OP-LoRA: The Blessing of Dimensionality with Overparameterized Low-Rank Adaptation

Supplementary Material

6. Theoretical Analysis

6.1. Extension of overparmeterization analysis to MLP

Although our analysis of the optimization benefits of OP-LoRA is carried out in the linear case (Section 3.2 of the main paper), the same principles extend to deeper ReLU networks. We formalize this below.

Lemma 6.1 (First-order expansion of ReLU activations)**.** *Let $h = \mathrm{ReLU}(a)$ with $a = W_1 z$. For a perturbation Δa, define the activation mask $\mathbf{M} = \mathbf{1}_{\{W_1 z > 0\}}$. Then a first-order Taylor expansion gives*

$$h_{new} \;\approx\; h + \mathbf{M} \odot \Delta a.$$

Lemma 6.2 (Expansion of pre-activation perturbations)**.** *Let $a = W_1 z$. For small perturbations $\Delta W_1, \Delta z$,*

$$a + \Delta a \;=\; (W_1 + \Delta W_1)(z + \Delta z).$$

Expanding and subtracting a yields

$$\Delta a = \Delta W_1 z \;+\; W_1 \Delta z \;+\; \Delta W_1 \Delta z.$$

Lemma 6.3 (Form of gradient updates)**.** *With gradients ∇_h, ∇_z, the updates take the form*

$$\Delta W_1 = -\eta(\nabla_h \odot \mathbf{M}) z^\top, \qquad \Delta z = -\eta \nabla_z.$$

Substituting into the expansion of Δa and dropping second order terms gives

$$\Delta a \approx -\eta((\nabla_h \odot \mathbf{M})\, z^\top z \;+\; W_1\, \nabla_z).$$

Theorem 6.4 (Approximation of activation perturbations)**.** *Combining the previous results,*

$$\Delta h \;\approx\; -\eta \mathbf{M} \odot \left[(\nabla_h \odot \mathbf{M})\, z^\top z \;+\; W_1\, \nabla_z\right].$$

Theorem 6.5 (Update rule in the ReLU case)**.** *Let v be the parameter vector defined as in the main paper. Then the update rule is*

$$\begin{aligned} v^{(t+1)} = v^{(t)} &- \eta \|h^{(t)}\|^2 \nabla_{v^{(t)}} \\ &- \eta W_2^{(t)} \Big(\mathbf{M} \odot \left[(W_2^\top \nabla_v \odot \mathbf{M})\, z^\top z + W_1 \nabla_z \right] \Big). \end{aligned}$$

Corollary 6.6 (Comparison to the linear case)**.** *In the linear case (Section 3.2 of the main paper), the update reduces to*

$$v^{(t+1)} = v^{(t)} - \eta \|h^{(t)}\|^2 \nabla_{v^{(t)}} - \eta W_2^{(t)} (W_2^{(t)\top} \nabla_v).$$

Thus both cases share two key terms:

- ***Trainable learning rate:*** *$-\|h^{(t)}\|^2 \nabla_v$, unchanged from the linear case.*
- ***Adaptive line search:*** *an update along the subspace spanned by the current columns of W_2.*

Remark 6.7 (Geometric interpretation in the ReLU case). The adaptive line search retains the same geometric role as in the linear case: shifting updates along the span of W_2. However, the ReLU nonlinearity induces a *composite over-parameterization*: the values h themselves are generated through an extra layer with nonlinearity, and each column of W_2 only contributes when its corresponding ReLU unit is active. This leads to more diverse update directions when different units activate across inputs or training steps.

6.1.1. LoRA Hessian

We begin with the LoRA reparameterization

$$W = W_0 + BA, \qquad B \in \mathbb{R}^{d \times r},\; A \in \mathbb{R}^{r \times d},\; W_0 \in \mathbb{R}^{d \times d}.$$

Let $L(W)$ denote the loss, and let H_W be the Hessian of L at W_0, viewed as a linear operator mapping perturbations in W to second–order variations of the loss.

Lemma 6.8 (Quadratic approximation)**.** *For any small perturbation ΔW, a second–order Taylor expansion gives*

$$\begin{aligned} L(W_0 + \Delta W) \;\approx\; & L(W_0) + \langle \nabla_W L,\, \Delta W \rangle \\ & + \tfrac{1}{2} \langle \Delta W,\, H_W(\Delta W) \rangle. \end{aligned}$$

Theorem 6.9 (Effective Hessian with respect to B)**.** *Fix A and consider variations in B. For a perturbation ΔB, we have $\Delta W = \Delta B A$. The corresponding effective curvature operator is*

$$H_B(\Delta B) \;=\; H_W(\Delta B A)\, A^\top.$$

Proof. Substitute $\Delta W = \Delta B A$ into the quadratic form:

$$\begin{aligned} \tfrac{1}{2} \langle \Delta W,\, H_W \Delta W \rangle &= \tfrac{1}{2} \langle \Delta B A,\, H_W(\Delta B A) \rangle \\ &= \tfrac{1}{2} \langle \Delta B,\, H_W(\Delta B A)\, A^\top \rangle, \end{aligned}$$

which establishes the claim. □

Corollary 6.10 (Operator and matrix forms of H_B)**.** *In operator notation,*

$$H_B \;=\; (\cdot\, A^\top) \circ H_W \circ (\cdot A),$$

where $(\cdot X)$ denotes right multiplication by X. In matrix form,

$$H_B \;=\; (A^\top \otimes I)^\top H_W\,(A^\top \otimes I), \qquad I \in \mathbb{R}^{d\times d}.$$

Theorem 6.11 (Effective Hessian with respect to A)**.** *Fix B and consider variations in A. For a perturbation ΔA, we have $\Delta W = B\Delta A$. The corresponding effective curvature operator is*

$$H_A(\Delta A) \;=\; B^\top H_W(B\Delta A).$$

Proof. Substitute $\Delta W = B\Delta A$ into the quadratic form:

$$\begin{aligned}\tfrac{1}{2}\langle \Delta W,\, H_W \Delta W\rangle &= \tfrac{1}{2}\langle B\Delta A,\, H_W(B\Delta A)\rangle \\ &= \tfrac{1}{2}\langle \Delta A,\, B^\top H_W(B\Delta A)\rangle,\end{aligned}$$

which establishes the claim. □

Corollary 6.12 (Operator and matrix forms of H_A)**.** *In operator notation,*

$$H_A \;=\; B^\top \circ H_W \circ B.$$

In matrix form,

$$H_A \;=\; (I \otimes B^\top)\, H_W\,(I \otimes B), \qquad I \in \mathbb{R}^{d\times d}.$$

6.1.2. Condition number bounds

In the main paper we state the following bound on the condition number of H_A :

$$\frac{\kappa(B)^2}{\kappa(H_W)} \;\le\; \kappa(H_A) \;\le\; \kappa(H_W)\,\kappa(B)^2,$$

where $\kappa(\cdot)$ denotes the spectral condition number. We first prove a general bound for quadratic forms of the type $S^\top H_W S$, then specialize to the LoRA case $S = I \otimes B$.

Lemma 6.13 (Spectral bounds for $S^\top H_W S$)**.** *Let $H_W \in \mathbb{R}^{n\times n}$ be symmetric positive definite (SPD), and let $S \in \mathbb{R}^{n\times m}$ be full column rank. Denote by $\lambda_{\min}(\cdot)$ and $\lambda_{\max}(\cdot)$ the minimal and maximal eigenvalues, and by $\sigma_{\min}(\cdot)$ and $\sigma_{\max}(\cdot)$ the minimal and maximal singular values. Define*

$$H_S \;=\; S^\top H_W S.$$

Then

$$\lambda_{\min}(H_W)\,\sigma_{\min}(S)^2 \;\le\; \lambda_{\min}(H_S) \;\le\; \lambda_{\max}(H_W)\,\sigma_{\min}(S)^2,$$
$$\lambda_{\min}(H_W)\,\sigma_{\max}(S)^2 \;\le\; \lambda_{\max}(H_S) \;\le\; \lambda_{\max}(H_W)\,\sigma_{\max}(S)^2.$$

Proof. For any nonzero $x \in \mathbb{R}^m$,

$$x^\top H_S x = x^\top S^\top H_W S x = (Sx)^\top H_W (Sx).$$

Since H_W is SPD,

$$\lambda_{\min}(H_W)\,\|Sx\|^2 \;\le\; (Sx)^\top H_W (Sx) \;\le\; \lambda_{\max}(H_W)\,\|Sx\|^2.$$

Using the singular value bounds $\sigma_{\min}(S)\,\|x\| \le \|Sx\| \le \sigma_{\max}(S)\,\|x\|$, we obtain, for all nonzero x,

$$\begin{aligned}\lambda_{\min}(H_W)\,\sigma_{\min}(S)^2\,\|x\|^2 \;&\le\; x^\top H_S x \\ &\le\; \lambda_{\max}(H_W)\,\sigma_{\max}(S)^2\,\|x\|^2.\end{aligned}$$

Taking the minimum over unit vectors x gives

$$\begin{aligned}\lambda_{\min}(H_W)\,\sigma_{\min}(S)^2 \;&\le\; \lambda_{\min}(H_S) \\ &\le\; \lambda_{\max}(H_W)\,\sigma_{\min}(S)^2,\end{aligned}$$

since $\|Sx\| \ge \sigma_{\min}(S)\|x\|$ holds for all x.

Taking the maximum over unit vectors x gives

$$\begin{aligned}\lambda_{\min}(H_W)\,\sigma_{\max}(S)^2 \;&\le\; \lambda_{\max}(H_S) \\ &\le\; \lambda_{\max}(H_W)\,\sigma_{\max}(S)^2,\end{aligned}$$

since $\|Sx\| \le \sigma_{\max}(S)\|x\|$ holds for all x. This establishes the bounds.

□

Corollary 6.14 (Condition number of $S^\top H_W S$)**.** *Under the assumptions of Lemma 6.13,*

$$\frac{\kappa(S)^2}{\kappa(H_W)} \;\le\; \kappa(H_S) \;\le\; \kappa(H_W)\,\kappa(S)^2,$$

where $\kappa(H_S) = \lambda_{\max}(H_S)/\lambda_{\min}(H_S)$ and $\kappa(S) = \sigma_{\max}(S)/\sigma_{\min}(S)$.

Proof. From Lemma 6.13, we have

$$\begin{aligned}\lambda_{\max}(H_S) \;&\le\; \lambda_{\max}(H_W)\,\sigma_{\max}(S)^2, \\ \lambda_{\min}(H_S) \;&\ge\; \lambda_{\min}(H_W)\,\sigma_{\min}(S)^2.\end{aligned}$$

Therefore

$$\begin{aligned}\kappa(H_S) &= \frac{\lambda_{\max}(H_S)}{\lambda_{\min}(H_S)} \\ &\le\; \frac{\lambda_{\max}(H_W)\,\sigma_{\max}(S)^2}{\lambda_{\min}(H_W)\,\sigma_{\min}(S)^2} \\ &= \kappa(H_W)\,\kappa(S)^2.\end{aligned}$$

Similarly, Lemma 6.13 also gives

$$\begin{aligned}\lambda_{\max}(H_S) \;&\ge\; \lambda_{\min}(H_W)\,\sigma_{\max}(S)^2, \\ \lambda_{\min}(H_S) \;&\le\; \lambda_{\max}(H_W)\,\sigma_{\min}(S)^2,\end{aligned}$$

so

$$\begin{aligned}\kappa(H_S) &= \frac{\lambda_{\max}(H_S)}{\lambda_{\min}(H_S)} \\ &\geq \frac{\lambda_{\min}(H_W)\,\sigma_{\max}(S)^2}{\lambda_{\max}(H_W)\,\sigma_{\min}(S)^2} \\ &= \frac{\kappa(S)^2}{\kappa(H_W)}.\end{aligned}$$

This establishes the claimed bounds. □

We now specialize to the LoRA Hessians H_A and H_B.

Theorem 6.15 (Condition number bounds for H_A)**.** *Assume H_W is SPD and B is full rank. Let H_A be the effective Hessian with respect to A,*

$$H_A = (I \otimes B^\top)\, H_W\, (I \otimes B).$$

Then

$$\frac{\kappa(B)^2}{\kappa(H_W)} \;\leq\; \kappa(H_A) \;\leq\; \kappa(H_W)\,\kappa(B)^2.$$

Proof. Set $S = I \otimes B$, so that $H_A = S^\top H_W S$. The singular values of $I \otimes B$ are exactly the singular values of B, repeated, so $\kappa(S) = \kappa(B)$. Applying Corollary 6.14 with this choice of S yields the stated bounds. □

Corollary 6.16 (Condition number bounds for H_B)**.** *Assume H_W is SPD and A is full rank. Let H_B be the effective Hessian with respect to B,*

$$H_B = (A^\top \otimes I)^\top\, H_W\, (A^\top \otimes I).$$

Then

$$\frac{\kappa(A)^2}{\kappa(H_W)} \;\leq\; \kappa(H_B) \;\leq\; \kappa(H_W)\,\kappa(A)^2.$$

Proof. Apply Corollary 6.14 with $S = A^\top \otimes I$, using again that $\kappa(A^\top \otimes I) = \kappa(A)$. □

7. Additional Results

7.1. Gradient Analysis LoRA and OP-LoRA

Recall that in the main paper we show

$$\frac{\kappa(B)^2}{\kappa(H_W)} \;\leq\; \kappa(H_A) \;\leq\; \kappa(H_W)\,\kappa(B)^2,$$

where A and B are LoRA matrices, W are the base weights, $\kappa(\cdot)$ denotes the condition number, and H are the corresponding Hessians. Equation (7.1) implies that LoRA can exhibit worse Hessian conditioning than full finetuning.

A high Hessian condition number indicates very high curvature in some directions relative to others. This matters because the step size must be small enough to avoid instabilities along the highest-curvature direction; the maximal stable learning rate is then limited by that direction and may be too small to make progress in low-curvature directions. This becomes problematic when the highest-curvature direction has already been minimized, but the low-curvature directions still require larger learning rates.

A useful diagnostic is the magnitude of the gradient in the direction of the principal singular vector of the Hessian of the trainable parameters. Let v be that direction (estimated by power iteration) and g the gradient. If $|v^\top g|$ is relatively large, the loss can still be reduced even with a learning rate small enough to remain stable in the largest-curvature direction. Conversely, if $|v^\top g| \approx 0$, a large condition number will prevent further decrease of the loss.

Setup. We use the power-iteration method to estimate v and then measure $|v^\top g|$ for a small-scale Rotated-MNIST problem. Pre-training is performed on MNIST and continued training is on Rotated MNIST (as in Fig. 2 of the main paper). We report the terminal values for LoRA and OP-LoRA.

Method	$\lvert v^\top g\rvert$
OP-LoRA	0.42
LoRA	0.06

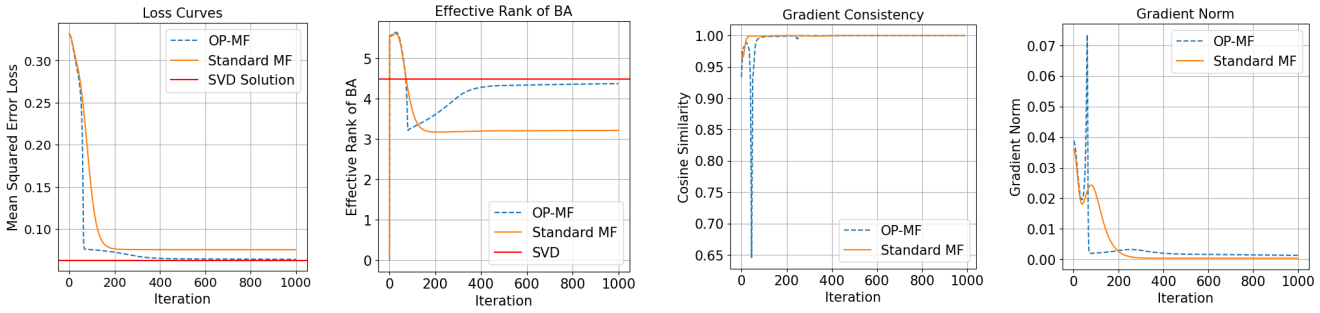
Table 5. $|v^\top g|$ at the end of training on Rotated MNIST. Higher is better (indicates remaining descent along the highest-curvature direction).

Findings. OP-LoRA exhibits a much larger $|v^\top g|$ in the direction of largest curvature than LoRA (Table 5). Empirically, this suggests OP-LoRA may be *less sensitive* to poor conditioning than LoRA, because it can continue to reduce the loss even when the step size is constrained by the highest-curvature direction. Thus, even if the OP-LoRA MLP were itself poorly conditioned, this sensitivity matters less than for LoRA.

These observations are consistent with the view that OP-LoRA adaptively reshapes the LoRA loss landscape via reparameterization, leading to better and faster optimization.

7.2. A Matrix Factorization Case Study

To verify that the gradient properties of MLP overparameterization results in observable changes, we design a controlled matrix factorization experiment com-



(a) **Loss Curve** (b) **Effective Rank** (c) **Gradient Consistency** (d) **Gradient Norm**

Figure 5. **Matrix Factorization (MF) Gradient Analysis: (a)** Loss Curve shows the reconstruction error for matrix factorization. OP-MF converges better and faster than standard MF. **(b)** Effective Rank of BA reveals changes in the rank of the learned solution. OP-MF learns an effective rank closer to that of the ground-truth SVD solution. **(c)** Gradient Consistency measures the similarity of gradients across iterations. OP-MF is able to make a sudden change in optimization direction, while standard MF cannot. **(d)** Gradient Norm illustrates the scale of gradients. OP-MF is able more quickly adjust optimization step size.

paring MLP-generated low-rank matrices A and B with freely learned parameter matrices and measure convergence and gradients.

Matrix factorization decomposes a target matrix $M \in \mathbb{R}^{m\times n}$ into two lower-dimensional matrices, $A \in \mathbb{R}^{r\times n}$ and $B \in \mathbb{R}^{m\times r}$, where r is the latent dimension or rank. This decomposition allows us to approximate M by BA. It can be solved exactly with SVD, or as in our study, one can use gradient descent to minimize the reconstruction error:

$$\|M - BA\|_F^2,$$

where $\|\cdot\|_F$ denotes the Frobenius norm. This resembles LoRA-tuning, where the pre-trained base weights are set to all zeros and the target matrix M is the full finetuning gradient matrix, making it an interesting proxy problem to study.

Experimental setup and training protocol: We construct a synthetic target matrix $M \in \mathbb{R}^{100\times 100}$ with entries initialized uniformly at random from 0 to 1. The resulting matrix has a poor condition number, defined as the ratio between the largest singular value and lowest, making the optimization difficult and therefore a good test for MLP reparameterization. We train for 1000 steps with SGD, with linear warmup for 50 steps and linear learning rate decay.

OP-MF Model: The OP-MF model generates matrices A and B through two separate MLPs. We enforce both matrices to be of rank 8. Each MLP receives a learned input vector $z \in \mathbb{R}^{128}$ and processes it through two fully connected layers with 32 hidden units and ReLU activations, outputting the entries for either A or B. The second layer is heavily overparameterized; the parameter count is number of hidden units in the MLP by the size of the parameter matrix A or B. To align with LoRA's initialization strategy, the MLP for matrix B is initialized to output zeros, setting the model close to a pre-trained state.

Matrix Factorization(MF) Model: We train a MF model with freely learnable matrices A and B, initialized with random values for A and zeros for B. Again, both matrices have rank 8.

Finding 1: OP-MF rapidly adapts step size and direction. We examine the gradients, looking for evidence that OP-LoRA adaptively changes step size and direction. Our results reveal that as predicted, OP-MF shows an ability to rapidly adapt step sizes. This can be seen in Fig. 5 (d), where the gradient norm experiences a sharp spike, followed by a collapse, corresponding to acceleration and slow down in the loss curve in Fig. 5 (a). The sudden phase change in gradient norm also corresponds to a direction change in trajectory, measured by the cosine similarity between gradients at 10-step intervals in in Fig. 5 (c). Therefore, MLP reparameterization rapidly changes step size and direction, as suggested by the mathematical analysis in Section 3 of the main paper.

Finding 2: OP-MF is more effective at reaching the SVD solution than standard MF trained with SGD. In Fig. 5(a), we study the loss curves for both the MF model and the OP-MF model. The plot tracks MSE loss over 1000 iterations. The red line represents the SVD solution as a baseline. Interestingly, OP-MF solutions reach the best-case reconstruction error achieved with SVD, while Standard-MF cannot.

In addition to reconstruction error, another way to track progress towards a solution in matrix factorization is plotting the effective rank of the predicted ma-

trix BA over the course of training. Effective rank ρ is defined as

$$\rho = \exp\left(-\sum_{i=1}^{r} \sigma_i \log \sigma_i\right)$$

where σ_i represents the normalized singular values of BA, and r is the rank of the matrix. One would expect the effective rank to converge towards the effective rank of the ground-truth SVD solution for successful optimization runs.

In Fig. 5 (b), we observe the behavior of effective rank across iterations for both MF and OP-MF. We find that OP-MF can approximate the effective rank of the best-case SVD solution much more closely than standard MF.

Finding 3: OP-MF composes well with both SGD with Momentum and Adam. A natural question is if using standard acceleration methods like SGD with Momentum or Adam is enough. In Figure 6 we present experiments by adding Momentum to SGD and replacing it entirely with Adam, an optimizer that combines adaptive learning rates with momentum . Both Adam and SGD with Momentum improve reconstruction error for MF, but neither reach the SVD solution. Moreover, OP-MF composes with even best-case and advanced optimizers to find the SVD solution even faster, indicating their complimentary nature.

7.3. DreamBooth Subject-Driven Generation

We extend our image generation experiments to DreamBooth [39], finetuning SDXL and evaluating on the DreamBench benchmark, which consists of 30 diverse subjects, 25 prompts per subject, and 4 generations per prompt. We report CLIP-I (semantic similarity via CLIP embeddings), DINO (fine-grained visual similarity via self-supervised features), and CLIP-T (text-image alignment).

Metric	OP-LoRA	LoRA
CLIP-I (subject fidelity)	**0.83**	0.82
DINO (subject fidelity)	**0.69**	0.65
CLIP-T (text alignment)	0.24	**0.25**

Table 6. DreamBench results with SDXL. OP-LoRA improves subject fidelity (DINO) without sacrificing text alignment.

OP-LoRA outperforms LoRA on fine-grained visual similarity (DINO: 0.69 vs 0.65), without sacrificing performance on more global CLIP semantic similarity and text-image alignment. Qualitatively, OP-LoRA better preserves subject details like color and patches (Fig. 7).

7.4. LLaVA Image Classification

Zhang et al. [54] recently demonstrated that by applying simple finetuning to adapters, large multimodal models like LLaVA can achieve surprisingly high performance on a variety of classification tasks. We leverage this finding, replacing full finetuning with LoRA and OP-LoRA.

Datasets: We use the Stanford Cars [25] dataset, which is a fine-grained dataset of about 8000 training examples consisting of 196 classes of cars.

finetuning and evaluation protocol: We follow [54], and convert image classification into a captioning task by using the format "⟨ image⟩ What type of object is in this photo? ⟨ class name ⟩" and training with a language modeling objective. We finetune the visual projector layers of LLaVA1.5-7B for 50 epochs. We set MLP width to 768, since memory resources are not an issue for training only the adapter. At evaluation, we parse the generations and search for the correct class label.

Results: We present the results in Tab. 7. OP-LoRA consistently outperforms LoRA at both rank levels, achieving 83.6% at rank 16 and 87.1% at rank 64, compared to 82.8% and 86.3% with LoRA. This result further reinforces the efficacy of MLP reparameterization.

Rank	LoRA	OP-LoRA
16	82.8	**83.6**
64	86.3	**87.1**

Table 7. LLaVA1.5-7B Image Classification, Top-1 Accuracy on Stanford Cars [25].

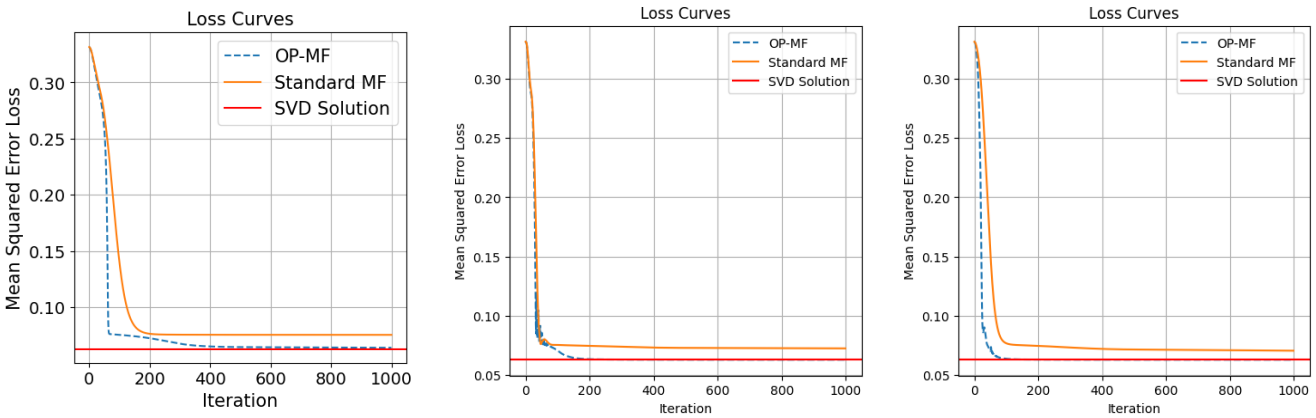
7.5. Stability of OP-DoRA

We found that OP-DoRA is more stable than DoRA, as shown with standard deviations across 3 runs. We hypothesize that this is a reflection of decreased learning rate sensitivity.

Method	Commonsense
DoRA ($r = 32$)	73.7 ± 6.7
OP-DoRA ($r = 32$)	77.5 ± 1.6

7.6. VeRA and OP-VeRA

We extend our method to VeRA [24], an ultra–low-parameter variant of LoRA, and evaluate on the GLUE[47] benchmark using a RoBERTa-base[30] backbone following the setup in [24]. We report results on



(a) SGD (b) Momentum (c) Adam

Figure 6. **Comparing Optimizers.** MLP reparameterization accelerates matrix factorization, even with advanced optimizers.

(a) LoRA (b) OP-LoRA (c) Reference subject

Figure 7. DreamBench: OP-LoRA preserves subject identity better, with more accurate color and patches. Prompt: *A [V] backpack in jungle.*

SST-2, CoLA, and QNLI. We exclude MNLI due to computational constraints and therefore also excludeMRPC/RTE/STS-B, which are commonly initialized from MNLI to mitigate overfitting [17].

	SST-2	CoLA	QNLI	**Avg.**
VeRA	94.0	59.8	**91.7**	81.8
OP-VeRA	**94.3**	**62.1**	91.5	**82.6**

Table 8. GLUE dev results with RoBERTa-base.

Averaged over the three tasks, OP-VeRA improves upon VeRA by **1.2** points, indicating the generality of the proposed optimization.

7.7. Improving Mix-of-Show with OP-LoRA

Following Zhang and Pilanci [52] directly, we apply Mix-of-Show to a small set of 14 training images of Harry Potter. We compare standard ScaledAdamW and OP-LoRA, keeping all training settings from Zhang and Pilanci [52]. In Figure 8, we generate images from the learned ¡potter¿ tokens as a prompt. We see that OP-LoRA better captures the subject of Harry Potter; there are fewer image of two people (not present in the training set) and the shape of the face is more accurate. Moreover, OP-LoRA generates Harry Potter in causal clothing less frequently.

7.8. Qualitative Stable Diffusion Results

In the main paper, we show the large quantitative gains OP-LoRA/OP-DoRA give in the image generation task (Table 1). We now present extensive random generations in Figures 9 though 26, with captions from the dataset as input. Several general trends emerge. First, there is a strong color bias towards red, however OP-LoRA and OP-DoRA reduce this dramatically. We attibute this improvement to the overparameterization easing optimization. Second, overparameterized LoRA is generates much more diverse and more complex scenes. Overall, qualitative results match the quantitative metrics.

8. Training Details

In this section, we summarize the training settings of our main experiments.

Hardware: Most experiments were done with a single H100 80GB HB3 GPU.

8.1. Initializing the OP-LoRA MLP

In Section 3, we introduce the MLP used to predict low rank parameters as

$$\begin{pmatrix} A \\ B \end{pmatrix} = W_2(\text{ReLU}(W_1 z + c_1)) + c_2$$

We initialize W_1 as Kaiming uniform. We also initalize W_2 as Kaiming uniform, except for parameters

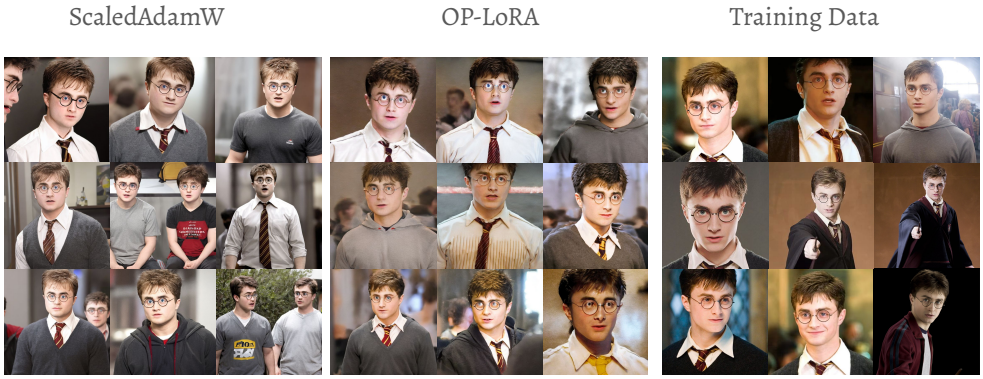


Figure 8. OP-LoRA vs ScaledAdamW[52] with Mix-of-Show[14].

Figure 9. GT Naruto[5] images

predicting the upsampling matrix B, which are initialized to zero to make the initialization not change the pre-trained model behavior. All biases are initialized to 0.

8.2. Training Hyperparameters

In Tables 9 through 12 we show training hyperparameters for experiments. These include batch sizes, learning rates, optimizer settings, and other configurations for each task. For the main results in the paper, we adopt hyperparameters tuned for the LoRA baselines and apply the same settings to OP-LoRA, without any additional tuning specific to OP-LoRA. Nevertheless, OP-LoRA consistently attains strong performance under these inherited hyperparameters, indicating that it is robust and effective even without additional hyperparameter optimization.

Figure 10. OP-LoRA Naruto[5] generated images.

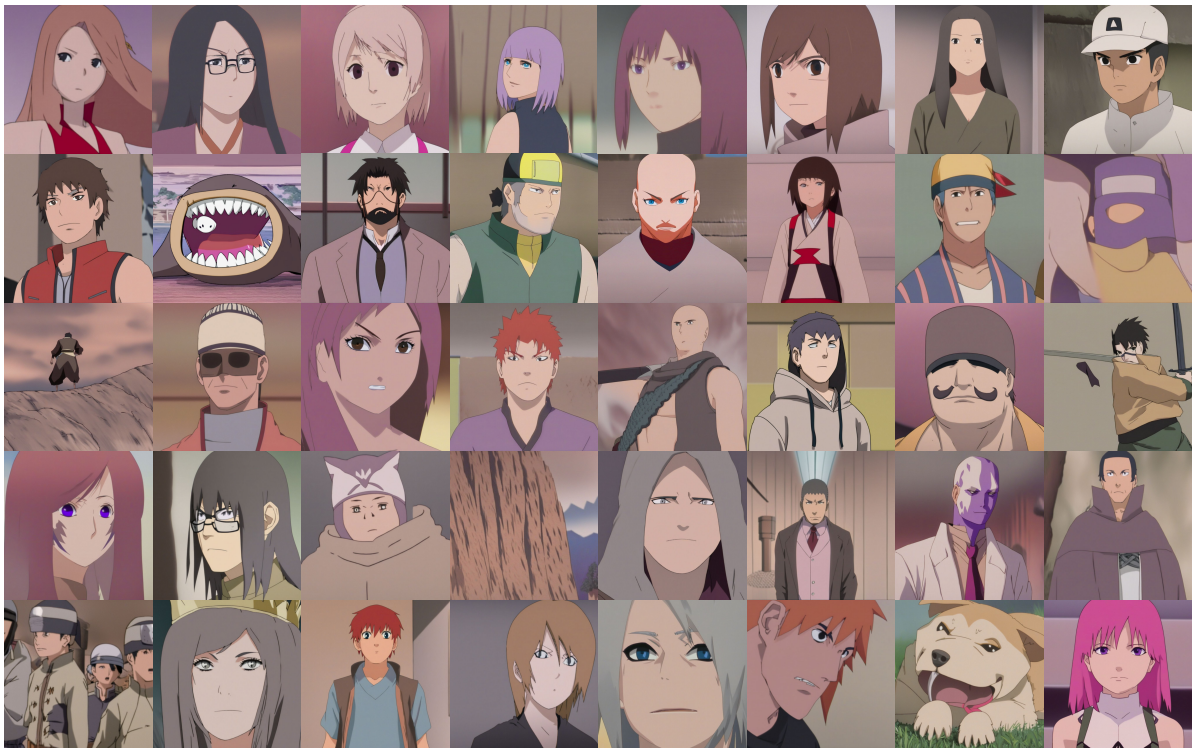


Figure 11. OP-DoRA Naruto[5] results

Figure 12. DoRA Naruto[5] generated images

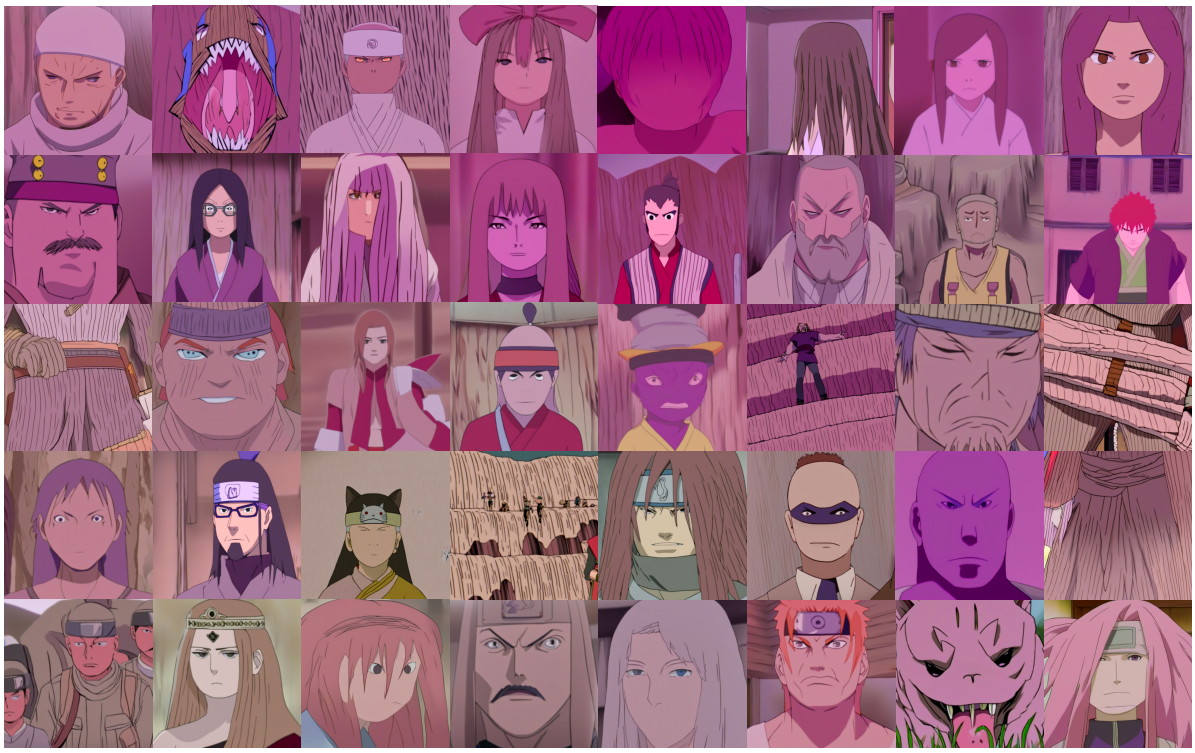


Figure 13. LoRA Naruto[5] generated images

Figure 14. PiSSA Naruto[5] generated images

Figure 15. LoRA-GA Naruto[5] generated images

Figure 16. LoRA-PRO Naruto[5] generated images

Figure 17. ScaledAdamW Naruto[5] generated images

Figure 18. GT Monet WikiArt[12] Images

Figure 19. OP-LoRA Monet WikiArt[12] Generated Images

Figure 20. OP-DoRA Monet WikiArt[12] Generated Images

Figure 21. DoRA Monet WikiArt[12] Generated Images

Figure 22. LoRA Monet WikiArt[12] Generated Images

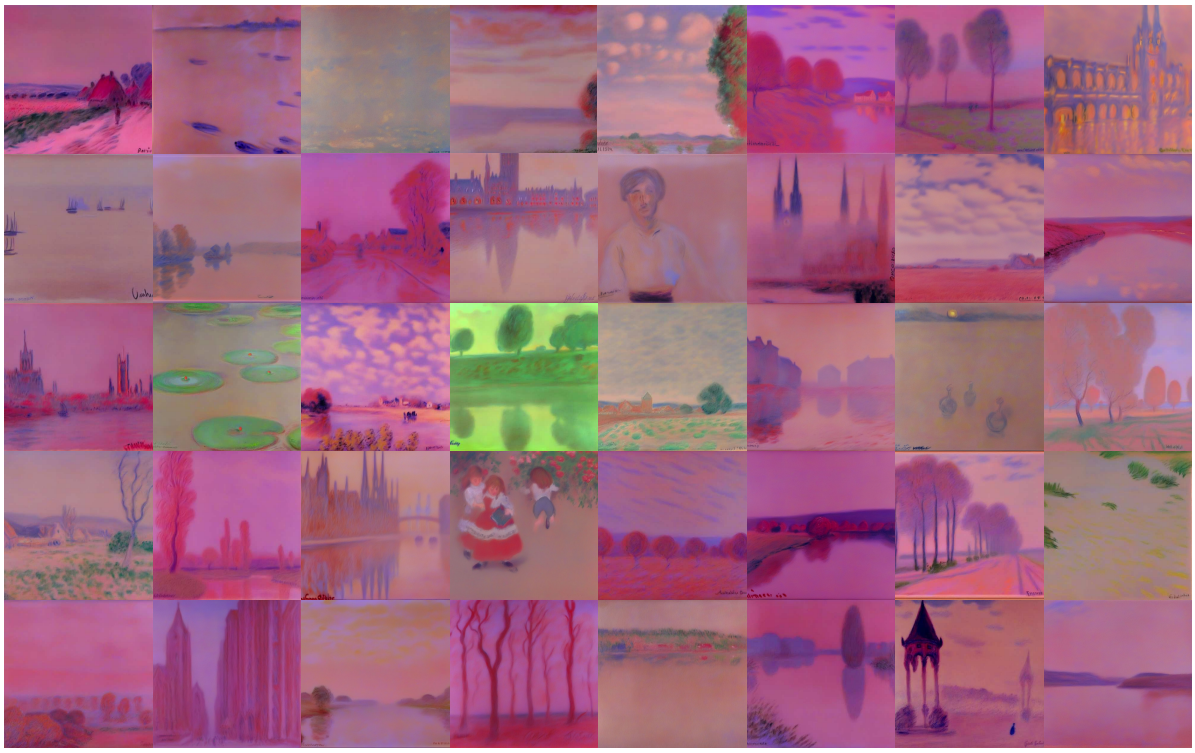


Figure 23. PiSSA Monet WikiArt[12] Generated Images

Figure 24. LoRA-GA Monet WikiArt[12] Generated Images

Figure 25. LoRA-Pro Monet WikiArt[12] Generated Images

Figure 26. ScaledAdamW Monet WikiArt[12] Generated Images

Hyperparameters	All
Base Model	Stable Diffusion XL 1.0[37]
Rank r	4
α	4
Dropout	0.0
Optimizer	AdamW
LR	1e-4
LR Scheduler	Constant
Batch size	1
Warmup Steps	0
Epochs	2
Where	U-Net Q, K, V, Out
MLP-Width(OP-LoRA/OP-DoRA)	32

Table 9. Training Details for Stable Diffusion Finetuning Experiments.

Hyperparameters	All
Base Model	VL-Bart[6]
Rank r	128
α	128
Dropout	0.0
Optimizer	AdamW
LR	1e-3
LR Scheduler	Linear
Batch size	300
Warmup ratio	0.1
Epochs	20
Where	Q,K (Bias Also Trained)
MLP-Width(OP-LoRA/OP-DoRA)	32(OP-LoRA)/4(OP-DoRA)

Table 10. Training Details for VL-Bart Finetuning Experiments.

Hyperparameters	All
Base Model	LLaVA1.5-7B[28]
Rank r	64/16
α	128/32
Dropout	0.0
Optimizer	AdamW
LR	5e-5
LR Scheduler	Cosine
Batch size	64
Warmup Ratio	0 .03
Epochs	50
Where	Multimodal Projector
MLP-Width(OP-LoRA/OP-DoRA)	768

Table 11. Training Details for LLaVA Classification Finetuning Experiments.

Hyperparameters	All
Base Model	LLaMA-7B[46]
Rank r	32
α	64
Dropout	0.05
Optimizer	AdamW
LR	1e-4
LR Scheduler	Linear
Batch size	16
Warmup ratio	0.03
Epochs	3
Where	Q,K, V, Up, Down
MLP-Width(OP-LoRA/OP-DoRA)	32

Table 12. Training Details for CommonSense Finetuning Experiments.