Supporting Information

LinearPartition: Linear-Time Approximation of RNA Folding Partition Function and Base Pairing Probabilities

He Zhang, Liang Zhang, David H. Mathews and Liang Huang

A Details of the Efficient Implementation

A.1 Data Structures

In the main text, for simplicity of presentation, Q is described as a hash from span $[i,j]$ to $Q_{i,j}$, but in our actual implementation, to make sure the overall runtime is $O(nb^2)$, we implement Q as an array of n hashes, where each $Q[j]$ is a hash mapping i to $Q[j][i]$ which is conveniently notated as $Q_{i,j}$ in the main text. It is important to note that the first dimension j is the right boundary and the second dimension i is the left boundary of the span $[i,j]$. See the following table for a summary of notations and the corresponding actual implementations. Here we use Python notation for simplicity, but in actual system we implement with C++.

notations in this paper	Python implementation
$Q \leftarrow$ hash()	`Q = [defaultdict(float) for _ in range(n)]`
$Q_{i,j}$	`Q[j][i]`
$[i,j]$ in Q	`i in Q[j]`
for each i such that $[i,j]$ in Q	`for i in Q[j]`
delete $[i,j]$ from Q	`del Q[j][i]`

A.2 Complexity Analysis

In the partition function calculation (inside phase) in Fig. 3, the number of states is $O(nb)$ because each $Q[j]$ contains at most b states ($Q_{i,j}$'s) after pruning. Therefore the space complexity is $O(nb)$. For time complexity, there are three nested loops, the first one (j) with n iterations, the second (i) and the third (k) loops both have $O(b)$ iterations thanks to pruning, so the overall runtime is $O(nb^2)$.

A.3 Outside Partition Function and Base Pairing Probability Calculation

After we compute the partition functions $Q_{i,j}$ on each span $[i,j]$ (known as the "inside partition function"), we also need to compute the complementary function $\widehat{Q}_{i,j}$ for each span known as the "outside partition function" in order to derive the base-pairing probabilities. Unlike the inside phase, this outside partition function is calculated from top down, with $\widehat{Q}_{1,n} = 1$ as the base case.

$$\begin{aligned}\widehat{Q}_{i,j} &= \widehat{Q}_{i,j+1} \cdot e^{-\frac{\delta(\mathbf{x},j+1)}{RT}} \\ &+ \sum_{k<i} \widehat{Q}_{k,j+1} \cdot Q_{k,i-2} \cdot e^{-\frac{\xi(\mathbf{x},i-1,j+1)}{RT}} \\ &+ \sum_{k>j+1} \widehat{Q}_{i,k} \cdot Q_{j+2,k-1} \cdot e^{-\frac{\xi(\mathbf{x},j+1,k)}{RT}}\end{aligned}$$

Note that the second line is only possible when $x_{i-1}x_{j+1}$ can form a base pair (otherwise $e^{-\frac{\xi(\mathbf{x},i-1,j+1)}{RT}} = 0$) and the third line has a constraint that $x_{j+1}x_k$ can form a base pair (otherwise $e^{-\frac{\xi(\mathbf{x},j+1,k)}{RT}} = 0$).

For each (i,j) where $x_i x_j$ can form a base pair, we compute its pairing probability:

$$p_{i,j} = \sum_{k \le i} \widehat{Q}_{k,j} \cdot Q_{k,i-1} \cdot e^{-\frac{\xi(\mathbf{x},i,j)}{RT}} \cdot Q_{i+1,j-1}$$

The whole "outside" computation takes $O(n^3)$ without pruning, but also $O(nb^2)$ with beam pruning. See Fig. SI 2 for the pseudocode to compute the outside partition function and base pairing probabilities.

B Details of datasets, baselines and methods

B.1 Datasets

We use sequences from two datasets, ArchiveII and RNAcentral. The archiveII dataset (available in `http://rna.urmc.rochester.edu/pub/archiveII.tar.gz`) is a diverse set with 3,857 RNA sequences and their secondary structures. It is first curated in the 1990s to contain sequences with structures that were well-determined by comparative sequence analysis (Mathews *et al.*, 1999)and updated later with additional structures (Sloma and Mathews, 2016). We remove 957 sequences that appear both in the ArchiveII and the S-Processed datasets (Andronescu *et al.*, 2007), because CONTRAfold uses S-Processed for training. We also remove all 11 Group II Intron sequences because there are so few instances of these that are available electronically. Additionally, we removed 30 sequences in the tmRNA family because the annotated structure for each of these sequences contains fewer than 4 pseudoknots, which suggests the structures are incomplete. These preprocessing steps lead to a subset of ArchiveII with 2,859 reliable secondary structure examples distributed in 9 families. See SI 1 for the statistics of the sequences we use in the ArchiveII dataset. Moreover, we randomly sampled 22 longer RNA sequences (without known structures) from RNAcentral (RNAcentral Consortium et al., 2017) (`https://rnacentral.org/`), with sequence lengths ranging from 3,048 *nt* to 244,296 *nt*. For the sampling, we evenly split the range from $3,000$ to $244,296$ (the longest) into 24 bins by log-scale, and for each bin we randomly select a sequence (there are bins with no sequences).

To show the approximation quality on random RNA sequences, we generated 30 sequences with uniform distribution over {A, C, G, U}. The lengths of these sequences are spaced in 100 nucleotide intervals from 100 to 3,000.

	# of seqs		length		
Family	total	used	avg	max	min
tRNA	557	74	77.3	88	58
5S rRNA	1,283	1,125	118.8	135	102
SRP RNA	928	886	186.1	533	28
RNase P RNA	454	182	344.1	486	120
tmRNA	462	432	369.1	433	307
Group I Intron	98	96	424.9	736	210
Group II Intron	11	0	-	-	-
telomerase RNA	37	37	444.6	559	382
16S rRNA	22	22	1,547.9	1995	950
23S rRNA	5	5	2,927.4	2968	2904
Overall	3,846	2,859	221.1	2968	28

Table SI 1. Statistics of the sequences in the ArchiveII dataset used in this work.

B.2 Baseline Software

We use two baseline software packages: (1) Vienna RNAfold (Version 2.4.11) from `https://www.tbi.univie.ac.at/RNA/download/sourcecode/2_4_x/ViennaRNA-2.4.11.tar.gz` and (2) CONTRAfold (Version 2.0.2) from `http://contra.stanford.edu/`. Vienna RNAfold is a widely-used RNA structure prediction package, while CONTRAfold is a successful machine learning-based RNA structure prediction system. Both provide partition function and base pairing probability calculations based on the classical cubic runtime algorithm. Our comparisons mainly focus on the systems with the same model, i.e., LinearPartition-V vs. Vienna RNAfold and LinearPartition-C vs. CONTRAfold. In this way the differences are based on algorithms themselves rather than models. We found a bug in CONTRAfold by comparing our results to CONTRAfold, which led to overcounting multiloops in the partition function calculation. We corrected the bug, and all experiments are based on this bug-fixed version of CONTRAfold.

B.3 Evaluation Metrics and Significance Test

Due to the uncertainty of base-pair matches existing in comparative analysis and the fact that there is fluctuation in base pairing at equilibrium, we consider a base pair to be correctly predicted if it is also displaced by one nucleotide on a strand (Mathews *et al.*, 1999). Generally, if a pair (i, j) is in the predicted structure, we consider it a correct prediction if one of (i, j), $(i-1, j)$, $(i+1, j)$, $(i, j-1)$, $(i, j+1)$ is in the ground truth structure.

We use Positive Predictive Value (PPV) and sensitivity as accuracy measurements. Formally, denote $\mathbf{y}$ as the predicted structure and $\mathbf{y}^*$ as the ground truth, we have:

$$\text{PPV} = \frac{\#_{\text{TP}}}{\#_{\text{TP}} + \#_{\text{FP}}} = \frac{|\text{pairs}(\mathbf{y}) \cap \text{pairs}(\mathbf{y}^*)|}{|\text{pairs}(\mathbf{y})|}$$

$$\text{Sensitivity} = \frac{\#_{\text{TP}}}{\#_{\text{TP}} + \#_{\text{FN}}} = \frac{|\text{pairs}(\mathbf{y}) \cap \text{pairs}(\mathbf{y}^*)|}{|\text{pairs}(\mathbf{y}^*)|}$$

where $\#_{\text{TP}}$ is the number of true positives (correctly predicted pairs), $\#_{\text{FP}}$ is the number of false positives (wrong predicted pairs) and $\#_{\text{FN}}$ is the number of false negatives (missing ground truth pairs).

We test statistical significance using a paired, two-sided permutation test (Aghaeepour and Hoos, 2013). We follow the common practice, choosing $10,000$ as the repetition number and $\alpha = 0.05$ as the significance threshold.

B.4 Curve Fitting

We determine the best exponent a for the scaling curve $O(n^a)$ for each data series in Figures 2 and 4. Specifically, we use $f(x) = ax + b$ to fit the log-log plot of those series in Gnuplot; e.g., fitting $\log t_n = a \log n + b$, where t_n is the running time on a sequence of length n, so that $t_n = e^b n^a$. Gnuplot uses the nonlinear least-squares Marquardt-Levenberg algorithm.

C Supporting Figures and Tables

```
1: function beamprune($Q, j, b$)
2:   $candidates \leftarrow$ hash()                          ▷ hash table: from candidates $i$ to score
3:   for each $i$ such that $[i, j]$ in $Q$ do
4:     $candidates[i] \leftarrow Q_{1,i-1} \cdot Q_{i,j}$     ▷ like LinearFold, use $Q_{1,i-1}$ as prefix score
5:   $candidates \leftarrow$ SelectTopB$(candidates, b)$      ▷ select top-$b$ states by score
6:   for each $i$ such that $[i, j]$ in $Q$ do
7:     if key $i$ not in $candidates$ then
8:       delete $[i, j]$ from $Q$                           ▷ prune low-scoring states
```

Fig. SI 1. The BeamPrune function from the Pseudocode of our main algorithm (Fig. 3).

```
1: function BasePairingProbs($\mathbf{x}$, $Q$)                 ▷ outside calculation
2:   $n \leftarrow$ length of $\mathbf{x}$
3:   $\widehat{Q} \leftarrow$ hash()                  ▷ hash table: from span $[i, j]$ to $\widehat{Q}_{i,j}$: outside partition function
4:   $p \leftarrow$ hash()                           ▷ hash table: from span $[i, j]$ to $p_{i,j}$: base-pairing probs
5:   $\widehat{Q}_{1,n} \leftarrow 1$                 ▷ base case
6:   for $j = n$ downto 1 do
7:     for each $i$ such that $[i, j-1]$ in $Q$ do
8:       $\widehat{Q}_{i,j-1} \mathrel{+}= \widehat{Q}_{i,j} \cdot e^{-\frac{\delta(\mathbf{x},j)}{RT}}$      ▷ skip
9:       if $x_{i-1}x_j$ in {AU, UA, CG, GC, GU, UG} then
10:        for each $k$ such that $[k, i-2]$ in $Q$ do
11:          $\widehat{Q}_{k,i-2} \mathrel{+}= \widehat{Q}_{k,j} \cdot Q_{i,j-1} \cdot e^{-\frac{\xi(\mathbf{x},i-1,j)}{RT}}$      ▷ pop: left
12:          $\widehat{Q}_{i,j-1} \mathrel{+}= \widehat{Q}_{k,j} \cdot Q_{k,i-2} \cdot e^{-\frac{\xi(\mathbf{x},i-1,j)}{RT}}$      ▷ pop: right
13:          $p_{i-1,j} \mathrel{+}= \frac{\widehat{Q}_{k,j} \cdot Q_{k,i-2} \cdot e^{-\frac{\xi(\mathbf{x},i-1,j)}{RT}} \cdot Q_{i,j-1}}{Q_{1,n}}$      ▷ accumulate base pairing probs
14:  return $p$                                   ▷ return the (sparse) base-pairing probability matrix
```

Fig. SI 2. Outside partition function and base pairing probabilities calculation for a simplified version of the LinearPartition. Q is the (inside) partition function calculated by the pseudocode in Fig. 3, and $\widehat{Q}$ is the outside partition function. The actual algorithm using the Turner model is in our GitHub codebase.

Family	percentage of violation sequences (%)	avg. violation ratio per sequence (%)
tRNA	85.1	8.6
5S rRNA	100.0	27.0
SRP RNA	92.3	33.1
RNase P RNA	100.0	45.0
tmRNA	100.0	40.6
Group I Intron	100.0	43.3
telomerase RNA	100.0	53.9
16S rRNA	100.0	52.3
23S rRNA	100.0	53.3
Overall	97.5	39.7

Table SI 2. Statistic of negative unpaired probabilities for Vienna RNAplfold. RNAplfold base pairing probabilities are not normalized, resulting in negative unpaired probabilities. The second column (percentage of violation sequences) shows the percentage of sequences that have at least one nucleotide with negative unpaired probability. The third column (avg. violation ratio per sequence) shows the percentage of nucleotides that have negative unpaired probability per sequence. The overall ratio is averaged on families. To avoid precision issues, we set the negative threshold as -0.01.

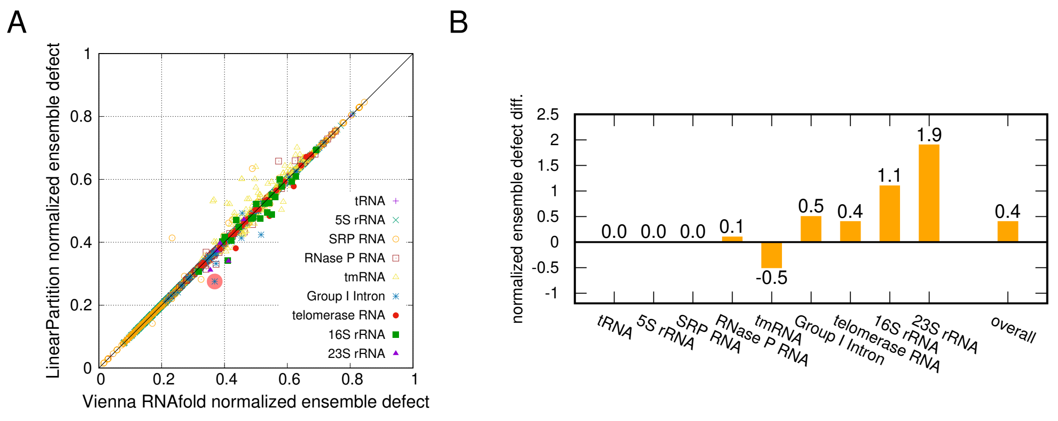

Fig. SI 3. The comparison of normalized ensemble defects (normalized by sequence length) on the ArchiveII dataset. **A**: Normalized ensemble defect between Vienna RNAfold and LinearPartition-V for each sequence; the trend is similar as Fig. 5A, but the deviations for tmRNAs are more apparent; the point with red shaded are the example in Fig. 6. **B**: Normalized ensemble defect difference for each family; for longer families, e.g., Group I Intron, telomerase RNA, 16S and 23S rRNA, LinearPartition has lower normalized ensemble defect differences; note that LinearPartition's normalized ensemble defects are significantly better than Vienna RNAfold on Group I Intron ($p < 0.01$), but significantly worse on tmRNA ($p < 0.01$).

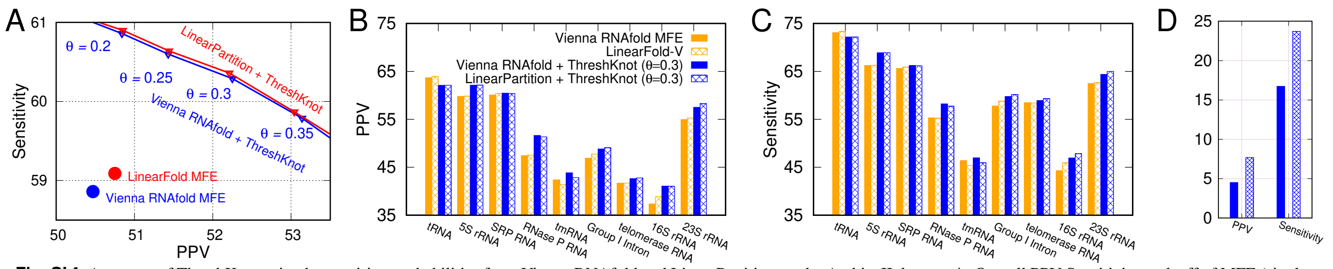

Fig. SI 4. Accuracy of ThreshKnot using base pairing probabilities from Vienna RNAfold and LinearPartition on the ArchiveII dataset. **A**: Overall PPV-Sensitivity tradeoff of MFE (single point) and ThreshKnot with varying θ. **B** & **C**: PPV and Sensitivity comparisons of ThreshKnot structures for each family. **D**: Accuracy comparison of long-distance base pairs (>500 *nt* apart) in the ThreshKnot structures. We conclude that ThreshKnot predictions based on those two are almost identical for all θ's. LinearPartition-V is substantially better on long-range base pairs in ThreshKnot predictions.

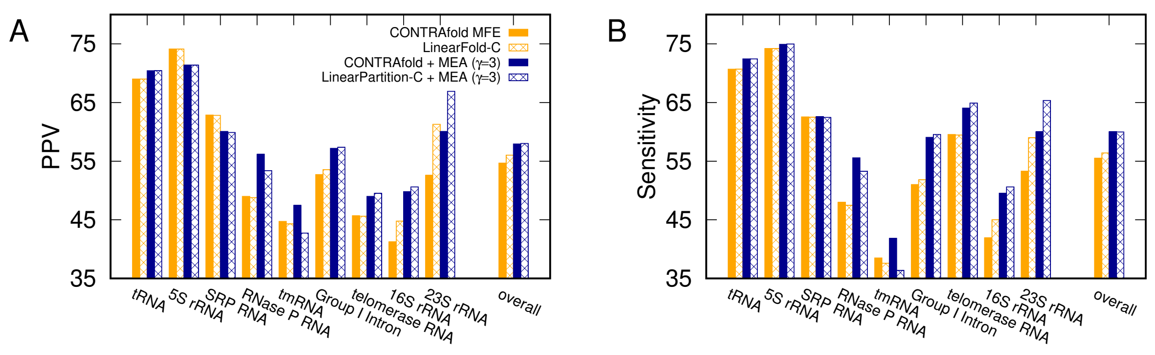

Fig. SI 5. Accuracy comparison of MEA structures ($\gamma = 3$) between CONTRAfold and LinearPartition-C on the ArchiveII dataset. γ is the hyperparameter balances PPV and Sensitivity. Note that LinearPartition-C + MEA is significantly worse than CONTRAfold + MEA on two families in both PPV and Sensitivity, tmRNA and RNase P RNA ($p < 0.01$).

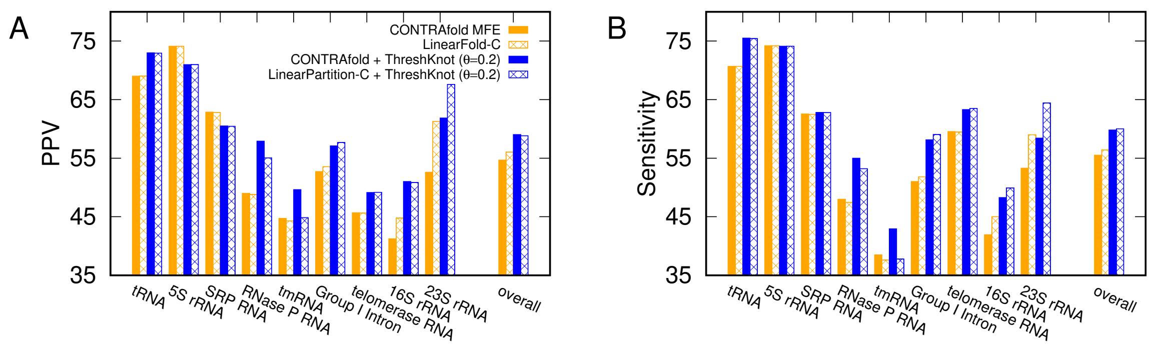

Fig. SI 6. Accuracy comparison of ThreshKnot structure ($\theta = 0.2$) between CONTRAfold and LinearPartition-C on ArchiveII dataset. θ is the hyperparameter that balances PPV and Sensitivity. Note that LinearPartition-C + ThreshKnot is significantly worse than CONTRAfold + ThreshKnot on two families in both PPV and Sensitivity, tmRNA and RNase P RNA ($p < 0.01$), and significantly better on three longer families in Sensitivity, Group I Intron ($p < 0.01$), telomerase RNA and 16S rRNA ($0.01 \leq p < 0.05$).

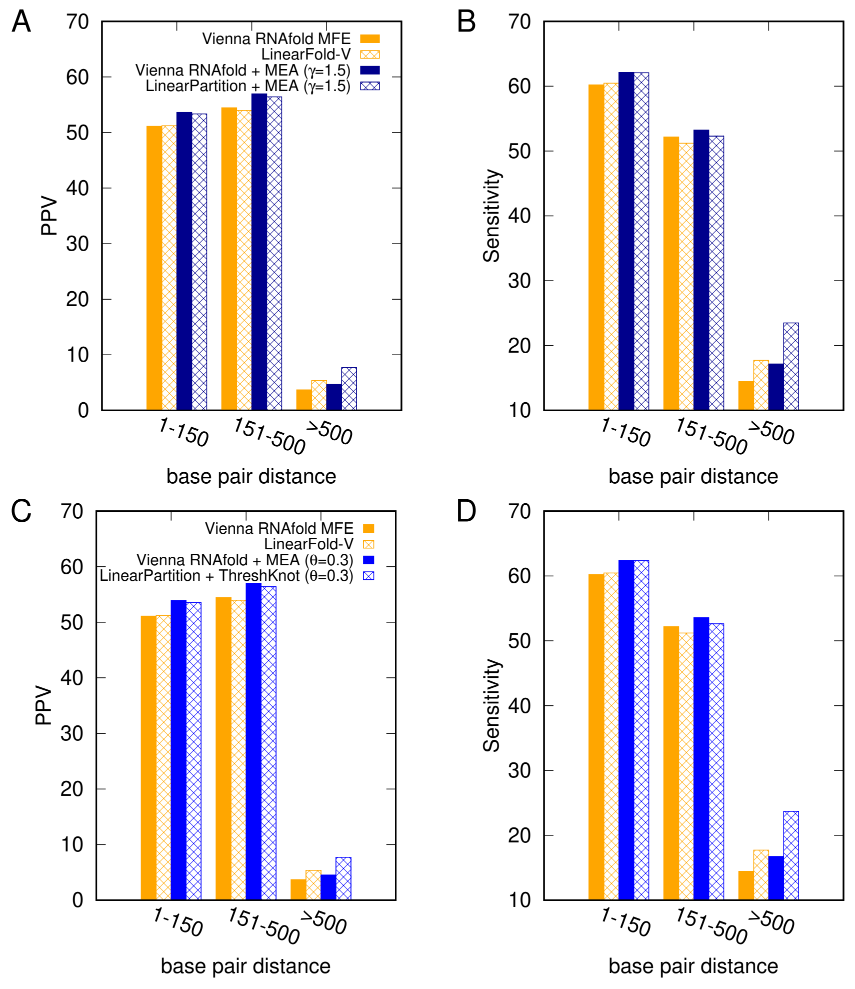

Fig. SI 7. Accuracy comparison of base pair prediction with different base pair distances. Each bar represents the overall PPV/sensitivity of all predicted base pairs in a certain length range across all sequences. LinearPartition performs best on long base pairs over four systems. **A** and **B**: Comparison using MEA structures. **C** and **D**: Comparison using ThreshKnot structures. In all cases, LinearPartition's base pair probabilities lead to substantially better accuracies on long-distance pairs (500+ *nt* apart).

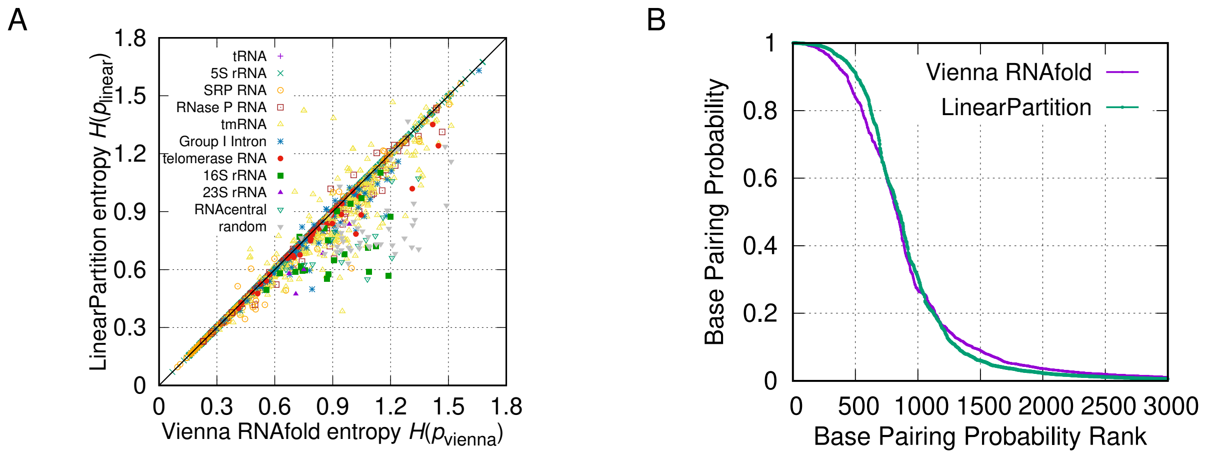

Fig. SI 8. Comparison of the uncertainty of the base pairing probability distribution from Vienna RNAfold and LinearPartition. **A**: Average positional structural entropy $H(p)$ comparison; LinearPartition has noticeably lower entropy. **B**: LinearPartition starts higher and finishes lower than Vienna RNAfold in a sorted probability curve for *E. coli* 23S rRNA, suggesting lower entropy.

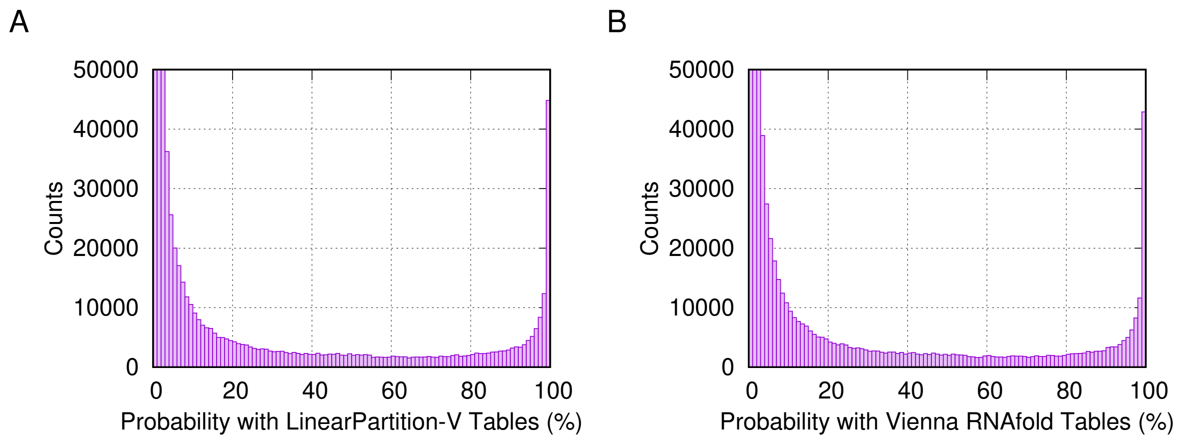

Fig. SI 9. Pair probability distributions of Vienna RNAfold and LinearPartition-V are similar. **A**: Pair probability distribution of Vienna RNAfold; **B**: Pair probability distribution of LinearPartition-V. The count of LinearPartition-V in bin [99,100) is slightly bigger than Vienna RNAfold, while the count in bin [0,1) (cut here at 50,000) is much less than Vienna RNAfold (2,068,758 for LinearPartition-V and 48,382,357 for Vienna RNAfold).

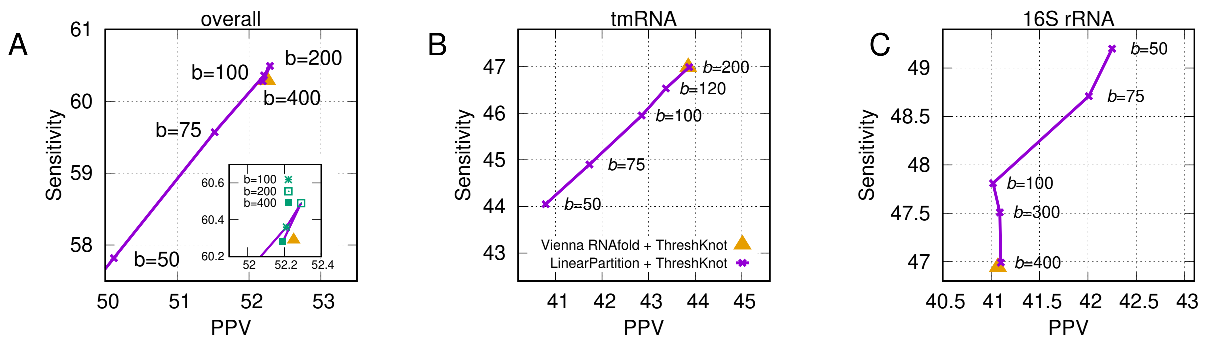

Fig. SI 10. Impact of beam size on accuracy. **A**: Overall PPV and Sensitivity with beam size. **B–C**: tmRNA and 16S rRNA PPV and Sensitivity against beam size, respectively. Note that the results of ThreshKnot using RNAfold (yellow triangles in **A–C**) are identical to ThreshKnot using the exact version of LinearPartition ($b = \infty$).

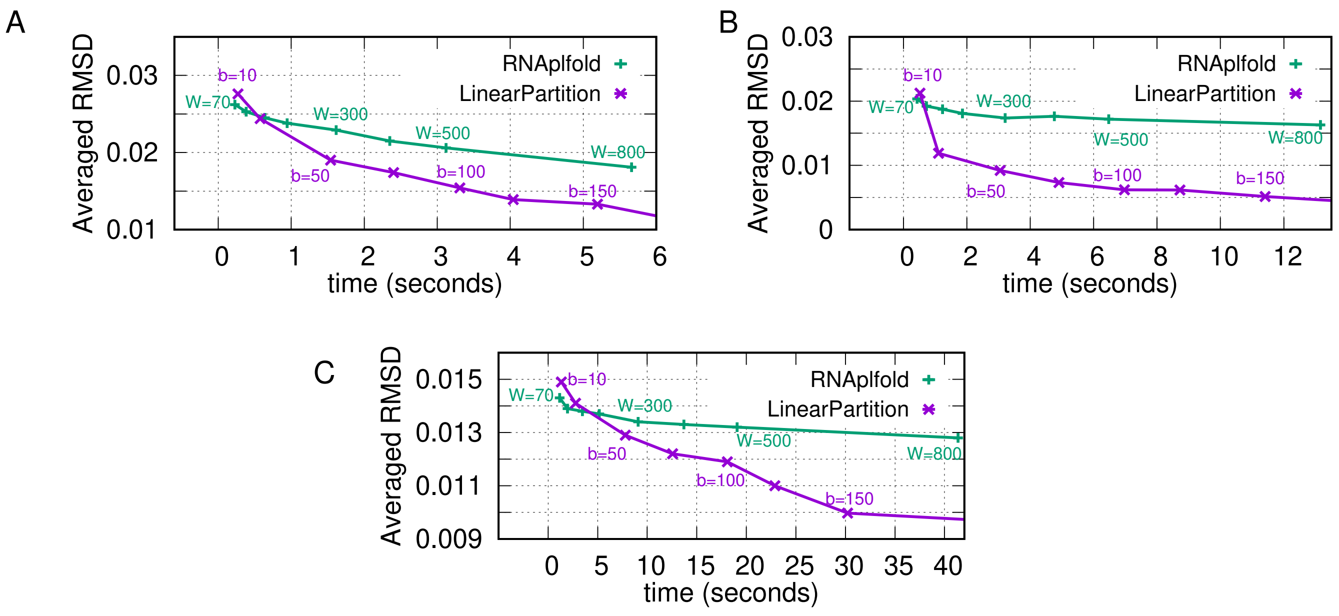

Fig. SI 11. Approximation quality comparisons of Vienna RNAplfold and LinearPartition on long sequences. Given the same runtime (x-axis), the averaged RMSDs of LinearPartition are smaller than RNAplfold in most cases. **A**: Comparison on 16S rRNA family. **B**: Comparison on 23S rRNA family. **C**: Comparison on lncRNAs from RNAcentral. In **A–C**, we choose window size of 70, 100, 150, 200, 300, 500, 800 for RNAplfold, and beam size of 10, 20, 50, 75, 100, 120, 150, 200 for LinearPartition.

input	$x_1 \dots x_n$	
states	$\mathsf{E}\ \langle 0,\ j\rangle : \langle \alpha, Q\rangle$	prefix structure
	$\mathsf{P}\ \langle i,\ j\rangle : \langle\ (\alpha)\ , Q\rangle$	pair
	$\mathsf{H}\ \langle i,\ j\rangle : \langle\ \mathbf{(} \dots , Q\rangle$	hairpin candidate
	$\mathsf{M}_1\ \langle i,\ j\rangle : \langle\ (\alpha)\,\beta, Q\rangle$	one or more pairs
	$\mathsf{M}_2\ \langle i,\ j\rangle : \langle\ (\alpha)\,\beta\,(\gamma)\ , Q\rangle$	two or more pairs
	$\mathsf{M}\ \langle i,\ j\rangle : \langle\ \mathbf{(} \dots (\alpha)\,\beta\,(\gamma) \dots , Q\rangle$	multiloop candidate
	Shaded substrings are balanced in brackets.	
	$Q^{X}_{i,j}$ is the partition function of state $X\ \langle i, j\rangle$	
	$Q^{X}_{i,j} \leftarrow 0$ for all $1 \leq i < j \leq n, X \in \{\mathsf{P}, \mathsf{M}_1, \mathsf{M}_2, \mathsf{M}\}$	
axiom	$Q^{\mathsf{E}}_{0,1} \leftarrow 1$	goal $\mathsf{E}\ \langle 0,\ n{+}1\rangle : \langle \alpha, Q\rangle$
push	$Q^{\mathsf{H}}_{j,\mathtt{next}(j,j)} \leftarrow 1$	
Hjump	$Q^{\mathsf{H}}_{i,\mathtt{next}(i,j)} \leftarrow 1$	$\mathrm{next}(i,j) \triangleq \min\{k \mid k > j,\ (x_i, x_k) \text{ match}\}$
skip	$Q^{\mathsf{E}}_{0,j+1} \mathrel{+}= Q^{\mathsf{E}}_{0,j} \cdot e^{-\frac{sc^{\mathrm{E}}_{\mathbf{w}}(\mathbf{x},j,j+1)}{RT}}$	
	$Q^{\mathsf{M}_1}_{i,j+1} \mathrel{+}= Q^{\mathsf{M}_1}_{i,j} \cdot e^{-\frac{w^{\text{multi}}_{\text{unpair}}}{RT}}$	
reduce	$Q^{\mathsf{M}_2}_{k,j} \mathrel{+}= \sum_{k<i<j} Q^{\mathsf{M}_1}_{k,i} \cdot Q^{P}_{i,j} \cdot e^{-\frac{w^{\text{multi}}_{\text{bp}}(\mathbf{x},i,j)}{RT}}$	
combine	$Q^{\mathsf{E}}_{0,j} \mathrel{+}= \sum_{0<i<j} Q^{\mathsf{E}}_{0,i} \cdot Q^{P}_{i,j} \cdot e^{-\frac{\mathrm{sc}^{\mathrm{E}}_{\mathbf{w}}(\mathbf{x},i,j)}{RT}}$	
XtoM$_1$	$Q^{\mathsf{M}_1}_{i,j} \mathrel{+}= Q^{\mathsf{P}}_{i,j} \cdot e^{-\frac{w^{\text{multi}}_{\text{bp}}(\mathbf{x},i,j)}{RT}}$	
	$Q^{\mathsf{M}_1}_{i,j} \mathrel{+}= Q^{\mathsf{M}_2}_{i,j}$	
Mleft	$Q^{\mathsf{M}}_{k,\mathtt{next}(k,j)} \mathrel{+}= Q^{\mathsf{M}_2}_{i,j} \cdot e^{-\frac{u \cdot w^{\text{multi}}_{\text{unpair}}}{RT}}$	$u = (\mathrm{next}(k,j){-}j){+}(i{-}k{-}1)$, $i{-}k{-}1 \leq 30$
Mjump	$Q^{\mathsf{M}}_{i,\mathtt{next}(i,j)} \mathrel{+}= Q^{\mathsf{M}}_{i,j} \cdot e^{-\frac{u \cdot w^{\text{multi}}_{\text{unpair}}}{RT}}$	$u = \mathrm{next}(i,j){-}j$
hairpin	$Q^{\mathsf{P}}_{i,j+1} \mathrel{+}= Q^{\mathsf{H}}_{i,j} \cdot e^{-\frac{sc^{\mathrm{H}}_{\mathbf{w}}(\mathbf{x},i,j)}{RT}}$	
singleloop	$Q^{\mathsf{P}}_{k,l} \mathrel{+}= Q^{\mathsf{P}}_{i,j} \cdot e^{-\frac{sc^{\mathrm{S}}_{\mathbf{w}}(\mathbf{x},i,j,k,l)}{RT}}$	(x_k, x_{l-1}) match, $(l{-}j{-}1){+}(i{-}k{-}1) \leq 30$
multiloop	$Q^{\mathsf{P}}_{i,j+1} \mathrel{+}= Q^{\mathsf{M}}_{i,j} \cdot e^{-\frac{w^{\text{multi}}_{\text{base}}+w^{\text{multi}}_{\text{bp}}(\mathbf{x},i,j)}{RT}}$	

Fig. SI 12. The partition function recursions used in LinearPartition real system. $sc^{\mathrm{E}}_{\mathbf{w}}(\mathbf{x},\cdot,\cdot)$, $w^{\text{multi}}_{\text{base}}$, $w^{\text{multi}}_{\text{bp}}(\mathbf{x},\cdot,\cdot)$, $w^{\text{multi}}_{\text{unpair}}$, $sc^{\mathrm{S}}_{\mathbf{w}}(\mathbf{x},\cdot,\cdot,\cdot,\cdot)$, $sc^{\mathrm{H}}_{\mathbf{w}}(\mathbf{x},\cdot,\cdot)$ are the various energy or scoring parameters (E stands for external loop, multi for multiloop, S for single loop, and H for hairpin loop). The $\mathtt{next}(i,j)$ returns the next position after x_j that can pair with x_i; CONTRAfold, ViennaRNA, and LinearFold also use the "jumping" trick.