TWO FOR THE PRICE OF ONE: CURVATURE-AWARE PRUNING PRESERVES INFLUENCE FUNCTIONS TOO

Anonymous authors
Paper under double-blind review

ABSTRACT

Post-training pruning enables efficient deployment, but conventional methods do not explicitly preserve internal-gradient alignment between sparse and dense models. We study novel connections between downstream performance of compressed models and the ability to retain gradient information via influence-function preservation. Central to this connection is a curvature-aware objective that accounts for both layer activations and backpropagated gradients. Despite its promise, this objective remains underexplored in pruning due to a lack of efficient solvers. We introduce CACTUS, a novel ADMM-based optimizer supporting both pruning and sparse-plus-low-rank decompositions of pre-trained weights. Compared to LLM-Surgeon which targets the same objective, CACTUS achieves over an order-of-magnitude speedup. At 50% pruning of Llama-3.2-1B, CACTUS reduces the C4 perplexity gap to the dense model by 24%, improves compressed-to-dense per-sample gradient cosine similarity on WikiText-2 from 0.445 to 0.512, and runs over $15\times$ faster than LLM-Surgeon on substantially weaker hardware. By preserving influence functions, CACTUS supports effective data selection. On Llama-3.2-3B, fine-tuning a fresh dense model on the 5% of data selected by CACTUS achieves 49.44% accuracy on BIG-Bench Hard, compared with 47.22% using ALPS and 49.54% using dense-model selection. Finally, integrating CACTUS with LoGRA, a recent gradient-projection strategy, on Qwen3-4B with MLP layers pruned to 2:4 sparsity and a rank-64 correction retains a 0.711 cosine alignment with dense LoGRA-projected C4 gradients.

1 INTRODUCTION

Large Language Models (LLMs) (Achiam et al., 2023; Dubey et al., 2024; Google, 2023) have driven advances in reasoning, code, and mathematics, but their sheer size often measured in billions of parameters make both deployment and downstream inference expensive. These costs have motivated **two parallel research threads** that have advanced largely independently.

The first thread is *post-training pruning*. Methods such as SparseGPT (Frantar and Alistarh, 2023), WANDA (Sun et al., 2024), and ALPS (Meng et al., 2024) use a small calibration dataset to minimize the *layerwise-reconstruction error* $\|\mathbf{X}\mathbf{W} - \mathbf{X}\widehat{\mathbf{W}}\|_F^2$ subject to a structural constraint, where $\mathbf{X}$ contains input activations and $\mathbf{W}$ and $\widehat{\mathbf{W}}$ are the dense and compressed weights, and $\|\cdot\|_F$ is the Frobenius norm. *Pruning* (pure sparsity) zeroes out the weights in an unstructured pattern or in hardware-friendly n:m patterns that keep n nonzeros in each contiguous block of m weights; the common 2:4 case is natively accelerated by recent NVIDIA sparse Tensor Cores (Zhou et al., 2021). Sparse-plus-low-rank (SLR) compression, denoted in this paper by $(\mathbf{S} + \mathbf{L})$, combines a sparse backbone $\mathbf{S}$ with a low-rank correction $\mathbf{L}$ under the same framework, improving compression-utility tradeoffs over either component alone (Zhang and Papyan, 2025; Makni et al., 2025; 2026).

The *layerwise-reconstruction* procedure above matches each layer's outputs given its input activations and is therefore *gradient-blind* to the LLM loss (see Figure 1). LLM-Surgeon (van der Ouderaa et al., 2024) (pruning) and YAQA (Tseng et al., 2026b) (quantization) additionally incorporate downstream gradients from the calibration dataset. This curvature-aware objective improves model quality, but its optimization can require multiple GPU hours per model (LLM-Surgeon in Table 1), limiting its adoption. Our solver minimizes the curvature-aware loss function more than an *order of*

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.

magnitude faster than LLM-Surgeon while improving perplexity (Table 1). It also extends beyond pure pruning to curvature-aware $(\mathbf{S}+\mathbf{L})$ compression. Figure 3 studies Llama-3-8B with 2:4 sparsity and a rank-256 correction, matching the $(\mathbf{S}+\mathbf{L})$ configuration with serving speed and memory gains reported by Mozaffari et al. (2024).

The second thread is *efficient influence function (IF) estimation*. Influence functions, with early roots in robust statistics (Hampel, 1974), estimate the counterfactual effect of a training example via $\mathcal{I}(z,z') = \nabla\ell(z')^\top\mathcal{H}^{-1}\nabla\ell(z)$, where z and z' are training and query examples, $\nabla\ell$ is the per-example parameter gradient, and $\mathcal{H}$ is the training-loss Hessian. Applications include data attribution (Pruthi et al., 2020; Park et al., 2023), training-data selection (Xia et al., 2024), model debugging and provenance (Grosse et al., 2023), and machine unlearning. At LLM scale, evaluating this expression is prohibitively expensive. Existing methods reduce the cost on two fronts: (i) inverse-Hessian approximations (Kronecker-Factored Approximate Curvature (KFAC) (Martens and Grosse, 2015; Grosse et al., 2023), its eigenvalue-corrected variant (EK-FAC), identity-Hessian as in TracIn (Pruthi et al., 2020), Adam-style preconditioners), and (ii) on the gradient side: random or architecture-aware gradient sketches (TRAK (Park et al., 2023), Low-rank Gradient Approximation (LoGRA) (Choe et al., 2025)); see Section 4. These methods still require *per-sample gradient computations*, which remain costly for large models and datasets.

IProX (Chen et al., 2026) introduced *compressed-model proxies*: with the idea of computing influence estimates using a smaller surrogate model whose gradients mimic those of the dense model. They show that minimizing the curvature-aware objective, also used by LLM-Surgeon, helps preserve per-sample gradients. IProX instantiates this idea using low-rank proxies with a two-stage pipeline: (i) each weight matrix is approximated by a product of two smaller matrices, obtained through a closed-form weighted singular value decomposition (SVD), termed influence-preserving SVD (IPSVD); and (ii) gradient-alignment distillation further adapts the low-rank proxy.

In this paper, we study the connection between these **two different research threads**. The bridge is the curvature-aware objective in Equation (1), and the enabler is a single algorithm that makes it practical for both high-quality compression and influence-preserving pruned proxies. Here, influence preservation means retaining the dense model's gradient-based influence estimates after compression. This is the sense in which the paper offers "Two for the Price of One."

Our **main contributions** can be summarized as follows.

1. **Efficient Curvature-Aware Compression.** We introduce `CACTUS`[1], a highly-efficient algorithm tailored for Equation (1) with convergence guarantees. It handles both pure pruning (`CACTUS`-prune) and sparse-plus-low-rank (`CACTUS`-SLR) constraints, and runs over an order of magnitude faster than LLM-Surgeon on a significantly weaker hardware on Llama-3.2-1B. The updates are derived in Section 2. With this new unifying algorithm, we make contributions in model compression on two fronts: preserving downstream utility and influence function preservation.
2. **Compression Quality.** On Llama-3.2-1B at 50% compression, `CACTUS` reduces the C4 perplexity gap to the dense model by 24% for pure pruning and 28% for $(\mathbf{S}+\mathbf{L})$ decompositions relative to the strongest corresponding baseline. Additionally, it maintains its advantage at different compression ratios, architectures, and model sizes (see Tables 1, 8 and 10 to 12). In particular, we show in Table 13 that `CACTUS` reduces the C4 perplexity gap to dense by over 25% compared to SparseGPT under 2:4 sparsity for Qwen3.8-27B.
3. **Influence Preservation.** On Llama-3.2-1B at 50% pruning, `CACTUS` improves the cosine similarity between compressed and dense per-example gradients on WikiText-2 from 0.445 (LLM-Surgeon) to 0.512 (Table 1). We then use pruned models with 50% sparsity to select 5% of data from an instruction-tuning mixture for fine-tuning a fresh Llama-3.2-3B model. `CACTUS` selected subset overlaps the dense model's selection by 54.13%, compared with 52.24% for ALPS, and yields 49.44% versus 47.22% BBH accuracy (Table 2). On Qwen3-4B, compressing MLP layers to 2:4 sparsity with a rank-64 correction retains a cosine similarity of 0.711 between compressed and dense per-example gradients on C4, measured using the same LoGRA projection.

[1] **C**urvature-**A**ware **C**ompression with **T**ractable **U**pdates via **S**ylvester Steps: Our method uses iterative closed-form updates to find high-quality solutions for Equation (1). One key update (smooth subproblem in ADMM) solves a Sylvester equation $\mathbf{HSG} + \frac{\rho}{2}\mathbf{S} = \text{RHS}$ in closed-form at matmul cost, hence the name of the algorithm–see more details in Section 2.2.

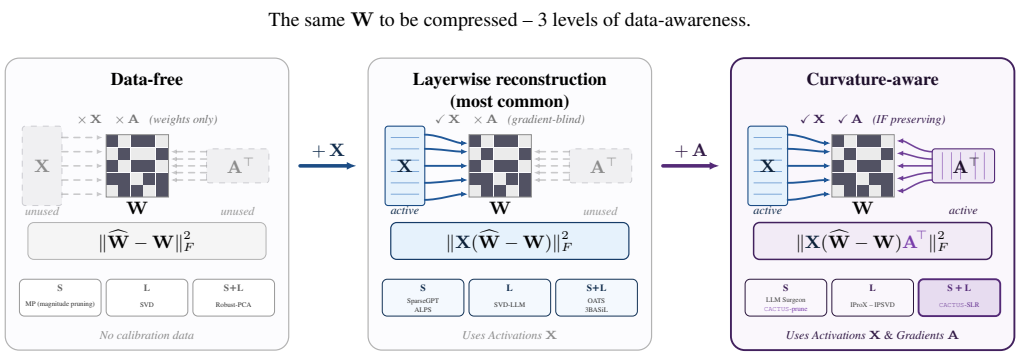


Figure 1: Each panel is a layer-level reconstruction problem centered on the same weight matrix $\mathbf{W}$. Input activations from the previously compressed layer are represented by $\mathbf{X} \in \mathbb{R}^{N_{\text{cal}} \times d_{\text{in}}}$; the downstream gradients incoming from the next layer are represented by $\mathbf{A} \in \mathbb{R}^{N_{\text{cal}} \times d_{\text{out}}}$. *(Left)* **Data-free.** Neither $\mathbf{X}$ nor $\mathbf{A}$ is available; both sides reduce to parallel gray passes and the objective collapses to $\|\widehat{\mathbf{W}} - \mathbf{W}\|_F^2$. *(Middle)* **Layerwise reconstruction (most common).** Calibration activations $\mathbf{X}$ light up the left side, giving $\|\mathbf{X}(\widehat{\mathbf{W}} - \mathbf{W})\|_F^2$; outputs are matched but the method remains *gradient-blind* (SparseGPT, ALPS, 3BASiL, SVD-LLM). *(Right)* **Curvature-aware.** Both activations and the downstream gradients are used, yielding $\|\mathbf{X}(\widehat{\mathbf{W}} - \mathbf{W})\mathbf{A}^\top\|_F^2$, as studied in this paper.

2 Curvature-Aware Model Compression

2.1 Problem Formulation

We compress a pretrained LLM on a layer-by-layer basis. For a linear layer $\mathbf{W} \in \mathbb{R}^{d_{\text{in}} \times d_{\text{out}}}$, we use N_{cal} calibration data points to introduce (i) its input activations $\mathbf{X} \in \mathbb{R}^{N_{\text{cal}} \times d_{\text{in}}}$, and (ii) its downstream gradients $\mathbf{A} \in \mathbb{R}^{N_{\text{cal}} \times d_{\text{out}}}$. We seek a compressed weight $\widehat{\mathbf{W}}$ satisfying a structural constraint $\widehat{\mathbf{W}} \in \mathcal{C}$ (see below) that minimizes the Kronecker-Factored Approximate Curvature (KFAC) loss:

$$\mathcal{L}_{\text{kfac}}(\widehat{\mathbf{W}}) \;=\; \left\| \mathbf{X}\,(\widehat{\mathbf{W}} - \mathbf{W})\,\mathbf{A}^\top \right\|_F^2 \;=\; \operatorname{Tr}\left(\mathbf{H}\,(\widehat{\mathbf{W}} - \mathbf{W})\,\mathbf{G}\,(\widehat{\mathbf{W}} - \mathbf{W})^\top\right), \tag{1}$$

where Tr denotes the matrix trace, $\mathbf{H} = \mathbf{X}^\top\mathbf{X} \in \mathbb{R}^{d_{\text{in}} \times d_{\text{in}}}$ is the input activations covariance and $\mathbf{G} = \mathbf{A}^\top\mathbf{A} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{out}}}$ is the downstream gradients covariance. For efficiency, we collect $\mathbf{G}$ for *all* layers at the start of the algorithm using a single forward-backward pass on the N_{cal} calibration data points and store them on CPU. We then proceed to compress layers sequentially by constructing $\mathbf{H}$ from outputs of previously compressed layers. Compared to the statistics collected using a layerwise reconstruction error method (e.g., SparseGPT), we only add the collection of $\mathbf{G}$ for all layers *once* at the start of the algorithm. We denote Eq (1) by *KFAC* loss because it can be derived using the Kronecker factorization approximation in two distinct ways (i) a downstream compression quality perspective in LLM-Surgeon and (ii) influence function preservation perspective in IProX, see more details in Section A (Figure 5). For compression, we consider two structural constraints $\mathcal{C}$:

- **Pure pruning:** either unstructured sparsity $\mathcal{C}_k = \{\mathbf{S} \in \mathbb{R}^{d_{\text{in}} \times d_{\text{out}}} : \|\mathbf{S}\|_0 \le k\}$, or $n{:}m$-structured sparsity $\mathcal{C}_{n:m}$ that keeps at most n non-zeros in each contiguous block of m entries. Here, $\|.\|_0$ counts the number of non-zero elements. In Algorithm 1, $\mathcal{C}_S$ denotes either sparsity set.
- **Sparse + low rank $(\mathbf{S}+\mathbf{L})$:** $\mathcal{C}_{\mathbf{S}+\mathbf{L}} = \mathcal{C}_S + \mathcal{C}_L$, i.e. $\widehat{\mathbf{W}} = \mathbf{S} + \mathbf{L}$ with simultaneous sparsity and rank budgets, where $\mathcal{C}_L = \{\mathbf{L} : \operatorname{rank}(\mathbf{L}) \le r\}$ and r is the rank budget.

2.2 CACTUS Algorithm Derivation

We treat the $(\mathbf{S}+\mathbf{L})$ family as the general case, with pure pruning (obtained by setting rank $r = 0$) as a special case. We use the Alternating Direction Method of Multipliers (ADMM) (Boyd et al., 2011) for the Kronecker-Factored Approximate Curvature (KFAC) objective (1). Using variable splitting for the sparse component, our problem formulation is given as:

$$\begin{aligned} \min_{\mathbf{S},\boldsymbol{\Delta},\mathbf{L}} \quad & \operatorname{Tr}\left(\mathbf{H}(\mathbf{W} - \mathbf{S} - \mathbf{L})\mathbf{G}(\mathbf{W} - \mathbf{S} - \mathbf{L})^\top\right) \\ \text{s.t.} \quad & \mathbf{S} \;=\; \boldsymbol{\Delta}, \qquad \boldsymbol{\Delta} \in \mathcal{C}_S, \quad \operatorname{rank}(\mathbf{L}) \le r, \end{aligned} \tag{2}$$

where $\mathbf{\Delta} \in \mathcal{C}_S$ is set to be sparse and $\mathbf{S}$ is its copy; $\mathbf{L}$ is the low-rank component with $\mathrm{rank}\,(\mathbf{L}) \le r$. The sparsity constraint is decoupled from the smooth objective through the consensus equality $\mathbf{S} = \mathbf{\Delta}$. The augmented Lagrangian with scaled dual variable $\mathbf{V}$ and penalty $\rho > 0$ is

$$\mathcal{L}_\rho(\mathbf{S}, \mathbf{L}, \mathbf{\Delta}, \mathbf{V}) \;=\; \mathrm{Tr}\left(\mathbf{H}(\mathbf{W}-\mathbf{S}-\mathbf{L})\mathbf{G}(\mathbf{W}-\mathbf{S}-\mathbf{L})^\top\right) \;+\; \tfrac{\rho}{2}\left\|\mathbf{S}-\mathbf{\Delta}+\mathbf{V}\right\|_F^2 \;-\; \tfrac{\rho}{2}\|\mathbf{V}\|_F^2. \quad (3)$$

In practice, we damp $\mathbf{H}$ and $\mathbf{G}$ using separate ridge parameters λ_H and λ_G to improve conditioning (see Section C). In what follows, $\mathbf{H}$ and $\mathbf{G}$ denote these damped factors. We perform an eigendecomposition of $\mathbf{H}$ and $\mathbf{G}$ once, then ADMM alternates across three primal blocks ($\mathbf{S}$, $\mathbf{\Delta}$, $\mathbf{L}$) one at a time followed by a dual update in $\mathbf{V}$. We minimize (3) over each primal block $\mathbf{S}$, $\mathbf{\Delta} \in \mathcal{C}_S$ and $\mathrm{rank}\,(\mathbf{L}) \le r$ one at a time. Each primal subproblem minimization admits a closed-form solution and the $\mathbf{S}$- and $\mathbf{L}$-updates reuse the same eigenspaces throughout the ADMM iterations:

$$\mathbf{H} \;=\; \mathbf{Q}_H\,\mathbf{\Lambda}_H\,\mathbf{Q}_H^\top, \qquad \mathbf{G} \;=\; \mathbf{Q}_G\,\mathbf{\Lambda}_G\,\mathbf{Q}_G^\top, \qquad \text{(eigendecompositions)} \quad (4)$$

with $\mathbf{\Lambda}_H = \mathrm{diag}\,(\lambda_{H,1}, \ldots, \lambda_{H,d_{\text{in}}})$ and $\mathbf{\Lambda}_G = \mathrm{diag}\,(\lambda_{G,1}, \ldots, \lambda_{G,d_{\text{out}}})$.

(i) S-update: Sylvester equation. Holding $\mathbf{L}, \mathbf{\Delta}, \mathbf{V}$ fixed, the augmented Lagrangian (3) is a strictly convex quadratic in $\mathbf{S}$. Setting $\nabla_{\mathbf{S}}\mathcal{L}_\rho = 0$ yields the matrix linear system

$$\mathbf{H}\,\mathbf{S}\,\mathbf{G} \;+\; \tfrac{\rho}{2}\,\mathbf{S} \;=\; \mathbf{H}\,(\mathbf{W}-\mathbf{L})\,\mathbf{G} \;+\; \tfrac{\rho}{2}\,(\mathbf{\Delta}-\mathbf{V}), \quad (5)$$

which is a *generalized Sylvester equation* in $\mathbf{S}$ (Sylvester, 1909). Crucially, this is what differentiates CACTUS from prior ADMM-based layerwise reconstruction methods such as ALPS (Meng et al., 2024) or 3BASiL (Makni et al., 2026): those methods only deal with a one-sided system $(\mathbf{H} + \frac{\rho}{2}\mathbf{I})\mathbf{S} = \mathrm{RHS}$ under our normalization, whereas the curvature-aware KFAC objective forces a *two-sided* coupling that, naively solved, would cost $O((d_{\text{in}}d_{\text{out}})^3)$ via a vectorized Kronecker formulation $(\mathbf{G}^\top \otimes \mathbf{H} + \frac{\rho}{2}\mathbf{I})\,\mathrm{vec}(\mathbf{S}) = \mathrm{vec}(\mathrm{RHS})$, which is completely **intractable** for LLM-scale matrices.

Closed-form solve in the two eigenbases. The Sylvester structure admits an exact solution by substituting $\mathbf{S} = \mathbf{Q}_H\widetilde{\mathbf{S}}\mathbf{Q}_G^\top$ into (5); the equation *decouples elementwise*:

$$\left(\lambda_{H,i}\,\lambda_{G,j} + \tfrac{\rho}{2}\right)\widetilde{\mathbf{S}}_{ij} \;=\; \widetilde{\mathrm{RHS}}_{ij}, \qquad \widetilde{\mathrm{RHS}} \;=\; \mathbf{Q}_H^\top\left[\mathbf{H}(\mathbf{W}-\mathbf{L})\mathbf{G} + \tfrac{\rho}{2}(\mathbf{\Delta}-\mathbf{V})\right]\mathbf{Q}_G. \quad (6)$$

We obtain each entry of $\widetilde{\mathbf{S}}$ by a single scalar division with $\lambda_{H,i}\lambda_{G,j} + \rho/2 > 0$—the $\mathbf{S}$-step remains nonsingular for any $\rho > 0$ and positive definitie $\mathbf{H}, \mathbf{G}$. The final solution is $\mathbf{S} = \mathbf{Q}_H\widetilde{\mathbf{S}}\mathbf{Q}_G^\top$.

Algorithmic implications. The whole inner solve thus reduces to (i) matmuls to form $\mathbf{Q}_H^\top(\cdot)\mathbf{Q}_G$, (ii) an elementwise division on a $d_{\text{in}} \times d_{\text{out}}$ matrix, and (iii) matmuls to undo the basis change. The smooth solve costs $O(d_{\text{in}}d_{\text{out}}(d_{\text{in}} + d_{\text{out}}))$ per iteration, with the eigenbases reused across all T iterations and in the $\mathbf{L}$-update for whitening. The ability to solve (5) *exactly and at matmul cost* is key to making curvature-aware ADMM tractable and yields the runtimes reported in Section 3.

(ii) L-update: IPSVD (Chen et al., 2026). Holding $\mathbf{S}, \mathbf{\Delta}, \mathbf{V}$ fixed, for the $\mathbf{L}$-update, we minimize $\mathrm{Tr}\left(\mathbf{H}(\mathbf{R}_{\mathbf{S}}-\mathbf{L})\mathbf{G}(\mathbf{R}_{\mathbf{S}}-\mathbf{L})^\top\right)$ (here, $\mathbf{R}_{\mathbf{S}} = \mathbf{W}-\mathbf{S}$) subject to $\mathrm{rank}\,(\mathbf{L}) \le r$. Since $\mathbf{H}$ and $\mathbf{G}$ are positive definite, the change of variables $\mathbf{M} = \mathbf{H}^{1/2}\mathbf{L}\mathbf{G}^{1/2}$ preserves rank and reduces the subproblem to ordinary low-rank approximation. Its exact minimizer is therefore

$$\mathbf{L}^\star \;=\; \mathbf{H}^{-1/2}\,\mathrm{SVD}_r\!\left(\mathbf{H}^{1/2}\,\mathbf{R}_{\mathbf{S}}\,\mathbf{G}^{1/2}\right)\mathbf{G}^{-1/2}, \quad (7)$$

where SVD_r denotes a best rank-r approximation obtained by truncated SVD, and $\mathbf{H}^{\pm 1/2} = \mathbf{Q}_H\mathbf{\Lambda}_H^{\pm 1/2}\mathbf{Q}_H^\top$ and $\mathbf{G}^{\pm 1/2} = \mathbf{Q}_G\mathbf{\Lambda}_G^{\pm 1/2}\mathbf{Q}_G^\top$ reuse eigendecompositions from (4). We accept this candidate only when its movement is bounded by the smooth-block movement:

$$\mathbf{L}^{(t+1)} = \begin{cases} \mathbf{L}^\star, & \|\mathbf{L}^\star - \mathbf{L}^{(t)}\|_F \le C_L\|\mathbf{S}^{(t+1)} - \mathbf{S}^{(t)}\|_F, \\ \mathbf{L}^{(t)}, & \text{otherwise}, \end{cases} \quad (8)$$

where $C_L \ge 0$ is fixed and finite. For pure pruning ($r = 0$), we keep $\mathbf{L} = \mathbf{0}$.

(iii) $\mathbf{\Delta}$-update: exact sparse projection. The only term involving $\mathbf{\Delta}$ is the consensus quadratic, so we retain the top-k entries by magnitude (unstructured) or the top-n entries within each m-block (n:m-structured) and set the remaining elements to 0. We denote this projection by $\Pi_{\mathcal{C}_S}(\mathbf{S}+\mathbf{V})$.

(iv) Scaled dual and penalty updates. Given the new penalty, we update $\mathbf{V}^{(t+1)} = \frac{\rho^{(t)}}{\rho^{(t+1)}}\big(\mathbf{V}^{(t)} + \mathbf{S}^{(t+1)} - \mathbf{\Delta}^{(t+1)}\big)$. The ratio rescales the dual variable to preserve the updated Lagrange multiplier.

Algorithm 1 CACTUS for one linear layer.

Require: Pretrained $\mathbf{W}$, damped positive-definite factors $\mathbf{H}, \mathbf{G}$ (eigendecomposed), sparsity set $\mathcal{C}_S$, rank r, movement bound C_L, positive penalties $\{\rho^{(t)}\}$, iterations T.

Initialize $\mathbf{S}^{(0)} \leftarrow \mathbf{W}$, $\mathbf{L}^{(0)} \leftarrow \mathbf{0}$, $\mathbf{\Delta}^{(0)} \leftarrow \mathbf{0}$, $\mathbf{V}^{(0)} \leftarrow \mathbf{0}$.
for $t = 0, 1, \ldots, T-1$ **do**
  $\mathbf{S}^{(t+1)} \leftarrow$ Sylvester solve (5) using $(\mathbf{L}^{(t)}, \mathbf{\Delta}^{(t)}, \mathbf{V}^{(t)}, \rho^{(t)})$.
  $\mathbf{L}^{\star} \leftarrow$ weighted rank-r SVD of $\mathbf{W} - \mathbf{S}^{(t+1)}$ via (7).
  $\mathbf{L}^{(t+1)} \leftarrow \mathbf{L}^{\star}$ **if** $\|\mathbf{L}^{\star} - \mathbf{L}^{(t)}\|_F \le C_L \|\mathbf{S}^{(t+1)} - \mathbf{S}^{(t)}\|_F$ **else** $\mathbf{L}^{(t)}$.
  $\mathbf{\Delta}^{(t+1)} \leftarrow \Pi_{\mathcal{C}_S}(\mathbf{S}^{(t+1)} + \mathbf{V}^{(t)})$.
  $\mathbf{V}^{(t+1)} \leftarrow \frac{\rho^{(t)}}{\rho^{(t+1)}}\big(\mathbf{V}^{(t)} + \mathbf{S}^{(t+1)} - \mathbf{\Delta}^{(t+1)}\big)$.
return $\widehat{\mathbf{W}} \leftarrow \mathbf{\Delta}^{(T)} + \mathbf{L}^{(T)}$.

2.3 Convergence

The convergence argument follows 3BASiL (Makni et al., 2026). Under the conditions below, the iterates converge to a feasible sparse-plus-low-rank decomposition.

Theorem 1 (Convergence to a feasible decomposition)**.** *Consider Algorithm 1 with positive definite* $\mathbf{H}, \mathbf{G}$*, exact smooth solves and sparse projections, and a fixed finite movement bound* C_L*. Suppose the positive penalties* (ρ_t) *are non-decreasing and satisfy* $\sum_{t=0}^{\infty} 1/\rho^{(t)} < \infty$*. Then*

$$\mathbf{S}^{(t)}, \mathbf{\Delta}^{(t)} \longrightarrow \bar{\mathbf{S}} \in \mathcal{C}_S, \qquad \mathbf{L}^{(t)} \longrightarrow \bar{\mathbf{L}}, \quad \operatorname{rank}\big(\bar{\mathbf{L}}\big) \le r.$$

The successive changes in $\mathbf{S}^{(t)}$ *and* $\mathbf{L}^{(t)}$*, and the consensus residual* $\mathbf{S}^{(t+1)} - \mathbf{\Delta}^{(t+1)}$*, are all* $O(1/\rho^{(t-1)})$ *in Frobenius norm.*

The two-sided curvature changes the smooth solve, but preserves the projection and boundedness arguments used in 3BASiL; Section B gives the adaptation in two lemmas. Increasing ρ by a fixed factor greater than one at fixed intervals satisfies the summability condition (see Section C).

3 Experimental Results

3.1 Experimental Setup

Models and Evaluation Protocol. We compute gradients using the language-model cross-entropy loss. We evaluate CACTUS on different settings and configurations using one of the following model architectures: Llama-3.2-1B/3B, Llama-2-7B, Llama-3-8B (Dubey et al., 2024), Qwen3-4B or Qwen3.8-27B (Yang et al., 2025). For model compression utility, we report C4 (Raffel et al., 2020) and WikiText-2 (WT-2) (Merity et al., 2017) perplexity (PPL) at sequence length 2048. Further, we evaluate influence function (IF) preservation under two gradient representations, since influence methods differ mainly in the inner product used to compare per-sample gradients. For every sample x and linear layer ℓ, let $\nabla_{x,\ell} \in \mathbb{R}^{d_{\ell,\text{in}} \times d_{\ell,\text{out}}}$ denote the per-sample weight gradient. The **TracIn** representation (Pruthi et al., 2020) uses the raw single-checkpoint gradient vector $r_x^{\mathrm{T}} = \operatorname{concat}_\ell \operatorname{vec}(\nabla_{x,\ell})$. The **KFAC-weighted** representation (Mlodozeniec et al., 2024; Grosse et al., 2023) uses $r_x^{\mathrm{K}} = \operatorname{concat}_\ell \operatorname{vec}(\nabla_{x,\ell}^{\mathrm{K}})$ with $\nabla_{x,\ell}^{\mathrm{K}} = \mathbf{H}_\ell^{-1/2} \nabla_{x,\ell} \mathbf{G}_\ell^{-1/2}$, where $\mathbf{H}_\ell$ and $\mathbf{G}_\ell$ are the dense model's activation and backpropagated-gradient covariance factors. We use the dense factors for both dense and compressed gradients: they are calibration-level curvature statistics, distinct from per-sample gradients, and can be precomputed once then applied on the fly. Before projection, the KFAC-weighted layerwise inner product is $\langle \nabla_{x,\ell}^{\mathrm{K}}, \nabla_{q,\ell}^{\mathrm{K}} \rangle_F = \operatorname{vec}(\nabla_{x,\ell})^\top (\mathbf{G}_\ell^{-1} \otimes \mathbf{H}_\ell^{-1}) \operatorname{vec}(\nabla_{q,\ell})$, corresponding to the standard KFAC approximation of the Hessian.

For either representation $r_x \in \{r_x^{\mathrm{T}}, r_x^{\mathrm{K}}\}$, we store $\tilde{\nabla}_x = \Pi r_x$ using the same TRAK (Park et al., 2023) Rademacher Johnson-Lindenstrauss (JL) sketch Π down to $d = 8192$. This projection preserves the relevant geometry: $\langle \Pi r_x, \Pi r_q \rangle \approx \langle r_x, r_q \rangle$ and $\|\Pi r_x\| \approx \|r_x\|$, so projected inner products

and cosine similarities approximate their full-gradient counterparts. In what follows, $\tilde{\nabla}^{\mathrm{d}}_x, \tilde{\nabla}^{\mathrm{c}}_x \in \mathbb{R}^d$ denote the dense and compressed projected gradients of the same sample x for a fixed representation, and $\mathbf{I}^{\mathrm{d}}, \mathbf{I}^{\mathrm{c}} \in \mathbb{R}^{n_{\mathrm{tr}} \times n_{\mathrm{te}}}$ denote each model's *influence matrix* with entries $\mathbf{I}_{ij} = \langle \tilde{\nabla}_{x_i}, \tilde{\nabla}_{q_j} \rangle$ (training row i, test column j). Following influence-function terminology, "training" rows are the source examples being ranked, while "test" columns are the query examples (distinct from a train/test split). The four metrics below are defined once on $\tilde{\nabla}$, and *each is reported twice*: TracIn and KFAC-weighted. Cosine metrics measure per-example gradient-direction fidelity before any cross-example influence score is formed:

$$\cos_{\mathrm{tr}} = \frac{1}{n_{\mathrm{tr}}} \sum_{i=1}^{n_{\mathrm{tr}}} \frac{\langle \tilde{\nabla}^{\mathrm{d}}_{x_i}, \tilde{\nabla}^{\mathrm{c}}_{x_i} \rangle}{\|\tilde{\nabla}^{\mathrm{d}}_{x_i}\| \, \|\tilde{\nabla}^{\mathrm{c}}_{x_i}\|}, \qquad \cos_{\mathrm{te}} = \frac{1}{n_{\mathrm{te}}} \sum_{j=1}^{n_{\mathrm{te}}} \frac{\langle \tilde{\nabla}^{\mathrm{d}}_{q_j}, \tilde{\nabla}^{\mathrm{c}}_{q_j} \rangle}{\|\tilde{\nabla}^{\mathrm{d}}_{q_j}\| \, \|\tilde{\nabla}^{\mathrm{c}}_{q_j}\|}.$$

Query-ranking metrics measure whether each query induces the same influence ordering over source examples: Spearman uses the full ordering, while recall checks overlap among the top 5%. Let $T_k(v)$ denote the indices of the $k = \lceil 0.05 n_{\mathrm{tr}} \rceil$ largest entries of v; then

$$\rho_{\mathrm{qry}} = \frac{1}{n_{\mathrm{te}}} \sum_{j=1}^{n_{\mathrm{te}}} \mathrm{Spearman}\big(\mathbf{I}^{\mathrm{d}}_{\cdot j}, \mathbf{I}^{\mathrm{c}}_{\cdot j}\big), \qquad R_{\mathrm{qry@5\%}} = \frac{1}{n_{\mathrm{te}}} \sum_{j=1}^{n_{\mathrm{te}}} \frac{\big|T_k(\mathbf{I}^{\mathrm{d}}_{\cdot j}) \cap T_k(\mathbf{I}^{\mathrm{c}}_{\cdot j})\big|}{k}.$$

Baselines. We group baselines by constraints imposed on compressed weights: (i) **Pure pruning**: `SparseGPT`, `ALPS`, and `LLM-Surgeon`; (ii) **Pure low-rank**: `SVD-LLM(W)`, `IPSVD`, and `IProX`; and (iii) **Sparse + low-rank** $(\mathbf{S} + \mathbf{L})$: `OATS`, `3BASiL`. IPSVD is the closed-form solution of the weighted SVD in the $\mathbf{L}$-update, while IProX further adds a distillation stage. For $(\mathbf{S}+\mathbf{L})$ runs, we use rank $r_\ell = \lfloor \eta(1-c)\, m_\ell n_\ell / (m_\ell + n_\ell) \rfloor$ for target compression c, where m_ℓ and n_ℓ are the layer dimensions and η is the fraction of retained parameters allocated to low rank; we use $\eta = 0.3$, a value recommended by OATS. Additional implementation details are provided in Section C.

3.2 One-shot Compression Results

We calibrate all methods on 128 sequences of length 2048 from C4's first shard and evaluate the four IF metrics on $n_{\mathrm{tr}} = 1000$ separate C4 training samples and $n_{\mathrm{te}} = 100$ WikiText-2 test samples, each of length 512. We examine sensitivity to calibration-set size in Section D.3 and curvature-factor damping in Table 3. Tables 1, 9, 11 and 12 report results for Llama-3.2-1B, Qwen3-4B, Llama-3.2-3B, and Llama-3-8B, respectively. At 50% compression on Llama-3.2-1B, `CACTUS` reduces the C4 perplexity (PPL) gap to dense by 28% over the closest $(\mathbf{S} + \mathbf{L})$ method and 24% over LLM-Surgeon(5sh), with `CACTUS`-prune running $>15\times$ faster than the latter (Table 1) while using one L40 GPU (againt 2 A100 GPUs for LLM-Surgeon). It also retains substantially better downstream model quality than IProX's low-rank proxy, which achieves 102.61 WikiTest-2 perplexity, against 14.38 for `CACTUS`-prune. Figure 2 shows the C4 perplexity comparison on Qwen3-4B at 50% compression. Under the hardware-friendly $(2{:}4{+}r256)$ configuration ($1.32\times$ inference speedup and 29% memory reduction on Llama-3-8B with specialized kernels by Mozaffari et al. (2024)), `CACTUS`-SLR improves per-query TracIn retention by 38% over the closest competitor (Figure 3). These results support important joint improvements in model quality and IF preservation; we next examine downstream data selection and integration with LoGRA projections.

Discussion. Better perplexity need not imply better gradient preservation. At 50% Qwen3-4B compression, 3BASiL improves WikiText-2 perplexity over `CACTUS`-prune (15.34 vs. 16.27), but has lower dense-gradient cosine alignment (0.743 vs. 0.762; Table 8).

3.3 Data Selection Results

Having established the large gains of `CACTUS` in terms of efficiency compared to LLM-Surgeon as well as utility and IF preservation compared to layerwise reconstruction pruning methods (e.g., SparseGPT), we now evaluate whether IF preservation also benefits downstream data selection using the LESS (Xia et al., 2024) pipeline on BIG-Bench Hard (BBH) (Suzgun et al., 2023). For Llama-2-7B and Llama-3.2-3B, we compare the pruning methods SparseGPT, ALPS, and `CACTUS`-prune at matched layerwise budgets, using dense-model and random selection as references. Each scoring model uses the same LESS procedure, with optimizer-aware gradients and TRAK projections, to select 5% of the $270,679$ example LESS instruction-tuning mixture comprising FLAN-v2, CoT, Dolly, and OpenAssistant.

Method	C4 PPL (↓)	WT-2 PPL (↓)	TracIn (↑) $\cos_{tr}$	TracIn (↑) $\cos_{te}$	TracIn (↑) ρ_{qry}	TracIn (↑) $R_{qry@5\%}$	KFAC (↑) $\cos_{tr}$	KFAC (↑) $\cos_{te}$	KFAC (↑) ρ_{qry}	KFAC (↑) $R_{qry@5\%}$	Compress Runtime
Dense	14.01	9.75	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	—
30% compression											
Pure pruning											
SparseGPT	15.94	10.92	0.737	0.730	0.524	0.293	0.689	0.690	0.429	0.277	10m 24s
ALPS	15.53	10.71	0.773	0.764	0.580	0.347	0.729	0.724	0.482	0.297	14m 48s
LLM-Surgeon(5sh)	15.18	**10.48**	0.817	0.802	0.644	0.391	0.772	0.762	0.540	0.319	4h 57m†
CACTUS-prune	**15.03**	10.49	**0.840**	**0.824**	**0.685**	**0.428**	**0.794**	**0.784**	**0.570**	**0.337**	15m 22s
Pure low-rank											
SVD-LLM(W)	135.87	184.95	0.029	0.029	0.013	0.047	0.012	0.010	0.004	0.046	18m 15s
IPSVD	72.01	91.87	0.098	0.096	0.028	0.050	0.050	0.051	0.005	0.051	18m 54s
IProX	**53.05**	**81.46**	**0.173**	**0.171**	**0.067**	**0.094**	**0.097**	**0.095**	0.015	**0.081**	36m 41s
$(\mathbf{S}+\mathbf{L})$											
OATS	15.77	10.78	0.754	0.740	0.542	0.315	0.713	0.703	0.458	0.275	36m 46s
3BASiL	14.92	10.27	0.845	0.835	0.678	0.428	0.814	0.810	0.609	0.388	50m 07s
CACTUS-SLR	**14.64**	**10.12**	**0.892**	**0.875**	**0.766**	**0.525**	**0.862**	**0.849**	**0.687**	**0.447**	59m 31s
50% compression											
Pure pruning											
SparseGPT	26.54	19.61	0.266	0.267	0.036	0.083	0.235	0.235	0.028	0.083	10m 54s
ALPS	22.20	16.44	0.395	0.394	0.130	0.117	0.336	0.333	0.090	0.120	14m 51s
LLM-Surgeon(5sh)	21.39	15.04	0.451	0.445	0.184	0.130	0.375	0.371	0.118	0.126	5h 18m†
CACTUS-prune	**19.60**	**14.38**	**0.524**	**0.512**	**0.252**	**0.152**	**0.444**	**0.434**	**0.171**	**0.136**	16m 26s
Pure low-rank											
SVD-LLM(W)	403.18	632.93	0.016	0.015	0.018	0.072	0.004	0.005	0.001	0.060	18m 03s
IPSVD	198.39	322.80	0.042	0.040	0.019	0.060	0.012	0.012	0.004	0.048	18m 52s
IProX	**68.82**	**102.61**	**0.151**	**0.145**	**0.075**	**0.090**	**0.060**	**0.060**	0.009	**0.078**	34m 28s
$(\mathbf{S}+\mathbf{L})$											
OATS	29.29	20.47	0.209	0.214	0.036	0.070	0.179	0.185	0.017	0.057	35m 44s
3BASiL	19.98	14.46	0.475	0.458	0.227	0.154	0.409	0.396	0.140	0.137	43m 06s
CACTUS-SLR	**18.27**	**13.02**	**0.585**	**0.569**	**0.327**	**0.199**	**0.505**	**0.493**	**0.219**	**0.163**	46m 51s

Table 1: One-shot compression of Llama-3.2-1B at 30% and 50% compression ratio. **Bold** = best within its constraint set; **underline** = best overall across all nine methods. †LLM-Surgeon uses two NVIDIA A100. All other methods use one L40 GPU. More results are deferred to Appendix D.

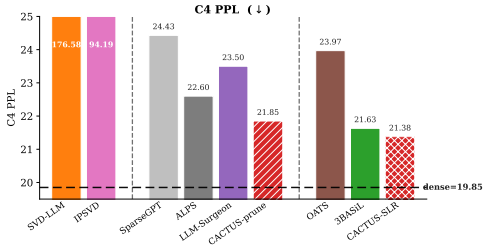


Figure 2: Qwen3-4B at matched 50% compression.

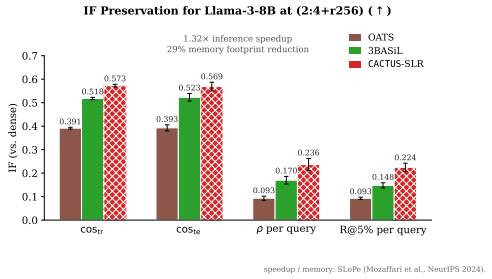


Figure 3: Llama-3-8B at (2:4+r256).

We then fine-tune a fresh dense model on each subset using the same training recipe as LESS and evaluate exact match (EM) on 1080 held-out examples across 27 BBH tasks with 3-shot chain-of-thought (CoT) prompting. Additional setup details are deferred to Section C. CACTUS achieves the highest score-rank correlation 0.638 against the closest competitor 0.594 and selected-set overlap with dense-selection at 47.20% compared with 44.07% for the better competitor. It also achieves the best BBH EM on Llama-3.2-3B 49.44 compared with 47.22 for ALPS.

Selector	Llama-2-7B ρ_{sel} ↑	Llama-2-7B Overlap ↑	Llama-2-7B EM ↑	Llama-3.2-3B ρ_{sel} ↑	Llama-3.2-3B Overlap ↑	Llama-3.2-3B EM ↑
Dense LESS	1.000	100.00	42.59	1.000	100.00	49.54
CACTUS-prune	**0.638**	**47.20**	41.57	**0.678**	**54.13**	**49.44**
ALPS	0.594	44.07	40.46	0.621	52.24	47.22
SparseGPT	0.587	39.70	**41.67**	0.598	49.60	46.02
Random 5%	—	4.45	39.81	—	4.94	48.89

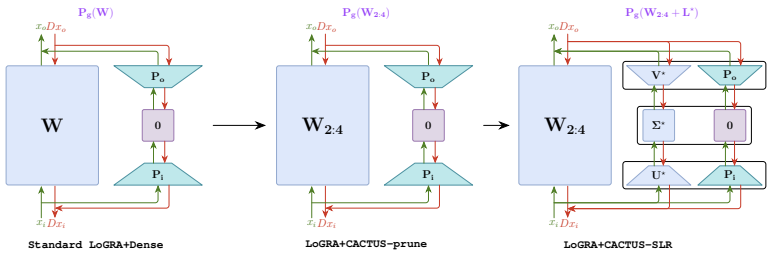
Table 2: LESS selection. ρ_{sel}: Spearman correlation of aggregate scores. Overlap: selected examples shared with dense selected data. EM: BBH accuracy (%).

3.4 LoGRA Integration

In the previous subsections, we examined the performance of CACTUS using TRAK to sketch full per-sample gradients to $d = 8192$ dimensions. We now examine its compatibility with another gradient projection strategy: Low-rank Gradient Approximation (LoGRA) (Choe et al., 2025). Lo-

GRA projects layerwise activations and backpropagated gradients without materializing the full per-sample gradient. It attaches a random rank-r probe $(\mathbf{P}_o, \mathbf{P}_i)$ to each tracked linear layer and records a projected per-sample gradient $\mathbf{\Phi}_\ell(x)=(\mathbf{P}_o^\top g_{x,\ell})(\mathbf{P}_i x_\ell)^\top \in \mathbb{R}^{r\times r}$, where x_ℓ and $g_{x,\ell}$ are the layer input and backpropagated gradient for sample x, and $\mathbf{P}_i$ and $\mathbf{P}_o$ are the fixed input and output projection matrices. Recall that the full per-sample gradient of the layer is given by $g_{x,\ell}x_\ell^\top$.

We treat Dense+LoGRA influence scores as the oracle and compare compressed models (also using the same LoGRA projection probes) against this reference. All rows use the same rank-64 random LoGRA factors on the MLP linears (default LoGRA setting). The **(middle)** panel shows the 2:4 pruning-only case, where the layer is compressed and then we append the same rank-r probe as the dense version of the **(left)** panel. For sparse-plus-low-rank models, we incorporate the low-rank correction into the LoGRA adapter while preserving exactly the same gradient projection used for the dense and pruned models. This allows a direct comparison under identical projection settings.



The **(right)** panel illustrates this construction for CACTUS-SLR, whose deployed weight is $\mathbf{S}^\star+\mathbf{L}^\star$. We decompose $\mathbf{L}^\star$ using SVD and concatenate its factors with the random LoGRA probes, widening the adapter to rank $2r$. The augmented branch contributes exactly $\mathbf{L}^\star$ to the forward pass, so the complete layer still computes with $\mathbf{S}^\star+\mathbf{L}^\star$; the random probe contributes zero to the output. After backpropagation, we obtain the per-sample gradient with respect to the adapter's central $2r \times 2r$ matrix and retain only the $r \times r$ block associated with the original random LoGRA probe. This block is exactly the standard rank-r LoGRA gradient evaluated on the deployed $(\mathbf{S} + \mathbf{L})$ model. On Qwen3-4B, CACTUS-SLR improves all LoGRA-projected IF preservation metrics over pruning procedures, reaching a per-query 0.804 Spearman and 0.597 recall at 5%. CACTUS-prune remains competitive with both LLM-Surgeon variants while running over $10\times$ and $50\times$ faster than the 1-shot and 5-shot variants, respectively.

Method MLP-Layers	$\cos_{\text{tr}}$	$\cos_{\text{te}}$	ρ_{qry}	$R_{\text{qry@5\%}}$	Compression
LoGRA + Dense (oracle)	1.000	1.000	1.000	1.000	—
LoGRA + SparseGPT	0.593	0.489	0.628	0.398	19m 47s
LoGRA + LLM-Surgeon(1sh)	0.630	0.557	0.708	0.445	4h 02m
LoGRA + LLM-Surgeon(5sh)	0.635	0.570	0.706	0.499	19h 55m
LoGRA + CACTUS-prune	0.641	0.567	0.700	0.489	22m 36s
LoGRA + CACTUS-SLR	**0.711**	**0.639**	**0.804**	**0.597**	1h 33m

Figure 4: Influence Function preservation metrics for Qwen3-4B with MLP layers pruned using 2:4 sparsity.

4 Related Work

One-shot pruning and low-rank decompositions for LLMs. Post-training one-shot compression is a training-free paradigm popularized by SparseGPT (Frantar and Alistarh, 2023), which compresses a pretrained LLM layer-by-layer on a small calibration set by minimizing the layerwise reconstruction error $\|\mathbf{X}\mathbf{W} - \mathbf{X}\widehat{\mathbf{W}}\|_F^2$ subject to a structural constraint on $\widehat{\mathbf{W}}$; methods differ along two axes: the constraint family and the per-layer solver. *Pure sparsity*: SparseGPT (Frantar and Alistarh, 2023) uses a sequential column-based sparsification scheme: after sparsifying one column, the remaining weights are updated to compensate for the incurred error, while ALPS (Meng et al., 2024) formulates the same problem as a constrained quadratic problem solved by the Alternating Direction Method of Multipliers (ADMM) (Boyd et al., 2011). *Pure low-rank*: SVD-LLM(W) (Wang et al., 2025) and ASVD (Yuan et al., 2025) use a single whitened Singular Value Decomposition (SVD) that is the closed-form minimizer of the layerwise reconstruction error. *Sparse + low-rank* $(\mathbf{S} + \mathbf{L})$: OATS (Zhang and Papyan, 2025) and Hassle-free (Makni et al., 2025) alternate a sparse fitting step with a low-rank projection step; OATS scores weights using a per-row activation-norm criterion whereas Hassle-free uses the full input correlation matrix. 3BASiL (Makni et al., 2026) runs a multi-block ADMM that jointly minimizes the sparse and low-rank components simultane-

ously. Our method CACTUS differs from all these layerwise reconstruction algorithms by incorporating downstream gradients in its objective, making it curvature-aware rather than *gradient-blind*. This is best illustrated in Figure 1.

Curvature-Aware Compression. A complementary thread incorporates downstream sensitivity into model compression. LLM-Surgeon (van der Ouderaa et al., 2024) and IProX (Chen et al., 2026) arrive at the curvature-aware objective $\|\mathbf{X}(\mathbf{W} - \widehat{\mathbf{W}})\mathbf{A}^\top\|_F^2$ from different motivations: LLM-Surgeon approximates the change in model loss, whereas IProX seeks to preserve gradient information for influence estimation (Section A and Figure 5). LLM-Surgeon targets pure pruning through curvature-based weight selection and correlated corrections over multiple rounds; IProX constructs a pure low-rank proxy through a weighted SVD followed by gradient-alignment fine-tuning. YAQA (Tseng et al., 2026a) is a related approach which applies downstream curvature to weight-only quantization using Kronecker-factored sketches of the Hessian. We make explicit the connection between model compression and influence-function preservation through this shared curvature-aware objective. CACTUS optimizes it for pruning and $(\mathbf{S}+\mathbf{L})$ using multi-block ADMM with efficient subroutines: a closed-form Sylvester solve in cached eigenbases, sparsity projections, and a randomized-SVD-based low-rank approximation (Section 2.2). On Qwen3-4B at 50% compression, CACTUS-prune is over $28\times$ faster than five-shot LLM-Surgeon, using one L40 rather than two A100 GPUs (Table 8 and Section C). Our pruning and $(\mathbf{S} + \mathbf{L})$ representations also achieve substantially lower perplexity than IProX's aligned low-rank proxy (Table 6). YAQA instead targets quantization; its curvature-based error feedback is conceptually closer to LLM-Surgeon's weight compensation than to our ADMM splitting.

Efficient Influence Function Estimation. Influence functions, rooted in robust statistics (Hampel, 1974) and introduced to deep learning by Koh and Liang (2017), estimate a training example's effect via $\mathcal{I}(z, z') = \nabla\ell(z')^\top \mathbf{H}^{-1}\nabla\ell(z)$, supporting data valuation, instruction tuning, and attribution (Pruthi et al., 2020; Xia et al., 2024; Choe et al., 2025; Grosse et al., 2023; Chen et al., 2026). At LLM scale, both the per-sample gradient $\nabla\ell(z)$ and the inverse-Hessian-vector product $\mathbf{H}^{-1}\nabla\ell(z)$ are prohibitively expensive to compute and store exactly. We focus on two broadly complementary directions that make influence estimation tractable. **(i) Compressed-model proxies** compute influence on smaller surrogates designed to preserve the target model's gradients. IProX (Chen et al., 2026) uses an influence-preserving SVD followed by gradient-alignment fine-tuning. CACTUS extends compressed-model proxies to pruning and $(\mathbf{S} + \mathbf{L})$ through the same curvature-aware objective. **(ii) Gradient projection** maps per-sample gradients to compact vectors for efficient similarity computation. TRAK (Park et al., 2023) uses a *data-free* Johnson Lindenstrauss (JL) sketch, while Arnoldi-IF (Schioppa et al., 2022) uses a *data-dependent* approximation to the leading Hessian eigenspace. Low-rank Gradient Approximation (LoGRA) (Choe et al., 2025) exploits the per-layer gradient structure $\mathrm{vec}(\mathcal{D}\mathbf{W}) = \sum_t x_{i,t} \otimes \mathcal{D}x_{o,t}$ through a matching projection $P = P_i \otimes P_o$, implemented as small low-rank adaptation (LoRA) add-on layers. Our work primarily contributes to direction **(i)**: where IProX is restricted to pure low-rank, CACTUS delivers pruning and $(\mathbf{S}+\mathbf{L})$ proxies with improved utility-compression tradeoffs. We also evaluate compatibility with both TRAK's random projections and LoGRA's architecture-aware projections from direction **(ii)**.

5 Conclusion and Limitations

In this paper, we first highlight the connection between downstream utility and influence preservation through the curvature-aware objective introduced in Equation (1). Our new algorithm CACTUS makes optimizing this objective practical for modern LLM pruning as well as $(\mathbf{S} + \mathbf{L})$ decomposition tasks. It runs at a substantially lower cost compared to prior state-of-the-art curvature-aware pruning methods. Across various compression settings, model architectures, and gradient projection strategies, CACTUS consistently improves downstream utility while retaining gradient fidelity.

Limitations. In this paper, we focus on pruning and sparse-plus-low-rank $(\mathbf{S} + \mathbf{L})$ compression. Extending our ADMM framework to quantization and mixed quantization-sparsity settings are directions for future work, especially when reparameterizations (e.g., rotations) change gradient geometry. We also only consider uniform compression budgets; it would be interesting to design specialized methods that can allocate sparsity and rank non-uniformly across the entire LLM layers. Additionally, we do not study post-compression refinements like full-model or LoRA-style finetuning. These may trade off deployment quality and gradient alignment fidelity in different ways. We leave these interesting follow-up research directions for future work.

AI USE STATEMENT

Generative AI assisted with code development, manuscript revision and polishing, and figure plotting and formatting. The authors are responsible for the final content and verification of the reported results.

REPRODUCIBILITY STATEMENT

The experimental section and appendix specify model families, pruning formats, calibration data, hyperparameters, and evaluation protocols. We will release the code of our curvature-aware algorithm and experimental setups after the paper is reviewed.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122, 2011.

Sirui Chen, Yunzhe Qi, Mengting Ai, Yifan Sun, Ruizhong Qiu, Jiaru Zou, and Jingrui He. Influence-preserving proxies for gradient-based data selection in LLM finetuning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=PDNpRLxDlI.

Sang Keun Choe, Hwijeen Ahn, Juhan Bae, Kewen Zhao, Youngseog Chung, Adithya Pratapa, Willie Neiswanger, Emma Strubell, Teruko Mitamura, Jeff Schneider, Eduard Hovy, Roger Baker Grosse, and Eric P. Xing. What is your data worth to GPT? LLM-scale data valuation with influence functions. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=zPKeJAEo27.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Elias Frantar and Dan Alistarh. SparseGPT: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning*, pages 10323–10337. PMLR, 2023.

Gemini Team Google. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Roger Grosse, Juhan Bae, Cem Anil, Nelson Elhage, Alex Tamkin, Amirhossein Tajdini, Benoit Steiner, Dustin Li, Esin Durmus, Ethan Perez, Evan Hubinger, Kamilė Lukošiūtė, Karina Nguyen, Nicholas Joseph, Sam McCandlish, Jared Kaplan, and Samuel R. Bowman. Studying large language model generalization with influence functions. *arXiv preprint arXiv:2308.03296*, 2023.

Frank R Hampel. The influence curve and its role in robust estimation. *Journal of the american statistical association*, 69(346):383–393, 1974.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning*, pages 1885–1894. PMLR, 2017.

Mehdi Makni, Kayhan Behdin, Zheng Xu, Natalia Ponomareva, and Rahul Mazumder. A unified framework for sparse plus low-rank matrix decomposition for LLMs. In *The Second Conference on Parsimony and Learning (Proceedings Track)*, 2025. URL https://openreview.net/forum?id=hyN75SAJTI.

Mehdi Makni, Xiang Meng, and Rahul Mazumder. 3basil: An algorithmic framework for sparse plus low-rank compression of llms. *arXiv preprint arXiv:2603.01376*, 2026.

Mitchell Marcus, Grace Kim, Mary Ann Marcinkiewicz, Robert MacIntyre, Ann Bies, Mark Ferguson, Karen Katz, and Britta Schasberger. The penn treebank: Annotating predicate argument structure. In *Proceedings of the Workshop on Human Language Technology*, HLT '94, page 114–119, USA, 1994. Association for Computational Linguistics. ISBN 1558603573. doi: 10.3115/1075812.1075835. URL https://doi.org/10.3115/1075812.1075835.

James Martens and Roger Grosse. Optimizing neural networks with kronecker-factored approximate curvature. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 2408–2417. PMLR, 2015. URL https://proceedings.mlr.press/v37/martens15.html.

Xiang Meng, Kayhan Behdin, Haoyue Wang, and Rahul Mazumder. Alps: Improved optimization for highly sparse one-shot pruning for large language models. In *Annual Conference on Neural Information Processing Systems, NeurIPS*, 2024.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=Byj72udxe.

Bruno Mlodozeniec, Runa Eschenhagen, Juhan Bae, Alexander Immer, David Krueger, and Richard Turner. Influence functions for scalable data attribution in diffusion models. *arXiv preprint arXiv:2410.13850*, 2024.

Mohammad Mozaffari, Amir Yazdanbakhsh, Zhao Zhang, and Maryam Mehri Dehnavi. Slope: Double-pruned sparse plus lazy low-rank adapter pretraining of llms. *arXiv preprint arXiv:2405.16325*, 2024.

Sung Min Park, Kristian Georgiev, Andrew Ilyas, Guillaume Leclerc, and Aleksander Madry. Trak: Attributing model behavior at scale. In *International Conference on Machine Learning*, pages 27074–27113. PMLR, 2023.

Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. Estimating training data influence by tracing gradient descent. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 19920–19930. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/e6385d39ec9394f2f3a354d9d2b88eec-Paper.pdf.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. URL http://jmlr.org/papers/v21/20-074.html.

Andrea Schioppa, Polina Zablotskaia, David Vilar, and Artem Sokolov. Scaling up influence functions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 8179–8186, 2022.

Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. A simple and effective pruning approach for large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=PxoFut3dWW.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, et al. Challenging BIG-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13003–13051, 2023.

James Joseph Sylvester. *The Collected Mathematical Papers of James Joseph Sylvester...*, volume 3. University Press, 1909.

Albert Tseng, Zhaofeng Sun, and Christopher De Sa. Model-preserving adaptive rounding. In *Forty-third International Conference on Machine Learning*, 2026a. URL https://openreview.net/forum?id=PKFilPWjMI.

Albert Tseng, Zhaofeng Sun, and Christopher De Sa. Model-preserving adaptive rounding. In *Forty-third International Conference on Machine Learning*, 2026b. URL https://openreview.net/forum?id=PKFilPWjMI.

Tycho F. A. van der Ouderaa, Markus Nagel, Mart Van Baalen, and Tijmen Blankevoort. The LLM surgeon. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=DYIIRgwg2i.

Xin Wang, Yu Zheng, Zhongwei Wan, and Mi Zhang. SVD-LLM: Truncation-aware singular value decomposition for large language model compression. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=LNYIUouhdt.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. LESS: Selecting influential data for targeted instruction tuning. In *International Conference on Machine Learning*, 2024.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Zhihang Yuan, Yuzhang Shang, Yue Song, Dawei Yang, Qiang Wu, Yan Yan, and Guangyu Sun. ASVD: Activation-aware singular value decomposition for compressing large language models, 2025. URL https://openreview.net/forum?id=HyPofygOCT.

Stephen Zhang and Vardan Papyan. OATS: Outlier-aware pruning through sparse and low rank decomposition. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=DLDuVbxORA.

Aojun Zhou, Yukun Ma, Junnan Zhu, Jianbo Liu, Zhijie Zhang, Kun Yuan, Wenxiu Sun, and Hongsheng Li. Learning n:m fine-grained structured sparse neural networks from scratch. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=K9bw7vqp_s.

A THE LLM-SURGEON AND IPROX DERIVATIONS

Figure 5 connects the two motivations for Equation (1). For a layer perturbation $\Delta W = \widehat{\mathbf{W}} - \mathbf{W}$, let x be its input, $\Delta y = (\Delta W)^\top x$ its output perturbation, and $g = \nabla_y \ell$ the downstream gradient of the per-example loss ℓ; $\mathcal{L} = \mathbb{E}[\ell]$ denotes the expected loss. The diagram makes each path's approximations explicit.

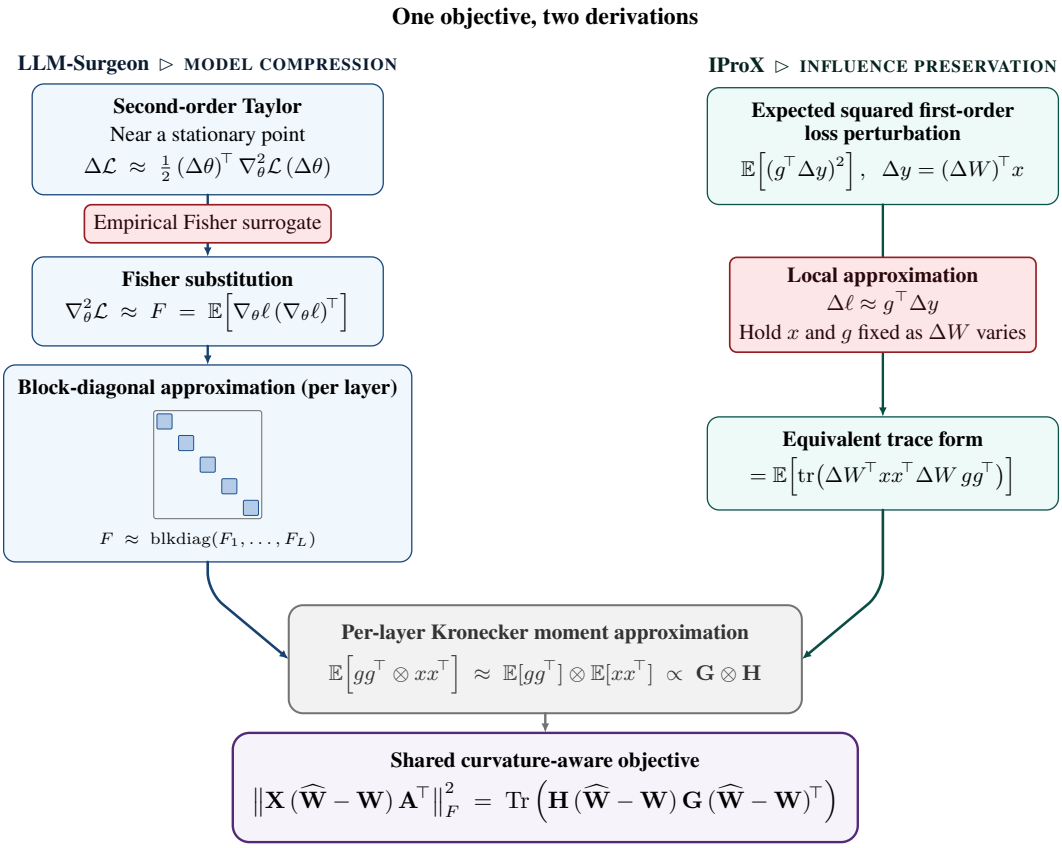


Figure 5: **One objective, two derivations.** LLM-Surgeon (van der Ouderaa et al., 2024) uses a second-order loss approximation, an empirical Fisher surrogate, and layerwise blocks. IProX (Chen et al., 2026) starts from a local first-order loss perturbation with fixed activations and downstream gradients. Both paths use a Kronecker moment approximation to recover Equation (1), up to fixed normalization factors. The Kronecker order follows column-wise vectorization of ΔW; $\mathbf{H} = \mathbf{X}^\top \mathbf{X}$ and $\mathbf{G} = \mathbf{A}^\top \mathbf{A}$ are the empirical Gram matrices.

B CONVERGENCE OF CACTUS

The proof follows the structure of the 3BASiL analysis (Makni et al., 2026, Appendix A). The two-sided objective changes the operator to $\mathcal{A}(\mathbf{M}) = 2\mathbf{HMG}$. The projection on sparsity set argument and the summable-penalty argument remain the same; we give their adapted forms below.

We write iteration indices as subscripts and set $\mathbf{Y}_t = \rho_t \mathbf{V}_t$ for the unscaled dual variable. Since $\mathbf{H}, \mathbf{G} \succ 0$, the operator $\mathcal{A}$ is positive definite and satisfies

$$\|\mathcal{A}(\mathbf{M})\|_F \le h\|\mathbf{M}\|_F, \qquad \|(\mathcal{A} + \rho\,\mathrm{Id})^{-1}(\mathbf{M})\|_F \le \frac{\|\mathbf{M}\|_F}{\rho}, \qquad h = 2\|\mathbf{H}\|_2\|\mathbf{G}\|_2.$$

Here Id is the identity operator on matrices. The acceptance rule in Equation (8) gives

$$\|\mathbf{L}_{t+1} - \mathbf{L}_t\|_F \le C_L\|\mathbf{S}_{t+1} - \mathbf{S}_t\|_F. \tag{9}$$

This bound holds for both accepted and rejected candidates by construction. For pure pruning, $\mathbf{L}_t = \mathbf{0}$ and the condition is automatic.

Lemma B.1 (Sparse and dual updates control)**.** *Let $\mathbf{F}_t = \mathcal{A}(\mathbf{S}_{t+1} + \mathbf{L}_t - \mathbf{W})$ be the smooth gradient at the $\mathbf{S}$-update. Then*

$$\|\mathbf{\Delta}_{t+1} - \mathbf{\Delta}_t\|_F \le \frac{2\|\mathbf{F}_t\|_F}{\rho_t}, \qquad \|\mathbf{Y}_{t+1}\|_F \le \|\mathbf{F}_t\|_F. \tag{10}$$

Proof. This is the projection comparison used in 3BASiL (Makni et al., 2026, Lemma A.2), applied to the two-sided gradient. Write $\mathbf{P}_t = \mathbf{S}_{t+1} + \mathbf{V}_t$. The smooth-step optimality condition gives

$$\mathbf{F}_t + \rho_t(\mathbf{S}_{t+1} - \mathbf{\Delta}_t + \mathbf{V}_t) = \mathbf{0}, \qquad \mathbf{P}_t = \mathbf{\Delta}_t - \mathbf{F}_t/\rho_t. \tag{11}$$

The previous sparse iterate $\mathbf{\Delta}_t$ is feasible, so

$$\|\mathbf{P}_t - \mathbf{\Delta}_{t+1}\|_F \le \|\mathbf{P}_t - \mathbf{\Delta}_t\|_F = \|\mathbf{F}_t\|_F/\rho_t.$$

The triangle inequality gives the sparse-movement bound. The rescaled dual update gives $\mathbf{Y}_{t+1} = \rho_t(\mathbf{P}_t - \mathbf{\Delta}_{t+1})$, proving the second bound. This argument uses only projection optimality and applies to both unstructured and $n{:}m$ sparsity. □

Lemma B.2 (Bounded iterates)**.** *Under the assumptions of Theorem 1, the sequences $\mathbf{S}_t$, $\mathbf{\Delta}_t$, $\mathbf{L}_t$, $\mathbf{Y}_t$, and $\mathbf{F}_t$ are bounded.*

Proof. As in the scalar recurrence underlying 3BASiL (Makni et al., 2026, Lemma A.3), it suffices to bound the growth of a running maximum. Define

$$M_t = 1 + \|\mathbf{W}\|_F + \max_{0\le j\le t}\big(\|\mathbf{S}_j\|_F + \|\mathbf{\Delta}_j\|_F + \|\mathbf{L}_j\|_F\big).$$

For $t \ge 2$, Lemma B.1 gives $\|\mathbf{Y}_t\|_F, \|\mathbf{Y}_{t-1}\|_F \le 3hM_t$. The smooth solve and the preceding dual update yield, respectively,

$$\begin{aligned}\mathbf{S}_{t+1} - \mathbf{\Delta}_t &= -(\mathcal{A} + \rho_t\mathrm{Id})^{-1}\big[\mathbf{Y}_t + \mathcal{A}(\mathbf{\Delta}_t + \mathbf{L}_t - \mathbf{W})\big],\\ \mathbf{S}_t - \mathbf{\Delta}_t &= (\mathbf{Y}_t - \mathbf{Y}_{t-1})/\rho_{t-1}.\end{aligned}$$

Their norms are at most $6hM_t/\rho_t$ and $6hM_t/\rho_{t-1}$. Since the penalties are non-decreasing, the triangle inequality and Equation (9) give

$$\|\mathbf{S}_{t+1} - \mathbf{S}_t\|_F \le \frac{12hM_t}{\rho_{t-1}}, \qquad \|\mathbf{L}_{t+1} - \mathbf{L}_t\|_F \le \frac{12C_LhM_t}{\rho_{t-1}}. \tag{12}$$

Also, the first bound above implies $\|\mathbf{F}_t\|_F \le aM_t$, where $a = h(3 + 6h/\rho_0)$. Combining these bounds with the sparse-movement bound in Lemma B.1 gives

$$M_{t+1} \le \left(1 + \frac{B}{\rho_{t-1}}\right) M_t, \qquad B = 12h(1 + C_L) + 2a.$$

Iterating this scalar inequality and using $1 + x \le e^x$ gives

$$M_t \le M_2 \exp\left(B \sum_{j=1}^{\infty} \rho_j^{-1}\right) =: M_\star < \infty.$$

The initial iterates are finite because the smooth solves are nonsingular and the movement rule bounds every accepted low-rank update. Thus the primal iterates are bounded, and the bounds on $\mathbf{F}_t$ and $\mathbf{Y}_t$ follow as well. □

Proof of Theorem 1. By Lemma B.2, let $K < \infty$ such that $\|\mathbf{F}_t\|_F, \|\mathbf{Y}_t\|_F \le K$ for all t. The dual update and Lemma B.1 now give

$$\|\mathbf{S}_{t+1} - \mathbf{\Delta}_{t+1}\|_F \le \frac{2K}{\rho_t}, \qquad \|\mathbf{\Delta}_{t+1} - \mathbf{\Delta}_t\|_F \le \frac{2K}{\rho_t}.$$

For $t \ge 1$, we have:

$$\begin{aligned}\|\mathbf{S}_{t+1} - \mathbf{S}_t\|_F &\le \|\mathbf{S}_{t+1} - \mathbf{\Delta}_{t+1}\|_F + \|\mathbf{\Delta}_{t+1} - \mathbf{\Delta}_t\|_F + \|\mathbf{\Delta}_t - \mathbf{S}_t\|_F \\ &\le \frac{6K}{\rho_{t-1}}, \qquad \|\mathbf{L}_{t+1} - \mathbf{L}_t\|_F \le \frac{6C_L K}{\rho_{t-1}}.\end{aligned}$$

Since $\sum_t \rho_t^{-1} < \infty$, all three primal sequences have finite total movement and hence converge. The consensus residual tends to zero, so $\mathbf{S}_t$ and $\mathbf{\Delta}_t$ have the same limit. Finally, every $\mathbf{\Delta}_t$ satisfies the sparsity constraint and every $\mathbf{L}_t$ has rank at most r. Both constraint sets are closed, so their limits retain these properties. □

C Additional Experimental Protocol Details

Compute environment and timing. Unless otherwise specified, compression runs use an Intel Xeon Gold 6248 machine with 16 CPU cores and a single GPU: L40/A100/H100. Runtime entries marked with † are LLM-Surgeon runs on Llama-3.2-1B and Qwen3-4B using two NVIDIA A100 GPUs. For Qwen3-4B, `CACTUS` and the other one-shot methods use a single L40 GPU. Compression runtime is measured end-to-end from covariance statistic estimation until algorithm compression finish. In particular, we only collect $\mathbf{G}$ for curvature-aware algorithms: LLM-Surgeon, IPSVD, `CACTUS`-prune, `CACTUS`-SLR. For IF preservation experiments, we allocate between 200 and 500 GB of disk storage per experiment and use the Rademacher JL sketch provided by `fast_jl` CUDA kernel (Park et al., 2023).

`CACTUS` implementation. Before eigendecomposition, we apply separate ridge damping to the activation and gradient factors:

$$\mathbf{H} \leftarrow \mathbf{H} + \lambda_H \,\mathrm{mean}(\mathrm{diag}(\mathbf{H}))\mathbf{I},$$
$$\mathbf{G} \leftarrow \mathbf{G} + \lambda_G \,\mathrm{mean}(\mathrm{diag}(\mathbf{G}))\mathbf{I}.$$

Here λ_H and λ_G are damping parameters. The algorithm minimizes the two-sided objective in Equation (1) using these damped factors. We present the ppl results using different ridge parameters in Table 3. Compression seems to be robust to the ridge values used. For large layers/model architectures, 0.005 can cause decomposition issues (low condition number). In all of our experiments, we use $\lambda_H = 0.01$ (same as SparseGPT) and $\lambda_G = 0.1$.

	λ_G			
λ_H	0.005	0.01	0.1	0.25
C4 perplexity (↓)				
0.005	19.729	19.647	19.448	19.449
0.01	19.719	19.596	19.432	**19.392**
0.1	19.626	19.601	19.536	19.395
0.25	19.681	19.780	19.548	19.509
WT-2 perplexity (↓)				
0.005	14.351	14.447	14.308	14.186
0.01	14.346	14.302	14.168	14.070
0.1	14.207	14.150	14.045	14.023
0.25	14.218	14.195	14.097	**13.972**

Table 3: Ridge-damping ablation for $\mathbf{H}$ and $\mathbf{G}$ using next-token cross-entropy on Llama-3.2-1B at 50% unstructured sparsity. Bold denotes the lowest perplexity within each dataset's grid.

By default, our ADMM-based algorithm `CACTUS` uses $C_F = 10$ for the low-rank movement condition (C_F denotes the threshold C_L in Algorithm 1).

C_F	Accept (%)	Reject (%)	Avg. iterations	C4 ↓	WT-2 ↓
∞	100.00	0.00	430	18.256	13.061
10 (default)	100.00	0.00	431	18.307	12.952
3	100.00	0.00	429	18.290	13.021
1	100.00	0.00	431	18.337	13.023
0.9	99.99	0.01	430	18.293	12.909
0.8	99.62	0.38	430	18.276	13.028
0.7	97.47	2.53	429	18.320	13.014
0.6	61.58	38.42	417	19.428	14.122
0.5	22.03	77.97	407	24.511	18.527
0.4	1.16	98.84	403	35.538	26.500
0.3	0.00	100.00	403	36.723	27.272
0.1	0.00	100.00	403	36.590	27.079

Table 4: Low-rank movement-threshold ablation from the rebuttal: `CACTUS`-$(\mathbf{S} + \mathbf{L})$ on Llama-3.2-1B at 50% total compression with rank-budget allocation $\eta = 0.3$. Rejection rates and iteration counts are averaged over all 112 compressed sublayers; acceptance is the complement of rejection. These historical runs used the mixed objective with coefficients $\alpha = 0.1$ and $\beta = 0.01$.

We use an initial penalty $\rho_0 = 0.1$ and update the ADMM penalty using the support-change rule from ALPS/3BASiL. Let $s_t = |\operatorname{Supp}(\mathbf{\Delta}^{(t)}) \triangle \operatorname{Supp}(\mathbf{\Delta}^{(t-m)})|$ be the symmetric difference in sparse support at the update period m, and let k be the number of retained nonzeros. The finite-run support-change schedules are summarized in Table 5. For the non-decreasing schedule analyzed in Theorem 1, a multiplier of at least $\eta_{\min} = 1.02$ at every fixed-period update gives $\rho_t \geq \rho_0 \eta_{\min}^{\lfloor t/m \rfloor}$, so $\sum_{t=0}^{\infty} 1/\rho_t \leq m\eta_{\min}/[\rho_0(\eta_{\min} - 1)] < \infty$.

Setting	Max iter.	Support-change condition	ρ multipliers
Pure pruning	100	$s_t/k > 0.1,\ s_t/k > 0.005,\ s_t > 0.5$	1.3, 1.2, 1.1
$(\mathbf{S} + \mathbf{L})$	2000	$s_t/k > 0.1,\ s_t/k > 0.005,\ s_t > 0.5$	1.1, 1.05, 1.02

Table 5: Support-change penalty multipliers for finite-run compression. At the update period m, the multiplier is selected according to which support-change condition holds: the first matching threshold sets the multiplier $\rho^{(t+1)}/\rho^{(t)}$. The $(\mathbf{S} + \mathbf{L})$ schedule is intentionally less aggressive as $(\mathbf{S} + \mathbf{L})$ decompositions are more sensitive.

Baselines. All one-shot baselines are run in the same layerwise compression framework and with the same calibration samples, except when a method explicitly includes a post-compression stage. `SparseGPT`, `ALPS`, `SVD-LLM(W)`, `OATS`, and `3BASiL` use activation-only objectives using $\mathbf{H}$, whereas `IPSVD`, `IProX`, `LLM-Surgeon`, and `CACTUS` use $\mathbf{G}$ as well. For $(\mathbf{S} + \mathbf{L})$ baselines at target compression ratio c, we match total parameter budget using the same layerwise rank allocation $r_\ell = \lfloor \eta(1 - c)m_\ell n_\ell/(m_\ell + n_\ell) \rfloor$ with $\eta = 0.3$; for named structured settings such as $(2{:}4{+}r256)$, the sparsity pattern and rank are fixed exactly as specified.

LLM-Surgeon is reported with the five-shot setting described in Section 3.1, since the one-shot unstructured pruning on Llama3.2-1B reproduction produced weak results. For 2:4 sparsity on MLP layers only, one-shot results are competitive, so we report both one-shot and five-shot results.

Compression Protocol. We estimate activation and gradient covariance factors from 128 C4 calibration sequences at sequence length 2048 (see effect of number of calibration data points at Table 14). This is a standard calibration set introduced by SparseGPT (Frantar and Alistarh, 2023) and adopted by most layerwise reconstruction algorithms like ALPS (Meng et al., 2024). We evaluate language model utility with C4 and WikiText-2 perplexity at sequence length 2048. Influence-preservation metrics use projected per-example gradients (via `fast_jl` as mentioned above) with the same seeded projection. Thus, TracIn entries are computed as $\langle \Pi g_i, \Pi g_j \rangle$ under a fixed JL sketch Π, which preserves inner products and ensures dense and compressed gradients are compared in the same projected space. The per-sample gradients are sampled from $n_{\text{tr}} = 1000$ C4 training sequences (different from calibration samples) and $n_{\text{te}} = 100$ WikiText-2 test sequences at sequence length 512.

Data Selection Protocol. We adapt LESS (Xia et al., 2024) to the released FLAN-v2, CoT, Dolly, and OpenAssistant mixture (270,679 candidates). Scorers share a 5% warm-up sample and train LoRA adapters on attention $q/k/v/o$ projections for four epochs (rank 128, scaling 512, dropout 0.1, initialization seed 0). Warm-up and target training use learning rate 2×10^{-5}, effective batch size 128, sequence length 2048, cosine decay, and 3% learning-rate warm-up.

We normalize shared 8192-dimensional TRAK projections of Adam-aware candidate updates and ordinary query gradients, then sum their inner products across four checkpoints, weighted by each epoch's mean learning rate. Scores are averaged over three queries per BBH task and maximized over 27 tasks. Each scorer selects 13,533 examples for four-epoch training of a fresh dense target; 81 validation examples determine the checkpoint. Evaluation uses 1080 disjoint test examples, greedy 3-shot CoT, and the released answer extraction and case-/punctuation-insensitive exact match.

In Table 2, Spearman compares aggregate scores on common valid candidates; overlap measures intersection with dense LESS divided by 13,533 (this corresponds to 5% of the entire dataset size). Figure 6 plots 3B scores in dense-ranked order. Training on the authors' released 7B LESS subset yields 41.30% BBH-EM.

LoGRA Integration Protocol. The setup in LoGRA Integration subsection reuses the standard one-shot calibration setup (128 C4 sequences at sequence length 2048, $\mathbf{H}$ and $\mathbf{G}$ collected with the same damping values as the one-shot compression subsection) and compresses Qwen3-4B with each method restricted to MLP linears (`gate_proj`, `up_proj`, `down_proj`). Hence, the attention $q/k/v/o$ remain dense. Sparsity is set to 2:4; the $(\mathbf{S}+\mathbf{L})$ experiment additionally adds rank-64 low-rank corrections using `CACTUS`-SLR on the same MLP layers. For `LLM-Surgeon`, we report both the 1-shot and 5-shot variants. The LoGRA influence-evaluation pipeline is identical for every row. We attach rank-64 LoGRA adapters to every tracked MLP linear (`gate_proj`, `up_proj`, and `down_proj`) of the deployed compressed model with $\mathbf{P}_o \in \mathbb{R}^{r \times d_{\text{out}}}$ and $\mathbf{P}_i \in \mathbb{R}^{r \times d_{\text{in}}}$ drawn from a single shared seed (so all methods share the *same* random subspaces and IF entries are pairwise comparable).

Concat-LoGRA construction. For `CACTUS`-SLR, we widen each tracked MLP-linear adapter to rank $2r$ by concatenating the SVD basis of that linear layer's low-rank correction denoted by $\mathbf{L}_\ell^\star = \mathbf{U}_\ell^\star \mathbf{\Sigma}_\ell^\star \mathbf{V}_\ell^{\star\top}$: $\mathbf{A}_{\text{cat},\ell} = [\,\mathbf{P}_o \,|\, \mathbf{U}_\ell^\star\,]$, $\mathbf{B}_{\text{cat},\ell} = [\,\mathbf{P}_i \,|\, \mathbf{V}_\ell^\star\,]$. Forward and backward use the full rank-$2r$ adapter with $\mathbf{E}_{\text{cat},\ell} = \text{blkdiag}(\mathbf{0}, \mathbf{\Sigma}_\ell)$; this is forward-equivalent to the deployed weight $\mathbf{S}_\ell + \mathbf{L}_\ell$ since the random block contributes zero to the forward. We capture the $2r \times 2r$ per-sample gradient and *slice the top-left* $r \times r$ *block* as the IF statistic, corresponding to a standard rank-r LoGRA gradient (of size $r \times r$), so all downstream IF analysis is identical.

D Additional Experimental Results

This appendix reports the four influence-preservation metrics introduced in Section 3.1 ($\cos_{\text{tr}}$, $\cos_{\text{te}}$, ρ_{qry}, $R_{\text{qry@5\%}}$) for all nine compression methods on Llama-3.2-1B at *three* compression levels ($30\%, 50\%, 70\%$). Each cell is annotated as mean $\pm$ $\widehat{\text{SEM}}$: the sample standard deviation divided by the square root of the number of evaluation examples. Here $n = n_{\text{tr}}$ for $\cos_{\text{tr}}$, and $n = n_{\text{te}}$ for $\cos_{\text{te}}$, ρ_{qry}, and $R_{\text{qry@5\%}}$. Bolding in both sub-tables follows the convention of Table 1 (**bold**: best within constraint family; **underline**: best overall). Runtime entries marked with † use two NVIDIA A100 GPUs; all other one-shot compression runtimes of Llama-3.2-3B and Qwen3-4B use the single L40 GPU setup described in Section C.

Method	C4 PPL (↓)	WT-2 PPL (↓)	TracIn (↑)				Compress Runtime
			$\cos_{\text{tr}}$	$\cos_{\text{te}}$	ρ_{qry}	$R_{\text{qry@5\%}}$	
Dense	14.01	9.75	1.000	1.000	1.000	1.000	—
30% compression							
Pure pruning							
SparseGPT	15.94	10.92	0.737 ± 0.005	0.730 ± 0.018	0.524 ± 0.014	0.293 ± 0.007	10m 24s
ALPS	15.53	10.71	0.773 ± 0.005	0.764 ± 0.017	0.580 ± 0.014	0.347 ± 0.007	14m 48s
LLM-Surgeon(5sh)	15.18	**10.48**	0.817 ± 0.005	0.802 ± 0.017	0.644 ± 0.015	0.391 ± 0.008	4h 57m†
`CACTUS-prune`	**15.03**	10.49	$\mathbf{0.840} \pm 0.005$	$\mathbf{0.824} \pm 0.017$	$\mathbf{0.685} \pm 0.015$	$\mathbf{0.428} \pm 0.009$	15m 22s
Pure low-rank							
SVD-LLM(W)	135.87	184.95	0.029 ± 0.001	0.029 ± 0.003	0.013 ± 0.004	0.047 ± 0.003	18m 15s
IPSVD	72.01	91.87	0.098 ± 0.002	0.096 ± 0.004	0.028 ± 0.005	0.050 ± 0.003	18m 54s
IProX	**53.05**	**81.46**	$\mathbf{0.173} \pm 0.002$	$\mathbf{0.171} \pm 0.005$	$\mathbf{0.067} \pm 0.004$	$\mathbf{0.094} \pm 0.004$	36m 41s
$(\mathbf{S} + \mathbf{L})$							
OATS	15.77	10.78	0.754 ± 0.005	0.740 ± 0.017	0.542 ± 0.014	0.315 ± 0.007	36m 46s
3BASiL	14.92	10.27	0.845 ± 0.004	0.835 ± 0.015	0.678 ± 0.015	0.428 ± 0.008	50m 07s
`CACTUS-slr`	**14.64**	**10.12**	$\mathbf{0.892} \pm 0.004$	$\mathbf{0.875} \pm 0.015$	$\mathbf{0.766} \pm 0.014$	$\mathbf{0.525} \pm 0.010$	59m 31s
50% compression							
Pure pruning							
SparseGPT	26.54	19.61	0.266 ± 0.003	0.267 ± 0.011	0.036 ± 0.005	0.083 ± 0.005	10m 54s
ALPS	22.20	16.44	0.395 ± 0.004	0.394 ± 0.012	0.130 ± 0.007	0.117 ± 0.005	14m 51s
LLM-Surgeon(5sh)	21.39	15.04	0.451 ± 0.004	0.445 ± 0.014	0.184 ± 0.008	0.130 ± 0.006	5h 18m†
`CACTUS-prune`	**19.60**	**14.38**	$\mathbf{0.524} \pm 0.005$	$\mathbf{0.512} \pm 0.016$	$\mathbf{0.252} \pm 0.010$	$\mathbf{0.152} \pm 0.006$	16m 26s
Pure low-rank							
SVD-LLM(W)	403.18	632.93	0.016 ± 0.001	0.015 ± 0.002	0.018 ± 0.004	0.072 ± 0.004	18m 03s
IPSVD	198.39	322.80	0.042 ± 0.001	0.040 ± 0.003	0.019 ± 0.004	0.060 ± 0.004	18m 52s
IProX	**68.82**	**102.61**	$\mathbf{0.151} \pm 0.002$	$\mathbf{0.145} \pm 0.005$	$\mathbf{0.075} \pm 0.004$	$\mathbf{0.090} \pm 0.005$	34m 28s
$(\mathbf{S} + \mathbf{L})$							
OATS	29.29	20.47	0.209 ± 0.003	0.214 ± 0.007	0.036 ± 0.004	0.070 ± 0.005	35m 44s
3BASiL	19.98	14.46	0.475 ± 0.004	0.458 ± 0.016	0.227 ± 0.010	0.154 ± 0.005	43m 06s
`CACTUS-slr`	**18.27**	**13.02**	$\mathbf{0.585} \pm 0.005$	$\mathbf{0.569} \pm 0.017$	$\mathbf{0.327} \pm 0.011$	$\mathbf{0.199} \pm 0.006$	46m 51s
70% compression							
Pure pruning							
SparseGPT	156.90	139.13	0.030 ± 0.001	0.030 ± 0.003	0.021 ± 0.004	0.049 ± 0.003	10m 25s
ALPS	68.21	63.47	0.065 ± 0.001	0.071 ± 0.003	0.021 ± 0.004	0.049 ± 0.003	10m 45s
LLM-Surgeon(5sh)	64.56	52.02	0.104 ± 0.002	0.107 ± 0.005	0.015 ± 0.006	0.055 ± 0.004	5h 33m†
`CACTUS-prune`	**52.19**	**41.08**	$\mathbf{0.122} \pm 0.002$	$\mathbf{0.124} \pm 0.004$	$\mathbf{0.043} \pm 0.005$	$\mathbf{0.068} \pm 0.004$	17m 09s
Pure low-rank							
SVD-LLM(W)	1081	2106	0.010 ± 0.001	0.008 ± 0.002	0.016 ± 0.004	0.075 ± 0.004	18m 59s
IPSVD	1095	1896	0.006 ± 0.001	0.006 ± 0.002	0.009 ± 0.004	0.051 ± 0.004	18m 13s
IProX	**156.30**	**293.62**	$\mathbf{0.075} \pm 0.001$	$\mathbf{0.073} \pm 0.003$	$\mathbf{0.051} \pm 0.004$	$\mathbf{0.104} \pm 0.005$	28m 18s
$(\mathbf{S} + \mathbf{L})$							
OATS	389.61	355.95	0.019 ± 0.001	0.023 ± 0.002	0.028 ± 0.004	0.062 ± 0.004	35m 20s
3BASiL	79.98	68.06	0.045 ± 0.001	0.047 ± 0.003	0.021 ± 0.004	0.048 ± 0.003	38m 01s
`CACTUS-slr`	**54.58**	**41.74**	$\mathbf{0.126} \pm 0.002$	$\mathbf{0.124} \pm 0.005$	$\mathbf{0.045} \pm 0.006$	$\mathbf{0.070} \pm 0.004$	37m 12s

Table 6: Influence-preservation results in the **TracIn** (unweighted) representation.

Method	C4 PPL (↓)	WT-2 PPL (↓)	KFAC (↑)				Compress Runtime
			$\cos_{\text{tr}}$	$\cos_{\text{te}}$	ρ_{qry}	$R_{\text{qry@5\%}}$	
Dense	14.01	9.75	1.000	1.000	1.000	1.000	—
30% compression							
Pure pruning							
SparseGPT	15.94	10.92	0.689 ± 0.005	0.690 ± 0.016	0.429 ± 0.011	0.277 ± 0.007	10m 24s
ALPS	15.53	10.71	0.729 ± 0.005	0.724 ± 0.016	0.482 ± 0.011	0.297 ± 0.008	14m 48s
LLM-Surgeon(5sh)	15.18	**10.48**	0.772 ± 0.005	0.762 ± 0.016	0.540 ± 0.012	0.319 ± 0.008	4h 57m†
CACTUS-prune	**15.03**	10.49	**0.794** ± 0.005	**0.784** ± 0.017	**0.570** ± 0.013	**0.337** ± 0.008	15m 22s
Pure low-rank							
SVD-LLM(W)	135.87	184.95	0.012 ± 0.001	0.010 ± 0.001	0.004 ± 0.003	0.046 ± 0.003	18m 15s
IPSVD	72.01	91.87	0.050 ± 0.001	0.051 ± 0.003	0.005 ± 0.004	0.051 ± 0.004	18m 54s
IProX	**53.05**	**81.46**	**0.097** ± 0.001	**0.095** ± 0.004	**0.015** ± 0.003	**0.081** ± 0.004	36m 41s
$(\mathbf{S}+\mathbf{L})$							
OATS	15.77	10.78	0.713 ± 0.004	0.703 ± 0.016	0.458 ± 0.011	0.275 ± 0.006	36m 46s
3BASiL	14.92	10.27	0.814 ± 0.004	0.810 ± 0.015	0.609 ± 0.012	0.388 ± 0.009	50m 07s
CACTUS-slr	**14.64**	**10.12**	**0.862** ± 0.004	**0.849** ± 0.014	**0.687** ± 0.012	**0.447** ± 0.010	59m 31s
50% compression							
Pure pruning							
SparseGPT	26.54	19.61	0.235 ± 0.003	0.235 ± 0.009	0.028 ± 0.004	0.083 ± 0.004	10m 54s
ALPS	22.20	16.44	0.336 ± 0.003	0.333 ± 0.010	0.090 ± 0.004	0.120 ± 0.005	14m 51s
LLM-Surgeon(5sh)	21.39	15.04	0.375 ± 0.004	0.371 ± 0.012	0.118 ± 0.005	0.126 ± 0.005	5h 18m†
CACTUS-prune	**19.60**	**14.38**	**0.444** ± 0.004	**0.434** ± 0.014	**0.171** ± 0.007	**0.136** ± 0.005	16m 26s
Pure low-rank							
SVD-LLM(W)	403.18	632.93	0.004 ± 0.001	0.005 ± 0.001	0.001 ± 0.004	0.060 ± 0.004	18m 03s
IPSVD	198.39	322.80	0.012 ± 0.001	0.012 ± 0.001	0.004 ± 0.004	0.048 ± 0.003	18m 52s
IProX	**68.82**	**102.61**	**0.060** ± 0.001	**0.060** ± 0.002	**0.009** ± 0.003	**0.078** ± 0.004	34m 28s
$(\mathbf{S}+\mathbf{L})$							
OATS	29.29	20.47	0.179 ± 0.002	0.185 ± 0.006	0.017 ± 0.004	0.057 ± 0.004	35m 44s
3BASiL	19.98	14.46	0.409 ± 0.004	0.396 ± 0.014	0.140 ± 0.006	0.137 ± 0.005	43m 06s
CACTUS-slr	**18.27**	**13.02**	**0.505** ± 0.005	**0.493** ± 0.016	**0.219** ± 0.008	**0.163** ± 0.005	46m 51s
70% compression							
Pure pruning							
SparseGPT	156.90	139.13	0.012 ± 0.001	0.011 ± 0.001	0.002 ± 0.004	0.033 ± 0.003	10m 25s
ALPS	68.21	63.47	0.037 ± 0.001	0.040 ± 0.002	0.004 ± 0.003	0.041 ± 0.003	10m 45s
LLM-Surgeon(5sh)	64.56	52.02	0.063 ± 0.001	0.064 ± 0.003	0.003 ± 0.004	0.039 ± 0.004	5h 33m†
CACTUS-prune	**52.19**	**41.08**	**0.071** ± 0.001	**0.072** ± 0.003	**0.009** ± 0.003	**0.055** ± 0.004	17m 09s
Pure low-rank							
SVD-LLM(W)	1081	2106	0.002 ± 0.001	0.001 ± 0.001	0.002 ± 0.003	0.067 ± 0.004	18m 59s
IPSVD	1095	1896	0.001 ± 0.001	0.003 ± 0.001	0.002 ± 0.004	0.046 ± 0.004	18m 13s
IProX	**156.30**	**293.62**	**0.020** ± 0.001	**0.019** ± 0.002	**0.010** ± 0.003	**0.094** ± 0.004	28m 18s
$(\mathbf{S}+\mathbf{L})$							
OATS	389.61	355.95	0.006 ± 0.001	0.006 ± 0.002	-0.002 ± 0.003	0.044 ± 0.003	35m 20s
3BASiL	79.98	68.06	0.024 ± 0.001	0.026 ± 0.002	0.002 ± 0.003	0.037 ± 0.003	38m 01s
CACTUS-slr	**54.58**	**41.74**	**0.071** ± 0.001	**0.072** ± 0.003	**0.009** ± 0.003	**0.058** ± 0.004	37m 12s

Table 7: Influence-preservation results in the **KFAC** (Fisher-weighted) representation.

Qwen3-4B compression results. Tables 8–9 report the same comparison of nine methods on Qwen3-4B at the three compression levels ($30\%, 50\%, 70\%$), under the identical evaluation protocol of Section 3.1 ($n_{\text{tr}} = 1000$ C4, $n_{\text{te}} = 100$ WT-2, projection dim $d = 8192$). Each cell is annotated with $\pm \widehat{\text{SEM}}$ following the same conventions as Tables 6–7. Bolding follows the same convention as 1B tables.

Method	C4 PPL (↓)	WT-2 PPL (↓)	TracIn (↑)				Compress Runtime
			$\cos_{\text{tr}}$	$\cos_{\text{te}}$	ρ_{qry}	$R_{\text{qry@5\%}}$	
Dense	19.85	13.66	1.000	1.000	1.000	1.000	—
30% compression							
Pure pruning							
SparseGPT	21.05	14.53	0.852 ± 0.004	0.862 ± 0.008	0.959 ± 0.002	0.746 ± 0.006	24m 53s
ALPS	20.44	**13.84**	0.892 ± 0.003	0.903 ± 0.006	0.977 ± 0.001	0.820 ± 0.004	43m 07s
LLM-Surgeon(5sh)	20.44	14.33	0.910 ± 0.004	0.915 ± 0.008	0.971 ± 0.003	0.829 ± 0.009	18h 41m†
CACTUS-prune	**20.21**	13.87	$\mathbf{0.920 \pm 0.003}$	$\mathbf{0.923 \pm 0.005}$	$\mathbf{0.984} \pm 0.001$	$\mathbf{0.837 \pm 0.003}$	45m 59s
Pure low-rank							
SVD-LLM(W)	47.31	70.50	0.178 ± 0.002	0.183 ± 0.007	0.309 ± 0.007	0.103 ± 0.004	54m 43s
IPSVD	**37.79**	**60.98**	$\mathbf{0.378 \pm 0.004}$	$\mathbf{0.383 \pm 0.013}$	$\mathbf{0.545} \pm 0.011$	$\mathbf{0.230 \pm 0.008}$	51m 00s
$(\mathbf{S} + \mathbf{L})$							
OATS	20.13	13.96	0.892 ± 0.003	0.889 ± 0.007	0.969 ± 0.001	0.748 ± 0.004	2h 13m
3BASiL	20.15	**13.72**	0.922 ± 0.003	0.928 ± 0.005	0.980 ± 0.001	0.819 ± 0.003	3h 03m
CACTUS-slr	**20.06**	13.73	$\mathbf{0.937 \pm 0.003}$	$\mathbf{0.945 \pm 0.004}$	$\mathbf{0.987} \pm 0.001$	$\mathbf{0.859 \pm 0.004}$	3h 50m
50% compression							
Pure pruning							
SparseGPT	24.43	18.04	0.656 ± 0.004	0.660 ± 0.008	0.874 ± 0.002	0.551 ± 0.005	24m 07s
ALPS	22.60	16.40	0.709 ± 0.004	0.710 ± 0.008	0.895 ± 0.002	0.585 ± 0.005	43m 24s
LLM-Surgeon(5sh)	23.50	19.80	0.680 ± 0.005	0.670 ± 0.015	0.864 ± 0.006	0.561 ± 0.007	20h 13m†
CACTUS-prune	**21.85**	**16.27**	$\mathbf{0.762 \pm 0.004}$	$\mathbf{0.755 \pm 0.010}$	$\mathbf{0.927} \pm 0.002$	$\mathbf{0.697 \pm 0.005}$	42m 18s
Pure low-rank							
SVD-LLM(W)	176.58	213.73	0.113 ± 0.002	0.118 ± 0.005	0.193 ± 0.004	0.052 ± 0.003	53m 15s
IPSVD	**94.19**	**167.49**	$\mathbf{0.146 \pm 0.003}$	$\mathbf{0.151 \pm 0.007}$	$\mathbf{0.354} \pm 0.008$	$\mathbf{0.143 \pm 0.004}$	53m 27s
$(\mathbf{S} + \mathbf{L})$							
OATS	23.97	16.73	0.652 ± 0.003	0.653 ± 0.009	0.804 ± 0.002	0.508 ± 0.004	2h 05m
3BASiL	21.63	**15.34**	0.743 ± 0.004	0.747 ± 0.008	0.919 ± 0.001	0.695 ± 0.004	2h 37m
CACTUS-slr	**21.38**	15.59	$\mathbf{0.782 \pm 0.004}$	$\mathbf{0.790 \pm 0.009}$	$\mathbf{0.937} \pm 0.001$	$\mathbf{0.698 \pm 0.005}$	2h 54m
70% compression							
Pure pruning							
SparseGPT	47.88	48.97	0.303 ± 0.003	0.298 ± 0.011	0.424 ± 0.009	0.128 ± 0.004	24m 01s
ALPS	34.95	32.32	0.406 ± 0.004	0.397 ± 0.011	0.534 ± 0.008	0.070 ± 0.003	44m 15s
LLM-Surgeon(5sh)	49.54	62.56	0.325 ± 0.004	0.323 ± 0.012	0.524 ± 0.025	0.225 ± 0.008	22h 01m†
CACTUS-prune	**30.73**	**29.19**	$\mathbf{0.475 \pm 0.004}$	$\mathbf{0.468 \pm 0.014}$	$\mathbf{0.666} \pm 0.015$	$\mathbf{0.303 \pm 0.008}$	47m 59s
Pure low-rank							
SVD-LLM(W)	**316.85**	521.24	0.101 ± 0.002	0.112 ± 0.005	0.161 ± 0.004	$\mathbf{0.062 \pm 0.003}$	52m 20s
IPSVD	328.27	**489.76**	$\mathbf{0.147 \pm 0.002}$	$\mathbf{0.154 \pm 0.006}$	$\mathbf{0.183} \pm 0.005$	0.061 ± 0.003	49m 25s
$(\mathbf{S} + \mathbf{L})$							
OATS	89.01	68.15	0.064 ± 0.002	0.059 ± 0.005	0.328 ± 0.009	0.113 ± 0.005	2h 07m
3BASiL	34.32	31.28	0.408 ± 0.004	0.400 ± 0.012	0.550 ± 0.010	0.186 ± 0.005	2h 11m
CACTUS-slr	**30.21**	**27.83**	$\mathbf{0.477 \pm 0.004}$	$\mathbf{0.468 \pm 0.013}$	$\mathbf{0.674} \pm 0.012$	$\mathbf{0.275 \pm 0.008}$	2h 14m

Table 8: Qwen3-4B influence-preservation results in the **TracIn** representation, three compression levels. Each cell is annotated with $\pm \widehat{\text{SEM}}$ following the same protocol as Tables 6–7.

Method	C4 PPL (↓)	WT-2 PPL (↓)	KFAC (↑)				Compress Runtime
			$\cos_{\text{tr}}$	$\cos_{\text{te}}$	ρ_{qry}	$R_{\text{qry@5\%}}$	
Dense	19.85	13.66	1.000	1.000	1.000	1.000	—
30% compression							
Pure pruning							
SparseGPT	21.05	14.53	0.739 ± 0.004	0.750 ± 0.010	0.498 ± 0.008	0.292 ± 0.008	24m 53s
ALPS	20.44	**13.84**	0.794 ± 0.004	0.805 ± 0.009	0.595 ± 0.007	0.375 ± 0.006	43m 07s
LLM-Surgeon(5sh)	20.44	14.33	$\mathbf{0.825 \pm 0.004}$	$\mathbf{0.836 \pm 0.009}$	$\mathbf{0.630} \pm 0.010$	0.373 ± 0.012	18h 41m†
CACTUS-prune	**20.21**	13.87	0.816 ± 0.004	0.816 ± 0.009	0.624 ± 0.008	$\mathbf{0.388 \pm 0.007}$	45m 59s
Pure low-rank							
SVD-LLM(W)	47.31	70.50	0.075 ± 0.001	0.079 ± 0.004	0.018 ± 0.004	0.075 ± 0.004	54m 43s
IPSVD	**37.79**	**60.98**	$\mathbf{0.141 \pm 0.002}$	$\mathbf{0.140 \pm 0.005}$	$\mathbf{0.036} \pm 0.004$	$\mathbf{0.081 \pm 0.004}$	51m 00s
$(\mathbf{S} + \mathbf{L})$							
OATS	20.13	13.96	0.790 ± 0.003	0.788 ± 0.009	0.588 ± 0.007	0.360 ± 0.007	2h 13m
3BASiL	20.15	**13.72**	0.857 ± 0.004	0.858 ± 0.010	0.698 ± 0.008	0.441 ± 0.008	3h 03m
CACTUS-slr	**20.06**	13.73	$\mathbf{0.866 \pm 0.004}$	$\mathbf{0.871 \pm 0.008}$	$\mathbf{0.715} \pm 0.007$	$\mathbf{0.460 \pm 0.008}$	3h 50m
50% compression							
Pure pruning							
SparseGPT	24.43	18.04	0.387 ± 0.003	0.388 ± 0.008	0.138 ± 0.004	0.128 ± 0.005	24m 07s
ALPS	22.60	16.40	0.458 ± 0.004	0.455 ± 0.010	0.190 ± 0.005	$\mathbf{0.150 \pm 0.006}$	43m 24s
LLM-Surgeon(5sh)	23.50	19.80	0.397 ± 0.004	0.383 ± 0.013	0.140 ± 0.005	0.125 ± 0.005	20h 13m†
CACTUS-prune	**21.85**	**16.27**	$\mathbf{0.493 \pm 0.005}$	$\mathbf{0.474 \pm 0.013}$	$\mathbf{0.211} \pm 0.006$	0.149 ± 0.006	42m 18s
Pure low-rank							
SVD-LLM(W)	176.58	213.73	0.007 ± 0.001	0.009 ± 0.001	0.013 ± 0.004	0.047 ± 0.004	53m 15s
IPSVD	**94.19**	**167.49**	$\mathbf{0.030 \pm 0.001}$	$\mathbf{0.032 \pm 0.002}$	$\mathbf{0.021} \pm 0.004$	$\mathbf{0.062 \pm 0.004}$	53m 27s
$(\mathbf{S} + \mathbf{L})$							
OATS	23.97	16.73	0.380 ± 0.003	0.383 ± 0.009	0.140 ± 0.004	0.130 ± 0.005	2h 05m
3BASiL	21.63	**15.34**	0.521 ± 0.004	0.521 ± 0.011	0.252 ± 0.005	$\mathbf{0.187 \pm 0.006}$	2h 37m
CACTUS-slr	**21.38**	15.59	$\mathbf{0.539 \pm 0.005}$	$\mathbf{0.538 \pm 0.013}$	$\mathbf{0.259} \pm 0.007$	0.173 ± 0.006	2h 54m
70% compression							
Pure pruning							
SparseGPT	47.88	48.97	0.079 ± 0.001	0.082 ± 0.004	0.024 ± 0.004	0.063 ± 0.003	24m 01s
ALPS	34.95	32.32	0.147 ± 0.002	0.145 ± 0.005	0.042 ± 0.004	0.074 ± 0.004	44m 15s
LLM-Surgeon(5sh)	49.54	62.56	0.085 ± 0.001	0.085 ± 0.003	0.025 ± 0.004	$\mathbf{0.094 \pm 0.005}$	22h 01m†
CACTUS-prune	**30.73**	**29.19**	$\mathbf{0.174 \pm 0.003}$	$\mathbf{0.172 \pm 0.007}$	$\mathbf{0.044} \pm 0.004$	0.087 ± 0.004	47m 59s
Pure low-rank							
SVD-LLM(W)	**316.85**	521.24	0.002 ± 0.001	0.003 ± 0.001	$\mathbf{0.008} \pm 0.004$	0.050 ± 0.003	52m 20s
IPSVD	328.27	**489.76**	$\mathbf{0.003 \pm 0.001}$	$\mathbf{0.006 \pm 0.001}$	0.000 ± 0.004	$\mathbf{0.050 \pm 0.003}$	49m 25s
$(\mathbf{S} + \mathbf{L})$							
OATS	89.01	68.15	0.024 ± 0.001	0.026 ± 0.002	0.026 ± 0.004	0.060 ± 0.004	2h 07m
3BASiL	34.32	31.28	0.150 ± 0.002	0.146 ± 0.006	0.032 ± 0.004	0.071 ± 0.004	2h 11m
CACTUS-slr	**30.21**	**27.83**	$\mathbf{0.173 \pm 0.002}$	$\mathbf{0.167 \pm 0.006}$	$\mathbf{0.038} \pm 0.004$	$\mathbf{0.082 \pm 0.004}$	2h 14m

Table 9: Qwen3-4B influence-preservation results in the **KFAC** representation, three compression levels. Each cell is annotated with $\pm \widehat{\text{SEM}}$ following the same protocol as Tables 6–7.

Llama-3-8B at $(2{:}4{+}r256)$. Table 10 reports the four influence-preservation metrics for the Llama-3-8B $(\mathbf{S}+\mathbf{L})$ configuration $(2{:}4{+}r256)$ alongside C4 and WikiText-2 perplexities. Evaluation protocol matches Section 3.1: $n_{\text{tr}} = 1000$ C4 training samples and $n_{\text{te}} = 100$ WikiText-2 test samples at sequence length 512, projection dim $d = 8192$. **Bold**: best across the three reported $(\mathbf{S}+\mathbf{L})$ methods.

Method	C4 PPL (↓)	WT-2 PPL (↓)	TracIn (↑)				Compress Runtime
			$\cos_{\text{tr}}$	$\cos_{\text{te}}$	ρ_{qry}	$R_{\text{qry@5\%}}$	
Dense	9.45	6.14	1.000	1.000	1.000	1.000	—
$2{:}4 + r256$ **compression**							
$(\mathbf{S}+\mathbf{L})$							
OATS	13.85	9.02	0.391 ± 0.004	0.393 ± 0.013	0.093 ± 0.009	0.093 ± 0.005	5h 28m
3BASiL	12.27	8.06	0.518 ± 0.005	0.523 ± 0.016	0.170 ± 0.016	0.148 ± 0.011	**2h 42m**
CACTUS-SLR	**12.21**	**8.04**	$\mathbf{0.573} \pm 0.006$	$\mathbf{0.569} \pm 0.019$	$\mathbf{0.236} \pm 0.026$	$\mathbf{0.224} \pm 0.018$	4h 07m

Table 10: Llama-3-8B influence-preservation results at the $(2{:}4{+}r256)$ $(\mathbf{S}+\mathbf{L})$ configuration in the **TracIn** (unweighted) representation. Each cell is annotated with $\pm\widehat{\text{SEM}}$ following the same convention as Tables 6–7. Compression runtime is measured on a single NVIDIA A100 GPU.

Llama-3-8B perplexity results across $2{:}4$ and $2{:}4{+}r$ $(\mathbf{S}+\mathbf{L})$ configurations. Table 11 reports C4, WikiText-2, and PTB (Marcus et al., 1994) perplexities (↓) across three compression budgets: $2{:}4$ pure pruning, and the $(\mathbf{S}+\mathbf{L})$ settings $(2{:}4{+}r64)$ and $(2{:}4{+}r256)$. For each budget, three methods are reported in order: activation-only: SparseGPT and ALPS for pure pruning, followed by OATS and 3BASiL for $(\mathbf{S}+\mathbf{L})$, and curvature-aware: CACTUS. Calibration: 128 C4 samples at sequence length 2048; PPL evaluated at sequence length 2048.

Dense	C4 = 9.45			WT-2 = 6.14			PTB = 11.18		
Method	$2{:}4$ pruning			**Method**	$(2{:}4{+}r64)$			$(2{:}4{+}r256)$		
	C4	WT-2	PTB		C4	WT-2	PTB	C4	WT-2	PTB
SparseGPT	22.27	15.98	24.30	OATS	22.28	15.36	23.33	13.85	9.02	14.47
ALPS	20.08	14.44	20.62	3BASiL	15.84	11.20	16.10	12.27	8.06	**12.73**
CACTUS-prune	**19.74**	**14.37**	**20.12**	CACTUS-SLR	**15.46**	**10.75**	**15.73**	**12.21**	**8.04**	12.80

Table 11: Llama-3-8B perplexity (↓) for one-shot $2{:}4$ pruning, and $(2{:}4{+}r)$ $(\mathbf{S}+\mathbf{L})$ compression. **Bold**: best across the three reported methods within each compression budget.

Llama-3.2-3B unstructured pruning. Table 12 reports C4 and WikiText-2 perplexities (↓) for unstructured one-shot pruning on Llama-3.2-3B across nine sparsity levels (10% through 90% in 10% increments). Calibration: 128 C4 samples at sequence length 2048.

Dense	C4 = 11.33			WT-2 = 7.81														
Method	10%		20%		30%		40%		50%		60%		70%		80%		90%	
	C4	WT-2	C4	WT-2	C4	WT-2	C4	WT-2	C4	WT-2	C4	WT-2	C4	WT-2	C4	WT-2	C4	WT-2
SparseGPT	11.42	7.86	11.68	8.01	12.34	8.42	13.80	9.37	17.31	12.24	29.38	21.73	77.82	69.13	212.5	261.5	999.1	1510.6
ALPS	11.39	7.85	11.61	7.98	12.18	8.33	13.40	9.14	15.85	11.30	22.92	18.19	45.57	42.39	106.5	132.9	365.7	563.6
CACTUS-prune	**11.38**	**7.84**	**11.56**	**7.94**	**12.07**	**8.27**	**13.17**	**9.01**	**15.36**	**11.12**	**21.49**	**17.37**	**41.92**	**35.58**	**101.2**	**125.2**	**354.7**	**482.7**

Table 12: Llama-3.2-3B perplexity (↓) for one-shot unstructured pruning across nine sparsity levels. **Bold**: best across the three reported methods within each sparsity level. CACTUS-prune wins all cells.

Method	C4	WT-2	PIQA	ARC-E	ARC-C	HellaSwag	WinoGrande	OBQA	RTE	Avg.	Time
Dense	11.27	7.05	81.72	73.06	58.96	82.91	75.85	46.40	82.31	71.60	—
SparseGPT	14.17	9.47	78.62	80.13	56.91	74.18	**72.85**	40.40	75.45	68.36	**58m 15s**
CACTUS-prune	**13.42**	**8.88**	**79.27**	**82.07**	**58.11**	**75.51**	72.77	**41.80**	**78.34**	**69.70**	10h 28m

Table 13: **Qwen3.8-27B at** 2:4 **sparsity** across all 496 decoder linear matrices: C4/WT-2 perplexity (↓), seven zero-shot accuracies (↑). **Bold**: better compressed method. Total time (↓) includes compression on 2 NVIDIA L40S GPUs. Our method additionally collected gradient statistics during 2h 21m, included in the overall compression time, and used 4 L40S GPUs for the statistics collection.

D.1 LESS SELECTION SCORES DISTRIBUTION

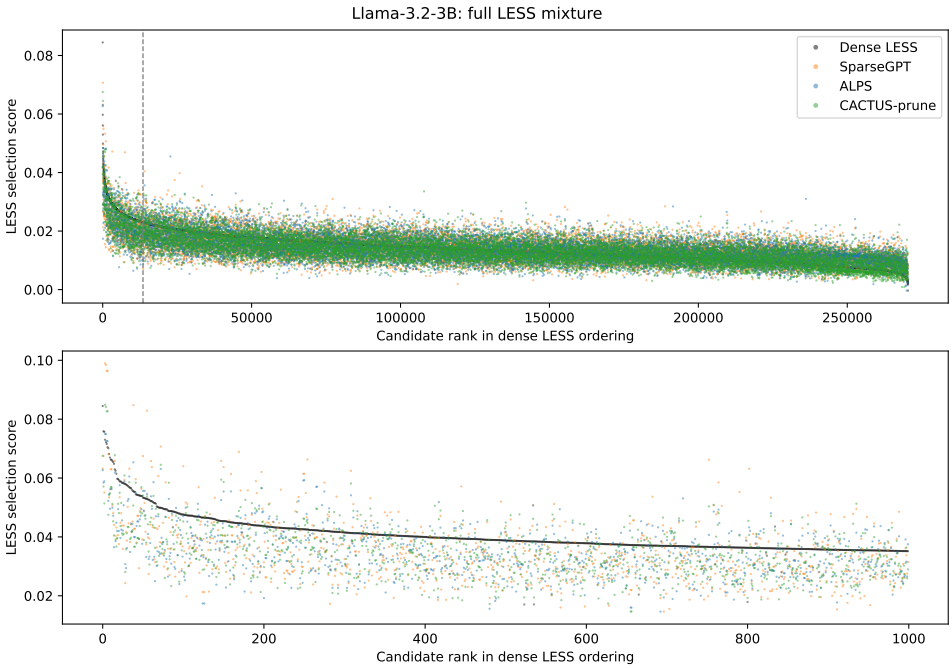


Figure 6: LESS selection scores for dense Llama-3.2-3B and its SparseGPT, ALPS, and CACTUS-prune proxies, with candidates ordered by the dense model's aggregate score. The top panel spans the full instruction mixture; the bottom panel shows the first 1000 candidates. The dashed line marks the top-5% cutoff.

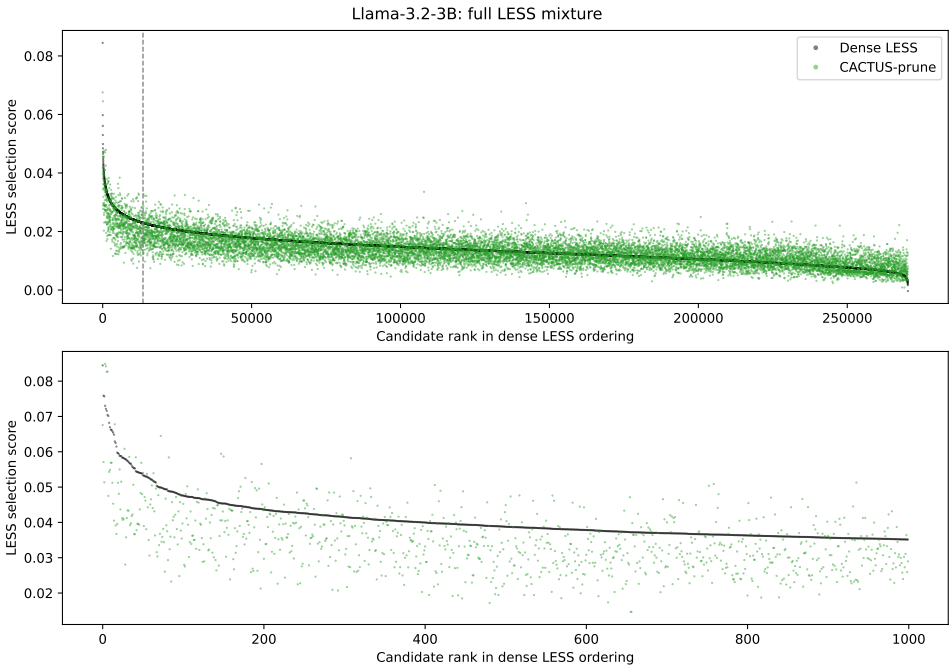


Figure 7: Dense LESS and CACTUS-prune scores from Figure 6, shown separately.

D.2 Per-layer Loss Decrease

We examine how the layerwise KFAC objective evolves over CACTUS's ADMM iterations. Figure 8 compares representative loss trajectories for pure pruning and $(\mathbf{S}+\mathbf{L})$ compression, illustrating their initial non-monotonic behavior and subsequent stabilization as the penalty increases.

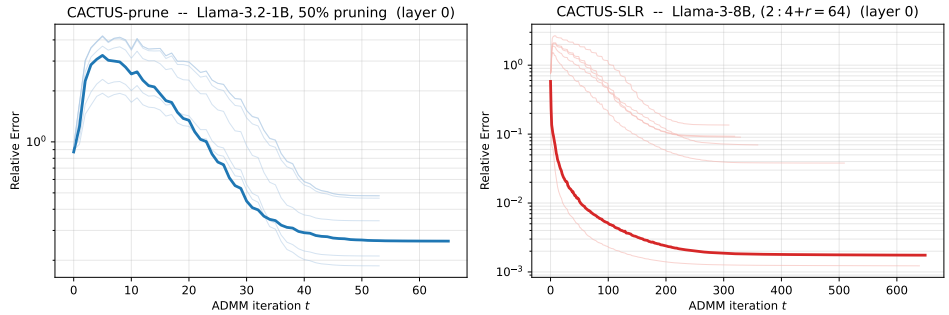


Figure 8: CACTUS relative-error trajectory for a representative layer. The KFAC objective can increase during the first ADMM iterations because the method is exploring feasible sparse candidates while the penalty schedule is still loose. As the penalty grows, primal feasibility tightens and the iterates settle, yielding the expected decrease and convergence of the objective.

D.3 Effect of Calibration-Set Size

We vary the number of C4 calibration sequences from 32 to 256 for CACTUS-prune at 50% unstructured sparsity on Llama-3.2-1B and Qwen3-4B (Table 14). All runs use sequence length 2048 and same calibration data and ridge coefficients $\lambda_H = 0.01$ and $\lambda_G = 0.1$. Perplexity improves with more calibration data on all three datasets. Doubling the sample count from 128 to 256 yields modest further C4 improvements of 1.7% and 0.4%, respectively, supporting 128 sequences as a practical calibration budget.

	Llama-3.2-1B			Qwen3-4B		
Sequences	C4	WT-2	PTB	C4	WT-2	PTB
32	21.113	15.455	24.121	22.429	17.235	28.703
64	20.105	14.566	23.181	21.952	16.811	28.576
128	19.465	14.211	22.613	21.743	16.102	27.924
256	19.132	13.842	22.399	21.663	16.057	27.660

Table 14: Effect of calibration-set size on perplexity (↓) for CACTUS-prune at 50% unstructured sparsity. All other settings are held fixed for each model.