Magdalena Rother

3D structure edition with ModeRNA

1. INTRODUCTION

1.1. THE PURPOSE. ModeRNA is a program for 3D RNA structure modeling. Besides the core functionality it facilitates RNA structure editing. This tutorial will help you to familiarize with it. The tutorial is divided into 6 independent exercises.

- Cleaning PDB files
- Working with sequence
- Secondary structure editing
- Checking geometry
- Working with fragments
- Modeling

They do not need to be carried out in the given order, so feel free to start with the category that interests you the most. Keep in mind that 'Cleaning PDB files' and 'Working with sequence' are the most basic ones.

1.2. MATERIALS for this tutorial contain:

- The ModeRNA software (http://genesilico.pl/moderna/download/)
- A directory with all input files (moderna_tutorial)
- The ModeRNA software documentation (http://genesilico.pl/moderna/commands/)

1.3. HOW TO GET STARTED? You can carry out all exercises in the python shell (after starting python and typing '`from moderna import *`' all commands mentioned in the TIPS sections will be available). You can also write a script. To make it work you should have ModeRNA installed to the site_packages folder or run all commands/scripts in the ModeRNA directory. Alternatively the moderna directory may be added to your PYTHONPATH.

2. CLEANING PDB FILES

2.1. Check what is inside your PDB file. For the files **tRNAphe_A.pdb**, **group2intron_A.pdb**, and **ribosome_0.pdb** check which have these features:

- Contains ions and a water molecules.

  How many ions residues are present in the structure?

- The phosphorus atom is missing in one of the residues.

  What is the base in the residue with the missing phosphorus (A, U, G, C)?

- The chain is discontinuous.

Which of the features mentioned above can be fixed with ModeRNA and which not?

TIPS: functions that may be useful to complete this task are `load_model`, `examine_structure`, and `clean_structure`. Notice that when you load a model you need to specify the chain name if it's different from 'A'.

2.2. The residues of the structure **aptamer_A.pdb** have a strange numbering. Change the residue numbers so that it starts with '1' and counts up continuously. Write the renumbered structure and examine the new file in a text editor or PyMOL to be sure that the numbering is correct.

TIPS: functions that may be useful to complete this task are `load_model`, `renumber_chain`, `write_model`.

3. WORKING WITH SEQUENCE

3.1. Check the sequences of the structures **tRNAcys_R.pdb** and **tRNAglu_C.pdb**. Which of them contains more uridine residues?

TIPS: functions that may be useful to complete this task are `load_model`, `get_sequence`.

3.2. Retrieve the sequence of **tRNApheYeast_A.pdb** and inspect this structure in PyMOL. Find out the meaning of:

- '_' character
- '.' character
- 'Y' character

TIPS: functions that may be useful to complete this task are `load_model`, `get_sequence`. You may also check the properties of residue objects: `residue.long_abbrev`, `residue.short_abbrev`, `residue.full_name`.

3.3. The anticodon loop of the glutamyl-tRNA (**tRNAglu_C.pdb**) is composed of C (residue 34), U (residue 535) and C (residue 536). Change it so that it can decode glutamine instead of glutamic acid (codon sequence: CAA, anticodon sequence UUG). Save the structure afterwards and check in PyMOL whether you can see any differences.

TIPS: functions that may be useful to complete this task are `load_model`, `exchange_single_base`.

3.4. Examine the structure **tRNAcys_R.pdb** for modified nucleotides.

How many modifications occur in this structure?

What are the different abbreviations and the full name for the modified residue 37?

TIPS: functions that may be useful to complete this task are `load_model`, `find_modifications`. You may also check the residue properties: `residue.long_abbrev`, `residue.short_abbrev`, `residue.full_name`.

3.5. Remove all modifications from the structure **tRNAcys_R.pdb**. Make the first residue of this structure 2-methylthio-N6-hydroxynorvalylcarbamoyladenosine (ms2hn6A). Write the structure to a PDB file and examine in PyMOL whether you were successful.

TIPS: functions that may be useful to complete this task are `load_model`, `remove_all_modifications`, `add_modification`, `write_model`.

4. SECONDARY STRUCTURE EDITING

4.1. Retrieve the secondary structure of the **aptamer_A.pdb**. How many canonical Watson-Crick pairs are formed in this structure?

TIPS: functions that may be useful to complete this task are `load_model`, `get_secstruc`.

4.2. Try to elongate **helix_A.pdb** with a helix having the sequence 'AGCU_AGCU' starting from the residues 7 and 26. What are the identifiers of the inserted residues?

TIPS: functions that may be useful to complete this task are `load_model`, `extend_helix`.

4.3. File **single_strand_A.pdb** contains the single strand of a helix. Add the second strand to the helix. Have you got a perfect helix?

TIPS: functions that may be useful to complete this task are `load_model`, `add_pair_to_base`, `fix_backbone`.

5. CHECKING GEOMETRY

5.1. Examine the geometry of the structure **model_tRNAglu_D.pdb**.

Does the structure have any geometrical irregularities? Which kind of problems can be found?

Do any residues clash inside the structure? What are the identifiers of the clashing residues?

TIPS: functions that may be useful to complete this task are `load_model`, `analyze_geometry`, `find_clashes`.

5.2. Try to fix the backbone discontinuity in the **model_tRNAglu_D.pd**b (between residues 14-15 and 19-20) and in the **ribosome_0.pdb** (between residues 125-129 and 714-716). Is it always possible to fix the backbone? Why?

TIPS: functions that may be useful to complete this task are `load_model`, `fix_backbone`.

6. WORKING WITH FRAGMENTS

6.1. Join the structure **helix_A.pdb** and **tetraloop_A.pdb** into one. Use residue 16 and 17 as anchors in the helix. What is the length (number of residues) of each structural fragment taken alone and of the resulting structure that you have obtained?

TIPS: functions that may be useful to complete this task are `load_model`, `create_fragment`, `insert_fragment`.

6.2. Make a hairpin structure out of the **helix_A.pdb**. Find one fragment candidate using residues 16 and 17 as anchors and 'GGUCCAAACCGGACC' as sequence. Insert the candidate to the structure and write the model to a file.

What is now the secondary structure of the model?

Now again find one fragment candidate with the same parameters as above but add the secondary structure constraint '(((((.....)))))'. Insert the candidate to the structure and write the model to a file. Has ModeRNA managed to enforce the secondary structure correctly?

Compare both models. Which fragment is better in your opinion?

TIPS: functions that may be useful to complete this task are `load_model`, `find_fragment`, `insert_fragment`, `write_model`, `get_secstruc`.

6.3. The anticodon arm of the broken_tRNAglu_C.pdb is missing. Find 10 fragment candidate that will fit to this structure. Use residues 527 and 543 as anchors. The sequence of the missing arm is 'GGCCCUCUCAAGGCCGA'. Write all candidates to files and examine them in PyMOL.

Compare candidate0.pdb and candidate9.pdb. Which would you choose to insert into the broken tRNA?

TIPS: functions that may be useful to complete this task are `load_model`, `find_fragment`, `write_fragment_candidates`. To load all 10 structure to PyMOL you may type 'pymol *' in the directory with structures.

7. MODELING

7.1. Build a model of the Tyr tRNA based on the template 1QF6_B_tRNA.pdb and the alignment aln_1QF6_E_coli_Tyr2_GUA.fasta.

Is the sequence of your model identical with the target?

Does the model have any geometrical irregularities?

Does it have any internal clashes?

TIPS: functions that may be useful to complete this task are `load_template`, `load_alignment`, `create_model`, `match_alignment_with_model`, `analyze_geometry`, `find_clashes`.