Secondary Structure Identification in RNA

WWU Department of Computer Science

Open Acidification Project

Ivan Guillen

Walla Walla University
SEVENTH-DAY ADVENTIST HIGHER EDUCATION

ABSTRACT

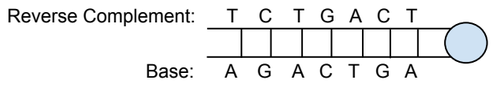
Dr. Kirt Onthank, an Associate Professor of Biology at WWU, studies ocean acidification and its effects on the physiology of marine invertebrates, especially cephalopods.

The goal of this project is to create a script using Python that, provided files containing potential edit sites and corresponding RNA sequences, will locate secondary structures at the site of each edit. This will facilitate Dr. Onthank's ability to identify actual RNA edits, which will in turn aid his research on the effects of ocean acidification on RNA editing in octopuses.

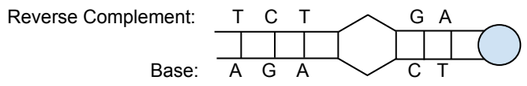
SECONDARY STRUCTURES

Nucleic acid secondary structures are base-pairing interactions that occur within the same sequence. While various types exist, my primary focus was on three types:

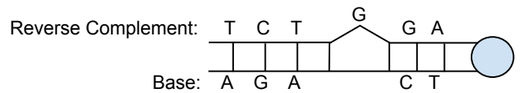
Hairpin Loops are a common type of secondary structure that are created when a sequence of RNA folds upon itself and forms base pairs with another section of the same sequence.



Internal Loops are similar, but feature a short sequence of unpaired bases within a larger sequence of paired bases.



Bulges are also similar, but feature regions on one side of the folded structure that have extra bases with no corresponding bases on the opposite side.



ALGORITHM

Our goal is to find the longest reverse complement at each edit site. Thus, given a sequence (**seq**) and edit site (**E**):

Step 1: Find Longest Reverse Complement to the Left

- Set **L** = **E**
- While the reverse complement of **seq[L:E]** exists in either **seq[:L]** or **seq[E:]**,
  - Decrement **L** by 1
- Set **longest** to **seq[L:E]**

Step 2: Find Longest Reverse Complement

- Set **R** = **E** + 1
- While **L** ≤ **E**,
  - While the reverse complement of **seq[L:R]** exists in either **seq[:L]** or **seq[R:]**,
    - Set **longest** to **seq[L:R]**
    - Increment **R** by 1
  - Increment **L** and **R** by 1
- Return **longest**

Iteration 1:

Iteration 2:

Iteration 3:

The base string **longest** (**"**GCTTAGACA") is returned, which is used to identify the longest reverse complement and its location in the sequence.

Note: Some steps, such as bounds checking, have been omitted for simplicity

INTERNAL LOOPS AND BULGES

While the underlying algorithm is similar for all three types, additional modifications were needed to account for unpaired/extra bases in internal loops and bulges. Regular expression searches were used instead of Python's built-in string.find() and periods (".") were used to simulate gaps between paired bases. A boolean flag was set if a jump over the unpaired/extra bases could be made and the jump would be made if necessary.

OUTPUT

id	position	length	base_string	base_string_loc	rev_comp	rev_comp_loc
hairpin	119	5	AAATT	[117, 121]	AATTT	[224, 228]
int_loop	8075	7	AGCGG.C	[8071, 8077]	G.CCGCT	[7748, 7754]
bulge	8075	7	AGGAGCG	[8068, 8074]	CG.CTCCT	[7702, 7709]

SUMMARY

By iteratively improving upon the algorithm, this project successfully allows users to efficiently identify three common types of secondary structures. This secondary structure identification will play a key role in allowing Dr. Onthank, and other researchers, to sort out actual edits from false positives.

Additional features and modifications will be implemented at the request of the customer. Some potential features include, but are not limited to, support for other types of structures such as pseudoknots, multi-thread capacity, and additional flags for identifying specific structures.

REFERENCES

- https://github.com/ivanguillen78/Octopus-RNA-Analysis
- https://open-acidification.github.io