
000 001 002 003 004 005 006 007 008 009 010 011 012 013 014 015 016 017 018 019 020 021 022 023 024 025 026 027 028 029 030 031 032 033 034 035 036 037 038 039 040 041 042 043 044 045 046 047 048 049 050 051 052 053 054 Emergent Biological Realism in RL-Trained DNA Language Models

Anonymous Authors¹

Abstract

Reinforcement learning post-training has driven the mass adoption of large language models by unlocking unexpected emergent capabilities, yet this approach remains largely unexplored for generative DNA models. We investigate whether similar post-training techniques can induce emergent biological realism in DNA language models, using plasmid generation as a testbed due to plasmids' relative simplicity, well-characterized functional constraints, and ubiquity in biotechnology. Using Group Relative Policy Optimization with a reward function based on synthetic biological constraints, our model achieves 97% quality control pass rate compared to 6% for the pretrained baseline. Remarkably, beyond explicitly optimized features, the model exhibits emergent biological parallels: generated sequences match natural plasmids in thermodynamic stability, codon usage patterns, and ORF length distributions, properties not directly encoded in the reward function. These results suggest that RL post-training can steer DNA language models toward biologically coherent regions of sequence space, analogous to how such techniques unlock emergent capabilities in natural language models.

1. Introduction

Plasmids are extrachromosomal DNA sequences, often found in bacteria, capable of replication independent of a host genome (Lederberg, 1952). These genetic elements are ubiquitous in biotechnology, serving as the primary vectors for protein expression, gene editing, and emerging DNA therapeutics (Kutzler & Weiner, 2008; Prather et al., 2003). Despite their widespread utility, plasmid engineering remains a complex, high-dimensional optimization problem. Traditional workflows are cost intensive and heuristic

driven, often requiring iterative cycles of manual sequence editing and experimental validation to resolve structural instabilities (Oliveira et al., 2009; Meng & Ellis, 2015). Suboptimal plasmid architectures, plagued by incompatible regulatory elements or unstable repeat regions, frequently lead to metabolic burden, reduced expression efficiency, and manufacturing bottlenecks (Wu et al., 2016; Brophy & Voigt, 2011).

Current approaches to plasmid design rely heavily on tacit domain knowledge and piecemeal assembly of genetic parts. Designers must simultaneously optimize for competing objectives such as copy number, transcriptional output, and host viability while navigating the strict biophysical constraints of DNA folding and context dependent regulatory interactions (Deng et al., 2025; Fung et al., 2025).

The dramatic success of reinforcement learning post-training in natural language processing—unlocking emergent reasoning, instruction following, and unexpected generalization—raises a compelling question: can similar techniques induce emergent biological realism in DNA language models? We investigate this using plasmid generation as a testbed, applying Group Relative Policy Optimization to the PlasmidGPT foundation model (Shao et al., 2024a). Beyond dramatically improving quality control pass rates (97% vs. 6% baseline), we observe emergent properties not explicitly optimized: generated sequences match natural plasmids in thermodynamic stability, codon usage patterns, and ORF length distributions. These results suggest that RL post-training may unlock similar emergent capabilities in genomic models as it has in language models, steering generation toward biologically coherent sequence space regions through reward-guided optimization.

2. Background

2.1. DNA Language Models

Natural language processing and genomics have developed in parallel, often with algorithms developed for bioinformatics used in NLP and vice versa (Durbin et al., 1998). Natural language models pretrained on massive amounts of data have displayed emergent capabilities and remarkable utility by understanding the structure of language as a whole (Wei et al., 2022). These capabilities have vastly

¹Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

increased the utility of language models in a wide range of tasks, and are one of the primary reasons why large language model adoption has skyrocketed in the last few years. This has inspired the application of many similar techniques to genomic language models, resulting in models capable of impressive performance on biological tasks including variant prediction (Ji et al., 2021), transcription factor binding site identification (Nguyen et al., 2023), and even whole-genome generation (Nguyen et al., 2024).

Plasmid DNA has received relatively little attention compared to other sequence types despite its importance in biomanufacturing and wet lab research. OriGen (Martinson et al., 2025) introduces a generative model to produce previously undiscovered origins of replication (ORIs) but does not model whole sequences. PlasmidGPT (Shao et al., 2024a) uses modern language modeling techniques to develop a generative model for whole plasmid sequences, and later work expands on this by synthesizing whole plasmids generated using a fine tuned version of the PlasmidGPT model (Cunningham et al., 2025). We use this as a base model and apply post-training techniques to improve results significantly.

2.2. Plasmid Design

Lab-designed plasmids are short circular DNA molecules (typically 2–15 kb) that must contain multiple functional components arranged in precise configurations. At minimum, a viable plasmid requires: (i) an origin of replication to enable autonomous replication, (ii) a selection marker (e.g., antibiotic resistance gene) for identifying successfully transformed cells, and (iii) a cloning site where genes of interest can be inserted.

The search space of valid plasmids is massive due to combinatorial explosion across components. There are many types of each component on the scale of hundreds to thousands per component, and additional regulatory elements (promoters, terminators, enhancers) and reporters may be required depending on the application. Multiple instances of certain components may be necessary, and the ordering and spacing of these elements significantly impacts function. Beyond sheer combinatorics, designers must navigate complex biological constraints including compatibility requirements (specific ORIs only function in certain hosts), physical stability issues (repeat regions can fold and bind to each other), and other design challenges.

3. Methods

We follow an established method that starts with a base model, fine-tunes it using supervised fine-tuning (SFT), and then applies reinforcement learning (RL) to optimize for specific attributes in the output.

3.1. Supervised Fine-Tuning

Supervised fine-tuning (SFT) was performed on a curated corpus of *E. coli* plasmid sequences assembled from PlasmidScope and Addgene. After deduplication and quality filtering, approximately 15k circular plasmids (≤ 30 kb) were retained, excluding linear entries, fragments, and incomplete records. Sequences were tokenized using the original PlasmidGPT byte-pair DNA tokenizer. The pretrained PlasmidGPT model was fine-tuned using an autoregressive next-token prediction objective with gradient accumulation and learning-rate warmup over three epochs.

3.2. Reinforcement Learning with GRPO

We implement a configurable reinforcement learning pipeline for plasmid design that uses Group Relative Policy Optimization (GRPO) (Shao et al., 2024b) with a domain-specific reward function described below. At each training iteration, the model generates a batch of candidate plasmids via autoregressive rollouts conditioned on short nucleotide prompts. Prompts are either stochastic (4–25 bp random seeds, excluding "ATG", used to promote rollout diversity) or structured (partial "cassette" seeds encoding canonical marker genes such as antibiotic-resistance or fluorescent reporters). Each candidate sequence is evaluated via our reward function, which captures structural plausibility, cassette organization, repeat content, and other biologically motivated constraints. GRPO is then applied to update the model parameters using these sequence-level rewards, enabling the policy to progressively shift toward generating plasmids with higher predicted validity.

3.3. Reward Function Design

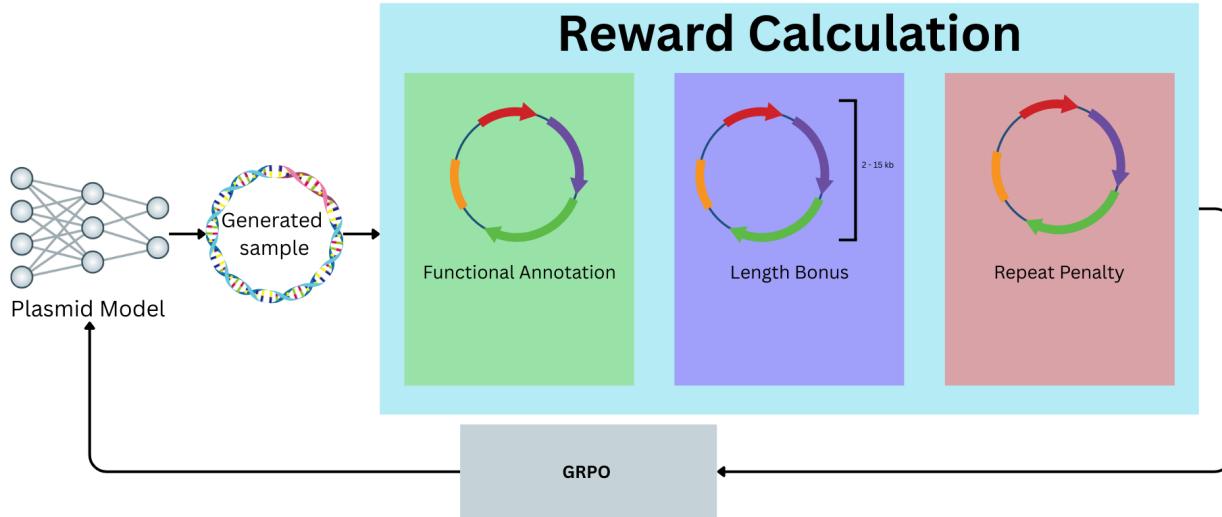
The reward function scores each generated plasmid according to its structural plausibility and expected stability. It is composed of three conceptual components:

Functional annotation scoring: Lightweight annotations identify origins of replication, promoters, terminators, coding sequences (CDS), and selectable markers, which are then scored according to a configuration designed with subject matter expert (SME) input to reflect biologically reasonable quantities (e.g., exactly one origin of replication and at least one selectable marker). To encourage coherent gene cassettes, this component also includes a location-aware bonus for promoter → CDS → terminator arrangements that appear in the correct order and within a reasonable proximity window.

Length prior: A length prior favors plasmid sizes within typical experimental ranges (5–15 kb) preferred for plasmid construction.

Repeat penalty: A repeat penalty down-weights sequences

110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135



136 *Figure 1.* Overview of the Plasmid-RL training pipeline. Starting from the pretrained PlasmidGPT base model, we apply supervised
137 fine-tuning on curated plasmid sequences, followed by reinforcement learning with Group Relative Policy Optimization using a
138 biologically-motivated reward function that evaluates functional annotations, length constraints, and repeat content.

139 containing long exact repeats that are associated with instability or recombination, specifically penalizing .1 reward for
140 each repeat of length 50 bp or greater.
141

142 These terms are combined into a single scalar in [0, 1], yielding a fast and interpretable proxy for “plasmid-likeness”
143 during reinforcement learning.
144

4. Experiments

4.1. Plasmid Quality Control and Uniqueness

151 We evaluate three model variants on held-out prompts not
152 seen during training. For each model, we sampled 50 roll-
153 outs with two prompts: (i) a minimal prompt (single ATG
154 codon) to test unconditional generation capability, and (ii)
155 a structured prompt containing a complete GFP expression
156 cassette to test the model’s ability to build around provided
157 components. These prompts were deliberately excluded
158 from the training corpus to ensure evaluation reflects gener-
159 alization rather than memorization.
160

4.1.1. VALIDITY ASSESSMENT

161 In silico plasmid validity was assessed using a bioinformat-
162 ics quality-control pipeline that leverages BLAST based
163

164 tools, requiring exactly one origin of replication ($\geq 95\%$ identity and coverage), one or two antimicrobial resistance genes ($\geq 99\%$ identity and coverage), and no internal repeats longer than 50 bp (Altschul et al., 1990). This pipeline has been validated as a reliable proxy for experimental synthesis success (Cunningham et al., 2025). While we do not perform wet-lab validation in this work, our focus is on establishing that RL post-training can successfully navigate the plasmid design space in silico, laying groundwork for future conditional generation systems where user-specified designs can be experimentally validated.

4.1.2. UNIQUENESS ASSESSMENT

To assess whether generated plasmids represent genuinely new designs rather than minor variants of existing constructs, we compute similarity to known sequences using the NCBI BLASTn API. Each generated plasmid is assigned to one of three categories based on identity and query-coverage thresholds: sequences with $\geq 99\%$ identity and $\geq 95\%$ coverage are classified as **Exists**; those with $\geq 95\%$ identity and $\geq 80\%$ coverage are **Similar**; and all others are classified as **Novel**.

This categorization is based on large scale data curation

165
166 **Table 1.** Novelty and diversity metrics across model variants. RL
167 achieves dramatically higher quality (97% pass rate) with reduced
168 diversity in samples.
169

MODEL	QC PASS RATE	% NOVEL	DIVERSITY
BASE	6%	95.5%	0.926
SFT	11%	100%	0.886
RL	97%	88%	0.391

175 efforts such as PLSDB that use these thresholds of similarity
176 to attempt to de-duplicate plasmids for not adding any
177 additional value to a dataset (Galata et al., 2018).
178

179 4.1.3. DIVERSITY ASSESSMENT

180 To detect and prevent model collapse, we attempt to measure
181 the diversity of the many samples of the model from the
182 same prompt. Due to the lack of utility of traditional NLP
183 metrics on this task, we use the mean Pairwise Jaccard
184 distance of the 21-mers of each sequence. Diversity of a
185 group of rollouts is calculated as follows:
186

$$187 D = 1 - \frac{1}{\binom{n}{2}} \sum_{i=1}^n \sum_{j=i+1}^n J(S_i, S_j)$$

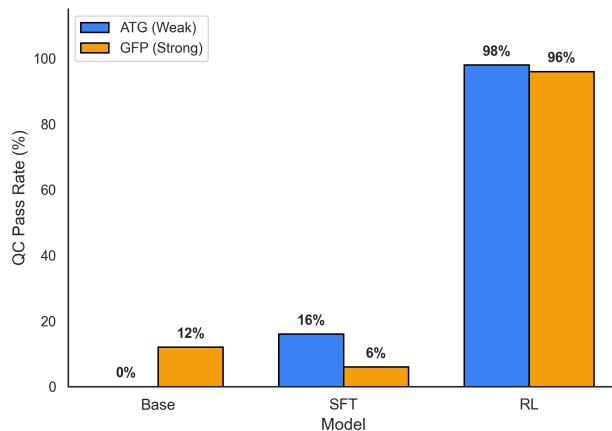
192 where $J(S_i, S_j)$ is the Jaccard similarity between MinHash
193 sketches of sequences i and j , and n is the number of se-
194 quences in the group.

195 The diversity metric (pairwise Jaccard distance) serves pri-
196 marily as a model collapse detector rather than a biolog-
197 ical validity measure. The RL model achieves 0.391 diver-
198 sity compared to 0.926 for the base model, indicating that
199 while the RL model concentrates probability mass on higher-
200 quality regions of sequence space, it does not collapse to
201 identical outputs. This is consistent with RL optimization
202 finding conserved “successful motifs” (e.g., proven origins
203 of replication, reliable resistance markers) while still main-
204 taining sequence-level uniqueness.
205

206 4.1.4. RESULTS

207 Reinforcement learning substantially increases the proba-
208 bility of generating plasmids that pass our bioinformatics
209 quality control (QC) pipeline, while supervised fine-tuning
210 provides modest improvements. Figure 2 shows pass rates
211 by prompt type, and Table 1 summarizes novelty and diver-
212 sity metrics across models.
213

214 When prompted with the weak ATG prompt, the base model
215 never produces a valid plasmid (0%), whereas SFT increases
216 the pass rate to 16%, and RL further increases it to 96%. A
217 similar trend holds with the stronger GFP-cassette prompt:
218 the base model achieves 12%, SFT drops to 4%, but the
219



220 **Figure 2.** Summary of QC outcomes by prompt. Base model is
221 only able to generate functional plasmids with a strong prompt.
222 SFT allows the model to overcome this limitation on occasion but
223 still often fails QC. Adding RL to the training process improves
224 pass rate dramatically.

225 RL-optimized model reaches 76%.

226 Aggregated across strong and weak prompts, the overall
227 QC pass rate rises from 6% with the base model to 11%
228 with the SFT model and finally to 97% with the RL model,
229 representing more than an order-of-magnitude improvement
230 in validity relative to the pretrained baseline.

231 Importantly, this increase does not come from the RL mod-
232 els repeating previously known, high scoring sequences.
233 Among passing sequences, the proportion classified as novel
234 remains substantial for all models. Using our novelty thresh-
235 olds, the RL model produces 88% novel plasmids among
236 its QC-passing samples, compared to 95.5% for the base
237 model. The SFT model generated 100% novel sequences.
238 When normalized over all 100 rollouts per model, the RL
239 model produces 85.4% sequences that are both QC-valid
240 and novel, compared to 11% for SFT and 6% for the base
241 model.

242 Taken together, these results show that RL post-training not
243 only dramatically improves biological plausibility but also
244 preserves meaningful novelty relative to published plasmids.
245 The model learns to satisfy constraints without collapsing to
246 memorized or trivial constructs, suggesting that sequence-
247 level RL can push generation toward realistic design regions
248 while still exploring new areas of plasmid space.

249 4.2. Distribution Comparison

250 We compute several statistics known to be relevant to the
251 performance of DNA from the raw sequences of both the
252 generated plasmids and a small subset of real plasmids used

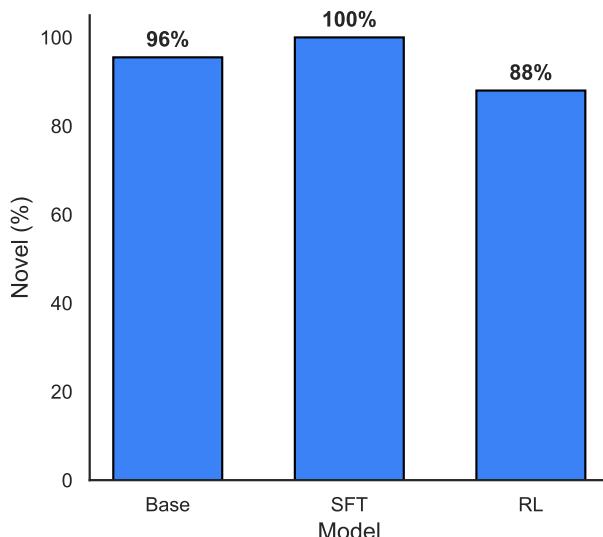


Figure 3. Summary of plasmid novelty as measured by comparison to NCBI database. SFT improves novelty of generated sequences from both base model and RL post-trained model.

for protein expression, genome editing, and other applications. See full details in the appendix. The distribution of the generated samples to each other and the real samples. We calculate the following summary statistics: sequence length distribution, GC content, longest open reading frame (ORF; calculated two ways), Jensen-Shannon divergence of the codon distribution, and Gibbs free energy (as calculated by ViennaRNA (Lorenz et al., 2011)).

Figure 4 shows that the RL post-trained model’s samples much more closely match the distributions of the real plasmids than the pretrained and supervised models, even when the metric is not directly encoded by the reward function.

Sequence length is directly encoded by the reward function, with optimal reward determined by a parameter sweep for training stability. GC content is not directly encoded, but regions selected for by the reward function likely have GC content distributed similarly to real plasmids. Despite generally making up a smaller percentage of the whole sequence, GC content is partially encoded by the reward function, as regions with higher GC content are more likely to be rewarded.

ORF length and codon distribution are not factored into the reward function directly. We calculate ORF length two ways: (1) maximum length of codons that are not stop codons on a single strand, and (2) longest stretch of non-stop codons after the presence of a start codon on either strand. These methods are disjoint from the method used to account for ORF in the reward function, which uses Prodigal (Hyatt

Table 2. Average log-probability on held-out continuation task. Higher values indicate better next-token prediction. RL shows unexpected improvement despite not being optimized for this task.

MODEL	MEAN LOG-PROB	STD DEV
BASE	-12.449	6.144
RL	-11.148	2.977

et al., 2010) to predict and reward correctly placed ORFs. The ORF length, measured by either method, converges closely to the distribution of the real plasmids.

The same pattern holds for codon distribution and Gibbs free energy. While not accounted for in the reward function, the RL model learns to generate tokens with a much more similar codon distribution to real plasmids than the two models that have seen the correct distribution in the training data. The similarity in the distribution of free energy measurements is particularly remarkable given that not only is free energy not optimized for directly, but no structural components are factored in at all excluding the weakly correlated repeat penalty originally included to solve recombination issues.

4.3. Held-Out Continuation

To evaluate how RL training affects nucleotide-level predictive performance, we measure how well each model can predict future bases given a real plasmid prefix. For each sequence, we provide the first 400 nucleotides as a prompt, have each model produce the next 100 nucleotides, and then compute the average log-probability of the true next 100 bases under each model. This allows us to compare the Base and RL models against real plasmids in terms of next-token prediction accuracy.

The RL model shows improved log-probability compared to the base model, with the standard deviation shrinking substantially from 6.144 to 2.977. This improvement is unexpected, as RL generally makes language models worse at next-token prediction tasks, a phenomenon known as the “alignment tax” (Lin et al., 2023).

4.4. Coding Sequence Surprisal Analysis

The reward function evaluates CDS regions using Prodigal rather than bioinformatics token-level approaches, reducing the risk of token-level information leakage. Because Prodigal predicts coding regions based only on generic statistical patterns and not on specific plasmid sequences, the reward signal is biologically grounded but distribution-agnostic. Interestingly, the RL-trained model achieves lower surprisal on real plasmids than the pretrained model, suggesting that the Prodigal-based reward sharpens the model’s understand-

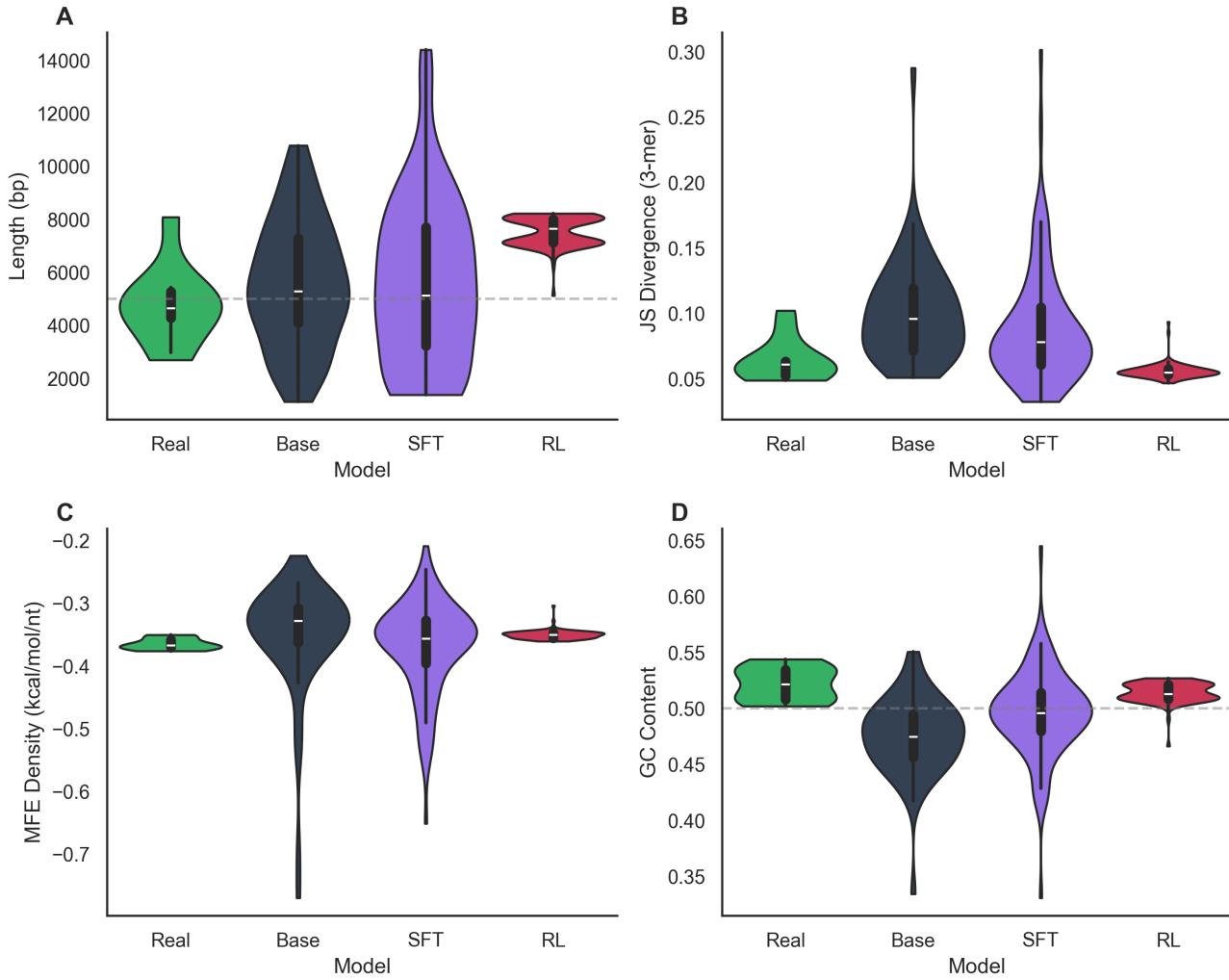


Figure 4. Distribution comparison across key biophysical metrics. The RL post-trained model (yellow) closely matches real plasmid distributions (blue) across sequence length, GC content, ORF length, codon usage (Jensen-Shannon divergence), and thermodynamic stability (Gibbs free energy), while the base model (green) and SFT model (orange) show substantial deviations. Notably, RL optimization produces realistic distributions even for metrics not explicitly encoded in the reward function.

ing of natural plasmid structure.

5. Discussion

5.1. Diversity Trade-offs in Post-Training

One caveat to note is the decreased diversity of the post-trained model. This is a known effect of post-training but is especially noticeable in the domain of DNA sequence generation. DNA in the context of this experiment can be thought of in two primary ways: (1) DNA that codes for a known region such as a promoter or an ORI, or (2) spacer DNA that exists primarily to contribute to structural stability.

While we observe a high degree of diversity in the sequences, there are some highly conserved regions that the post-trained model relies on to make the plasmid viable even more so than the base model. Conserved regions are necessary and common in nature, but the post-training process reduces the frequency with which some of the less common regions are used by the model. To make this concrete, when we sample the base model and the RL model 500 times each with the weak ATG prompt and annotate all outputs, we see that the base model uses 10 unique ORIs while the RL model only uses 7.

This might explain some of the distribution patterns in Section 4.2, as the model finds a less diverse set of policies that

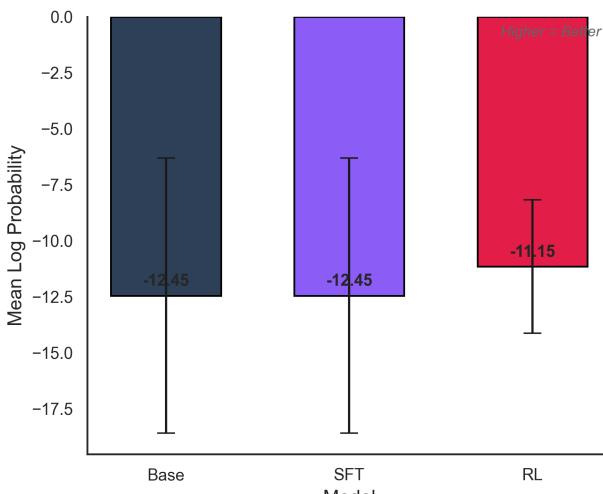


Figure 5. Held-out continuation performance across models. The RL model shows improved log-probability on real plasmid sequences, demonstrating better alignment with natural plasmid structure despite not being explicitly optimized for next-token prediction.

work and explores less, leading to the shrunken variance and a more similar mean to the real plasmids. However, the experiments in Sections 4.3 and 4.4 suggest that the policy learned by the RL process is better aligned with natural plasmids, even on tasks involving next-token prediction and surprisal.

5.2. Limitations of Bioinformatic Evaluation

Our training and evaluation process is based almost entirely on bioinformatics, which depends on having a library of known regions. These libraries are naturally incomplete. Therefore, if our model ever generates a sequence that would work as an ORI, for example, but that sequence is missing from the library either by omission or because it has never been observed in nature before, the model will not receive a reward. This sharply limits how creative the model can be, and contrasts with other known examples of RL working well in domains such as natural language processing or protein design, where preference reward models or biophysical models are used, respectively.

5.3. Unexpected Improvement on Token-Level Tasks

Reinforcement learning typically makes language models worse on token-level tasks. This phenomenon, known as the “alignment tax,” makes sense intuitively: the model is optimized toward a policy to maximize a reward function and therefore away from the next-token prediction policy. Our model shows an interesting reversal of this trend.

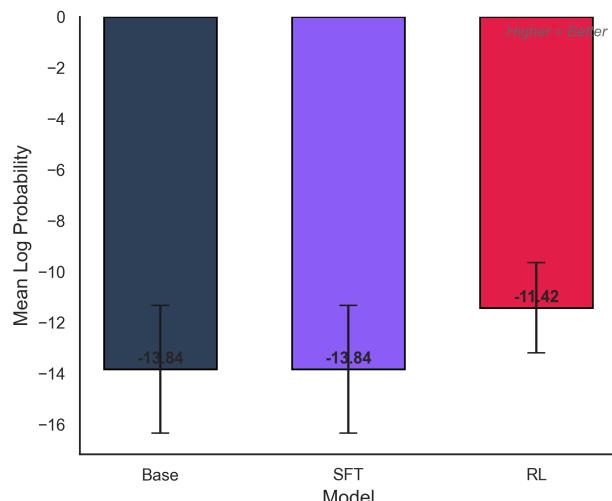


Figure 6. Coding sequence surprisal analysis. The RL model achieves lower surprisal on real plasmid coding sequences compared to the base and SFT models, indicating better capture of natural coding patterns despite using only Prodigal-based structural rewards.

One possible explanation uses the “steering vector” hypothesis as a way to understand how reinforcement learning interacts with the base model (Zou et al., 2023). This hypothesis states that the post-training process doesn’t give the model any new skills (more easily proven in NLP) but simply moves the latent space that the model decodes from to a more favorable region.

This offers a potential explanation as to why the reward function used was able to not only optimize plasmid validity as measured in silico (which is highly correlated with the reward function) but also optimize for thermodynamic stability and other characteristics that should be weakly correlated with the reward function. The RL process may have pushed the region of latent space toward a region of more valid plasmids in several aspects by being optimized for these bioinformatic features.

5.4. Emergent Properties and Evolutionary Analogies

The emergence of biologically realistic properties, reflexing those of real plasmids, while explicitly optimized mirros an evolutionary processes in that produce complex, correlated traits as byproducts of selecting for primary fitness criteria. Just as evolution optimizes for survival and reproduction while producing emergent phenomena like protein folding patterns and metabolic network architecture, our RL process optimizes for functional annotations while inducing realistic thermodynamic and sequence composition properties. This suggests that appropriately structured reward functions may

385 capture sufficient biological constraints to guide models
386 toward broadly realistic sequence space regions.
387

388 **5.5. Future Work: Conditional Generation**

389 This work further supports the ongoing hypothesis that modern
390 language modeling techniques can be applied to DNA
391 language models in a similar fashion to how they have been
392 applied to natural language in the past. The real utility of
393 natural language models came from instruction tuning them
394 to respond to questions and follow directions. This work
395 shows that post-training techniques can be applied to DNA
396 language models to achieve similar results.
397

398 We hope to build out a dataset designed for conditional
399 generation where the user can prompt the model with the
400 specifics of the plasmid they want, and the model will de-
401 velop it from there. This would enable practical use cases
402 such as “design a plasmid for expressing protein X in E. coli
403 with high copy number” or “create a mammalian expression
404 vector with constitutive GFP expression.”

405 In addition to being far more practical than unconditional
406 generation, the conditioning will promote more diversity of
407 sample and make evaluation of the model’s performance on
408 the task of plasmid design much more understandable.

410 **6. Conclusion**

411 We demonstrate that reinforcement learning post-training,
412 the technique that unlocked emergent capabilities in large
413 language models, can induce analogous emergent biolog-
414 ical realism in DNA language models. Our model gener-
415 ates structurally valid plasmids at a 97% rate (compared
416 to 6% for the base model), but more significantly, exhibits
417 emergent properties not directly encoded in the reward func-
418 tion: realistic thermodynamic stability, natural codon usage
419 patterns, and appropriate ORF length distributions. These
420 parallels to natural plasmids emerge from optimizing only
421 functional annotations, length constraints, and repeat pen-
422 alties.

423 This finding suggests that RL post-training operates simi-
424 larly across modalities—steering models toward coherent,
425 naturalistic regions of their respective sequence spaces. Just
426 as RL unlocked reasoning and generalization in language
427 models beyond their pretraining objectives, it guides DNA
428 models toward biologically realistic sequences beyond their
429 explicit rewards. The unexpected improvement in next-
430 token prediction further supports this interpretation: the
431 model learns general principles of plasmid structure rather
432 than memorizing specific solutions.

433 These findings open several directions for future work. First,
434 extending to conditional generation (e.g., “express protein
435 X in E. coli with high copy number”) would enable prac-

436 tical applications and experimental validation of model-
437 generated sequences. Second, applying RL post-training
438 to other genomic sequence classes (promoters, regulatory
439 elements, entire genomes) could test whether emergent bio-
440 logical realism generalizes across domains. Finally, inves-
441 tigating what reward function properties induce such
442 emergence could provide principles for designing effective
443 biological reward functions—potentially accelerating the
444 broader application of RL to computational biology.

Impact Statement

This paper presents work whose goal is to advance the field of computational biology and machine learning for biological sequence design. The ability to generate novel, valid plasmid sequences could accelerate research in synthetic biology, biomanufacturing, and therapeutic development. While the immediate applications are primarily beneficial for scientific research, we acknowledge the dual-use nature of synthetic biology tools. The methods described here are limited to generating plasmid sequences based on patterns in existing databases and do not enable the design of harmful organisms without substantial additional effort and expertise. We believe the benefits to research efficiency and accessibility outweigh the potential risks, particularly given existing biosafety regulations and oversight mechanisms in synthetic biology research.

Code Availability

Code for model training and evaluation will be released upon publication. Training data is derived from PlasmidScope and Addgene databases, available under their respective licenses.

Acknowledgements

We thank the PlasmidGPT team for providing the base model and the computational biology community for making plasmid sequence databases publicly available.

440 A. Training Configuration Details

441 A.1. Supervised Fine-Tuning Hyperparameters

- 442 • **Batch size:** 1
- 443 • **Learning rate:** 5×10^{-5} with 500 warmup steps
- 444 • **Optimizer:** AdamW
- 445 • **Epochs:** 3
- 446 • **Gradient accumulation steps:** 8
- 447 • **Hardware:** [single/multi] NVIDIA L4 GPU(s)

448 A.2. Reinforcement Learning (GRPO)

449 Hyperparameters

- 450 • **Rollout batch size:** 50
- 451 • **Policy learning rate:** 3.55×10^{-5} (found via hyperparameter sweep)
- 452 • **GRPO group size:** 16
- 453 • **Training steps:** Varies between 1000-2500 (scheduled for 2500 but rarely reached due to early convergence)
- 454 • **Convergence criteria:** Reward plateau or dip indicating collapse
- 455 • **Prompt types:** Random 4–25bp seeds (excluding ATG) and structured cassette seeds
- 456 • **Total training time:** ~10–20 hours on NVIDIA L4 GPU

457 A.3. Reward Function Configuration

- 458 • **Origin of replication (exactly 1):** Weight = 1.0
- 459 • **Selectable markers (≥ 1):** Weight = 1.5
- 460 • **Promoter→CDS→terminator bonus:** Weight = 1.5
- 461 • **Length prior (2–15kb):** Weight = 0.6, linear with maximum reward at 5kb and minimum at 15kb, zero reward beyond 15kb
- 462 • **Repeat penalty (>50bp):** -0.1 from total reward for each repeat > 50bp

463 A.4. Reference Plasmids for Distribution Comparison

464 The following common laboratory plasmids were used for distribution comparison and benchmarking analyses in Section 4.2: pUC19 (Yanisch-Perron et al., 1985), pBluescript (Short et al., 1988), pBR322 (Bolivar et al., 1977), pACYC184 (Chang & Cohen, 1978), pBAD24 (Guzman et al., 1994), pEGFP (Clontech Laboratories, 1999), pGEX-4T-1

(GE Healthcare, 2000), pET-28a (Novagen, 2005), pcDNA3 (Kaufman et al., 1991), and px330 (Cong et al., 2013). These plasmids represent widely-used vectors spanning diverse applications and size ranges (2.7–8.1 kb), providing a representative benchmark for evaluating the biological realism of generated sequences.

References

- Altschul, S. F., Gish, W., Miller, W., Myers, E. W., and Lipman, D. J. Basic local alignment search tool. *Journal of Molecular Biology*, 215(3):403–410, 1990.
- Bolivar, F., Rodriguez, R. L., Greene, P. J., Betlach, M. C., Heyneker, H. L., Boyer, H. W., Crosa, J. H., and Falkow, S. Construction and characterization of new cloning vehicles. II. A multipurpose cloning system. *Gene*, 2(2): 95–113, 1977.
- Brophy, J. A. and Voigt, C. A. Plasmid design for tunable gene expression in bacteria. *Methods in Enzymology*, 497: 371–388, 2011.
- Chang, A. C. and Cohen, S. N. Construction and characterization of new cloning vehicles. V. Mobility and coding functions of pbr322 and pbr325 'derived' plasmids. *Journal of Bacteriology*, 134(3):1141–1156, 1978.
- Clontech Laboratories. pEGFP vector information. Product Manual, 1999. Catalog #6077-1.
- Cong, L., Ran, F. A., Cox, D., Lin, S., Barretto, R., Habib, N., Hsu, P. D., Wu, X., Jiang, W., Marraffini, L. A., et al. Multiplex genome engineering using CRISPR/Cas systems. *Science*, 339(6121):819–823, 2013.
- Cunningham, A. G., Dekker, L., Shcherbakova, A., and Barnes, C. P. Generative design and construction of functional plasmids with a dna language model. *bioRxiv*, 2025. doi: 10.64898/2025.12.06.692736. URL <https://www.biorxiv.org/content/early/2025/12/07/2025.12.06.692736>.
- Deng, Y., Maurais, H. E., Etheridge, K., and Sarpehkar, R. Gene syntaxes modulate gene expression and circuit behavior on plasmids. *Journal of Biological Engineering*, 19(1):25, 2025. doi: 10.1186/s13036-025-00493-0. URL <https://doi.org/10.1186/s13036-025-00493-0>.
- Durbin, R., Eddy, S. R., Krogh, A., and Mitchison, G. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, 1998. ISBN 9780521629713.
- Fung, V., Tiwade, P. B., and Fenton, O. S. Clonefast: A simple plasmid design and construction guide

- 495 for labs venturing into synthetic biology. *STAR*
496 *Protocols*, 6(3):104025, 2025. ISSN 2666-1667.
497 doi: <https://doi.org/10.1016/j.xpro.2025.104025>.
498 URL <https://www.sciencedirect.com/science/article/pii/S2666166725004319>.
- 500 Galata, V., Fehlmann, T., Backes, C., and Keller, A. Plsdb:
501 a resource of complete bacterial plasmids. *Nucleic Acids*
502 *Research*, 47(D1):D195–D202, 10 2018. ISSN 0305-
503 1048. doi: 10.1093/nar/gky1050. URL <https://doi.org/10.1093/nar/gky1050>.
- 504 GE Healthcare. pGEX vectors. Product Manual, 2000. GST
505 Gene Fusion System.
- 506 Guzman, L.-M., Belin, D., Carson, M. J., and Beckwith, J.
507 Making transcriptional silencers work in Escherichia coli.
508 *Molecular Microbiology*, 13(4):655–662, 1994.
- 509 Hyatt, D., Chen, G.-L., LoCascio, P. F., Land, M. L.,
510 Larimer, F. W., and Hauser, L. J. Prodigal: prokaryotic
511 gene recognition and translation initiation site identifi-
512 cation. *BMC Bioinformatics*, 11(1):1–11, 2010. doi:
513 10.1186/1471-2105-11-119.
- 514 Ji, Y., Zhou, Z., Liu, H., and Davuluri, R. V. DNABERT:
515 pre-trained bidirectional encoder representations from
516 transformers model for DNA-language in genome. *Bioin-
517 formatics*, 37(15):2112–2120, 2021.
- 518 Kaufman, R. J., Davies, M. V., Pathak, V. K., and Hershey,
519 J. W. A mammalian expression vector for the expression
520 of complete open reading frames. *Nucleic Acids Research*,
521 19(16):4485–4490, 1991.
- 522 Kutzler, M. A. and Weiner, D. B. Plasmid DNA vaccines:
523 an overview. *Vaccine*, 26:S59–S75, 2008.
- 524 Lederberg, J. Cell genetics and hereditary symbiosis. *Phys-
525 iological Reviews*, 32(4):403–430, 1952. Seminal paper
526 defining the term “plasmid”.
- 527 Lin, Y., Dong, H., Wang, H., Zhang, J., Liu, J., Pi, R.,
528 Pan, R., Zhang, H., Hu, W., Zhao, H., et al. Mitigating
529 the alignment tax of RLHF. *arXiv preprint arXiv:2309.06256*, 2023.
- 530 Lorenz, R., Bernhart, S. H., Höner zu Siederdissen, C., Tafer,
531 H., Flamm, C., Stadler, P. F., and Hofacker, I. L. Vien-
532 narna package 2.0. *Algorithms for Molecular Biology*, 6
533 (1):26, 2011.
- 534 Martinson, J. N. V. et al. Generating functional plasmid
535 origins with OriGen. *Nucleic Acids Research*, 53(22):
536 gkaf1198, 2025. doi: 10.1093/nar/gkaf1198.
- 537 Meng, F. and Ellis, T. Challenges in rational design of
538 synthetic promoters. *New Biotechnology*, 32(3):337–344,
539 2015.
- 540 Nguyen, E., Poli, M., Faizi, M., Thomas, A., Birch-Sykes,
541 C., Wornow, M., Patel, A., Rabideau, C., Massaroli, S.,
542 Bengio, Y., et al. HyenaDNA: Long-range genomic se-
543 quence modeling at single nucleotide resolution. *arXiv
544 preprint arXiv:2306.15794*, 2023.
- 545 Nguyen, E., Poli, M., Durrant, M. G., Kang, B., Katrekar,
546 D., Li, D. B., Bartie, A., Thomas, A. W., King, S. H.,
547 Bifulco, G., et al. Sequence modeling and design from
548 molecular to genome scale with Evo. *bioRxiv*, 2024. doi:
549 10.1101/2024.02.27.582234.
- 550 Novagen. pET system manual. Product Manual, 11th Edi-
551 tion, 2005. TB055.
- 552 Oliveira, P. H., Prather, K. L., Prazeres, D. M., and Monteiro,
553 G. A. Structural instability of plasmid biopharmaceuti-
554 cals: challenges and implications. *Trends in Biotechnol-
555 ogy*, 27(9):503–511, 2009.
- 556 Prather, K. J., Sagar, S., Murphy, J., and Chartrain, M. In-
557 dustrial scale production of plasmid DNA for vaccine and
558 gene therapy: plasmid design, production, and purifica-
559 tion. *Enzyme and Microbial Technology*, 33(7):865–883,
560 2003.
- 561 Shao, L. et al. PlasmidGPT: a generative framework for
562 plasmid design and annotation. *bioRxiv*, 2024a. doi:
563 10.1101/2024.09.30.615762. Preprint.
- 564 Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Zhang, M.,
565 Li, Y. K., Wu, Y., and Guo, D. DeepSeekMath: Pushing
566 the limits of mathematical reasoning in open language
567 models. *arXiv preprint arXiv:2402.03300*, 2024b.
- 568 Short, J., Fernandez, J., Sorge, J., and Huse, W. Bluescript:
569 a high copy number Escherichia coli plasmid vector. *Nu-
570 cleic Acids Research*, 16(15):7583–7600, 1988.
- 571 Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B.,
572 Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D.,
573 Metzler, D., Chi, E. H., Hashimoto, T., Vinyals, O.,
574 Liang, P., Dean, J., and Fedus, W. Emergent abili-
575 ties of large language models, 2022. URL <https://arxiv.org/abs/2206.07682>.
- 576 Wu, G., Yan, Q., Jones, J. A., Tang, Y. J., Fong, S. S.,
577 and Koffas, M. A. Metabolic burden: cornerstones in
578 synthetic biology and metabolic engineering applications.
579 *Trends in Biotechnology*, 34(8):652–664, 2016.
- 580 Yanisch-Perron, C., Vieira, J., and Messing, J. New M13
581 vectors for cloning. *Gene*, 33(1):103–119, 1985.
- 582 Zou, A., Phan, L., Chen, S., Campbell, J., Guo, P., Ren, R.,
583 Pan, A., Yin, X., Mazeika, M., Dombrowski, A.-K., et al.
584 Representation engineering: A top-down approach to AI
585 transparency. *arXiv preprint arXiv:2310.01405*, 2023.