SPQR 0.9.5.1
A simulation package for the prediction, refinement and simulation of RNA structures.

Simón Poblete

1 Introduction.

SPQR (SPlit and conQueR) is a coarse-grained representation of RNA [1, 2], which is implemented in the present software. The energy function can be used to score structures, predict three-dimensional structures from a given sequence, explore the conformational space and optimize structures as well.

2 Features.

As a coarse-grained model, SPQR is defined by its degrees of freedom and its energy function. As shown in Figure 1, each nucleoside is represented by an anisotropic particle, while the phosphate group is a point particle.

Thus, the base is an anisotropic object with a virtual site which stands for the sugar group. This representation allows the introduction of directional interactions and the possibility of forming stacking, canonical and non-canonical base-pairs and base-phosphate interactions. In addition, through the interactions between a nucleoside and its topologically connected phosphate groups, it can specify the state of the glycosidic bond angle and sugar pucker. The relative weight of the interactions considers the number of hydrogen bonds present in the base-pairs while it has been adjusted by fitting a set of x-ray structures for the rest of interactions.

3 To compile - Binary files.

To compile, one must run the script

```
./make_here
```

with the option `-MPI` in case one wants to add the option for multiple processors. The compilers for each case can be set in the scripts `make_mpi` and `make_no_mpi`.

The binary files generated have the prefix `MPI_` for the cases which support or need multiple cores. In addition, they have as a suffix the current version of the code.

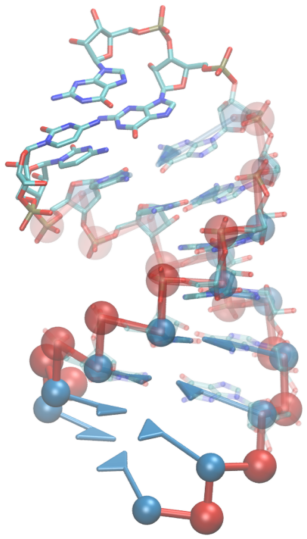

Figure 1: SPQR representation of a RNA duplex.

In order to run any of these programs one must include the file `input.spqr` in the same directory, plus other files specified in the next section. The following list summarizes the binaries

- `SPQR_MC`: Runs a simple Monte Carlo simulation at constant temperature. Requires also two empty directories for the initial conditions and the output configurations. Must be run as

  ```
  ./SPQR_MC input.spqr
  ```

- `SPQR_MCermsd`: Runs a simple Monte Carlo simulation with the option of including a harmonic restraint on the $\mathcal{E}$RMSD with respect to a target structure. Requires also two empty directories for the initial conditions and the output configurations.

- `SPQR_SA`: Runs a simulated annealing. Requires also two empty directories for the initial conditions and the output configurations.

  ```
  ./SPQR_SA input.spqr <job_index>
  ```

  where `<job_index>` is a positive integer used to label the job, in case many simulations with different initial conditions are ran. In addition, it ensures that the random seed of each run is different for each replica. Pretty useful for running job arrays.

- `SPQR_wrmSA`: Runs a simulated annealing with capped interactions. That is, whenever there is a clash or broken bond, the interactions do not collapse but instead push the system to an allowed configuration.
- `SPQR_ENERG`: A tool for calculating the energy and its contributions of a given structure. In addition, one can perform annotations and also detect possible clashes.

4 Input files.

4.1 input.spqr

The most important file is `input.spqr`, which contains the parameters for any type of run. The list is below

- `TEMPERATURE`: temperature of the system in the CG units.
- `MC_PH_XYZ` : Monte Carlo step for the translation of phosphate particles.
- `MC_NT_XYZ` : Monte Carlo step for the translation of nucleoside particles.
- `MC_STEPS` : Number of Monte Carlo trials for the simulation. Each of these steps consists of a trial move of each of the nucleotides of the system.
- `MC_TRAJ_STEPS`: Number of steps between each configuration saved in trajectory files.
- `MC_CHKP_STEPS`: Number of steps between checkpoint saving.
- `RANDOM_SEED`: Integer larger than zero. Still, two runs with the same random seed but different `mpi_id` or process id will have internally a different seed and therefore, produce different results.
- `MC_NT_ANGLE` : Monte Carlo parameters for the maximum rotations of the nucleoside, around the base and the sugar group.
- `MC_RCUT` : Cut-off for the interactions. Currently optimized.
- `VL_SKIN` : Skin of the Verlet lists.
- `ENERGS_PATH` : Path leading to the file `intrac.btb` . This file is a binary which contains the information of the whole tables and interaction parameters. Such files can be found in the same directory by default, `interactions`. In case a modification to the tables is added, one can generate the corresponding binary file again with the tools included in the same directory.

The order of the input parameters is irrelevant, and the lines can be commented by introducing the character `#` at the beginning of a line.

Additionally, there is a number of parameters needed for running a simulated annealing. All of them have the prefix `SA_` . Considering that the temperature starts with a value T_M and that $T_i = \lambda^i T_M$, for $i = 0, 1, 2, ...$, the annealing consists of a set of simulations (using the parameters aforementioned) with temperature T_i, until a minimum value T_0 is reached and the temperature is rounded to zero. In addition, if a maximum number of steps N_m is reached, the simulation will also stop.

- `SA_TINI`: Initial, or maximum temperature T_M, from where the annealing starts. Usually around 20.
- `SA_TMIN`: Minimum temperature, or where the annealing will round it to zero to stop at the next simulation.
- `SA_TFAC`: The value of λ; the prefactor with decreases the temperature.
- `SA_STEP`: Initial step for the run. This must be set to zero if the annealing is not starting from a checkpoint.
- `SA_NT`: The maximum number of annealing steps, N_m.
- `SA_PREENERG`: The energy of the previous annealing step. It must be zero if the annealing is not starting from a checkpoint.
- `SA_SFAC` : A prefactor which decreases the size of the Monte Carlo steps. This follows the procedure of Snow et al. [3], for better convergence.
- `SA_RTIMES`: The number of times the reduction of the Monte Carlo steps has been applied. It must be zero if the simulation does not start from a checkpoint.

All the parameters which are zero unless the simulation starts from a checkpoint can be set manually to specific values. Nevertheless, the same checkpoint contains the information for these values and their setup is automatic. In future versions, their direct manipulation will be removed.

4.2 pdb_inits

The directory `pdb_inits` contains the initial conditions. The files can be in `.pdb` or `.mc` binary format, which saves the configurations and more information in full precision, allowing to evaluate energies and restarting simulations without complications. Several tools are provided for converting between these formats, modify them and generate them from a sequence or other pdb (see the Tools section). The names of the initial conditions must be `init.p<XX>.<format>` . `XX` corresponds to the index of the simulation. It can also be that only the file `init.<format>` is provided, starting all the simulations from them but with different random seeds.

The SPQR pdbformat contains some particular features. Each nucleotide is composed of five particles, with atom names BASE, XVEC, YVEC, SUGR and PHOS. The sugar position is only contained for human-reading purposes, since the SPQR binaries rewrite it anyway. Apart from the atom, residue and chain indexes, it contains the coordinates as any pdb file. However, the BASE atoms can contain additional information for a simulation. Starting from the column 56, four parameters G, P, Land F can be specified.

- G: Corresponds to the glycosidic bond angle state. It can be A, H or S for anti, high-anti and syn. The syn state is not allowed in pyrimidines.
- P: Corresponds to the sugar pucker. It can be 3 or 2, for C3' endo and C2' endo states.
- L: Specifies if the glycosidic bond angle and sugar pucker states of a nucleotide will be fixed or not during a simulation. It is Aif both parameters can change, Pif only the pucker can, Gif only the glycosidic bond angle is allowed to change, and Nif both are fixed.
- F: Specifies if the position of a nucleotide, or part of it, will be allowed to move during the simulation. It is A if the whole nucleotide is movable, P if only the phosphate can move, B if the nucleoside can move and N if the whole nucleotide is frozen.

If any of the previous parameters is wrongly initialized, of absent, **the state of the nucleotide will be set by default to** A3N3, which is a nucleotide flexible and free to move, but with glycosidic bond angle in anti conformation and sugar pucker C3' endo, both fixed.

4.3 Interactions

As mentioned before, the intrac.btb binary file must be located somewhere, such that the input.spqr file knows its path. The interaction tables are written in binary format, which are available somewhere else.

5 Output files.

After a simulation is performed, several files are created. Inside the configs directory, the trajectories and the checkpoints are stored. Both of them are in binary format, which is accessible by using the analysis template in the tools directory. The trajectories are stored in the file confs.pXX.mc, where XX corresponds to the index of the simulation running. The checkpoints, on the other hand, are named chk.YY.pXX.mc, with YYdenoting the Monte Carlo step. These files can also be transformed to pdb format easily with the tool extract_traj.c, and modified with similar methods found in the same directory (See the Tools section). The final configuration is also written in pdb format in the local

directory under the name `final.pXX.pdb`. The checkpoints contain all the information of the conformation with double precision, and the parameters for the simulated annealing in case this is interrupted.

6 Annotation, energies and secondary structure.

The `SPQR_ENERG`binary allows to calculate the energy of a configuration. Regardless its format (`mc` or `pdb`), it has several options to run as

```
SPQR_ENERG  <filename>  -option
```

The option can be :

- `a`: annotation.
- `s`: secondary structure.
- `t`: total energy (with a detail of its contributions).
- `f`: the total energy.
- `b`: the backbone energy.
- `B`: the backbone energy plus the stacking energy of contiguous nucleotides.
- `n`: non-bonded stacking energy and finally.
- `w`: the base-pairing energy.

7 Tools.

The directory tools contains several binaries with their sources for the creation and manipulation of structures and conformation files. The files are listed here:

- `analysis_template.c`: Contains functions to open and access the positions and orientations of the nucleotides of a given conformation in `mc`format. The file can contain multiple time steps, from which one can select some of them for further analysis.
- `EXTRACT`: In a similar fashion, it open, reads and converts to `pdb`format a trajectory in `mc`binary format. It has to be run as

  ```
  EXTRACT <filename> NTOT INI
  ```

  where `NTOT` is the total number of snapshots to be read, discarding everything before the `NINI` configuration. Its source is `extract_trajs.c`.
- `MODIF_MC`: It allows to change the state of a glycosidic bond angle or sugar pucker, to make it fixed or allowed to change during a simulation. Its source is `modif_mc.c`.

- `assemble.py`: A python script which allows to generate a simple SPQR `pdb` file from a sequence. It must be used as

  ```
  python assemble.py (-d) <seq> > my_spqr_file.pdb
  ```

  where the `-d`option can be added in case one wants to generate a duplex.

- `pdb2spqr.py`: Transforms an all-atom `pdb` file into a SPQR `pdb` file. It classifies automatically the glycosidic bond angles and sugar puckers of each nucleotide.

- `spqr2ccc.py`: It converts a SPQR `pdb` file into a `pdb` with the positions of the C2, C4 and C6 atoms, ready to be used with the Gromacs and Plumed packages for ERMSD pulling.

- `spqr_brute_backmap.py`: A trivial and direct way of backmapping SPQR files into all-atom representations. It simply mounts an atomistic template nucleotide on top of each SPQR base. It uses specific templates depending on the glycosidic and pucker states. Further energy minimization can follow this step for later use in MD simulations.

8 An example.

A simple test corresponds to fold the GCAA tetraloop closed by a duplex, belonging to the PDB:1zih. Its initial conformation can be created using the `assemble.py` script with the sequence GGGCGCAAGCCU . In the directory `example` one can find the `input.spqr` and the initial condition in the `pdb_inits` directory. By simply setting the correct directory for the interactions in the input file and running it with `MPI_SPQR_SA`, one will generate a series of final structures from which the one with the minimum energy will have the correct contacts, as found in the native structure. Try it!

References

[1] S. Poblete, S. Bottaro and G. Bussi, *A nucleobase-centered coarse-grained representation for structure prediction of RNA motifs*, Nucleic Acids Res. 46, 1674-1683 (2018).

[2] S. Poblete, S. Bottaro and G. Bussi, *Effects and limitations of a nucleobase-driven backmapping procedure for nucleic acids using steered molecular dynamics*, Biochem. Biophys. Res. Comm. 498, 352-358 (2018).

[3] M. E. Snow, *Powerful simulated-annealing algorithm locates global minimum of protein-folding potentials from multiple starting conformations.*, J. Comput. Chem. 13, 579-584 (1991).