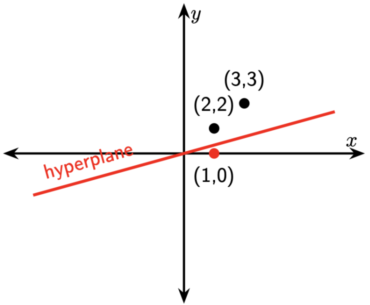
- $n = 3$ samples in $\mathbb{R}^d$, $d = 2$. $X = \begin{bmatrix} x_1^T \\ x_2^T \\ x_3^T \end{bmatrix} = \begin{bmatrix} 3 & 3 \\ 2 & 2 \\ 1 & 0 \end{bmatrix}, y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}$.



$$D_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, D_1 X = \begin{bmatrix} 3 & 3 \\ 2 & 2 \\ 1 & 0 \end{bmatrix}.$$

Hyperplane Arrangements

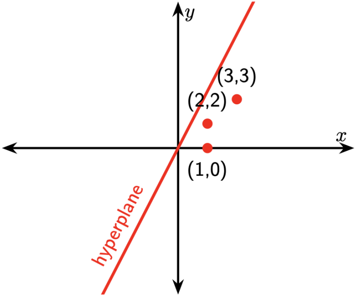
- $n = 3$ samples in $\mathbb{R}^d$, $d = 2$. $X = \begin{bmatrix} x_1^T \\ x_2^T \\ x_3^T \end{bmatrix} = \begin{bmatrix} 3 & 3 \\ 2 & 2 \\ 1 & 0 \end{bmatrix}, y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}$.



$$D_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, D_2X = \begin{bmatrix} 3 & 3 \\ 2 & 2 \\ 0 & 0 \end{bmatrix}.$$

Hyperplane arrangements

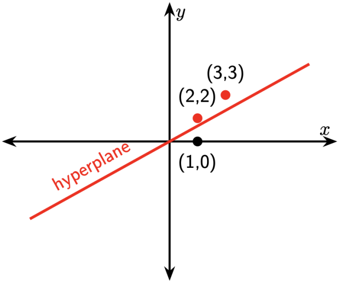
- $n = 3$ samples in $\mathbb{R}^d$, $d = 2$. $X = \begin{bmatrix} x_1^T \\ x_2^T \\ x_3^T \end{bmatrix} = \begin{bmatrix} 3 & 3 \\ 2 & 2 \\ 1 & 0 \end{bmatrix}, y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}$.



$$D_3 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, D_3 X = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}.$$

Hyperplane arrangements

- $n = 3$ samples in $\mathbb{R}^d$, $d = 2$. $X = \begin{bmatrix} x_1^T \\ x_2^T \\ x_3^T \end{bmatrix} = \begin{bmatrix} 3 & 3 \\ 2 & 2 \\ 1 & 0 \end{bmatrix}, y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}$.



$$D_4 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, D_4 X = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \end{bmatrix}.$$

Questions: Lanscape of Neural Network

- Convex program finds an optimal solution for the nonconvex training problem.
- How to find all global optima of neural networks?
- How do local minimizers (Clarke stationary points) look like?

Q&A: Lanscape of Neural Network

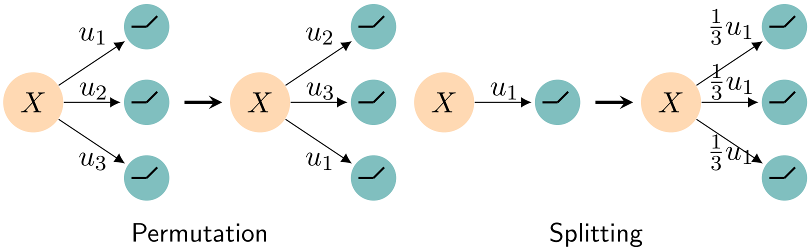
- Convex program finds an optimal solution for the nonconvex training problem.
- How to find all global optima of neural networks?
  - All global optima can be found from the convex program up to permutation and splitting[1].
- How do local minimizers (Clarke stationary points) look like?
  - All Clarke stationary points can be found from the convex program with subsampled hyperplane arrangements.
  - Popular local optimizers (SGD, Adam etc) converge to such stationary points

[1] Yifei Wang, Jonathan Lacotte, Mert Pilanci, The Hidden Convex Optimization Landscape of Two-Layer ReLU Neural Networks: an Exact Characterization of the Optimal Solutions, International Conference on Learning Representations (ICLR) 2022 Oral.

Global Optima Characterization

Theorem

Assume that $m \geq m^*$, where $m^* \leq n+1$ is a critical threshold. All optimal solution of p_{noncvx} can be found from the optimal solutions of p_{convex} up to permutation and splitting.



Clarke Stationary Point

- Denote $\mathcal{L}(\theta)$ as the objective of the nonconvex problem.
- Clarke's subdifferential:

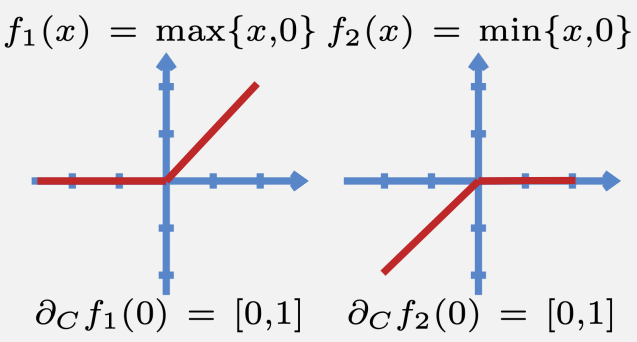
$$\partial_C \mathcal{L}(x) = \mathbf{Co}\left\{\lim_{k\to\infty} \nabla \mathcal{L}\left(x_k\right) \mid x_k \to x, x_k \in D, \lim_{k\to\infty} \nabla \mathcal{L}\left(x_k\right) \text{ exists } \right\}$$

- Clarke stationary point:

$$\theta : 0 \in \partial_C \mathcal{L}(\theta),$$

- Any local minimizer of $\mathcal{L}$ is a Clarke stationary point.
- The limit points of SGD are almost surely Clarke stationary with respect to the nonconvex problem.

Clarke Subdifferential



Characterization of Clarke Stationary Points

Theorem

Suppose that $\theta = (\mathbf{W}_1, \mathbf{w}_2)$ is a Clarke's stationary point of the nonconvex problem. Then, θ corresponds to a global optimum of the subsampled convex program:

$$\min_{(\mathbf{u}_i, \mathbf{u}_i')_{i\in\mathcal{I}}} \ell\Big(\sum_{i\in\mathcal{I}} \mathbf{D}_i \mathbf{X}(\mathbf{w}_i - \mathbf{w}_i'), \mathbf{y}\Big) + \beta \sum_{i\in\mathcal{I}} (\|\mathbf{w}_i\|_2 + \|\mathbf{w}_i'\|_2),$$

$$\text{s.t. } (2\mathbf{D}_i - \mathbf{I}_n)\mathbf{X}\mathbf{w}_i \geq 0, (2\mathbf{D}_i - \mathbf{I}_n)\mathbf{X}\mathbf{w}_i' \geq 0, i \in \mathcal{I},$$

where $\mathcal{I} = \{i \in [p] |$ there exists $k \in [m]$ s.t. $D_i = \mathrm{diag}(\mathbb{I}(Xu \geq 0))\}$.

Questions: Neural Recovery

- Does global optima generalize well?
- Under which conditions, global optima generalize well?
- It has become a common practice in ML to use overly complex models. If we use a more complicated model that contains the true model (say a linear model), what is the price we pay compared to not using the linear model?

Q&A: Neural Recovery

- Do global optima generalize well?
  - Global optima picks the simplest model.[1]
  - Sparse in terms of number of neurons due to the group L1 penalty
- Under which conditions, global optima generalize well?
  - Linear model recovery: the global optima generalize well only when $n > 2d$.
- If we use a more complicated model that contains the true model (say a linear model), what is the price we pay compared to not using the linear model?
  - **We need exactly $2\times$ samples compared to using the linear model.**

[1] Yifei Wang, Yixuan Hua, Emmanuel Candes, Mert Pilanci, Overparameterized ReLU Neural Networks Learn the Simplest Models: Neural Isometry and Exact Recovery. Transactions on Information Theory 2025.

Linear Model Recovery

- Suppose that the ground truth model is linear, i.e., $y = Xw^*$ for some unknown $w^* \in \mathbb{R}^d$
- Suppose we train a ReLU neural network with linear skip connection

$$f^{\mathrm{ReLU-skip}}(\mathbf{X};\Theta) = \mathbf{X}\mathbf{w}_{1,1}w_{2,1} + \sum_{i=2}^{m}(\mathbf{X}\mathbf{w}_{1,i})_+ w_{2,i}, \Theta = \{\mathbf{W}_1, \mathbf{w}_2\}.$$

- **Question: Does the neural network recover the ground truth linear model?**
- **Surprising result: We can characterize precisely when this happens for Gaussian training data:**

  ReLU network requires $2\times$ samples compared to a linear model

Convex Formulation

- Consider the minimum norm interpolation problem

$$\min_{\Theta} \underbrace{\|\mathbf{W}_1\|_F^2 + \|\mathbf{w}_2\|_2^2}_{\|\Theta\|_F^2}, \text{ s.t. } f^{\mathrm{ReLU-skip}}(\mathbf{X};\Theta) = \mathbf{y}.$$

- Equivalent to the following convex problem

$$\begin{aligned}\min_{\mathbf{w}_0,\left(\mathbf{w}_j,\mathbf{w}_j'\right)_{j=1}^p} \quad & \sum_{j=1}^p \left(\|\mathbf{w}_j\|_2 + \left\|\mathbf{w}_j'\right\|_2\right) \\ \text{s.t.} \quad & \mathbf{X}\mathbf{w}_0 + \sum_{j=1}^p \mathbf{D}_j\mathbf{X}\left(\mathbf{w}_j - \mathbf{w}_j'\right) = \mathbf{y}, \\ & (2\mathbf{D}_j - \mathbf{I}_n)\mathbf{X}\mathbf{w}_j \geq 0, (2\mathbf{D}_j - \mathbf{I}_n)\mathbf{X}\mathbf{w}_j' \geq 0, j \in [p].\end{aligned}$$

- **Intuition:** Most variable blocks will be zero due to the group Lasso regularization

Linear Neural Isometry Condition

Definition (Linear Neural Isometry Condition)

The linear neural isometry condition for recovering the linear model $\mathbf{y} = \mathbf{X}\mathbf{w}^*$ is given by:

$$\left\| \mathbf{X}^T \mathbf{D}_j \mathbf{X} \left(\mathbf{X}^T \mathbf{X} \right)^{-1} \hat{\mathbf{w}}^* \right\|_2 < 1, \forall j \in [p], \tag{NIC-L}$$

where $\hat{\mathbf{w}}^* := \frac{\mathbf{w}^*}{\|\mathbf{w}^*\|_2}$.

- This is a variant of the Restricted Isometry Property. It holds for random i.i.d. data

Sharp Phase Transition

Theorem

Suppose that the training data matrix $\mathbf{X} \in \mathbb{R}^{n\times d}$ is i.i.d. Gaussian, and $f(\mathbf{X};\Theta)$ is a two-layer ReLU network containing arbitrarily many neurons with skip connection. Assume that the response is a noiseless linear model $\mathbf{y} = \mathbf{X}\mathbf{w}^*$. The condition $n > 2d$ is sufficient for ReLU networks with skip connections or normalization layers to recover the planted model exactly with high probability. Furthermore, when $n < 2d$, the recovery fails with high probability.

- Therefore, $n = 2d$ precisely characterizes the phase transition for the ReLU network to recover the linear ground truth.
- **Why this value?** $\frac{n}{2}$ is the Gaussian Width of the positive orthant, which is due to the ReLU activation

Sharp Phase Transition

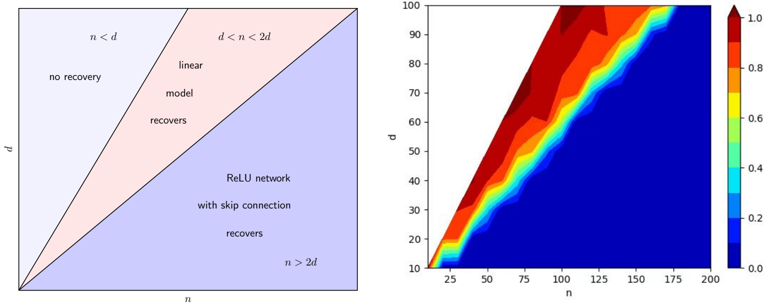


Figure: Phase transition in recovering a linear neuron. Left: when $n \in (d, 2d)$, ReLU network fails to recover a planted linear model, while a simple linear model succeeds in recovery. Right: Empirical generalization error in recovering a linear neuron by solving the convex program numerically.

2D Illustration

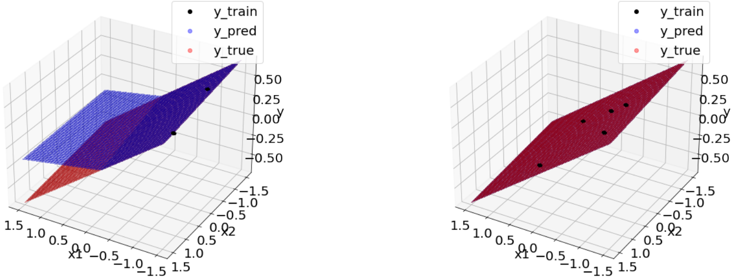
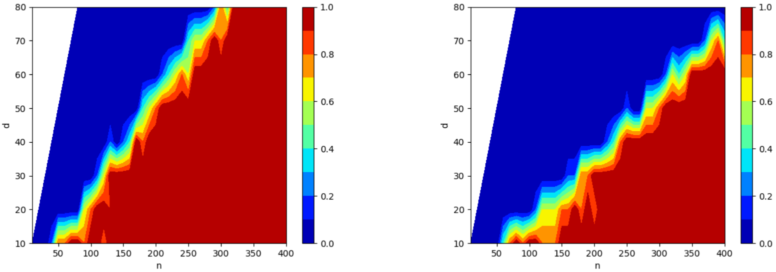


Figure: Optimal ReLU NNs found via the convex program. Left: A ReLU neuron is fitted to the observations generated from a linear model when $n = 2, d = 2$. Right: Only a linear neuron is fitted to the observations generated from a linear model when $n = 5, d = 2$.

Recovery of Multiple Neurons



(a) $k = 2,\ \mathbf{w}_1^* = \mathbf{e}_1,\ \mathbf{w}_2^* = \mathbf{e}_2,$

(b) $k = 3,\ \mathbf{w}_i^* = \mathbf{e}_i (i = 1, 2, 3)$

Figure: The empirical probability of exact recovery of the planted ReLU neurons by solving the group ℓ_1-minimization problem.

- We prove that multiple neurons can be recovered exactly via the convex NN solution under i.i.d. Gaussian data assumption

Question: Convex formulations for deep networks?

- Can we generalize the convex duality result to deep networks?
- Can we characterize the duality gap (P-D)?
- Is there an architecture for which strong duality holds regardless of the depth?

Q&A: Convex formulations for deep networks?

- Can we generalize the convex duality result to deep networks?
  - Yes, but it depends on the network architecture.[1]
- Can we characterize the duality gap (P-D)?
  - Yes, we have a closed-form expression of the duality gap for deep linear networks.
- Is there an architecture for which strong duality holds regardless of the depth?
  - Yes, **parallel architectures** have zero duality gap, i.e., there are **exact convex formulations**
  - In contrast, non-parallel architectures have non-zero duality gap

[1] Yifei Wang, Tolga Ergen, Mert Pilanci, Parallel Deep Neural Networks Have Zero Duality Gap, International Conference on Learning Representations (ICLR) 2023.

Standard Architecture and Parallel Architecture

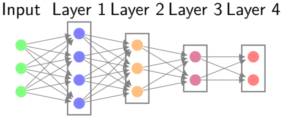


Figure: Standard Architecture

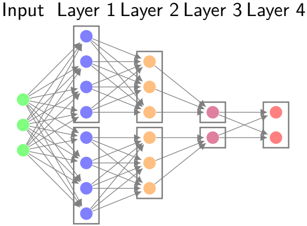


Figure: Parallel Architecture

- Standard Architecture

$$f_{\boldsymbol{\theta}}(\mathbf{X}) = \mathbf{A}_{L-1}\mathbf{W}_L, \mathbf{A}_l = \phi(\mathbf{A}_{l-1}\mathbf{W}_l), \forall l \in [L-1], \mathbf{A}_0 = \mathbf{X},$$

- Parallel Architecture ($j \in [m]$, m is the number of branches)

$$f_{\boldsymbol{\theta}}^{\mathrm{prl}}(\mathbf{X}) = \mathbf{A}_{L-1}\mathbf{W}_L, \mathbf{A}_{l,j} = \phi(\mathbf{A}_{l-1,j}\mathbf{W}_{l,j}), \forall l \in [L-1], \mathbf{A}_{0,j} = \mathbf{X}$$

Main Result

Theorem

For $L \geq 3$, there exists an activation function ϕ and an L-layer standard neural network such that the strong duality does not hold, i.e., $P > D$. In contrast, for any L-layer parallel neural network with linear or ReLU activations and sufficiently large number of branches, strong duality holds, i.e., $P = D$.

Negative Result for Standard Networks

- With a standard linear activation NN

$$f(\mathbf{X};\Theta) = \mathbf{X}\mathbf{W}_1 \dots \mathbf{W}_L,$$

The minimum norm optimization problem writes as

$$P_{\text{lin}} = \min_{\{\mathbf{W}_l\}_{l=1}^L} \frac{1}{2}\sum_{l=1}^{L} \|\mathbf{W}_l\|_F^2, \quad \text{s.t. } \mathbf{X}\mathbf{W}_1 \dots \mathbf{W}_L = \mathbf{Y},$$

Primal Problem Reformulation

- By introducing a scale parameter t, the primal problem can be reformulated as

$$P_{\text{lin}} = \min_{t>0} \frac{L-2}{2} t^2 + P_{\text{lin}}(t),$$

where the subproblem $P_{\text{lin}}(t)$ is defined as

$$\begin{aligned} P_{\text{lin}}(t) = \min_{\{\mathbf{W}_l\}_{l=1}^{L}} & \sum_{j=1}^{K} \|\mathbf{w}_{L,j}^{\text{row}}\|_2, \\ \text{s.t. } & \mathbf{X}\mathbf{W}_1 \dots \mathbf{W}_L = \mathbf{Y}, \|\mathbf{W}_i\|_F \le t, i \in [L-2], \\ & \|\mathbf{w}_{L-1,j}^{\text{col}}\|_2 \le 1, j \in [m_{L-1}]. \end{aligned}$$

- The dual problem follows

$$\begin{aligned} D_{\text{lin}}(t) = & \max_{\mathbf{\Lambda}} \operatorname{tr}(\mathbf{\Lambda}^T \mathbf{Y}) \\ \text{s.t. } & \max_{\|\mathbf{W}_i\|_F \le t, i \in [L-2], \|\mathbf{w}_{L-1}\|_2 \le 1} \|\mathbf{\Lambda}^T \mathbf{X} \mathbf{W}_1 \dots \mathbf{W}_{L-2} \mathbf{w}_{L-1}\|_2 \le 1. \end{aligned}$$

Duality Gap

Theorem

Assume that $m_l \geq \text{rank}(\mathbf{X}^\dagger \mathbf{Y})$ *for* $l = 1, \ldots, L-1$. *For fixed* $t > 0$, *the optimal value of* $P_{\text{lin}}(t)$ *and* $D_{\text{lin}}(t)$ *are given by*

$$P_{\text{lin}}(t) = t^{-(L-2)} \|\mathbf{X}^\dagger \mathbf{Y}\|_{S_{2/L}},$$

and

$$D_{\text{lin}}(t) = t^{-(L-2)} \|\mathbf{X}^\dagger \mathbf{Y}\|_*.$$

Here $\|\cdot\|_*$ *represents the nuclear norm.* $P_{\text{lin}}(t) = D_{\text{lin}}(t)$ *if and only if the singular values of* $\mathbf{X}^\dagger \mathbf{Y}$ *are equal.*

- Implies that the duality gap is non-zero for non-parallel NN architectures

Question: Sampling Hyperplane Arrangements

- Enumerate all hyperplane arrangements can be computationally expensive.
- Can we sample hyperplane arrangements more efficiently?

Q & A: Sampling Hyperplane Arrangements

- Enumerate all hyperplane arrangements can be computationally expensive.
- Can we sample hyperplane arrangements more efficiently?
  - Yes, we can sample hyperplane arrangements more efficiently using geometric algebra.

Practical Algorithm

Algorithm Convex neural network training via Gaussian sampling

Require: Number of hyperplane arrangement samples k, regularization parameter $\beta > 0$.

1: Sample k i.i.d. random vectors $v_1, \ldots, v_k$ following $\mathcal{N}(0, I)$.
2: Compute $\bar{D}_i = \mathrm{diag}(\mathbb{I}(Xv_i \geq 0))$ for $i \in [k]$.
3: Solve the convex optimization problem with the subsampled patterns.

Geometric Algebra

- $\mathbb{G}^d$: geometric algebra over a d-dimensional Euclidean space
- Hypercomplex numbers: extension of complex numbers/quaternions
- Each $M \in \mathbb{G}^d$ is a multivector

$$M = \langle M \rangle_0 + \langle M \rangle_1 + \cdots + \langle M \rangle_d \, .$$

  where $\langle M \rangle_k$ denotes the k-vector part of M
- A k-blade $M = \alpha_1 \wedge \cdots \wedge \alpha_k$ is a k-vector that can be expressed as the wedge product of k vectors $\alpha_1, \ldots, \alpha_k \in \mathbb{R}^d$.

Example of 2-blade and 3-blade

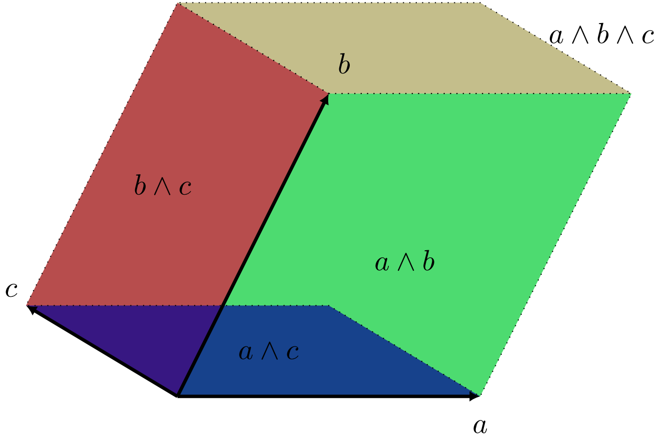


Figure: $a \wedge b$ is a 2-blade, which represents the signed area of the parallelogram spanned by a and b. $a \wedge b \wedge c$ is a 3-blade, which represents the signed volume of the parallelepiped spanned by a, b, c.

Calculation of generalized wedge product

Definition

Let $x_1, \dots, x_{d-1} \in \mathbb{R}^d$ be a set of $d-1$ vectors and denote $A = \begin{bmatrix} x_1 & \dots & x_{d-1} \end{bmatrix}$ as the matrix whose columns are the vectors $\{x_i\}_{i=1}^{d-1}$. The generalized cross-product of $\{x_i\}_{i=1}^{d-1}$ is defined as

$$\times(x_1, \dots, x_{d-1}) \triangleq \sum_{i=1} (-1)^{i-1} |A_i| e_i,$$

where $|A_i|$ is the determinant of the square matrix A_i, A_i is the square matrix obtained from A by deleting its i-th row.

- The generalized cross-product forms a vector which is orthogonal to all of them.

Relation between cross product and wedge product

- The cross product and the wedge product are related via the formula

$$\begin{aligned} x^T \times (x_1, \dots, x_{d-1}) =& \textbf{Vol}(\mathcal{P}(x, x_1, \dots, x_{d-1})) \\ =& (x \wedge x_1 \wedge \dots \wedge x_{d-1}) \mathbf{I}^{-1}, \end{aligned}$$

  where $\mathcal{P}(x, x_1, \dots, x_{d-1})$ is the parallelotope spanned by vectors $\{x, x_1, \dots, x_{d-1}\}$, whose volume is given by the determinant $\mathbf{det}[x, x_1, ..., x_d]$.

Convex NN from a Geometric Algebra Perspective

- Convex optimization formulation[1]

$$\min_{z} \ell\left(Kz, y\right) + \beta \|z\|_1,$$

where $K_{i,j} = \kappa(x_i, x_{j_1}, \ldots, x_{j_{d-1}})$ for $j = (j_1, \ldots, j_{d-1})$ which enumerates over all combinations of $d-1$ rows of $X \in \mathbb{R}^{n\times d}$ and

$$\begin{aligned}\kappa(x, u_1, \ldots, u_{d-1}) &= \frac{\left(x^T \times (u_1, \ldots, u_{d-1})\right)_+}{\| \times (u_1, \ldots, u_{d-1})\|_2} \\ &= \frac{\left(\textbf{Vol}(\mathcal{P}(x, u_1, \ldots, u_{d-1}))\right)_+}{\| \times (u_1, \ldots, u_{d-1})\|_2}.\end{aligned}$$

Here, $\times$ is the generalized cross-product and $\mathcal{P}(x, u_1, \ldots, u_{d-1})$ is the parallelotope spanned by vectors $\{x, u_1, \ldots, u_{d-1}\}$.

[1] Mert Pilanci. From Complexity to Clarity: Analytical Expressions of Deep Neural Network Weights via Clifford Algebra and Convexity. Transactions on Machine Learning Research 2024.

Optimal Weights in NN

- From an optimal solution z^* to the Lasso problem, an optimal ReLU neural network can be constructed as follows:

$$f^{\text{ReLU}}(x;\Theta^*) = \sum_{j=(j_1,\dots,j_{d-1})} z_j^* \kappa(x, x_{j_1}, \dots, x_{j_{d-1}}).$$

- The optimal weights in the training problem have a closed-form formula $\times(x_{j_1}, \dots, x_{j_{d-1}})$, where $\{x_{j_i}\}_{i=1}^{d-1}$ is a subset of training data indexed by $j_1, \dots, j_{d-1}$ and $\times(x_{j_1}, \dots, x_{j_{d-1}})$ is the generalized cross-product of $\{x_{j_i}\}_{i=1}^{d-1}$.

Approximate Generalized Cross-product by Sketching

- The hyperplane arrangement patterns of the optimal neural network take the form:

$$D = \mathrm{diag}(\mathbb{I}(Xh \geq 0)), \quad h = \times(x_{j_1}, \ldots, x_{j_{d-1}}).$$

- sketch size: $r \ll d$, embedding matrix $S \in \mathbb{R}^{r \times d}$
- project the training data to dimension r, i.e., XS^T.
- approximate the generalized cross-product by the one computed from the projected data:

$$\tilde{v} = \times(Sx_{j_1}, \ldots, Sx_{j_{r-1}}).$$

- embed $\tilde{v} \in \mathbb{R}^r$ to $\mathbb{R}^d$ by $v = S^T \tilde{v}$.

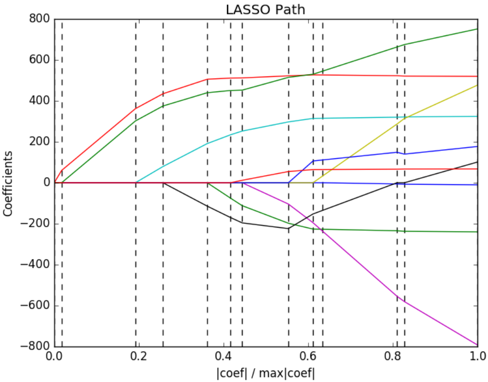
Algorithm Convex neural network training via randomized Geometric Algebra

Require: Number of hyperplane arrangement samples k, regularization parameter $\beta > 0$, sketching matrix $S \in \mathbb{R}^{m \times d}$.

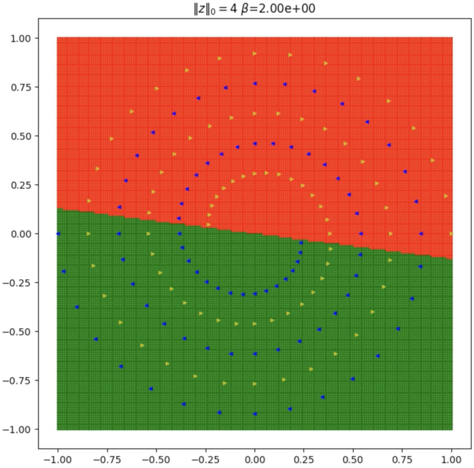
1: **for** $i = 1, \ldots, k$ **do**
2: Sample $\{j_i\}_{i=1}^{r-1}$ from $[n]$.
3: Compute $v_i = S^T \times (Sx_{j_1}, \ldots, Sx_{j_{r-1}})$.
4: Compute $\bar{D}_i = \mathrm{diag}(\mathbb{I}(Xv_i \geq 0))$.
5: **end for**
6: Solve the convex optimization problem with subsampled arrangements.

Lasso Path

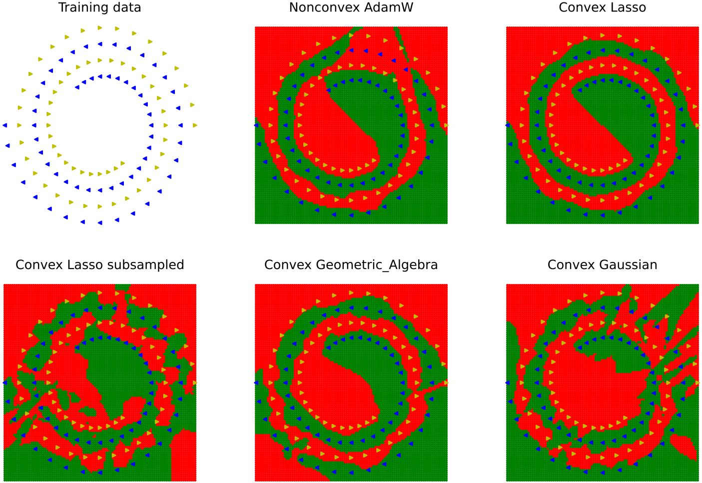
- we can find the full **Lasso path** as the regularization β changes
- produces a **path of neural networks** with varying number of neurons



Video Illustration

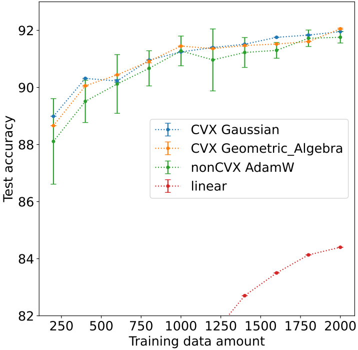


Spiral Dataset

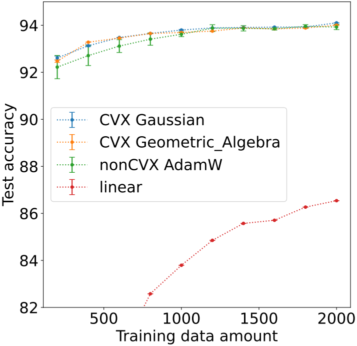


Sentiment Classification

- fine-tuning OpenAI GPT4 embeddings via two-layer ReLU networks

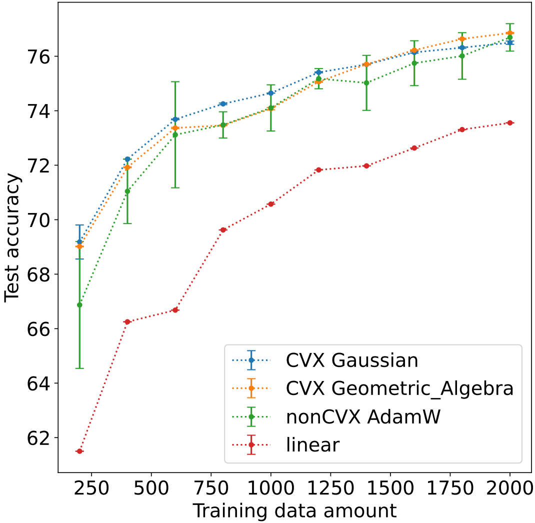


IMDB



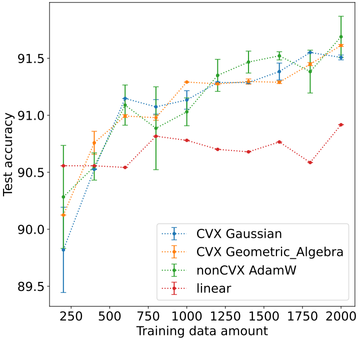
Amazon

Semantic Understanding

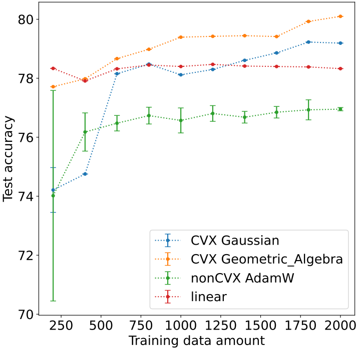


GLUE-QQP

ECG Classification

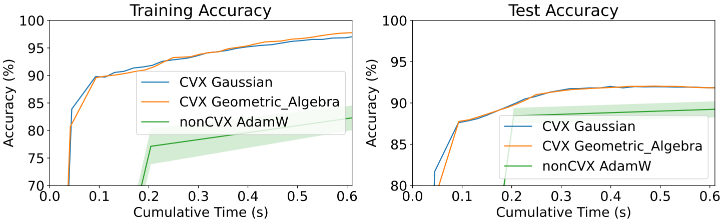


ECG-report

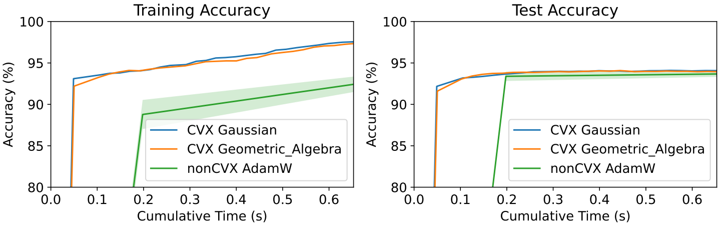


ECG-signal

Efficiency Comparison

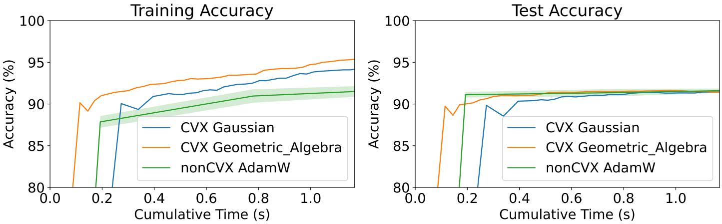


IMDB

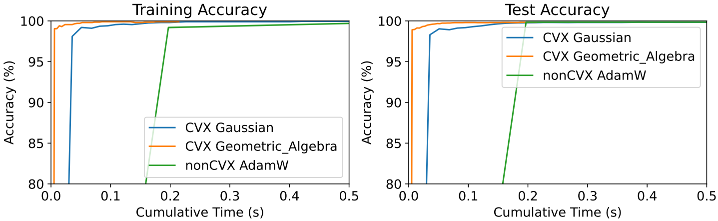


Amazon

Efficiency Comparison



ECG-report



MNIST

Question: Complexity Analysis

- Can we characterize the difficulty of solving the neural network training problem to global optimality?
- Can we find a polynomial-time algorithm to solve the neural network training problem?

Q & A: Complexity Analysis

- Can we characterize the difficulty of solving the neural network training problem to global optimality?
  - Yes, we can provide a negative result by relating the training problem to the NP-hard max-cut problem[1].
- Can we find a polynomial-time algorithm to solve the neural network training problem?
  - Yes, this is doable for structured datasets, for example, random Gaussian datasets[2],orthogonal separable datasets and datasets with negative correlation.

[1] Yifei Wang, Mert Pilanci, Polynomial-Time Solutions for ReLU Network Training: A Complexity Classification via Max-Cut and Zonotopes.

[2] Kim, Sungyoon, and Mert Pilanci. "Convex Relaxations of ReLU Neural Networks Approximate Global Optima in Polynomial Time. ICML 2024.

Complexity Upper Bound

- The complexity of solving the convex problem mainly depends on the number of hyperplane arrangement patterns.
- For $X \in \mathbb{R}^{N\times d}$, $p = \#\{\mathbf{1}(Xw \geqslant 0) | w \in \mathbb{R}^d\}$ is bounded by

$$p \leq 2r \left(\frac{e(N-1)}{r}\right)^r,$$

  where r is the rank of X.[3]
- For CNNs, the number of hyperplane arrangement patterns reduces to

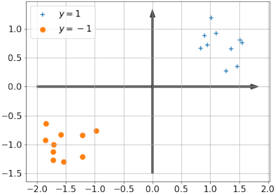
$$O(r^3(n/r)^{3r}),$$

  where r is the filter size, e.g., $r = 9$ for a 3×3 filter.

[1] Thomas M Cover. Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. IEEE transactions on electronic computers. 1965.

Hardness Characterization

- $(\mathbf{X}, \mathbf{y}) \in \mathbb{R}^{n\times d} \times \{-1, 1\}^n$ is orthogonally separable, i.e., for all $i, i' \in [n]$,
$$\mathbf{x}_i^T \mathbf{x}_{i'} > 0, \text{ if } y_i = y_{i'},$$
$$\mathbf{x}_i^T \mathbf{x}_{i'} \leq 0, \text{ if } y_i \neq y_{i'}.$$
- $(\mathbf{X}, \mathbf{y}) \in \mathbb{R}^{n\times d} \times \{-1, 1\}^n$ is negatively correlated, if $x_i^T x_{i'} \leq 0$ for all $y_i \neq y_{i'}$.



Positive Result for a Special Case: Orthogonal Separable

Theorem

Suppose that (X, y) is orthogonal separable, i.e.,

$$x_i^T x_j > 0, \text{ if } y_i = y_j, \quad x_i^T x_j \leq 0, \text{ if } y_i \neq y_j.$$

Then, for arbitrary $\epsilon > 0$, we can find a near-optimal neural network with ϵ multiplicative error in polynomial-time.

Positive Result for another Special Case: Negative Correlation

Theorem

Suppose that (X, y) *has negative correlation, i.e.,* $x_i x_j \leq 0$ *for* $y_i \neq y_j$*. For* $\epsilon = \sqrt{\pi/2} - 1$*, we can find a near-optimal neural network solution with* ϵ *multiplicative error in polynomial-time.*

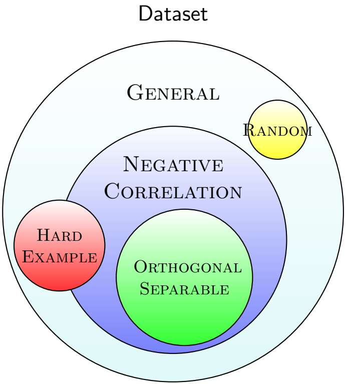
Negative Result

Theorem

Suppose that $P \neq NP$ *and set a multiplicative error* $\epsilon \leq \sqrt{84/83} - 1$. *Then, there does not exist a polynomial-time algorithm to find a solution with* ϵ *multiplicative error. This result holds for a generic loss* ℓ *for which the conjugate* $\ell^*(\lambda) = -g(-\lambda)$ *satisfies* $g(a\lambda) \geq ag(\lambda), \forall a > 1$.

- **To our knowledge, this is the first result that shows hardness of approximation for ReLU networks**
- **Proved by relating the convex NN problem to MaxCut**

Complexity Analysis



Polynomial-time algorithms to solve the NN problem

- approximation with multiplicative error in terms of a geometric ratio
- approximation with multiplicative error $\epsilon = \sqrt{\pi/2} - 1$
- exact optimal solution, i.e., $\epsilon = 0$
- impossible for approximation with multiplicative error $\epsilon \leq \sqrt{84/83} - 1$
- Randomized algorithms to solve the convex NN problem with $\epsilon = O(\sqrt{\log(n)})$

Figure: Difficulty of approximation of the ReLU neural network problem using a polynomial-time algorithm.

Takeaways

- The convex optimization formulation elucidates the optimization landscape, characterizes all global optima and Clarke stationary points, and decouples model performance from hyper-parameter choices
- Over-parameterized neural networks inherently learn simple models that effectively explain the data.
- The convex duality results extend to deep networks with parallel architecture.
- Geometric algebra provides an alternative perspective on the practical implementation of a convex neural network.

Journal Publications

1. **Yifei Wang**, Yixuan Hua, Emmanuel Candés, Mert Pilanci, Overparameterized ReLU Neural Networks Learn the Simplest Models: Neural Isometry and Exact Recovery, Transactions on Information Theory 2025.
2. **Yifei Wang**, Peng Chen, Mert Pilanci, Wuchen Li, Optimal Neural Network Approximation of Wasserstein Gradient Direction via Convex Optimization, SIAM Journal on Mathematics of Data Science 2024.
3. **Yifei Wang**, Mert Pilanci, Sketching the Krylov Subspace: Faster Computation of the Entire Ridge Regularization path, Springer Journal of Supercomputing 2023.
4. **Yifei Wang**, Kangkang Deng, Haoyang Li, Zaiwen Wen, A Decomposition Augmented Lagrangian Method for Low-rank Semidefinite Programming, SIAM Journal on Optimization 2023.

Conference Publications

5. Ertem Nusret Tas, David Tse, **Yifei Wang**, A Circuit Approach to Constructing Blockchains on Blockchains, Advances in Financial Technologies (AFT) 2024.
6. **Yifei Wang**, Tolga Ergen, Mert Pilanci, Parallel Deep Neural Networks Have Zero Duality Gap, International Conference on Learning Representations (ICLR) 2023 Poster.
7. **Yifei Wang**, Tavor Baharav, Yanjun Han, Jiantao Jiao, David Tse, Beyond the Best: Distribution Functional Estimation in Infinite-Armed Bandits, Conference on Neural Information Processing Systems (NeurIPS) 2022.
8. **Yifei Wang**, Jonathan Lacotte, Mert Pilanci, The Hidden Convex Optimization Landscape of Two-Layer ReLU Neural Networks, International Conference on Learning Representations (ICLR) 2022 Oral.
9. **Yifei Wang**, Mert Pilanci, The Convex Geometry of Backpropagation, International Conference on Learning Representations (ICLR) 2022 Poster.
10. Jonathan Lacotte, **Yifei Wang**, Mert Pilanci, Adaptive Newton Sketch: Linear-time Optimization with Quadratic Convergence, International Conference on Machine Learning (ICML) 2021 Poster.

Preprints and Other Works

11. **Yifei Wang**, Sungyoon Kim, Paul Chu, Indu Subramaniam, Mert Pilanci, Randomized Geometric Algebra Methods for Convex Neural Networks.
12. Emi Zeger, **Yifei Wang**, Aaron Mishkin, Tolga Ergen, Emmanuel Candes, Mert Pilanci, A Library of Mirrors: Deep Neural Nets in Low Dimensions are Convex Lasso Models with Reflection Features.
13. **Yifei Wang**, Mert Pilanci, Polynomial-Time Solutions for ReLU Network Training: A Complexity Classification via Max-Cut and Zonotopes.

---

14. **Yifei Wang**, Peng Chen, Wuchen Li, Projected Wasserstein gradient descent for high-dimensional Bayesian inference, SIAM Journal on Uncertainty Quantification 2022.
15. **Yifei Wang**, Wuchen Li, Accelerated Information Gradient flow, Springer Journal of Scientific Computing 2022.
16. **Yifei Wang**, Zeyu Jia, Zaiwen Wen, Search Direction Correction with Normalized Gradient Makes First-Order Methods Faster, SIAM Springer Journal on Scientific Computing 2021.