WEIGHT-SPACE GRAPH SIGNAL PROCESSING FOR MULTI-RESOLUTION MODEL MERGING

Anonymous authors
Paper under double-blind review

ABSTRACT

Model merging often suffers from destructive interference arising from conflicting parameter updates across tasks. Prior work has proposed various methods in Euclidean parameter space to mitigate this issue. However, it remains unclear how to identify which components of the parameters cause interference and which can be safely shared. To address this, we study model merging with graph signal processing and propose a novel model merging method, Spectral Graph Merging (SGM), based on a multi-resolution merging process. By viewing weight matrices as bipartite neuron-interaction graphs, we apply graph wavelet transformation to decompose the task vectors into low-frequency components capturing shared global structure and higher-frequency components associated with task-specific conflicts. SGM then reduces the interference in the high-frequency components while aggregating the low-frequency components. Evaluation results on MergeBench and OptMerge benchmarks, covering conflicting tasks and multi-modal language models, demonstrate that SGM outperforms state-of-the-art merging methods on both the benchmarks, while maintaining robust reasoning and safety under highly conflicting task combinations. Our results suggest that exploiting spectral structure in weight space provides a principled and effective mechanism for mitigating destructive interference in large-scale model merging.

1 INTRODUCTION

The rise of open-source foundation models and advanced post-training techniques has shifted LLM development from monolithic pre-training toward modular, task-specialized adaptation Kumar et al. (2025); Belcak et al. (2025). Methods such as supervised fine-tuning (SFT) Chen et al. (2025); Raffel et al. (2020), parameter-efficient fine-tuning (PEFT) Belanec et al. (2026); Hu et al. (2022); Liu et al. (2022), and RL-based alignment Shao et al. (2024); Rafailov et al. (2023); Ethayarajh et al. (2024); Wu et al. (2025) have led to a proliferation of task-specialized models. Deploying these models independently, however, incurs substantial storage, memory, and operational overhead. Model merging addresses this challenge by consolidating multiple task-specialized models into a single checkpoint without additional model training, to retain their respective capabilities Yang et al. (2026).

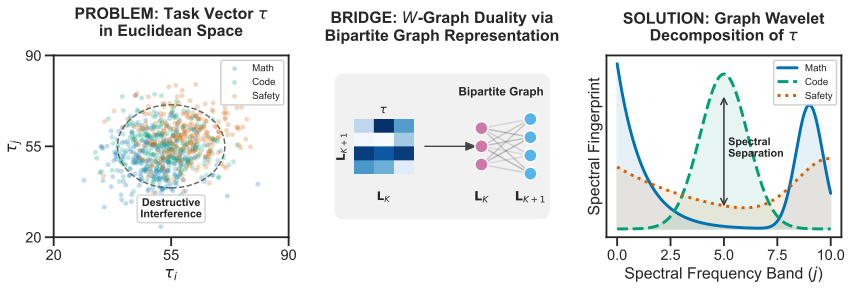

Figure 1: **Conceptual overview of SGM.** Direct parameter-space fusion merges conflicting task updates (left). Our method instead represents layer-wise task vectors τ through neural connectivity bipartite graphs (middle) and decomposes them into multiple resolutions via graph-wavelet transformation for band-dependent fusion while mitigating task interference (right).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

A central obstacle to retaining these capabilities is destructive interference, which occurs when conflicting task-specific updates degrade the merged model Yadav et al. (2023). Parameter-space methods combine updates through task-vector arithmetic or parameter averaging, as in Task Arithmetic Ilharco et al. (2022), Model Soups Wortsman et al. (2022), and Fisher Merging Matena & Raffel (2022), but do not explicitly separate compatible from conflicting updates. Subsequent methods mitigate this problem through sign-consistent update selection (TIES-Merging Yadav et al. (2023)), stochastic dropping and rescaling (DARE Davari & Belilovsky (2024); Yu et al. (2024)), statistics-based or dataless objectives (RegMean and WUDI Jin et al. (2022); Cheng et al. (2025)), localization of task-relevant parameters (Localize-and-Stitch He et al. (2024)), or dynamic selection of task-specific modules and subspaces (Twin-Merging and MASS Lu et al. (2024); Crisostomi et al. (2025)).

Recent methods analyze task updates in alternative representations. SVD-based approaches, including TSV, KnOTS, Iso-C, and CART Gargiulo et al. (2025); Stoica et al. (2024); Marczak et al. (2025); Choi et al. (2024), characterize task updates through global matrix subspaces, whereas FREE-Merging Zheng & Wang (2024) applies a Fourier basis defined over parameter coordinates. These methods demonstrate that the choice of representation affects how task interference is identified and treated. However, the interference mitigation is structure-dependent: different methods treat the same task updates differently, which may lead to agreement and disagreement across components. This structural dependence shifts attention to which properties of a neural layer remain meaningful under reparameterization. In particular, equivariant neuron permutations can change the neural connectivity of a weight matrix without changing the functional space Zhou et al. (2023); Kofinas et al. (2024). This raises the question of whether a permutation-invariant representation can provide a complementary view of task interference.

To investigate further, we formulate model merging as signal processing problem over neural connectivity graphs Kofinas et al. (2024). For each layer, the respective task vectors (τ) are defined as weighted bipartite graph in which input and output neurons form the two node sets and the values themselves define the edges (Figure 1, middle). We construct a shared normalized graph Laplacian (from the mean of task vectors) and use it as a common operator for all experts. The Graph Wavelet Transform decomposes each task vector into aligned, multi-resolution spectral bands (Figure 1, right). These bands organize signal variation by smoothness relative to the graph Laplacian: lower frequencies represent smoother variation over the graph, whereas higher frequencies capture finer-scale variation Mallat (2002); Ortega et al. (2018). Our layer-wise analysis indicates that, within this graph-wavelet basis, higher-frequency bands ($j > 0$) generally exhibit greater cross-task variance than the lowest-frequency band ($j = 0$) (Appendix A.8).

Motivated by the bipartite graph representation of τ and its multi-resolution analysis, we propose **Spectral Graph Merging (SGM)**, which performs frequency-aware merging over the graph-wavelet decomposition. SGM densely aggregates the more consistent lower-frequency components while applying conflict-aware fusion and band-dependent sparsification to higher-frequency specialized components. The fused bands are then synthesized into a single task vector and combined with the base model, producing a static merged model without additional training or inference-time overhead.

We evaluate SGM on MergeBench He et al. (2025) across four LLMs and five task domains. Among the evaluated methods, SGM achieves the best average score. On OptMerge Wei et al. (2026) with VLM, SGM obtains the best average score and outperforms the evaluated SOTA SVD/subspace-based and Fourier-based methods. Component ablations separately support the contributions of graph topology, multi-resolution decomposition, and band-dependent fusion, while the runtime analysis shows a favorable offline performance–efficiency trade-off with no inference-time overhead.

In summary, our **contributions** are:

1. We formulate model merging as graph signal processing over layer-wise bipartite graphs, providing a topology-dependent, multi-resolution basis for mitigating task interference.

2. We propose **Spectral Graph Merging (SGM)**, which applies band-dependent fusion to graph-wavelet components and uses efficient Chebyshev approximation to produce a static merged model without additional training or inference-time overhead.

3. We provide comprehensive empirical validation across established language and vision-language benchmarks from 2B to 7B models, where SGM performs the best.

2 Related Work

Current Model merging methods have evolved from a simple averaging Choi et al. (2024) to complex interference mitigation in LLMs Vaswani et al. (2017); Yang et al. (2026). Early methods like Model Soups Wortsman et al. (2022); Choi et al. (2024) average the weights of multiple models fine-tuned with different hyperparameters. Fisher Merging Matena & Raffel (2022) treats merging as approximately maximizing the joint likelihood of the posterior distribution. Task Arithmetic Ilharco et al. (2022) treats weight deltas as task vectors, τ, and merges the τ using arithmetic computation. However, arithmetic merging often causes catastrophic forgetting in divergent tasks like math and safety He et al. (2025); Song & Zheng (2026).

To mitigate destructive interference, RegMean Nguyen et al. (2025) addresses the model merging by minimizing prediction differences between the merged model and the expert model. Standard methods include sign-based consensus (TIES Yadav et al. (2023)), stochastic dropping (DARE Yu et al. (2024)), magnitude-based pruning (Model Breadcrumbs Davari & Belilovsky (2024), DELLA Deep et al. (2024)), SVD-based redundancy reduction (TSV Gargiulo et al. (2025), KNOTS Stoica et al. (2024), ISO-C Marczak et al. (2025),MASS Crisostomi et al. (2025)), subspace optimization (WUDI Cheng et al. (2025), AWD Xiong et al. (2024), DOGE Wei et al. (2025)) , permutation alignment (Git Re-basin Ainsworth et al. (2022)), evolutionary strategies Akiba et al. (2025); Lu et al. (2024).

While SVD-based methods decompose individual task matrices $\boldsymbol{\tau}_i = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$ into task-specific, coordinate-dependent bases, and SGM constructs a unified bipartite graph-derived Laplacian that captures permutation-invariant functional interactions and shared support structure across tasks. Unlike global SVD truncations or uniform singular-value thresholding, SGM leverages localized spectral graph wavelets for multi-resolution decomposition, applying wavelet-specific processing (Appendix A.10). Furthermore, SGM avoids dynamic test-time optimization like AdaRank Lee et al. (2026), preserving the original parameter count without requiring routing or calibration data.

FREE-Merging Zheng & Wang (2024), the model merging method in Fourier-transform domain, defines frequency via Fourier Transform over 2D task-vector coordinate grids, SGM utilizes Graph Wavelet Transform over a normalized Laplacian on bipartite neuron graphs to capture permutation-invariant connectivity. Because both methods operate on distinct spectral bases, their notions of smoothness and locality differ fundamentally, concentrating parameter conflicts in low-frequency Fourier bands for FREE-Merging versus high-frequency graph-wavelet bands for SGM Hammond et al. (2018). SGM preserves shared foundational knowledge by averaging low-frequency components and applying sign-consensus and thresholding to high-frequency components, opposed to different basis in FREE-Merging. Consequently, SGM resolves spatial coordinate constraints without contradiction, outperforming FREE-Merging on OptMerge Wei et al. (2026) by over 8% on average while remaining parameter-free and router-free.

Our work transcends these methods by treating weights as bipartite graphs Kofinas et al. (2024) and task vectors as graph signals, and using Spectral Graph Wavelets Hammond et al. (2018) for multi-resolution spectral fusion of LLMs. Graph Signal Processing (GSP) Ortega et al. (2018); Shuman et al. (2013) provides a mathematical framework to treat neural network weights as permutation-equivariant Zhou et al. (2023) signals on a graph Kofinas et al. (2024), allowing for the decomposition of model parameters into interpretable frequency components via the Graph Wavelet Transform Hammond et al. (2018) to enable precise, multi-resolution, spectral-based model merging.

3 Spectral Graph Merging of LLMs via Wavelet Transformation

In Section 3.1, we introduce how to treat the **task vector as a bipartite graph**. In Section 3.2, we use the graph wavelet transformation to address **task interference** by separating the shared low-frequency band (shared general information across models) and higher-frequency band (task-specific information). Based on this analysis, we introduce the **frequency-aware fusion** (merging) in Section 3.3, reducing the task interference in the higher-frequency bands and reconstructs from the spectral to the weight space domain. Algorithm 1 (Appendix A.9) provides pseudo-code of SGM.

3.1 CONVERT TASK VECTOR TO BIPARTITE GRAPH

Let θ_{base} denote a base model and $\{\theta_i\}_{i=1}^{N}$ denote N expert models obtained by independently fine-tuning θ_{base}. For a base-model weight matrix $W_{\text{base}} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$, let W_i denote the corresponding matrix in expert i. Following task-vector arithmetic Ilharco et al. (2022), we define the layer-wise task update as $\tau_i = W_i - W_{\text{base}} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$, where d_{in} and d_{out} are the input and output dimensions of the layer, respectively. As illustrated in Figure 1, the input–output structure of each layer defines a bipartite graph $\mathcal{G} = (\mathcal{V}_{\text{in}} \cup \mathcal{V}_{\text{out}}, \mathcal{E})$, where $\mathcal{V}_{\text{in}}$ and $\mathcal{V}_{\text{out}}$ contain the input and output neurons, respectively Godsil & Royle (2013); Bapat (2010). Each entry $(\tau_i)_{vu}$ assigns a weight to the edge connecting input neuron u to output neuron v. Thus, all task updates share the same bipartite support while assigning task-specific weights to its edges.

Because τ_i is a map from input to output neurons, we lift it to the symmetric bipartite graph domain as $\mathbf{X}_i = \begin{bmatrix} \mathbf{0} & \tau_i \\ \tau_i^\top & \mathbf{0} \end{bmatrix}$, where $\mathbf{X}_i \in \mathbb{R}^{(d_{\text{out}}+d_{\text{in}}) \times (d_{\text{out}}+d_{\text{in}})}$. $\mathbf{X}_i$'s off-diagonal blocks preserve the distinction between the input and output partitions and contain the original task update exactly. This lifting allows $\mathbf{X}_i$ to be processed as a graph signal using an undirected graph Laplacian $\mathcal{L}$. Under independent permutations of the input and output neurons, $\mathbf{X}_i$ transforms by the corresponding joint permutation, making the representation permutation equivariant Zhou et al. (2023).

3.2 FREQUENCY ANALYSIS OF BIPARTITE GRAPH TO IDENTIFY TASK INTERFERENCE

To mitigate task interference, we depart from traditional parameter-space methods and use spectral decomposition based on Graph Signal Processing (GSP) Shuman et al. (2013); Ortega et al. (2018). This approach allows us to localize interference within specific frequency bands, providing a granular view of how task signals interact across the graph topology (structure). The wavelet transform Mallat (2002) guarantees a multi-resolution decomposition into approximation (or low frequency) and detail (or higher frequency) bands, where approximation bands capture the general or shared structure of decomposed signals and detail bands capture the specialized structure of the decomposed signals Hammond et al. (2011).

We first define the underlying geometry that supports our spectral operators. We construct a representative bipartite graph that captures the shared structural relationships of task-specific models operating within a common loss landscape Theus et al. (2025). The graph is defined by a sparse block-symmetric adjacency matrix $\mathbf{A} \in \mathbb{R}^{(d_{out}+d_{in}) \times (d_{out}+d_{in})}$:

$$\mathbf{A} = \begin{bmatrix} \mathbf{0} & \overline{\tau} \\ \overline{\tau}^\top & \mathbf{0} \end{bmatrix} \tag{1}$$

where $\overline{\tau}$ is the normalized mean task vector across N models. $\overline{\tau}$ provides a shared spectral basis aligned with all tasks. To ensure stable signal analysis, we employ the Symmetric Normalized Laplacian, $\mathcal{L} = I - D^{-1/2}|\mathbf{A}|D^{-1/2}$, where D is the Diagonal matrix Bapat (2010). This topography serves as the fixed support for our multi-resolution analysis (see Appendix A.1 and A.2).

Multi-Resolution Spectral Decomposition: Leveraging the GWT Hammond et al. (2018), we decompose the graph signal $\mathbf{X}_i$ into J distinct frequency bands using a bank of spectral kernels $\{g_j(\lambda)\}_{j=0}^{J-1}$ (Appendix A.5 for formulation). Each band, associated with a specific range of eigenvalues λ, represents a unique level of signal variation across the topology. To maintain linear computational complexity and avoid the $O(n^3)$ cost of full eigen-decomposition, we approximate the spectral components $\Phi_{i,j}$ for each band j using M^{th}-order Chebyshev polynomials Hammond et al. (2018) (details in Appendix A.7):

$$\Phi_{i,j} \approx \sum_{k=0}^{M} c_{j,k} T_k(\tilde{\mathcal{L}}) X_i \qquad \tilde{\mathcal{L}} = \frac{2}{\lambda_{\max}} \mathcal{L} - I \tag{2}$$

where $c_{j,k}$ and T_k are the coefficients and Chebyshev polynomials respectively, and $\tilde{\mathcal{L}}$ is the linear transformation required for Chebyshev approximation. This shifts the interference analysis to spectral domain, allowing a granular understanding of interaction in multi-task signals across frequencies.

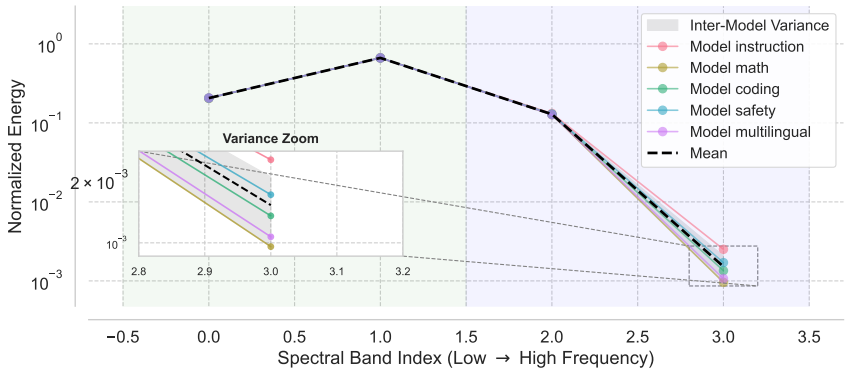

Figure 2: **Spectral Energy Concentration** for `query_proj` in Block 24 τ_i from *google/gemma-2-2b* model He et al. (2025). The Green area represents structural prior and general knowledge, while the Purple region represents the task specific learning. It represents divergence in normalized spectral energy Gutman (2001) for experts in highest frequency band and need for interference mitigation.

The number of spectral bands, J, can be fixed based on empirical analysis. To avoid this hyperparameter search, J can also be computed per layer based on the effective rank (χ) Roy & Vetterli (2007), using spectral norm $||\tau||_2$ Shuman et al. (2013), allocating finer resolution to layers with more complex task learning (Appendix A.6). To manage complexity, $||\tau||_2$ is estimated via power iteration Meidani & Ghanem (2014). We ensure the filter bank satisfies the Partition of Unity condition Behjat & Van De Ville (2018), $\sum_{j=0}^{J-1} |g_j(\lambda)|^2 = 1$, ensuring total spectral energy is conserved during decomposition and subsequent reconstruction (see appendix A.5).

Interference Localization: By analyzing in the spectral domain, we can see where conflicts arise via multi-resolution wavelet decomposition Hammond et al. (2011); Mallat (2002) (Figure 9, Appendix A.8). We compute spectral energy Gutman (2001) for each band and models to interpret the conflicts in spectral domain. We observe clear separation of normalized spectral energy among fine-tuned models in Figure 2, implying the mitigation should specifically target these higher spectral components.

3.3 Frequency-Aware Spectral Fusion and Task Vector Reconstruction

Once τ are decomposed into J spectral components $\Phi_{i,j}$, SGM performs multi-resolution merging. This enables us to treat global knowledge (low-pass bands) and sparse specialized features (higher-pass bands) Zhang & Pilanci (2024) with distinct strategies to mitigate destructive interference.

Dual-Fusion Strategy: Based on the spectral energy separation observed in Figure 2, task-specific divergence is localized predominantly in the high-frequency bands($j > 0$). Consequently, we use a dual-fusion strategy to obtain the fused task vector $\hat{\Psi}_j$ for each band: (1) Low-Pass Approximation ($j = 0$): We apply simple summation, $\hat{\Psi}_0 = \sum_{i=1}^{N} \Phi_{i,0}$, as this band captures stable, shared structural priors Hammond et al. (2011) where interference is minimal. (2) High-Pass Detail Bands ($j > 0$): To resolve the higher-frequency conflicts identified in our analysis, we apply Sign Resolution and Sparsity (ρ_j, can be static or defined per j^{th} band), as defined in Equation 3. Specifically, we resolve sign conflicts using the majority agreement $\mathcal{S}_j$ (following TIES-merging Yadav et al. (2023)) and apply a soft-thresholding Kusupati et al. (2020) operator to prune noise. Performing sign resolution in the spectral domain, rather than the spatial weight domain, allows for a more surgical removal of task interference while preserving the functional integrity of specialized bands.

$$\hat{\Psi}_j = \Psi_j \odot \text{ReLU}(|\Psi_j| - \text{quantile}(|\Psi_j|, \rho_j)) \quad \text{where} \quad \Psi_j = \sum_{i=1}^{N} (\Phi_{i,j} \odot [\text{sgn}(\Phi_{i,j}) = \mathcal{S}_j]) \quad (3)$$

Inverse Transform and Energy Rescaling: Finally, the merged spectral components are projected back to the spatial domain via an inverse GWT using Chebyshev polynomials:

$$\mathbf{X}_{fused} = \sum_{j=0}^{J-1} g_j(\mathcal{L})\Psi_j \approx \sum_{j=0}^{J-1} \left(\sum_{k=0}^{M} c_{j,k} T_k(\tilde{\mathcal{L}})\Psi_j \right) \tag{4}$$

From $\mathbf{X}_{fused}$, we extract the synthesized task vector $\tau_{synthesized}$. To prevent the magnitude collapse with model merging Yang et al. (2026); Yu et al. (2024); Skorobogat et al. (2025), we apply Energy Rescaling. The final task vector τ_{final} is rescaled to match variance of $\bar{\tau}$ in Equation 5, which is then added to θ_{base} via task arithmetic Ilharco et al. (2022):

$$\tau_{final} = \tau_{synthesized} \cdot \left(\frac{\sigma(\sum \tau_i)}{\sigma(\tau_{synthesized})} \right) \tag{5}$$

Additional Methodological Details: The appendix provides a step-by-step illustration of the neural-to-graph transformation and further discusses the induced bipartite topology (Sections A.1 and A.2). It also details the wavelet filter-bank construction, band selection, Chebyshev approximation, spectral-fusion procedure, computational tractability, and complete SGM algorithm (Sections A.4 to A.9).

4 Experiments

4.1 Experimental Setup

Benchmarks: We evaluate SGM on MergeBench He et al. (2025) and OptMergeWei et al. (2026), two standardized benchmarks designed to evaluate destructive interference when merging models specialized for diverse and conflicting tasks. For each base-model configuration, MergeBench and OptMerge provide five independently fine-tuned expert models specialized in conflicting tasks. Through MergeBench, we evaluated the model merging capability on language models, whereas with OptMerge, we evaluated merging on multi-modal language models. All five experts are derived from the same base checkpoint. In MergeBench, we evaluate four representative model configurations from two model families, covering both pretrained and instruction-tuned base checkpoints: `Gemma-2-2B` Team et al. (2024), `Gemma-2-2B-IT`, `Llama-3.2-3B` Grattafiori et al. (2024), and `Llama-3.2-3B-Instruct`. In OptMerge benchmark, we evaluate on `Qwen2-VL-7B`Wang et al. (2024). This setting merges five vision-language experts and evaluates the resulting model across ten tasks grouped into five task categories. Our experiment design allows us to evaluate SGM's performance on varying scales of models (2B, 3B and 7B) across multiple modalities.

Baselines: Following the official MergeBench evaluation protocol, we compare SGM with nine established baselines: Task Arithmetic Ilharco et al. (2022), RegMean Nguyen et al. (2025), Consensus TA He et al. (2025), Localize-and-Stitch (L&S) He et al. (2025), TIES Yadav et al. (2023), DARE Davari & Belilovsky (2024), Dataless L&S He et al. (2025), Model Soup Wortsman et al. (2022), and Fisher Merging Matena & Raffel (2022). On the OptMerge benchmark, we compare SGM with baselines already existing the benchmark like Weight Averaging, Task Arithmetic, TIES, DARE-based variants, WUDICheng et al. (2025). We also compare with additional baselines such as TSV Gargiulo et al. (2025), KnOTS Stoica et al. (2024), Iso-C Marczak et al. (2025), CART Choi et al. (2024), and the Fourier-based FR and FREE-Merging Zheng & Wang (2024).

Evaluation Metrics and Implementation Details: We follow the official MergeBench and OptMerge implementation and task-specific protocols. OptMerge evaluation protocol were completely reproducible in our infrastructure. In MergeBench, the original Safety environment could not be reproduced exactly due to environment–library mis-match, so we re-evaluated SGM and all baselines in a common local environment. The specifications are documented in Appendix A.11.3. Hardware, software, inference, hyperparameter-selection, and evaluation details are provided in Appendix A.11. We also summarize performance in MergeBench using four aggregate metrics which capture overall performance rank-wise. Formal definitions and results are provided in Appendix A.12 and A.13.

4.2 Results

Results on MergeBench: To examine how this aggregate advantage manifests across individual task domains, Table 1 presents task-wise results for `Gemma-2-2B` and `Llama-3.2-3B-Instruct`,

representing pretrained and instruction-tuned configurations from the two evaluated model families. SGM achieves the highest average performance on both representative configurations: 41.3% on `Gemma-2-2B` and 65.2% on `Llama-3.2-3B-Instruct`, exceeding the strongest baselines by 3.2 and 3.5 percentage points, respectively. Across the ten task–model combinations, SGM obtains the best or tied-best result in seven, with particularly large gains in mathematics. Results for `Gemma-2-2B-IT` and `Llama-3.2-3B` are provided in Appendix A.13.

Table 1: Task-wise performance on `Gemma-2-2B` and `Llama-3.2-3B-Instruct`. Scores marked with $*$ are reproducible from original MergeBench He et al. (2025) evaluation environment; Scores marked with † are re-evaluated locally using the official evaluation protocols.

Task	Model soup	Task arith.	Fisher Merg.	Reg-Mean	TIES	DARE	Cons. TA	Data. L&S	L&S	SGM (ours)
Gemma-2-2B Base Model										
Inst. following*	19.6	29.4	22.6	24.6	19.8	26.3	26.3	17.0	23.1	**30.9**
Math*	25.2	28.2	30.3	27.9	26.3	26.9	27.6	37.9	37.1	**42.6**
Multilingual*	47.9	47.9	41.1	48.2	48.2	**48.3**	**48.3**	47.5	47.4	**48.3**
Coding*	30.3	35.2	28.0	31.4	30.4	33.3	33.3	33.5	33.1	**37.1**
Safety†	34.1	45.9	7.1	40.1	40.1	46.9	42.6	34.8	**49.7**	47.5
Avg. Acc	31.4	37.3	25.8	34.4	33.0	36.3	35.6	34.1	38.1	**41.3**
Llama-3.2-3B-Instruct Base Model										
Inst. following*	56.0	59.7	53.6	56.9	56.6	48.6	58.8	55.8	57.2	**60.3**
Math*	53.9	55.1	49.8	61.0	56.7	56.7	51.1	59.8	59.9	**73.0**
Multilingual*	45.0	45.2	44.7	**48.4**	44.6	43.9	45.1	44.1	44.2	47.5
Coding*	52.4	49.8	51.2	51.8	52.5	53.2	48.0	51.8	52.8	**53.4**
Safety†	90.7	87.7	85.9	90.2	89.5	86.2	87.8	88.7	**94.0**	91.7
Avg. Acc	59.6	59.5	57.0	61.7	60.0	57.7	58.2	60.0	61.6	**65.2**

Table 2: Performance on OptMerge Wei et al. (2026) with `Qwen2-VL-7B`. Five vision-language experts are merged and evaluated on ten tasks grouped into five categories. The final column reports the mean across all ten tasks. The base and instruction-tuned checkpoints are included as references and are not merging methods. Bold and underlined values denote the best and second-best results.

Method	VQA	Geometry	Chart	OCR	Grounding	Mean (↑)
Weight Averaging	49.40	42.21	59.56	69.47	74.59	60.55
Task Arithmetic	51.42	33.34	79.67	70.30	71.05	60.29
TIES-Merging	50.23	40.54	67.24	69.98	74.55	61.24
TA with DARE	51.51	33.49	79.76	70.19	71.01	60.32
TIES with DARE	50.80	40.63	70.68	70.58	74.72	61.88
TSV	49.37	42.63	59.44	69.53	74.60	60.63
Iso-C	12.88	33.80	2.80	18.09	44.85	26.69
WUDI	46.82	35.30	67.84	**72.74**	69.65	58.65
FR-Merging	47.53	37.56	26.00	66.81	59.45	50.81
FREE-Merging	46.20	36.06	77.72	65.40	59.43	55.14
KnOTS	50.77	41.59	69.76	71.08	74.02	61.87
CART	49.47	41.81	58.16	69.50	74.48	60.31
OptMerge	51.39	**44.73**	74.08	70.80	75.02	63.30
SGM (ours)	**52.14**	41.12	**79.92**	71.12	**76.16**	**63.72**

Results on OptMerge: To extend the evaluation beyond language-only merging, we evaluate SGM on OptMerge using five `Qwen2-VL-7B` experts and ten vision-language tasks grouped into five categories. Table 2 compares SGM with direct parameter-space, SVD and subspace-based, and Fourier-based merging methods under the OptMerge benchmark. SGM achieves the highest overall mean score of 63.72, compared with 63.30 for OptMerge, the strongest baseline. It obtains the best merging performance in VQA, Chart, and Grounding tasks, and the second-best result in OCR task. OptMerge performs best in Geometry, while WUDI achieves the highest OCR score. SGM also outperforms the SVD and subspace-based methods like TSV, KnOTS, Iso-C, and CART, as well as the Fourier-based FREE-Merging, which are some of the notable closest work to SGM, suggesting that mutli-scale graph-spectral decomposition isolates task-specific knowledge more effectively than global singular vector truncation. It also demonstrate that SGM remains competitive across multiple vision-language tasks.

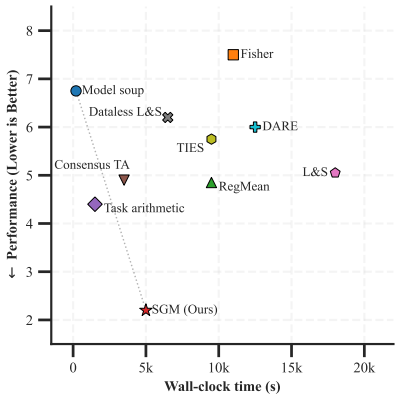

Figure 3: Performance–efficiency trade-off on MergeBench. Mean Rank is plotted against merging time; ↓ values (both axes) are better.

Table 3: Performance across model scales. Each row compares methods within the respective benchmark setting. Only Task arithmetic and TIES-merging are shown, as they are the only algorithms common to both MergeBench and OptMerge.

Setting	Task Arithmetic	TIES-Merging	SGM
`Gemma-2-`**2B** (MergeBench)	37.30	33.00	**41.30**
`Llama-3.2-`**3B**-Instruct (MergeBench)	59.50	60.00	**65.20**
`Qwen2-VL-`**7B** (OptMerge)	60.29	61.24	**63.72**

Runtime and Efficiency: Figure 3 compares Mean Rank (from Appendix A.13) with the wall-clock time required to construct the merged model. SGM achieves the best Mean Rank with a moderate offline merging cost, providing a favorable performance–efficiency trade-off. SGM avoids full eigen-decomposition by approximating graph-wavelet filters with Mth-order Chebyshev polynomials. Its per-layer complexity is $\mathcal{O}(M|\mathcal{E}|)$ for J spectral bands using vectorization, where $|\mathcal{E}|$ is the number of edges in the sparse bipartite graph. Since SGM produces a single static model, it introduces no additional parameters or inference-time overhead. Further details are in Appendix A.14.

SGM at scale: We examine whether SGM remains competitive across varying scales of the language-model settings. Table 3 summarizes the results of SGM, Task Arithmetic, and TIES-Merging in the 2B, 3B, and 7B settings. SGM achieves the highest mean performance in each setting: 41.30 on `2B`, 65.20 on `3B`, and 63.72 on `7B`. Although these settings differ in architecture, modality, and benchmark, SGM outperforms the baseline methods within each independently evaluated setting. Table 3 clearly demonstrates consistent competitive performance of SGM at different scales, including larger multimodal setting. Due to infrastructure resource constraints, it is not possible to run SGM on larger models.

4.3 Ablation Studies

We isolate the contributions of SGM's components across four axes: topography, spectral resolution, fusion and energy re-scaling. All ablations are conducted on `Llama-3.2-3B-Instruct` under the same controlled configuration He et al. (2025). Table 4 reports task-wise and average performance.

Graph Topography: Replacing the $\bar{\tau}$-derived Laplacian with identity, randomized-adjacency, or base-model-derived alternatives reduces average performance by $> 3\%$. The largest reductions occur in instruction following, where performance decreases by $> 12.5\%$. These results indicate that the $\bar{\tau}$-derived $\mathcal{L}$ provides a better support for graph-spectral analysis than the evaluated alternatives.

Multi-Resolution Decomposition: Retaining only the low-frequency band ($j = 0$) reduces mathematics and coding performance by $> 5.5\%$ and $> 5\%$, respectively. Retaining only the highest-frequency band ($j = J - 1$) produces larger reductions of $> 8.5\%$ and $> 5.4\%$. Our results indicate that the multi-resolution decomposition combines complementary information across spectral bands.

Fusion Strategy: Replacing SGM's band-dependent fusion with DARE, maximum, or mean fusion reduces average performance by $> 2\%$, respectively. DARE primarily reduces instruction-following performance, whereas maximum and mean fusion produce larger reductions in mathematics and coding. This result supports treating graph-wavelet bands separately rather than uniform operation.

Energy Rescaling: Removing energy rescaling reduces average performance from 65.2% to 63.54%. Although the change in average performance is modest, mathematics and coding decrease by > 3.5 and > 3 percentage points, respectively. This indicates that energy rescaling helps maintain the magnitude of reconstructed task updates relevant to these tasks.

Table 4: Component ablations on `Llama-3.2-3B-Instruct`. Δ Avg. denotes the change in average performance relative to full SGM. Bold and underlined are the best and second-best results.

Component	Configuration	IF	Math	Multi.	Code	Safety	Avg.	Δ Avg.
Full SGM	Full configuration	<u>60.3</u>	**73.0**	**47.5**	<u>53.4</u>	**91.7**	**65.2**	**0.00**
Topology ($\mathcal{L}$)	Identity support ($\mathcal{L} = I$)	47.50	72.86	47.16	52.55	87.60	61.53	-3.67
	Randomized support ($\mathcal{L}_{\mathrm{Rand}(A)}$)	48.24	72.18	47.16	52.52	88.78	61.78	-3.42
	Base-model support ($\mathcal{L}_{\theta_{\mathrm{base}}}$)	49.91	71.27	47.15	**53.57**	88.49	62.08	-3.12
Bands ($\Phi_{i,j}$)	Low-frequency only ($j = 0$)	**63.96**	67.17	47.38	47.87	88.47	62.97	-2.23
	Highest-frequency only ($j = J - 1$)	58.23	65.13	47.13	47.40	86.31	60.84	-4.36
Fusion ($\widehat{\Psi}_j$)	DARE	50.83	<u>72.93</u>	47.14	53.17	89.50	62.71	-2.49
	Element-wise maximum	61.37	69.67	47.37	49.35	88.27	63.21	-1.99
	Element-wise mean	63.22	66.87	47.33	48.69	88.91	63.00	-2.2
Energy Rescaling	Disabled	61.18	69.29	<u>47.45</u>	50.03	<u>89.74</u>	<u>63.54</u>	<u>-1.66</u>

Overall, these results show that SGM's components make complementary contributions. The $\bar{\tau}$-derived topology defines the support for graph-spectral filtering, the multi-resolution decomposition retains information across frequency ranges, the band-dependent fusion controls how these components are combined, and energy rescaling stabilizes the reconstructed task vector. Multilingual performance varies only slightly across ablations (~ 0.37), likely because multilingual representations are primarily learned during pretraining Anil et al. (2023); Lin et al. (2022). Reasoning-oriented tasks depend more heavily on task-specific updates and are more sensitive to ablation changes.

The appendix reports sensitivity analyses of band and sparsity configurations and the Chebyshev approximation order (Sections A.6.1 and A.7). It also provides complete task-wise scores for remaining 2B and 3B models, formal definitions of the aggregate metrics, and detailed evaluation and reproducibility settings (Sections A.11 to A.13).

5 Conclusion

Spectral Graph Merging (SGM) demonstrates that destructive interference in LLM model merging can be mitigated without any optimization by operating in a structured spectral representation of the weight space. Empirical results on MergeBench and OptMerge show that our approach enables consistent preservation of reasoning, alignment, and task specialization across highly conflicting merges. Broadly, our findings suggest that graph-based signal processing is critical during merging.

Future work includes extending spectral merging beyond shared-base models and exploring different graph operators. Our method could enable high-performance AI through resource-efficient model merging, while mitigating the environmental impact of redundant large-scale training and fostering a safer, privacy-preserving, transparent ecosystem of specialized and possibly interpretable modular AI agents. Our work suggests graph signal processing as an research direction in weight-space learning.

6 Limitations

Although SGM is optimization-free and data-free, it costs higher than arithmetic-based approaches due to bipartite $\mathcal{G}$. SGM require $O(|(d_{out} + d_{in})^2|)$ space per parameter with linear time complexity (Appendix A.4). While amortized at deployment and not affecting inference, this is an implementation and future work may explore *computational optimizations*. Consistent with prior observations He et al. (2025); Song & Zheng (2026); Yang et al. (2026), performance depends on initialization regimes, reflecting assumptions such as Neural Thickets Gan & Isola (2026) and Linear Mode Connectivity Theus et al. (2025). *Relaxing the θ_{base}* prior remains an important direction for extending SGM. SGM relies on a fixed $\mathcal{L}$ operator constructed from $\bar{\tau}$, providing stable topological support for spectral analysis. While effective, this assumes a shared geometrical structure and may underrepresent strongly task-specific synaptic importance. *Developing adaptive $\mathcal{L}$* operator offers a promising direction to consider highly sparse specializations.

REFERENCES

Samuel K Ainsworth, Jonathan Hayase, and Siddhartha Srinivasa. Git re-basin: Merging models modulo permutation symmetries. *arXiv preprint arXiv:2209.04836*, 2022.

Takuya Akiba, Makoto Shing, Yujin Tang, Qi Sun, and David Ha. Evolutionary optimization of model merging recipes. *Nature Machine Intelligence*, 7(2):195–204, 2025.

Rohan Anil, Andrew M Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, et al. Palm 2 technical report. *arXiv preprint arXiv:2305.10403*, 2023.

Ravindra B Bapat. *Graphs and matrices*. Springer, 2010.

Hamid Behjat and Dimitri Van De Ville. Spectral design of signal-adapted tight frames on graphs. In *Vertex-Frequency Analysis of Graph Signals*, pp. 177–206. Springer, 2018.

Robert Belanec, Branislav Pecher, Ivan Srba, and Maria Bielikova. Peft-bench: A parameter-efficient fine-tuning methods benchmark. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3035–3054, 2026.

Peter Belcak, Greg Heinrich, Shizhe Diao, Yonggan Fu, Xin Dong, Saurav Muralidharan, Yingyan Celine Lin, and Pavlo Molchanov. Small language models are the future of agentic ai. *arXiv preprint arXiv:2506.02153*, 2025.

Liang Chen, Xueting Han, Li Shen, Jing Bai, and Kam-Fai Wong. Beyond two-stage training: Cooperative sft and rl for llm reasoning. *arXiv preprint arXiv:2509.06948*, 2025.

Runxi Cheng, Feng Xiong, Yongxian Wei, Wanyun Zhu, and Chun Yuan. Whoever started the interference should end it: Guiding data-free model merging via task vectors. *arXiv preprint arXiv:2503.08099*, 2025.

Jiho Choi, Donggyun Kim, Chanhyuk Lee, and Seunghoon Hong. Revisiting weight averaging for model merging. *arXiv preprint arXiv:2412.12153*, 2024.

F. R. K. Chung. *Spectral Graph Theory*. American Mathematical Society, 1997.

Donato Crisostomi, Alessandro Zirilli, Antonio Andrea Gargiulo, Maria Sofia Bucarelli, Simone Scardapane, Fabrizio Silvestri, Iacopo Masi, and Emanuele Rodolà. Mass: Moerging through adaptive subspace selection. *arXiv preprint arXiv:2504.05342*, 2025.

MohammadReza Davari and Eugene Belilovsky. Model breadcrumbs: Scaling multi-task model merging with sparse masks. In *European Conference on Computer Vision*, pp. 270–287. Springer, 2024.

Pala Tej Deep, Rishabh Bhardwaj, and Soujanya Poria. Della-merging: Reducing interference in model merging through magnitude-based sampling. *arXiv preprint arXiv:2406.11617*, 2024.

David L Donoho and Iain M Johnstone. Ideal spatial adaptation by wavelet shrinkage. *Biometrika*, 81(3):425–455, 1994.

Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. Kto: Model alignment as prospect theoretic optimization. *arXiv preprint arXiv:2402.01306*, 2024.

Yulu Gan and Phillip Isola. Neural thickets: Diverse task experts are dense around pretrained weights. *arXiv preprint arXiv:2603.12228*, 2026.

Antonio Andrea Gargiulo, Donato Crisostomi, Maria Sofia Bucarelli, Simone Scardapane, Fabrizio Silvestri, and Emanuele Rodola. Task singular vectors: Reducing task interference in model merging. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 18695–18705, 2025.

Chris Godsil and Gordon F Royle. *Algebraic graph theory*. Springer Science & Business Media, 2013.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

Swati Goswami, CA Murthy, and Asit K Das. Sparsity measure of a network graph: Gini index. *Information Sciences*, 462:16–39, 2018.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Ivan Gutman. The energy of a graph: old and new results. In *Algebraic Combinatorics and Applications: Proceedings of the Euroconference, Algebraic Combinatorics and Applications (ALCOMA), held in Gößweinstein, Germany, September 12–19, 1999*, pp. 196–211. Springer, 2001.

David K Hammond, Pierre Vandergheynst, and Rémi Gribonval. Wavelets on graphs via spectral graph theory. *Applied and computational harmonic analysis*, 30(2):129–150, 2011.

David K Hammond, Pierre Vandergheynst, and Rémi Gribonval. The spectral graph wavelet transform: Fundamental theory and fast computation. In *Vertex-Frequency Analysis of Graph Signals*, pp. 141–175. Springer, 2018.

Yifei He, Yuzheng Hu, Yong Lin, Tong Zhang, and Han Zhao. Localize-and-stitch: Efficient model merging via sparse task arithmetic. *arXiv preprint arXiv:2408.13656*, 2024.

Yifei He, Siqi Zeng, Yuzheng Hu, Rui Yang, Tong Zhang, and Han Zhao. Mergebench: A benchmark for merging domain-specialized llms. *arXiv preprint arXiv:2505.10833*, 2025.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *Iclr*, 1(2):3, 2022.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. *arXiv preprint arXiv:2212.04089*, 2022.

Xisen Jin, Xiang Ren, Daniel Preotiuc-Pietro, and Pengxiang Cheng. Dataless knowledge fusion by merging weights of language models. *arXiv preprint arXiv:2212.09849*, 2022.

Miltiadis Kofinas, Boris Knyazev, Yan Zhang, Yunlu Chen, Gertjan J Burghouts, Efstratios Gavves, Cees GM Snoek, and David W Zhang. Graph neural networks for learning equivariant representations of neural networks. *arXiv preprint arXiv:2403.12143*, 2024.

Komal Kumar, Tajamul Ashraf, Omkar Thawakar, Rao Muhammad Anwer, Hisham Cholakkal, Mubarak Shah, Ming-Hsuan Yang, Phillip HS Torr, Fahad Shahbaz Khan, and Salman Khan. Llm post-training: A deep dive into reasoning large language models. *arXiv preprint arXiv:2502.21321*, 2025.

Aditya Kusupati, Vivek Ramanujan, Raghav Somani, Mitchell Wortsman, Prateek Jain, Sham Kakade, and Ali Farhadi. Soft threshold weight reparameterization for learnable sparsity. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5544–5555. PMLR, 13–18 Jul 2020.

Chanhyuk Lee, Jiho Choi, Chanryeol Lee, Donggyun Kim, and Seunghoon Hong. Adarank: Adaptive rank pruning for enhanced model merging. In *International Conference on Learning Representations*, volume 2026, pp. 6484–6510, 2026.

Xi Victoria Lin, Todor Mihaylov, Mikel Artetxe, Tianlu Wang, Shuohui Chen, Daniel Simig, Myle Ott, Naman Goyal, Shruti Bhosale, Jingfei Du, et al. Few-shot learning with multilingual generative language models. In *Proceedings of the 2022 conference on empirical methods in natural language processing*, pp. 9019–9052, 2022.

Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin A Raffel. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. *Advances in Neural Information Processing Systems*, 35:1950–1965, 2022.

Zhenyi Lu, Chenghao Fan, Wei Wei, Xiaoye Qu, Dangyang Chen, and Yu Cheng. Twin-merging: Dynamic integration of modular expertise in model merging. *Advances in Neural Information Processing Systems*, 37:78905–78935, 2024.

Stephane G Mallat. A theory for multiresolution signal decomposition: the wavelet representation. *IEEE transactions on pattern analysis and machine intelligence*, 11(7):674–693, 2002.

Daniel Marczak, Simone Magistri, Sebastian Cygert, Bartłomiej Twardowski, Andrew D Bagdanov, and Joost Van De Weijer. No task left behind: Isotropic model merging with common and task-specific subspaces. *arXiv preprint arXiv:2502.04959*, 2025.

Michael S Matena and Colin A Raffel. Merging models with fisher-weighted averaging. *Advances in Neural Information Processing Systems*, 35:17703–17716, 2022.

Hadi Meidani and Roger Ghanem. Spectral power iterations for the random eigenvalue problem. *AIAA journal*, 52(5):912–925, 2014.

The-Hai Nguyen, Dang Huu-Tien, Takeshi Suzuki, and Le-Minh Nguyen. Regmean++: Enhancing effectiveness and generalization of regression mean for model merging. *arXiv preprint arXiv:2508.03121*, 2025.

Antonio Ortega, Pascal Frossard, Jelena Kovačević, José MF Moura, and Pierre Vandergheynst. Graph signal processing: Overview, challenges, and applications. *Proceedings of the IEEE*, 106 (5):808–828, 2018.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

Olivier Roy and Martin Vetterli. The effective rank: A measure of effective dimensionality. In *2007 15th European signal processing conference*, pp. 606–610. IEEE, 2007.

Mark Rudelson and Roman Vershynin. Sampling from large matrices: An approach through geometric functional analysis. *Journal of the ACM (JACM)*, 54(4):21–es, 2007.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

David I Shuman, Sunil K Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE signal processing magazine*, 30(3):83–98, 2013.

Ronald Skorobogat, Karsten Roth, and Mariana-Iuliana Georgescu. Subspace-boosted model merging. *arXiv preprint arXiv:2506.16506*, 2025.

Mingyang Song and Mao Zheng. Model merging in the era of large language models: Methods, applications, and future directions. *arXiv preprint arXiv:2603.09938*, 2026.

George Stoica, Pratik Ramesh, Boglarka Ecsedi, Leshem Choshen, and Judy Hoffman. Model merging with svd to tie the knots. *arXiv preprint arXiv:2410.19735*, 2024.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.

Alexander Theus, Alessandro Cabodi, Sotiris Anagnostidis, Antonio Orvieto, Sidak Pal Singh, and Valentina Boeva. Generalized linear mode connectivity for transformers. *arXiv preprint arXiv:2506.22712*, 2025.

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

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.

Yongxian Wei, Anke Tang, Li Shen, Zixuan Hu, Chun Yuan, and Xiaochun Cao. Modeling multi-task model merging as adaptive projective gradient descent. *arXiv preprint arXiv:2501.01230*, 2025.

Yongxian Wei, Runxi Cheng, Weike Jin, Enneng Yang, Li Shen, Lu Hou, Sinan Du, Chun Yuan, Xiaochun Cao, and Dacheng Tao. Optmerge: Unifying multimodal llm capabilities and modalities via model merging. In *International Conference on Learning Representations*, volume 2026, pp. 61571–61595, 2026.

Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International conference on machine learning*, pp. 23965–23998. PMLR, 2022.

Yongliang Wu, Yizhou Zhou, Zhou Ziheng, Yingzhe Peng, Xinyu Ye, Xinting Hu, Wenbo Zhu, Lu Qi, Ming-Hsuan Yang, and Xu Yang. On the generalization of sft: A reinforcement learning perspective with reward rectification. *arXiv preprint arXiv:2508.05629*, 2025.

Feng Xiong, Runxi Cheng, Wang Chen, Zhanqiu Zhang, Yiwen Guo, Chun Yuan, and Ruifeng Xu. Multi-task model merging via adaptive weight disentanglement. *arXiv preprint arXiv:2411.18729*, 2024.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin A Raffel, and Mohit Bansal. Ties-merging: Resolving interference when merging models. *Advances in neural information processing systems*, 36:7093–7115, 2023.

Enneng Yang, Li Shen, Guibing Guo, Xingwei Wang, Xiaochun Cao, Jie Zhang, and Dacheng Tao. Model merging in llms, mllms, and beyond: Methods, theories, applications, and opportunities. *ACM Computing Surveys*, 58(8):1–41, 2026.

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 International Conference on Machine Learning*, 2024.

Fangzhao Zhang and Mert Pilanci. Spectral adapter: Fine-tuning in spectral space. *arXiv preprint arXiv:2405.13952*, 2024.

Shenghe Zheng and Hongzhi Wang. Free-merging: Fourier transform for model merging with lightweight experts. *arXiv preprint arXiv:2411.16815*, 2, 2024.

Allan Zhou, Kaien Yang, Kaylee Burns, Adriano Cardace, Yiding Jiang, Samuel Sokota, J Zico Kolter, and Chelsea Finn. Permutation equivariant neural functionals. *Advances in neural information processing systems*, 36:24966–24992, 2023.

AI USE STATEMENT

We used generative AI for: (1) Debugging infrastructure and CUDA issues and (2) Grammatical and spelling checks.

We have **not used** generative AI for: (1) Generate synthetic data sets, (2) Develop or polish theoretical models or conceptual frameworks, (3) Formulate mathematical claims, (4) Provide critical ingredients for proving mathematical claims, (5) Assist in the writing of proofs, (6) Propose or refine hypotheses,

(7) Design or provide feedback on research methodology or experiments, (8) Implement methods, (9) Assist with translation, (10) Clean and reformat dataset, (11) Support qualitative and thematic data analysis, and (12) interpret results.

We take complete responsibility for all the content of this work, including text, claims or artifacts produced in this paper.

Ethics Statement

The work presented in this paper is an algorithmic contribution with benchmarks and baselines comparisons using publicly available models, datasets and algorithms, without any privacy, security or ethical violations.

Reproducibility Statement

The paper primarily introduces a new algorithm, which is well described in Section 3 and Appendix A.9. The baseline empirical studies are conducted based on the code framework, models and datasets provided by original MergeBench (He et al., 2025) and OptMerge (Wei et al., 2026) works. The code for the algorithm is submitted in the Supplementary Materials. Our work utilizes publicly available benchmarks and models are available on HuggingFace by the respective authors of the MergeBench and OptMerge papers. The environment reproduction details are provided in Appendix A.11 and A.12.

A Appendix: Supplementary Material

A.1 Toy Illustration of Neural to Graph to Spectral Analysis

In Figure 4, a toy illustration of a simple MLP is depicted, where first a bipartite graph is constructed, which then, via the Laplacian, is decomposed through Spectral Analysis using Eigen Decomposition. After decomposition, the low- and high-frequency components (signals) are separated, which provides the basis for our proposed mechanistically interpretable analysis and model merging.

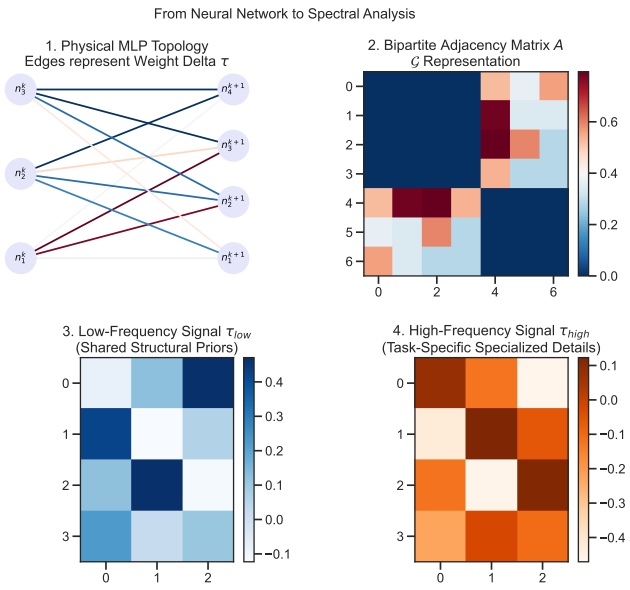

Figure 4: A toy illustration of how our method constructs a bi-partite graph from Neural Network Parameters and performs Spectral analysis for low-frequency and high-frequency analysis.

Godsil & Royle (2013); Chung (1997) define a graph as bipartite if and only if it contains no odd-length cycles. While DAGs are not inherently bipartite, neural network layers induce a bipartite

structure because the weight matrices W represent connections between two distinct, non-overlapping sets of vertices: the input neurons and output neurons. In this topology, edges only exist between these two disjoint sets, fulfilling the definition of a bipartite graph. Standard feed-forward neural networks (MLPs, CNNs without recurrence, Transformers without recurrence in the computation graph sense and W) have directed edges without any cycles, hence form a directed acyclic graph by construction as discussed by Goodfellow et al. (2016) in chapter 6. Therefore, the underlying neural connectivity W matrix is a bipartite graph, which is induced by layers and computation graph Goodfellow et al. (2016). Kofinas et al. (2024) extends this idea even further to implicit neural representations.

A.2 Topology

We have plotted the bipartite graph for 5 fine-tuned models in Figure 5.

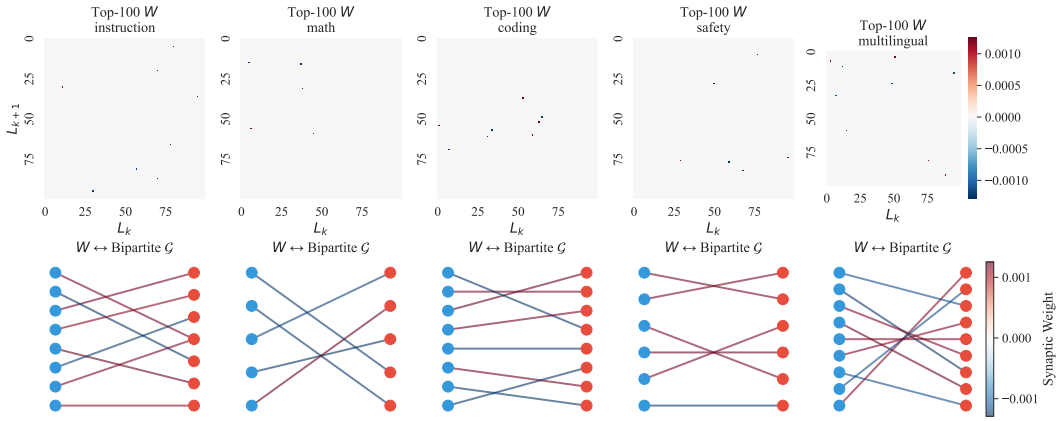

Figure 5: Bipartite Graph ($\mathcal{G}$) for top-100 absolute weight values in `down_proj` parameter in Block 3 of 5 fine-tuned models from *google/gemma-2-2b* model family in Mergebench He et al. (2025).

While SGM typically operates on models sharing identical layer widths d_{L_K}, the proposed graph-spectral framework provides a natural bridge toward cross-architecture merging through the interpolation of the bipartite adjacency matrix A. By viewing neural weights not as static parameters but as discrete samplings of a continuous graph signal, a perspective motivated by the (Kofinas et al., 2024), we can theoretically motivate the alignment of heterogeneous architectures. Within this regime, Graph Signal Processing Ortega et al. (2018); Hammond et al. (2018) allows for the projection of task vectors onto a canonical spectral support via interpolation, enabling the decomposition and construction of signals across different internal dimensions. This topological approach suggests that task-specific knowledge may reside within consistent frequency bands of the graph Laplacian $\mathcal{L}$, regardless of the underlying neuron count. While the bipartite graph $\mathcal{G}$ induced by a weight matrix W provides the structural substrate for spectral operators, the use of graph wavelet transforms facilitates a multi-resolution analysis that is invariant to node re-indexing and potentially robust to width scaling. Although the full reconstruction of merged signals into divergent target architectures via spectral synthesis remains an open question, the decomposition of weights into a permutation-equivariant graph space offers a theoretically grounded pathway for merging models where standard Euclidean arithmetic fails.

A.3 Expanded Discussion on Neural Thickets Hypothesis

The "Neural Thicket" Gan & Isola (2026) hypothesis suggests that specialized and fine-tuned large models like LLM have task-specific parameters, which are "intertwined" with general-purpose parameters and sparse to find. In SGM, we view these thickets as clusters in the graph domain. Low-frequency wavelets act as a "global smoothing" operator, effectively averaging the shared backbone of the models, while high-frequency wavelets act as "feature detectors" that isolate the sparse, high-magnitude updates unique to specific tasks. This separation allows us to apply different fusion rules (e.g., dense summation for low-freq, TIES-style sign resolution for high-freq) without destructive interference.

We observe consistent spectral sparsity and energy concentration across models and tasks. This empirical behavior aligns with the Neural Thicket hypothesis Gan & Isola (2026), which posits that specialized capabilities occupy dense, localized regions in parameter space. Crucially, SGM does not require this hypothesis to operate. Spectral decomposition and fusion are well-defined independent of any semantic interpretation. Neural thickets serve instead as a post-hoc explanatory lens for why spectral separation emerges consistently in practice.

A.4 Computational Tractability of Layer-wise Processing

While Transformer layers exhibit deep dependencies, we adopt the layer-wise independent merging paradigm Ilharco et al. (2022). This decision significantly reduces the curse of dimensionality from building the Laplacian $\mathcal{L}$. For a hidden dimension $h = 4096$, the bipartite vertex set $\mathcal{V}$ is size 8192.

While a global Laplacian $\mathcal{L}$ could theoretically be constructed for the entire network, its dimensions would correspond to the total number of neurons across all parameter, say ν_{total}. For modern LLMs, ν_{total} is on the order of 10^6 to 10^7, making the computation of $\mathcal{L}^k X$ globally inefficient due to memory constraints and the loss of layer-specific spectral locality. Furthermore, since neural networks are organized as sequences of bipartite transformations, layer-wise GWT allows us to align features within the specific manifold of each weight matrix, which is more effective for merging than a global alignment that ignores the hierarchical structure of the model. Our layer-wise spectral domain approach provides a balance between capturing local neural geometry and maintaining tractable computational efficiency.

Generally, with increasing parameter size of LLMs (like $3B \rightarrow 3B \rightarrow 70B$), the number of transformer blocks increase quickly (depth of LLM) than the width of each parameter. One such evidence is in *Llama-3* model families. The number of transformer blocks (depth) for $3B$, $8B$ and $70B$ is $28, 32, 80$ respectively. While the standard self-attention `q` parameter dimension for $3B$, $8B$ and $70B$ is $\mathbb{R}^{3072\times3072}$, $\mathbb{R}^{4096\times4096}$ and $\mathbb{R}^{8192\times8192}$ respectively.

In this work, SGM is applied only to `self-attention` and `MLP` weight matrices of Transformer blocks, and not to `embedding` layers or `LayerNorm` parameters (they are merged via task arithmetic). `Embedding` layers are excluded because token semantics and vocabulary statistics are often misaligned across fine-tuned checkpoints, making their weight deltas poorly suited for topology-aware spectral fusion and commonly omitted in prior model-merging practice. `LayerNorm` parameters primarily perform scale and shift calibration and do not typically encode task-specific knowledge relevant to interference mitigation. We therefore restrict spectral analysis to parameters with explicit neural connectivity and structured task interference, leaving topology-aware treatment of embeddings and normalization layers as future work.

A.5 Partition of Unity for Wavelet Filter Bank

To implement the Graph Wavelet Transform (GWT), we construct a filter bank $\{g_j(\lambda)\}_{j=0}^{J-1}$ that satisfies a tight frame condition through a Partition of Unity Behjat & Van De Ville (2018) normalization. Unlike standard spectral filters, our kernels are defined using a log-normal distribution in the spectral domain to provide balanced frequency coverage. For a given spectral value λ and a set of dyadic scales $s_j = 2^j$, the non-normalized kernels $\hat{g}_j(\lambda)$ are defined as:

$$\hat{g}_j(\lambda) = \begin{cases} \exp(-\tau_0\lambda) & \text{for } j = 0 \\ exp\left(-0.5\left(\ln\frac{\lambda}{1/s_j}\right)^2\right) & \text{for } j > 0 \end{cases}$$

where $\tau_0 = 1/\lambda_{\max}$.

To ensure the Partition of Unity ($\sum |g_j(\lambda)|^2 = 1$), we compute an energy map $G(\lambda) = \sqrt{\sum_j |\hat{g}_j(\lambda)|^2}$ and define the normalized kernels as $g_j(\lambda) = \hat{g}_j(\lambda)/G(\lambda)$. These normalized functional forms are then projected onto a M-th order Chebyshev basis using the Gauss-Chebyshev quadrature points $\lambda_j = \frac{\lambda_{\max}}{2}(\cos\theta_j + 1)$, where $\theta_j = \frac{\pi(j+0.5)}{M}$. The resulting coefficients $c_{i,k}$ are calculated as:

$$c_{j,k} = \frac{2}{M}\sum_{j=0}^{M-1} g_j(\lambda_j)\cos(k\theta_j)$$

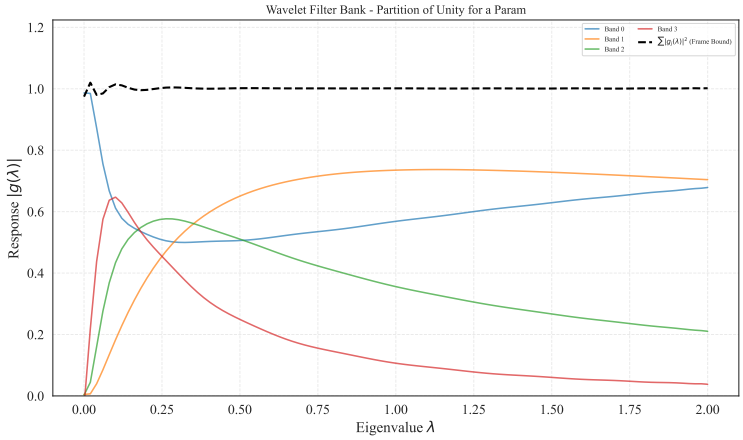

Figure 6: Wavelet Filter Bank for parameter of `down_proj` in Block 3 for $\overline{\tau}$ of 5 fine-tuned models from *google/gemma-2-2b* model family, described in Mergebench He et al. (2025).

(with a $1/M$ multiplier for $k = 0$), which are used to execute the efficient filtering described in Algorithm 1.

The figure 6 shows the wavelet filter banks for a given parameter based on the $\mathcal{L}$ calculated using $\overline{\tau}$.

A.6 Adaptive Bands based on Effective Rank

SGM additionally supports ***Adaptive Band Selection***, which dynamically determines the number of spectral bands J based on the Effective Rank (χ) Roy & Vetterli (2007); Rudelson & Vershynin (2007) of $\overline{\tau}$. It transitions model merging from empirical tuning to algorithmic determinism, which ensures that the merging process is tailored to the geometry of the specific weights being fused. While static baselines can be tuned to outperform adaptive methods on specific benchmarks (as seen in Table 4), the adaptive configuration provides a 'tuning-free' generalization that maintains high performance across varying model scales without human intervention. High-χ layers (complex knowledge) receive finer-grained decomposition, while low-χ layers are processed with fewer bands to mitigate noise. We compute J with dyadic scales Mallat (2002) as: $J = \lfloor (\log_2(\chi + 1)) \rfloor$, where $\chi = \frac{\|\overline{\tau}\|_F^2}{\|\overline{\tau}\|_2^2}$. is the squared spectral norm Shuman et al. (2013) and $||\overline{\tau}||_F^2$ is the Frobenius norm Shuman et al. (2013).

In figure 7, we show how the number of frequency bands is directly proportional to the effective rank of the $\overline{\tau}$. We also show in the figure 7 how effective rank is inversely proportional to spectral norm, which is also supported by Roy & Vetterli (2007); Rudelson & Vershynin (2007): $\chi = \frac{\|\overline{\tau}\|_F^2}{\|\overline{\tau}\|_2^2}$.

A.6.1 Adaptive Configuration and Sensitivity Analysis

Building on the adaptive band-selection rule described in Appendix A.6, SGM provides an optional tuning-free configuration for both the number of graph-wavelet bands J and the band-wise sparsity level ρ_j. While J is determined from the effective rank of each layer, ρ_j is adjusted according to the energy concentration across spectral bands. Together, these mechanisms adapt the spectral resolution and sparsification to the layer-wise structure of the task updates without model-specific hyperparameter search or validation data.

SGM in its primary form uses static default hyperparameters and requires zero data, tuning, or adaptive routing during merging or inference. The adaptive formulation in Appendix A.6.1 is an optional extension for extreme task heterogeneity, not a requirement for SGM's core performance.

Figure 8 compares the adaptive configuration with fixed band counts and alternative sparsity schedules on `Llama-3.2-3B-Instruct`. The fixed configurations $J = 2$, $J = 4$, and $J = 8$ achieve average accuracies of 64.76%, 64.17%, and 63.88%, respectively. The alternative sparsity schedules produce similarly small variations in performance.

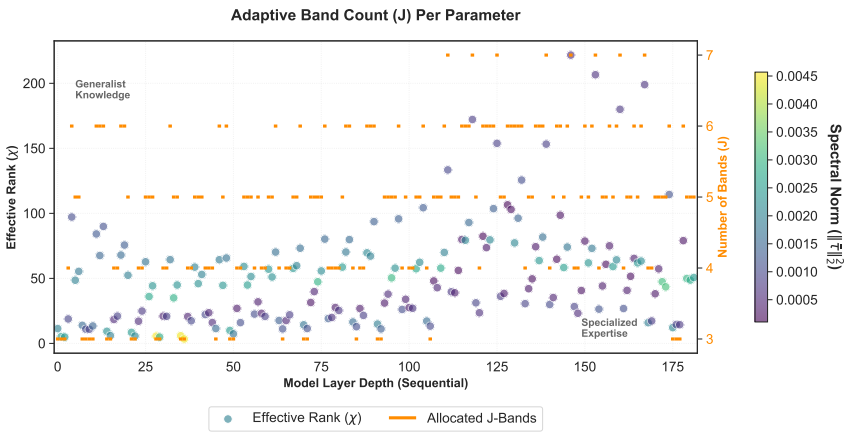

Figure 7: **Adaptive Band Count** Number of frequency bands, $J \propto \chi \propto \frac{1}{\|\bar{\tau}\|_2^2}$.

These results demonstrate that SGM remains stable across a range of band-count and sparsity configurations. The adaptive mechanism provides a practical, layer-aware default that automatically determines J and ρ_j, accommodating variations in effective rank and spectral-energy concentration across layers.

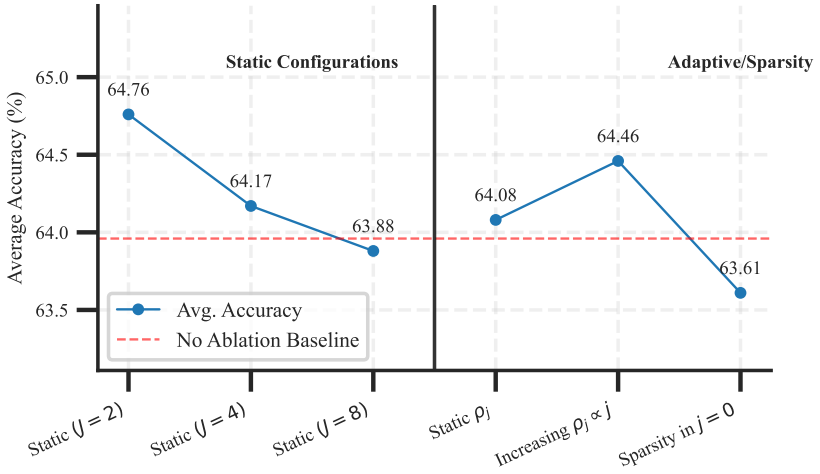

Figure 8: Sensitivity to band count and sparsity configuration. Average accuracy on `Llama-3.2-3B-Instruct` under fixed band counts, alternative sparsity schedules, and the adaptive SGM configuration.

Controlled Heterogeneous-Rank Setting. To examine adaptivity under substantial layer-wise heterogeneity, we construct a controlled three-layer network in which the task updates contain a shared low-rank component and layer-dependent high-frequency specialization. Table 5 reports the mean accuracy over eight trials. In this setting, adaptive SGM achieves an average accuracy of 29.29%, outperforming all evaluated fixed configurations and sparsity schedules. The strongest fixed alternative achieves 26.22%. This result shows that adaptive configuration is particularly effective when spectral complexity and sparsity requirements vary across layers.

Table 5: **Adaptive configuration in a controlled heterogeneous-rank setting.** Mean accuracy over eight trials.

Configuration	Mean Acc. (↑)
Fixed $J = 2$	24.06
Fixed $J = 4$	<u>26.22</u>
Fixed $J = 8$	23.36
Linear sparsity	26.17
Fixed sparsity	24.18
Sparsity only at $j = 0$	25.17
Adaptive SGM	**29.29**

Overall, adaptive selection extends SGM with an optional tuning-free mechanism for configuring spectral resolution and sparsification. The graph-wavelet decomposition and band-dependent fusion remain the core operations of SGM, while adaptivity determines their layer-specific configuration.

A.7 Chebyshev Approximation and Filter Order

To bypass the computationally prohibitive requirement of diagonalizing the graph Laplacian $\mathcal{L}$ for large-scale weight matrices, we utilize Chebyshev polynomial expansion for spectral decomposition in place of full eigen decomposition. The spectral response of each wavelet filter $g_j(\lambda)$ is approximated by a truncated series of Mth order Chebyshev polynomials $T_k(x)$. This allows the filtering operation to be expressed as a recursive matrix-vector multiplication:

$$\Phi_{i,j} \approx \sum_{k=0}^{M} c_{j,k} T_k(\tilde{\mathcal{L}}) X_i$$

To evaluate the Chebyshev polynomials $T_k(\cdot)$, we map the spectrum of the Laplacian $\mathcal{L}$, which lies in $[0, \lambda_{\max}]$, to the required Chebyshev domain $[-1, 1]$. This is achieved via the linear transformation:

$$\tilde{\mathcal{L}} = \frac{2}{\lambda_{\max}} \mathcal{L} - I$$

where I is the identity matrix and $\lambda_{\max}$ is the upper bound of the Laplacian's eigenvalues (typically $\lambda_{\max} = 2$ for normalized Laplacians). All polynomial evaluations $T_k(\tilde{\mathcal{L}})$ in Algorithm 1 are performed on this rescaled operator to ensure numerical stability and spectral accuracy.

The choice of the approximation order M represents a trade-off between spectral resolution and computational efficiency. Higher values of M allow the filter to capture sharper transitions in the spectral domain—specifically the narrow bands of the higher-order wavelets. However, since the cost of the Chebyshev recursion is $O(M \cdot |\mathcal{E}|)$, where $|\mathcal{E}|$ is the number of non-zero entries in the graph, increasing M scales linearly with time and memory.

Following established literature in Graph Signal Processing (GSP), an order of $M \approx 20$ is generally sufficient to achieve high-fidelity approximations of smooth kernels like those used in tight-frame constructions Hammond et al. (2018). In the context of model merging, our empirical analysis confirms that the performance of the merged model remains remarkably stable beyond $M = 20$. As shown in Table 6, increasing M from 20 to 32 yields negligible performance drop in accuracy across benchmark evaluation from Mergebench He et al. (2025), while significantly increasing the merging latency. Consequently, we use $M = 20$ as it provides the optimal settings, which is backed by empirical studies in other GSP work Hammond et al. (2018), ensuring the multi-resolution decomposition is spectrally accurate without becoming a bottleneck in the merging pipeline.

Table 6: Sensitivity Analysis of Chebyshev Polynomial Order M

M	**SGM Avg Acc.**
8	48.54
16	48.5
20	**48.49**
24	48.46
32	48.41

A.8 Extended Discussion on Spectral Fusion

We can observe in Figure 9 how the task interference is observed significantly in higher bands ($j > 0$) compared to lower bands.

Band-Adaptive Sparsity Our analysis of the cumulative spectral energy share along with Gini Index Goswami et al. (2018) in Figure 11 indicates that higher bands concentrate energy on a much

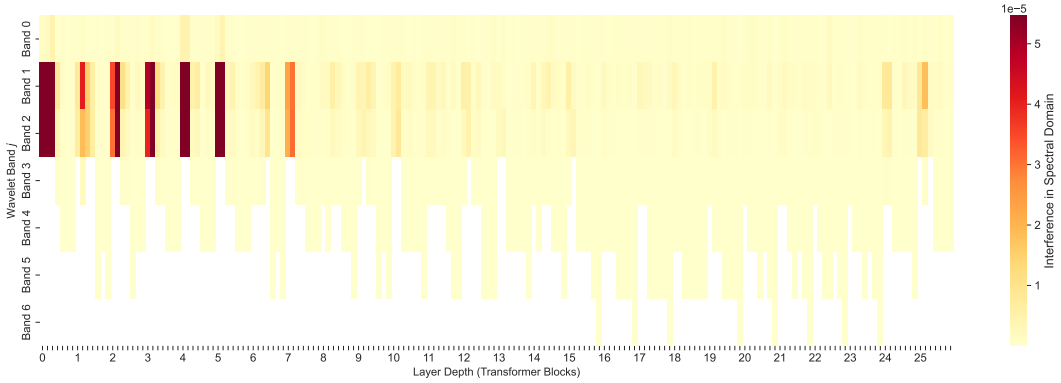

Figure 9: Spectral Energy Conflict via Variance over 5 fine-tuned models from *google/gemma-2-2b* model family, described in Mergebench He et al. (2025). It evident that lower spectral bands have more commonality across fine-tuned models, whereas higher frequency bands have conflicts (variance).

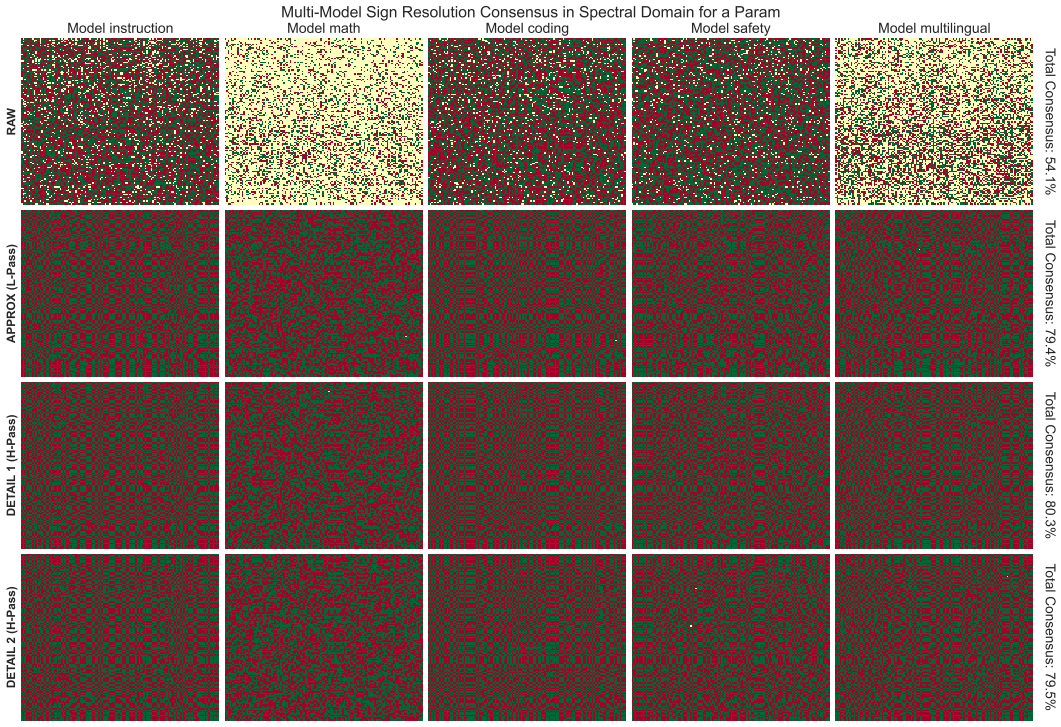

Figure 10: Sign resolution for `q_proj` in Block 13 τ_i of 5 fine-tuned models from *google/gemma-2-2b* model family, described in Mergebench He et al. (2025), showing effectiveness of sign resolution in spectral domain as compared to weight domain.

smaller subset of weights Goswami et al. (2018). Since uniform pruning can "wash out" these fragile, high-frequency features, we introduce Band-Adaptive Sparsity (ρ_j). Here, lower frequency bands are pruned more aggressively, while higher frequency bands are treated with lower ρ_j to preserve specialized knowledge as their Gini Index is generally higher, as observed in Figure 11.

Our design of adaptive sparsity is based on the well-established from the seminal work by Donoho & Johnstone (1994), Shuman et al. (2013), Yadav et al. (2023) and Yu et al. (2024), where wavelet shrinkage in higher frequency is observed, along with spectral energy decay observed in graph signals. To preserve fine-grained specialist knowledge captured in higher spectral frequencies, we employ a scale-dependent sparsity schedule.

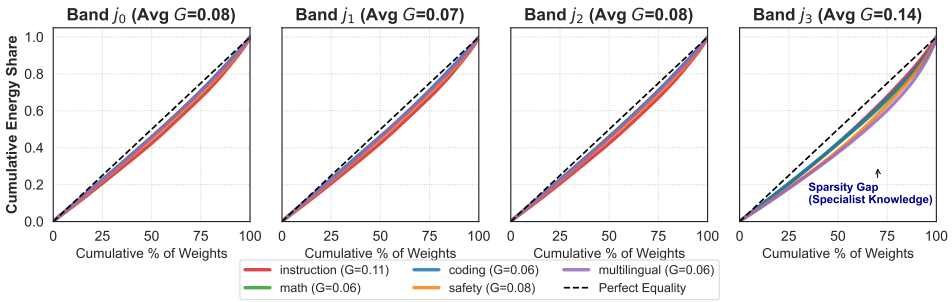

Figure 11: **Standardized Spectral Energy Lorenze Curve** for `value_proj` in Block 13 τ_i from *google/gemma-2-2b* model, described in Mergebench He et al. (2025), representing that cumulative energy (knowledge) in higher bands in concentrated only on fewer values of τ (G = Gini coefficient Goswami et al. (2018)).

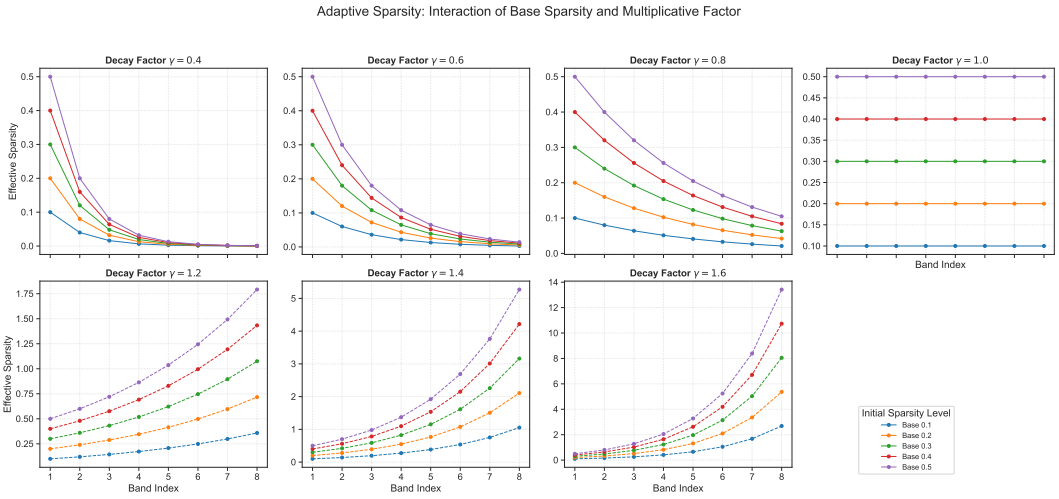

Figure 12: Effective ρ_b with respect to ζ and γ.

The sparsity threshold ρ_j for band j is defined as

$$\rho_j = \zeta \cdot \gamma^{(j-1)}$$

where ζ is the global sparsity and $\gamma \in (0, 1]$ is the Sparsity Decay Factor.

Specifically, the threshold for the jth detail band is modulated by a decay factor γ^{j-1}, where the final selected $\gamma = 0.8$, based on Figure 12. This heuristic is motivated by the principle of Wavelet Shrinkage Donoho & Johnstone (1994), which suggests that noise and signal interference are non-uniformly distributed across multi-resolution scales. By reducing pruning pressure on higher-frequency bands, SGM retains the high-fidelity 'details' of divergent tasks while enforcing structural consistency in the approximation band.

To prevent **variance collapse** Yang et al. (2026); Skorobogat et al. (2025); Yu et al. (2024), we apply a Spectral Energy Rescale:

$$\tau_{final} = \tau_{synthesized} \cdot \left(\frac{\sigma(\sum \tau_i)}{\sigma(\tau_{synthesized})} \right)$$

where $\sigma(\cdot)$ is the standard deviation, which ensures the merged delta maintains the synaptic amplitude of the original task vectors. We apply layer-wise energy rescaling, matching the $\sigma(\cdot)$ of the reconstructed $\tau_{synthesized}$ to that of the $\sum \tau_i$ (vector addition of task vectors). This preserves relative variance within parameters rather than enforcing a global magnitude constraint, avoiding calibration drift across depth.

A.9 ALGORITHM

Algorithm 1 specifies the methodologies and working of our work, as discussed in Section 3.

Algorithm 1 Spectral Graph Merging (SGM)

Require: Base model θ_{base}, Fine-tuned models $\{\theta_i\}_{i=1}^{N}$, Sparsity ζ, Scaling Coefficient α, Chebyshev Order M, Max Eigenvalue $\lambda_{max} = 2.0$
Ensure: Merged model θ_{merged}

for each parameter tensor $W \in \theta_{base}$ targeted for merging **do**
  Calculate task vectors: $\tau_i = W_i - W_{base} \quad \forall i \in \{1 \dots N\}$
  Compute mean task vector: $\bar{\tau} = \frac{1}{N} \sum_{i=1}^{N} \tau_i$
  {1. Bipartite Spectral Construction — Section 3.1}
  Normalize $\bar{\tau}$ row-wise: $\hat{\tau} = \bar{\tau}_{jk} / (\|\bar{\tau}_{j\cdot}\|_2 + \epsilon)$
  Construct Bipartite Adjacency $A = \begin{pmatrix} 0 & \hat{\tau} \\ \hat{\tau}^\top & 0 \end{pmatrix} \in \mathbb{R}^{(d_{L_{K+1}} + d_{L_K}) \times (d_{L_{K+1}} + d_{L_K})}$
  Compute Symmetric Normalized Laplacian: $\mathcal{L} = I - D^{-1/2}|A|D^{-1/2}$
  {2. Adaptive Spectral Band Determination — Section 3.2}
  Estimate Effective Rank: $\chi = \|\bar{\tau}\|_F^2 / \lambda_{max}(\bar{\tau}^\top \bar{\tau})$ {via Power Iteration}
  Set number of bands: $J = \lfloor \log_2(\chi + 1) \rfloor$
  {3. Spectral Analysis (Chebyshev Decomposition) — Section 3.2}
  Compute frame coefficients $\{c_{j,k}\}$ for J kernels $g_j(\lambda)$ ensuring Partition of Unity
  for each model $i \in \{1 \dots N\}$ **do**
    Lift τ_i to bipartite signal $X_i = \begin{pmatrix} 0 & \tau_i \\ \tau_i^\top & 0 \end{pmatrix}$
    Filter into bands: $\Phi_{i,j} = \sum_{k=0}^{M} c_{j,k} T_k(\tilde{\mathcal{L}}) X_i \quad \forall j \in \{0 \dots J-1\}$
  end for
  {4. Multi-Resolution Fusion — Section 3.3}
  Approx. Band ($j = 0$)**:** $\Psi_0 = \sum_{i=1}^{N} \Phi_{i,0}$ {Dense global fusion}
  for Detail Bands $j = 1 \dots J-1$ **do**
    $\mathcal{S}_j = \text{sgn}(\sum_{i=1}^{N} \text{sgn}(\Phi_{i,j}))$ {Sign consensus}
    $\hat{\Psi}_j = \sum_{i=1}^{N} (\Phi_{i,j} \odot [\text{sgn}(\Phi_{i,j}) == \mathcal{S}_j])$ {Masked summation}
    Compute adaptive sparsity $\rho_j = \zeta \cdot (\gamma)^{j-1}$
    Apply thresholding: $\Psi_j = \text{sgn}(\hat{\Psi}_j) \odot \text{ReLU}(|\hat{\Psi}_j| - \text{quantile}(|\hat{\Psi}_j|, \rho_j))$
  end for
  {5. Synthesis and Rescaling — Section 3.3}
  Reconstruct bipartite task vector: $X_{fused} = \sum_{j=0}^{J-1} (\sum_{k=0}^{M} c_{j,k} T_k(\tilde{\mathcal{L}}) \Psi_j)$
  Extract weight delta: τ_{final} from X_{fused}
  Apply Energy Rescale: $\tau_{final} \leftarrow \tau_{final} \cdot (\frac{\text{std}(\sum \tau_i)}{\text{std}(\tau_{final})})$
  $W_{merged} = W_{base} + \alpha \tau_{final}$
end for
return θ_{merged}

A.10 METHODOLOGICAL DISTINCTION FROM SVD FACTORIZATION

A common question is whether SGM's gains stem from the downstream fusion rules or the Graph Wavelet basis itself compared to Singular Value Decomposition (SVD). We clarify two fundamental differences:

1. **Global vs. Local Topology:** SVD operates directly on 2D weight matrices $W \in \mathbb{R}^{m \times n}$, treating parameter space as flat and Euclidean. It extracts global singular vectors spanning entire rows/columns. SGM constructs an adjacency-based Graph Laplacian $\mathcal{L}$ over parameter nodes, encoding functional connectivity across neurons.
2. **Multi-Resolution Localization:** Standard SVD lacks spatial-frequency filter banks. SGM applies band-pass wavelet filters Ψ_j, enabling localized spectral soft-thresholding across parameter neighborhoods.

Furthermore, as shown in Table 2, SGM consistently outperforms SVD-based methods (TSV, KnOTS, Iso-C) that utilize similar truncation logic on standard SVD bases, suggesting the empirical advantage of the graph-wavelet representation.

A.11 Experimental Setup and Environment

A.11.1 Software and Hardware Environment

Aligned with MergeBench He et al. (2025) official code implementation and paper, we have designed our experimental setup in such a way that evaluation of our proposed method is a sound apple-to-apple comparison with scientific rigor. In this regard, we implement our algorithm aligned with the MergeBench framework as a separate merging algorithm, keeping everything else exactly the same, such as loading models, saving merged models, evaluation, scripts, initializations and metrics. We also ran a lighter evaluation for sanity check on the baseline merging methods to validate the consistency of the evaluation results reported in the original MergeBench paper and local evaluation. While we found evaluation for Math, Multilingual, Code and Instruction Following were replicated perfectly in local sanity check compared to MergeBench paper, Safety evaluation was not reproducible due to dependencies and hardware (GPU drivers) compatibility. Hence, we upgraded only the dependencies of the Safety evaluation framework (discussed in details in subsequent subsections) and evaluated all the baseline methods and our method on the upgraded version of Safety evaluation framework, while keeping other evaluation frameworks as is.

The details of the python software packages for model merging (in an independent virtual environment) are available in table 7. Table 8 and 9 have details of the python software packages for evaluation Math, Instruction Following, Multi-lingual and Code generation tasks (in an independent virtual environment). We have mentioned the hardware specifications of the experimental environment in table 10.

Table 7: Software Environment and Key Dependencies for Model Merging, aligned with MergeBench He et al. (2025)

Package	**Version**	**Primary Role in Merging Pipeline**
Python	3.10.9	Core runtime environment.
PyTorch	2.1.2+cu121	Backend framework for tensor operations and gradient calculations.
Transformers	$< 5.0.0$	Model loading (AutoModel), architecture definitions, and weight manipulation, keeping it below version 5, which has major upgrades with breaking changes.
Datasets	(latest)	Interface for loading evaluation benchmarks.
TRL	(latest)	For training and fine-tuning (for data-based or optimization-based merging algorithms) LLMs.
DeepSpeed	0.14.5	Memory-efficient training and distributed inference for large-scale merges.
Flash-Attention	2.5.7	Optimized attention kernels to reduce memory overhead during long-context evaluation.
Protobuf	3.20.3	Compatibility layer for model serialization and tokenizer protocols.
TQDM	(latest)	Progress tracking for iterative merging and evaluation loops.

Table 8: Software Environment and Key Dependencies for Evaluation of Math, Instruction Following and Multi-lingual tasks, aligned with MergeBench He et al. (2025)

Package	**Version**	**Function**
Python	3.10.9	Base language runtime.
PyTorch	2.1.2+cu121	Tensor backend with CUDA 12.1 support for GPU acceleration.
LM-Eval Harness	(latest/source)	Core framework for standardizing zero-shot/few-shot evaluations.
Langdetect	(latest)	Language identification for multilingual task filtering.
Immutabledict	(latest)	Data structure management for frozen task configurations.
Torchvision/audio	0.16.2 / 2.1.2	Ancillary dependencies for multimodal compatibility.

Table 9: Software Environment and Key Dependencies for Evaluation of Code generation task, aligned with MergeBench He et al. (2025)

Package	Version	Function
Python	3.10.9	Base language runtime.
PyTorch	2.1.2+cu121	Tensor backend with CUDA 12.1 support.
BigCode Harness	(latest/source)	Framework for HumanEval, MBPP, and code-specific benchmarks.
NumPy	1.24.1	Strict versioning for numerical consistency in evaluation metrics.
Torchvision/audio	0.16.2 / 2.1.2	Ancillary dependencies for environment consistency.

Table 10: Hardware Specifications for model merging and evaluation, aligned with MergeBench He et al. (2025)

Component	Specification
GPU Model	NVIDIA A100 (80GB) and NVIDIA H100 (80GB)
VRAM Requirement	80GB (to support models up to 9B parameters like Gemma-2-9B and Llama-3.1-8B for the layers of five fine-tuned models at any given point)
CUDA Version	CUDA 12.1 (verified via `torch==2.1.2+cu121` dependency)
Driver Support	NVIDIA drivers compatible with CUDA 12.1 or higher
PyTorch Version	PyTorch 2.1.2
GPU Architecture	Ampere (A-series) or Hopper (H-series) for Flash-Attention support
Storage	Capacity for the specialized model checkpoints and base models (2B to 9B range) along with variations and ablations; $> 800GB$

A.11.2 Algorithm Configurations

In this section, we detail the three global, calibration-related configurations for Spectral Graph Merging (SGM): the Scaling Coefficient (α), Global Sparsity (ζ), and the Chebyshev Polynomial Order (M). Unlike traditional merging methods that may rely on data-driven optimization, these parameters are fixed architectural constants that modulate the spectral response of the SGM mechanism.

Scaling Coefficient (α) The scaling coefficient α modulates the magnitude of the merged task vector τ_{final} before its integration with the base model weights. This global multiplier prevents weight saturation and catastrophic forgetting by maintaining stability across different model scales. We explored $\alpha \in \{0.3, 0.4, 0.5\}$ and observed that lower values typically preserve instruction-following capabilities, while higher values (approaching 0.5) enhance performance in high-reasoning domains like Mathematics and Coding. We used 0.4 in our experiments, after calibrating with small held-out data.

Global Sparsity (ζ) The global sparsity ζ defines the baseline pruning percentage for the frequency-aware spectral fusion stage. It serves as the foundation for our **Band-Adaptive Sparsity** strategy, where the pruning threshold for each band is calculated as $\rho_j = \zeta \cdot \gamma^{(j-1)}$. This mechanism ensures that lower-frequency bands, which encode shared structural knowledge, are pruned more aggressively, while high-frequency bands containing specialized task signals are preserved. In our experiments with held-out calibration data, $\zeta = 0.1$ provided an optimal balance. Notably, the decay factor γ (fixed at 0.8 in our implementation) allows the model to adaptively distribute sparsity across the spectrum without manual intervention (Appendix A.8).

Chebyshev Polynomial Order (M) The order M determines the degree of the Chebyshev polynomials used to approximate spectral filters, serving as a computationally efficient proxy for full eigen-decomposition. It dictates the precision of the spectral decomposition; higher values of M

provide a more accurate approximation of the ideal filter response, which is essential for isolating task-specific signals. We explored $M \in \{8, 16, 20, 24, 32\}$ (see Appendix A.7) and found $M = 20$ to be the optimal Hammond et al. (2018) point where spectral resolution is maximized without incurring the bottlenecks of excessive matrix-vector multiplications.

Generalization and Evaluation Protocol SGM incorporates internal adaptive mechanisms, such as Adaptive Band Computation via power iteration and effective rank (χ), which automatically determines the number of spectral bands utilized per layer based on the weight geometry. This ensures that even with fixed global configurations, the merging process remains task and frequency-aware. These global configurations are one time exploration with small held-out data, which are needed for calibration of the merging process and doesn't impact the merging algorithm itself.

A.11.3 Safety Evaluation Infrastructure Differences

For safety evaluation, MergeBench used a specific fork of the official safety-eval framework. The version used in MergeBench had pinned dependencies which were not compatible with our environment, hence we had to upgrade few core dependencies to be able to run in our infrastructure. We also had issues running *vLLM* in our infrastructure, due to which we had to fallback to using native *transformers* library directly for evaluation generation and omit *vLLM* altogether for inference. Since the framework dependencies and LLM response generation framework changed, we evaluated all the baseline methods locally on our infrastructure along with our proposed method on the upgraded version of safety-eval framework, to ensure apple-to-apple comparison in safety scores with scientific rigor. Table 11 summarizes the major changes.

Table 11: Summary of the dependencies changes and upgrades for safety-eval framework in MergeBench He et al. (2025), specific to evaluating Safety tasks.

Feature/Package	Original State	Updated State	Rationale
Inference Engine	`vLLM` ($\geq 0.6.2$)	Native `Transformers`	Shift to native `transformers` generation for broader architecture support and `vLLM` compatibility issue with CUDA driver.
`transformers`	Unconstrained	`< 5.0.0`	Version pinning for stability.
`vllm`	$\geq 0.6.2$	Removed	Deprecated as part of the architecture shift.
`torch`	$\geq 2.4.0$	Implicit	Moved to library-managed dependency for cleaner environment.

A.12 Metrics used in Evaluation and Results

This subsection provides the formal mathematical definitions for the metrics utilized in the benchmarking of **Spectral Graph Merging (SGM)** and baseline methods, reported in Table 12. These metrics are designed to provide a multi-faceted view of model merging performance, accounting for absolute accuracy, cross-domain stability, and relative improvement over simple average scores.

A.12.1 Notation

Let $\mathcal{M}$ be the set of merging methods, $\mathcal{F}$ be the set of model families (e.g., Llama-3.2-3B, Gemma-2-2b), and $\mathcal{T}$ be the set of evaluation tasks (e.g., Math, Coding, Safety). For a specific method $m \in \mathcal{M}$, let $x_{m,t,f}$ denote its raw performance score on task $t \in \mathcal{T}$ within model family $f \in \mathcal{F}$.

A.12.2 Mean Rank

Notation in the table 12: Mean Rank (↓)

Notation in the mathematical formulation below: R_μ

The Mean Rank is a scale-invariant metric that measures the relative dominance of a method across diverse benchmarks. For each task-family pair (t, f), we assign a rank $r_{m,t,f} \in \{1, \ldots, |\mathcal{M}|\}$ based on the raw scores $x_{\cdot,t,f}$, where the highest performing method receives rank 1. The global Mean Rank for method m is defined as:

$$R_\mu(m) = \frac{1}{|\mathcal{F}| \cdot |\mathcal{T}|} \sum_{f \in \mathcal{F}} \sum_{t \in \mathcal{T}} r_{m,t,f} \tag{6}$$

A lower R_μ indicates that a method consistently outperforms competitors regardless of the task domain or model architecture.

A.12.3 Rank Standard Deviation

Notation in the table 12: Rank Std (↓)

Notation in the mathematical formulation below: σ_R

To quantify the reliability and stability of a merging algorithm across different configurations, we calculate the standard deviation of the ranks:

$$\sigma_R(m) = \sqrt{\frac{1}{|\mathcal{F}| \cdot |\mathcal{T}| - 1} \sum_{f,t} (r_{m,t,f} - R_\mu(m))^2} \tag{7}$$

A low σ_R suggests that the method's performance is predictable and robust across different base models.

A.12.4 Relative Win Share (Normalized Win Rate)

Notation in the table 12: Win Share % (↑)

Notation in the mathematical formulation below: W_r

To compare the comparative dominance of methods within a fixed set of baselines $\mathcal{M}$, we define the Relative Win Share W_r.

We first calculate the raw win count $C_m = \sum_{f \in F} \sum_{t \in T} \mathbb{I}(r_{m,t,f} = 1)$. The normalized metric is then defined as:

$$\hat{W}_m = \frac{C_m}{\sum_{j \in \mathcal{M}} C_j} \times 100\%$$

This metric ensures that $\sum_{m \in \mathcal{M}} \hat{W}_m = 100\%$, representing each method's "share" of the total SOTA outcomes achieved across the benchmark. This metric highlights methods that frequently achieve the best overall performance in specialized tasks post merging.

A.12.5 Average Relative Gain

Notation in the table 12: Relative Gain % (↑)

Notation in the mathematical formulation below: G_{rel}

Relative Gain measures the percentage improvement a method provides over the collective average of all tested merging strategies. This isolates the specific benefit of the merging algorithm from the inherent capabilities of the base models. Let $\bar{x}_{t,f} = \frac{1}{|\mathcal{M}|} \sum_{m \in \mathcal{M}} x_{m,t,f}$ be the mean performance for a given task-family pair. The Average Relative Gain is:

$$G_{rel}(m) = \frac{1}{|\mathcal{F}| \cdot |\mathcal{T}|} \sum_{f,t} \left(\frac{x_{m,t,f} - \bar{x}_{t,f}}{\bar{x}_{t,f}} \right) \times 100\% \tag{8}$$

Positive values for G_{rel} signify that the method effectively mitigates destructive interference compared to the baseline ensemble mean.

Table 12: **How SGM compares to existing merging methods**. We evaluated SGM against nine leading merging algorithms across various model types and tasks from MergeBench He et al. (2025). We used four key metrics: Mean Rank (↓) provides a scale-invariant measure of dominance across diverse expertise (*lower is better*). Rank Std (↓) quantifies the reliability of the merging across various base model (*lower is better*). Win Share % (↑) is the relative win share with which a method achieves SOTA performance in specific Task × Model configurations against baseline methods (*higher is better*). Rel. Gain % (↑) measures the % improvement over the ensemble mean to isolate the specific performance jump enabled by spectral decomposition (*higher is better*).

Method	Mean Rank (↓)	Rank Std (↓)	Win Share % (↑)	Relative Gain % (↑)
SGM (**ours**)	**2.20**	**1.91**	**45.83**	**17.53**
Task arithmetic	<u>4.40</u>	<u>2.30</u>	4.17	<u>3.82</u>
RegMean	4.85	2.56	<u>12.50</u>	2.99
Consensus TA	4.90	2.75	8.33	2.86
L&S	5.05	2.76	8.33	3.44
TIES	5.75	2.55	8.33	-4.10
DARE	6.00	2.94	4.17	-0.63
Dataless L&S	6.20	2.88	4.17	-2.14
Model soup	6.75	2.88	4.17	-10.18
Fisher Merging	7.50	2.50	0.00	-13.59

A.13 Raw Accuracy of Evaluation on MergeBench Baseline

As shown in Table 12, SGM attains the lowest Mean Rank, highest Win Share, and largest Relative Gain among all baselines. Raw accuracy scores of our evaluations on the instruction fine-tuned 2B and pre-trained 3B base model based on MergeBench He et al. (2025) are reported below in table 13, 14.

Table 13: Accuracy performance of merged models on five task domains from *gemma-2-2b-it* base model. We have compared SGM against all the baseline methods and tasks described in MergeBench He et al. (2025). For non-SGM algorithms, Safety results (†) are reproduced locally using the official implementation due to safety evaluation framework compatibility issue, while other tasks (∗) are reported directly from He et al. (2025) as our sanity tests yielded identical results.

Task	Model soup	Task arith.	Fisher Merg.	Reg-Mean	TIES	DARE	Cons. TA	Data. L&S	L&S	SGM (ours)
Inst. following*	51.9	51.9	53.0	<u>54.2</u>	49.2	46.4	**54.9**	48.8	49.2	51.4
Math*	38.7	38.7	39.7	40.5	38.5	36.7	38.4	<u>47.9</u>	<u>47.9</u>	**53.8**
Multilingual*	<u>49.2</u>	<u>49.2</u>	48.7	48.7	**49.3**	49.0	<u>49.2</u>	48.8	48.7	49.0
Coding*	**40.2**	**40.2**	<u>40.1</u>	39.3	39.6	38.3	39.7	**40.2**	38.2	39.5
Safety†	79.9	78.8	72.1	62.6	**81.0**	79.6	80.3	77.1	79.7	<u>80.4</u>
Avg. Acc	52.0	51.8	50.7	49.1	51.5	50.0	52.5	52.6	<u>52.7</u>	**54.8**

Figure 13 shows task-wise ranks aggregated across all the evaluated base models.

Figure 14 represents the task wise performance of selected merging methods from MergeBench He et al. (2025) against our proposed method. The figure visually confirms the capability to preserve task wise performance from all the 5 fine tuned models post model merge.

Figure 15 helps us to understand SGM's ability to be a reasoning effective model merging method, while preserving global and overall capabilities from all the fine tuned models.

Table 14: Accuracy performance of merged models on five task domains from *Llama-3.2-3B* base model. We have compared SGM against all the baseline methods and tasks described in MergeBench He et al. (2025). For non-SGM algorithms, Safety results (†) are reproduced locally using the official implementation due to safety evaluation framework compatibility issue, while other tasks (∗) are reported directly from He et al. (2025) as our sanity tests yielded identical results.

Task	Model soup	Task arith.	Fisher Merg.	Reg-Mean	TIES	DARE	Cons. TA	Data. L&S	L&S	SGM (ours)
Inst. following*	7.2	25.3	12.0	14.2	9.6	18.7	30.5	10.4	23.7	**34.0**
Math*	16.2	27.7	26.6	33.8	26.6	27.1	27.6	41.1	38.9	**51.6**
Multilingual*	46.8	47.0	47.6	**48.4**	47.6	47.5	46.9	46.7	47.0	48.1
Coding*	37.0	41.1	37.2	39.3	37.6	40.2	40.7	40.0	40.7	**43.7**
Safety†	23.5	36.1	10.8	**57.7**	40.3	37.3	32.6	23.8	16.0	34.9
Avg. Acc	26.1	35.4	26.8	38.7	32.3	34.2	35.7	32.4	33.3	**42.5**

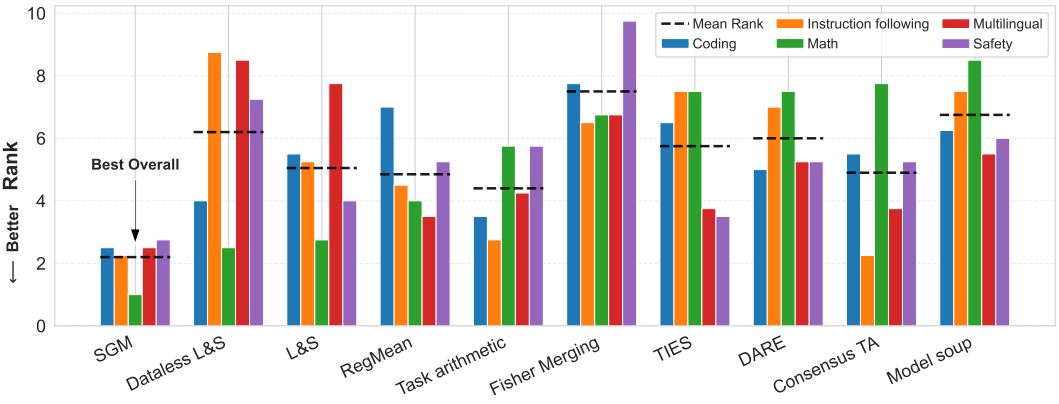

Figure 13: **Performance Ranks** of merging methods grouped by task, aggregated over all the base models. Lower the rank, better is the performance of the merging method. We can observe that SGM preserves the overall performance across tasks and base models from MergeBench He et al. (2025) post-merging.

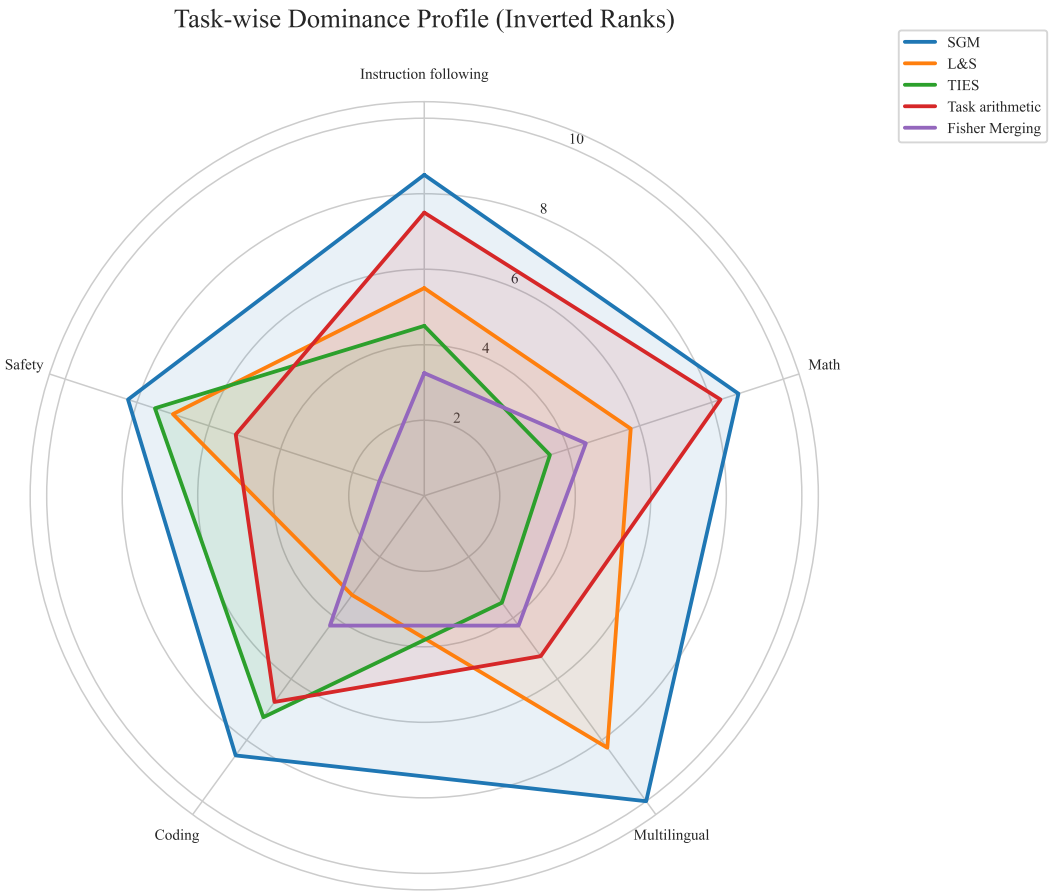

Figure 14: Task-wise dominance of merging methods. The ranks are inverted here (higher rank is better) to visually depict how merging methods are able to preserve the individual task performance post merging. SGM is compared with TIES and Task Arithmetic (foundational baseline), Localize-and-Stitch (an advanced method) and Fisher Merging (advanced and computationally expensive method with training and optimization in weight space merging). We can see how SGM is able to preserve capabilities across different tasks.

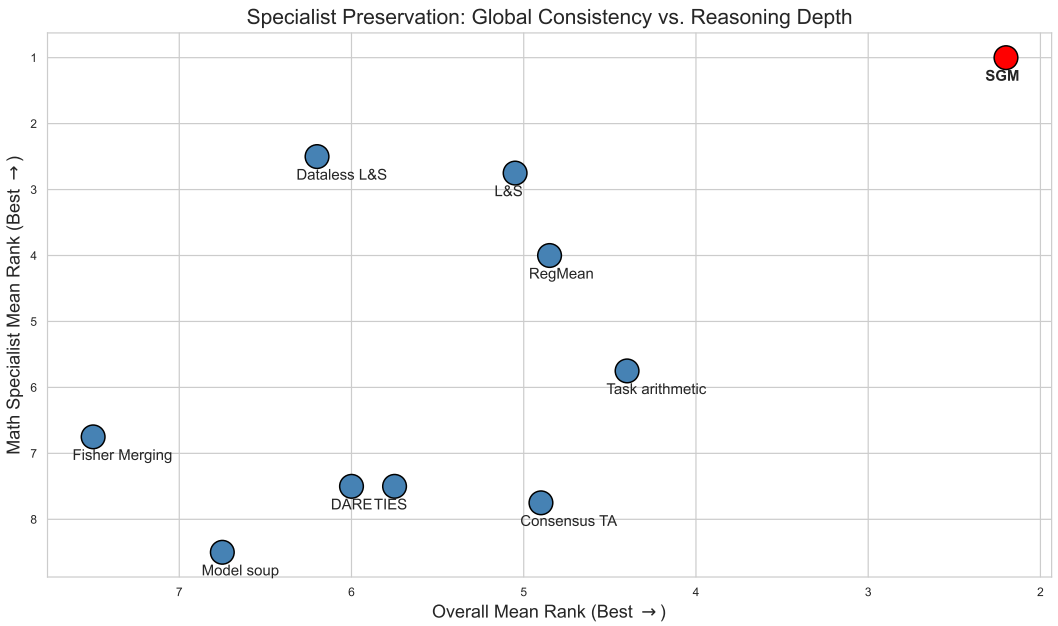

Figure 15: Specialization of tasks and how they are preserved post merging by SGM and other baseline algorithms from MergeBench He et al. (2025). We can see how SGM is consistent in preserving overall performance across different tasks in x-axis, yet performs quite well with reasoning heavy task like Math in y-axis.

A.14 Computational Performance Comparison

Figure 3 plots performance of methods against wall-clock merging time (without validation complexity). Despite employing spectral decomposition, SGM remains computationally efficient and favorable against all the baseline methods, due to its layer-wise design, sparse bipartite graph construction, and Chebyshev polynomial approximation ($O(M \cdot |\mathcal{E}|)$, where $|\mathcal{E}|$ is the number of non-sparse edges in $\mathcal{G}$), comparable to a single forward pass and amortized over merging (not inference), given **SGM is a one-time offline operation**.

SGM achieves top performance without data, training, or optimization, offering a favorable accuracy–efficiency trade-off compared to data-based methods (e.g., RegMean, Fisher).

A.15 Ablation Studies

We ablate SGM along three axes: graph topography, spectral partitioning, and fusion strategy—to isolate the source of performance gains. Table 4 reports the complete results of these studies, while Figure 16 visually illustrates the effect of key ablations, highlighting that SGM (our propose method) maintains the most balanced performance profile across all divergent task domains compared to alternative configurations.

Replacing our **Bipartite Graph Topology**, A, with an Identity (I) or a randomly permuted $\bar{\tau}$ causes a performance drop ($> 2.5\%$), as seen in Table 4. This confirms that weight-space adjacency is essential for meaningful spectral decomposition and interference localization (**RQ2**). $\mathcal{L}$ from the base model (θ_{base}) also underperformed, suggesting fine-tuned knowledge resides in task vectors.

Spectral decomposition and fusion isolates global structural priors (low-pass) from task-specific details (high-pass). Using only the $j = 0$ low-pass band maintained instruction-following but caused a $> 6\%$ drop in Math, suggesting that while general behavior occupies low-frequency structures, reasoning requires high-frequency features. While adaptive banding is theoretically optimal, a static $J = 2$ (Base and Specialist split) remains a robust zero-shot default for most SFT Wu et al. (2025); Chen et al. (2025) and PEFT Belanec et al. (2026) fine-tunings. Conversely, combining all

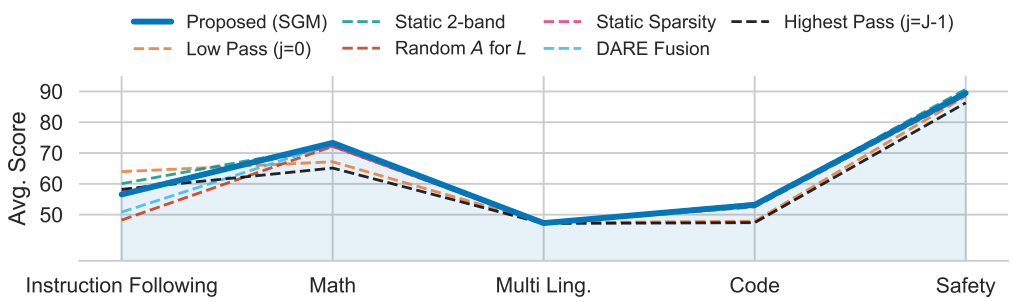

Figure 16: Impact of Architectural Choices of SGM on Multi-Domain Performance. Performance trade-offs across five benchmarks He et al. (2025) when ablating core SGM components. The removal of the Bipartite Graph Topography (Random A for L) causes a visible collapse in instruction following. Omitting Multi-Resolution Spectral Bands (e.g., Low Pass or $J = 2$) reveals a distinct performance optimality, we believe this result highlights a specific spectral characteristic of the used base model (best performing avg. accuracy), since adaptive bands makes SGM efficient and informed through χ. In a production or multi-model environment, the adaptive mechanism serves as a safeguard against the sub-optimal spectral reconstruction that would occur if a static $J = 2$ were applied to a model family with higher-rank weight fine-tunings. The Highest Pass ($j = J - 1$) line consistently tracks as the performance floor for most tasks. SGM achieves the most robust Pareto-frontier.

detail bands $(j_1, \ldots, J-1)$ outperformed standard baselines in multilingual tasks, justifying our multi-resolution approach to mitigating interference.

Although $J = 2$ achieved the highest average score (64.76%), suggesting a *Saliency Gap* where critical info concentrates in coarse structures, **spectral partitioning** remains methodologically grounded. Crucially, **adaptive sparsity** (decreasing sparsity with j) performed better than global static sparsity in Math. This validates that high-frequency bands contain isolated *specialist* signals easily destroyed by aggressive pruning through mechanistic view, requiring higher preservation to maintain reasoning depth (see Figure 11).

Band-wise sign-consensus outperformed both mean (63.00 avg.) and max (63.21 avg.) fusion, proving that simple averaging cannot resolve spectral sign conflicts. DARE-style random dropping performed poorly, likely by compromising the structural integrity of wavelet reconstruction. Finally, omitting **Energy Rescaling** caused notable degradation in Math ($> 4\%$ drop), confirming its necessity for preserving signal strength during the spectral-to-spatial transition.

Why Secondary Ablations Can Outperform the Fully Adaptive Configuration. An initially counterintuitive observation in Table 4 is that certain secondary ablations—such as fixing the spectral band count J or simplifying the band-wise sparsity schedule ρ_j slightly outperform the fully adaptive SGM configuration on average. This behavior is theoretically well-motivated and consistent with both the design of SGM and classical results in multi-resolution signal estimation.

At a high level, the fully adaptive setting relies on estimating layer-wise effective rank χ and translating it into adaptive choices of spectral resolution J and sparsity ρ_j. While this adaptivity is principled and critical for robustness across model families and scales, it also introduces an additional estimation step. In finite-sample regimes and for high-performing base models such as `Llama-3.2-3B-Instruct`, small errors or noise in estimating χ can lead to mild over-segmentation of the spectral domain, effectively dispersing signal energy across more bands than necessary. Fixing J or enforcing a simpler sparsity schedule acts as a form of implicit regularization on the spectral reconstruction, reducing sensitivity to this estimation noise.

This effect closely mirrors classical results from wavelet shrinkage theory. Donoho & Johnstone (1994) show that near-minimax performance is often achieved not by aggressively adaptive multiscale schemes, but by coarse and stable decompositions combined with conservative thresholding. Their analysis demonstrates that least-favorable signals in *Besov* and *Triebel* spaces concentrate energy densely in low-resolution coefficients while exhibiting sparsity at fine resolutions. As a result, estimators that preserve a clear low–high frequency separation can outperform more finely adaptive methods when the underlying signal structure does not warrant additional resolution.

SGM exhibits the same structural properties. As discussed in Section 3.3 and Appendix A.3, task-relevant knowledge in merged LLMs tends to localize in a small number of dominant spectral regimes: shared structural priors in low-frequency bands and sparse specialist signals in higher-frequency bands. Secondary ablations that fix J (e.g., $J = 2$ or $J = 4$) effectively enforce this coarse separation, ensuring that low-frequency components are robustly preserved while high-frequency components remain interpretable and protected from excessive pruning.

Importantly, this observation does not contradict the role of adaptivity in SGM. Rather, it highlights a known bias–variance trade-off in adaptive multi-resolution methods: while adaptivity is essential for generalization across heterogeneous architectures and training regimes, simpler static configurations can be locally optimal for specific, already well-aligned base models. This phenomenon is explicitly acknowledged in wavelet theory, where fixed or weakly adaptive thresholding rules are shown to achieve near-minimax risk while offering greater robustness to noise and model mis-specification Donoho & Johnstone (1994).

In practice, this explains why secondary ablations occasionally outperform the default configuration in average score, while the fully adaptive SGM remains the most reliable and tuning-free choice across models, tasks, and scales. The observed gains should thus be interpreted as evidence that SGM's spectral formulation captures the correct inductive structure of weight-space interference, rather than as a limitation of the adaptive design.

A.16 Qualitative Reasoning Example.

To complement the aggregate mathematics results, Figure 17 presents a representative response from SGM and Consensus TA. In this example, Consensus TA incorrectly factorizes 196 as $2^3 \cdot 7^2$, whereas SGM identifies the correct factorization $2^2 \cdot 7^2$ and consequently returns the correct number of divisors. This example illustrates the type of reasoning error observed in a competing merged model but is not used as standalone evidence of SGM's overall performance or spectral mechanism.

How many positive whole-number divisors does 196 have?

Consensus TA (incorrect): To find the number of positive whole-number divisors of 196, we first find its prime factorization. The prime factorization of 196 is $2^3 \cdot 7^2$. Therefore, the number of divisors is $(3 + 1)(2 + 1) = 12$.

SGM (correct): The prime factorization of 196 is $2^2 \cdot 7^2$. Therefore, the number of positive divisors is $(2+1)(2+1) = 9$. Hence, 196 has nine positive whole-number divisors.

Consensus TA incorrectly uses $2^3 \cdot 7^2 = 392$, whereas SGM uses the correct factorization $196 = 2^2 \cdot 7^2$ and returns nine divisors.

Figure 17: **Representative mathematics reasoning example.** Consensus TA makes an incorrect prime-factorization step, whereas SGM produces the correct factorization and answer.

A.17 Future Work

Training Stage and Spectral Robustness Our results reveal substantial differences in task sensitivity to spectral modifications: mathematics and coding vary considerably across ablations, whereas multilingual performance remains comparatively stable. A promising direction is to investigate whether this behavior depends on the training stage at which a capability is acquired. As modern foundation models increasingly develop reasoning, multilingual, and other specialized capabilities during pretraining or intermediate training, these capabilities may be distributed differently across graph-spectral components than capabilities introduced through task-specific post-training. Evaluating models with documented training stages could clarify the relationships among capability acquisition, spectral distribution, and robustness during model merging.