PREDICTOR-GUIDED LATENT SPACE CODON OPTIMIZATION FOR MAXIMIZING PROTEIN EXPRESSION

Anonymous authors
Paper under double-blind review

ABSTRACT

Codon optimization, the process of selecting synonymous codons to improve mRNA translation efficiency and protein expression, is central to therapeutic protein production and mRNA vaccines, yet it remains a hard problem. The design space is discrete and combinatorially large, precluding gradient-based methods, and existing tools rely on heuristic proxies (e.g., Codon Adaptation Index or GC-content) that poorly capture true protein expression. We introduce Latent-Space Codon Optimization (LSCO), which recasts this discrete problem as a continuous one by mapping sequences into the latent space of a pretrained mRNA language model, enabling efficient gradient-based search. LSCO combines four components: a data-driven expression objective from an uncertainty-aware deep ensemble, whose epistemic uncertainty steers the search away from out-of-distribution designs; a Minimum-Free-Energy regularizer for structural stability; a naturalness prior from a protein-to-codon back-translation model; and constrained decoding for protein fidelity. On a real-world, wet-lab antibody expression dataset, LSCO attains the highest predicted expression among the compared methods under its native predictor and two evaluators held out from optimization, outperforming naturalness-driven and deep generative RNA foundation models, while retaining suitable biophysical properties. Notably, these gains do not come from reproducing conventional codon-bias heuristics. Ablations further show that search in the pretrained latent space outperforms direct input-space optimization, both on the guiding expression surrogate and under held-out evaluators.

1 INTRODUCTION

Synonymous codons, i.e., nucleotide triplets that redundantly encode the same amino acid, create a vast design space for tailoring mRNA without changing the associated protein (Brule & Grayhack, 2017). This space grows combinatorially with protein length: a therapeutic antibody such as trastuzumab (made of a 449 and 214 amino acid heavy and light chains, respectively) admits on the order of 10^{300} synonymous encodings (Carter et al., 1992). Codon choice within this space strongly affects translation efficiency and protein yield, and even minor synonymous changes can dramatically alter yields for industrial enzymes and therapeutic antibodies (Mauro & Chappell, 2014; Burgess-Brown et al., 2008). Codon optimization is therefore a key lever in applications from monoclonal antibody (mAb) manufacturing to mRNA therapeutics (Paremskaia et al., 2024; Mauro, 2018).

Codon optimization faces two primary challenges. First, the synonymous mRNA sequence space is discrete and combinatorially vast, ruling out exhaustive search and gradient-based methods. Second, the true objective, an accurate mRNA-to-protein expression model, is inaccessible: the biology is only partly understood and labelled data is scarce, forcing reliance on heuristic proxies (CAI, GC-content, codon-pair bias, "naturalness") that enable tractable algorithms, but often fail to capture true protein expression (Sharp & Li, 1987; Qian et al., 2012; Constant et al., 2023). Replacing these proxies with a learned expression surrogate, however, only relocates the difficulty. Such a surrogate is fit on limited data and is reliable only near its training distribution, so an optimizer that drifts far from biologically plausible codon usage will exploit model-specific artifacts rather than find genuinely improved designs. The problem is thus not merely to find high-scoring sequences, but to find them where the score can be trusted. Note also that, while the objective is formulated in

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

terms of protein expression, experimental validation typically uses A280 absorbance as a practical proxy for protein yield and abundance (Pace et al., 1995).

To address both challenges, we propose Latent-Space Codon Optimization (LSCO), which recasts codon optimization as a continuous, gradient-based problem grounded in a data-driven expression objective. By mapping sequences into the latent space of an mRNA language model, pretrained on the large-scale Observed Antibody Space (OAS) (Olsen et al., 2022) antibody-mRNA dataset, LSCO bypasses discrete search while inheriting realistic codon-usage priors. Within this space, it optimizes a multi-criteria objective built from four components: (i) an expression surrogate trained directly on mRNA-protein data rather than frequency-based proxies, specified as a deep ensemble whose epistemic uncertainty allows to steer optimization away from out-of-distribution regions; (ii) a Minimum Free Energy (MFE) regularizer that prevents drift toward near-adversarial sequences with inflated predicted yields but poor structural stability; (iii) a naturalness prior, derived from a protein-to-codon BERT back-translation model pretrained on the same OAS dataset (Olsen et al., 2022), that initializes and anchors candidates to biologically plausible codon usage; and (iv) constrained decoding that guarantees fidelity to the reference protein. Our contributions can be summarized as follows:

- We propose LSCO, a continuous latent-space codon optimizer featuring the four novel components listed above that replaces heuristic proxies with reliable expression surrogate as main objective.
- On a real-world, wet-lab antibody expression dataset, we benchmark LSCO against frequency-based, naturalness-driven, and modern generative baselines (ICOR, CodonTransformer, GEMORNA), and show it attains the highest predicted expression on its native surrogate while ranking favorably under two evaluators held out from optimization, at no cost in structural stability or host-appropriate GC content. We additionally show that these gains do not come from rediscovering conventional codon-bias heuristics, as optimized sequences move away from CAI and codon-pair signatures rather than reproducing them
- Ablations show that pretrained latent-space search optimizes the guiding expression surrogate more effectively than direct input-space optimization and that the MFE regularizer yields a controllable expression-stability trade-off.

The remainder of the paper is organized as follows. Section 2 provides background on codon optimization, defining key concepts and challenges. Section 3 covers the essential related work. Section 4 introduces Latent-Space Codon Optimization (LSCO), detailing its components. Section 5 presents our experimental setup, baselines, evaluation metrics, and results on antibody expression data. Finally, Section 6 discusses limitations and outlines directions for future work.

2 Background on Codon Optimization

A protein sequence of length L is denoted by $\mathbf{a} = (a_1, a_2, \ldots, a_L)$, with $a_i \in \mathcal{A}$, where $\mathcal{A}$ is the set of 20 canonical amino acids. Each amino acid a_i can be encoded by a subset of synonymous codons $\mathcal{C}(a_i) \subseteq \mathcal{C}$, where $\mathcal{C}$ is the full codon alphabet consisting of 64 nucleotide triplets. An mRNA sequence corresponding to $\mathbf{a}$ is thus a sequence of codons $\mathbf{c} = (c_1, c_2, \ldots, c_L)$ with $c_i \in \mathcal{C}(a_i)$. For algorithmic purposes, codon sequences are often represented in one-hot form $R \in \{0,1\}^{L\times 64}$ where each row $\mathbf{r}_i$ is a one-hot vector indicating the codon index chosen at position i. We can denote $\tilde{\mathbf{r}}_i$ instead as a probability distribution of codons at position i, i.e., $c_i \sim \text{Categorical}(\tilde{\mathbf{r}}_i)$. We drop the position index i for ease of notation in most of the following sections. Due to codon synonymity, protein $\mathbf{a}$ can be encoded by multiple synonymous mRNAs. The synonymous design space for given protein $\mathbf{a}$ is the combinatorial space $\mathcal{S}(\mathbf{a}) = \mathcal{C}(a_1) \times \cdots \times \mathcal{C}(a_L)$.

The codon optimization task revolves around parsing $\mathcal{S}(\mathbf{a})$ to find the codon sequence which maximizes expression of the protein $\mathbf{a}$, or formally:

$$\mathbf{c}^\star = \arg\max_{\mathbf{c}\in\mathcal{S}(\mathbf{a})} \Phi(\mathbf{c}) \tag{1}$$

where $\Phi : \mathcal{S}(\mathbf{a}) \to \mathbb{R}$ denotes an mRNA-to-protein expression model that admits an mRNA sequence and evaluates its expression efficiency. Ideally Φ would be a perfect biophysical simulator or empirical oracle, accounting for translation initiation, elongation rates, ribosomal availability, mRNA stability, and host-specific cellular dynamics (Zur & Tuller, 2016). In practice it is inaccessible: the underlying biology is only partly understood, labelled data is scarce, and experimen-

tally evaluating sequences across $\mathcal{S}(\mathbf{a})$ is prohibitively costly. Together with the discrete nature of $\mathcal{S}(\mathbf{a})$, which precludes gradient-based methods and demands specialized search strategies (Moore & Maranas, 2002; Arbib et al., 2020), this constitutes the *two major challenges of codon optimization*: finding a good proxy for Φ, and searching a combinatorially large discrete space.

3 Related work

Codon Optimization We begin by noting that the term "codon optimization" is often used loosely to mean *generation*, i.e., modifying/re-weighting codon distributions, rather than to indicate actual optimization towards an explicit objective; we refer to these methods as *codon generation methods*. Early examples of such methods focused heavily on maximizing the use of frequent codons within the host organism, leading to tools like JCat (Grote et al., 2005) and OPTIMIZER (Puigbo et al., 2007) that emphasize Codon Adaptation Index (CAI) and codon pair bias as key metrics. Such approaches have shown limitations, including insufficient attention to other genomic features such as mRNA stability and resulting protein expression (Zhang et al., 2023; Demissie et al., 2025; Paremskaia et al., 2024). Recent advances have seen the incorporation of deep learning into the codon generation pipeline. Pretrained mRNA language models based on transformer architectures, such as CodonBERT (Li et al., 2024a), CodonTransformer (Fallahpour et al., 2025) and GEMORNA (Zhang et al., 2025), leverage large-scale, multi-species datasets to learn complex codon usage patterns across diverse organisms via amino-acid–codon pair representations and novel tokenization, ensuring species specificity and preservation of regulatory context. *Codon optimization methods per se* explicitly optimize a set of objectives in addition to generative modelling. RNop (Gong et al., 2025) combines a suite of custom loss functions that target fidelity, tRNA availability, mRNA secondary structure, and codon adaptation within a transformer architecture. Other recent works leverage recurrent networks and active learning to improve translational efficiency across expression systems (Jain et al., 2023; Li et al., 2024b). Comparative studies have highlighted that combining multiple criteria, i.e., CAI, GC content, folding energy, codon-pair bias, outperforms single-metric objectives (Zhang et al., 2023).

Discrete Sequence Optimization General-purpose discrete sequence optimizers vary along two axes. The first is the *search space geometry*: whether the optimizer works directly in sequence space, or in a learned continuous latent space that is decoded back into sequences. The second axis is the *trust mechanism*: what stops the optimizer from exploiting the surrogate model where it is unreliable, i.e., on out-of-distribution designs (e.g., via a conservative surrogate, posterior uncertainty, etc.). We organize prior work by search space geometry, and comment on the trust mechanism in each case. *Sequence-Space Search* methods operate directly on sequences (or distributions over them). Population-based methods evolve candidates through mutation and selection: Genetic Algorithms (GA) (Deb et al., 2002) are simple but sample-inefficient when evaluations are costly (Turner et al., 2021), and model-based variants add a learned model to screen proposals (Yang et al., 2019) or guide mutation (Zhang et al., 2021). Density estimation methods instead maintain a learned distribution $q_\theta(x)$ over sequences, repeatedly sampling, reweighting toward high fitness, and refitting (Brookes & Listgarten, 2018; Brookes et al., 2019). The VAE in Brookes et al. (2019) acts only as a density *prior* to keep sampling on the same manifold, so the search still lives in sequence space. A second line of work trains a fitness surrogate and runs gradient ascent on a continuous relaxation of the sequence. This approach can fail miserably as the surrogate tends to over-predict on the very inputs the optimizer discovers, since they drift off the training distribution: a failure mode known as reward hacking (Trabucco et al., 2021; 2022; Kim et al., 2025). The proposed remedies adjust the *trust mechanism* rather than the geometry: conservative objective models regularize the surrogate to under-predict off-distribution (Trabucco et al., 2021); Reinforcement Learning methods generate tokens under a surrogate reward with per-round model selection (Angermueller et al., 2020); and Bayesian optimization over string kernels or embeddings uses posterior uncertainty and trust regions (Benjamins et al., 2024). *Latent-Space Optimization* methods learn a continuous latent space, shared by a decoder that reconstructs sequences and a model that predicts their properties, and optimizes in that latent space (Gómez-Bombarelli et al., 2018; Jin et al., 2021; Tripp et al., 2020). Latent-Space Bayesian Optimization (LSBO) adds a probabilistic surrogate, generally a Gaussian process or Deep Ensemble, to quantify epistemic uncertainty and balance exploration-exploitation across rounds of propose, test, and update (Stanton et al., 2022; Gruver et al., 2023; Amin et al., 2024). The upside is that a smooth, low-dimensional space makes gradient-based and Bayesian

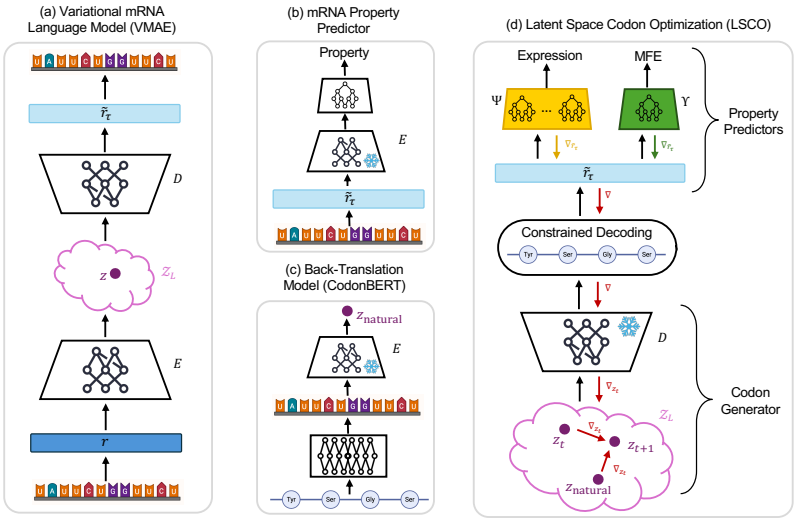


Figure 1: **LSCO framework. (a)** mRNA sequences are mapped to codon-latent space using a pretrained variational masked language model. **(b)** mRNA property predictor reuses the pretrained encoder in (a) and takes the soft and continuous version of the discrete sequence. **(c)** We find a natural mRNA sequence using a pretrained protein-to-codon back-translation model, whose representation $\mathbf{z}_{\text{nat}}$ from the pretrained encoder in (a) is used as the naturalness prior. **(d)** Latent representations $\mathbf{z}$ are optimized with respect to a data-driven expression predictor Ψ, an MFE predictor, and a naturalness prior $\mathbf{z}_{\text{nat}}$ from (c). Constrained decoding ensures valid codon choices, preserving the target protein sequence. The result is an optimized mRNA that balances expression and stability.

search easy to apply and additionally acts as a prior that keeps candidates realistic. The downside is the extra decoder, which needs to be trained and can potentially mis-decode, creating the chance for a design to fall off-distribution. 'Reward hacking' is therefore not unique to sequence space search, and motivates the use of uncertainty safeguards.

LSCO belongs to the latent-space family but differs in three ways. First, we optimize for *predicted expression* directly, not proxy metrics, within a multi-criteria objective that also weighs mRNA stability and naturalness while maintaining target protein fidelity. Second, rather than training a task-specific autoencoder, we pre-train and reuse an mRNA foundation model and conduct search in its latent space, inheriting realistic codon-usage priors. Third, our expression surrogate model is trained as a deep ensemble, whose epistemic uncertainty estimates help steer the search away from unreliable regions.

4 Latent-Space Codon Optimization

We propose Latent-Space Codon Optimization (LSCO) to address both challenges of codon optimization at once. First, LSCO recasts the discrete search into a continuous one: rather than searching the mRNA sequence space directly, we optimize codon-level embeddings in the latent space of a pretrained mRNA language model, then decode them back to the discrete sequence space at the end. Second, instead of relying on heuristic proxies such as CAI or naturalness, we train a model $\Psi : [0,1]^{L\times 64} \to \mathbb{R}$ to predict mRNA-to-protein expression directly from data, and use an uncertainty-aware utility over its predictions, U_Ψ, as the main objective. Since optimizing predicted expression alone can produce near-adversarial sequences with inflated yields but poor stability or unfavourable GC content, we add two regularizers: a predicted Minimum Free Energy term, $\text{MFE} : [0,1]^{L\times 64} \to \mathbb{R}$, and a naturalness prior U_{nat}, together producing sequences that are highly expressed, structurally stable, and within a host-appropriate GC range. The overall framework is depicted in Figure 1. Given a target protein $\mathbf{a}$, the LSCO objective is

$$\hat{\mathbf{z}} = \arg\max_{\mathbf{z}} \Big\langle \lambda_\Psi \cdot U_\Psi(D(\mathbf{z})) - \lambda_{\text{MFE}} \cdot \text{MFE}(D(\mathbf{z})) + \lambda_{\text{nat}} \cdot U_{\text{nat}}(\mathbf{z}) \Big\rangle \quad \textit{s.t.} \quad D(\mathbf{z}) \in \mathcal{S}(\mathbf{a}), \quad (2)$$

Algorithm 1: Latent-Space Codon Optimizer (LSCO)

Require: Protein sequence $\mathbf{a}$, pretrained encoder E, decoder D, predictors Ψ and MFE, weights $\lambda_\Psi, \lambda_{\text{MFE}}, \lambda_{\text{nat}}$, initial τ_{high}, final τ_{low}, optimization steps T, learning rate η
Ensure: Optimized mRNA sequence $\mathbf{c}^\star$

Initialize $\mathbf{z} \in \mathcal{Z}_L$ ▷ randomly from $\mathcal{N}(0, I)$
for $t = 1$ to T **do**
  $\tau \leftarrow \tau_{\text{high}} + (t/T)\,(\tau_{\text{low}} - \tau_{\text{high}})$ ▷ temperature annealing
  $\tilde{\mathbf{r}}_\tau \leftarrow \text{softmax}_\tau(D(\mathbf{z}) + M(\mathbf{a}))$ ▷ constrained soft sequence
  $\mathcal{L} \leftarrow \lambda_\Psi U_\Psi(\tilde{\mathbf{r}}_\tau) - \lambda_{\text{MFE}}\,\text{MFE}(\tilde{\mathbf{r}}_\tau) + \lambda_{\text{nat}}\,U_{\text{nat}}(\mathbf{z}, \mathbf{z}_{\text{nat}})$
  $\mathbf{z} \leftarrow \mathbf{z} + \eta\,\nabla_{\mathbf{z}}\mathcal{L}$ ▷ gradient *ascent* on the objective
end for
$\mathbf{r} \leftarrow \arg\max(D(\mathbf{z}) + M(\mathbf{a}))$ ▷ discrete one-hot mRNA
$\mathbf{c}^\star \leftarrow$ map $\mathbf{r}$ to codon sequence
return $\mathbf{c}^\star$

where positive $\lambda_\Psi, \lambda_{\text{MFE}}, \lambda_{\text{nat}}$ re-weight expression, MFE, and the naturalness prior; D is the latent-to-sequence decoder (generator); and $\mathcal{S}(\mathbf{a})$ is the synonymous codon space for protein $\mathbf{a}$. We describe each component below, along with the constrained decoding that enforces $\tilde{\mathbf{r}}_\tau \in \mathcal{S}(\mathbf{a})$; further implementation details are in Appendix A.

i) Variational mRNA language model. The backbone is a bidirectional transformer, since mRNA molecules exhibit non-causal long-range interactions that strongly influence their structural and biochemical properties. The model is pretrained on the curated OAS-mRNA dataset (Yazdani-Jahromi et al., 2025) (15M+ sequences) with a Variational Masked Auto-Encoder objective (Hu et al., 2022). The encoder $E : \mathbb{R}^{L\times 64} \to \mathcal{Z}_L$ maps codon sequences into a continuous latent space $\mathcal{Z}_L$, while the decoder $D : \mathcal{Z}_L \to \mathbb{R}^{L\times 64}$ reconstructs codon identities through a temperature-τ softmax, yielding a continuous relaxation $\tilde{\mathbf{r}}_\tau$ that is a smooth, differentiable surrogate for the discrete mRNA. The resulting representation preserves the explicit sequence-length dimension, fixed for a given protein, and captures biologically meaningful codon usage patterns learned from large-scale mRNA data.

ii) Expression predictor and utility. To guide the optimizer, we train an mRNA-to-protein expression network $\Psi : [0, 1]^{L\times 64} \to \mathbb{R}$ built on top of the pretrained encoder E, which we found improves predictive accuracy. The predictor takes the relaxed sequence representation $\tilde{\mathbf{r}}_\tau$ as input, rather than the latent code $\mathcal{Z}_L$, for two reasons: i) early experiments showed that gradient-based optimization in the latent space can cause large shifts in predicted expression and latent values without corresponding changes at the sequence level; ii) the softmax sequence representation admits a simple constrained decoding step enforcing $\tilde{\mathbf{r}}_\tau \in \mathcal{S}(\mathbf{a})$. To quantify predictive uncertainty, we implement Ψ as a deep ensemble (Lakshminarayanan et al., 2017) of M=10 members trained from separate random initializations (Appendix B.5), yielding an ensemble mean μ_Ψ and standard deviation σ_Ψ. The expression utility in Eq. 2 is a Lower Confidence Bound (LCB) on these predictions,

$$U_\Psi(\tilde{\mathbf{r}}_\tau) = \mu_\Psi(\tilde{\mathbf{r}}_\tau) - \lambda_{\text{LCB}}\,\sigma_\Psi(\tilde{\mathbf{r}}_\tau), \tag{3}$$

which favours high predicted yield while penalizing high-uncertainty, out-of-distribution designs, thereby regularizing the optimization trajectory to remain in-distribution.

iii) Minimum Free Energy regularizer. To prevent near-adversarial sequences and ensure mRNA stability, we train a predictor $\text{MFE} : [0, 1]^{L\times 64} \to \mathbb{R}$ on MFE labels computed with the ViennaRNA package (Lorenz et al., 2011), enabling it to approximate folding energies rapidly without explicit biophysical simulation. Lower MFE indicates more stable secondary structure, which enhances mRNA longevity and translational efficiency in the host cell (Higgs, 2000). The MFE predictor shares the architecture of the expression predictor Ψ (though without ensembling) and is likewise trained on $\tilde{\mathbf{r}}_\tau$ for differentiability and compatibility with constrained decoding.

iv) Naturalness prior. We train a *CodonBERT* (Devlin et al., 2019; Li et al., 2024a) back-translation model that predicts codon sequences from amino-acids in a single forward pass using a joint amino-acid/codon vocabulary. The model is pretrained on the curated OAS-mRNA dataset (Yazdani-Jahromi et al., 2025) with a masked language modelling objective: amino-acid tokens are randomly masked, and the model predicts the corresponding codons via cross-entropy loss. This learns the conditional distribution $p(\mathbf{c} \mid \mathbf{a}) = \prod_i p(c_i \mid \mathbf{a})$, capturing context-dependent codon usage

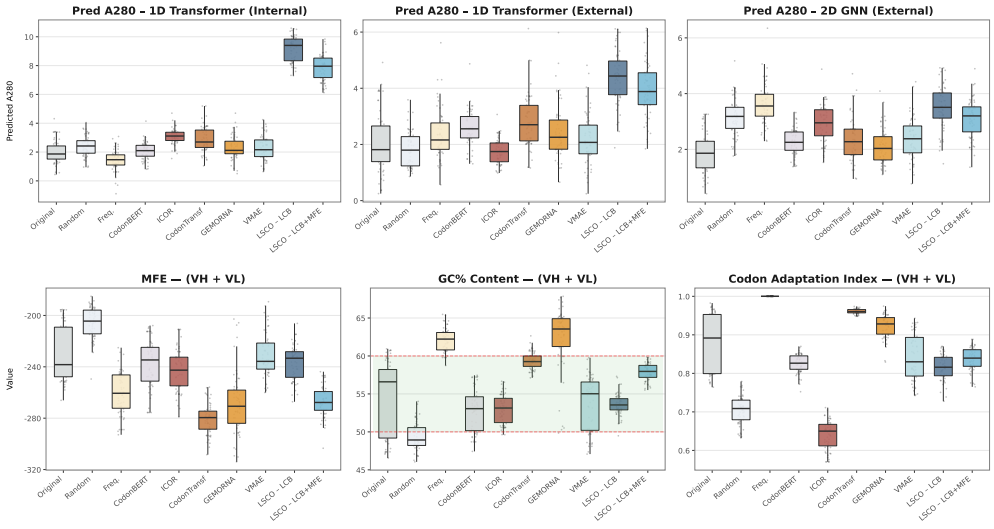


Figure 2: **Optimized sequences comparison on 54 starting mAbs.** *Top row:* predicted A280 expression from three different models: LSCO's native 1D transformer-based predictor, an external 1D transformer-based predictor with a Transformer (External) backbone, and an external structure-aware 2D GNN. LSCO variants (*LSCO–LCB*, *LSCO–LCB+MFE*) attain high or comparable predicted expression across all three predictors. *Bottom row:* biophysical metrics for the joint heavy + light chain construct (VH+VL): MFE (lower = more stable), GC% (green band marks the host-appropriate range), and CAI. The MFE-regularized LSCO (*LSCO–LCB+MFE*) recovers stability and nudges GC and CAI upward at a modest expression cost (while keeping sequences within host-appropriate ranges).

and long-range dependencies in natural mRNAs, and produces a biologically plausible reference $\mathbf{c}_{\text{nat}} = \arg\max_{\mathbf{c}} p(\mathbf{c} \mid \mathbf{a})$ reflecting usage patterns learned from millions of antibody transcripts. During optimization, the latent-space similarity between the current candidate and this reference,

$$U_{\text{nat}}(\mathbf{z}) = -\|E(\mathbf{c}) - E(\mathbf{c}_{\text{nat}})\|_2 = -\|\mathbf{z} - \mathbf{z}_{\text{nat}}\|_2, \tag{4}$$

penalizes drift toward codon compositions unlikely to be efficiently translated by the cellular machinery.

Constrained decoding. Both the back-translation model and the optimizer require every decoded mRNA to translate exactly to the target protein $\mathbf{a}$, which we enforce with a shared masking step. Let $\mathbf{l} = D(\mathbf{z}) \in \mathbb{R}^{L\times 64}$ be the pre-softmax decoder logits. For each position a_j we add a validity mask $M(a_j)[i] = 0$ if codon $i \in \mathcal{C}(a_j)$ and $-\infty$ otherwise, i.e. $l_{j,i} \leftarrow l_{j,i} + M(a_j)[i]$, before the temperature-τ softmax $\tilde{\mathbf{r}}_\tau = \text{softmax}_\tau(\mathbf{l})$. This drives non-synonymous logits to $-\infty$, guaranteeing that every decoded $\tilde{\mathbf{r}}_\tau \in \mathcal{S}(\mathbf{a})$ encodes exactly the target protein while the optimizer explores synonymous choices freely. LSCO pseudocode is given in Algorithm 1.

5 Experiments

We benchmark LSCO against codon optimization methods spanning three different families: i) simple frequency-based host-preferred codon selection (Sharp & Li, 1987); ii) an antibody-naturalness back-translation CodonBERT model (Li et al., 2024a); iii) other modern state-of-the-art deep generative models such as ICOR optimizer (Jain et al., 2023), CodonTransformer (Fallahpour et al., 2025) and GEMORNA (Zhang et al., 2025). We additionally include random synonymous codon substitution as a lower bound on optimized-sequence quality, isolating the contribution of each method above pure randomness, and the unoptimized VMAE generator we train.

Evaluation Metrics. We assess generated sequences along three axes. **(i) Predicted expression.** Protein expression is measured as absorbance at 280 nm (A280), predicted by our native 1D transformer Ψ and, to guard against surrogate exploitation, by two evaluators held out from optimization:

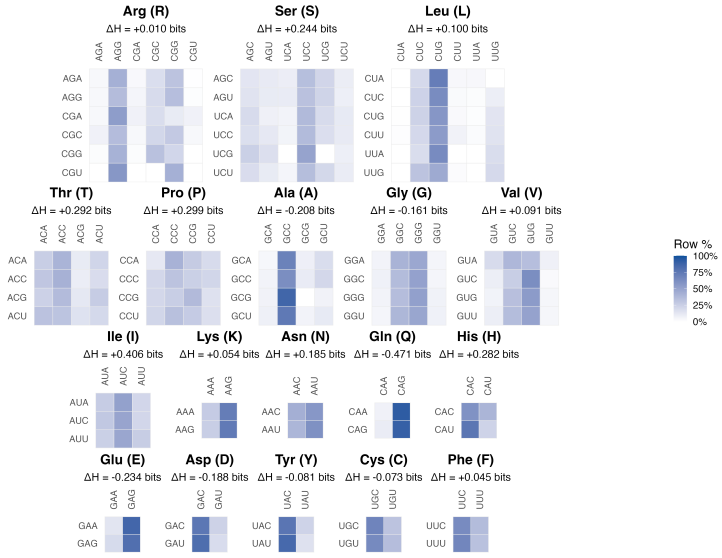


Figure 3: **Codon substitutions induced by LSCO optimization.** Rows denote original codons while columns optimized codons. Colours show row-normalized transition probabilities for synonymous substitutions produced by the *LSCO–LCB+MFE* variant. Strong diagonal structure would indicate codons conservation, while systematic off-diagonal transitions reveal learned codon preferences. Titles use three-letter (one-letter) amino-acid codes. Subtitles gives ΔH (change in Shannon entropy after optimization; negative = more peaked codon usage, positive = more uniform).

an external 1D transformer with a different encoder backbone and a structure-aware 2D GNN. A280 is a faithful expression readout for synonymous codon designs: a linear fit of yield (mg/ml) on A280 explains 98.5% of A280 variance. **(ii) Biophysical proxies.** Thermodynamic stability is quantified by the Minimum Free Energy (MFE) of the mRNA secondary structure (Lorenz et al., 2011) (lower = more stable), and host compatibility by GC content (GC%), since excessively high or low GC can impair translation (Kudla et al., 2009). **(iii) Codon-usage descriptors.** To characterize *how* each method reshapes codon usage, we report the Codon Adaptation Index (CAI), a conventional but coarse proxy for host-favoured codons, in addition to other descriptors featured in Appendix B. These include: the Codon-Pair Score (2-gram) and KL-divergence codon usage gap between high versus low abundance proteins (1- and 2-gram), which capture bias toward a highly-expressed codon signature; the L_1 distance of the codon (1-gram) and codon-pair (2-gram) distributions to a highly-translated Chinese hamster ovary (CHO) reference (Ang et al., 2016); Shannon entropy of the 59 non-degenerate codon distribution, measuring usage uniformity; and codon Hamming distance from the original sequence, measuring the number of synonymous edits. Unless noted, all quantities are computed on the joint heavy and light V-regions and reported as medians across the 54 mAbs. These codon usage metrics are reported in the appendix (Figure 4 and Table 3).

5.1 LSCO EVALUATION

We evaluate LSCO against the baselines on 54 selected antibody sequences, chosen from the mRNA expression dataset to stratify across antigens (6 mAbs per antigen, 9 antigens) while favouring mAbs with moderate A280 values that leave room for improvement. Every optimized design is scored by predicted A280 with LSCO's native 1D transformer predictor and with two evaluators held out from optimization: an external 1D transformer with a different pretrained encoder backbone, and a structure-aware 2D GNN. We test two variants: *LSCO–LCB*, optimizing expression U_Ψ and naturalness U_{nat}; and *LSCO–LCB+MFE*, which adds MFE regularization MFE ($\lambda_{\text{MFE}} = 0.1$).

LSCO ranks high across all three predictors. LSCO attains the highest predicted A280 among compared baselines on both 1D predictors and scores favorably on the 2D GNN (top row, Fig. 2). Because the two external evaluators were held out from optimization and carry different inductive biases from the native surrogate, their consistent ranking of LSCO indicates the gain is not an artifact of the optimizer exploiting its own predictor. Cross-evaluator rank agreement is lower for optimized

λ_{MFE}	λ_{LCB}	Predicted A280			Metrics (VH + VL)		
		1D Transf (Internal)	1D Transf (External)	2D GNN	MFE	GC%	CAI
0	0	$\mathbf{9.585 \pm 0.218}$	$\mathbf{4.518 \pm 0.223}$	$\mathbf{3.492 \pm 0.203}$	-235.187 ± 3.555	53.303 ± 0.423	0.816 ± 0.008
	1	9.156 ± 0.241	4.400 ± 0.254	3.485 ± 0.212	-236.711 ± 3.603	53.601 ± 0.390	0.815 ± 0.009
0.1	0	8.180 ± 0.237	3.900 ± 0.224	3.175 ± 0.217	-266.274 ± 2.783	57.756 ± 0.285	0.830 ± 0.009
	1	7.858 ± 0.235	3.935 ± 0.265	3.099 ± 0.187	$\mathbf{-267.050 \pm 3.211}$	57.916 ± 0.300	0.837 ± 0.008
1	0	8.065 ± 0.232	3.887 ± 0.200	3.319 ± 0.201	-263.765 ± 3.122	58.003 ± 0.304	0.825 ± 0.009
	1	7.899 ± 0.256	3.951 ± 0.271	2.923 ± 0.182	-266.365 ± 3.381	58.018 ± 0.317	$\mathbf{0.838 \pm 0.008}$

Table 1: **MFE sensitivity analysis for LSCO variants.** Predicted A280 values from the three models (1D Transformer (Internal), 1D Transformer (External), 2D GNN) and biophysical metrics (MFE, GC content, CAI) reported for the joint heavy + light chain, on the same 54 starting mAbs as Figure 2. The second row ($\lambda_{\text{MFE}} = 0$, $\lambda_{\text{LCB}} = 1$) is the LCB-only LSCO baseline ('LSCO–LCB' in Figure 2). The fourth row ($\lambda_{\text{MFE}} = 0.1, \lambda_{\text{LCB}} = 1$) is the LCB+MFE baseline ('LSCO–LCB+MFE' in Figure 2) that best balances A280 and MFE objectives. **Bold** indicates the best value per column: highest for A280 predictions and CAI, most negative for MFE. GC% is not bolded as "best" is target-dependent. Values are mean $\pm$ 95% confidence bands across starting sequences.

than for original antibodies and decreases as methods introduce larger synonymous changes (Suppl. Fig. 5), consistent with optimization pushing candidates into regions where predictors extrapolate (motivating our use of multiple evaluators rather than a single surrogate). LSCO remains top-ranked across all three, suggesting its advantage stems from directly optimizing a learned expression surrogate in a continuous codon-latent space rather than hand-crafted proxies.

LSCO offers controllable expression-stability trade-off. Expression-only variant (*LSCO–LCB*) maximizes predicted A280 but yields weak secondary-structure stability, with MFE comparable to the original antibodies (median -233 vs -238) and well above strong baselines such as CodonTransformer and GEMORNA. Adding the MFE regularizer (*LSCO–LCB+MFE*) recovers stability (median around -267) at modest expression cost (bottom row, Fig. 2). This stability is not an artifact of GC inflation. Both LSCO variants keep GC within the host-appropriate band (54% and 58%), unlike frequency-based and CodonTransformer/GEMORNA that reach low MFE partly by pushing GC above the admissible range, which can impede ribosome accessibility (Gustafsson et al., 2004).

LSCO learns a distinct, more balanced codon strategy. Figure 3 shows codon-level transition probabilities between original and optimized sequences per amino acid. Several amino acids exhibit strong off-diagonal structure, indicating optimization redirects usage toward a preferred subset of synonymous codons, with a moderate shift toward GC-richer codons, consistent with the GC and MFE gains of *LSCO–LCB+MFE*. This is not a rediscovery of conventional codon-bias heuristics. Relative to the original antibodies, LSCO *lowers* the classical targets (CAI and codon-pair score) rather than raising them (unlike frequency-based methods and CodonTransformer, which drive CAI toward its ceiling and codon-pair scores positive). It also avoids a peaked codon-usage signature, keeping the high- vs. low-protein-abundance KL gap at or below the original. Codon-usage uniformity increases (higher Shannon entropy), whereas frequency-based optimization, CodonTransformer and GEMORNA reduce it (Suppl. Fig. 4, Table 3). LSCO–LCB variant exhibits a slight shift towards a canonical highly-translated CHO reference, but this disappears once MFE regularization is added. Other meaningful codon-bias statistics can be found in the appendix. Together, the picture that emerges is that LSCO improves predicted expression while preserving structural stability and host compatibility, a balance prior codon optimizers fail to strike.

5.2 Ablation and Sensitivity Studies

We conduct two additional experimental studies on LSCO components: a sensitivity analysis over the MFE regularization strength (Table 1), and an ablation that isolates the contribution of the pretrained VMAE mRNA language model, comparing search in the pretrained model's latent space vs. directly in the input space (Table 2). Throughout, we report predicted A280 from the native 1D Transformer (Internal) predictor and the two held-out evaluators (1D Transformer (External), 2D GNN), alongside the usual biophysical metrics (MFE, GC%, CAI).

MFE regularization and expression-stability trade-off. Table 1 sweeps λ_{MFE} hyperparameter values, while toggling λ_{LCB}, the uncertainty penalty on σ_Ψ (Eq. 3). The expression-only variant

Family	Variant	λ_{nat}	Predicted A280			Metrics (VH + VL)		
			1D Transformer (Internal)	1D Transformer (External)	2D GNN (External)	MFE	GC%	CAI
LSCO	LCB	0.01	$\mathbf{9.156 \pm 0.241}$	4.400 ± 0.254	$\mathbf{3.485 \pm 0.212}$	-236.711 ± 3.603	53.601 ± 0.390	0.815 ± 0.009
	LCB	0	9.024 ± 0.247	$\mathbf{4.564 \pm 0.240}$	3.248 ± 0.193	-236.491 ± 3.900	53.428 ± 0.465	0.808 ± 0.008
	LCB+MFE	0.01	7.858 ± 0.235	3.935 ± 0.265	3.099 ± 0.187	$\mathbf{-267.050 \pm 3.211}$	57.916 ± 0.300	0.837 ± 0.008
	LCB+MFE	0	7.956 ± 0.208	3.885 ± 0.249	3.259 ± 0.254	-265.006 ± 3.066	57.860 ± 0.305	0.836 ± 0.009
DISO	LCB	0.01	8.547 ± 0.314	4.423 ± 0.240	3.220 ± 0.192	-239.578 ± 3.904	53.594 ± 0.461	0.829 ± 0.007
	LCB	0	8.528 ± 0.305	4.393 ± 0.230	3.036 ± 0.174	-239.561 ± 3.678	53.688 ± 0.421	0.829 ± 0.007
	LCB+MFE	0.01	7.631 ± 0.268	3.828 ± 0.241	2.939 ± 0.206	-264.987 ± 3.625	57.428 ± 0.353	0.839 ± 0.010
	LCB+MFE	0	7.598 ± 0.246	3.840 ± 0.247	2.923 ± 0.187	-264.994 ± 3.639	57.382 ± 0.355	$\mathbf{0.840 \pm 0.010}$
Δ = LSCO–DISO	LCB	0.01	$\mathbf{0.609 \pm 0.202}^{***}$	-0.023 ± 0.270	0.264 ± 0.247	2.867 ± 3.991	0.007 ± 0.369	$\mathbf{-0.014 \pm 0.009}^{*}$
	LCB	0	$\mathbf{0.496 \pm 0.173}^{***}$	0.171 ± 0.275	0.212 ± 0.268	3.070 ± 3.847	-0.260 ± 0.395	$\mathbf{-0.020 \pm 0.009}^{***}$
	LCB+MFE	0.01	$\mathbf{0.227 \pm 0.209}^{*}$	0.107 ± 0.307	0.159 ± 0.303	-2.063 ± 4.059	$\mathbf{0.487 \pm 0.266}^{**}$	-0.002 ± 0.010
	LCB+MFE	0	$\mathbf{0.359 \pm 0.185}^{***}$	0.044 ± 0.283	0.336 ± 0.302	-0.011 ± 3.852	$\mathbf{0.477 \pm 0.291}^{*}$	-0.004 ± 0.010

Table 2: **Ablation study of LSCO and DISO.** Predicted A280 from three surrogate models (1D Transformer (Internal), 1D Transformer (External), and 2D GNN) together with sequence-level metrics (MFE, GC content, and CAI) for joint VH+VL constructs, evaluated on the same 54 starting mAbs as Fig. 2. Ablations include removal of the naturalness prior ($\lambda_{\text{nat}} = 0$) for both LSCO and DISO. Values reported are mean $\pm 95\%$ confidence interval across starting sequences. In the LSCO and DISO blocks, **bold** denotes the best value within a column (highest for A280 and CAI; most negative for MFE). GC% is not bolded as no optimum exists. The Δ block reports paired differences (LSCO$-$DISO) as mean difference $\pm 95\%$ confidence interval. Positive values indicate higher metrics under LSCO. Significance markers denote Holm-adjusted paired Wilcoxon signed-rank tests (* $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$).

($\lambda_{\text{MFE}} = 0$) attains the highest predicted A280 across all three predictors, but at the weakest stability (MFE) and lowest GC. Introducing even a light MFE penalty ($\lambda_{\text{MFE}} = 0.1$) drives MFE substantially lower and lifts GC and CAI into the host-appropriate range, at only a modest expression cost; larger weights ($\lambda_{\text{MFE}} \geq 1$) yield no further stability gain. Enabling the uncertainty penalty only slightly lowers A280, while keeping the optimization from drifting out of distribution.

Latent-Space vs Direct Input-Space Optimization. Table 2 compares LSCO against Direct Input-Space Optimization (DISO) and ablates the naturalness prior ($\lambda_{\text{nat}} = 0$). DISO differs only in the search space: it optimizes per-position codon logits directly, with no VMAE decoder in the loop, sharing all other components with LSCO (Appendix C). We test the *LCB* and *LCB + MFE* variants of Figure 2 and report paired, Holm-adjusted Wilcoxon differences $\Delta = \text{LSCO} - \text{DISO}$. Under the native predictor that guides optimization, LSCO exceeds DISO in all four settings (all $p_{\text{adj}}! < 0.05$), showing latent-space search maximizes the learned objective more effectively than input-space logits; on the two held-out evaluators the methods are indistinguishable, so this comes at no cost to generalization. Biophysical profiles are comparable (MFE n.s.), and where CAI differs LSCO is significantly *lower*, consistent with the naturalness prior restraining CAI-inflating drift. Ablating the prior changes A280 only slightly and leaves other metrics largely unchanged, suggesting it acts as a lightweight anti-drift anchor rather than a primary driver of the gains.

6 Discussion

We introduced Latent-Space Codon Optimization (LSCO), which reformulates codon optimization from a discrete, heuristic-driven search into a continuous, gradient-based one in the latent space of a pretrained mRNA language model. On antibody data, LSCO outperforms frequency-based and deep generative baselines (ICOR, CodonTransformer, GEMORNA) under both the internal predictor and two external evaluators, and latent-space search optimizes the guiding surrogate more effectively than input-space optimization at no cost to held-out predicted expression. Notably, LSCO does not simply recover established codon-usage rules: CAI, CHO reference frequency, and codon-pair scores often decrease relative to the original antibodies even as predicted expression rise. The pretrained model thus serves not merely as a generative prior, but as an optimization geometry for an uncertainty-aware design with controllable trade-offs between translation and structural stability. LSCO's performance depends on the quality of the data used to train the mRNA language model and expression predictors. Public mRNA datasets with reliable expression are scarce and often proprietary, making benchmarking beyond surrogate metrics difficult; like most work in this area, our evaluation relies on in-silico rather than empirical expression. The natural next step is wet-lab validation, as LSCO can be incorporated into a lab-in-the-loop Bayesian Optimization pipeline.

AI USE STATEMENT

In this work, we used generative AI tools to assist with code implementation and to improve the clarity and concision of the writing; all other disclosure categories are not applicable to this work. All AI-assisted code was verified and tested by the authors, all AI-assisted text was reviewed and edited, and we take responsibility for the final content of this work

ETHICS STATEMENT

This work does not involve human subjects, personally identifiable data, or applications that raise foreseeable ethical concerns. All data used are proprietary antibody sequence datasets analyzed for protein expression optimization.

REPRODUCIBILITY STATEMENT

We have sought to describe our approach in sufficient detail to support independent replication. The method and its components are specified in Section 4, with the overall procedure summarized in Algorithm 1. The experimental setup, baselines and evaluation protocol are described in Section 5, and the Appendix provides further implementation details, ablations and sensitivity analyses supporting the claims made in the main text. All baselines and pretrained components used in our comparisons are publicly available and are cited accordingly. The data underlying the surrogate models are subject to confidentiality obligations and are not available for release.

REFERENCES

Alan Nawzad Amin, Nate Gruver, Yilun Kuang, Lily Li, Hunter Elliott, Calvin McCarter, Aniruddh Raghu, Peyton Greenside, and Andrew Gordon Wilson. Bayesian optimization of antibodies informed by a generative model of evolving sequences. *arXiv preprint arXiv:2412.07763*, 2024.

Kok Siong Ang, Sarantos Kyriakopoulos, Wei Li, and Dong-Yup Lee. Multi-omics data driven analysis establishes reference codon biases for synthetic gene design in microbial and mammalian cells. *Methods*, 102:26–35, 2016. doi: 10.1016/j.ymeth.2016.01.016. URL https://doi.org/10.1016/j.ymeth.2016.01.016.

Christof Angermueller, David Dohan, David Belanger, Ramya Deshpande, Kevin Murphy, and Lucy Colwell. Model-based reinforcement learning for biological sequence design. In *International Conference on Learning Representations (ICLR)*, 2020. URL https://openreview.net/forum?id=HklxbgBKvr.

Claudio Arbib, Mustafa Ç Pınar, Fabrizio Rossi, and Alessandra Tessitore. Codon optimization by 0-1 linear programming. *Computers & Operations Research*, 119:104932, 2020.

Carolin Benjamins, Shikha Surana, Oliver Bent, Marius Lindauer, and Paul Duckworth. Bayesian optimization for protein sequence design: Back to simplicity with gaussian processes. In *NeurIPS 2024 AI4MAT Workshop*, 2024. URL https://openreview.net/forum?id=zori4pHmFP.

David H. Brookes and Jennifer Listgarten. Design by adaptive sampling. In *NeurIPS Workshop on Machine Learning for Molecules and Materials*, 2018. doi: 10.48550/arXiv.1810.03714. URL https://arxiv.org/abs/1810.03714.

David H. Brookes, Hahnbeom Park, and Jennifer Listgarten. Conditioning by adaptive sampling for robust design. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, volume 97 of *Proceedings of Machine Learning Research*, pp. 773–782. PMLR, 2019. doi: 10.48550/arXiv.1901.10060. URL https://proceedings.mlr.press/v97/brookes19a.html.

Christina E Brule and Elizabeth J Grayhack. Synonymous codons: choose wisely for expression. *Trends in Genetics*, 33(4):283–297, 2017.

Nicola A Burgess-Brown, Sujata Sharma, Frank Sobott, Christoph Loenarz, Udo Oppermann, and Opher Gileadi. Codon optimization can improve expression of human genes in escherichia coli: A multi-gene study. *Protein expression and purification*, 59(1):94–102, 2008.

Paul Carter, Leonard Presta, Cornelia M. Gorman, John B. B. Ridgway, Dennis Henner, Wai Lee T. Wong, Ann M. Rowland, Claire Kotts, Monique E. Carver, and H. Michael Shepard. Humanization of an anti-p185^{HER2} antibody for human cancer therapy. *Proceedings of the National Academy of Sciences of the USA*, 89(10):4285–4289, 1992. doi: 10.1073/pnas.89.10.4285.

David A Constant, Jahir M Gutierrez, Anand V Sastry, Rebecca Viazzo, Nicholas R Smith, Jubair Hossain, David A Spencer, Hayley Carter, Abigail B Ventura, Michael TM Louie, et al. Deep learning-based codon optimization with large-scale synonymous variant datasets enables generalized tunable protein expression. *bioRxiv*, pp. 2023–02, 2023.

Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE transactions on evolutionary computation*, 6(2): 182–197, 2002.

Eden A Demissie, Seo-Young Park, Je Hun Moon, and Dong-Yup Lee. Comparative analysis of codon optimization tools: Advancing toward a multi-criteria framework for synthetic gene design. *Journal of Microbiology and Biotechnology*, 35:e2411066, 2025.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding, 2019. URL https://arxiv.org/abs/1810.04805.

Adibvafa Fallahpour, Vincent Gureghian, Guillaume J Filion, Ariel B Lindner, and Amir Pandi. CodonTransformer: a multispecies codon optimizer using context-aware neural networks. *Nature Communications*, 16(1):3205, 2025.

Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, 4(2):268–276, 2018.

Zheng Gong, Ziyi Jiang, Weihao Gao, Deng Zhuo, and Lan Ma. A new deep-learning-based approach for mRNA optimization: High fidelity, computation efficiency, and multiple optimization factors. *arXiv preprint arXiv:2505.23862*, 2025.

Andreas Grote, Karsten Hiller, Maurice Scheer, Richard Münch, Bernd Nörtemann, Dietmar C Hempel, and Dieter Jahn. JCat: a novel tool to adapt codon usage of a target gene to its potential expression host. *Nucleic acids research*, 33(suppl_2):W526–W531, 2005.

Nate Gruver, Samuel Don Stanton, Nathan C. Frey, Tim G. J. Rudner, Isidro Hotzel, Julien Lafrance-Vanasse, Arvind Rajpal, Kyunghyun Cho, and Andrew Gordon Wilson. Protein design with guided discrete diffusion. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=MfiK69Ga6p.

Claes Gustafsson, Sridhar Govindarajan, and Jeremy Minshull. Codon bias and heterologous protein expression. *Trends in biotechnology*, 22(7):346–353, 2004.

Paul G Higgs. RNA secondary structure: physical and computational aspects. *Quarterly reviews of biophysics*, 33(3):199–253, 2000.

Dou Hu, Xiaolong Hou, Xiyang Du, Mengyuan Zhou, Lianxin Jiang, Yang Mo, and Xiaofeng Shi. Varmae: Pre-training of variational masked autoencoder for domain-adaptive language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pp. 6276–6286, 2022.

Rishab Jain, Aditya Jain, Elizabeth Mauro, Kevin LeShane, and Douglas Densmore. ICOR: improving codon optimization with recurrent neural networks. *BMC bioinformatics*, 24(1):132, 2023.

Wengong Jin, Jeremy Wohlwend, Regina Barzilay, and Tommi Jaakkola. Iterative refinement graph neural network for antibody sequence-structure co-design. *arXiv preprint arXiv:2110.04624*, 2021.

Minsu Kim, Jiayao Gu, Ye Yuan, Taeyoung Yun, Zixuan Liu, Yoshua Bengio, and Can Chen. Offline model-based optimization: Comprehensive review. *Transactions on Machine Learning Research (TMLR)*, 2025. doi: 10.48550/arXiv.2503.17286. URL https://arxiv.org/abs/2503.17286. Survey Certification.

Grzegorz Kudla, Andrew W Murray, David Tollervey, and Joshua B Plotkin. Coding-sequence determinants of gene expression in *escherichia coli*. *science*, 324(5924):255–258, 2009.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

Sizhen Li, Saeed Moayedpour, Ruijiang Li, Michael Bailey, Saleh Riahi, Lorenzo Kogler-Anele, Milad Miladi, Jacob Miner, Fabien Pertuy, Dinghai Zheng, et al. CodonBERT large language model for mRNA vaccines. *Genome research*, 34(7):1027–1035, 2024a.

Yupeng Li, Fan Wang, Jiaqi Yang, Zirong Han, Linfeng Chen, Wenbing Jiang, Hao Zhou, Tong Li, Zehua Tang, Jianxiang Deng, et al. Deep generative optimization of mRNA codon sequences for enhanced protein production and therapeutic efficacy. *bioRxiv*, pp. 2024–09, 2024b.

Ronny Lorenz, Stephan H Bernhart, Christian Höner zu Siederdissen, Hakim Tafer, Christoph Flamm, Peter F Stadler, and Ivo L Hofacker. Viennarna package 2.0. *Algorithms for molecular biology*, 6(1):26, 2011.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Vincent P Mauro. Codon optimization in the production of recombinant biotherapeutics: potential risks and considerations. *BioDrugs*, 32(1):69–81, 2018.

Vincent P Mauro and Stephen A Chappell. A critical analysis of codon optimization in human therapeutics. *Trends in molecular medicine*, 20(11):604–613, 2014.

Gregory L Moore and Costas D Maranas. e codonopt: a systematic computational framework for optimizing codon usage in directed evolution experiments. *Nucleic acids research*, 30(11):2407–2416, 2002.

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

C Nick Pace, Felix Vajdos, Lanette Fee, Gerald Grimsley, and Theronica Gray. How to measure and predict the molar absorption coefficient of a protein. *Protein science*, 4(11):2411–2423, 1995.

Anastasiia Iu Paremskaia, Anna A Kogan, Anastasiia Murashkina, Daria A Naumova, Anakha Satish, Ivan S Abramov, Sofya G Feoktistova, Olga N Mityaeva, Andrei A Deviatkin, and Pavel Yu Volchkov. Codon-optimization in gene therapy: promises, prospects and challenges. *Frontiers in Bioengineering and Biotechnology*, 12:1371596, 2024.

Mangal Prakash, Artem Moskalev, Peter DiMaggio Jr., Steven Combs, Tommaso Mansi, Justin Scheer, and Rui Liao. Bridging biomolecular modalities for knowledge transfer in bio-language models. In *Neurips 2024 Workshop Foundation Models for Science: Progress, Opportunities, and Challenges*, 2024. URL https://openreview.net/forum?id=dicOSQVPLm.

Pere Puigbo, Eduard Guzmán, Antoni Romeu, and Santiago Garcia-Vallvé. OPTIMIZER: a web server for optimizing the codon usage of DNA sequences. *Nucleic acids research*, 35(suppl_2): W126–W131, 2007.

Wenfeng Qian, Jian-Rong Yang, Nathaniel M Pearson, Calum Maclean, and Jianzhi Zhang. Balanced codon usage optimizes eukaryotic translational efficiency. *PLoS genetics*, 8(3):e1002603, 2012.

Paul M Sharp and Wen-Hsiung Li. The codon adaptation index-a measure of directional synonymous codon usage bias, and its potential applications. *Nucleic acids research*, 15(3):1281–1295, 1987.

Samuel Stanton, Wesley Maddox, Nate Gruver, Phillip Maffettone, Emily Delaney, Peyton Greenside, and Andrew Gordon Wilson. Accelerating bayesian optimization for biological sequence design with denoising autoencoders. In *International conference on machine learning*, pp. 20459–20478. PMLR, 2022.

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

Brandon Trabucco, Aviral Kumar, Xinyang Geng, and Sergey Levine. Conservative objective models for effective offline model-based optimization. *Proceedings of the 38th International Conference on Machine Learning (ICML)*, 2021. doi: 10.48550/arXiv.2107.06882. URL https://arxiv.org/abs/2107.06882.

Brandon Trabucco, Xinyang Geng, Aviral Kumar, and Sergey Levine. Design-bench: Benchmarks for data-driven offline model-based optimization. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*, volume 162 of *PMLR*, pp. 21658–21676, 2022. doi: 10.48550/arXiv.2202.08450. URL https://proceedings.mlr.press/v162/trabucco22a.html.

Austin Tripp, Erik Daxberger, and José Miguel Hernández-Lobato. Sample-efficient optimization in the latent space of deep generative models via weighted retraining. *Advances in Neural Information Processing Systems*, 33:11259–11272, 2020.

Ryan Turner, David Eriksson, Michael McCourt, Juha Kiili, Eero Laaksonen, Zhen Xu, and Isabelle Guyon. Bayesian optimization is superior to random search for machine learning hyperparameter tuning: Analysis of the black-box optimization challenge 2020. In *NeurIPS 2020 competition and demonstration track*, pp. 3–26. PMLR, 2021.

Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, Nathan Cooper, Griffin Adams, Jeremy Howard, and Iacopo Poli. Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference, 2024. URL https://arxiv.org/abs/2412.13663.

Kevin K Yang, Zachary Wu, and Frances H Arnold. Machine-learning-guided directed evolution for protein engineering. *Nature methods*, 16(8):687–694, 2019.

Mehdi Yazdani-Jahromi, Mangal Prakash, Tommaso Mansi, Artem Moskalev, and Rui Liao. HELM: Hierarchical encoding for mRNA language modeling. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=MMHqnUOnl0.

Dinghuai Zhang, Jie Fu, Yoshua Bengio, and Aaron Courville. Unifying likelihood-free inference with black-box optimization and beyond. *arXiv preprint arXiv:2110.03372*, 2021.

He Zhang, Liang Zhang, Ang Lin, Congcong Xu, Ziyu Li, Kaibo Liu, Boxiang Liu, Xiaopin Ma, Fanfan Zhao, Huiling Jiang, et al. Algorithm for optimized mRNA design improves stability and immunogenicity. *Nature*, 621(7978):396–403, 2023.

He Zhang, Hailong Liu, Yushan Xu, Haoran Huang, Yiming Liu, Jia Wang, Yan Qin, Haiyan Wang, Lili Ma, Zhiyuan Xun, et al. Deep generative models design mRNA sequences with enhanced translational capacity and stability. *Science*, 390(6773):eadr8470, 2025.

Hadas Zur and Tamir Tuller. Predictive biophysical modeling and understanding of the dynamics of mrna translation and its evolution. *Nucleic acids research*, 44(19):9031–9049, 2016.

A IMPLEMENTATION DETAILS

Back-translation model The back-translation model uses the standard BERT architecture with absolute position embeddings, configured with a hidden dimension of 777, 9 transformer layers, 21 attention heads per layer, a feed-forward dimension of 777, and a maximum sequence length of 2048 tokens. It operates over a joint vocabulary of 91 tokens: 27 amino-acid tokens (20 canonical amino acids plus 7 special tokens `[BOS]`, `[EOS]`, `[CLS]`, `[SEP]`, `[MASK]`, `[PAD]`, and `[UNK]`) as the input alphabet, and 64 codon tokens (one per sense codon) as the output alphabet. During training, amino-acid input tokens are masked with probability $p = 0.2$.

Variational mRNA language model We use a 12-layer ModernBERT architecture (Warner et al., 2024) with Rotary Position Embeddings (RoPE) (Su et al., 2024). The model uses a vocabulary of 70 tokens (64 codons plus 6 special tokens `[BOS]`, `[EOS]`, `[CLS]`, `[SEP]`, `[MASK]`, and `[PAD]`), a hidden dimension of 768, 12 attention heads per layer, and a feed-forward dimension of 3072. The VMAE masking ratio is set to $p = 0.15$. The model is split at layer $k = 6$, using the first six layers as the encoder E and the remaining six as the decoder D. We use a standard Gaussian prior $\mathcal{N}(0, I)$ over the latent variables $\mathbf{z} \in \mathcal{Z}_L$. Pretraining is performed for 40 epochs using AdamW (Loshchilov & Hutter, 2017) with a batch size of 256, an initial learning rate of 10^{-4}, linear warmup over the first 10% of training steps, and cosine learning-rate decay thereafter.

Expression predictor We implement Ψ as a dual-chain transformer that models heavy and light chain mRNA separately, then aggregates the two chain representations using Multiple Instance Learning (MIL) attention before a final linear layer predicts antibody expression. In practice, we found that reusing the pretrained language model encoder E as the backbone for Ψ improves the expression prediction. To enable this, we replace the first `nn.Embedding` layer of E with a dense `nn.Linear` layer to handle soft non-one-hot sequences. Once trained, Ψ is differentiable with respect to the soft-sequence inputs $\tilde{\mathbf{r}}_\tau$, enabling continuous optimization. Training proceeds for 120 epochs with Negative Log-Likelihood loss (for each member m of the ensemble) on the mAb expression (absorbance measured at 280 nm) data (Prakash et al., 2024; Yazdani-Jahromi et al., 2025). We use AdamW optimization with a batch size of 64, initial learning rate of 5×10^{-5}, and the same warmup and decay schedule as the language model. The Spearman's rank correlation between the predicted and ground truth expression on the test split reaches 0.65.

MFE predictor Analogously, the MFE predictor mirrors Ψ's architecture. It is trained for 50 epochs using MSE loss on 50,000 sub-sampled OAS mRNA sequences, with ground-truth MFE labels computed using the ViennaRNA package. Optimization follows identical settings to Ψ. The Spearman's rank correlation between the predicted and ground truth MFE on the test split reaches 0.91.

LSCO hyperparameters We determined the weights λ_Ψ and λ_{MFE} via grid search on each dataset to maximize predicted A280 while ensuring the optimized sequence's MFE is generally lower than the wild-type/original MFE, with flexibility to relax this constraint when predicted A280 ≤ 1. In practice, LSCO proved robust to a wide range of these weights. Grid search identified $\lambda_\Psi = 10$ and $\lambda_{\text{MFE}} = 0.1$ as effective. For decoding, we set the initial temperature $\tau_{\text{high}} = 0.5$ and the final temperature $\tau_{\text{low}} = 0.01$. These temperature parameters showed higher sensitivity, particularly τ_{low}; excessively high values caused significant rounding errors, reducing expression, while overly low values hindered optimization progress. We ran LSCO for $T = 3000$ iterations, observing that additional iterations improved convergence marginally. The learning rate was set to $\eta = 5 \times 10^{-3}$, and, leveraging the differentiability of the LSCO objective, we employed the Adam optimizer for robust gradient-based updates.

Decoding Temperature In practice, we also observed the importance of carefully selecting the decoding temperature τ. Too high temperature allows for broader exploration of the synonymous sequence space but can introduce significant rounding errors when mapping $\tilde{\mathbf{r}}_\tau$ to the nearest discrete mRNA sequence $\mathbf{r}$ via argmax. Conversely, too low temperature hinders exploration of the soft-sequence space by creating high-energy barriers between sequences. Empirically, we found that linearly annealing τ from a high to a low value during optimization strikes the best balance. This

AA	# Syn Codons	χ^2 p-value	Storey q-value	Distribution shift (initial → optimized)[1]		Shannon entropy (bits)[2]		
				EMD	SKL (bits)	H_{init}	H_{opt}	ΔH
R	6	1.33×10^{-20}	1.55×10^{-19}	0.533	2.653	1.925	1.935	+0.010
S	6	1.92×10^{-14}	1.12×10^{-13}	0.389	1.678	2.139	2.384	+0.244
P	4	3.94×10^{-10}	1.53×10^{-9}	0.369	1.267	1.649	1.948	+0.299
G	4	1.51×10^{-6}	2.94×10^{-6}	0.373	1.119	1.692	1.531	-0.161
N	2	8.41×10^{-3}	9.82×10^{-3}	0.334	0.699	0.798	0.983	+0.185
I	3	8.68×10^{-4}	1.13×10^{-3}	0.223	0.529	1.098	1.504	+0.406
Q	2	1.63×10^{-2}	1.73×10^{-2}	0.232	0.525	0.906	0.435	-0.471
A	4	2.10×10^{-7}	6.12×10^{-7}	0.242	0.469	1.579	1.370	-0.208
L	6	5.09×10^{-7}	1.19×10^{-6}	0.193	0.386	1.525	1.625	+0.100
V	4	1.65×10^{-4}	2.40×10^{-4}	0.193	0.341	1.403	1.494	+0.091
H	2	2.93×10^{-2}	2.63×10^{-2}	0.196	0.289	0.670	0.952	+0.282
C	2	2.37×10^{-2}	2.31×10^{-2}	0.199	0.236	0.996	0.923	-0.073
T	4	1.61×10^{-5}	2.68×10^{-5}	0.136	0.215	1.568	1.860	+0.292
E	2	4.40×10^{-1}	3.02×10^{-1}	0.144	0.165	0.904	0.670	-0.234
D	2	2.63×10^{-1}	2.20×10^{-1}	0.146	0.151	0.947	0.758	-0.188
Y	2	1.00	6.49×10^{-1}	0.048	0.018	0.827	0.746	-0.081
F	2	3.68×10^{-1}	2.76×10^{-1}	0.044	0.013	0.892	0.937	+0.045
K	2	3.78×10^{-1}	2.76×10^{-1}	0.034	0.009	0.778	0.832	+0.054

Table 3: **Codon-usage redistribution within synonymous families under LSCO–LCB+MFE.** For each amino-acid family, aggregated over the 54 optimized mAbs: the significance of the initial-to-optimized shift (chi-square p-value and Storey q-value); its magnitude (EMD and symmetrized KL divergence); and the change in codon diversity (Shannon entropy before and after optimization, $\Delta H = H_{\text{opt}} - H_{\text{init}}$). Rows are ordered by decreasing SKL. Families with the largest shifts (R, S, P, G) are highly significant, while E, D, Y, F and K are statistically indistinguishable from their original usage, indicating that redistribution is selective rather than uniform across the genetic code.

strategy allows the optimizer to first explore the domain freely to identify promising candidates and then commit to a single solution at the end of the process

B Additional Results and Experiments

B.1 Pretrained mRNA language models comparison

To evaluate whether pretrained mRNA language models learn biologically meaningful codon patterns, we compared four sequence generation strategies: random codon assignment (*Random*), codon sampling according to empirical codon frequencies (*Freq Sample*), an amino-acid-to-codon translation model (*CodonBERT*) pretrained on OAS sequences, and a codon-to-codon Variational Masked Auto-Encoder (*VMAE*) also pretrained on the OAS data. For VMAE generation, we include two variants, one that samples new sequences starting from the prior latent space (*VMAE – Prior*) and one that samples sequences starting from the encoded CodonBERT generated sequences (*VMAE – BERT*). All sequences are evaluated on Minimum Free Energy (MFE), GC content, Codon Adaptation Index (CAI), and codon-level Hamming distance from the reference starting sequence.

As shown in the bottom row of the main body's Figure 2, modern pretrained mRNA language models generate sequences with substantially improved biological properties than other baselines. CodonTransf, GEMORNA and VMAE all score favorably in terms of lower MFE, higher GC content (with GEMORNA falling just above the ideal host-specific GC range), higher CAI values, and lower codon Hamming distances. This indicates that modern transformer-based pretrained models efficiently learn context-dependent synonymous codon substitutions that improve quality without unnecessary sequence changes.

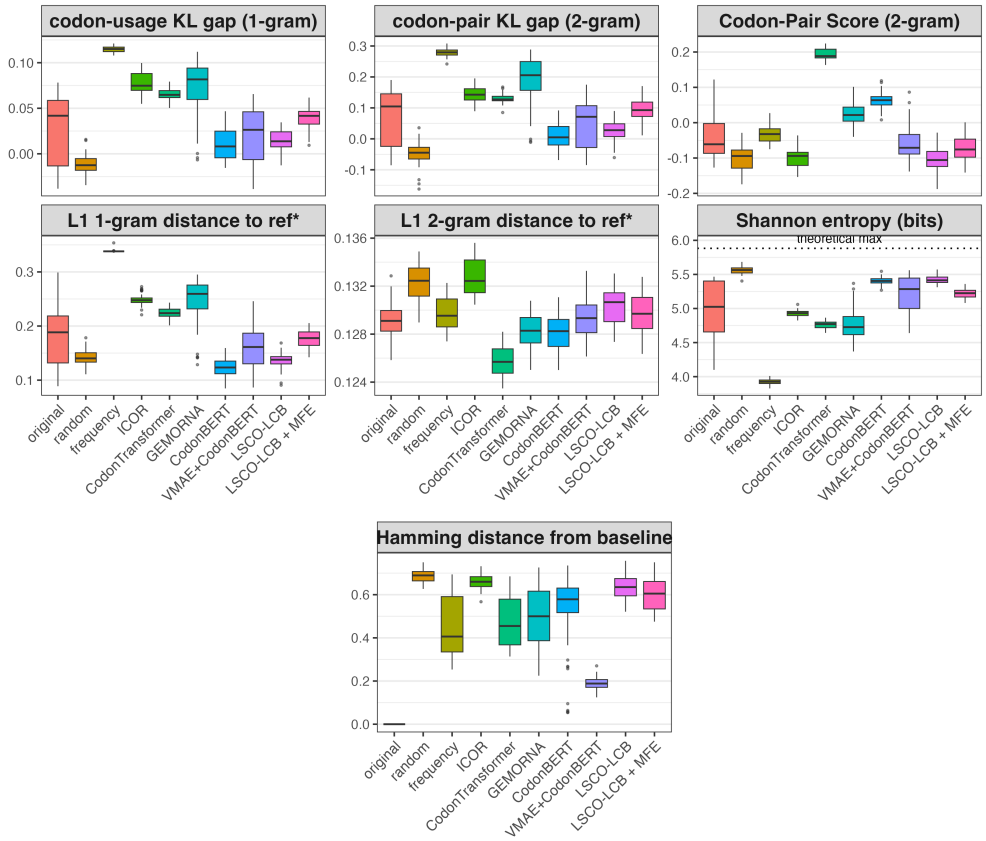


Figure 4: **Codon-usage descriptors and biophysical proxies across generation and optimization methods.** Medians over 54 mAbs (joint VH+VL); boxes span Q1–Q3. **Codon-bias targets:** Codon Adaptation Index (CAI), Codon-Pair Score, and the codon-usage KL gap toward high- vs. low-abundance proteins (1- and 2-gram). Codon generators drive these upward (CAI: frequency 1.000, CodonTransformer 0.960, GEMORNA 0.928; 1-gram KL gap: frequency 0.115, GEMORNA 0.082), whereas both LSCO variants stay at or below the original antibodies (CAI 0.892 $\rightarrow$ 0.816/0.840; 1-gram KL gap 0.042 $\rightarrow$ 0.014/0.042; 2-gram 0.105 $\rightarrow$ 0.028/0.093). **Distance to reference:** L_1 distance to a highly-translated CHO usage reference (Ang et al., 2016); LSCO remains near the original (0.188 $\rightarrow$ 0.138 for *LCB*, 0.178 for *LCB+MFE*; 2-gram flat at $\sim$0.130). **Uniformity:** Shannon entropy over the 59 non-degenerate codons (higher = more uniform; dotted line = theoretical maximum) increases under LSCO (5.025 $\rightarrow$ 5.414/5.225) but decreases for frequency (3.925), CodonTransformer (4.772) and GEMORNA (4.727). **Biophysical:** MFE (lower = more stable), GC% (shaded band = host-appropriate 50–60%), and codon Hamming distance from the original (Δ; lower = fewer synonymous edits).

B.2 Codon-Usage Descriptors of Optimized Sequences

Descriptors. To characterize *how* each method reshapes codon usage, beyond whether it improves predicted expression, we score every optimized sequence along four groups of descriptors, all com-

[1]EMD = earth-mover distance; on the unordered codon set with the discrete ground metric this equals total variation, $0.5 \cdot \Sigma|p-q|$ (range 0–1). SKL = symmetrized (Jeffreys) Kullback–Leibler divergence $KL(p||q) + KL(q||p)$, in bits. Both compare the initial codon distribution with the optimized one; 0 = no shift in overall codon proportions.

[2]$H = -\Sigma\ p \cdot \log_2(p)$ (range 0 to $\log_2(k)$, where k = codon-family size; 0 = one codon used exclusively, $\log_2(k)$ = uniform). $\Delta H = H_{opt} - H_{init}$; negative = lower entropy after optimization (more concentrated codon usage). The chi-square test already tests whether the codon distributions differ; if $p < 0.05$, their entropies differ significantly as well (entropy is a deterministic function of the distribution).

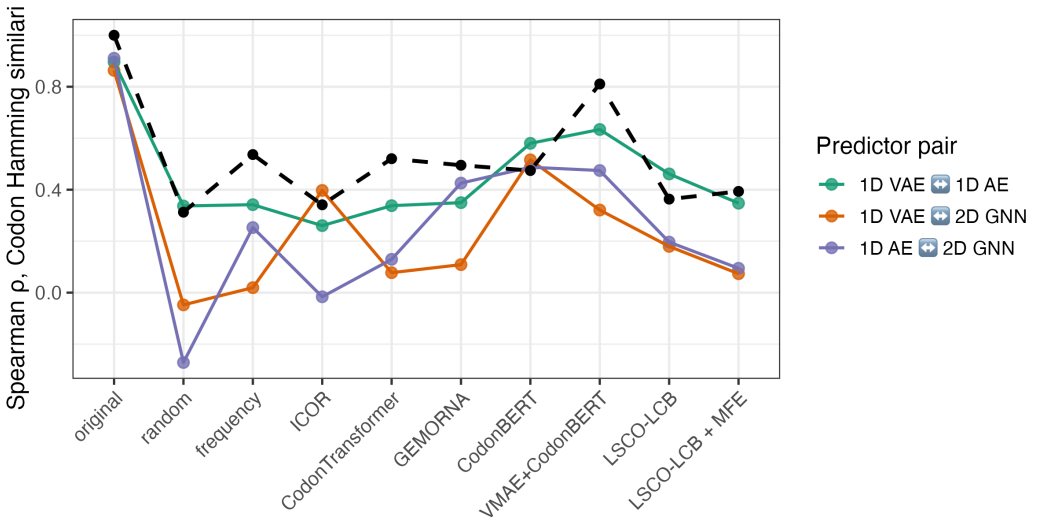


Figure 5: **Spearman rank correlation between A280 predictor pairs (colored lines) and codon Hamming similarity to original, computed separately for each method (54 mAbs per point).** Cross-predictor agreement was positively associated with codon conservation across methods (Spearman $\rho = 0.564$), although this did not reach significance at the method level ($p = 0.09, n = 10$).

puted on the joint heavy and light V-regions and reported as medians across the 54 mAbs. *(i) Codon-bias targets:* the Codon Adaptation Index (Sharp & Li, 1987), the Codon-Pair Score (2-gram), and the KL-divergence gap between codon usage in high- versus low-abundance proteins (1- and 2-gram). These are the quantities conventional codon optimizers explicitly maximize, so they measure the extent to which a method reproduces the classical highly-expressed-gene signature. *(ii) Distance to a host reference:* the L_1 distance of the codon (1-gram) and codon-pair (2-gram) distributions to a highly-translated Chinese hamster ovary (CHO) reference (Ang et al., 2016), measuring convergence toward canonical host usage. *(iii) Uniformity:* the Shannon entropy of the distribution over the 59 non-degenerate codons, which is maximal when synonymous codons are used evenly and falls as usage concentrates on a preferred subset. *(iv) Edit magnitude:* the codon Hamming distance from the original sequence, i.e. the number of synonymous substitutions made. We additionally resolve the shift within each synonymous family, by testing the initial-versus-optimized codon distribution per amino acid with a chi-square test (reporting Storey q-values to control the false discovery rate across the 18 multi-codon families), and quantifying its magnitude with the earth-mover distance (EMD, equal to total variation on the unordered codon set) and the symmetrized Jeffreys KL divergence (SKL).

LSCO learns a distinct and more balanced codon substitution strategy. Figure 4 shows that LSCO's expression gains are not obtained by rediscovering conventional codon-bias heuristics. Relative to the original antibodies, both LSCO variants *lower* the classical targets rather than raising them (CAI $0.892 \rightarrow 0.816$ for *LCB* and 0.840 for *LCB+MFE*; codon-pair score likewise decreases), in sharp contrast to frequency-based optimization and CodonTransformer, which drive CAI toward its ceiling (1.000 and 0.960 respectively) and push codon-pair scores positive. LSCO also avoids acquiring a peaked usage signature, keeping the high- versus low-abundance KL gap at or below the original in both the 1-gram ($0.042 \rightarrow 0.014/0.042$) and 2-gram ($0.105 \rightarrow 0.028/0.093$) settings, whereas frequency-based generation more than doubles it. Consistently, codon-usage uniformity *increases* under LSCO (entropy $5.025 \rightarrow 5.414/5.225$) while every generative and frequency-based baseline reduces it (3.925–4.772): LSCO spreads usage across synonymous codons rather than collapsing onto a favoured subset. With respect to the CHO reference, *LSCO–LCB* drifts slightly toward canonical host usage (L_1 $0.188 \rightarrow 0.138$), but this effect largely disappears once MFE regularization is added (0.178), and the 2-gram distance is essentially unchanged across variants.

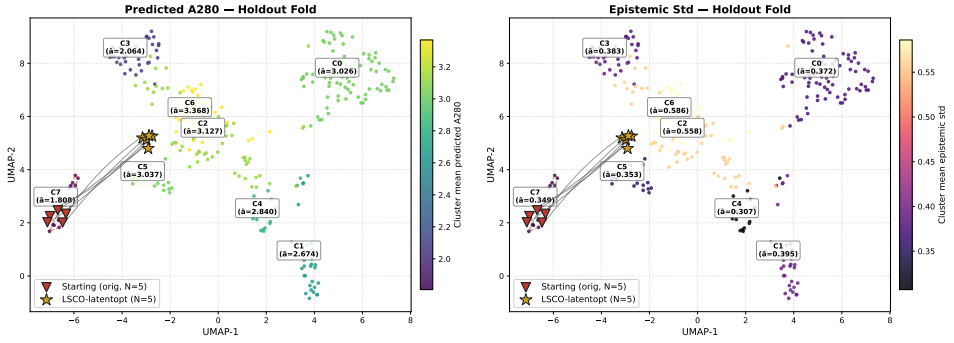


Figure 6: **LSCO optimization trajectories in latent space.** UMAP projection of the pretrained VMAE latent space on a holdout fold, coloured by predicted A280 (left) and by predictor epistemic uncertainty (right). Starting sequences (red triangles) lie in a low-expression cluster (C7, $\bar{a} = 1.808$); LSCO-optimized sequences (gold stars) are relocated toward high-expression clusters ($\bar{a} \approx 3.0$–3.4). Trajectories (grey arrows) follow a consistent direction and remain in low-uncertainty regions.

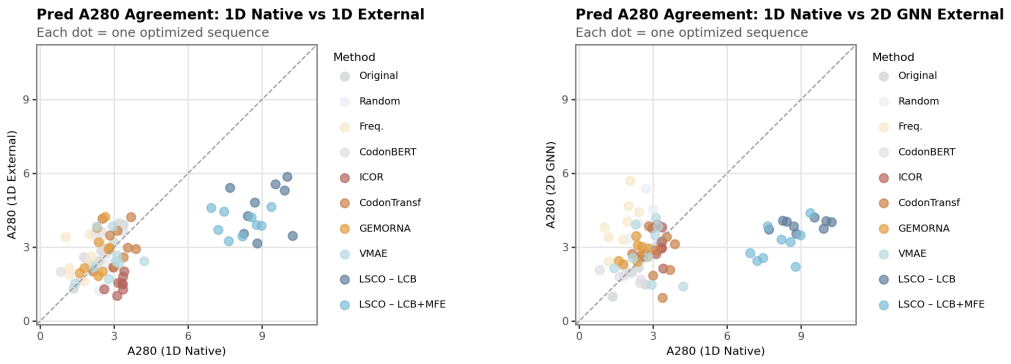
Redistribution is selective, not uniform. Table 3 resolves this picture within synonymous families. Optimization does not perturb the genetic code evenly: arginine, serine, proline and glycine undergo large and highly significant redistributions (SKL 1.12–2.65 bits, $q < 10^{-5}$), while lysine, phenylalanine, tyrosine, aspartate and glutamate are statistically indistinguishable from their original usage ($q > 0.2$, SKL < 0.17 bits). The entropy column shows that the direction of the shift is also family-specific: usage broadens for most families (e.g. isoleucine $\Delta H = +0.406$, proline $+0.299$, threonine $+0.292$) but concentrates for others (glutamine -0.471, glutamate -0.234, alanine -0.208). Taken together with the aggregate entropy increase above, this indicates that LSCO is not applying a single global rule such as "prefer GC-rich" or "prefer frequent" codons, but is making context-dependent choices whose net effect is a more balanced, less canonical codon composition than that produced by heuristic optimizers.

B.3 LSCO Optimization Trajectory

Figure 6 illustrates how LSCO reshapes the distribution of sequences in the learned latent space. The starting sequences (red triangles) reside in a cluster (C7) with low averaged predicted expression ($\bar{a} = 1.808$), whereas the LSCO-optimized sequences (gold stars) are consistently relocated toward clusters with substantially higher averaged predicted A280 ($\bar{a} \approx 3.0$–3.4). All optimization trajectories follow a similar direction, indicating that LSCO discovers coherent ascent paths on the underlying fitness landscape rather than arbitrary sequence perturbations. The companion uncertainty panel (right) shows that the optimized sequences stay in regions of low epistemic uncertainty, confirming that the LCB utility keeps the search in-distribution rather than exploiting the predictor off-manifold.

B.4 Expression Predictors and Agreement

1D Sequence-Based Predictors. Both 1D predictors follow the same dual-chain design: heavy and light chain mRNAs are encoded separately, each chain is reduced to a fixed-size representation, and the two are aggregated with a Multiple Instance Learning (MIL) attention module before a final linear layer predicts A280. The two variants then differ mainly in the encoder backbone. The *native* predictor (Ψ) reuses the pretrained VMAE encoder E as its backbone (with the first embedding layer replaced by a dense layer so it accepts the relaxed, soft-sequence input $\tilde{\mathbf{r}}_\tau$); this is the model whose gradients drive LSCO. The *external* 1D predictor keeps similar dual-chain + MIL head but swaps in a different, shallower autoencoder backbone that never takes part in optimization. It therefore provides an independent read-out that shares the 1D sequence view but not the representation LSCO is optimized against.



(a) Native 1D vs External 1D

(b) Native 1D vs External 2D GNN.

Figure 7: **Predicted-A280 agreement across evaluators.** Each dot is one optimized sequence, coloured by method; the dashed line marks perfect agreement. The horizontal axis is the native 1D predictor Ψ; the vertical axis is an independent external predictor withheld from optimization, the 1D Shallow-AE model (a) and the structure-aware 2D GNN (b). LSCO variants (blue) are rated highest by the native model, but score favorably also under the two external models in terms of ordering/ranking preserving.

2D Structure-Based Predictor. The external "2D GNN" predictor takes a genuinely different view of each sequence. For each chain, the mRNA is folded and its dot-bracket secondary structure is converted into a graph, with one-hot nucleotide node features and edges from both the backbone and the predicted base pairs. A graph neural network encodes this structure, and the same MIL-attention aggregation combines the two chains before the regression head. Because it consumes secondary-structure graphs rather than raw sequence embeddings, the 2D GNN carries a structural inductive bias absent from both 1D models, making it the most independent of the three evaluators.

Cross-predictor agreement. Figure 7 plots, for every optimized sequence, the native 1D A280 against each external predictor. In both panels the LSCO points (blue) sit clearly at the center-right, separated from all baselines: this implies they are rated highest by the native predictor and rated comparably or slightly higher by the external 1D and the structure-aware 2D GNN. The external predictors are more conservative in absolute terms in that most points lie below the diagonal, so agreement is more on *ranking* rather than exact value prediction. Overall ordering is more or less preserved across all three, despite distinct structural views. This consistency across predictors with different backbones and inductive biases presents some evidence that the gains are not an artifact of Ψ being exploited off-distribution, but reflect genuinely higher predicted expression.

B.5 Deep-Ensemble and Epistemic Uncertainty Aware Objective

Probabilistic Deep Ensemble. We implement the expression predictor Ψ as a deep ensemble of $M = 10$ *probabilistic* members (Lakshminarayanan et al., 2017). Each member m does not output a point estimate but a full Gaussian predictive distribution over A280:

$$p_m(y \mid \tilde{\mathbf{r}}_\tau) = \mathcal{N}\big(y;\, \mu_m(\tilde{\mathbf{r}}_\tau),\, \sigma_m^2(\tilde{\mathbf{r}}_\tau)\big)\,, \tag{5}$$

where the head outputs both a mean μ_m and a variance σ_m^2. Members are trained *independently*, each with its own random initialization, optimizer, and backward pass, by minimizing the Gaussian negative log-likelihood loss:

$$\mathcal{L}_{\text{NLL}}^{(m)} = \tfrac{1}{2}\log \sigma_m^2(\tilde{\mathbf{r}}_\tau) + \frac{(y - \mu_m(\tilde{\mathbf{r}}_\tau))^2}{2\,\sigma_m^2(\tilde{\mathbf{r}}_\tau)}, \tag{6}$$

so that the predicted variance is calibrated to each member's own residuals. The variance head itself is configurable (constant, scalar, per-group, or fully heteroscedastic); we use the scalar σ_m^2 setting, as we noticed data exhibit somehow uniform noise.

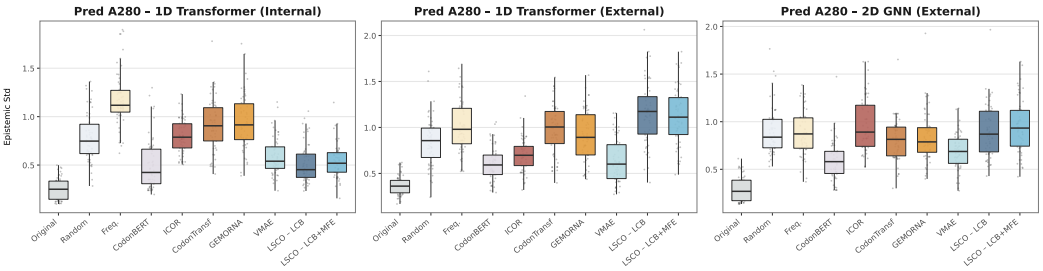


Figure 8: **Epistemic uncertainty of the optimized sequences.** Epistemic standard deviation (across-member disagreement, $\sqrt{\text{epistemic}}$) of the three probabilistic deep-ensemble predictors — 1D Transformer Internal/Native (left), 1D Transformer External (middle), 2D GNN External (right), evaluated on the same 54 optimized mAbs as Figure 2; each dot is one mAb. Under the native predictor, LSCO variants (blue) show low, concentrated epistemic uncertainty, comparable to the unoptimized VMAE and well below the other optimizers, confirming the LCB utility keeps the search in-distribution. Under the external predictors — never seen during optimization — LSCO's uncertainty is comparable to other high-expression methods, as expected for held-out evaluators.

Aleatoric/Epistemic Uncertainty Decomposition. Treating the ensemble as a uniformly-weighted Gaussian mixture, the ensemble mean and total predictive variance are

$$\mu_\Psi(\tilde{\mathbf{r}}_\tau) = \frac{1}{M}\sum_{m=1}^{M} \mu_m(\tilde{\mathbf{r}}_\tau), \tag{7}$$

$$\sigma^2_\Psi(\tilde{\mathbf{r}}_\tau) = \underbrace{\frac{1}{M}\sum_{m=1}^{M} \sigma^2_m(\tilde{\mathbf{r}}_\tau)}_{\text{aleatoric}} + \underbrace{\frac{1}{M}\sum_{m=1}^{M} \big(\mu_m(\tilde{\mathbf{r}}_\tau) - \mu_\Psi(\tilde{\mathbf{r}}_\tau)\big)^2}_{\text{epistemic}}. \tag{8}$$

The two terms have distinct meanings. The *aleatoric* term is the average of the members' own predicted variances and captures irreducible noise in the measurement/data. The *epistemic* term is the variance *across* the members' means and captures model uncertainty: it is large where the members disagree, which happens precisely in regions of sequence space that are under-represented in the training data. Because epistemic uncertainty shrinks as designs stay close to the training distribution and grows off-distribution, it is exactly the signal the LCB utility $U_\Psi = \mu_\Psi - \lambda_{\text{LCB}}\,\sigma_\Psi$ (Eq. 3) exploits: penalizing σ_Ψ steers the optimizer away from over-optimistic, out-of-distribution sequences. The same probabilistic ensemble construction is used for all three predictors (1D native, 1D external, 2D GNN); the 2D GNN members simply own an independent structure-graph backbone each.

Epistemic uncertainty of the optimized sequences. Figure 8 reports the epistemic standard deviation ($\sqrt{\text{epistemic}}$) of each predictor on the same 9 optimized mAbs as Figure 2. Two patterns stand out. First, under the *native* predictor (left), the LSCO variants sit at low and tightly-concentrated epistemic uncertainty, comparable to the unoptimized VMAE sequences and well below frequency-based, ICOR, CodonTransformer, and GEMORNA. This confirms that the LCB penalty keeps the search in-distribution rather than chasing high-variance designs. Second, under the two *external* predictors (middle, right), LSCO's epistemic uncertainty rises to a level similar to the other high-expression methods. This is because the external models were never optimized against, so any strong-expression design is somewhat novel to them, and LSCO is no more off-distribution for them than the competing optimizers. Read together with Figure 2, LSCO therefore attains the *highest* predicted A280 across all three predictors while keeping the *lowest* epistemic uncertainty under the predictor that actually guides optimization, i.e., its gains do not come from exploiting high-uncertainty regions of its own surrogate.

		Predicted A280			Metrics (VH + VL)		
λ_{MFE}	λ_{LCB}	1D Transf (Internal)	1D Transf (External)	2D GNN	MFE	GC%	CAI
0	0	$\mathbf{9.585 \pm 0.218}$	$\mathbf{4.518 \pm 0.223}$	$\mathbf{3.492 \pm 0.203}$	-235.187 ± 3.555	53.303 ± 0.423	0.816 ± 0.008
	1	9.156 ± 0.241	4.400 ± 0.254	3.485 ± 0.212	-236.711 ± 3.603	53.601 ± 0.390	0.815 ± 0.009
0.1	0	8.180 ± 0.237	3.900 ± 0.224	3.175 ± 0.217	-266.274 ± 2.783	57.756 ± 0.285	0.830 ± 0.009
	1	7.858 ± 0.235	3.935 ± 0.265	3.099 ± 0.187	-267.050 ± 3.211	57.916 ± 0.300	0.837 ± 0.008
1	0	8.065 ± 0.232	3.887 ± 0.200	3.319 ± 0.201	-263.765 ± 3.122	58.003 ± 0.304	0.825 ± 0.009
	1	7.899 ± 0.256	3.951 ± 0.271	2.923 ± 0.182	-266.365 ± 3.381	58.018 ± 0.317	$\mathbf{0.838 \pm 0.008}$
5	0	7.921 ± 0.260	3.729 ± 0.220	3.057 ± 0.220	-265.006 ± 3.443	57.959 ± 0.307	0.824 ± 0.009
	1	7.635 ± 0.271	3.852 ± 0.228	3.161 ± 0.249	-263.954 ± 3.377	57.973 ± 0.321	0.832 ± 0.008
10	0	7.490 ± 0.248	3.514 ± 0.263	3.089 ± 0.248	$\mathbf{-267.381 \pm 3.033}$	58.494 ± 0.282	0.818 ± 0.008
	1	7.390 ± 0.264	3.500 ± 0.256	3.169 ± 0.241	-265.220 ± 2.735	58.233 ± 0.258	0.829 ± 0.009

Table 4: **MFE sensitivity analysis for LSCO variants.** Predicted A280 values from three surrogate models (1D Transformer (Internal), 1D Transformer (External), 2D GNN) and biophysical metrics (minimum free energy, GC content, codon adaptation index) reported for the joint heavy + light chain construct on the 54 starting sequences. The second row ($\lambda_{MFE} = 0, \lambda_{LCB} = 1$) is the LCB-only LSCO baseline ('LSCO – LCB' in Figure 2). The fourth row ($\lambda_{MFE} = 0.1, \lambda_{LCB} = 1$) is the LCB+MFE baseline ('LSCO – LCB + MFE' in Figure 2) that best balances A280 and MFE objectives. **Bold** indicates the best value per column: highest for A280 predictions and CAI, most negative for MFE. GC% is not bolded as "best" is target-dependent. Values are mean $\pm$ 95% confidence bands across starting sequences.

B.6 MFE REGULARIZATION SENSITIVITY

Table 4 reports the full sweep over the MFE weight λ_{MFE} and the uncertainty-penalty toggle λ_{LCB}, extending the two settings shown in the main text (Figure 2). Increasing λ_{MFE} from 0 drives MFE substantially lower and lifts GC and CAI into the host-appropriate range, at a modest cost in predicted A280; gains in stability saturate beyond $\lambda_{\text{MFE}} = 0.1$, confirming it as the sweet spot used throughout.

B.7 SEED SENSITIVITY STUDY

To assess how much results depend on the random seed for *stochastic* codon optimization methods, we fix a single representative starting mAb (chosen near the median A280 of the dataset) and re-run each stochastic method across 10 seeds. Since the evaluators are deterministic given a sequence, the spread in Figure 9 reflects methods' stochasticity alone. Deterministic methods (original, natural, frequency, ICOR) are seed-inert and therefore omitted from the boxes. We report both LSCO variants (*LSCO – LCB*, *LSCO – LCB + MFE*) alongside the stochastic baselines CodonTransformer (temperature 0.4) and GEMORNA, plus the LSCO random-substitution baseline.

Across seeds, the results in terms of performance generally hold with some caveats. Under the two 1D predictors, both LSCO variants stay above the stochastic baselines on predicted A280 for every seed, with bounded spread; *LCB* variant is tightest, while *LCB + MFE* shows slightly more variability and slightly lower absolute A280. Under the more conservative and less aligned 2D GNN, the boxes overlap more and seed spread is larger for all methods. Biophysical metrics are stable: both LSCO variants keep GC within the 50–60% band on every seed, and *LCB + MFE* consistently yields lower MFE than *LCB*. These results are limited to a single starting mAb and characterize generator stochasticity rather than variation across antibodies.

C LSCO VS. DIRECT INPUT-SPACE OPTIMIZATION (DISO)

DISO is an ablation of LSCO designed to isolate the contribution of the pretrained VMAE mRNA language model. The two methods share everything except *where* the optimization happens: DISO reuses the exact same expression-predictor ensemble Ψ, MFE head, tokenizers, valid-codon mask, and candidate-selection gate as LSCO. The only thing that changes is the set of variables being optimized, and, as a direct consequence, how the naturalness prior is expressed. The empirical outcome of this comparison is reported in the main text (Table 2): LSCO generally matches or exceeds DISO across evaluators, confirming that the pretrained latent space, and not merely the shared components, drives the gains.

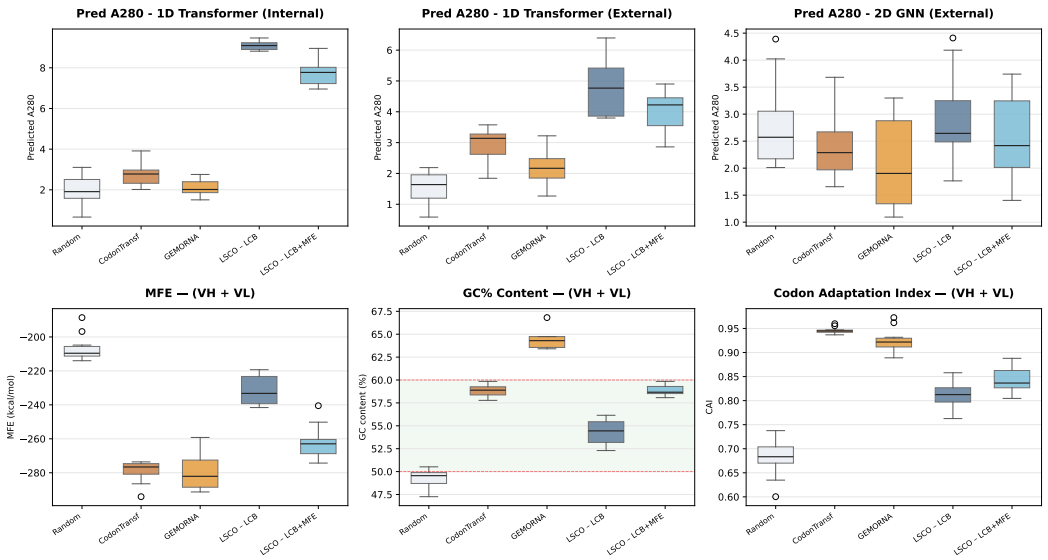


Figure 9: **Seed sensitivity on a single mAb (10 seeds).** Boxes show the spread over different 10 seeded runs, reflecting the stochasticity behind different codon optimization methods. *Top:* predicted A280 from the native 1D transformer (internal), an external 1D transformer, and an external 2D GNN. *Bottom:* biophysical metrics for the joint heavy+light construct (VH+VL): MFE (lower = more stable), GC% (shaded band = host-appropriate 50–60%), and CAI. Both LSCO variants remain above the stochastic baselines on the 1D predictors across all seeds with bounded spread, keep GC within the admissible band, and *LSCO – LCB + MFE* consistently lowers MFE relative to *LSCO – LCB*; seed spread is largest under the more conservative 2D GNN evaluator.

What is being optimized (search geometry). LSCO optimizes a continuous latent code z in the VMAE latent space, and turns it into a codon sequence by passing it through the pretrained decoder D_τ at every step. DISO removes the decoder from the loop entirely: it optimizes one set of *per-position codon logits* directly in sequence space, one logit vector for each position of the heavy and light chains. In other words, LSCO moves through the smooth, learned manifold of the language model, while DISO moves through the raw codon-logit space with no generative model shaping the trajectory. Everything downstream — the temperature-controlled softmax, the valid-codon masking that keeps the sequence translating back to the target protein, the expression and MFE predictors — is identical.

Naturalness prior: latent anchoring vs. distribution matching. This is the component that changes most, and it changes because it *has* to. In LSCO the naturalness prior lives in the latent space: we encode the BERT back-translated "natural" sequence to a reference code z_{nat} and pull the current code toward it with a squared-distance penalty, $U_{\text{nat}}(z, z_{\text{nat}}) = -\|z - z_{\text{nat}}\|^2$. DISO has no latent space, so this anchor cannot be defined the same way. Instead, DISO anchors the sequence directly in codon space: it builds a fixed per-position categorical distribution from the BERT-natural sequence (using a fixed reference temperature $\tau_{\text{BERT}} = 0.5$) and, at each step, penalizes the Kullback–Leibler divergence between the current per-position codon distribution and this fixed reference, $\text{KL}(p_{\text{current}} \,\|\, p_{\text{BERT}})$. For a two-chain antibody the term is the average of the heavy- and light-chain KL divergences. Conceptually both do the same job — keep candidates close to biologically plausible codon usage — but LSCO measures "closeness" as a distance between latent codes, whereas DISO measures it as a distance between probability distributions over codons. Setting $\lambda_{\text{nat}} = 0$ removes the anchor in both methods.

Initialization. LSCO initializes the latent code z from a standard Gaussian and lets the decoder map it to a starting sequence. DISO offers two starting points. The default (`init_mode=bert`) warm-starts the logits at the BERT-natural codon for each position (a logit of 0 for the natural codon and $-\delta$ for the others, with $\delta = 3.0$), so DISO begins from a natural sequence. The "fairer-ablation"

	LSCO	**DISO**
Optimized variable	latent code z	per-position codon logits
Generative decoder in loop	Yes (D_τ)	No
Naturalness anchor	latent ℓ^2 distance $\|z - z_{\text{nat}}\|^2$	KL to fixed BERT distribution
Naturalness weight λ_{nat}	10^{-2} (0 to disable)	10^{-2} (0 to disable)
Initialization	$z \sim \mathcal{N}(0, I)$	BERT-natural warm start (or $\mathcal{N}(0, 0.5^2)$)
Temperature schedule τ	$0.5 \to 0.03$	$0.5 \to 0.03$
Optimization steps	3000	3000
Learning rate	5×10^{-3}	5×10^{-3}
MFE weight λ_{MFE}	$\{0, 0.1, 1\}$	$\{0, 0.1, 1\}$
Optimizer	AdamW	AdamW

Table 5: Configuration differences between LSCO and DISO. All model components (expression ensemble, MFE head, tokenizers, valid-codon mask, candidate gate) are shared; only the search geometry and its downstream consequences differ.

variant (`init_mode=random_gauss`) instead draws logits from $\mathcal{N}(0, \sigma^2)$ with $\sigma = 0.5$ and no bias toward the natural codon, immediately suppressing invalid codons via the valid-codon mask so optimization still starts from a protein-consistent point. Importantly, the naturalness anchor target is *always* the BERT-natural distribution regardless of where the logits are initialized — initialization only decides the starting point, not the anchor.

Optimization schedule and other hyperparameters. To keep the comparison as clean as possible, both methods share the same optimization budget and schedule: an identical decoding-temperature anneal from a high value (broad exploration) down to a low value (commit to a single codon per position), the same ramp on the constraint that enforces protein consistency, the same number of steps, learning rate, optimizer, and range of MFE weights. This is deliberate — because DISO differs from LSCO only in the search geometry (raw codon logits vs. the pretrained latent code) and the induced form of the naturalness anchor, matching all remaining settings ensures that any performance gap is attributable to those two factors rather than to differing training budgets. Table 5 lists the settings for both.