

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

Modeling Plasmid DNA using Reinforcement Learning

Anonymous Authors¹

Abstract

Plasmids are essential DNA molecules widely used in research and biomanufacturing, yet their design remains complex and iterative. We introduce Plasmid-RL, a DNA language model framework that applies supervised fine-tuning and reinforcement learning to generate structurally valid and functionally coherent plasmid sequences. Using Group Relative Policy Optimization with a biologically-motivated reward function, our model achieves an 97% quality control pass rate compared to 6% for the pretrained baseline, while maintaining 88% novelty among valid sequences. Remarkably, the RL-trained model improves on next-token prediction tasks and generates sequences with distribution characteristics closely matching natural plasmids, including thermodynamic properties not explicitly optimized. These results demonstrate that modern language modeling techniques can effectively navigate complex biological design spaces at the primary structure level, while preserving sequence diversity.

1. Introduction

Plasmids are extrachromosomal DNA sequences 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).

To address these limitations, we introduce Plasmid-RL, a DNA language modeling framework for unconditional, end-to-end plasmid generation. Using PlasmidGPT as a testbed, Plasmid-RL adapts modern post-training techniques, including supervised fine-tuning (SFT) and reinforcement learning (RL), to the foundational genomic model (Shao et al., 2024a). This approach enables the model to internalize biological constraints and requirements, allowing it to reliably generate structurally valid and functionally coherent plasmid sequences. As demonstrated in our evaluation, Plasmid-RL achieves significantly higher quality control pass rates while maintaining acceptable sequence diversity, highlighting its potential to accelerate early stage design and minimize downstream verification efforts.

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. 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 (Martin-

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.

son 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 (ORI) 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 a well-tested pipeline 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 (ORI), 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 (2–15 kb) preferred for plasmid construction.

Repeat penalty: A repeat penalty down-weights sequences containing long exact repeats that are associated with instability or recombination.

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

4. Experiments

4.1. Plasmid Quality Control and Uniqueness

We evaluate four model variants, the base pretrained model, the supervised fine-tuned (SFT) model, and the RL post-trained model, and finally the SFT + RL model, by compar-

110 ing their ability to generate unique and functionally plausible plasmids. For each model, we sampled 50 rollouts with two prompts: (i) a weak, information-poor prompt consisting of a single ATG codon, and (ii) a strong prompt containing an entire canonical GFP expression cassette.

116 4.1.1. VALIDITY ASSESSMENT

117 In silico plasmid validity was assessed using a bioinformatics quality-control pipeline that leverages BLAST based
118 tools, requiring exactly one origin of replication ($\geq 95\%$
119 identity and coverage), one or two antimicrobial resistance
120 genes ($\geq 99\%$ identity and coverage), and no internal repeats
121 longer than 50 bp (Altschul et al., 1990). This pipeline has
122 been shown to be a reasonable proxy for ability to synthesize
123 the plasmid in a lab setting (Cunningham et al., 2025).

126 4.1.2. UNIQUENESS ASSESSMENT

128 To assess whether generated plasmids represent genuinely
129 new designs rather than minor variants of existing constructs,
130 we compute similarity to known sequences using the NCBI
131 BLASTn API. Each generated plasmid is assigned to one
132 of three categories based on identity and query-coverage
133 thresholds:

Category	Identity	Coverage
Exists	$\geq 99\%$	$\geq 95\%$
Similar	$\geq 95\%$	$\geq 80\%$
Novel	Outside above criteria	

139 This categorization is based on large scale data curation
140 efforts such as PLSDB that use these thresholds of similarity
141 to attempt to de-duplicate plasmids for not adding any
142 additional value to a dataset (Galata et al., 2018).

144 4.1.3. DIVERSITY ASSESSMENT

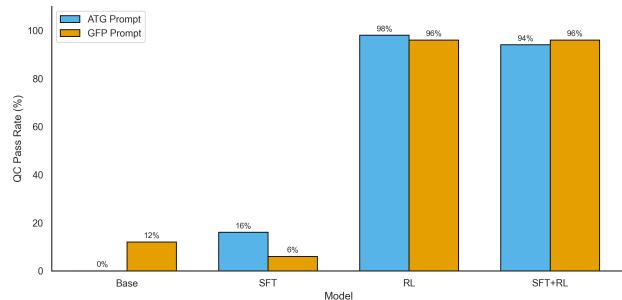
146 To detect and prevent model collapse, we attempt to measure
147 the diversity of the many samples of the model from the
148 same prompt. Due to the lack of utility of traditional NLP
149 metrics on this task, we use the mean Pairwise Jaccard
150 distance the the 21-mers of each sequence. Diversity of a
151 group of rollouts is calculated as follows:

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

157 Where, $J(S_i, S_j)$ is the Jaccard similarity between Min-
158 Hash sketches of sequences i and j , and n is the number of
159 sequences in the group.

160 4.1.4. RESULTS

162 Reinforcement learning substantially increases the probability
163 of generating plasmids that pass our bioinformatics



165 *Figure 1.* Summary of QC outcomes by prompt. Base model is
166 only able to generate functional plasmids with a strong prompt.
167 SFT allows the model to overcome this limitation on occasion but
168 still often fails QC. Adding RL to the training process improves
169 pass rate dramatically.

170 *Table 1.* Novelty and diversity metrics across model variants. RL
171 achieves dramatically higher quality (97% pass rate) with reduced
172 but acceptable diversity, while SFT+RL balances high quality
173 (95%) with improved diversity and novelty (96%).

MODEL	QC PASS RATE	% NOVEL	DIVERSITY
BASE	6%	95.5%	0.926
SFT	11%	100%	0.886
RL	97%	88%	0.391
SFT + RL	95%	96%	0.453

174 QC pipeline, while supervised fine-tuning provides modest
175 improvements. Figure 1 shows pass rates by prompt type,
176 and Table 1 summarizes novelty and diversity metrics across
177 models.

178 When prompted with the weak ATG prompt, the base model
179 never produces a valid plasmid (0%), whereas SFT increases
180 the pass rate to 16%, and RL alone further increases it to
181 96% and SFT+RL achieves a 94% pass rate. A similar
182 trend holds with the stronger GFP-cassette prompt: the
183 base model achieves 12%, SFT drops to 4%, but the RL-
184 optimized model reaches 76%.

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

192 Importantly, this increase does not come from the RL
193 models repeating previously known, high scoring sequences.
194 Among passing sequences, the proportion classified as novel
195 remains substantial for all models. Using our novelty thresh-
196 olds, the RL model produces 88% novel plasmids among its
197 QC-passing samples, compared to 95.5% for the base model.
198 The SFT model and SFT+RL model generated 100% and

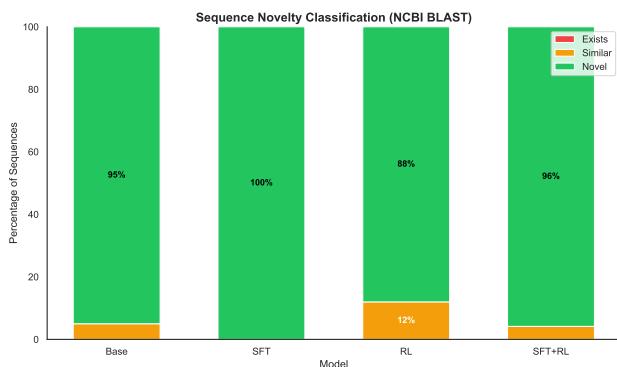


Figure 2. Summary of plasmid novelty as measured by comparison to NCBI database. SFT improves novelty of generated sequences from both base model and RL post-trained model.

96% novel sequences respectively. When normalized over all 100 rollouts per model, the RL model produces 85.4% sequences that are both QC-valid and novel, compared to 11% for SFT and 6% for the base model.

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

4.2. Distribution Comparison

We compute several statistics known to be relevant to the performance of DNA from the raw sequences of both the generated plasmids and a small subset of real plasmids used for protein expression. 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 3 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.

ORF length and codon distribution are not factored into the

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	-X.XX	Y.YY
SFT	-X.XX	Y.YY
RL	-X.XX	Y.YY

Note: $t = -2.442$, $p = 0.017$ (RL vs. others)

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 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.

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, SFT, and RL models against real plasmids in terms of next-token prediction accuracy.

We observe a small but statistically significant improvement was observed ($t = -2.442$, $p = 0.017$) for the RL model compared to the other models, and the standard deviation shrinks substantially. The magnitude of the improvement is small but positive, which is unexpected. RL generally makes language models worse at next-token prediction tasks, a phenomenon known as the “alignment tax” (Lin et al., 2023).

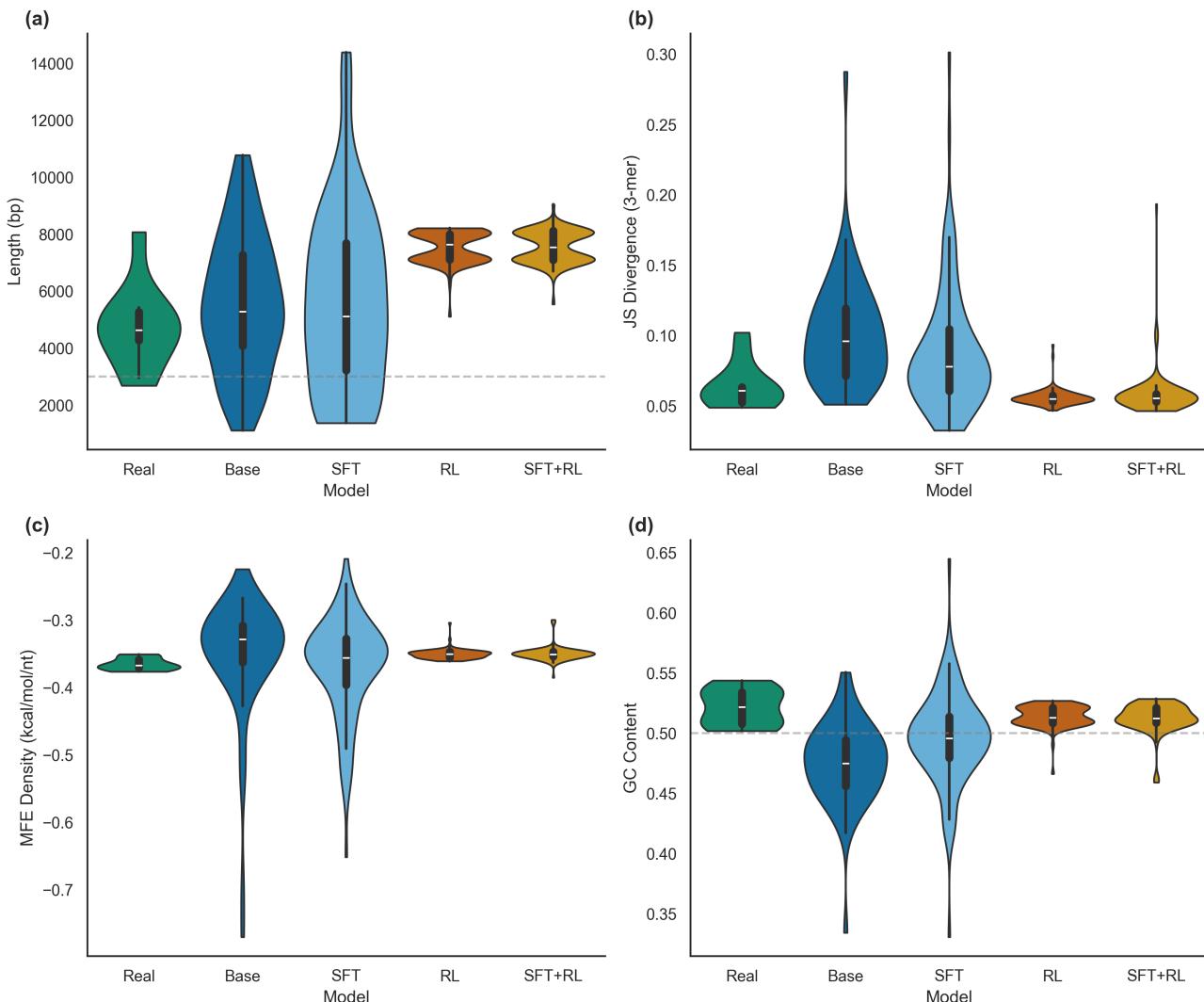


Figure 3. Distribution comparison across key biophysical metrics. The RL post-trained model (orange) closely matches real plasmid distributions (blue) across sequence length, GC content, ORF length, codon usage (Jensen-Shannon divergence), and thermodynamic stability (Gibbs free energy), while the base model (green) and SFT model (red) show substantial deviations. Notably, RL optimization produces realistic distributions even for metrics not explicitly encoded in the reward function.

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 understanding 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,

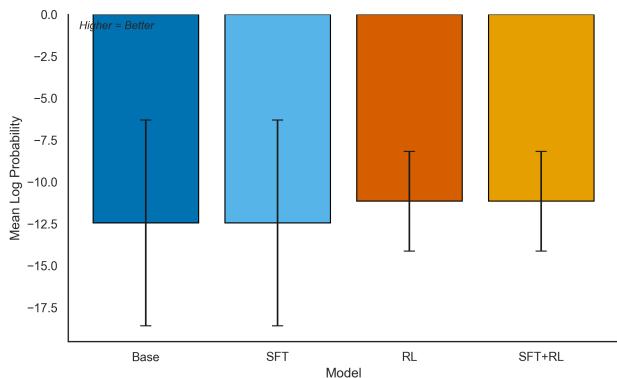


Figure 4. Held-out continuation performance across models. The RL model shows improved log-probability on real plasmid sequences, demonstrating better alignment with natural plasmid structure despite not being explicitly optimized for next-token prediction.

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 work and explores less, hence 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 error 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.

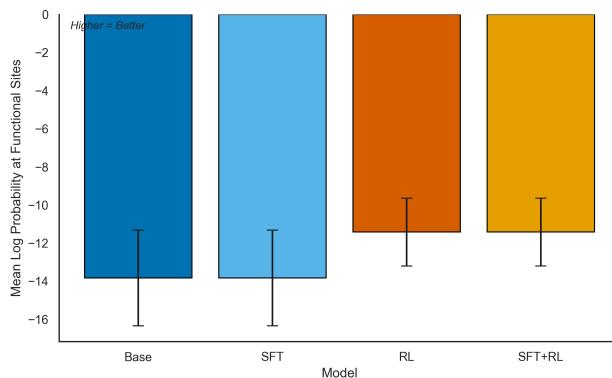


Figure 5. Coding sequence surprisal analysis. The RL model achieves lower surprisal on real plasmid coding sequences compared to the base and SFT models, indicating better capture of natural coding patterns despite using only Prodigal-based structural rewards.

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.

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. Future Work: Conditional Generation

This work proves that modern language modeling techniques can be applied to DNA language models in a similar fashion to how they have been applied to natural language in the past. The real utility of natural language models came from instruction tuning them to respond to questions and follow directions. Even if the latent space hypothesis

330 we outline is correct, navigating this latent space is far too
331 difficult to be practical without explicit conditioning.

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

340 6. Conclusion

341 We have introduced Plasmid-RL, a reinforcement learning-
342 enhanced DNA language model framework for plasmid
343 design that achieves substantial improvements in both
344 validity and novelty compared to pretrained baselines. Our
345 model generates structurally valid plasmids at a 97% rate
346 (compared to 6% for the base model) while maintaining
347 88% novelty among valid sequences. Remarkably, the RL
348 training not only optimizes for explicitly rewarded features
349 but also induces emergent properties such as realistic ther-
350 modynamic stability and codon usage patterns.

351 These results demonstrate that modern post-training tech-
352 niques from natural language processing can be effectively
353 adapted to biological sequence design, opening new av-
354 enues for accelerating synthetic biology workflows. The
355 unexpected improvement in next-token prediction suggests
356 that RL may be steering the model toward more biologi-
357 cally coherent regions of sequence space rather than simply
358 memorizing solutions.

359 Future work will focus on applying the framework to other
360 models, developing conditional generation capabilities to en-
361 able user-directed plasmid design, addressing the limitations
362 of purely bioinformatic evaluation through integration of ex-
363 perimental validation data, and extending these techniques
364 to other classes of biological sequences.

365 Impact Statement

366 This paper presents work whose goal is to advance the field
367 of computational biology and machine learning for biolog-
368 ical sequence design. The ability to generate novel, valid
369 plasmid sequences could accelerate research in synthetic
370 biology, biomanufacturing, and therapeutic development.
371 While the immediate applications are primarily beneficial
372 for scientific research, we acknowledge the dual-use nature
373 of synthetic biology tools. The methods described here
374 are limited to generating plasmid sequences based on pat-
375 terns in existing databases and do not enable the design of
376 harmful organisms without substantial additional effort and
377 expertise. We believe the benefits to research efficiency and
378 accessibility outweigh the potential risks, particularly given

379 existing biosafety regulations and oversight mechanisms in
380 synthetic biology research.

381 Acknowledgements

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

385 References

- 386 Altschul, S. F., Gish, W., Miller, W., Myers, E. W., and
387 Lipman, D. J. Basic local alignment search tool. *Journal
388 of Molecular Biology*, 215(3):403–410, 1990.
- 389 Brophy, J. A. and Voigt, C. A. Plasmid design for tunable
390 gene expression in bacteria. *Methods in Enzymology*, 497:
391 371–388, 2011.
- 392 Cunningham, A. G., Dekker, L., Shcherbakova, A., and
393 Barnes, C. P. Generative design and construction
394 of functional plasmids with a dna language model.
395 *bioRxiv*, 2025. doi: 10.64898/2025.12.06.692736.
396 URL <https://www.biorxiv.org/content/early/2025/12/07/2025.12.06.692736>.
- 397 Deng, Y., Maurais, H. E., Etheridge, K., and Sarapeshkar,
398 R. Gene syntaxes modulate gene expression and
399 circuit behavior on plasmids. *Journal of Biolog-
400 ical Engineering*, 19(1):25, 2025. doi: 10.1186/
401 s13036-025-00493-0. URL <https://doi.org/10.1186/s13036-025-00493-0>.
- 402 Durbin, R., Eddy, S. R., Krogh, A., and Mitchison, G. *Bio-
403 logical Sequence Analysis: Probabilistic Models of Pro-
404 teins and Nucleic Acids*. Cambridge University Press,
405 1998. ISBN 9780521629713.
- 406 Fung, V., Tiwade, P. B., and Fenton, O. S. Clonefast:
407 A simple plasmid design and construction guide
408 for labs venturing into synthetic biology. *STAR
409 Protocols*, 6(3):104025, 2025. ISSN 2666-1667.
410 doi: <https://doi.org/10.1016/j.xpro.2025.104025>.
411 URL <https://www.sciencedirect.com/science/article/pii/S2666166725004319>.
- 412 Galata, V., Fehlmann, T., Backes, C., and Keller, A. Plsdb:
413 a resource of complete bacterial plasmids. *Nucleic Acids
414 Research*, 47(D1):D195–D202, 10 2018. ISSN 0305-
415 1048. doi: 10.1093/nar/gky1050. URL <https://doi.org/10.1093/nar/gky1050>.
- 416 Hyatt, D., Chen, G.-L., LoCascio, P. F., Land, M. L.,
417 Larimer, F. W., and Hauser, L. J. Prodigal: prokaryotic
418 gene recognition and translation initiation site identifi-
419 cation. *BMC Bioinformatics*, 11(1):1–11, 2010. doi:
420 10.1186/1471-2105-11-119.

- 385 Ji, Y., Zhou, Z., Liu, H., and Davuluri, R. V. DNABERT:
386 pre-trained bidirectional encoder representations from
387 transformers model for DNA-language in genome. *Bioin-*
388 *formatics*, 37(15):2112–2120, 2021.
- 389 Kutzler, M. A. and Weiner, D. B. Plasmid DNA vaccines:
390 an overview. *Vaccine*, 26:S59–S75, 2008.
- 391 Lederberg, J. Cell genetics and hereditary symbiosis. *Phys-392
393 iological Reviews*, 32(4):403–430, 1952. Seminal paper
394 defining the term “plasmid”.
- 395 Lin, Y., Dong, H., Wang, H., Zhang, J., Liu, J., Pi, R.,
396 Pan, R., Zhang, H., Hu, W., Zhao, H., et al. Mitigating
397 the alignment tax of RLHF. *arXiv preprint arXiv:2309.06256*, 2023.
- 398 Lorenz, R., Bernhart, S. H., Höner zu Siederdissen, C., Tafer,
399 H., Flamm, C., Stadler, P. F., and Hofacker, I. L. Vien-
400 narna package 2.0. *Algorithms for Molecular Biology*, 6
401 (1):26, 2011.
- 402 Martinson, J. N. V. et al. Generating functional plasmid
403 origins with OriGen. *Nucleic Acids Research*, 53(22):
404 gkaf1198, 2025. doi: 10.1093/nar/gkaf1198.
- 405 Meng, F. and Ellis, T. Challenges in rational design of
406 synthetic promoters. *New Biotechnology*, 32(3):337–344,
407 2015.
- 408 Nguyen, E., Poli, M., Faizi, M., Thomas, A., Birch-Sykes,
409 C., Wornow, M., Patel, A., Rabideau, C., Massaroli, S.,
410 Bengio, Y., et al. HyenaDNA: Long-range genomic se-
411 quence modeling at single nucleotide resolution. *arXiv
412 preprint arXiv:2306.15794*, 2023.
- 413 Nguyen, E., Poli, M., Durrant, M. G., Kang, B., Katrekar,
414 D., Li, D. B., Bartie, A., Thomas, A. W., King, S. H.,
415 Bifulco, G., et al. Sequence modeling and design from
416 molecular to genome scale with Evo. *bioRxiv*, 2024. doi:
417 10.1101/2024.02.27.582234.
- 418 Oliveira, P. H., Prather, K. L., Prazeres, D. M., and Monteiro,
419 G. A. Structural instability of plasmid biopharmaceuti-
420 cals: challenges and implications. *Trends in Biotechnol-*
421 *ogy*, 27(9):503–511, 2009.
- 422 Prather, K. J., Sagar, S., Murphy, J., and Chartrain, M. In-
423 dustrial scale production of plasmid DNA for vaccine and
424 gene therapy: plasmid design, production, and purifica-
425 tion. *Enzyme and Microbial Technology*, 33(7):865–883,
426 2003.
- 427 Shao, L. et al. PlasmidGPT: a generative framework for
428 plasmid design and annotation. *bioRxiv*, 2024a. doi:
429 10.1101/2024.09.30.615762. Preprint.
- 430 Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Zhang, M.,
431 Li, Y. K., Wu, Y., and Guo, D. DeepSeekMath: Pushing
432 the limits of mathematical reasoning in open language
433 models. *arXiv preprint arXiv:2402.03300*, 2024b.
- 434 Wu, G., Yan, Q., Jones, J. A., Tang, Y. J., Fong, S. S.,
435 and Koffas, M. A. Metabolic burden: cornerstones in
436 synthetic biology and metabolic engineering applications.
437 *Trends in Biotechnology*, 34(8):652–664, 2016.
- 438 Zou, A., Phan, L., Chen, S., Campbell, J., Guo, P., Ren, R.,
439 Pan, A., Yin, X., Mazeika, M., Dombrowski, A.-K., et al.
440 Representation engineering: A top-down approach to AI
441 transparency. *arXiv preprint arXiv:2310.01405*, 2023.