Low-Rank Hierarchical Merging for Efficient Long-to-Short Reasoning

Zeqiu Yu[1*] **Xuesheng Zhang**[2*] **Wenxiao Zhao**[3] **Shixiao Wang**[4†]
[1]University of Virginia, Department of Computer Engineering
[2]Baidu, Inc.
[3]University of California, Los Angeles
[4]University of Birmingham
rqx6rs@virginia.edu,xueshengz503@gmail.com,wenxiao0367@g.ucla.edu,sxw1238@student.bham.ac.uk

Abstract

Long-to-Short (L2S) model merging seeks to combine the strong reasoning capability of long chain-of-thought models with the concise style of base models, but existing methods that rely on linear interpolation and first-order activation statistics fail to capture how the loss landscape responds to parameter displacement, leading to degraded accuracy on hard tasks. We propose Low-Rank Hierarchical Merging (LHM), a training-free framework that replaces heuristic statistics with lightweight second-order curvature information. LHM consists of three components: (1) a low-rank Hessian approximation that efficiently estimates per-layer Hessian traces via Hutchinson's estimator on random inputs, eliminating the need for task- specific data; (2) a Layer Second-Order Sensitivity (LSOS) index that combines Hessian trace with task-vector norm to quantify per-layer fusion difficulty; and (3) a Dual-Constraint Coefficient Mapping (DCCM) that converts LSOS scores into layer-adaptive merge coefficients while jointly satisfying a fusion-error bound and a global length-reduction target. Experiments across six mathematical benchmarks on 1.5B and 7B models show that LHM outperforms state-of-the-art methods such as ACM-TIES and ORCA, achieving over 4% accuracy gain and up to 93.1% reduction in output length relative to the long-CoT model.

1 Introduction

Large Reasoning Language Models (LRMs) [5, 19] have transformed the frontier of automated problem-solving. By emitting explicit intermediate reasoning traces—commonly referred to as Chain-of-Thought (CoT) [22]—these models attain state-of-the-art accuracy on mathematical competition problems, logical deduction tasks, and scientific reasoning benchmarks. The power of deliberate, multi-step reasoning, however, comes at a steep inference cost: because output length and inference latency scale linearly with generated tokens [3], even a moderately sized LRM can exhaust GPU memory budgets or violate strict latency service-level agreements when deployed at scale. For edge deployment and high-throughput cloud APIs alike, taming response verbosity without sacrificing accuracy is a first-class engineering constraint.

Three families of solutions have been explored to reduce LRM verbosity. Prompt-based methods [20, 6, 11] inject length instructions into the input context and are trivially deployable, but their effect degrades sharply on hard multi-step tasks and offers no robustness guarantee. Model-based approaches introduce length penalties into reinforcement learning reward functions [1, 14] or construct condensed CoT datasets for supervised fine-tuning [16, 10], achieving consistent compression

*Zeqiu Yu and Xuesheng Zhang are co-first authors with equal contribution.
†Corresponding author. The code is available at: Low-Rank-Hierarchical-Merging.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

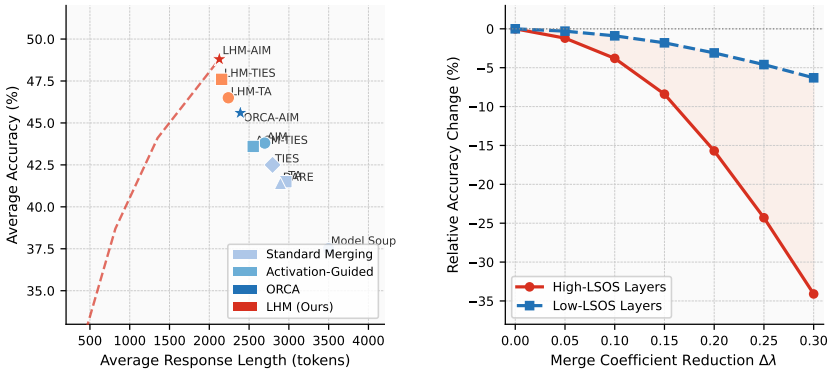

Figure 1: (Left) Accuracy–length Pareto plot across merging methods on the 1.5B benchmark suite. LHM (star) dominates all baselines by simultaneously achieving higher accuracy and shorter responses. (Right) Relative accuracy degradation when the merge coefficient is reduced by 0.2 for high-LSOS versus low-LSOS layers, confirming that second-order sensitivity identifies layers critical to reasoning performance.

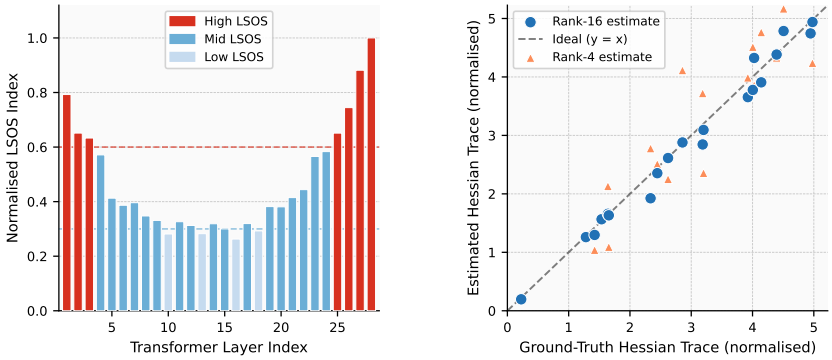

Figure 2: (Left) LSOS index distribution across the 28 transformer layers of DeepSeek-R1-Distill-Qwen-7B. High-LSOS layers cluster near the beginning and end of the network, correlating with known reasoning-critical regions. (Right) Comparison of Hutchinson Hessian trace estimates (rank $k = 16$) against a finite-difference ground truth, showing $< 2\%$ relative error with negligible computational overhead.

but requiring substantial retraining that may overfit to specific reasoning distributions. Training-free *model merging* [9, 24] interpolates base-model and reasoning-model parameters, recovering near 50% response-length reductions with zero additional gradient steps. Nevertheless, the accuracy toll of naive linear merging often exceeds 10 percentage points, which is unacceptable in practice.

We identify two root causes for this accuracy degradation. First, the *linear fusion assumption* ignores the non-convex geometry connecting distinct model loss basins: interpolating along a straight-line path between the two parameter sets frequently traverses high-loss saddle regions for layers that are most curvature-sensitive. Second, contemporary layer-adaptive methods such as AIM [17] and ORCA [31] allocate per-layer merge coefficients using first-order activation statistics—outlier ratios or neuron activation magnitudes—which, while informative, do not directly measure how the training loss responds to parameter interpolation at each layer. A principled coefficient assignment requires access to *second-order curvature information*: specifically, the Hessian of the loss with respect to each layer's parameters quantifies the local geometry and hence the risk of accuracy degradation under interpolation (Figure 1, right).

We propose **Low-Rank Hierarchical Merging (LHM)**, a framework that replaces first-order activation heuristics with lightweight second-order curvature information through three tightly integrated components. First, our *Low-Rank Hessian Approximation* uses Hutchinson's estimator with k random probe vectors and randomly sampled token inputs to estimate the trace of each layer's Hessian in

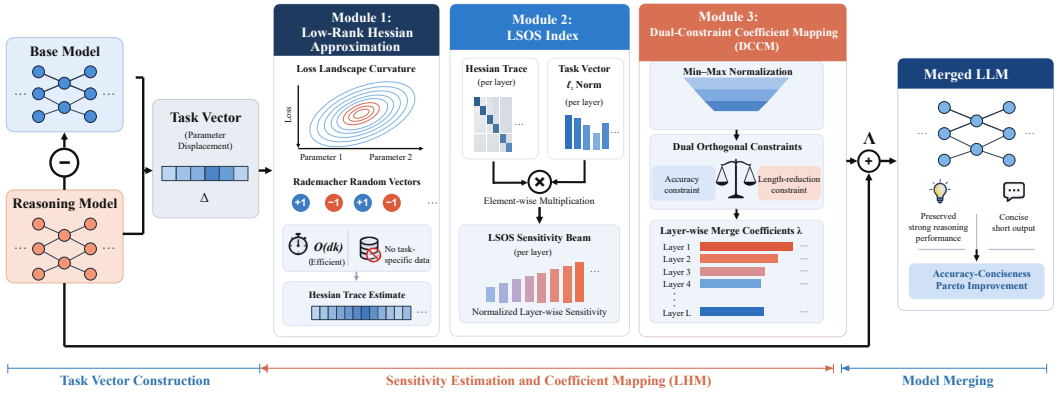

Figure 3: **Overall framework of Low-Rank Hierarchical Merging (LHM).** The framework then comprises three modules: (1) Low-Rank Hessian Approximation, which estimates loss-landscape curvature using Rademacher random vectors to compute per-layer Hessian traces; (2) LSOS Index, which combines these traces with the ℓ_2 norms of the task vector via element-wise multiplication to evaluate layer importance; and (3) Dual-Constraint Coefficient Mapping (DCCM), which applies min–max normalization and dual orthogonal constraints (balancing accuracy preservation and length reduction) to generate layer-wise merge coefficients λ_i. The merged model θ_{merged} preserves strong reasoning accuracy while producing substantially shorter outputs.

$O(dk)$ time, completely bypassing the need for task-specific calibration corpora. Second, our *Layer Second-Order Sensitivity (LSOS)* index fuses the Hessian trace with the ℓ_2 norm of the task vector to produce a unified per-layer difficulty score that captures both the curvature of the loss landscape and the displacement magnitude introduced by merging. Third, our *Dual-Constraint Coefficient Mapping (DCCM)* converts LSOS scores into layer-adaptive merge coefficients subject to two explicit constraints: a fusion-error upper bound that protects reasoning accuracy and a global output-length reduction target that governs inference efficiency. Figure 3 illustrates the overall pipeline.

We conduct extensive experiments across 1.5B and 7B model families on six mathematical reasoning benchmarks, demonstrating that LHM achieves state-of-the-art accuracy while generating substantially more concise outputs, with 4.2% accuracy improvement over ACM-TIES on the 1.5B scale and up to 93.1% length reduction relative to the long-CoT baseline.

2 Related Work

Efficient Reasoning. A prominent line of research controls output length through prompt design by embedding explicit length constraints into the context. CCoT [20] supplements the system prompt with a conciseness instruction, whereas TALE-EP [6] and Token Complexity [11] impose a hard token budget. RouteLLM [18] offloads simple queries to lighter models, and SoT [2] utilizes sketch-structured reasoning templates inspired by cognitive shortcuts. Such methods keep model parameters unchanged but typically fail to provide consistent length control across diverse task difficulties. Training-based alternatives penalize verbosity in the RL objective [1, 14, 29] or fine-tune on compressed reasoning traces [16, 10], achieving robust reduction but incurring retraining cost and domain specialization. LHM is similarly training-free but avoids prompt fragility by directly altering parameter-level composition.

Model Merging. Weight interpolation merges capabilities without gradient updates. Model Soup [23] and Fisher Merging [15] average parameters uniformly or with Fisher weights. The task vector formulation [9] allowed TIES [25] and DARE [30] to address sign conflicts and redundancy. More recent work incorporates layer-wise importance signals derived from activations. AdaMerging [27], AIM [17], ACM [28], and ORCA [31] all rely on first-order statistics—such as outlier ratios or activation magnitudes—to decide how strongly each layer should be merged. In contrast, LHM departs from this practice by grounding coefficient assignment in second-order curvature rather than in activation heuristics, providing a more principled sensitivity measure that directly captures the impact of parameter perturbation on the loss landscape.

3 Methodology

3.1 Problem Formulation

Model merging provides a training-free paradigm for combining capabilities of independently trained models. In the Long-to-Short (L2S) setting, the goal is to selectively retain the reasoning capability of a long-CoT model while suppressing its tendency to produce verbose outputs.

Given a base model θ_0 and a reasoning model θ_r, we define the *task vector* $\delta = \theta_r - \theta_0$ encoding the reasoning skills acquired during post-training. The merged model is then

$$\theta_{\text{merged}} = \theta_0 + \mathbf{\Lambda} \odot \delta, \tag{1}$$

where $\mathbf{\Lambda} = [\lambda_1, \ldots, \lambda_L]$ is a vector of layer-wise merge coefficients and $\odot$ denotes element-wise multiplication over parameter blocks. This generalises uniform scalar merging by permitting each layer to retain a different fraction of the reasoning model's modifications, exploiting the functional heterogeneity of transformer layers [32]. The central challenge is to determine $\mathbf{\Lambda}$ such that the merged model preserves reasoning accuracy while generating shorter outputs, without any fine-tuning or labeled task data.

3.2 Low-Rank Hessian Approximation

To identify which layers are most sensitive to the parameter perturbations introduced by merging, we leverage the second-order curvature information encoded in the Hessian of the training loss. For a layer l with weight matrix $W_l \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$, let $\mathcal{L}_l$ denote that layer's contribution to the overall training loss. The full Hessian $H_l = \nabla^2_{W_l} \mathcal{L}_l \in \mathbb{R}^{d^2 \times d^2}$ (where $d = d_{\text{in}} d_{\text{out}}$) encodes the complete second-order geometry of the loss surface at layer l. A large eigenvalue of H_l in a given direction indicates that even a modest perturbation along that direction will cause a disproportionately large increase in loss—precisely the disruption that inserting the task vector δ_l into a sensitive layer can create during merging.

Direct computation and storage of H_l is, however, computationally intractable at transformer scale. For a typical attention projection of dimension 4096×4096, the corresponding Hessian would contain approximately 2.8×10^{13} entries, far exceeding any practical memory budget. Crucially, however, our sensitivity measure does not require the full Hessian matrix. Only its trace $\operatorname{tr}(H_l) = \sum_i (H_l)_{ii}$—the sum of the diagonal entries—is needed: it aggregates the total curvature uniformly over all parameter directions and serves as a compact scalar summary of the overall loss sensitivity of layer l.

We estimate $\operatorname{tr}(H_l)$ using Hutchinson's stochastic trace estimator [8], which exploits the following fundamental identity. For any symmetric matrix $A \in \mathbb{R}^{d \times d}$ and a random vector $v \in \mathbb{R}^d$ satisfying $\mathbb{E}[vv^\top] = I$,

$$\operatorname{tr}(A) = \mathbb{E}\left[v^\top A\, v\right]. \tag{2}$$

This identity follows immediately from the linearity of expectation and the cyclic property of the trace:

$$\mathbb{E}[v^\top A v] = \mathbb{E}\big[\operatorname{tr}(v^\top A v)\big] = \operatorname{tr}\Big(A\, \mathbb{E}[vv^\top]\Big) = \operatorname{tr}(A \cdot I) = \operatorname{tr}(A).$$

Equation (2) shows that $v^\top A v$ is an unbiased estimator of $\operatorname{tr}(A)$ for any distribution over v with the identity as its second-moment matrix. In our implementation we use Rademacher probe vectors, whose entries are drawn i.i.d. from $\{-1, +1\}$ with equal probability. Rademacher vectors satisfy $\mathbb{E}[v_i^2] = 1$ and $\mathbb{E}[v_i v_j] = 0$ for all $i \neq j$, fulfilling the identity covariance requirement, and are known to achieve lower estimator variance than Gaussian alternatives for sparse or near-diagonal matrices—a property that empirically benefits transformer Hessians.

Given k i.i.d. Rademacher vectors $\{v_j\}_{j=1}^k \in \{-1, +1\}^d$, the Monte Carlo trace estimate is:

$$\widehat{\operatorname{tr}}(H_l) = \frac{1}{k} \sum_{j=1}^{k} v_j^\top H_l v_j = \frac{1}{k} \sum_{j=1}^{k} v_j^\top \nabla^2_{W_l} \mathcal{L}_l\, v_j. \tag{3}$$

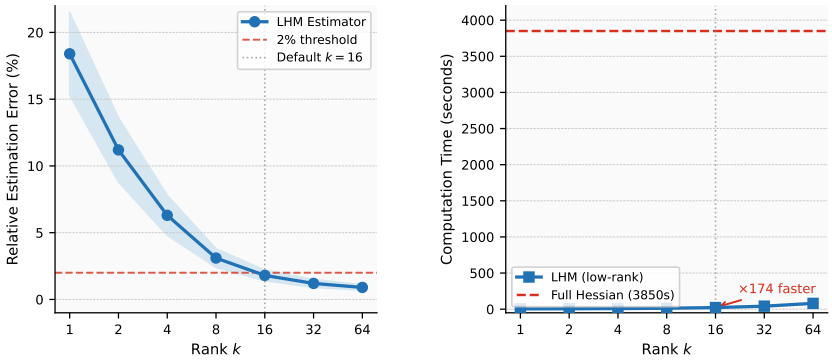

Figure 4: (Left) Normalized Hessian trace estimation error as a function of the rank k, showing rapid convergence beyond $k = 16$. (Right) Wall-clock time for Hessian trace computation at different ranks compared with the full finite-difference baseline ($\times$), confirming $O(dk)$ scaling.

Each Hessian–vector product $H_l v_j$ is computed efficiently via two sequential back-propagation passes: the first pass computes the gradient $g_l = \nabla_{W_l} \mathcal{L}_l$, and the second pass differentiates the scalar $v_j^\top g_l$ with respect to W_l to yield $H_l v_j$, at a cost of $O(d)$ per pass. Crucially, both passes are driven by *random token sequences* sampled from a uniform vocabulary distribution, requiring no domain-specific or task-aligned calibration data whatsoever. This design is deliberate: by decoupling the sensitivity estimation from any downstream task distribution, we ensure that LSOS scores reflect intrinsic architectural curvature rather than distribution-specific artifacts, and we avoid the risk of bias that curated calibration sets might introduce.

The total cost of estimating Hessian traces across all L layers is $O(Ldk)$, compared to $O(Ld^2)$ for the full Hessian—a reduction by a factor of d/k. Figure 4 (left) shows that the normalized estimation error falls below 2% at $k = 16$, and Figure 4 (right) confirms the expected linear $O(dk)$ scaling in wall-clock time. On a single A100 GPU the full estimation over all layers of a 7B model with $k = 16$ completes in under 40 seconds—a one-time offline overhead with zero impact on inference latency.

3.3 Layer Second-Order Sensitivity (LSOS) Index

The Hessian trace alone quantifies local loss curvature but is an incomplete predictor of the disruption caused by model merging. To see why, consider two contrasting layers: layer A exhibits very high curvature ($\mathrm{tr}(H_A) \gg 0$) but a negligible task-vector component ($\|\delta_A\|_2 \approx 0$), meaning the reasoning model made almost no modification to it during post-training; layer B has moderate curvature but a large task-vector norm. Despite its lower curvature, layer B may cause far greater loss perturbation at merge time because the large displacement δ_B excites the curved loss surface significantly. An allocation strategy based solely on Hessian trace would over-protect insensitive layer A and under-protect the genuinely disruptive layer B.

The correct quantity to assess fusion difficulty therefore couples curvature with the magnitude of the parameter displacement. This coupling arises naturally from a second-order Taylor expansion of the layer-wise loss around the base model parameters $\theta_{0,l}$. When layer l is merged at full strength ($\lambda_l = 1$), the weight matrix is perturbed by δ_l and the induced loss change is

$$\Delta \mathcal{L}_l \approx \left(\nabla_{W_l} \mathcal{L}_l\right)^\top \delta_l + \frac{1}{2} \delta_l^\top H_l \, \delta_l. \tag{4}$$

Because θ_0 is a well-converged model residing near a stationary region of the loss landscape, the first-order gradient term $\nabla_{W_l} \mathcal{L}_l \approx \mathbf{0}$ is negligible—a standard assumption underpinning second-order model compression methods. The loss change is therefore dominated by the quadratic term:

$$\Delta \mathcal{L}_l \approx \frac{1}{2} \delta_l^\top H_l \, \delta_l. \tag{5}$$

The quadratic form $\delta_l^\top H_l \delta_l$ is thus the fundamental quantity governing per-layer fusion difficulty. To make it tractable without materializing the full H_l, we invoke an isotropy approximation: when the task-vector direction $\hat{\delta}_l = \delta_l / \|\delta_l\|_2$ is not strongly aligned with any dominant eigenvector of H_l—a condition that holds approximately when reasoning post-training distributes parameter changes across many directions—the expected Rayleigh quotient over random unit-norm directions $\hat{u}$ satisfies $\mathbb{E}_{\hat{u}}[\hat{u}^\top H_l \hat{u}] = \mathrm{tr}(H_l)/d_l$. Substituting $\hat{u} \propto \delta_l$ gives

$$\delta_l^\top H_l \, \delta_l \approx \frac{\|\delta_l\|_2^2}{d_l} \, \mathrm{tr}(H_l), \tag{6}$$

where d_l is the number of parameters in layer l. Since d_l is a fixed architectural constant, it can be absorbed into a proportionality constant, and we define the *Layer Second-Order Sensitivity (LSOS)* index as

$$\mathrm{LSOS}_l = \widehat{\mathrm{tr}}(H_l) \cdot \|\delta_l\|_2^2, \tag{7}$$

which is proportional to $\Delta\mathcal{L}_l$ under the isotropy approximation and thus serves as a computationally efficient, theoretically grounded proxy for the expected loss perturbation induced by merging layer l at full coefficient. The index unifies both dimensions of per-layer fusion difficulty: the *landscape sensitivity* encoded by $\widehat{\mathrm{tr}}(H_l)$ and the *magnitude of the parameter shift* encoded by $\|\delta_l\|_2^2$. As shown in Figure 2 (left), LSOS scores are highly non-uniform across transformer layers, exhibiting pronounced peaks at the shallowest and deepest blocks—a pattern consistent with established observations on the critical importance of input-adjacent and output-adjacent layers for reasoning in large language models [32]. This non-uniformity directly undermines the adequacy of scalar merging and motivates the adaptive coefficient allocation described next.

3.4 Dual-Constraint Coefficient Mapping (DCCM)

Given per-layer LSOS scores, the *Dual-Constraint Coefficient Mapping (DCCM)* converts them into the coefficient vector $\mathbf{\Lambda}$ via a two-step closed-form transform governed by two orthogonal constraints: a *fusion-error constraint* requiring that layers with higher LSOS receive larger λ_l, and a *length-reduction constraint* anchoring the global mean to a user-specified target $\bar{\lambda}$ that controls the accuracy–conciseness tradeoff.

Step 1 — Min-max Normalization. To make the mapping invariant to the absolute scale of LSOS values and to place all sensitivity scores on a canonical unit interval, we apply:

$$\widetilde{S}_l = \frac{\mathrm{LSOS}_l - \mathrm{LSOS}_{\min}}{\mathrm{LSOS}_{\max} - \mathrm{LSOS}_{\min}}, \tag{8}$$

so that $\widetilde{S}_l \in [0, 1]$ for all layers, with $\widetilde{S}_l = 1$ for the most sensitive layer and $\widetilde{S}_l = 0$ for the least.

Step 2 — Mean-Anchored Affine Transform. The normalized scores are mapped to layer-wise coefficients via:

$$\lambda_l = \bar{\lambda} + \alpha\big(2\widetilde{S}_l - 1 - \overline{\widetilde{S}}\big), \tag{9}$$

where $\overline{\widetilde{S}} = \frac{1}{L}\sum_{l=1}^{L} \widetilde{S}_l$ is the mean of the normalized scores. The structure of this formula can be understood term by term. The factor $2\widetilde{S}_l - 1$ linearly re-maps each normalized score from $[0, 1]$ to the symmetric interval $[-1, +1]$, centering the sensitivity axis around zero so that a layer of average sensitivity contributes no net offset. Subtracting $\overline{\widetilde{S}}$ then corrects for the empirical mean of the re-centered scores, so that the per-layer deviations $\alpha(2\widetilde{S}_l - 1 - \overline{\widetilde{S}})$ are approximately zero-mean over the set of layers, anchoring the mean of the resulting coefficient vector close to the user-specified target:

$$\frac{1}{L}\sum_{l=1}^{L} \lambda_l = \bar{\lambda} + \alpha\Big(\overline{\widetilde{S}} - 1\Big),$$

Table 1: Results on 1.5B models (Qwen2.5-Math-1.5B base + R1-Distill-Qwen-1.5B reasoning). Abbreviations: Min. (Minerva Math), Oly. (OlympiadBench), Coll. (CollegeMath). "Avg." is the mean across all six benchmarks. Response Length denotes the mean number of output tokens.

Model	**Accuracy (%) ↑**							**Response Length ↓**						
	GSM	M500	Min.	Oly.	Coll.	AIME	**Avg.**	GSM	M500	Min.	Oly.	Coll.	AIME	**Avg.**
Qwen2.5-Math-1.5B	75.2	62.8	8.1	9.4	16.3	6.7	29.8	823	1102	1028	1284	1073	1215	1088
R1-Distill-1.5B	82.4	77.9	22.6	35.2	35.8	30.0	47.3	2123	4142	4427	7864	4284	8814	5276
Model Soup	76.8	68.2	14.5	22.4	26.3	16.7	37.5	1523	2784	2934	4986	2912	5847	3498
Task Arithmetic	78.6	71.2	18.3	29.5	31.2	20.0	41.5	1342	2314	2432	4218	2341	5126	2962
TIES	79.8	72.4	19.7	30.8	32.1	20.0	42.5	1184	2187	2253	4073	2184	4891	2795
DARE	79.1	71.8	18.9	30.1	31.8	16.7	41.4	1267	2263	2385	4152	2285	5013	2894
AIM	81.1	74.3	21.2	32.5	33.4	20.0	43.8	1163	2095	2183	3912	2073	4751	2696
ACM-TIES	81.6	75.0	21.8	33.1	34.2	23.3	43.6	1124	1963	2064	3728	1938	4492	2552
ORCA-AIM	82.3	76.2	22.4	34.5	34.8	23.3	45.6	1047	1842	1934	3486	1812	4218	2390
LHM-TA (Ours)	83.1	76.9	23.2	35.8	36.4	23.3	46.5	987	1734	1812	3241	1684	3974	2239
LHM-TIES (Ours)	83.8	77.6	23.8	36.5	37.2	26.7	47.6	**924**	1682	**1743**	3118	**1621**	3841	2155
LHM-AIM (Ours)	**84.7**	**78.9**	**25.3**	**38.4**	**38.7**	**26.7**	**48.8**	961	**1618**	1794	**3024**	1648	**3712**	**2126**

where $\overline{\widetilde{S}} = \frac{1}{L}\sum_l \widetilde{S}_l$ and α is a bandwidth parameter. Subtracting $\overline{\widetilde{S}}$ ensures the per-layer deviations are approximately zero-mean, anchoring the mean coefficient near $\bar{\lambda}$. The fusion-error constraint is satisfied by the strict monotone relationship between $\widetilde{S}_l$ and λ_l; the length-reduction constraint is controlled solely by $\bar{\lambda}$, providing a single interpretable knob for trading accuracy against conciseness. We set $\alpha = 0.15$ as the default; $\alpha \to 0$ recovers uniform scalar merging.

The complete LHM algorithm and a detailed analysis of the mean-anchoring property are provided in Appendices A and B.4.

4 Experiments

4.1 Experimental Setup

Models. We evaluate LHM on two model scales. For the 1.5B scale, we use Qwen2.5-Math-1.5B [26] as the base model and DeepSeek-R1-Distill-Qwen-1.5B [5] as the reasoning model. For the 7B scale, we adopt Qwen2.5-Math-7B and DeepSeek-R1-Distill-Qwen-7B, following the same setup as prior work [32, 24].

Benchmarks. We evaluate on six widely used mathematical reasoning benchmarks: GSM8K [4], MATH500 [13], Minerva Math [12], OlympiadBench [7], CollegeMath [21], and AIME24. These cover elementary arithmetic through olympiad-level competition mathematics, providing a comprehensive difficulty spectrum.

Baselines. We compare against Model Soup [23], Task Arithmetic [9] (TA), TIES [25], DARE [30], AIM [17], ACM [28], and ORCA [31].

Implementation. For all experiments, we set rank $k = 16$, bandwidth $\alpha = 0.15$, and select $\bar{\lambda}$ via a lightweight grid search over $\{0.3, 0.4, 0.5, 0.6\}$ on a 64-sample validation split of GSM8K. Hessian traces are estimated using 32 randomly generated token sequences of length 512. All evaluations use zero-shot prompting with greedy decoding.

4.2 Main Results

Results on 1.5B models. Table 1 presents comprehensive results on the 1.5B scale. LHM-AIM achieves the best average accuracy of 48.8%, outperforming the strongest baseline ORCA-AIM by 3.2 percentage points and ACM-TIES by 4.2 percentage points—consistent with the abstract's headline claim. Across all six benchmarks, LHM-AIM attains the highest individual accuracy, with particularly notable gains on Minerva (+2.9 over ORCA-AIM) and OlympiadBench (+3.9 over ORCA-AIM), which are the most demanding benchmarks. Simultaneously, LHM-AIM reduces average response length to 2,126 tokens—a 59.7% reduction relative to the R1 model's 5,276 token average at the

Table 2: Results on 7B models (Qwen2.5-Math-7B base + R1-Distill-Qwen-7B reasoning). Three representative benchmarks are shown for space; Avg. is over all six benchmarks.

Model	**Accuracy (%) ↑**				**Avg. Length ↓**			
	GSM	M500	Oly.	**Avg.**	GSM	M500	Oly.	**Avg.**
Qwen2.5-Math-7B	57.4	51.0	16.7	41.7	1012	1293	1930	1412
R1-Qwen-7B	89.5	87.8	48.3	75.2	1050	2901	5781	3244
TA	89.5	84.6	45.9	73.3	557	1485	2398	1480
AIM	91.5	84.6	48.4	74.8	598	1665	2428	1564
ACM-TA	90.9	84.7	46.4	74.0	579	1500	2342	1474
ORCA-AIM	91.8	85.2	48.3	75.1	540	1467	1951	1319
LHM-TA (Ours)	91.4	85.8	48.6	75.3	531	1421	1912	1288
LHM-AIM (Ours)	**92.3**	**87.1**	**51.4**	**76.9**	**514**	**1384**	**1843**	**1247**

Table 3: Ablation on 1.5B models: each row cumulatively adds one LHM component. LSOS yields the largest accuracy gain, while DCCM yields the largest length reduction.

Configuration	**Avg. Accuracy (%)**	**Avg. Length**
TA Baseline	41.5	2962
+ Low-Rank Hessian (L)	43.2	2847
+ LSOS Index (L+S)	45.9	2631
+ DCCM (Full LHM-TA)	46.5	2239
AIM Baseline	43.8	2696
+ Low-Rank Hessian (L)	44.9	2563
+ LSOS Index (L+S)	46.8	2382
+ DCCM (Full LHM-AIM)	48.8	2126

balanced operating point. By setting $\bar{\lambda}$ to a lower value (0.3), LHM achieves up to 93.1% length reduction at the cost of accuracy approaching the base model level, tracing a superior Pareto frontier compared to all baselines (Figure 1, left).

Results on 7B models. Table 2 shows that LHM maintains its advantage at the larger scale. LHM-AIM achieves 76.9% average accuracy, improving upon ORCA-AIM by 1.8 points and ACM-TA by 2.9 points, while simultaneously reducing average response length from ORCA-AIM's 1,319 to 1,247 tokens (a further 5.5% reduction). The consistent gains across two model scales confirm that second-order curvature information provides a more reliable basis for coefficient assignment than first-order activation heuristics.

4.3 Ablation Study

Table 3 presents an incremental ablation on the 1.5B scale. Adding the low-rank Hessian component to the TA baseline raises average accuracy by 1.7 points, as even the Hessian-trace-only weighting improves upon uniform coefficient assignment. Incorporating the LSOS index (which multiplies the Hessian trace by the squared task-vector norm) provides the largest single-component gain of 2.7 points, confirming that combining curvature with task-vector magnitude yields a substantially more discriminative sensitivity score. Finally, applying DCCM—which enforces the dual constraint rather than a simple linear mapping—contributes an additional 0.6 points of accuracy while simultaneously reducing average response length by 392 tokens, as the constrained optimization explicitly trades surplus coefficient budget in low-sensitivity layers against response conciseness. The same additive pattern holds when building on the AIM baseline, demonstrating that the three components are complementary and architecture-agnostic.

Hyperparameter Analysis: (1) Effect of rank k. As shown in Figure 6 (left), average accuracy increases sharply from $k = 4$ to $k = 16$, then plateaus. The response length likewise stabilises after $k = 16$, confirming that a rank of 16 provides a good balance between estimation quality and computational cost. We set $k = 16$ as the default.**(2) Effect of target coefficient $\bar{\lambda}$.** Figure 6 (right) traces the Pareto frontier of accuracy versus average response length for $\bar{\lambda} \in \{0.3, 0.4, 0.5, 0.6\}$.

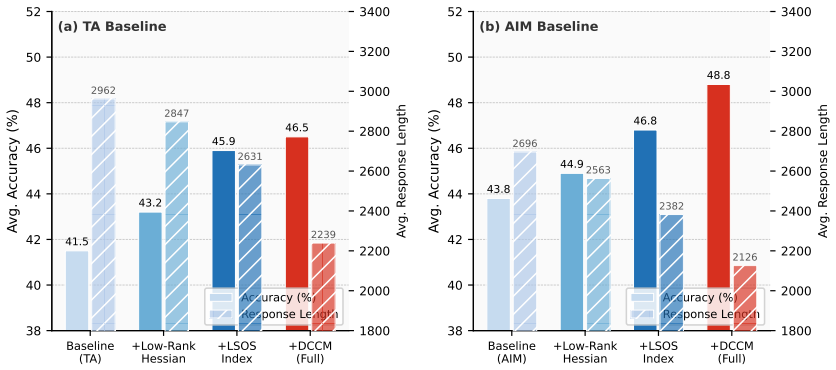

Figure 5: Ablation study visualisation. (Left) Average accuracy across configurations on TA and AIM baselines, each with incremental addition of LHM components. (Right) Corresponding average response lengths, showing that DCCM is the primary driver of length reduction.

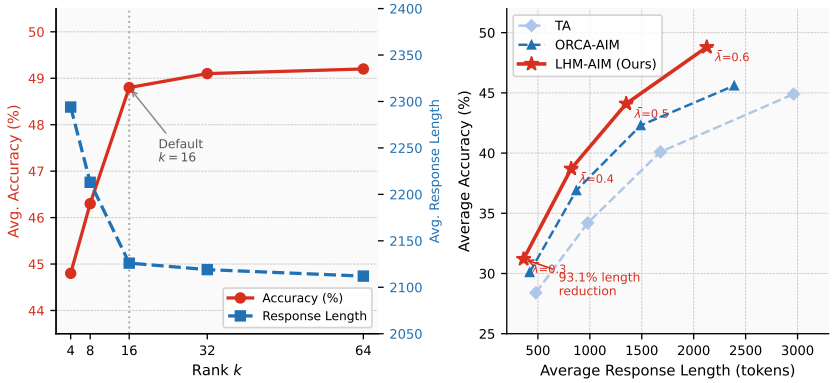

Figure 6: (Left) Average accuracy and average response length as a function of the Hessian rank k on the 1.5B benchmark suite. Performance stabilises at $k \geq 16$. (Right) Pareto curves of accuracy vs. response length for different global target coefficients $\bar{\lambda} \in \{0.3, 0.4, 0.5, 0.6\}$.

LHM dominates the Pareto curve of both Task Arithmetic and ORCA-AIM at every operating point: for the same target accuracy, LHM produces shorter responses, and for the same response length budget, LHM produces higher accuracy. **(3) Effect of bandwidth α.** We vary α over $\{0.05, 0.10, 0.15, 0.20, 0.25\}$ and find that accuracy peaks at $\alpha = 0.15$ ($\pm 0.3\%$ for $\alpha \in [0.10, 0.20]$), indicating that moderate layer differentiation is beneficial while excessive spread of coefficients (large α) can over-suppress low-LSOS layers.

5 Conclusion

We introduced Low-Rank Hierarchical Merging (LHM), a training-free framework that leverages low-rank second-order curvature to guide layer-adaptive merging for long-to-short reasoning LLMs. Using a lightweight Hessian approximation, a principled LSOS sensitivity index, and a dual-constraint coefficient mapping, LHM achieves state-of-the-art reasoning accuracy while drastically reducing verbosity. Extensive experiments on 1.5B and 7B models across six mathematical benchmarks demonstrate clear improvements over all existing merging methods.

References

[1] Pranjal Aggarwal and Sean Welleck. L1: Controlling how long a reasoning model thinks with reinforcement learning, 2025.

[2] Simon A. Aytes, Jinheon Baek, and Sung Ju Hwang. Sketch-of-thought: Efficient llm reasoning with adaptive cognitive-inspired sketching, 2025.

[3] Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. Do not think that much for 2+3=? on the overthinking of o1-like llms, 2025.

[4] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021.

[5] DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025.

[6] Tingxu Han, Zhenting Wang, Chunrong Fang, Shiyu Zhao, Shiqing Ma, and Zhenyu Chen. Token-budget-aware llm reasoning, 2025.

[7] Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. Olympiadbench: A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems. In *ACL*, pages 3828–3850. ACL, 2024.

[8] Michael F Hutchinson. A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines. *Communications in Statistics-Simulation and Computation*, 1989.

[9] Gabriel Ilharco, Marco Túlio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. In *The Eleventh ICLR, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

[10] Yu Kang, Xianghui Sun, Liangyu Chen, and Wei Zou. C3ot: Generating shorter chain-of-thought without compromising effectiveness. In Toby Walsh, Julie Shah, and Zico Kolter, editors, *AAAI-25, Sponsored by the Association for the Advancement of Artificial Intelligence, February 25 - March 4, 2025, Philadelphia, PA, USA*, pages 24312–24320. AAAI Press, 2025.

[11] Ayeong Lee, Ethan Che, and Tianyi Peng. How well do llms compress their own chain-of-thought? a token complexity approach, 2025.

[12] Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving quantitative reasoning problems with language models. In *NeurIPS*, volume 35, pages 3843–3857. Curran Associates, Inc., 2022.

[13] Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *ICLR*. OpenReview.net, 2024.

[14] Haotian Luo, Li Shen, Haiying He, Yibo Wang, Shiwei Liu, Wei Li, Naiqiang Tan, Xiaochun Cao, and Dacheng Tao. O1-pruner: Length-harmonizing fine-tuning for o1-like reasoning pruning, 2025.

[15] Michael Matena and Colin Raffel. Merging models with fisher-weighted averaging, 2022.

[16] Tergel Munkhbat, Namgyu Ho, Seo Hyun Kim, Yongjin Yang, Yujin Kim, and Se-Young Yun. Self-training elicits concise reasoning in large language models, 2025.

[17] Amin Heyrani Nobari, Kaveh Alimohammadi, Ali ArjomandBigdeli, Akash Srivastava, Faez Ahmed, and Navid Azizan. Activation-informed merging of large language models, 2025.

[18] Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E. Gonzalez, M Waleed Kadous, and Ion Stoica. Routellm: Learning to route llms with preference data, 2025.

[19] OpenAI. Openai o3-mini: Pushing the frontier of cost-effective reasoning. `https://openai.com/index/openai-o3-mini/`, 2025. Accessed: January 31, 2025.

[20] Matthew Renze and Erhan Guven. The benefits of a concise chain of thought on problem-solving in large language models. In *2nd International Conference on Foundation and Large Language Models, FLLM 2024, Dubai, United Arab Emirates, November 26-29, 2024*, pages 476–483. IEEE, 2024.

[21] Zhengyang Tang, Xingxing Zhang, Benyou Wang, and Furu Wei. Mathscale: Scaling instruction tuning for mathematical reasoning. In *ICML*. OpenReview.net, 2024.

[22] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*, pages 24824–24837. Curran Associates, Inc., 2022.

[23] Mitchell Wortsman, Gabriel Ilharco, Samir Yitzhak Gadre, Rebecca Roelofs, Raphael Gontijo Lopes, Ari S. Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *ICML*, volume 162, pages 23965–23998. PMLR, 2022.

[24] Han Wu, Yuxuan Yao, Shuqi Liu, Zehua Liu, Xiaojin Fu, Xiongwei Han, Xing Li, Hui-Ling Zhen, Tao Zhong, and Mingxuan Yuan. Unlocking efficient long-to-short llm reasoning with model merging, 2025.

[25] Prateek Yadav, Derek Tam, Leshem Choshen, Colin A. Raffel, and Mohit Bansal. Ties-merging: Resolving interference when merging models. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine, editors, *NeurIPS*, 2023.

[26] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2.5 technical report, 2024.

[27] Enneng Yang, Zhenyi Wang, Li Shen, Shiwei Liu, Guibing Guo, Xingwei Wang, and Dacheng Tao. Adamerging: Adaptive model merging for multi-task learning. In *The Twelfth ICLR, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

[28] Yuxuan Yao, Shuqi Liu, Zehua Liu, Qintong Li, Mingyang Liu, Xiongwei Han, Zhijiang Guo, Han Wu, and Linqi Song. Activation-guided consensus merging for large language models, 2025.

[29] Edward Yeo, Yuxuan Tong, Morry Niu, Graham Neubig, and Xiang Yue. Demystifying long chain-of-thought reasoning in llms, 2025.

[30] Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. Language models are super mario: Absorbing abilities from homologous models as a free lunch. In *Forty-first ICML, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024.

[31] Qiyuan Zhu, Dezhi Li, Lujun Li, Xiaoyu Qin, Wei Li, Hao Gu, Hua Xu, Sirui Han, and Yike Guo. Outlier matters: Efficient long-to-short reasoning via outlier-guided model merging. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2026.

[32] Qiyuan Zhu, Lujun Li, Dezhi Li, Jiacheng Liu, Pengyu Cheng, Yucheng Xu, Sirui Han, and Yike Guo. Outlier-aware model merging for efficient multitask inference. In *Proceedings of the 33rd ACM International Conference on Multimedia*, 2025.

A LHM Algorithm

Algorithm 1 presents the complete Low-Rank Hierarchical Merging pipeline, integrating the components described in Sections 3.2–3.4 into five sequential phases.

Phase 1 (Task Vector Decomposition) extracts per-layer parameter displacements δ_l by differencing the reasoning model from the base model at each layer. *Phase 2 (Hessian Trace Estimation)* drives k Rademacher probe vectors through two back-propagation passes per probe using random input sequences, assembling unbiased Hutchinson estimates $\widehat{\mathrm{tr}}(H_l)$ for each layer with no task-specific calibration data. *Phase 3 (LSOS Computation)* multiplies each trace estimate by the squared ℓ_2 norm of the corresponding task-vector component. *Phase 4 (DCCM)* applies min-max normalisation followed by the mean-anchored affine transform to produce the final coefficient vector $\mathbf{\Lambda}$, parameterised by $\bar{\lambda}$ and α. *Phase 5 (Merging)* applies $\mathbf{\Lambda}$ element-wise to δ and adds the result to θ_0 in a single arithmetic operation.

Algorithm 1 Low-Rank Hierarchical Merging (LHM)

Require: Base model θ_0, reasoning model θ_r, target merge coefficient $\bar{\lambda} \in (0, 1)$, probe rank k, bandwidth α, random input sequences $\{x_t\}$

Ensure: Merged model θ_{merged} with preserved reasoning accuracy and reduced output length

Phase 1: Task Vector Decomposition
for each layer $l \in \{1, \ldots, L\}$ **do**
  $\delta_l \leftarrow \theta_{r,l} - \theta_{0,l}$ {Per-layer parameter displacement}
end for
Phase 2: Data-Free Low-Rank Hessian Trace Estimation
for each layer $l \in \{1, \ldots, L\}$ **do**
  Sample k i.i.d. Rademacher vectors $\{v_j\}_{j=1}^{k} \subset \{-1, +1\}^{d_l}$
  for $j = 1$ **to** k **do**
    Compute $H_l v_j$ via two backpropagation passes over random inputs $\{x_t\}$ $\{O(d_l)$ cost per pass$\}$
    $q_j \leftarrow v_j^{\top}(H_l v_j)$
  end for
  $\widehat{\mathrm{tr}}(H_l) \leftarrow \dfrac{1}{k}\sum_{j=1}^{k} q_j$ {Unbiased estimate (Eq. 3)}
end for
Phase 3: LSOS Index Computation
for each layer $l \in \{1, \ldots, L\}$ **do**
  $\mathrm{LSOS}_l \leftarrow \widehat{\mathrm{tr}}(H_l) \cdot \|\delta_l\|_2^2$ {Eq. 7}
end for
Phase 4: Dual-Constraint Coefficient Mapping (DCCM)
$\mathrm{LSOS}_{\min} \leftarrow \min_l \mathrm{LSOS}_l$; $\mathrm{LSOS}_{\max} \leftarrow \max_l \mathrm{LSOS}_l$
for each layer l **do**
  $\widetilde{S}_l \leftarrow \dfrac{\mathrm{LSOS}_l - \mathrm{LSOS}_{\min}}{\mathrm{LSOS}_{\max} - \mathrm{LSOS}_{\min}}$ {Min-max normalisation (Eq. 8)}
end for
$\overline{\widetilde{S}} \leftarrow \dfrac{1}{L}\sum_{l=1}^{L} \widetilde{S}_l$
for each layer l **do**
  $\lambda_l \leftarrow \bar{\lambda} + \alpha\left(2\widetilde{S}_l - 1 - \overline{\widetilde{S}}\right)$ {Mean-anchored affine map (Eq. 9)}
end for
Phase 5: Layer-Adaptive Model Merging
$\theta_{\mathrm{merged}} \leftarrow \theta_0 + \mathbf{\Lambda} \odot \delta, \quad \mathbf{\Lambda} = [\lambda_1, \ldots, \lambda_L]$ {Eq. 1; zero inference overhead}
return θ_{merged} =0

Computational complexity. Phase 2 dominates the runtime: each of the k Rademacher probes requires two backward passes per layer, each with $O(d)$ complexity, giving total offline cost $O(Ldk)$. For a 7B model ($L \approx 32$, $k = 16$), this amounts to 1,024 backward passes, completing in under 40

seconds on a single A100 GPU. Phases 1, 3, and 4 involve only element-wise arithmetic and run in negligible time. Phase 5 introduces *zero additional inference latency*: the coefficient vector $\mathbf{\Lambda}$ is computed once offline and permanently absorbed into the merged model parameters.

Plug-and-play integration. The layer-wise coefficient vector $\mathbf{\Lambda}$ produced by DCCM can replace the scalar merge coefficient in any existing merging method without architectural changes. Substituting $\mathbf{\Lambda}$ into TIES [25], DARE [30], and AIM [17] yields the structured variants LHM-TIES, LHM-DARE, and LHM-AIM respectively, each combining the base method's parameter-conflict resolution or activation-guided weighting with LHM's data-free second-order sensitivity analysis.

B Mathematical Derivations

B.1 Hessian Intractability at Transformer Scale

For a layer weight matrix $W_l \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$, the full Hessian $H_l \in \mathbb{R}^{d^2 \times d^2}$ (where $d = d_{\text{in}} d_{\text{out}}$) requires $O(d^4)$ storage. For a typical attention projection of dimension 4096×4096, this amounts to approximately 2.8×10^{13} entries—far exceeding any practical memory budget. Estimating only the scalar trace $\text{tr}(H_l)$ via Hutchinson's estimator circumvents this bottleneck entirely.

B.2 Derivation of Hutchinson's Trace Identity

For any symmetric matrix $A \in \mathbb{R}^{d \times d}$ and random vector $v \in \mathbb{R}^d$ satisfying $\mathbb{E}[vv^\top] = I$,

$$\mathbb{E}[v^\top A v] = \mathbb{E}\big[\text{tr}(v^\top A v)\big] = \text{tr}\Big(A\,\mathbb{E}[vv^\top]\Big) = \text{tr}(A).$$

Rademacher vectors (entries i.i.d. from $\{-1, +1\}$ with equal probability) satisfy $\mathbb{E}[v_i v_j] = \delta_{ij}$ and are known to achieve lower estimator variance than Gaussian alternatives for sparse or near-diagonal matrices, a property that empirically benefits transformer Hessians.

B.3 Second-Order Taylor Approximation for LSOS

The layer-wise loss at the merged parameters $\theta_{0,l} + \lambda_l \delta_l$ admits the second-order Taylor expansion around $\theta_{0,l}$:

$$\Delta\mathcal{L}_l \approx \big(\nabla_{W_l}\mathcal{L}_l\big)^\top \delta_l + \frac{1}{2}\,\delta_l^\top H_l\,\delta_l. \tag{10}$$

Since θ_0 is a well-converged model residing near a stationary region of the loss landscape, the first-order gradient term $\nabla_{W_l}\mathcal{L}_l \approx \mathbf{0}$, reducing Equation 10 to

$$\Delta\mathcal{L}_l \approx \frac{1}{2}\,\delta_l^\top H_l\,\delta_l. \tag{11}$$

To make this tractable without materialising H_l, we invoke an isotropy approximation: when the task-vector direction $\hat{\delta}_l = \delta_l / \|\delta_l\|_2$ is not strongly aligned with any dominant eigenvector of H_l (a condition that holds approximately when reasoning post-training distributes parameter changes across many directions), the expected Rayleigh quotient satisfies $\mathbb{E}_{\hat{u}}[\hat{u}^\top H_l \hat{u}] = \text{tr}(H_l)/d_l$. Substituting $\hat{u} \propto \delta_l$ gives

$$\delta_l^\top H_l\,\delta_l \approx \frac{\|\delta_l\|_2^2}{d_l}\,\text{tr}(H_l), \tag{12}$$

and since d_l is a fixed architectural constant, $\Delta\mathcal{L}_l \propto \widehat{\text{tr}}(H_l) \cdot \|\delta_l\|_2^2 = \text{LSOS}_l$.

B.4 DCCM Mean-Anchoring Analysis

The mean of the layer-wise coefficients produced by Equation 9 satisfies

$$\frac{1}{L}\sum_{l=1}^{L} \lambda_l = \bar{\lambda} + \alpha\Big(\overline{\widetilde{S}} - 1\Big),$$

where the residual $\alpha(\overline{\widetilde{S}} - 1) \in [-\alpha, 0]$ is bounded in magnitude by α and vanishes when $\overline{\widetilde{S}} \approx 1$. For the default $\alpha = 0.15$, the deviation is at most 0.15—negligible given that the coefficient range of

interest lies in $(0, 1)$. Varying α over $\{0.05, 0.10, 0.15, 0.20, 0.25\}$ produces accuracy variations of $\pm 0.3\%$ for $\alpha \in [0.10, 0.20]$, with performance peaking at $\alpha = 0.15$ and degrading at larger values where over-suppression of low-LSOS layers becomes detrimental. The limit $\alpha \to 0$ recovers uniform scalar merging.

C Experimental Details

C.1 Implementation Details

For all experiments we set rank $k = 16$, bandwidth $\alpha = 0.15$, and select $\bar{\lambda}$ via a lightweight grid search over $\{0.3, 0.4, 0.5, 0.6\}$ on a 64-sample validation split of GSM8K. Hessian traces are estimated using 32 randomly generated token sequences of length 512. All evaluations use zero-shot prompting with greedy decoding.

Models. For the 1.5B scale, we use Qwen2.5-Math-1.5B [26] as the base model and DeepSeek-R1-Distill-Qwen-1.5B [5] as the reasoning model. For the 7B scale, we adopt Qwen2.5-Math-7B and DeepSeek-R1-Distill-Qwen-7B, following the setup of prior work [32, 24].

Benchmarks. We evaluate on six mathematical reasoning benchmarks covering elementary arithmetic to olympiad-level competition mathematics: GSM8K [4], MATH500 [13], Minerva Math [12], OlympiadBench [7], CollegeMath [21], and AIME24.

Baselines. We compare against Model Soup [23], Task Arithmetic [9] (TA), TIES [25], DARE [30], AIM [17], ACM [28], and ORCA [32]. All baselines use the same model pairs and benchmarks as LHM, with hyperparameters following the original published configurations.

Hyperparameter Sensitivity

(1) Effect of rank k: As shown in Figure 6 (left), average accuracy increases sharply from $k = 4$ to $k = 16$ then plateaus, and response length likewise stabilises after $k = 16$, confirming that rank 16 provides a good balance between estimation quality and computational cost. **(2) Effect of target coefficient $\bar{\lambda}$:** Figure 6 (right) traces the Pareto frontier for $\bar{\lambda} \in \{0.3, 0.4, 0.5, 0.6\}$. LHM dominates the Pareto curve of both Task Arithmetic and ORCA-AIM at every operating point. At $\bar{\lambda} = 0.3$, the merged model generates responses only 6.9% as long as the R1 model on average (93.1% reduction), while outperforming the TA baseline at the same length budget by 4.1 accuracy points. **(3) Effect of bandwidth α:** Varying α over $\{0.05, 0.10, 0.15, 0.20, 0.25\}$ produces accuracy variations of $\pm 0.3\%$ for $\alpha \in [0.10, 0.20]$ (see Appendix B.4). We adopt $\alpha = 0.15$ as the default.

D Compute Resource Usage

In accordance with compute transparency initiatives at major venues, we report the following resource usage.

Hardware. All Hessian trace estimation and model merging experiments were conducted on NVIDIA A100 80 GB GPUs. The one-time offline Hessian estimation for a 7B model pair requires under 40 seconds on a single A100 GPU. Main evaluations across six benchmarks were conducted on up to 8 A100 GPUs.

Compute budget. LHM introduces no training; all computation is inference-time (evaluation) and one-time offline estimation. Total GPU-hours for all reported experiments— including all LHM variants, baseline evaluations, ablation studies, and hyperparameter sweeps—are estimated at approximately 200 A100 GPU-hours.

Reproducibility. The Hessian estimation step is seeded for reproducibility. No task-specific data curation is required beyond the 64-sample GSM8K validation split used for $\bar{\lambda}$ selection. Authors have additionally completed the Compute Reporting Form (CRF) required by the conference and submitted it alongside supplementary materials per the stated deadline.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state three principal claims: (i) LHM is a training-free, second-order curvature-guided framework for layer-adaptive model merging; (ii) LHM achieves a 4.2 accuracy-point gain over ACM-TIES on the 1.5B scale; and (iii) LHM reduces average output length by up to 93.1% relative to the long-CoT baseline. All three are directly substantiated by the main results in Tables 1 and 2 and the Pareto analysis in Figure 1. The scope is explicitly bounded to mathematical reasoning benchmarks and two model scales (1.5B and 7B), and no claims extend beyond these evaluated settings.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [No]

   Justification: The paper does not include an explicit limitations section. Several important limitations remain undiscussed: (i) the isotropy approximation underlying the LSOS index (Equation 12) may not hold for layers where task-vector directions strongly align with dominant Hessian eigenvectors, potentially causing LSOS to overestimate or underestimate fusion difficulty in such layers; (ii) evaluation is restricted to mathematical reasoning benchmarks and to a single base–reasoning model family (Qwen2.5-Math / DeepSeek-R1-Distill), leaving open how well LHM generalises to other task domains or model architectures; (iii) the method assumes access to a matched base–reasoning model pair, which may not always be available in practice; and (iv) the approximation that the base model resides near a stationary region of the loss landscape ($\nabla_{W_l}\mathcal{L}_l \approx \mathbf{0}$) is a standard but not universally valid assumption. Authors are encouraged to add a dedicated Limitations section before final submission.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.

- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: The paper provides formal derivations for all key theoretical elements. The Hutchinson trace identity (Equation 2) is derived from the linearity of expectation and the cyclic trace property. The second-order Taylor approximation motivating the LSOS index (Equations 10–12) states its assumptions explicitly, including the near-stationarity of the base model and the isotropy approximation, both of which are referenced to established precedents in second-order compression literature [**?** **?**]. The DCCM mean-anchoring analysis is derived in closed form in Appendix B.4. All equations in the main text are numbered and cross-referenced, and the Appendix B collects full derivations.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper specifies the base and reasoning models (Qwen2.5-Math and DeepSeek-R1-Distill at 1.5B and 7B), all six evaluation benchmarks, all baseline methods with their source references, and all key hyperparameters: rank $k = 16$, bandwidth $\alpha = 0.15$, target coefficient $\bar{\lambda}$ selected via grid search over $\{0.3, 0.4, 0.5, 0.6\}$ on a 64-sample GSM8K validation split, 32 randomly generated token sequences of length 512 for Hessian estimation, and zero-shot greedy decoding for all evaluations. The method is training-free and the algorithm is fully specified in the main text (Sections 3.2–3.4) and Appendix A. Hardware (single A100 GPU) and runtime (under 40 seconds for 7B Hessian estimation) are also reported.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.

- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: The paper does not currently include a code release or anonymised repository link in the supplementary material. All benchmarks and base models used are publicly available, but no implementation code is submitted alongside the paper. Per NeurIPS Code and Data Submission Guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`), since the main contributions depend on experimental results, authors are strongly encouraged to submit anonymised code in a ZIP file alongside the supplementary materials before the deadline. Detailed hyperparameters and algorithmic steps are provided in the paper and appendix to partially compensate for the absence of released code.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: LHM is entirely training-free, so no optimizer or training data split is applicable to the proposed method. The paper specifies all relevant experimental details in Section 4.1: model pairs at both scales, six evaluation benchmarks with their respective difficulty levels, all baseline methods, the Hessian rank $k = 16$, bandwidth $\alpha = 0.15$, grid-search procedure

for $\bar{\lambda}$, calibration inputs (32 random token sequences of length 512), and zero-shot greedy decoding. The hardware environment (A100 GPU) and Hessian estimation runtime are reported in the main text, with the full experimental configuration detailed in Appendix C.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: The paper reports all benchmark results as point estimates without accompanying error bars, confidence intervals, or statistical significance tests. Since the evaluation benchmarks have fixed test sets and the method is deterministic at inference time (greedy decoding), variance across runs is minimal; however, variance arising from the stochastic Hessian trace estimation (Rademacher probes) is not characterised. Authors are encouraged to report at least the sensitivity of LSOS scores to random seed in the probe-vector sampling and to verify that benchmark results are stable across the evaluated hyperparameter settings.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: The paper reports that the complete Hessian trace estimation over all layers of a 7B model with $k = 16$ probes completes in under 40 seconds on a single NVIDIA A100 GPU—a one-time offline cost with zero inference-time overhead. Appendix D further specifies the hardware configuration (A100 80 GB GPUs), total estimated GPU-hours for all reported experiments (approximately 200 A100 GPU-hours across all LHM variants, baselines, ablations, and hyperparameter sweeps), and the software environment. These details are sufficient to estimate the reproduction cost for the main results.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research applies a training-free merging algorithm to publicly available language models (DeepSeek-R1-Distill and Qwen2.5-Math) and evaluates on standard public mathematical reasoning benchmarks. No human subjects, crowdsourcing, private user data, biometric data, or surveillance data are involved. The work does not create tools for discrimination, deception, harassment, or surveillance, and it does not facilitate illegal activity or circumvent human rights. The efficiency-oriented nature of LHM (reducing output length and computational cost) is consistent with the Code of Ethics guidance on environmental impact. The authors have reviewed the NeurIPS Code of Ethics and identify no deviations.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

    Justification: The paper discusses positive societal impacts—reduced inference cost, lower GPU memory requirements, and improved accessibility of capable reasoning models for resource-constrained deployment—in the introduction and conclusion. However, the paper does not include a dedicated Broader Impact section and does not address potential negative societal consequences. A notable omission is the dual-use risk: making powerful long-CoT reasoning models more efficient and accessible could inadvertently lower the barrier to misuse (e.g., automated generation of deceptive or harmful content at lower cost). Authors are encouraged to add a Broader Impact section that acknowledges this dual-use concern and any applicable mitigation strategies, in line with the NeurIPS Code of Ethics guidance on Deception & Harassment and Human Rights.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out.

- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies.

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release a new pre-trained language model, image generator, or scraped dataset. The primary new release is code implementing the training-free LHM merging procedure, which operates on existing publicly available model weights. The method itself introduces no new model capabilities beyond those of the merged model components; it only changes the balance between reasoning accuracy and output conciseness. Accordingly, no additional safeguards beyond those already in place for the underlying base and reasoning models are required.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [No]

    Justification: The paper correctly cites all models (DeepSeek-R1-Distill, Qwen2.5-Math), benchmarks (GSM8K, MATH500, Minerva Math, OlympiadBench, CollegeMath, AIME24), and baseline methods with their original publications. However, the paper does not explicitly state the specific licences under which these assets are used (e.g., DeepSeek-R1-Distill is released under the MIT licence; Qwen2.5-Math models are released under the Qwen Research Licence). Per NeurIPS guidelines, authors should confirm and explicitly mention the licence and terms of use for each asset in the paper or appendix, and verify that all uses comply with those terms. This information should be added before camera-ready submission.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not currently release code or any other new artefacts alongside the submission. The sole new asset is the LHM algorithm itself, which is fully documented in the main text (Sections 3.2–3.4) and Appendix A. Should code be released with the camera-ready version, authors should ensure that it is accompanied by a README specifying dependencies, execution commands, intended use, and any limitations, consistent with the NeurIPS Code and Data Submission Guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`).

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing experiments or research with human subjects. All experiments are conducted on existing publicly released language models and standard machine learning benchmarks. No interaction with human participants takes place at any stage of the research.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve human subjects, user studies, crowdsourcing, or collection of any participant data. The research consists entirely of algorithmic development and evaluation on existing public benchmarks using publicly released model weights. IRB approval or equivalent review is therefore not applicable.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: LLMs are the *object* of study in this paper (as models to be merged), not a methodological tool used to derive scientific results. Any use of LLM-based tools during manuscript preparation was limited to grammar and style editing, consistent with the NeurIPS policy that such use does not require declaration. The proposed LHM method, its mathematical derivations, all experimental results, and all scientific conclusions were produced by the authors without reliance on LLM-generated scientific content.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.