ADAPTIVE DCT-DOMAIN SPECTRAL MERGING: RETHINKING FREQUENCY-DOMAIN MODEL MERGING

Anonymous authors
Paper under double-blind review

ABSTRACT

Model merging combines task-specific fine-tunes of a shared base into one model without retraining or training data. Frequency-domain methods such as FR-MERGING reduce cross-task interference by applying a fixed-cutoff 2D FFT high-pass filter to each task vector before combining them. We revisit this recipe along two axes: replacing the FFT with the discrete cosine transform (DCT), which better matches the real-valued, non-periodic structure of weight matrices, and replacing the global cutoff with a per-layer adaptive cutoff driven by a cheap, data-free cross-task sign-interference score. On MergeBench, however, we find that two combination-rule pitfalls dominate the transform choice: an uncalibrated edit-scale that produces larger updates than the baselines at matched layers, and energy-proportional weighting under which each task's contribution scales with the square of its task-vector norm, starving smaller-update domains (instruction-following receives under 6% of the combination weight versus an equal 20% share). Correcting both raises aggregate normalized performance from 0.806 to 0.966 on Llama-3.1-8B-Instruct, matching DARE (0.965) on the four domains we aggregate. A replication across two model families at four scales (Llama-3.1-8B, Llama-3.2-3B, Gemma-2-2B/9B; 36 spectral merges, 144 domain evaluations) confirms the combination rules as the dominant effect: edit-scale calibration lifts pooled four-domain scores by $+0.11$ on Llama-3.1-8B and $+0.07$ on Gemma-2-9B and is neutral to slightly negative on the two smaller bases, versus at most 0.015 for any DCT-vs-FFT contrast; the DCT is ahead in most pairwise comparisons, but by margins we do not claim as significant. We release the corrected method, the diagnostics, and comparisons against Task Arithmetic, TIES, DARE, and an FR-MERGING-style FFT baseline.

1 INTRODUCTION

Fine-tuning a large pretrained model separately for each of several tasks or domains is simple and effective, but produces one checkpoint per task. *Model merging* instead tries to combine several such fine-tunes – sharing a common base model – into a single set of weights that retains as much of each fine-tune's capability as possible, without retraining and without access to the original fine-tuning data. This is attractive whenever training or serving several full-size specialist checkpoints is expensive, and it has become a standard building block for combining instruction-following, reasoning, coding, multilingual, and safety-tuned variants of a shared open-weight base model.

The dominant family of merging methods operates on *task vectors*: for a domain d with fine-tuned weights θ_d and shared base weights θ_0, the task vector is $\tau_d = \theta_d - \theta_0$. A merged model is then produced as $\theta_{\text{merge}} = \theta_0 + \lambda \cdot f(\tau_1, \ldots, \tau_n)$ for some combination function f and edit-scale λ. **Task Arithmetic** (Ilharco et al., 2023) takes f to be a simple sum. **TIES-Merging** (Yadav et al., 2023) trims each task vector to its top-magnitude elements, elects a majority sign per parameter, and merges only the sign-agreeing entries, to reduce destructive interference between tasks that disagree on the direction of a given weight update. **DARE** (Yu et al., 2024) instead randomly drops a fraction of each task vector's entries and rescales the survivors before summing, which empirically reduces redundancy between task vectors and tends to improve merge quality, especially at higher numbers of merged tasks.

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 newer family of methods operates in the *frequency domain*. FR-MERGING (Zheng & Wang, 2025) applies a 2D FFT to each (2D) task-vector tensor, applies a fixed-radius high-pass mask to remove low-frequency content, inverts the transform, and combines the filtered task vectors with coefficients proportional to each task vector's mean; the full FREE-MERGING framework additionally extracts a rescaled sparse "expert" per task (its top-magnitude task-vector entries) and uses a trained router to add one expert per input at inference, recovering detail that filtering discards. The premise is that cross-task interference is concentrated in the low-frequency (broad, smooth) component of a task vector, while task-specific detail lives in the higher-frequency component.

We ask two questions about this frequency-domain approach. First, is the FFT the right transform? The FFT assumes an implicit periodic (circular) boundary condition that has no particular reason to hold for a weight matrix, and is known to compact less energy into few coefficients than the discrete cosine transform (DCT) for smooth, non-periodic, real-valued signals – a property already exploited for DCT-domain parameter-efficient adaptation of LLM weights (Du et al., 2025), but to our knowledge not yet applied to model merging. Second, does a single global frequency cutoff make sense across every parameter tensor in a network, when different layers plausibly carry different amounts of cross-task interference?

We propose DCT-ADAPTIVE: DCT-based spectral filtering with a per-layer adaptive cutoff set from a measured cross-task sign-disagreement statistic, in place of FR-MERGING's fixed global FFT cutoff. In implementing and evaluating this idea, however, we found that the more consequential problems were not about the transform at all, but about how the filtered task vectors are recombined. We report both the method and this diagnostic process, since we believe the two combination-rule pitfalls we identify (Section 3.4) are easy to introduce silently in any spectral- or filtering-based merging pipeline, and are large enough in effect size to dominate comparisons between merging methods if left uncontrolled.

Contributions.

1. We introduce DCT-domain high-pass filtering for model merging, and an adaptive per-layer cutoff rule driven by a cheap, data-free cross-task sign-interference score, in place of FR-MERGING's fixed global cutoff.
2. We identify and empirically diagnose two failure modes in the standard spectral-merging combination rule: an uncalibrated edit-scale hyperparameter, and an energy-proportional weighting scheme that quadratically discriminates against task vectors with naturally smaller norms. We measure both effects directly (layer-wise norm comparisons; per-domain combination-weight statistics averaged over 224 linear layers) rather than inferring them from end-task scores alone.
3. We evaluate on MergeBench (He et al., 2025b) with Llama-3.1-8B-Instruct and its five domain experts (instruction-following, math, multilingual, coding, safety), and replicate on Llama-3.2-3B-, Gemma-2-2B-, and Gemma-2-9B-Instruct, comparing against Task Arithmetic, TIES, DARE, and an FR-MERGING-style FFT baseline (FFT-FIXED). After the two combination-rule fixes, DCT-ADAPTIVE matches DARE in aggregate normalized performance on the four domains we aggregate, and it edges out FFT-FIXED at the uncalibrated $\lambda = 1.0$ stage on all four bases, by margins we do not claim as significant.

2 RELATED WORK

Task-vector arithmetic. Ilharco et al. (2023) showed that the difference between a fine-tuned and pretrained checkpoint (a "task vector") behaves additively: adding several task vectors to a shared base can combine their respective capabilities, and negating a task vector can suppress a capability. This underlies most subsequent merging methods, including ours.

Interference-aware merging. TIES-Merging (Yadav et al., 2023) identifies that naive summation of task vectors suffers from sign conflicts (different tasks pushing the same weight in opposite directions) and redundancy (many near-zero, uninformative updates), and addresses both with magnitude trimming and majority sign election before merging. DARE (Yu et al., 2024) instead shows that randomly dropping a large fraction of each task vector's entries (with rescaling to preserve expected magnitude) barely harms, and can even help, single-task performance, and that this sparsification substantially reduces destructive interference when merging several such sparsified task vectors –

making DARE a strong, simple baseline that we compare against throughout. Other data-free methods localize or restructure task-specific updates before merging, including Consensus TA (Wang et al., 2024), dataless Localize-and-Stitch (He et al., 2025a), and Task Singular Vectors (Gargiulo et al., 2025); MergeBench evaluates the first two, and we do not compare against them here (Section 7).

Frequency-domain merging. FR-MERGING (Zheng & Wang, 2025) reframes interference reduction as a signal-processing problem: it takes a 2D FFT of each task-vector tensor, removes low-frequency content with a fixed-radius high-pass mask (the same cutoff radius, as a fraction of the tensor's maximum frequency-domain radius, for every tensor in the network), and combines the filtered vectors with coefficients proportional to each task vector's mean (Zheng & Wang, 2025, Eq. 3). The full FREE-MERGING framework adds lightweight, rescaled experts built from each task's top-magnitude parameters, which a trained router adds to the merged model per input at inference. We adopt a static, data-free version of this recipe (transform, high-pass mask, magnitude-proportional weighting, and a sparse correction added directly to each filtered task vector) as our starting point and baseline, FFT-FIXED (Section 3), and modify two of its four components.

DCT for LLM weight updates. Separately from merging, DCT-domain parameterizations have been used for parameter-efficient fine-tuning of LLMs: Du et al. (2025) parameterize the weight update as an inverse DCT over selectively learnable frequency locations, showing both theoretically and empirically that frequency-domain adaptation with carefully chosen components can surpass the expressivity of low-rank decompositions, and that iDCT is a more efficient basis than iDFT for this purpose. We are not aware of prior work applying the DCT to the model-*merging* setting, where the goal is not parameter-efficient adaptation but reducing cross-task interference before combination.

Benchmarks. MergeBench (He et al., 2025b) standardizes evaluation of merging methods across five domains (instruction-following, math, multilingual understanding, coding, and safety) and several base-model families and scales, and defines a normalized-performance metric (merged-model score divided by the corresponding single-task fine-tuned expert's score) that we adopt. We use MergeBench's released domain-expert checkpoints for Llama-3.1-8B-Instruct (main experiments) and for Llama-3.2-3B-, Gemma-2-2B-, and Gemma-2-9B-Instruct (Section 5.4), and follow its evaluation protocol with three exceptions: we use fixed edit-scales rather than tuning them on MergeBench's surrogate validation tasks (Section 4), we evaluate math on GSM8K only, and we exclude math from the aggregate (Section 4.1).

3 METHOD

3.1 PRELIMINARIES

For a shared base model θ_0 and n domain-specialized fine-tunes $\theta_1, \ldots, \theta_n$, the task vector for domain d is $\tau_d = \theta_d - \theta_0$ (computed independently for every parameter tensor). A merging method produces $\theta_{\text{merge}} = \theta_0 + \lambda \cdot f(\tau_1, \ldots, \tau_n)$, where f combines the n task vectors into one update and λ is a global edit-scale applied uniformly. All spectral filtering described below operates independently on each 2D weight tensor (embeddings, projections, MLP weights); 1D tensors (norms, biases) are merged with plain task arithmetic.

3.2 SPECTRAL FILTERING: FFT BASELINE AND DCT VARIANT

FR-MERGING applies a 2D transform T to a task vector τ_d, zeroes all transform-domain coefficients within radius D_0 of the origin (a high-pass mask M_c parameterized by a cutoff fraction $c = D_0/D_{\max}$, where $D_{\max}$ is the tensor's maximum transform-domain radius), and inverts:

$$\text{filt}_d = T^{-1}\big(M_c \odot T(\tau_d)\big).$$

FR-MERGING takes T to be the 2D FFT. We additionally implement $T =$ 2D DCT-II (orthonormal). The FFT is complex-valued and implicitly treats the tensor as periodic; a weight matrix has no such periodicity, and this boundary-condition mismatch can smear energy across frequency components near tensor edges. The DCT is real-valued and constructed for exactly this non-periodic, real-signal setting, and is known to concentrate more of a smooth signal's energy into fewer low-frequency coefficients than the FFT does for the same signal (Ahmed et al., 1974; Rao & Yip, 1990). We

hypothesize that this lets a DCT high-pass filter at a given cutoff more cleanly separate broad, shared structure from genuinely localized, task-specific detail than an FFT filter at a matched cutoff.

Inspired by the lightweight experts of FREE-MERGING (Zheng & Wang, 2025), we add back a sparse correction: the top $k\%$ (by magnitude) of the raw, unfiltered task vector's entries, which the high-pass filter would otherwise discard along with the rest of the low-frequency content. Unlike FREE-MERGING, which rescales each task's expert and uses a trained router to add one expert per input at inference, we add every task's correction statically, which keeps the method data-free and routing-free:

$$\text{filt}'_d = \text{filt}_d + \text{Expert}_k(\tau_d), \qquad \text{Expert}_k(\tau_d)_i = \begin{cases} (\tau_d)_i & i \in \text{top-}k\% \text{ by } |(\tau_d)_i| \\ 0 & \text{otherwise} \end{cases}$$

We use $k = 1\%$ throughout, matching the expert size used by FREE-MERGING. We refer to this pipeline with T = FFT and a fixed cutoff as FFT-FIXED; it is our reimplementation of the FR-MERGING recipe rather than the original method.

3.3 ADAPTIVE PER-LAYER CUTOFF

FR-MERGING uses a single fixed cutoff fraction c for every 2D parameter tensor in the network. We instead compute a cutoff per tensor, based on how much the n task vectors disagree in sign at that tensor. For each parameter element, let $\bar{s}$ be the mean of the n task-vector signs at that element, and define the per-element disagreement as $1 - |\bar{s}|$ (for nonzero entries this equals $1 - |2(\text{agree} - 0.5)|$, where agree is the fraction of tasks whose sign matches any fixed reference task). It is 0 when all tasks agree in sign and grows as the signs split more evenly; with our $n = 5$ tasks it takes only the values 0, 0.4, and 0.8 (for 5–0, 4–1, and 3–2 splits). Averaging over all elements of a tensor gives a scalar interference score $I \in [0, 0.8]$ for that tensor. We then set

$$c = c_{\min} + I \cdot (c_{\max} - c_{\min}),$$

with $c_{\min} = 0.05$, $c_{\max} = 0.35$ in our experiments; since $I \leq 0.8$ for $n = 5$, the effective range of c is $[0.05, 0.29]$. Tensors where the domain experts disagree more in the direction of their weight updates receive a more aggressive (larger) high-pass cutoff, discarding more low-frequency content on the theory that this is where the disagreement-driven interference concentrates; tensors with little cross-task disagreement keep more of their low-frequency content. This adds no learned parameters and costs one extra pass over each tensor's task vectors relative to the fixed-cutoff variant.

3.4 COMBINING FILTERED TASK VECTORS, AND TWO PITFALLS

Like FR-MERGING, which weights each task by the mean of its task vector (Zheng & Wang, 2025, Eq. 3), FFT-FIXED weights tasks in proportion to their magnitude; we use the ℓ_2 norm of each filtered-and-corrected task vector, which we refer to as energy-proportional weighting:

$$w_d = \frac{\|\text{filt}'_d\|}{\sum_j \|\text{filt}'_j\|}, \qquad f(\tau_1, \ldots, \tau_n) = \sum_d w_d \cdot \text{filt}'_d.$$

This is a reasonable-looking convex combination, but it has two consequences that only become apparent when compared quantitatively, layer-by-layer, against non-spectral baselines.

(a) Edit-scale calibration. Our Task Arithmetic and DARE baselines apply an edit-scale of $\lambda = 0.3$ to their combined update, a common choice that keeps the merged model close to the base; unlike MergeBench's protocol, we do not tune it on validation data. Throughout, "calibration" refers to adopting this same $\lambda = 0.3$ for the spectral methods, not to a tuning procedure. A spectral-merging implementation that inherits a different default (we started from $\lambda = 1.0$) produces larger edits at matched layers: at one representative `mlp.down_proj` tensor in our Llama-3.1-8B-Instruct experiments, the final combined update had norm 2.62 for our uncalibrated spectral method versus 2.27 for Task Arithmetic and 0.64 for DARE, both at $\lambda = 0.3$. This alone is enough to substantially change downstream task performance, independent of anything about frequency-domain filtering; we quantify its effect on our benchmark in Section 5.2.

(b) Weighting-scheme bias. Because the combination weight w_d is itself proportional to the magnitude $\|\text{filt}'_d\|$, and that same vector filt'_d is what w_d multiplies, a task's net contribution to $\sum_d w_d \cdot \text{filt}'_d$

scales roughly with the *square* of its own task-vector magnitude, not linearly with it. If different domains' fine-tuning naturally produces task vectors of very different typical magnitude – which we find to be the case (Section 5.3) – this weighting rule does not merely give smaller-magnitude domains a smaller-than-equal share; it can give them a share far below even what their magnitude alone would justify. We replace w_d with a uniform weight $w_d = 1/n$ and show this closes the remaining gap to DARE (Section 5.3).

4 EXPERIMENTAL SETUP

Models. We use MergeBench's Llama-3.1-8B-Instruct setting: a shared base model (`meta-llama/Llama-3.1-8B-Instruct`; Grattafiori et al., 2024) and five domain-specialized fine-tunes released by MergeBench, one per evaluation domain (instruction-following, math, multilingual, coding, safety). The math expert is trained with SFT followed by GRPO (Shao et al., 2024); we discuss a consequence of this for evaluation in Section 4.1.

Evaluation domains and tasks. Following MergeBench's protocol: *instruction-following* via IFEval (Zhou et al., 2023); *math* via GSM8K (Cobbe et al., 2021) with chain-of-thought 8-shot prompting (`gsm8k_cot` in lm-evaluation-harness; Gao et al., 2024); *multilingual* via m_MMLU, ARC, and HellaSwag (Lai et al., 2023) in French, Spanish, German, and Russian (12 tasks, averaged); *coding* via HumanEval+ and MBPP+ (Chen et al., 2021; Austin et al., 2021; Liu et al., 2023) (bigcode-evaluation-harness, Ben Allal et al., 2022; 10 samples per problem, temperature 0.2, pass@1 averaged over both tasks); *safety* via WildGuardTest (Han et al., 2024), HarmBench (Mazeika et al., 2024), XSTest (Röttger et al., 2024), and Do-Anything-Now (Shen et al., 2024), generated with vLLM (Kwon et al., 2023) and scored with the WildGuard classifier, aggregated into a single 0–1 safety score (mean of $1 -$ harm rate across the harm-rate metrics and raw accuracy for XSTest).

Baselines. Task Arithmetic ($\lambda = 0.3$); TIES (density 0.2, disjoint sign-merge, $\lambda = 1.0$ following the standard TIES convention where λ scales an already-sparsified, already-normalized update); DARE (drop rate 0.5, $\lambda = 0.3$, sparsified task vectors averaged over tasks with a fixed random seed); FFT-FIXED (our static reimplementation of the FR-MERGING recipe: FFT, fixed cutoff fraction 0.15, energy-proportional weights, statically added sparse correction; Section 3). All edit-scales are fixed rather than tuned on validation data. MergeBench instead tunes $\lambda \in \{0.1, \ldots, 1.0\}$ (and sparsity) on surrogate validation tasks, so our Task Arithmetic and TIES numbers are not directly comparable to its tuned results.

Our methods. DCT-FIXED (DCT in place of FFT, same fixed cutoff 0.15) and DCT-ADAPTIVE (DCT with the adaptive per-layer cutoff of Section 3.3). We report both, together with FFT-FIXED, at three stages to isolate the effect of each combination-rule fix: (i) as originally implemented ($\lambda = 1.0$, energy-proportional weighting), (ii) with edit-scale corrected to $\lambda = 0.3$, and (iii) with both the edit-scale correction and equal ($1/n$) combination weighting. We do not evaluate an FFT-ADAPTIVE variant, so the transform and cutoff effects are not fully separated (Section 6).

Metrics. For each domain we report the merged model's raw task score, and MergeBench's normalized performance, defined as the merged model's score divided by the corresponding single-task fine-tuned expert's score on that same domain. We report the average normalized performance across domains as our headline aggregate metric.

4.1 A CAVEAT ON THE MATH DOMAIN

The released math expert is GRPO-trained and characteristically answers in LaTeX `\boxed{N}` notation. The `gsm8k_cot` task's exact-match answer extractor does not parse this format, and scores the math expert's raw accuracy at 0.047 (flexible-extract) / 0.106 (strict-match) – *lower than every merged model we evaluate*, despite the expert's reasoning and final numeric answers being correct in the large majority of samples we manually inspected. Because normalizing by an artificially deflated expert score produces uninterpretably large ($\sim$17$\times$) normalized-performance values for math specifically, we report raw math scores for completeness but exclude math from the aggregate normalized-performance average throughout. We flag this as an evaluation-harness compatibility issue with GRPO-style outputs, not a property of any merging method.

5 RESULTS

5.1 MAIN COMPARISON

Table 1 reports raw domain scores and normalized performance (excluding math from the aggregate, per Section 4.1) for all baselines and for our method after both combination-rule fixes (edit-scale calibration and equal weighting). DCT-ADAPTIVE achieves the highest aggregate normalized performance among the spectral variants, though all three are within 0.005 of one another, and matches the strongest baseline in our comparison, DARE (0.966 vs. 0.965). On raw math, which we do not aggregate (Section 4.1), DCT-ADAPTIVE scores lowest of all methods (0.798 vs. 0.809–0.829).

Table 1: Raw domain scores and normalized performance (merged / single-task expert), Llama-3.1-8B-Instruct, MergeBench. Math is excluded from the aggregate average (Section 4.1); raw math scores are shown for completeness. All three spectral rows (FFT-FIXED, DCT-FIXED, DCT-ADAPTIVE) reflect both combination-rule fixes ($\lambda = 0.3$, equal weighting); see Table 2 for the effect of each fix in isolation.

	Raw score					Normalized (excl. math)	
Method	Instr.	Math	Multiling.	Coding	Safety	Per-domain range	**Avg**
Single-task expert	0.694	0.047[†]	0.495	0.625	0.941	—	—
Task Arithmetic	0.356	0.822	0.473	0.573	0.850	0.51–0.96	0.823
TIES	0.428	0.829	0.452	0.556	0.857	0.62–0.91	0.832
DARE	0.623	0.809	0.486	0.609	0.948	0.90–1.01	**0.965**
FFT-FIXED	0.597	0.811	0.487	0.622	0.947	0.86–1.01	0.961
DCT-FIXED	0.614	0.809	0.486	0.612	0.945	0.88–1.00	0.962
DCT-ADAPTIVE (ours)	0.619	0.798	0.487	0.615	0.945	0.89–1.00	**0.966**

[†] Deflated by an answer-format mismatch with the GRPO-trained expert; see Section 4.1. Not used to compute the aggregate.

Per-domain, DCT-ADAPTIVE is tied with DARE on multilingual (0.984 vs. 0.983 normalized), slightly ahead on coding (0.984 vs. 0.974), and slightly behind on safety (1.004 vs. 1.007) and instruction-following (0.892 vs. 0.897). These differences are small relative to benchmark sampling error (for scale, one binomial standard error on IFEval's 541 prompts is about 0.02 at the accuracies we observe), and we did not run significance tests. Instruction-following is also where, before the combination-rule fixes, our method trailed DARE by a much larger margin (Section 5.3).

5.2 ABLATION: EDIT-SCALE CALIBRATION

Table 2 shows the effect of correcting the edit-scale λ from its uncalibrated default of 1.0 to 0.3 (matching Task Arithmetic and DARE), before any change to the combination weighting. This single change raises aggregate normalized performance from 0.803/0.784/0.806 to 0.940/0.942/0.945 for FFT-FIXED, DCT-FIXED, and DCT-ADAPTIVE respectively – the majority of the total improvement we report.

Table 2: Aggregate normalized performance (excl. math) at each stage of the combination-rule fix.

Method	λ=1.0, energy-wt. (orig.)	λ=0.3, energy-wt.	λ=0.3, equal-wt. (final)
FFT-FIXED	0.803	0.940	0.961
DCT-FIXED	0.784	0.942	0.962
DCT-ADAPTIVE	0.806	0.945	**0.966**

5.3 ABLATION: COMBINATION WEIGHTING

To directly test the weighting-scheme bias described in Section 3.4, we measured, for every 2D linear-layer parameter tensor in the network (224 tensors), the energy-proportional weight w_d each of the 5 domains receives under the energy-proportional rule of Section 3.4, and compared it to the

equal $1/5 = 0.20$ share. Table 3 reports the average across all 224 tensors. Instruction-following and multilingual receive 28% and 10% of their fair share respectively, while math, safety, and coding receive 117–195% of theirs – consistent with these domains' fine-tuning producing task vectors of systematically larger norm (we measure a 7–$18\times$ disparity in raw task-vector norm between domains at a representative layer). Switching to equal weighting (rightmost column of Table 2) improves every spectral method, with the largest single-domain gain on instruction-following, the domain most starved by the original scheme. The ordering of these shares matches the size of each domain's fine-tuning set in MergeBench (math, then safety, coding, instruction-following, and multilingual; He et al., 2025b), consistent with the norm differences reflecting how much each expert was trained rather than how much each domain should contribute. Equal weighting also shrinks the combined update, because energy-proportional weights concentrate it on the largest task vectors; we have not separated this effect from the share correction with a norm-matched control, so part of the gain may come from a smaller edit.

Table 3: Average energy-proportional combination weight received by each domain (fair share = 0.20), averaged over 224 linear-layer parameter tensors.

	Instruction	Math	Multilingual	Coding	Safety
Avg. weight received	0.056	0.391	0.020	0.235	0.299
% of fair share	28%	195%	10%	117%	149%

5.4 Replication across model families and scales

To test whether the two combination-rule findings generalize beyond Llama-3.1-8B-Instruct, we replicate the full spectral-merging pipeline (FFT-Fixed/DCT-Fixed/DCT-Adaptive at each of the three stages) on three additional bases from MergeBench: Llama-3.2-3B-Instruct, Gemma-2-2B-Instruct, and Gemma-2-9B-Instruct (Grattafiori et al., 2024; Gemma Team, 2024), using each base's MergeBench domain experts, the same $\lambda = 0.3$, and equal weighting in the final stage. This adds 27 spectral merges and 108 domain evaluations to the 9 merges and 36 evaluations on Llama-3.1-8B. Table 4 reports the pooled raw score over the four non-math domains (math is excluded per Section 4.1); normalized scores against per-model expert checkpoints follow the same ordering.

Table 4: Replication across four bases. "Pooled raw score" is the simple mean over the four non-math domains (instruction, multilingual, coding, safety), averaged across the three spectral methods at each stage. This is a raw-score pooling summary and is not the same as the normalized-average metric reported in Table 1. "Lift" is λ=0.3 minus λ=1.0; "+EW" additionally applies equal weighting.

Base model	λ=1.0	λ=0.3	λ=0.3+EW	Lift	+EW vs. λ=1.0	DARE
Llama-3.1-8B	0.538	0.646	**0.665**	$+0.108$	$+0.127$	0.667
Llama-3.2-3B	**0.623**	0.619	0.618	-0.003	-0.004	0.621
Gemma-2-2B	**0.546**	0.535	0.536	-0.012	-0.011	0.536
Gemma-2-9B	0.574	0.642	**0.650**	$+0.069$	$+0.076$	0.650

Three observations. (i) *Calibration is the dominant effect, but it is model-dependent.* On the two larger bases (Llama-3.1-8B, Gemma-2-9B), correcting the edit-scale lifts pooled scores by $+0.108$ and $+0.069$ and brings the spectral methods to parity with DARE; on the two smaller bases, λ=1.0 is the best of the three stages, reducing λ lowers pooled scores slightly (-0.003 and -0.012), and the uncalibrated merges match or exceed DARE (0.623 vs. 0.621 and 0.546 vs. 0.536). The over-editing failure mode therefore appears only on the two larger bases. MergeBench trained the 8B/9B experts with a $5\times$ higher learning rate than the 2B/3B experts and added GRPO for their math experts (He et al., 2025b), so this split may reflect larger task vectors on those bases rather than base-model strength as such. Either way, it argues for tuning λ per base rather than adopting any single value. (ii) *The DCT-vs-FFT contrast remains small everywhere.* The maximum pooled gap between DCT-Adaptive and FFT-Fixed within any model$\times$stage cell is at most 0.012 (at most 0.015 for either DCT variant; in normalized units, the largest reported gap is DCT-Fixed trailing FFT-Fixed by 0.019 at λ=1.0 on Llama-3.1-8B, Table 2). We ran one seed per configuration and did not test

these differences for significance. DCT-ADAPTIVE wins the λ=1.0 stage on all four bases. We therefore claim consistency, not significance, for the transform choice. (iii) *Weighting matters least once λ is calibrated.* Equal weighting adds $+0.019$ and $+0.008$ on the two larger bases and is flat on the smaller ones, consistent with the weighting bias being a second-order correction on top of the edit-scale fix.

6 DISCUSSION

Two findings stand out. First, the transform and cutoff choices have small, mixed effects. At the uncalibrated $\lambda = 1.0$ stage, DCT-ADAPTIVE edges out FFT-FIXED on all four bases, but on Llama-3.1-8B the DCT with a fixed cutoff trails FFT-FIXED (0.784 vs. 0.803), and DCT-ADAPTIVE recovers the gap (0.806). At the calibrated stages the three variants are within 0.005 of one another on Llama-3.1-8B, and the FFT is slightly ahead on the two smaller bases. Because the adaptive rule can also change the average cutoff relative to the fixed 0.15 (Section 3.3), and we do not evaluate an FFT-ADAPTIVE variant, we cannot attribute these differences to per-layer adaptivity or to the transform itself, and we claim neither as significant. Second, and more surprisingly, these differences are small compared to the effect of two combination-rule issues that have nothing to do with frequency-domain filtering as such, and that we suspect are easy to introduce unnoticed in any spectral-merging implementation, since neither produces an obvious error – both merges run to completion and produce plausible-looking merged weights; the failure only shows up as a quantitative performance gap against calibrated baselines. We would encourage future work in this area to report edit-scale and combination-weight statistics (e.g., the per-task weight share we compute in Table 3) alongside end-task numbers, since the latter alone cannot distinguish a transform-related improvement from a combination-rule artifact.

After both fixes, DCT-ADAPTIVE and DARE are within 0.006 of each other (raw) on every aggregated domain, so we see no meaningful remaining gap. Instruction-following is, however, the domain most sensitive to the combination rule: it gains the most from equal weighting (Section 5.3), and our method trailed DARE there by a much larger margin before the fixes. Its task vectors are among the smallest in our setting (only multilingual receives less weight; Table 3), and it is plausible that whatever fine-grained, distributed pattern instruction tuning relies on is more easily disrupted by *any* shared per-tensor spatial-frequency mask (ours or FR-MERGING's) than by DARE's independent per-element dropout, which never removes a coordinated pattern across an entire frequency band at once. An alternative explanation is that larger edits from the other domains overwrite instruction-following ability the Instruct base already has; comparing against the base model's own scores would distinguish the two. Domain-aware cutoff calibration, rather than a single sign-interference-driven schedule, is a natural next step.

7 LIMITATIONS

Our method inherits the central premise of frequency-domain merging: that cross-task interference concentrates in the low spatial frequencies of a task vector. Because spatial frequency in a weight matrix is defined relative to an arbitrary ordering of hidden units, this premise is a heuristic rather than a guarantee, consistent with the small transform effects we observe. The adaptive cutoff is driven by a sign-only interference score that ignores update magnitudes, and its range depends on the number of tasks (for $n = 5$ the cutoff never exceeds 0.29), so $c_{\min}$ and $c_{\max}$ may need re-tuning as n changes. The cutoff is also shared by all tasks at a tensor, so a domain whose useful signal lies in the filtered band can lose it; a domain-aware cutoff is a natural extension. The sparse correction re-adds each task's largest raw entries statically, with a fixed k and no resolution of sign conflicts among them. On the combination side, the method still requires an edit-scale λ, which our results show is base-dependent and which the method provides no data-free way to set, and equal weighting ignores genuine differences in how much each domain should contribute. Empirical caveats, namely fixed edit-scales (Section 4), a single seed per configuration (Section 5), and the math-evaluation confound (Section 4.1), are discussed alongside the results, and we leave comparisons with Consensus TA and Localize-and-Stitch to future work.

8 CONCLUSION

We revisited frequency-domain model merging along two axes – transform choice (DCT vs. FFT) and cutoff granularity (adaptive per-layer vs. fixed global) – and found their effects to be small and mixed relative to single-run noise. We additionally found and corrected two combination-rule issues, an uncalibrated edit-scale and an energy-proportional weighting scheme that quadratically penalizes smaller-magnitude task vectors, that dominate the eventual comparison against strong non-spectral baselines far more than either axis we set out to study. After both fixes, our method matches DARE in aggregate normalized performance across the four MergeBench domains we aggregate on Llama-3.1-8B-Instruct, and edges out the FFT baseline at the uncalibrated $\lambda = 1.0$ stage on all four bases. Our merging code and the diagnostic scripts used to measure both combination-rule issues are included in the anonymized supplementary material, and we encourage future spectral-merging work to report the same layer-wise norm and weight-share statistics we used to find them.

REPRODUCIBILITY STATEMENT

All experiments use public artifacts: MergeBench's domain-expert checkpoints and evaluation harness (He et al., 2025b) and the open-weight Llama-3.1-8B-, Llama-3.2-3B-, Gemma-2-2B-, and Gemma-2-9B-Instruct models. Section 3 specifies the merging procedures (transforms, masks, adaptive-cutoff rule, sparse correction, and combination weights), and Section 4 specifies the baselines, hyperparameters ($\lambda \in \{1.0, 0.3\}$, $c_{\min} = 0.05$, $c_{\max} = 0.35$, $k = 1\%$, orthonormal DCT-II), and evaluation settings. The anonymized supplementary material contains the merging code, the diagnostic scripts behind Section 3.4 and Table 3, the evaluation launch scripts, and the compilation script that produces every reported number from a single results file (`results_summary.csv`) of raw per-domain outputs.

ETHICS STATEMENT

This work merges publicly released models and evaluates them on public benchmarks; it involves no human subjects and no new data collection. Our safety evaluation uses adversarial and harmful prompts from established benchmarks (WildGuardTest, HarmBench, XSTest, and Do-Anything-Now) solely to measure refusal behavior, and we release no new harmful content. Because merging can weaken safety behavior contributed by an aligned base model or a safety expert, we report safety scores for every merge and recommend evaluating safety after merging, since edit-scale and weighting choices can change it.

THE USE OF LARGE LANGUAGE MODELS (LLMS)

In this work, we used generative AI tools for the following tasks that require disclosure under the ICLR 2027 policy: providing feedback on the research methodology, experiments, and interpretation of results, and implementing parts of the merging and evaluation scripts, based on a draft of this paper. We did not use generative AI tools to generate data, formulate hypotheses, or design the experiments, and the remaining required-disclosure tasks are not applicable to this work. We also used generative AI tools for tasks with recommended disclosure: drafting and debugging evaluation scripts, restructuring the LaTeX source, checking bibliographic metadata against primary sources, identifying relevant literature, and drafting revisions to parts of the text. We reviewed all AI-assisted work: AI-assisted code was tested by the authors, the bibliography and reported numbers were checked against their source material and raw outputs, and the manuscript was revised by the authors before submission. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REFERENCES

Nasir Ahmed, T. Natarajan, and K. R. Rao. Discrete cosine transform. *IEEE Transactions on Computers*, C-23(1):90–93, 1974. doi: 10.1109/T-C.1974.223784.

Jacob Austin et al. Program synthesis with large language models. In *arXiv preprint arXiv:2108.07732*, 2021.

Loubna Ben Allal, Niklas Muennighoff, Logesh Kumar Umapathi, Ben Lipkin, and Leandro von Werra. A framework for the evaluation of code generation models. `https://github.com/bigcode-project/bigcode-evaluation-harness`, 2022.

Mark Chen, Jerry Tworek, et al. Evaluating large language models trained on code. In *arXiv preprint arXiv:2107.03374*, 2021.

Karl Cobbe et al. Training verifiers to solve math word problems. In *arXiv preprint arXiv:2110.14168*, 2021.

Zhekai Du, Yinjie Min, Jingjing Li, Ke Lu, Changliang Zou, Liuhua Peng, Tingjin Chu, and Mingming Gong. LoCA: Location-aware cosine adaptation for parameter-efficient fine-tuning. In *International Conference on Learning Representations (ICLR)*, 2025.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The language model evaluation harness, July 2024. URL `https://zenodo.org/records/12608602`.

Antonio Andrea Gargiulo, Donato Crisostomi, Maria Sofia Bucarelli, Simone Scardapane, Fabrizio Silvestri, and Emanuele Rodolà. Task singular vectors: Reducing task interference in model merging. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18695–18705, 2025.

Gemma Team. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.

Aaron Grattafiori et al. The Llama 3 Herd of Models. In *arXiv preprint arXiv:2407.21783*, 2024.

Seungju Han, Kavel Rao, Allyson Ettinger, Liwei Jiang, Bill Yuchen Lin, Nathan Lambert, Yejin Choi, and Nouha Dziri. WildGuard: Open one-stop moderation tools for safety risks, jailbreaks, and refusals of LLMs. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2024.

Yifei He, Yuzheng Hu, Yong Lin, Tong Zhang, and Han Zhao. Localize-and-stitch: Efficient model merging via sparse task arithmetic. *Transactions on Machine Learning Research (TMLR)*, 2025a.

Yifei He, Siqi Zeng, Yuzheng Hu, Rui Yang, Tong Zhang, and Han Zhao. MergeBench: A benchmark for merging domain-specialized LLMs. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2025b. arXiv:2505.10833; code: github.com/uiuctml/MergeBench.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. In *International Conference on Learning Representations (ICLR)*, 2023.

Woosuk Kwon et al. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles (SOSP)*, 2023.

Viet Dac Lai, Chien Van Nguyen, Nghia Trung Ngo, Thuat Nguyen, Franck Dernoncourt, Ryan A. Rossi, and Thien Huu Nguyen. Okapi: Instruction-tuned large language models in multiple languages with reinforcement learning from human feedback. In *Conference on Empirical Methods in Natural Language Processing (EMNLP): System Demonstrations*, pp. 318–327, 2023.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by ChatGPT really correct? rigorous evaluation of large language models for code generation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Mantas Mazeika et al. HarmBench: A standardized evaluation framework for automated red teaming and robust refusal. In *arXiv preprint arXiv:2402.04249*, 2024.

K. Ramamohan Rao and P. C. Yip. *Discrete Cosine Transform: Algorithms, Advantages, Applications*. Academic Press, San Diego, CA, 1990. ISBN 978-0-12-580203-1.

Paul Röttger, Hannah Rose Kirk, Bertie Vidgen, Giuseppe Attanasio, Federico Bianchi, and Dirk Hovy. XSTest: A test suite for identifying exaggerated safety behaviours in large language models. In *Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pp. 5377–5400, 2024.

Zhihong Shao, Peiyi Wang, et al. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. In *arXiv preprint arXiv:2402.03300*, 2024.

Xinyue Shen et al. Do anything now: Characterizing and evaluating in-the-wild jailbreak prompts on large language models. In *arXiv preprint arXiv:2308.03825*, 2024.

Ke Wang, Nikolaos Dimitriadis, Guillermo Ortiz-Jiménez, François Fleuret, and Pascal Frossard. Localizing task information for improved model merging and compression. In *International Conference on Machine Learning (ICML)*, pp. 50268–50287, 2024.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. TIES-Merging: Resolving interference when merging models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

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 *International Conference on Machine Learning (ICML)*, 2024.

Shenghe Zheng and Hongzhi Wang. FREE-Merging: Fourier transform for efficient model merging. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025. arXiv:2411.16815.

Jeffrey Zhou et al. Instruction-following evaluation for large language models. In *arXiv preprint arXiv:2311.07911*, 2023.