THE BEHAVIORAL RATE OF FINE-TUNING: COMPRESSION AS A MEASURE OF USEFUL INFORMATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Low-Rank Adaptation (LoRA) adapts a frozen language model through a compact learned update, making it a natural setting for asking how much information fine-tuning needs to write. Yet adapter rank, parameter count, or numerical reconstruction error do not directly tell us how many bits are required to preserve the acquired behavior. We therefore define an operational behavioral rate $R^\star$ as the smallest serialized size of a compressed adapter that, once reloaded into the frozen base model, retains a fixed fraction of its held-out fine-tuning gain. Across LoRA fine-tuning experiments, this rate increases with distinct training content at fixed optimization budgets, but is not determined by intuitive metrics like source description length, fine-tuning gain, or weight-space reconstruction error. Most importantly, the same datasets exhibit substantially different, and even reordered, rates across frozen base models, showing that fine-tuning compressibility is a property of the dataset–receiver pair rather than of the dataset alone. However, we show that correction spectra measured on the data are able to rank these rates within a receiver. Finally, we study a striking noise-reducing effect of compression: adapters trained on deliberately corrupted datasets fall far below the frozen model, yet after compression score above it (23.7% to 82.9% on GSM8K, against 77.6% for frozen Qwen2.5-7B), consistently across tested families and sizes. The damage is unevenly concentrated in the adapter's leading singular directions, which compression appears to filter. Away from its training task, the compressed corrupted adapter also generalizes better than a clean fine-tune: on harder math it retains 97% of base accuracy, versus 44% for the clean adapter. Together, these results suggest that the information cost of fine-tuning depends on the receiver, target behavior, and how the required corrections are represented in the learned update.

1 INTRODUCTION

Fine-tuning changes the behavior of a pretrained model by writing an update relative to what that model already knows. How many bits are actually required to preserve this acquired behavior? Parameter count, adapter rank, and numerical precision measure storage, but need not measure the information that matters functionally. Low-Rank Adaptation (LoRA) (Hu et al., 2022) provides a clean setting for studying this question: the base model remains frozen while the learned update is isolated in a separate adapter that can be compressed after training.

We take a receiver-relative rate–distortion view of this process. As illustrated in Figure 1, fine-tuning acts as an encoder, the adapter as a finite-rate message, and the frozen model as side information available when that message is decoded. We measure rate by the exact serialized size of a compressed adapter and distortion by the behavior of the resulting composed model. For a fixed retention target, the *behavioral rate* $R^\star$ is the smallest measured rate that preserves the required fraction of the held-out fine-tuning gain.

This operational definition exposes a gap between numerical and behavioral compression. Lower weight reconstruction error need not preserve more learned behavior, and comparable bit budgets can have different effects depending on where compression acts. Behavioral rate also increases with distinct training content at fixed optimization budget. Most importantly, the same datasets can require substantially different rates across frozen models, with their ordering changing across receivers. A correction-gradient spectrum computed from the data and receiver ranks $R^\star$ well within

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.

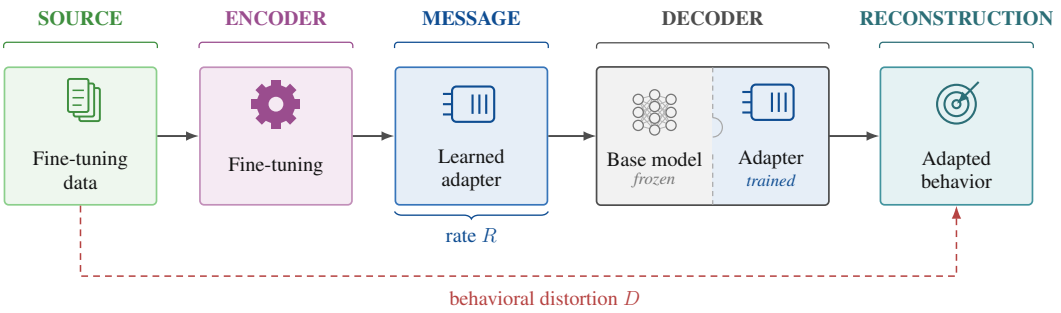

Figure 1: Information-theoretic view of parameter-efficient fine-tuning. R is the number of bits used by the adapter, D measures failure to preserve the behavior induced by fine-tuning.

a fixed receiver, but does not transfer reliably across new receivers.

Compression can also reveal structure within a learned update. Under deliberately corrupted reasoning supervision, fine-tuning can strongly damage the frozen model, yet aggressive compression recovers much of the lost performance and can exceed the frozen baseline. On Qwen2.5-7B, for example, GSM8K accuracy falls from 77.6% to 23.7% after corrupted fine-tuning and returns to 82.9% after compression. Singular-direction analyses show that harmful and useful effects are distributed unevenly across the update, while matched-norm controls show that the recovery is not exactly equivalent to scalar rescaling.

Finally, once an update has been trained, its behavioral rate closely tracks how far it can be shrunk before losing its gain. A simple codec-specific rule, $R^\star(0.90) \approx \alpha_{90}/\gamma$, achieves an RMSE of 0.068 bits per value on 42 checkpoints from five receivers not used to fix the rule, although the evaluation remains retrospective. Together, these results suggest that the information cost of fine-tuning is not an intrinsic property of the data or of parameter displacement alone: it depends on the frozen receiver, the behavior to preserve, and how the learned corrections are represented within the update.

2 Related Work

Parameter-efficient fine-tuning and adapter structure. LoRA (Hu et al., 2022) and QLoRA (Dettmers et al., 2023) provide compact parameterizations for adapting frozen language models, while earlier work on intrinsic dimensionality showed that effective fine-tuning can lie in a low-dimensional subspace (Aghajanyan et al., 2021). More recent work has emphasized that the components of a learned adapter need not be functionally equivalent. Zhu et al. (2024) study asymmetric roles within low-rank adapters, Shuttleworth et al. (2025) identify high-ranking singular directions introduced by LoRA whose attenuation can reduce forgetting, and Tian et al. (2026) identify neutral or harmful singular directions that can be reweighted using calibration gradients. Our work is complementary: we use post-training compression to probe which components of an update are required to preserve its acquired behavior.

Quantization and adapter compression. Low-bit post-training quantization has been extensively studied for language models, including ZeroQuant (Yao et al., 2022), GPTQ (Frantar et al., 2023), SmoothQuant (Xiao et al., 2023), and AWQ (Lin et al., 2024); more recent work characterizes behavior in extremely low-bit regimes (Liu et al., 2025). Quantization has also been extended to trained adapters: Mirzaei et al. (2025) compress LoRA updates to ultra-low precision while preserving task performance, while Tan et al. (2026) study adapter capacity and memorization directly in bits. In contrast, we use compression primarily as a measurement tool: our rate is the serialized size required to preserve the behavior induced by fine-tuning.

Compression, description length, and rate–distortion. Model compressibility has been connected to generalization and description length (Arora et al., 2018; Lotfi et al., 2022; Urdshals et al., 2025), and rate–distortion ideas have been applied to language-model compression (Young, 2025). Our setting differs in two respects: the frozen model acts as shared side information, and distortion is measured behaviorally after the compressed adapter is decoded back into that receiver. The resulting rate is therefore relative to both the learned update and the frozen model with which it is composed.

3 FUNCTIONAL STRUCTURE OF ADAPTER COMPRESSION

We begin by examining how post-training compression affects fine-tuned behavior. Controlled compression lets us separate numerical reconstruction from functional preservation while keeping both the learned checkpoint and the frozen model fixed. These diagnostics show that ordinary parameter-space fidelity is not sufficient to predict which compressed adapters preserve the behavior acquired during fine-tuning.

3.1 EXPERIMENTAL SETTING: ADAPTER COMPRESSION

We study post-training compression of LoRA adapters across frozen language models from 0.5B to 32B parameters, including Qwen2.5-{0.5B, 1.5B, 3B, 7B, 14B, 32B}, Qwen3-8B-Base, Mistral-7B-v0.1, Llama-3.1-8B, and Gemma-2-9B. Each adapter is trained once and then compressed over a range of operating points, without retraining. At one bit per value and above, we quantize the LoRA factors row-wise; below one bit, we additionally remove paired rank directions. Each compressed representation, including its decoding metadata, is serialized to disk, reloaded, and evaluated in composition with the frozen model. We report realized file size normalized by the number of scalar values in the uncompressed adapter. Full training configurations, codec details, datasets, and evaluation procedures are given in Appendix A.

3.2 MEASURING COMPRESSION THROUGH BEHAVIOR

The object we ultimately want to preserve is not the adapter itself, but the behavior added by fine-tuning. Because every operating point is obtained from the same trained checkpoint, differences after compression isolate the effect of representation from variation in optimization.

Let L_0 denote the held-out loss of the frozen model, L_{raw} that of the uncompressed adapter, and L_r the loss after compression to realized rate r. We define retained gain as

$$\mathrm{Retain}(r) = \frac{L_0 - L_r}{L_0 - L_{\mathrm{raw}}}, \tag{1}$$

so that 0 corresponds to the frozen model and 1 to the uncompressed adapter; values above one are possible when compression improves held-out loss. For comparisons across conditions, we define

$$R^\star(\tau) = \min\{r : \mathrm{Retain}(r) \geq \tau\}, \tag{2}$$

and use $\tau = 0.90$ unless stated otherwise. $R^\star(0.90)$ is therefore an operational, codec-dependent behavioral rate: the smallest measured reloadable adapter preserving at least 90% of the original fine-tuning gain.

Figure 2 (left and center) illustrates the resulting rate-retention frontier. At fixed optimization budget, more distinct training content shifts the frontier toward higher rates; at fixed dataset size, $R^\star(0.90)$ also tends to increase with source-problem diversity. We return to what determines these shifts in Section 6.

A natural alternative would be to measure preservation through numerical reconstruction error. We test this by comparing uniform quantization with an adaptive allocator that minimizes LoRA weight reconstruction error under a rate constraint. Figure 2 (right) gives a direct counterexample: near one bit per value, the adaptive code achieves lower relative weight RMSE than uniform quantization (0.54 versus 0.56), yet retains only 3% of the held-out gain, compared with 86% for the uniform code. The allocator partly obtains its lower error by removing rows that are inexpensive under the Euclidean objective but behaviorally important; at higher rates, once rows are no longer removed, the two codecs converge.

Thus, lower parameter-space distortion need not imply lower behavioral distortion. Numerical reconstruction alone does not identify which perturbations are harmless once the adapter is composed with the frozen model. Additional codec details and controls are given in Appendix A.

4 A RECEIVER-RELATIVE RATE-DISTORTION VIEW OF FINE-TUNING

This section develops a receiver-relative rate–distortion view of fine-tuning. Our goal is not to derive an exact information-theoretic law for LoRA, but to formalize the empirical observations of

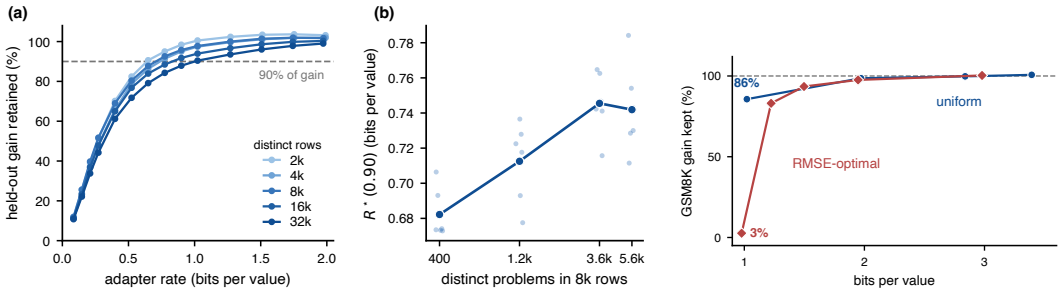

Figure 2: Behavioral rate is not a weight-space quantity. Left: retained fine-tuning gain versus exact serialized adapter rate across 2k–32k MetaMathQA rows at fixed compute. **Center:** $R^\star(0.90)$ at 8k rows as source-problem diversity increases. **Right:** near one bit per value, an RMSE-optimized codec achieves lower relative weight error than uniform quantization (0.54 vs. 0.56) but retains only 3% of the GSM8K gain, versus 86%. Curves on the left show seed means; small points are individual seeds. Mistral-7B, NF4, rank-16 LoRA, three seeds.

Section 3 and motivate the quantities studied later in the paper. We first treat the frozen model as shared side information, then characterize behavioral preservation through a functional geometry over adapter perturbations, and finally show how compression can improve behavior when the learned update is misaligned with the target utility. Detailed derivations and assumptions are given in Appendix B.

4.1 The Frozen Model as Side Information

A fine-tuning adapter is not used independently from the model on which it was trained. At inference time, its parameters act through their composition with the frozen base model. As illustrated in Figure 1, the adapter can therefore be viewed as a finite-rate message whose effect is interpreted by a particular receiver. Let M denote a frozen model, A a trained adapter, and c a finite code of length $\ell(c)$ from which a shared decoder reconstructs $\hat{A} = \mathrm{Dec}(c)$. The reconstructed model is $M \oplus \hat{A}$, where $\oplus$ denotes composition of the frozen model with its adapter. Since M is available at both encoding and decoding time, it acts as shared side information: the code only needs to specify the additional modification required by this receiver.

Compression should therefore be evaluated through the behavior of the composed model. For a utility U for which larger values are better, we define $D_{M,U}(A, \hat{A}) = \left[U(M \oplus A) - U(M \oplus \hat{A})\right]_+$, with the corresponding difference reversed for a loss. Given a tolerance ε, we define the *receiver-relative behavioral description length*

$$L^\star_{M,U}(A, \varepsilon) = \min_c \ell(c) \quad \text{s.t.} \quad D_{M,U}(A, \mathrm{Dec}(c)) \le \varepsilon. \tag{3}$$

Equation 3 makes explicit two dependencies: changing M changes how an update is interpreted, while changing U changes which differences between decoded updates matter. This is an operational one-shot coding problem for a deterministic checkpoint, not a Shannon rate-distortion function; the empirical rates studied here are therefore codec-dependent achievable rates rather than information-theoretic optima.

4.2 Behavioral Distortion Induces a Functional Geometry

The definition above does not yet specify which parameter perturbations are costly. Let a compressed adapter satisfy $\hat{A} = A + e$, where e is the compression error. Under a local expansion of the utility U, and when the first-order term is negligible for the perturbations considered, the leading decrease in utility can be written as

$$D_{M,U}(A, A + e) \approx \frac{1}{2} e^\top H_U e, \tag{4}$$

where $H_U \succeq 0$ denotes minus the local utility Hessian, or equivalently the local curvature of the corresponding loss near a local optimum.

This differs from ordinary weight reconstruction error, $D_{\text{weight}}(A, A+e) = \|e\|_2^2$, which assigns the same cost to every parameter direction. Equation 4, in contrast, weights an error by its orientation relative to the behavioral sensitivity of the composed model. Euclidean and functional distortion induce the same ordering over perturbations only in the isotropic case, when H_U is proportional to the identity on the subspace explored by compression.

This gives a local interpretation of the RMSE counterexample in Section 3: behavioral cost depends not only on the magnitude of a compression error, but on its orientation relative to the receiver-induced geometry. The matched-budget layer-location controls in Appendix C.3 provide complementary evidence that this geometry is strongly anisotropic across the network.

4.3 Why Compression Can Improve on the Base Model

Equation 4 assumes that the first-order term vanishes, i.e. that the trained adapter sits at an optimum of the behavior U we care about. Training only guarantees an optimum of the training loss. When the data ask for corrections that U does not reward, as with the corrupted supervision of Section 5, the two optima differ. Consider moving along the trained update, $M \oplus \alpha A$ for $\alpha \in [0, 1]$, so that $\alpha = 0$ is the frozen model and $\alpha = 1$ the adapter. To second order,

$$U(M \oplus \alpha A) \approx U(M) + \alpha\, g_U^\top A - \frac{\alpha^2}{2}\, A^\top H_U A, \tag{5}$$

with g_U the gradient of U at the frozen model. If the update first helps ($g_U^\top A > 0$) but is too large, U peaks at $\alpha^\star = g_U^\top A / A^\top H_U A < 1$, and every scale between 0 and $2\alpha^\star$ beats the frozen model while the full update may fall far below it. A code that effectively moves the update toward such an intermediate scale can then score above both. The same argument applies direction by direction: if the harmful excess lives in a few strong directions of A, cutting or shrinking those directions recovers the gain the rest of the update carries.

Compression-based generalization bounds motivate asking whether shorter adapter descriptions also induce smaller off-task changes (Arora et al., 2018; Lotfi et al., 2022). We evaluate this possibility in Appendix D.5, where the compressed corrupted adapter remains substantially closer to the frozen model on off-task probes than a clean fine-tune. This pattern is consistent with compression suppressing task-specific perturbations while preserving the useful part of the learned update, although it does not by itself establish a general relation between shorter adapter descriptions and better out-of-distribution behavior.

The same framework also suggests quantities that might anticipate compression difficulty before training. The per-example correction gradients $g_i = \nabla_\theta \ell_i(\theta_0)$ describe what the data ask of a particular receiver. Their second-moment geometry yields a spectrum whose log-determinant is analogous to the rate term appearing in classical quadratic rate-distortion models (Thomas & Joy, 2006). We derive this proxy in Appendix B.3 and test its empirical scope in Section 6.

5 Recovering Task-Relevant Structure through Compression

So far, compression has been treated as a source of distortion. We now ask whether its non-uniform action can instead be useful when the learned update itself contains harmful structure. We construct deliberately corrupted fine-tunes by modifying the supervision used during training. Our main intervention pairs each MetaMathQA question with the worked rationale of another example while retaining its original final answer. The resulting adapter is therefore trained on the correct final answer together with reasoning that is systematically mismatched to the input. This sharply degrades task performance and provides a controlled setting in which useful task signal and harmful supervision can coexist within the same learned update.

5.1 Compression Repairs Corrupted Fine-Tunes

Figure 3 shows that aggressive compression can reverse much of this damage across several base models. On Qwen2.5-7B, the corrupted adapter reduces GSM8K accuracy from 77.6% to 23.7%; after lossy compression, the same learned update reaches 82.9%. The effect also appears on Mistral-7B-Instruct and Llama-3.1-8B, despite their different frozen-model baselines.

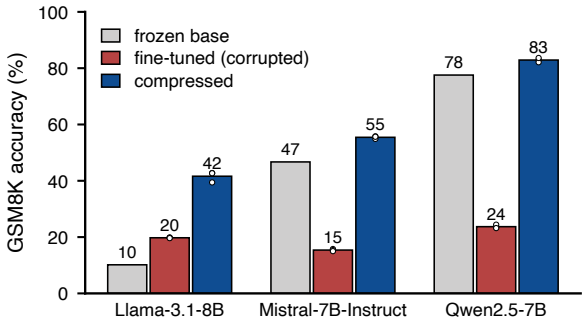

Figure 3: Compression can repair a corrupted fine-tune. GSM8K accuracy of the frozen model, an adapter trained on permuted rationales, and the same adapter after lossy compression, across three base models.

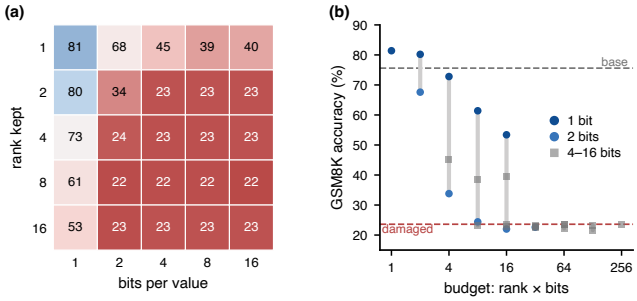

Figure 4: Rank and precision are not interchangeable under corrupted fine-tuning. (a) GSM8K accuracy after retaining rank r and coding each value at b bits. (b) The same configurations grouped by nominal budget $r \times b$.

Recovery depends strongly on how the compression budget is spent. Figure 4 varies both the number of retained rank directions and their precision for the same corrupted rank-16 adapter. At nominal budgets $r \times b \leq 16$, retaining more directions at one bit consistently outperforms retaining fewer directions at higher precision. Most strikingly, the rank-1 truncation reaches 81.4% accuracy at one bit, but only 39.6% at 16 bits. Thus, recovery cannot be explained by parameter count or nominal budget alone.

A substantial part of the effect is associated with shrinkage: rescaling the full-precision rank-1 update recovers most of the accuracy obtained by one-bit coding. Matched-norm controls nevertheless leave a smaller residual advantage for coarse coding, indicating that norm reduction does not explain the entire effect. Recovery is also not specific to selecting a single leading singular direction: full-rank one-bit coding and random low-rank projections can recover substantial performance as well. Finally, compression is not equivalent to training the final small adapter directly: adapters trained with additional rank recover more strongly when all are compressed to the same rank-1, one-bit endpoint. Full controls are reported in Appendix D.

5.2 Damage Is Structured Across Adapter Directions

The previous results suggest that the harmful and useful effects of the update are not uniformly distributed across its rank directions. We test this directly by decomposing corrupted adapters into singular directions and evaluating every contiguous window $[i, j)$ of the rank-16 update at full precision. Figure 5 reveals a consistent qualitative structure across model scales. Leading directions are strongly detrimental, an intermediate range contributes positively, and trailing directions have comparatively little effect. The location and width of these regimes vary across models, but the behavioral effect is clearly non-uniform across directions. This shows that the corruption is represented anisotropically within the learned update, and provides one mechanism by which compression can preferentially attenuate harmful behavior. The recovery controls above, however, show that directional selection alone does not account for the full effect.

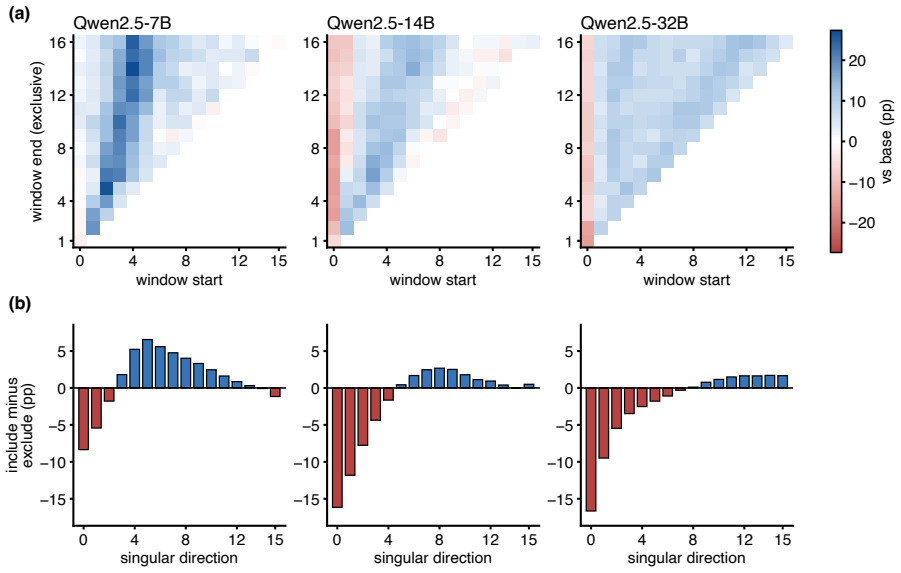

Figure 5: Corruption is structured across singular directions. (a) Accuracy relative to the frozen model for every contiguous window $[i, j)$ of singular directions retained at full precision. (b) Marginal contribution of each direction, computed by comparing windows that include it with those that do not.

The effect is not restricted to a single model size or task. Additional experiments reproduce recovery across Qwen2.5 scales and other corrupted settings. On 72 off-task probes, the compressed corrupted adapter also remains closer to the frozen model than a clean fine-tune with comparable in-domain performance. Scaling, matched-norm, training-rank, cross-model, and off-task controls are reported in Appendix D. Together, these results show that compression need not merely degrade a fine-tuned update. When supervision induces both useful and harmful effects, those effects can be distributed very differently across the learned update, and aggressive bottlenecks can suppress them unequally. In the next section, we ask whether related properties of a successful update also predict how much rate is required to preserve its behavior.

6 What Sets the Behavioral Rate?

The receiver-relative view of Section 4 suggests two complementary questions. First, can compression difficulty be anticipated from the data and the frozen receiver before training? Second, once an update has been learned, what property of that update tracks how aggressively it can be compressed? We address these questions in turn.

6.1 Behavioral Rate Is Receiver-Relative

We first test whether behavioral rate can be treated as an intrinsic property of the fine-tuning data. Figure 6(b) measures $R^\star(0.90)$ for the same three corpora across seven frozen receivers. The ordering changes with the receiver: XBRL tagging is the lowest-rate corpus for six receivers, but not for Qwen2.5-7B, where math is cheaper. Even closely related receivers can differ substantially: preserving 90% of the math gain requires 1.03 bits per value on Qwen2.5-Math-7B, compared with 0.41 on Qwen2.5-7B. Behavioral rate therefore depends on the dataset–receiver pair, rather than on the dataset alone.

Section 4 motivates a receiver-relative statistic computed before training from the corrections requested by the data. We summarize the corresponding correction-gradient spectrum by its log-determinant (Appendix B.3). Within a fixed receiver, this quantity ranks behavioral rate well: across 24 arms and seven task families, Spearman correlation with $R^\star(0.90)$ is $\rho = 0.92$ on Mistral-7B and $\rho = 0.59$ on Qwen2.5-7B, with a selection-adjusted permutation $p = 0.0003$ (Figure 6(a)).

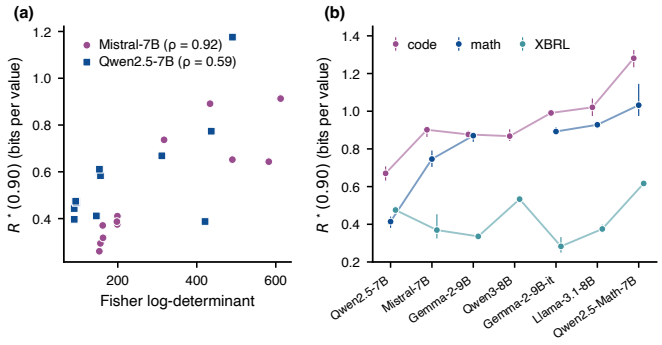

Figure 6: Behavioral rate is receiver-relative. (a) Correction-spectrum log-determinant against $R^\star(0.90)$ within the two development receivers. (b) $R^\star(0.90)$ for the same three corpora across seven receivers; bars span seeds. The ordering of corpora changes with the receiver.

This association does not transfer reliably across receivers. After fixing the statistic, correlations on additional receiver panels are -0.1 on Llama-3.1-8B, -0.4 on Qwen3-8B, and $+0.1$ on a fresh Qwen2.5-7B panel. Corpus-only quantities and quantities obtained by scoring the corpus with the frozen model likewise fail to consistently order receivers. We therefore view correction spectra as informative *within-receiver* descriptors of compression difficulty, rather than as universal predictors of behavioral rate. Predicting $R^\star$ from the data and receiver alone remains open.

6.2 Behavioral Rate Tracks Update Shrinkability

The previous analysis asks what can be inferred before the update is trained. Once the learned update is available, a simpler relationship emerges. Section 5 showed that aggressive compression often acts behaviorally like shrinking the update, although it is not exactly equivalent to scalar rescaling. This suggests that compression difficulty may depend on how far the learned update can be reduced before its acquired behavior is lost.

Let Δ denote the trained update and let $\widehat{\Delta}_{1\text{bit}}$ denote its one-bit coded version. We define

$$\gamma = \frac{\langle \widehat{\Delta}_{1\text{bit}}, \Delta \rangle}{\|\Delta\|_2^2}, \tag{6}$$

which measures how strongly one-bit coding shrinks the update along its original direction. Under our rank-pair codec, retaining a fraction p of the directions at one bit behaves approximately like scaling the uncompressed update by $p\gamma$. Along this compression path, the realized rate is also approximately p bits per original adapter value. Figure 7(a) tests this correspondence directly. Across three Llama-3.1-8B adapters, compressed operating points placed at their effective scale $p\gamma$ closely follow the behavioral response obtained by explicitly scaling the uncompressed update by α.

Let α_{90} denote the smallest scale at which the uncompressed update retains 90% of its original fine-tuning gain. Matching $p\gamma \approx \alpha_{90}$ then gives the codec-specific approximation

$$R^\star(0.90) \approx \frac{\alpha_{90}}{\gamma}. \tag{7}$$

Estimating α_{90} requires evaluating only five scales of the uncompressed update on 64 held-out examples, while γ is obtained from the adapter weights; no regression coefficient is fitted.

After fixing the rule on six adapters from two receivers, Figure 7(b) compares this estimate with measured rates on 42 checkpoints from five other receivers. It achieves an RMSE of 0.068 bits per value ($R^2 = 0.95$). On a broader archive of 58 adapters spanning 34 corpora, the RMSE is 0.105 bits per value. Additional baselines, disjoint-evaluation controls, and trained-update predictors are reported in Appendix C.4.

These experiments remain retrospective: although the rule was fixed independently of the reported receiver panels, their behavioral rates had already been measured when it was formulated. We therefore interpret Equation 7 as a retrospective empirical approximation rather than a prospectively validated prediction law.

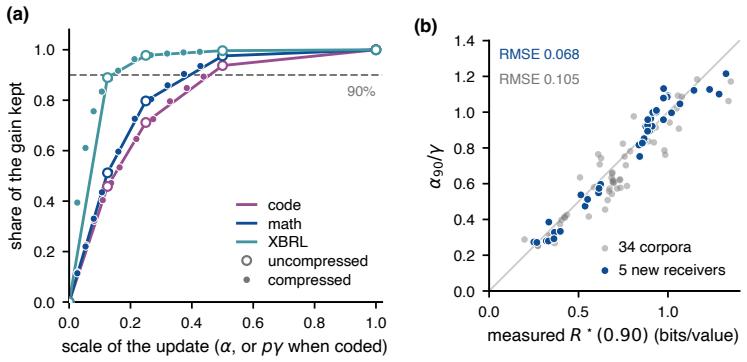

Figure 7: Behavioral rate tracks update shrinkability. (a) Retained gain for three Llama-3.1-8B adapters when the uncompressed update is scaled by α (lines), together with compressed operating points placed at their effective scale $p\gamma$ (dots). (b) α_{90}/γ against measured $R^\star(0.90)$ for 42 checkpoints on five receivers not used to fix the rule (blue) and 58 archived adapters spanning 34 corpora (grey). No regression coefficient is fitted.

The same response curve connects the two main empirical findings of the paper. In Section 5, moving back along the learned update can remove harmful behavior and recover performance; here, the distance one can move before losing useful behavior closely tracks the rate required to preserve the fine-tune.

7 Discussion and Limitations

Our experiments show that behavioral rate is receiver-relative: the same dataset can require different rates across frozen models, while parameter-space reconstruction error alone does not predict which compressed updates preserve behavior. At the same time, the success of the shrinkability approximation indicates that, once an update has been learned, much of its compression difficulty can be summarized by the behavioral response obtained when moving back along that update.

Several limitations qualify these conclusions. First, $R^\star$ is an operational, codec-dependent rate rather than an information-theoretic optimum; different adapter parameterizations or codecs may induce different measured rates. Second, the correction-spectrum statistic is only a proxy for the full receiver-relative geometry and does not transfer reliably across receivers. Third, the shrinkability rule is supported by broad but retrospective evaluation, and remains to be validated prospectively on rates not observed when the rule is formulated. Finally, our recovery experiments use deliberately corrupted supervision to create a controlled mixture of useful and harmful effects. Whether the same structure governs naturally occurring fine-tuning errors remains an open question.

8 Conclusion

We introduced the behavioral rate of fine-tuning: the smallest serialized adapter rate that preserves a target fraction of the behavior acquired by fine-tuning. Across models and tasks, this rate is not determined by parameter count, weight reconstruction error, or the dataset alone, but depends on the interaction between the learned update and its frozen receiver.

Compression also exposes structure within the update itself. Under corrupted supervision, useful and harmful effects can occupy components with sharply different behavioral consequences, allowing aggressive bottlenecks to recover performance that the full update destroys. More generally, once an update has been trained, the rate required to preserve it closely tracks how far that update can be shrunk before its acquired behavior is lost.

These results separate three notions that are often conflated: how much data are presented during fine-tuning, how large the resulting parameter update is, and how many bits of that update are behaviorally necessary for a particular receiver. Adapter compression therefore serves not only as a storage mechanism, but as an operational probe of which parts of a learned update are needed to preserve a specified behavior.

AI USE STATEMENT

Generative AI tools were used for language editing and debugging support. All AI-assisted text were reviewed by the authors. The authors take full responsibility for the paper's claims, results, citations, and final content.

REPRODUCIBILITY STATEMENT

We provide the experimental details required to reproduce our results in the appendices. Appendix A specifies the models, datasets, fine-tuning configurations, compression procedures, serialization protocol, and evaluation setup used to measure behavioral rate. In particular, reported rates correspond to realized serialized adapter sizes after including the metadata required for decoding and reloading the adapter. Theoretical assumptions and derivations for the receiver-relative rate-distortion view are given in Appendix B, while additional predictor, compression, recovery, and generalization controls are reported in the remaining appendices. We report multiple seeds where indicated in the text and figure captions, and distinguish explicitly between multi-seed results and single-checkpoint diagnostic analyses. Code for training, compression, evaluation, and figure generation will be released upon publication.

REFERENCES

Armen Aghajanyan, Sonal Gupta, and Luke Zettlemoyer. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 7319–7328, Online, 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.568. URL https://aclanthology.org/2021.acl-long.568/.

Sanjeev Arora, Rong Ge, Behnam Neyshabur, and Yi Zhang. Stronger generalization bounds for deep nets via a compression approach. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 254–263. PMLR, 2018. URL https://proceedings.mlr.press/v80/arora18b.html.

Klaudia Bałazy, Mohammadreza Banaei, Karl Aberer, and Jacek Tabor. Lora-xs: Low-rank adaptation with extremely small number of parameters. In *ECAI 2025: 28th European Conference on Artificial Intelligence 25-30 October 2025, Bologna, Italy-Including 14th Conference on Prestigious Applications of Intelligent Systems (PAIS 2025) Proceedings*, pp. 3194–3201, 2025.

Valérie Castin, Kimia Nadjahi, Pierre Ablin, and Gabriel Peyré. Balanced loRA: Removing parameter invariance to accelerate convergence. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=kHInw3cjCP.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient finetuning of quantized LLMs. In *Advances in Neural Information Processing Systems*, volume 36, pp. 10088–10115, 2023. doi: 10.52202/075280-0441. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/1feb87871436031bdc0f2beaa62a049b-Abstract.html.

Subhabrata Dutta, Joykirat Singh, Soumen Chakrabarti, and Tanmoy Chakraborty. How to think step-by-step: A mechanistic understanding of chain-of-thought reasoning. *Transactions on Machine Learning Research*, 2024. URL https://mlanthology.org/tmlr/2024/dutta2024tmlr-think/.

Erwan Fagnou, Paul Caillon, Blaise Delattre, and Alexandre Allauzen. Chain and causal attention for efficient entity tracking. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 13174–13188, Miami, Florida, USA, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.731. URL https://aclanthology.org/2024.emnlp-main.731/.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. GPTQ: Accurate post-training quantization for generative pre-trained transformers. In *The Eleventh International Conference on Learning Representations*. OpenReview.net, 2023. URL https://openreview.net/forum?id=tcbBPnfwxS.

Soufiane Hayou, Nikhil Ghosh, and Bin Yu. Lora+: efficient low rank adaptation of large models. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations*. OpenReview.net, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Chengsong Huang, Qian Liu, Bill Yuchen Lin, Tianyu Pang, Chao Du, and Min Lin. Lorahub: Efficient cross-task generalization via dynamic loRA composition. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=TrloAXEJ2B.

Ting Jiang, Shaohan Huang, Shengyue Luo, Zihan Zhang, Haizhen Huang, Furu Wei, Weiwei Deng, Feng Sun, Qi Zhang, Deqing Wang, et al. Mora: High-rank updating for parameter-efficient fine-tuning. *arXiv preprint arXiv:2405.12130*, 2024.

Dawid Kopiczko, Tijmen Blankevoort, and Yuki Asano. Vera: Vector-based random matrix adaptation. In *International Conference on Learning Representations*, pp. 6815–6835, 2024.

Zeju Li, Jianyuan Zhong, Ziyang Zheng, Xiangyu Wen, Zhijian Xu, Yingying Cheng, Fan Zhang, and Qiang Xu. Making slow thinking faster: Compressing LLM chain-of-thought via step entropy. In *The Fourteenth International Conference on Learning Representations*. OpenReview.net, 2026. URL https://openreview.net/forum?id=cGLqQfS5wH.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. AWQ: Activation-aware weight quantization for on-device LLM compression and acceleration. In *Proceedings of Machine Learning and Systems*, volume 6, pp. 87–100, 2024. URL https://proceedings.mlsys.org/paper_files/paper/2024/hash/42a452cbafa9dd64e9ba4aa95cc1ef21-Abstract-Conference.html.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. DoRA: Weight-decomposed low-rank adaptation. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 32100–32121. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/liu24bn.html.

Zechun Liu, Changsheng Zhao, Hanxian Huang, Sijia Chen, Jing Zhang, Jiawei Zhao, Scott Roy, Lisa Jin, Yunyang Xiong, Yangyang Shi, Lin Xiao, Yuandong Tian, Bilge Soran, Raghuraman Krishnamoorthi, Tijmen Blankevoort, and Vikas Chandra. ParetoQ: Improving scaling laws in extremely low-bit LLM quantization. In *Advances in Neural Information Processing Systems*, volume 38, pp. 101395–101420, 2025. doi: 10.52202/085713-3055. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/83b17fb3369b1effa97ca5409526b02e-Abstract-Conference.html.

Sanae Lotfi, Marc Finzi, Sanyam Kapoor, Andres Potapczynski, Micah Goldblum, and Andrew Gordon Wilson. PAC-bayes compression bounds so tight that they can explain generalization. In *Advances in Neural Information Processing Systems*, volume 35, pp. 31459–31473, 2022. doi: 10.52202/068431-2281. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/cbeec55c50c3367024bafab2438a021b-Abstract-Conference.html.

Fanxu Meng, Zhaohui Wang, and Muhan Zhang. Pissa: Principal singular values and singular vectors adaptation of large language models. *Advances in Neural Information Processing Systems*, 37:121038–121072, 2024.

Amir Reza Mirzaei, Yuqiao Wen, Yanshuai Cao, and Lili Mou. LoRAQuant: Mixed-precision quantization of LoRA to ultra-low bits, 2025. URL https://arxiv.org/abs/2510.26690.

Chen Qian, Dongrui Liu, Haochen Wen, Zhen Bai, Yong Liu, and Jing Shao. Demystifying reasoning dynamics with mutual information: Thinking tokens are information peaks in LLM reasoning. In *Advances in Neural Information Processing Systems*, volume 38, pp. 14065–14104, 2025. doi: 10.52202/085713-0421. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/12867c8f1888d97d4e9f95f03c0f3ec9-Abstract-Conference.html.

Reece Shuttleworth, Jacob Andreas, Antonio Torralba, and Pratyusha Sharma. LoRA vs full fine-tuning: An illusion of equivalence. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/ff541950d1e885af90f523571564a401-Abstract-Conference.html.

Kaizhen Tan, Heqing Du, and Yang Feng. How many bits can an adapter write? measuring the capacity and memorization of parameter-efficient fine-tuning, 2026. URL https://arxiv.org/abs/2607.21351.

MTCAJ Thomas and A Thomas Joy. *Elements of information theory*. Wiley-Interscience, 2006.

Zailong Tian, Yanzhe Chen, Zhuoheng Han, and Lizi Liao. Spectral surgery: Training-free refinement of LoRA via gradient-guided singular value reweighting, 2026. URL https://arxiv.org/abs/2603.03995.

Einar Urdshals, Edmund Lau, Jesse Hoogland, Stan van Wingerden, and Daniel Murfet. Compressibility measures complexity: Minimum description length meets singular learning theory, 2025. URL https://arxiv.org/abs/2510.12077.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 24824–24837, 2022. doi: 10.52202/068431-1800. URL https://proceedings.neurips.cc/paper/2022/hash/9d5609613524ecf4f15af0f7b31abca4-Abstract-Conference.html.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. SmoothQuant: Accurate and efficient post-training quantization for large language models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 38087–38099. PMLR, 2023. URL https://proceedings.mlr.press/v202/xiao23c.html.

Yukang Yang, Declan Iain Campbell, Kaixuan Huang, Mengdi Wang, Jonathan D. Cohen, and Taylor Whittington Webb. Emergent symbolic mechanisms support abstract reasoning in large language models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 70515–70549. PMLR, 2025. URL https://proceedings.mlr.press/v267/yang25c.html.

Zhewei Yao, Reza Yazdani Aminabadi, Minjia Zhang, Xiaoxia Wu, Conglong Li, and Yuxiong He. ZeroQuant: Efficient and affordable post-training quantization for large-scale transformers. In *Advances in Neural Information Processing Systems*, volume 35, pp. 27168–27183, 2022. doi: 10.52202/068431-1970. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/adf7fa39d65e2983d724ff7da57f00ac-Abstract-Conference.html.

Sean I. Young. Radio: Rate–distortion optimization for large language model compression. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 72819–72836. PMLR, 2025. URL https://proceedings.mlr.press/v267/young25b.html.

Longteng Zhang, Lin Zhang, Shaohuai Shi, Xiaowen Chu, and Bo Li. Lora-fa: Memory-efficient low-rank adaptation for large language models fine-tuning. *arXiv e-prints*, pp. arXiv–2308, 2023a.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. Adaptive budget allocation for parameter-efficient fine-tuning. In *The Eleventh International Conference on Learning Representations*, 2023b. URL https://openreview.net/forum?id=lq62uWRJjiY.

Hongyun Zhou, Xiangyu Lu, Wang Xu, Conghui Zhu, Tiejun Zhao, and Muyun Yang. Lora-drop: Efficient lora parameter pruning based on output evaluation. In *Proceedings of the 31st international conference on computational linguistics*, pp. 5530–5543, 2025.

Jiacheng Zhu, Kristjan Greenewald, Kimia Nadjahi, Haitz S áez De Ocáriz Borde, Rickard Brüel Gabrielsson, Leshem Choshen, Marzyeh Ghassemi, Mikhail Yurochkin, and Justin Solomon. Asymmetry in low-rank adapters of foundation models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 62369–62385. PMLR, 2024. URL https://proceedings.mlr.press/v235/zhu24c.html.

A Experimental Details

A.1 Models, training and evaluation

Models. All frozen models are base (pretrained) checkpoints loaded in 4-bit NF4 with double quantization (bitsandbytes), except Mistral-7B-Instruct-v0.3 and Gemma-2-9B-it, which are instruction-tuned and prompted through their chat template. We use Qwen2.5 at 0.5B, 1.5B, 3B, 7B, 14B and 32B parameters, Qwen3-8B-Base, Llama-3.1-8B, Mistral-7B-v0.1, Mistral-7B-Instruct-v0.3 and Gemma-2-9B-it. Most experiments use Qwen2.5-7B. Headline comparisons use three seeds unless stated otherwise; some diagnostic analyses use a single checkpoint, as indicated below and in the corresponding figure captions.

Adapters and training. Unless stated otherwise, adapters are LoRA of rank 16 with scale $\alpha = 32$ ($\alpha/r = 2$) and no dropout, on the attention and MLP projections of every layer (q, k, v, o, gate, up and down). We train with AdamW at learning rate 2×10^{-4}, $\beta = (0.9, 0.95)$, no weight decay, 3% linear warmup, gradient clipping at 1.0, effective batch size 16 (micro-batches of 4) and maximum sequence length 1,024 tokens, with the loss on the whole response. The corruption experiments of Section 5 train for one epoch on 32,000 MetaMathQA rows (2,000 updates), and evaluate the validation loss every 200 updates. The adapters of Figure 3 and of the 14B and 32B studies keep the checkpoint with the lowest validation loss; the Qwen2.5-7B window-search adapters keep the final one. Seeds are 11, 22 and 33 where three are reported, and 11 otherwise.

Evaluation. Decoding is greedy. GSM8K-style tasks allow 512 new tokens and are scored by exact match of the final answer; MATH-500 allows 1,024 tokens and is checked with Math-Verify; multiple-choice probes ask for a letter within 32 tokens. GSM8K uses all 1,319 test problems where stated and a fixed subset of 500 otherwise. Intervals are 95% paired bootstrap intervals over questions (2,000 to 10,000 draws), clustered by template for GSM-Symbolic.

Selecting spectral filters. For the window searches of Section 5, the corrupted update is decomposed by SVD into its 16 singular directions. Each candidate keeps a set of directions at full precision or codes them. Candidates are scored on 128 MetaMathQA rows reserved from training (every seed problem of the training slice is excluded), the finalists are re-scored on 128 further reserved rows, and only the selected filter is scored on the test sets.

A.2 Corruptions

Table 1 lists the corruptions and how many configurations use each. The main text uses the rationale permutation; the XBRL rows of Table 8 use the response mismatch.

Table 1: Corruptions used, with the number of configurations that use each.

	corruption	configs
rationale permuted	each worked explanation/CoT moved to a different problem; question and answer line kept	19
response mismatched	the whole response dealt to a different item, for tasks without an answer line	10
rationale steps shuffled	steps of the explanation reordered; steps, arithmetic and answer kept	2
arithmetic corrupted	half the computed results moved by up to a third, and the answer follows the wrong chain	2
labels scrambled / flipped	each multiple-choice letter, or short text label, moved to another of its choices	2 + 2

A.3 Codecs and rate accounting

Rates are the size of the serialized file, decoding metadata included, divided by the number of scalar values in the uncompressed adapter. The uniform code is a zero-free midrise quantizer fitted per row; the exact-zero control is a midtread quantizer scaled by the row's absolute maximum. LoRAQuant

runs in four configurations: 2@0.8, 2@0.9, 3@0.8 and 3@0.9. Below one bit per value we remove paired rank directions before coding. Table 2 lists the rates each code realises.

Table 2: Realised rate and retained GSM8K gain per codec (Mistral-7B, 3 seeds, raw gain 61.5 points).

codec	bits / value	gain kept (%)	relative RMSE
midtread, 2 levels (exact zero)	0.63	61.1	0.881
adaptive, may drop rows, target 1	0.98	2.6	0.537
uniform zero-free, 1 bit	1.02	85.7	0.563
adaptive, keeps rows, target 1.1	1.03	85.9	0.561
LoRAQuant 2@0.8	1.59	95.4	–
uniform zero-free, 2 bits	1.97	98.6	0.314
LoRAQuant 3@0.9	2.69	99.1	–
uniform zero-free, 3 bits	2.84	99.8	0.184
uniform zero-free, 4 bits	3.39	100.7	0.112
uniform zero-free, 8 bits	7.28	100.1	0.008

The compressed adapters of Figure 3 use the uniform code below one bit: half of the rank directions are removed and the rest are coded at 1 bit, 0.52 bits per value overall and 2.63 to 2.74 MB depending on the model. The rank-1 codes of Figure 4 and Table 8 keep the leading singular direction of the update; a rank-1 adapter at 1 bit is a 0.40 MB file for Qwen2.5-7B.

A.4 Candidates for the behavioral rate

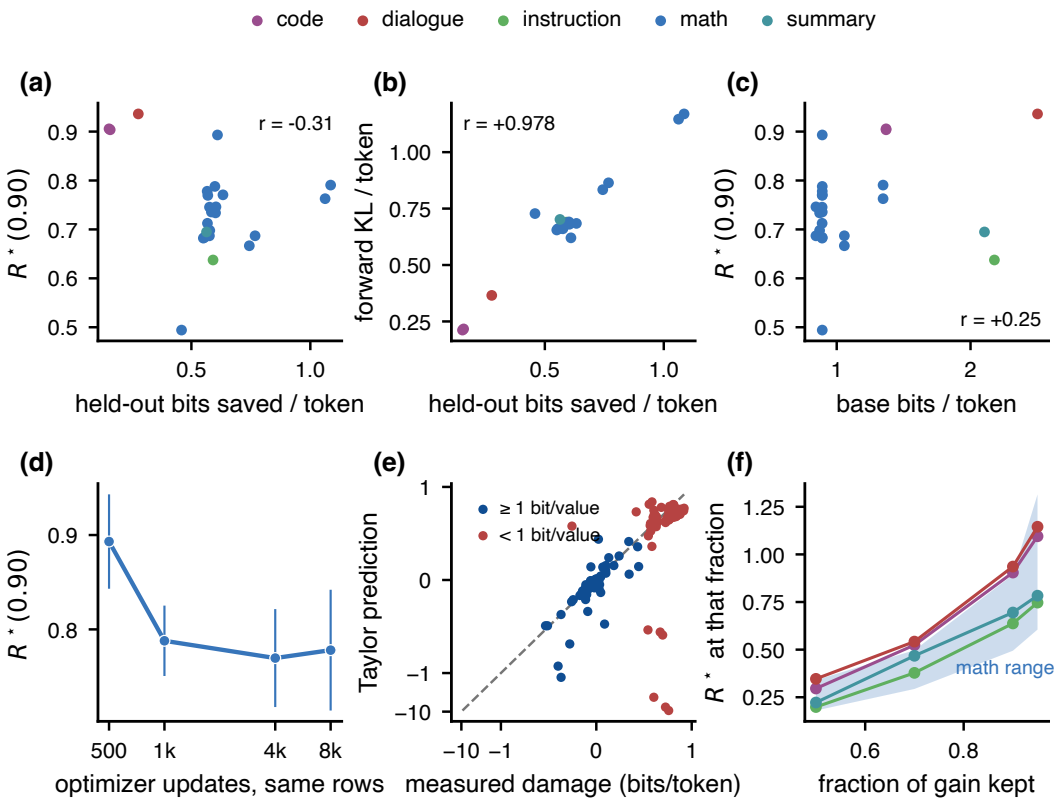

Figure 8: Candidates for predictors of $R^\star(0.90)$, one point per arm (seed means; 24 arms, Mistral-7B). (a) Held-out bits saved: $r = -0.31$. (b) Forward KL tracks bits saved at $r = +0.978$, so it is not a second measurement. (c) Base bits per token on the corpus: $r = +0.25$. (d) Sixteen times the optimizer updates on the same rows lowers $R^\star$; bars are seed s.d. (e) Second-order Taylor prediction of the damage against measured damage, symmetric log axes: close above one bit per value, wrong in sign and size below it. (f) $R^\star$ at other retention thresholds: code and dialogue sit at the top of the math range at every threshold.

Figure 8 shows the candidates that fail on the 24 Mistral-7B development arms: the size of the behavioral gain, distribution shift, base-model surprisal, optimizer depth, a second-order Taylor model of the damage, and the choice of retention threshold.

The audit re-scored 268 cells from 67 arms on four receivers with 46 candidate measures, after six repairs to the measurement pipeline; nothing was retrained. The repairs lower the held-out-receiver error from 0.239 (receiver mean) to 0.152, but the best predictor is text cross-row redundancy, two zlib calls that never look at the model (Figure 9). A measure of the corpus alone assigns one value to a corpus on every receiver, so it cannot produce the corpus–receiver reorderings of Section 6.

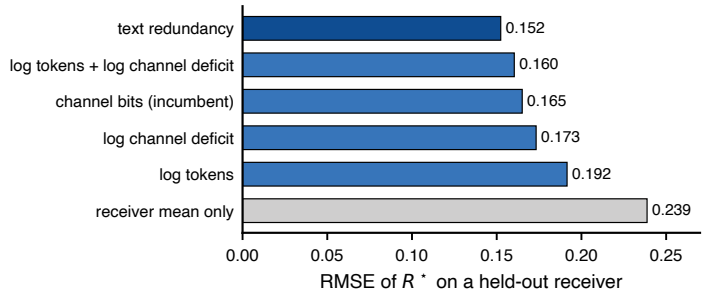

Figure 9: Leave-one-receiver-out RMSE of $R^\star(0.90)$ for the rate-law models, four receivers.

Table 3 lists every test of a correction statistic on a receiver outside the development pair. Only the locked rows were chosen before seeing that receiver's $R^\star$; the rest are shown for completeness.

Table 3: Correction statistics on new receivers. ρ: Spearman between measure and $R^\star(0.90)$ over natural corpora (seed means). RMSE: absolute prediction from the development fit, against the base-code-length fit. Gate: $\rho \geq 0.70$ and lower RMSE than base code length.

receiver	measure	locked	corpora	ρ	RMSE	base RMSE
Llama-3.1-8B	Fisher log-volume	yes	5	-0.10	0.357	0.327
Qwen3-8B	Fisher effective rank	yes	4	-0.40	0.408	0.306
Qwen2.5-7B, new panel	coherent energy / entropy	yes	5	0.10	0.296	0.241
Llama-3.1-8B	coherent energy / entropy	no	5	0.70	0.132	0.327
Qwen3-8B	coherent energy / entropy	no	4	0.80	0.163	0.306
Qwen3-8B	coherent energy	no	4	1.00	0.206	0.306
Llama-3.1-8B	Fisher log-determinant	no	5	-0.50	0.483	0.327
Qwen3-8B	Fisher log-determinant	no	4	0.40	0.461	0.306
Qwen2.5-7B, new panel	Fisher log-determinant	no	5	0.70	0.234	0.241

A.5 Known payload

Sixteen single-token labels are assigned to (family, item) pairs; 128 of 192 mappings are taught, and only the rule that generates labels changes between conditions, so the number of independent 4-bit draws behind them is known. This separates the information the data carries relative to the base model, the weights installed, and the serialized rate by construction.

We additionally estimate the description length of this synthetic source with a conditional prequential code. After allowing later blocks to fall back to the frozen model when this gives the shorter code, the estimate correctly orders the 4-, 64-, 128-, 256-, and 512-bit source conditions and remains within 1.1-3.5 times the known source description length. The corresponding behavioral rates do not inherit a simple ordering by source description length. Thus, even when the amount of source information is controlled and independently calibrated, it is not sufficient to determine $R^\star(0.90)$.

A.6 All generalization probes

Table 4 lists the 72 probes behind Figure 13.

Table 4: All 72 generalization probes. Accuracy on all rows; the last two columns are, respectively, the held-out half as a fraction of base, where † marks probes the clean adapter hurts on the first half.

probe	tier	base	clean	compr.	clean/base	compr./base
gsm8k	0	0.756	0.810	0.814	1.13	1.13
asdiv	1	0.885	0.925	0.902	1.07	1.03
gsm symbolic	1	0.762	0.830	0.790	1.10	1.09
svamp	1	0.790	0.847	0.890	1.08	1.15
aqua rat†	2	0.402	0.059	0.394	0.02	1.00
math500†	2	0.585	0.495	0.590	0.89	1.07
MMLU abstract algebra†	2	0.470	0.320	0.510	0.70	1.15
MMLU college mathematics†	2	0.510	0.170	0.470	0.34	0.90
MMLU elementary mathematics†	2	0.701	0.188	0.693	0.28	0.99
MMLU high school mathematics†	2	0.556	0.011	0.419	0.01	0.69
MMLU high school statistics†	2	0.690	0.449	0.671	0.63	0.97
MMLU anatomy	3	0.681	0.696	0.674	1.00	0.98
MMLU astronomy†	3	0.816	0.796	0.829	1.02	1.02
MMLU college biology	3	0.840	0.806	0.847	0.92	1.00
MMLU college chemistry†	3	0.480	0.390	0.460	0.81	1.04
MMLU college computer science†	3	0.650	0.550	0.640	0.84	1.03
MMLU college physics†	3	0.451	0.284	0.431	0.48	0.91
MMLU computer security	3	0.810	0.840	0.810	1.05	1.00
MMLU conceptual physics†	3	0.711	0.634	0.719	0.86	0.99
MMLU electrical engineering†	3	0.703	0.648	0.697	0.92	1.02
MMLU high school biology	3	0.839	0.816	0.829	0.98	0.98
MMLU high school chemistry†	3	0.665	0.463	0.645	0.71	0.99
MMLU high school computer science†	3	0.800	0.720	0.780	0.87	0.92
MMLU high school physics†	3	0.523	0.272	0.543	0.55	1.05
MMLU machine learning†	3	0.491	0.411	0.509	0.73	1.08
arc challenge†	3	0.880	0.825	0.887	0.94	1.02
arc easy	3	0.955	0.955	0.958	0.98	0.99
openbookqa	3	0.858	0.848	0.860	0.98	1.01
qasc	3	0.823	0.807	0.830	0.98	1.00
commonsenseqa	4	0.815	0.818	0.830	1.01	1.02
hellaswag†	4	0.855	0.710	0.845	0.76	0.99
winogrande	4	0.650	0.637	0.632	0.97	0.96
MMLU business ethics	4	0.750	0.720	0.740	0.91	0.94
MMLU clinical knowledge	4	0.770	0.751	0.766	0.94	1.00
MMLU college medicine†	4	0.611	0.558	0.621	0.93	1.04
MMLU econometrics	4	0.561	0.509	0.579	0.85	1.06
MMLU global facts	4	0.440	0.470	0.410	0.96	0.96
MMLU high school geography†	4	0.889	0.859	0.884	1.00	1.01
MMLU high school government and politics	4	0.938	0.912	0.938	0.96	1.00
MMLU high school macroeconomics	4	0.772	0.751	0.764	0.99	0.99
MMLU high school microeconomics	4	0.857	0.853	0.840	1.00	0.99
MMLU high school psychology	4	0.873	0.870	0.873	0.98	0.99
MMLU human aging	4	0.731	0.722	0.758	1.01	1.02
MMLU human sexuality†	4	0.832	0.771	0.809	0.91	0.96
MMLU management	4	0.922	0.903	0.903	0.96	0.96
MMLU marketing	4	0.919	0.919	0.906	1.00	0.97
MMLU medical genetics	4	0.820	0.780	0.790	0.88	0.90
MMLU miscellaneous	4	0.850	0.840	0.860	0.98	1.01
MMLU nutrition	4	0.765	0.758	0.758	1.00	0.98
MMLU professional accounting†	4	0.525	0.482	0.535	0.92	1.04
MMLU professional medicine	4	0.746	0.739	0.746	0.99	1.03
MMLU professional psychology	4	0.735	0.715	0.740	0.98	1.03
MMLU public relations†	4	0.682	0.664	0.682	1.00	1.00
MMLU security studies	4	0.763	0.771	0.771	1.01	0.99
MMLU sociology	4	0.856	0.861	0.861	1.00	1.02
MMLU us foreign policy†	4	0.940	0.870	0.900	0.96	1.00
MMLU virology†	4	0.554	0.512	0.530	0.93	0.98

probe	tier	base	clean	compr.	clean/base	compr./base
race high	4	0.882	0.875	0.882	1.00	0.99
humaneval	5	0.421	0.415	0.427	0.86	1.00
MMLU formal logic†	5	0.532	0.429	0.508	0.84	1.06
MMLU high school european history	5	0.855	0.812	0.830	0.95	0.99
MMLU high school us history	5	0.848	0.858	0.853	1.02	1.01
MMLU high school world history	5	0.873	0.882	0.869	1.01	1.00
MMLU international law	5	0.810	0.793	0.818	0.96	1.00
MMLU jurisprudence	5	0.824	0.806	0.815	1.00	1.00
MMLU logical fallacies	5	0.828	0.804	0.828	0.96	0.99
MMLU moral disputes	5	0.780	0.757	0.769	0.96	0.99
MMLU moral scenarios	5	0.292	0.395	0.237	1.26	0.76
MMLU philosophy	5	0.740	0.720	0.746	0.98	1.04
MMLU prehistory	5	0.809	0.784	0.796	0.98	0.97
MMLU professional law	5	0.490	0.475	0.460	0.97	0.94
MMLU world religions	5	0.860	0.877	0.860	1.01	0.99

B Details on the Receiver-Relative Rate-Distortion View

This appendix provides additional details for the local arguments used in Section 4. The purpose of these derivations is not to establish an exact rate-distortion theory of LoRA, but to make explicit the assumptions under which the geometric quantities introduced in the main text arise.

B.1 Local Behavioral Distortion

Let A denote a trained adapter and let $\hat{A} = A + e$ be a perturbed version, where e is the error induced by compression. For a differentiable evaluation loss $\mathcal{L}_U$, a Taylor expansion around A gives

$$\mathcal{L}_U(M \oplus (A + e)) - \mathcal{L}_U(M \oplus A) = g_U^\top e + \frac{1}{2} e^\top H_U e + o(\|e\|^2), \tag{8}$$

where

$$g_U = \nabla_A \mathcal{L}_U(M \oplus A), \qquad H_U = \nabla_A^2 \mathcal{L}_U(M \oplus A).$$

The quadratic approximation used in the main text therefore requires the first-order contribution to be negligible for the perturbations considered. This is exact at a stationary point of the evaluation objective and can also be a useful local approximation when $|g_U^\top e|$ is small compared with the quadratic term.

The exact Hessian need not be positive semidefinite. When interpreting H_U as a local distortion metric, one may instead use a positive-semidefinite curvature approximation, such as a generalized Gauss–Newton matrix. The relevant point for our argument is not the particular approximation, but that behavioral distortion generally defines a non-isotropic metric over parameter perturbations.

B.2 When Does Weight Error Recover Functional Distortion?

Squared weight reconstruction error corresponds to the Euclidean quadratic form

$$D_{\text{weight}}(e) = e^\top e,$$

whereas the local functional criterion is

$$D_{\text{func}}(e) = e^\top H_U e.$$

Let $\mathcal{S}$ denote the subspace containing the compression errors under consideration. If

$$H_U|_{\mathcal{S}} = \lambda I, \qquad \lambda > 0, \tag{9}$$

then

$$D_{\text{func}}(e) = \lambda D_{\text{weight}}(e)$$

for every $e \in \mathcal{S}$, and the two criteria induce identical rankings of compression errors.

Conversely, if all perturbations with equal Euclidean norm have equal functional cost, the restriction of H_U to $\mathcal{S}$ must be proportional to the identity. To see this, diagonalize the symmetric quadratic form on $\mathcal{S}$. If two eigenvalues differ, perturbations of equal norm along their associated eigenvectors have different functional costs.

Thus, ordinary reconstruction error is sufficient only under an isotropic functional geometry on the errors produced by the codec. The RMSE counterexample and layer-location controls in the main text provide empirical evidence against this approximation in our setting.

B.3 Correction Statistics and the Learned Update

At the frozen parameters θ_0, let

$$g_i = \nabla_\theta \ell_i(\theta_0), \qquad \mu = \mathbb{E}[g_i], \qquad J = \mathbb{E}[g_i g_i^\top].$$

The centered correction covariance is

$$C_g = \mathbb{E}\big[(g_i - \mu)(g_i - \mu)^\top\big] = J - \mu\mu^\top. \tag{10}$$

A quadratic approximation to the average training objective has the form

$$\mathcal{L}_T(\theta_0 + \Delta\theta) \approx \mathcal{L}_T(\theta_0) + \mu^\top \Delta\theta + \frac{1}{2}\Delta\theta^\top H_T \Delta\theta.$$

When H_T is invertible on the relevant subspace, its minimizer is

$$\Delta\theta^\star = -H_T^{-1}\mu. \tag{11}$$

Throughout, H_T^{-1} denotes the inverse restricted to the relevant subspace. When H_T is singular, the same local expressions can instead be read using a pseudoinverse or a suitably damped inverse.

Equation 11 makes clear that J does not determine the deterministic local update. For example, in one dimension, a dataset for which every example has gradient $+1$ and a dataset containing equal numbers of gradients $+1$ and -1 both satisfy

$$J = 1,$$

but their means are respectively $\mu = 1$ and $\mu = 0$. Their local optimal updates are therefore different despite having identical second moments.

We consequently interpret J as a descriptor of the geometry of the corrections requested by the data, rather than as an estimate of the adapter update itself. Its potential relevance to compression comes from the directional diversity that it captures in addition to the mean correction.

B.4 From Correction Statistics to a Receiver-Relative Spectrum

To motivate a link between correction statistics and compressibility, consider the following local linear model. Let G be a random correction gradient and suppose that it induces a local response

$$\Delta\theta = -H_T^{-1}G. \tag{12}$$

This should be understood as an idealized response model, not as a description of an SGD trajectory.

Under Equation 12, the mean update is

$$\mathbb{E}[\Delta\theta] = -H_T^{-1}\mu,$$

its covariance is

$$\mathrm{Cov}(\Delta\theta) = H_T^{-1} C_g H_T^{-1}, \tag{13}$$

and its second moment is

$$\mathbb{E}[\Delta\theta\Delta\theta^\top] = H_T^{-1} J H_T^{-1}. \tag{14}$$

Behavioral distortion introduces a second geometry through H_U. Applying it to the centered update covariance in Equation 13 gives

$$C_{\mathrm{rel}} = H_U^{1/2} H_T^{-1} C_g H_T^{-1} H_U^{1/2}. \tag{15}$$

This is the quantity that corresponds most directly to the Gaussian rate-distortion argument below: it is the covariance of the local update distribution expressed in behaviorally weighted coordinates.

For prediction, however, we are also interested in the systematic correction encoded by μ. Since

$$J = C_g + \mu\mu^\top,$$

retaining the uncentered second moment motivates the related geometry

$$\widetilde{C}_{\mathrm{rel}} = H_U^{1/2} H_T^{-1} J H_T^{-1} H_U^{1/2}. \tag{16}$$

Unlike C_{rel}, $\widetilde{C}_{\mathrm{rel}}$ is not itself a Gaussian source covariance. It is a second-moment extension that retains both the mean correction and its directional variability, and is introduced as a candidate geometry for predicting behavioral rate.

When training and evaluation share the same local geometry, $H_T = H_U = H$, this reduces to

$$\widetilde{C}_{\mathrm{rel}} = H^{-1/2} J H^{-1/2}. \tag{17}$$

This whitening has a simple interpretation. A correction direction is discounted when the training geometry makes it easy to realize, and amplified when it corresponds to a behaviorally sensitive direction of the receiver.

B.5 Gaussian Rate-Distortion Motivation

The spectral summaries used in the main text are motivated by the classical rate-distortion function of a Gaussian source under quadratic distortion (Thomas & Joy, 2006, Ch. 10).

Let

$$X \sim \mathcal{N}(0, \Sigma)$$

and consider a quadratic distortion

$$d(x, \hat{x}) = (x - \hat{x})^\top H (x - \hat{x}), \qquad H \succeq 0.$$

After the change of coordinates

$$Z = H^{1/2} X,$$

the problem becomes ordinary squared-error coding for a Gaussian variable with covariance

$$C = H^{1/2} \Sigma H^{1/2}.$$

Let $\lambda_1, \ldots, \lambda_d$ denote the eigenvalues of C. The Gaussian rate-distortion function is then given by reverse water filling:

$$R(D) = \frac{1}{2} \sum_j \left[\log_2 \frac{\lambda_j}{\nu} \right]_+, \tag{18}$$

where ν is chosen so that

$$D = \sum_j \min(\lambda_j, \nu).$$

Equation 18 highlights two properties relevant to our setting. Rate increases when the distortion-weighted variance becomes larger, but also when it is spread across more active directions. This motivates considering both scale-sensitive and dimensionality-sensitive summaries of correction spectra.

A convenient smooth summary is

$$I_{\mathrm{rel}}(\alpha) = \frac{1}{2} \log\det\left(I + \alpha C_{\mathrm{rel}}\right) = \frac{1}{2} \sum_j \log(1 + \alpha \lambda_j). \tag{19}$$

Unlike Equation 18, this expression does not implement an explicit distortion threshold and should not be interpreted as the rate-distortion function of our adapters. We use it only as a smooth statistic that increases with both the magnitude and dimensionality of the relevant spectrum.

Table 5: Alternative quantities do not determine $R^\star(0.90)$. Results cover 81 adapters over five corpora (MetaMathQA, Magicoder, XSum, hh-rlhf, Alpaca) on Mistral-7B and Qwen2.5-7B, together with the broader rate-law audit described in Appendix A.

candidate	result
distinct rows	ranks four conditions correctly within a corpus, but reverses across corpora
behavioral change (bits saved)	slope 1.90 within a 0.16-wide window; 0.12 on answer rewrites and approximately flat under optimizer-budget changes
optimizer depth	increasing the number of updates by $16\times$ on the same rows changes bits saved by 0.04 while $R^\star$ decreases
base surprisal on the corpus	$r = +0.25$ with $R^\star$
weight reconstruction error	approximately constant to within 1% across all five corpora at a fixed rate
local curvature ($g^\top\delta$, $\frac{1}{2}\delta^\top H\delta$)	the second-order term adds no ranking power, and the local expansion breaks down below one bit per value

Our empirical correction spectrum is a substantially coarser approximation. It is constructed from projected output-head correction gradients rather than from gradients in the full LoRA parameter space, and we do not explicitly estimate H_T or H_U in the experiments reported here. Consequently,

$$\log\det(I + \alpha J)$$

should be interpreted as a tractable proxy for the second-moment geometry $\widetilde{C}_{\mathrm{rel}}$, which is itself an extension of the covariance geometry directly motivated by Gaussian rate-distortion theory. It is therefore neither a direct estimate of Equation 15 nor an information-theoretic rate.

C Additional Analyses of Behavioral Rate

This appendix collects additional controls supporting the behavioral-rate analysis in Sections 3 and 6. We first summarize alternative quantities that fail to determine $R^\star(0.90)$, then examine the effect of where a fixed compression budget is spent, and finally provide additional receiver-level results.

C.1 Behavioral Rate and Distinct Training Content

Figure 2 isolates two forms of training-content diversity. First (left), we vary the number of distinct MetaMathQA rows while holding the optimization budget fixed , so that larger datasets are seen fewer times. Across 2k–32k distinct rows, $R^\star(0.90)$ increases with the amount of non-repeated training content rather than with the number of optimization updates.

Second (center), we hold the dataset size fixed at 8,000 rows while varying how many distinct source problems generated those rows. Increasing source-problem diversity again increases the measured behavioral rate. This indicates that the effect is not explained solely by the number of stored rows or sample presentations.

These controls should be interpreted within a fixed corpus and receiver. Distinct-example count does not provide a universal predictor of $R^\star$ across unrelated corpora; Table 5 reports this failure together with other alternative predictors.

C.2 Alternative Predictors of Behavioral Rate

Table 5 summarizes several intuitive candidates that do not by themselves determine $R^\star(0.90)$. Some correlate with rate within restricted experimental families, but none consistently orders compression difficulty across datasets and receivers.

C.3 Equal Bit Budgets Are Not Functionally Interchangeable

Section 3 shows that the magnitude of parameter-space reconstruction error is not sufficient to predict behavioral distortion. Here we additionally test whether the location of that error matters.

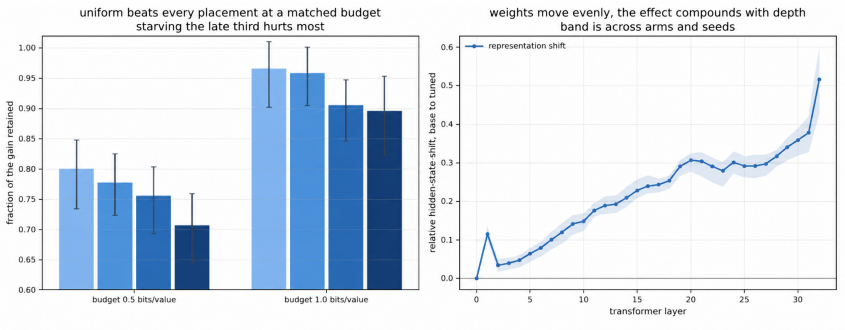

Figure 10: Equal bit budgets are not functionally interchangeable. Left: retained gain under uniform or depth-selective compression at matched budgets. Right: hidden-state shift grows with depth despite approximately flat update magnitude.

We compare uniform precision with matched-rate variants in which precision is reduced preferentially in the early, middle, or late third of the transformer. Figure 10 shows that uniform allocation preserves more fine-tuning gain at both 0.5 and 1.0 bits per value, while reducing precision in the final third is consistently the most damaging.

This difference is not explained by larger parameter updates in later layers. The RMS magnitude of the learned update is approximately flat across depth, whereas the relative hidden-state shift induced by fine-tuning grows toward the final layers. Thus, comparable parameter-space perturbations can have different behavioral effects depending on where they act in the frozen model.

C.4 Alternative Trained-Update Predictors

The main text distinguishes between quantities available before training and properties of the learned update itself. Here we report additional trained-update predictors and controls omitted from the main discussion.

A simple candidate is the fraction of update energy carried by the leading singular direction. We fixed this statistic before evaluating a confirmation panel of 64 adapters spanning four receivers and 34 corpora. On held-out receivers, it orders receiver pairs correctly 73% of the time, compared with 48% for the best correction-spectrum statistic, and reduces prediction error from 0.167 bits per value for a corpus-only baseline to 0.129.

We also tested a more behaviorally informed directional statistic that measures how much training-data gain is carried by individual singular directions. Despite using both the trained adapter and the receiver's behavior on the training data, this statistic does not transfer reliably across receivers, achieving 17% receiver-pair sign accuracy in the same evaluation.

These results motivated the shrinkability rule used in Section 6.2. For completeness, we report additional validation of

$$R^\star(0.90) \approx \alpha_{90}/\gamma.$$

On rates remeasured using evaluation rows disjoint from those used to estimate α_{90}, the rule obtains an RMSE of 0.048 bits per value. On the five new-receiver panel and the broader 34-corpus archive, the corresponding RMSEs are 0.068 and 0.105, compared with 0.093, 0.076, and 0.177 for the strongest alternative baselines in the respective evaluations.

Because the behavioral rates in these panels had already been measured when the shrinkability rule was formulated, these experiments remain retrospective. We therefore use them as evidence that update shrinkability is a strong empirical descriptor of behavioral rate, rather than as a prospectively validated prediction law.

D Additional Controls for Compression-Based Recovery

This appendix provides additional controls for Section 5. We test alternative compression operations, separate coarse coding from scalar shrinkage, vary the rank available during training, replicate the

Table 6: Recovery under alternative compression operations. Accuracy (%) on GSM8K (1,319 examples) and GSM-Symbolic (500 examples, intervals clustered by template), averaged over three seeds. Frozen-base accuracy is 77.56 / 71.40.

condition	GSM8K	GSM-Symbolic
rank 1 @ 1 bit	**83.24**	**77.60**
rank 1 @ 16 bits	42.81	33.80
full rank @ 1 bit	65.05	55.00
full rank @ 16 bits (damaged)	23.81	15.53
random rank-1 projection @ 16 bits	80.09	72.13
full update scaled to the rank-1 norm	23.93	13.80
residual rank 15 @ 16 bits	30.86	25.20

Table 7: Uniform low-bit coding can itself recover corrupted adapters. GSM8K accuracy (%) of aligned and permuted adapters under three uniform codes, averaged over three seeds. Realized rates are 1.97, 1.50, and 1.02 bits per value.

	aligned				permuted			
model	raw	2 bits	1.5 bits	1 bit	raw	2 bits	1.5 bits	1 bit
Mistral-7B	68.9	67.3	64.2	59.3	15.7	15.3	14.3	24.2
Qwen2.5-7B	83.2	83.4	82.9	82.4	23.7	23.2	27.4	65.8
Llama-3.1-8B	73.2	72.2	70.0	65.7	19.8	19.4	17.1	34.7

directional analysis across model families, and evaluate recovery across model scales, tasks, and out-of-distribution probes.

D.1 Recovery under Alternative Compression Operations

The main recovery experiments combine aggressive rank reduction with coarse quantization. We first test whether recovery is specific to this particular compression path. Table 6 compares several alternative interventions on the same corrupted Qwen2.5-7B adapters.

The rank-1, one-bit representation gives the strongest recovery in this comparison, but neither rank reduction nor quantization alone is sufficient to explain the phenomenon. In particular, a random rank-1 projection at full precision also recovers substantial performance, whereas simply scaling the full update to the norm of the rank-1 update does not.

Uniformly coding the complete adapter, without removing rank directions, already repairs part of the damage while having a comparatively small effect on an aligned adapter. For example, at one bit per value the corrupted Qwen2.5-7B adapter rises from 23.7% to 65.8% GSM8K accuracy, whereas the aligned adapter changes from 83.2% to 82.4%.

D.2 Shrinkage and Training-Rank Controls

A natural explanation for recovery is that coarse quantization simply reduces the magnitude of the corrupted update. We therefore compare coded rank-1 updates with the corresponding full-precision rank-1 update rescaled by a scalar α.

Figure 11a shows that shrinkage explains a substantial fraction of the recovery: an appropriately rescaled full-precision update can approach the performance of the one-bit code. However, matched-norm controls do not eliminate all differences between the coded and rescaled updates, indicating that coarse coding is not exactly equivalent to scalar shrinkage.

Effect of training rank. The final recovered adapter is rank 1 and one bit, raising the possibility that the same result could be obtained by training a small adapter from the start. We therefore train corrupted adapters at ranks 1, 2, 4, and 16, and compress every checkpoint to the same rank-1, one-bit endpoint. Figure 11b shows that recovery is substantially stronger when additional rank was

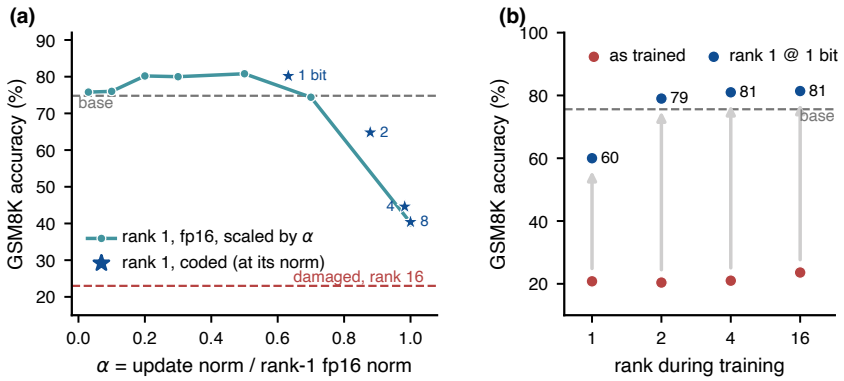

Figure 11: Shrinkage and training-rank controls. (a) GSM8K accuracy of the full-precision rank-1 update as it is rescaled by α, compared with coded rank-1 updates at their realized norms. (b) GSM8K accuracy after corrupted adapters trained at different ranks are all compressed to the same rank-1, one-bit endpoint.

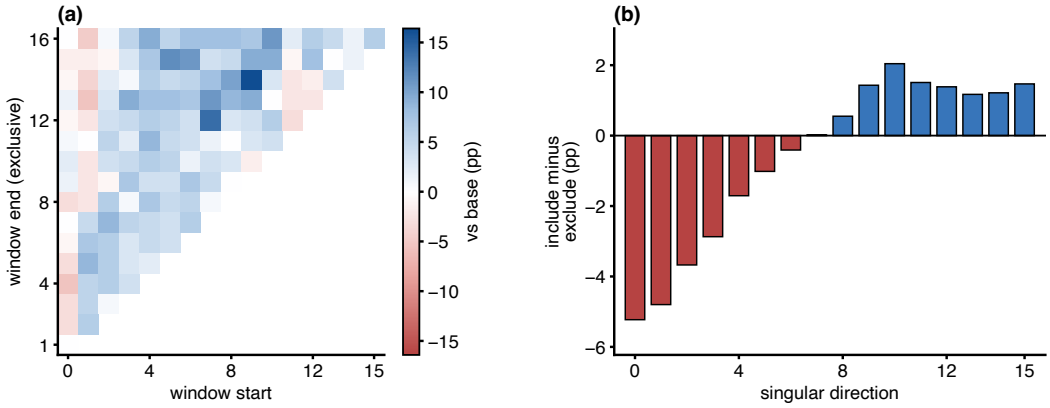

Figure 12: Directional structure on Mistral-7B-Instruct. (a) Accuracy relative to the frozen model for every contiguous window $[i, j)$ of singular directions retained at full precision. (b) For each direction, mean accuracy of windows containing it minus that of windows excluding it. Results use 128 reserved MetaMathQA examples.

available during training: final GSM8K accuracy rises from approximately 60% when trained at rank 1 to approximately 78–81% when trained at ranks 2–16.

Compression is therefore not equivalent to directly training the final low-rank representation. Higher training rank appears to make useful behavior more recoverable under the subsequent bottleneck, although these experiments do not establish the mechanism by which this separation emerges.

D.3 Directional Structure across Model Families

We repeat the singular-direction analysis of Section 5.2 on Mistral-7B-Instruct in Figure 12. The same qualitative structure appears as for Qwen2.5: windows containing the leading directions are generally harmful, while windows beginning later in the spectrum recover task performance.

Every window containing the leading direction remains at or below the frozen baseline (24.0% on average versus 25.0% for the base), whereas windows starting between directions 2 and 9 average 28.7–31.8%. Consistently, the per-direction marginal contribution changes from negative for early directions to positive for later ones.

D.4 Recovery across Model Scales and Tasks

We next test whether recovery is specific to Qwen2.5-7B and GSM8K. Table 8 reports corresponding experiments across Qwen2.5 model scales and on a second structured task.

Table 8: Recovery across model scales and tasks. For models up to 7B, entries are GSM8K accuracy (%) after training on permuted MetaMathQA rationales. For 14B and 32B, entries are XBRL exact match after corrupted response assignments.

size	base	damaged	clean	rank 1 @ 1 bit	overshoot
0.5B	23.0	9.4	45.2	25.0	+2.0
1.5B	51.6	13.6	67.6	59.0	+7.4
3B	60.4	17.4	74.0	71.0	**+10.6**
7B	75.6	22.8	81.0	80.8	+5.2
14B, XBRL tagging	24.6	53.0	–	73.8	+49.2
32B, XBRL tagging	32.4	51.8	–	74.4	+42.0

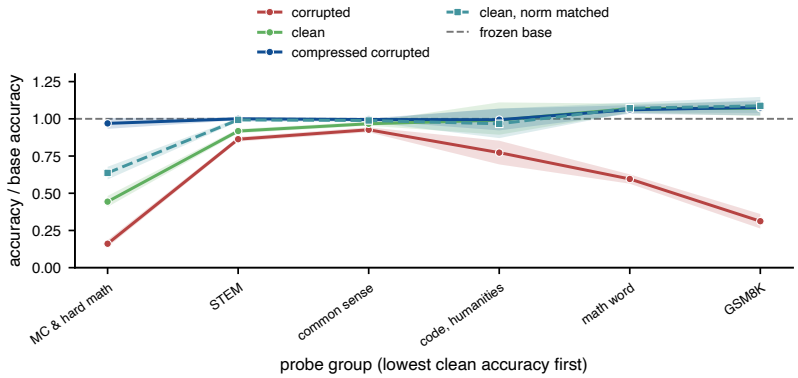

Figure 13: Off-task behavior after compression-based recovery. Accuracy on 72 probes normalized by the frozen Qwen2.5-7B model and grouped by task proximity to GSM8K. The compressed corrupted adapter remains close to the frozen model away from its training task, unlike the clean fine-tune even under a norm-matched control.

For models up to 7B, corrupted MetaMathQA rationales consistently damage GSM8K accuracy and the compressed adapter recovers a substantial fraction of the loss. For the 14B and 32B models, we additionally report XBRL tagging under corrupted response assignments, where the frozen models leave more headroom. Recovery remains visible in this setting.

The XBRL corruption differs from the permuted-rationale intervention: mismatched responses can still teach the model task-specific output vocabulary, so the corrupted adapter may already outperform the frozen model. The relevant comparison is therefore whether compression further separates task-relevant structure from the corruption-induced behavior.

D.5 Off-Task Behavior after Recovery

Finally, we ask whether recovering in-domain performance comes at the cost of larger behavioral changes elsewhere. We evaluate the frozen model, a clean fine-tune, and the compressed corrupted adapter on 72 probes of varying proximity to GSM8K.

Figure 13 shows that the compressed corrupted adapter remains close to the frozen model on distant task groups despite recovering strong GSM8K performance. The contrast is particularly large on harder mathematical tasks: the clean adapter retains 44% of the frozen model's accuracy, whereas the compressed corrupted adapter retains 97%. A norm-matched clean control indicates that this difference is not explained solely by the magnitude of the learned update.

E Additional Related Work

Low-rank parameterizations and capacity. A large body of work has refined the original LoRA parameterization by changing how adaptation capacity is allocated or represented. AdaLoRA adaptively allocates a parameter budget across weight matrices and prunes less important singular com-

ponents of the learned update (Zhang et al., 2023b). VeRA substantially reduces trainable storage by sharing frozen low-rank matrices across layers and learning only small scaling vectors (Kopiczko et al., 2024), while LoRA-XS uses frozen SVD-derived bases and trains only a small $r \times r$ matrix between them (Bałazy et al., 2025). PiSSA instead initializes the adaptive subspace from the principal singular directions of the pretrained weights (Meng et al., 2024). In the opposite direction, MoRA argues that restricting the update itself to low rank can limit learning and memorization, and obtains higher-rank updates under a comparable parameter budget (Jiang et al., 2024). These methods demonstrate that trainable parameter count, update rank, and the subspace in which adaptation occurs are distinct design choices. Our behavioral rate measures a different quantity: how many serialized bits of a trained update must survive in order to preserve its behavior when composed with a particular frozen receiver.

Optimization and parameterization of LoRA updates. Other work shows that even when the represented update class is similar, its factorization and optimization can substantially affect learning. LoRA+ assigns different learning rates to the two LoRA factors to improve feature learning (Hayou et al., 2024), while LoRA-FA freezes one factor to reduce activation-memory cost (Zhang et al., 2023a). DoRA decomposes pretrained weights into magnitude and direction and applies low-rank adaptation to the directional component (Liu et al., 2024). More recently, Balanced LoRA observes that many pairs of LoRA factors represent the same weight update but can have very different conditioning, and projects optimization onto a balanced factorization (Castin et al., 2026). This distinction between the parameterization used to learn an update and the functional effect of the resulting update is complementary to our setting: our compression analysis is performed after training and evaluates the decoded update through the frozen model rather than through its factorization alone.

Selecting and composing adapter components. Several methods exploit non-uniform functional importance within or across adapters. LoRA-drop ranks layers by their LoRA output contribution, retains separate modules for important layers, and shares LoRA parameters across the remaining layers (Zhou et al., 2025), while LoraHub composes independently trained LoRA modules to transfer across tasks (Huang et al., 2024). These approaches further illustrate that adapter components are not interchangeable. Our experiments ask a complementary question: after an adapter has been learned, which parts of its representation must remain after compression to preserve the acquired behavior?

Reasoning structure and chain-of-thought compression. Several of our corruption experiments use chain-of-thought supervision (Wei et al., 2022). Qian et al. (2025) study information dynamics along reasoning trajectories, while Dutta et al. (2024) analyze mechanisms supporting step-by-step reasoning. Related mechanistic work studies symbolic computation and structured state tracking (Yang et al., 2025; Fagnou et al., 2024). Li et al. (2026) directly compress chain-of-thought traces using step-level information measures. Our experiments instead use corrupted reasoning supervision as a controlled way to induce useful and harmful effects within the same learned update.