

From Sequence to 2D Structure

Practicals

Setup

- Clone the Git Repository
<https://github.com/ViennaRNA/ViennaRNA-Tutorial>
- You need an editor to view Jupyter-Notebooks or Jupyter-Lab and Conda installed
- Alternatively, upload the Notebooks and the folder 2_input to a Google-Colab environment (easy alternative for Windows)

Structure of the Practical

- First Part:
 - How can we predict secondary structures using the ViennaRNA Package
 - 1_ViennaRNA_Introduction.ipynb
 - Brief introductions; Small tasks; Discussion of results
- Second Part:
 - Reproduce a research paper using the techniques from session 1
 - 2_Practical_Example.ipynb and 2_input

Part 1: ViennaRNA Package Introduction

- Very expansive package covering RNA secondary structures
- Usage modes: **Command-Line Tools**, **Python-API**, C-API
- Object-oriented programming
 - Create *RNA.fold_compound(sequence)*
 - Use member function to change settings or start computations
 - fc stores sequence, settings and intermediate calculation results

Energy Evaluation

- Evaluate the energy of a given RNA sequence/structure pair
- Also provides detailed information on what each loop contributes to the total energy

```
import RNA

sequence = "CUACGGCGCGGCGCCUUGGCGA"
ss = ".....((((.....)))"

fc = RNA.fold_compound(sequence)

energy = fc.eval_structure_verbose(ss)
```

Minimum Free Energy

- To find the secondary structure with the lowest free energy
- Can use the command-line tool RNAfold

```
import RNA

sequence = "CUACGGCGCGGCGCCCUUGGCGA"

fc = RNA.fold_compound(sequence)
(ss, mfe) = fc.mfe()
```

Structure Ensemble

- Sequences can fold into a whole ensemble of structures, which can have very similar free energies
- Properties of the ensemble are often more informative than only looking at the MFE-structure
- After storing the partition function in the *fold_compound*, can access properties of the ensemble

```
import RNA

sequence = "CUACGGCGCGGCGCCCUUGGCGA"

fc = RNA.fold_compound(sequence)
bpp, efe = fc.pf()
```

Structure Ensemble Properties

- Free energy of the ensemble
- Base pairing probabilities of individual positions
 - Given in a upper triangular matrix
- Frequency of a given structure in the ensemble
- Centroid structure
 - Minimizes the weighted average distance to other structures of the ensemble

Suboptimal Structures

- Find structures which have an energy close to the MFE

```
import RNA
```

```
sequence = "CUACGGCGCGGCGCCCUUGGCGA"
```

```
fc = RNA.fold_compound(sequence)
```

```
subopt_structures = fc.subopt(delta=100, sorted=True)
```

- Or sample representative structures from the ensemble
 - Can be used to approximate uncommon properties where an exact algorithm is not implemented

Hard Constraints

- Find the lowest energy structure under constraints that certain positions are paired or unpaired.
- Useful when parts of the structure are already known
- Are represented using a pseudo dot-bracket notation

```
sequence = "CUACGGCGCGGCGCCCUUGGCGA"  
constraints_bp = ".....((xxx))"  
fc = RNA.fold_compound(sequence)  
fc.constraints_add(constraints_bp, RNA.CONSTRAINT_DB_DEFAULT | RNA.CONSTRAINT_DB_ENFORCE_BP)  
(ss, mfe) = fc.mfe()
```

Soft Constraints

- Allow for a more subtle guiding of the folding by adding a bonus energy to certain motifs like
 - Individual base pairs
 - Unpaired positions
 - Larger structures like hairpins
- Can be used to integrate chemical probing data or RNA-ligand interactions into a prediction

```
import RNA  
sequence = "CUACGGCGCGGCGCCCUUGGCGA"  
fc = RNA.fold_compound(sequence)  
fc.sc_add_bp(5, 15, -0.5)  
(ss, mfe) = fc.mfe()
```

Consensus Structure

- Used when multiple sequences fold into the same structure
- Calculate an alignment of the sequences
- Use covariation in the alignment to inform structure prediction

```
CCCCAAAGGGG  
GCCCAAUGGGC  
AUGCUAAGCAU
```

Consensus Structure

- Used when multiple sequences fold into the same structure
- Calculate an alignment of the sequences
- Use covariation in the alignment to inform structure prediction

```
C C C C A A A G G G G  
G C C C A A U G G G C  
A U G C U A A G C A U
```

Consensus Structure

- Used when multiple sequences fold into the same structure
- Calculate an alignment of the sequences
- Use covariation in the alignment to inform structure prediction

```
CCCCAAAGGGG  
GCCCAAUGGGC  
AUGCUAAGCAU
```

Consensus Structure

- Used when multiple sequences fold into the same structure
- Calculate an alignment of the sequences
- Use covariation in the alignment to inform structure prediction

```
CCCCAAAGGGG  
GCCCAAUGGGC  
AUGCUAAGCAU
```

Consensus Structure

- Used when multiple sequences fold into the same structure
- Calculate an alignment of the sequences
- Use covariation in the alignment to inform structure prediction

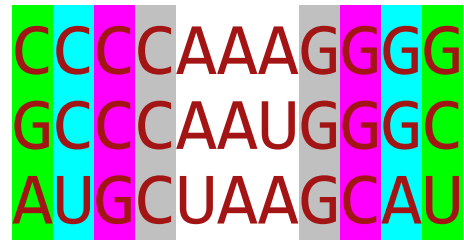


Diagram illustrating sequence alignment and covariation. The sequences are:

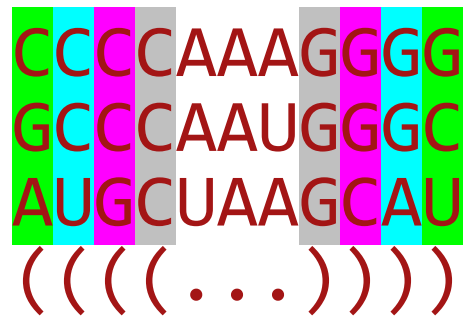
```
CCCCAAAGGGG  
GCCCAAUGGGC  
AUGC UAAGCAU
```

The alignment shows three sequences with highlighted regions indicating covariation (indels or mismatches):

- Position 1: C (green), G (cyan), A (magenta)
- Position 2: C (green), C (cyan), U (magenta)
- Position 3: C (green), C (cyan), G (magenta)
- Position 4: A (green), A (cyan), A (magenta)
- Position 5: A (green), A (cyan), A (magenta)
- Position 6: G (green), U (cyan), U (magenta)
- Position 7: G (green), G (cyan), G (magenta)
- Position 8: G (green), G (cyan), C (magenta)
- Position 9: G (green), C (cyan), A (magenta)
- Position 10: G (green), A (cyan), U (magenta)

Consensus Structure

- Used when multiple sequences fold into the same structure
- Calculate an alignment of the sequences
- Use covariation in the alignment to inform structure prediction
 - RNAalifold
 - *fold_compound* can also be created using an alignment



Pair Table Representation

- Dot-bracket notation not perfect for many applications

- Pair table: $((((\dots)))$

- [9, 9, 8, 7, 0, 0, 0, 3, 2, 1]

Length

0 if unpaired

Base 1 index of partner

Pair Table Representation

- Dot-bracket notation not perfect for many applications

- Pair table: `((((...)))`

- [9, 9, 8, 7, 0, 0, 0, 3, 2, 1]

Length

0 if unpaired

Base 1 index of partner

```
import RNA
```

```
ss = "..(((.....)))...."
```

```
ss_pt = RNA.pltable(ss)
```

```
for pos_i in range(1, ss_pt[0]+1):
```

```
    if structure_pt[pos_i] == 0:  
        #Unpaired  
        break
```

```
    if structure_pt[pos_i] < i:  
        continue
```

```
    if structure_pt[pos_i] > pos_i:
```

```
        # pos_i paired with structure_pt[pos_i]
```

Part 2: Introduction

- Predicting the structure of an alignment often more reliable due to covariation information (See RNAalifold)
- Structure of the alignment can not always be directly translated to individual sequences (focal sequence)

CACUAAAUGUG	CCGG--ACCGG	CCCAUCGACAUUUGUCGGAGGG
((((. .. .))))	((((.. .. .))))	((((.....))))

Strategy I – Hard constraints

- Extract all valid base pairs from the consensus structure
- Use them as hard constraints to refold the focal sequence

CACUAAAUGUG	CCGG--ACCGG	CCCAUCGACAUUUGUCGGAGGG
(((((...))))	(((((...))))	(((.)))
(((.)))	(((. - - . .)))	(((.)))

Strategy II – Soft constraints

- Use the structure ensemble of the alignment to calculate base pair probabilities
- For features μ (base pairs) with a non-zero probability, determine a bonus energy Γ_μ which can be added as a soft constraint
- $\Gamma_\mu = G_\epsilon[\mu] - G_\epsilon[\neg \mu]$

CACUAAAUGUG
 (((((...)))
 (((.....)))

CCGG--ACCGG
 (((((...)))
 (((. -- .)))

CCCAUAGGGAUUUCUUUGAAAU
 (((...)).....
 (((...)).....