Automatic Rank Determination for Low-Rank Adaptation via Submodular Function Maximization *

Yihang Gao[1], **Vincent Y. F. Tan**[2]

[1]Department of Mathematics, National University of Singapore
[2]Department of Mathematics and Department of Electrical and Computer Engineering, National University of Singapore
gaoyh@nus.edu.sg, vtan@nus.edu.sg

Abstract

In this paper, we propose SubLoRA, a rank determination method for Low-Rank Adaptation (LoRA) based on submodular function maximization. In contrast to prior approaches, such as AdaLoRA, that rely on first-order (linearized) approximations of the loss function, SubLoRA utilizes second-order information to capture the potentially complex loss landscape by incorporating the Hessian matrix. We show that the linearization becomes inaccurate and ill-conditioned when the LoRA parameters have been well optimized, motivating the need for a more reliable and nuanced second-order formulation. To this end, we reformulate the rank determination problem as a combinatorial optimization problem with a quadratic objective. However, solving this problem exactly is NP-hard in general. To overcome the computational challenge, we introduce a submodular function maximization framework and devise a greedy algorithm with approximation guarantees. We derive a sufficient and necessary condition under which the rank-determination objective becomes submodular, and construct a closed-form projection of the Hessian matrix that satisfies this condition while maintaining computational efficiency. Our method combines solid theoretical foundations, second-order accuracy, and practical computational efficiency. We further extend SubLoRA to a joint optimization setting, alternating between LoRA parameter updates and rank determination under a rank budget constraint. Extensive experiments on fine-tuning physics-informed neural networks (PINNs) for solving partial differential equations (PDEs) demonstrate the effectiveness of our approach. Results show that SubLoRA outperforms existing methods in both rank determination and joint training performance.

Introduction

Low-rank adaptation (LoRA) (Hu et al. 2022) has demonstrated promising performance as an efficient fine-tuning technique across a wide range of domains, including large language models (Zhang et al. 2024; Liu et al. 2024; Valipour et al. 2023; Ding et al. 2023), vision models (Chen et al. 2022; Liang et al. 2025; Jia et al. 2022; Zhang, Zhou, and Liu 2024), and physics-informed neural networks (Majumdar et al. 2023; Wang et al. 2025). Rather than updating all model parameters during fine-tuning, LoRA introduces a parameter-efficient strategy by applying low-rank decomposition to selected weight matrices. A broader review of recent developments in LoRA, including algorithmic enhancements, theoretical analyses, and various applications, can be found in Zeng and Lee (2024); Jang, Lee, and Ryu (2024); Wang et al. (2024); Hayou, Ghosh, and Yu (2024); Han et al. (2024); Mao et al. (2025); Dettmers et al. (2023); Schmirler, Heinzinger, and Rost (2024); Kim, Kim, and Ryu (2025).

However, rank determination in LoRA remains an underexplored but critical aspect of its effectiveness. Despite its importance, relatively few works have addressed this problem in depth. Most existing methods rely on singular value decomposition (SVD) to factorize parameter updates, motivated by the fundamental relationship between singular values and matrix rank. A computationally efficient method considers the importance of each singular value through loss sensitivity analysis. AdaLoRA (Zhang et al. 2023), for instance, linearizes the loss function with respect to the singular values of the parameter updates. The first-order approximation provides a sensitivity measure that guides adaptive pruning by identifying which components of the singular values contribute significantly to the loss.

Our work builds upon the direction of AdaLoRA by determining the rank allocation for each LoRA layer based on the importance of the singular values in the parameter updates. However, we observe that first-order linearization suffers from poor approximation accuracy, particularly when the LoRA fine-tuning reaches a stationary point. Motivated by this limitation, we propose to adopt a second-order expansion of the objective by incorporating Hessian information, enabling a more accurate approximation that captures the curvature of the loss landscape. Although the second-order formulation offers a better geometric understanding of the objective, it introduces an NP-hard optimization challenge, unlike the linearized formulation, which admits closed-form solutions. To address this, we draw inspiration from the theoretical guarantees of the greedy algorithm in submodular function maximization. We introduce a Hessian projection that transforms the original set-valued quadratic objective into a submodular function. We prove that this projection is well-defined, admits a closed-form solution, and is computationally efficient. As a result, the projected second-order objective becomes submodular, allowing us to apply greedy algorithms to obtain provably near-optimal solutions. We ap-

*This paper appears in **MATH4AI Workshop@AAAI 2026**.

ply the proposed technique to LoRA fine-tuning in physics-informed neural networks (PINNs) under rank budget constraints, where the loss landscape is especially complex and curvature information is valuable.

Preliminaries

Notation

In this paper, we use bold lowercase letters (e.g., $\boldsymbol{x}$) to denote vectors and bold uppercase letters (e.g., $\boldsymbol{A}$) to represent matrices. Scalars are represented using regular (non-bold) lowercase letters (e.g., a). The operator $\mathrm{diag}(\cdot)$ constructs a square diagonal matrix from a given vector. Specifically, for $\boldsymbol{a} \in \mathbb{R}^n$, we define $\boldsymbol{A} = \mathrm{diag}(\boldsymbol{a}) \in \mathbb{R}^{n\times n}$ such that $A_{i,i} = a_i$ and $A_{i,j} = 0$ for $i \neq j$. We denote the set $\{1, 2, \cdots, n\}$ by $[n]$. Calligraphic letters (e.g., $\mathcal{S}$) are used to denote sets, and $|\mathcal{S}|$ denotes the cardinality (i.e., the number of elements) of set $\mathcal{S}$. For a given vector $\boldsymbol{a} \in \mathbb{R}^n$ and a set $\mathcal{S} \subseteq [n]$, we define $[\boldsymbol{a}]_{\mathcal{S}} \in \mathbb{R}^{|\mathcal{S}|}$ as the subvector of $\boldsymbol{a}$ that contains only the entries indexed by $\mathcal{S}$. Similarly, for a matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, we define $[\boldsymbol{A}]_{\mathcal{S}} \in \mathbb{R}^{|\mathcal{S}|\times|\mathcal{S}|}$ as the principal submatrix of $\boldsymbol{A}$ formed by retaining only the rows and columns indexed by $\mathcal{S}$. For a matrix $\boldsymbol{B} \in \mathbb{R}^{m\times n}$, We define $(\boldsymbol{B})_{\mathcal{S}} \in \mathbb{R}^{m\times|\mathcal{S}|}$ as the column submatrix of $\boldsymbol{B}$, constructed by selecting the columns indexed by $\mathcal{S}$.

Submodular Function

We consider a set-valued function $f : 2^{\Omega} \to \mathbb{R}$, where 2^{Ω} denotes the power set of the finite ground set Ω. The function f is said to be submodular if it satisfies the following diminishing returns property:

Definition 1 (See (Krause and Golovin 2014; Fujishige 2005)) *A set-valued function $f : 2^{\Omega} \to \mathbb{R}$ is submodular if for any sets $\mathcal{X}, \mathcal{Y} \subseteq \Omega$ with $\mathcal{X} \subseteq \mathcal{Y}$ and every element $x \in \Omega \setminus \mathcal{Y}$, the following holds:*

$$f(\mathcal{X} \cup \{x\}) - f(\mathcal{X}) \geq f(\mathcal{Y} \cup \{x\}) - f(\mathcal{Y}).$$

Equivalently, submodularity can be characterized by the following inequality:

$$f(\mathcal{X}) + f(\mathcal{Y}) \geq f(\mathcal{X} \cup \mathcal{Y}) + f(\mathcal{X} \cap \mathcal{Y}),$$

for any $\mathcal{X}, \mathcal{Y} \subseteq \Omega$.

Submodular functions play a crucial role in combinatorial optimization, as convex and concave functions do in continuous optimization.

Definition 2 *A set-valued function f is monotone if for all $\mathcal{X} \subseteq \mathcal{Y} \subseteq \Omega$, we have $f(\mathcal{X}) \leq f(\mathcal{Y})$.*

Low Rank Adaptation

In transfer learning, the model parameters $\boldsymbol{W}_{\text{ft}}$ are fine-tuned based on pre-trained weights $\boldsymbol{W}_{\text{pt}}$ as follows:

$$\boldsymbol{W}_{\text{ft}} = \boldsymbol{W}_{\text{pt}} + \Delta\boldsymbol{W},$$

where $\boldsymbol{W}_{\text{pt}}, \boldsymbol{W}_{\text{ft}}, \Delta\boldsymbol{W} \in \mathbb{R}^{n_2\times n_1}$. Standard fine-tuning updates the entire parameter matrix $\boldsymbol{W}_{\text{pt}}$ by $\Delta\boldsymbol{W}$, providing full flexibility but at the cost of significant computational and memory overhead. LoRA (Hu et al. 2022) addresses this inefficiency by imposing a low-rank structure on the update matrix $\Delta\boldsymbol{W}$, reducing both parameter count and computational cost. Specifically, LoRA parameterizes the update $\Delta\boldsymbol{W}$ by the Burer–Monteiro factorization:

$$\Delta\boldsymbol{W} = \boldsymbol{B}\boldsymbol{A},$$

where $\boldsymbol{A} \in \mathbb{R}^{r\times n_1}$ and $\boldsymbol{B} \in \mathbb{R}^{n_2\times r}$, and $r \ll \min\{n_1, n_2\}$. Instead of optimizing the full matrix $\Delta\boldsymbol{W}$, LoRA focuses on training the smaller matrices $\boldsymbol{A}$ and $\boldsymbol{B}$, thereby reducing the number of parameters and computational complexity.

Rank Determination of LoRA

For LoRA-based fine-tuning of a multi-layer neural network $\phi(\cdot;\Theta)$, the model parameters are denoted as $\Theta := \{\boldsymbol{W}_1, \boldsymbol{W}_2, \cdots, \boldsymbol{W}_L\}$, where each layer ℓ follows the update rule:

$$\boldsymbol{W}_{\text{ft},\ell} = \boldsymbol{W}_{\text{pt},\ell} + \boldsymbol{B}_\ell\boldsymbol{A}_\ell,$$

where $\boldsymbol{W}_{\text{ft},\ell} \in \mathbb{R}^{n_{\ell+1}\times n_\ell}$ denotes the fine-tuned parameters, $\boldsymbol{W}_{\text{pt},\ell} \in \mathbb{R}^{n_{\ell+1}\times n_\ell}$ the pre-trained parameters, $\boldsymbol{A}_\ell \in \mathbb{R}^{r_\ell\times n_\ell}$, and $\boldsymbol{B}_\ell \in \mathbb{R}^{n_{\ell+1}\times r_\ell}$ the LoRA components. Here, n_ℓ represents the width of the ℓ-th layer, and r_ℓ is the assigned LoRA rank. Given a global rank budget b (i.e., $\sum_{\ell=1}^{L} r_\ell \leq b$)), a key challenge is deciding how to allocate the individual ranks r_ℓ across layers.

An alternative to the Burer–Monteiro factorization for rank determination in LoRA is to consider the singular value decomposition (SVD) of the parameter updates. Specifically, the update $\Delta\boldsymbol{W}$ is factorized by

$$\boldsymbol{W}_{\text{ft},\ell} = \boldsymbol{W}_{\text{pt},\ell} + \boldsymbol{U}_\ell\boldsymbol{\Sigma}_\ell\boldsymbol{V}_\ell, \tag{1}$$

where $\boldsymbol{U}_\ell \in \mathbb{R}^{n_{\ell+1}\times r_\ell}$, $\boldsymbol{V}_\ell \in \mathbb{R}^{r_\ell\times n_\ell}$, and $\boldsymbol{\Sigma}_\ell := \mathrm{diag}(\boldsymbol{\sigma}_\ell) \in \mathbb{R}^{r_\ell\times r_\ell}$ is a diagonal matrix with $\boldsymbol{\sigma}_\ell \in \mathbb{R}^{r_\ell}$. In this formulation, rank determination reduces to identifying which entries of $\boldsymbol{\sigma}_\ell$ are effectively zero, as they correspond to components that do not contribute to the rank.

Zhang et al. (2023) regards $\boldsymbol{\sigma}_\ell$ as a deterministic parameter and performs pruning based on the dynamics of the loss function after fine-tuning. For notational simplicity, we denote the pre-trained model parameters as $\Theta_{\text{pt}} := \{\boldsymbol{W}_{\text{pt},1}, \cdots, \boldsymbol{W}_{\text{pt},L}\}$, and the fine-tuned model parameters as $\Theta_{\text{ft}} := \{\boldsymbol{W}_{\text{ft},1}, \cdots, \boldsymbol{W}_{\text{ft},L}\}$ obtained by Equation (1). The corresponding LoRA components are denoted by $\Theta_{\text{LoRA}} := \{\boldsymbol{U}_\ell, \boldsymbol{V}_\ell, \boldsymbol{\sigma}_\ell\}_{\ell\in[L]}$. For convenience, we define the operator $\oplus$ such that the fine-tuned parameters can be written as $\Theta_{\text{ft}} := \Theta_{\text{pt}} \oplus \Theta_{\text{LoRA}}$ in accordance with the decomposition in Equation (1). LoRA fine-tuning aims to minimize the following objective function:

$$\mathcal{L}_{\text{ft}}(\Theta_{\text{LoRA}}) = \frac{1}{N}\sum_{i=1}^{N} \mathrm{Loss}\left(\phi\left(\boldsymbol{x}_i; \Theta_{\text{pt}} \oplus \Theta_{\text{LoRA}}\right); \boldsymbol{y}_i\right), \tag{2}$$

where $\mathrm{Loss}(\cdot;\cdot)$ represents the loss function used to compare predictions and ground-truth labels, and $\{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^{N}$ denotes the training dataset. The trainable parameters in this setting are those within Θ_{LoRA}. Under this setup, the rank determination problem seeks to prune the diagonal entries of $\boldsymbol{\Sigma}_\ell$ (equivalently, the entries of $\boldsymbol{\sigma}_\ell$), such that the total number

of nonzero singular values across all layers remains within a given budget b, while maintaining competitive model performance. The key challenge lies in determining both (i) how to allocate the rank budget across layers and (ii) which components of $\boldsymbol{\sigma}_\ell$ should be retained. To formalize this, we define:

$$\mathcal{L}\left(\{\mathcal{S}_\ell\}_{\ell\in[L]}\right) := \mathcal{L}_{\text{ft}}\left(\left[\Theta_{\text{LoRA}}\right]_{\{\mathcal{S}_\ell\}_{\ell\in[L]}}\right),$$

where $\mathcal{S}_\ell \subseteq [r_\ell]$ is the set of indices of the retained entries at the ℓ-th layer, and $\left[\Theta_{\text{LoRA}}\right]_{\{\mathcal{S}_\ell\}_{\ell\in[L]}} := \{(\boldsymbol{U}_\ell)_{\mathcal{S}_\ell}, (\boldsymbol{V}_\ell)_{\mathcal{S}_\ell}, [\boldsymbol{\sigma}_\ell]_{\mathcal{S}_\ell}\}_{\ell\in[L]}$. The goal is to solve the following combinatorial optimization problem with cardinality constraints:

$$\min_{\{\mathcal{S}_\ell\}_{\ell\in[L]}} \mathcal{L}\left(\{\mathcal{S}_\ell\}_{\ell\in[L]}\right), \quad \text{s.t.} \quad \sum_{\ell=1}^{L} |\mathcal{S}_\ell| \le b, \tag{3}$$

where b denotes the total rank budget for the whole model. Discarding an entry of $\boldsymbol{\sigma}_\ell$ is equivalent to setting its corresponding singular value to zero. Therefore, the optimization seeks to prune singular values that contribute the least to performance, such that the loss increase is minimized. However, Equation (3) is a combinatorial optimization problem over $\sum_{\ell=1}^{L} r_\ell$ binary decision variables. Solving such a problem exactly is NP-hard and computationally intractable in general.

For analytical insight, we decompose the pruning procedure as:

$$\mathcal{L}\left(\{\mathcal{S}_\ell\}_{\ell\in[L]}\right) - \mathcal{L}_{\text{ft}}\left(\Theta_{\text{LoRA}}\right) + \mathcal{L}_{\text{ft}}\left(\Theta_{\text{LoRA}}\right). \tag{4}$$

where the first two terms reflect the change in loss due to pruning, and the third term is the loss of the unpruned fine-tuned model. Note that $\mathcal{L}_{\text{ft}}\left(\Theta_{\text{LoRA}}\right)$ is a fixed quantity after the fine-tuning stage. Previous work has adopted a first-order (linear) relaxation of the loss difference $\mathcal{L}\left(\{\mathcal{S}_\ell\}_{\ell\in[L]}\right) - \mathcal{L}_{\text{ft}}\left(\Theta_{\text{LoRA}}\right)$, resulting in the following approximation:

$$\begin{aligned}
&\mathcal{L}\left(\{\mathcal{S}_\ell\}_{\ell\in[L]}\right) - \mathcal{L}_{\text{ft}}\left(\Theta_{\text{LoRA}}\right) \\
&= \mathcal{L}_{\text{ft}}\left(\left[\Theta_{\text{LoRA}}\right]_{\{\mathcal{S}_\ell\}_{\ell\in[L]}}\right) - \mathcal{L}_{\text{ft}}\left(\Theta_{\text{LoRA}}\right) \\
&\approx \sum_{\ell=1}^{L} \left\langle \nabla_{\boldsymbol{\sigma}_\ell} \mathcal{L}_{\text{ft}}\left(\Theta_{\text{LoRA}}\right), [\boldsymbol{\sigma}_\ell]_{\mathcal{S}_\ell} - \boldsymbol{\sigma}_\ell \right\rangle \\
&= \sum_{\ell=1}^{L} \left\langle \left[\nabla_{\boldsymbol{\sigma}_\ell} \mathcal{L}_{\text{ft}}\left(\Theta_{\text{LoRA}}\right)\right]_{\mathcal{S}_\ell^-}, -[\boldsymbol{\sigma}_\ell]_{\mathcal{S}_\ell^-} \right\rangle,
\end{aligned}$$

where $\mathcal{S}_\ell^- := [r_\ell] \setminus \mathcal{S}_\ell$ denotes the complement of the selected indices. This leads to the following surrogate cardinality-constrained combinatorial optimization problem:

$$\begin{aligned}
\min_{\{\mathcal{S}_\ell\}_{\ell\in[L]}} & \sum_{\ell=1}^{L} \left\langle \left[\nabla_{\boldsymbol{\sigma}_\ell} \mathcal{L}_{\text{ft}}\left(\Theta_{\text{LoRA}}\right)\right]_{\mathcal{S}_\ell^-}, -[\boldsymbol{\sigma}_\ell]_{\mathcal{S}_\ell^-} \right\rangle, \\
\text{s.t.} \quad & \sum_{\ell=1}^{L} |\mathcal{S}_\ell| \le b.
\end{aligned} \tag{5}$$

Zhang et al. (2023) proposed AdaLoRA, which prunes singular values based on their estimated sensitivities. Their method builds upon the linear relaxation in Equation (5), with the additional use of elementwise absolute values in the objective to prioritize components with large magnitude impact.

The Proposed Method

Motivation

The first-order expansion of the loss function used in Equation (5) may not be a reliable or effective approximation for rank determination. Here, we elaborate on its limitations in detail. The rank determination typically involves two stages. In the first stage, we fine-tune the model using LoRA with overestimated ranks based on limited prior knowledge. In the second stage, we prune some less important singular values to fit within a total rank budget. Our focus is specifically on the second stage that selects singular values whose removal has less significant impact on performance. Suppose that the LoRA fine-tuning has been sufficiently optimized and the resulting parameters have reached a stationary point Θ_{LoRA} of the objective in Equation (2). In this case, we have

$$\nabla_{\boldsymbol{\sigma}_\ell} \mathcal{L}_{\text{ft}}\left(\Theta_{\text{LoRA}}\right) = \mathbf{0},$$

which implies that the first-order expansion in Equation (5) evaluates to zero. Consequently, the linearized objective provides no meaningful signal for pruning, making the optimization problem ill-posed.

To further illustrate this issue, let us consider a simple toy example: $\mathcal{L}_{\text{ft}}(\sigma_1, \sigma_2) = (\sigma_1 - \sigma_2 + 1)^2$. Also consider the point $(\sigma_1, \sigma_2) = (1, 2.1)$ which is close to the stationary point $(1, 2)$. In this case, it would be more appropriate to prune σ_1, as $\mathcal{L}_{\text{ft}}(0, 2.1) = 1.1^2$ and $\mathcal{L}_{\text{ft}}(1, 0) = 4$. However, the gradient at $(\sigma_1, \sigma_2) = (1, 2.1)$ is $\nabla_{\boldsymbol{\sigma}} \mathcal{L}(\sigma_1, \sigma_2) = (-0.2, 0.2)$, which would mislead the linear relaxation in Equation (5) into pruning σ_2. Now consider a slight perturbation of the original point $(\sigma_1, \sigma_2) = (1, 2.1)$ to $(\sigma_1, \sigma_2) = (1.1, 2)$, where the gradient becomes positive for σ_1, and the linear relaxation would correctly prune σ_1. This demonstrates that small variations near stationary points can drastically alter the decision of which singular value to prune, despite the fact that the underlying model behavior is essentially unchanged. This sensitivity arises because the linearization overemphasizes singular values that deviate more from optimality, rather than those that contribute most meaningfully to the loss.

Hessian-Guided Rank Determination

To address the limitations of the first-order approximation discussed earlier, we propose adopting a second-order (quadratic) expansion of the loss function. This approach provides a more accurate approximation of the objective and avoids the degeneracy that arises when the fine-tuning step reaches near-stationarity. Let $\boldsymbol{\sigma} \in \mathbb{R}^{\sum_{\ell=1}^{L} r_\ell}$ denote the concatenation of all singular value vectors $\boldsymbol{\sigma}_\ell$ across layers $\ell \in [L]$. We define $\mathcal{S} \subseteq \left[\sum_{\ell=1}^{L} r_\ell\right]$ as the set of indices corresponding to the singular values $\boldsymbol{\sigma}$ of the whole model we choose to retain. Recall that the rank-pruned objective

difference can be approximated using a second-order Taylor expansion:

$$\begin{aligned}
&\mathcal{L}\left(\{\mathcal{S}_\ell\}_{\ell\in[L]}\right) - \mathcal{L}_{\text{ft}}\left(\Theta_{\text{LoRA}}\right) \\
&= \mathcal{L}_{\text{ft}}\left(\left[\Theta_{\text{LoRA}}\right]_{\{\mathcal{S}_\ell\}_{\ell\in[L]}}\right) - \mathcal{L}_{\text{ft}}\left(\Theta_{\text{LoRA}}\right) \\
&\approx \left\langle \nabla_{\boldsymbol{\sigma}} \mathcal{L}_{\text{ft}}\left(\Theta_{\text{LoRA}}\right), [\boldsymbol{\sigma}]_{\mathcal{S}} - \boldsymbol{\sigma} \right\rangle \\
&\quad + \frac{1}{2}\left([\boldsymbol{\sigma}]_{\mathcal{S}} - \boldsymbol{\sigma}\right)^\top \nabla^2_{\boldsymbol{\sigma}} \mathcal{L}_{\text{ft}}\left(\Theta_{\text{LoRA}}\right)\left([\boldsymbol{\sigma}]_{\mathcal{S}} - \boldsymbol{\sigma}\right) \\
&= \left\langle \left[\nabla_{\boldsymbol{\sigma}} \mathcal{L}_{\text{ft}}\left(\Theta_{\text{LoRA}}\right)\right]_{\mathcal{S}^-}, -[\boldsymbol{\sigma}]_{\mathcal{S}^-} \right\rangle \\
&\quad + \frac{1}{2}[\boldsymbol{\sigma}]^\top_{\mathcal{S}^-}\left[\nabla^2_{\boldsymbol{\sigma}} \mathcal{L}_{\text{ft}}\left(\Theta_{\text{LoRA}}\right)\right]_{\mathcal{S}^-}[\boldsymbol{\sigma}]_{\mathcal{S}^-},
\end{aligned}$$

where $\nabla^2_{\boldsymbol{\sigma}} \mathcal{L}_{\text{ft}}\left(\Theta_{\text{LoRA}}\right)$ is the Hessian matrix of the fine-tuning objective function with respect to singular values $\boldsymbol{\sigma}$ and $\mathcal{S}^- := \left[\sum_{\ell=1}^{L} r_\ell\right] \setminus \mathcal{S}$ represents the complement of $\mathcal{S}$. Then, we reformulate the approximation of Equation (3) into the following combinatorial problem with quadratic objective:

$$\begin{aligned}
\min_{\mathcal{S}\subseteq\left[\sum_{\ell=1}^{L} r_\ell\right]} \; & \left\langle \left[\nabla_{\boldsymbol{\sigma}} \mathcal{L}_{\text{ft}}\left(\Theta_{\text{LoRA}}\right)\right]_{\mathcal{S}^-}, -[\boldsymbol{\sigma}]_{\mathcal{S}^-} \right\rangle \\
& + \frac{1}{2}[\boldsymbol{\sigma}]^\top_{\mathcal{S}^-}\left[\nabla^2_{\boldsymbol{\sigma}} \mathcal{L}_{\text{ft}}\left(\Theta_{\text{LoRA}}\right)\right]_{\mathcal{S}^-}[\boldsymbol{\sigma}]_{\mathcal{S}^-}, \\
\text{s.t.} \quad & |\mathcal{S}| \le b.
\end{aligned} \tag{6}$$

Beyond interpreting Equation (6) as a second-order Taylor expansion of the nonlinear objective function, it can also be understood from a geometric perspective. When the fine-tuned parameters are at or near a local minimum, the gradient vanishes $\nabla_{\boldsymbol{\sigma}} \mathcal{L}_{\text{ft}} = \mathbf{0}$, and the Hessian matrix $\nabla^2_{\boldsymbol{\sigma}} \mathcal{L}_{\text{ft}}$ is positive semi-definite. Under this condition, Equation (6) simplifies to the problem of minimizing the quadratic form $\|\boldsymbol{\sigma}\|^2_{\nabla^2_{\boldsymbol{\sigma}} \mathcal{L}_{\text{ft}}} := \boldsymbol{\sigma}^\top \nabla^2_{\boldsymbol{\sigma}} \mathcal{L}_{\text{ft}} \boldsymbol{\sigma}$, which represents the magnitude of $\boldsymbol{\sigma}$ measured in the norm induced by the Hessian.

However, the quadratic formulation in Equation (6), while more intuitively convincing and theoretically sound, remains NP-hard due to its combinatorial and non-separable structure. A natural relaxation is to consider only the *diagonal* elements of the Hessian matrix (i.e., considering $\boldsymbol{D} := \operatorname{diag}\left(\nabla^2_{\boldsymbol{\sigma}} \mathcal{L}_{\text{ft}}\left(\Theta_{\text{LoRA}}\right)\right)$), rather than the full second-order structure.

While the diagonal relaxation makes the problem in Equation (6) tractable, it comes at the cost of discarding much of the Hessian information. This simplification may lead to suboptimal decisions and performance degradation for rank determination. It raises a natural question: *can we strike a better balance between retaining second-order information and ensuring computational tractability?* We address this challenge by proposing an intermediate relaxation, one that preserves more of the Hessian structure than the diagonal approximation, while still allowing for efficient optimization. To achieve this, we reformulate the objective as a submodular function maximization problem, enabling the use of greedy algorithms with provable theoretical approximation guarantees.

Algorithm 1: Greedy Algorithm

1: Initialize $\mathcal{S} \leftarrow \emptyset$
2: **for** $i = 1$ to b **do**
3: Select $e \in [r] \setminus \mathcal{S}$ that maximizes the marginal gain:

$$e = \arg\max_{j\in[r]\setminus\mathcal{S}} f(\mathcal{S}\cup\{j\}) - f(\mathcal{S})$$

4: Update the solution: $\mathcal{S} \leftarrow \mathcal{S}\cup\{e\}$
5: **end for**
6: **return** $\mathcal{S}$

Submodular Function Maximization

Motivated by the well-developed theoretical properties of submodular functions, we propose to adopt the greedy algorithm as an efficient solver for approximately solving Equation (6). The following theorem summarizes the classical theoretical guarantees for greedy algorithms applied to cardinality-constrained submodular maximization.

Theorem 1 (See Krause and Golovin (2014); Fujishige (2005)) *Let $f : 2^{[r]} \to \mathbb{R}$ be a non-negative objective function. Consider the following cardinality-constrained combinatorial optimization problem:*

$$\begin{aligned}
\max_{\mathcal{S}\subseteq[r]} \quad & f(\mathcal{S}), \\
\textit{s.t.} \quad & |\mathcal{S}| \le b,
\end{aligned}$$

If f is submodular and monotone, then the solution $\mathcal{S}^\#$ obtained from the greedy algorithm (Algorithm 1) satisfies:

$$f(\mathcal{S}^\#) \ge (1 - 1/e)\cdot f(\mathcal{S}^*),$$

where $\mathcal{S}^*$ denotes an optimal solution.

The above results imply that the greedy algorithm, which is computationally practical, achieves a $(1 - 1/e)$-approximation ratio to the optimal solution when the objective function is submodular and monotone. The submodularity property imposes useful structure and regularity on the objective function, ensuring that the solutions obtained from greedy algorithms are provably close to optimal in terms of objective value.

The following theorem provides a key condition under which a set-valued quadratic function is submodular, thus enabling the use of greedy algorithms with approximation guarantees.

Theorem 2 *Let $\boldsymbol{c} \in \mathbb{R}^r$, $\boldsymbol{x} \in \mathbb{R}^r$, and $\boldsymbol{H} \in \mathbb{S}^r$ be given vectors and a symmetric matrix, respectively. Consider the set-valued function:*

$$f(\mathcal{S}) = [\boldsymbol{c}]^\top_{\mathcal{S}^-}[\boldsymbol{x}]_{\mathcal{S}^-} + \frac{1}{2}[\boldsymbol{x}]^\top_{\mathcal{S}^-}[\boldsymbol{H}]_{\mathcal{S}^-}[\boldsymbol{x}]_{\mathcal{S}^-},$$

where $\mathcal{S} \subseteq [r]$ and $\mathcal{S}^- = [r] \setminus \mathcal{S}$. Then f is submodular if and only if

$$H_{i,j} x_i x_j \le 0,$$

for any $i, j \in [r]$ and $i \ne j$.

To clearly present our method, we begin by rewriting Equation (6) as the following maximization problem:

$$\begin{aligned}\max_{\mathcal{S}\subseteq\left[\sum_{\ell=1}^{L} r_\ell\right]} & \left\langle \left[\nabla_{\boldsymbol{\sigma}} \mathcal{L}_{\text{ft}}\left(\Theta_{\text{LoRA}}\right)\right]_{\mathcal{S}^-}, \left[\boldsymbol{\sigma}\right]_{\mathcal{S}^-} \right\rangle \\ & - \frac{1}{2} \left[\boldsymbol{\sigma}\right]_{\mathcal{S}^-}^{\top} \left[\nabla^2_{\boldsymbol{\sigma}} \mathcal{L}_{\text{ft}}\left(\Theta_{\text{LoRA}}\right)\right]_{\mathcal{S}^-} \left[\boldsymbol{\sigma}\right]_{\mathcal{S}^-}, \\ \text{s.t.} \quad & |\mathcal{S}| \leq b,\end{aligned} \tag{7}$$

According to Theorem 2, the objective function is submodular if $\left[\nabla^2_{\boldsymbol{\sigma}} \mathcal{L}_{\text{ft}}\left(\Theta_{\text{LoRA}}\right)\right]_{i,j} \cdot \sigma_i \sigma_j \geq 0$, for all $i \neq j$.

To balance between tractability of solving Equation (7) and the fidelity of Hessian information, we propose modifying the Hessian to preserve as much curvature information as possible while still ensuring submodularity. Specifically, we solve the following projection problem:

$$\begin{aligned}\min_{\boldsymbol{G}\in\mathbb{S}^{\sum_{\ell=1}^{L} r_\ell}} & \left\|\boldsymbol{G} - \nabla^2_{\boldsymbol{\sigma}} \mathcal{L}_{\text{ft}}\left(\Theta_{\text{LoRA}}\right)\right\|_F^2, \\ \text{s.t.} \quad & G_{i,j} \cdot \sigma_i \sigma_j \geq 0, \text{ for } i, j \in \left[\sum_{\ell=1}^{L} r_\ell\right] \text{ and } i \neq j.\end{aligned} \tag{8}$$

This projection preserves as much of the original Hessian as possible under the Frobenius norm, while enforcing the structural condition required for submodularity. Since all $G_{i,j}$ terms are separable in both the objective and the constraint of Equation (8), we can derive a closed-form solution for the projection:

$$G_{i,j} = \begin{cases} \left[\nabla^2_{\boldsymbol{\sigma}} \mathcal{L}_{\text{ft}}\left(\Theta_{\text{LoRA}}\right)\right]_{i,j}, \\ \quad \text{if } \left[\nabla^2_{\boldsymbol{\sigma}} \mathcal{L}_{\text{ft}}\left(\Theta_{\text{LoRA}}\right)\right]_{i,j} \cdot \sigma_i \sigma_j \geq 0 \text{ and } i \neq j, \\ 0, \quad \text{if } \left[\nabla^2_{\boldsymbol{\sigma}} \mathcal{L}_{\text{ft}}\left(\Theta_{\text{LoRA}}\right)\right]_{i,j} \cdot \sigma_i \sigma_j < 0 \text{ and } i \neq j, \\ \left[\nabla^2_{\boldsymbol{\sigma}} \mathcal{L}_{\text{ft}}\left(\Theta_{\text{LoRA}}\right)\right]_{i,i}, \quad \text{if } i = j. \end{cases} \tag{9}$$

This projection is computationally efficient due to its closed-form nature. To convert Equation (7) into a submodular function maximization problem, we apply this projection to the Hessian matrix, resulting in a modified matrix denoted by $\boldsymbol{G}$. We then solve the following cardinality-constrained submodular function maximization problem:

$$\begin{aligned}\max_{\mathcal{S}\subseteq\left[\sum_{\ell=1}^{L} r_\ell\right]} & \left\langle \left[\nabla_{\boldsymbol{\sigma}} \mathcal{L}_{\text{ft}}\left(\Theta_{\text{LoRA}}\right)\right]_{\mathcal{S}^-}, \left[\boldsymbol{\sigma}\right]_{\mathcal{S}^-} \right\rangle \\ & - \frac{1}{2} \left[\boldsymbol{\sigma}\right]_{\mathcal{S}^-}^{\top} \left[\boldsymbol{G}\right]_{\mathcal{S}^-} \left[\boldsymbol{\sigma}\right]_{\mathcal{S}^-}, \\ \text{s.t.} \quad & |\mathcal{S}| \leq b,\end{aligned} \tag{10}$$

by greedy algorithms. Although solving Equation (10) exactly remains NP-hard, the greedy algorithm provides a tractable approximation with a known $1 - 1/e$ guarantee. This trade-off reflects our goal of striking a balance between two competing aspects: preserving second-order information from the Hessian and ensuring the problem remains solvable with provable guarantees.

The complete algorithm of the proposed SubLoRA for LoRA rank determination is summarized in Algorithm 2. Stage 1 involves standard LoRA fine-tuning, where we apply (stochastic) gradient descent to minimize the loss function in Equation (2), providing an approximate solution. This step follows classical LoRA training procedures and is not the primary focus of this work. Stages 2 and 3 constitute the core contribution of this paper. In Stage 2, we approximate the objective in Equation (4) using a second-order Taylor expansion, as formulated in Equation (6). To balance the trade-off between computational tractability and fidelity to the curvature of the loss landscape, we project the Hessian matrix according to the constraint in Equation (8), using the closed-form solution in Equation (9). This projection ensures that the resulting quadratic objective in Equation (10) is submodular. In Stage 3, we solve the resulting cardinality-constrained submodular function maximization problem using either the greedy or randomized greedy algorithm. The final output is a selected subset of singular values to retain, while the remaining LoRA components are discarded accordingly.

Algorithm 2: SubLoRA

1: **Stage 1 (LoRA fine-tuning):** Obtain LoRA parameters Θ_{LoRA} by minimizing Equation (2)
2: **Stage 2 (Problem formulation):** Calculate the gradient vector $\nabla_{\boldsymbol{\sigma}} \mathcal{L}_{\text{ft}}\left(\Theta_{\text{LoRA}}\right)$ and Hessian matrix $\nabla^2_{\boldsymbol{\sigma}} \mathcal{L}_{\text{ft}}\left(\Theta_{\text{LoRA}}\right)$ with respect to singular values $\boldsymbol{\sigma}$.
3: Project the Hessian and obtain $\boldsymbol{G}$ as in Equation (9)
4: Formulate the submodular function maximization problem as in Equation (10)
5: **Stage 3 (Solvers):** Apply greedy algorithm (Algorithm 1) to obtain the set of singular values to be kept

In submodular function maximization, it is well known that without the monotonicity condition, the greedy algorithm (Algorithm 1) may be trapped in suboptimal or ill-conditioned points, potentially leading to poor performance (Krause and Golovin 2014; Fujishige 2005). The following lemma provides a sufficient condition under which the objective function is monotone.

Lemma 1 *Suppose the LoRA fine-tuning stage (Stage 1 of Algorithm 2) minimizes the loss in Equation* (2) *to a point satisfying the second-order necessary optimality conditions. Then, the objective function in Equation* (10) *is monotone.*

Note that the above lemma requires that the LoRA fine-tuning step satisfies second-order necessary optimality conditions. Recent theoretical results (Ge et al. 2015; Ge, Lee, and Ma 2016) suggest that many widely adopted optimizers, such as Adam and SGD, converge to points satisfying second-order optimality conditions with high probability under mild assumptions. As a result, the objective in Equation (10) tends to exhibit approximate monotonicity in practice after a sufficient number of iterations in the fine-tuning stage.

Alternating Training and Rank Determination

The previous sections focus on a training-free, post-hoc method for LoRA rank determination, where the LoRA components are fixed and the effective rank is reduced by pruning unimportant singular values. Here, we extend our

Algorithm 3: Alternating SubLoRA

for $t = 0, 1, \cdots, T$ **do**
  Stage 1 (Parameter update): Run several steps of optimization algorithms for LoRA parameters Θ_{LoRA}.
  Stage 2 (Problem formulation): Calculate the gradient vector $\nabla_{\boldsymbol{\sigma}} \mathcal{L}_{\text{ft}}(\Theta_{\text{LoRA}})$ and Hessian matrix $\nabla^2_{\boldsymbol{\sigma}} \mathcal{L}_{\text{ft}}(\Theta_{\text{LoRA}})$ with respect to singular values $\boldsymbol{\sigma}$.
  Project the Hessian and obtain $\boldsymbol{G}$ as in Equation (9)
  Formulate the submodular function maximization problem as in Equation (10)
  Stage 3 (Solvers): Apply the greedy algorithm (Algorithm 1) to obtain the set of singular values to be retained
end for

method to a more integrated setting, where LoRA fine-tuning and rank determination are performed simultaneously. The full procedure is described in Algorithm 3.

The exact computation of the Hessian matrix is often a major concern in deep neural networks due to the large number of trainable parameters. However, in the context of SubLoRA, this challenge is significantly mitigated. In contrast to traditional second-order optimization methods such as Newton's method, which require computing the Hessian with respect to all model parameters, SubLoRA computes the Hessian $\nabla^2_{\boldsymbol{\sigma}} \mathcal{L}_{\text{ft}}(\Theta_{\text{LoRA}})$ only with respect to the singular values $\boldsymbol{\sigma}$. This dramatically reduces the computational and memory requirements.

Experiments

Training-Free Rank Determination

We evaluate the training-free rank determination introduced in Algorithm 2 for solving a class of PDEs with varying physical parameters. For a given PDE, we first pre-train the neural network (MLPs) and obtain network parameters Θ_{pt}. We then fine-tune the network by LoRA (with uniform pre-defined ranks across all layers) on a new PDE with another physical parameter and obtain Θ_{LoRA}. The total LoRA rank is intentionally set much higher than the desired budget. We then apply the proposed rank determination method (e.g., SubLoRA) to remove less informative components, producing budget-constrained model parameters $\hat{\Theta}_{\text{ft}}$ by reallocating rank across layers. To evaluate the quality of the rank determination, we compute the relative error (rel) between the predicted solution $\phi(\boldsymbol{x}; \Theta)$ and the ground truth solution $\boldsymbol{u}(\boldsymbol{x})$ on a test dataset $\{(\boldsymbol{x}_i, \boldsymbol{u}(\boldsymbol{x}_i))\}_{i=1}^{N_t}$, defined as

$$\sqrt{\frac{\sum_{i=1}^{N_t} \|\phi(\boldsymbol{x}_i; \Theta) - \boldsymbol{u}(\boldsymbol{x}_i)\|_2^2}{\sum_{i=1}^{N_t} \|\boldsymbol{u}(\boldsymbol{x}_i)\|_2^2}}.$$

In the experiments, we compare four different methods of LoRA rank determination. The first is the linearized objective function method in Equation (5), denoted as "LinearLoRA". The second method, named "DiagLoRA", utilizes the diagonal second-order approximation. The third method "SubLoRA" corresponds to the proposed approach that formulates the rank selection problem as a submodular maximization using the projected Hessian in Equation (10), solved by either greedy or randomized greedy algorithms. To evaluate the effectiveness of the Hessian projection step in Equation (8), we also include a fourth method, termed "HessLoRA", which uses the exact (unprojected) Hessian from Equation (7) and applies the greedy algorithm for optimization. All experiments are conducted using a four-layer MLP architecture with three hidden layers. Each hidden layer consists of 1000 neurons and is fine-tuned with LoRA using a fixed pre-defined rank of 50 per layer, resulting in a total initial LoRA rank of 100. We vary the total rank budget in the experiments, and each method determines its own layer-wise rank allocation under the given constraint.

Allen–Cahn Equations We consider a class of Allen–Cahn equations with varying physical parameters $\boldsymbol{\lambda} \in \mathbb{R}^2$:

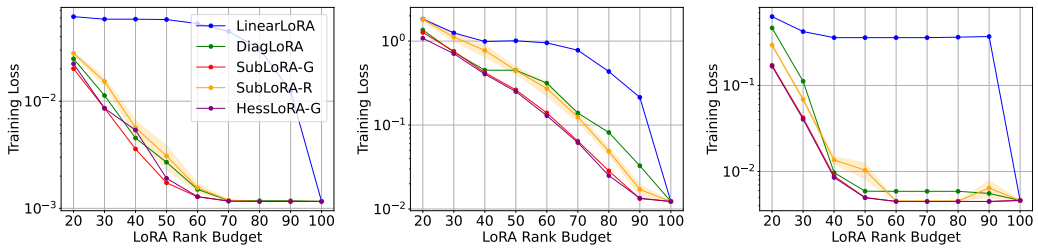
$$\begin{aligned} \frac{\partial u(t, \boldsymbol{x}; \boldsymbol{\lambda})}{\partial t} &- \Delta u(t, \boldsymbol{x}; \boldsymbol{\lambda}) - u(t, \boldsymbol{x}; \boldsymbol{\lambda})^3 + u(t, \boldsymbol{x}; \boldsymbol{\lambda}) \\ &= g(t, \boldsymbol{x}; \boldsymbol{\lambda}), \quad (t, \boldsymbol{x}) \in [0, 1] \times \Omega, \\ u(t, \boldsymbol{x}; \boldsymbol{\lambda}) &= h_1(t, \boldsymbol{x}; \boldsymbol{\lambda}), \quad (t, \boldsymbol{x}) \in [0, 1] \times \partial\Omega, \\ u(0, \boldsymbol{x}; \boldsymbol{\lambda}) &= h_2(\boldsymbol{x}; \boldsymbol{\lambda}), \quad \boldsymbol{x} \in \Omega, \end{aligned} \tag{11}$$

where the temporal and spatial domains are defined as $[0, 1]$ and $\Omega = \{\boldsymbol{x} \in \mathbb{R}^2 : \|\boldsymbol{x}\|_2 \leq 1\}$, respectively. Similarly, the exact solution $u(\boldsymbol{x}; \boldsymbol{\lambda})$ with the parameter $\boldsymbol{\lambda}$ is defined as $u(\boldsymbol{x}; \boldsymbol{\lambda}) = e^{-t} \sin\left(\frac{\pi\lambda_1}{2}\left(1 - \|\boldsymbol{x}\|_2\right)^{2.5}\right) + \lambda_2 \cdot e^{-t} \sin\left(\frac{\pi}{2}\left(1 - \|\boldsymbol{x}\|_2\right)\right)$.

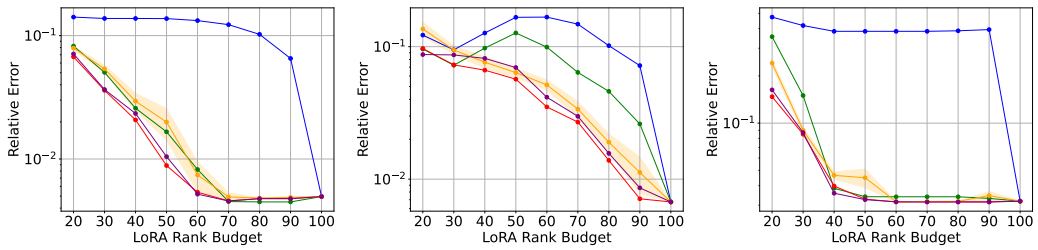
We visualize the experimental results in Figure 1. Similar trends to those observed in the elliptic equation experiments are evident here, further validating the effectiveness of incorporating second-order information compared to using only first-order approximations. Notably, in these examples, SubLoRA-G slightly outperforms HessLoRA-G. This highlights the advantage of transforming the objective into a submodular function by the Hessian projection defined in Equation (8). It is important to note that a general (non-submodular) quadratic objective is not guaranteed to perform well under the greedy algorithm, where the performance can degrade significantly in certain cases. However, by projecting the Hessian to enforce the submodularity, as done in Equation (8), we ensure that applying the greedy algorithm becomes theoretically sound and practically robust. Therefore, the use of Hessian projection followed by greedy algorithm, as implemented in Algorithm 2, is a reasonable and effective strategy for rank determination.

Alternating Training with Rank Determination

We also evaluate the alternating algorithm for LoRA fine-tuning and rank determination, as described in Algorithm 3, on a class of PDEs with varying physical parameters. The Allen-Cahn equations are considered. We begin by pre-training neural networks (MLPs) to obtain the pre-trained parameters Θ_{pt}. Then, we apply Algorithm 3 to alternate between LoRA parameter updates and rank determination. In all experiments, we use a four-layer MLP (three hidden layers) as the base architecture for PINN training. During each



(a) $\boldsymbol{\lambda} = (1, 1)$, training loss (b) $\boldsymbol{\lambda} = (1, 5)$, training loss (c) $\boldsymbol{\lambda} = (2, 1)$, training loss



(d) $\boldsymbol{\lambda} = (1, 1)$, relative error (e) $\boldsymbol{\lambda} = (1, 5)$, relative error (f) $\boldsymbol{\lambda} = (2, 1)$, relative error

Figure 1: Performance comparison of different rank determination methods on Allen–Cahn equations under varying LoRA rank budgets and physical parameters.

outer-loop iteration, Stage 1 of Algorithm 3 performs LoRA parameter updates using the Adam optimizer for 100 epochs. In Stages 2 and 3, rank determination methods are applied to prune less informative singular values, ensuring that the number of non-zero singular values remains within the rank budget. This alternating procedure is repeated for $T = 5$ iterations.

To illustrate the training dynamics, we visualize the optimization trajectories of Algorithm 3 in Figure 2, comparing the alternating LinearLoRA (first-order) and the alternating SubLoRA (second-order) methods. In the figure, circular markers represent the outputs of Stage 1 (parameter updates), while cross markers indicate the results after rank determination in Stages 2 and 3. The visualization shows that the proposed SubLoRA method facilitates more accurate rank determination, which in turn guides Stage 1 to progressively converge to parameter configurations that better align with the rank budget. This interaction leads to stable convergence of the alternating process. In contrast, the first-order method of LinearLoRA fails to decide LoRA ranks effectively, often discarding important components, thereby disrupting learning and preventing convergence to a meaningful low-rank structure.

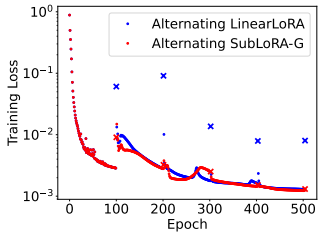


Figure 2: Training trajectories of Algorithm 3 using (first-order) LinearLoRA and (second-order) SubLoRA-G as rank determination methods on Allen–Cahn equations with $\boldsymbol{\lambda} = (1, 1)$. The SubLoRA-G method leads to more reliable rank determination, which effectively guides the alternating optimization to convergence. In contrast, the LinearLoRA method tends to discard critical components, resulting in divergence of the training process.

Conclusion

In this paper, we introduce SubLoRA, a rank determination method for LoRA based on submodular function maximization. We formulate the rank determination problem as a combinatorial optimization problem with a set-valued quadratic objective derived from the second-order Taylor expansion of the fine-tuning loss. To address the computational challenges of optimizing this objective, we design a Hessian projection that transforms the objective into a submodular function. The transformation enables the use of efficient greedy algorithms with theoretical approximation guarantees, making the method both computationally tractable and effective. To further enhance LoRA fine-tuning with rank determination, we propose an alternating algorithm that iteratively updates LoRA parameters and performs rank determination. We apply SubLoRA to LoRA fine-tuning and rank determination of PINNs for solving a class of PDEs due to their inherent smoothness. Experimental results demonstrate that our method consistently outperforms first-order and diagonal second-order baselines. The use of Hessian information and the submodular projection leads to better capturing of the loss landscape and more effective rank allocation under budget constraints.

References

Chen, S.; Ge, C.; Tong, Z.; Wang, J.; Song, Y.; Wang, J.; and Luo, P. 2022. AdaptFormer: Adapting vision Transformers for scalable visual recognition. *Advances in Neural Information Processing Systems*, 35: 16664–16678.

Dettmers, T.; Pagnoni, A.; Holtzman, A.; and Zettlemoyer, L. 2023. QLoRA: Efficient finetuning of quantized LLMs. *Advances in Neural Information Processing Systems*, 36: 10088–10115.

Ding, N.; Qin, Y.; Yang, G.; Wei, F.; Yang, Z.; Su, Y.; Hu, S.; Chen, Y.; Chan, C.-M.; Chen, W.; et al. 2023. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, 5(3): 220–235.

Fujishige, S. 2005. *Submodular functions and optimization*, volume 58. Elsevier.

Ge, R.; Huang, F.; Jin, C.; and Yuan, Y. 2015. Escaping from saddle points—online stochastic gradient for tensor decomposition. In *Conference on Learning Theory*, 797–842. PMLR.

Ge, R.; Lee, J. D.; and Ma, T. 2016. Matrix completion has no spurious local minimum. *Advances in Neural Information Processing Systems*, 29.

Han, Z.; Gao, C.; Liu, J.; Zhang, J.; and Zhang, S. Q. 2024. Parameter-Efficient Fine-Tuning for Large Models: A Comprehensive Survey. *Transactions on Machine Learning Research*.

Hayou, S.; Ghosh, N.; and Yu, B. 2024. LoRA+: Efficient Low Rank Adaptation of Large Models. In *International Conference on Machine Learning*, 17783–17806. PMLR.

Hu, E. J.; yelong shen; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; and Chen, W. 2022. LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations*.

Jang, U.; Lee, J. D.; and Ryu, E. K. 2024. LoRA Training in the NTK Regime has No Spurious Local Minima. In *Forty-first International Conference on Machine Learning*.

Jia, M.; Tang, L.; Chen, B.-C.; Cardie, C.; Belongie, S.; Hariharan, B.; and Lim, S.-N. 2022. Visual prompt tuning. In *European conference on computer vision*, 709–727. Springer.

Kim, J.; Kim, J.; and Ryu, E. K. 2025. LoRA Training Provably Converges to a Low-Rank Global Minimum Or It Fails Loudly (But it Probably Won't Fail). In *Forty-second International Conference on Machine Learning*.

Krause, A.; and Golovin, D. 2014. Submodular function maximization. *Tractability*, 3(71-104): 3.

Liang, J.; Huang, W.; Wan, G.; Yang, Q.; and Ye, M. 2025. Lorasculpt: Sculpting lora for harmonizing general and specialized knowledge in multimodal large language models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 26170–26180.

Liu, Z.; Lyn, J.; Zhu, W.; Tian, X.; and Graham, Y. 2024. ALoRA: Allocating Low-Rank Adaptation for Fine-tuning Large Language Models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, 622–641. Mexico City, Mexico: Association for Computational Linguistics.

Majumdar, R.; Jadhav, V.; Deodhar, A.; Karande, S.; Vig, L.; and Runkana, V. 2023. PIHLoRA: Physics-informed hypernetworks for low-ranked adaptation. In *AI for Accelerated Materials Design-NeurIPS 2023 Workshop*.

Mao, Y.; Ge, Y.; Fan, Y.; Xu, W.; Mi, Y.; Hu, Z.; and Gao, Y. 2025. A survey on LoRA of large language models. *Frontiers of Computer Science*, 19(7): 197605.

Schmirler, R.; Heinzinger, M.; and Rost, B. 2024. Fine-tuning protein language models boosts predictions across diverse tasks. *Nature Communications*, 15(1): 7407.

Valipour, M.; Rezagholizadeh, M.; Kobyzev, I.; and Ghodsi, A. 2023. DyLoRA: Parameter-Efficient Tuning of Pre-trained Models using Dynamic Search-Free Low-Rank Adaptation. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, 3274–3287. Dubrovnik, Croatia: Association for Computational Linguistics.

Wang, Y.; Bai, J.; Eshaghi, M. S.; Anitescu, C.; Zhuang, X.; Rabczuk, T.; and Liu, Y. 2025. Transfer Learning in Physics-Informed Neurals Networks: Full Fine-Tuning, Lightweight Fine-Tuning, and Low-Rank Adaptation. *International Journal of Mechanical System Dynamics*.

Wang, Y.; Shi, H.; Han, L.; Metaxas, D. N.; and Wang, H. 2024. BLoB: Bayesian Low-Rank Adaptation by Backpropagation for Large Language Models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Zeng, Y.; and Lee, K. 2024. The Expressive Power of Low-Rank Adaptation. In *The Twelfth International Conference on Learning Representations*.

Zhang, Q.; Chen, M.; Bukharin, A.; He, P.; Cheng, Y.; Chen, W.; and Zhao, T. 2023. Adaptive Budget Allocation for Parameter-Efficient Fine-Tuning. In *The Eleventh International Conference on Learning Representations*.

Zhang, Y.; Wang, J.; Yu, L.-C.; Xu, D.; and Zhang, X. 2024. Personalized LoRA for human-centered text understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 17, 19588–19596.

Zhang, Y.; Zhou, K.; and Liu, Z. 2024. Neural prompt search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.