Tutorial: Relative Reaction Rate Estimator (*rrre*) Version 1.0 beta

Mahmoud Moradi[1,*]

[1] *Physics Department, North Carolina State University*
and
Beckman Institute, University of Illinois at Urbana-Champaign.

(Dated: April 11, 2013)

The *relative reaction rate estimator* (*rrre*) package provides several tools to estimate free energies, committor functions, and relative reaction rates of different mechanisms/pathways associated with a reaction, sampled by a nonequilibrium driven scheme such as steered MD. This tutorial provides all the details needed to set up the program to perform the desired analysis. We also use three simple examples to illustrate the workings of the code. These examples are based on examples generated by a simple standalone code as well as well as NAMD and AMBER simulation packages.

I. INTRODUCTION

Many physical/chemical transitions/reactions occur on time scales beyond what is currently accessible computationally using conventional sampling techniques such as room-temperature molecular dynamics (MD) simulations. This problem is particularly acute for biomolecular systems, whose free energy surfaces are often quite rugged and complex, with several different competing pathways and transition states. In order to enhance the sampling of rare events (*i.e.*, crossing the large free energy barriers), one may use a biasing potential. Using a time-independent bias (*e.g.*, umbrella sampling technique [1]) has an advantage of keeping the system near equilibrium; however, reaching a reliable statistical convergence could be extremely expensive computationally particularly when the free energy landscape is rugged. On the other hand, simulations generated by time-dependent biasing protocols require a nonequilibrium treatment. In the context of MD simulations using an external (time-dependent) "driving" force is a well-known scheme to enhance the sampling and estimate the free energies. Referred to as "nonequilibrium driven scheme" here, this scheme is used most prominently in methods such as steered MD [2] (SMD) in which the biasing potential is a harmonic constraint with a center in the collective variable space, moving by time linearly.

Using a time-independent bias (*e.g.*, umbrella sampling technique [1]) has an advantage of keeping the system near equilibrium; however, reaching a reliable statistical convergence could be extremely expensive computationally particularly when the free energy landscape is rugged. On the other hand, trajectories generated by time-dependent biasing protocols such as SMD require a nonequilibrium treatment. Nonequilibrium work relations such as Jarzynski's equality [3] (JE), Crooks equation [4] (CE), and Hummer-Szabo relation (HSR) provide a framework for analyzing such trajectories. If a reaction is associated with competing mechanisms (*i.e.*, distinct transition paths), the nonequilibrium work relations can be used to estimate their relative reaction rates [5].

The *relative reaction rate estimator* (*rrre*) package provides several tools to estimate free energies, committor functions, and relative reaction rates of different mechanisms/pathways associated with a reaction, sampled by a nonequilibrium driven scheme such as SMD. Specificaly, the *rrre* package can be used for the following tasks:

- Reconstructing perturbed and unperturbed free energy profiles associated with a transition from uni- or bidirectional SMD simulations using Hummor-Szabo approach [6] (Minh-Adib [7] variaton for the bidirectional case).
- Reconstructing perturbed and unperturbed free energy profiles associated with any particular subclass of the pathways generated by SMD simulations. The classification of the pathways must be done independently based on certain mechanistically relevant criteria.
- Assuming each class of pathways is associated with a particular transition tube (in the language of transition path theory [8]), *rrre* can measure the committor functions associated with each tube and estimate the relative reaction rates of different tubes. The reaction rate associated with each tube will be given relative to the reaction rate of the other tubes along with the overall probability of each tube.

*Electronic address: emad@life.illinois.edu

II. SYNTAX

The *rrre* programs takes only one argument that is the name of the parameter file in which specifies all the parametres needed for the analysis. What is needed from the simulations is loaded via the standard input stream that must include collective variable trajectories and work measurements. The standard output stream will include all the results including free energies, committor functions, and transition path probabilities.

Typical Usage: ./rrre ParameterFile < InpuFile > OutputFile

A. Parameter File

All the parameter specifiers start with a "/" character (*e.g.*, "/temperature"). Once a parameter specifier is used, it must be followed by its specified value(s) as explained below (*e.g.*, "/temperature 300.0"). The order of parameter specifiers is not important unless specified otherwise below. Anything that is not a parameter specifier and comes before, after, or in between parameter specifiers (combined with their values) is considered a comment and will not be parsed.

/temperature [*temperature*]
[*temperature*]: a real number; unit: Kelvin; default: "300.0".

/runs [*runs*]
[*runs*]: a positive integer; default: "10000"
[*runs*] specifies the number of trajectories to be loaded. If the actual number of trajectories in InputFile, PathFile (see /paths), or ProtocolFile (see /protocol) is larger than [*runs*], the additional trajectories will not be loaded. If the actual number of trajectories in InputFile, PathFile, or ProtocolFile is smaller than [*runs*], the [*runs*] parameter will be automatically set to the smallest number; thus, only trajectories with known path and protocol will be used in the analysis.

/protocol [*protocol*] $\langle$[$specifier$] [$ProtocolFile$] [$ForwardRuns$]$\rangle$
[*protocol*]: "unidirectional" or "birirectional"; default: "unidirectional"
"unidirectional": all the trajectories are assumed to use the same protocol.
"bidirectional": the trajectories are assumed to come from different (*i.e.*, forward or reverse) protocols.
[$specifier$]" "file" or "sorted" or "alternate"; default: none; required only if [*protocol*]="bidirectional"
"file": the forward and reverse simulations will be loaded in an arbitrary order, specified in the file [$ProtocolFile$] ([$ProtocolFile$]is required only if [$specifier$]="file").
"sorted": all the forward trajectories will be loaded prior to the reverse trajectories. [$ForwardRuns$] specifies the number of forward protocols ([$ForwardRuns$] is required only if [$specifier$]="sorted").
"alternate": the forward and reverse trajectories are loaded in an alternating manner.

/paths [*paths*] $\langle$[$specifier$] [$runs1$] [$runs2$] ...$\rangle$
[*paths*]: positive integer; default: "1"
[*paths*] specifies the number of classes of pathways identified. If [*paths*]="1", the relative reaction rate will not be meaningful but the free energy profile and committor function will be calculated. [$specifier$]" "file" or "sorted" or "alternate"; default: none; required only if [*paths*]¿1.
"file": the trajectories will be loaded in an arbitrary order whose classes are specified in the file [$PathFile$] ([$PathFile$] is required only if [$specifier$]="file").
"sorted": if the trajectories are sorted based on their classes, only the number of trajectories that belong to each class need to be specified (*i.e.*, [$runs1$] [$runs2$] ...). The number of trajectories of the last class will be determined based on the number of remaining trajectories and not the number specified here so only [*paths*]-1 numbers need to be specified.
"separate": assumes the trajectories are sorted based on their classes (similar to "sorted" option) but treats forward and reverse trajectories independently (if [*protocol*]="bidirectional"). Two sets of numbers must be be specified: [*paths*]-1 numbers for forward and [*paths*]-1 numbers for reverse trajectories (*i.e.*, [$ForwardRuns1$] [$ForwardRuns2$] ... [$ReverseRuns1$] [$ReverseRuns2$] ...). If [$specifier$]="separate", /paths must not come prior to /protocol.

/metric [*metric_1*] [*metric_2*] ...
[*metric_i*]: a real number; default: "1" (Eucledean metric).
[*metric_i*] is an estimate $|d\phi_i/d\lambda|^2$ factor in which $\phi|$ is the centerline of the pathway i and λ is the collective variable.

/initial [*initial*]
[*initial*]: a real number; default: "0"
[*initial*] specifies the initial value of the colvar.

/final [*final*]
[*final*]: a real number; default: "1"
[*final*] specifies the final/target value of the colvar.

/bin [*bin*]
[*bin*]: a real number; default: "0.1"
[*bin*] specifies the bin size, used to build the free energy and other functions in the colvar space.

/kernel [*kernel*]
[*kernel*]: a real number; default: "0"
[*kernel*] specifies the kenel width, used to reconstruct the unperturbed free energy in the colvar space. A Gaussian Kernel will be used for any [*kernel*] > 0 while [*kernel*]="0" will replace the kernel estimatation method by a simple histogram based algorithm using [*bin*] for the histograms.

/snapshots [*SnapShots*]
[*SnapShots*]: a positive integer number; default: "10"
[*SnapShots*] specifies the number of snapshots from each run to be loaded. This number has to the same for all trajectories and equal to [*SnapShots*], otherwise the results will be incorrect although you may not be notified of any error.

/harmonic [*harmonic*]
[*harmonic*]: a real number; unit:(kcal/mol)/X^2 (X: colvar unit); default: "1"
[*harmonic*] specifies the force constant, used in the simulations.

/convergence: [*convergence*]
[*convergence*]: a real number; unit: kcal/mol; default: "0.1"
[*convergence*] specifies the convergence precision used as a criterion for the self-consistent solution of free energies.

/iteration [*iteration*]
[*iteration*]: a positive integer number; default: "100"
[*iteration*] specifies the maximum number of iterations used as a secondary criterion for the self-consistent solution of free energies.

/report [*report*]
[*report*]: "0" or a positive integer number; default: "0"
Results will be reported at the end of final iteration and every [*report*] iterations. If [*report*]=0, only the final results will be reported.

/column_x [*column_x*]
[*column_x*]: a positive integer number; default: "1"
[*column_x*] specifies the column corresponding to the colvar value in the standard input stream.

/column_w [*column_w*]
[*column_w*]: a positive integer number; default: "2"
[*column_w*] specifies the column corresponding to the work value in the standard input stream.

/adjust [*adjust*]
[*adjust*]: "forward" or "reverse" or "average" or "optimized"; default: "forward"
[*adjust*] specifies the adjustment method from the options above. The "forward" option shifts the free energy profiles of different classes to match their forward probabilities (*i.e.*, $F_i(x_0) = -log(\pi_i^F)/\beta$ in which x_0=[*initial*]) while the "reverse" option does this for the reverse probabilities. The average option averages over the required forward and reverse shifts (*i.e.*, $F_i(x_0) + F_i(x_1) = -(\log(\pi_i^F) + \log(pi_i^R))/\beta$). The "optimized" option is not currently implemented. The choice of the adjustment method is only relevent if [*protocol*]="bidirectional".

B. Input File

rrre reads the standard input stream (a file or a pipeline) to import the trajectories. All the lines starting with "#" character will be ignored. The [*column_x*] and [*column_w*] specify the columns corresponding to the colvar and work values. The rest of the columns will be ignored. The columns can be separated by any number of space or tab characters. The number of snapshots from each trajectory must be exactly [*SnapShots*], otherwise the results will be incorrect although you may not be notified of any error. [*runs*] specifies the number of trajectories; however, If the actual number of trajectories coming from the standard input stream is larger than [*runs*], the additional trajectories will not be loaded. Also if the number of data coming from files [*PathFile*] or [*ProtocolFile*] is smaller than the the actual number of trajectories coming from the standard input stream, the additional trajectories will not be loaded.

C. Output File

rrre writes the results of the analyses to the standard output stream (a file or a pipeline). Lines starting with "#" character include human readable information regarding the parameters used, the trajectories loaded (*e.g.*, the number of trajectories associated with each class or protocol), and final driven and equilibrium probabilities estimated. The remaining lines start with the word "PFE" or "UFE" (perturbed or unperturbed free energy). These include free energy estimates reported every [*report*] iteration while solving the equations self-consistently (if [*report*] $\neq 0$) as well as the final free energies (shifted using the [*adjust*] method) and committor functions at the end of the self-consistent algorithm. The results reported during the self-consistent solution start with "PFE_" or "UFE_" followed by the current iteration number as the first column while the first field in the final results for perturbed and unperturbed free energies only include "PFE" and "UFE" words. The second column is the path identification number: "0" (for all paths combined) or a positive integer between 1 and [*paths*] for each particular path (or class of paths). One can easily extract the perturbed or unperturbed free energies associated with each path (or all paths combined) from the output using a simple script. For instance "grep 'UFE 0'" extracts the unperturbed free energy profile of all paths combined.

III. EXAMPLES

In order to illustrate the workings of the *rrre* package, here we provide several simple examples including (i) a one-dimentional toy model, (ii) a proline dipeptide, and (iii) a dimeric proline peptide.

A. Toy Model

Here we consider a one-dimentional potential-free overdamped Langevin equation with a $[0, 1)$ periodic boundary conditions with a drag coefficient ζ, thermodynamic parameter β, and diffusion constant $D = (\beta\zeta)^{-1/2}$:

$$\dot{x} = \sqrt{2D}\eta,\ \ \langle \eta(t)\eta(t') \rangle = \delta(t - t'). \tag{3.1}$$

If the system starts at $x_0 = l$, we define π_1 (π_0) to be the probability that the system reaches the boundary through $x = 1$ ($x = 0$) rather than $x = 0$ ($x = 1$). The problem is finding π_0 and pi_1 or π_0/π_1 (note that $\pi_0 + \pi_1 = 1$). This problem is reminiscent of a circular random walk with the following exact answers:

$$\pi_1 = l,\ \ \pi_0 = 1 - l. \tag{3.2}$$

The task here is to reproduce these (already known) answers from nonequilibrium simulations. Note that in the driven trajectories π_i^{dr} are not necessarily the same as pi_i^{eq} above and work measurements are needed to estimate the correct equilibrium transition rates.

We define an order parameter:

$$\alpha(x) = \begin{cases} 1 - x/l & \text{if } x \leq l \\ (x - l)/(1 - l) & \text{if } x > l. \end{cases} \tag{3.3}$$

Clearly, α increases from 0 to 1 as the transition progresses from x_0 to the boundary through either pathway. In order to push the system along these pathways, we use a time-dependent external potential in the form of a moving harmonic constraint:

$$V(\alpha, t) = \frac{1}{2} k \left(\alpha - \frac{t}{T} \right)^2 , \tag{3.4}$$

in which $0 \leq t \leq T$ and k is a spring constant. The system starts from $x = l$, but we let the system equilibrate around this point by applying $V(\alpha, 0)$ for a time T_{eq}. At $t = 0$, we start pulling and let the transition take place through one of the two possible mechanisms. We also need to use the right metric associated with the α space to measure $|d\phi_i/d\alpha|^2$. Note that $\phi_i(\alpha)$ is equal to x here for both $i = 0, 1$. Therefore the $|d\phi_i/d\alpha|^2$ would be the same as $|\frac{dx}{d\alpha}|^2$ that is l^2 and $(1-l)^2$ for $i = 0$ and 1, respectively. These numbers can be easily passed to the *rrre* via the "/metric" identifier in the parameter file. We also have a code in which generates sample trajectories based on the algorithm described above.

B. Proline Dipeptide

In order to test the *rrre* package in a more realistic example (while keeping the system simple enough to allow enough sampling with a very modest computational cost) we use a bidirectional SMD on a proline dipeptide whose prolyl bond is pushed along $\cos\omega$ (ω is the backbone dihedral angle) between cis ($\cos\omega = 1$) and trans ($\cos\omega = -1$) isomers. There are two possible pathways since the intermediate $\cos\omega = 0$ is associate with two states: syn $\omega = 90^\circ$ and anti $\omega = -90^\circ$. We use NAMD 2.8 simulation package along with a short tcl script in which implements the SMD algorithm and $\cos\omega$ collective variables. The *rrre* package not only reconstructs the free energy profile along $\cos\omega$ but it also estimates the relative reaction rates between the two possible mechanisms (*i.e.*, syn and anti intermediates).

C. Dimeric Proline Peptide

Here we show how *rrre* package can be used to analyze a typical set of SMD trajectories generated by AMBER simulation package. The example chosen here is a dimeric proline peptide pushed along a collective variable $\Omega = \cos\omega_N + \cos\omega_C$ (ω_N and ω_C are associated with the N- and C-terminal prolyl dihedral angles) between Cis-Cis ($\Omega = 2$) and Trans-Trans ($\Omega = -2$) isomers. There are two possible pathways since the intermediate $\Omega = 0$ is associate with two states: Cis-Trans and Trans-Cis. We use AMBER 11 simulation package and its built-in SMD ("ncsu_smd" module) with the collective variable "COS_OF_DIHEDRAL". After running several forward and backward simulations, the *rrre* package will be able to reconstruct the free energy profile along Ω and estimates the relative reaction rates and committor functions.

---

[1] Torrie and Valleau, Journal of Computational Physics **23**, 187 (1977).
[2] S. Izrailev, S. Stepaniants, M. Balsera, Y. Oono, and K. Schulten, **72**, 1568 (1997).
[3] C. Jarzynski, Phys. Rev. Lett. **78**, 2690 (1997).
[4] G. E. Crooks, Phys. Rev. E **61**, 2361 (2000).
[5] M. Moradi, C. Sagui, and C. Roland, Chem. Phys. Lett. **518**, 109 (2011).
[6] G. Hummer and A. Szabo, Proc. Natl. Aca. Sci. USA **98**, 3658 (2001).
[7] D. D. L. Minh and A. B. Adib, Phys. Rev. Lett. **100**, 180602 (2008).
[8] W. E and E. Vanden-Eijnden, Annu. Rev. Phys. Chem. **61**, 391 (2010).