GROUP-LINEARIZED BACKPROPAGATION: SCHEDULING RESIDUAL PATHS FOR DEEP NETWORK TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Residual networks provide exponentially many implicit paths for gradient propagation, yet standard backpropagation still follows a sequential chain whose critical path grows with depth. Existing methods either modify the forward computation, changing the network being trained, or retain the original backward operator and approximate its solution through iterative truncation. We introduce Path-Scheduled Residual Training, built on Group-Linearized Backpropagation (GLBP). Specifically, GLBP partitions the residual chain into contiguous groups, where blocks within each group receive a shared entry gradient and are aggregated in parallel—replacing the standard sequential product within each group with a sum while preserving the product structure across groups. This rewriting requires exactly one block gradient per residual position per step, regardless of grouping granularity. Coarser groupings trade within-group path interactions for reduced serial depth, and which paths are retained is determined by where group boundaries fall rather than by a global order cutoff. The grouping granularity is then scheduled during training: a coarse-to-fine progression starts from short serial depth and reinstates higher-order path content over training, reaching the exact operator at its final stage. Experiments on ResNets and fine-tuning of LLaMA-3.1-8B show that scheduled GLBP achieves near-standard training quality and is competitive with standard backpropagation on downstream benchmarks.

1 INTRODUCTION

Backpropagation computes gradients along a chain whose length equals the network depth: the cotangent must traverse all n residual blocks one after another. For deep models and large models this backward chain is a critical path—in a pipeline it is the segment that determines stage balance, and on a single device it can make the backward pass a major contributor to training time. Veit et al. (2016) showed that residual networks admit a path view in which the backward polynomial expands into 2^n terms. This suggests that the *shape* of the backward computation—which interactions are formed and when—is a design variable rather than a fixed fact.

Existing work explores two ways to exploit this. One modifies the forward path: Stochastic Depth (Huang et al., 2016), LayerDrop (Fan et al., 2019), and layer-sampling methods drop or sample blocks in the forward pass, training a subnetwork and accepting a train–test discrepancy. The other modifies the solve: Highway-BP keeps the exact backward operator and truncates its Neumann iteration, approximating the backward solution at a cost of $(k+1)n$ block-VJPs for k iterations.

We take a third route: *rewrite the backward operator itself, then solve the rewritten operator exactly, once*. We call this framework Path-Scheduled Residual Training; its core operator is Group-Linearized Backpropagation (GLBP). Grouping the chain and replacing each intra-group product by a sum (Figure 1) yields a coarse operator H_Π whose retained paths are those selecting at most one branch per group, Eq. equation 4. Three properties follow. First, the block-VJP cost is exactly n regardless of how the chain is grouped. Second, the singleton partition recovers standard backpropagation at zero overhead—the construction has an exact anchor. Third, the partition may vary over training: the path family becomes a schedulable object, and the member VJPs inside a group are

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

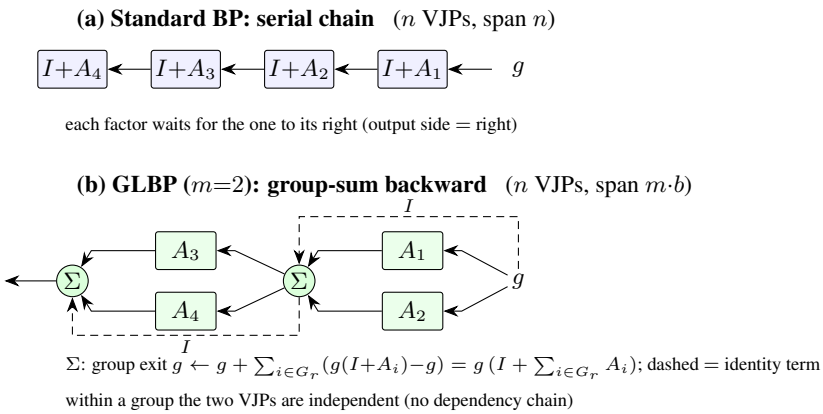


Figure 1: GLBP rewrites the residual backward chain. **(a)** Standard backpropagation composes the factors $(I + A_i)$ serially: span n. **(b)** GLBP with two groups $G_1 = \{1, 2\}$, $G_2 = \{3, 4\}$ shares one entrance cotangent across a group, computes member VJPs independently, and combines additively at the group exit (Σ); inter-group products are kept. The block-VJP count is n for both; only the span changes ($m{\cdot}b$). By Theorem 1 the retained paths are those selecting at most one branch per group.

independent nodes of the backward dependency graph, a structural parallelism that does not depend on physical devices.

Contributions.

- **Operator rewriting with a path criterion.** A group product-to-sum transformation of the residual backward chain, with an exact characterization of the retained path family $\mathcal{R}(\Pi)$ and a fixed block-VJP cost structure (Section 4; verified against the exact gradient in Appendix F).
- **A schedulable grouping operator.** The partition Π_t may change over training; fixed and progressive groupings are two representative instances (Sections 4.4, 5.2).
- **Structural parallelism.** Member VJPs are dependency-independent by construction (Section 4.3); a *constrained* GPU-boundary instance illustrates that this structure can be realized in hardware (Section 5.5).
- **Cross-scale evaluation.** Quality–cost tradeoffs on CIFAR ResNets from $n = 15$ to $n = 600$ (Sections 5.1, 5.3) and LLaMA-3.1-8B (Grattafiori et al., 2024) fine-tuning on instruction following, code, and math, where a progressive schedule approaches the Standard BP baseline (Section 5.4).

2 Related Work

We organize prior work by the plane on which it operates. GLBP acts on the backward operator and makes the retained path family a schedulable quantity—a combination we are not aware of in prior work.

Residual path expansion. Veit et al. (2016) cast residual networks as ensembles of shallow paths; He et al. (2016) give the identity-mapping block our chain model assumes, and Chen et al. (2024) formalize residual computation as a jet expansion. These read path structure as an object of *analysis*; we treat it as one of *control*. We take the 2^n-term expansion as background (Veit et al., 2016) and add the grouping criterion of Eq. equation 4; "shorter paths matter more" is motivation, not a law we assume. Balduzzi et al. (2017) and Bengio et al. (1994) give the backdrop for why the backward chain is hard.

Forward-path interventions. Stochastic Depth (Huang et al., 2016), LayerDrop (Fan et al., 2019), FractalNet (Larsson et al., 2017), Dropout (Srivastava et al., 2014), and dynamic-depth routers (SkipNet (Wang et al., 2018), Mixture-of-Depths (Raposo et al., 2024)) drop or route computation in the forward pass, changing what is trained or inferred. GLBP leaves the forward pass unchanged and modifies only the backward operator.

Gated and scaled residuals. Highway Networks (Srivastava et al., 2015), SkipInit (De & Smith, 2020), Fixup (Zhang et al., 2019), ReZero (Bachlechner et al., 2021), and LayerScale (Touvron et al., 2021) rescale residual branches, changing the *coefficients* of the path expansion; Chen et al. (2024) weight paths by $\lambda^{|S|}$. GLBP instead assigns coefficient 1 to surviving paths and 0 to the rest—a structural, not magnitude-based, selection, compatible with forward scaling.

Approximate backward solvers. Highway-BP (Anonymous, 2025) keeps the exact operator and truncates its iterative solution by order (our closest comparison; Appendix B). Truncated-BPTT corrections ARTBP (Tallec & Ollivier, 2018) and UORO (Tallec & Ollivier, 2017) truncate-then-reweight; synthetic gradients (Jaderberg et al., 2017), feedback alignment (Lillicrap et al., 2016; Nøkland, 2016), and Forward-Forward (Hinton, 2022) replace or remove the true backward signal. GLBP retains the true chain, linearized only within groups.

Parallelizing the backward chain. DeepPCR (Danieli et al., 2023), ELK (Gonzalez et al., 2024), and layer-parallel training (Günther et al., 2020) reduce the span of an exact/fixed-point computation; GLBP reduces span by changing the operator. Decoupled parallel backprop (Huo et al., 2018b) and features replay (Huo et al., 2018a) group layers with stale activations (error = staleness), whereas GLBP groups over the path polynomial (error = omitted intra-group interactions).

3 Notation and Assumptions

We use a row-covector convention and index residual blocks in backward-processing order: block 1 is adjacent to the terminal cotangent and block n is at the network input. Products of backward operators therefore follow this processing order. Cotangents are written in plain italic; in particular, g is the terminal cotangent and h is an intermediate cotangent. The symbol m is reserved for the number of GLBP groups; it is never used for a cotangent.

3.1 Residual blocks and backward factors

We use the clean residual setting: block i has backward factor $I + A_i$, where A_i is the Jacobian contribution of its residual branch and the residual connection is not followed by a further transformation. A non-identity post-residual transformation (e.g. a post-activation) contributes an additional local map and is absorbed as described in Appendix A.6; it does not change the GLBP rewrite of the residual factors.

3.2 Path polynomial

With g the terminal cotangent, exact backpropagation in the clean residual setting is

$$H_{\text{exact}} = g \prod_{i=1}^{n} (I + A_i) = \sum_{S \subseteq [n]} g\, P_S, \tag{1}$$

where the product follows the backward-processing convention above and P_S is the ordered product of the A_i selected by S; $P_\varnothing = I$. Each block contributes either its identity path or its residual-branch path, so the expansion contains 2^n terms and an order-r path selects r residual branches.

3.3 Block–bundle–group hierarchy

The computation has three levels. A *block* is the atomic mathematical unit. A *bundle* contains b consecutive blocks whose internal composition is computed exactly; b is an execution-granularity parameter, not a modeling variable. A *group* is a collection of consecutive bundles on which GLBP

Table 1: GLBP control parameters. Background symbols (n, g, A_i, P_S) and the Highway-BP count k are defined where they first appear.

Symbol	Meaning	Scope / note
b, B	bundle size and bundle count $B = \lceil n/b \rceil$	exact execution granularity
m (pps)	number of GLBP groups; in staged models pps is the per-stage-body group count, so the network total is $m \approx \texttt{pps} \times$ (#stage bodies)	main control variable; not a cotangent
Π, G_r	grouping partition and its r-th contiguous group	Π_t may vary over training
$\mathcal{R}(\Pi)$	retained path family	Eq. equation 4
H_Π, L_Π	grouped backward map and its inducing operator ($H_\Pi = \sum_{r\geq 0} L_\Pi^r g$)	Appendix B

applies the product-to-sum rewriting. In staged implementations, groups may align with physical layer or GPU stages; pps denotes the number of position groups within one stage body.

The partition is $\Pi = (G_1, \ldots, G_m)$, where G_r is the r-th group and m is the number of groups. During scheduled training, $\Pi_t = (m_t, \mathrm{cuts}_t)$ may vary, while b remains an implementation choice.

Operational assumption. The path polynomial is a bookkeeping representation of the backward operator, not a claim that a nonlinear network decomposes into 2^n independent subnetworks. Shared parameters, nonlinearities, and activation-dependent Jacobians couple the terms. Our statements concern which interactions the specified backward operator generates and retains; they do not assert independent subnetworks or task-independent path importance. We use fixed and progressive groupings as representative instances rather than searching for an optimal schedule, and the GPU-boundary system instance (Section 5.5) is one constrained hardware realization, not a general deployment claim.

4 Group-Linearized Backpropagation

GLBP modifies the algebraic structure of the backward chain while leaving the forward pass untouched. We first define the rewriting and its path semantics (Section 4.1), develop the retention and error structure (Section 4.2), state the cost invariants and structural parallelism (Section 4.3), and formalize the schedulable operator (Section 4.4).

4.1 Group product-to-sum rewriting

Consider a residual chain of n blocks with backward factors $(I + A_i)$, where A_i is the transposed Jacobian of the residual branch of block i. Exact backpropagation composes the factors as

$$H_{\text{exact}} \;=\; g\prod_{i=1}^{n}(I + A_i) \;=\; \sum_{S\subseteq[n]} gP_S, \tag{2}$$

where gP_S denotes the path term selecting residual branch A_i for $i \in S$ and the identity otherwise. The expansion contains 2^n terms and is realized by standard backpropagation as a strictly sequential chain. We index blocks in *backward processing order*: block 1 is adjacent to the terminal cotangent g (output side) and block n is at the input; products of row-covector actions compose left to right in this order, so equation 2 applies $(I + A_n)$ last and $(I + A_1)$ first to g.

GLBP partitions the chain into m contiguous groups $G_1, \ldots, G_m$ and replaces the *intra-group* product by a sum while keeping the inter-group product:

$$\boxed{\prod_{i\in G_r}(I + A_i) \;\longrightarrow\; I + \sum_{i\in G_r} A_i} \qquad H_\Pi \;=\; g\prod_{r=1}^{m}\Big(I + \sum_{i\in G_r} A_i\Big). \tag{3}$$

Operationally, a group receives one entrance cotangent; its members compute independent local VJPs and their outputs are added at the group exit. The partition $\Pi = (G_1, \ldots, G_m)$, equivalently its cut positions, determines the operator; the execution granularity b is fixed by the implementation and is not part of Π.

4.2 PATH RETENTION AND ERROR STRUCTURE

The expansion of equation 3 is fully characterized by the following statement.

Theorem 1 (Path retention)**.** *For any partition* Π *(statements are at block level,* $b = 1$*; with* $b > 1$ *the same statements hold at bundle granularity, where a bundle operator replaces* A_i*),*

$$H_\Pi \;=\; \sum_{S \in \mathcal{R}(\Pi)} g P_S, \qquad \mathcal{R}(\Pi) \;=\; \big\{\, S \subseteq [n] \;:\; |S \cap G_r| \le 1,\ \forall r \,\big\}. \tag{4}$$

The omitted content is exactly the set of paths with at least two selected branches inside one group:

$$\Delta_\Pi \;:=\; H_{\text{exact}} - H_\Pi \;=\; \sum_{\substack{S \subseteq [n] \\ \exists r:\, |S \cap G_r| \ge 2}} g P_S. \tag{5}$$

Proof sketch. Expanding each factor selects the identity or one A_i per group, so gP_S appears iff S picks at most one branch per group; equation 5 is the complement. Full proof in Appendix A. □

The retained family is *anisotropic*: it depends on where the group boundaries fall, not only on the order $|S|$. In particular, with m groups no path of order $|S| > m$ survives, so $\mathcal{R}(\Pi) \subseteq \{S : |S| \le m\}$; the bound is tight but $\mathcal{R}(\Pi)$ does not contain all paths of order $\le m$ in general. The lowest-order omitted terms are the intra-group pairs, whose count is $\sum_r \binom{|G_r|}{2}$; all cross-group pairs are retained (Proposition 2). This structural count describes the polynomial error and does not imply task-independent path dominance; the full refinement, together with the positional entrance-error decomposition (Theorem 2), is in Appendix A.

4.3 COST STRUCTURE AND STRUCTURAL PARALLELISM

Proposition 1 (Cost invariant)**.** *For any partition* Π *and any execution granularity* b*, the number of block-VJPs per step equals* n*:* $W_{\text{GLBP}} = n$*. The dependency-graph span of the grouped backward is* $D_{\text{GLBP}} \approx m \cdot b$*. The singleton partition* $m = n$ *(with* $b = 1$*) satisfies* $H_\Pi = H_{\text{exact}}$ *at zero overhead.*

Every block contributes one VJP, giving $W_{\text{GLBP}} = n$ for every partition; the group dependency graph has span $D_{\text{GLBP}} \approx m \cdot b$. The proof and the exactness boundary are given in Appendix A.4.

In the Brent model $T_p \approx \max(W/p, D)$ with p effective workers, GLBP trades retained path content for structural span at fixed work. Cost figures in Sections 5.1–5.4 are block-VJP counts and dependency-graph spans (Section 5.5 reports end-to-end step times for its hardware case study).

Definition 1 (Structural parallelism)**.** *The backward computation of a group* G_r *is* structurally parallel *if its member VJPs depend only on the shared entrance cotangent and on pairwise disjoint parameter subsets. Structural parallelism is a property of the dependency graph and is independent of the number of physical devices.*

GLBP satisfies this definition by construction; hardware is only needed to convert structural span into elapsed time, and the constrained GPU-boundary realization is reported in Section 5.5.

For the clean-residual setting $K_i = I$, let $h^{(r)}$ be the shared entrance cotangent of G_r (with $h^{(1)} = g$) and let $\text{rv}_i = \text{VJP}(I + A_i, h^{(r)})$. The group exit is

$$h^{(r+1)} = h^{(r)} + \sum_{i \in G_r} \Big(\text{rv}_i - h^{(r)} \Big), \tag{6}$$

Algorithm 1 implements this group update.

4.4 THE SCHEDULABLE GROUPING OPERATOR

Because H_Π depends on Π only through the group algebra, the partition may vary across training steps:

$$\Pi_t = \big(m_t,\ \text{cuts}_t\big), \tag{7}$$

Algorithm 1 GLBP backward (one step)

Require: partition $\Pi = (G_1, \ldots, G_m)$; terminal cotangent g

1: $w \leftarrow g$ ▷ entrance cotangent of the output-side group G_1
2: **for** $r = 1$ to m **do** ▷ groups processed from output toward input
3: **for each** $i \in G_r$ **in parallel**: $\mathrm{rv}_i \leftarrow \mathrm{VJP}(I + A_i,\, w)$ ▷ $= w(I + A_i)$; independent member VJPs
4: $w \leftarrow w + \sum_{i \in G_r} (\mathrm{rv}_i - w)$ ▷ $= w\,(I + \sum_{i \in G_r} A_i)$
5: **end for**
6: flush w through the input embedding / stem

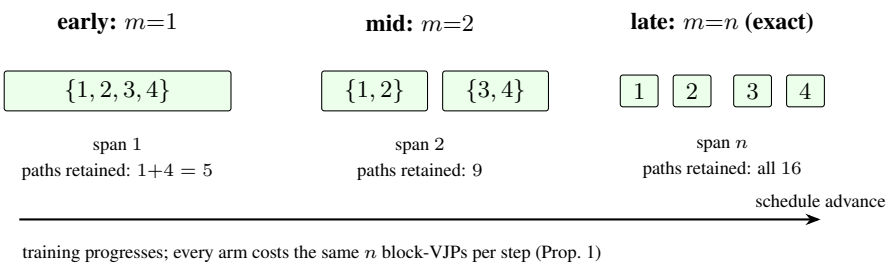


Figure 2: A progressive schedule increases the group count m over training (illustrated on a chain of four blocks). Coarse groups (left) give small span and retain low-order paths; refining toward singletons (right) reinstates higher-order path content and reaches the exact operator at m=n. The block-VJP budget is constant across the schedule; only the span and the retained family change (Section 4.4). Cut positions at a fixed m are an additional, optional degree of freedom that we do not tune here.

and the path family represented at step t is $\mathcal{R}(\Pi)_t$. Varying Π_t turns the path structure into a controllable training variable; we call this property *path scheduling*.

The training loop is unchanged except that each step draws Π_t from the schedule and runs Algorithm 1 with it before the optimizer update. Figure 2 illustrates a coarse-to-fine schedule. Two representative instances are studied in this paper: a *fixed* grouping with constant m, and a *progressive* grouping that increases m over training, reinstating higher-order path content and, when scheduled to the boundary, reaching the exact backward operator. Optional policies within the same framework—rotating cut positions at fixed m, cycling through all legal cuts, adaptive fidelity—are orthogonal to the operator definition and are not developed further here.

Remark 1 (Coverage is not unbiasedness)**.** *Varying Π_t can cover paths across training, but coverage is not equivalent to reconstructing the exact gradient on any individual step. We therefore distinguish schedule coverage from per-step gradient fidelity; the formal coefficient condition is given in Appendix C.4.*

Implementation note. The parameter b bundles b consecutive blocks into one VJP unit: inside a bundle the composition is exact, across bundles the algebra of equation 3 applies to bundle operators. We use $b = 1$ on small models and a larger b on LLM fine-tuning as an engineering configuration; a systematic study of b is out of scope.

5 Experiments

We organize the evaluation around four questions. (1) Does GLBP give a controllable quality–cost tradeoff at fixed compute (Section 5.1)? (2) Does the grouping operator remain usable when scheduled during training (Section 5.2)? (3) How does the method behave as depth grows to very deep chains (Section 5.3)? (4) Does it transfer to LLM fine-tuning (Section 5.4)? Section 5.5 reports one hardware realization on GPU-boundary cuts as a constrained case study.

Protocol and metric discipline. We report three metric families separately: *task metrics* (test accuracy, benchmark scores) measure quality, *gradient diagnostics* (cosine, relative ℓ_2) measure

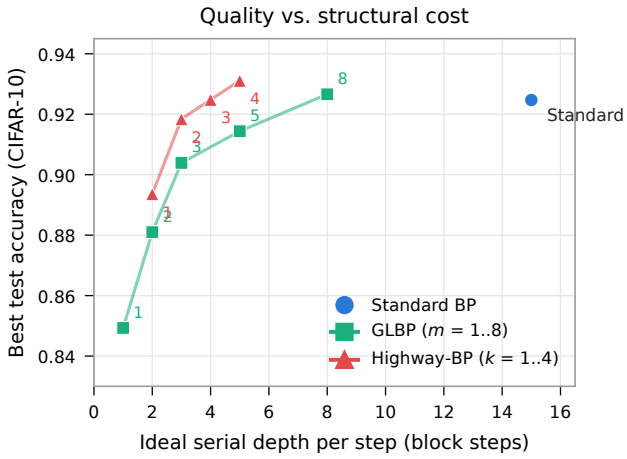


Figure 3: Quality versus structural cost on PreactResNet-32 (CIFAR-10, 200 epochs). GLBP sweeps the group count m at a constant budget of 15 block-VJPs per step; Highway-BP pays $(k{+}1)n$ VJPs for k iterations. Point labels are m (squares) and k (triangles).

Table 2: ResNet-32 on CIFAR-10 (200 epochs). All GLBP arms (fixed and scheduled) keep block-VJP$= 15$ per step; Highway-BP pays $(k{+}1) \cdot 15$. The two Standard BP rows are separate, unpaired reference runs (top: fixed sweep; bottom: schedules; run-to-run variation a few tenths of a point). For schedules the span is the schedule average $\sum_t m_t/T$. Best test accuracy; bold marks the best GLBP arm within each block (not the block maximum); ideal span/step is the dependency-graph span, not wall-clock.

Method	m / k / schedule	Best acc.	Block-VJPs/step	Ideal span/step
Standard BP (sweep ref.)	–	0.9247	15	15
GLBP	$m = 1$	0.8493	15	1
GLBP	$m = 2$	0.8810	15	2
GLBP	$m = 3$	0.9039	15	3
GLBP	$m = 5$	0.9144	15	5
GLBP	$m = 8$	**0.9266**	15	8
Highway-BP	$k = 1$	0.8936	30	2
Highway-BP	$k = 2$	0.9184	45	3
Highway-BP	$k = 3$	0.9248	60	4
Highway-BP	$k = 4$	0.9311	75	5
Standard BP (sched. ref.)	–	0.9304	15	15
GLBP fixed	$m = 1$	0.8523	15	1
GLBP fixed	$m = 4$	0.9138	15	4
GLBP prog.	$1 \to 2 \to 4 \to 8 \to 15$	0.9148	15	6
GLBP prog.	end-weighted	**0.9230**	15	10.5

approximation fidelity, and *structural counts* (block-VJPs, dependency-graph span) measure compute cost. Bench experiments train CIFAR-10 ResNets for 200 epochs with SGD; LLM experiments fine-tune LLaMA-3.1-8B (Grattafiori et al., 2024) with AdamW (Kingma & Ba, 2015; Loshchilov & Hutter, 2019). We use $b = 1$ on small models and $b = 4$ on LLM fine-tuning, and count Highway-BP under the $(k+1)n$ convention (including the final parameter-gradient sweep). Full hyperparameters and dataset details are in Appendix E. All runs are single-seed; we report best scores without error bars or significance tests.

Table 3: Very deep ResNets on CIFAR-10. `pps` denotes position groups per stage body; dimension-changing transitions are handled outside the grouped body. Best test accuracy.

	Best accuracy		
Model	Standard	pps=4	pps=6
ResNet-110 post-act (n=54)	0.9298	0.9274	0.9255
ResNet-110 pre-act	0.9289	0.9232	0.9155
ResNet-1202 post-act (n=600)	0.9380	0.8920	0.9098
ResNet-1202 pre-act	0.8661	0.8418	0.8529

5.1 Quality–cost tradeoff at fixed compute

Figure 3 and the top block of Table 2 sweep the group count on PreactResNet-32 ($n = 15$). The cost invariant of Proposition 1 is visible: every GLBP configuration costs 15 block-VJPs/step, and among fixed-grouping runs quality rises monotonically with m, from 84.93% ($m = 1$) to 92.66% ($m = 8$). At $m = 8$, GLBP reaches 92.66% at the budget where Standard BP reaches 92.47% (within 0.2 points). Highway-BP reaches 92.48% only at $k = 3$—60 block-VJPs/step, four times the budget. The comparison illustrates the two cost structures of Table 5: GLBP fixes compute and varies path shape, Highway-BP buys path content with VJPs.

Gradient fidelity. At the gradient level (Appendix F, Figure 6), cosine against the exact gradient rises monotonically with m on all three diagnostic models. In omitted-monomial counts this matches Proposition 2; the measured error grows in the same direction (term magnitudes are task-dependent).

5.2 Scheduling the grouping operator

The design motivation for scheduling is that a fixed grouping keeps a fixed path family while training proceeds. A diagnostic run (Appendix F, Figure 8) quantifies this: along a Standard BP trajectory, the cosine between the GLBP(m=4) gradient and the exact gradient falls from 0.80 at initialization to 0.39 at the end of training, while larger m degrades more slowly. The approximation error of a fixed grouping therefore evolves over training—a fact that motivates, but does not by itself justify, adapting the operator.

The lower block of Table 2 (curves in Appendix F, Figure 7) reports two representative instances that refine m over training. Under the same block-VJP budget the end-weighted progressive arm reaches 92.30%, versus 91.38% (fixed m=4) and 85.23% (fixed m=1). Per Remark 1, these are representative feasibility instances, not an optimized schedule; we make no claim that progressive dominates fixed grouping in general.

5.3 Scaling to very deep chains

Table 3 moves to ResNet-110 ($n = 54$) and ResNet-1202 ($n = 600$). The post-activation tracks apply a ReLU after the residual addition. Its local highway map is not itself replaced by the product-to-sum rewrite; within-group residual interactions remain approximated as described in Appendix A.6. Coarse grouping becomes expensive as depth grows: on ResNet-1202, `pps` $= 4$ loses 4.6 points (post-act) and 2.4 points (pre-act) against Standard BP, while finer grouping (`pps` $= 6$) recovers part of the gap; on ResNet-110 the coarse cost is under one point (post-act). This is a boundary statement, not a rule—the granularity needed to retain quality grows with depth, so $n = 15$ settings do not transfer to $n = 600$. The count of deleted intra-group interactions grows with group size (Proposition 2); whether error compounds with depth is an observation here, not a theorem. Highway-BP is compared at $n = 15$; at $n = 54$ we report one point ($k = 2$: 0.9291 at $3\times$ the budget, separate track); at $n = 600$ the $(k + 1)n$ cost makes the sweep impractical.

5.4 LLM fine-tuning

Table 4 fine-tunes LLaMA-3.1-8B on instruction following (Alpaca (Taori et al., 2023)→IFEval (Zhou et al., 2023)), code (Evol-CodeAlpaca→HumanEval (Chen et al., 2021)),

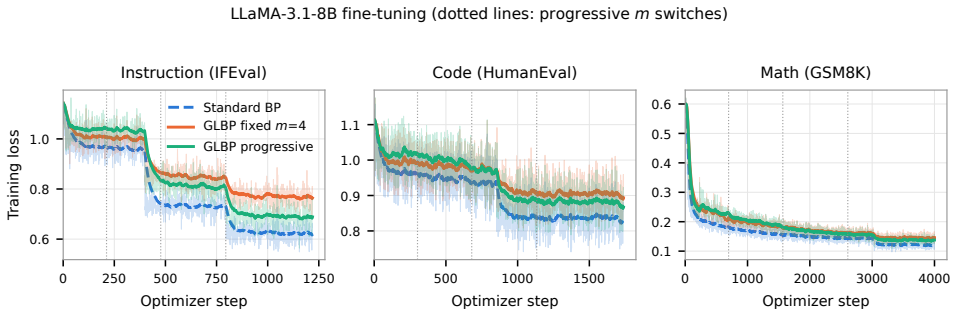


Figure 4: LLaMA-3.1-8B fine-tuning losses. Bold lines are EMA-smoothed; faint lines are raw. Dotted vertical lines mark the progressive schedule's m-switch points. Training-loss ordering is Standard $<$ progressive $<$ fixed m=4 on all three tasks; downstream metrics (Table 4) do not follow the same order.

Table 4: LLaMA-3.1-8B fine-tuning. Three tasks, three arms, and the untuned base model. Fixed and progressive GLBP both use $b = 4$; progressive increases m over training. Bold marks the largest value in each row, reported descriptively. All arms share the same block-VJP budget per task.

Task	Metric	Base	Standard	Fixed m=4	Progressive
Instruct	IFEval (p-loose)	0.111	**0.311**	0.296	**0.311**
Code	HumanEval	0.372	0.518	0.451	**0.543**
Math	GSM8K (strict)	0.538	0.647	0.552	**0.676**
Math	GSM8K (flexible)	0.563	**0.755**	0.684	0.721

and math (MetaMathQA (Yu et al., 2023)→GSM8K (Cobbe et al., 2021)), comparing Standard BP, fixed GLBP ($m = 4$), and progressive GLBP at equal block-VJP budget. The progressive arm approaches the Standard baseline on all three tasks (IFEval tie at 0.311; HumanEval/GSM8K-strict $0.543/0.676$ vs. $0.518/0.647$; GSM8K-flex 0.721 vs. 0.755). Fixed $m = 4$ is lower than both on every reported metric.

Figure 4 shows all three arms train stably with the same curve shape (no divergence from the grouped operator), and that the training-loss ordering (Standard $<$ progressive $<$ fixed) differs from the downstream-metric ordering of Table 4—a dissociation we report without explaining.

5.5 A SYSTEM INSTANCE ON GPU-BOUNDARY CUTS

Finally, as one constrained illustration—not a core evaluation—we map GLBP groups to GPU-boundary stages to check that the structural parallelism of Definition 1 can be realized on hardware. The instance reduces step time by $1.34\times$ and $1.63\times$ with close final losses; full setup, timings, and curves are in Appendix E (Table 6, Figure 5). This is one fixed-device, GPU-boundary realization—evidence that the structural parallelism *can* become end-to-end speedup, not that arbitrary partitions would.

6 DISCUSSION AND CONCLUSION

Group-linearized backpropagation provides an analyzable interpolation between approximate and exact backpropagation under a constant block-VJP budget, with path fidelity a controllable function of the grouping structure: on small residual networks it trades path content for span at fixed compute, on very deep chains it exposes the cost of over-coarse grouping, and on LLM fine-tuning a progressive schedule approaches the standard baseline. The main open directions are schedule design (cut rotation, coverage cycles, task-adaptive policies, and per-task group counts), turning the structural parallelism of the constrained GPU-boundary instance into a general multi-device speedup, and tighter theoretical control of the omitted-interaction error with depth.

AI USE STATEMENT

Generative AI tools assisted with literature organization, draft editing, LaTeX formatting, code inspection, and figure/layout iteration. The authors independently verified all mathematical statements, citations, experimental descriptions, numerical results, and generated artifacts, and take responsibility for the final content of this submission.

REFERENCES

Anonymous. Accelerated training through iterative gradient propagation along the residual path. In *International Conference on Learning Representations*, 2025.

Thomas Bachlechner, Bodhisattwa Prasad Majumder, Huanru Henry Mao, Garrison W. Cottrell, and Julian McAuley. Rezero is all you need: Fast convergence at large depth. In *Uncertainty in Artificial Intelligence*, 2021.

David Balduzzi, Marcus Frean, Lennox Leary, JP Lewis, Kurt Wan-Duo Ma, and Brian McWilliams. The shattered gradients problem: If resnets are the answer, then what is the question? In *International Conference on Machine Learning*, 2017.

Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994.

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

Yihong Chen, Xiangxiang Xu, Yao Lu, Pontus Stenetorp, and Luca Franceschi. Jet expansions of residual computation. *arXiv preprint arXiv:2410.06024*, 2024.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Federico Danieli, Miguel Sarabia, Xavier Suau, Pau Rodríguez, and Luca Zappella. Deeppcr: Parallelizing sequential operations in neural networks. In *Advances in Neural Information Processing Systems*, 2023.

Soham De and Samuel L. Smith. Batch normalization biases residual blocks towards the identity function in deep networks. *Advances in Neural Information Processing Systems*, 2020.

Angela Fan, Edouard Grave, and Armand Joulin. Reducing transformer depth on demand with structured dropout. In *International Conference on Learning Representations*, 2019.

Leo Gao, Jonathan Tow, Baber Abbasi, et al. A framework for few-shot language model evaluation. `https://github.com/EleutherAI/lm-evaluation-harness`, 2023.

Xavier Gonzalez, Andrew Warrington, Jimmy T.H. Smith, and Scott W. Linderman. Towards scalable and stable parallelization of nonlinear rnns. *arXiv preprint arXiv:2407.19115*, 2024.

Aaron Grattafiori, Abhinav Dubey, Abhinav Jauhri, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Stefanie Günther, Lars Ruthotto, Jacob B. Schroder, Eric C. Cyr, and Nicolas R. Gauger. Layer-parallel training of deep residual neural networks. *SIAM Journal on Mathematics of Data Science*, 2(1):1–23, 2020.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European Conference on Computer Vision*, 2016.

Geoffrey Hinton. The forward-forward algorithm: Some preliminary investigations. *arXiv preprint arXiv:2212.13345*, 2022.

Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *European Conference on Computer Vision*, 2016.

Zhouyuan Huo, Bin Gu, and Heng Huang. Training neural networks using features replay. In *Advances in Neural Information Processing Systems*, 2018a.

Zhouyuan Huo, Bin Gu, Qian Yang, and Heng Huang. Decoupled parallel backpropagation with convergence guarantee. In *International Conference on Machine Learning*, 2018b.

Max Jaderberg, Wojciech Marian Czarnecki, Simon Osindero, Oriol Vinyals, Alex Graves, David Silver, and Koray Kavukcuoglu. Decoupled neural interfaces using synthetic gradients. In *International Conference on Machine Learning*, 2017.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Fractalnet: Ultra-deep neural networks without residuals. In *International Conference on Learning Representations*, 2017.

Timothy P Lillicrap, Daniel Cownden, Douglas B Tweed, and Colin J Akerman. Random synaptic feedback weights support error backpropagation for deep learning. *Nature Communications*, 2016.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Arild Nøkland. Direct feedback alignment provides learning in deep neural networks. In *Advances in Neural Information Processing Systems*, 2016.

David Raposo, Sam Ritter, Blake Richards, Timothy Lillicrap, Peter Conway Humphreys, and Adam Santoro. Mixture-of-depths: Dynamically allocating compute in transformer-based language models. *arXiv preprint arXiv:2404.02258*, 2024.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. Highway networks. *arXiv preprint arXiv:1505.00387*, 2015.

Corentin Tallec and Yann Ollivier. Unbiased online recurrent optimization. *arXiv preprint arXiv:1702.05043*, 2017.

Corentin Tallec and Yann Ollivier. Unbiasing truncated backpropagation through time. In *International Conference on Learning Representations Workshop*, 2018.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. Alpaca: A strong, replicable instruction-following model. `https://crfm.stanford.edu/2023/03/13/alpaca.html`, 2023.

Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. In *IEEE/CVF International Conference on Computer Vision*, 2021.

Andreas Veit, Michael J Wilber, and Serge Belongie. Residual networks behave like ensembles of relatively shallow networks. In *Advances in Neural Information Processing Systems*, 2016.

Xin Wang, Fisher Yu, Zi-Yi Dou, Trevor Darrell, and Joseph E. Gonzalez. Skipnet: Learning dynamic routing in convolutional networks. In *European Conference on Computer Vision*, 2018.

An Yang, Anfeng Li, Baosong Yang, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T. Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. Metamath: Bootstrap your own mathematical questions for large language models. *arXiv preprint arXiv:2309.12284*, 2023.

Hongyi Zhang, Yann N. Dauphin, and Tengyu Ma. Fixup initialization: Residual learning without normalization. In *International Conference on Learning Representations*, 2019.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, David Dohan, and Denny Zhou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023.

A PROOFS

Throughout, blocks are indexed in backward processing order (block 1 next to the terminal cotangent g; block n at the input), groups $G_1, \dots, G_m$ partition $[n]$ into contiguous segments in that order, and products of row-covector actions compose left to right. For $S = \{i_1 < \cdots < i_k\}$ the path monomial $P_S := A_{i_1} A_{i_2} \cdots A_{i_k}$ is taken in processing order, so that $H_{\text{exact}} = \sum_{S \subseteq [n]} g P_S$ with $P_\emptyset = I$.

A.1 PROOF OF THEOREM 1

Consider one group factor $I + \sum_{i \in G_r} A_i$. Expanding the product

$$H_\Pi \;=\; g \prod_{r=1}^{m} \Big(I + \sum_{i \in G_r} A_i \Big)$$

amounts to choosing, for each group r, either I or exactly one operator A_i with $i \in G_r$. A choice is therefore a subset $S \subseteq [n]$ with $|S \cap G_r| \le 1$ for every r; because the groups are contiguous and ordered, the product of the chosen operators is exactly the ordered product P_S. Hence

$$H_\Pi \;=\; \sum_{\substack{S \subseteq [n] \\ |S \cap G_r| \le 1,\ \forall r}} g P_S \;=\; \sum_{S \in \mathcal{R}(\Pi)} g P_S.$$

Since H_{exact} sums over *all* subsets, the complement of $\mathcal{R}(\Pi)$ in $2^{[n]}$ is the family of subsets with $|S \cap G_r| \ge 2$ for at least one r, which gives equation 5. □

A.2 ARRIVAL AND ERROR DECOMPOSITION (THEOREM 2)

This positional view refines Theorem 1: omitted paths at a position—two branches in one earlier group, or a branch from the position's own group on the output side—are propagated through the exact local VJP $\mathcal{B}_v$, so positions inside one group share an approximate entrance cotangent. The formal statement and its propagation law follow.

Theorem 2 (Arrival and error decomposition)**.** *Let v lie in group $G_{r(v)}$ and let $D_v = \{i : i < v\}$ be the blocks processed before v (the output side of v). The entrance cotangent of v under equation 3 is*

$$h_v^{\text{sched}} \;=\; g \prod_{r < r(v)} \Big(I + \sum_{i \in G_r} A_i \Big), \tag{8}$$

i.e. the path gP_S arrives at v iff (i) $S \subseteq D_v$, (ii) $S \cap G_{r(v)} = \varnothing$, and (iii) $|S \cap G_r| \le 1$ for all $r < r(v)$. The entrance error $\Delta_v := h_v^{\text{exact}} - h_v^{\text{sched}}$ is the sum over the omitted families: type A*, sets $S \subseteq D_v$ with at least two branches in one earlier group G_r, $r < r(v)$; and* type B*, sets $S \subseteq D_v$ using at least one branch of v's own group $G_{r(v)}$. The two families overlap (a set may violate both conditions); Δ_v is the sum over their* union*, and no term is counted twice. Writing $\Gamma_v = \mathcal{B}_v(h)$ for the local VJP map at v (linear in its entrance cotangent h), the parameter-gradient error is*

$$\Gamma_v^{\text{exact}} - \Gamma_v^{\text{sched}} = \mathcal{B}_v(\Delta_v). \tag{9}$$

Proof. Write $r(v)$ for the group containing v. In the grouped backward (Algorithm 1) the cotangent entering group $G_{r(v)}$ is produced only by the groups processed before it (the groups G_r with $r < r(v)$):

$$h_v^{\text{sched}} \;=\; g \prod_{r < r(v)} \Big(I + \sum_{i \in G_r} A_i \Big).$$

Expanding as in the proof of Theorem 1, a path gP_S appears in h_v^{sched} iff S draws at most one branch from each group G_r with $r < r(v)$ and none from $G_{r(v)}$. Every block of such a group satisfies $i < v$, so conditions (i)–(iii) of the theorem follow; conversely any S satisfying (i)–(iii) is generated. The exact entrance is $h_v^{\text{exact}} = g \prod_{i<v}(I + A_i) = \sum_{S \subseteq D_v} g P_S$. Comparing the two expansions, the missing terms are exactly those $S \subseteq D_v$ that either (a) use two branches of one earlier group G_r, $r < r(v)$ (type A) or (b) use a branch of $G_{r(v)}$ (type B)—or both; Δ_v is the sum over the union of the two families, so overlapping terms appear once.

For equation 9, observe that the parameter gradient of block v is linear in its entrance cotangent: $\Gamma_v = \mathcal{B}_v(h)$, where $\mathcal{B}_v$ is the linear VJP map of block v's local Jacobian with respect to its parameters. Both the exact and the scheduled computation apply the *same* local map to their respective entrances, so subtracting gives $\Gamma_v^{\text{exact}} - \Gamma_v^{\text{sched}} = \mathcal{B}_v(h_v^{\text{exact}} - h_v^{\text{sched}}) = \mathcal{B}_v(\Delta_v)$. □ □

Error propagation across a group. Write Δ_q^{in} and Δ_q^{out} for the entrance and exit errors of a unit q. If q is processed as a full residual unit with its own factor $(I + A_q)$, then

$$\Delta_q^{\text{out}} = \Delta_q^{\text{in}}(I + A_q) = \Delta_q^{\text{in}} + \Delta_q^{\text{in}} A_q.$$

If q belongs to a group G whose exit receives the shared entrance, the group error also absorbs the type-A content deleted inside G:

$$\Delta^{\text{out}} = \Delta^{\text{in}}\Big(I + \sum_{i\in G} A_i\Big) + \Delta_{\text{group}}, \qquad \Delta_{\text{group}} := h^{\text{in}}\Big[\prod_{i\in G}(I + A_i) - \Big(I + \sum_{i\in G} A_i\Big)\Big],$$

where Δ_{group} is a definition: the content that the grouped factor drops relative to the exact factor at the group's own entrance h^{in} (it is the type-A contribution of G). These identities are used in the error analysis (they quantify how an entrance error propagates toward the input) and are not executed by Algorithm 1, which computes no error terms.

A.3 Order bound and dominant error

Proposition 2 (Dominant error term)**.** *The lowest-order part of Δ_Π consists of the intra-group second-order terms: $\sum_r \binom{|G_r|}{2}$ pair terms gA_iA_j with $i \neq j$ in a common group. All retained pair terms are exactly the cross-group pairs.*

Order bound. A set $S \in \mathcal{R}(\Pi)$ picks at most one branch from each of m groups, hence $|S| \leq m$; so $\mathcal{R}(\Pi) \subseteq \{S : |S| \leq m\}$. Conversely, every S with $|S| \leq m$ whose branches lie in distinct groups is retained, but sets with two branches in one group are excluded, so the inclusion is generally strict. □

Dominant error term. By equation 5, the omitted paths of order exactly 2 are the pairs $\{i, j\}$ with $i \neq j$ in a common group. The number of such pairs in group G_r is $\binom{|G_r|}{2}$, giving $\sum_r \binom{|G_r|}{2}$ omitted pairs. All cross-group pairs satisfy $|S \cap G_r| \leq 1$ and are retained. □

A.4 Proof of Proposition 1

Each block index appears in exactly one group; Algorithm 1 runs one VJP per block per step, so $W = n$ for every partition and every bundle granularity b (a bundle of b consecutive blocks is one VJP unit that internally composes exactly; the count of block-VJPs is unchanged). The group loop visits m groups sequentially and the members of a group are dependency-independent (Definition 1), so with enough workers the span is one group per level, $D \approx m \cdot b$. For $m = n$ and $b = 1$ every group factor is $I + A_i$, so $H_\Pi = H_{\text{exact}}$ and the overhead is zero. □

A.5 A bias bound (optional refinement)

For residual branches scaled by α, Theorem 1 gives the exact omitted polynomial; writing $gP_S = O(\alpha^{|S|})$, the leading error of the full operator is $O(\alpha^2)$ with constant $\sum_r \binom{|G_r|}{2}$ (Proposition 2). For the entrance error of Theorem 2 the situation differs: type-B families contain first-order terms (a single branch of v's own group), so Δ_v may contain $O(\alpha)$ contributions. Either way, with $\|\mathcal{B}_v\|$ denoting the operator norm of the local VJP map,

$$\|\Delta_v\| \;\leq \sum_{\substack{S\subseteq D_v \\ S\notin\mathcal{R}(\Pi)_v}} \|gP_S\| \;\leq \sum_{r\geq 1}\Big|\big\{S \subseteq D_v : |S| = r,\ S \notin \mathcal{R}(\Pi)_v\big\}\Big| \cdot \max_S \|gP_S\|,$$

$$\|\Gamma_v^{\text{exact}} - \Gamma_v^{\text{sched}}\| \leq \|\Delta_v\| \cdot \|\mathcal{B}_v\|,$$

where $\mathcal{R}(\Pi)_v$ denotes the family that arrives at v (conditions (i)–(iii) of Theorem 2) and the maximum is over the finite set of omitted monomials. This is a structural statement about the polynomial: it bounds the error in terms of the sizes of the omitted families and the magnitudes of their terms, without assuming that low-order terms dominate on any particular task.

A.6 POST-RESIDUAL TRANSFORMATIONS

The main derivation uses the clean residual setting (Section 3). In general a residual block can be written as a residual branch followed by a post-residual transformation,

$$f_i(x) = K_i(x + \phi_i(x)), \qquad \frac{\partial f_i}{\partial x} = K_i (I + J_i), \tag{10}$$

where J_i is the Jacobian of the residual branch and K_i is the Jacobian of the transformation applied after the residual addition (e.g. a post-activation or normalization). The clean setting is $K_i = I$, giving backward factor $I + A_i$ with $A_i = J_i$; this is the case treated by the theorems above. For $K_i \neq I$, the exact group-level analogue of the retained-path rule is obtained by preserving the K transport and allowing at most one residual insertion per group. For a contiguous group $G = [a, b]$, it is

$$\widetilde{T}_G = \prod_{i=a}^{b} K_i + \sum_{j=a}^{b} \left(\prod_{i=a}^{j-1} K_i \right) K_j J_j \left(\prod_{i=j+1}^{b} K_i \right), \qquad h_{\text{out}} = h_{\text{in}} \widetilde{T}_G. \tag{11}$$

The products are ordered in the same backward-processing convention as above. Thus K is not itself replaced by a sum, while all terms containing two or more J insertions within one group are omitted. For a post-activation block, $K_i = D_i = \operatorname{diag}(\mathbf{1}[u_i > 0])$ at the current forward point, while batch-normalization operations inside the residual branch belong to J_i, not to K_i. The post-activation ResNet-110 and ResNet-1202 tracks (Section 5.3) implement this $K_i \neq I$ case with the singleton-bundle setting used by the engine.

B RELATION TO HIGHWAY-BP

For this comparison we use the standard path-order interpretation of Highway-BP: after k iterations, the estimate contains the path terms up to order k. We do not require a literal identification of the scalar sum of the A_i with the noncommutative product in Eq. equation 1; all GLBP statements are made directly for the ordered path polynomial.

Highway-BP keeps the exact backward operator and truncates its iterative (Neumann-type) solution after k iterations; the estimate then coincides with the sum of all path terms of order at most k, i.e. $\mathcal{R}_{\text{HB}}(k) = \{S : |S| \leq k\}$. GLBP replaces the operator by the grouped operator L_Π induced by equation 3 and solves the resulting nilpotent system exactly in one sweep. Concretely, L_Π is the strictly lower-triangular (hence nilpotent) operator on the group-boundary cotangents whose single application propagates a cotangent across one group boundary, $w \mapsto w\big(\sum_{i \in G_r} A_i\big)$, following the same output-to-input processing order as Algorithm 1. Because there are m groups, L_Π is nilpotent of index $m + 1$ ($L_\Pi^{m+1} = 0$), so the Neumann series terminates and $H_\Pi = (I - L_\Pi)^{-1} g = \sum_{r=0}^{m} L_\Pi^r g$ is exact in a single sweep. In short: *Highway-BP truncates the solver; GLBP coarsens the operator*. Both methods lie on the same path–cost surface but spend different currencies: Highway-BP spends compute to recover paths by order, GLBP spends span to recover paths by grouping structure. Since the grouped operator is nilpotent, GLBP's solution is exact for H_Π after one sweep, and the residual error against H_{exact} is precisely Δ_Π of equation 5.

Lemma 1 (Inclusion relation)**.** *For $m = k$, $\mathcal{R}(\Pi) \subseteq \mathcal{R}_{\text{HB}}(k) = \{S : |S| \leq k\}$, with equality iff $m \in \{1, n\}$ (for $n \geq 2$). The difference $\mathcal{R}_{\text{HB}}(m) \setminus \mathcal{R}(\Pi)$ consists of the intra-group interaction paths of Theorem 1 of order at most m.*

Proposition 3 (Coincidence at first order)**.** *At $k = 1$ and $m = 1$, Highway-BP and GLBP produce the same retained family (the identity term and all first-order terms). For $m = k > 1$ the families separate: the same-group exclusion removes intra-group pairs that the order-m family of Highway-BP still contains.*

Proof. Coincidence. At $m = k = 1$ both retained families are $\{\emptyset\} \cup \{\{i\} : i \in [n]\}$. For $m = k > 1$, the pair drawn from any group of size ≥ 2 lies in $\mathcal{R}_{\text{HB}}(m)$ but not in $\mathcal{R}(\Pi)$, so the families separate. *Inclusion.* Every $S \in \mathcal{R}(\Pi)$ satisfies $|S| \leq m$, so $\mathcal{R}(\Pi) \subseteq \mathcal{R}_{\text{HB}}(m) = \{S : |S| \leq m\}$. The difference $\mathcal{R}_{\text{HB}}(m) \setminus \mathcal{R}(\Pi)$ is the family of sets with $|S| \leq m$ that violate the same-group condition for some G_r—i.e. the intra-group interaction paths of order at most m. For the equality cases: if $m = n$ all groups are singletons and both families are $2^{[n]}$; if $m = 1$ then $\mathcal{R}(\Pi) = \{S : |S| \leq 1\} =$

$\mathcal{R}_{\mathrm{HB}}(1)$, again equality. Conversely, if $1 < m < n$, some group has at least two blocks and the pair drawn from it has order $2 \le m$, so it lies in $\mathcal{R}_{\mathrm{HB}}(m) \setminus \mathcal{R}(\Pi)$ and the inclusion is strict. Hence equality holds iff $m \in \{1, n\}$. □ □

Table 5: GLBP and Highway-BP in the same path–cost space. VJP counts include the final parameter-gradient sweep, giving $(k+1)n$ for Highway-BP.

	Highway-BP	GLBP
What is altered	solver truncation (k)	operator ($L \to L_\Pi$)
Retained paths	$\{S : \lvert S\rvert \le k\}$	$\{S : \lvert S \cap G_r\rvert \le 1\}$
Path family	order-based, isotropic	group-based, anisotropic
Block-VJPs per step	$(k+1)n$	n
Exact recovery	$k \ge n$, cost $(n{+}1)n$	$m = n$, cost n
Parallelism source	iteration-level, device-free	group-level, device-free

C Counting

C.1 Retained-path counts

Let $s_r = |G_r|$ be the group sizes ($\sum_r s_r = n$). Each group contributes either the identity or one of its s_r branches, so

$$N_{\mathrm{keep}} \;=\; \prod_{r=1}^{m}(s_r+1), \qquad N_{\mathrm{err}} \;=\; 2^n - \prod_{r=1}^{m}(s_r+1).$$

The generating function $\prod_r(1+s_r x)$ expands to $\sum_k e_k(s_1,\dots,s_m)x^k$, where e_k is the elementary symmetric function; the coefficient of x^k counts the retained paths of order k. In particular the number of retained 2-order paths is $e_2 = \sum_{r<t} s_r s_t$ (cross-group pairs), and the number of omitted pairs is $\binom{n}{2} - e_2 = \sum_r \binom{s_r}{2}$, confirming Proposition 2.

C.2 Cut sets that cover a given path

Fix a path $S = \{i_1 < i_2 < \cdots < i_d\}$. There are $n-1$ candidate cut positions (cut t lies after block t). A cut t separates the consecutive selected blocks i_j and i_{j+1} iff $i_j \le t < i_{j+1}$; the number of such positions is $s_j := i_{j+1} - i_j \ge 1$. S is retained by a cut set iff every gap contains at least one cut. The number of cut sets with this property is

$$\#\{\text{cut sets covering } S\} \;=\; 2^{(n-1)-\sum_j s_j} \prod_j \big(2^{s_j}-1\big),$$

because positions outside the gaps are free ($2^{(n-1)-\sum s_j}$ choices) and each gap must receive a nonempty subset of its s_j separating positions ($2^{s_j}-1$ choices). For $d=2$ this reduces to $2^{n-1-s}(2^s-1)$ with $s = i_2 - i_1$, and for $d=3$ to $2^{n-1-s_1-s_2}(2^{s_1}-1)(2^{s_2}-1)$, matching direct enumeration. In particular adjacent pairs ($s_j = 1$) are the hardest to cover: they require the single cut between the two blocks, and are covered by exactly half of all cut sets.

C.3 Cut-set families at fixed group count

A partition of $[n]$ into m contiguous groups is specified by choosing $m-1$ cut positions among the $n-1$ candidates, so $|\mathcal{F}_m| = \binom{n-1}{m-1}$. Enumerating $\mathcal{F}_m$ in lexicographic order and applying one partition per step yields a deterministic cycle; the union of retained families over a full cycle covers every path of order up to m, but per Remark 1 this coverage does not make the per-step gradients unbiased.

C.4 THE COEFFICIENT CONDITION FOR COVERAGE

Fix a schedule $(\Pi_t)_{t=1}^T$ with nonnegative aggregation weights ω_t (for example, $\omega_t = 1/T$ for a uniform average). Because each grouped backward is linear in the terminal cotangent, the aggregated operator is again a path polynomial,

$$\sum_{t=1}^{T} \omega_t \, H_{\Pi_t} \;=\; \sum_{S \subseteq [n]} c_S \, g P_S, \qquad c_S \;=\; \sum_{t=1}^{T} \omega_t \, \mathbf{1}[\, S \in \mathcal{R}(\Pi_t)\,].$$

A path S is *covered* by the schedule iff $c_S > 0$, i.e. some step retains it. The aggregated operator equals the exact backward operator $H_{\text{exact}} = \sum_S g P_S$ iff $c_S = 1$ for every $S \subseteq [n]$. Coverage ($c_S > 0 \; \forall S$) is therefore necessary but not sufficient for this coefficient condition: a full lexicographic cycle achieves $c_S > 0$ for all S of order up to m, yet the individual coefficients generally differ from 1, so no finite schedule makes a single step unbiased. This is the formal content of Remark 1.

D WORKED EXAMPLES

D.1 THE FOUR-BLOCK EXPANSION

For $n = 4$ the exact expansion has 16 terms. With the cut after block 2 ($G_1 = \{1, 2\}$, $G_2 = \{3, 4\}$), $H_\Pi = g(I + A_1 + A_2)(I + A_3 + A_4)$ retains 9 terms:

Order	Retained	Omitted	Total
0	1 (g)	–	1
1	4 ($gA_1, \ldots, gA_4$)	–	4
2	4 (cross pairs $\{1,3\}, \{1,4\}, \{2,3\}, \{2,4\}$)	2 (same-group pairs $\{1,2\}, \{3,4\}$)	6
3	0	4	4
4	0	1	1
Total	9	7	16

For comparison, Highway-BP at $k = 1$ retains 5 terms (order ≤ 1), at $k = 2$ retains 11 (order ≤ 2), at $k = 3$ retains 15, and at $k = 4$ the full 16. GLBP with $m = 2$ thus covers 9 of 16 terms at 4 block-VJPs; Highway-BP covers 11 at 12 block-VJPs ($k = 2$).

D.2 A SCHEDULE THAT COMPLETES THE MISSING INTERACTIONS ($n = 3$)

Consider $n = 3$. Under one group $\{1, 2, 3\}$ the retained family is $\{\emptyset, \{1\}, \{2\}, \{3\}\}$; the pairs and triple are absent. Switching to $\{1, 2\}, \{3\}$ restores gA_1A_3, gA_2A_3 and the pair $\{1, 2\}$ remains absent; switching to $\{1\}, \{2, 3\}$ restores gA_1A_2. This illustrates Remark 1: across steps every path of order at most 2 can be covered by these two-group partitions (the triple $\{1, 2, 3\}$ requires $m \geq 3$), yet no single step computes the exact gradient.

E EXPERIMENTAL DETAILS

Small models (Sections 5.1–5.3). CIFAR-10 with standard augmentation (random crop, horizontal flip). PreactResNet-32 has $n = 15$ residual blocks; ResNet-110 and ResNet-1202 use staged realizations with $n = 54$ and $n = 600$, respectively. Dimension-changing transitions are handled outside the grouped stage bodies. Here `pps` denotes the number of position groups within each stage body, not the total number of groups in the network; ResNet-110 has stage-body sizes $18/17/17$, while ResNet-1202 has $200/199/199$. The first block of the latter two stages is a dimension-changing transition and is handled by standard sequential backpropagation outside the grouped body. The reported deep comparisons use Standard BP and GLBP with $b = 1$ and `pps`$= 4, 6$; pps$= 9$, progressive, and singleton-boundary runs are not included. All runs use 200 epochs, batch size 128, SGD with momentum 0.9, weight decay 10^{-4}, and a MultiStep learning-rate schedule with drops at epochs 100 and 150. The learning rate is 0.1 for post-activation tracks, 0.03 for pre-activation

ResNet-110, and 0.01 for pre-activation ResNet-1202. No learning-rate warmup or gradient clipping is used in these reported tracks. Pre-activation and post-activation results are compared only within their respective tracks.

Highway-BP comparison (Sections 5.1 and 5.3). The ResNet-32 Highway-BP sweep uses the Highway-BP–aligned pre-activation track with $n = 15$, $b = 1$, and $k \in \{1, 2, 3, 4\}$ under the same CIFAR-10 training recipe as its Standard BP/GLBP references. Its per-step cost is $(k + 1)n = 30, 45, 60, 75$ block-VJPs, including the final full parameter-gradient sweep. These runs form a same-protocol comparison sweep rather than a strictly paired trajectory. The $n = 54$, $k = 2$ point is a separate Highway-BP–aligned space-to-depth reference track, not a paired baseline for the staged ResNet-110 experiments; it costs $(2 + 1) \cdot 54 = 162$ block-VJPs per step. We use Highway-BP here as a local quality–cost reference, not as a separate deep or LLM evaluation.

LLM fine-tuning (Section 5.4). LLaMA-3.1-8B (from the LLaMA family (Grattafiori et al., 2024)), 32 layers, four GPUs with 8 layers per GPU, bfloat16. Tasks: Alpaca (Taori et al., 2023) (52K examples, sequence length 512, 1,219 optimizer steps) evaluated with IFEval (Zhou et al., 2023); Evol-CodeAlpaca (111K, sequence length 1216, 1,738 optimizer steps) evaluated with HumanEval (Chen et al., 2021); MetaMathQA (Yu et al., 2023) (395K, sequence length 512, 4,000 optimizer steps) evaluated with GSM8K (Cobbe et al., 2021) (strict and flexible extraction). AdamW (Kingma & Ba, 2015; Loshchilov & Hutter, 2019) with learning rate $2{\cdot}10^{-5}$, linear warmup plus cosine decay, gradient clipping 1.0, effective batch 128 (the code task uses micro-batch 16 with 8 accumulation steps to fit longer sequences; the effective batch is unchanged). Fixed GLBP uses $b = 4$ and $m = 4$. Progressive GLBP uses $b = 4$, hence $B = 32/4 = 8$ bundles, and schedules $m \in \{4, 5, 6, 8\}$. The four stages receive proportional weights $(4, 5, 6, 8)$: for a task with T optimizer steps, the ideal stage lengths are

$$\widetilde{T}_j = T \frac{w_j}{\sum_{\ell=1}^{4} w_\ell}, \qquad (w_1, w_2, w_3, w_4) = (4, 5, 6, 8),$$

with integer lengths obtained by largest-remainder rounding. This gives stage lengths $(212, 265, 318, 424)$ for IFEval, $(302, 378, 453, 605)$ for HumanEval, and $(696, 870, 1043, 1391)$ for GSM8K. Cut positions rotate within each stage. Evaluation follows lm-eval-harness (Gao et al., 2023) defaults per task. The untuned base model is evaluated with the same protocol.

System instance (Section 5.5). Qwen3-8B (Yang et al., 2025) ($n = 36$ layers) uses $b = 9$ layers per bundle, giving $B = \lceil n/b \rceil = 4$ bundles. The GPU-boundary partition has $m = 2$ groups, with cuts after bundle 3: the first group contains the three prefix bundles and the final group is an exact singleton. Qwen3-32B ($n = 64$ layers) uses $b = 8$, giving $B = 8$ bundles; its GPU-boundary partition has $m = 3$ groups with cuts after bundles 6 and 7, leaving two exact singleton tail groups. Thus the two instances use the same GLBP group algebra with different (b, B, m, cuts) settings. Both use bfloat16, sequence length 512, Alpaca, AdamW, and global gradient clipping at 1.0; the batch sizes are 4 and 2, and the learning rates are $2 \cdot 10^{-5}$ and 10^{-5}, respectively, with warmup. Bundle entrances are detached at the GPU-stage boundaries to expose each stage as an independent subgraph; the inter-group cotangent is then re-fed across these boundaries so the inter-group product is preserved: the boundary detachment isolates subgraphs for staging, while the re-fed cotangent leaves the inter-group gradient intact. The approximate prefix groups are evaluated on separate CUDA streams while the exact singleton tail groups are processed sequentially; the prefix streams are synchronized before the step returns. Step time covers forward, loss, backward, gradient clipping, and the optimizer-step call, while excluding data loading, learning-rate updates, and gradient clearing. The timing boundaries synchronize the current CUDA device; the reported mean drops the first five timing samples and averages the remaining 1,995 steps of each 2,000-step run.

F GRADIENT DIAGNOSTICS

On fixed synthetic probe batches for three models (an $n = 8$ linear chain, PreactResNet-32 with $n = 15$, and a small Transformer with $n = 12$), the cosine similarity between the GLBP gradient and the exact gradient rises monotonically with the group count m. On PreactResNet-32, cosine is 0.833, 0.912, and 0.948 at $m = 1, 5, 8$, respectively; the corresponding relative ℓ_2 errors are 0.554,

Table 6: GLBP system instance on GPU-boundary cuts (Alpaca, 2,000 steps, bfloat16). End-to-end step time including forward, backward, and optimizer. Final loss is the mean over the last 50 optimizer steps (single seed). This is a constrained hardware case study: cuts are fixed to device boundaries and the device count is fixed.

Model	GPUs	Step time (ms)		Final loss	
		Standard	GLBP inst.	Standard	GLBP inst.
Qwen3-8B (36 layers)	4	922	690	1.089	1.097
Qwen3-32B (64 layers)	8	2085	1282	0.997	1.032

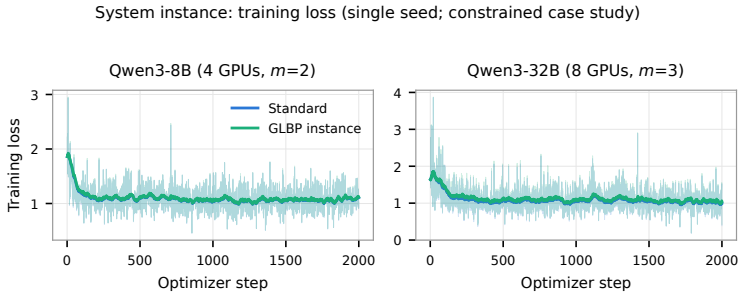


Figure 5: Training loss of the GLBP system instance versus Standard BP on the two models of Section 5.5. Curves track closely throughout; this is a constrained hardware case study (fixed device count, GPU-boundary cuts), not a core result.

0.410, and 0.319 (Figure 6). Figure 6 reports cosine only; the relative ℓ_2 values are supplementary diagnostics.

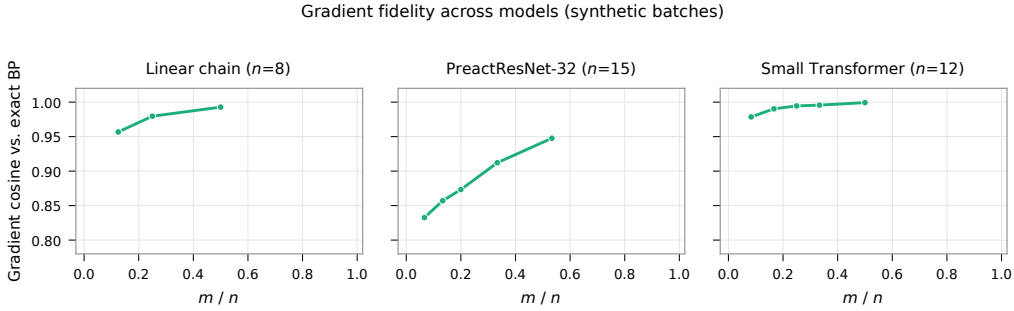


Figure 6: Gradient cosine against exact backpropagation vs. normalized group count m/n on three diagnostic models (synthetic batches). Fidelity rises monotonically with m.

G Notes on cost accounting

All block-VJP counts include the final parameter-gradient sweep. For Highway-BP this gives $(k+1)n$ block-VJPs for k Neumann iterations rather than the kn figure that omits the sweep; both methods are counted with the same convention. The dependency-graph span reported for GLBP, $m \cdot b$, is the ideal parallel span of the backward graph and is not a wall-clock measurement; the bench-scale experiments intentionally report no wall-clock comparisons.

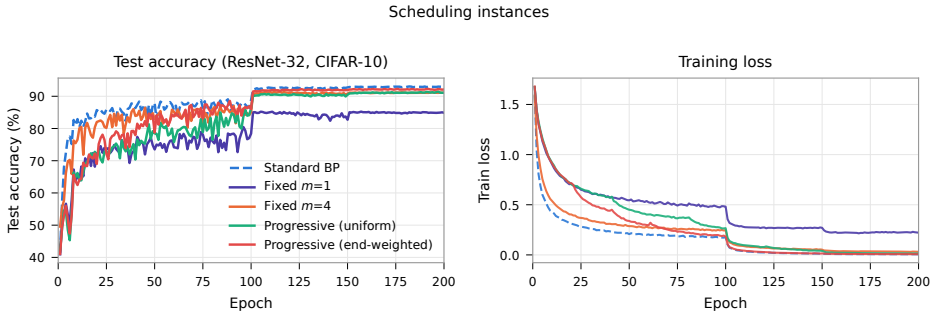


Figure 7: Scheduling instances on ResNet-32 (CIFAR-10, 200 epochs). Left: test accuracy; right: training loss. Progressive schedules increase m over training; all arms cost 15 block-VJPs per step.

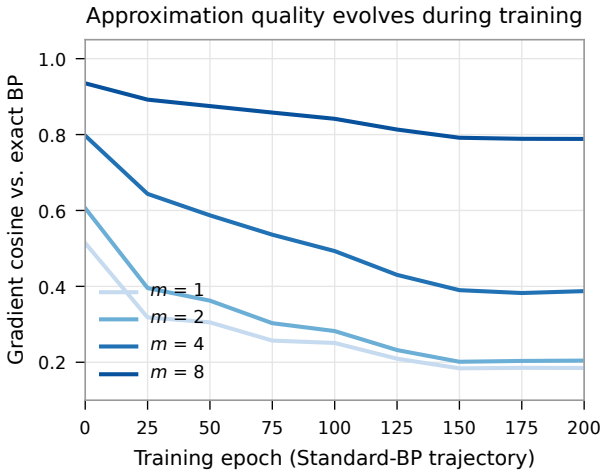


Figure 8: Approximation quality over training: cosine between the GLBP(m) gradient and the exact gradient at nine checkpoints on a Standard BP trajectory (PreactResNet-32, fixed probe batch). Larger m degrades more slowly.