1 Introduction

As the problem of ab initio folding of proteins is close to being solved [1], [2], we still got a long way for the same achievement with ab initio folding of biologically active RNA structures [3] - that is, the folding of the 3-dimensional structure of an RNA molecule given its primary structure. In this report, I look at some of the earliest milestones in this field.

1.1 RNA base pairing

RNA consists of the nucleotides (bases) A, C, G, and U. The bases pair following the Watson-Crick pairing, thus A and U pair while C and G pair. Further, a wobble pair between G and U exists [4]. The pairing of bases gives rise to the secondary structure of RNA and the features associated with it, e.g. loops and hairpins, see Figure 1.

1.2 The Nussinov algorithm

In 1978, Nussinov et al. [5] proposed a computational algorithm to determine maximum base pairs in an RNA sequence of n nucleotides. The algorithm recursively goes through all subsequences between the first and last nucleotide to determine the maximum possible base pairs on all subsequence of the RNA sequence [5], [6].

Given a sequence of length n, such that $nt_1, \dots, nt_n$, it is possible to recursively calculate the maximum number of base pairs for each substructure in the sequence $nt_{1+i}, \dots, nt_{j=n}$, with the assumption that the maximum number of base pairs for all smaller subsequences have been calculated earlier. This recursion can mathematically be expressed as [6]:

$$M(i,j) = max \begin{cases} M(i,j-1) \\ M(i+1,j) \\ M(i+1,j-1) + s(i,j) \\ \max_{i<k<j-1}\{M(i,k) + M(k+1,j)\} \end{cases}, \qquad \textit{(Eq. 1)}$$

$$s(i,j) = \begin{cases} 1 & if\ nt_i\ and\ nt_j\ pair \\ 0 & if\ nt_i\ and\ nt_j\ don't\ pair \end{cases} \qquad \textit{(Eq. 2)}$$

In equation 1, $M(i,j)$ is the maximum base pairs in the subsequence $nt_i, \dots, nt_j$ given the four conditions. The fourth condition in $M(i,j)$ is a recursion accounting for branches, that is possible base pair between nt_i and nt_k for any subsequence in $i > k > j$ [6]. All values for maximum base pairs are put into a matrix M of dimensions $\{n \times n\}$.

To get a secondary structure, a backtracking step is needed. The backtracking starts at $M(m,n)$ $m = 1$ and using the conditions in equation 1, one move to the cell resulting in the score of the current cell. Only when the change is $n-1$ and $m-1$ (a diagonal shift due to a base pair or bifurcation), nt_m and nt_n are recorded as pairing [6].

1.3 Energy folding

RNA structures fold into the structure with the least free energy which is in accordance with the principals of thermodynamics. This is due to the different structural features affecting the stability of the overall RNA structure, where e.g. stacking of base pairs improves stability and loops decrease stability. The folding of linear RNA strands thus becomes a problem of finding the structure with the lowest free energy [7].

1.4 Extending the Nussinov algorithm with nearest neighbour energy

Because the RNA structure must have the lowest free energy, the Nussinov algorithm can be rewritten as a minimization problem. By using the structural information, e.g. the free energy contribution of neighbouring base pair stacking, see table 1, the Nussinov equation 1 and 2 can be adjusted to become a minimization problem:

$$M(i,j) = min\begin{cases} M(i,j-1) \\ M(i+1,j) \\ M(i+1,j-1) + s(i,j) \\ \min_{i<k<j-1}\{M(i,k) + M(k+1,j)\} \end{cases}, \quad \textit{(Eq. 3)}$$

$$s(i,j) = \begin{cases} table\ 1[nt_i nt_j, nt_{i+1} nt_{j-1}] & if\ nt_i\ and\ nt_j\ pair \\ 0 & if\ nt_i\ and\ nt_j\ don't\ pair \end{cases} \quad \textit{(Eq. 4)}$$

2 Materials and methods

2.1 Dataset

100 RNA sequences of 120 nucleotides have been downloaded from https://rth.dk/resources/bioinf2024/ using KUID PGT781. The downloaded dataset contains primary RNA sequence, secondary structure in dot-bracket notation as determined by the RNAfold algorithm, and minimum free energy for the predicted structure.

2.2 Implementation of the Nussinov algorithm and backtracking

The Nussinov algorithm and backtracking have been implemented using Python 3.7 and the following packages: NumPy ver. 1.21.6 and Pandas ver. 1.2.3. See section 6.1 for implementation of the Nussinov algorithm and section 6.2 for implementation of the backtracking.

2.3 Implementation of nearest neighbour energy in Nussinov algorithm

The nearest neighbour energy was implemented in the Nussinov algorithm by referring to table 2 for the pair and nearest neighbour pair energy contribution. Table 2 is used instead of Table 1 as requested in the exam. See implementation in section 6.3, where the yellow highlights are changes compared to section 6.1.

3 Results

All 100 RNA sequences have been predicted a secondary structure using my implementation of the unmodified Nussinov algorithm and the modified Nussinov algorithm. Pairwise base distance for the resulting structure have been made for the predictions of 1) unmodified Nussinov vs modified Nussinov, 2) unmodified Nussinov vs RNAfold, and 3) modified Nussinov vs RNAfold. Scatterplots with Pearsons correlation coefficient of the three pairwise base distances are reported in Figure 2A-C and the cumulative distribution of base pair distance for the pairwise comparisons are reported in Figure 2D.

3.1 Results of comparing pairwise base pair distances

There is a weak positive correlation between the unmodified Nussinov and RNAfold ($r = 0.24$), a weak positive correlation between the modified Nussinov and RNAfold ($r = 0.37$), and a moderate positive correlation between the unmodified and modified Nussinov ($r = 0.52$).

The cumulative distribution of base pair distances shows there are fewer base pair distances between the unmodified and modified Nussinov algorithm. Further, there are fewer base pair

distances between the unmodified Nussinov and RNAfold, than between the modified Nussinov and RNAfold, though the range of base pair distances is greater between the modified Nussinov and RNAfold, than the unmodified Nussinov and RNAfold.

4 Discussion

It is expected to find a correlation between the unmodified and modified Nussinov algorithm predictions, as the implementation is basically the same besides using experimentally determined stacking energies of nearest neighbour in the modified Nussinov algorithm ($r = 0.52$). Further, the pairwise comparisons showed that the modified Nussinov algorithm was closer to the predictions of RNAfold than the unmodified Nussinov algorithm ($r = 0.37$ vs $r = 0.24$), although the cumulative probability of distribution of base pair distance was less favourable than that of the unmodified Nussinov compared to RNAfold.

In the modified Nussinov, only energies for Watson-Crick and wobble pairs have experimentally determined values, while all other possible pairings have the value 1. Thus, a way to strengthen the minimization problem in this context would be to include the experimentally determined values for all nearest neighbour base pairings as described by Turner et al. and Vienna RNA server for RNAfold [8], [9].

Also, energy contributions of RNA features as loop, hairpins, helices etc. should be incorporated in the algorithm to account for these structural features effect on minimum free energy of the overall structure.

5 Figures and tables

5.1 RNA secondary structures

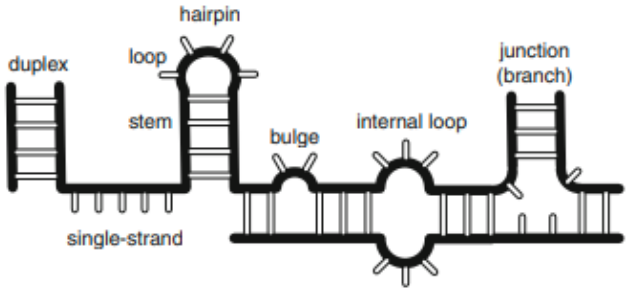

Figure 1 *Secondary structures of RNA. Figure from Zweib, C.* [4]

5.2 Nearest neighbour base pair energies

		$(i+1, j-1)$					
		AU	CG	GC	GU	UA	UG
(i,j)	AU	-1.1	-2.1	-2.2	-1.4	-0.9	-0.6
	CG	-2.1	-2.4	-3.3	-2.1	-2.1	-1.4
	GC	-2.2	-3.3	-3.4	-2.5	-2.4	-1.5
	GU	-1.4	-2.1	-2.5	1.3	-1.3	-0.5
	UA	-0.9	-2.1	-2.4	-1.3	-1.3	-1.0
	UG	-0.6	-1.4	-1.5	-0.5	-1.0	0.3

Table 1 Energy contribution between current base pair and nearest neighboring base pair. Table adapted from [10]

		$(i+1, j-1)$					
		AU	CG	GC	GU	UA	UG
(i,j)	AU	2.4	3.4	3.5	2.7	2.2	1.9
	CG	3.4	3.7	4.6	3.4	3.4	2.7
	GC	3.5	4.6	4.7	3.8	3.7	2.8
	GU	2.7	3.4	3.8	0.0	2.6	1.8
	UA	2.2	3.4	3.7	2.6	2.6	2.3
	UG	1.9	2.7	2.8	1.8	2.3	1.0

Table 2 Energy contribution between current base pair and nearest neighboring base pair. Table from [11]

5.3 Comparison of pairwise and cumulative distribution of base pair distances

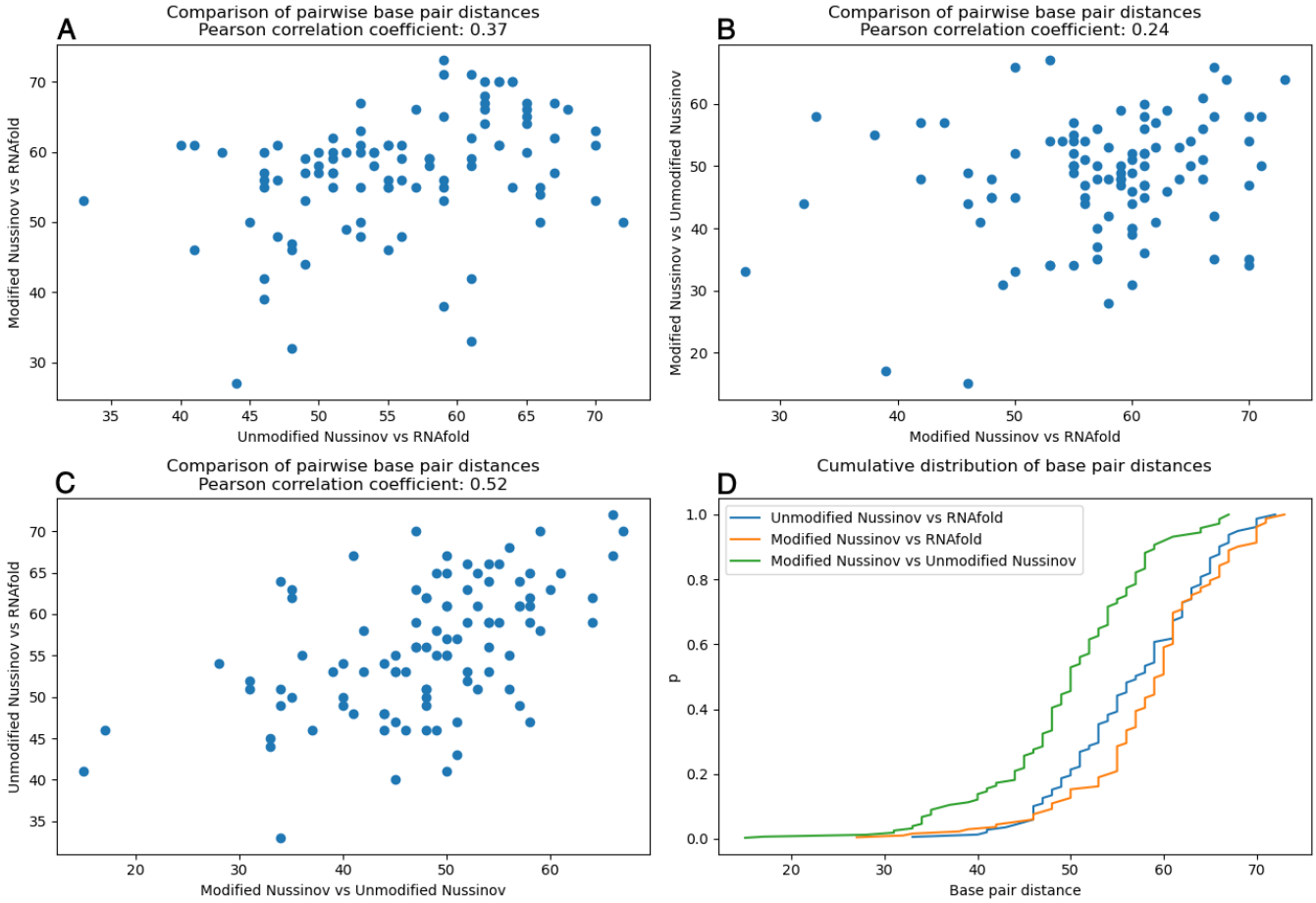

Figure 2 A-C: Scatterplot comparison of pairwise base pair distances for 100 RNA sequences of length 120 nucleotides between the unmodified Nussinov algorithm, the modified Nussinov algorithm, and the RNAfold algorithm. Pearson correlation coefficient is displayed above the scatterplot. **D:** The cumulative distribution of three pairwise base pair distances for 100 RNA sequences of length 120 nucleotides using the unmodified Nussinov algorithm, the modified Nussinov algorithm, and the RNAfold algorithm.

6 Code snippets

6.1 Implementation of the Nussinov algorithm

```
def nussinov(sequence, min_loop_size=0):
    """
    Implementation of the Nussinov algorithm to generate RNA secondary structure

    Args:
        sequence (Str): RNA sequence
        min_loop_size (Int): Minimum loop size, default: 0

    Returns:
        A matrix of Nussinov scores
    """

    # Get a zeroed matrix of the sequence in question
    seq_matrix = seq_to_matrix(sequence)

    # Loop through the cells of diagonals
    for i, j in loop_matrix_diag(seq_matrix, min_loop=min_loop_size):

        # Get the value of the cell to the left of the current cell (i, j-1)
        left_cell = seq_matrix[i, j - 1]

        # Get the value of the cell below the current cell (i+1, j)
        down_cell = seq_matrix[i + 1, j]

        # Check if the current cell is a basepair and
        if sequence[i] + sequence[j] in PAIRS.keys():

            # Get value of the cell diagonally down left and add 1 to the value
            diagonal_down_left = seq_matrix[i + 1, j - 1] + 1

        else:
            # Get value of the cell diagonally down left
            diagonal_down_left = seq_matrix[i + 1, j - 1]

        # Max value from bifurcation
        bifurcation = 0

        for k in range(i, j - 1):
            if (seq_matrix[i, k] + seq_matrix[k + 1, j]) > bifurcation:
                bifurcation = seq_matrix[i, k] + seq_matrix[k + 1, j]

        # Determine which value of the cells and bifurcation were biggest and write it
        # to the current cell
        biggest_value = max(left_cell, down_cell, diagonal_down_left, bifurcation)
        seq_matrix[i, j] = biggest_value

    # Return the matrix of Nussinov values
    return seq_matrix
```

6.2 Implementation of backtracking to get a predicted secondary structure

```
def backtrack(matrix, sequence):
    """
    Backtrack through a matrix using the same rules as in the Nussinov algorithm to
    predict a structure

    Args:
        matrix (Matrix): Matrix containing Nussinov scores
        sequence (Str): RNA sequence in interest

    Returns:
        A dot-bracket notation of RNA secondary structure
    """

    # Starting values of in the matrix depending on the sequence in question
    stack = [(0, len(sequence) - 1)]

    # Basepairs found
    basepairs = []

    # Continue to backtrack as long as the stack is not empty
    while len(stack) > 0:

        # Get first tuple values from stack
        i, j = stack.pop(0)

        # If i is equal to or bigger than j, move to next iteration
        if i >= j:
            continue

        # Check the value of the cell to the left of the current cell (i, j-1)
        if matrix[i, j - 1] == matrix[i, j]:
            stack.append((i, j - 1))  # Move one column to the left

        # Check the value of the cell below the current cell (i+1, j)
        elif matrix[i + 1, j] == matrix[i, j]:
            stack.append((i + 1, j))  # Move one row down

        # Check the value of the cell diagonally down left from the current cell
        # (i+1, j-1)
        elif sequence[i] + sequence[j] in PAIRS.keys():
            basepairs.append((i, j))  # Add found basepair to list
            stack.append((i + 1, j - 1))  # Move diagonally down left

        # If none of above is true, check if bifurcation occurred
        else:
            for k in range(i + 1, j - 1):
                stack.append((k + 1, j))
                stack.append((i, k))
                break

    # Return dot-bracket string of found basepairs (predicted secondary structure)
    dot_bracket_string = basepairs_to_string(basepairs, sequence)

    return dot_bracket_string
```

6.3 Implementation of nearest neighbour energy in the Nussinov algorithm

```
def nussinov(sequence, min_loop_size=0, use_scoring=False):
    """
    Implementation of the Nussinov algorithm to generate RNA secondary structure

    Args:
        sequence (Str): RNA sequence
        min_loop_size (Int): Minimum loop size, default: 0
        use_scoring (Bool): Use scoring matrix for stacked basepairs, default: False

    Returns:
        A matrix of Nussinov scores
    """

    # Get a zeroed matrix of the sequence in question
    seq_matrix = seq_to_matrix(sequence)

    # Loop through the cells of diagonals
    for i, j in loop_matrix_diag(seq_matrix, min_loop=min_loop_size):

        # Get the value of the cell to the left of the current cell (i, j-1)
        left_cell = seq_matrix[i, j - 1]

        # Get the value of the cell below the current cell (i+1, j)
        down_cell = seq_matrix[i + 1, j]

        # Check if the current cell is a basepair
        if sequence[i] + sequence[j] in PAIRS.keys():

            # Add score to value of cell diagonally down left
            if sequence[i + 1] + sequence[j - 1] in PAIRS.keys() and use_scoring:
                diagonal_down_left = seq_matrix[i + 1, j - 1] +
                                     bp_scoring.at[sequence[i] + sequence[j],
                                                   sequence[i + 1] + sequence[j - 1]]

            else:
                diagonal_down_left = seq_matrix[i + 1, j - 1] + 1

        else:
            # Get value of the cell diagonally down left
            diagonal_down_left = seq_matrix[i + 1, j - 1]

        # Max value from bifurcation
        bifurcation = 0

        for k in range(i, j - 1):
            if (seq_matrix[i, k] + seq_matrix[k + 1, j]) > bifurcation:
                bifurcation = seq_matrix[i, k] + seq_matrix[k + 1, j]

        # Determine which value of the cells and bifurcation were biggest and write it
        # to the current cell
        biggest_value = max(left_cell, down_cell, diagonal_down_left, bifurcation)
        seq_matrix[i, j] = biggest_value

    # Return the matrix of Nussinov values
    return seq_matrix
```

7 References

[1] J. Jumper *et al.*, “Highly accurate protein structure prediction with AlphaFold,” *Nature*, vol. 596, no. 7873, pp. 583–589, Aug. 2021, doi: 10.1038/s41586-021-03819-2.

[2] R. Wu *et al.*, “Title: High-resolution de novo structure prediction from primary sequence,” 2022, doi: 10.1101/2022.07.21.500999.

[3] J. Zhang, Y. Fei, L. Sun, and Q. C. Zhang, “Advances and opportunities in RNA structure experimental determination and computational modeling,” *Nature Methods*, vol. 19, no. 10. Nature Research, pp. 1193–1207, Oct. 01, 2022. doi: 10.1038/s41592-022-01623-y.

[4] C. Zwieb, “The Principles of RNA Structure Architecture,” in *RNA Sequence, Structure, and Function: Computational and Bioinformatic Methods*, vol. 1097, J. Gorodkin and W. L. Ruzzo, Eds., in Methods in Molecular Biology, vol. 1097. , Totowa, NJ: Humana Press, 2014, pp. 33–41. doi: 10.1007/978-1-62703-709-9.

[5] R. Nussinov, G. Pieczenik, J. R. Griggs, and D. J. Kleitman, “Algorithms for Loop Matchings,” *SIAM J Appl Math*, vol. 35, no. 1, pp. 68–82, Jul. 1978, doi: 10.1137/0135006.

[6] J. Gorodkin, I. L. Hofacker, and W. L. Ruzzo, “Concepts and Introduction to RNA Bioinformatics,” in *RNA Sequence, Structure, and Function: Computational and Bioinformatic Methods*, vol. 1097, J. Gorodkin and W. L. Ruzzo, Eds., in Methods in Molecular Biology, vol. 1097. , Totowa, NJ: Humana Press, 2014, pp. 7–10. doi: 10.1007/978-1-62703-709-9.

[7] I. L. Hofacker, “Energy-Directed RNA Structure Prediction,” in *RNA Sequence, Structure, and Function: Computational and Bioinformatic Methods*, vol. 1097, J. Gorodkin and W. L. Ruzzo, Eds., in Methods in Molecular Biology, vol. 1097. , Totowa, NJ: Humana Press, 2014, pp. 71–78. doi: 10.1007/978-1-62703-709-9.

[8] D. H. Mathews, J. Sabina, M. Zuker, and D. H. Turner, “Expanded sequence dependence of thermodynamic parameters improves prediction of RNA secondary structure,” *J Mol Biol*, vol. 288, no. 5, pp. 911–940, May 1999, doi: 10.1006/jmbi.1999.2700.

[9] I. L. Hofacker, “Vienna RNA secondary structure server,” *Nucleic Acids Res*, vol. 31, no. 13, pp. 3429–3431, Jul. 2003, doi: 10.1093/nar/gkg599.

[10] S. Seemann, “Energy Based RNA Secondary Structure Prediction,” Copenhagen, Jan. 2024.

[11] J. Gorodkin, S. Seemann, and T. Hamelryck, “Exam Structural Bioinformatics 2023-2024,” Copenhagen, Jan. 2024.