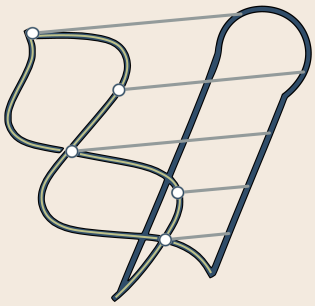


PlanarFold

a CGMD model of RNA in 2D space

PlanarFold

Version 1.1

User manual

Developed by

Lan Xiang

Mar. 2025

Contents

1. Framework of PlanarFold

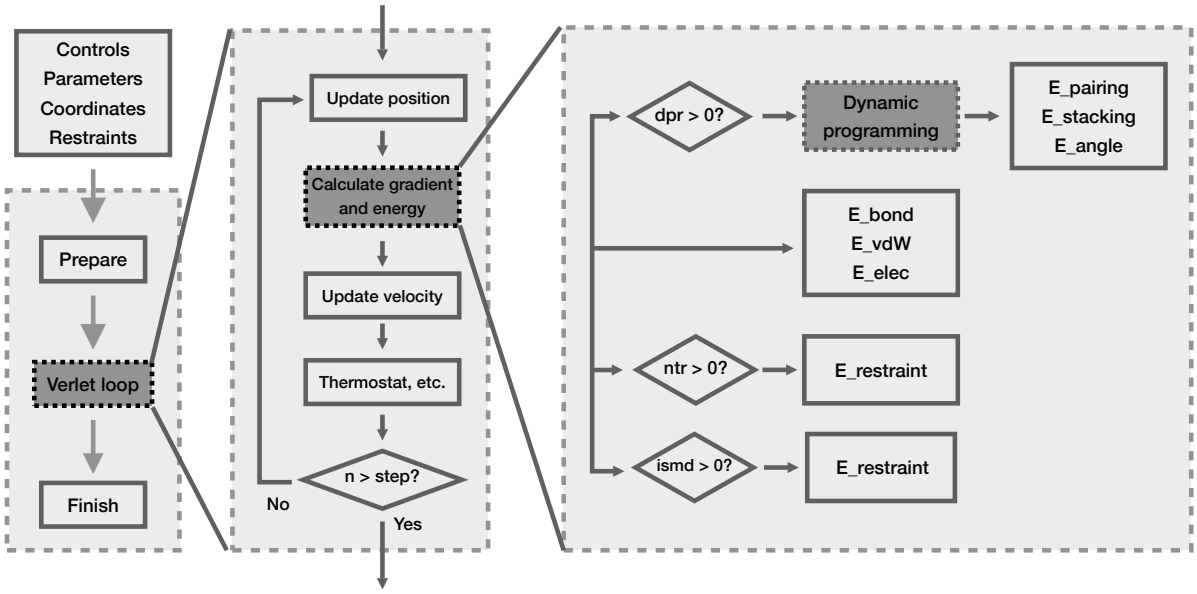


Figure 1. Workflow of PlanarFold

PlanarFold is a coarse-grained molecular dynamics (CGMD) model designed to study RNA dynamics on a two-dimensional (2D) plane. Inspired by the planar visualization of RNA secondary structures, PlanarFold represents each nucleotide as a single bead, enabling efficient exploration of large conformational changes while focusing on secondary structure formation and dynamics. The model employs a physics-based potential function with six energy terms: bond, stem angle, base pairing, base stacking, van der Waals interaction, and electrostatics. A unique feature of PlanarFold is its integration of a dynamic programming (DP) algorithm, which dynamically identifies stem regions in real time and selectively applies energy terms. PlanarFold is optimized for studying secondary-structure-centered dynamics, excluding pseudoknots and other tertiary interactions. Its standalone design, compatible with Amber format, makes it a powerful tool for simulating RNA behavior in a straightforward yet effective manner.

The pipeline of PlanarFold follows the architecture of *Amber*'s *mdgx* program, which is written in C. PlanarFold is written in Python, C, and C++. The Python code handles input file parsing (*PlanarFold.py*). The C code performs the core calculations (*planarfold_core.a*), while the C++ code manages data transmission between the Python and C modules (*dsflyext.cpp*)..

All the analysis tools of PlanarFold are written in Python. Users proficient in Python are encouraged to modify these scripts to generate a wider variety of visualizations and analysis outputs.

2. PlanarFold's I/O and analysis

2.1. Usage

The *PlanarFold* is a Python executable script.

```
$ PlanarFold    -i controls.in  -f fparm.in
                -p ssRNA.prmtop  -c ssRNA.inpcrd
                -o run.out  -x run.nc  -r run.rst
                -ntr restraint_ntr.in  -dpr  restraint_dpr.in
```

or

```
$ PlanarFold
```

See more description about the input and output files:

```
$ PlanarFold -h
```

The first command line explicitly describes all the default input and output files. The second command line executes simulation using all default files implicitly.

2.2. Input files

2.2.1. Control parameter file

The control parameter file (*control.in*) contains several frequency related parameters, parameters of restraints, the length and time step of simulations, etc. Several basic control parameters follow the name of *AMBER*'s corresponding parameters.

- Parameters are explained as follows:

—*dt* The time step, unit in picosecond (ps). Default = 0.004. Due to single-bead coarse-graining, the time step can be chosen larger than 1 fs and 2 fs. The longer time step is not recommended, as the length of the time step affects the calibrated temperature. A time step of 4 fs is tested through all case studies in the PlanarFold paper, and has been carefully calibrated with experimental temperature and its thermostats. Changing this parameter, then all the quantitative results from the original paper might may no longer apply. Do not change this parameter unless with expertise.

—*nstlim* Number of MD steps to be performed. Default = 1000.

—*ntpr* Every *ntpr* steps, energy information will be printed in a human-readable energy output file (*run.out*). Default = 1000.

—*ntwx* Every *ntwx* steps, coordinates will be printed in NetCDF-formatted coordinate output file (*run.nc*). Default = 1000.

—*nscm* Every *nscm* steps, translational and rotational momentum are removed. Default = 1000. The removal of rotational momentum is extremely important for simulation of small RNAs. Compared to all-atom simulation, the synchronization of all atoms' motion is nearly impossible, so the rotation of the entire system will not happen. However, for our small system, such as a 12-nt hairpin, it will rotate very fast, which contributes to all the kinetic energy, and the RNA will be stuck in one particular conformation.

- Parameters for constant harmonic restraints:

—*ntr* Flag for applying constant harmonic restraints, default = 0. If ntr > 0, the restrained pair-wise residues will be determined from "restraint file" (*restraint_ntr.in*). If ntr = 1, constant harmonic restraints will be applied throughout the simulation. If ntr = 2, constant harmonic restraints will be applied only after the target base pairs approach and form spontaneously. If ntr = 2, extra lines will be written into the energy output file describing the successfully "targeted" base pairs.

—*rcut* The cutoff for applying constant harmonic restraints, unit in Å. Default = 11.5. Only when target residue pairs are in a distance larger than *rcut*, harmonic restraints are applied. In this way, when systems are cooled to 0 Kelvin, most harmonic restraints are released to yield target secondary structure without restraining energy. Setting *rcut* to a large value can be useful for plotting of prescribed secondary structure. However, a large *rcut* can cause futile restraints because target base pairs will not be stretched too close to contact (larger than *vcut*).

—*restraint_wt* The strength of constant harmonic restraints, unit in kcal·mol^{-1}·Å^{-2}. Default = 0.1.

—*restraint_eq* The equilibrium distance of constant harmonic restraints, unit in Å. Default = 10.0.

- Parameters for temperature control:

—*ntt* The option for temperature controlling. Default = 0. If *ntt* = 1, Berendsen thermostat will be used. More sophisticated thermostats will be implemented for specific usage.

—*tautp* Thermostat coupling constant, unit in ps. Default = 0.005. Tight coupling is necessary for PlanarFold as the temperature fluctuation is much stronger than all-atom system.

—*iCollide* The option for mimicking random water collision. Default = 0. If *iCollide* = 1, water with velocity following Maxwell-Boltzmann distribution is assumed to collide with each residue every *nscm* steps. It is important to introduce random collision for

simulating RNAs with long handles (SMD). This functionality can be replaced with thermostats considering random fluctuations.

—*ntemp* The number of temperature controlling stages for simulated annealing. Default = 1. If *ntemp* > 1, simulated tempering or annealing is performed.

—*tempi* Initial temperature(s), unit in Kelvin. If more than one *tempi* is described or *tempi* ≠ *temp0*, simulated tempering or annealing is performed.

—*temp0* Reference or final temperature(s), unit in Kelvin. If more than one *temp0* is described or *tempi* ≠ *temp0*, simulated tempering or annealing is performed. The temperature is linearly interpolated from *tempi* to *temp0* within *nstlim* steps.

—*duration* The number of MD steps at each temperature controlling stage. If *tempi* = *temp0, duration* can be set to any non-zero number, indicating a constant temperature simulation. If *tempi* ≠ *temp0, duration* must be set to *nstlim*, such that the temperature can be regulated from *tempi* to *temp0*. When multiple *duration* is assumed, and the sum of all *duration* is smaller than *nstlim*, simulated annealing cycles are performed.

- Parameters for neighbor list:

—*verlet_cut* The cutoff for creating Verlet neighbor lists, unit in Å. Default = 30.0. For residue i, all other residues within *verlet_cut* are stored into its neighbor list. Then, the pair-wise calculation of vdW, electrostatic, and base pairing interactions between residue i and other residues can be searched from its neighbor list, avoiding the search for all the residues. Instead, *Amber* uses box list in combination with periodic boundary condition, which is inappropriate for our coarse-grained small system of non-periodic boundary. V*erlet_cut* must be larger than *vcut*, *bcut*, and *ecut*. When a small *verlet_cut* is chosen, a small *nVerlet* should be employed as well.

—*nVerlet* Every *nVerlet* step, the Verlet neighbor lists are updated. Default = 1000. The choice of a proper *nVerlet* is determined by the velocity of the residue. At least in *nVerlet* step, no residue beyond the Verlet neighbor list of residue i should enter into the range such that it can interact with residue i.

- Parameters for steered MD:

—*ismd* The option for performing steered MD. Default = 0. If *ismd* > 0, steering force will be applied to the 5’ and 3’ termini of the RNA molecule. Currently, only terminal residues can be steered. The functionality of steering of any pair of residues can be readily achievable in the future. If *ismd* = 1, constant force (c.f.) mode of steering is employed, and the steering force will be kept at *stretch_force* (without any fluctuation). In the future, more realistic c.f. mode, where fluctuation should be

allowed as a result of the feedback frequency limited by wet experiment (optical tweezer or AFM), should be implemented. If *ismd* = 2, constant velocity (c.v.) mode of steering is employed. To achieve this, a harmonic restraint is applied to the RNA of interest. By gradually changing the equilibrium value of this harmonic spring, a virtual manipulation of optical trap is mimicked. The c.v. mode of PlanarFold is similar to *Amber*'s SMD mode, which is non-directional steering. To simulate the direction steering, two dumb beads can be added, following *NAMD*'s c.v. mode. The stretching or relaxing speed is determined as (*dist_final* - *dist_init*) / (*nstlim*dt*). If *dist_final* = *dist_init*, passive mode is employed.

—*dumpfreq* Every *dumpfreq* steps, the SMD-related arguments (equilibrium value of steering spring, terminal distance, force, and work) will be written. The equilibrium values of the steering spring interpolate between *dist_init* and *dist_final* every *dumpfreq* steps.

—*stretch_force* The stretching force in c.f. mode, unit in kcal·mol^{-1}·Å^{-2}. Default = 1.0. 1 kcal·mol^{-1}·Å^{-2} converts to 69.5 pN·Å^{-1}. Of note, the *stretch_force* should be set as the half of the experiment stiffness, as in

—*stretch_spring* The spring constant (or stiffness) in c.v. mode, unit in kcal·mol^{-1}·Å^{-2}. Default = 0.01. 1 kcal·mol^{-1}·Å^{-2} converts to 69.5 pN·Å^{-1}. Of note, the *stretch_spring* should be set as the half of the experiment stiffness, as PlanarFold follows *Amber*'s format, that the Force = 2·*stretch_spring*·(X-X(t)), where X and X(t) is the extension of the RNA and the equilibrium value of the virtual spring, respectively. Careful calculation of the *stretch_spring* is very important to match with experimental trap stiffness.

—*dist_init* The initial equilibrium value of the virtual spring in c.v. mode, unit in Å. Default = 10.0.

—*dist_final* The final equilibrium value of the virtual spring in c.v. mode, unit in Å. Default = 100.0. When *dist_init* < *dist_final*, the RNA is stretched to unfold. When *dist_init* > *dist_final*, the RNA is relaxed to refold. When *dist_init* = *dist_final*, passive mode is simulated.

2.2.2. Force parameter file

The force parameter file (*fparm.in*) contains parameters related to all the potential function and dynamic programming. The most unique design of PlanarFold force field is the three terms in the potential (stem angle, base pairing, and base stacking) are not applied to all residues but are selectively applied to residues identified to form stems. Specifically, these stems are not pre-defined but are determined in real time at each step by the dynamic programming (DP) algorithm.

The potential function of PlanarFold:

$$E_{potential} = k_{scaling} \cdot (E_{bond} + E_{angle} + E_{stacking} + E_{pairing} + E_{vdW} + E_{elec})$$

Where

$$E_{bond} = k_{bond} \cdot (r - r_0)^2$$

$$E_{angle} = k_{angle} \cdot (\theta - \theta_0)^2 + p_{angle}$$

$$E_{stacking} = k_{stk} \cdot k_{stacking}[i] \cdot [(\psi - \psi_0)^2 + p_{stacking}]$$

$$E_{paring} = n_{HB}[i] \cdot [k_{paring} \cdot (d - d_0)^2 + p_{paring}]$$

$$E_{vdW} = \epsilon \cdot \{(\frac{\sigma}{r})^{12} - (\frac{\sigma}{r})^6\}$$

$$E_{elec} = \frac{Q^2}{r} \cdot e^{-\frac{r}{\lambda}}$$

- Parameters for bond:

  —*bond_eq* The equilibrium value (r_0) of bond potential, unit in Å. Default = 5.5.

  —*bond_k* The spring constant (k_{bond}) of bond potential, unit in kcal·mol^{-1}·Å^{-2}. Default = 20.0.

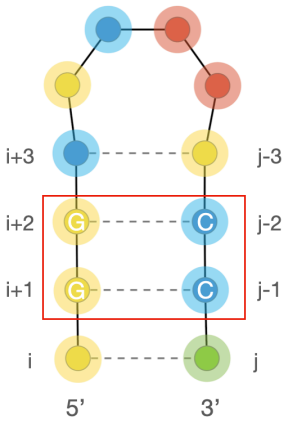
- Parameters for stem angle (rigidity):

  —*angle_eq* The equilibrium value (θ_0) of stem angle potential, unit in radian. Default = 3.1415926535898 (180° in degree).

  —*angle_k* The spring constant (k_{angle}) of stem angle potential, unit in kcal·mol^{-1}·radian^{-2}. Default = 4.0.

  —*angle_p* The penalty/bonus (p_{angle}) of stem angle potential, unit in kcal·mol^{-1}. Default = -3.7.

- Parameters for base-stacking (harmonic):



5' \ 3'	AU	UA	GC	CG	GU	UG
AU	-0.9	-1.1	-2.1	-2.2	-0.6	-1.4
UA	-1.3	-0.9	-2.1	-2.4	-1.0	-1.3
GC	-2.4	-2.2	**-3.3**	-3.4	-1.5	-2.5
CG	-2.1	-2.1	-2.4	-3.3	-1.4	-2.1
GU	-1.3	-1.4	-2.1	-2.5	-0.5	1.3
UG	-1.0	-0.6	-1.4	-1.5	0.3	-0.5

Figure 2. 36 bp-steps and corresponding stk_k

—*stk_ks* The uniform scaling factor (k_{stk}) for base-stacking potential. Default = 1.5.

—*stk_eq* The equilibrium value (ψ_0) of base-stacking potential, unit in radian. Default = 1.57079633 (90° in degree).

—*stk_k* The spring constant ($k_{stacking}[i]$) of base-stacking potential, unit in kcal·mol^{-1}·radian^{-2}. There are 36 stk_*k* parameter for different bp-steps (Fig.2), which is adopted 36 Turner's(2004) parameters for Watson-Crick Helix.

—*stk_p* The penalty/bonus ($p_{stacking}$) of base-stacking potential, unit in radian^2. Default = -3.0.

- Parameters for base pairing (harmonic):

—*bphm_eq* The equilibrium value (d_0) of base pairing potential, unit in Å. Default = 10.0.

—*bphm_k* The spring constant ($k_{pairing}$) of base pairing potential, unit in kcal·mol^{-1}·Å^{-2}. Default = 0.4.

—*bphm_p* The penalty/bonus ($p_{pairing}$) of base pairing potential, unit in kcal·mol^{-1}. Default = 9.6.

—*nHB* The number of hydrogen bonds (nHB[i]). The GC/CG, AU/UA, and GU/UG base pair are assumed to form 3, 2, and 2 hydrogen bonds, respectively. Other non-canonical base pairs are excluded from base pairing potential, even though they are considered in dynamic programming.

- Parameters for van der Waals potential:

—*sigma* The distance (σ) where vdW potential is zero, unit in Å. Default = 9.8.

—*epsilon* The well depth (ε) of vdW potential, unit in kcal·mol^{-1}. Default = 1.1.

—*vcut* The cutoff for calculating vdW potential, unit in Å. Default = 13.5.

- Parameters for electrostatic potential:

—*charge* The charge (Q) of each residue, unit in kcal$^{0.5}$·mol$^{-0.5}$·Å$^{0.5}$. Default = -2.93. Currently, only uniform charge is supported.

—*len_Debye* The Debye length (λ) of electrostatic potential, unit in Å. Default = 10.0.

—*ecut* The cutoff for calculating electrostatic potential, unit in Å. Default = 20.0.

- Parameters for dynamic programming:

$$S(i, j) = \max (S(i-1, j+1) + cmap(i-1, j+1) \times bpFLAG(i-1, j+1) ,$$
$$S(i-1, j) + cmap(i-1, j) \times bpFLAG(i-1, j) + bulge\_p ,$$
$$S(i, j+1) + cmap(i, j+1) \times bpFLAG(i, j+1) + bulge\_p),$$
$$\text{for } |j-i| > min\_Loop.$$

—*bcut* The cutoff for calculating inter-residue contacts, unit in Å. Default = 14.0.

—*bulge_p* The penalty for bulge loop and asymmetric internal loop. Default = 0.02.

—*bpFLAG* The factor for modulating different contact strength. bpFLAG is 6, 4, 4, 2, and 1 for GC/CG, AU/UA, GU/UG, GA/AG, and other base pairs, respectively, which is roughly proportional to the hydrogen bonds involving in them.

—*minbpFLAG* The minimum factor of contact that are allowed to form base pairing. Default = 2. For *bpFLAG* ≥ *minbpFLAG*, such contact is extracted during back-tracking of dynamic programming. By setting *minbpFLAG* = 2, only GC, AU, and GU contact will be extracted.

—*minStembp* The minimum number of continuous base pairing allowed to form base pairing potential. Default = 2. After backtracking in dynamic programming, canonical contacts (GC, AU, and GU) are screened into separate stems. If *minStembp* = 1, lone-pairs are allowed. Lone-pairs only have base pairing energy, but no base-stacking and stem angle energy. If *minStembp* = 2, lone-pairs are excluded.

—*min_Loop* The smallest apical loop allowed. Default = 4 (tetraloop). There are also tri-loops identified, but most of them turn to be pentaloop, as their closing base pair are either unstable or non-canonical. For special case where triloop formation is confident, setting *min_Loop* = 3 might be helpful.

- Parameters for holistic scaling of all potential:

—*f_scale* The scaling factor ($k_{scaling}$) that uniformly re-scales all the potential proportionally. Default = 0.33. A larger *f_scale* generally yields a more stagnant trajectory (with rarer conformation change).

2.2.3. Topology file

The topology parameter file (*ssRNA.prmtop*) follows *Amber*'s ".prmtop" file format. Compared with *Amber*'s complete description of all SECTIONS in prmtop file, PlanarFold's prmtop file only keeps the SECTIONS that are necessary for visualization (by *VMD*) and analyzing tools (*pytraj*). A full description of prmtop file can be found at *https://ambermd.org/FileFormats.php#topology*.

The SECTIONs being kept are as follows:

—POINTERS This section contains the information about how many parameters are present in each section. Compared to all-atom prmtop, most parameters in PlanarFold's prmtop are obsolete.

—ATOM_NAME This section contains the bead name (nucleotide name in PlanarFold) for each bead in the prmtop. Options are A, U, C, and G.

—CHARGE This section contains the charge for each nucleotide in the prmtop. The charges described in prmtop file are currently obsolete. The parameter *"charge"* in the force parameter file (*fparm.in*) describes the uniform charge of nucleotide. In the future, this section can be activated for non-uniform description of charges.

—ATOMIC_NUMBER This section contains the nucleotide index for each nucleotide in the prmtop. A, U, G, and C are indexed as 1, 2, 3, and 4, respectively.

—MASS This section contains the mass for each nucleotide in the prmtop. Currently, the mass of all nucleotides are uniformly assumed as 321.45 $g \cdot mol^{-1}$ for simplicity. In the future, this section can be activated for non-uniform description of masses.

—ATOM_TYPE_INDEX This section contains the nucleotide index for each nucleotide in the prmtop, identical to ATOMIC_NUMBER section. A, U, G, and C are indexed as 1, 2, 3, and 4, respectively.

—NONBONDED_PARM_INDEX This section contains the pointers for each pair of Lennard-Jones bead types for each nucleotide in the prmtop. Index = NONBONDED_PARM_INDEX [NTYPES × (ATOM_TYPE_INDEX(i) – 1) + ATOM_TYPE_INDEX(j)]. In this way, a pair of same residues (e.g., AU and UA) share the same index for LJ potential. This section would be helpful if different residue pairs share different vdW parameter.

—RESIDUE_LABEL This section contains the residue name (nucleotide name in PlanarFold) for each bead in the prmtop, identical to ATOM_NAME section. Options are A, U, C, and G.

—RESIDUE_POINTER This section lists the first atom in each residue. Since PlanarFold uses single-bead model, they are identical to RNA primary sequence position number (from 5' to 3').

—BOND_FORCE_CONSTANT This section contains the bond force constant (k_{bond}) for each bond in the prmtop. The bond force constant described in prmtop file is currently obsolete. The parameter *"bond_k"* in the force parameter file (*fparm.in*) describes the uniform bond force constant of all bonds. In the future, this section can be activated for non-uniform description of bond force constant.

—BOND_EQUIL_VALUE This section contains the bond equilibrium value (r_0) for each bond in the prmtop. The bond equilibrium value described in prmtop file is currently obsolete. The parameter *"bond_eq"* in the force parameter file (*fparm.in*) describes the uniform

	bond equilibrium value of all bonds. In the future, this section can be activated for non-uniform description of bond equilibrium value.
—ANGLE_FORCE_CONSTANT	This section contains the stem angle force constant (k_{angle}) for each stem angle in the prmtop. The stem angle force constant described in prmtop file is currently obsolete. The parameter *"angle_k"* in the force parameter file (*fparm.in*) describes the uniform stem angle force constant of all stem angles. In the future, this section can be activated for non-uniform description of stem angle force constant.
—ANGLE_EQUIL_VALUE	This section contains the stem angle equilibrium value (θ_0) for each stem angle in the prmtop. The stem angle equilibrium value described in prmtop file is currently obsolete. The parameter *"angle_eq"* in the force parameter file (*fparm.in*) describes the uniform stem angle equilibrium value of all stem angles. In the future, this section can be activated for non-uniform description of stem angle equilibrium value.
—BOND_WITHOUT_HYDROGEN	This section contains a list of every bond in the system in which neither atom is hydrogen. Since PlanarFold is a coarse-grained model with only A, U, C, and G, all the bonds are bonds without hydrogen. This section is important to describe the connectivity of the RNA chain.

Beyond the above-mentioned sections, all other sections are kept vacant, with only header for file format consistency.

2.2.4. Coordinate file

The coordinate file (*ssRNA.inpcrd*) follows *Amber*'s ".inpcrd" or ".rst" file format. Both coordinate file formats are identical to *Amber*'s corresponding file format. A full description of inpcrd/rst file can be found at *https://ambermd.org/FileFormats.php#topology*. For inpcrd file, bead coordinates and velocities are written in human-readable txt file format. Different from amber, the velocities in inpcrd file will be read by PlanarFold, instead of reassigning initial velocities using build-in procedure. For rst file, the coordinates and velocities are written in NetCDF format, with data stored in double precision (64-bit) for the precise restart of the simulation. Of note, both inpcrd and rst file store three-dimensional coordinates and velocities, with the coordinates and velocities set to zero, to be compatible with VMD and other analysis tools.

2.2.5. Restraint and confinement file

The restraint file (*restraint_ntr.in*) and confinement file (*restraint_dpr.in*) follow *.fasta* file format, with an extra line describing secondary structure in dot-bracket format. Currently, only nested base pairs are allowed. Pseudoknots and lone-pairs are excluded. The parameter that controls the on/off and modes of the restraint and confinement are "*ntr*" and "*dpr*" in *control.in*, respectively.

2.3. Output files

2.3.1. Energy output file

There are 6 sections in PlanarFold's energy output file (*run.out*).

—Section 1: the execution time-log

—Section 2: the control parameters

—Section 3: the force field parameters

—Section 4: the restraints and confinements

—Section 5: the energies, base pairs, etc. Writing frequency is controlled by parameter "*ntpr*".

```
*******************************************************************************
  PlanarFold's ENERGIES
*******************************************************************************
//-----------------------------------------------------------------------------
Frame:          200000   Temperature:        534.2130   Restraint:        0.0000
Bond:          67.1919   Angle:             -15.7995   Stacking:      -192.0957
Vdw:          -58.8444   Elec:               36.2225   BasePair:       208.2984
Ektot:        170.9171   Eptot:              44.9732   Etot:           215.8902
!!BasePairs: 24
  0-54    1-53    2-52    3-51   11-43   12-42   13-41   14-40   23-39   24-38
 69-87   70-86   71-85   94-136  95-135 103-130 104-129 105-128 108-127 109-126
111-122 112-121 113-120 114-119
```

Figure 3. An example of energy outputs

The 6 energy terms, bond, stem angle, base-stacking, van der Waals, electrostatics, and base pairing are recorded as *Bond*, *Angle*, *Stacking*, *Vdw*, *Elec*, and *BasePair*, respectively. The kinetic energy, potential energy, and total internal energy are indicated as *Ektot*, *Eptot*, and *Etot* (= *Ektot* + *Eptot*). The harmonic restraint energy (including SMD virtual spring) is recorded as *Restraint*. There is no energy term for confinement, as it does not add any harmonic potential to the system. The unit for energies is in kcal·mol^{-1}. The temperature is in unit Kelvin.

Besides, the base pair information is also recorded, which is determined by dynamic programming and several post-processing procedures. The base pairs are arranged by the index of their 5' residue.

—Section 6: the calculation timing

2.3.2. Trajectory file

The trajectory file (*run.nc*) records the coordinates every "*ntwx*" (a control parameter in *control.in*) frames, in NetCDF format. The trajectory is written in binary format, NetCDF package is required to extract the data. The trajectory file of PlanarFold completely follows the trajectory file format of *Amber*, such that PlanarFold's trajectory file can be visualized by VMD and analyzed by several cut-and-dried tools from *Amber* (e.g., *pytraj*).

2.3.3. Restart file

The restart file (*run.rst*) recorded the coordinates and velocities of the last frame, in NetCDF format with double precision for resuming the simulation. Like the trajectory file, the restart file requires NetCDF package to extract the binary data. The restart file of PlanarFold completely follows the restart file format of *Amber* as well.

2.3.4. Force-Extension file

The force-extension file (*dist.RST*) will be generated only if *ismd* > 0. A detailed description of this file format can be found in *Steered MD* section.

2.4. Analysis

2.4.1. generate_PlanarFold_Inputs

The tool "*generate_PlanarFold_Inputs*" can generate input files required for PlanarFold, including topology file (*ssRNA.prmtop*), coordinate file (*ssRNA.inpcrd*), restraint (*restraint_ntr.in*) and confinement (*restraint_dpr.in*) files, together or separately.

```
$ generate_PlanarFold_Inputs --seq ssRNA.seq --iCirc 1 --InitV 0 --vscale 0.1
                             -p ssRNA.prmtop  -c ssRNA.inpcrd
                             -ntr restraint_ntr.in  -dpr restraint_dpr.in
```

The sequence file (*ssRNA.seq*) should be in standard *.fasta* file format. Besides, below the sequence line in this file, a dot-bracket (dbn) formatted secondary structure can be added. If so, the prescribed secondary structure will be written into the restraint and confinement files. Of note, the initial conformation should be generated with initial velocity (especially at the Y-axis). Otherwise, the RNA cannot fold, but will only vibrate along the X-axis. User can also manually modify the coordinate file directly to assign desired velocity for specific residue(s).

If velocities are assigned, they will be assigned randomly using a uniform distribution. The random seeds are given by the time clock. To reproduce the simulation, the identical input files are required. To conduct parallel simulations at identical conditions, their coordinates must be different to yield different trajectories, which can be achieved by manual modification of the coordinate file, or by generating the coordinate files using *generate_PlanarFold_Inputs* by several times.

2.4.2. extract_Frame_into_inpcrd

The tool "*extract_Frame_into_inpcrd*" extracts the coordinates of a chosen frame, and converts them into a coordinate file (*ssRNA_extract.inpcrd*). Using this extracted frame, further studies of this conformation can be possible. Nonetheless, it is worth mentioning that, starting the simulation from this frame extracted from trajectory file (*run.nc*) is different from the original trajectory, as the velocities of this frame are not recorded along the simulation for restarting.

```
$ extract_Frame_into_inpcrd -x run.nc  -c ssRNA_extract.inpcrd  --iframe 0
```

2.4.3. read_PlanarFold_energy

The tool "*read_PlanarFold_energy*" extracts the energies or other data from the energy output file (*run.out*).

```
$ read_PlanarFold_energy  -o run.out  --ref 180bp_hairpin.ct  --var 9  --s 10
                          --needFrame 1 --frame0 0  --frame1 1000 --step 1
```

The command above gives the following results:

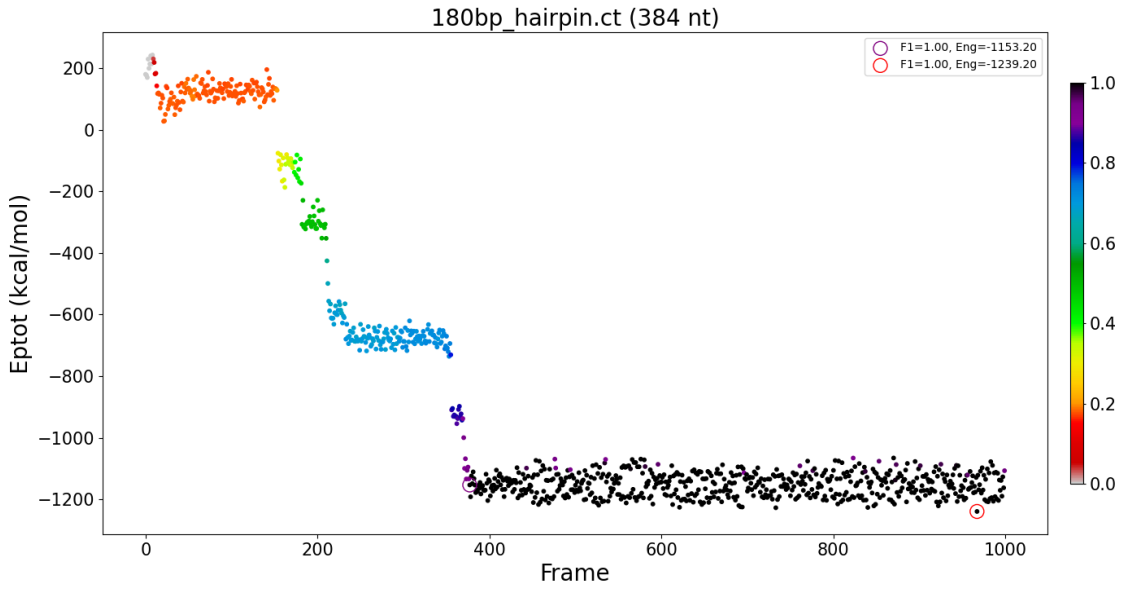


Figure 4. The potential energy evolvement of a 180-bp hairpin

It will plot the chosen energy (or other data, by variable "*var*") colored with the F1-score of each conformation. A secondary structure file (in "*.ct*" or "*.dbn*/*.dot*" file format) is required to calculate

the F1-score, which can be converted from dot-bracket format easily. Of note, *read_PlanarFold_energy* only supports "*.ct*" file in which the first data in the first line is RNA length, since it needs to read this number to determine RNA length. While for the ".dbn/.dot" format, the first line must be RNA sequence, and the second line must be dot-bracket.

See more description about the variables:

```
$ read_PlanarFold_energy -h
```

For *Python*-proficient users, the modification of the copy of the *read_PlanarFold_energy* script is recommended for more sophisticated variations. We also encourage other users to keep the usage of the rainbow-color spectrum to indicate F1-score, as it straightforwardly delineates the F1-score from low to high.

2.4.4. map_PlanarFold_BP

The tool "*map_PlanarFold_BP*" maps the base pairs (or secondary structures) along the trajectory. The base pairing information is extracted from the energy output file (*run.out*).

```
$ map_PlanarFold_BP -o run.out -p ssRNA.prmtop --step 1
                    --needFrame 1 --frame0 0 --frame1 1000 --ceiling -1
```

The command above gives the following results.

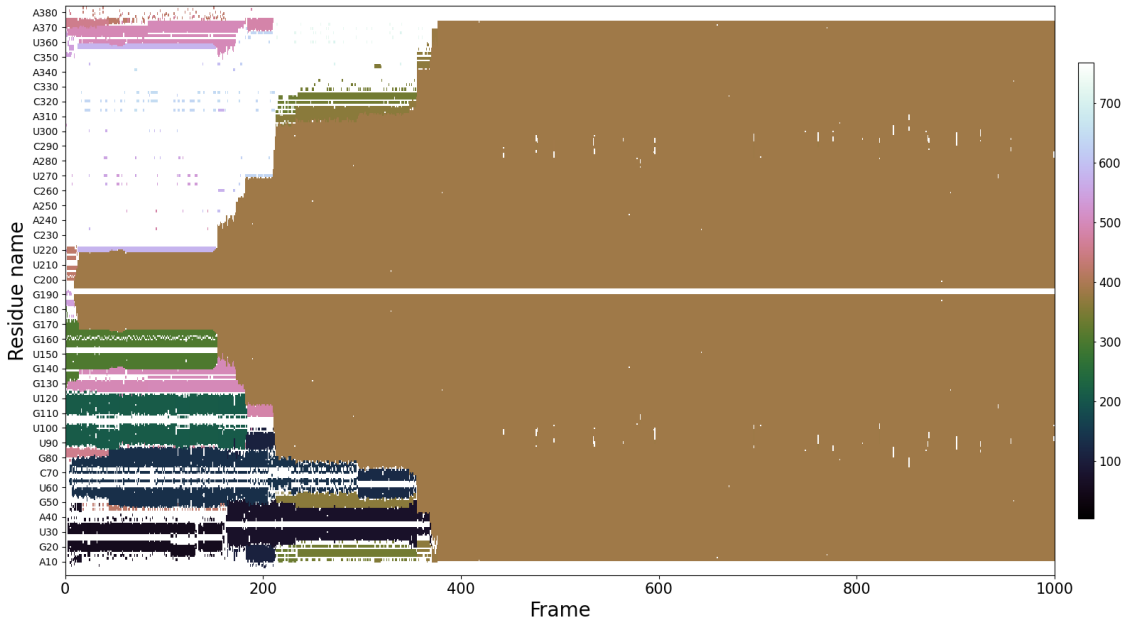


Figure 5. The base pair evolvement of a 180-bp hairpin

The essence of this tool is the indexing of the base pair. For continuous base pairs in the same stem (at least two continuous canonical base pairs), the summation of indexes of a base pair is always the same. For instance, for a stem of 4 bp (*Figure.1*), if residue i and j form base pair, their summation of index is i+j. Their neighboring residue i+1 and j-1 also form base pair, and their summation of index is (i+1) + (j-1) = i+j. There the summation of these 4 base pairs will share the same summation of index (subsequently naming as stem index). The stem index of stems close to

5’ terminus are smaller than that of stems close to 3’ terminus. This index difference can be employed to straightforwardly represent the secondary structure by mapping to appropriate color gradient. In the above map, each paired residue is colored according to its stem index. Of note, index = residue number -1, e.g., the index of the first residue is 0. Unpaired residues are colored as white by being set to the maximum of index. *Figure.4* and *Figure.5* use the same data from an energy output file (*run.out*) of a 180-bp hairpin.

See more description about the variables:

```
$ map_PlanarFold_BP -h
```

For *Python*-proficient users, the modification of the copy of the *map_PlanarFold_BP* script is recommended for more sophisticated variations. We also encourage other users to keep the stem indexing to better distinguish the secondary structures. Other colormaps can be found on Python website (*https://matplotlib.org/stable/users/explain/colors/colormaps.html*) for stronger contrast, The default colormap of *map_PlanarFold_BP* is “*cubehelix*”.

2.4.5. plot_Frames

The tool “*plot_Frames*” plots the exact coordinates of a single frame or several frames. If the base pair map drawn by *map_PlanarFold_BP* is not straightforward enough, *plot_Frames* can be employed to plot the exact coordinates and planar secondary structure.

```
$ plot_Frames -x run.nc -p ssRNA.prmtop -o run.out --ref 180bp_hairpin.ct
                --frame0 1 --frame1 402 --step 200 --subplot 1-3
                -s 3 --needRotate 0 --needEnergy 1
```

The command above gives the following results.

Figure 6. The coordinates and secondary structure of three frames of a 180-bp hairpin

This tool requires three input files: energy output file (*run.out*) for base pair information and energy information (if *needEnergy* =1); trajectory file (*run.nc*) for coordinates; and topology file (*ssRNA.prmtop*) for RNA sequence. To distinguish between native and non-native base pairs (by gray lines and red lines, respectively), a secondary structure file is required, either in "*.ct*" format or "*.dbn/.dot*" format. If *needRotate* is set to 0 (by default), the genuine coordinates extracted from trajectory file will be displayed. If *needRotate* is set to 1, the 5' residue of the first base pair (arrange from 5' to 3') will be moved to origin, and the vector of the first base pair (form 5' to 3') will be aligned to positive X axis. *Figure.4, 5, and 6* use the same data from an energy output file (*run.out*) of a 180-bp hairpin. Combining these 3 tools (*read_PlanarFold_energy*, *map_PlanarFold_BP ,* and *plot_Frames*), the secondary structure evolution and related energy changes of the RNA can be clearly characterized.

For *Python*-proficient users, the modification of the copy of the *plot_Frames* script is recommended for more sophisticated variations. We also encourage other users to keep the coloring of A, U, G, and C as red, green, orange, and cyan respectively for representation consistency.

2.4.6. visualization by VMD

The trajectory (*.nc* format) of PlanarFold can be directly visualized by *VMD*.

— open VMD program

— load the topology file:

at "*VMD main*" window, click [*File*] -> [*New Molecule…*];
at "*Molecule File Browser*" window, click [*Browse…*], choose "*ssRNA.prmtop"* file.

— load the trajectory file:

at "Molecule File Browser" window, click [*Browse…*], choose "*run.nc"* file.

— Change the "*Drawing Method*" (Optional):

at "*VMD main*" window, click [*Graphics*] -> [*Representation…*];
at "*Graphical Representation*" window, choose [*CPK*] at "*Drawing Method*" column;
change the "*Sphere scale*" to 4.0 and "*Bond Radius*" to 3.0.

— Change the Coloring of residues (Optional):

at "*VMD main*" window, click [*Graphics*] -> [*Colors…*];
at "Color Controls" window, choose [*Name*] at "*Categories*" column;
choose [*C*] at "Name" column, and change the [R, G, B] to [0.00, 0.75, 1.00] for cytosine;
choose [*A*] at "Name" column, and change the [R, G, B] to [1.00, 0.00, 0.00] for adenine;

choose [*G*] at “Name” column, and change the [R, G, B] to [1.00, 0.75, 0.00] for guanine;
choose [*U*] at “Name” column, and change the [R, G, B] to [0.00, 1.00, 0.00] for uracil.

An example of the above coloring and drawing method is shown as follows:

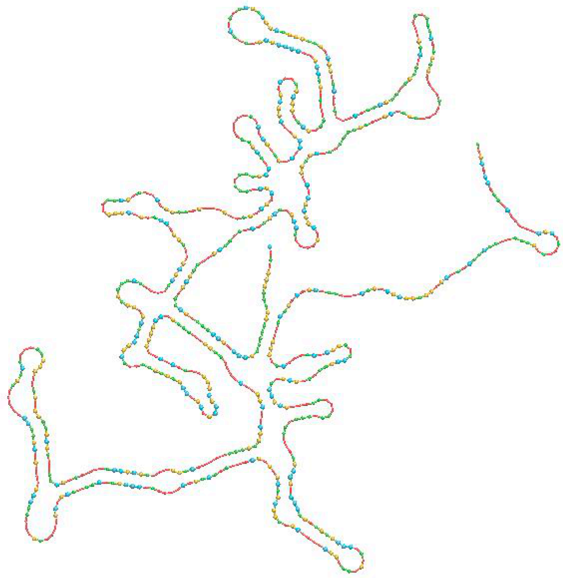


Figure 7. An example of PlanarFold’s trajectory visualized by VMD

Detailed usage of VMD can be found at: https://www.ks.uiuc.edu/Research/vmd/.

2.4.7. extract_DBs_from_MDout

The tool “*extract_DBs_from_MDout*” extracts the base pairs information and potential energies from energy output file (run.out), and store the secondary structures into dot-bracket format.

```
$ extract_DBs_from_MDout  -o run.out  --ref 1gid.dbn  --dots run.dots
                          --needFrame 1  --frame0 0  --frame1 25000  --step 250
```

The command above gives the following results.

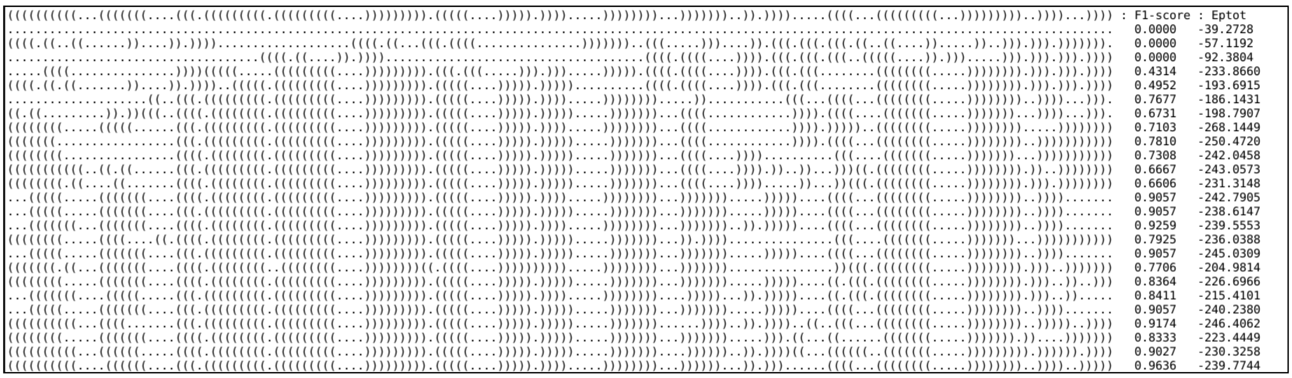


Figure 8. An example of run.dots file

This tool is useful for extracting the 0K conformations from simulated annealing cycles and for providing formatted secondary structures and calculated F1-scores for further analysis..

2.4.8. pca_PlanarFold_traj

The tool "*pca_PlanarFold_traj*" performs the principal component analysis (PCA) of the coordinates from trajectory file (*run.nc*) using contact map and plots the conformation in reduced space (2D).

```
$ pca_PlanarFold_traj  -x run.nc  -p ssRNA.prmtop  -dots run.dots
                       --bcut 14.0  --minLoop 4  -s 20
                       --needFrame 1  --frame0 0  --frame1 25000  --step 25
                       --bdwidth 0.005  --temp 300
```

or

```
$ pca_PlanarFold_traj --step 25  -s 20  --temp 300
```

The command above gives the following results.

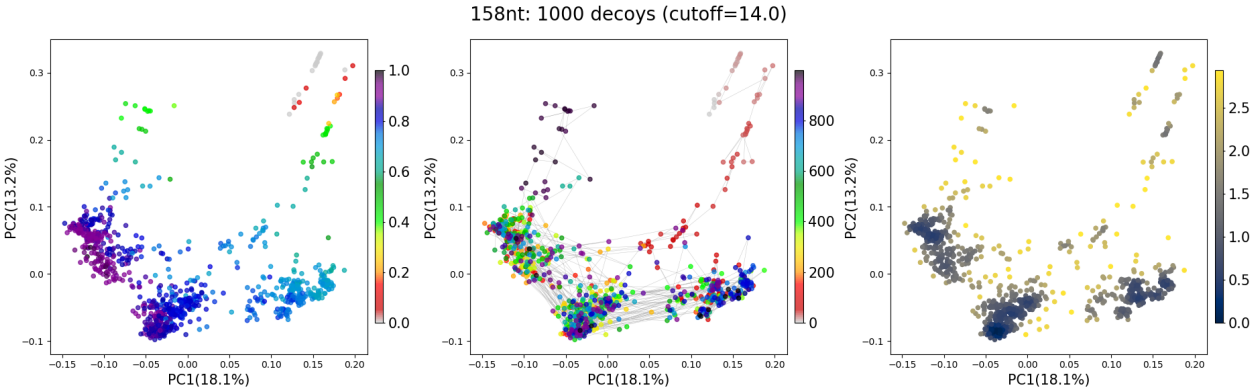


Figure 9. PCA results of a trajectory of a 180-bp hairpin

The same reduced trajectory is represented in three ways. In the left plot, the color indicates the F1-score of each conformation. In the middle plot, the color indicates the frame index along the trajectory, from which the time evolvement can be inferred. In the right plot, the color indicates the potential mean force (PMF) of each conformation, controlled by bandwidth (-- bdwidth) and temperature (-- temp). Of note, the value of the PMF is only qualitative, but not quantitative, since the sampling of such large conformation space is non-ergodic even for PlanarFold. The *.dots* input file should be prepared in advance using *extract_DBs_from_MDout*.

Users proficient in Python are encouraged to modify the copy of the *pca_PlanarFold_traj* script is recommended for more sophisticated variations. Except that the colormap of the left plot (F1-score) is encouraged, using other colormaps (*https://matplotlib.org/stable/users/explain/colors/colormaps.html*) is allowed.

2.4.9. draw_Contact_map

The tool "*draw_Contact_map*" draws the contact map of a specific frame using the coordinates recorded in trajectory file (*run.nc*). The values B(i,j) of the contact map are calculated as the same as the procedure in dynamic programming: B(i,j) = Cmap(i,j) × BpFlag(i,j). Cmap(i,j) is the

reciprocal of the square of distance of i and j: Cmap(i,j) = 1.0/dist2(i,j). BpFlag(i,j) is the weight for different types of base pairs: BpFlag(i, j) is 6, 4, 4, 2, and 1 for GC/CG, AU/UA, GU/UG, GA/AG, and other base pairs, respectively.

```
$ draw_Contact_map  -x run.nc  -p ssRNA.prmtop  --frame 200
                    --bcut 14.0  --minLoop 4
```

or

```
$ draw_Contact_map --frame 200
```

The command above gives the following results.

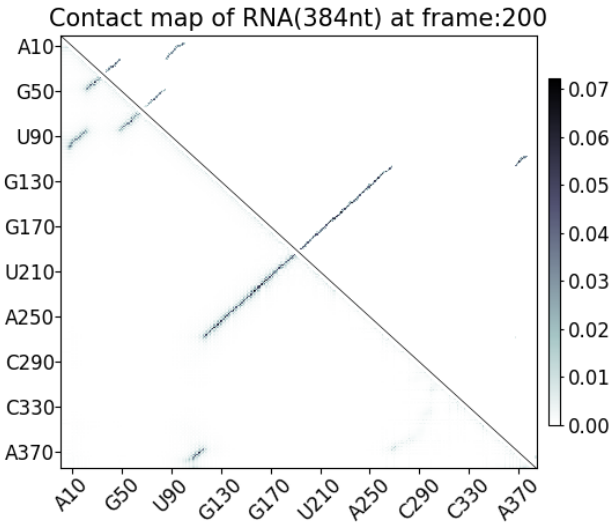


Figure 10. Contact map of frame-200 of a 180-bp hairpin

The contact map plotted is essentially symmetric, except that the top right triangle are screened with a cutoff (pairwise distance smaller than "*bcut*"). The top right triangle is identical to the inner data constructed at each frame for dynamic programming to determine the optimal base pairing.

3. Predict secondary structure using PlanarFold

3.1. Principle

PlanarFold's potential energy has been optimized to distinguish high energy (unfolded and misfolded) states and low energy (native and near-native) states using genetic algorithm. Using the default force field parameters described in section 2.2.2. (Force parameter file), the energy of the conformations can be directly used to predict secondary structure, as the conformation with the lowest potential energy should share the highest similarity with native state. The similarity with the native secondary structure is commonly indicated by F1-score, with an F1 of 1.0 being identical to native state.

PlanarFold employs a sample-and-select strategy, where conformation pool of diverse secondary structures should be generated at first, rendering it much less efficient in predicting secondary structure compared to traditional methods and deep-learning (DL) based methods. However, its secondary structure can be comparable to higher than traditional and some DL-based predictors.

To generate such diverse conformation pool, both restrained sampling and unrestrained sampling are resorted.

3.2. Benchmark

In the following, we use 6U8D as an example. The related files are stored at "*PlanarFold/benchmarks/predict_2Dstructure/6u8d*".

First, the sequence file should be provided (*6u8d.seq*). The description of secondary structure is optional.

```
>6U8D
GGGUCUCGCGGAACCGGUGAGUACACCGGAAUCCAGGAAACUGGAUUUGGGCGUGC
CCCCGCGAGACC
```

Second, use "*generate_PlanarFold_Inputs*" to generate input files.

```
$ generate_PlanarFold_Inputs -s 6u8d.seq --iCirc 1 --InitV 0
                             -p ssRNA.prmtop  -c ssRNA.inpcrd
                             -ntr restraint_ntr.in  -dpr  restraint_dpr.in
```

Third, perform the unrestrained sampling and restrained sampling.

For the unrestrained sampling, 1000 (a general option) cycles of Heating-and-Cooling (H&C) simulation to sample 1000 conformers for 6U8D (68 nt) should be performed.

The control parameter file (*control.in*) is listed below. In each H&C cycle, the temperature increases from 0 K to 800 K in 50,000 steps; then cooled from 800K to 300 K in 50,000 steps; and then cooled more slowly from 300K to 0 K in 100,000 steps; and eventually maintained at 0 K for

50,000 steps. The conformer at last frame (0K) of each H&C cycle is extracted, yielding a total number of 1,000 randomly sampled conformers from a 1-*μ*s simulation of 6U8D. The force parameter file has been prepared with all the default parameters.

```
&cntl
 ntpr = 10000,
 ntwx = 10000,
 ntr = 0,
 nscm = 1000,
 nstlim = 250000000,
 dt = 0.004,
 ntt = 1,
 ntemp = 4,
 tautp = 0.005,
 temp0 = 800,  300,   0,   0,
 tempi =    0, 800, 300,   0,
 duration = 50000, 50000, 100000, 50000,
 dpr = 0,
/
```

Figure 11. The contents of controls.in

Perform the simulation by following the command:

```
$ PlanarFold   -i control.in  -f fparm.in
               -p ssRNA.prmtop  -c ssRNA.inpcrd
               -ntr restraint_ntr.in  -dpr  restraint_dpr.in
```

For the restrained sampling, the *RNAstructure's Fold* module is resorted to generate vast number of diverse secondary structures in the first place.

To ensure that both near-native and diverse far-native conformations can be sampled while considering the computational burden for subsequent MD simulations, the top (up to) 500 structures at 4 window sizes sampled. The *RNAstructure* can be downloaded at https://rna.urmc.rochester.edu/RNAstructure.html. The commands to sample use *Fold* are as follows:

```
$ Fold 6u8d.seq  6u8d_w0.ct  --maximum 500  --percent 50  --window 0
$ Fold 6u8d.seq  6u8d_w1.ct  --maximum 500  --percent 50  --window 1
$ Fold 6u8d.seq  6u8d_w2.ct  --maximum 500  --percent 50  --window 2
$ Fold 6u8d.seq  6u8d_w3.ct  --maximum 500  --percent 50  --window 3
```

The following commands convert the ct files generated by RNAstructure's Fold module into dot-bracket formatted files for latter merging:

```
$ ct2dot_RNAstructure  -ct 6u8d_w0.ct  --dots 6u8d_w0.dots
$ ct2dot_RNAstructure  -ct 6u8d_w1.ct  --dots 6u8d_w1.dots
$ ct2dot_RNAstructure  -ct 6u8d_w2.ct  --dots 6u8d_w2.dots
$ ct2dot_RNAstructure  -ct 6u8d_w3.ct  --dots 6u8d_w3.dots
```

The conformations sampled at different windows are merged, with identical secondary structures removed, following the commands:

```
$ merge_Dots_windows  -n 4  --merge 6u8d_w0123_merge.dots
                    —dots 6u8d_w0.dots,6u8d_w1.dots,6u8d_w2.dots,6u8d_w3.dots
```

The above command gives the following information:

```
fnDots: ['6u8d_w0.dots', '6u8d_w1.dots', '6u8d_w2.dots', '6u8d_w3.dots']
nRes: 68
fnDot: 6u8d_w0.dots      nCand: 207
fnDot: 6u8d_w1.dots      nCand: 38
fnDot: 6u8d_w2.dots      nCand: 10
fnDot: 6u8d_w3.dots      nCand: 6
6u8d_w0123_merge.dots
 nCand_merge: 207
```

For short RNA such as 6u8d (68 nt), the conformations sampled under option “window=0” should cover all the conformations sampled under option “window=1, 2, or 3”, since the conformation space of small RNA is comparatively small. However, for larger RNAs (> 200 nt), the options other than “window=1” are necessary for generating a diverse conformation pool. Then, the merged dots file (*6u8d_w0123_merge.dots*) can be used for restrained MD simulation. Note that for restrained MD, the initial conformation should be circular rather than linear to prevent the formation of trans-isomers. The *.dots* files are deposited at directory “*/benchmarks/predict_2Dstructure/6u8d/rsn_RNAstructure/restraints/*”.

```
$ dots2restaints  --dots 6u8d_w0123_merge.dots
```

The above command generates 207 restraint and 207 confinement files (*restraint_ntr_NNN.in* and *restraint_dpr_NNN.in*) using all the sampled secondary structures from *6u8d_w0123_merge.dots*. To accelerate restrained MD, we apply confinement to avoid the time-consuming dynamic programming as only the restrained secondary structure is desired.

```
#!/bin/bash

istart=1
iend=207

I=$istart
while [ $I -le $iend ]
do
  suffix=`printf %03d $I`
  # pre-equilibrium to release long-range restraints tension
  PlanarFold  -i controls_prep.in  -f fparm.in -p ssRNA.prmtop  -c ssRNA.inpcrd \
              -o run.out  -x run.nc  -r run_prep_$suffix.rst \
              -ntr ./restraints/restraint_ntr_$suffix.in \
              -dpr ./restraints/restraint_dpr_$suffix.in
  # production run
  PlanarFold  -i controls.in  -f fparm.in -p ssRNA.prmtop  -c run_prep_$suffix.rst \
              -o run$suffix.out  -x run$suffix.nc  -r run$suffix.rst \
              -ntr ./restraints/restraint_ntr_$suffix.in \
              -dpr ./restraints/restraint_dpr_$suffix.in
  I=$((I+1))
done
```

Figure 12. The content of “run_multiple_restrainedMD.sh”

To avoid the trans-isomer, the restrained MD is arranged to two stages The first stage is the pre-equilibrium stage, where only long-range restraints are applied (by setting rcut=60.0, see /

benchmarks/predict_2Dstructure/6u8d/rsn_RNAstructure/controls_prep.in). The second stage is the production stage, using the restart file generated at the end of the first stage as the initial coordinates to run H&C cycle (the procedure is identical to unrestrained MD) with all restraints applied (see */benchmarks/predict_2Dstructure/6u8d/rsn_RNAstructure/controls.in*). In production stage, 5 cycles of H&C are performed, yielding 5 conformers under each restraint. In total, 1035 (207 x 5) conformer are generated using restrained sampling, along with 1000 conformer generated using unrestrained sampling. To accelerate the restrained sampling, the simulations can be performed at several sub-directory, which is necessary for long RNAs.

Eventually, 2035 (1035+1000) conformers (at 0 K) are extracted and merged into one file following the command:

```
$ merge_0K_conformers --unrestrained ./rsn_free/run.out
                      --restrained ./rsn_RNAstructure/  --Nrsn 207
                      --merge run_merge.out  --step 25
```

The merged energy output file (*run_merge.out*) can be used for subsequent analysis.

3.3. Analysis

The secondary structure of 6U8D is predicted as the conformer of lowest potential energy among all its sampled conformers.

```
$ read_PlanarFold_energy -o run_merge.out --ref 6U8D.ct -s 10
```

The command above gives the following results.

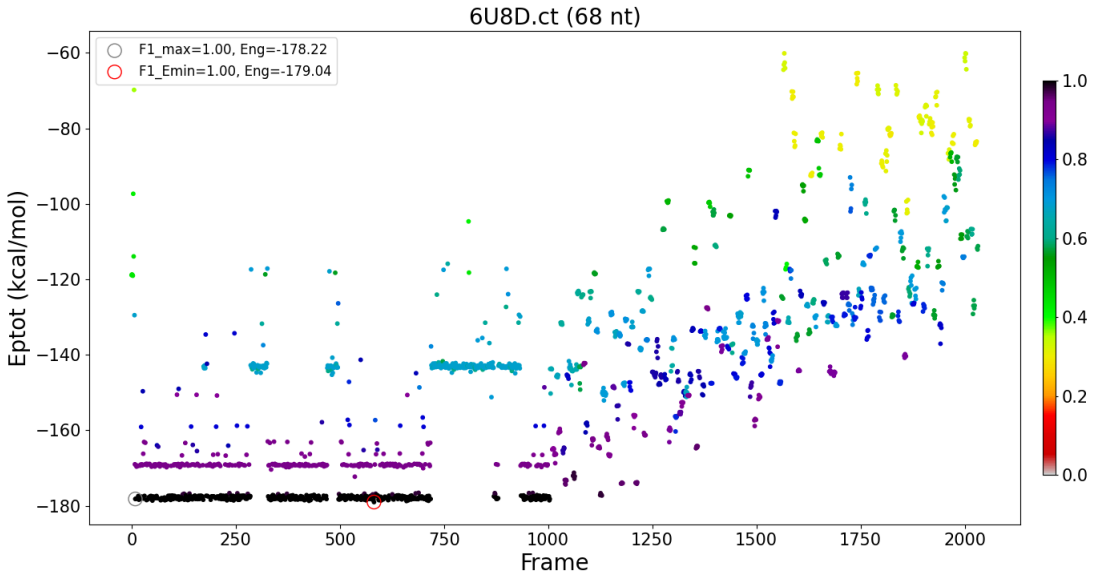


Figure 13. The potential energy of sampled conformers of 6u8d

From the above figure, the F1-score of the conformer with the lowest potential (red circle, Eptot=-179.04) is given, which is 1.0 at frame 582. And the highest F1-score is also shown (1.0, at frame 9, which is the first frame among all conformers with the highest F1-score).

```
Output: run_merge.out
nRes: 68
Calculating F1-score ...
nData = 2033
Highest F1-score at frame-9: F1=1.00, Eng=-178.22
dbn: .((((((((((..((((((....))))))((((((....)))))).(((....)))))))))))))
Lowest Energy at frame-582: F1=1.00, Eng=-179.04
dbn: .((((((((((..((((((....))))))((((((....)))))).(((....)))))))))))))
```

Figure 14. The printout information of read_PlanarFold_energy

Therefore, the secondary structure of 6u8d predicted by PlanarFold is F1-score = 1.0.

Inspecting the conformers and their associated energies (*Figure.13*), the unrestrained sampling may prove less important than restrained sampling, as generally restrained sampling yields more diverse conformers with lower energy than unrestrained sampling does. Therefore, to speed up secondary structure prediction process, users can perform only restrained sampling simulations. Certainly, a larger conformation pool will produce more robust results, if computation resources are abundant.

Besides, top-N predictions can be made following the command:

```
$ extract_DBs_from_MDout  -o run_merge.out  --ref 6U8D.ct  --dots run_merge.dots
$ reRank_decoys  --dots run_merge.dots  --rerank run_merge_reRanked.dots
$ predict_topN  --rerank run_merge_reRanked.dots  --topN 5
```

The commands above give the following results.

```
seq:GGGUCUCGCGGAACCGGUGAGUACACCGGAAUCCAGGAAACUGGAUUUGGGCGUGCCCCCGCGAGACC  68nt
topN: 5 nCand: 2035
  0 .((((((((((..((((((....))))))((((((....)))))).(((....)))))))))))))) 1.0000 -179.0393
  1 .((((((((((..((((((....)))))).((((((....))))))..(((....)))))))))))))) 0.9804 -177.3071
  2 .((((((((((..((((((....))))))..((((((....))))))...(((....)))))))))))))) 0.9600 -174.3888
  3 .((((((((((..((((((....))))))((((((......))))))).(((....)))))))))))))) 0.9804 -172.8505
  4 .((((((((((((((((((....))))))).((((((....))))))))))(((....)))))))))))) 0.9434 -172.3176
Best among top-5
  0 .((((((((((..((((((....))))))((((((....)))))).(((....)))))))))))))) 1.0000 -179.0393
```

Figure 15. The printout information of predict_topN

Since the top-1 prediction of 6U8D is already optimal (F1-score=1.0), considering more (top-5 tested here) conformers will not further elevate its secondary structure prediction performance. However, for many RNAs, especially the long ones, there is considerable room for improvements by incorporating top-N conformers.

For RNAs with unknown native secondary structure, the "*6U8D.ct*" file can be replaced with a simple "*unfolded.dbn*" file, where the dot-bracket can be simply represented with all dots, indicating an unfolded state. If so, all the conformations will have an F1-score of 0.0. The commands are as follows:

```
$ read_PlanarFold_energy -o run_merge.out --ref unfolded.dbn -s 10
$ extract_DBs_from_MDout  -o run_merge.out  --ref unfolded.dbn  --dots run_merge.dots
$ reRank_decoys  --dots run_merge.dots  --rerank run_merge_reRanked.dots
$ predict_topN  --rerank run_merge_reRanked.dots  --topN 5
```

The commands above give the following results.

```
seq:GGGUCUCGCGGAACCGGUGAGUACACCGGAAUCCAGGAAACUGGAUUUGGGCGUGCCCCCGCGAGACC  68nt
topN: 5 nCand: 2035
  0 .((((((((((..((((((....))))))(((((((....)))))))..(((....)))))))))))) 0.0000 -179.0393
  1 .((((((((((..((((((....))))))..((((((....))))))..(((....)))))))))))) 0.0000 -177.3071
  2 .((((((((((..((((((....))))))..(((((....)))))...(((....)))))))))))) 0.0000 -174.3888
  3 .((((((((((..((((((....))))))(((((......))))))..(((....)))))))))))) 0.0000 -172.8505
  4 .((((((((((((((((((....))))))..((((((....))))))(((....)))))))))))))) 0.0000 -172.3176
Best among top-5
  0 .((((((((((..((((((....))))))(((((((....)))))))..(((....)))))))))))) 0.0000 -179.0393
  1 .((((((((((..((((((....))))))..((((((....))))))..(((....)))))))))))) 0.0000 -177.3071
  2 .((((((((((..((((((....))))))..(((((....)))))...(((....)))))))))))) 0.0000 -174.3888
  3 .((((((((((..((((((....))))))(((((......))))))..(((....)))))))))))) 0.0000 -172.8505
  4 .((((((((((((((((((....))))))..((((((....))))))(((....)))))))))))))) 0.0000 -172.3176
```

Figure 16. The printout information of "predict_topN" (for unknown secondary structure)

For RNAs with unknown secondary structure, the top-N conformers will be ranked by the top-N lowest-potential-energy conformers. Users can select the top-N predicted conformations with the aid of human expertise and extra experimental information (such as SHAPE reactivity, etc.) for further validation.

4. Predict free energy difference and kinetic rates of transition

4.1. Principle

PlanarFold is tailored for predicting RNA dynamics, including thermodynamics and kinetics. PlanarFold can estimate the free energy differences between several conformation transitions of various modes. Moreover, the exchange rates estimated by PlanarFold highly correlate with experimental measurements. Therefore, PlanarFold can be used for predicting RNA excited states, the free energy difference between the excited state and ground state or between bistable states (thermodynamics), and the exchange rate between their transitions (kinetics). Although the exchange rate verified by PlanarFold has an error within approximately 1 order of magnitude, it should be very useful in helping researchers to estimate the timescale of uncharacterized conformation transitions and to guide the subsequent design of appropriate experiments for validation.

To bear in mind, only thermodynamics and kinetics, governed by secondary structure rearrangement instead of tertiary interactions, can be estimated by PlanarFold. Besides, the conformation transitions that occur at extremely slow rates (< 0.01 s^{-1}) are beyond the detection range of PlanarFold, as they would require at least 1000 μs of simulations to acquire statistically reliable exchange rates and populations of states. Future versions equipped with more efficient algorithms and GPU acceleration will be able to simulate those slow transition events.

4.2. Benchmark

In the following, we use a hairpin with 1-nt register shift (T1)[1] and a bistable RNA (Bi55)[2] as an example (*Figure.17*). The related files are stored at "*PlanarFold/benchmarks/thermodynamics_kinetics/*".

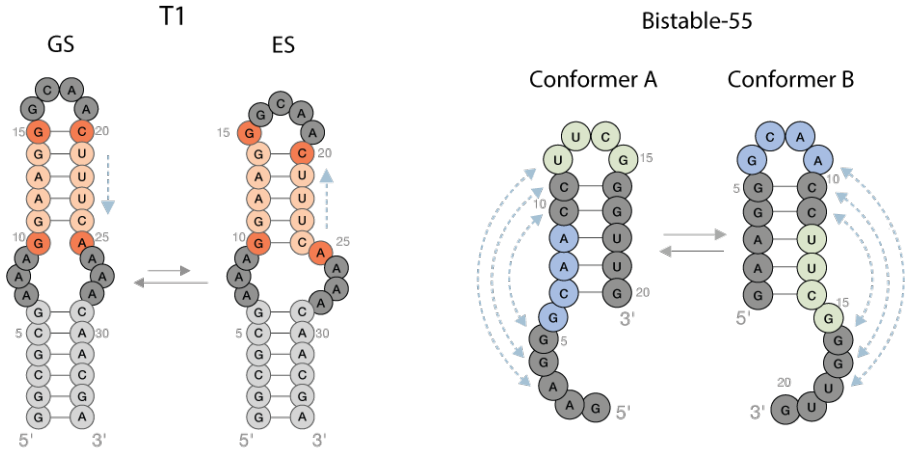


Figure 17. Two types of conformation transitions

First, the sequence file of T1 and Bi55 should be provided (*t1.seq* and *bi55.seq*).

```
>t1
GGCGCGAAAGGAAGGGCAACUUUCAAAACGCGCC
```

And

```
>bistable_55
GAAGGGCAACCUUCGGGUUG
```

Second, use "*generate_PlanarFold_Inputs*" to generate input files.

```
$ generate_PlanarFold_Inputs  -s t1.seq  --iCirc 1  --InitV 0
        -p ssRNA.prmtop -c ssRNA.inpcrd -ntr restraint_ntr.in -dpr restraint_dpr.in
```

And

```
$ generate_PlanarFold_Inputs  -s bi55.seq  --iCirc 1  --InitV 0
        -p ssRNA.prmtop -c ssRNA.inpcrd -ntr restraint_ntr.in -dpr restraint_dpr.in
```

Third, perform the simulation at desired temperature (283 K for T1, 443 K for Bi55).

The T1 is simulated for 1-μs under 283 K, with 100,000 frames recorded.

```
&cntl
 ntpr = 2500,
 ntwx = 2500,
 ntr = 1,
 restraint_wt = 0.001,
 restraint_eq = 10.0,
 rcut = 20.0,
 nscm = 1000,
 nstlim = 250000000,
 dt = 0.004,
 ntt = 1,
 ntemp = 1,
 tautp = 0.005,
 temp0 = 283,
 tempi = 283,
 duration = 50000,
 nVerlet = 1000,
 verlet_cut = 30.0,
/
```

```
&cntl
 ntpr = 25000,
 ntwx = 25000,
 ntr = 0,
 restraint_wt = 0.1,
 restraint_eq = 10.0,
 rcut = 11.0,
 nscm = 1000,
 nstlim = 250000000,
 dt = 0.004,
 ntt = 1,
 ntemp = 1,
 tautp = 0.005,
 temp0 = 433.0,
 tempi = 433.0,
 duration = 50000,
/
```

Figure 18. The control parameter file (controls.in) of T1 (left) and Bi55 (right)

The Bi55 is simulated for 1-μs under 443 K, with 10,000 frames recorded in each run. The sampling frequency is lower than that of T1 as the exchange rate of Bi55 (0.182 s^{-1} at 298 K) [2] is much smaller than that of T1 (453 s^{-1} at 283 K).

Importantly, users should pay attention to the simulation conducted other than at 283 K, since the simulation temperature of PlanarFold is not identical to the experiment temperature except for 283 K. A conversion relationship between PlanarFold simulation temperatures and realistic temperatures has been established during the **temperature calibration**: ***T_{sim} = (T_{exp} - 283) x 10 + 283 (in Kelvin)***. For instance, to simulate an experiment performed at 298 K, the simulation temperature in PlanarFold should be set to 443 K, calculated as T_{sim}(298 K) = (298 - 283)*10 + 283 = 433 (Kelvin). Moreover, we applied very weak restraints (*ntr* = 1, *restraint_wt* = 0.01, *rcut* =20.0) to the two terminal base pairs of T1, to accelerate the folding into its ground state (GS) or excited state (ES). Start the simulation of T1 simply by typing the following command:

```
$ PlanarFold
```

On the other hand, for transitions with much lower frequency (exchange rates < 1 s^{-1}), such as bistable RNAs, the simulation might require as long as 1000 μs to guarantee sufficient sampling. For Bi55, we can split the 1000-μs simulation into 5 independent parallel simulations, with each replica lasting 200 μs. Since the parameter *nstlim* is 32-bit (C language int type), the maximum of *nstlim* is 2^{31}-1 (2,147,483,647), suggesting that one round of simulation cannot exceed ~8.5 μs. Hence, simulations longer than 8.5 μs must be conducted in discontinuous fashion. Besides, a single continuous long simulation is not suggested, as any program abortion or interruption can obsolete the simulation that might have been running for days.

Perform simulation of Bi55 by executing the following script:

```
$ ./run.sh
```

The contents of “run.sh” is listed:

```
#!/bin/bash

# start simulation using ssRNA.inpcrd
PlanarFold  -i controls.in  -f fparm.in -p ssRNA.prmtop  -c ssRNA.inpcrd \
              -o run001.out  -x run001.nc  -r run001.rst \
              -ntr restraint_ntr.in -dpr restraint_dpr.in

# extend simulation using restart file
istart=2
iend=100
I=$istart
while [ $I -le $iend ]
do
  suffix=`printf %03d $I`
  suffix2=`printf %03d $((I-1))`
  # production run
  PlanarFold  -i controls.in  -f fparm.in -p ssRNA.prmtop  -c run$suffix2.rst \
              -o run$suffix.out  -x run$suffix.nc  -r run$suffix.rst \
              -ntr restraint_ntr.in -dpr restraint_dpr.in
  I=$((I+1))
done
```

Figure 19. Execute “run.sh” to run discontinuous simulation of Bi55

The above commands list in “*run.sh*” conduct a 100-μs simulation. If 200 μs is required, start the simulation at the second stage (*# extend simulation using restart file*), and change the variable “istart” and “iend” to 101 and 200, respectively, after the first 100 μs simulation is finished. Or, directly change the “iend” to 200 and execute the above script.

4.3. Analysis

First, use the tool *map_PlanarFold_BP* to directly check the conformation transition of T1 and Bi55 following the commands, respectively:

```
$ map_PlanarFold_BP  -p ssRNA.prmtop --ceiling 0 --intY 2
                     --iMulti 0 -o run.out
                     --step 1 --needFrame 1 --frame0 0 --frame1 1000
```

```
$ map_PlanarFold_BP  -p ssRNA.prmtop --ceiling 1 --intY 1
                     --iMulti 1 --prefix run --files 1-30 --len 3
                     --step 100 --needFrame 1 --frame0 0 --frame1 10000
```

The commands above give the following results.

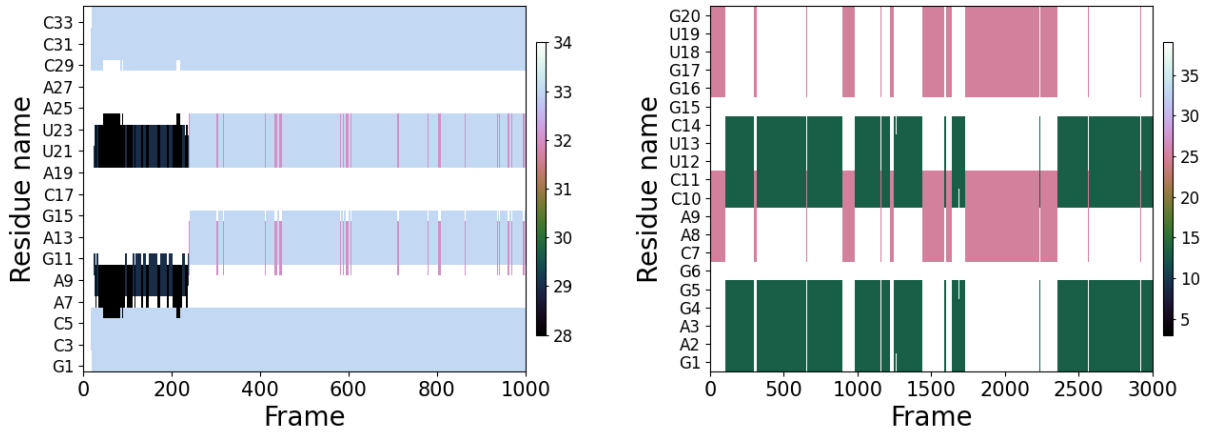


Figure 20. The conformation transition of T1 (left) and Bi55 (right)

For T1, the GS and ES only differ at the upper stem. In *Figure.20* (left), we only plot the first 1,000 frames of the entire 100,000 frames for analysis. Before frame about 220, the upper stem is misfolded. After frame 220, the upper stem switches between GS and ES, with much more GS (light blue, index=33) than ES (pink, index=32).

For Bi55, the two alternative states share entirely different base pairings. In *Figure.20* (right), we only plot the first 30 μs of the entire 100-μs simulation. Conformer A is colored pink (index=25), while conformer B is colored green (index=13). Compared with T1, the transition frequency is much lower for Bi55, in agreement with experiments.

To quantitatively analyze the thermodynamics and kinetics, the tool *calc_ddG_and_kinetics* should be used:

```
$ calc_ddG_and_kinetics  --seq t1.seq --iMulti 0  -o run.out --temp 283
```

To study the transition events of interest, the secondary structure of two states should be specified in a file (*t1.seq*), with the first line being the sequence, and the second and third lines being the secondary structure of the two states in dot-bracket format. Besides, the temperature should be specified too. To note, the temperature here should be the experiment temperature, which can be converted from PlanarFold's simulation temperature following the equation: T_{sim} = (T_{exp} - 283) x 10 + 283 (in Kelvin).

The command above gives the following results for T1.

```
State   Secondary structure                Population   Percent
  A:   ((((((....(((((....)))))....))))))     93821      94.13%
  B:   ((((((...(((((.....)))))....))))))      5855       5.87%
SEQ    GGCGCGAAAGGAAGGGCAACUUUCAAAACGCGCC

Temp= 283K, ΔΔG(B->A) = ΔG(A)-ΔG(B) = -1.56 kcal/mol

(nB+nA)/nTotal = 99676/100000 = 99.676%

Transition times (A->B) = 1899
Transition times (B->A) = 1899
```

Figure 21. The simulated thermodynamics and kinetics of T1

The free energy difference between two states is calculated following the Boltzmann distribution: $\Delta\Delta G(B->A) = \Delta G(A) - \Delta G(B) = -8.3145 \times Temp \times \ln(pA/pB) / 4184$, where the Temp is in unit Kelvin; pA and pB are the population of state A and state B, respectively; the value 4184 is the conversion rate from kcal to Joule. Here, the free energy difference between the state A (GS) and state B (ES) is -1.56 kcal/mol, very close to experimental measurement (-1.57 kcal/mol, at 283 K). The GS transits to ES 1899 times in 1-μs simulation, and vice versa. The forward (backward) rates, defined as the rates from state A to B (state B to A), were determined by dividing the number of forward (backward) transitions by the dwell time at state A (state B). Hence, the forward rate (GS->ES) is 1899/(pGS × Simulation time) = 2017.5 μs^{-1}. And the backward rate (ES->GS) is 1899/(pES × Simulation time) = 32329 μs^{-1}. The exchange rate is calculated as: $k_{ex} = k_{forward} + k_{backward}$. The exchange rate is 34346.5 μs^{-1}, which is 5 x10^4 of magnitude higher than experimental measurement (453 s^{-1}). This is the reason why the equivalent time should be adopted (20 μs in PlanarFold's simulation time converts to 1 s in realistic time). Besides, the GS and ES occupy ~99.7% of all sampled conformers, suggesting such simulation is valid for calculation.

In comparison, several trajectories (energy files) of Bi55 should be analyzed as follows:

```
$ calc_ddG_and_kinetics --seq bi55.seq --iMulti 1 --prefix run --files 1-50 --len 3 --temp 298
```

The command above gives the following results for Bi55.

```
State   Secondary structure  Population   Percent
  A:    (((((....)))))......    371273      74.89%
  B:    ......(((((....)))))    124497      25.11%
SEQ     GAAGGGCAACCUUCGGGUUG

Temp= 298.0K, ΔΔG(B->A) = ΔG(A)-ΔG(B) = -0.65 kcal/mol

(nB+nA)/nTotal = 495770/500000 = 99.154%

Transition times (A->B) = 19
Transition times (B->A) = 19
```

Figure 22. The simulated thermodynamics and kinetics of Bi55

The secondary structures are described in file bi55.seq. The free energy calculated between conformer A and conformer B is -0.65 kcal/mol, close to experimental measurement (-0.11 kcal/mol, at 298 K). The GS transits to ES 19 times in 50-μs simulation, and vice versa. Hence, the forward rate is calculated as: 19/(0.7489*50 μs) = 0.507 μs^{-1}. And the backward rate is 1.513 μs^{-1}.

The exchange rate is 2.02 μs^{-1}, which is also about the same time scale as the experimental measurement (0.182 s^{-1}) by multiply 5 x10^4 (equivalent time ration). For slow transition of Bi55, simulation should be extended for statistically significant sampling.

5. Simulate co-transcriptional folding

5.1. Principle

PlanarFold is equipped to simulate the co-transcriptional folding of RNA by progressively adding nascent residues along the transcribed RNA chains. Currently, PlanarFold only supports adding 1 nucleotide each time to avoid crashes caused by collisions with transcribed residues. With PlanarFold's drastically increased sampling efficiency, it could capture the conformation transitions ranging from microseconds to second, using the equivalent simulation time. The transcription speed can be modulated by adjusting the simulation time accordingly.

Given the stochastic nature of MD simulations, several parallel simulations should be conducted to better predict the co-transcriptional folding pathway and end products. Additionally, PlanarFold records a continuum of transcripts in diverse conformations across simulation time. This dynamic ensemble may provide valuable insights into how RNAs exert their biological functions, with the aid of complementary experimental information.

5.2. Benchmark

Here, we use a bistable switch (reverse switch) for benchmarking. This RNA can adopt two disparate conformations, which has been validated by experiments[3]. The related files are stored at "*PlanarFold/benchmarks/cotranscriptional/reverse_switch/*".

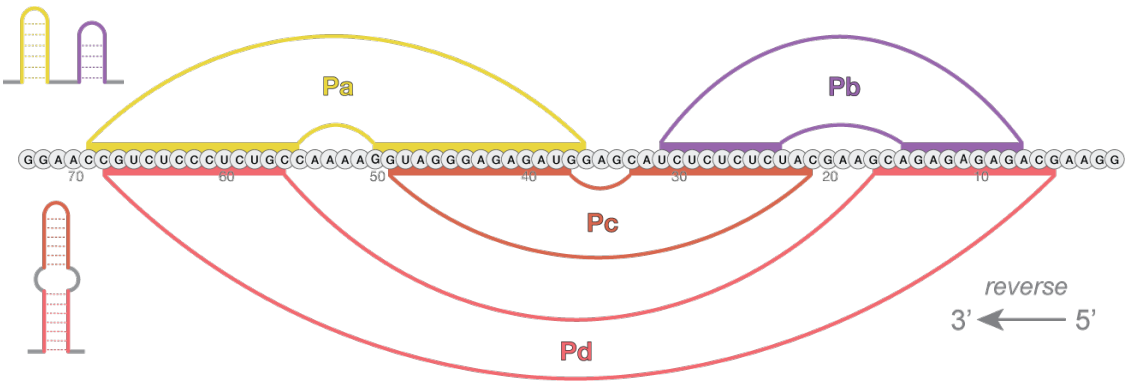


Figure 23. The two conformations of reverse switch

First, the sequence file of reverse switch should be provided (*reverse_switch.seq*).

```
>cofold_r
GGAAGCAGAGAGAGACGAAGCAUCUCUCUCUACGAGGUAGAGGGAGAUGGAAAACCG
UCUCCCUCUGCCAAGG
```

Then, use the following script (*fold.sh*) to perform co-transcriptional folding:

```
$ ./fold.sh
```

```
#!/bin/bash

istart=10
iend=73

# Step1. prepare the initial unfolded conformation (skip this step if conformation is already provided)
cut -c1-$istart reverse_switch.seq > transcript.seq
generate_PlanarFold_Inputs -s transcript.seq -c ssRNA`printf %03d $istart`.inpcrd

# Step2. start the simulation
I=$istart
while [ $I -le $iend ]
do
  suffix=`printf %03d $I`
  suffix2=`printf %03d $((I+1))`

  # Generate the sequence file for this transcript
  cut -c1-$I reverse_switch.seq > transcript.seq

  # Generate topology file of this transcript using sequence file, but do not generate coordinate file
  generate_PlanarFold_Inputs -s transcript.seq -p ssRNA$suffix.prmtop -ntr restraint_ntr.in -dpr restraint_dpr.in

  # Production run
  PlanarFold  -i controls.in  -f fparm.in -p ssRNA$suffix.prmtop  -c ssRNA$suffix.inpcrd \
              -o run$suffix.out  -x run$suffix.nc  -r run$suffix.rst \
              -ntr restraint_ntr.in -dpr restraint_dpr.in

  # Elongate the transcript (only adding 1-nt is allowed) using restart file, generate extended coordinate file
  elongate_Transcript -r run$suffix.rst -c ssRNA$suffix2.inpcrd

  I=$((I+1))
done
```

Figure 24. The contents of fold.sh

The script above performs the co-transcriptional folding simulation of reverse switch from 10 nt to 73 nt (its full length). Three files should be provided beforehand: 1). the sequence file of reverse switch in .fasta format (reverse_switch.seq); 2). the force parameter file (*fparm.in*); 3). the control parameter file (*controls.in*). Because transcripts have different lengths, each simulation for a specific transcript needs a topology file, a coordinate file, a restraint file, and a confinement file, which are generated automatically running this script. At step1, an initial coordinate file of unfolded (either circular or linear, circular by default) conformation of 10-nt is generated. If the coordinate file is provided for a transcript (X nt), then comment out the “Step1" section and change the “10” to “X”.

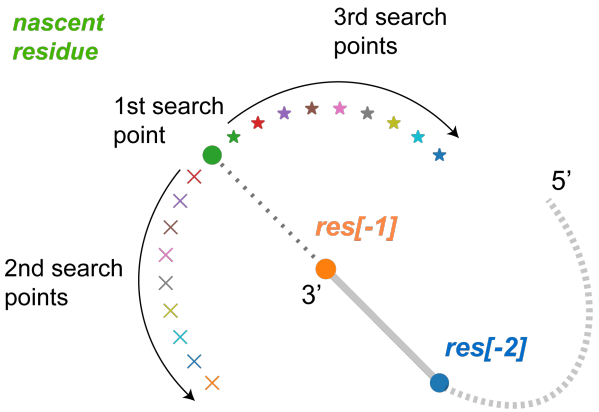


Figure 25. The placement of the nascent residue

In *Figure.25*, we elucidate the principle of our transcript extension strategy. First, the nascent residue is placed at the “1st search point” with a distance of 5.5 Å to the last 3’ residue (res[-1]), which is along the extensive line of the last two residues (res[-2] and res[-1]). If this point does not

collide with other transcribed residues (distance larger than 7.7 Å, with residues other than res[-1]), this placement is adopted. Otherwise, an array of nine “2nd search points” are searched sequentially, with each point representing 10° clockwise rotation of “1st search point” with an half radius of 5.5 Å around res[-1]. The collision with transcribed residues is also checked. If none of those positions satisfy the requirements, another array of nine “3rd search points” are searched, in counter-clockwise direction, and same checking process is performed. Eventually, if none of those points satisfy the requirements, the program “elongate_Transcript” will abort, giving a suggestion for selecting another frame for transcript extension. The users can choose the second last frame from the trajectory for transcript extension, and so on.

To better mimic the transcription *in vitro*, usually catalyzed by T7 polymerase, the simulation time must be carefully calculated to match with the experimental transcription speed. Generally, the transcription speed of T7 polymerase is 200-400 nt/s. Hence, the life span of each transcript is 2.5-5.0 ms. Employing equivalent time, the simulation time for each transcript should be set to 50-100 ns. Of note, the equivalent time has an error of one order of magnitude. Therefore, any simulation time within the range 5 ns to 1000 ns is acceptable. Simulations at these two extremes may be tested both, to determine whether variation of transcription speed within this range could affect the dynamics or other characteristics of interest.

5.3. Analysis

To visualize the co-transcriptional folding pathway of reverse switch, following the command:

```
$ map_Cotranscriptional_folding -s reverse_switch.seq --prefix run --files 10-73
        --step 50 --needFrame 1 --frame0 0 --frame1 500 --len 3 --intY 5
```

Combine with the following commands to check the secondary structure of the end product of this simulation more clearly.

```
$ plot_Frames -o run073.out -p ssRNA073.prmtop -x run073.nc --frame0 500 --frame1 501
```

The commands above give the following results.

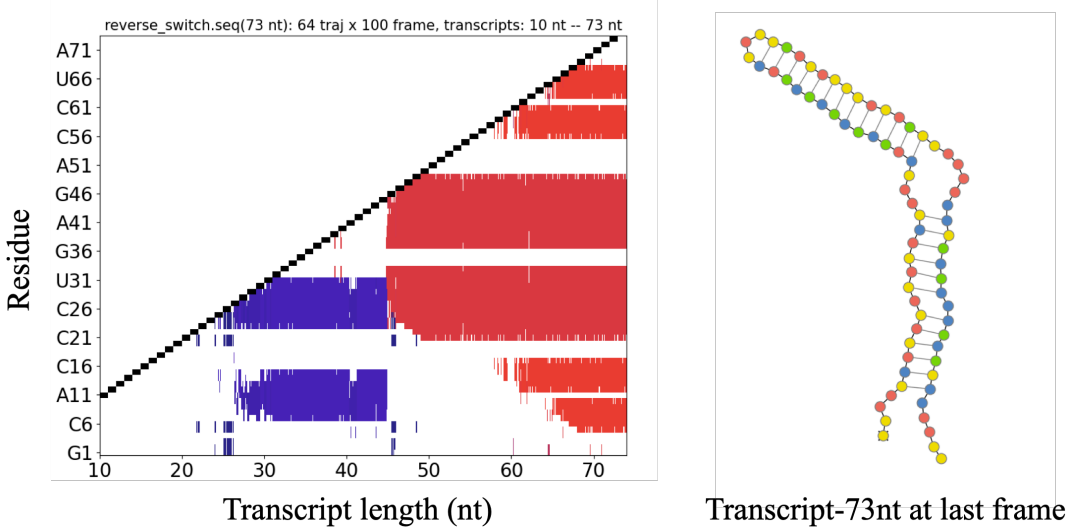


Figure 26. The co-transcriptional folding pathway and end product (rod-like) of reverse switch

The base pairing evolution shows that helix Pb forms first, followed by Pc, which competes with the 3' segment of Pb for pairing. And eventually, Pd forms to yield the rod-like conformer. Of note, this result is only one of the possible fates of reverse switch. To acquire a more comprehensive understanding of the co-transcriptional folding pathways of reverse switch, at least 10 repeats of co-transcriptional folding simulations should be conducted. Another folding pathway, identified through multiple simulation replicates, is shown below and can be visualized using the commands:

```
$ map_Cotranscriptional_folding -s reverse_switch.seq --prefix run --files 10-73
                --step 50 --needFrame 1 --frame0 0 --frame1 500 --len 3 --intY 5

$ plot_Frames  -o run073.out -p ssRNA073.prmtop -x run073.nc
                -s 12 needRotate 1 --frame0 500 --frame1 501
```

The commands above give the following results.

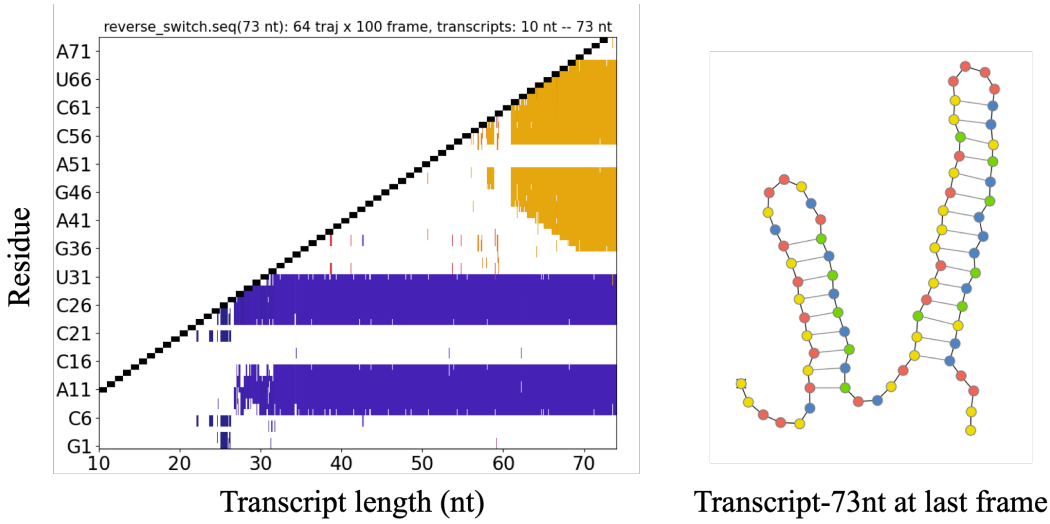


Figure 27. The co-transcriptional folding pathway and end product (branched) of reverse switch

6. Steered MD simulation

6.1. Principle

We implemented steered MD (SMD) simulation into PlanarFold to mimic the force-steered pulling and relaxing by optical tweezer (OT) or atomic force microscopy (AFM). *In vivo*, the mechanical unfolding also occurs under the regulation of cellular machineries. Similar to co-transcriptional folding pathways, the mechanical unfolding or refolding pathways are diverse and require parallel simulations. Each simulation trajectory can be deemed as a record of a single molecule event.

Currently, we have achieved the constant speed (c.v.) mode and constant force (c.f.) mode of SMD. The passive mode of steered can be readily achieved by setting the initial distance and final distance as the same in c.v. mode.

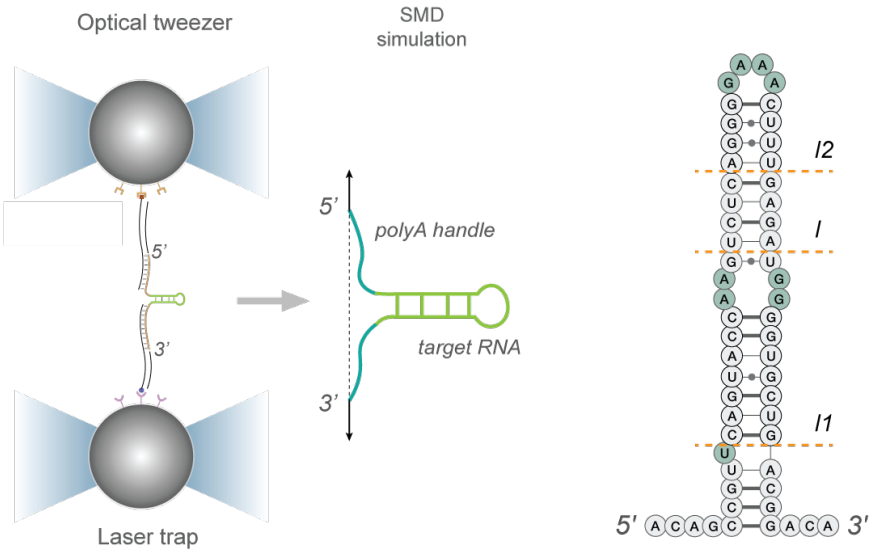


Figure 28. The set of OT experiment and SMD simulation (left) and benchmark (p5ab, right)

6.2. Benchmark

6.2.1. Constant velocity mode

In the following, we use a simple hairpin p5ab (*Figure.28*, right)[4], which has been extensively tested by optical tweezer experiments, as a benchmark. This hairpin is modified from the p5abc domain of the *Tetrahymena thermophila* group-I intron. The related files are stored at "*PlanarFold/benchmarks/mechanics/p5ab*".

First, the sequence file of p5ab should be provided (*p5ab.seq*).

```
>p5ab
ACAGCCGUUCAGUACCAAGUCUCAGGGGAAACUUUGAGAUGGGGUGCUGACGGACA
```

Of note, in experiment, the RNAs are connected to handles, either DNA duplex or DNA/RNA hybrid duplex ranging from 2 kb to 10 kb, which is not yet compatible with PlanarFold model. Therefore, polyA handles are generally added to the 5' terminus and 3' terminus of the RNA of interest (*Figure.28*, left), to account for the dynamic effect of handle fluctuation in OT experiments. Poly G, U, and C can be used as well. However, G and U have a higher change of forming base pair (GC/UA and GU/UG) than A, and C can form CG base pair, which is stronger than AU. Therefore, poly-A handles are recommended, with a length of 10 nt to 200 nt. Simply add the polyA sequences into the sequence file (.seq) to generate input files for simulation of RNAs with handles,

Second, use "*generate_PlanarFold_Inputs*" to generate input files.

```
$ generate_PlanarFold_Inputs -s p5ab.seq --iCirc 0 --InitV 1 --vscale 0.1
        -p ssRNA.prmtop -c ssRNA.inpcrd -ntr restraint_ntr.in -dpr restraint_dpr.in
```

Third, perform the simulation at desired conditions by modifying the related variables in force parameter file (*fparm.in*) and control parameter file (*controls.in*).

```
&cntl
 ntpr = 500000,
 ntwx = 500000,
 ntr = 0,
 ismd = 2,
 dumpfreq = 5000,
 stretch_force = 1.0,
 stretch_spring = 0.000033,
 dist_init = 5900.0,
 dist_final = 900.0,
 restraint_wt = 0.1,
 restraint_eq = 10.0,
 rcut = 11.0,
 nscm = 1000,
 nstlim = 625000000,
 dt = 0.004,
 ntt = 1,
 ntemp = 1,
 tautp = 0.005,
 temp0 = 433.0,
 tempi = 433.0,
 duration = 50000,
 dpr = 0,
/
```

```
&parm
 sigma = 9.8,
 epsilon = 1.1,
 vcut = 13.5,
 bphm_eq = 10.0,
 bphm_k = 0.3,
 bphm_p = 9.6,
 angle_eq = 3.1415926535898,
 angle_k = 4.0,
 angle_p = -3.7,
 bond_eq = 5.5,
 bond_k = 20.0,
 charge = 1.0,
 ecut = 20.0,
 len_Debye = 10.0,
 min_loop = 4,
 minbpFLAG = 2,
 minStembp = 2,
 bulge_p = 0.02,
 bcut = 14.0,
 stk_eq = 1.57079633,
 stk_k = 0.9,  1.1,  2.1,  2.2,  0.6,  1.4, 1.
3,  0.9,  2.1,  2.4,  1.0,  1.3, 2.4,  2.2,  3.
3,  3.4,  1.5,  2.5, 2.1,  2.1,  2.4,  3.3,  1.
4,  2.1, 1.3,  1.4,  2.1,  2.5,  0.5, -1.3, 1.0
,  0.6,  1.4,  1.5, -0.3,  0.5,
 stk_ks = 1.5,
 stk_p = -3.0,
 bpFLAG = 1, 4, 1, 2, 4, 1, 1, 1, 6, 1,
 nHB = 0, 2, 0, 0, 2, 0 ,0 ,0, 3, 0,
 f_scale = 0.33,
/
```

Figure 29. The control parameter file and force parameter file for c.v.SMD

There are several critical variables in OT experiment: trap stiffness, pulling/relaxing speed, force region, and acquisition frequency.

1). The trap stiffness (often denoted by κ), in optical tweezer experiments, is a measure of the restoring force exerted by the optical trap on a trapped particle when it is displaced from the equilibrium position. In MD simulation, the trap stiffness essentially equals to the force constant of the virtual spring, which are both the spring constant of harmonic potential. To bear in mind, the

apparent stiffness in OT experiment is the modulation of trap stiffness and handle stiffness. The handle stiffness should be reflected by the handles themselves, which is hard to simulate yet. Hence, the apparent stiffness is still hard to reach by PlanarFold yet. More detailed and systematic simulation of OT experiment might be achieved in the future.

PlanarFold adopts the same SMD strategy of *Amber*: $f_{steering} = 2 \cdot k_{spring} \cdot (X(t) - X_{RNA})$, where X_{RNA} is the distance between 5' end and 3' end of the RNA, and X(t) is the equilibrium value of the virtual spring. X(t) is linearly interpolated between the initial distance (*dist_init*) and final distance (*dist_final*) set by the user, to attain the desired pulling or relaxing speed. And, k_{spring} is the spring constant (*stretch_spring*) of the virtual pulling spring.

The experimental trap stiffness we want to simulate is 0.046 pN/nm. Therefore, according to equation: $2 \cdot k_{spring} = 0.046$ pN/nm, k_{spring} is 0.023 pN/nm. Since the variable "stretch_spring" is in unit kcal·mol^{-1}·Å^{-2}, and 1 kcal·mol^{-1}·Å^{-2} converts to 69.5 pN·Å^{-1}. The k_{spring} is set to 0.023/69.5/10 kcal·mol^{-1}·Å^{-2} = 0.000033 kcal·mol^{-1}·Å^{-2}.

2). Loading speed or pulling/relaxing speed is the moving speed of optical trap devices. Owing to the enhanced sampling of PlanarFold, experimental loading speed can be comparatively closely reached. In OT experiments, the loading speed is usually in the range of 10 nm/s to 400 nm/s, which translates to simulation speed of 10 nm/μs to 400 nm/μs in PlanarFold, by employing equivalent time. Here, the experiment loading speed is 200 nm/s, therefore, the simulation speed is 200 nm/μs (20-fold of experiment loading speed even after converting to equivalent time). In PlanarFold, the simulation speed is determined by several parameters: (*dist_final* - *dist_init*) / (*nstlim*dt)*, which should be determined by the force range of interest.

3). Force region is the window where conformation transitions (unzipping or rezipping, etc.) of interest happen. For p5ab, the experimental measured ripping region is around 13-16 pN, and the force-extension curves (FEC) exhibit force generally ranging from 5 pN to 25 pN. Therefore, users should set the final distance (*dist_final*) and initial distance (*dist_init*) to certain value such that their corresponding force be around 25 pN and 5 pN, respectively. Here, we try to simulate the refolding process, instead of the unfolding process, so the final force and initial force should be 5 pN and 25 pN, respectively.

In the final state, we assume p5ab has completely folded into native state (*Fig.27*). Since folded p5ab has 4-nt and 3-nt dangling loop at its 5' and 3' terminus, respectively, the finial distance X_{RNA} should be the distance 8 inter-residue interval: 3 for 4 nt 5' dangling loop, 2 for 3 nt 3' dangling loop, and 2 for a terminal base pair (whose distance is nearly two inter-residue interval). In PlanarFold, the distance of inter-residue interval is 5.5 Å (the equilibrium value of *Bond* potential), so we have the equation: $f_{steering}(final) = 2 \cdot k_{spring} \cdot (X(final) - X_{RNA}) = 0.046$ pN/nm · (X(final) - 8 · 5.5Å) = 5 pN. Then, X(final) = 5 pN / 0.0046 pN/Å + 8 · 5.5Å = 1131 Å.

Similarly, in the initial state, we assume p5ab has completely unfolded into native state. Since p5ab is 56 nt, we have the equation: $f_{steering}(initial) = 2 \cdot k_{spring} \cdot (X(initial) - X_{RNA}) = 0.046$ pN/nm · (X(initial) - (56-1) · 5.5Å) = 25 pN. And, X(initial) = 25 pN / 0.0046 pN/Å + 55 · 5.5Å = 5737 Å.

Now, we can determine the simulation steps: loading speeding = (*dist_final* - *dist_init*) / (*nstlim*dt).* Here, we can choose *dist_final, dist_init* is 900 Å (slight smaller than 1131 Å) and 5900 Å (slightly larger than 5737 Å), respectively, and dt is set to 0.004 ps. And the desired loading speeding is 200 nm/μs. So the simulation steps: *nstlim* = (*dist_final* - *dist_init*) / (*loading speeding*dt) = (5900* Å - *900* Å) / (200 nm/μs * 0.004 ps) = 625,000,000.

4). The acquisition frequency (or sampling frequency) refers to the rate at which the position (or other properties) of a trapped particle is measured, typically in Hertz (Hz). This parameter is crucial because it determines the temporal resolution of the experiment and affects the accuracy of force and displacement measurements. MD simulation can achieve incredibly high acquisition frequency, by setting the variable "*dumpfreq*" as small as possible. In here, we set *dumpfreq* = *5000*, which translates to an acquisition frequency of 50 kHz. Such acquisition frequency is 10 times higher than experiment acquisition frequency (generally 4-5 kHz). Such high simulation sampling frequency guarantees the more precise states visualized in FEC.

```
5890.44000  283.42167    0.37006 2222.17554
5890.40000  271.67967    0.37084 2226.72491
5890.36000  266.93096    0.37115 2228.55550
5890.32000  280.34537    0.37026 2223.32491
5890.28000  274.04184    0.37067 2225.75993
5890.24000  261.89999    0.37147 2230.46463
5890.20000  269.48975    0.37097 2227.49852
5890.16000  283.10049    0.37007 2222.19175
5890.12000  267.29640    0.37111 2228.32004
5890.08000  267.25292    0.37111 2228.32140
```

Figure 30. The example lines of SMD recording file (*dist.RST*)

The variable "*dumpfreq*" controls both the frequency of adjusting the equilibrium value of the virtual spring X(t), and the frequency of recording SMD-related data. The SMD output file DO NOT supports arbitrary filename, but adopts the uniform filename *dist.RST*. In this file, the 4 columns records the X(t) (Å), X_{RNA} (Å), $f_{steering}$ (kcal·mol^{-1}·Å^{-1}), and work (kcal·mol^{-1}), respectively.

Eventually, start the SMD simulation of p5ab following the command:

```
$ PlanarFold
```

If the unfolding simulation at the same loading condition is needed, one can simply switch the value of the final distance (*dist_final*) and initial distance (*dist_init*). Besides, the initial conformation should be already at the native state, instead of the unfolded state. Such native conformation can be acquired by performing restrained MD at first using the native secondary structure as restraint. The script of running restrained MD can be acquired from script of running normal MD and simply changing the variable ntr=1 to turn on the restraints, and manually add the secondary structure into the restraint file (*restraint_ntr.in*). Then, use the tool *extract_Frame_into_inpcrd* to extract the restrained native conformation as the initial conformation for the SMD simulation of unfolding.

It is worth mentioning that, in our previous tests, partially reducing the charge ("charge") from 2.86 to 1.0 and decreasing the strength of base pairing ("bphm_k") from 0.4 to 0.3 can better reproduce the OT experimental measurements. Such parameter fine-tuning is based on the buffer

condition difference. The parameters we originally refined is based on free energy differences measured in NMR buffer, which usually differs from OT buffer. The OT buffer for p5ab consists of 250–300 mM NaCl or KCl and 3–20 mM $MgCl_2$, while the NMR buffer generally contains 10 mM sodium phosphate and 0.01 mM EDTA.

6.2.2. Constant force mode

Generally, c.v. SMD simulations are conducted prior to c.f.SMD, to determine the ripping force range. The c.v. SMD should be conducted in low loading rates to avoid strong hysteresis, such that the zipping force and unzipping force of p5ab are close enough. Several c.f.SMD should be simulation at forces near the ripping force.

To perform c.f.SMD, simply set “ismd=1” and set the “stretch_force” to desired force. Here, we have identified that p5ab hops around 13.5 pN, so we set “stretch_force = 0.1942”, since the unit of “stretch_force” is kcal·mol^{-1}·Å^{-1}. The spring constant can be kept as 0.046 pN/nm. However, the current c.f.SMD algorithm of PlanarFold is different from how OT device achieve constant force mode. In OT experiment, the designated force is reached by adjusting the position of optical traps, restricted by the feedback frequency. In PlanarFold, the forces are precisely maintained at designated value by adjusting the equilibrium value of the spring X(t) from the equation: $f_{steering} = 2 \cdot k_{spring} \cdot (X(t) - X_{RNA})$. The feedback frequency is same to sampling frequency, which is 50 kHz (*dumpfreq* = 5000). Implementing c.f.SMD following OT experiment principle is underway.

The c.f.SMD usually requires longer sampling (>10 μs), so the simulation is also conducted in a discontinuous fashion using the following script. To further accelerate computation, we turn “confinements” on by setting dpr=1 in the controls.in and provide the native secondary structure in the confinement file (restriant_dpr.in), such that the time-consuming dynamic programming is replaced with a much faster algorithm, which only considers designated base pairs within the cutoff (bcut). Such acceleration is only possible when native secondary structure is definite.

```
#!/bin/bash

# start simulation using ssRNA.inpcrd
PlanarFold  -i controls.in  -f fparm.in -p ssRNA.prmtop  -c ssRNA.inpcrd \
              -o run001.out  -x run001.nc  -r run001.rst \
              -ntr restraint_ntr.in -dpr restraint_dpr.in
cat dist.RST > dist001.RST

# extend simulation using restart file
istart=2
iend=10
I=$istart
while [ $I -le $iend ]
do
  suffix=`printf %03d $I`
  suffix2=`printf %03d $((I-1))`
  # production run
  PlanarFold  -i controls.in  -f fparm.in -p ssRNA.prmtop  -c run$suffix2.rst \
              -o run$suffix.out  -x run$suffix.nc  -r run$suffix.rst \
              -ntr restraint_ntr.in -dpr restraint_dpr.in
  cat dist.RST > dist$suffix.RST
  I=$((I+1))
done
```

Figure 31. The contents of run.sh

User should pay attention to rename or copy the content of SMD output file ("*dist.RST*") into a new file, since each simulation will re-write "*dist.RST*".

6.3. Analysis

6.3.1. Constant velocity mode

There are typical two groups of data used in OT experimental analysis: force-extension data and extension-time data, for c.v.SMD and c.f.SMD, respectively.

To plot the unfolding and refolding FECs of p5ab, type the following commands at directory "*PlanarFold/benchmarks/mechanics/p5ab/c.v.smd*":

```
$ smd_Force_Extension  --fndist [./refold/dist.RST,./unfold/dist.RST]
                       --compression 100  --lw 0.5  --s 2
```

And to analysis the base pair information more explicitly:

```
$ map_PlanarFold_BP -o ./refold/run.out -p ./refold/ssRNA.prmtop
```

The command above gives the following results.

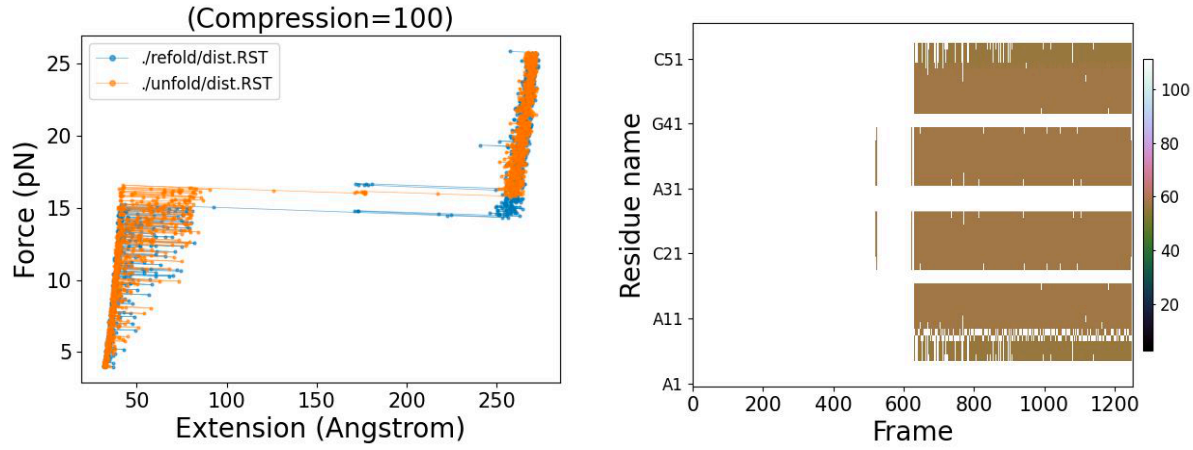


Figure 32. The force-extension curves and base pairing evolvement (refolding) of p5ab

From the FECs, we can observe that unfolding and refolding of p5ab is around 15 pN and XX pN, respectively. Such transition barely has any intermediates, indicating the formation of disruption of the entire p5ab hairpin. Around 160 Å and 75 Å, two intermediates can be spotted, which agree with experimentally verify intermediate I and I1 (*Fig.27*), respectively.

6.3.2. Constant force mode

First, several SMD output files should be merged for subsequent data processing. Following the benchmark commands, we should have generated 10 SMD output files, from dist001.RST to dist010.RST.

Now, type the following commands at directory "*PlanarFold/benchmarks/mechanics/p5ab/c.f.smd*" to merge these SMD output files:

```
$ cat dist*.RST > all_dist.RST
```

Then, we use the merged file (*all_dist.RST*) to analyze the extension-time data from c.f.SMD,:

```
$ smd_Extension_Time  --fndist all_dist.RST --compression 10
```

The command above gives the following results.

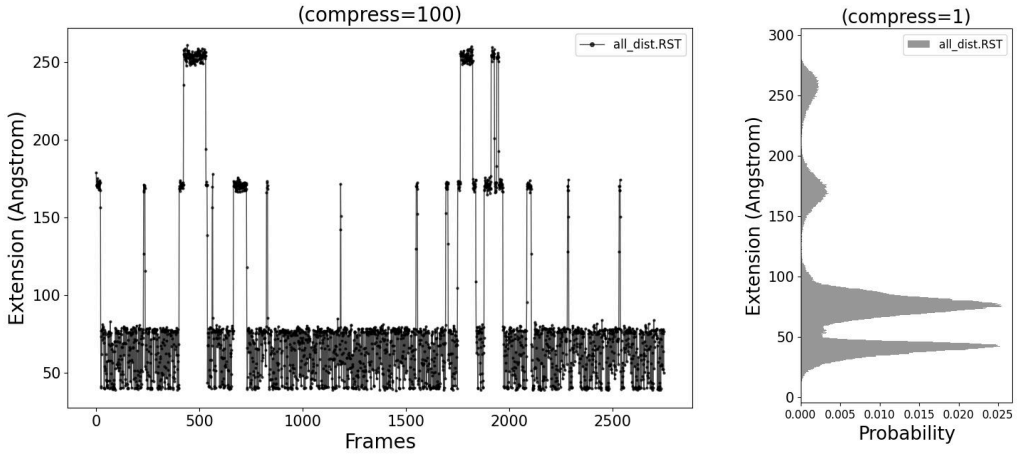


Figure 33. The extension-time data (left) and the extension distribution (right) of p5ab

From the extension-time data, one can clearly observe the folded and unfolded states, along with two intermediate states. Their relative population are determined by the force we set. If the force is set too large (> 20 pN for p5ab), the RNA stays at unfolded state. If the force is set too small (< 5 pN for p5ab), the RNA stays at folded state. Only when the force is set close to the ripping point (~ 14 pN for p5ab), the RNA can hop between unfolded and folded state. Therefore, c.v.SMD at appropriate speed must be performed first to ascertain the ripping force.

To bear in mind, if confinement is employed, all the intermediates probed by c.f.SMD can only form base pairs that are confined in the designate (usually native) secondary structure. Disarm confinement can achieve sampling of all possible conformations, at the cost of sacrificing computation efficiency.

7. Plot secondary structure using PlanarFold

PlanarFold can be used as a plotting software, since it generates coordinates that can be directly visualized on the 2D plane. Here, we designed a pipeline for users to plot designated secondary structure of RNA using PlanarFold. Compared with other plotting software, PlanarFold does not generate overlapping chains, due to van der Waals repulsion. However, plotting with PlanarFold is more computationally intensive.

First, users should place the sequence and secondary structure (in dot-bracket format) into the sequence file (*ssRNA.seq*). The related files and scripts can be found in "*PlanarFold/benchmarks/plot/*".

```
>7PKT_1
CAGCUAUAAUUUAGCGUACCUUUUGCAUUAUGGGUUAGCUACUCAAUAGACGUGAGUUAAGCAGAAAGCUUACUGAGAUUUUUUUUAAC
GUCUAUUGGACCCGAAACCGAUCGAUCUAGCGACGUGCAGGUCUGACUUUAAUCAGGACCGAACCCGUGUCUGUUGCAAGAGACUGGGAUG
ACACGUGGCUAGGGGUGAAAGGCCAAUCAAGUUCGGUAAUAGCUGGUUUUCUGCGAAAUCUAUUCAGGUAAAGAGUGUGUACUAAGACCAUA
UACCGGUAGCGCAAGCGAAUGGUAAUGGAUCUAAUACAACACAUUCAGACAAUCGGCGGUAAGGUCAAUUGUCAAGAGGAGAACAGUCCAGA
GGCGAGGCUAAGGUGCCUAACAAUUAAGGUAGAGUCUGCGUUGAACACCAGAACAUAGGCUUGGAAGCAGCCACUGUGCUUGGAAAGCGUAA
UAGCUCACUGGUCUAGCAGUCUCCGCUUAUAUAUGUCCGGCACUAUUGCCGAAGCCCGCCUAUGACAAAUUUUUUUUUUUUU

..((((.....)))).....................((((((((((....(((((.....)))))..............))))))))))((((.....((((.......(((.(((((((((((((.....)))))..))))...(((((.
((((.......))))))))).)).)))))..)))).(((((.....)))..))....))))........))))..............(((....))).((((((((.....((((((.((((.................))))))))).))....))).)).)).((((((.((.........)).))))))....
(((.....))))..(((((.((((..((((((.......((((...(((.((((.......(((((.(((..((((.......))))..)))).....((..((.....)))).)))))...)))).)).)))).....)))).....)))..)))))))).....................
```

Then, run the following scripts, one by one:

```
$ gen.sh

$ run.sh

$ plot.sh
```

The above commands can generate satisfactory planar secondary structures, in most cases.

```
#!/bin/bash

# generate topology and coordinate files
generate_PlanarFold_Inputs -s ssRNA.seq -c ssRNA.inpcrd -p ssRNA.prmtop\
                           -ntr restraint_ntr.in -dpr restraint_dpr.in \
                           --iCirc 1 --InitV 0 --vscale 0.0
```

Figure 34. The contents of gen.sh

The first script "*gen.sh*" generates the input files required for performing MD simulation. The initial conformation is set as "circular" since it is much easier to fold than linear conformation.

```
#!/bin/bash

# Step-1: Filter the secondary structure to keep only apical stems
extract_Apical_loops --seq ssRNA.seq  --seq_apical ssRNA_apical.seq

# Step-2: Generate restraint and confinement files for apical stems
generate_PlanarFold_Inputs -s ssRNA_apical.seq  -ntr restraint_ntr_apical.in -dpr restraint_dpr_apical.in

# Step-3: Pre-equilibrium to restrain the apical stems to avoid trans-isomer
PlanarFold  -i controls_prep.in  -f fparm.in -p ssRNA.prmtop  -c ssRNA.inpcrd \
            -o run_prep.out  -x run_prep.nc  -r run_prep.rst \
            -ntr restraint_ntr_apical.in -dpr restraint_dpr_apical.in


# Step-4: Production run with all restraints applied
PlanarFold  -i controls.in  -f fparm.in -p ssRNA.prmtop  -c run_prep.rst \
            -o run.out  -x run.nc  -r run.rst \
            -ntr restraint_ntr.in -dpr restraint_dpr.in
```

Figure 35. The contents of run.sh

The second script “*run.sh*” performs two stages of MD simulation to acquire the satisfying coordinates under the designated secondary structure. In the first stage, a 4-ps simulation is conducted under the restraints of only apical stems in the designated secondary structure structure, to avoid the formation of the trans-isomer. Since we employ confinements (dpr=1) and restraint (ntr=1) in both stages, the trans-isomer exclusion algorithm has not yet been implemented under this regime to save time. Such a practical solution is fast and can tackle the restrained folding of most complex RNAs. If the trans-isomer emerges (in rare cases), users can modify “*gen.sh*” to assign random initial velocities (-- InitV 1 -- vscale 0.01) to yield different outcomes. In the second stage, a 1-ns simulation is conducted under all the base pair restraints. More relaxed conformations can be obtained using longer simulation time by changing the file "controls.in". Running the secondary-structure plot generally takes tens of seconds.

```
#!/bin/bash

# plot
plot_Frames -o run.out -p ssRNA.prmtop -x run.nc \
            --frame0 25 --frame1 26 --step 25\
            --needRotate 1 -s 5 --needEnergy 1
```

Figure 36. The contents of plot.sh

The third script “*plot.sh*” visualizes the last frame (25) using the coordinates recorded in trajectory file (*run.nc*) and base pair information stored in energy output file (*run.out*).

Running the above three commands gives the following results:

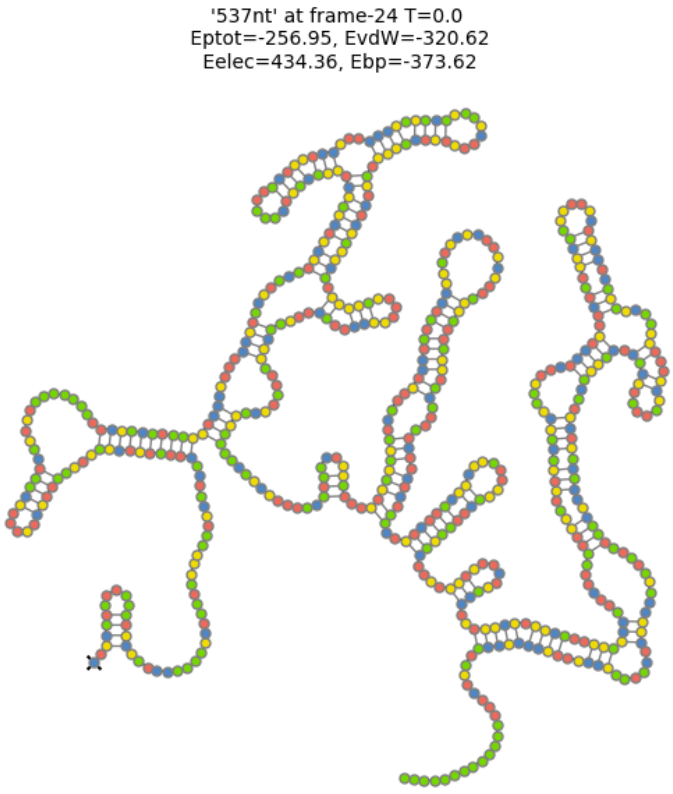


Figure 37. The secondary structure of 7PKT_1 plotted by PlanarFold

Reference

1. Han, G. & Xue, Y. Rational design of hairpin RNA excited states reveals multi-step transitions. ***Nature Communications*** 13(2022).

2. Wenter, P., Fürtig, B., Hainard, A., Schwalbe, H. & Pitsch, S. A caged uridine for the selective preparation of an RNA fold and determination of its refolding kinetics by real time NMR. ***Chembiochem*** 7, 417-420 (2006).

3. Xayaphoummine, A., Viasnoff, V., Harlepp, S. & Isambert, H. Encoding folding paths of RNA switches. ***Nucleic Acids Research*** 35, 614-622 (2007).

4. Xu, H.Z. et al. Direct Observation of Folding Energy Landscape of RNA Hairpin at Mechanical Loading Rates. ***Journal of Physical Chemistry B*** 121, 2220-2229 (2017).