An Iterative Algorithm for Rescaled Hyperbolic Functions Regression

Yeqi Gao
The University of Washington

Zhao Song
Simons Institute for the Theory of Computing, UC Berkeley

Junze Yin
Rice University

Abstract

Large language models (LLMs) have numerous real-life applications across various domains, such as natural language translation, sentiment analysis, language modeling, chatbots and conversational agents, creative writing, text classification, summarization, and generation. LLMs have shown great promise in improving the accuracy and efficiency of these tasks, and have the potential to revolutionize the field of natural language processing (NLP) in the years to come. Exponential function based attention unit is a fundamental element in LLMs. Several previous works have studied the convergence of exponential regression and softmax regression.

In this paper, we propose an iterative algorithm to solve a rescaled version of the slightly different formulation of the softmax regression problem that arises in attention mechanisms of large language models. Specifically, we consider minimizing the squared loss between a certain function, which can be either the exponential function, hyperbolic sine function, or hyperbolic cosine function, and its inner product with a target n-dimensional vector b, scaled by the normalization term. This "rescaled softmax regression" differs from classical softmax regression in the location of the normalization factor.

The efficiency and generalizability of this framework to multiple hyperbolic functions make it relevant for optimizing attention mechanisms. The analysis also leads to a corollary bounding solution changes under small perturbations for in-context learning. Limitations and societal impact are discussed.

Proceedings of the 28[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2025, Mai Khao, Thailand. PMLR: Volume 258.

1 INTRODUCTION

The background of large language models (LLMs) can be traced back to a series of groundbreaking models, including the Transformer model (Vaswani et al., 2017), GPT-1 (Radford et al., 2018), BERT (Devlin et al., 2018), GPT-2 (Radford et al., 2019), and GPT-3 (Brown et al., 2020). These models are trained on massive amounts of textual data to generate natural language text and have shown their power on various real-world tasks, including natural language translation (He et al., 2021), sentiment analysis (Usama et al., 2020), language modeling (Martin et al., 2019), and even creative writing (OpenAI, 2023). The success of the new version of LLM named GPT-4 (OpenAI, 2023) has exemplified the use of LLMs in human-interaction tasks and suggests that LLMs are likely to continue to be a key area of research in the years to come.

LLMs rely heavily on attention computations to improve their performance in natural language processing tasks. The attention mechanism enables the model to selectively focus on specific parts of the input text (Vaswani et al., 2017; Devlin et al., 2018; Radford et al., 2019; Brown et al., 2020; Radford et al., 2018), enhancing its ability to identify and extract relevant information. A crucial component of the attention mechanism is the attention matrix, a square matrix in which each entry represents the correlations between words or tokens in the input text. The entries in the matrix are computed using a soft attention mechanism, which generates weights by applying a softmax function over the input sequence. Through this process, LLMs can identify and prioritize important parts of the input text, resulting in more accurate and efficient language processing.

Mathematically, one layer of forward computation is defined as follows:

Definition 1.1 (ℓ-th layer forward computation and attention optimization)**.** *Let n denote the length of the input token, and d denote the hidden dimension.*

For $\mathbf{1}_n$ being a vector whose entries are all 1's and dimension is n, diag *being a function mapping a vector*

in $\mathbb{R}^n$ to a matrix in $\mathbb{R}^{n\times n}$ (each of the entries of the vector in $\mathbb{R}^n$ is mapped to the diagonal entries of the $n\times n$ matrix), $Q, K, V \in \mathbb{R}^{d\times d}$ being the weights of the query, key, and value, respectively, $X_\ell \in \mathbb{R}^{n\times d}$ being the ℓ-th layer input, the ℓ-th layer forward computation is

$$X_{\ell+1} \leftarrow D^{-1}\exp(X_\ell Q K^\top X_\ell^\top) X_\ell V,$$

where $D := \mathrm{diag}(\exp(X_\ell Q K^\top X_\ell^\top)\mathbf{1}_n)$.

Therefore, the attention optimization is defined as

$$\min_{X,Y\in\mathbb{R}^{d\times d}} \|D(X)^{-1}\exp(A_1 X A_2^\top) A_3 Y - B\|_F^2, \quad (1)$$

where $\|\cdot\|_F$ is the Frobenius norm, $QK^\top$ is merged into $X \in \mathbb{R}^{d\times d}$, and $Y = V \in \mathbb{R}^{d\times d}$ are the weights which are interested to learn. $D(X) = \mathrm{diag}(\exp(A_1 X A_2^\top)\mathbf{1}_n) \in \mathbb{R}^{n\times n}$ and $A_1, A_2, A_3, B \in \mathbb{R}^{n\times d}$.

The attention mechanism has a computational complexity of $\widetilde{O}(n^2)$ with respect to the input sequence length n. The quadratic complexity of the attention computation makes it challenging for LLMs to efficiently process very long input sequences, which further limits the efficiency of training LLMs. Consequently, there has been growing interest in addressing the quadratic computational complexity by analyzing various regression problems derived from the attention computation (Definition 1.1). Several recent studies investigate the computation of the attention matrix in LLMs, including Zandieh et al. (2023); Brand et al. (2023); Deng et al. (2023b); Alman and Song (2023). Specifically, Zandieh et al. (2023); Brand et al. (2023); Alman and Song (2023) explore:

$$D^{-1}\exp(QK^\top)V,$$

where compared to Eq. (1), $A_1 X$ is merged into one matrix Q and $A_3 Y$ is merged into one matrix V. To get an almost linear time algorithm to approximate the attention optimization problem, Alman and Song (2023) relies on strict assumptions that $d = O(\log n)$ and all entries of Q, K, V are bounded by $o(\sqrt{\log n})$.

Deng et al. (2023b), on the other hand, studies

$$D^{-1}\exp(A_2 A_2^\top),$$

where $A_3 Y$ is not considered and only considers the symmetric matrix. Kacham et al. (2023) also replaces the softmax function exp in the attention mechanism with polynomials. While simplifying the attention optimization problem is acceptable and can reduce quadratic complexity to accelerate the training of LLMs, making too many modifications will inevitably have a negative impact on their performance (Dong et al., 2023). Thus, there is a trade-off between the efficiency of LLM training and its performance. It is natural to ask:

Is it possible to address quadratic computational complexity and accommodate more than the softmax unit with minimum simplifications to the attention optimization problem?

In this work, we provide a positive answer to this question: we focus on and develop the direction of regression tasks from Deng et al. (2023a); Li et al. (2023a), called the softmax regression

$$\min_{x\in\mathbb{R}^d} \|\langle\exp(Ax), \mathbf{1}_n\rangle^{-1}\exp(Ax) - b\|_2,$$

to define and analyze the following novel regression problem:

Definition 1.2 (Rescaled softmax/hyperbolic functions regression)**.** *Let $A \in \mathbb{R}^{n\times d}$ and $x \in \mathbb{R}^d$. Let $u(x)$ be applied entry-wise to the vector x and $u(x) \in \{\exp(Ax), \cosh(Ax), \sinh(Ax)\}$. Let $b \in \mathbb{R}^n$. The goal of the rescaled softmax/hyperbolic functions regression problem is to solve*

$$\min_{x\in\mathbb{R}^d} \|u(x) - \langle u(x), \mathbf{1}_n\rangle \cdot b\|_2,$$

where $\mathbf{1}_n$ is the n-dimensional vector whose entries are all 1.

Remark 1.3. *When $u(x) = \exp(Ax)$, this regression problem is called the rescaled softmax regression; when $u(x) = \cosh(Ax)$ or $u(x) = \sinh(Ax)$, this is called the rescaled hyperbolic functions regression. In this paper, for simplicity, we may use "rescaled softmax regression" or "rescaled hyperbolic functions regression" to denote both types of regression.*

Compared to Deng et al. (2023b); Zandieh et al. (2023); Brand et al. (2023), the softmax regression from Deng et al. (2023a) is the problem with the smallest change from the original attention optimization problem, where only $A_3 Y$ is not considered. For Deng et al. (2023b), $A_3 Y$ is not considered and only considered for the symmetric matrix, and for Zandieh et al. (2023); Brand et al. (2023), $A_1 X$ is merged into one matrix Q and $A_3 Y$ is merged into one matrix V. We minimize the simplifications of the attention computation (Definition 1.1) and design a sub-quadratic algorithm (Algorithm 1), which may lead to faster training in transformer models with minimum sacrifice in their performance.

Our contributions. Our contributions can be summarized as follows:

- The first contribution of this paper is defining and analyzing the rescaled version of the softmax

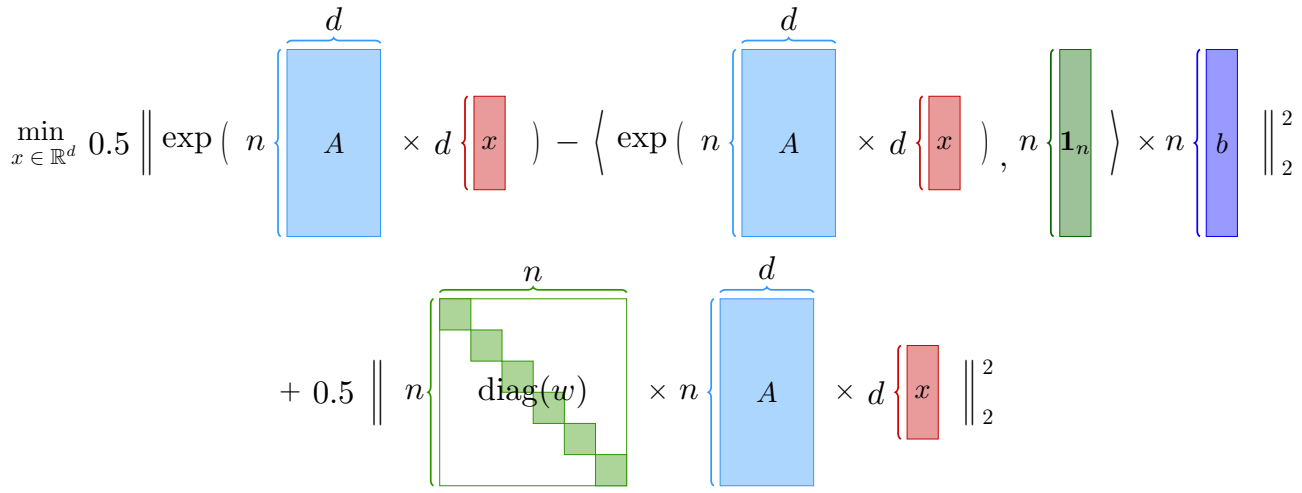

Figure 1: A visualization of the rescaled softmax regression problem with the regularization term (see Eq. (2)). We take $u(x) = \exp(Ax)$ as an example in this figure, and $u(x)$ can also be $\sinh(Ax)$ and $\cosh(Ax)$. Let $A \in \mathbb{R}^{n \times d}$, $x \in \mathbb{R}^d$, $b \in \mathbb{R}^n$, and $w \in \mathbb{R}^n$. We first get $\exp(Ax) \in \mathbb{R}^n$. Then, we find the inner product $\langle \exp(Ax), \mathbf{1}_n \rangle \in \mathbb{R}$ and multiplying it with $b \in \mathbb{R}^n$. Finding the ℓ_2 norm of the difference between $\exp(Ax) \in \mathbb{R}^n$ and $\langle \exp(Ax), \mathbf{1}_n \rangle b \in \mathbb{R}^n$ would be the top part of this figure. Regarding the bottom part, we first get $\mathrm{diag}(w) \in \mathbb{R}^{n \times n}$ and multiply it with $Ax \in \mathbb{R}^n$. Then, we find its ℓ_2 norm.

regression problem (Definition 1.2) and creating a randomized algorithm to solve it in subquadratic time of n (Theorem F.1 and Theorem 1.4).

- We remark the major difference between classical softmax regression and our new rescaled softmax regression (Definition 1.2) is the location of the normalization factor $\langle u(x), \mathbf{1}_n \rangle$. Due to the difference, the analysis for rescaled softmax regression will be quite different. This is the second contribution of this work.
- The third contribution of this paper is that our framework is very general and can handle several hyperbolic functions at the same time, including exp, cosh, and sinh, which is comparable to Kacham et al. (2023) that handles polynomial function.

1.1 Our Results

Note that we follow the assumption that $\|b\|_2 \leq R$ as in Li et al. (2023e). The reason why Deng et al. (2023a) can assume that $\|b\|_2 \leq 1$ is because they solve a normalized version. Therefore, in our re-scaled version, we only assume $\|b\|_2 \leq R$. Moreover, inspired by the empirical success of using weight decay in training transformers as explained in Li et al. (2023b), we explore a regularized version of Definition 1.2, namely

$$\min_{x \in \mathbb{R}^d} 0.5 \cdot \|u(x) - \langle u(x), \mathbf{1}_n \rangle \cdot b\|_2^2 + 0.5 \cdot \|\mathrm{diag}(w) A x\|_2^2, \tag{2}$$

where $\mathbf{1}_n, u(x), b \in \mathbb{R}^n$, $x \in \mathbb{R}^d$, and $A \in \mathbb{R}^{n \times d}$ are defined as in Definition 1.2. Also, $w \in \mathbb{R}^n$ and $\mathrm{diag}(w) \in \mathbb{R}^{n \times n}$ is a diagonal matrix that moves the entries of w to the diagonal entries of $\mathrm{diag}(w)$.

The informal version of our main result is presented as follows:

Theorem 1.4 (Main Result, Informal version of Theorem F.1)**.** *Let $\epsilon, \delta \in (0, 0.1)$ be the accuracy parameter and the failure probability, respectively.*

Let $x_0, x^* \in \mathbb{R}^d$ denote the initial point and the optimal solution respectively, $\mathrm{nnz}(A)$ denote the number of non-zero entries of A, and $\omega \approx 2.37$ be the exponent of matrix multiplication (Williams, 2012; Le Gall, 2014; Alman and Williams, 2021; Duan et al., 2023; Le Gall, 2023; Williams et al., 2023).

Then, there exists a randomized algorithm (Algorithm 1) solving Eq. (2) such that, with probability at least $1 - \delta$, runs $T = \log(\|x_0 - x^*\|_2 / \epsilon)$ iterations, spends

$$O((\mathrm{nnz}(A) + d^\omega) \cdot \mathrm{poly}(\log(n/\delta))$$

time in each iteration, and outputs a vector $\widetilde{x} \in \mathbb{R}^d$ such that

$$\|\widetilde{x} - x^*\|_2 \leq \epsilon.$$

Roadmap. Our paper is organized as follows. In Section 2, we discuss related work. In Section 3, we introduce several basic mathematical notations that we use in this paper. In Section 4, we provide a technique

overview. In Section 5, we present several properties of Hessian of loss functions. In Section 6, we present an analysis of our regression algorithm. In Section 7, we provide a conclusion.

2 RELATED WORK

Optimization and Convergence Studies in the field of optimization have investigated diverse facets of optimization methods and their applications. Snell et al. (2021) investigated the behavior of the mechanism of single-head attention for Seq2Seq model learning, providing insights into how to choose parameters for better performance. Zhang et al. (2020) emphasized the importance of adaptive methods for attention models and proposed a new adaptive method for the attention mechanism. Gao et al. (2023a) studied the convergence of over-parameterized neural networks with exponential activation functions, addressing the over-parametrization problem. Li et al. (2023e) proposed an algorithm for regularized exponential regression that runs in input sparsity time and demonstrated its effectiveness on various datasets. Finally, Li et al. (2023b) provided a thorough clarification of how transformers can learn the "semantic structure" to detect the patterns of word co-occurrence, exploring the optimization techniques used in transformers and highlighting their strengths and weaknesses.

Learning in-context Research on in-context learners based on transformers has been exploring various aspects of their abilities and mechanisms. As an example, Akyürek et al. (2022) showed that these learners can implicitly perform traditional learning algorithms through updating them continuously with new examples and encoding smaller models within their activations. Another work by Garg et al. (2022) focused on training a model that is under the in-context conditions which are used for learning a class of functions, like the linear functions, aiming to determine whether or not a model that has been given information obtained from specific functions within a class can learn the "majority" of functions in this class through training. In their research, Oswald et al. (2022) described how Transformers operate as in-context learners and revealed similarities between a few meta-learning formulations, which are based on gradient descent, and the training process of Transformers in in-context tasks. In general, these studies provide valuable insights into the abilities and mechanisms of in-context learners based on transformers, which possess the huge potential to significantly improve the applications of machine learning. Li et al. (2023a) proved a theoretical result about the in-context learning under softmax regression formulation (Deng et al., 2023a).

Fast Attention Computation The computation of attention has been explored in several works, with a focus on both dynamic and static attention. Brand et al. (2023) investigated the dynamic version of attention computation, where the input data is very dynamic and subject to constant changes, showing both positive results and negative results. They utilized lazy update techniques in their algorithmic results while the hardness result was based on the Hinted MV conjecture. On the other hand, Zandieh et al. (2023) and Alman and Song (2023) focused on static attention computation. Alman and Song (2023) proposed an algorithm for static attention and provided a hardness result based on the exponential time hypothesis. Meanwhile, Zandieh et al. (2023) explored the efficiency of static attention algorithms in various applications. Deng et al. (2023b) investigated the sparsification of feature dimension in attention computation, providing both randomized and deterministic algorithms. Song et al. (2024) studies the attention kernel regression problem, which utilizes the mathematical induction to generalize the algorithms of solving regression problems $\min_{x\in\mathbb{R}^d} \|AA^\top Ax - y\|_2^2$ and $\min_{x\in\mathbb{R}^d} \|A^\top AA^\top Ax - y\|_2^2$ to $\min_{x\in\mathbb{R}^d} \|A(A^\top A)^j x - b\|_2$ and $\min_{x\in\mathbb{R}^d} \|(A^\top A)^j x - b\|_2$ respectively, where j is any arbitrary positive integer. Song et al. (2023b) provides an algorithm to solve the exact attention regression problem by using the tensor and support vector machine tricks. Moreover, Song et al. (2023c) analyzes the polynomial based attention problem, where the $\exp(x)$ function from Eq. (1) is replaced by the x^β function, where $\beta \geq 2$. Furthermore, Song et al. (2023a) combines the softmax regression analyzed in Deng et al. (2023a) and the residual neural network developed in He et al. (2016) to study a unified regression problem. Li et al. (2023d) proposes a two-layer regression problem, where the inner layer is the ReLU function and the outer layer is the softmax regression studied in Deng et al. (2023a). Finally, Liang et al. (2024a) studies the masked version of the attention computation showing that any lower triangular matrices can be decomposed into the convolution basis.

3 PRELIMINARIES

In this section, we first introduce basic notations. Then, in Section 3.1, we define several functions that we use in later sections; in Section 3.2, we present a basic mathematical fact.

Notation We use $\mathbb{Z}_+$ to represent a set that contains all positive integers, and we use n to be an arbitrary element in $\mathbb{Z}_+$. We define $[n]$ to be the set, i.e., $[n] := \{1, 2, \ldots, n\}$.

Let $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^n$ be two vectors. For any

$i \in [n]$, we let $x_i \in \mathbb{R}$ denote the i -th entry of x. We use $x \circ y \in \mathbb{R}^n$ to represent the vector satisfying $(x \circ y)_i = x_i y_i$ for each $i \in [n]$. We use $\|x\|_p$ (where $p \in \{1, 2, \infty\}$) to represent the ℓ_p norm of x, where $\|x\|_1 := \sum_{i=1}^n |x_i|$ (ℓ_1 norm), $\|x\|_2 := (\sum_{i=1}^n x_i^2)^{1/2}$ (ℓ_2 norm), and $\|x\|_\infty := \max_{i \in [n]} |x_i|$ (ℓ_∞ norm). For a scalar $z \in \mathbb{R}$, we let $\exp(z)$ represent the standard exponential function.

Note that $\cosh(z) = \frac{1}{2}(\exp(z)+\exp(-z))$ and $\sinh(z) = \frac{1}{2}(\exp(z) - \exp(-z))$. Therefore, by the definitions of $\exp(z)$, $\cosh(z)$, and $\sinh(z)$, we have $\exp(z)' = \exp(z), \cosh(z)' = \sinh(z), \sinh(z)' = \cosh(z)$ and

$$\begin{aligned} \exp(z)'' &= \exp(z), \\ \cosh(z)'' &= \cosh(z), \\ \sinh(z)'' &= \sinh(z). \end{aligned}$$

For an arbitrary vector $x \in \mathbb{R}^n$, we use $\exp(x) \in \mathbb{R}^n$ to denote a vector whose i-th entry $\exp(x)_i$ is $\exp(x_i)$. We use $\langle x, y \rangle$ to denote $\sum_{i=1}^n x_i y_i$.

$\mathbf{1}_n$ represents a n-dimensional vector whose entries are all 1, and $\mathbf{0}_n$ represents a n-dimensional vector whose entries are all 0. We use I_n to denote an identity matrix that has size $n \times n$ and all the diagonal are ones.

For an arbitrary vector $u \in \mathbb{R}^n$, let $\mathrm{diag}(u) \in \mathbb{R}^{n \times n}$ represent a diagonal matrix whose i-th entry on the diagonal is u_i. For an arbitrary symmetric matrix $B \in \mathbb{R}^{n \times n}$, we say B is positive definite ($B \succ 0$) if for all vectors $x \in \mathbb{R}^n \backslash \{\mathbf{0}_n\}$, $x^\top B x > 0$.

For a symmetric matrix $B \in \mathbb{R}^{n \times n}$, we say B is positive semidefinite ($B \succeq 0$) if for all vectors $x \in \mathbb{R}^n$, $x^\top B x \geq 0$. For symmetric matrices B and C, we say $B \succeq C$ if for all x, $x^\top B x \geq x^\top C x$. For any matrix A, we use $\|A\|$ to denote the spectral norm of A, i.e., $\|A\| = \max_{\|x\|_2 = 1} \|Ax\|_2$. For each $i \in [d]$, we use $A_{*,i} \in \mathbb{R}^n$ to denote the i-th column of matrix $A \in \mathbb{R}^{n \times d}$.

3.1 General Functions: Definitions

In this section, we present the definitions of the basic functions appearing in our loss function.

Definition 3.1. *Let $u(x)$ be one of the following*

- *Case 1.* $u(x) = \exp(Ax)$
- *Case 2.* $u(x) = \cosh(Ax)$
- *Case 3.* $u(x) = \sinh(Ax)$

We define a helpful function as follows.

Definition 3.2. *Let $v(x)$ be one of the following*

- *Case 1.* $v(x) = \exp(Ax)$ *(when $u(x) = \exp(Ax)$)*
- *Case 2.* $v(x) = \sinh(Ax)$ *(when $u(x) = \cosh(Ax)$)*
- *Case 3.* $v(x) = \cosh(Ax)$ *(when $u(x) = \sinh(Ax)$)*

In the above two definitions, we introduce two basic notations $u(x)$ and $v(x)$. Those two notations are utilized in various locations, especially when we compute first derivatives and second derivatives. Note that $x \in \mathbb{R}^d$ is a vector. Therefore, we expect to use $v(x)$ to express a certain part of the derivative of $u(x)$ to simplify our mathematical expression.

We define L_u in the following sense:

Definition 3.3 (Loss function L_u)**.** *Given a matrix $A \in \mathbb{R}^{n \times d}$ and a vector $b \in \mathbb{R}^n$. We define loss function $L_u : \mathbb{R}^d \to \mathbb{R}$ as*

$$L_u(x) := 0.5 \cdot \|u(x) - \langle u(x), \mathbf{1}_n \rangle \cdot b\|_2^2.$$

For convenience, we define two helpful functions α and c:

Definition 3.4 (Rescaled coefficients)**.** *Given a matrix $A \in \mathbb{R}^{n \times d}$, we define $\alpha : \mathbb{R}^d \to \mathbb{R}$ as $\alpha(x) := \langle u(x), \mathbf{1}_n \rangle$. Then, the $L_u(x)$ (see Definition 3.3) can be rewritten as*

$$L_u(x) = 0.5 \cdot \|u(x) - b \cdot \alpha(x)\|_2^2.$$

Definition 3.5. *Given a matrix $A \in \mathbb{R}^{n \times d}$ and a vector $b \in \mathbb{R}^n$. Let $\alpha(x)$ be defined as in Definition 3.4. We define function $c : \mathbb{R}^d \to \mathbb{R}^n$ as follows $c(x) := u(x) - b \cdot \alpha(x)$.*

Then, we can rewrite $L_u(x)$ (see Definition 3.3) as $L_u(x) = 0.5 \cdot \|c(x)\|_2^2$.

Now, we define the regularization function, L_{reg}.

Definition 3.6. *Given a matrix $A \in \mathbb{R}^{n \times d}$ and $W = \mathrm{diag}(w) \in \mathbb{R}^{n \times n}$ where $w \in \mathbb{R}^n$ is a vector, we define $L_{\mathrm{reg}} : \mathbb{R}^d \to \mathbb{R}$ as*

$$L_{\mathrm{reg}}(x) := 0.5 \|WAx\|_2^2.$$

3.2 A Basic Mathematical Property

In this section, we present a basic mathematical property that is useful in later analysis. The following fact provides upper bounds on the norms of exponential, hyperbolic cosine, and hyperbolic sine functions and also establishes an approximation property when input values of these functions are close to each other.

Fact 3.7 (Informal version of Fact A.8)**.** *For vectors $a, b \in \mathbb{R}^n$, we have the following results:*

- $\|\exp(a)\|_\infty \leq \exp(\|a\|_2)$
- $\|\cosh(a)\|_\infty \leq \cosh(\|a\|_2) \leq \exp(\|a\|_2)$

- $\|\sinh(a)\|_\infty \leq \sinh(\|a\|_2) \leq \cosh(\|a\|_2) \leq \exp(\|a\|_2)$
- $\cosh(a) \circ \cosh(a) - \sinh(a) \circ \sinh(a) = \mathbf{1}_n$

Approximation in a small range: If two vectors $a, b \in \mathbb{R}^n$ are close, meaning $\|a-b\|_\infty \leq 0.01$, then, we can get

- $\|\exp(a) - \exp(b)\|_2 \leq \|\exp(a)\|_2 \cdot 2\|a-b\|_\infty$,
- $\|\cosh(a) - \cosh(b)\|_2 \leq \|\cosh(a)\|_2 \cdot 2\|a-b\|_\infty$, *and*
- $\|\sinh(a) - \sinh(b)\|_2 \leq \|\cosh(a)\|_2 \cdot 2\|a-b\|_\infty$.

This fact shows that the three distinct functions—exponential, hyperbolic cosine, and hyperbolic sine—actually share some similar mathematical properties.

4 TECHNIQUE OVERVIEW

An overview of our techniques is presented in this section.

General Functions For the purpose of applying our theory to exp, sinh, and cosh functions at the same time, we will introduce our generalized definition first. $u(x)$ is used to represent the functions including $\exp, \sinh$ and $\cosh$. With the aim that we can only use $u(x)$ in the following proof, the common property used in our proof of $u(x)$ will be proposed. To elaborate further, the expression for $u(x)$ is defined as Definition 3.1. Based on Fact 3.7 and $\|A\| \leq R$, we have

$$\|u(x)\|_2 \leq \sqrt{n} \exp(R^2)$$

And $v(x)$ is as defined as Definition 3.2. A unified version of the Hessian computation and the gradient computation can also be obtained as follows:

- $\frac{\mathrm{d}u(x)}{\mathrm{d}x} = (v(x)\mathbf{1}_d^\top) \circ A$
  - $\frac{\mathrm{d}u(x)}{\mathrm{d}x_i} = v(x) \circ A_{*,i}$ for each $i \in [d]$
- $\frac{\mathrm{d}^2 u(x)}{\mathrm{d}x_i^2} = A_{*,i} \circ u(x) \circ A_{*,i}$ for each $i \in [d]$
- $\frac{\mathrm{d}^2 u(x)}{\mathrm{d}x_i \mathrm{d}x_j} = A_{*,i} \circ u(x) \circ A_{*,j}$ for each $i, j \in [d] \times [d]$

Hessian Computation Taking $w \in \mathbb{R}^d$ into account as well, the target function we are focusing on is listed as follows:

$$\min_{x \in \mathbb{R}^d} L(x) = \min_{x \in \mathbb{R}^d} \underbrace{0.5 \cdot \|u(x) - \langle u(x), \mathbf{1}_n \rangle \cdot b\|_2^2}_{:=L_u} + \underbrace{0.5 \cdot \|\operatorname{diag}(w) A x\|_2^2}_{:=L_{\mathrm{reg}}}. \tag{3}$$

The computation of the Hessian for the problem directly is complex. We will introduce some techniques used in the computation of the Hessian function. To enhance the clarity of our expression, we draw a comparison between our Hessian Computation and the ones presented in Li et al. (2023e); Deng et al. (2023a). Specifically, we introduce the function $\alpha(x) := \langle u(x), \mathbf{1}_n \rangle$ and note that in Li et al. (2023e), there is no need to compute $\alpha(x)$, while $\alpha^{-1}(x)$ is the focus of Deng et al. (2023a). However, our emphasis is on the function $\alpha(x)$.

Additionally, with the definition $c(x) := u(x) - b \cdot \alpha(x)$, the computation of the Hessian can be divided into the $\frac{\mathrm{d}^2 u(x)}{\mathrm{d}x^2}$, $\frac{\mathrm{d}^2 \alpha(x)}{\mathrm{d}x^2}$ and $\frac{\mathrm{d}^2 c(x)}{\mathrm{d}x^2}$.

$\frac{\mathrm{d}^2 L}{\mathrm{d}x^2}$ is Positive Definite The first property we need to establish in order to apply the Newton optimization method is the positive definiteness of the Hessian. This is inspired by the semidefinite programming literature (Anstreicher, 2000; Huang et al., 2022). We have defined $R_0 := \max\{\|v(x)\|_2, \|b\|_2, \|c(x)\|_2, \|u(x)\|_2, 1\}$. Give that

$$\frac{\mathrm{d}^2 L(x)}{\mathrm{d}x_i^2} = \underbrace{A_{*,i}^\top B(x) A_{*,i}}_{\frac{\mathrm{d}^2 L_u(x)}{\mathrm{d}x_i^2}} + \underbrace{A_{*,i}^\top W^2 A_{*,i}}_{\frac{\mathrm{d}^2 L_{\mathrm{reg}}(x)}{\mathrm{d}x_i^2}}$$

and the bound on $B(x)$

$$-10R_0^4 \cdot I_n \preceq B(x) \preceq 10R_0^4 \cdot I_n,$$

by choosing $w_i \geq 10R_0^4 + l/\sigma_{\min}(A)^2$, the Hessian function is Positive definite now (see Section C for detail).

$\frac{\mathrm{d}^2 L}{\mathrm{d}x^2}$ is Lipschitz with respect to variable x To apply the Newton optimization method, it is also necessary to ensure the Lipschitz property. To finish the proof, we will get the upper bound of $\|H(x) - H(y)\|$ by $c \cdot \|x - y\|_2$ where c is a scalar. $H(x)$ can be decomposed into G_i where $i \in [n]$.

$$\|H(x) - H(y)\| \leq \|A\| \cdot (\sum_{i=1}^{5} \|G_i\|) \|A\|$$

The idea of how to bound each term G_i is quite standard neural network literature (for example, see Allen-Zhu et al. (2019b,a)).

With

$$R_\infty := \max\{\|u(x)\|_2, \|u(y)\|_2, \|v(x)\|_2, \|v(y)\|_2, \|c(x)\|_2, \|c(y)\|_2, \|b\|_2, 1\}$$

and then we get the following bound on $\|H(x)-H(y)\|$ by the following equation:

$$\sum_{i=1}^{5} \|G_i\| \leq 20R_\infty^3 \cdot \max\{\|u(x)-u(y)\|_2, \|c(x)-c(y)\|_2\}.$$

Furthermore, we can prove that the Hessian is Lipschitz continuous $\|H(x)-H(y)\| \leq n^4 \exp(20R^2) \cdot \|x-y\|_2$ (see details in Section D).

Approximated Newton Algorithm Based on the property of the Hessian function we have, we can apply the approximated Newton Method to the function regression problem. Building on the assumption of a (l, M)-good loss function, we can guarantee the correctness of our algorithm.

Given $M = n^4 \exp(20R^2)$, x_* as the optimization of Eq. (3) and x_0 as the initialization, we have a good initialization assumption

$$\underbrace{\|x_0 - x_*\|}_{:=r_0} M \leq 0.1l$$

To expedite the algorithm computation, it is natural to introduce a method for approximating the Hessian and its inverse (for example, see Cohen et al. (2019); Lee et al. (2019); Song (2019); Brand (2020); Jiang et al. (2021b); Song and Yu (2021); Huang et al. (2022); Gu and Song (2022); Deng et al. (2022); Song et al. (2022b); Jiang et al. (2023)). Given that $H(x_t)$ is a diagonal matrix, $\frac{\mathrm{d}^2 L}{\mathrm{d}x^2}$ can be transformed into a format $A^\top H(x_t) A$. With $\epsilon_0 \in (0, 0.1)$, an alternative way to obtain a sparse method is to substitute $H(x_t)$ with a sparse matrix $\widetilde{H}(x_t)$ where

$$(1-\epsilon_0) \cdot H(x_t) \preceq \widetilde{H}(x_t) \preceq (1+\epsilon_0) \cdot H(x_t)$$

The running time of Hessian computation can be reduced to $\widetilde{O}(\mathrm{nnz}(A) + d^\omega)$. To ensure the convergence of our algorithm, the number of iterations is expected to be $\log(1/\epsilon)$ based on the assumption above, leading to a total running time of

$$\widetilde{O}((\mathrm{nnz}(A) + d^\omega) \cdot \log(1/\epsilon).$$

Here $\mathrm{nnz}(A)$ denotes the number of nonzero entries in matrix A. Thus, we can derive our main result Theorem 1.4.

From Lipschitz with respect to x to Lipschitz with to A In Section D, we already proved a number of results for Lipschitz with respect to x. To present the application to in-context learning for rescaled softmax regression, we generalize the Lipschitz with respect x to Lipschitz with respect to A (see Setion E). To analyze the Lipschitz property, we bound $\|c(A)-c(B)\|_2$ using two terms $\|u(A)-u(B)\|_2$ and $|\alpha(A)-\alpha(B)|$.

Let $u(x)$ be in Definiton 3.1 and $u(A) = \exp(Ax)$, we have

$$\|u(A)-u(B)\|_2 \leq 2\sqrt{n} R \exp(R^2) \|A-B\|$$

We can also have

$$|\alpha(A)-\alpha(B)| \leq \sqrt{n} \cdot \|u(A)-u(B)\|_2$$

Then $\|c(A)-c(B)\|_2$ can be bounded as follows

$$\|c(A)-c(B)\|_2 \leq \|u(A)-u(B)\|_2 + |\alpha(A)-\alpha(B)| \cdot \|b\|_2.$$

The Lipschitz property of $c(A)$ with respect to A is guaranteed by $\|c(A)-c(B)\|_2 \leq C\|A-B\|$, where C is a scalar that can be determined as described above. Finally, we present the Corollary F.2 as our in-context learning application.

5 PROPERTIES OF HESSIAN

In this section, we introduce and analyze two crucial components of our main result (see Theorem F.1). In Section 5.1, we show that Hessian is a positive definite matrix. In Section 5.2, we analyze the Lipschitz property of Hessian. Both of the properties are the promise of correctness and efficiency of our Algorithm 1.

5.1 Hessian is Positive Definite

In this section, we present the result that Hessian is positive definite, which is the promise in computing $\widetilde{H}$ efficiently (see Lemma 6.4). Due to space limitation, we only present the informal Lemma statement here and defer the formal Lemma statement and the proof to Section C.3.

Lemma 5.1 (Informal version of Lemma C.4)**.** *Let $A \in \mathbb{R}^{n \times d}$, where $u(x)$ is defined according to Definition 3.1, and $v(x)$ follows Definition 3.2. Furthermore, $L_u(x)$ is defined as per Definition 3.3, and $L_{\mathrm{reg}}(x)$ corresponds to Definition 3.6. The combined loss function is denoted as*

$$L(x) = L_{\mathrm{reg}}(x) + L_u(x).$$

Given a vector $w \in \mathbb{R}^n$, the diagonal matrix $W = \mathrm{diag}(w) \in \mathbb{R}^{n \times n}$, and W^2 represents the matrix with w_i^2 as the i-th diagonal entry. Here, $\sigma_{\min}(A)$ denotes the minimum singular value of A, and $l > 0$ is a scalar. Let $R_0 = \max\{\|u(x)\|_2, \|v\|_2, \|b\|_2, \|c(x)\|_2, 1\}$. Suppose for all $i \in [n]$, $w_i^2 \geq 10R_0^4 + l/\sigma_{\min}(A)^2$.

Then we have

$$\frac{\mathrm{d}^2 L(x)}{\mathrm{d}x^2} \succeq l \cdot I_d.$$

5.2 Hessian is Lipschitz

In the following lemma, we show that the Hessian is Lipschitz, which is used to demonstrate that our loss function is (l, M)-good (see Definition 6.1). The proof is deferred to Section D.

Lemma 5.2 (Informal version of Lemma D.1)**.** *Let $H(x) = \frac{\mathrm{d}^2 L}{\mathrm{d}x^2}$ and $R > 4$. Let $\|x\|_2 \leq R$, $\|y\|_2 \leq R$, where $x, y \in \mathbb{R}^d$. Let $\|A(x-y)\|_\infty < 0.01$, where $A \in \mathbb{R}^{n \times d}$, $\|A\| \leq R$, $\|b\|_2 \leq R$, where $b \in \mathbb{R}^n$, and*

$$\begin{aligned} R_\infty := & \max\{\|u(x)\|_2, \|u(y)\|_2, \|c(x)\|_2, \|c(y)\|_2, 1\} \\ \leq & 2nR\exp(R^2). \end{aligned}$$

Then we have

$$\|H(x) - H(y)\| \leq n^4 \exp(20R^2) \cdot \|x - y\|_2$$

6 REGRESSION ALGORITHM

Algorithm 1 Rescaled Hyperbolic Functions Regression.

procedure RescaledHyperbolicFunctionsRegression($A \in \mathbb{R}^{n \times d}, b \in \mathbb{R}^n, w \in \mathbb{R}^n, \epsilon, \delta$) ▷ Theorem F.1
  We choose x_0 (suppose it satisfies Definition 6.1)
  We use $T \leftarrow \log(\|x_0 - x^*\|_2/\epsilon)$ to denote the number of iterations.
  for $t = 0 \to T$ **do**
    $D \leftarrow B_{\mathrm{diag}}(x_t) + \mathrm{diag}(w \circ w)$
    $\widetilde{D} \leftarrow$ SubSample($D, A, \epsilon_1 = \Theta(1), \delta_1 = \delta/T$) ▷ Lemma 6.4
    $g \leftarrow A^\top(c(x_t) \circ v(x_t) - v(x_t)\langle b, c(x_t)\rangle) + A^\top W^2 A x_t$ ▷ Definition 3.2 and Definition 3.5
    $\widetilde{H} \leftarrow A^\top \widetilde{D} A$
    $x_{t+1} \leftarrow x_t - \widetilde{H}^{-1} g$ ▷ Definition 6.3
  end for
  $\widetilde{x} \leftarrow x_{T+1}$
  return $\widetilde{x}$
end procedure

We provide an overview of our algorithm and its key components in this section. To help readers better understand our contribution and how it relates to the results in Section 5, we explain some of the key parts of our algorithm in this section. Given that $L(x) = L_u(x) + L_{\mathrm{reg}}(x)$, we consider the following optimization problem $\min_{x \in \mathbb{R}^d} L(x)$.

Specifically, in Section 6.1, we introduce the (l, M)-good Loss function. In Section 6.2, we present the approximate Hessian and update rule.

6.1 (l, M)-good Loss function

In this section, we explain the meaning of (l, M)-good loss function used in Lemma 6.5. Now, we provide the following definition:

Definition 6.1 ((l, M)-good Loss function)**.** *For a function $L : \mathbb{R}^d \to \mathbb{R}$, we say L is (l, M)-good if satisfies the following conditions,*

- ***l*-local Minimum.** *We define $l > 0$ to be a positive scalar. If there exists a vector $x^* \in \mathbb{R}^d$ such that $\nabla L(x^*) = \mathbf{0}_d$ and $\nabla^2 L(x^*) \succeq l \cdot I_d$.*
- **Hessian is *M*-Lipschitz.** *If there exists a positive scalar $M > 0$ such that $\|\nabla^2 L(y) - \nabla^2 L(x)\| \leq M \cdot \|y - x\|_2$*
- **Good Initialization Point.** *Let x_0 denote the initialization point. If $r_0 := \|x_0 - x_*\|_2$ satisfies $r_0 M \leq 0.1l$.*

Based on Lemma 5.2, our loss function (see Definition 1.2) satisfies the (l, M)-good assumption above. Now, we turn to two key steps in our main Algorithm 1: Line 8 and Line 9.

6.2 Approximate of Hessian and Update Rule

In this section, we present the concept of approximate update and its properties. The approximate update replaces the Hessian matrix $H(x_t) \in \mathbb{R}^{d \times d}$ in the well-known Newton method $x_{t+1} = x_t - H(x_t)^{-1} \cdot g(x_t)$ by the approximate Hessian $\widetilde{H}(x_t) \in \mathbb{R}^{d \times d}$, which is close to $H(x_t)$. The formal definition of the approximate Hessian is presented as follows:

Definition 6.2 (ϵ_0-approximate Hessian)**.** *Let $x \in \mathbb{R}^d$ and $H(x) \in \mathbb{R}^{d \times d}$ be a Hessian matrix. For all $\epsilon_0 \in (0, 0.1)$, we define an ϵ_0-approximate Hessian*[1] *$\widetilde{H}(x) \in \mathbb{R}^{d \times d}$ to be a matrix that satisfies:*

$$(1 - \epsilon_0) \cdot H(x) \preceq \widetilde{H}(x) \preceq (1 + \epsilon_0) \cdot H(x).$$

Using the definition of the approximate Hessian, we define the approximate Newton method as follows:

Definition 6.3 (ϵ_0-approximate update Newton's method (Deng et al., 2023a))**.** *Let $L : \mathbb{R}^d \to \mathbb{R}$ be a loss function. Suppose it has the gradient function $g : \mathbb{R}^d \to \mathbb{R}^d$ and the Hessian matrix $H : \mathbb{R}^d \to \mathbb{R}^{d \times d}$. Let $\widetilde{H} : \mathbb{R}^d \to \mathbb{R}^{d \times d}$ be an ϵ_0-approximate Hessian defined in Definition 6.2, for any $\epsilon_0 \in (0, 0.1)$.*

An ϵ_0-approximate update of Newton's method is a recurrence relation defined on L:

$$x_{t+1} = x_t - \widetilde{H}(x_t)^{-1} \cdot g(x_t).$$

[1] This approximate Hessian does not need to be a Hessian matrix. It is used to approximate the Hessian $H(x) \in \mathbb{R}^{d \times d}$.

In Line 8, we need to compute an approximated $\widetilde{H}$ (see Definition 6.2). In order to get the approximated Hessian $\widetilde{H}(x_t)$ efficiently, we present a standard tool that can be found in Lemma 4.5 of Deng et al. (2022).

Lemma 6.4 ((Deng et al., 2022; Song et al., 2022b)). *Let $\epsilon_0, \delta \in (0, 0.1)$ be the precision parameter and failure probability, respectively. Let $A \in \mathbb{R}^{n \times d}$.*

Then, for all $i \in [n]$, for all $D \in \mathbb{R}^{n \times n}$ satisfying $D_{i,i} > 0$, there exists an algorithm which runs in time

$$O((\mathrm{nnz}(A) + d^\omega) \operatorname{poly}(\log(n/\delta)))$$

and outputs an $O(d \log(n/\delta))$ sparse diagonal matrix $\widetilde{D} \in \mathbb{R}^{n \times n}$, i.e. a diagonal matrix where most of the entries are zeros, and the number of non-zero entries is less than or equal to a constant time $d \log(n/\delta)$, such that

$$(1 - \epsilon_0) A^\top D A \preceq A^\top \widetilde{D} A \preceq (1 + \epsilon_0) A^\top D A.$$

Here ω denotes exponent of matrix multiplication, currently $\omega \approx 2.373$ (Williams, 2012; Le Gall, 2014; Alman and Williams, 2021; Duan et al., 2023; Le Gall, 2023; Williams et al., 2023).

Given the importance of the approximated Hessian computation in the update step (see Line 9), we now focus on this particular step of Algorithm 1, where

$$x_{t+1} = x_t - \widetilde{H}(x_t)^{-1} \cdot g(x_t).$$

To establish the correctness of our algorithm, we leverage Lemma 6.9 of Li et al. (2023e):

Lemma 6.5 (Iterative shrinking, Lemma 6.9 on page 32 of Li et al. (2023e)). *For a positive integer t, we define $x_t \in \mathbb{R}^d$ to be the t-th iteration of the approximate Newton method (see Definition 6.3). We let $x^* \in \mathbb{R}^d$ be defined as in Definition 6.1, for fixed $A \in \mathbb{R}^{n \times d}$, $b \in \mathbb{R}^n$, and $w \in \mathbb{R}^n$. Let $L : \mathbb{R}^d \to \mathbb{R}$ be a loss function which is (l, M)-good (see Definition 6.1). Let $r_t := \|x_t - x^*\|_2$. Let $\overline{r}_t := M \cdot r_t$.*

Then, for all $\epsilon_0 \in (0, 0.1)$, we have

$$r_{t+1} \leq 2 \cdot (\epsilon_0 + \overline{r}_t / (l - \overline{r}_t)) \cdot r_t.$$

This lemma allows us to shrink the distance $\|x_t - x^*\|_2$ by one step using our assumption that the loss function is (l, M)-good, as verified in Section 5. To apply Lemma 6.5, we need the following induction hypothesis lemma. This is very standard in the literature, see Li et al. (2023e).

Lemma 6.6 (Induction hypothesis, Lemma 6.10 on page 34 of Li et al. (2023e)). *For a positive integer t, for each $i \in [t]$, we define $x_i \in \mathbb{R}^d$ to be the i-th iteration of the approximate Newton method (see Definition 6.3). We let $x^* \in \mathbb{R}^d$ be defined as in Definition 6.1, for fixed $A \in \mathbb{R}^{n \times d}$, $b \in \mathbb{R}^n$, and $w \in \mathbb{R}^n$.*

For each $i \in [t]$, we define $r_i := \|x_i - x^*\|_2$. Let $\epsilon_0 \in (0, 0.1)$. Suppose $r_i \leq 0.4 \cdot r_{i-1}$, for all $i \in [t]$. For M and l to be defined for Definition 6.1, we assume $M \cdot r_i \leq 0.1 l$, for all $i \in [t]$.

Then we have

- $r_{t+1} \leq 0.4 r_t$.
- $M \cdot r_{t+1} \leq 0.1 l$.

By applying this induction hypothesis and choosing a sufficiently large value of the number of iterations, we can then establish the correctness of our algorithm. The running time of our algorithm in each iteration is dominated by

$$O((\mathrm{nnz}(A) + d^\omega) \operatorname{poly}(\log(n/\delta))).$$

Because of the page limit, we delay our formal proof to Appendix F.

7 CONCLUSION

The exponential function-based attention unit is a crucial component in LLMs, enabling the model to selectively focus on different parts of the input sequence and improving its ability to capture long-range dependencies. In this paper, we focus on a slightly different version of softmax regression, namely

$$\min_{x \in \mathbb{R}^d} \|u(x) - \langle u(x), \mathbf{1}_n \rangle \cdot b\|_2.$$

We propose an efficient algorithm for this problem that operates on sparse inputs, leveraging the positive-definite and Lipschitz properties of the Hessian. Our mathematical analysis provides a deeper theoretical understanding of optimization problems related to the attention mechanism in LLMs. This could spur further advances and innovations in the architecture and training of language models.

Moreover, our algorithm framework is highly general and can be applied to a variety of functions, including $\exp(\cdot)$, $\cosh(\cdot)$, and $\sinh(\cdot)$.

Acknowledgments

This work was mostly done when Junze Yin was at Boston University. Junze Yin is supported by the Rice University graduate fellowship.

References

Nir Ailon and Bernard Chazelle. Approximate nearest neighbors and the fast johnson-lindenstrauss transform. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 557–563, 2006.

Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? investigations with linear models. *arXiv preprint arXiv:2211.15661*, 2022.

Maryam Aliakbarpour, Ilias Diakonikolas, and Ronitt Rubinfeld. Differentially private identity and equivalence testing of discrete distributions. In *International Conference on Machine Learning*, pages 169–178. PMLR, 2018.

Maryam Aliakbarpour, Ilias Diakonikolas, Daniel Kane, and Ronitt Rubinfeld. Private testing of distributions via sample permutations. *Advances in Neural Information Processing Systems*, 32, 2019.

Maryam Aliakbarpour, Rose Silver, Thomas Steinke, and Jonathan Ullman. Differentially private medians and interior points for non-pathological data. *arXiv preprint arXiv:2305.13440*, 2023.

Maryam Aliakbarpour, Konstantina Bairaktari, Adam Smith, Marika Swanberg, and Jonathan Ullman. Privacy in metalearning and multitask learning: Modeling and separations. *arXiv preprint arXiv:2412.12374*, 2024a.

Maryam Aliakbarpour, Syomantak Chaudhuri, Thomas A Courtade, Alireza Fallah, and Michael I Jordan. Enhancing feature-specific data protection via bayesian coordinate differential privacy. *arXiv preprint arXiv:2410.18404*, 2024b.

Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. A convergence theory for deep learning via over-parameterization. In *International Conference on Machine Learning*, pages 242–252. PMLR, 2019a.

Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. On the convergence rate of training recurrent neural networks. *Advances in neural information processing systems*, 32, 2019b.

Josh Alman and Zhao Song. Fast attention requires bounded entries. *arXiv preprint arXiv:2302.13214*, 2023.

Josh Alman and Virginia Vassilevska Williams. A refined laser method and faster matrix multiplication. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 522–539. SIAM, 2021.

Josh Alman, Jiehao Liang, Zhao Song, Ruizhe Zhang, and Danyang Zhuo. Bypass exponential time preprocessing: Fast neural network training via weight-data correlation preprocessing. *arXiv preprint arXiv:2211.14227*, 2022.

Kurt M Anstreicher. The volumetric barrier for semidefinite programming. *Mathematics of Operations Research*, 2000.

Shahab Asoodeh, Maryam Aliakbarpour, and Flavio P Calmon. Local differential privacy is equivalent to contraction of an f-divergence. In *2021 IEEE International Symposium on Information Theory (ISIT)*, pages 545–550. IEEE, 2021.

Haim Avron, Huy Nguyen, and David Woodruff. Subspace embeddings for the polynomial kernel. *Advances in neural information processing systems*, 27, 2014.

Song Bian, Zhao Song, and Junze Yin. Federated empirical risk minimization via second-order method. *arXiv preprint arXiv:2305.17482*, 2023.

Jan van den Brand. A deterministic linear program solver in current matrix multiplication time. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 259–278. SIAM, 2020.

Jan van den Brand, Zhao Song, and Tianyi Zhou. Algorithm and hardness for dynamic attention maintenance in large language models. *arXiv preprint arXiv:2304.02207*, 2023.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Yang Cao, Bo Chen, Xiaoyu Li, Yingyu Liang, Zhizhou Sha, Zhenmei Shi, Zhao Song, and Mingda Wan. Force matching with relativistic constraints: A physics-inspired approach to stable and efficient generative modeling. *arXiv preprint arXiv:2502.08150*, 2025a.

Yuefan Cao, Xiaoyu Li, Yingyu Liang, Zhizhou Sha, Zhenmei Shi, Zhao Song, and Jiahao Zhang. Dissecting submission limit in desk-rejections: A mathematical analysis of fairness in ai conference policies. *arXiv preprint arXiv:2502.00690*, 2025b.

Ya-Ting Chang, Zhibo Hu, Xiaoyu Li, Shuiqiao Yang, Jiaojiao Jiang, and Nan Sun. Dihan: A novel dynamic hierarchical graph attention network for fake news detection. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 197–206, 2024.

Bo Chen, Xiaoyu Li, Yingyu Liang, Jiangxuan Long, Zhenmei Shi, and Zhao Song. Circuit complexity bounds for rope-based transformer architecture. *arXiv preprint arXiv:2411.07602*, 2024a.

Bo Chen, Xiaoyu Li, Yingyu Liang, Zhenmei Shi, and Zhao Song. Bypassing the exponential dependency: Looped transformers efficiently learn in-context by multi-step gradient descent. In *International Conference on Artificial Intelligence and Statistics*, 2025a.

Bo Chen, Yingyu Liang, Zhizhou Sha, Zhenmei Shi, and Zhao Song. Hsr-enhanced sparse attention acceleration. In *Conference on Parsimony and Learning*. PMLR, 2025b.

Yifang Chen, Jiayan Huo, Xiaoyu Li, Yingyu Liang, Zhenmei Shi, and Zhao Song. Fast gradient computation for rope attention in almost linear time. *arXiv preprint arXiv:2412.17316*, 2024b.

Yifang Chen, Xiaoyu Li, Yingyu Liang, Zhenmei Shi, and Zhao Song. The computational limits of state-space models and mamba via the lens of circuit complexity. *arXiv preprint arXiv:2412.06148*, 2024c.

Yifang Chen, Xiaoyu Li, Yingyu Liang, Zhenmei Shi, and Zhao Song. Universal approximation of visual autoregressive transformers. *arXiv preprint arXiv:2502.06167*, 2025c.

Kenneth L Clarkson and David P Woodruff. Low-rank approximation and regression in input sparsity time. *Journal of the ACM (JACM)*, 63(6):1–45, 2017.

Michael B Cohen, Yin Tat Lee, and Zhao Song. Solving linear programs in the current matrix multiplication time. In *STOC*, 2019.

Yichuan Deng, Zhao Song, and Omri Weinstein. Discrepancy minimization in input-sparsity time. *arXiv preprint arXiv:2210.12468*, 2022.

Yichuan Deng, Zhihang Li, and Zhao Song. Attention scheme inspired softmax regression. *arXiv preprint arXiv:2304.10411*, 2023a.

Yichuan Deng, Sridhar Mahadevan, and Zhao Song. Randomized and deterministic attention sparsification algorithms for over-parameterized feature dimension. *arxiv preprint: arxiv 2304.03426*, 2023b.

Yichuan Deng, Zhao Song, and Junze Yin. Faster robust tensor power method for arbitrary order. *arXiv preprint arXiv:2306.00406*, 2023c.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Huaian Diao, Zhao Song, Wen Sun, and David Woodruff. Sketching for kronecker product regression and p-splines. In *International Conference on Artificial Intelligence and Statistics*, pages 1299–1308. PMLR, 2018.

Ye Dong, Wen-jie Lu, Yancheng Zheng, Haoqi Wu, Derun Zhao, Jin Tan, Zhicong Huang, Cheng Hong, Tao Wei, and Wenguang Cheng. Puma: Secure inference of llama-7b in five minutes. *arXiv preprint arXiv:2307.12533*, 2023.

Ran Duan, Hongxun Wu, and Renfei Zhou. Faster matrix multiplication via asymmetric hashing. In *FOCS*, 2023.

Yeqi Gao, Sridhar Mahadevan, and Zhao Song. An over-parameterized exponential regression. *arXiv preprint arXiv:2303.16504*, 2023a.

Yeqi Gao, Zhao Song, Weixin Wang, and Junze Yin. A fast optimization view: Reformulating single layer attention in llm based on tensor and svm trick, and solving it in matrix multiplication time. *arXiv preprint arXiv:2309.07418*, 2023b.

Yeqi Gao, Zhao Song, and Junze Yin. Gradientcoin: A peer-to-peer decentralized large language models. *arXiv preprint arXiv:2308.10502*, 2023c.

Shivam Garg, Dimitris Tsipras, Percy Liang, and Gregory Valiant. What can transformers learn in-context? a case study of simple function classes. *arXiv preprint arXiv:2208.01066*, 2022.

Yuzhou Gu and Zhao Song. A faster small treewidth sdp solver. *arXiv preprint arXiv:2211.06033*, 2022.

Yuzhou Gu, Zhao Song, Junze Yin, and Lichen Zhang. Low rank matrix completion via robust alternating minimization in nearly linear time. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=N0gT4A0jNV.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Weihua He, Yongyun Wu, and Xiaohua Li. Attention mechanism for neural machine translation: A survey. In *2021 IEEE 5th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, volume 5, pages 1485–1489. IEEE, 2021.

Baihe Huang, Shunhua Jiang, Zhao Song, Runzhou Tao, and Ruizhe Zhang. Solving sdp faster: A robust ipm framework and efficient implementation. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 233–244. IEEE, 2022.

Haotian Jiang, Yin Tat Lee, Zhao Song, and Lichen Zhang. Convex minimization with integer minima in $\widetilde{O}(n^4)$ time. *arXiv preprint arXiv:2304.03426*, 2023.

Shuli Jiang, Dennis Li, Irene Mengze Li, Arvind V Mahankali, and David Woodruff. Streaming and distributed algorithms for robust column subset selection. In *International Conference on Machine Learning*, pages 4971–4981. PMLR, 2021a.

Shunhua Jiang, Zhao Song, Omri Weinstein, and Hengjie Zhang. Faster dynamic matrix inverse for faster lps. In *STOC*, 2021b.

William B Johnson and Joram Lindenstrauss. Extensions of lipschitz mappings into a hilbert space. *Contemporary mathematics*, 26(189-206):1, 1984.

Praneeth Kacham, Vahab Mirrokni, and Peilin Zhong. Polysketchformer: Fast transformers via sketches for polynomial kernels. *arXiv preprint arXiv:2310.01655*, 2023.

Yekun Ke, Xiaoyu Li, Yingyu Liang, Zhenmei Shi, and Zhao Song. Advancing the understanding of fixed point iterations in deep neural networks: A detailed analytical study. *arXiv preprint arXiv:2410.11279*, 2024.

Yekun Ke, Yingyu Liang, Zhenmei Shi, Zhao Song, and Chiwun Yang. Curse of attention: A kernel-based perspective for why transformers fail to generalize on time series forecasting and beyond. In *Conference on Parsimony and Learning*. PMLR, 2025.

Valero Laparra, Diego Marcos Gonzalez, Devis Tuia, and Gustau Camps-Valls. Large-scale random features for kernel regression. In *2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pages 17–20. IEEE, 2015.

François Le Gall. Powers of tensors and fast matrix multiplication. In *Proceedings of the 39th international symposium on symbolic and algebraic computation*, pages 296–303, 2014.

François Le Gall. Faster rectangular matrix multiplication by combination loss analysis. *arXiv preprint arXiv:2307.06535*, 2023.

Yin Tat Lee, Zhao Song, and Qiuyi Zhang. Solving empirical risk minimization in the current matrix multiplication time. In *Conference on Learning Theory (COLT)*, pages 2140–2157. PMLR, 2019.

Chenyang Li, Yingyu Liang, Zhenmei Shi, and Zhao Song. Exploring the frontiers of softmax: Provable optimization, applications in diffusion model, and beyond. *arXiv preprint arXiv:2405.03251*, 2024a.

Chenyang Li, Yingyu Liang, Zhenmei Shi, Zhao Song, and Tianyi Zhou. Fourier circuits in neural networks and transformers: A case study of modular arithmetic with multiple inputs. In *International Conference on Artificial Intelligence and Statistics*, 2025a.

Shuai Li, Zhao Song, Yu Xia, Tong Yu, and Tianyi Zhou. The closeness of in-context learning and weight shifting for softmax regression. *arXiv preprint arXiv:2304.13276*, 2023a.

Xiaoyu Li, Yingyu Liang, Zhenmei Shi, and Zhao Song. A tighter complexity analysis of sparseGPT. In *Workshop on Machine Learning and Compression, NeurIPS 2024*, 2024b.

Xiaoyu Li, Yingyu Liang, Zhenmei Shi, Zhao Song, and Mingda Wan. Theoretical constraints on the expressive power of rope-based tensor attention transformers. *arXiv preprint arXiv:2412.18040*, 2024c.

Xiaoyu Li, Yingyu Liang, Zhenmei Shi, Zhao Song, and Yufa Zhou. Fine-grained attention i/o complexity: Comprehensive analysis for backward passes. *arXiv preprint arXiv:2410.09397*, 2024d.

Xiaoyu Li, Yingyu Liang, Jiangxuan Long, Zhenmei Shi, Zhao Song, and Zhen Zhuang. Neural algorithmic reasoning for hypergraphs with looped transformers. *arXiv preprint arXiv:2501.10688*, 2025b.

Xiaoyu Li, Yingyu Liang, Zhenmei Shi, Zhao Song, Wei Wang, and Jiahao Zhang. On the computational capability of graph neural networks: A circuit complexity bound perspective. *arXiv preprint arXiv:2501.06444*, 2025c.

Xiaoyu Li, Yingyu Liang, Zhenmei Shi, Zhao Song, and Junwei Yu. Fast john ellipsoid computation with differential privacy optimization. In *Conference on Parsimony and Learning*. PMLR, 2025d.

Yuchen Li, Yuanzhi Li, and Andrej Risteski. How do transformers learn topic structure: Towards a mechanistic understanding. *arXiv preprint arXiv:2303.04245*, 2023b.

Yunfan Li and Lin Yang. On the model-misspecification in reinforcement learning. In *International Conference on Artificial Intelligence and Statistics*, pages 2764–2772. PMLR, 2024.

Yunfan Li, Yiran Wang, Yu Cheng, and Lin Yang. Low-switching policy gradient with exploration via online sensitivity sampling. In *International Conference on Machine Learning*, pages 19995–20034. PMLR, 2023c.

Zhihang Li, Zhao Song, Zifan Wang, and Junze Yin. Local convergence of approximate newton method for two layer nonlinear regression. *arXiv preprint arXiv:2311.15390*, 2023d.

Zhihang Li, Zhao Song, and Tianyi Zhou. Solving regularized exp, cosh and sinh regression problems. *arXiv preprint, 2303.15725*, 2023e.

Jiehao Liang, Somdeb Sarkhel, Zhao Song, Chenbo Yin, Junze Yin, and Danyang Zhuo. A faster k-means++ algorithm. *arXiv preprint arXiv:2211.15118*, 2022.

Yingyu Liang, Heshan Liu, Zhenmei Shi, Zhao Song, Zhuoyan Xu, and Junze Yin. Conv-basis: A new paradigm for efficient attention inference and gradient computation in transformers. *arXiv preprint arXiv:2405.05219*, 2024a.

Yingyu Liang, Zhizhou Sha, Zhenmei Shi, Zhao Song, and Yufa Zhou. Multi-layer transformers gradient can be approximated in almost linear time. *arXiv preprint arXiv:2408.13233*, 2024b.

Yingyu Liang, Zhenmei Shi, Zhao Song, and Chiwun Yang. Toward infinite-long prefix in transformer. *arXiv preprint arXiv:2406.14036*, 2024c.

Yingyu Liang, Zhenmei Shi, Zhao Song, and Yufa Zhou. Differential privacy of cross-attention with provable guarantee. *arXiv preprint arXiv:2407.14717*, 2024d.

Yingyu Liang, Zhenmei Shi, Zhao Song, and Yufa Zhou. Tensor attention training: Provably efficient learning of higher-order transformers. *arXiv preprint arXiv:2405.16411*, 2024e.

Yingyu Liang, Jiangxuan Long, Zhenmei Shi, Zhao Song, and Yufa Zhou. Beyond linear approximations: A novel pruning approach for attention matrix. In *International Conference on Learning Representations*, 2025.

Junyan Liu, Yunfan Li, Ruosong Wang, and Lin Yang. Uniform last-iterate guarantee for bandits and reinforcement learning. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

S Cliff Liu, Zhao Song, Hengjie Zhang, Lichen Zhang, and Tianyi Zhou. Space-efficient interior point method, with applications to linear programming and maximum weight bipartite matching. In *ICALP*, 2023.

Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suarez, Yoann Dupont, Laurent Romary, Eric Villemonte de La Clergerie, Djame Seddah, and Benoit Sagot. Camembert: a tasty french language model. *arXiv preprint arXiv:1911.03894*, 2019.

Jelani Nelson and Huy L Nguyên. Osnap: Faster numerical linear algebra algorithms via sparser subspace embeddings. In *2013 ieee 54th annual symposium on foundations of computer science*, pages 117–126. IEEE, 2013.

OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Johannes von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. Transformers learn in-context by gradient descent. *arXiv preprint arXiv:2212.07677*, 2022.

Rasmus Pagh. Compressed matrix multiplication. *ACM Transactions on Computation Theory (TOCT)*, 5(3):1–17, 2013.

Ninh Pham and Rasmus Pagh. Fast and scalable polynomial kernels via explicit feature maps. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 239–247, 2013.

Lianke Qin, Rajesh Jayaram, Elaine Shi, Zhao Song, Danyang Zhuo, and Shumo Chu. Adore: Differentially oblivious relational database operators. *arXiv preprint arXiv:2212.05176*, 2022.

Lianke Qin, Zhao Song, and Yuanyuan Yang. Efficient sgd neural network training via sublinear activated neuron identification. *arXiv preprint arXiv:2307.06565*, 2023a.

Lianke Qin, Zhao Song, Lichen Zhang, and Danyang Zhuo. An online and unified algorithm for projection matrix vector multiplication with application to empirical risk minimization. In *AISTATS*, 2023b.

Lianke Qin, Zhao Song, and Ruizhe Zhang. A general algorithm for solving rank-one matrix sensing. *arXiv preprint arXiv:2303.12298*, 2023c.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training, 2018.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Zhenmei Shi, Yifei Ming, Xuan-Phi Nguyen, Yingyu Liang, and Shafiq Joty. Discovering the gems in early layers: Accelerating long-context llms with 1000x input token reduction. *arXiv preprint arXiv:2409.17422*, 2024a.

Zhenmei Shi, Junyi Wei, Zhuoyan Xu, and Yingyu Liang. Why larger language models do in-context learning differently? *arXiv preprint arXiv:2405.19592*, 2024b.

Charlie Snell, Ruiqi Zhong, Dan Klein, and Jacob Steinhardt. Approximating how single head attention learns. *arXiv preprint arXiv:2103.07601*, 2021.

Aleksandros Sobczyk and Efstratios Gallopoulos. pylspack: Parallel algorithms and data structures for sketching, column subset selection, regression, and leverage scores. *ACM Transactions on Mathematical Software*, 48(4):1–27, 2022.

Zhao Song. *Matrix theory: optimization, concentration, and algorithms*. The University of Texas at Austin, 2019.

Zhao Song and Zheng Yu. Oblivious sketching-based central path method for linear programming. In *International Conference on Machine Learning*, pages 9835–9847. PMLR, 2021.

Zhao Song, David Woodruff, and Peilin Zhong. Towards a zero-one law for column subset selection. *Advances in Neural Information Processing Systems*, 32, 2019.

Zhao Song, Lichen Zhang, and Ruizhe Zhang. Training multi-layer over-parametrized neural network in subquadratic time. *arXiv preprint arXiv:2112.07628*, 2021.

Zhao Song, Zhaozhuo Xu, Yuanyuan Yang, and Lichen Zhang. Accelerating frank-wolfe algorithm using low-dimensional and adaptive data structures. *arXiv preprint arXiv:2207.09002*, 2022a.

Zhao Song, Xin Yang, Yuanyuan Yang, and Tianyi Zhou. Faster algorithm for structured john ellipsoid computation. *arXiv preprint arXiv:2211.14407*, 2022b.

Zhao Song, Weixin Wang, and Junze Yin. A unified scheme of resnet and softmax. *arXiv preprint arXiv:2309.13482*, 2023a.

Zhao Song, Yitan Wang, Zheng Yu, and Lichen Zhang. Sketching for first order method: Efficient algorithm for low-bandwidth channel and vulnerability. In *ICML*, 2023b.

Zhao Song, Guangyi Xu, and Junze Yin. The expressibility of polynomial based attention scheme. *arXiv preprint arXiv:2310.20051*, 2023c.

Zhao Song, Xin Yang, Yuanyuan Yang, and Lichen Zhang. Sketching meets differential privacy: Fast algorithm for dynamic kronecker projection maintenance. In *ICML*, 2023d.

Zhao Song, Mingquan Ye, Junze Yin, and Lichen Zhang. A nearly-optimal bound for fast regression with ℓ_∞ guarantee. In *International Conference on Machine Learning (ICML)*, pages 32463–32482. PMLR, 2023e.

Zhao Song, Junze Yin, and Ruizhe Zhang. Revisiting quantum algorithms for linear regressions: Quadratic speedups without data-dependent parameters. *arXiv preprint arXiv:2311.14823*, 2023f.

Zhao Song, Junze Yin, and Lichen Zhang. Solving attention kernel regression problem via pre-conditioner. In *International Conference on Artificial Intelligence and Statistics*, pages 208–216. PMLR, 2024.

Zhao Song, Weixin Wang, Chenbo Yin, and Junze Yin. Fast and efficient matching algorithm with deadline instances. In *The Second Conference on Parsimony and Learning (Proceedings Track)*, 2025a. URL https://openreview.net/forum?id=TIneXGrWZt.

Zhao Song, Mingquan Ye, Junze Yin, and Lichen Zhang. Efficient alternating minimization with applications to weighted low rank approximation. In *The Thirteenth International Conference on Learning Representations*, 2025b. URL https://openreview.net/forum?id=rvhu4V7yrX.

Mohd Usama, Belal Ahmad, Enmin Song, M Shamim Hossain, Mubarak Alrashoud, and Ghulam Muhammad. Attention-based sentiment analysis using convolutional and recurrent neural network. *Future Generation Computer Systems*, 113:571–578, 2020.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Jiayu Wang, Yifei Ming, Zhenmei Shi, Vibhav Vineet, Xin Wang, Yixuan Li, and Neel Joshi. Is a picture worth a thousand words? delving into spatial reasoning for vision language models. *Advances in Neural Information Processing Systems*, 36, 2024.

Virginia Vassilevska Williams. Multiplying matrices faster than coppersmith-winograd. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 887–898, 2012.

Virginia Vassilevska Williams, Yinzhan Xu, Zixuan Xu, and Renfei Zhou. New bounds for matrix multiplication: from alpha to omega, 2023.

Zhaozhuo Xu, Zhao Song, and Anshumali Shrivastava. Breaking the linear iteration cost barrier for some well-known conditional gradient methods using maxip data-structures. *Advances in Neural Information Processing Systems*, 34:5576–5589, 2021.

Zhuoyan Xu, Zhenmei Shi, and Yingyu Liang. Do large language models have compositional ability? an investigation into limitations and scalability. In *Conference on Language Modeling*, 2024a.

Zhuoyan Xu, Zhenmei Shi, Junyi Wei, Fangzhou Mu, Yin Li, and Yingyu Liang. Towards few-shot adaptation of foundation models via multitask finetuning. In *International Conference on Learning Representations*, 2024b.

Amir Zandieh, Insu Han, Majid Daliri, and Amin Karbasi. Kdeformer: Accelerating transformers via kernel density estimation. *arXiv preprint arXiv:2302.02451*, 2023.

Haochen Zhang, Xi Chen, and Lin F Yang. Adaptive liquidity provision in uniswap v3 with deep reinforcement learning. *arXiv preprint arXiv:2309.10129*, 2023.

Haochen Zhang, Junze Yin, Guanchu Wang, Zirui Liu, Tianyi Zhang, Anshumali Shrivastava, Lin Yang, and Vladimir Braverman. I3s: Importance sampling subspace selection for low-rank optimization in llm pretraining. *arXiv preprint arXiv:2502.05790*, 2025.

Jiahao Zhang, Feng Liu, and Aimin Zhou. Off-tanet: A lightweight neural micro-expression recognizer with optical flow features and integrated attention mechanism. In *Pacific Rim International Conference on Artificial Intelligence*, pages 266–279. Springer, 2021.

Jingzhao Zhang, Sai Praneeth Karimireddy, Andreas Veit, Seungyeon Kim, Sashank Reddi, Sanjiv Kumar, and Suvrit Sra. Why are adaptive methods good for attention models? *Advances in Neural Information Processing Systems*, 33:15383–15393, 2020.

Lichen Zhang. Speeding up optimizations via data structures: Faster search, sample and maintenance. Master's thesis, Carnegie Mellon University, 2022.

Zhi Zhang, Chris Chow, Yasi Zhang, Yanchao Sun, Haochen Zhang, Eric Hanchen Jiang, Han Liu, Furong Huang, Yuchen Cui, and Oscar Hernan Madrid Padilla. Statistical guarantees for lifelong reinforcement learning using pac-bayesian theory. *arXiv preprint arXiv:2411.00401*, 2024.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Not Applicable]

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes]
   (b) Complete proofs of all theoretical results. [Yes]
   (c) Clear explanations of any assumptions. [Yes]

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Not Applicable]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Not Applicable]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Not Applicable]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Not Applicable]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Not Applicable]
   (b) The license information of the assets, if applicable. [Not Applicable]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Not Applicable]
   (d) Information about consent from data providers/curators. [Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

An Iterative Algorithm for Rescaled Hyperbolic Functions Regression: Supplementary Materials

Roadmap. We define the notations and propose approximate algebra, differential computation, and math tools for exact algebra used in our paper in Section A. In Section B, we introduce the computation of Hessian and Gradient. In Section C, we prove $L = L_u + L_{\mathrm{reg}}$ is convex function. The hessian of $L = L_u + L_{\mathrm{reg}}$ is proved to be Lipschitz in Section D. In Section E, we analyze the Lipschitz with respect to A, where $A \in \mathbb{R}^{n \times d}$. In Section F, we introduce our main result and algorithm.

A PRELIMINARIES

In Section A.1, we introduce several basic notations and mathematics symbols, which are used in this paper. In Section A.2, we present the algebraic properties for $\circ$ and diag. In Section A.3, the properties of the inner product are explained. In Section A.4, the properties of the $\preceq$ and its relationship with diag and $\circ$ are introduced. In Section A.5, we present several standard derivative rules, both for the scalar variables and for the vector variables. In Section A.6, we demonstrate the properties of the vector norm bound, including the Cauchy-Schwarz inequality and other inequalities of the bound containing $\circ$ and diag. In Section A.7, we illustrate the properties of the matrix norm bound, namely the inequalities of the spectral norms. In Section A.8, we introduce the properties of the hyperbolic functions, which take the scalar as an element of their domains. On the other hand, in Section A.9, we elucidate the properties of the hyperbolic functions, which take the vector as an element of their domains. In Section A.10, we define the regularization function, $L_{\mathrm{reg}} : \mathbb{R}^d \to \mathbb{R}$, and analyze its basic properties. In Section A.11, we define the gradient and Hessian of an arbitrary loss function L and define the update of the Newton method.

A.1 Notation

In this section, we explain the several basic notations. We use $\mathbb{Z}_+$ to represent a set that contains all the positive integers, and we use n to be an arbitrary element in $\mathbb{Z}_+$. We define $[n]$ to be the set, i.e., $[n] := \{1, 2, \ldots, n\}$. Let $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^n$ be two vectors. For any $i \in [n]$, we let $x_i \in \mathbb{R}$ to denote the i-th entry of x. We use $x \circ y \in \mathbb{R}^n$ to represent the vector satisfying $(x \circ y)_i = x_i y_i$ for each $i \in [n]$. We use $\|x\|_p$ (where $p \in \{1, 2, \infty\}$) to represent the ℓ_p norm of x, where $\|x\|_1 := \sum_{i=1}^n |x_i|$ (ℓ_1 norm), $\|x\|_2 := (\sum_{i=1}^n x_i^2)^{1/2}$ (ℓ_2 norm), and $\|x\|_\infty := \max_{i \in [n]} |x_i|$ (ℓ_∞ norm). For a scalar $z \in \mathbb{R}$, we let $\exp(z)$ represent the standard exponential function. We then define $\cosh(z) := \frac{1}{2}(\exp(z) + \exp(-z))$ and $\sinh(z) := \frac{1}{2}(\exp(z) - \exp(-z))$. Note that

$$\exp(z)' = \exp(z), \cosh(z)' = \sinh(z), \sinh(z)' = \cosh(z)$$

and

$$\exp(z)'' = \exp(z), \cosh(z)'' = \cosh(z), \sinh(z)'' = \sinh(z).$$

For an arbitrary vector $x \in \mathbb{R}^n$, we use $\exp(x) \in \mathbb{R}^n$ to denote a vector whose i-th entry $\exp(x)_i$ is $\exp(x_i)$. We use $\langle x, y \rangle$ to denote $\sum_{i=1}^n x_i y_i$. $\mathbf{1}_n$ represents a n-dimensional vector whose entries are all 1, and $\mathbf{0}_n$ represents a n-dimensional vector whose entries are all 0. For an arbitrary vector $u \in \mathbb{R}^n$, let $\mathrm{diag}(u) \in \mathbb{R}^{n \times n}$ represent a diagonal matrix whose i-th entry on diagonal is u_i. For an arbitrary symmetric matrix $B \in \mathbb{R}^{n \times n}$, we say B is positive definite ($B \succ 0$) if for all vectors $x \in \mathbb{R}^n \backslash \{\mathbf{0}_n\}$, $x^\top B x > 0$. For a symmetric matrix $B \in \mathbb{R}^{n \times n}$, we say B is positive semidefinite ($B \succeq 0$) if for all vectors $x \in \mathbb{R}^n$, $x^\top B x \geq 0$. For symmetric matrices B and C, we say $B \succeq C$ if for all x, $x^\top B x \geq x^\top C x$. For any matrix A, we use $\|A\|$ to denote the spectral norm of A, i.e., $\|A\| = \max_{\|x\|_2 = 1} \|Ax\|_2$. For each $i \in [d]$, we use $A_{*,i} \in \mathbb{R}^n$ to denote the i-th column of matrix $A \in \mathbb{R}^{n \times d}$. We use I_n to denote an identity matrix that has size $n \times n$ and all the diagonal are ones.

A.2 Basic Algebra for $\circ$ and diag

In this section, we provide a fact that includes the basic algebraic properties of $\circ$ and diag.

Fact A.1. *Given vectors $a \in \mathbb{R}^n$, $b \in \mathbb{R}^n$, $c \in \mathbb{R}^n$ and $d \in \mathbb{R}^n$, we have*

- $a \circ b = \mathrm{diag}(a) \cdot b = \mathrm{diag}(a) \cdot \mathrm{diag}(b) \cdot \mathbf{1}_n$
  - $a \circ b = b \circ a$
  - $\mathrm{diag}(a) b = \mathrm{diag}(b) a$
  - $\mathrm{diag}(a) \cdot \mathrm{diag}(b) \cdot \mathbf{1}_n = \mathrm{diag}(b) \cdot \mathrm{diag}(a) \cdot \mathbf{1}_n$
- $\mathrm{diag}(a \circ b) = \mathrm{diag}(a)\,\mathrm{diag}(b)$
- $\mathrm{diag}(a) + \mathrm{diag}(b) = \mathrm{diag}(a + b)$
- $a^\top (b \circ c) = a^\top \mathrm{diag}(b) c$
  - $a^\top (b \circ c) = b^\top (a \circ c) = c^\top (a \circ b)$
  - $a^\top \mathrm{diag}(b) c = b^\top \mathrm{diag}(a) c = a^\top \mathrm{diag}(c) b$
- $\langle a, b \circ c \rangle = a^\top \mathrm{diag}(b) c$

A.3 Basic Inner Product

Now, we present the inner product properties.

Fact A.2. *Given vectors $a \in \mathbb{R}^n$, $b \in \mathbb{R}^n$ and $c \in \mathbb{R}^n$, we have*

- $\langle a, b \rangle = \langle b, a \rangle$
- $\langle a \circ b, c \rangle = \langle a \circ b \circ c, \mathbf{1}_n \rangle$
- $\langle a, b \rangle = a^\top b = b^\top a$
- $\langle a, b \rangle = \langle a \circ b, \mathbf{1}_n \rangle$
- $\|a - b\|_2^2 = \|a\|_2^2 + \|b\|_2^2 - 2\langle a, b \rangle$
- $\langle a, b \rangle \langle c, d \rangle = a^\top b c d^\top = b^\top a c d^\top$

A.4 Positive Semi-definite

In this section, we explain the properties of the mathematics operation $\preceq$.

Fact A.3. *Let $u, v \in \mathbb{R}^n$. We have:*

- $u u^\top \preceq \|u\|_2^2 \cdot I_n$.
- $\mathrm{diag}(u) \preceq \|u\|_2 \cdot I_n$
- $\mathrm{diag}(u \circ u) \preceq \|u\|_2^2 \cdot I_n$
- $\mathrm{diag}(u \circ v) \preceq \|u\|_2 \cdot \|v\|_2 \cdot I_n$
- $u v^\top + v u^\top \preceq u u^\top + v v^\top$
- $u v^\top + v u^\top \succeq -(u u^\top + v v^\top)$
- $(v \circ u)(v \circ u)^\top \preceq \|v\|_\infty^2 u u^\top$
- $(v \circ u) u^\top \preceq \|v\|_\infty u u^\top$
- $(v \circ u) u^\top \succeq -\|v\|_\infty u u^\top$

A.5 Basic Calculus and Chain Rule

In this section, we present the basic calculus rules, including the derivative rules for scalars and the derivative rules for vectors.

Fact A.4. *We have*

- *Let $\alpha \in \mathbb{R}$ be a fixed scalar, let $x \in \mathbb{R}^d$ denote variable, then we have $\frac{\mathrm{d}\alpha x}{\mathrm{d}t} = \alpha \frac{\mathrm{d}x}{\mathrm{d}t}$*
- *Let $f(x) \in \mathbb{R}^n$, we have $\frac{\mathrm{d}0.5\|f(x)\|_2^2}{\mathrm{d}t} = \langle f(x), \frac{\mathrm{d}f(x)}{\mathrm{d}t} \rangle$*
- *Let $b \in \mathbb{R}^n$ be a fixed vector, we have $\frac{\mathrm{d}(b \circ f(x))}{\mathrm{d}t} = b \circ \frac{\mathrm{d}f(x)}{\mathrm{d}t}$*
- *Let $z \in \mathbb{R}$ denote a scalar variable, we have the following calculus rules*
  - $\frac{\mathrm{d}\exp(z)}{\mathrm{d}t} = \exp(z) \frac{\mathrm{d}z}{\mathrm{d}t}$
  - $\frac{\mathrm{d}\cosh(z)}{\mathrm{d}t} = \sinh(z) \frac{\mathrm{d}z}{\mathrm{d}t}$
  - $\frac{\mathrm{d}\sinh(z)}{\mathrm{d}t} = \cosh(z) \frac{\mathrm{d}z}{\mathrm{d}t}$
- *Let $x \in \mathbb{R}^n$ denote a vector variable, we have the following calculus rules*
  - $\frac{\mathrm{d}\exp(x)}{\mathrm{d}t} = \exp(x) \circ \frac{\mathrm{d}x}{\mathrm{d}t}$
  - $\frac{\mathrm{d}\cosh(x)}{\mathrm{d}t} = \sinh(x) \circ \frac{\mathrm{d}x}{\mathrm{d}t}$
  - $\frac{\mathrm{d}\sinh(x)}{\mathrm{d}t} = \cosh(x) \circ \frac{\mathrm{d}x}{\mathrm{d}t}$

A.6 Basic Vector Norm Bounds

Now, we analyze the norm bounds for vectors.

Fact A.5. *Given vectors $a \in \mathbb{R}^n$ and $b \in \mathbb{R}^n$, we have*

- $\langle a, b \rangle \leq \|a\|_2 \cdot \|b\|_2$ *(Cauchy-Schwarz inequality)*
- $\|\operatorname{diag}(a)\| \leq \|a\|_\infty \leq \|a\|_2$
- $\|a \circ b\|_2 \leq \|a\|_\infty \cdot \|b\|_2 \leq \|a\|_2 \cdot \|b\|_2$
- $\|a\|_\infty \leq \|a\|_2 \leq \sqrt{n} \cdot \|a\|_\infty$ *(ℓ_∞-norm vs ℓ_2-norm)*
- $\|a\|_2 \leq \|a\|_1 \leq \sqrt{n} \cdot \|a\|_2$ *(ℓ_2-norm vs ℓ_1-norm)*

A.7 Basic Matrix Norm Bound

Then, we analyze the norm bounds for matrices.

Fact A.6. *For matrices A and B, we have*

- *Let $a, b \in \mathbb{R}^d$ denote two vectors, then we have $\|ab^\top\| \leq \|a\|_2 \cdot \|b\|_2$.*
- $\|Ax\| \leq \|A\| \cdot \|x\|_2$.
- $\|AB\| \leq \|A\| \cdot \|B\|$
- *Let $\alpha \in \mathbb{R}$ be a scalar, then we have $\|\alpha \cdot A\| \leq |\alpha| \cdot \|A\|$.*

A.8 Basic Hyperbolic Functions: Scalar Version

In this section, we analyze the properties of the hyperbolic functions, including sinh and cosh, and exponential functions, exp, where the elements of the domains of these functions are all scalars.

Fact A.7. *For a scalar $z \in \mathbb{R}$, we have*

- *Part 1.* $\cosh^2(z) - \sinh^2(z) = 1$
- *Part 2.* $|\exp(z)| \leq \exp(|z|)$
- *Part 3.* $|\cosh(z)| = \cosh(z) = \cosh(|z|) \leq \exp(|z|)$
- *Part 4.* $|\sinh(z)| = \sinh(|z|) \leq \exp(|z|)$
- *Part 5.* $\sinh(|z|) \leq \cosh(|z|) \leq \exp(|z|)$

Taylor expansions

- $\exp(z) = \sum_{i=0}^{\infty} \frac{1}{i!} z^i$
- $\cosh(z) = \sum_{i=0}^{\infty} \frac{1}{(2i)!} z^{2i}$
- $\sinh(z) = \sum_{i=0}^{\infty} \frac{1}{(2i+1)!} z^{2i+1}$

Approximation in small range,

- *For all $x \in \mathbb{R}$ satisfy that $|x| \leq 0.1$, we can get $|\exp(x) - 1| \leq 2|x|$*
- *For all $x \in \mathbb{R}$ satisfy that $|x| \leq 0.1$, we can get $|\cosh(x) - 1| \leq x^2$*
- *For all $x \in \mathbb{R}$ satisfy that $|x| \leq 0.1$, we can get $|\sinh(x)| \leq 2|x|$*
- *For all $x, y \in \mathbb{R}$ satisfy that $|x - y| \leq 0.1$, we can get $|\exp(x) - \exp(y)| \leq \exp(x) \cdot 2|x - y|$*
- *For all $x, y \in \mathbb{R}$ satisfy that $|x - y| \leq 0.1$, we can get $|\cosh(x) - \cosh(y)| \leq \cosh(x) \cdot 2|x - y|$*
- *For all $x, y \in \mathbb{R}$ satisfy that $|x - y| \leq 0.1$, we can get $|\sinh(x) - \sinh(y)| \leq \cosh(x) \cdot 2|x - y|$*

Proof. Most of the proofs are trivial or standard. We only provide some proofs.

Proof of Part 4.

We have

$$\begin{aligned} |\sinh(z)| = & |0.5 \exp(z) - 0.5 \exp(-z)| \\ = & 0.5 \exp(|z|) - 0.5 \exp(-|z|) \\ = & \sinh(|z|) \end{aligned}$$

where second step is true because it's for $z \geq 0$ and also true for $z < 0$.

We have

$$\begin{aligned} |\sinh(z)| = & |0.5 \exp(z) - 0.5 \exp(-z)| \\ \leq & 0.5 \exp(|z|) + 0.5 \exp(|z|) \\ = & \exp(|z|) \end{aligned}$$

Proof of Part 5.

We have

$$\sinh(|z|) = 0.5 \exp(|z|) - 0.5 \exp(-|z|)$$

$$\begin{aligned} &\leq 0.5\exp(|z|) + 0.5\exp(-|z|) \\ &= \cosh(|z)| \end{aligned}$$

We have

$$\begin{aligned} \cosh(|z)| &= 0.5\exp(|z|) + 0.5\exp(-|z|) \\ &\leq 0.5\exp(|z|) + 0.5\exp(|z|) \\ &= \exp(|z|) \end{aligned}$$

□

A.9 Basic Hyperbolic Functions: Vector Version

In this section, we keep analyzing the properties of the hyperbolic functions, namely sinh and cosh, and exponential functions, exp, but the elements of the domains of these functions are all vectors.

Fact A.8 (Formal version of Fact 3.7)**.** *For vectors $a, b \in \mathbb{R}^n$*

- $\|\exp(a)\|_\infty \leq \exp(\|a\|_2)$
- $\|\cosh(a)\|_\infty \leq \cosh(\|a\|_2) \leq \exp(\|a\|_2)$
- $\|\sinh(a)\|_\infty \leq \sinh(\|a\|_2) \leq \cosh(\|a\|_2) \leq \exp(\|a\|_2)$
- $\cosh(a) \circ \cosh(a) - \sinh(a) \circ \sinh(a) = \mathbf{1}_n$

Approximation in a small range,

- *For any $\|a - b\|_\infty \leq 0.01$, we have $\|\exp(a) - \exp(b)\|_2 \leq \|\exp(a)\|_2 \cdot 2\|a - b\|_\infty$*
- *For any $\|a - b\|_\infty \leq 0.01$, we have $\|\cosh(a) - \cosh(b)\|_2 \leq \|\cosh(a)\|_2 \cdot 2\|a - b\|_\infty$*
- *For any $\|a - b\|_\infty \leq 0.01$, we have $\|\sinh(a) - \sinh(b)\|_2 \leq \|\cosh(a)\|_2 \cdot 2\|a - b\|_\infty$*

Proof. Since $\exp, \cosh, \sinh$ are all applied entrywisely, it directly follows from Fact A.7. □

A.10 Regularization

Now, we define the regularization function, L_{reg}, and analyze its properties.

Definition A.9 (Restatement of Definition 3.6)**.** *Given a matrix $A \in \mathbb{R}^{n \times d}$ and $W = \mathrm{diag}(w) \in \mathbb{R}^{n \times n}$ where $w \in \mathbb{R}^n$ is a vector, we define $L_{\mathrm{reg}} : \mathbb{R}^d \to \mathbb{R}$*

$$L_{\mathrm{reg}}(x) := 0.5\|WAx\|_2^2$$

Lemma A.10 (Folklore, see Li et al. (2023e); Deng et al. (2023a) as an example)**.** *If the following condition holds*

- *Let $L_{\mathrm{reg}}(x)$ be defined as Definition A.9.*

Then, we have

- *The gradient is*

$$\frac{\mathrm{d}L_{\mathrm{reg}}}{\mathrm{d}x} = A^\top W^2 Ax$$

- *The Hessian is*

$$\frac{\mathrm{d}^2 L_{\mathrm{reg}}}{\mathrm{d}x^2} = A^\top W^2 A$$

A.11 Gradient and Hessian

Finally, in this section, we define the gradient and Hessian of the loss function and present the definition of the update of the Newton method.

Definition A.11 (Gradient and Hessian)**.** *Let $L(x)$ be some loss function. The gradient $g : \mathbb{R}^d \to \mathbb{R}^d$ of the loss function is defined as*

$$g(x) := \nabla L(x) = \frac{\mathrm{d}L}{\mathrm{d}x}$$

The Hessian $H : \mathbb{R}^d \to \mathbb{R}^{d \times d}$ of the loss function is defined as follow:

$$H(x) := \nabla^2 L(x) = \frac{\mathrm{d}^2 L}{\mathrm{d}x^2}$$

Definition A.12 (Update of the Newton method)**.** *Given that the gradient function $g : \mathbb{R}^d \to \mathbb{R}^d$ and the Hessian matrix $H : \mathbb{R}^d \to \mathbb{R}^{d \times d}$, the exact process of the Newton method can be defined as follows:*

$$x_{t+1} = x_t - H(x_t)^{-1} \cdot g(x_t)$$

B GENERAL FUNCTION: GRADIENT AND HESSIAN COMPUTATIONS

In Section B.1, we compute the gradients of $u(x)$, $\alpha(x)$, $c(x)$, and $L_u(x)$. In Section B.2, we present the second-order derivatives of $u(x)$ with respect to x_i^2 and $x_i x_j$, where x_i and x_j are two arbitrary entries of the vector $x \in \mathbb{R}^d$. In Section B.3, we present the second-order derivatives of $\alpha(x)$ with respect to x_i^2 and $x_i x_j$, where x_i and x_j are two arbitrary entries of the vector $x \in \mathbb{R}^d$. In Section B.4, we compute the second-order derivatives of $c(x)$ with respect to x_i^2 and $x_i x_j$. Finally, in Section B.5, we compute the second-order derivatives of $L_u(x)$ with respect to x_i^2 and $x_i x_j$.

B.1 Gradient Computations

In this section, we compute the gradients of $u(x)$, $\alpha(x)$, $c(x)$, and $L_u(x)$, namely their first-order derivative with respect to x_i.

Lemma B.1 (Gradient)**.** *If the following conditions hold*

- *Let $A \in \mathbb{R}^{n \times d}$ and $b \in \mathbb{R}^n$.*
- *For all $i \in [d]$, $A_{*,i} \in \mathbb{R}^n$ denotes the i-th column of matrix $A \in \mathbb{R}^{n \times d}$.*
- *Let $u(x)$ be defined in Definition 3.1.*
- *Let $v(x)$ be defined in Definition 3.2.*
- *Let $\alpha(x)$ be defined in Definition 3.4.*
- *Let $c(x)$ be defined in Definition 3.5.*
- *Let $L_u(x)$ be defined in Definition 3.3*

Then, for each $i \in [d]$, we have

- *Part 1. (see Part 1 in Lemma 5.6 in page 11 in Deng et al. (2023a))*

$$\frac{\mathrm{d}u(x)}{\mathrm{d}x_i} = v(x) \circ A_{*,i}$$

- *Part 2. (see Part 2 in Lemma 5.6 in page 11 in Deng et al. (2023a))*

$$\frac{\mathrm{d}\alpha(x)}{\mathrm{d}x_i} = \langle v(x), A_{*,i} \rangle$$

- *Part 3.*

$$\frac{\mathrm{d}c(x)}{\mathrm{d}x_i} = v(x) \circ A_{*,i} - b \cdot \langle v(x), A_{*,i} \rangle$$

- *Part 4.*

$$\frac{\mathrm{d}L_u(x)}{\mathrm{d}x_i} = A_{*,i}^\top (c(x) \circ v(x) - v(x) \langle b, c(x) \rangle)$$

Proof. **Proof of Part 1.** For each $j \in [n]$, we have

$$\begin{aligned} \frac{\mathrm{d}(u(x))_j}{\mathrm{d}x_i} &= v(x)_j \cdot \frac{\mathrm{d}(Ax)_j}{\mathrm{d}x_i} \\ &= v(x)_j \cdot \frac{(A\mathrm{d}x)_j}{\mathrm{d}x_i} \\ &= v(x)_j \cdot A_{j,i} \end{aligned}$$

where the first step follows from chain rule (Fact A.4), the second step follows from basic chain rule (Fact A.4), the third step follows from basic calculus rule (Fact A.4).

Since the above equation is true for all $j \in [n]$, we have

$$\underbrace{\frac{\mathrm{d}u(x)}{\mathrm{d}x_i}}_{n \times 1} = \begin{bmatrix} \frac{\mathrm{d}u(x)_1}{\mathrm{d}x_i} \\ \frac{\mathrm{d}u(x)_2}{\mathrm{d}x_i} \\ \vdots \\ \frac{\mathrm{d}u(x)_n}{\mathrm{d}x_i} \end{bmatrix} = \underbrace{v(x)}_{n \times 1} \circ \underbrace{A_{*,i}}_{n \times 1}$$

Proof of Part 2. It trivially follows from arguments in **Part 1**.

Proof of Part 3.

$$\begin{aligned} \frac{\mathrm{d}c(x)}{\mathrm{d}x_i} &= \frac{\mathrm{d}}{\mathrm{d}x_i}(u(x) - b \cdot \alpha(x)) \\ &= v(x) \circ A_{*,i} - b \cdot \langle v(x), A_{*,i} \rangle \end{aligned}$$

where the first step is due to the definition of $c(x)$ (see Definition 3.5), the second step follows from **Part 1** and **Part 2**.

Proof of Part 4.

$$\begin{aligned} \frac{\mathrm{d}L_u(x)}{\mathrm{d}x_i} &= \frac{\mathrm{d}}{\mathrm{d}x_i}(0.5 \cdot \|c(x)\|_2^2) \\ &= c(x)^\top \frac{\mathrm{d}}{\mathrm{d}x_i} c(x) \\ &= c(x)^\top (v(x) \circ A_{*,i} - b \cdot \langle v(x), A_{*,i} \rangle) \\ &= A_{*,i}^\top (c(x) \circ v(x)) - A_{*,i}^\top v(x) \langle b, c(x) \rangle \\ &= A_{*,i}^\top (c(x) \circ v(x) - v(x) \langle b, c(x) \rangle) \end{aligned}$$

where the first step is due to the definition of $L_u(x)$ (see Definition 3.3), the second step follows from chain rule (Fact A.4), the third step is due to **Part 3**, the fourth step is obtained by using Fact A.1, and the fifth step follows from simple algebra. □

B.2 Hessian Computation Step 1. Vector Function $\exp(Ax)$

We state a tool from previous work (Li et al., 2023e; Deng et al., 2023a).

Lemma B.2 (Lemma 5.9 in Deng et al. (2023a) and implicitly in Li et al. (2023e))**.** *If the following conditions hold*

- *Let $A \in \mathbb{R}^{n \times d}$*
- *Let $x \in \mathbb{R}^d$.*

We have

- *Part 1.*

$$\frac{\mathrm{d}^2 u(x)}{\mathrm{d} x_i^2} = A_{*,i} \circ u(x) \circ A_{*,i}$$

- *Part 2.*

$$\frac{\mathrm{d}^2 u(x)}{\mathrm{d} x_i \mathrm{d} x_j} = A_{*,j} \circ u(x) \circ A_{*,i}.$$

B.3 Hessian Computation Step 2. Scalar Function $\alpha(x)$

We state a tool from previous work (Li et al., 2023e; Deng et al., 2023a).

Lemma B.3 (Lemma 5.10 in Deng et al. (2023a), implicitly in Li et al. (2023e))**.** *If the following conditions hold*

- *Let $\alpha(x)$ be defined as Definition 3.4.*
- *Let $u(x)$ be defined as in Definition 3.1.*
- *Let $A \in \mathbb{R}^{n \times d}$.*
- *Let $x \in \mathbb{R}^d$.*

Then, we have

- *Part 1.*

$$\frac{\mathrm{d}^2 \alpha(x)}{\mathrm{d} x_i^2} = \langle u(x), A_{*,i} \circ A_{*,i} \rangle$$

- *Part 2.*

$$\frac{\mathrm{d}^2 \alpha(x)}{\mathrm{d} x_i \mathrm{d} x_j} = \langle u(x), A_{*,i} \circ A_{*,j} \rangle$$

B.4 Hessian Computation Step 3. Vector Function $c(x)$

Now, we compute the second-order derivative of $c(x)$ with respect to x_i^2 and $x_i x_j$.

Lemma B.4. *If the following conditions hold*

- *Let $c(x)$ be defined as Definition 3.5.*
- *Let $A \in \mathbb{R}^{n \times d}$.*
- *Let $x \in \mathbb{R}^d$.*

- *Let $b \in \mathbb{R}^n$.*

Then, we have

- *Part 1.*

$$\frac{\mathrm{d}^2 c(x)}{\mathrm{d} x_i^2} = A_{*,i} \circ u(x) \circ A_{*,i} - b \cdot \langle u(x), A_{*,i} \circ A_{*,i} \rangle$$

- *Part 2.*

$$\frac{\mathrm{d}^2 c(x)}{\mathrm{d} x_i \mathrm{d} x_j} = A_{*,i} \circ u(x) \circ A_{*,j} - b \cdot \langle u(x), A_{*,i} \circ A_{*,j} \rangle$$

Proof. **Proof of Part 1.**

$$\begin{aligned} \frac{\mathrm{d}^2 c(x)}{\mathrm{d} x_i^2} = & \frac{\mathrm{d}^2}{\mathrm{d} x_i^2} (u(x) - b \cdot \alpha(x)) \\ = & A_{*,i} \circ u(x) \circ A_{*,i} - b \cdot \langle u(x), A_{*,i} \circ A_{*,i} \rangle \end{aligned}$$

where the first step follows from definition of $c(x)$ (see Definition 3.5), the second step follows from Lemma B.2 and Lemma B.3.

Proof of Part 2.

$$\begin{aligned} \frac{\mathrm{d}^2 c(x)}{\mathrm{d} x_i \mathrm{d} x_j} = & \frac{\mathrm{d}^2}{\mathrm{d} x_i \mathrm{d} x_j} (u(x) - b \cdot \alpha(x)) \\ = & A_{*,i} \circ u(x) \circ A_{*,j} - b \cdot \langle u(x), A_{*,i} \circ A_{*,j} \rangle \end{aligned}$$

where the first step follows from definition of $c(x)$ (see Definition 3.5), the second step follows from Lemma B.2 and Lemma B.3. □

B.5 Hessian Computation Step 4. Scalar Function $L_u(x)$

Then, we compute the second-order derivative of $L_u(x)$ with respect to x_i^2 and $x_i x_j$, by first introducing some functions, $B_{1,1}, B_{1,2}, B_{1,3}, B_{1,4}, B_{2,1}, B_{2,2}$ (see Definition B.5), to simplify the process of computation.

Definition B.5. *Given the following objects*

- *Let $A \in \mathbb{R}^{n \times d}$.*
- *Let $x \in \mathbb{R}^d$.*
- *Let $b \in \mathbb{R}^n$.*

Then, we define the functions $B_{1,1}, B_{1,2}, B_{1,3}, B_{1,4}, B_{2,1}, B_{2,2} : \mathbb{R}^d \to \mathbb{R}^{n \times n}$ as

$$\begin{aligned} B_{1,1}(x) := & \operatorname{diag}(v(x) \circ v(x)) \\ B_{1,2}(x) := & -(v(x) \circ b) \cdot v(x)^\top \\ B_{1,3}(x) := & -v(x) \cdot (v(x) \circ b)^\top \\ B_{1,4}(x) := & \|b\|_2^2 \cdot v(x) v(x)^\top \end{aligned}$$

We define

$$B_{2,1}(x) := \operatorname{diag}(c(x) \circ u(x))$$

$$B_{2,2}(x) := -\langle c(x), b\rangle \operatorname{diag}(u(x))$$

We define $B : \mathbb{R}^d \to \mathbb{R}^{n\times n}$ *as follows:*

$$\begin{aligned} B(x) := & B_{1,1}(x) + B_{1,2}(x) + B_{1,3}(x) + B_{1,4}(x) \\ & + B_{2,1}(x) + B_{2,2}(x) \end{aligned}$$

Lemma B.6. *If the following conditions hold*

- *Let* $B(x)$ *be defined as in Definition B.5.*
- *Let* $A \in \mathbb{R}^{n\times d}$.
- *Let* $L_u(x)$ *be defined as in Definition 3.3.*

Then, we have

- *Part 1.*

$$\frac{\mathrm{d}^2 L_u(x)}{\mathrm{d}x_i^2} = A_{*,i}^\top B A_{*,i}$$

- *Part 2.*

$$\frac{\mathrm{d}^2 L_u(x)}{\mathrm{d}x_i \mathrm{d}x_j} = A_{*,i}^\top B A_{*,j}$$

Proof. **Proof of Part 1.**

$$\begin{aligned} \frac{\mathrm{d}^2 L_u(x)}{\mathrm{d}x_i^2} = & \frac{\mathrm{d}}{\mathrm{d}x_i}(\frac{\mathrm{d}L_u(x)}{\mathrm{d}x_i}) \\ = & \frac{\mathrm{d}}{\mathrm{d}x_i}(c(x)^\top \frac{\mathrm{d}c(x)}{\mathrm{d}x_i}) \\ = & \langle \frac{\mathrm{d}c(x)}{\mathrm{d}x_i}, \frac{\mathrm{d}c(x)}{\mathrm{d}x_i}\rangle + \langle c(x), \frac{\mathrm{d}^2 c(x)}{\mathrm{d}x_i^2}\rangle \end{aligned}$$

where the first step follows from simple algebra, the second step follows from basic chain rule (see Fact A.4), and the last step follows from basic calculus.

For the first term, in the above equation, we have

$$\begin{aligned} \langle \frac{\mathrm{d}c(x)}{\mathrm{d}x_i}, \frac{\mathrm{d}c(x)}{\mathrm{d}x_i}\rangle = & \|v(x) \circ A_{*,i} - b \cdot \langle v(x), A_{*,i}\rangle\|_2^2 \\ = & A_{*,i}^\top \operatorname{diag}(v(x) \circ v(x)) A_{*,i} \\ & - A_{*,i}^\top (v(x) \circ b) v(x)^\top A_{*,i} \\ & - A_{*,i}^\top (v(x))(v(x) \circ b)^\top A_{*,i} \\ & + A_{*,i}^\top \|b\|_2^2 v(x) v(x)^\top A_{*,i} \\ = & A_{*,i}^\top (B_{1,1}(x) + B_{1,2}(x) + B_{1,3}(x) + B_{1,4}(x)) A_{*,i} \end{aligned}$$

where the first step is due to **Part 3** of Lemma B.1, the second step follows from Fact A.2, and the last step follows from the definition of $B_{1,i}(x)$ for each $i \in [4]$ (see Definition B.5).

For the second term, we have

$$\langle c(x), \frac{\mathrm{d}^2 c(x)}{\mathrm{d}x_i^2}\rangle = \langle c(x), A_{*,i} \circ u(x) \circ A_{*,i} - b \cdot \langle u(x), A_{*,i} \circ A_{*,i}\rangle\rangle$$

$$\begin{aligned} &= A_{*,i}^\top \operatorname{diag}(c(x) \circ u(x)) A_{*,i} \\ &\quad - A_{*,i}^\top \langle c(x), b \rangle \operatorname{diag}(u(x)) A_{*,i} \\ &= A_{*,i}^\top (B_{2,1}(x) + B_{2,2}(x)) A_{*,i} \end{aligned}$$

where the first step is due to **Part 1** of Lemma B.4, the second step follows from Fact A.2, and the last step follows from $B_{2,i}$ for all $i \in [2]$ (see Definition B.5).

Thus, we finally have

$$\frac{\mathrm{d}^2 L_u(x)}{\mathrm{d} x_i^2} = A_{*,i}^\top B(x) A_{*,i}$$

Proof of Part 2.

The proof is similar, and we omitted the details here. □

C GENERAL FUNCTION: PSD LOWER BOUND

In Section C.1, we provide the upper bound for the ℓ_2 norms of $u(x), v(x), c(x) \in \mathbb{R}^n$ and for the absolute value of $\alpha(x) \in \mathbb{R}$. In Section C.2, we compute both the upper bound and the lower bound of $B(x)$ in terms of $\preceq$. In Section C.3, we analyze the lower bound of Hessian.

C.1 Upper Bound for Several Basic Quantities

In this section, we compute the bounds for the ℓ_2 norms of the vectors $u(x), v(x), c(x) \in \mathbb{R}^n$ and compute the bound for the absolute value of $\alpha(x)$.

Claim C.1. *If the following conditions hold*

- *Let* $R \geq 2$.
- $\|A\| \leq R$
- $\|x\|_2 \leq R$
- $\|b\|_2 \leq R$
- *Let* $u(x) \in \mathbb{R}^n$ *be defined as Definition 3.1.*
- *Let* $v(x) \in \mathbb{R}^n$ *be defined as Definition 3.2.*
- *Let* $\alpha(x) \in \mathbb{R}$ *be defined as Definition 3.4.*
- *Let* $c(x) \in \mathbb{R}^n$ *be defined as in Definition 3.5.*

Then, we have

- *Part 1. (see Li et al. (2023e); Deng et al. (2023a); Li et al. (2023a))*

$$\begin{aligned} \|u(x)\|_2 &\leq \sqrt{n} \exp(R^2) \\ \|v(x)\|_2 &\leq \sqrt{n} \exp(R^2) \end{aligned}$$

- *Part 2.*

$$|\alpha(x)| \leq n \exp(R^2)$$

- *Part 3.*

$$\|c(x)\|_2 \leq nR \exp(R^2)$$

Proof. **Proof of Part 1.** The proof is standard in the literature, and we omit the details here.

Proof of Part 2. We can show

$$\begin{aligned} |\alpha(x)| &= |\langle u(x), \mathbf{1}_n \rangle| \\ &\leq \sqrt{n} \cdot \|u(x)\|_2 \\ &\leq n \cdot \exp(R^2) \end{aligned}$$

where the first step follows from definition of $\alpha(x)$ (see Definition 3.4), the second step follows from Fact A.5, the third step follows from **Part 1**.

Proof of Part 3. We can show

$$\begin{aligned} \|c(x)\|_2 &= \|u(x) - \alpha(x) b\|_2 \\ &\leq \|u(x)\|_2 + \|\alpha(x) b\|_2 \\ &= \sqrt{n} \exp(R^2) + |\alpha(x)| \cdot \|b\|_2 \\ &\leq \sqrt{n} \exp(R^2) + |\alpha(x)| \cdot R \\ &\leq \sqrt{n} \exp(R^2) + nR \exp(R^2) \\ &\leq 2nR \exp(R^2), \end{aligned}$$

where the first step comes from the definition of $c(x)$ (see Definition 3.5), the second step follows from the triangle inequality, the third step is because of **Part 1**, the fourth step follows from the assumption on b, the fifth step follows from **Part 2**, and the last step follows from simple algebra.

□

C.2 PSD Bounds for Several Basic Matrix Functions

In this section, we first define the matrices $B^1_{\text{rank}}, B^2_{\text{rank}}, B^3_{\text{rank}}, B^1_{\text{diag}}, B^2_{\text{diag}} \in \mathbb{R}^{n \times n}$ and find the $\preceq$-bound for them. Then, we combine them together to form the bound for $B(x) \in \mathbb{R}^{n \times n}$

Definition C.2. *Given the following objects*

- *Let $u(x)$ be defined as in Definition 3.1.*
- *Let $c(x)$ be defined as in Definition 3.5.*
- *Let $b \in \mathbb{R}^n$.*

We define $B : \mathbb{R}^d \to \mathbb{R}^{n \times n}$ as follows:

$$B(x) := B_{\text{rank}} + B_{\text{diag}}$$

We define

$$\begin{aligned} B_{\text{rank}} &:= B^1_{\text{rank}} + B^2_{\text{rank}} + B^3_{\text{rank}} \\ B_{\text{diag}} &:= B^1_{\text{diag}} + B^2_{\text{diag}} \end{aligned}$$

We define

- $B^1_{\text{rank}} := -u(x)(u(x) \circ b)^\top$
- $B^2_{\text{rank}} := -(u(x) \circ b) u(x)^\top$
- $B^3_{\text{rank}} := \|b\|_2^2 u(x) u(x)^\top$
- $B^1_{\text{diag}} := \text{diag}((u(x) + c(x)) \circ u(x) + q)$
  - $q = \mathbf{0}_n$ *(when $u(x) = \exp(Ax)$)*

  - $q = -\mathbf{1}_n$ *(when* $u(x) = \cosh(Ax)$*)*
  - $q = \mathbf{1}_n$ *(when* $u(x) = \sinh(Ax)$*)*
- $B^2_{\mathrm{diag}} := -\langle c(x), b\rangle \operatorname{diag}(u(x))$

Lemma C.3. *If the following situations hold*

- $B(x)$ *is a* $n \times n$ *dimension matrix (See Definition C.2).*
- B^1_{rank}, B^2_{rank}, B^3_{rank} *are defined in Definition C.2.*
- B^1_{diag}, B^2_{diag} *are defined in Definition C.2.*

Then, we have

- *Part 1.*
$$-\|b\|_2 v(x)v(x)^\top \preceq B^1_{\mathrm{rank}} \preceq \|b\|_2 v(x)v(x)^\top$$
- *Part 2.*
$$-\|b\|_2 v(x)v(x)^\top \preceq B^2_{\mathrm{rank}} \preceq \|b\|_2 v(x)v(x)^\top$$
- *Part 3.*
$$B^3_{\mathrm{rank}} = \|b\|_2^2 v(x)v(x)^\top$$
- *Part 4.*
$$-(1 + (\|u(x)\|_\infty + \|c(x)\|_\infty) \cdot \|u(x)\|_\infty) \cdot I_n \preceq B^1_{\mathrm{diag}} \preceq (1 + (\|u(x)\|_\infty + \|c(x)\|_\infty) \cdot \|u(x)\|_\infty) \cdot I_n$$
- *Part 5.*
$$-\|b\|_2 \|c(x)\|_2 \|u(x)\|_\infty I_n \preceq B^2_{\mathrm{diag}} \preceq \|b\|_2 \|c(x)\|_2 \|u(x)\|_\infty I_n$$
- *Part 6.*
  - *Let* $R_0 = \max\{\|u(x)\|_2, \|v(x)\|_2, \|b\|_2, \|c(x)\|_2, 1\}$
  - *Then, we have*
$$-10R_0^4 \cdot I_n \preceq B(x) \preceq 10R_0^4 \cdot I_n$$

Proof. **Proof of Part 1.** First, we focus on the lower bound of B^1_{rank}. We have

$$\begin{aligned} B^1_{\mathrm{rank}} = & -v(x)(v(x) \circ b)^\top \\ \succeq & -\|b\|_2 \cdot v(x)v(x)^\top, \end{aligned}$$

where the first step follows from the definition of B^1_{rank} (see Definition C.2) and the second step follows from Fact A.3.

Similarly, we have

$$\begin{aligned} B^1_{\mathrm{rank}} = & -v(x)(v(x) \circ b)^\top \\ \preceq & \|b\|_2 \cdot v(x)v(x)^\top, \end{aligned}$$

where the first step follows from the definition of B^1_{rank} (see Definition C.2) and the second step follows from Fact A.3

Proof of Part 2. According to what we obtained in the Part 1, we can also have

$$-\|b\|_2 v(x)v(x)^\top \preceq B_{\mathrm{rank}}^2 \preceq \|b\|_2 v(x)v(x)^\top$$

Proof of Part 3.

The proof is trivially following from definition of B_{rank}^3. We have

$$B_{\mathrm{rank}}^3 = \|b\|_2^2 \cdot v(x)v(x)^\top$$

Proof of Part 4. For $i \in [n]$, $u(x)_i > 0$, we have

$$\begin{aligned} B_{\mathrm{diag}}^1 = & \ \mathrm{diag}((u(x) + c(x)) \circ u(x) + q) \\ \preceq & \ (1 + (\|u(x)\|_\infty + \|c(x)\|_\infty)\|u(x)\|_\infty) \cdot I_n, \end{aligned}$$

where the first step is due to the definition of B_{diag}^1 (see Definition C.2) and the second step follows from Fact A.3.

On the other hand, we have

$$B_{\mathrm{diag}}^1 \succeq -(1 + (\|u(x)\|_\infty + \|c(x)\|_\infty)\|u(x)\|_\infty) \cdot I_n$$

Proof of Part 5.

$$\begin{aligned} B_{\mathrm{diag}}^2 = & \ -\langle c(x), b\rangle \, \mathrm{diag}(u(x)) \\ \preceq & \ \|b\|_2 \cdot \|c(x)\|_2 \cdot \mathrm{diag}(u(x)) \\ \preceq & \ \|b\|_2 \cdot \|c(x)\|_2 \cdot \|u(x)\|_\infty \cdot I_n, \end{aligned}$$

where the first step follows from the definition of B_{diag}^2 (see Definition C.2), the second step follows from Fact A.3, and the third step follows from Fact A.3.

Similarly, we have

$$\begin{aligned} B_{\mathrm{diag}}^2 = & \ -\langle c(x), b\rangle \, \mathrm{diag}(u(x)) \\ \succeq & \ -\|b\|_2 \cdot \|c(x)\|_2 \cdot \mathrm{diag}(u(x)) \\ \succeq & \ -\|b\|_2 \cdot \|c(x)\|_2 \cdot \|u(x)\|_\infty \cdot I_n, \end{aligned}$$

where the first step comes from the definition of B_{diag}^2 (see Definition C.2), the second step follows from Fact A.3, and the third step follows from Fact A.3.

Proof of Part 6. Using Fact A.3

$$u(x)u(x)^\top \preceq \|u(x)\|_2^2 I_n$$

We also have

$$\max\{B_{\mathrm{rank}}^1, B_{\mathrm{rank}}^2, B_{\mathrm{rank}}^3, B_{\mathrm{diag}}^1, B_{\mathrm{diag}}^2\} \leq 2R_0^4 \cdot I_n$$

□

C.3 Lower bound on Hessian

In this section, we compute the lower bound for Hessian.

Lemma C.4. *If conditions as follows are satisfied*

- *Let* $A \in \mathbb{R}^{n \times d}$.

- *Let $u(x)$ be defined as Definition 3.1.*
- *Let $v(x)$ be defined as Definition 3.2.*
- *$L_u(x)$ is defined in Definition 3.3.*
- *$L_{\mathrm{reg}}(x)$ is defined in Definition A.9.*
- *$L(x) = L_{\mathrm{reg}}(x) + L_u(x)$.*
- *Given $w \in \mathbb{R}^n$, $W = \mathrm{diag}(w) \in \mathbb{R}^{n \times n}$ and W^2 denotes the matrix with w_i^2 as the i-th diagonal.*
- *We use $\sigma_{\min}(A)$ as the minimum singular value of A.*
- *We let $l > 0$ as a scalar.*
- *Let $R_0 = \max\{\|u(x)\|_2, \|v\|_2, \|b\|_2, \|c(x)\|_2, 1\}$.*

Then we have

- *Part 1. If all $i \in [n]$, $w_i^2 \geq 10R_0^4 + l/\sigma_{\min}(A)^2$, then we have*
$$\frac{\mathrm{d}^2 L}{\mathrm{d}x^2} \succeq l \cdot I_d$$
- *Part 2. If all $i \in [n]$, $w_i^2 \geq 200R_0^4 + l/\sigma_{\min}(A)^2$, then we have*
$$(1 - 1/10) \cdot (B(x) + W^2) \preceq W^2 \preceq (1 + 1/10) \cdot (B(x) + W^2).$$

Proof. By Lemma B.6, we have

$$\frac{\mathrm{d}^2 L_u}{\mathrm{d}x^2} = A^\top B(x) A, \tag{4}$$

By Lemma A.10, we have

$$\frac{\mathrm{d}^2 L_{\mathrm{reg}}}{\mathrm{d}x^2} = A^\top W^2 A. \tag{5}$$

By what we have in the Lemma statement, we also have

$$\frac{\mathrm{d}^2 L}{\mathrm{d}x^2} = \frac{\mathrm{d}^2 L_u}{\mathrm{d}x^2} + \frac{\mathrm{d}^2 L_{\mathrm{reg}}}{\mathrm{d}x^2} \tag{6}$$

By combining Eq. (4), Eq. (5), and Eq. (6), we can rewrite the equation above as follows:

$$\begin{aligned} \frac{\mathrm{d}^2 L}{\mathrm{d}x^2} &= A^\top B(x) A + A^\top W^2 A \\ &= A^\top (B(x) + W^2) A, \end{aligned}$$

where the second step follows from simple algebra.

Now we define

$$D := B(x) + W^2$$

Now we get the bound of D

$$\begin{aligned} D \succeq & -10R_0^4 I_n + w_{\min}^2 I_n \\ = & (w_{\min}^2 - 10R_0^4) I_n \end{aligned}$$

$$\succeq \frac{l}{\sigma_{\min}(A)^2} I_n,$$

where the first step follows from **Part 6** of Lemma C.3, the second step follows from simple algebra, and the third step is because of the assumption of this part.

Since D is positive definite, then we have

$$A^\top D A \succeq \sigma_{\min}(D) \cdot \sigma_{\min}(A)^2 \cdot I_d \succeq l \cdot I_d$$

Proof of Part 2.

Using **Part 6** of Lemma C.3, we have

$$-10R_0^4 I_n \preceq B(x) \preceq 10R_0^4 I_n.$$

From assumption on W, we also have

$$\begin{aligned} W^2 &\succeq 200R_0^4 I_n \\ -W^2 &\preceq -200R_0^4 I_n \end{aligned}$$

Combining the above three equations,

$$-\frac{1}{20}W^2 \preceq B(x) \preceq \frac{1}{20}W^2$$

Thus,

$$(1 - \frac{1}{20})W^2 \preceq B(x) + W^2 \preceq (1 + \frac{1}{20})W^2$$

which implies that

$$-(1 + \frac{1}{10})(B(x) + W^2) \preceq W^2 \preceq (1 + \frac{1}{10})(B(x) + W^2)$$

□

D GENERAL FUNCTION: HESSIAN IS LIPSCHITZ WITH RESPECT TO x

In Section D.1, we summarize all of the important properties that we derive in the following subsections to form an upper bound for $\|H(x) - H(y)\|$. In Section D.2, we analyze the upper bound for $\|u(x) - u(y)\|_2$. In Section D.3, we analyze the upper bound for $|\alpha(x) - \alpha(y)|$. In Section D.4, we prove the upper bound for $\|c(x) - c(y)\|_2$. In Section D.5, we evaluate the upper bound of the sum of all the spectral norms of the matrices $G_i \in \mathbb{R}^{n \times n}$, for all $i \in [5]$, where the spectral norms of each of the matrix G_i is evaluated in each of the following subsection. In Section D.6, we analyze the upper bound of the spectral norm of $G_1 \in \mathbb{R}^{n \times n}$. In Section D.7, we find the upper bound of the spectral norm of $G_2 \in \mathbb{R}^{n \times n}$. In Section D.8, we study the upper bound of the spectral norm of $G_3 \in \mathbb{R}^{n \times n}$. In Section D.9, we prove the upper bound of the spectral norm of $G_4 \in \mathbb{R}^{n \times n}$. In Section D.10, we show the upper bound of the spectral norm of $G_5 \in \mathbb{R}^{n \times n}$.

D.1 Main Result

In this section, we introduce our main result, which is the combination of all the important concepts in Section D.

Lemma D.1. *If the following condition holds*

- *Let* $H(x) = \frac{\mathrm{d}^2 L}{\mathrm{d}x^2}$
- *Let* $R > 4$

- $\|x\|_2 \leq R$, $\|y\|_2 \leq R$, *where* $x, y \in \mathbb{R}^d$
- $\|A(x-y)\|_\infty < 0.01$, *where* $A \in \mathbb{R}^{n \times d}$
- $\|A\| \leq R$
- $\|b\|_2 \leq R$, *where* $b \in \mathbb{R}^n$
- *Let* $R_\infty := \max\{\|u(x)\|_2, \|u(y)\|_2, \|c(x)\|_2, \|c(y)\|_2, 1\}$
  - *where* $R_\infty \leq 2nR\exp(R^2)$
  - *this is proved by Part 1 and Part 3 in Claim C.1*

Then we have

$$\|H(x) - H(y)\| \leq n^4 \exp(20R^2) \cdot \|x - y\|_2$$

Proof.

$$\begin{aligned}
& \|H(x) - H(y)\| \\
\leq & \|A\| \cdot (\|G_1\| + \|G_2\| + \cdots + \|G_5\|)\|A\| \\
\leq & R^2 \cdot (\|G_1\| + \|G_2\| + \cdots + \|G_5\|) \\
\leq & R^2 \cdot 5 \cdot R_\infty^3 \|b\|_2 \sqrt{n} \cdot \|u(x) - u(y)\|_2 \\
\leq & R^2 \cdot 5 \cdot R_\infty^3 \|b\|_2 \sqrt{n} \cdot 2\sqrt{n} R \exp(R^2) \cdot \|x - y\|_2 \\
\leq & 80 n^4 R^6 \exp(4R^2) \cdot \|x - y\|_2 \\
\leq & n^4 \exp(20R^2) \cdot \|x - y\|_2,
\end{aligned}$$

where the first step is due to the definition of G_i (see Lemma D.5) and Fact A.6, the second step follows from $\|A\| \leq R$, the third step follows from Lemma D.5, the fourth step is because of Lemma D.2, the fifth step is due to the assumption on R_∞, and the last step is from simple algebra. □

D.2 Lipschitz for $u(x)$

We use a tool from Deng et al. (2023a).

Lemma D.2 (Part 1 of Lemma 7.2 in Deng et al. (2023a))**.** *If the following conditions hold*

- *Let* $u(x)$ *be defined in definition 3.1.*
- *Let* $\|A(x-y)\|_\infty < 0.01$
- *Let* $\|A\| \leq R$, *where* $A \in \mathbb{R}^{n \times d}$
- *Let* $\|x\|_2, \|y\|_2 \leq R$ *, where* $x, y \in \mathbb{R}^d$

then, we have

$$\|u(x) - u(y)\|_2 \leq 2\sqrt{n} R \exp(R^2) \|x - y\|_2$$

D.3 Lipschitz for $\alpha(x)$

We use a tool from previous work, namely Deng et al. (2023a).

Lemma D.3 (Part 2 of Lemma 7.2 in Deng et al. (2023a))**.** *If the following conditions hold*

- *Let* $\alpha(x)$ *be defined as Definition 3.4.*
- *Let* $u(x)$ *be defined as Definition 3.1.*

then, we have

$$|\alpha(x) - \alpha(y)| \leq \sqrt{n} \cdot \|u(x) - u(y)\|_2$$

D.4 Lipschitz for $c(x)$

We find the upper bound of $\|c(x) - c(y)\|_2$.

Lemma D.4. *If the following situations hold*

- *Let $c(x)$ be defined in Definition 3.5.*
- *Let $\alpha(x)$ be defined as Definition 3.4.*
- *Let $u(x)$ be defined as Definition 3.1.*
- *Let $b \in \mathbb{R}^n$.*

Then, we have

$$\|c(x) - c(y)\|_2 \leq \|u(x) - u(y)\|_2 + |\alpha(x) - \alpha(y)| \cdot \|b\|_2$$

Proof. We have

$$\begin{aligned} \|c(x) - c(y)\|_2 &= \|(u(x) - \alpha(x) \cdot b) - (u(y) - \alpha(y) \cdot b)\|_2 \\ &\leq \|u(x) - u(y)\|_2 + \|(\alpha(x) - \alpha(y)) \cdot b\|_2 \\ &= \|u(x) - u(y)\|_2 + |\alpha(x) - \alpha(y)| \cdot \|b\|_2 \end{aligned}$$

where the first step is from how we defined c (Definition 3.5), the second step is due to the triangle inequality, and the last step follows from simple algebra. □

D.5 Summary of Five Steps

In this section, we analyze the upper bound of the sum of $\|G_i\|$, for all $i \in [5]$.

Lemma D.5. *If the following conditions hold*

- $G_1 = v(x)(v(x) \circ b)^\top - v(y)(v(y) \circ b)^\top$
- $G_2 = (v(x) \circ b)v(x)^\top - (v(y) \circ b)v(y)^\top$
- $G_3 = \|b\|_2^2 v(x)v(x)^\top - \|b\|_2^2 v(y)v(y)^\top$
- $G_4 = \mathrm{diag}((u(x) + c(x)) \circ u(x)) - \mathrm{diag}((u(y) + c(y)) \circ u(y))$
- $G_5 = \langle c(x), b \rangle \mathrm{diag}(u(x)) - \langle c(y), b \rangle \mathrm{diag}(u(y))$
- *Let $R_\infty := \max\{\|u(x)\|_2, \|u(y)\|_2, \|v(x)\|_2, \|v(y)\|_2, \|c(x)\|_2, \|c(y)\|_2, \|b\|_2, 1\}$*

Then, we have

- *Part 1.*

$$\sum_{i=1}^5 \|G_i\| \leq 20R_\infty^3 \cdot \max\{\|u(x) - u(y)\|_2, \|c(x) - c(y)\|_2\}.$$

- *Part 2. Let $\|b\|_2 \leq R$*

$$\sum_{i=1}^5 \|G_i\| \leq 100R_\infty^3 R\sqrt{n}\|u(x) - u(y)\|_2$$

Proof. **Proof of Part 1.**

Using Lemma D.6, Lemma D.7, Lemma D.8, Lemma D.9 and Lemma D.10, we can show for each $i \in [5]$, we have

$$\|G_i\| \leq 20R_\infty^3 \cdot \max\{\|u(x) - u(y)\|_2, \|c(x) - c(y)\|_2\}.$$

Proof of Part 2.

Note that

$$\begin{aligned}
\|c(x) - c(y)\|_2 \leq & \|u(x) - u(y)\|_2 + |\alpha(x) - \alpha(y)| \cdot \|b\|_2 \\
\leq & \|u(x) - u(y)\|_2 + \|u(x) - u(y)\|_2 \sqrt{n} \|b\|_2 \\
\leq & \|u(x) - u(y)\|_2 + \|u(x) - u(y)\|_2 \sqrt{n} R \\
\leq & 2\sqrt{n} R \|u(x) - u(y)\|_2,
\end{aligned}$$

where the first step follows from Lemma D.4, the second step follows from Lemma D.3, the third step follows from the assumption on $\|b\|_2 \leq R$, and the last step follows from simple algebra.

□

D.6 Lipschitz Calculations: Step 1. Lipschitz for Matrix Function $v(x)(v(x) \circ b)^\top$

We find the upper bound of $\|G_1\|$.

Lemma D.6. *If the following conditions hold*

- $G_1 = v(x)(v(x) \circ b)^\top - v(y)(v(y) \circ b)^\top$

Then, we have

$$\|G_1\| \leq 2 \max\{\|v(x)\|_2, \|v(y)\|_2\} \cdot \|b\|_2 \cdot \|v(x) - v(y)\|_2.$$

Proof. We define

$$\begin{aligned}
G_{1,1} := & v(x)(v(x) \circ b)^\top - v(y)(v(x) \circ b)^\top \\
G_{1,2} := & v(y)(v(x) \circ b)^\top - v(y)(v(y) \circ b)^\top
\end{aligned}$$

We have

$$G_1 = G_{1,1} + G_{1,2}$$

We can show

$$\begin{aligned}
\|G_{1,1}\| = & \|(v(x) - v(y)) \cdot (v(x) \circ b)^\top\| \\
\leq & \|v(x) - v(y)\|_2 \cdot \|v(x) \circ b\|_2 \\
\leq & \|v(x) - v(y)\|_2 \cdot \|v(x)\|_2 \cdot \|b\|_2
\end{aligned}$$

where the first step is due to the definition of $G_{1,1}$, the second step follows from Fact A.6, and the last step follows from Fact A.5.

Similarly, we can also show

$$\begin{aligned}
\|G_{1,2}\| = & \|v(y) \cdot ((v(x) - v(y)) \circ b)^\top\| \\
\leq & \|v(y)\|_2 \cdot \|(v(x) - v(y)) \circ b\|_2 \\
\leq & \|v(y)\|_2 \cdot \|v(x) - v(y)\|_2 \cdot \|b\|_2
\end{aligned}$$

where the first step is due to the definition of $G_{1,2}$, the second step follows from Fact A.6, and the last step follows from Fact A.5. Thus, we complete the proof. □

D.7 Lipschitz Calculations: Step 2. Lipschitz for Matrix Function $(v(x) \circ b)v(x)^\top$

We look for the upper bound of $\|G_2\|$.

Lemma D.7. *If the following conditions hold*

- $G_2 = (v(x) \circ b)(v(x))^\top - (v(y) \circ b)v(y)^\top$

Then, we have

$$\|G_2\| \leq 2\max\{\|v(x)\|_2, \|v(y)\|_2\} \cdot \|b\|_2 \cdot \|v(x) - v(y)\|_2.$$

Proof. The proof is very similar to the previous Lemma. So we omit the details here. □

D.8 Lipschitz Calculations: Step 3. Lipschitz for Matrix Function $\|b\|_2^2 v(x)v(x)^\top$

We analyze the upper bound of $\|G_3\|$.

Lemma D.8. *If the following conditions hold*

- $G_3 = \|b\|_2^2 v(x)v(x)^\top - \|b\|_2^2 v(y)v(y)^\top$

Then, we have

$$\|G_3\| \leq 2\max\{\|v(x)\|_2, \|v(y)\|_2\} \cdot \|b\|_2^2 \cdot \|v(x) - v(y)\|_2.$$

Proof. We define

$$\begin{aligned} G_{3,1} :=& \|b\|_2^2 v(x)v(x)^\top - \|b\|_2^2 v(y)v(x)^\top \\ G_{3,2} :=& \|b\|_2^2 v(y)v(x)^\top - \|b\|_2^2 v(y)v(y)^\top \end{aligned}$$

We have

$$G_3 = G_{3,1} + G_{3,2}.$$

We can show that

$$\begin{aligned} \|G_{3,1}\| =& \|b\|_2^2 \cdot \|v(x)v(x)^\top - v(y)v(x)^\top\| \\ =& \|b\|_2^2 \cdot \|(v(x) - v(y))v(x)^\top\| \\ \leq& \|b\|_2^2 \cdot \|v(x) - v(y)\|_2 \cdot \|v(x)\|_2, \end{aligned}$$

where the first step comes from the definition of $G_{3,1}$, the second step is due to simple algebra, and the third step follows from Fact A.6.

Similarly, we can show that

$$\|G_{3,2}\| \leq \|b\|_2^2 \cdot \|v(x) - v(y)\|_2 \cdot \|v(x)\|_2.$$

Thus, we complete the proof. □

D.9 Lipschitz Calculations: Step 4. Lipschitz for Matrix Function $\mathrm{diag}((u(x) + c(x)) \circ u(x))$

We show the upper bound of $\|G_4\|$.

Since we need to prove the Lipschitz, the effect of q make no difference. The q will be canceled. Thus, we define the terms without having q at all.

Lemma D.9. *If the following conditions hold*

- $G_4 = \mathrm{diag}((u(x) + c(x)) \circ u(x)) - \mathrm{diag}((u(y) + c(y)) \circ u(y))$

Then, we have

$$\|G_4\| \leq 4 \max\{\|u(x)\|_2, \|u(y)\|_2, \|c(x)\|_2, \|c(y)\|_2\} \cdot (\|u(x) - u(y)\|_2 + \|c(x) - c(y)\|_2)$$

Proof. We define

$$\begin{aligned} G_{4,1} :=& \ \mathrm{diag}((u(x) + c(x)) \circ u(x)) - \mathrm{diag}((u(y) + c(y)) \circ u(x)) \\ G_{4,2} :=& \mathrm{diag}((u(y) + c(y)) \circ u(x)) - \mathrm{diag}((u(y) + c(y)) \circ u(y)) \end{aligned}$$

Then we have

$$\begin{aligned} \|G_{4,1}\| =& \ \| \mathrm{diag}((u(x) + c(x)) \circ u(x)) - \mathrm{diag}((u(y) + c(y)) \circ u(x))\| \\ \leq& \ \|(u(x) + c(x) - u(y) - c(y)) \circ u(x)\|_2 \\ \leq& \ \|u(x) + c(x) - u(y) - c(y)\|_2 \cdot \|u(x)\|_2 \\ \leq& \ (\|u(x) - u(y)\|_2 + \|c(x) - c(y)\|_2) \cdot \|u(y)\|_2 \end{aligned}$$

where the first step is due to the definition of $G_{4,1}$, the second step is due to Fact A.5, and the third step is due to Fact A.5 and the last step follows from triangle inequality.

Similarly, we have

$$\begin{aligned} \|G_{4,2}\| =& \ \| \mathrm{diag}((u(y) + c(y)) \circ u(x)) - \mathrm{diag}((u(y) + c(y)) \circ u(y))\| \\ \leq& \ \|(u(y) + c(y)) \circ u(x) - (u(y) + c(y)) \circ u(y)\|_2 \\ \leq& \ (\|u(y)\|_2 + \|c(y)\|_2) \cdot \|u(x) - u(y)\|_2 \end{aligned}$$

where the first step is due to the definition of $G_{4,2}$, the second step is due to Fact A.5, and the third step is due to Fact A.5. □

D.10 Lipschitz Calculations: Step 5. Lipschitz for Matrix Function $\langle c(x), b\rangle \mathrm{diag}(u(x))$

We compute the upper bound of $\|G_5\|$.

Lemma D.10. *If the following conditions hold*

- $G_5 = \langle c(x), b\rangle \mathrm{diag}(u(x)) - \langle c(y), b\rangle \mathrm{diag}(u(y))$

Then, we have

$$\|G_5\| \leq 4 \max\{\|u(x)\|_2, \|u(y)\|_2, \|c(x)\|_2, \|c(y)\|_2\} \cdot \|b\|_2 (\|u(x) - u(y)\|_2 + \|c(x) - c(y)\|_2)$$

Proof. We define

$$\begin{aligned} G_{5,1} :=& \ \langle c(x), b\rangle \mathrm{diag}(u(x)) - \langle c(x), b\rangle \mathrm{diag}(u(y)) \\ G_{5,2} :=& \ \langle c(x), b\rangle \mathrm{diag}(u(y)) - \langle c(y), b\rangle \mathrm{diag}(u(y)) \end{aligned}$$

We can show

$$\begin{aligned} \|G_{5,1}\| =& \ \|\langle c(x), b\rangle \cdot (\mathrm{diag}(u(x)) - \mathrm{diag}(u(y)))\| \\ =& \ |\langle c(x), b\rangle| \cdot \| \mathrm{diag}(u(x)) - \mathrm{diag}(u(y))\| \\ \leq& \ \|c(x)\|_2 \cdot \|b\|_2 \cdot \| \mathrm{diag}(u(x)) - \mathrm{diag}(u(y))\| \\ \leq& \ \|c(x)\|_2 \cdot \|b\|_2 \cdot \|u(x) - u(y)\|_2 \end{aligned}$$

where the first step is due to the definition of $G_{5,1}$, the second step follows from Fact A.6, the second step follows from Fact A.5, and the last step follows from Fact A.5.

Similarly, we have

$$\begin{aligned} \|G_{5,2}\| &= |\langle c(x) - c(y), b \rangle| \cdot \| \operatorname{diag}(u(y)) \| \\ &\leq \|c(x) - c(y)\|_2 \cdot \|b\|_2 \cdot \|u(y)\|_2 \end{aligned}$$

where the first step is due to Fact A.5, the definition of $G_{5,2}$ and simple algebra, and the second follows from Fact A.5 and Fact A.3. □

E LIPSCHITZ WITH RESPECT TO A

In Section E.1, we consider the x case, which is to upper bound $|\alpha(x)^{-1}|$. In Section E.2, we consider the A case, namely computing the upper bound of $|\alpha(A)^{-1}|$. In Section E.3, we analyze the bound for $\|u(A) - u(B)\|_2$. In Section E.4, we prove the bound for $|\alpha(A) - \alpha(B)|$. In Section E.5, we analyze the bound for $\|c(A) - c(B)\|_2$.

E.1 For the x case

In this section, the goal is to bound $|\alpha(x)^{-1}|$. We start from the following definition.

Definition E.1. *We define δ_b be to the vector that satisfies*

$$\|u(x_{t+1}) - \alpha(x_{t+1})b\|_2^2 = \|u(x_t) - \alpha(x_t)(b - \delta_b)\|_2^2$$

Lemma E.2. *We have*

$$\|\delta_b\|_2 \leq |\alpha(x_t)^{-1}| \cdot \|c(x_{t+1}) - c(x_t)\|_2$$

Proof. Similarly as Liu et al. (2023) described, there could be multiple solutions, e.g. 2^n possible solutions

$$u(x_{t+1}) - \alpha(x_{t+1})b = (u(x_t) - \alpha(x_t)(b - \delta_b)) \circ \{-1, +1\}^n$$

The norm of all the solutions are same. Therefore, we can just consider one solution, which is the following solution

$$u(x_{t+1}) - \alpha(x_{t+1})b = u(x_t) - \alpha(x_t)(b - \delta_b)$$

Thus,

$$\begin{aligned} \delta_b &= \alpha(x_t)^{-1}(u(x_{t+1}) - u(x_t) - b(\alpha(x_{t+1}) - \alpha(x_t))) \\ &= \alpha(x_t)^{-1}(c(x_{t+1}) - c(x_t)) \end{aligned}$$

□

We use a tool, which is from Deng et al. (2023a).

Lemma E.3 (Lemma 8.9 in Deng et al. (2023a))**.** *If the following condition hold*

- *Let $\|A\| \leq R$*
- *Let $\|x\|_2 \leq R$*

We have

$$|\alpha(x)^{-1}| \leq \exp(R^2)$$

The proof is standard, we omit the details here.

E.2 For the A case

Here, we bound $|\alpha(A)^{-1}|$.

Definition E.4. *We define δ_b be to the vector that satsifies*

$$\|u(x_{t+1}) - \alpha(x_{t+1})b\|_2^2 = \|u(x_t) - \alpha(x_t)(b - \delta_b)\|_2^2$$

Lemma E.5. *We have*

$$\|\delta_b\|_2 \leq |\alpha(x_t)^{-1}| \cdot \|c(x_{t+1}) - c(x_t)\|_2$$

Lemma E.6 (Lemma 8.9 in Deng et al. (2023a))**.** *If the following points hold*

- *Let $\|A\| \leq R$*
- *Let $\|x\|_2 \leq R$*

We have

$$|\alpha(A)^{-1}| \leq \exp(R^2)$$

E.3 Lipschitz for $u(A)$

We state a tool that directly implies by previous work. The proof is very standard, so we omit the details here.

Lemma E.7 (A variation of Part 1 of Lemma 7.2 in Deng et al. (2023a))**.** *If the following conditions hold*

- *Let $u(A)$ be defined as definition 3.1 with reparamerization by A instead of x.*[2]
- *Let $\|(A - B)x\|_\infty < 0.01$*
- *Let $\|A\|, \|B\| \leq R$, where $A, B \in \mathbb{R}^{n \times d}$*
- *Let $\|x\|_2 \leq R$, where $x \in \mathbb{R}^d$*

then, we have

$$\|u(A) - u(B)\|_2 \leq 2\sqrt{n}R\exp(R^2)\|A - B\|$$

E.4 Lipschitz for $\alpha(A)$

We state a tool which directly implies by previous work. The proof is very standard, so we omit the details here.

Lemma E.8 (A variation of Part 2 of Lemma 7.2 in Deng et al. (2023a))**.** *If the following conditions hold*

- *Let $\alpha(A)$ be defined in Definition 3.4 with reparameterization by A instead of x.*
- *Let $u(A)$ be defined as Definition 3.1 with reparameterization by A instead of x.*

then, we have

$$|\alpha(A) - \alpha(B)| \leq \sqrt{n} \cdot \|u(A) - u(B)\|_2$$

[2] Instead of calling $u(x) = \exp(Ax)$. We call $u(A) = \exp(Ax)$.

E.5 Lipschitz for $c(x)$

In this section, we bound $\|c(A) - c(B)\|_2$.

Lemma E.9 (A variation of Lemma D.4)**.** *If the following conditions hold*

- *Let $c(A)$ be defined as Definition 3.5 with reparametrization by A.*
- *Let $\alpha(A)$ be defined as Definition 3.4 with reparameterization by A.*
- *Let $u(A)$ be defined as Definition 3.1 with reparameterization by A.*
- *Let $b \in \mathbb{R}^n$.*

Then, we have

$$\|c(A) - c(B)\|_2 \leq \|u(A) - u(B)\|_2 + |\alpha(A) - \alpha(B)| \cdot \|b\|_2$$

Proof. We have

$$\begin{aligned} \|c(A) - c(B)\|_2 &= \|(u(A) - \alpha(B) \cdot b) - (u(A) - \alpha(B) \cdot b)\|_2 \\ &\leq \|u(A) - u(B)\|_2 + \|(\alpha(A) - \alpha(B)) \cdot b\|_2 \\ &= \|u(A) - u(B)\|_2 + |\alpha(A) - \alpha(B)| \cdot \|b\|_2 \end{aligned}$$

where the first step comes from how we defined c (see Definition 3.5), the second step is because of the triangle inequality, and the last step follows from simple algebra. □

F MAIN RESULT

In Section F.1, we introduce our algorithm (see Algorithm 1) and use our main Theorem (see Theorem F.1) to analyze its properties, including running time and the output $\widetilde{x}$. In Section F.2, we introduce a corollary which is the application of in-context learning.

F.1 Convergence

Now, we introduce our main algorithm and Theorem.

Theorem F.1. *Given that vectors $b, w \in \mathbb{R}^n$ and a matrix $A \in \mathbb{R}^{n \times d}$, we define x^* as the optimal solution of the following problem*

$$\min_{x \in \mathbb{R}^d} 0.5 \cdot \| \exp(Ax) - \langle \exp(Ax), \mathbf{1}_n \rangle \cdot b \|_2^2 + 0.5 \| \operatorname{diag}(w) A x \|_2^2$$

And then if the conditions as follows hold:

- $R \geq 4$.
- $g(x^*) = \mathbf{0}_d$.
- $\|x^*\|_2 \leq R$.
- $\|A\| \leq R$.
- $\|b\|_2 \leq R$.
- $w_i^2 \geq 100 + l/\sigma_{\min}(A)^2$ *for all* $i \in [n]$
- $M = \exp(O(R^2 + \log n))$.
- *Let accuracy* $\epsilon \in (0, 0.1)$
- *Let failure probability* $\delta \in (0, 0.1)$

- *Let x_0 denote an initial point for which it holds that $M\|x_0 - x^*\|_2 \leq 0.1l$.*

Then there exists a randomized algorithm (Algorithm 1) such that, with probability at least $1 - \delta$,

- *it runs $T = \log(\|x_0 - x^*\|_2/\epsilon)$ iterations*
- *spends*

$$O((\mathrm{nnz}(A) + d^\omega) \cdot \mathrm{poly}(\log(n/\delta)).$$

- *outputs a vector $\widetilde{x} \in \mathbb{R}^d$ such that*

$$\|\widetilde{x} - x^*\|_2 \leq \epsilon$$

Here ω denote the exponent of matrix multiplication. Currently $\omega \approx 2.373$ (Williams, 2012; Le Gall, 2014; Alman and Williams, 2021).

Proof. **Proof of Hessian is PD.**

We can obtain this conclusion from Lemma C.4.

Proof of Hessian is Lipschitz.

The proof is due to Lemma D.1.

Proof of Cost per iteration.

This follows from Lemma 6.4.

Proof of Convergence per Iteration.

By Lemma 6.5, we have

$$\|x_k - x^*\|_2 \leq 0.4 \cdot \|x_{k-1} - x^*\|_2.$$

Proof of Number of Iterations.

After T iterations, we have

$$\|x_T - x^*\|_2 \leq 0.4^T \cdot \|x_0 - x^*\|_2$$

By choice of T, we get the desired bound. The failure probability is following from union bound over T iterations.

□

F.2 Application to In-context Learning

In this section, we introduce the application to in-context learning.

Corollary F.2 (Bounded shift for Learning in-context)**.** *If the following conditions hold*

- *Let $A \in \mathbb{R}^{n \times d}$.*
- *Let $b \in \mathbb{R}^n$.*
- *$\|A\| \leq R$.*
- *Let $\|x\|_2 \leq R$.*
- *$\|A(x_{t+1} - x_t)\|_\infty < 0.01$.*
- *$\|(A_{t+1} - A_t)x\|_\infty < 0.01$.*

- *Let* $R \geq 4$.
- *Let* $M := \exp(O(R^2 + \log n))$.
- *Let* $u(x) \in \{\exp(Ax), \cosh(Ax), \sinh(Ax)\}$.

We consider the rescaled softmax regression (Definition 1.2) problem

$$\min_{x \in \mathbb{R}^d} \|u(x) - \alpha(x)b\|_2.$$

- **Part 1.** *If we move the* x_t *to* x_{t+1}*, then we're solving a new rescaled softmax regression problem with*

$$\min_{x \in \mathbb{R}^d} \|u(x) - \alpha(x)\widetilde{b}\|_2$$

where

$$\|\widetilde{b} - b\|_2 \leq M \cdot \|x_{t+1} - x_t\|_2$$

- **Part 2.** *If we move the* A_t *to* A_{t+1}*, then we're solving a new rescaled softmax regression with*

$$\min_{x} \|u(x) - \alpha(x)\widehat{b}\|_2$$

where

$$\|\widehat{b} - b\|_2 \leq M \cdot \|A_{t+1} - A_t\|$$

Proof. **Proof of Part 1.** The proof follows from by combining Lemma E.2, Lemma E.3, Lemma D.2, Lemma D.3, Lemma D.4.

Proof of Part 2. The proof follows from by combining Lemma E.5, Lemma E.6, Lemma E.7, Lemma E.8, Lemma E.9. □

G MORE RELATED WORKS

One of the important ideas in this work is to use sketching to speed up the iterative algorithm in optimization. The fundamental concept of sketching is to decompose a large input matrix into a significantly smaller sketching matrix, but this sketching matrix retains the important characteristics of the original matrix. Therefore, the algorithms can only operate on this smaller matrix instead of the unwieldy original one, resulting in a substantial reduction in computational time. There are numerous prior studies that have devised sketching algorithms with robust theoretical assurances. For example, the Johnson-Lindenstrauss lemma in Johnson and Lindenstrauss (1984) demonstrates that, in certain high-dimensional spaces, projecting points onto lower-dimensional subspaces can preserve pairwise distances between the points. This property supports the development of faster algorithms for tasks such as nearest neighbor search. Furthermore, as elucidated in Ailon and Chazelle (2006), the Fast Johnson-Lindenstrauss Transform (FJLT) introduces a specific family of structured random projections that can be applied to an input matrix in time proportional to its sparsity.

There are two ways to utilize the sketching matrices. The first way is known as sketch-and-solve, which uses sketching a predetermined number of times. This may lead to faster algorithms in several domains, like in the linear regression (Nelson and Nguyên, 2013; Clarkson and Woodruff, 2017) and low-rank approximation (Song et al., 2025b), in column subset selection (Sobczyk and Gallopoulos, 2022; Song et al., 2019; Jiang et al., 2021a), where, with provable approximation guarantees, the column selection can be speed up by sketching the data matrix, in kernel methods (Laparra et al., 2015), where the sketching methods can be applied to large kernel matrices approximation, in tensor method (Avron et al., 2014; Pagh, 2013; Pham and Pagh, 2013; Diao et al., 2018; Deng et al., 2023c), tensors can be compressed down to much smaller core tensors. Additionally, it can be employed to determine the optimal bound as demonstrated in Song et al. (2023e) and to design an efficient method for training neural networks, as shown in Qin et al. (2023a). Moreover, a recent work (Song et al., 2023f)

has applied the sketching method to the quantum algorithm, which solves the linear regression problem. Finally, it has been used to study the matrix completion problem in Gu et al. (2024).

The second way is known as iterate-and-sketch, which is applied in each iteration of the optimization algorithm and establishes a robust analysis framework. It has been widely used in numerous important tasks such as linear programming (Jiang et al., 2021b; Song and Yu, 2021; Liu et al., 2023; Cohen et al., 2019; Gu and Song, 2022), empirical risk minimization (Lee et al., 2019; Qin et al., 2023b), John Ellipsoid algorithm (Song et al., 2022b), online weighted matching problem (Song et al., 2025a), the Frank-Wolfe algorithm (Song et al., 2022a; Xu et al., 2021), semidefinite programming (Gu and Song, 2022; Song et al., 2023d), federated learning (Song et al., 2023b; Bian et al., 2023), attention approximation (Gao et al., 2023c,b), k means clustering (Liang et al., 2022), discrepancy algorithm (Deng et al., 2022), training over-parametrized neural network (Song et al., 2021; Alman et al., 2022; Zhang, 2022), rational database (Qin et al., 2022), matrix sensing (Qin et al., 2023c).

Other theoretical machine learning works focus on LLMs efficiency (Zhang et al., 2025; Xu et al., 2024b,a; Li et al., 2024a; Liang et al., 2024e; Shi et al., 2024b; Liang et al., 2024c; Wang et al., 2024; Liang et al., 2024d; Li et al., 2024b; Liang et al., 2024b; Shi et al., 2024a; Li et al., 2024d; Ke et al., 2024; Chen et al., 2024a,b; Li et al., 2024c; Liang et al., 2025; Chen et al., 2025a; Li et al., 2025a; Ke et al., 2025; Chen et al., 2025b; Li et al., 2025b; Chen et al., 2025c; Cao et al., 2025a; Chang et al., 2024; Zhang et al., 2021), reinforcement learning (Zhang et al., 2024, 2023; Li et al., 2023c; Li and Yang, 2024; Liu et al., 2024), circuit complexity (Chen et al., 2024c; Li et al., 2025c), fairness analysis (Cao et al., 2025b), and differential privacy (Aliakbarpour et al., 2024b,a, 2023; Asoodeh et al., 2021; Aliakbarpour et al., 2019, 2018; Li et al., 2025d).