REINFORCEMENT LEARNING FOR BLACK-BOX OBJECTIVES IN LOGIT-BASED PROTEIN HALLUCINATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Gradient-based "hallucination" methods such as Germinal and BindCraft enable efficient protein binder design by optimizing a continuous relaxation of the sequence (a logit matrix) using gradients from differentiable structure predictors and protein language models. However, many practically useful objectives for ranking or filtering designs are non-differentiable with respect to sequence (e.g., external confidence metrics from AlphaFold3 or Chai, experimental readouts, or arbitrary black-box scores), preventing direct backpropagation through the objective. We extend the Germinal pipeline with an *optional* policy-gradient update on the sequence logits, enabling direct optimization of black-box rewards while preserving Germinal's differentiable optimization backbone. Our implementation reuses Germinal's existing filter metrics as modular reward components and supports both Chai-1 and AlphaFold3 backends for reward evaluation. On a nanobody design task against PD-L1, adding a small policy-gradient term during the *high-softness* portion of Germinal's optimization yields a 2.3× improvement in acceptance rate among processed trajectories, from 0.161 ± 0.020 to 0.370 ± 0.055 (mean±std over six seeds), while maintaining comparable confidence metrics for accepted designs. This suggests that combining policy-gradient-based black-box optimization helps improve design success rates, potentially improving downstream wet-lab metrics.

1 INTRODUCTION

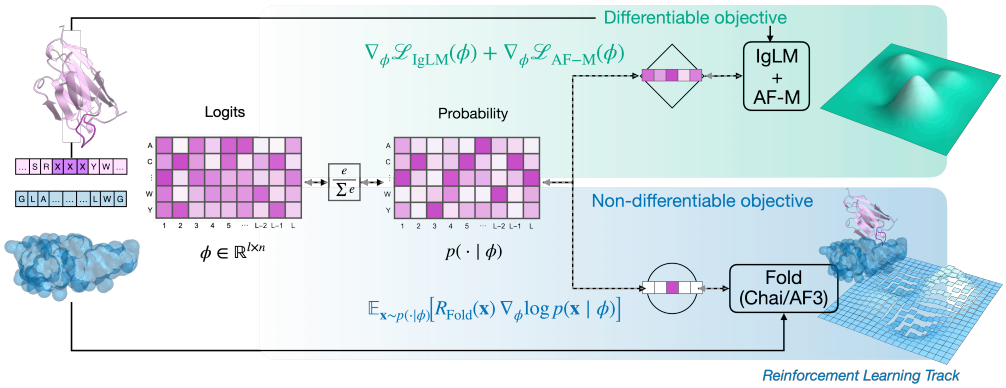

Figure 1: Overview of the RL-augmented logit-optimization pipeline. Logits are updated by combining differentiable hallucination gradients with a policy-gradient term computed from black-box rewards.

Designing functional protein binders requires searching a large discrete sequence space under structural and interface constraints. Recent progress has come from gradient-based design pipelines that optimize sequence logits through differentiable structure models and sequence priors Frank et al. (2024); Pacesa et al. (2025); Mille-Fragoso et al. (2025). These approaches are effective, but they are naturally limited to objectives that can be backpropagated.

Two families dominate current practice. Hallucination-based methods optimize sequences directly against structure-model losses and can produce strong sequence–structure consistency Frank et al. (2024); Pacesa et al. (2025); Mille-Fragoso et al. (2025). Diffusion-based pipelines sample candidate

structures quickly and then design sequences conditioned on those structures Bennett et al. (2024); Watson et al. (2023); Gruver et al. (2023). Both families still rely heavily on downstream filtering and cannot directly optimize black-box success criteria. For a larger discussion of related works, please see A.1

For example, confidence metrics from external structure predictors like AlphaFold3 Abramson et al. (2024) are often useful for ranking candidates, but are not differentiable with respect to sequence. Reinforcement learning offers a natural route to optimize such objectives without requiring differentiability da Silva et al. (2023); Ding et al. (2024); Lee et al. (2025).

We propose a minimal, modular RL augmentation for gradient-based protein hallucination pipelines. Our method adds an optional policy-gradient term to the existing logit optimization loop, supports rewards from either internal design metrics or external predictors (e.g., Chai-1 or AlphaFold3 Chai Discovery (2024); Abramson et al. (2024)), and can be activated only in selected optimization regions (e.g., late logits / softmax) to control variance and compute. When RL is disabled, the pipeline is reduced to the baseline hallucination method.

The central question is practical: *can sparse RL updates improve end-to-end design yield without destabilizing the core hallucination workflow?* We evaluate this question on PD-L1 nanobody design and report acceptance-rate behavior across multiple independent seeds.

2 METHOD

RALO is implemented as an *optional extension* to Germinal Mille-Fragoso et al. (2025). We retain Germinal's core optimization loop—gradient-based hallucination on a logit parameterization of sequences—and add a policy-gradient term that can optimize *black-box* objectives that do not admit backpropagation. Figure 1 provides a high-level overview.

2.1 BACKGROUND: GERMINAL OPTIMIZES A LOGIT DISTRIBUTION

Let $\mathbf{x} = (x_1, \ldots, x_N)$ be an amino-acid sequence of length N over an alphabet of size M (typically $M = 20$). Germinal maintains logits $\phi \in \mathbb{R}^{N \times M}$ and defines an independent categorical distribution using Eq. (1). Gradients from a structure prediction loss (AlphaFold2-Multimer Evans et al. (2022)) and sequence prior (IgLM Shuai et al. (2023)) are merged (e.g., with PCGrad Yu et al. (2020)) and applied to the logits during a three-phase schedule: *Phase 1 (logits)*, *Phase 2 (softmax)*, and *Phase 3 (semi-greedy)*.

2.2 POLICY-GRADIENT TERM FOR BLACK-BOX REWARDS

Many useful scoring functions for binder design are not differentiable with respect to ϕ, including confidence scores produced by external structure predictors (Chai-1, AlphaFold3 Chai Discovery (2024); Abramson et al. (2024)), or experimental measurements. To enable optimization under such objectives, we add an *optional* policy-gradient update. At selected iterations, we sample K sequences $\mathbf{x}^{(k)} \sim p(\cdot \mid \phi)$, evaluate a black-box reward $R(\mathbf{x}^{(k)})$, and apply the REINFORCE Williams (1992) estimator:

$$\nabla_\phi \mathbb{E}_{\mathbf{x}\sim p(\cdot|\phi)}[R(\mathbf{x})] \;\approx\; \frac{1}{K}\sum_{k=1}^{K}(R(\mathbf{x}^{(k)}) - b)\,\nabla_\phi \log p(\mathbf{x}^{(k)} \mid \phi)\,,$$

where b is a baseline (we use a per-update batch mean) to reduce variance. In practice, we implement this by minimizing the surrogate loss

$$\mathcal{L}_{\text{RL}}(\phi) \;=\; -\frac{1}{K}\sum_{k=1}^{K}(\tilde{R}^{(k)} - b)\;\log p(\mathbf{x}^{(k)} \mid \phi)\,,$$

where $b = \frac{1}{K}\sum_k \tilde{R}^{(k)}$ is the per-update baseline. For the setting used in this paper, the scalar reward is clipped as

$$\tilde{R}^{(k)} = \text{clip}(R(\mathbf{x}^{(k)}); -c, c),$$

where the clipping threshold c is a tunable hyperparameter. When reward variance within an update is effectively zero ($\text{std}(R) \le \epsilon$, with ϵ a small constant), the implementation skips the RL update to avoid adding numerical noise. The total logit update combines the differentiable Germinal gradients with the RL term:

$$g_{\text{total}} \;=\; g_{\text{AF}} \;+\; g_{\text{IgLM}} \;+\; \lambda_{\text{RL}}\, g_{\text{RL}}\,,$$

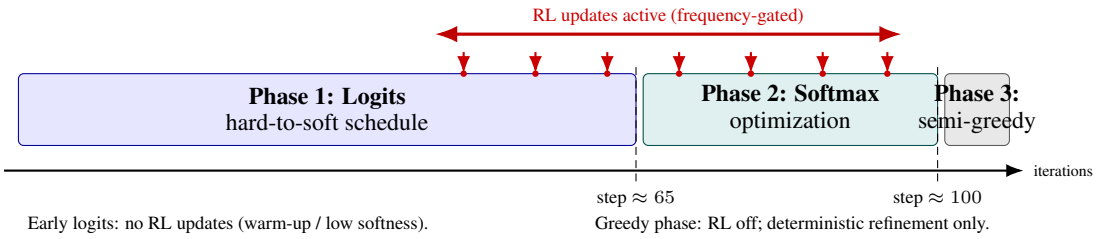

Figure 2: Temporal view of Germinal phases and where the optional RL gradient is injected. In the RALO configuration, RL is applied only in late logits and softmax steps, at frequency-gated update events. The RL gradient is added to the logit-gradient tensor and merged with standard Germinal gradients before each optimizer step.

with a tunable coefficient λ_{RL}. When $\lambda_{\text{RL}} = 0$ (or the feature flag is disabled), behavior is identical to the unmodified Germinal pipeline.

2.3 Reward Construction from Germinal Metrics

We define $R(\mathbf{x})$ as a weighted sum of modular reward components: $R(\mathbf{x}) = \sum_i w_i \, r_i(\mathbf{x})$. To maximize reuse and avoid backend-specific code paths, each r_i wraps an existing Germinal Mille-Fragoso et al. (2025) metric. In particular, we support: (i) *internal* metrics from Germinal (or optional per-sample AlphaFold Evans et al. (2022) reevaluation), and (ii) *external* metrics computed by an external predictor (Chai-1 or AlphaFold3 Chai Discovery (2024); Abramson et al. (2024)), using the same backend selected by the run configuration.

Margin Mode. To increase filter pass rates, we introduce a margin reward. Given a filter threshold τ and a metric $m(\mathbf{x})$, we define a signed margin:

$$r^{\text{margin}}(\mathbf{x}) = \begin{cases} m(\mathbf{x}) - \tau, & \text{for filters of the form } m(\mathbf{x}) \geq \tau, \\ \tau - m(\mathbf{x}), & \text{for filters of the form } m(\mathbf{x}) \leq \tau. \end{cases}$$

Positive margin indicates the constraint is satisfied; negative margin indicates violation. This simple transformation makes the RL term "push" designs toward passing the same constraints used downstream.

2.4 Scheduling: Apply RL When the Distribution Has Variance

Policy-gradient updates require reward variance under $p(\cdot \mid \phi)$. If the logit distribution collapses to (nearly) a point mass, or if sampling is deterministic, then the sampled rewards become (nearly) constant and the REINFORCE update becomes ineffective. We therefore introduce simple scheduling controls: (i) a warm-up step, (ii) a minimum "softness" threshold, (iii) an update frequency, and (iv) a sampling strategy. Figure 2 provides a temporal view of these controls across Germinal's three phases and highlights where RL updates are active. Here "softness" refers to Germinal's scalar `soft` knob that interpolates between a hard, nearly deterministic sequence representation and a fully soft categorical distribution over amino acids. In the experiments in Section 3, we restrict RL updates to the late steps of the logits phase and the softmax phase; this controls compute and helps avoid no-variance updates.

3 Experiments

We evaluate RALO as an *extension* of the Germinal pipeline Mille-Fragoso et al. (2025) on a nanobody binder design task against PD-L1. All runs use the same Germinal backbone, which attempts 50 trajectories and uses Germinal's standard three-phase schedule (logits → softmax → semi-greedy). We add a small RL term in the high-softness region of optimization and test its effect on final filter acceptance rates (see Appendix A.4 for filter details).Our baseline is the unmodified Germinal pipeline, which we compare to an RL-augmented configuration that uses sparse policy-gradient updates with rewards computed by Chai-1 Chai Discovery (2024).

RL-augmented. We enable the RL term in the *high-softness* region of optimization (when logits are in their raw and softmax phases) and gate RL contributions until after a warm-up period. We compute

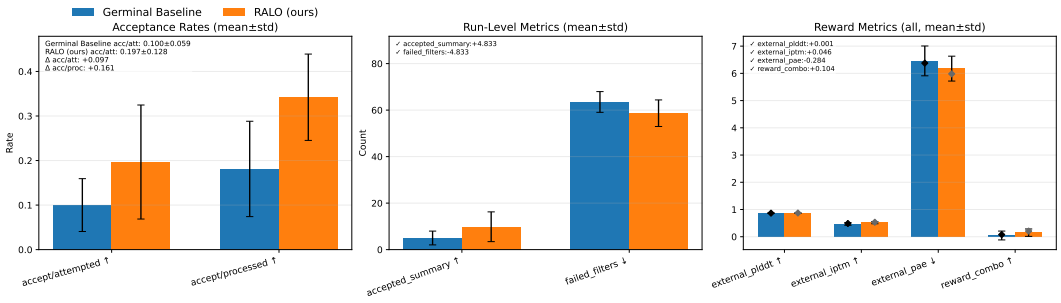

Figure 3: Baseline Germinal vs. RL-augmented Germinal (RALO) on PD-L1 nanobody design. Left: acceptance rates (mean±std across six seeds). Middle: counts of accepted and failed designs over entire runs (mean±std across six seeds). Right: accepted-set metrics (mean±std across seeds of per-run accepted-set means).

a margin reward as described above over the external confidence metrics already used by Germinal filters: pLDDT, ipTM, and PAE, all computed by Chai-1 with weights $w = \{+1, +1, -0.2\}$ and reward clipping $c = 2.0$. We enable sparse, small RL updates ($\lambda_{\text{RL}} = 0.02$) every 20 optimization steps. See Appendix A.3 for details.

Protocol and metrics. For each condition (baseline and RL), we run 6 independent seeds (6 separate executions). We report mean±std over seeds. We track two acceptance rates: (i) *accept/attempted*, the fraction of attempted trajectories that yield an accepted design, and (ii) *accept/processed*, the fraction of trajectories that reach the final evaluation stage that are accepted.

Results. Across the seeds, adding a small RL term in the high-softness region increases acceptance rates. The baseline achieves accept/attempted 0.100 ± 0.059 and accept/processed 0.181 ± 0.107. The RL-augmented configuration achieves accept/attempted 0.197 ± 0.128 and accept/processed 0.342 ± 0.097. The higher variance in accept/attempted results stems from the stochastic sampling performed by policy gradients; this variance tightens in accept/processed as RL contributions cease in the final optimization stage. These results suggest policy gradients enhance sequence space exploration, thus improving final performance. Figure 3 summarizes both acceptance rates, design counts, and reward metrics (Chai-1 confidence scores) to verify that improvements in acceptance rates are accompanied by improvements in the underlying reward metrics. The RL-augmented configuration achieves higher acceptance rates by improving the underlying reward metrics, which are directly related to the final filters.

4 Conclusion

We propose RALO, an extension to the Germinal protein design pipeline to enable optimization of important downstream signals ordinarily reserved for filtering of designs. We leverage policy gradients to optimize these objectives during the raw logits and softmax phases.

Our results show that integrating policy gradient methods for black-box optimization accelerates antibody design: across run seeds, RALO significantly increases the number of passing designs.

A limitation of our results is that our evaluation is limited to a single target (PD-L1) and nanobody scaffold. Future work should expand evaluation to diverse targets and scaffolds.

Future work will also explore greater algorithmic changes in more depth, study the effect of adding a policy gradient optimization step on generated designs, expand the *in silico* benchmark suite, and include wet lab validation of selected designs.

References

Josh Abramson, Jonas Adler, Jack Dunger, Richard Evans, Tim Green, Alexander Pritzel, Olaf Ronneberger, Lindsay Willmore, Andrew J. Ballard, Joshua Bambrick, Sebastian W. Bodenstein, David A. Evans, Chia-Chun Hung, Michael O'Neill, David Reiman, Kathryn Tunyasuvunakool, Zachary Wu, Akvilė Žemgulytė, Eirini Arvaniti, Charles Beattie, Ottavia Bertolli, Alex Bridgland, Alexey Cherepanov, Miles Congreve, Alexander I. Cowen-Rivers, Andrew

Cowie, Michael Figurnov, Fabian B. Fuchs, Hannah Gladman, Rishub Jain, Yousuf A. Khan, Caroline M. R. Low, Kuba Perlin, Anna Potapenko, Pascal Savy, Sukhdeep Singh, Adrian Stecula, Ashok Thillaisundaram, Catherine Tong, Sergei Yakneen, Ellen D. Zhong, Michal Zielinski, Augustin Žídek, Victor Bapst, Pushmeet Kohli, Max Jaderberg, Demis Hassabis, and John M. Jumper. Accurate structure prediction of biomolecular interactions with alphafold 3. *Nature*, 630(8016):493–500, 2024. doi: 10.1038/s41586-024-07487-w. URL https://doi.org/10.1038/s41586-024-07487-w.

N. Bennett, Joseph L. Watson, R. Ragotte, Andrew J. Borst, DéJenaé L. See, Connor Weidle, Riti Biswas, Yutong Yu, Ellen L. Shrock, Russell Ault, P. J. Leung, Buwei Huang, Inna Goreshnik, John Tam, Kenneth D. Carr, Benedikt Singer, Cameron Criswell, B. Wicky, Dionne K. Vafeados, Mariana Garcia Sanchez, Ho Min Kim, Susana Vázquez Torres, Sidney Chan, Shirley M. Sun, Timothy T Spear, Yi Sun, Keelan O'Reilly, John M. Maris, Nikolaos G. Sgourakis, Roman A. Melnyk, Chang C. Liu, and David Baker. Atomically accurate de novo design of antibodies with rfdiffusion. *bioRxiv*, 2024.

Chai Discovery. Chai-1: Decoding the molecular interactions of life. *bioRxiv*, 2024. doi: 10.1101/2024.10.10.615955. URL https://www.biorxiv.org/content/early/2024/10/11/2024.10.10.615955.

Felipe Leno da Silva, Andre Gonçalves, Sam Nguyen, Denis Vashchenko, R. Glatt, Thomas A. Desautels, Mikel Landajuela, Daniel M. Faissol, and Brenden K. Petersen. Language model-accelerated deep symbolic optimization. *Neural Computing and Applications*, 37:25601 – 25617, 2023.

Kerr Ding, Michael Chin, Yunlong Zhao, Wei Huang, Binh Khanh Mai, Huanan Wang, Peng Liu, Yang Yang, and Yunan Luo. Machine learning-guided co-optimization of fitness and diversity facilitates combinatorial library design in enzyme engineering. *Nature Communications*, 15, 2024.

Richard Evans, Michael O'Neill, Alexander Pritzel, Natasha Antropova, Andrew Senior, Tim Green, Augustin Žídek, Russ Bates, Sam Blackwell, Jason Yim, Olaf Ronneberger, Sebastian Bodenstein, Michal Zielinski, Alex Bridgland, Anna Potapenko, Andrew Cowie, Kathryn Tunyasuvunakool, Rishub Jain, Ellen Clancy, Pushmeet Kohli, John Jumper, and Demis Hassabis. Protein complex prediction with alphafold-multimer. *biorxiv*, 2022. doi: 10.1101/2021.10.04.463034.

Christopher Frank, Ali Khoshouei, Lara Fuß, Dominik Schiwietz, Dominik Putz, Lara Weber, Zhixuan Zhao, Motoyuki Hattori, Shihao Feng, Yosta de Stigter, Sergey Ovchinnikov, and Hendrik Dietz. Scalable protein design using optimization in a relaxed sequence space. *Science*, 386:439 – 445, 2024.

Nate Gruver, S. Stanton, Nathan C Frey, Tim G. J. Rudner, I. Hotzel, J. Lafrance-Vanasse, A. Rajpal, Kyunghyun Cho, and A. Wilson. Protein design with guided discrete diffusion. *ArXiv*, abs/2305.20009, 2023.

Chak Shing Lee, Conor F. Hayes, Denis Vashchenko, and Mikel Landajuela. Reinforcement learning for antibody sequence infilling. *bioRxiv*, 2025.

Luis S. Mille-Fragoso, John N. Wang, Claudia L. Driscoll, Haoyu Dai, Talal Widatalla, Xiaowei Zhang, Brian L. Hie, and Xiaojing J. Gao. Efficient generation of epitope-targeted de novo antibodies with germinal. *bioRxiv*, 2025.

Martin Pacesa, Lennart Nickel, Christian Schellhaas, Joseph Schmidt, Ekaterina Pyatova, Lucas Kissling, Patrick Barendse, Jagrity Choudhury, Srajan Kapoor, A. Alcaraz-Serna, Yehlin Cho, Kourosh H. Ghamary, Laura Vinué, B. Yachnin, Andrew M. Wollacott, Stephen Buckley, A. Westphal, S. Lindhoud, S. Georgeon, Casper A. Goverde, G. Hatzopoulos, P. Gönczy, Yannick D. Muller, Gerald Schwank, D. C. Swarts, A. Vecchio, Bernard L. Schneider, Sergey Ovchinnikov, and Bruno E. Correia. One-shot design of functional protein binders with bindcraft. *Nature*, 646: 483 – 492, 2025.

Richard W. Shuai, Jeffrey A. Ruffolo, and Jeffrey J. Gray. Iglm: Infilling language modeling for antibody sequence design. *Cell systems*, 2023.

Joseph L. Watson, David Juergens, Nathaniel R. Bennett, Brian L. Trippe, Jason Yim, et al. De novo design of protein structure and function with rfdiffusion. *Nature*, 620(7976):1089–1100, 2023. doi: 10.1038/s41586-023-06415-8.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3):229–256, 1992. doi: 10.1007/BF00992696. URL https://doi.org/10.1007/BF00992696.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning, 2020. URL https://arxiv.org/abs/2001.06782.

Wensi Zhu, Aditi Shenoy, Petras Kundrotas, and Arne Elofsson. Evaluation of alphafold-multimer prediction on multi-chain protein complexes. *Bioinformatics*, 39(7):btad424, 07 2023. ISSN 1367-4811. doi: 10.1093/bioinformatics/btad424. URL https://doi.org/10.1093/bioinformatics/btad424.

A Appendix

A.1 Related Work

Logit representations for protein sequence design. Modern hallucination pipelines Frank et al. (2024); Pacesa et al. (2025); Mille-Fragoso et al. (2025) often optimize continuous logits $\phi \in \mathbb{R}^{N\times M}$ that induce a factorized categorical policy over sequences:

$$p_{i,j} = \frac{\exp(\phi_{i,j})}{\sum_{k=1}^{M}\exp(\phi_{i,k})}, \qquad p(\mathbf{x} \mid \phi) = \prod_{i=1}^{N} p_{i,x_i}. \tag{1}$$

Equation (1) is the common parameterization used in logit-based sequence optimization and underlies both differentiable hallucination updates and policy-gradient updates.

Policy gradients optimize via the score-function estimator:

$$\nabla_\phi J = \mathbb{E}_{\mathbf{x}\sim p(\cdot|\phi)[(R\mathbf{x}-b)\nabla_\phi \log(p\mathbf{x}|\phi)]}$$

enabling optimization of black-box objectives.

RL objective and policy-gradient approaches. When design quality is measured by a black-box reward $R(\mathbf{x})$, the objective and its score-function gradient can be written jointly as

$$J(\phi) = \mathbb{E}_{x\sim p(\cdot|\phi)}[R(\mathbf{x})], \qquad \nabla_\phi J = \mathbb{E}_{\mathbf{x}\sim p(\cdot|\phi)}[R(\mathbf{x}) - b\nabla_\phi \log p(\mathbf{x} \mid \phi)] \tag{2}$$

with baseline b for variance reduction. RL-style sequence optimization has been used for difficult, multi-objective fitness landscapes and diversity-constrained design Ding et al. (2024); da Silva et al. (2023); Lee et al. (2025).

Hallucination trick: moving from expectation to deterministic surrogate. Hallucination-style design commonly replaces the stochastic expected objective Eq. (2) with a differentiable objective evaluated on soft sequence representations. Let R denote a sequence-scoring functional and let p_ϕ be induced by logits in Eq. (1). The usual approximation is

$$J(\phi) = \mathbb{E}_{\mathbf{x}\sim p(\cdot|\phi)}[R(\mathbf{x})] \;\approx\; R\big(\mathbb{E}_{\mathbf{x}\sim p(\cdot|\phi)}[\mathbf{x}]\big) \;\approx\; \hat{R}(p_\phi), \qquad \nabla_\phi J \approx \nabla_\phi \hat{R}(p_\phi). \tag{3}$$

where $\hat{R}$ is differentiable with respect to soft marginals. This is accurate when distributional variance is sufficiently small or R is locally well-approximated around the mean representation, which is exactly the regime targeted by entropy reduction / sharpened logits in many hallucination pipelines Frank et al. (2024); Pacesa et al. (2025); Mille-Fragoso et al. (2025).

Why combine both. The two paradigms are complementary: Eq. (3) gives low-variance, efficient gradients for differentiable objectives, while Eq. (2) includes the score-function term that enables optimization of non-differentiable black-box rewards Ding et al. (2024); Frank et al. (2024). We refer to this hybrid as **RALO** (*RL-Augmented Logit Optimization*): it retains the hallucination backbone and injects an optional RL gradient term only in selected optimization regions, yielding a practical method that preserves baseline behavior when RL is disabled.

A.2 Pseudo-code for RL-augmented Germinal Step

A.3 Policy gradient hyperparameters

For policy gradient optimization, we keep the RL term small ($\lambda_{\text{RL}} = 0.02$) and sparse (every 20 optimization steps), and we only start RL updates after the optimization has warmed up and the distribution is sufficiently soft. Our default is to allow a 30-gradient-step warm-up period and require distribution softness of 0.8.

A.4 Relaxed Filter Set

For our experiments, we use a relaxed set of Germinal's original filters.

A.5 Reward function weights

Here we list the weights placed on the reward function derived from Chai-1 outputs.

We also apply reward clipping with $c = 2.0$.

Algorithm 1 RL-augmented Germinal step

Require: current logits ϕ_t, Germinal gradient tensor g_t (`aux["grad"]["seq"]`), step index t, softness s_t
Require: design mode $m_t \in \{\texttt{logits}, \texttt{soft}, \texttt{greedy}\}$
Require: RALO hyperparameters: $\lambda_{\mathrm{RL}} = 0.02$, $K = 10$, frequency $F = 20$, start step $t_0 = 30$, minimum softness $s_{\min} = 0.8$

if RL disabled **then**
  Use Germinal update only
  return
end if
if $m_t \notin \{\texttt{logits}, \texttt{soft}\}$ **or** $t < t_0$ **or** $s_t < s_{\min}$ **or** $t \bmod F \neq 0$ **then**
  Skip RL update for this step
else
  Sample K sequences $\mathbf{x}^{(k)} \sim p(\cdot \mid \phi_t)$ with stochastic categorical sampling
  for $k = 1, \ldots, K$ **do**
    Run Chai-1 prediction for $\mathbf{x}^{(k)}$
    Extract $\{m^{(k)}_{\text{plddt}}, m^{(k)}_{\text{iptm}}, m^{(k)}_{\text{pae}}\}$
    Convert each metric to margin against its corresponding final-filter threshold
    $R^{(k)} \leftarrow 1.0 \cdot \Delta^{(k)}_{\text{plddt}} + 1.0 \cdot \Delta^{(k)}_{\text{iptm}} - 0.2 \cdot \Delta^{(k)}_{\text{pae}}$
    $R^{(k)} \leftarrow \mathrm{clip}(R^{(k)}, -2.0, 2.0)$
  end for
  if $\mathrm{std}(\{R^{(k)}\}_{k=1}^{K}) \leq \epsilon$ **then**
    Skip RL update (`rl_skip_no_variance=true`)
  else
    $b \leftarrow \frac{1}{K} \sum_{k=1}^{K} R^{(k)}$
    $g_{\mathrm{RL}} \leftarrow \frac{1}{K} \sum_{k=1}^{K} (R^{(k)} - b) \nabla_\phi \log p(\mathbf{x}^{(k)} \mid \phi_t)$
    $g_t \leftarrow g_t + \lambda_{\mathrm{RL}}\, g_{\mathrm{RL}}$
  end if
end if
Apply the standard Germinal optimizer step using updated g_t

Parameter	Value
RL enabled	`enable_rl_structural_term=true`
Reward backend	Chai-1 (`structure_model=chai`)
Use external prediction	`rl_use_structure_prediction=true`
Reward components	`[external_plddt, external_iptm, external_pae]`
Reward weights	`{external_plddt:1, external_iptm:1, external_pae:-0.2}`
Reward mode	`margin` (relative to filter thresholds)
Reward clip	`rl_reward_clip=2.0`
Allow soft-mode updates	`rl_apply_in_soft=true`
RL coefficient	$\lambda_{\mathrm{RL}} = 0.02$
Update frequency	`rl_update_frequency=20`
Samples per update	`rl_num_samples=10`
Sampling strategy	`stochastic`
Warm-up	`rl_start_step=30`
Min softness	`rl_min_soft=0.8`

Table 1: RL configuration used for the PD-L1 Germinal extension experiments.

Filter	Condition
Clashes	< 2
self-consistency RMSD (scRMSD)	< 8.0
Binder near hotspot	True
CDR3 Hotspot contacts	> 0
Percent Interface is CDR	$> 40\%$
Interface shape comlpementarity	≥ 0.5
Interface hydrogen bonds	> 2
Surface hydrophobicity	≤ 0.5
Interface hydrophobicity	≥ 40
pDockQ2 Zhu et al. (2023)	> 0.15
External pLDDT	> 0.82
External ipTM	> 0.60
External pTM	> 0.75
External PAE	< 12.0

Table 2: Filter set used for our experiments.

Metric	Weight
pLDDT	+1
ipTM	+1
PAE	-0.2

Table 3: Weighting terms placed on metrics used in the reward function.