JOURNAL OF COMPUTATIONAL BIOLOGY
Volume 31, Number 3, 2024

Pp. 1–18
DOI: 10.1089/cmb.2023.0283

Research Article

DERNA Enables Pareto Optimal RNA Design

XINYU GU,[1] YUANYUAN QI,[1] and MOHAMMED EL-KEBIR[1,2]

ABSTRACT

The design of an RNA sequence v that encodes an input target protein sequence w is a crucial aspect of messenger RNA (mRNA) vaccine development. There are an exponential number of possible RNA sequences for a single target protein due to codon degeneracy. These potential RNA sequences can assume various secondary structure conformations, each with distinct minimum free energy (MFE), impacting thermodynamic stability and mRNA half-life. Furthermore, the presence of species-specific codon usage bias, quantified by the codon adaptation index (CAI), plays a vital role in translation efficiency. While earlier studies focused on optimizing either MFE or CAI, recent research has underscored the advantages of simultaneously optimizing both objectives. However, optimizing one objective comes at the expense of the other. In this work, we present the Pareto Optimal RNA Design problem, aiming to identify the set of Pareto optimal solutions for which no alternative solutions exist that exhibit better MFE and CAI values. Our algorithm DEsign RNA (DERNA) uses the weighted sum method to enumerate the Pareto front by optimizing convex combinations of both objectives. We use dynamic programming to solve each convex combination in $\mathcal{O}(|\mathbf{w}|^3)$ time and $\mathcal{O}(|\mathbf{w}|^2)$ space. Compared with a CDSfold, previous approach that only optimizes MFE, we show on a benchmark data set that DERNA obtains solutions with identical MFE but superior CAI. Moreover, we show that DERNA matches the performance in terms of solution quality of LinearDesign, a recent approach that similarly seeks to balance MFE and CAI. We conclude by demonstrating our method's potential for mRNA vaccine design for the SARS-CoV-2 spike protein.

Keywords: dynamic programming, multiobjective optimization, reverse translation and mRNA vaccine design, RNA sequence design.

1. INTRODUCTION

With the emergence of the COVID-19 pandemic, messenger RNA (mRNA) vaccines have garnered significant attention due to their effectiveness in combating the disease (Mahase, 2020; Meo et al., 2021). However, there remain significant challenges in the delivery (Crommelin et al., 2021) as well as in

[1]Department of Computer Science and [2]Cancer Center at Illinois, University of Illinois Urbana-Champaign, Urbana, Illinois, USA.

An earlier version of this article was published in WABI 2023 (doi: 10.4230/LIPIcs.WABI.2023.21).

the in vitro and in vivo stability of mRNA-based vaccines and therapeutics (Wayment-Steele et al., 2021). Importantly, due to codon degeneracy with $4^3 = 64$ codons encoding for 20 distinct amino acids as well as translation termination signals, there are exponentially many RNA sequences $\mathbf{v}$ for a single target protein $\mathbf{w}$. Synonymous codon choice impacts translational efficiency and mRNA stability in two interrelated ways. First, a subset of ''optimal'' codons occur at a higher frequency in highly expressed genes (Gustafsson et al., 2004) and ''nonoptimal'' codons lead to increased ribosomal pausing and decreased mRNA half-life (Presnyak et al., 2015; Weinberg et al., 2016).

Second, depending on codon choice, each candidate RNA sequence folds into a distinct *secondary structure* or conformation, affecting its thermodynamic stability and consequently mRNA half-life. Recent studies have demonstrated the importance of both factors, showing that optimizing one factor at the expense of the other leads to degraded protein expression (Tuller et al., 2010; Mauger et al., 2019). This leads to the following key question of this article. How does one identify RNA sequences that optimize both criteria?

Different organisms and even different genes within the same organism can have distinct codon usage patterns. The *codon adaptation index* (CAI) is a measure that quantifies the degree of codon usage bias in a protein coding sequence relative to a reference set of highly expressed genes (Sharp and Li, 1987). The reference set is often chosen based on the assumption that these genes have evolved to use codons that are mostly efficiently translated by the ribosome. Thus, an RNA sequence with high CAI is expected to have higher rates of translation (Gustafsson et al., 2004; Presnyak et al., 2015; Weinberg et al., 2016).

Specifically, for a reference gene set, we are given the relative frequencies $g(\mathbf{x})$ of each codon $\mathbf{x}$ in the gene set. Then, the CAI of an RNA sequence $\mathbf{v}$ is the geometric mean of the ratios $g(\mathbf{x})/\max_{\mathbf{y}\in S(\mathbf{x})} g(\mathbf{y})$ of each codon $\mathbf{x}$ versus the maximum relative frequency of a synonymous codon $\mathbf{y} \in S(\mathbf{x})$ (see Eq. (1)). RNA sequences that are composed of only optimal codons with maximum relative frequencies have by definition a CAI of 1. In our setting, it is trivial to identify such an RNA sequence with a CAI equal to 1 by simply choosing the codon with maximum relative frequency for each amino acid of the target protein. However, such an RNA sequence with an optimal CAI may exhibit reduced secondary structure (Fig. 1), and ultimately decreased protein expression (Tuller et al., 2010; Mauger et al., 2019).

RNA molecules adopt a secondary structure and three-dimensional conformations as the nucleotides within the RNA molecule and the surrounding solvent interact with each other. When an RNA molecule folds into its conformation, it forms base-pairing interactions between nucleotides that result in the lowest possible free energy (Freier et al., 1986). This conformation is said to have the *minimum free energy* (MFE). In general, an RNA molecule with a lower MFE is more likely to be stable and maintain its integrity over time, whereas an RNA molecule with a higher MFE is more likely to be degraded. Thermodynamic stability is an important factor in identifying the most stable RNA sequences that are likely to be functional and efficient in producing a target protein (Tuller et al., 2010; Mauger et al., 2019).

Zuker and Stiegler (1981) introduced a dynamic programming algorithm to identify the conformation of RNA molecules with MFE from a given RNA sequence $\mathbf{v}$. This approach was later extended independently by Terai et al. (2016) and Cohen and Skiena (2003) to identify an RNA sequence $\mathbf{v}$ and a corresponding secondary structure with overall minimum MFE for a given target protein sequence $\mathbf{w}$. However, an RNA sequence with optimal MFE may have lower CAI values (Fig. 1), leading to diminished protein expression (Tuller et al., 2010; Mauger et al., 2019). Recognizing the importance of examining both CAI and MFE, Zhang et al. (2023) introduced LinearDesign, which uses stochastic context-free grammars and deterministic finite automata. The article describes both an exact algorithm and a beam search heuristic to optimize $\text{MFE} + \lambda_{\text{LD}} \log \text{CAI}$ where λ_{LD} is a user-specified parameter.

In this work, we model the trade-off between CAI and MFE as a multiobjective optimization problem. Motivated by Mauger et al. (2019), showing the importance of sampling a wide range of solutions that achieve different trade-offs between CAI and MFE, we introduce the Pareto Optimal RNA Design (PORD) problem. In this problem, we seek the set of *Pareto optimal solutions* for which no other solution exists that is better in terms of both MFE and CAI (Fig. 1). We use the weighted sum method (Zadeh, 1963) to enumerate the Pareto front by optimizing convex combinations of both objectives—leading to the Balanced RNA Design (BRD) problem (Fig. 1). Our resulting algorithm, DEsign RNA (DERNA), extends the Zuker and Stiegler (1981) dynamic programming recurrence to support codon selection for each input amino acid as well as incorporate CAI in the objective function in addition to MFE.

We solve each convex combination of the two objectives in $\mathcal{O}(|\mathbf{w}|^3)$ time and $\mathcal{O}(|\mathbf{w}|^2)$ space. Unlike LinearDesign, where key functions are closed source, DERNA is fully open source with all code and functionality available to the user under a permissive license. We show on a benchmark data set that

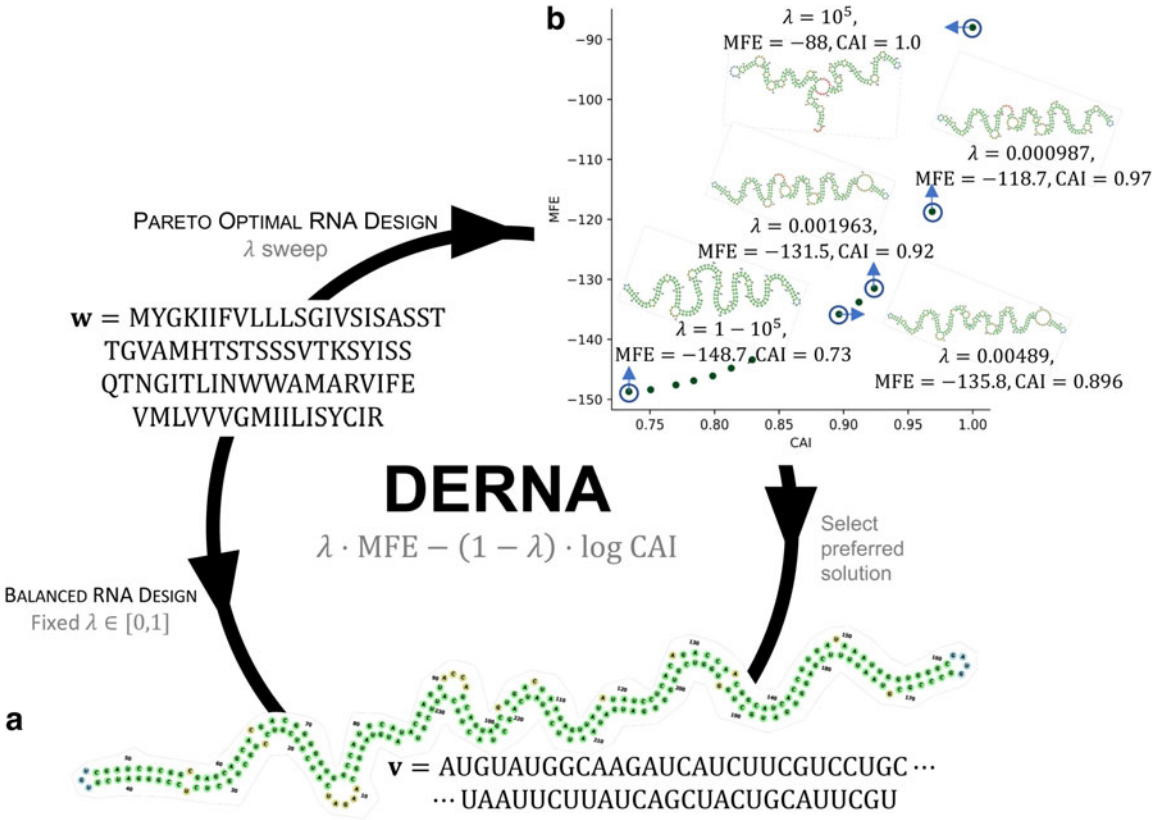

FIG. 1. DERNA seeks Pareto optimal RNA sequences **v** for a target protein **w**, balancing the MFE and CAI. **(a)** For the BRD problem, DERNA takes as input the parameter $\lambda \in [0, 1]$ and returns the RNA sequence **v** whose corresponding secondary structure P minimizes $\lambda \cdot \text{MFE}(\mathbf{v}, P) - (1-\lambda) \cdot \overline{\text{CAI}}(\mathbf{v}, \mathbf{w})$. **(b)** For the PARETO OPTIMAL RNA DESIGN problem, DERNA performs a systematic sweep on λ, solving multiple BRD instances and returning a set of Pareto optimal solutions $(\mathbf{v}, P)$. BRD, BALANCED RNA DESIGN; CAI, codon adaptation index; DERNA, DEsign RNA; MFE, minimum free energy.

DERNA obtains solutions with identical MFE but superior CAI compared with CDSfold (Terai et al., 2016). In addition, we show that DERNA matches LinearDesign's performance in terms of solution quality. Finally, we run our method on the SARS-CoV-2 spike protein and demonstrate its potential for mRNA vaccine design.

2. PROBLEM STATEMENT

A secondary structure for an RNA sequence with length n is a set of ordered base pairings $(i, j) \in [n] \times [n]$ such that each base is paired with at most one other base and there are no crossings base pairings (also known as pseudoknots). More formally, we define a secondary structure as follows.

Definition 2.1. *A set $P \subseteq [n] \times [n]$ of base pairings is a* secondary structure *provided (i) for each base pairing $(i, j) \in P$ it holds that $i < j$, and for any two base pairings $(i, j), (i', j') \in P$ it holds that (ii) $i = i'$ if and only if $j = j'$ and (iii) if $i < i' < j$ then $i < i' < j' < j$.*

Following Zuker and Stiegler (1981), a secondary structure P can be decomposed into several secondary structure elements such that the free energy of the secondary structure P is the sum of the free energies contributed by each secondary structure element, defined as follows.

Definition 2.2. *A subset $P' = \{i, p_1, q_1, \ldots, p_k, q_k, j\} \subseteq [n]$ of bases is a* secondary structure element *of P provided (i) $i < p_1 < q_1 < \ldots < p_k < q_k < j$, (ii) $(i, j) \in P$, (iii) $(p_l, q_l) \in P$ for each $l \in [k]$ and (iv) there exists no base pairing $(i', j') \in P$ such that $i < i', j' < p_1 < j$; $i < q_k < i', j' < j$; or $q_l < i', j' < p_{l+1}$ for all $l \in \{1, \ldots, k-1\}$.*

The conditions in the above definition ensure that each secondary structure element P' corresponds to a unique face of a planar embedding of the secondary structure. Each base pairing (i,j) induces a unique secondary structure element $P'=\{i,p_1,q_1,\ \ldots,p_k,q_k,j\}$ that can be classified into one of five types. First, if (i,j) is immediately followed by pairing $(i+1,j-1)$ then P' is a stacking element (Fig. 2a). Second, if no other pairings exist involving bases $\{i+1,\ \ldots,j-1\}$ then P' is a hairpin loop (Fig. 2b). Third, if P' has exactly one enclosing base pairing, that is, $k=1$, and only one end of this pairing is contiguous with (i,j) then P' is a bulge loop (Fig. 2c). Fourth, if both ends of the one enclosing base pairing of P' are noncontiguous with (i,j) then P' is an internal loop (Fig. 2d). Fifth, if P' contains more than one enclosing base pairing, that is, $k>1$, then P' is a multibranch loop (Fig. 2e). We refer to Supplementary Data SA.1 for more precise definitions.

As mentioned, we define the MFE $\mathrm{MFE}(\mathbf{v},P)=\sum_{(i,j)\in P}\mathrm{MFE}(\mathbf{v},P,(i,j))$ of an RNA sequence $\mathbf{v}$ as the sum of the MFEs $\mathrm{MFE}(\mathbf{v},P,(i,j))$ of the secondary structure elements induced by each base pairing $(i,j)\in P$.

Definition 2.3. *The* MFE $\mathrm{MFE}(\mathbf{v},P)$ *of secondary structure* P *of RNA sequence* $\mathbf{v}$ *equals* $\sum_{(i,j)\in P}\mathrm{MFE}(\mathbf{v},P,(i,j))$ *where* $\mathrm{MFE}(\mathbf{v},P,(i,j))$ *is the contribution of the secondary structure element induced by a base pairing* $(i,j)\in P$ *defined as*

$$\mathrm{MFE}(\mathbf{v},P,(i,j)) = \begin{cases} f_{\mathrm{s}}(\mathbf{v}(P')), & \text{if } P'=\{i,i+1,j-1,j\} \text{ is a stacking element},\\ f_{\mathrm{h}}(\mathbf{v}(P')), & \text{if } P'=\{i,j\} \text{ is a hairpin},\\ f_{\mathrm{b}}(\mathbf{v}(P')), & \text{if } P'=\{i,p_1,q_1,j\} \text{ is a bulge loop},\\ f_{\mathrm{i}}(\mathbf{v}(P')), & \text{if } P'=\{i,p_1,q_1,j\} \text{ is an internal loop},\\ f_{\mathrm{m}}(\mathbf{v}(P')), & \text{if } P'=\{i,p_1,q_1,\ \ldots,p_k,q_k,j\} \text{ is a multi-branch loop}. \end{cases}$$

The actual definitions of f_s,f_h,f_b,f_i and f_m depend on the used energy model. Briefly, in the widely used Turner energy model (Turner et al., 1988), the stacking energy value f_s is computed using a lookup table indexed by the four nucleotides comprising the base pairings $(i,j),(i+1,j-1)$. Similarly, the hairpin energy value f_h is a function of the four nucleotides v_i,v_{i+1},v_{j-1},v_j and the length $j-i+1$ of the hairpin loop. For a bulge loop, the energy value f_b is a function of the four nucleotides in the base pairings $(i,j),(p_1,q_1)$ and the number of unpaired nucleotides in the loop $\mathbf{v}(\{i,p_1,q_1,j\})$. For an internal loop, the energy value f_h is a function of the eight nucleotides $v_i,v_{i+1},v_{p_1-1},v_{p_1},v_{q_1},v_{q_1+1},\ v_{j-1},v_j$ surrounding the base pairings $(i,j),(p_1,q_1)$ as well as the number of unpaired nucleotides in the loop $\mathbf{v}(\{i,p_1,q_1,j\})$. Finally, the energy value f_m is a function of the number k of base pairings enclosed in the multiloop, the four nucleotides surrounding each base pairing and the number of unpaired nucleotides in the loop $\mathbf{v}(\{i,p_1,q_1,\ \ldots,p_k,q_k,j\})$. We refer to Supplementary Data SA.2 for more details.

The classical RNA Secondary Structure Prediction problem is defined as follows.

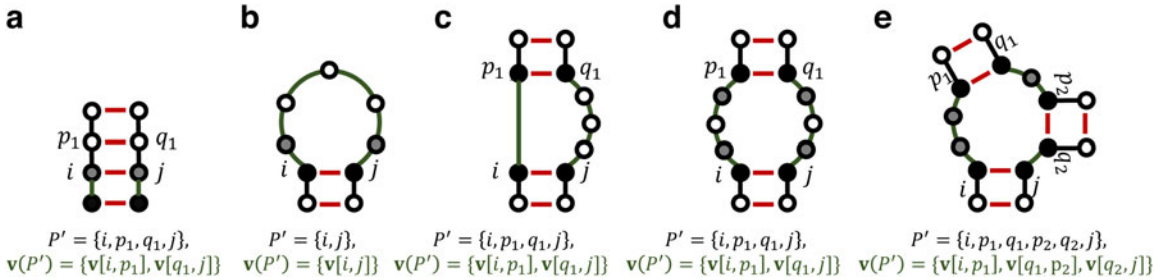

FIG. 2. There are five secondary structure elements. **(a)** Stacking. **(b)** Hairpin loop. **(c)** Bulge loop. **(d)** Internal loop. **(e)** Multibranch loop. Each structural element is defined by a unique set P' of nucleotide indices involved in base pairings (indicated in *red*). In addition, each structural element corresponds to a unique face of a planar embedding comprising subsequences $\mathbf{v}(P')$ (indicated in *green*). Nucleotides next to the base pairings (indicated in *gray*) are involved in providing a free energy contribution to some structural components.

Problem 2.1 (RNA Secondary Structure Prediction). *Given an RNA sequence* $\mathbf{v} \in \Sigma_{\text{rna}}^n$, *find a secondary structure* P *such that* $\text{MFE}(\mathbf{v}, P)$ *is minimized.*

This problem can be solved in $O(n^3)$ time using the Zuker and Stiegler (1981) algorithm. In this work we are interested in a reverse translation variant of the problem. That is, given a protein sequence $\mathbf{w} \in \Sigma_{\text{prot}}^m$ where Σ_{prot} is the set of 20 amino acids, we seek a corresponding RNA sequence $\mathbf{v} \in \Sigma_{\text{rna}}^{3m}$ that translates into $\mathbf{w}$. To that end, we use the function $S : \Sigma_{\text{prot}} \to \mathcal{P}(\Sigma_{\text{rna}}^3)$ such that $S(\alpha)$ is the set of codons that encode amino acid $\alpha \in \Sigma_{\text{prot}}$. We define $\sigma(a, s) = 3(a-1)+s$ to indicate the RNA sequence index corresponding to protein sequence index $a \in [m]$ and codon index $s \in \{1, 2, 3\}$.

Definition 2.4. *RNA sequence* $\mathbf{v} \in \Sigma_{\text{rna}}^n$ *encodes for protein sequence* $\mathbf{w} \in \Sigma_{\text{prot}}^m$ *provided (i)* $|\mathbf{v}| = n = 3m = 3|\mathbf{w}|$ *and (ii)* $\mathbf{v}[\sigma(a, 1), \sigma(a, 3)] \in S(w_a)$ *for all protein indices* $a \in [m]$.

Rather than only considering the MFE $\text{MFE}(\mathbf{v}, P)$, we also take species-specific codon usage bias into account. In other words, given the species-specific relative codon frequencies $g : \Sigma_{\text{rna}}^3 \to [0, 1]$, we compute the CAI $\text{CAI}(\mathbf{v}, \mathbf{w})$ defined as follows.

Definition 2.5. *The* CAI $\text{CAI}(\mathbf{v}, \mathbf{w})$ *of RNA sequence* $\mathbf{v}$ *that translates into protein sequence* $\mathbf{w}$ *is defined as*

$$\text{CAI}(\mathbf{v}, \mathbf{w}) = \sqrt[m]{\prod_{a=1}^{m} \frac{g(\mathbf{v}[\sigma(a, 1), \sigma(a, 3)])}{\max_{\mathbf{x} \in S(w_a)} g(\mathbf{x})}} \tag{1}$$

where $g(\mathbf{x})$ *is the* species-specific relative frequency *of codon* $\mathbf{x} \in \Sigma_{\text{rna}}^3$ *such that* $g(\mathbf{x}) \geq 0$ *for all codons* $\mathbf{x}$ *and* $\sum_{\mathbf{x} \in \Sigma_{\text{rna}}^3} g(\mathbf{x}) = 1$.

The CAI ranges from 0 to 1, where a value of 1 indicates that for each amino acid w_a the maximum frequency codon $\arg\max_{\mathbf{x} \in S(w_a)} g(\mathbf{x})$ is used (Sharp and Li, 1987). Thus, given a target protein sequence $\mathbf{w}$, there are two competing objective functions; we seek a corresponding RNA sequence $\mathbf{v}$ and a secondary structure P that simultaneously minimizes $\text{MFE}(\mathbf{v}, P)$ and maximizes $\text{CAI}(\mathbf{v}, \mathbf{w})$. Equivalently, rather than maximizing $\text{CAI}(\mathbf{v}, \mathbf{w})$, we maximize $\overline{\text{CAI}}(\mathbf{v}, \mathbf{w})$ defined as

$$\begin{aligned} \text{CAI}(\mathbf{v}, \mathbf{w}) &= \sqrt[m]{\prod_{a=1}^{m} \frac{g(\mathbf{v}[\sigma(a, 1), \sigma(a, 3)])}{\max_{\mathbf{x} \in S(w_a)} g(\mathbf{x})}} \\ &\propto \sum_{a=1}^{m} \log \frac{g(\mathbf{v}[\sigma(a, 1), \sigma(a, 3)])}{\max_{\mathbf{x} \in S(w_a)} g(\mathbf{x})} = \overline{\text{CAI}}(\mathbf{v}, \mathbf{w}). \end{aligned}$$

We model the trade-off between MFE and CAI by introducing a parameter $\lambda \in [0, 1]$ and minimizing a convex combination of $\text{MFE}(\mathbf{v}, P)$ and $-\overline{\text{CAI}}(\mathbf{v}, \mathbf{w})$.

Problem 2.2 (BRD). *Given a protein sequence* $\mathbf{w} \in \Sigma_{\text{prot}}^m$ *and parameter* $\lambda \in [0, 1]$, *find an RNA sequence* $\mathbf{v} \in \Sigma_{\text{rna}}^{3m}$ *with secondary structure* P *such that (i)* $\mathbf{v}$ *encodes for* $\mathbf{w}$ *and (ii) solution* $(\mathbf{v}, P)$ *minimizes* $\lambda \cdot \text{MFE}(\mathbf{v}, P) - (1-\lambda) \cdot \overline{\text{CAI}}(\mathbf{v}, \mathbf{w})$.

We say that a solution $(\mathbf{v}, P)$ is *Pareto optimal* if $(\mathbf{v}, P)$ is better than all other feasible solutions in at least one of the two objectives. In other words, there does not exist another solution $(\mathbf{v}', P')$ that is better in both objectives, or equal in one objective and better in the other. In our final problem, we seek all Pareto optimal RNA sequences $\mathbf{v}$.

Problem 2.3 (PORD). *Given a protein sequence* $\mathbf{w} \in \Sigma_{\text{prot}}^m$, *enumerate all RNA sequences* $\mathbf{v} \in \Sigma_{\text{rna}}^{3m}$ *each with a secondary structure* P *such that (i)* $\mathbf{v}$ *encodes for* $\mathbf{w}$ *and (ii)* $(\mathbf{v}, P)$ *is Pareto optimal with respect to* $\text{MFE}(\mathbf{v}, P)$ *and* $\text{CAI}(\mathbf{v}, \mathbf{w})$.

3. METHODS

3.1. RNA design with fixed λ

In the BRD problem (Problem 2.2), we are given a protein sequence $\mathbf{w} \in \Sigma_{\text{prot}}^m$ and parameter $\lambda \in [0, 1]$ that models the trade-off between MFE and CAI. In this section, we show how to solve this problem using dynamic programming. Specifically, for protein sequence indices $a, b \in [m]$, codon indices $s, t \in \{1, 2, 3\}$,

codons $\mathbf{x} \in S(w_a)$ and $\mathbf{y} \in S(w_b)$, $O[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$ is the minimum objective value when solving a problem instance restricted to the RNA sequence $\mathbf{v}[\sigma(a, s), \sigma(b, t)]$ such that codons $\mathbf{x}$ and $\mathbf{y}$ are used to encode amino acids w_a and w_b, respectively. Using $O[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$, we express the objective value of an optimal solution as

$$\min_{\mathbf{x} \in S(w_1),\, \mathbf{y} \in S(w_m)} O[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}. \tag{2}$$

To see why this is the case, observe that $O[1, 1]\_{\mathbf{x}}[m, 3]\_{\mathbf{y}}$ equals the minimum objective value for the complete RNA sequence $\mathbf{v}[\sigma(1, 1), \sigma(m, 3)] = \mathbf{v}[1, 3m] = \mathbf{v}$ restricted to using codons $\mathbf{x}$ and $\mathbf{y}$ for amino acid w_1 and w_m, respectively. Thus, the overall minimum objective value is obtained for the codon pair $(\mathbf{x}, \mathbf{y}) \in S(w_1) \times S(w_m)$ that minimizes $O[1, 1]_{\mathbf{x}}[m, 3]_{\mathbf{y}}$.

Let $\Gamma = \{(A, U), (U, A), (G, C), (C, G), (G, U), (U, G)\}$ be the set of allowed base pairings in the Turner energy model (Turner et al., 1988; Mathews et al., 2004). To express the contribution of the CAI, we introduce the shorthand $\bar{g}(w, \mathbf{x}) = \log\left(\frac{g(\mathbf{x})}{\max_{\mathbf{y} \in S(w)} g(\mathbf{y})}\right)$ such that

$$\overline{\mathrm{CAI}}(\mathbf{v}, \mathbf{w}) = \sum_{a=1}^{m} \bar{g}(w_a, \mathbf{v}[\sigma(a, 1), \sigma(a, 3)]). \tag{3}$$

We define $O[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$ recursively as

$$\min \begin{cases} -(1-\lambda)\bar{g}(w_a, \mathbf{x}), & \text{if } a = b,\ \mathbf{x} = \mathbf{y}, \\ \infty, & \text{if } a = b,\ \mathbf{x} \neq \mathbf{y}, \\ O[a, s+1]_{\mathbf{x}}[b, t]_{\mathbf{y}}, & \text{if } a < b,\ s \in \{1, 2\}, \\ O[a, s]_{\mathbf{x}}[b, t-1]_{\mathbf{y}}, & \text{if } a < b,\ t \in \{2, 3\}, \\ \min\limits_{\mathbf{x}' \in S(w_{a+1})} \{O[a+1, 1]_{\mathbf{x}'}[b, t]_{\mathbf{y}}\} - (1-\lambda)\bar{g}(w_a, \mathbf{x}), & \text{if } a \leq b-1,\ s = 3, \\ \min\limits_{\mathbf{y}' \in S(w_{b-1})} \{O[a, s]_{\mathbf{x}}[b-1, 3]_{\mathbf{y}'}\} - (1-\lambda)\bar{g}(w_b, \mathbf{y}), & \text{if } a \leq b-1,\ t = 1, \\ \min\limits_{\substack{a \leq c < b \\ t' \in \{1,2\} \\ \mathbf{x}' \in S(w_c)}} \left\{ \begin{array}{l} O[a, s]_{\mathbf{x}}[c, t']_{\mathbf{x}'} + E[c, t'+1]_{\mathbf{x}'}[b, t]_{\mathbf{y}} \\ \quad + (1-\lambda)\bar{g}(w_c, \mathbf{x}') \end{array} \right\}, & \text{if } a < b, \\ \min\limits_{\substack{a \leq c < b-1 \\ \mathbf{y}' \in S(w_c) \\ \mathbf{x}' \in S(w_{c+1})}} O[a, s]_{\mathbf{x}}[c, 3]_{\mathbf{y}'} + E[c+1, 1]_{\mathbf{x}'}[b, t]_{\mathbf{y}}, & \text{if } a < b-1, \\ E[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}, & \text{if } a < b,\ (x_s, y_t) \in \Gamma. \end{cases}$$

There are two components in the objective function, the CAI and the MFE. We account for MFE upon identifying structural elements at base pairing $(\sigma(a, s),\ \sigma(b, t))$ using the energy functions in Definition 2.3. To avoid double counting, we must ensure that CAI is only accounted for once for each codon. As such, we include a CAI contribution when crossing codon boundaries or reaching a valid base case.

The first case in the above recurrence corresponds to the base case where $a = b$ and $\mathbf{x} = \mathbf{y}$. In that case, base pairing between $\sigma(a, s)$ and $\sigma(b, t)$ is not possible as the Turner energy model (Turner et al., 1988; Mathews et al., 2004) requires at least two nucleotides in between a pairing. In this base case, we must account for the CAI contribution of codon $\mathbf{x}$. The other base case occurs when $a = b$ and $\mathbf{x} \neq \mathbf{y}$, which is not allowed as any one amino acid must be encoded by a single codon—this case thus receives a value of ∞.

The next two cases correspond to, respectively, incrementing either the left index $\sigma(a, s)$ or decrementing the right index $\sigma(b, t)$ without crossing any codon boundary and leaving the corresponding nucleotide unpaired. As such, we do not have to account for CAI. However, in the following two cases, we additionally cross the codon boundary and thus must account for the CAI contribution of, respectively, codons $\mathbf{x}$ and $\mathbf{y}$. Next, we include two cases corresponding to bifurcating into two parts, one part is between nucleotides $\sigma(a, s)$ and $\sigma(c, t')$ and the other part is between nucleotides $\sigma(c, t') + 1$ and $\sigma(b, t)$. In the first case, the split happens inside a codon, that is, $t' \in \{1, 2\}$. We must include a correction of $+(1-\lambda)\bar{g}(w_c, \mathbf{x}')$ as both parts will include a CAI contribution of the same codon $\mathbf{x}'$. On the contrary, when the split happens outside a codon, that is, $t' = 3$ then no such correction is needed.

The last case corresponds to base pairing between $\sigma(a, s)$ and $\sigma(b, t)$. Specifically, $E[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$ denotes the optimal objective value when nucleotides $v_{\sigma(a, s)}$ and $v_{\sigma(b, t)}$ correspond to codons $\mathbf{x}$ and $\mathbf{y}$, respectively,

and form a base pairing. When calculating $E[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$, we consider the minimum among the five cases corresponding the five secondary structures elements defined in Section 2. That is, $E[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$ equals $\min\{E_s[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}},\ E_h[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}},\ E_b[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}},\ E_i[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}},\ E_m[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}\}$.

3.1.1. Recurrences for structural elements. In this section, we present the recurrence $E_s[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$ for the stacking element, the recurrence $E_h[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$ for the hairpin loop element, the recurrence $E_b[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$ for the bulge loop element, the recurrence $E_i[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$ for the internal loop element, and the recurrence $E_m[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$ for the multibranch loop. In particular, we require two additional recurrences $M[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$ and $N[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$ for solving the multibranch loop case.

3.1.1.1. Stacking

We start with the contribution $E_s[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$ for the stacking element. In addition to base pairing between $\sigma(a, s)$ and $\sigma(b, t)$, the stacking element requires base pairing of the adjacent nucleotides $\sigma(a, s)+1$ and $\sigma(b, t)-1$.

Let c and s' be the protein sequence and codon indices of nucleotide $\sigma(a, s)+1$, that is, $\sigma(c, s')=\sigma(a, s)+1$. Similarly, let d and t' be the protein sequence and codon indices of nucleotide $\sigma(b, t)-1$, that is, $\sigma(d, t')=\sigma(b, t)-1$. Then, $E_s[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$ is recursively defined as

$$\min_{\substack{\mathbf{h}\in S(w_c),\ \mathbf{r}\in S(w_d)\\ (h_{s'},\, r_{t'})\in\Gamma}} \{\, E[c, s']_{\mathbf{h}}[d, t']_{\mathbf{r}} + \lambda f_{\mathrm{s}}(\mathbf{v}(P')) - (1-\lambda)\bar{g}_s(a, \mathbf{x}, b, \mathbf{y}) \quad \}$$

where $P'=\{\sigma(a, s), \sigma(c, s'), \sigma(d, t'), \sigma(b, t)\}$, and $\bar{g}_s$ equals the CAI contributed by the stacking structure defined as

$$\bar{g}_s(a, \mathbf{x}, b, \mathbf{y}) = 1\{c \neq a\} \cdot \bar{g}(w_a, \mathbf{x}) + 1\{d \neq b\} \cdot \bar{g}(w_b, \mathbf{y}).$$

More specifically, condition $1\{c \neq a\}$ checks if the transition from $\sigma(a, s)$ to $\sigma(c, s')$ crosses the codon boundary, in which the CAI contribution of codon $\mathbf{x}$ must be accounted for. Condition $1\{d \neq b\}$ performs a similar check for codon $\mathbf{y}$.

3.1.1.2. Hairpin loop

As before, we let c be the protein sequence index of nucleotide $\sigma(a, s)+1$, and d be the protein sequence index of nucleotide $\sigma(b, t)-1$. Since adjacent nucleotides $\sigma(a, s)+1$ and $\sigma(b, t)-1$ are used in the MFE computation f_h (see Supplementary Data SA.2), we must identify their respective codons $\mathbf{h}$ and $\mathbf{r}$ that achieve the minimum objective value. Thus, we define $E_h[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$ as

$$\min_{\mathbf{h}\,\in\, S(w_c),\ \mathbf{r}\,\in\, S(w_d)} \{\quad \lambda f_{\mathrm{h}}(\mathbf{v}(P')) - (1-\lambda)\bar{g}_h(a, \mathbf{x}, c, \mathbf{h}, d, \mathbf{r}, b, \mathbf{y}) \quad \}$$

where $P'=\{\sigma(a, s), \sigma(b, t)\}$, and $\bar{g}_h$ equals the CAI contributed by the hairpin structure defined as

$$\begin{aligned}\bar{g}_h(a, \mathbf{x}, c, \mathbf{h}, d, \mathbf{r}, b, \mathbf{y}) = \bar{g}(w_a, \mathbf{x}) \quad & + \bar{g}(w_b, \mathbf{y}) + 1\{c \neq a \wedge c \neq b\} \cdot \bar{g}(w_c, \mathbf{h}) \\ & + 1\{d \neq c \wedge d \neq a \wedge d \neq b\} \cdot \bar{g}(w_d, \mathbf{r}).\end{aligned}$$

Given that $a < b$, we cross the two codon boundaries at both ends. Therefore, $\bar{g}_h$ accounts for the CAI contributions of codons $\mathbf{x}$ and $\mathbf{y}$ via the terms $\bar{g}(w_a, \mathbf{x})$ and $\bar{g}(w_b, \mathbf{y})$. We must additionally account for the CAI contributions of codons $\mathbf{h}$ and $\mathbf{r}$ if codon boundaries were crossed, explaining the last two terms. Finally, note that nucleotides corresponding to amino acids between $a+1$ and $b-1$ are not considered in the MFE computation f_h. As such, for each such amino acid w, we are free to choose the codon $\mathbf{x}^*$ that achieves the maximum value $\bar{g}(w, \mathrm{x}^*)=0$—since the maximum value is 0, we do not need to include these terms in recurrence.

3.1.1.3. Bulge loop

In addition to base pairing between $\sigma(a, s)$ and $\sigma(b, t)$, the bulge loop element requires another base pairing between nucleotides $\sigma(c, s')$ and $\sigma(d, t')$ where $a \leq c < d \leq b$ and $s', t' \in \{1, 2, 3\}$ and the bulge occurs either adjacent to $\sigma(a, s)$ (i.e., $\sigma(b, t)-\sigma(d, t')=1$) or adjacent to $\sigma(b, t)$ (i.e., $\sigma(c, s')-\sigma(a, s)=1$).

We define $E_b[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$ as

$$\min_{\substack{\mathbf{h}\in S(w_c),\ \mathbf{r}\in S(w_d)\\ (h_{s'},\, r_{t'})\in\Gamma}} \{\, E[c, s']_{\mathbf{h}}[d, t']_{\mathbf{r}} \quad + \lambda f_{\mathrm{b}}(\mathbf{v}[P'])\} - (1-\lambda)\bar{g}_b(a, c, \mathbf{x}, d, b, \mathbf{y})$$

where $P' = \{\sigma(a, s), \sigma(c, s'), \sigma(d, t'), \sigma(b, t)\}$, and $\bar{g}_b$ equals the CAI contributed by the bulge loop structure defined as

$$\bar{g}_b(a, c, \mathbf{x}, d, b, \mathbf{y}) = 1\{c \neq a\} \cdot \bar{g}(w_a, \mathbf{x}) + 1\{b \neq d\} \cdot \bar{g}(w_b, \mathbf{y}).$$

Specifically, $1\{c \neq a\}$ checks if the transition from $\sigma(a, s)$ to $\sigma(c, s')$ crosses a codon boundary, and $1\{d \neq b\}$ does so for $\sigma(b, t)$ to $\sigma(d, t')$. When either condition is met, we account for the CAI contribution of codons $\mathbf{x}$ and $\mathbf{y}$, respectively. Note that the nucleotides corresponding to amino acids between a and c or between d and b do not contribute to f_b. As such, we choose the corresponding codons that would provide the largest CAI contribution, which each have a log value of 0 and thus do not contribute to the objective function.

3.1.1.4. Internal loop

Similarly to the bulge loop, the internal loop requires another base pairing between $\sigma(c, s')$ and $\sigma(d, t')$ in addition to the pairing between $\sigma(a, s)$ and $\sigma(b, t)$. As per Supplementary Data SA.1 Definition 7, we have $\sigma(c, s') - \sigma(a, s) > 1$ and $\sigma(b, t) - \sigma(d, t') > 1$. In addition, let a_1, b_1, c_1 and d_1 be the protein sequence indices of adjacent nucleotides $\sigma(a, s)+1, \sigma(b, t)-1, \sigma(c, s')-1$ and $\sigma(d, t')+1$, respectively. Then, $E_i[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$ is recursively defined as

$$\min_{\substack{\mathbf{h}\in S(w_c),\, \mathbf{r}\in S(w_d) \\ \mathbf{x}'\in S(w_{a_1}),\, \mathbf{y}'\in S(w_{b_1}) \\ \mathbf{h}'\in S(w_{c_1}),\, \mathbf{r}'\in S(w_{d_1})}} \left\{ \begin{array}{l} E[c, s']_{\mathbf{h}}[d, t']_{\mathbf{r}} \\ \quad + \lambda f_i(\mathbf{v}(P')) - (1-\lambda)\bar{g}_i(a, \mathbf{x}, a_1, \mathbf{x}', c_1, \mathbf{h}', d_1, \mathbf{r}', b_1, \mathbf{y}', b, \mathbf{y}) \end{array} \right\}$$

where $P' = \{\sigma(a, s), \sigma(c, s'), \sigma(d, t'), \sigma(b, t)\}$, and $\bar{g}_i$ equals the CAI contributed by the internal loop structure defined as

$$\begin{aligned} \bar{g}_i(a, \mathbf{x}, a_1, \mathbf{x}', c_1, \mathbf{h}', d_1, \mathbf{r}', b_1, \mathbf{y}', b, \mathbf{y}) = \quad & 1\{c \neq a\} \cdot \bar{g}(w_a, \mathbf{x}) \\ & + 1\{a_1 \neq a \wedge a_1 \neq c\} \cdot \bar{g}(a_1, \mathbf{x}') \\ & + 1\{c_1 \neq c \wedge c_1 \neq a \wedge c_1 \neq a_1\} \cdot \bar{g}(c_1, \mathbf{h}') \\ & + 1\{d \neq b\} \cdot \bar{g}(w_b, \mathbf{y}) \\ & + 1\{b_1 \neq b \wedge b_1 \neq d\} \cdot \bar{g}(b_1, \mathbf{y}') \\ & + 1\{d_1 \neq d \wedge d_1 \neq b \wedge d_1 \neq b_1\} \cdot \bar{g}(d_1, \mathbf{r}'). \end{aligned}$$

As can be seen, $\bar{g}_i$ includes bookkeeping to prevent double counting CAI contributions. In the worst case, the internal loop element would only span the single codon $\mathbf{x}$ on the left side (i.e., $\sigma(a, s)+2 = \sigma(c, s')$) and the single codon $\mathbf{y}$ on the right side (i.e., $\sigma(d, t')+2 = \sigma(b, t)$), in which case we should not account for the CAI contributions of $\mathbf{x}$ and $\mathbf{y}$ and defer these to $E[c, s']_{\mathbf{h}}[d, t']_{\mathbf{r}}$. Using a similar argument as in previous sections, for the amino acids between a and c as well as between d and b, we choose the corresponding codons that would provide the largest CAI contribution, which in each case equals 0.

3.1.1.5. Multibranch loop

As per Supplementary Data SA.1 Definition 8, a multibranch loop has $k \geq 2$ base pairings in addition to $(\sigma(a, s), \sigma(b, t))$. In the Turner energy model (Mathews et al., 2004), there are three constants associated with the MFE contribution f_m of a multibranch loop: (1) A is the energy penalty paid upon the creation of a multibranch loop, which also accounts for the base pairing $(\sigma(a, s), \sigma(b, t))$, (2) B is the energy penalty paid for each unpaired nucleotide that is part of a multibranch loop, and (3) C is the energy penalty when including a base pairing as part of the multibranch loop (this is paid k times) (Jaeger et al., 1989; Poznanović et al., 2020). We define $E_m[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$ as

$$\min \begin{cases} M[a, s+1]_{\mathbf{x}}[b, t-1]_{\mathbf{y}} + \lambda A, & \text{if } a < b,\ s \in \{1, 2\},\ t \in \{2, 3\}, \\ \min_{\mathbf{x}'\in S(w_{a+1})}\{M[a+1, 1]_{\mathbf{x}'}[b, t-1]_{\mathbf{y}}\} + \lambda A - (1-\lambda)\bar{g}(w_a, \mathbf{x}), & \text{if } a < b-1,\ s = 3,\ t \in \{2, 3\}, \\ \min_{\mathbf{y}'\in S(w_{b-1})}\{M[a, s+1]_{\mathbf{x}}[b-1, 3]_{\mathbf{y}'}\} + \lambda A - (1-\lambda)\bar{g}(w_b, \mathbf{y}), & \text{if } a < b-1,\ s \in \{1, 2\},\ t = 1, \\ \min_{\substack{\mathbf{x}'\in S(w_{a+1}) \\ \mathbf{y}'\in S(\mathbf{w}_{b-1})}}\{M[a+1, 1]_{\mathbf{x}'}[b-1, 3]_{\mathbf{y}'}\} + \lambda A - (1-\lambda)\bar{g}(w_a, \mathbf{x}) + \bar{g}(w_b, \mathbf{y}), & \text{if } a < b-2,\ s = 3,\ t = 1. \end{cases}$$

In other words, we pay the energy penalty A for creating the multibranch loop associated with base pairing $(\sigma(a, s), \sigma(b, t))$. There are four distinct cases to account for crossing codon boundaries and paying

the associated CAI; in the first case no codon boundaries are crossed, in the second case $\sigma(a, s)+1$ crosses a codon boundary and we must identify the optimal codon for amino acid w_{a+1} and account for the CAI for the previous codon $\mathbf{x}$ for amino acid w_a, the third case is symmetrical for the other side $\sigma(b, t)-1$, and in the fourth case we cross codon boundaries at both sides. After accounting for penalty A and potential CAI contributions associating with crossing codon boundaries when moving to $\sigma(a, s)+1$ and $\sigma(b, t)-1$, we model the remainder of the value of $E_m[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$ with the recurrence $M[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$. Specifically, we use that there are at least $k \geq 2$ more pairings between $v_{\sigma(a, s)}$ and $v_{\sigma(b, t)}$, and split it into structures consisting of fewer pairings, that is

$$M[a, s]_{\mathbf{x}}[b, t']_{\mathbf{y}} = \min_{\substack{a<c\leq d<b \\ 1\leq s', t'\leq 3 \\ \mathbf{y}'\in S(w_c),\, \mathbf{x}'\in S(w_d)}} \{N[a, s]_{\mathbf{x}}[c, t']_{\mathbf{y}'} + N[d, s']_{\mathbf{x}'}[b, t]_{\mathbf{y}}\}$$

where $N[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$ is the objective value of part of a multibranch loop that contains at least one pairing. We define $N[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$ recursively as

$$\min \begin{cases} E[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}} + \lambda C, & \text{if } a < b,\ (x_s, y_t) \in \Gamma, \\ N[a, s+1]_{\mathbf{x}}[b, t]_{\mathbf{y}} + \lambda B, & \text{if } a < b,\ s \in \{1, 2\}, \\ N[a, s]_{\mathbf{x}}[b, t-1]_{\mathbf{y}} + \lambda B, & \text{if } a < b,\ t \in \{2, 3\}, \\ \min_{\mathbf{x}'\in S(w_{a+1})} \{N[a+1, s]_{\mathbf{x}'}[b, t]_{\mathbf{y}}\} - (1-\lambda)\bar{g}(w_a, \mathbf{x}) + \lambda B, & \text{if } a < b-1,\ s=3, \\ \min_{\mathbf{y}'\in S(w_{b-1})} \{N[a, s]_{\mathbf{x}}[b-1, t]_{\mathbf{y}'}\} - (1-\lambda)\bar{g}(w_b, \mathbf{y}) + \lambda B, & \text{if } a < b-1,\ t=1, \\ \min\limits_{k<3,\, a<c<b,\, \mathbf{y}'\in S(w_c)} \left\{ \begin{array}{c} N[a, s]_{\mathbf{x}}[c, k]_{\mathbf{y}'} + N[c, k+1]_{\mathbf{y}'}[b, t]_{\mathbf{y}} \\ -(\lambda-1)\bar{g}(w_c, \mathbf{y}') \end{array} \right\}, & \text{if } a < b-1, \\ \min\limits_{a<c<b-1,\, \mathbf{y}'\in S(w_c),\, \mathbf{x}'\in S(v_{c+1})} \{ N[a, s]_{\mathbf{x}}[c, 3]_{\mathbf{y}'} + N[c+1, 1]_{\mathbf{x}'}[b, t]_{\mathbf{y}} \}, & \text{if } a < b-2. \end{cases}$$

The first case in the above recurrence for N corresponds to base pairing between $\sigma(a, s)$ and $\sigma(b, t)$, and we pay the energy penalty C for including another base pairing as part of the multibranch loop. The next two cases correspond to incrementing either the left index $\sigma(a, s)$ or decrementing the right index $\sigma(b, t)$ without crossing any codon boundary and for each case leaving the corresponding nucleotide unpaired. For these two cases, we pay the energy penalty B for including an unpaired nucleotide as part of the multibranch loop. In the following two cases, incrementing the left index or decrementing the right index crosses a codon boundary, thus incurring a CAI contribution for the respective codons $\mathbf{x}$ and $\mathbf{y}$. In addition, these two cases include the energy penalty C for including an unpaired nucleotide.

The last two cases correspond to bifurcating into two parts, one part is between nucleotide $\sigma(a, s)$ and $\sigma(c, t')$ and the other part is between nucleotide $\sigma(c, t')+1$ and $\sigma(b, t)$. In the first bifurcation case, there is a correction of $+(1-\lambda)\bar{g}(w_c, x')$ as both parts will include a CAI contribution for the same codon $\mathbf{x}'$. In the other bifurcation case, as the split happens between two codons, that is, $t'=3$, no such correction is needed.

3.1.2. Dynamic programming, time, and space complexity. We store the following four tables: (1) $O[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$, (2) $E[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$, (3) $M[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$, and (4) $N[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$, each with the same dimensions. In particular, as each potential base pairing $(\sigma(a, s), \sigma(b, t))$ corresponds to exactly one of five structural elements, we do not store the corresponding values $E_s[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$, $E_h[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$, $E_b[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$, $E_i[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$, and $E_m[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$ separately, but only their minimum value in $E[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$. Note that the four stored tables have the same dimensions that comprised protein sequence indices $a, b \in [m]$, codon indices $s, t \in \{1, 2, 3\}$, and codons $\mathbf{x} \in S(w_a)$ and $\mathbf{y} \in S(w_b)$. Letting K denote the maximum number of codons associated with a single amino acid—the amino acids leucine (L), serine (S), and arginine (R) each have $K=6$ of codons—we conclude that the space complexity is $\mathcal{O}(m^2K^2)$.

Inspection of the recurrences reveals that the computation of each entry $[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$ in the four tables does not require access to entries $[a, s]_{\mathbf{x}'}[b, t]_{\mathbf{y}'}$ using other codons $\mathbf{x}' \neq \mathbf{x}$ and $\mathbf{y}' \neq \mathbf{y}$. On the contrary, we do require access to entries $[a', s']_{\mathbf{x}'}[b', t']_{\mathbf{y}'}$ where $\sigma(a', s') \geq \sigma(a, s)$, $\sigma(b', t') \leq \sigma(b, t)$ (indicated with dashed lines in Fig. 3). Moreover, with the exception of the base cases for table O, where $a=b$, the recurrences require $a < b$. This means we can organize the four tables as two-dimensional tables where the rows correspond to entries (a, s) and the columns correspond to entries (b, t), both sorted in an increasing lexicographical order. Each entry $[(a, s)][(b, t)]$ corresponds to another two-dimensional table whose rows correspond to codons $\mathbf{x} \in S(w_a)$ and columns to codons $\mathbf{y} \in S(w_b)$—see Figure 3b. We fill out the tables diagonally.

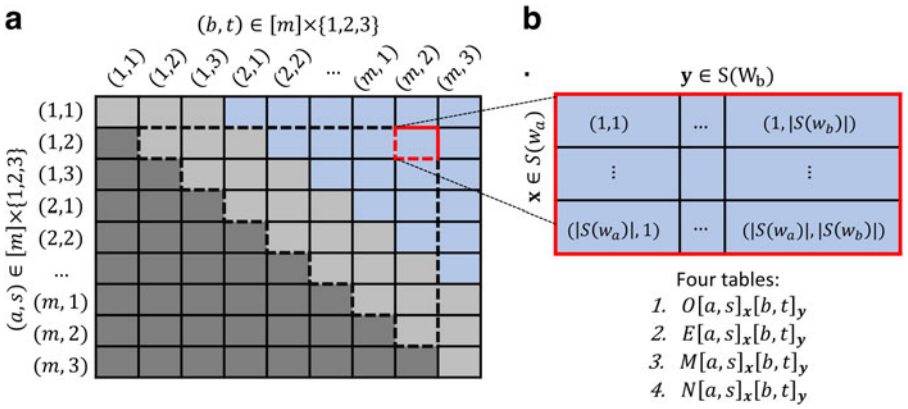

FIG. 3. Dynamic programming for solving the BRD problem. **(a)** To solve this problem, we store four dynamic programming tables O, E, M, and N with identical dimensions indexed as $[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$. Rows and columns correspond to pairs (a, s), $(b, t) \in [m]\times\{1, 2, 3\}$, respectively, both ordered lexicographically in increasing order. With the exception of the base cases for table O where $a = b$ (indicated in *light gray*), the recurrences require $a < b$ (indicated in *blue*). The dashed lines outline the entries of the table on which the red entry depends. **(b)** Each entry $[(a, s)][(s, t)]$ expands into another codon-by-codon table, whose rows are codons $\mathbf{x} \in S(w_a)$ and columns are codons $\mathbf{y} \in S(w_b)$.

More precisely, filling out the four entries indexed by $[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$, we check if base pairing between $\sigma(a, s)$ and $\sigma(b, t)$ is possible, that is, if $(x_s, y_t) \in \Gamma$. If so, we first fill out the entry in E followed by N and then finally M. On the contrary, we first fill out the entry in N, then M, and finally E. After completely filling out tables E, M, and N, we fill out table O. This ordering follows the recurrences. We store back pointers to be able to identify the optimal solution $(\mathbf{v}, P)$ when performing the back trace.

For each entry $O[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$, the running time is dominated by the case to determine $E[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$. That is, for each entry $E[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$, it takes $\mathcal{O}(K^2)$ time to compute a stacking element or a hairpin loop element, $\mathcal{O}(mK^2)$ time to compute a bulge loop element, and worst-case $\mathcal{O}(m^2K^6)$ time to determine the contribution of an internal loop element. To remedy the worst-case $\mathcal{O}(m^2K^6)$ time, we follow other secondary structure prediction methods and use a parameter L to bound the maximum interior loop size, which affects bulge loop and internal loop elements (Hofacker et al., 1994; Lyngso et al., 1999). Then, the time to determine the contribution of an internal loop element can be reduced to $\mathcal{O}(mLK^6)$. Since there are $\mathcal{O}(m^2K^2)$ entries to compute, the overall time complexity of solving the dynamic program is $\mathcal{O}(m^2K^2) \cdot \mathcal{O}(mLK^6) = \mathcal{O}(m^3LK^8)$.

When disregarding CAI, that is, $\lambda = 1$, we can adapt the recurrences such that for each entry $E[a, s]_{\mathbf{x}}[b, t]_{\mathbf{y}}$, it would take $\mathcal{O}(1)$ time to compute a stacking or a hairpin loop element, $\mathcal{O}(m)$ time to compute a bulge loop element, and worst-case $\mathcal{O}(m^2)$ time to determine the contribution of an internal loop element. With a similar implementation of a maximum interior loop size L, the time to compute an internal loop element can be reduced to $\mathcal{O}(mL)$. Thus, the overall time complexity drops to $\mathcal{O}(m^2K^2) \cdot \mathcal{O}(mL) = \mathcal{O}(m^3LK^2)$ when $\lambda = 1$.

3.2. *Pareto optimal RNA design*

In the PORD problem (Problem 2.3), we are given a protein sequence $\mathbf{w} \in \Sigma_{prot}^m$ and seek a set of Pareto optimal solutions $(\mathbf{v}, P)$. We use the weighted sum method (Zadeh, 1963). In this method, distinct convex combinations of the multiple objective functions are optimized. In our case this corresponds to solving distinct convex combinations of the two objectives MFE (Definition 2.3) and $\overline{\text{CAI}}$ (Eq. (2)), which correspond to solving distinct instances of the BRD problem with varying values of the parameter $\lambda \in [0, 1]$. The weighted sum method has several limitations: (1) multiple λs may generate the same solution, (2) the nonconvex part of the Pareto front cannot be recovered, and (3) there are nonuniform sampling issues (Das and Dennis, 1997; Cohon, 2004).

We mitigate the first limitation by recursively examining λ values. More specifically, we maintain a queue Q of intervals $[\lambda^-, \lambda^+]$ as well as a hash table X such that $X[\lambda]$ yields the solution $(\mathbf{v}, P)$ of the BRD problem instance $(\mathbf{w}, \lambda)$. Initially, Q contains a single interval $[\epsilon, 1-\epsilon]$ where ϵ is a small constant (the default value in our implementation is $\epsilon = 10^{-5}$). In addition, we initialize $X[\epsilon]$ and $X[1-\epsilon]$ with the

solutions of BRD problem instances $(\mathbf{w}, \epsilon)$ and $(\mathbf{w}, 1-\epsilon)$, respectively. As long as the queue Q is not empty, we obtain an interval $[\lambda^-, \lambda^+]$ from Q, and solve a new BRD instance $(\mathbf{w}, \lambda)$ where $\lambda = \lambda^- + (\lambda^+ - \lambda^-)/2$, yielding solution $(\mathbf{v}, P)$. If this solution differs from $X[\lambda^-]$ and $X[\lambda^+]$, we set $X[\lambda] = (\mathbf{v}, P)$ and add (λ^-, λ) and (λ, λ^+) to the queue Q if $\lambda - \lambda^- > \tau$. We use a default value of 10^{-3} for the threshold parameter τ.

3.3. Implementation details of DERNA

We implemented our algorithms for solving the BRD and PORD problems in *C*++11. The resulting method, DERNA, is open source and available at https://github.com/elkebir-group/derna.git under the BSD 3-clause license. Usage instructions and examples are also available on the GitHub site. Alternatively, prebuilt binaries are available on Bioconda (Grüning et al., 2018).

DERNA uses the same energy model (Mathews et al., 1999) as CDSfold (Terai et al., 2016). For codon usage data, DERNA use the *Homo sapiens* codon usage table published in the codon usage database (Nakamura et al., 2000). In addition, DERNA accepts alternative energy models and codon usage data in the CSV format.

To validate the correctness of our algorithm and its implementation, we split our recurrences into two separate components and utilized two separate tables to store the MFE and the CAI separately. Using the real data instances examined in Section 4, for each solution $(\mathbf{v}, P)$ identified by DERNA, we confirmed that the MFE predicted by DERNA matched the MFE calculated using the Zuker algorithm (Zuker and Stiegler, 1981) when given DERNA's inferred RNA sequence $\mathbf{v}$. In addition, we recomputed the CAI of DERNA's inferred RNA sequence $\mathbf{v}$ and confirmed that the resulting value matched the CAI inferred by DERNA.

4. RESULTS

We compare DERNA with CDSfold (Terai et al., 2016) and LinearDesign (Zhang et al., 2023) on 100 protein sequences from the UniProt database (The UniProt Consortium, 2022) (Section 4.1) as well as on a case study involving the SARS-CoV-2 spike protein (Section 4.2). While the LinearDesign article (Zhang et al., 2023) describes both an exact and a heuristic algorithm, only the heuristic algorithm was publicly available. As such, we were only able to include the heuristic algorithm in our benchmarking. All experiments were performed on a laptop with an Apple M1 Max 10-core CPU and 64 GB of RAM.

4.1. Benchmarking on 100 UniProt protein sequences

We begin by performing experiments that prioritize MFE over CAI in Section 4.1.1. In Section 4.1.2, we focus on the PORD problem, seeking solutions that collectively capture the trade-off between MFE and CAI.

4.1.1. Prioritizing MFE. The goal of this section is to assess the ability of RNA design methods to prioritize MFE over CAI. We seek solutions that achieve the minimum MFE and, as a secondary criterion, achieve the largest CAI—that is, among the space of solutions that achieve minimum MFE, we prefer those solutions that have the largest CAI value. We benchmarked using the same 100 protein sequences used in the CDSfold article (Terai et al., 2016), which come from the UniProt database (The UniProt Consortium, 2022) and having lengths ranging from 78 to 2828 amino acids (Fig. 4a and Supplementary Table S1). By design, CDSfold does not take CAI into account. Both DERNA and LinearDesign support balancing MFE and CAI. For DERNA, we set $\lambda = 1 - \epsilon = 1 - 10^{-5}$.

We note that LinearDesign's objective function is slightly different than DERNA's, seeking an RNA sequence $\mathbf{v}$ and secondary structure P that minimize $\mathrm{MFE}(\mathbf{v}, P) - \lambda_{\mathrm{LD}} \cdot \log \overline{\mathrm{CAI}}(\mathbf{v}, \mathbf{w})$ for a target protein sequence $\mathbf{w}$. To similarly prioritize MFE, we set $\lambda_{\mathrm{LD}} = \epsilon = 10^{-5}$ for LinearDesign and ran it with default parameters.

With the exception of the longest sequence (Q9NR99) with 2828 amino acids, which LinearDesign failed to complete within 24 hours (after which we killed the process), all methods ran successfully on all sequences. Moreover, with the exception of protein sequence Q9HAE3 (with 211 amino acids), all methods achieved the same minimum MFE (Fig. 4b and Supplementary Fig. S1). However, the CAI values varied between methods. In particular, DERNA with $\lambda = \epsilon$ and LinearDesign λ_{LD} achieved larger CAI values than CDSfold for all instances (Fig. 4c and Supplementary Fig. S2). This makes sense because CDSfold only

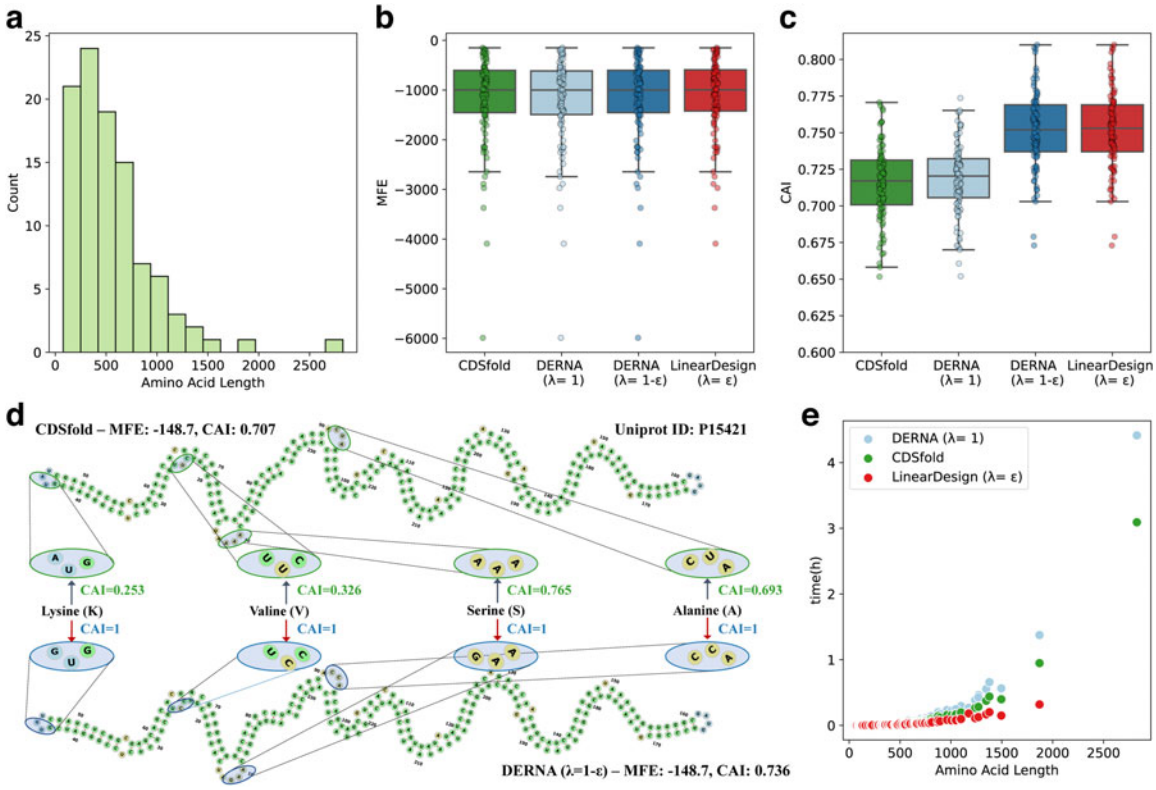

FIG. 4. DERNA with $\lambda = 1 - \epsilon$ identifies solutions that achieve optimal MFE and largest CAI as a secondary objective. **(a)** We used 100 UniProt sequences, with varying lengths as shown. **(b)** With one exception (discussed in the text), all methods returned solutions with the same MFE. **(c)** However, the CAI values differed drastically between methods, with DERNA ($\lambda = 1 - \epsilon$) and LinearDesign outperforming CDSfold. **(d)** As an example, we show protein sequence P15421 for which CDSfold (top) and DERNA (bottom) inferred the same MFE and identical secondary structures. However, the solutions contain different codons resulting in different CAI values. **(e)** Wall-clock running times in hours.

optimizes MFE but not CAI. The improved CAI values suggest that the sequences generated using our approach may exhibit a higher in vivo translational efficiency without sacrificing the mRNA half-life (Mauger et al., 2019).

To further illustrate this point, we highlight the results for protein sequence P15421 with 78 amino acids. Both CDSfold and DERNA achieved the same MFE value of -148.7, yielding identical secondary structures (in terms of complementary base pairings) consisting of mostly stacking elements that achieve the lowest MFE. CDSfold, however, identified a different RNA sequence than DERNA resulting in a CAI of 0.707, whereas DERNA achieved a CAI of 0.736. The two RNA sequences differ at four codons encoding four distinct amino acids. For each such amino acid, DERNA used the codon that achieved the largest CAI value. For example, for the first codon encoding for the amino acid lysine (K), DERNA used the codon GUG with a relative usage frequency of 1, whereas CDSfold used the codon GUA with a smaller relative frequency of 0.253.

The other three codons differed in a similar manner. We note that LinearDesign identified the same RNA sequence as DERNA for this instance.

As for the CAI values inferred by LinearDesign, these largely match those inferred by DERNA (Fig. 4c and Supplementary Fig. S2). The only exception is protein sequence Q9HAE3 where LinearDesign (run with $\lambda_{\mathrm{LD}} = 10^{-5}$) performed better in terms of CAI with a value of 0.754 versus 0.748 for DERNA. The solution inferred by DERNA and CDSfold, however, has a better MFE of -369.9 compared with -369.4 for LinearDesign (Supplementary Fig. S3), which is a nonoptimal solution in terms of MFE. Even when running with other 5 other values of λ_{LD} sampled from the range $[0, 10^{-5}]$, LinearDesign still failed to achieve an MFE of -369.9.

Although LinearDesign could potentially reach the MFE-optimal solution with an increased beam size or by using the exact search algorithm described in the article (Zhang et al., 2023), we were unable to assess this as only the heuristic algorithm with a fixed beam size (of 500) available. Using a smaller $\lambda = 0.062509 < 1 - \epsilon$, which slightly increases the priority for CAI, DERNA was able to recover LinearDesign's solution.

Finally, we consider the running times of CDSfold, LinearDesign, and DERNA. Leaving out the largest instance (for which LinearDesign failed), we found that LinearDesign was the fastest algorithm with running times ranging from 1.80 to 1149.02 seconds, followed by CDSfold with running times ranging from 1.86 to 3411.91 seconds and then DERNA with running times ranging from 16 to 21, 434 seconds. It is important to note that DERNA is an exact algorithm. On the contrary, while Zhang et al. (2023) describe both an exact and a heuristic algorithm, the publicly available version of LinearDesign contains only the heuristic utilizing a fixed-size beam search. Indeed, as discussed above there was one instance where LinearDesign returned a suboptimal solution (in terms of the lexicographical objective of prioritizing MFE first followed by CAI). We note the difference in running times between DERNA and CDSfold because DERNA takes into consideration both MFE and CAI, whereas CDSfold only considers MFE.

As discussed in Section 3.1.2, leaving out CAI from the objective value reduces the asymptotic running time from $O(m3LK^8)$ to $O(m^3LK^2)$ where m is the protein sequence length and K and L are constants corresponding to the maximum number of codons per amino acid and the maximum interior loop length, respectively. Indeed, this is also reflected in wall-clock times when running an altered version of DERNA that only considers MFE, reducing the running times to between 2 and 4951 seconds, closely matching those of LinearDesign (Fig. 4e). As expected, however, this comes at the expense of decreased CAI values for the inferred RNA sequences (Fig. 4c and Supplementary Fig. S2).

4.1.2. Balancing MFE and CAI. We now assess DERNA's ability to identify Pareto optimal solutions. To that end, we ran DERNA in λ-sweep mode with a termination threshold value of $\tau = 0.001$. Note that the number of λ values explored by DERNA depends on both the value of τ and the input instance itself. Unlike our method, LinearDesign does not include an automated way of altering their λ_{LD} parameter. As such, we varied $\lambda_{\text{LD}} \in [0, 100, 000]$—we did not set λ_{LD} to the same instance-specific λ values examined by DERNA as the two parameters play different roles in the corresponding objective functions of both methods.

However, for a fair comparison, we applied the same number of λ_{LD} values to LinearDesign as the λ values examined by DERNA for each instance. We refer to Supplementary Algorithm S1 in the Supplementary Data for precise details. Due to an increased number of runs per instance, we restricted our analysis to the 50 smallest instances with lengths ranging from 78 to 494 amino acids (Supplementary Table S1).

We begin by discussing the results for protein sequence P15421, which has 78 amino acids. DERNA examined 25 distinct λ values, leading to 12 distinct solutions (Fig. 5a). On the contrary, the list of 25 manually selected λ_{LD} values resulted in 7 distinct solutions identified by LinearDesign. Two of these solutions were identified by both DERNA and LinearDesign, Supplementary Fig. S4). Recall that $\lambda = 1$ prioritizes MFE for DERNA, whereas $\lambda = 0$ prioritizes CAI. Moreover, recall that each value of $\lambda \in [0, 1]$ leads to a Pareto optimal solution. A natural question is what is the smallest value λ_{MFE} that resulted in the optimal MFE? For protein sequence P15421 this was $\lambda_{\text{MFE}} = 0.070321$. Given that $\tau = 0.001$, this means that DERNA does not explore the part of the Pareto front that contains solutions with higher CAI values. Indeed, for this protein sequence, the largest nonoptimal CAI value identified by DERNA equals 0.996524, obtained using $\lambda = 0.000498$, followed by a CAI of 0.990816 using $\lambda = 0.000987$.

On the contrary, the largest nonoptimal CAI value identified by LinearDesign equals 0.999, which was obtained using $\lambda_{\text{LD}} = 40$. A downside of LinearDesign's objective function, which is of the form $\text{MFE}(\mathbf{v}, P) - \lambda_{\text{LD}} \cdot \log \overline{\text{CAI}}(\mathbf{v}, \mathbf{w})$, is that a nonbounded $\lambda_{\text{LD}} = \infty$ is required to exclusively prioritize CAI as opposed to a bounded value of $\lambda = 1$ for DERNA. Here, LinearDesign obtained this CAI-optimal solution only using $\lambda_{\text{LD}} \geq 100$. Supplementary Figure S5 demonstrates the effect of changing the termination threshold parameter τ—with lower values, we find that DERNA identifies more unique solutions, particularly those that favor CAI.

We now extend these analyses to all 50 protein sequences, where we found that DERNA identified 988 unique solutions and LinearDesign identified 830 unique solutions, with 383 solutions in common (Supplementary Fig. S6). Moreover, we observed that the median value of λ_{MFE}—the smallest λ that produces an MFE optimal solution—equals 0.089852 (Supplementary Fig. S7). For $\tau = 0.001$, the median number of λs examined by DERNA is 35 (Supplementary Fig. S8a), yielding a median number of 18 solutions

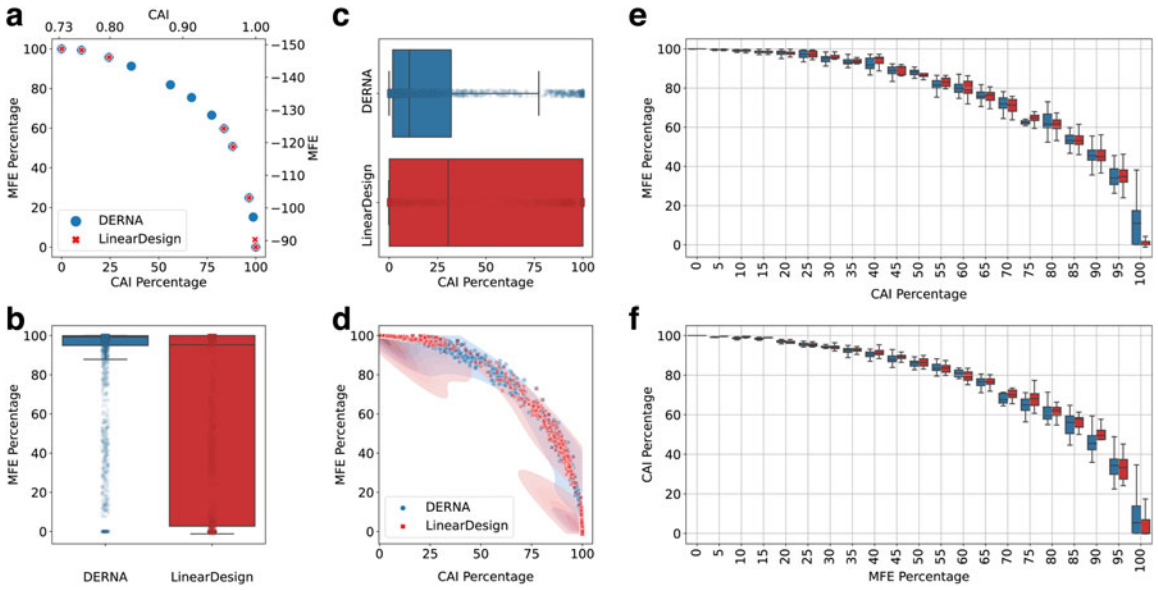

FIG. 5. DERNA models the trade-off between MFE and CAI. These analyses are restricted to the 50 smallest UniProt sequences in Supplementary Table S1. **(a)** Solutions identified by DERNA (*blue*) and LinearDesign (*red*) for proteins sequence P15421. The right *y*-axis shows the MFE, whereas the left *y*-axis shows the range-normalized MFE percentage. Similarly, the top *x*-axis shows the CAI, whereas the bottom *x*-axis shows the range-normalized CAI percentage. **(b–d)** MFE and CAI percentages inferred by both methods across all 50 instances. **(e)** For each instance, we show the best MFE percentage on the *y*-axis when only considering solutions that achieve the CAI percentage specified on the *x*-axis. **(f)** For each instance, we show the best CAI percentage on the *y*-axis when only considering solutions that achieve the MFE percentage specified on the *x*-axis.

(Supplementary Fig. S8b). The median number of λ_{LD} examined by LinearDesign is also 35, yielding a median number of 15 solutions (Supplementary Fig. S8c). To compare MFEs across instances, we define the MFE percentage as $(\text{MFE}(\mathbf{w}, \lambda) - \text{MFE}(\mathbf{w}, 0))/(\text{MFE}(\mathbf{w}, 1) - \text{MFE}(\mathbf{w}, 0))$ for each protein sequence $\mathbf{w}$ where $\text{MFE}(\mathbf{w}, \lambda)$ equals the MFE value of the solution obtained using λ. In other words, an MFE percentage of 100% means that the identified solution achieved the best possible MFE, whereas an MFE percentage of 0% means that the worst MFE that favors CAI was obtained.

We define CAI percentage similarly. Matching the previous analysis, we indeed see that DERNA favored the part of the Pareto front that prioritizes MFE (Fig. 5b). Conversely, for our choices of λ_{LD}, LinearDesign more heavily favored the part of the Pareto front that prioritizes CAI (Fig. 5c).

Finally, we delve more into the trade-off between CAI and MFE. To that end, we explored the following two questions. First, if one is willing to accept a certain CAI percentage, what is the best MFE that one can obtain? Second, for a specified minimum MFE percentage, what is the best CAI that one can obtain? Among the 50 considered instances, we found that if we accept solutions with a CAI percentage of at least 50%, the corresponding best MFE percentages for these solutions identified by DERNA range from 84.989% to 90.758% with a median of 88.077% (Fig. 5e). However, increasing the minimum CAI percentage to at least 80%, resulted in a decrease in best MFE of solutions identified by DERNA, with MFE percentages ranging from 52.398% to 72.967% with a median of 61.577%.

Conversely, for an MFE percentage of at least 50%, DERNA obtained solutions that have CAI percentages ranging from 82.755% to 89.290% with a median of 86.253% (Fig. 5f). Increasing the minimum MFE percentage to at least 80%, resulted in a decrease in best CAI of solutions identified by DERNA, with CAI percentages ranging from 54.946% to 71.356% with a median of 59.684%. When designing an RNA sequence for a target protein it is important to understand the trade-off between MFE and CAI, especially when trying to identify a single solution on the Pareto front.

4.2. Case study: SARS-CoV-2 spike protein

The spike (S) protein on the surface of the SARS-CoV-2 virus is responsible for recognizing and binding to the host cell's receptors, as well as merging itself with the host cell membrane, without which the virus

would be unable to interact with the host cells and initiate infection (Huang et al., 2020). The SARS-CoV-2 S protein, with its 1273 amino acids, is therefore the primary target of the Moderna and Pfizer-BioNTech mRNA vaccines (Salvatori et al., 2020).

We ran DERNA on the S protein with termination threshold $\tau = 0.0001$, 10 times smaller than the previous analysis in Section 4.1.2. DERNA evaluated 57 distinct λ values and generated 34 distinct solutions. Similarly, we applied LinearDesign to the S protein using the same number 57 values for λ_{LD} (sampled using Supplementary Algorithm S1). LinearDesign generated 25 distinct solutions corresponding to the 57 chosen λ values. The set of solutions obtained through LinearDesign overlaps with those generated by DERNA (Fig. 6a), with 3 identical solutions identified by both LinearDesign and DERNA.

Finally, we compared DERNA's solutions to the Pfizer-BioNTech and Moderna mRNA sequences. The Pfizer-BioNTech mRNA sequence has an MFE of -1217 and a CAI of 0.95 (Fig. 6b). For the same CAI value, DERNA identified a solution with a better MFE of -1955.2 (Fig. 6c). On the contrary, the Moderna mRNA sequence has an MFE of -1369.2 and a CAI of 0.98. Similarly, for the same CAI value, DERNA identified a solution with a better MFE of -1724.8. These two alternative solutions might lead to increased mRNA half-life without sacrificing the translational efficacy (Tuller et al., 2010; Mauger et al., 2019). We note that the overall minimum MFE equals -2486.7 with a corresponding CAI of 0.737 (Supplementary Fig. S9a), whereas solutions with overall maximum CAI of 1 lead to a decreased best MFE of -1384.3 (Supplementary Fig. S9b).

5. DISCUSSION

Given a target protein sequence $\mathbf{w}$, we introduced the PORD problem of identifying a set of Pareto optimal solutions $(\mathbf{v}, P)$ composed of an RNA sequence $\mathbf{v}$ that encodes for $\mathbf{w}$ and its corresponding secondary structure P that together balance the MFE and CAI. In addition, we introduced the BRD problem, where we additionally take as input the parameter $\lambda \in [0, 1]$ and return an RNA sequence $\mathbf{v}$ whose corresponding secondary structure P minimizes $\lambda \cdot \text{MFE}(\mathbf{v}, P) - (1-\lambda) \cdot \overline{\text{CAI}}(\mathbf{v}, \mathbf{w})$. To solve both problems, we introduced DERNA. Extending the work of Zuker and Stiegler (1981), DERNA solves the BRD problem via dynamic programming in $\mathcal{O}(|\mathbf{w}|^3)$ time and $\mathcal{O}(|\mathbf{w}|^2)$ space. In addition, DERNA solves the PORD problem via the weighted sum method (Zadeh, 1963), enumerating the Pareto front by solving multiple distinct instances of the BRD problem via a systematic sweep on λ.

On a benchmark data set of 100 protein sequences, we demonstrated that DERNA obtained solutions with an identical MFE but superior CAI compared with CDSfold (Terai et al., 2016), a previous approach that only optimizes MFE. In addition, we showed that DERNA matched LinearDesign's performance in terms of solution quality, a recent approach that similarly seeks to balance MFE and CAI. In addition, the

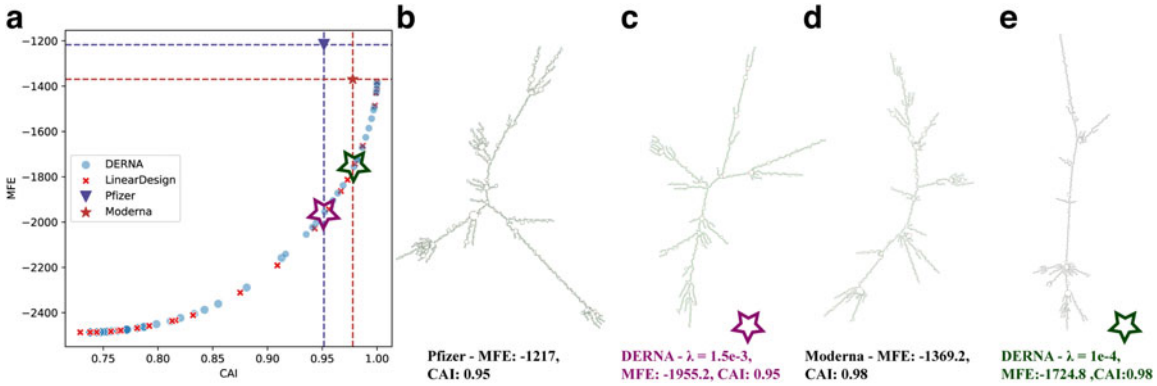

FIG. 6. DERNA identifies alternative sequences for the SARS-CoV-2 spike (S) protein. **(a)** Solution identified by DERNA (*blue*) and LinearDesign (*red*). **(b)** Secondary structures of the Pfizer-BioNTech and Moderna messenger RNA vaccine sequences and alternative solutions provided by DERNA, from left to right are Pfizer-BioNTech, DERNA with $\lambda = 1.5 \cdot 10^{-3}$, Moderna, and DERNA with $\lambda = 10^{-4}$ respectively.

key functionality of LinearDesign is closed source, whereas DERNA is fully open source. Finally, we demonstrated our method's potential for mRNA vaccine design using SARS-CoV-2 spike as the target protein.

For future development, it is important to consider other factors beyond CAI and MFE that affect protein expression levels. These factors include protein translation initiation motifs (Gingold and Pilpel, 2011) such as the Shine-Dalgarno sequence (Shine and Dalgarno, 1975) in prokaryotes or the Kozak sequence (Kozak, 1999) in eukaryotes that surround or occur upstream of the start codon, respectively. One can optimize for these initiation factors through the inclusion of an additional optimization criterion that evaluates the fit of the RNA sequence with $\mathbf{v}$ with the corresponding initiation motif, similarly to how we evaluate CAI. An additional factor that is positively correlated with increased protein expression is the presence of low secondary structure in the $5'$ untranslated region as well as the first ~ 30 nucleotides of the coding sequence (Ding et al., 2014; Wan et al., 2014; Tuller and Zur, 2015; Mauger et al., 2019).

To accomplish this goal, we will probably require similar techniques as used in a traditional RNA design where one seeks an RNA sequence that folds into a desired RNA secondary structure (Hofacker et al., 1994; Kleinkauf et al., 2015)—in this particular case we desire low amounts of secondary structure in a prefix of $\mathbf{v}$. We believe that the Pareto optimization framework developed in this article will support the inclusion of these two and potentially other additional optimization criteria. In addition, it will be valuable to investigate computing the Pareto front through algebraic dynamic programming (Saule and Giegerich, 2015). Alternatively, rather than using the weighted sum method (Zadeh, 1963) as done in this work, one could explore more sophisticated approaches for enumerating the Pareto front (Das and Dennis, 1998). Finally, one can use MFE and CAI criteria to prioritize neoantigens for inclusion in personalized mRNA cancer vaccines (He et al., 2022; Vishweshwaraiah and Dokholyan, 2022).

AUTHORS' CONTRIBUTIONS

X.G.: Conceptualization, implementation, and formal analysis. Y.Q.: Conceptualization and formal analysis. M.E.-K.: Conceptualization, validation, and writing—review and editing.

AUTHOR DISCLOSURE STATEMENT

The authors declare they have no conflicting financial interests.

FUNDING INFORMATION

M.E-K. was supported by the National Science Foundation (CCF-2046488) as well as funding from the Cancer Center at Illinois.

SUPPLEMENTARY MATERIAL

Supplementary Data SA.1
Supplementary Data SA.2
Supplementary Figure S1
Supplementary Figure S2
Supplementary Figure S3
Supplementary Figure S4
Supplementary Figure S5
Supplementary Figure S6
Supplementary Figure S7
Supplementary Figure S8
Supplementary Figure S9
Supplementary Table S1

REFERENCES

Cohen, B., and Skiena, S. 2003. Natural selection and algorithmic design of mRNA. *J. Comput. Biol.* 10, 419–432.

Cohon, J.L. 2004. *Multiobjective Programming and Planning*, volume 140. Courier Corporation.

Crommelin, D.J., Anchordoquy, T.J., Volkin, D.B., et al. 2021. Addressing the cold reality of mRNA vaccine stability. *J. Pharm. Sci*. 110, 997–1001.

Das, I., and Dennis, J.E. 1997. A closer look at drawbacks of minimizing weighted sums of objectives for Pareto set generation in multicriteria optimization problems. *Struct. Optim*. 14, 63–69.

Das, I., and Dennis, J.E. 1998. Normal-boundary intersection: A new method for generating the Pareto surface in nonlinear multicriteria optimization problems. *SIAM J. Optim.* 8, 631–657.

Ding, Y., Tang, Y., Kwok, C.K., et al. 2014. In vivo genome-wide profiling of RNA secondary structure reveals novel regulatory features. *Nature* 505, 696–700.

Freier, S.M., Kierzek, R., Jaeger, J.A., et al. 1986. Improved free-energy parameters for predictions of RNA duplex stability. *Proc. Natl Acad. Sci. U. S. A.* 83, 9373–9377.

Gingold, H., and Pilpel, Y. 2011. Determinants of translation efficiency and accuracy. *Mol. Syst. Biol.* 7, 481.

Grüning, B., Dale, R., Sjödin, A., et al. 2018. Bioconda: Sustainable and comprehensive software distribution for the life sciences. *Nat. Methods*, 15, 475–476.

Gustafsson, C., Govindarajan, S., and Minshull, J. 2004. Codon bias and heterologous protein expression. *Trends Biotechnol.* 22, 346–353.

He, Q., Gao, H., Tan, D., et al. 2022. mRNA cancer vaccines: Advances, trends and challenges. *Acta Pharm. Sin. B*, 12, 2969–2989.

Hofacker, I.L., Fontana, W., Stadler, P.F., et al. 1994. Fast folding and comparison of RNA secondary structures. *Monatsh. Chem*. 125, 167–167.

Huang, Y., Yang, C., Xu, X.-f., et al. 2020. Structural and functional properties of SARS-CoV-2 spike protein: Potential antivirus drug development for COVID-19. *Acta Pharmacol. Sin*. 41, 1141–1149.

Jaeger, J.A., Turner, D. H., and Zuker, M. 1989. Improved predictions of secondary structures for RNA. *Proc. Natl Acad. Sci. U. S. A.* 86, 7706–7710.

Kleinkauf, R., Mann, M., and Backofen, R. 2015. antaRNA: Ant colony-based RNA sequence design. *Bioinformatics* 31, 3114–3121.

Kozak, M. 1999. Initiation of translation in prokaryotes and eukaryotes. *Gene* 234, 187–208.

Lyngso, R.B., Zuker, M., and Pedersen, C. 1999. Fast evaluation of internal loops in RNA secondary structure prediction. *Bioinformatics* 15, 440–445.

Mahase, E. 2020. Covid-19: Moderna vaccine is nearly 95% effective, trial involving high risk and elderly people shows. *BMJ* 371, m4471.

Mathews, D.H., Sabina, J., Zuker, M., et al. 1999. Expanded sequence dependence of thermodynamic parameters improves prediction of RNA secondary structure. *J. Mol. Biol.* 288, 911–940.

Mathews, D.H., Disney, M.D., Childs, J.L., et al. 2004. Incorporating chemical modification constraints into a dynamic programming algorithm for prediction of RNA secondary structure. *Proc. Natl Acad. Sci. U. S. A.* 101, 7287–7292.

Mauger, D.M., Cabral, B.J., Presnyak, V., et al. 2019. mRNA structure regulates protein expression through changes in functional half-life. *Proc. Natl Acad. Sci. U. S. A.* 116, 24075–24083.

Meo, S., Bukhari, I., Akram, J., et al. 2021. COVID-19 vaccines: Comparison of biological, pharmacological characteristics and adverse effects of Pfizer/BioNTech and Moderna vaccines. *Eur. Rev. Med. Pharmacol. Sci.* 25, 1663–1669.

Nakamura, Y., Gojobori, T., and Ikemura, T. 2000. Codon usage tabulated from international DNA sequence databases: Status for the year 2000. *Nucleic Acids Res.* 28, 292–292.

Poznanović, S., Barrera-Cruz, F., Kirkpatrick, A., et al. 2020. The challenge of RNA branching prediction: A parametric analysis of multiloop initiation under thermodynamic optimization. *J. Struct. Biol.* 210, 107475.

Presnyak, V., Alhusaini, N., Chen, Y.-H., et al. 2015. Codon optimality is a major determinant of mRNA stability. *Cell* 160, 1111–1124.

Salvatori, G., Luberto, L., Maffei, M., et al. 2020. SARS-CoV-2 spike protein: An optimal immunological target for vaccines. *J. Transl. Med.* 18, 222.

Saule, C., and Giegerich, R. 2015. Pareto optimization in algebraic dynamic programming. *Algorithms Mol. Biol.* 10, 1–20.

Sharp, P.M., and Li, W.-H. 1987. The codon adaptation index-a measure of directional synonymous codon usage bias, and its potential applications. *Nucleic Acids Res*. 15, 1281–1295.

Shine, J., and Dalgarno, L. 1975. Determinant of cistron specificity in bacterial ribosomes. *Nature* 254, 34–38.

Terai, G., Kamegai, S., and Asai, K. 2016. CDSfold: An algorithm for designing a protein-coding sequence with the most stable secondary structure. *Bioinformatics* 32, 828–834.

The UniProt Consortium. 2022. UniProt: The universal protein knowledgebase in 2023. *Nucleic Acids Res.* 51(D1), D523–D531.

Tuller, T., and Zur, H. 2015. Multiple roles of the coding sequence 5' end in gene expression regulation. *Nucleic Acids Res.* 43, 13–28.

Tuller, T., Waldman, Y.Y., Kupiec, M., et al. 2010. Translation efficiency is determined by both codon bias and folding energy. *Proc. Natl Acad. Sci. U. S. A.* 107, 3645–3650.

Turner, D.H., Sugimoto, N., and Freier, S.M. 1988. RNA structure prediction. *Annu. Rev. Biophys. Biophys. Chem.* 17, 167–192.

Vishweshwaraiah, Y.L., and Dokholyan, N.V. 2022. mRNA vaccines for cancer immunotherapy. *Front. Immunol.* 13:1029069.

Wan, Y., Qu, K., Zhang, Q.C., et al. 2014. Landscape and variation of RNA secondary structure across the human transcriptome. *Nature* 505, 706–709.

Wayment-Steele, H.K., Kim, D.S., Choe, C.A., et al. 2021. Theoretical basis for stabilizing messenger RNA through secondary structure design. *Nucleic Acids Res.* 49, 10604–10617.

Weinberg, D.E., Shah, P., Eichhorn, S.W., et al. 2016. Improved ribosome-footprint and mRNA measurements provide insights into dynamics and regulation of yeast translation. *Cell Rep.* 14, 1787–1799.

Zadeh, L. 1963. Optimality and non-scalar-valued performance criteria. *IEEE Trans. Automat. Contr.* 8, 59–60.

Zhang, H., Zhang, L., Lin, A., et al. 2023. Algorithm for optimized mRNA design improves stability and immunogenicity. *Nature* 621, 396–403.

Zuker, M., and Stiegler, P. 1981. Optimal computer folding of large RNA sequences using thermodynamics and auxiliary information. *Nucleic Acids Res.* 9, 133–148.

Address correspondence to:
Dr. Mohammed El-Kebir
Department of Computer Science
University of Illinois Urbana-Champaign
201 N Goodwin Avenue
Urbana, IL 61801
USA

E-mail: melkebir@illinois.edu