RECAMPLIFY: LARGE-SCALE PRETRAINING OF RECURSIVE PROTEIN LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Protein language models (pLMs) are powerful tools for protein annotation and mutation-effect prediction, but progress has largely depended on scaling model size and data, making them expensive and prone to memorization. Test-time scaling offers a complementary path, but methods developed for autoregressive LLMs do not transfer directly: masked encoders lack a generation loop, and amino acids do not verbalize in a reasoning language. Instead, we adapt recursive pretraining to masked pLMs, where a weight-tied Transformer core refines residue-aligned hidden states over an adjustable number of cycles. We instantiate this recipe as recAMPLIFY, a pLM with 155M unique parameters trained on the same 2T tokens as AMPLIFY-350M. Despite 56% fewer parameters, it improves CASP contact recovery and three out of four supervised tasks, notably raising fold accuracy by 22.8%. Because optimal depth for recAMPLIFY varies across tasks and assays, we then introduce *Gated Mixture Recurrence* (GMR), a framework that decodes each protein from a probability-weighted mixture of hidden states visited during recursion. GMR employs a learned residual gate at scalar, low-rank, or per-dimension resolution to control how much of each cycle's proposed update is accepted into the state. This unifies halting and gating into a single adaptive computation objective while recovering PonderNet-style halting as a special case without gates. Layered on top of recAMPLIFY, GMR further reduces expected layer depth by up to 37%, although the accuracy trade-off varies by task. Mechanistic analyses show that residual gating controls recurrence stability beyond the trained depth and that halt depth tracks biological properties.

1 INTRODUCTION

Proteins are sequences of amino acids shaped by evolutionary pressure to code diverse biological functions, including the ability to form specific folded structures. Protein language models (pLMs) attempt to learn these constraints directly from sequence (Rives et al., 2021; Elnaggar et al., 2022; Nijkamp et al., 2023; Ferruz et al., 2022), typically using bidirectional Transformer (Vaswani et al., 2017) encoders trained with masked language modeling (MLM) (Devlin et al., 2019). By predicting masked residues from their surrounding context, these models must capture essential interactions, including those that are far apart in sequence but close in 3D space. The resulting conditional distributions enable zero-shot mutation scoring and contact recovery, while contextual representations enable downstream classification and regression tasks.

Recent progress in pLMs has largely been driven by scaling both model size and training data (Lin et al., 2023; Nijkamp et al., 2023; Candido et al., 2026). However, parameter count alone does not determine performance. AMPLIFY-350M (Fournier et al., 2024), despite having $43\times$ fewer parameters than ESM-2 15B (Lin et al., 2023), achieves comparable pseudo-perplexity on its curated validation set and better residue recovery on the human proteome. Several works further indicate that scaling can lead to undesirable memorization in larger pLMs (Cheng et al., 2024; Fournier et al., 2024; Hou et al., 2026). These observations motivate test-time scaling and adaptive computation as complementary axes to increase compute while keeping the number of parameters fixed.

Test-time compute scaling in autoregressive LLMs typically relies either on generating verbalized reasoning tokens (Wei et al., 2022) or iteratively updating continuous latent states (Dehghani et al., 2019; Giannou et al., 2023; Hao et al., 2025). For masked pLMs with no autoregressive generation

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.

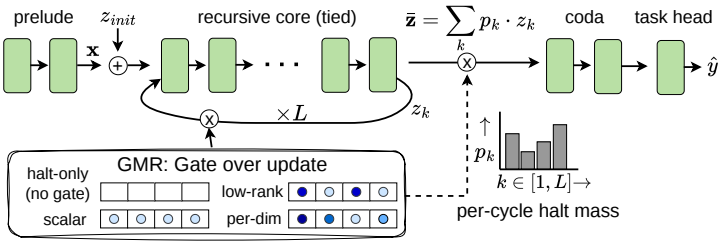


Figure 1: **recAMPLIFY architecture.** A non-recurrent prelude maps input embeddings to a residue-aligned latent state z_k, a weight-tied Transformer core iteratively refines this state, and a non-recurrent coda maps the refined state to the output space. Gated Mixture Recurrence (GMR) decodes from a probability-weighted mixture over the recurrence trajectory, with a per-sequence residual gate at scalar, low-rank, or per-dimension channel resolution controlling how much of each cycle's proposed update is accepted. Blue circles illustrate these channel-level parameterizations.

loop, the discrete-token approach does not transfer. Furthermore, amino-acid sequences do not admit verbalized rationales, and inserting thought tokens would break the residue-to-state alignment required for masked prediction. In contrast, iterative refinement of continuous latent representations, hereafter latent refinement, offers a natural way to increase pLM computation. Latent refinement already has a strong precedent in protein structure prediction, where AlphaFold (Jumper et al., 2021) and ESMFold (Lin et al., 2023) recycle internal representations for a predefined number of cycles. However, these recycles operate on structural rather than sequence representations, leaving input-dependent test-time latent refinement unexplored in pLMs.

To approach test-time latent refinement, we hold a set of parameters as a recurrent unit (Nakano, 1972; Amari, 1972) that iteratively refines its hidden states in a recursive fashion. Building on depth-recurrent language models (Dehghani et al., 2019; Giannou et al., 2023; Geiping et al., 2025), we adapt a recursive pretraining framework for test-time scaling in pLMs. Applied to AMPLIFY, this framework yields *recursive AMPLIFY* (recAMPLIFY) that separates effective model depth from unique parameter count. In recAMPLIFY, a non-recurrent *prelude* first maps token embeddings into a residue-aligned latent representation space where recursion can be unrolled. A weight-tied *core* refines this state over a number of recurrent cycles, increasing compute without changing sequence length or residue alignment. A non-recurrent *coda* then maps the refined state to the MLM output space for masked-residue prediction (Figure 1). Because the core can be reapplied, the number of recurrent cycles, and therefore the effective depth, remains adjustable after pretraining.[1]

Exposing recurrence depth as a test-time variable leaves open how to allocate the compute budget across inputs: fixed-depth recursion applies the same number of cycles to every input, yet we observe task- and assay-specific performance optima rather than monotonic gains from more cycles (Figures 2 and 3b). An effective recursion mechanism should thus decide both *when* to stop the update and *how* each cycle's update should shape the hidden state. Recent work in limited data regimes addresses the *when* question by training halting heads from verifiable answers (Jolicoeur-Martineau, 2025). However, no such verifier exists for MLM since determining which residues are biologically valid at a masked position is itself the problem the pLM is being trained to solve. Instead, we propose *Gated Mixture Recurrence* (GMR), which decodes from a probability-weighted mixture of the hidden states visited during recursion and gates each cycle's residual update by a learned survival signal at a *scalar*, *low-rank*, or *per-dimension* channel resolution. When the state update is ungated and the decode weights come from an independent halt head, we recover PonderNet-style halting (Banino et al., 2021) as a special case of GMR. By unifying halt and gate into one framework, every learned GMR survival makes the recurrence trajectory itself input-adaptive, with the three gate resolutions progressing from coarser sequence-wide to finer feature-resolved control.

Our experiments on zero-shot and supervised probes show that recAMPLIFY with 155M unique parameters, after being pretrained on the same 2T tokens, remains competitive with the best performing AMPLIFY-350M with 56% fewer unique parameters. Namely, recAMPLIFY improves fold accuracy by 22.8% (from 56.7% to 69.6%) and localization accuracy by 2.3% (from 70.7% to 72.3%). It also outperforms the similar-sized AMPLIFY-120M across zero-shot probes by up to

[1]We define *depth recurrence* as the architectural mechanism of repeatedly applying a weight-tied block and *recursive computation* as the resulting strategy of increasing effective depth through repeated latent refinement.

11.7% and supervised probes by up to 26.5%. We further observe that more recursion is not always better: recAMPLIFY trained with fixed recurrence depth can degrade beyond the recursion horizon seen during training. While variable-depth pretraining improves robustness to unseen depths, it does not provide a single optimal test-time depth. In contrast, GMR enables adaptive recursion while reducing the expected number of recurrent cycles by 32–40% relative to a recAMPLIFY trained with six cycles, with task-dependent performance trade-offs. Our mechanistic analyses further show that residual gating, rather than halting alone, stabilizes recurrence beyond the trained horizon, while the expected halt depth reflects a sequence's composition entropy. Our main contributions are:

- We introduce a recursive pretraining framework for test-time scaling in masked pLMs by separating effective depth from unique parameter count.
- We propose recAMPLIFY, a recursive pLM with 56% fewer unique parameters than AMPLIFY-350M, and show its competitiveness on downstream tasks after training on the same 2T tokens.
- To enable adaptive recursion in recAMPLIFY, we propose *Gated Mixture Recurrence* (GMR), a unified framework that decodes from a mixture over recurrence-depth hidden states and gates their updates by a learned survival at scalar, low-rank, or per-dimension resolution.
- We characterize the recurrent dynamics and the biological signal learned by recAMPLIFY, showing residual gating to stabilize recurrence and adaptive depth to track biological properties.

2 Related Work

pLMs and iterative refinement. ESM-1b (Rives et al., 2021), ProtTrans (Elnaggar et al., 2022), and ESM-2 (Lin et al., 2023) established that unsupervised protein representations discover aspects of structure, function, and variant-effect prediction. AMPLIFY (Fournier et al., 2024) showed that data curation and architecture can matter more than parameter count. Iterative refinement is not new to protein modeling either. The folding trunks of AlphaFold (Jumper et al., 2021) and ESMFold (Lin et al., 2023) repeatedly refine sequence and pair representations. However, these recycles belong to a supervised structure-prediction trunk and use a globally chosen count. Instead, we pretrain with recurrence inside the sequence encoder, and study both fixed and input-dependent test-time depth.

Scaling beyond parameters. Popular methods for scaling language models with fixed parameter count include repeated sampling (Cobbe et al., 2021; Baek et al., 2026), chain-of-thought (CoT) reasoning (Wei et al., 2022), or both (Wang et al., 2023; Yoran et al., 2023). Nevertheless, these come with their own trade-offs. Repeated sampling needs additional verifiers for assessing candidates, which is not always possible for scientific discovery tasks (Cornelio et al., 2026). CoT reasoning calls for committing to discrete verbalized tokens. Addressing these pitfalls, a third scaling axis involves thinking recursively in continuous latent spaces (Amari, 1972; Hao et al., 2025). Universal (Dehghani et al., 2019) and Looped Transformers (Giannou et al., 2023) approach this through a weight-tied, iterative refinement lineage (Liao & Poggio, 2016). Given the effectiveness of recursive thinking for limited data regimes (Wang et al., 2025; Jolicoeur-Martineau, 2025; Liao & Poggio, 2026) and how it helps compact models with better learning priors (Schwarzschild et al., 2021; Bansal et al., 2022; Kohli et al., 2026), we hypothesize that weight-tied recurrence is crucial for scaling compact pLMs, where larger models are known to memorize training data (Fournier et al., 2024). A desirable property for such recurrence is adaptive computation (Graves, 2016) that refines the latent state in an input-dependent manner (Blayney et al., 2026; Fein-Ashley & Rashidinejad, 2026). Here, a halting head can be trained either in a supervised (Jolicoeur-Martineau, 2025) or self-supervised manner (Banino et al., 2021) to recognize convergence. For a pLM encoder, supervised halting requires a verifier to recognize which residues are valid at each masked position, which is exactly the biology the model is being trained to learn. We thus adopt the latter self-supervised objective.

Mechanistic analyses of recursion. Depth recurrence defines a learned dynamical system whose hidden states can converge or oscillate, and where additional iterations can degrade a previously useful representation. Prior work links successful depth extrapolation to input recall (Bansal et al., 2022), path-independent attractors (Anil et al., 2022; Geiping et al., 2025), and stable cyclic fixed points (Prairie et al., 2026), and identifies "overthinking" beyond the trained depth as a recurring failure mode (Kohli et al., 2026; Blayney et al., 2026). These studies treat recursion dynamics, such as depth sweeps, state changes, output convergence, and evolutionary properties, as operational rather than interpretive. We build on these works in our analysis of gated recurrence and protein properties.

3 PROBLEM SETUP

Test-time scaling through recursive pretraining. We formulate test-time scaling for masked pLMs using AMPLIFY (Fournier et al., 2024), whose released pretraining data, code and architecture recipe enable controlled comparisons under a fixed data budget. Let $S = (a_1, \ldots, a_T)$ denote a protein sequence of length T, where a_i denotes the residue at position i, drawn from the alphabet of 20 standard amino acids. AMPLIFY pretrains a pre-norm Transformer encoder f by minimizing the masked-token cross-entropy $\mathcal{L}_{\mathrm{MLM}}$ to recover masked residues from a bidirectional context. Its hidden states remain in one-to-one alignment with the input residues, supporting masked-marginal mutation scoring and frozen-embedding probes. We seek to build a recursive AMPLIFY encoder whose inference-time computation can be increased per protein without changing its unique parameter count or disrupting residue alignment. To this end, we expose the number L of applications of a weight-tied Transformer core as a test-time variable. This allows the model to iteratively refine its residue-aligned latent state under varying compute budgets.

Desired properties of a recursive pLM. Proteins vary widely in biological complexity, from short signal peptides to multi-domain enzymes and from structured domains to intrinsically disordered chains. pLMs must accommodate diverse proteins, and the per-assay optima suggest that no single recurrence depth is optimal across all proteins (Figure 2). These observations motivate learning a recurrence depth budget that adapts to each sequence. We thus seek a recursive pLM that can allocate compute adaptively while preserving stable latent dynamics. We then study whether its expected halt depth carries interpretable information about proteins.

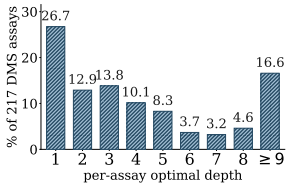


Figure 2: recAMPLIFY's per-assay optimal recurrence depth on ProteinGym DMS.

4 METHOD

We start by adapting the minimal recurrent design of Geiping et al. (2025) from a causal decoder to the bidirectional AMPLIFY encoder f, yielding recAMPLIFY. Here, the early p layers of the Transformer encoder form the *prelude* that runs once to produce the input representation $\mathbf{x} \in \mathbb{R}^{T \times D}$ from token embeddings. This is followed by an r-layer weight-tied *recurrent core* that repeatedly refines a single state $\mathbf{z} \in \mathbb{R}^{T \times D}$ over L cycles before passing it to a c-layer *coda* that decodes the final state $\widetilde{\mathbf{z}}$ from the latest cycle: $\widetilde{\mathbf{z}} = \mathbf{z}_L$. For the core to act as a stable data-dependent iterative operator, we re-inject the input $\mathbf{x}$ at every cycle (Bansal et al., 2022). To encourage a more general solution, we randomly initialize the latent vector as $\mathbf{z}_0 \sim \mathcal{N}(0, \sigma^2 I)$ (Anil et al., 2022):

$$\mathbf{x} = f_{\mathrm{p}}(\mathrm{Embed}(S)), \qquad \mathbf{z}_k = f_r(\mathbf{z}_{k-1}, \mathbf{x}), \qquad \hat{\mathbf{y}}_k = \mathrm{LMHead}(f_{\mathrm{c}}(\widetilde{\mathbf{z}})), \tag{1}$$

where $k \in \{1, \ldots, L\}$ is the recurrent cycle. The coda output is passed to the LM head for masked-token prediction and zero-shot probing. For supervised evaluation, we train a two-layer MLP on frozen recurrent-core representations, with coda representations considered in Figure 8. We use the triplet (p, r, c) for the unique layer counts and L for the number of cycles. This gives an effective depth of $p + (r \cdot L) + c$ while exposing L as a globally specified inference-time compute value. Motivated by the heterogeneous depth need in §3, we next make this budget input-dependent.

4.1 GATED MIXTURE RECURRENCE

Adaptive recursion requires allocating compute along two axes: *when* recurrent refinement should stop, and *how* each cycle's update should shape the hidden state along the way. We propose *Gated Mixture Recurrence* (GMR), a framework that treats both these axes under a single learned survival signal aimed to unify halting and gating. The recurrent core produces a trajectory of intermediate hidden states $\{\mathbf{z}_k\}_{k=1}^{L}$. GMR decodes each input from a probability-weighted mixture $\bar{\mathbf{z}}$ while gating each state update $\mathbf{z}_k = \mathbf{z}_{k-1} + S_k \odot (\widetilde{\mathbf{z}}_k - \mathbf{z}_{k-1})$ by a channel-resolved survival S_k.

When to stop updating? As a supervised adaptive computation head (Jolicoeur-Martineau, 2025) would require a per-position verifier (§2), we instead adopt a self-supervised approach (Banino et al., 2021) within GMR. Here, a halt head reads the state and emits a conditional halt probability $\lambda_k = \mathtt{sigmoid}(\mathtt{HaltHead}(\mathrm{sg}\,\widetilde{\mathbf{z}}_k))$, where the stopgrad operator sg prevents the head's gradients

from directly steering the LM features. This yields the unconditional halt mass p_k that acts as mixture weights, with the training loss $\mathcal{L}$ additionally maximizing the entropy H to encourage more cycles:

$$p_k \;=\; \lambda_k \textstyle\prod_{j<k}(1-\lambda_j), \qquad \bar{\mathbf{z}} = \textstyle\sum_k p_k\, \mathbf{z}_k, \qquad \mathcal{L} \;=\; \mathcal{L}_{\mathrm{MLM}}(\mathrm{LMHead}(f_c(\bar{\mathbf{z}}))) \;-\; H(\{p_k\}), \tag{2}$$

where $\mathbf{z}_k$ is the $k-$th cycle output. We decode predictions from the mass-weighted state $\bar{\mathbf{z}}$ and stop once the continuation mass $\prod_{j<k}(1-\lambda_j)$ falls below a threshold ε and discard this residual mass. Since the continuation mass is non-increasing, probability committed to halting at cycle j cannot be redistributed to later cycles.

How to update? In the halt-only case described above, a single halt probability weights the decode mixture but leaves state updates ungated, treating all hidden-state features as converging together. We relax this by allowing survival to vary across c channels of the feature space, so different features can stabilize at different cycles. To this end, a residual gate reads the proposal $\widetilde{\mathbf{z}}_k$ and emits channel-wise values $g_k^{\mathrm{ch}} = \mathtt{sigmoid}(\mathtt{HaltHead}(\mathrm{sg}\,\widetilde{\mathbf{z}}_k))$, where $g^{\mathrm{ch}} \approx 1$ preserves the channel's future update potential and $g^{\mathrm{ch}} \approx 0$ suppresses subsequent updates. Gate values accumulate across cycles into the survival S_k^{ch}, which is broadcast or projected to full hidden dimension D to yield the per-feature update weights S_k^D. Put together, the gated state update is:

$$\mathbf{z}_k \;=\; \mathbf{z}_{k-1} \;+\; S_k^D \odot \big(\widetilde{\mathbf{z}}_k - \mathbf{z}_{k-1}\big), \qquad S_k^D = \mathtt{project}(S_k^{\mathrm{ch}}), \qquad S_k^{\mathrm{ch}} = \prod_{j<k} g_j^{\mathrm{ch}}, \tag{3}$$

where S_k^{ch} is the update magnitude at cycle k set by the accumulated survival of previous gates, while the newly predicted gate g_k^{ch} contributes only to the current cycle's halt mass. Since the survival S_k^{ch} is monotonically non-increasing, this inherits the halt semantics of equation 2, where once a channel has committed to halting, its future updates are downweighted.

We instantiate the gate at three channel-level granularities: (a) the *scalar* gate broadcasts a single survival value (ch $= 1$) across all feature channels. This is similar to the halt scalar λ_k from equation 2, but acts on state-update magnitude rather than on decode; (b) the *low-rank* gate maintains ch$\ll D$ shared survival trajectories (e.g., ch$=16$, $D=960$) and projects them onto every hidden feature through a learned row-stochastic assignment. This ensures that all D features are steered by the same shared survivals; (c) the *per-dimension* gate is our most granular variant, using one independently predicted survival trajectory per channel (ch$=D$). Halting occurs either when we reach the predefined limit L or when every channel has a survival below the threshold ε (see Listing 1 for the pseudocode).

5 Experiments

Training and evaluation. Following AMPLIFY's pretraining recipe and using its publicly released code and data (Fournier et al., 2024), we pretrain recAMPLIFY on the same mixture of the UniRef100 (Suzek et al., 2007), SCOP (Murzin et al., 1995), and OAS (Olsen et al., 2022) databases, making the original AMPLIFY our direct baseline. Further details are provided in §B.

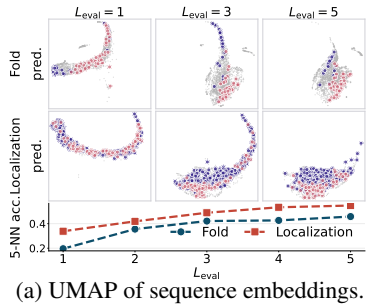
Following Lin et al. (2023), we evaluate using two types of probes. *Zero-shot probes:* KL calibration (Le Breton et al., 2026), ProteinGym DMS (Notin et al., 2023), and CASP16 contact recovery (Yuan et al., 2026). *Supervised probes:* we use the 18-task benchmark of Chen et al. (2025) and study four representative tasks reported to show clear benefits from pretraining (Le Breton et al., 2026): Fold accuracy (Rao et al., 2019), Localization accuracy (Dallago et al., 2021), Stability prediction (Rao et al., 2019; Dallago et al., 2021), and Secondary-structure prediction (SSP) (Rao et al., 2019). For each supervised task, we report test performance across 5 seeds after re-evaluating the best configuration from a 50-trial Bayesian TPE hyperparameter search. Further evaluation and hyperparameter details are provided in §C.

5.1 Scaling and Depth Behavior

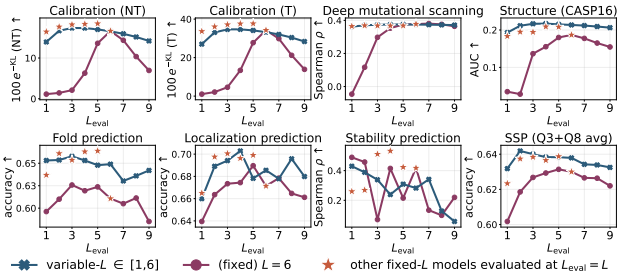
Does recursive pretraining scale? We first examine whether the minimal recurrent design of recAMPLIFY remains viable at the full pLM pretraining scale of 1M steps. For a fair comparison, we match AMPLIFY-350M's 15 attention heads and a hidden dimension of 960, and set the number of layers to $(p, r, c) = (2, 10, 2)$, i.e., roughly the budget of the compact AMPLIFY-120M baseline.

Table 1: **Zero-shot and supervised performance.** The top block reports ESM-2 for context, as its pretraining data favor structural modeling while AMPLIFY's emphasizes evolutionary calibration (KL). The middle block reports full-scale comparisons, where recurrence remains competitive on zero-shot tasks and helps improve three out of four supervised probes. The bottom block reports ablations with variable-depth training and four GMR variants. Effective depth $\mathcal{D}_{\text{eff}} = p+Lr+c$ counts layer applications, using L_{eval} at inference and expected cycles for GMR. Best scores within each block are **bold**. See Table 4 for standard deviations.

	Zero-shot probes			Supervised probes				Param / compute		
Model	KL ↑	DMS ρ ↑	CASP P@T ↑	Fold acc. ↑	Loc. acc. ↑	Stability ρ ↑	SSP (Q3+Q8) ↑	N	$\mathcal{D}_{\text{eff}}^{\text{train}}$	$\mathcal{D}_{\text{eff}}^{\text{eval}}$
ESM-2 baselines for context (500K pretraining steps on UniRef50/90)										
ESM-2-150M	6.69	0.401	0.257	0.648	0.745	0.525	0.771	149M	30	30
ESM-2-650M	16.61	**0.439**	**0.269**	0.679	0.768	0.515	0.799	652M	33	33
ESM-2-3B	**23.66**	0.433	0.267	**0.723**	**0.779**	**0.609**	**0.802**	2.8B	36	36
1M pretraining steps on UniRef100										
AMPLIFY-120M	21.23	0.368	0.154	0.563	0.642	0.396	0.639	118M	24	24
AMPLIFY-350M	**24.30**	**0.386**	0.163	0.567	0.707	0.461	**0.651**	354M	32	32
fixed-L recAMPLIFY	22.31	0.378	**0.172**	**0.696**	**0.723**	**0.501**	**0.651**	155M	64	64
100K pretraining steps on UniRef100										
AMPLIFY-120M	16.94	0.366	0.145	0.602	0.675	0.562	0.621	118M	24	24
AMPLIFY-350M	**19.57**	**0.382**	0.126	0.574	0.664	0.440	0.641	354M	32	32
vary-L recAMPLIFY	16.55	0.375	0.157	**0.649**	0.685	0.282	0.638	133M	52	52
fixed-L recAMPLIFY	16.63	0.378	0.143	0.611	0.671	0.418	0.630	133M	52	52
+ halt-only GMR	17.96	0.370	0.154	0.632	**0.702**	**0.572**	**0.644**	133M	52	33
+ scalar gated GMR	18.40	0.371	**0.172**	0.630	0.699	0.346	0.636	133M	52	37
+ rank-16 gated GMR	18.69	0.372	0.158	0.623	0.568	0.419	0.636	133M	52	36
+ per-dim gated GMR	18.92	0.367	0.158	0.622	0.700	0.469	0.637	133M	52	37



(a) UMAP of sequence embeddings.



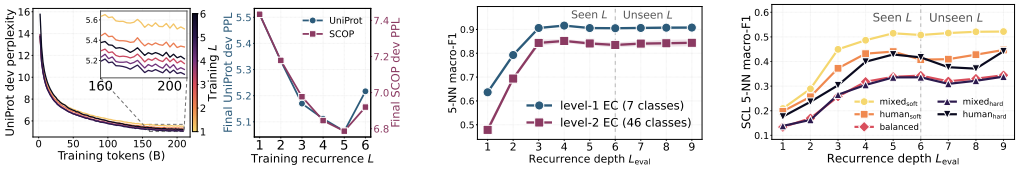
(b) Test-time L_{eval} sweep across 100K pretraining setups.

Figure 3: **recAMPLIFY at varying recurrence depths.** (a) For 1M-step recAMPLIFY, the pre-classifier embeddings become progressively more class-separable (5-NN accuracy) with larger test-time depth L_{eval}. (b) For 100K-step recAMPLIFY, as L_{eval} increases past the trained depth (L=6), the fixed-L model (red) degrades while the variable-L variant (blue) remains robust.

Table 1 shows that recAMPLIFY, with 155M unique parameters, remains competitive with AMPLIFY-350M across downstream probes when both are trained on the same 2T tokens. The clearest gains appear on the four supervised probe tasks that most reliably measure information acquired during pretraining. Visualizing the pre-classifier embeddings across recurrent cycles (Figure 3a) shows that each additional cycle brings the embedding manifold closer to the downstream label geometry, improving 5-NN accuracy. Finally, we verify that the cost of our recursive pretraining recipe scales with depth alone. Table 5 shows that our 133M recAMPLIFY at $L = 3$ achieves measured effective throughput comparable to AMPLIFY-350M, which has approximately $3\times$ as many unique parameters.

Comparison with ESM-2. A direct comparison with ESM-2 is confounded by differences in model size and pretraining data: its larger checkpoints have 4–18$\times$ as many parameters as recAMPLIFY and use an earlier UniRef release with a different clustering scheme. ESM-2 samples uniformly over UniRef50 clusters, whereas AMPLIFY's UniRef100 recipe retains more of the observed sequence redundancy, resulting in different training distributions (§B). ESM-2's unreleased pretraining code and exact data splits also make reproducing its training setup difficult. With these caveats, larger ESM-2 checkpoints outperform recAMPLIFY on supervised probes, DMS, and CASP, while recAMPLIFY achieves better KL calibration than smaller ESM-2 checkpoints.

A 100K-step regime for controlled ablations. Training recAMPLIFY for 1M steps makes extensive ablations computationally impractical. Following Le Breton et al. (2026), we next ask whether



(a) Held out set perplexity (b) Enzyme classification (c) Subcellular localization

Figure 4: **Effect of recurrence depth on sequence likelihood and downstream separability at 100K steps.** (a) UniProt perplexity for fixed-L pretraining, evaluated at $L_{\text{eval}}{=}L$. (b) EC level-1 and level-2 5-NN macro-F1 across L_{eval} for recAMPLIFY trained with $L{=}6$, using 14K Swiss-Prot enzymes balanced across the 7 top-level classes re-labelled into 46 level-2 subclasses ($\geq$30 samples each). (c) SCL 10-way 5-NN macro-F1 across five FLIP DeepLoc-1.0 splits (Didi et al., 2026), fitted on 11,181 shared training sequences and evaluated on each held-out test set.

recursive pretraining at a smaller scale remains representative. Table 1 shows that a similar recAMPLIFY configuration, $(p, r, c) = (2, 8, 2)$, achieves a representative task-performance profile at 100K steps. We therefore use 100K steps for subsequent experiments.

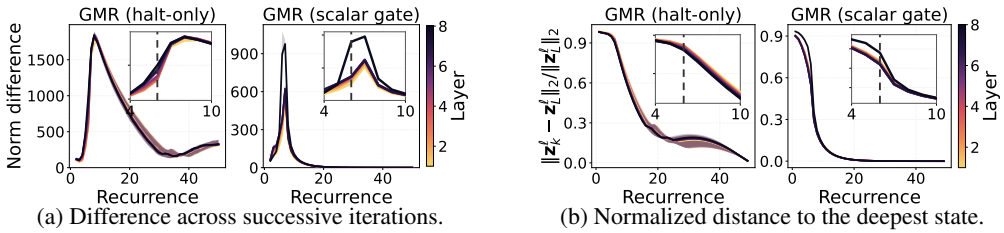
How does recursion behave with depth? Figure 4 shows that greater depth improves both validation perplexity and downstream separability for enzyme classes (EC) and subcellular localization (SCL). For SCL, deeper recursion yields macro-F1 gains of 20–31 pp across FLIP's train/test cluster splits. Given the diverse criteria used to construct these splits, this suggests benefits from recursion under distribution shift. Figure 3b shows downstream probe performance as recAMPLIFY is trained with larger cycles $L \in \{1, \ldots, 6\}$ (marked by stars). For the model trained with $L = 6$, further varying L_{eval} shows that $L_{\text{eval}} = L$ performs best across most tasks. Finally, at a fixed depth, we find the physical layer counts $r = 10$ (Figure 9a) and $(p, c) = (2, 2)$ (Figure 9b) to perform best overall.

Going beyond fixed-depth recursion. The fixed-depth recAMPLIFY baseline uses L cycles during training and exposes recurrence depth as a test-time variable L_{eval}. Figure 3b shows that this model (red lines) generalizes poorly to unseen depths ($L_{\text{eval}} > 6$) across supervised tasks. In contrast, pretraining with a variable number of cycles sampled uniformly from $\{1, \ldots, 6\}$ per minibatch generally improves robustness to unseen depths (blue lines). Several probes peak near the variable-depth model's mean training depth, suggesting a simple heuristic: vary the cycle count during training, then fix L_{eval} near that mean. However, the per-assay optima in Figure 2 suggest that no single depth is optimal across all assays. Put together, these observations support our motivation for an adaptive recursion mechanism that predicts when further refinement is no longer useful.

Gated Mixture Recurrence. Table 1 shows that every GMR variant outperforms both fixed-L and varying-L recAMPLIFY on KL calibration, the metric most representative of AMPLIFY's pretraining recipe (§B), while reducing the *expected* test-time depth from 52 to 33–37 layer applications. We quantify depth as the mean of the model's predicted halting distribution: $\mathbb{E}[k] = \sum_k k\, p_k$. This characterizes the learned allocation policy rather than measured wall-clock savings, since our evaluation decodes the full mixture $\sum_k p_k \mathbf{z}_k$ over the cycle budget. Crucially, KL improves monotonically with gate resolution, placing per-dim GMR within $\sim$3% of AMPLIFY-350M's KL at only $\sim$37% of its parameter count and a comparable $\mathbb{E}[k]$. This monotonic gate-resolution effect on our primary metric empirically confirms that channel-resolved survival carries non-trivial capacity. Even the halt-only case, our weakest gate, outperforms both recAMPLIFY baselines on KL and is the strongest supervised performer in the 100K-step block for localization, stability, and SSP, at $\mathbb{E}[k] = 3.57$, corresponding to an expected test-time depth of $2 + 3.57{\times}8 + 2 \approx 33$ layers. To understand where gating helps, we next examine whether the gates produce an empirical fixed point of the recurrent state (§5.1.1); whether recursion preserves the depth-heterogeneous layerwise mixing profile of a feedforward Transformer (§5.1.2); and what the GMR halt signal encodes about proteins (§5.1.3).

5.1.1 Hidden State Evolution

Let $\mathbf{z}_k^\ell$ denote the hidden state at physical layer $\ell \in \{1, \ldots, r\}$ of the recurrent block during cycle $k \in \{1, \ldots, L_{\text{eval}}\}$. The block-level state $\mathbf{z}_k$ used in §4.1 is its top-layer output, i.e., $\mathbf{z}_k \equiv \mathbf{z}_k^r$. We analyze how recursion affects hidden-state dynamics within and beyond the depth seen during training. Following Blayney et al. (2026), we unroll each recAMPLIFY variant to $L_{\text{eval}} = 50$ and examine



(a) Difference across successive iterations. (b) Normalized distance to the deepest state.

Figure 5: **Dynamics of recurrent block layer** ℓ on stability prediction upon training with $L = 6$ cycles (vertical dashed lines). (a) The successive difference norm $\|\mathbf{z}_k^\ell - \mathbf{z}_{k-1}^\ell\|_2$ shows gating drives each layer's step size toward zero beyond the trained depth. (b) The normalized difference to each layer's deepest recorded state highlights the rate and timing of decay as key convergence diagnostics.

state changes across recurrent layers along three complementary axes: (a) the size of successive updates $\|\Delta\mathbf{z}\|_2$, (b) the normalized distance of each cycle's state to its deepest recorded state $\mathbf{z}_{L_{\text{eval}}}^\ell$, taken as an approximate fixed point, and (c) the induced attention geometry, measured by pairwise Frobenius distances between attention maps across cycles.

Gates deliver the state-level stopping primitive that halting alone cannot. The per-layer step norms in Figures 5a and 13a reveal three distinct behaviors. Recurrence without GMR shows no consistent limiting behavior, although input injection keeps it bounded (Figure 12). Halt-only GMR changes only *when* updates stop, not the underlying step size. GMR with residual gates provides what halt-only lacks: a state-level "stop-moving" mechanism that drives the step norm toward zero for the scalar and per-dim variants (§E).

Scalar and per-dimension residual gates deliver empirical fixed-point convergence. We use "state convergence" for variants whose update magnitude $\|\Delta\mathbf{z}\|_2$ approaches zero (Figures 5a and 13a) *and* whose normalized residual $\|\mathbf{z}_k^\ell - \mathbf{z}_L^\ell\|_2 / \|\mathbf{z}_L^\ell\|_2$ decays rapidly (Figures 5b and 13b). From this perspective, the four variants separate cleanly: scalar and per-dim gates reach near-zero residual within a few cycles of the trained horizon $L = 6$, and their step sizes vanish. Halt-only decays only linearly with an unbounded norm while the rank-16 gate drifts radially. Cosine alignment to $\mathbf{z}_L^\ell$ is near-perfect for every variant and is therefore only an indirect diagnostic (Figure 13c). Lastly, the induced attention geometry shows scalar and per-dim gates collapse cross-cycle attention distances past the trained horizon, whereas halt-only and rank-16 do not (Figure 14). Empirical fixed-point convergence therefore requires a residual gate rather than the halting head alone.

5.1.2 Stages of Inference

Feedforward Transformer layers exhibit heterogeneous compute profiles, where early layers often cluster residues by local context, middle layers integrate over increasing spans, and late layers commit to global structural predictions (Vig et al., 2021; Siji et al., 2026). We thus ask whether, for each recAMPLIFY variant, the tied block preserves this depth-heterogeneous mixing schedule within the trained recurrence depth $L = 6$ and under extrapolation. Following de Llano et al. (2026), we measure attention geometry with ColSum concentration, $C \in [0, 1]$ (defined in §E). A non-flat layerwise profile of C indicates mixing heterogeneity across physical layers.

Within the trained depth, recursion preserves mixing heterogeneity. Within the trained horizon $k \leq 6$ (solid curves in Figure 15), all recAMPLIFY variants retain a structured, non-flat ColSum profile across the eight tied core layers, with the earliest cycle closest to feedforward reference.

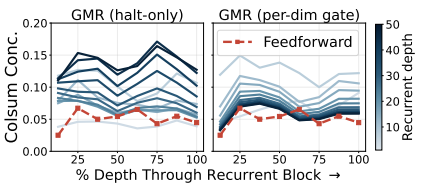


Figure 6: **ColSum concentration** through the recurrent block at L_{eval}=50. Cycles colored light (early) to dark (late). Dashed red line is the feedforward reference.

Residual gates stabilize mixing beyond the trained horizon. Figure 15 shows that a stable per-layer state under deeper unrolling is marked by dashed extrapolated per-cycle profiles piled onto the solid trained range. Only the three GMR gates achieve this, all within a few cycles of the seen L. Stability alone, however, does not tell us whether the frozen profile reproduces the depth-heterogeneous mixing of

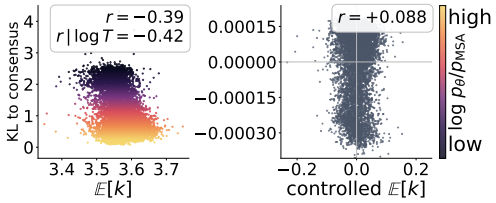


Figure 7: **Per-protein expected depth $\mathbb{E}[\mathbf{k}]$ vs mean per-position KL to MSA consensus.** Colors code the sharpness proxy $\log(p_\theta/p_{\mathrm{MSA}})$ at consensus AA. *Left:* original depth scatter, *right:* depth scatter when controlled for sequence length T and sharpness.

Table 2: **Sequence composition Cohen's d by controlled depth $\mathbb{E}[k]$** on held-out UniProt.

Feature	Cohen's d
Per-protein composition (+ means over-pondered)	
sequence Shannon entropy	+1.47
hydrophobic (FILVMWY) fraction	+1.45
aromatic (FWY) fraction	+0.98
polar-neutral (STNQ) fraction	−0.51
charged (DEKRH) fraction	−0.28
histidine fraction	+0.11
TM-like 19-win / 100 aa (KD≥1.6)	+0.35
sequence length	−0.57
Same-gene splice-isoform similarity	
Isoforms of the same human gene (335 genes)	+0.79

a standard feedforward Transformer or has flattened into a distorted schedule. We next compare the two most contrastive variants at extrapolated depth: halt-only GMR, which never stabilizes, and per-dim GMR, which settles. Figure 6 shows that only per-dim gating approaches the feedforward reference in both shape and amplitude. Lastly, the scalar and per-dimension gates stabilize attention geometry beyond the trained horizon, whereas halt-only and rank-16 do not (Figure 14).

5.1.3 Does the GMR halt depth encode biological signal?

For per-protein halt depth $\mathbb{E}[k]$ to be a reliable diagnostic signal, the model's distribution around an amino acid should align *sharply* with what evolution prefers at the position. However, once $\mathbb{E}[k]$ is controlled for sequence length and sharpness, its correlation with KL collapses by $\approx 78\%$ (Figure 7). $\mathbb{E}[k]$ thus tracks sharpness at consensus amino acid, which is a projection of sequence composition (more tasks in Table 7). To study other such composition properties, we bucket the same UniProt held out sequences by the controlled $\mathbb{E}[k]$ into top- and bottom-decile buckets intersected across all four GMR variants and report Cohen's d between the buckets. Table 2 identifies *compositional diversity* as the most robust signal, with over-pondered proteins reflecting higher sequence entropy and hydrophobic content. Unusually hydrophobic composition is only viable inside a specific fold or membrane pocket, so recovering the wild-type residue there requires long-range structural evidence and thus more cycles. In contrast, extreme polar or charged composition is an alphabet-reduction cue, where once a residue class dominates, most masked positions become predictable from local statistics and fewer cycles suffice. §F shows hydrophobic content driving an overall depth increase while other class overrepresentations produce an inverted *U-shape* consistent with alphabet reduction.

Next, we extend this test from per-protein composition to per-gene identity, and study how GMR depth varies across alternative splice-isoforms of the same human gene. Across 335 human genes and 1,054 isoforms, same-gene isoforms cluster nearly twice as tightly in halt depth as size-matched random pairs of proteins (Table 2). This gap indicates that gates do not follow a fixed, input-independent decay schedule. Instead, the input governs the gate trajectory, at least at gene granularity. Moreover, what gates read is a property of the shared backbone rather than of exact residues since alternative isoforms differ only by one included or excluded functional domain, and $\mathbb{E}[k]$ consistently tracks the compositional constraint of the retained portion across all family members (see §F for details).

6 Conclusion

We introduced recursive pretraining for test-time compute scaling in masked pLMs. With 56% fewer unique parameters, the resulting recAMPLIFY is competitive with AMPLIFY-350M on zero-shot probes and improves fold, localization, and mean stability scores. Our Gated Mixture Recurrence (GMR) unifies self-supervised halting and residual gating into one adaptive computation framework, reducing the expected per-protein cycle count by 32–40% while its KL calibration improves monotonically with gate resolution. Mechanistically, scalar and per-dimension gates drive the recurrent state toward an empirical fixed point beyond the trained horizon, and only per-dimension gating recovers the feedforward layerwise mixing profile. Finally, halt depth tracks sequence entropy and hydrophobic composition, making it an interpretable compute allocator.

AI USE STATEMENT

In this work, we used generative AI tools for scheduling experiments, gathering and plotting results from logs, analyzing the grammatical errors and logical flaws in writing, crawling and gathering the splice isoform dataset from UniProt REST API, cross-checking the equations and their notational consistency against other contemporary works on depth recurrence, and interpreting results particularly with biological property analyses like hydrophobic composition. We have not used generative AI tools for coming up with the core idea of the work, and researching related works on protein language models and recurrence depth. Additionally, we used generative AI tools for creating or editing the pretraining and evaluation code, formulating paragraphs and headings for showcasing our findings under coherent themes. We have reviewed all AI-assisted work. The LLM-generated code was verified and tested for correctness on small-scale training setup before scaling up to pretraining (100K and 1M steps). We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We build upon the codebase of the publicly released AMPLIFY model, and we will release our code. Also, we have provided details on hyperparameter tuning, dataset splits and the evaluation setup.

REFERENCES

José Juan Almagro Armenteros, Casper Kaae Sønderby, Søren Kaae Sønderby, Henrik Nielsen, and Ole Winther. Deeploc: prediction of protein subcellular localization using deep learning. *Bioinformatics*, 33(21):3387–3395, 11 2017. ISSN 1367-4803. doi: 10.1093/bioinformatics/btx431.

S.-I. Amari. Learning patterns and pattern sequences by self-organizing nets of threshold elements. *IEEE Transactions on Computers*, C-21(11):1197–1206, 1972. doi: 10.1109/T-C.1972.223477.

Cem Anil, Ashwini Pokle, Kaiqu Liang, Johannes Treutlein, Yuhuai Wu, Shaojie Bai, J. Zico Kolter, and Roger B. Grosse. Path independent equilibrium models can better exploit test-time computation. In *Advances in Neural Information Processing Systems*, volume 35, pp. 7796–7809, 2022.

Junyeob Baek, Mingyu Jo, Minsu Kim, Mengye Ren, Yoshua Bengio, and Sungjin Ahn. Generative recursive reasoning. *arXiv preprint arXiv:2605.19376*, 2026.

Andrea Banino, Jan Balaguer, and Charles Blundell. PonderNet: Learning to ponder. In *ICML 2021 Workshop on Automated Machine Learning*, 2021.

Arpit Bansal, Avi Schwarzschild, Eitan Borgnia, Zeyad Emam, Furong Huang, Micah Goldblum, and Tom Goldstein. End-to-end algorithm synthesis with recurrent networks: Extrapolation without overthinking. *Advances in Neural Information Processing Systems*, 35:20232–20242, 2022.

Hugh Blayney, Álvaro Arroyo, Johan Obando-Ceron, Pablo Samuel Castro, Aaron Courville, Michael M. Bronstein, and Xiaowen Dong. A mechanistic analysis of looped reasoning language models. *arXiv preprint arXiv:2604.11791*, 2026.

Salvatore Candido, Thomas Hayes, Alexander Derry, Roshan Rao, Zeming Lin, Robert Verkuil, Bryan Z. Wu, Jin Sub Lee, Elise S. Bruguera, Jehan A. Keval, Mykhailo Kopylov, John E. Pak, Wesley Wu, Neil Thomas, Samson Mataraso, Alvin Hsu, Ashton C. Trotman-Grant, Kilian Fatras, Allan dos Santos Costa, Rohil Badkundri, Halil Akın, Deniz Oktay, Jonathan Deaton, Elizabeth Montabana, Hrishita Sitwala, Yue Yu, Marius Wiggert, Dylan Alexander Carlin, Anthony W. Goering, Tomasz Blazejewski, McCullen Sandora, Michael Hla, Tina Z. Jia, Leon H. Kloker, Nicholas J. Sofroniew, Masatoshi Uehara, Jassi Pannu, Sharrol Bachas, Daniel S. Liu, Tom Sercu, and Alexander Rives. Language modeling materializes a world model of protein biology. *bioRxiv*, 2026. doi: 10.64898/2026.06.03.729735.

Bo Chen, Xingyi Cheng, Pan Li, Yangli-ao Geng, Jing Gong, Shen Li, Zhilei Bei, Xu Tan, Boyan Wang, Xin Zeng, Chiming Liu, Aohan Zeng, Yuxiao Dong, Jie Tang, and Le Song. xtrimopglm: unified 100-billion-parameter pretrained transformer for deciphering the language of proteins. *Nature Methods*, 22(5):1028–1039, 2025. doi: 10.1038/s41592-025-02636-z.

Xingyi Cheng, Bo Chen, Pan Li, Jing Gong, Jie Tang, and Le Song. Training compute-optimal protein language models. *Advances in Neural Information Processing Systems*, 37:69386–69418, 2024.

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

Cristina Cornelio, Takuya Ito, Ryan Cory-Wright, Sanjeeb Dash, and Lior Horesh. The need for verification in artificial intelligence-driven scientific discovery. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 384(2317):20240591, 04 2026. ISSN 1364-503X. doi: 10.1098/rsta.2024.0591.

Christian Dallago, Jody Mou, Kadina E Johnston, Bruce Wittmann, Nick Bhattacharya, Samuel Goldman, Ali Madani, and Kevin K Yang. FLIP: Benchmark tasks in fitness landscape inference for proteins. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

Enrique Queipo de Llano, Alvaro Arroyo, Federico Barbero, Xiaowen Dong, Michael M. Bronstein, Yann LeCun, and Ravid Shwartz-Ziv. Attention sinks and compression valleys in LLMs are two sides of the same coin. In *The Fourteenth International Conference on Learning Representations*, 2026.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Lukasz Kaiser. Universal transformers. In *International Conference on Learning Representations*, 2019.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pp. 4171–4186, 2019.

Kieran Didi, Sarah Alamdari, Alex Xijie Lu, Bruce James Wittmann, Kadina E Johnston, Ava P Amini, Ali Madani, Maya Czeneszew, Christian Dallago, and Kevin K Yang. FLIP2: Expanding protein fitness landscape benchmarks for real-world machine learning applications. In *Forty-third International Conference on Machine Learning*, 2026.

Ahmed Elnaggar, Michael Heinzinger, Christian Dallago, Ghalia Rehawi, Yu Wang, Llion Jones, Tom Gibbs, Tamas Feher, Christoph Angerer, Martin Steinegger, Debsindhu Bhowmik, and Burkhard Rost. ProtTrans: Toward understanding the language of life through self-supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(10):7112–7127, 2022. doi: 10.1109/TPAMI.2021.3095381.

Jacob Fein-Ashley and Paria Rashidinejad. Solve the loop: Attractor models for language and reasoning. *arXiv preprint arXiv:2605.12466*, 2026.

Noelia Ferruz, Steffen Schmidt, and Birte Höcker. ProtGPT2 is a deep unsupervised language model for protein design. *Nature Communications*, 13(1):4348, 2022. doi: 10.1038/s41467-022-32007-7.

Quentin Fournier, Robert M. Vernon, Almer van der Sloot, Benjamin Schulz, Sarath Chandar, and Christopher James Langmead. Protein language models: Is scaling necessary? *bioRxiv*, 2024. doi: 10.1101/2024.09.23.614603.

Naomi K. Fox, Steven E. Brenner, and John-Marc Chandonia. Scope: Structural classification of proteins—extended, integrating scop and astral data and classification of new structures. *Nucleic Acids Research*, 42(D1):D304–D309, 01 2014. ISSN 0305-1048. doi: 10.1093/nar/gkt1240.

Jonas Geiping, Sean McLeish, Neel Jain, John Kirchenbauer, Siddharth Singh, Brian Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, and Tom Goldstein. Scaling up test-time compute with latent reasoning: A recurrent depth approach. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 41340–41391. Curran Associates, Inc., 2025. doi: 10.52202/085713-1380.

Angeliki Giannou, Shashank Rajput, Jy-Yong Sohn, Kangwook Lee, Jason D. Lee, and Dimitris Papailiopoulos. Looped transformers as programmable computers. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, pp. 11398–11442. PMLR, 2023.

Alex Graves. Adaptive computation time for recurrent neural networks. *arXiv preprint arXiv:1603.08983*, 2016.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason E Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. In *Second Conference on Language Modeling*, 2025.

Chao Hou, Di Liu, Aziz Zafar, and Yufeng Shen. Understanding language model scaling for protein fitness prediction. *Nature Computational Science*, 6(7):778–788, 2026. doi: 10.1038/s43588-026-01010-z.

Alexia Jolicoeur-Martineau. Less is more: Recursive reasoning with tiny networks. *arXiv preprint arXiv:2510.04871*, 2025.

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *nature*, 596(7873):583–589, 2021.

Michael Schantz Klausen, Martin Closter Jespersen, Henrik Nielsen, Kamilla Kjærgaard Jensen, Vanessa Isabell Jurtz, Casper Kaae Sønderby, Morten Otto Alexander Sommer, Ole Winther, Morten Nielsen, Bent Petersen, and Paolo Marcatili. Netsurfp-2.0: Improved prediction of protein structural features by integrated deep learning. *Proteins: Structure, Function, and Bioinformatics*, 87(6):520–527, 2019. doi: https://doi.org/10.1002/prot.25674.

Harsh Kohli, Srinivasan Parthasarathy, Huan Sun, and Yuekun Yao. Loop, think, & generalize: Implicit reasoning in recurrent-depth transformers. In *Third Conference on Language Modeling*, 2026.

Lola Le Breton, David Heurtel-Depeiges, Douglas C Millar, Lara E Zetzsche, Robert M Vernon, Christopher James Langmead, Sarath Chandar, and Quentin Fournier. plm representations unlock metagenomic space beyond homology. *bioRxiv*, 2026. doi: 10.64898/2026.07.28.739874.

Qianli Liao and Tomaso Poggio. Bridging the gaps between residual learning, recurrent neural networks and visual cortex. *arXiv preprint arXiv:1604.03640*, 2016.

Qianli Liao and Tomaso Poggio. Simple recursive model: Simplified, single-state reasoning with skip connections, 01 2026.

Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, Allan dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Salvatore Candido, and Alexander Rives. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130, 2023. doi: 10.1126/science.ade2574.

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

Joshua Meier, Roshan Rao, Robert Verkuil, Jason Liu, Tom Sercu, and Alexander Rives. Language models enable zero-shot prediction of the effects of mutations on protein function. In *Advances in Neural Information Processing Systems*, volume 34, pp. 29287–29303, 2021.

Alexey G Murzin, Steven E Brenner, Tim Hubbard, and Cyrus Chothia. Scop: a structural classification of proteins database for the investigation of sequences and structures. *Journal of molecular biology*, 247(4):536–540, 1995.

Kaoru Nakano. Associatron-a model of associative memory. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-2(3):380–388, 1972. doi: 10.1109/TSMC.1972.4309133.

Erik Nijkamp, Jeffrey A. Ruffolo, Eli N. Weinstein, Nikhil Naik, and Ali Madani. ProGen2: Exploring the boundaries of protein language models. *Cell Systems*, 14(11):968–978.e3, 2023. doi: 10.1016/j.cels.2023.10.002.

Pascal Notin, Aaron Kollasch, Daniel Ritter, Lood van Niekerk, Steffanie Paul, Han Spinner, Nathan Rollins, Ada Shaw, Rose Orenbuch, Ruben Weitzman, Jonathan Frazer, Mafalda Dias, Dinko Franceschi, Yarin Gal, and Debora Marks. Proteingym: Large-scale benchmarks for protein fitness prediction and design. In A. Oh, T. Neumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 64331–64379. Curran Associates, Inc., 2023.

Tobias H Olsen, Fergus Boyles, and Charlotte M Deane. Observed antibody space: A diverse database of cleaned, annotated, and translated unpaired and paired antibody sequences. *Protein Science*, 31 (1):141–146, 2022.

Hayden Prairie, Zachary Novack, Taylor Berg-Kirkpatrick, and Daniel Y Fu. Parcae: Scaling laws for stable looped language models. *arXiv preprint arXiv:2604.12946*, 2026.

Roshan Rao, Nicholas Bhattacharya, Neil Thomas, Yan Duan, Peter Chen, John Canny, Pieter Abbeel, and Yun Song. Evaluating protein transfer learning with tape. *Advances in neural information processing systems*, 32, 2019.

Alexander Rives, Joshua Meier, Tom Sercu, Siddharth Goyal, Zeming Lin, Jason Liu, Demi Guo, Myle Ott, C. Lawrence Zitnick, Jerry Ma, and Rob Fergus. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proceedings of the National Academy of Sciences*, 118(15):e2016239118, 2021. doi: 10.1073/pnas.2016239118.

Gabriel J. Rocklin, Tamuka M. Chidyausiku, Inna Goreshnik, Alex Ford, Scott Houliston, Alexander Lemak, Lauren Carter, Rashmi Ravichandran, Vikram K. Mulligan, Aaron Chevalier, Cheryl H. Arrowsmith, and David Baker. Global analysis of protein folding using massively parallel design, synthesis, and testing. *Science*, 357(6347):168–175, 2017. doi: 10.1126/science.aan0693.

Avi Schwarzschild, Eitan Borgnia, Arjun Gupta, Furong Huang, Uzi Vishkin, Micah Goldblum, and Tom Goldstein. Can you learn an algorithm? generalizing from easy to hard problems with recurrent networks. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 6695–6706. Curran Associates, Inc., 2021.

Aleena Siji, Amir Mohammad Karimi Mamaghan, Ferdinand Kapl, Tobias Höppe, Emmanouil Angelis, Andrea Dittadi, Maurice Brenner, Michael Heinzinger, Karl Henrik Johansson, Kaitlin Maile, Johannes von Oswald, and Stefan Bauer. From words to amino acids: Does the curse of depth persist? In *Learning Meaningful Representations of Life (LMRL) Workshop at ICLR 2026*, 2026.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Baris E. Suzek, Hongzhan Huang, Peter McGarvey, Raja Mazumder, and Cathy H. Wu. Uniref: comprehensive and non-redundant uniprot reference clusters. *Bioinformatics*, 23(10):1282–1288, 05 2007. ISSN 1367-4803. doi: 10.1093/bioinformatics/btm098.

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

Jesse Vig, Ali Madani, Lav R. Varshney, Caiming Xiong, Richard Socher, and Nazneen Rajani. BERTology meets biology: Interpreting attention in protein language models. In *International Conference on Learning Representations*, 2021.

Guan Wang, Jin Li, Yuhao Sun, Xing Chen, Changling Liu, Yue Wu, Meng Lu, Sen Song, and Yasin Abbasi-Yadkori. Hierarchical reasoning model. *arXiv preprint arXiv:2506.21734*, 2025.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*, 2023.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 24824–24837, 2022.

Ori Yoran, Tomer Wolfson, Ben Bogin, Uri Katz, Daniel Deutch, and Jonathan Berant. Answering questions by meta-reasoning over multiple chains of thought. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 5942–5966, 2023.

Rongqing Yuan, Jing Zhang, Andriy Kryshtafovych, R Dustin Schaeffer, Jian Zhou, Qian Cong, and Nick V Grishin. Casp16 protein monomer structure prediction assessment. *Proteins: Structure, Function, and Bioinformatics*, 94(1):86–105, 2026.

Zhidian Zhang, Hannah K. Wayment-Steele, Garyk Brixi, Haobo Wang, Dorothee Kern, and Sergey Ovchinnikov. Protein language models learn evolutionary statistics of interacting sequence motifs. *Proceedings of the National Academy of Sciences*, 121(45):e2406285121, 2024. doi: 10.1073/pnas.2406285121.

A FORWARD-PASS PSEUDOCODE FOR RECAMPLIFY WITH GMR

Listing 1 shows a single training-time forward pass of recAMPLIFY for all four GMR variants (halt-only and the three residual gate resolutions). Notation follows §4.1: `sg` is the stopgrad operator, `halt_head` and `gate_head` are the linear heads producing λ_k and g^{ch}, `Norm` is RMSNorm, and Map_A is the channel-to-feature broadcast for c=1 scalar gate, row-stochastic low-rank projection, or identity for per-dim gate. The final `Loss` is the MLM cross-entropy on the mixture-decoded state $\bar{\mathbf{z}}$ plus the halt-distribution entropy regularizer H from equation 2. The backward pass is standard autograd with the stopgrad boundaries as marked.

```
x, z = Prelude(Embed(u)), z_init
inj  = x + f(z_init)                          # f = optional SRM-styled context encoder

if mode == "halt_only_GMR":                              # scalar, per-sequence halting
    p_left, z_bar, p = 1, 0, []
    for k in 1..L:
        z = f(z + inj)
        lam_k = sigmoid(halt_head(sg(z_bos)))
        if k == L: lam_k = 1                    # absorb remaining mass
        p_k = p_left * lam_k
        p_left *= 1 - lam_k
        p.append(p_k)
        z_bar += p_k * z
        if all_sequences(p_left < eps): break
else:                # scalar | low_rank | per_dim gate
    S_c, z_bar, p_ch = ones(C), 0, []
    for k in 1..L:
        z_new = f(z + inj)                    # recurrent proposal
        g_ch = sigmoid(gate_head(sg(z_new_bos)))
        if k == L: g_ch = 0                    # absorb remaining channel mass
        S_D = Map_A(S_c)                      # map accumulated survival to features
        z = z + S_D * (z_new - z)
        p_k_ch = S_c * (1 - g_ch)               # channel-wise halt mass
        p_ch.append(p_k_ch)
        w_k = Map_A(p_k_ch) if per_dim_decode else mean(p_k_ch)
        z_bar += w_k * Norm(z)
        S_c *= g_ch
        if all_sequences(max_channel(S_c) < eps): break

Loss  = L_MLM(LM_head(Norm(Coda(z_bar))), labels)
Loss -= entropy(p if mode == "halt_only_GMR" else p_ch)
return Loss
```

Listing 1: **Python-styled pseudocode** for a single training-time forward pass of recAMPLIFY with Gated Mixture Recurrence (GMR). Map_A broadcasts a scalar gate, applies the learned low-rank assignment for the low-rank gate, or is the identity for per-dimension gate.

B PRETRAINING SETUP

Following AMPLIFY (Fournier et al., 2024), we train for 100K steps with a 210B-token budget on 8 NVIDIA H100 GPUs. This includes 32,768 tokens per device, packed into a single FLASHATTENTION varlen buffer with 8 gradient-accumulation steps on the training mixture of UniRef100 (Suzek et al., 2007) for general proteome, SCOP (Murzin et al., 1995) for structural domains, and OAS (Olsen et al., 2022) for antibody sequences. We use the standard MLM objective with masking rate of 0.15 and truncate the input sequences to a max length of 512. Optimization uses AdamW (Loshchilov & Hutter, 2019) with (β_1=0.9, β_2=0.95) alongside a weight decay factor of 10^{-2}. We apply gradient clipping with a cosine schedule with 10% minimum-LR floor and 1,000-step warm-up. Unlike AMPLIFY, we use a 10× larger peak learning rate for the recurrent models (10^{-3} vs. the baseline's 10^{-4}). We empirically found that the shared recurrent block tolerates and requires the higher rate to converge in the same step budget. Architecturally, our 155M-parameter backbone matches AMPLIFY-350M 's hidden width of 960 with 15 attention heads each of dimension 64 alongside FFN width of 2,560 and RoPE (Su et al., 2024) with θ=10,000. Following Banino et al. (2021), we set $\varepsilon = 0.001$.

Dataset. We pretrain on the mixture of three corpora: UniRef100 (Suzek et al., 2007) for the general proteome, SCOP-e (Fox et al., 2014) for structural domain diversity, and OAS (Olsen et al., 2022) for antibody sequences. During training, we do not apply explicit resampling or per-corpus weighting.

Each sequence therefore appears in the training stream in proportion to the raw size of its source corpus, so UniRef100 contributes the vast majority of samples while SCOP and OAS make up for the underrepresented structural and antibody coverage. Training batches are formed by a token-budget sampler that greedily packs sequences of variable length into blocks of $32{,}768$ tokens per device, with per-epoch shuffling over the concatenated index, so every batch is a well-mixed sample drawn across all three corpora. The three held-out validation splits are evaluated separately, yielding one perplexity (PPL) curve per corpus (Figure 4a).

Why do we choose the pretraining recipe of AMPLIFY over ESM-2's UniRef50/90? Fournier et al. (2024) argue that UniRef50, obtained by clustering UniRef100 at 50% identity, systematically distorts the natural sequence distribution in three ways. First, cluster size in UniRef100 is an observational signal. Since large clusters correspond to families for which many independent sequences have been deposited, their redundancy reflects real evidence for the underlying subspace of the fitness landscape. However, 50%-identity clustering discards this signal and upweights singletons, which are more likely to derive from sequencing errors. Second, the fraction of ambiguous reads, fragmentary sequences, and hypothetical proteins in UniRef50 is markedly higher than in UniRef100. Third, specific functional classes are depleted under 50%-identity clustering. AMPLIFY's own 350M model trained on the mixture of UniRef100, OAS and SCOP matches or exceeds ESM-2 15B on the proteome completeness weighted validation despite the $\sim 43\times$ scale gap, which we take as empirical confirmation of the recipe.

C Evaluation Setup

C.1 Evaluation tasks

We describe the four-task rubric introduced in §5 in detail. The full pipeline is a two-stage protocol: a zero-shot stage in which the model is queried through its frozen MLM head, and a supervised stage in which frozen per-token embeddings are extracted once and used to train a downstream MLP head under a controlled hyperparameter search.

C.1.1 Zero-shot tasks

We evaluate our frozen MLM heads on the following three zero-shot tasks:

A) **KL-calibration to human MSA consensus (T, NT).** We consider $6{,}868$ human proteins. For each of these, we mask each residue position i in turn and read the model's per-position 20-way amino-acid distribution $p_\theta(\cdot \mid S_{\setminus i})$ from the MLM head. Independently, we compute a consensus profile $\mathbf{f}_i \in \Delta^{19}$ at each position from a curated multiple sequence alignment (MSA) of homologs. For each position, we report the scaled calibration score: $\mathrm{cal}_i = 100\, \exp\big(-D_{\mathrm{KL}}(\mathbf{f}_i \| p_\theta)\big) \in [0, 100]$, where higher is better. We separately average across positions where the wild-type residue is the consensus (target, T) and where it is not (non-target, NT). This split isolates the target (T) predictions that the model should trivially get right from the non-target (NT) predictions that require distinguishing evolutionarily-tolerated variants. Higher T rewards accurate consensus recovery while higher NT implies better generalization.

B) **ProteinGym DMS mutation-effect ranking.** We use the full substitution benchmark of ProteinGym (Notin et al., 2023) containing 217 deep mutational scanning assays covering $2{,}465{,}767$ single- and multi-mutant variants. For each assay, mutation effects are scored via the ESM-2-style masked-marginal formula popularised by Meier et al. (2021):

$$s(\mathrm{mt}) \;=\; \sum_{i \in \mathcal{M}(\mathrm{mt})} \log p_\theta(\mathrm{mt}_i \mid S_{\setminus i}) - \log p_\theta(\mathrm{wt}_i \mid S_{\setminus i}), \tag{4}$$

where $\mathcal{M}(\mathrm{mt})$ enumerates the mutated positions. Per-assay Spearman rank correlation between predicted and measured DMS scores is reported, then averaged across assays. No supervision from DMS labels is used. It is worth noting this is a strictly newer benchmark than the DMS collections used in Lin et al. (2023), which pre-dates ProteinGym.

C) **CASP16 categorical-Jacobian contacts.** We evaluate residue-residue contact prediction on 41 CASP16 targets with a median length of 291 and a max length of 466, via the categorical-Jacobian construction of Zhang et al. (2024). Here, for each residue pair (i, j), we measure how much perturbing (masking and re-sampling) the input at position j shifts the MLM output distribution at position i, aggregate across all 20 amino-acid perturbations, and rank residue pairs by the resulting influence score. Precision P at the length T of the protein sequence in residues over long-range contacts ($|i - j| \geq 6$) is reported, i.e. $P@T$.

C.1.2 SUPERVISED TASKS

The supervised tasks in our rubric are based on the 18-task biomap-research suite curated by Chen et al. (2025) that spans four biological categories — *structure*, *developability*, *interaction*, and *function* — and provides one benchmark per level of granularity (residue, sequence, and sequence-pair) with a mix of classification 'C' and regression 'R' targets. It is the union of tasks inherited from the widely used FLIP (Dallago et al., 2021) and TAPE (Rao et al., 2019) suites (secondary-structure, fold, fluorescence, stability, GB1 fitness, DeepLoc localization, solubility). Our setup matches the composition of the suite from Chen et al. (2025) except that the residue-level SSP task is split into Q3 *and* Q8 variants. The contact-map task from the original suite is skipped as it duplicates the CASP16 structure probe in our zero-shot rubric (§C.1.1). Table 3 summarizes every task and Table 6 compares the results of baseline AMPLIFY models against our recursive variant at 1M training steps.

Table 3: **The 18-task biomap-research supervised suite** from Chen et al. (2025). "Type" distinguishes classification 'C' from regression 'R' tasks. Sizes are the standard train/test split, where if multiple test splits exist, we report the aggregate size. Results for all 18 tasks are in Table 6.

Task	Type	Metric	Prediction target	Source dataset	|train| / |test|
SSP Q3	C	acc.	3-class per-residue secondary structure (H/E/C)	NetSurfP-2.0 / Ankh split	∼11k / 39
SSP Q8	C	acc.	8-class per-residue secondary structure	NetSurfP-2.0 (Q8 head)	∼11k / 39
Fold prediction	C	acc.	1,195-way fold class from SCOP 1.75	SCOPe / DeepSF split	12,312 / 3,244
Solubility prediction	C	acc.	binary soluble vs. insoluble	DeepSol	62,478 / 6,942
Stability prediction	R	Spearman ρ	protease-concentration fold-retention threshold	TAPE / Rocklin et al. 2017	53,614 / 12,851
Temperature stability	C	acc.	structural stability at 65°C	TemStaPro	283k / 73k
Optimal temperature	R	Spearman ρ	enzyme optimal reaction temperature	DeepET	1,706 / 190
Optimal pH	R	Spearman ρ	enzyme optimal reaction pH	EpHod	7,124 / 1,971
Cloning failure	C	acc.	cloning-stage failure in X-ray crystallography	PredPPCrys	23,375 / 4,791
Material production	C	acc.	material-stage failure in X-ray crystallography	PredPPCrys	23,339 / 4,791
Metal-ion binding	C	acc.	presence of any metal-ion binding site	Cheng et al. 2023	6,000 / 1,332
Enzyme catalytic efficiency	R	Spearman ρ	k_{cat} from substrate + protein pair	DLKcat	13,470 / 1,684
Peptide affinity	C	acc.	peptide binding to HLA/MHC-I (paired input)	CcBHLA	57,357 / 8,406
TCR affinity	C	acc.	TCR binding to peptide-MHC (paired input)	epiTCR / VDJdb	19,526 / 4,485
Fluorescence prediction	R	Spearman ρ	fluorescence intensity of engineered GFP (extrapolation split)	TAPE / Sarkisyan 2016	21,446 / 27,217
Fitness prediction	R	Spearman ρ	GB1 4-position combinatorial fitness	FLIP GB1	6,289 / 1,745
Localization prediction	C	acc.	10-way subcellular localization	DeepLoc	6,622 / 1,842
Antibiotic resistance	C	acc.	19-way antibiotic-resistance class	CARD	2,072 / 1,344

Why do we report only four of the eighteen supervised tasks? Le Breton et al. (2026) find that the aggregate F1 / MCC score over the 18-tasks differs by less than 5% across 5 pLMs (AMPLIFY 120M/350M, ESM-2 35M/150M/650M). To separate genuine pretraining signal from task-intrinsic ceiling effects, they measure the score gap between a shallow head on pretrained embeddings and the same head on randomly initialized weights, averaged over the five models. The results show that only four tasks account for essentially all the pretraining boost: stability prediction (+57.7%), fold prediction (+44.0%), and secondary-structure prediction (+30.4% Q3, +33.7% Q8). Every other task either saturates with random embeddings already recovering most of the accuracy or depends on features that MLM captures only weakly. For instance, TCR-pMHC affinity is 25% lower with pretrained embeddings than with random ones. We follow this observation and report the four structural/stability-sensitive tasks (Fold, SSP mean of Q3 and Q8, Stability, and Localization) in the main-paper Table 1 and Appendix Table 4. We further detail these tasks below.

A) **Fold prediction.** Given an amino-acid sequence, the model must predict its top-1 fold class out of 1,195 SCOP 1.75 folds. We use the DeepSF split with 12,312 train and 3,244 test samples. Because remote homologs are grouped into the same fold, high accuracy requires representations that recognize structural similarity across sequences with $< 25\%$ identity. This is exactly the regime where handcrafted alignment features fail and where pretraining should help. The metric is the sequence-level total accuracy (top-1 fold match) (Rao et al., 2019; Fox et al., 2014).

B) **Localization prediction.** A 10-way sequence-level classification of subcellular localization derived from UniProt annotations of $\sim$8.5k eukaryotic proteins with 6,622 train and 1,842 test samples (Almagro Armenteros et al., 2017). Signal peptides and transmembrane spans are informative but the label ultimately requires integrating long-range motifs and organelle-specific sequence patterns, which frozen bag-of-embedding baselines only partially capture. We report the total test accuracy here.

C) **Stability prediction.** A sequence-level regression on the protease concentration at which a designed mini-protein retains its native fold, from the massively parallel proteolysis assay of Rocklin et al. (2017) with 53,614 train and 12,851 test sequences. The target is a real-valued proxy for thermodynamic stability, and the reported score is the Spearman rank correlation between predicted and measured values across the test split. Le Breton et al. (2026) find stability to be the single largest pretraining-driven task where pretraining performs up to $+57.7\%$ over the random baseline. This makes it a highly noise-sensitive metric to compare pLMs at similar overall scale.

D) **Secondary-structure prediction (SSP).** A per-residue classification into either 3 classes (Q3 with helix, strand and coil) or 8 classes (Q8 with DSSP labels) trained on the NetSurfP-2.0 corpus (Klausen et al., 2019) and evaluated on the CASP12 and CASP14 targets as an average over $\sim$ 39 test proteins with $\sim$ 700 residues each. Q3 and Q8 use the same encoder feature map and differ only in the final softmax head. We report the equal-weight mean of the two per-residue accuracies. Secondary structure is the residue-level task where pretraining most clearly beats random embeddings (Le Breton et al., 2026) because the label is a direct function of local backbone geometry that MLM implicitly learns during masked reconstruction.

C.2 Task head and hyperparameter search for downstream supervised tasks

While evaluating the downstream supervised tasks, the encoder is frozen. Per-token embeddings either from the coda or the recurrent core are extracted once and cached. On top of these embeddings we fit a two-layer MLP head with a task-appropriate output layer, i.e., softmax for classification, single-scalar linear head for regression. The head is fit under a 50-trial Bayesian TPE hyperparameter search over hidden size, dropout, weight decay, learning rate, and batch size. Trials are evaluated on the task's validation split. The trial with the highest validation score is then re-trained and evaluated on the held-out test split with 5 different random seeds. We report the mean (main-paper Table 1) and the standard deviation (Appendix Table 4) of the test-set metric.

D Additional Results

Table 4: **Mean $\pm$ standard deviation scores for supervised probes** reported by rerunning the best-of-50-trial HPO setup over 5 seeds. Small standard deviations signify that most of the gains from recursion are beyond the noise regime.

Model	Fold ↑	Localization ↑	Stability ρ ↑	SSP ↑
1M pretraining steps				
AMPLIFY-120M (1M)	0.563 ± 0.004	0.642 ± 0.004	0.396 ± 0.053	0.639 ± 0.000
AMPLIFY-350M (1M)	0.567 ± 0.005	0.707 ± 0.002	0.461 ± 0.036	0.651 ± 0.000
recAMPLIFY (1M)	0.696 ± 0.003	0.723 ± 0.005	0.501 ± 0.137	0.651 ± 0.001
100K pretraining steps				
AMPLIFY-120M (100K)	0.602 ± 0.004	0.675 ± 0.003	0.562 ± 0.012	0.621 ± 0.000
AMPLIFY-350M (100K)	0.574 ± 0.003	0.664 ± 0.004	0.440 ± 0.049	0.641 ± 0.000
recAMPLIFY (fixed-L)	0.611 ± 0.005	0.671 ± 0.003	0.418 ± 0.139	0.630 ± 0.001
recAMPLIFY (varying-L)	0.649 ± 0.004	0.685 ± 0.014	0.282 ± 0.073	0.638 ± 0.001
+GMR (halt-only)	0.632 ± 0.005	0.702 ± 0.002	0.572 ± 0.007	0.644 ± 0.000
+GMR (scalar gate)	0.630 ± 0.002	0.699 ± 0.003	0.346 ± 0.087	0.636 ± 0.001
+GMR (rank-16 gate)	0.623 ± 0.002	0.568 ± 0.171	0.419 ± 0.024	0.636 ± 0.000
+GMR (per-dim gate)	0.622 ± 0.004	0.700 ± 0.006	0.469 ± 0.012	0.637 ± 0.001

Table 5: **Throughput comparison for our recursive pretraining recipe** at three cycle budgets L against that of the non-recurrent AMPLIFY baselines. "Blocks" is the effective depth $\mathcal{D}_{\text{eff}} = p + Lr + c$ used in Table 1, i.e., prelude, L applications of the tied recurrent core, and coda. The 133M recAMPLIFY rows use $(p, r, c) = (2, 8, 2)$. The last column normalizes the throughput by that count. All reported values are measured over 500 optimizer steps on one NVIDIA A100-40GB with 32 sequences per device packed into a single FLASHATTENTION varlen buffer ($\approx$10.9K tokens) using `bf16` precision without gradient accumulation.

Model	Model size			Measured training cost		
	Trainable parameters	Hidden width D	Blocks per forward pass	Throughput (K tokens/s)	Relative throughput	Throughput per block (M blk·tok/s)
AMPLIFY-120M	118M	640	24	142.0	1.00×	3.41
AMPLIFY-350M	354M	960	32	59.4	0.42×	1.90
recAMPLIFY (fixed $L = 3$)	133M	960	28	55.8	0.39×	1.56
recAMPLIFY (fixed $L = 5$)	133M	960	44	38.9	0.27×	1.71
recAMPLIFY (fixed $L = 7$)	133M	960	60	29.9	0.21×	1.79

Table 6: **Full 18-task supervised comparison at 1M pretraining steps.** Each cell is mean $\pm$ standard deviation of the test-set metric across 5 random-seed re-trainings of the best-of-50-trial HPO winner (§C.2). "Metric" is accuracy for classification and Spearman rank correlation for regression; higher is better for every metric. AMPLIFY-120M and AMPLIFY-350M are the standard AMPLIFY 1M-step checkpoints and recAMPLIFY is our recursive 1M checkpoint with $(p, r, c) = (2, 10, 2)$ that is trained and evaluated at L_{eval}=L=6. Best cell per task is **bold**. recAMPLIFY wins on 14 of 18 tasks despite having 56% fewer unique parameters than AMPLIFY-350M.

Task	Metric	AMPLIFY-120M	AMPLIFY-350M	recAMPLIFY
SSP Q3	acc.	0.706 ± 0.000	0.715 ± 0.001	**0.717 ± 0.001**
SSP Q8	acc.	0.573 ± 0.001	**0.586 ± 0.000**	0.586 ± 0.001
Fold prediction	acc.	0.563 ± 0.004	0.567 ± 0.005	**0.696 ± 0.003**
Solubility prediction	acc.	0.689 ± 0.013	**0.717 ± 0.000**	0.713 ± 0.002
Stability prediction	Sp. ρ	0.396 ± 0.053	0.461 ± 0.036	**0.501 ± 0.137**
Temperature stability	acc.	0.917 ± 0.002	**0.938 ± 0.002**	0.922 ± 0.001
Optimal temperature	Sp. ρ	**0.285 ± 0.027**	0.282 ± 0.003	0.020 ± 0.100
Optimal pH	Sp. ρ	-0.109 ± 0.004	-0.101 ± 0.001	**0.214 ± 0.097**
Cloning failure	acc.	0.699 ± 0.005	0.703 ± 0.003	**0.747 ± 0.011**
Material production	acc.	0.782 ± 0.001	0.774 ± 0.005	**0.796 ± 0.001**
Metal-ion binding	acc.	0.714 ± 0.003	0.737 ± 0.005	**0.800 ± 0.007**
Enzyme catalytic efficiency	Sp. ρ	0.003 ± 0.002	0.028 ± 0.003	**0.408 ± 0.015**
Peptide affinity	acc.	0.654 ± 0.010	0.729 ± 0.001	**0.854 ± 0.004**
TCR affinity	acc.	0.834 ± 0.000	0.834 ± 0.000	**0.862 ± 0.013**
Fluorescence prediction	Sp. ρ	-0.093 ± 0.057	-0.007 ± 0.011	**0.123 ± 0.148**
Fitness prediction	Sp. ρ	-0.040 ± 0.124	-0.171 ± 0.151	**0.369 ± 0.009**
Localization prediction	acc.	0.642 ± 0.004	0.707 ± 0.002	**0.723 ± 0.005**
Antibiotic resistance	acc.	0.943 ± 0.010	0.829 ± 0.169	**0.967 ± 0.002**

Recursion offers a better representation choice for training supervised probes. Besides the standard coda output $f_c(\widetilde{\mathbf{z}})$, recAMPLIFY exposes a second readout choice for training downstream supervised task probes: the refined latent state $\bar{\mathbf{z}}$. Figure 8 shows that the latent state $\bar{\mathbf{z}}$ provides a better representation space across most of the supervised tasks for recurrence depth $L < 4$ seen during training. However, in the unseen depth regime (L_{eval}>3), this representation space lags in robustness compared to downstream probes trained on coda outputs. Given its superior performance for seen recurrence depths, we report our supervised probe results using the latent state throughout the paper.

Design decisions that do not scale well for pretraining. Inspired by recent line of works on recursive refinement in limited data regime, we ablate the effect of truncated backpropagation (BPTT) (Jolicoeur-Martineau, 2025) and context type (Liao & Poggio, 2026). Figures 10 and 11 show that neither design helps at pretraining scale: truncated BPTT degrades supervised probes, most sharply on stability, and a learned context transform (fzL0) collapses, while the identity ('zero') and

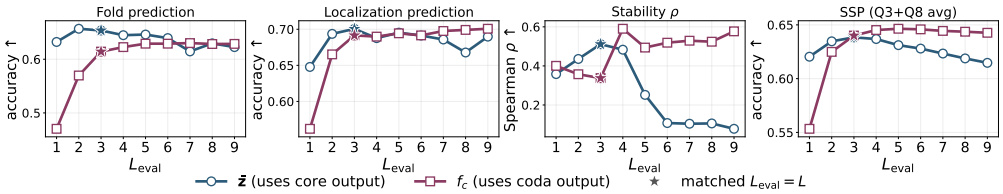
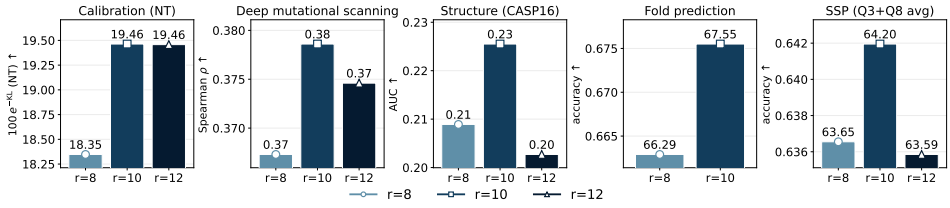
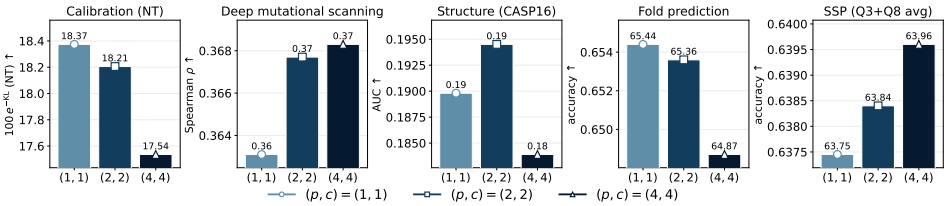


Figure 8: **Performance comparison of different representation choices used for training downstream supervised task probes:** z uses the output of the recurrent core while *coda* is the standard pre-LM head output $f_c(\tilde{\mathbf{z}})$. We sweep $L_{\text{eval}} \in \{1, \ldots, 9\}$ on the same recAMPLIFY trained with $(p, r, c) = (2, 8, 2)$, and L=3 ($\star$ denotes L_{eval} with matched L).



(a) Varying recursive layers r while fixing $(p, c) = (2, 2)$.



(b) Varying the prelude p and coda c while adjusting r to keep the number of layers $p + r + c = 12$.

Figure 9: **Ablation of physical layer counts:** we train each variant for 100K steps with $L = 4$ and evaluate at the same $L_{\text{eval}} = 4$. Overall, $r = 10$ performs best in (a), while the prelude/coda split in (b) changes every probe by less than about 1 pp.

zL0 contexts perform comparably. We therefore train with full BPTT and an identity transformation of the previous cycle's latent state.

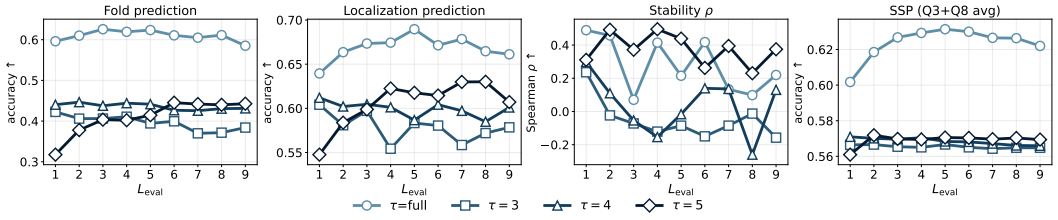


Figure 10: **Effect of truncated backpropagation (BPTT) steps on supervised probes.** We sweep across $L_{\text{eval}} \in \{1, \ldots, 9\}$ for four checkpoints that are trained with $L = 6$ but differ in the truncated backprop-through-time window τ: full BPTT (τ =full, gradients through every cycle) and truncated windows $\tau \in \{3, 4, 5\}$. Here, the last τ cycles receive gradients, while the earlier $L - \tau$ do not. Full-BPTT holds up across the sweep and remains the only recurrent variant competitive with AMPLIFY-350M on the supervised probes (main-paper Table 1). Truncated variants degrade sharply, especially on the stability prediction task which is known to benefit the most from pretraining (Le Breton et al., 2026).

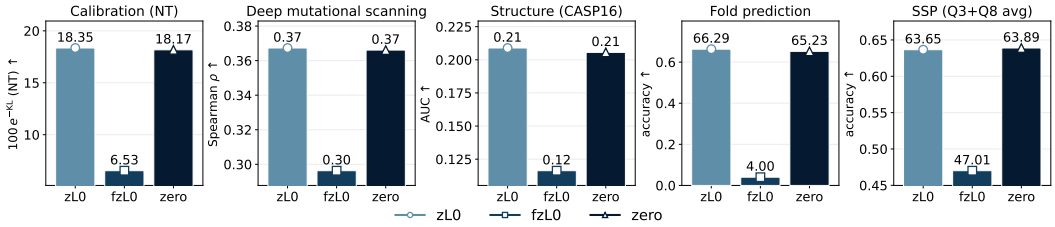


Figure 11: **Varying context type for $L = 4$.** All models are evaluated at the matched $L_{\text{eval}} = 4$ depth (Liao & Poggio, 2026). The simplest context type 'zero' which uses an identity transformation of the previous cycle's latent z_{k-1} performs surprisingly well throughout all tasks.

E Additional mechanistic analyses

Gates help achieve stable attention geometry. For the recurrent block to converge to an actual fixed point, the attention pattern it induces must also stabilize in addition to the hidden states. Figure 14 shows that once past the trained horizon ($L > 6$), the scalar gate's cross-cycle attention-map distances collapse, whereas fixed-depth and baseline GMR retain cycle-dependent attention maps alongside their continued state drift. Low-rank gate's attention distances also fail to collapse which is consistent with its magnitude drift.

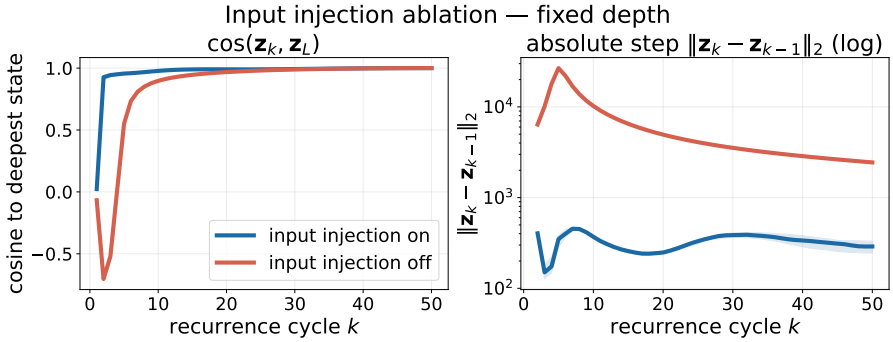
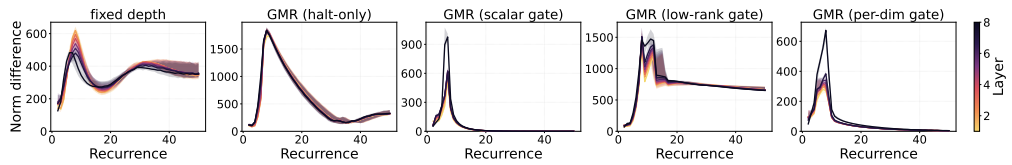


Figure 12: **Input-injection ablation** for recAMPLIFY. Blue line shows a normal forward pass while red denotes injection stream removed from the recurrent core. **Left:** $\cos(\mathbf{z}_k, \mathbf{z}_L)$. **Right:** $\|\mathbf{z}_k - \mathbf{z}_{k-1}\|_2$ on log-y. With input injection, the state is a bounded iterated map ($\|\Delta\mathbf{z}\|_2$ plateaus below 10^3). Without the injection, the state diverges by three orders of magnitude within a few cycles. Input injection thus helps keep the recurrence bounded, matching the finding of Blayney et al. (2026) on causal-LM looped Transformers. We note that this boundedness does not imply convergence. The blue plateau in the right panel stays around $\|\Delta\mathbf{z}\|_2 \approx 300$ throughout the OOD range, motivating main-paper Figure 5.

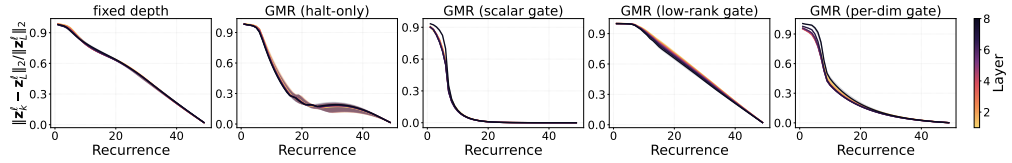
ColSum concentration. For a row-stochastic attention matrix A of a sequence with real-token length T, we define the column mass $c_j = \sum_i A_{ij}$ and its length-normalized version $\hat{c}_j = c_j/T$. The ColSum concentration is then given as:

$$C \;=\; 1 \;+\; \frac{1}{\log T} \sum_j \hat{c}_j \log \hat{c}_j \quad \in \quad [0, 1], \tag{5}$$

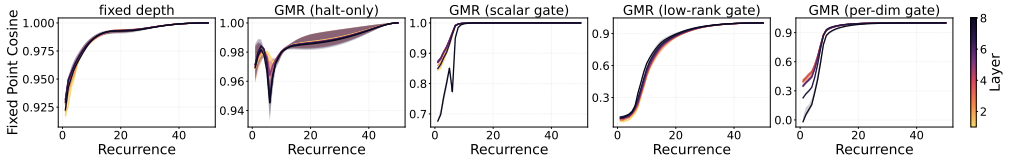
where the right hand side term is the residual of the length-normalized entropy of column-mass distribution. $C{=}0$ therefore corresponds to uniform mixing, where every key receives an equal share of the attention mass and $C{=}1$ corresponds to all mass pooled on a single key. We compute C per attention head, per protein, and per physical layer, and report the mean over heads and over the 32 probe proteins of our stability probe. Query positions are already summed inside $\hat{c}_j$ so no averaging over query indices is needed. The dashed red curve throughout §5.1.2 is our *feedforward reference* consisting of a non-recurrent recAMPLIFY with identical width, attention heads, and $p{+}r{+}c = 12$



(a) Difference across successive iterations $\|\mathbf{z}_k^\ell - \mathbf{z}_{k-1}^\ell\|_2$.



(b) Normalized distance to the deepest state.



(c) Cosine similarity $\cos(\mathbf{z}_k^\ell, \mathbf{z}_L^\ell)$ with the deepest recorded state.

Figure 13: **Per-layer dynamics of each recurrent block layer ℓ on the stability prediction task** (median over 32 probes). (a) Successive-difference norm: gating drives each layer's step size toward zero past the L=6 cycles seen during training. (b) Normalized operator residual to each layer's deepest recorded state $\mathbf{z}_L^\ell$ (taken as an approximate fixed point) over $k \in \{1, \ldots, L-1\}$ cycles shows the key diagnostic being the *rate* and *timing* of decay. Scalar and per-dim gates reach near-zero residual within a few cycles of the trained horizon, whereas halt-only and rank-16 decay only slowly and remain substantially non-zero at intermediate cycles. (c) Cosine similarity to $\mathbf{z}_L^\ell$: alignment is near-perfect for every variant, so cosine is an *indirect* diagnostic and must be read jointly with (a) and (b). Read together, only the scalar and per-dimension gates deliver empirical fixed-point convergence.

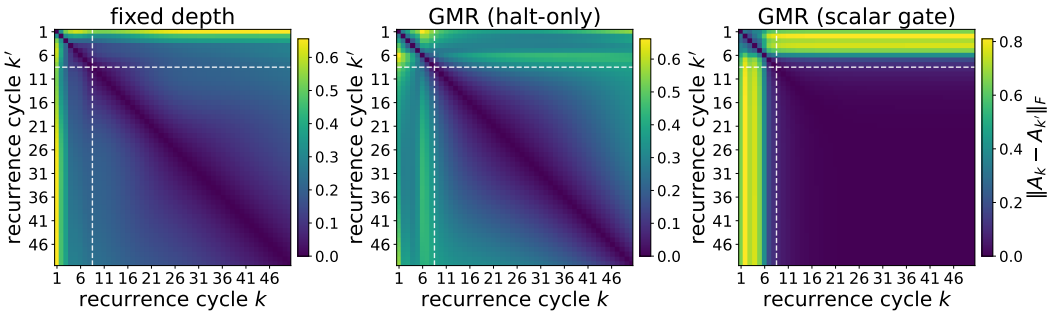


Figure 14: **Post-halt attention stability.** Head-averaged attention Frobenius distance between recurrent cycles for three representative variants at L_{eval}=50, each cell (k, k') is $\|A_k - A_{k'}\|_F$ averaged over B=4 probe sequences from stability prediction task, and the dashed line marks the trained halt horizon at $k \approx 7$. For the variant with fixed test-time depth and varying-L pretraining (left), attention keeps drifting through all 50 cycles, despite the recurrence being bounded (Figure 12). For halt-only GMR (middle), a discontinuity appears at the halt horizon but off-diagonal cells continue to darken smoothly. For scalar gate, cycles 8–50 collapse into uniform differences. The comparison isolates the role of the residual gate where neither variable-L training nor a halt signal on its own produces a post-halt attention fixed point, only the gate achieves this.

total layers, trained at $L = 1$ so the core layers are applied untied. A non-flat layerwise profile of C signals mixing heterogeneity across physical layers while proximity to the dashed red curve signals fidelity to the feedforward reference in both shape and amplitude.

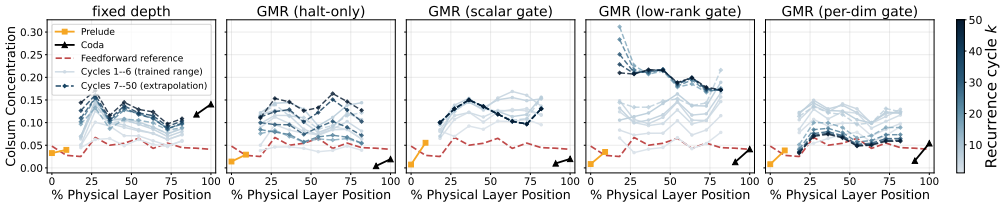


Figure 15: ColSum concentration vs. physical layer position across five recAMPLIFY variants, evaluated at $L_{\text{eval}} = 50$. Solid lines with circles are the trained cycles ($[1, 6]$). Dashed lines with diamonds: the six evenly-spaced samples from the extrapolation range $[7, 50]$, orange square: prelude, black triangle: coda. The dashed red is the feedforward reference which is a recAMPLIFY trained with the same $(p, r, c) = (2, 8, 2)$ physical layer counts without recursion, i.e., $L = 1$.

Table 7: Partial correlations between GMR halt depth and sharpness across probing tasks alongside the compute budget: **raw** $= |r(\mathbb{E}[k], \text{err})|$ | $\log T$ denotes the partial correlation between per-protein expected halt depth $\mathbb{E}[k]$ and task-specific per-protein error err, after regressing out the log sequence length $\log T$; **ctrl** $= |r(\mathbb{E}[k], \text{err})|$ | $\log T, \log \text{Sharp}$ denotes the same after additionally regressing out the task-specific sharpness proxy $\log \text{Sharp}$; **coup** $= r(\mathbb{E}[k], \log \text{Sharp})$ | $\log T$ is the signed length-controlled partial correlation of $\mathbb{E}[k]$ with $\log \text{Sharp}$. Per-task sharpness proxy: $\log(p_\theta / p_{\text{MSA}})$ on the consensus amino acid for KL; negative softmax entropy of the classifier for Fold and Localization. The two rightmost columns give per-protein $\mathbb{E}[k]$ (on the KL probe) and the potential budget reduction $1 - \mathbb{E}[k]/L$ with respect to the fixed L=6 budget under hard stopping.

	KL calibration			Fold prediction			Localization prediction			ProteinGym DMS			Compute	
Variant	raw	ctrl	coup	raw	ctrl	coup	raw	ctrl	coup	raw	ctrl	coup	$\mathbb{E}[k]$	reduction
Halt-only GMR	0.42	0.00	+0.42	0.09	0.00	−0.15	0.07	0.01	−0.15	0.13	0.08	+0.10	3.57	40%
scalar gate	0.28	0.04	+0.28	0.12	0.05	−0.13	0.12	0.07	−0.16	0.06	0.01	+0.10	4.07	32%
low-rank-16	**0.59**	0.00	**+0.59**	0.06	0.02	+0.07	0.07	0.04	−0.08	0.00	0.04	+0.05	4.06	32%
per-dim	0.21	0.00	+0.21	0.01	0.03	−0.03	0.11	0.08	−0.10	0.05	0.04	−0.04	4.08	32%

F Additional Results on GMR Depth Encoding Biological Signal

Compositional axes of the GMR halt signal. Main-paper Table 2 shows two independent per-protein composition signals for the length- and sharpness-controlled GMR halt depth $\mathbb{E}[k]$: high sequence entropy and high hydrophobic (FILMVWY) content both correlate with depth. We now show that these are two projections of a single compositional-vocabulary axis. Figure 17 bins the 10K UniProt held out sequences into deciles based on their hydrophobic content against either Shannon entropy (left) or Jensen–Shannon (JS) divergence to uniform (right), where the JS divergence particularly remains less sensitive to rare amino acids (AAs). Both metrics tell us the same story in that they are near perfectly anti-correlated (Pearson -0.99) and both show the highest cycles to be clustered at the high-hydrophobicity end, with halt depth mostly dropping as hydrophobicity falls.

Figure 16 decomposes the effect per amino acid, with the seven hydrophobic/aromatic residues grouped at the bottom and columns being within-AA composition deciles. Each cell reports the median $\mathbb{E}[k]$ residual for sequences in that AA's decile bin. The pattern splits cleanly along the horizontal separator: every one of the seven hydrophobic AAs (F, L, I, V, M, W, Y) shows an overall increase, i.e., higher composition of a hydrophobic residue drives more GMR cycles in general. By contrast, every non-hydrophobic AA (from A downward) shows an inverted U-shape, where extreme dominance of a single polar or charged residue drives *fewer* recurrent cycles.

The two mechanisms are the two sides of the same question — how many independent decisions does this residue distribution actually require? Overrepresentation of a hydrophobic residue is evolutionarily rare because random sequences with that composition would aggregate. The ones that survive do so through a specific fold or a membrane pocket, and recovering the wild-type residue at masked positions requires the model to integrate long-range structural evidence. Overrepresentation of any other residue is instead an alphabet-reduction cue, where once a polar or charged class dominates, many masked positions are predictable from local statistics and the model can commit early. The halt signal thus reads a reduced-alphabet grammar in the sense that restricted vocabularies get fewer cycles, unless the vocabulary happens to be the demanding kind.

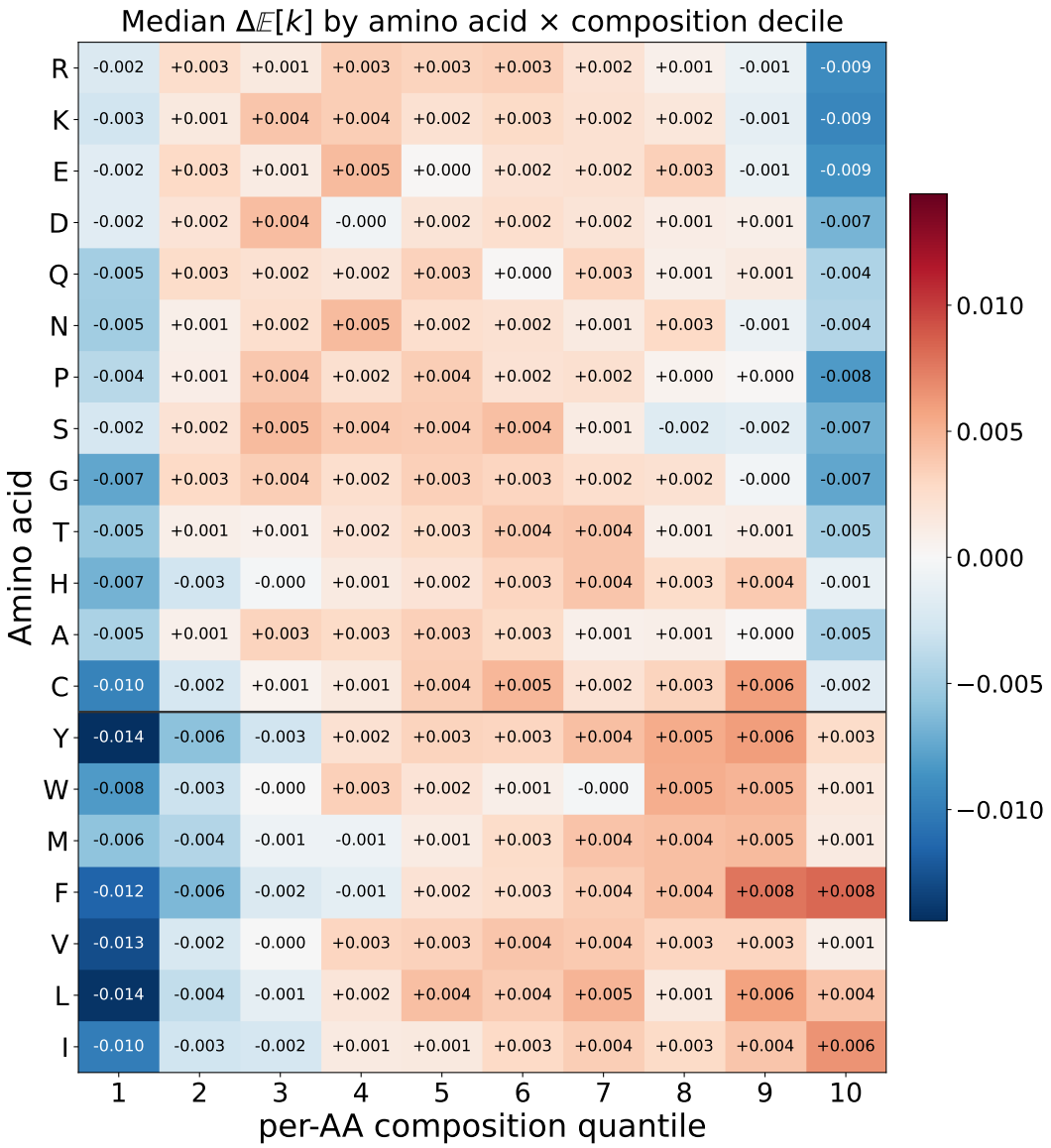


Figure 16: **Median residual halt depth $\mathbb{E}[k]$ by amino acid $\times$ per-AA composition decile.** Rows are grouped into the seven hydrophobic/aromatic residues (bottom) and the remaining residues (top). The horizontal line separates the hydrophobic block (I, L, V, F, M, W, Y) from the non-hydrophobic block above. Columns are within-AA composition deciles (bin 1: lowest fraction, bin 10: highest). Hydrophobic AAs show increases over most of the range: heavier hydrophobic content in general drives more GMR cycles. Non-hydrophobic AAs show an inverted U-shape with extreme composition dominance driving fewer cycles.

Splice-isoform collection and analysis. For our analyses on alternative isoform sequences in Table 2, we use curator-annotated Swiss-Prot alternative isoforms, fetched from UniProt REST (`rest.uniprot.org/uniprotkb/stream`) with `includeIsoform=true` on the human accessions of our 10K UniProt held out split. UniProt returns for each accession the canonical sequence together with every alternative isoform Swiss-Prot curators derived from experimental transcript evidence (alternative splicing, alternative promoter usage, alternative initiation, etc., depending on the entry). We keep the isoforms as returned (no truncation, mutation, or ad-hoc reconstruction) and identify them by the standard UniProt scheme `P12345-1`, `P12345-2`, and so on. The raw fetch returns 1,730 isoform sequences across the human accessions in the held out split. After restricting to families with at least two isoforms and applying the length- and sharpness-controlled

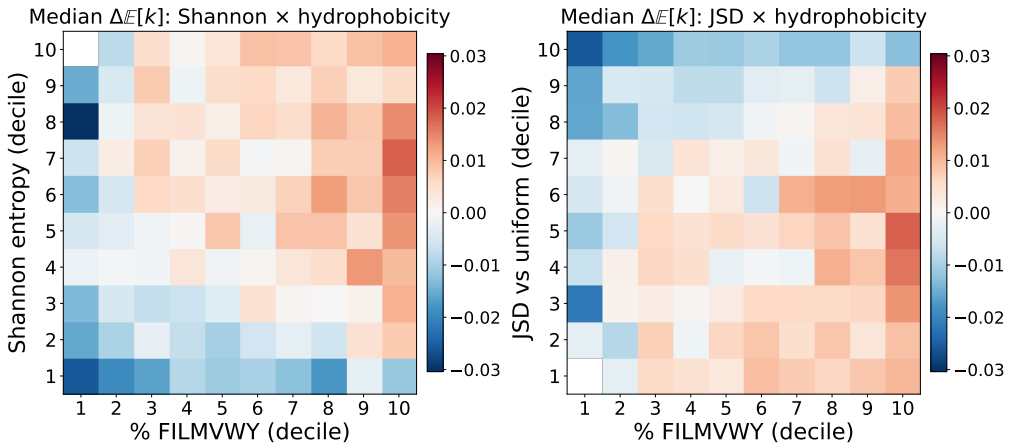


Figure 17: **Median residual halt depth $\mathbb{E}[k]$ binned by composition-diversity $\times$ hydrophobicity.** Left: Shannon entropy of the AA distribution (rows) against % of FILMVWY (columns). Right: Jensen–Shannon divergence of the AA distribution to the uniform $1/20$ prior (rows) against % of FILMVWY (columns). Both panels are computed on the 10K UniProt held out sequences of §5.1.3 and use $\mathbb{E}[k]$ residualised against sequence length and sharpness. Diverging colour scale is centred at 0. Positive (red) means the model spends more cycles than a length- and sharpness-matched null.

halt-depth aggregation of §5.1.3, we obtain 335 genes covering 1,054 measured isoforms (family-size distribution: 163 genes with two isoforms, 83 with three, 44 with four, and a long right tail up to one gene with 17). For each family we compute the within-family standard deviation of the controlled $\mathbb{E}[k]$. The observed mean within-family std (0.013) is 46% smaller than the size-matched bootstrap null over the full held out population (0.025; $p < 10^{-4}$ over 10^4 resamples), yielding the Cohen's $d = 0.79$ reported in the main paper.

G Limitations

This is the first work targeting adaptive computation for test-time scaling of pLMs. As such, several caveats bound our claims. First, per-protein expected halt depth $\mathbb{E}[k]$ varies within only $\sim$1% of its mean across the 10K UniProt held out set. GMR's adaptivity is thus a small, structured modulation around a learned mean. Second, state-level convergence is characterized empirically through vanishing update norms and residual decay. We do not establish local stability of the recurrent operator via a Jacobian analysis. Third, gates are per-sequence rather than per-token, i.e., computation adapts across proteins but not within a sequence, and moving to per-token survivals is a natural extension. Fourth, our ESM-2 comparison is confounded by both a 4–18$\times$ parameter-count gap and the unreleased ESM-2 code and validation splits (§B). We therefore treat those rows as reference, not as directly comparable baselines. Finally, all experiments are on recAMPLIFY variants trained under the AMPLIFY data recipe, and generalization to substantially different scales, architectures, or modalities remains open.

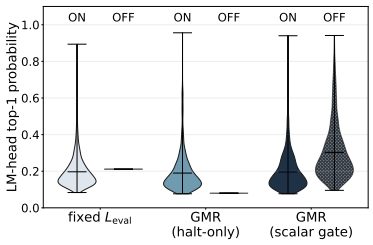


Figure 18: LM-head top-1 probability per real token on stability prediction, evaluated with (ON) and without (OFF) input-injection.

Gate's stability comes at the price of reduced input sensitivity. Figure 18 shows that removing input injection *collapses* the fixed depth and the halt-only GMR states toward a null attractor with the halt-only variant's decode particularly collapsing to a low-confidence generic token. On the contrary, the scalar gate becomes more confident about a token that disagrees with its own input-injected prediction on $\sim$90% of positions. This is a specific artefact of residual gating with low degrees of freedom. With only one survival channel, the scalar gate's survival distribution collapses to an input-invariant trajectory. The gate thus buys representational stability by substituting a stored default in place of a truly input-driven recurrence.