Tangent Space Least Adaptive Clustering

James Buenfil, Samson Koelle, Marina Meila

Modeling time-evolution of Molecular Systems

- In Molecular Dynamics, we want to model (for example) the configuration of a collection of atoms that comprise a molecule as it changes in time under atomic forces
- A protein folding is one such example of a molecule of interest
- We can model such systems as stochastic dynamical systems, for example the Langevin Equation:

$$\frac{d}{dt}x(t) = \frac{1}{\gamma}\nabla E(x(t)) + \eta(t)\sqrt{Tk_B\gamma}$$

Dynamical Systems

- A dynamical system is a system of differential equations which describes the behavior of a physical system
- We are given the equations which describe such a system, but in this context, there is usually no closed form solution $x(t)$ and we must resort to simulating them.
- Simulating means producing a discrete sequence of pairs (time,state of system) $(t_1, x_1), \ldots (t_N, x_N)$ which we hope approximates $x(t)$, this sequence is called a trajectory

Timestepping to produce trajectories

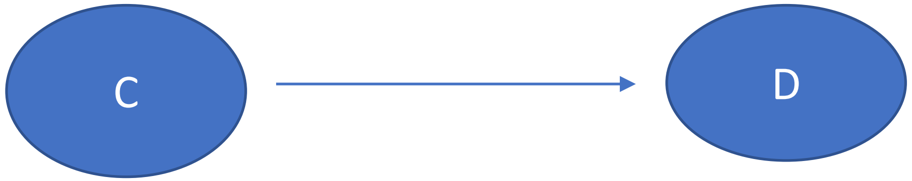
- Given our differential equation which models the dynamics, how we can produce trajectories from it?
- We can timestep them!
- Timestepping means we start from an initial point, x_0 at time $t_0$and then use the differential equation approximately in order to produce a new point (t_1, x_1). Then we repeat this process starting from (t_2, x_2) to produce (t_3, x_3) and so on.

Single-trajectory timestepping

- On a high level, we can think of the process which produces (t_1, x_1) from (t_0, x_0) as a function $A(t, x)$ into which we plug in (t_0, x_0) and are given $A(t_0, x_0)$ = (t_1, x_1).
- Iterating A repeatedly to produce a trajectory is called single-trajectory timestepping
- In this problem, A actually contains randomness, so $A(x)$ is a random variable (sampling from a stochastic dynamical system)

Problems with single-trajectory timestepping

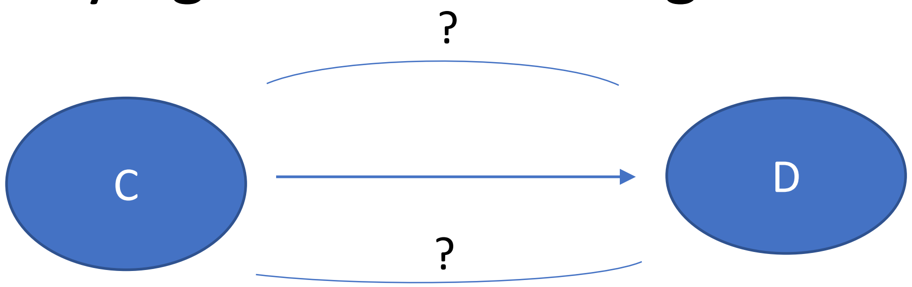
- The problem with single-trajectory timestepping is that computational bottlenecks can occur, where the state of the system will take a long time to pass through to other regions of interest.



- These bottlenecks are so costly that in Molecular Dynamics sometimes we are not interested in the sequence of pairs $(t_0, x_0), (t_1, x_1)$…, we are happy to only produce a sequence of states x_0, x_1…,

What we want: a valid trajectory

- We want to understand how the system will transition from state C to state D, without worrying about how long it will take



- How can we deal with these bottlenecks when we want to produce a valid trajectory?

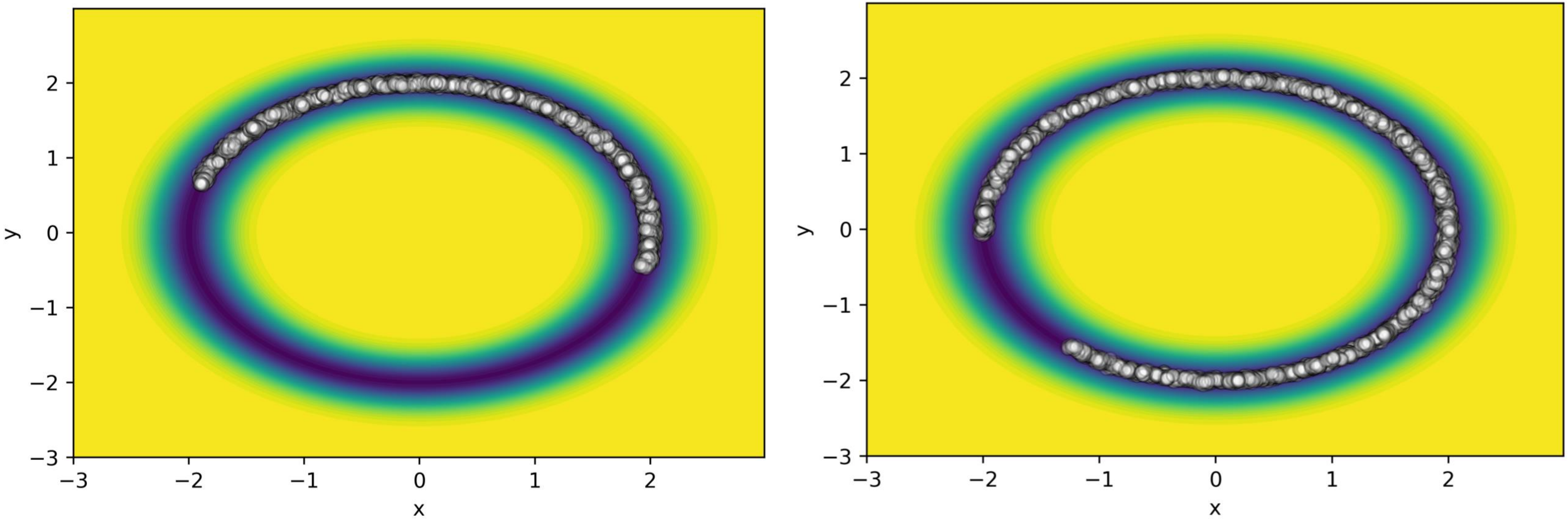
Solution: Cheating just a little

- Throughout this process what we want is to simulate a true trajectory which could arise from the molecular system
- Rather than repeatedly using $A(x_0) \ = x_1, x_2 = \ A(x_1)$ etc., in the process of producing a trajectory we instead will use any previous state seen so far.
- For example, if we are simulating and we currently have the sequence of states (x_0, x_1 , x_2 , x_3), then to produce x_4, rather than using x_4 = $A(x_3)$, we could use x_4 = $A(x_1)$, or x_4 = $A(x_2)$
- We only use previously seen states so that our trajectory is still representative of the physical process we're modeling

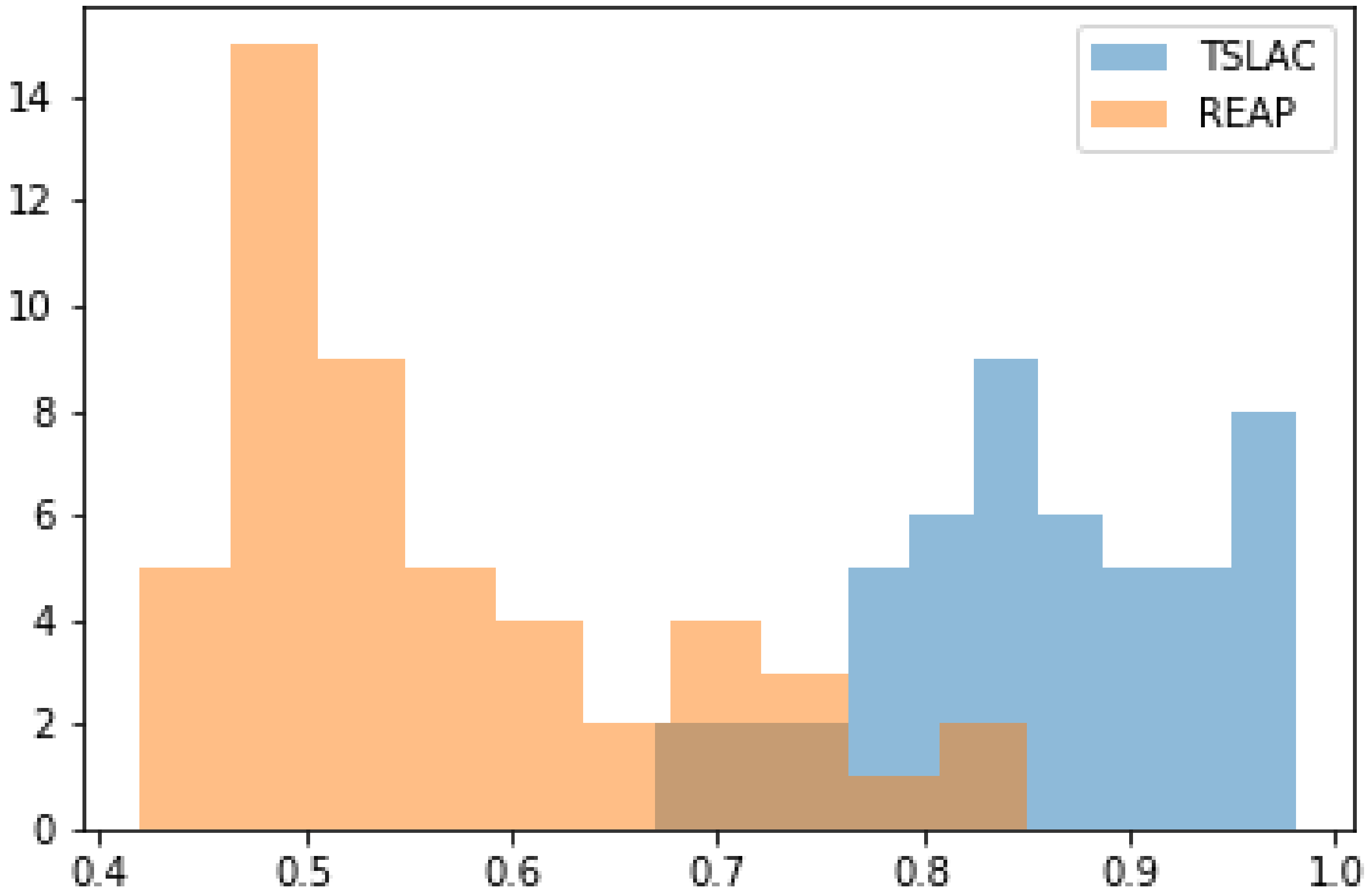
Can we cheat more efficiently?

- Now that we are choosing points to timestep from, for the purpose of exploration, some strategies will produce a trajectory faster than others
- This project was about building on an existing algorithm’s approach to choosing which points to sample from, that approach is called REAP: REinforcement learning based Adaptive sampling.
- Our approach is called Tangent Space Least Adaptive Clustering

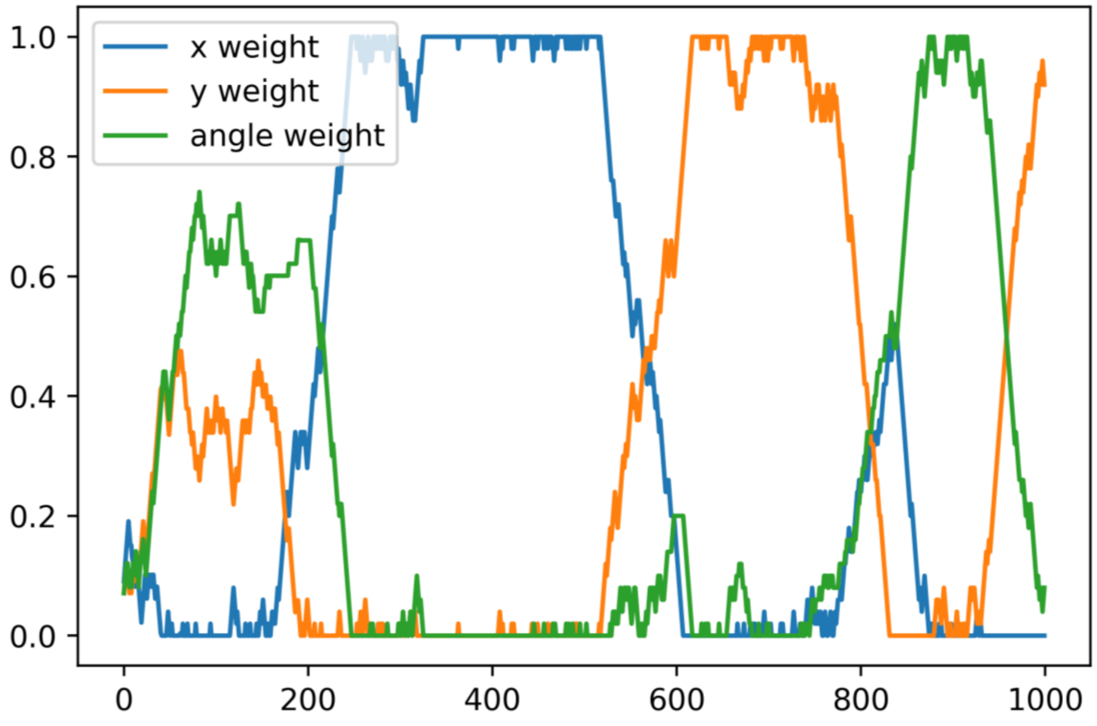
Typical trajectories after 700 iterations of each algorithm (REAP on left, TSLAC on right)

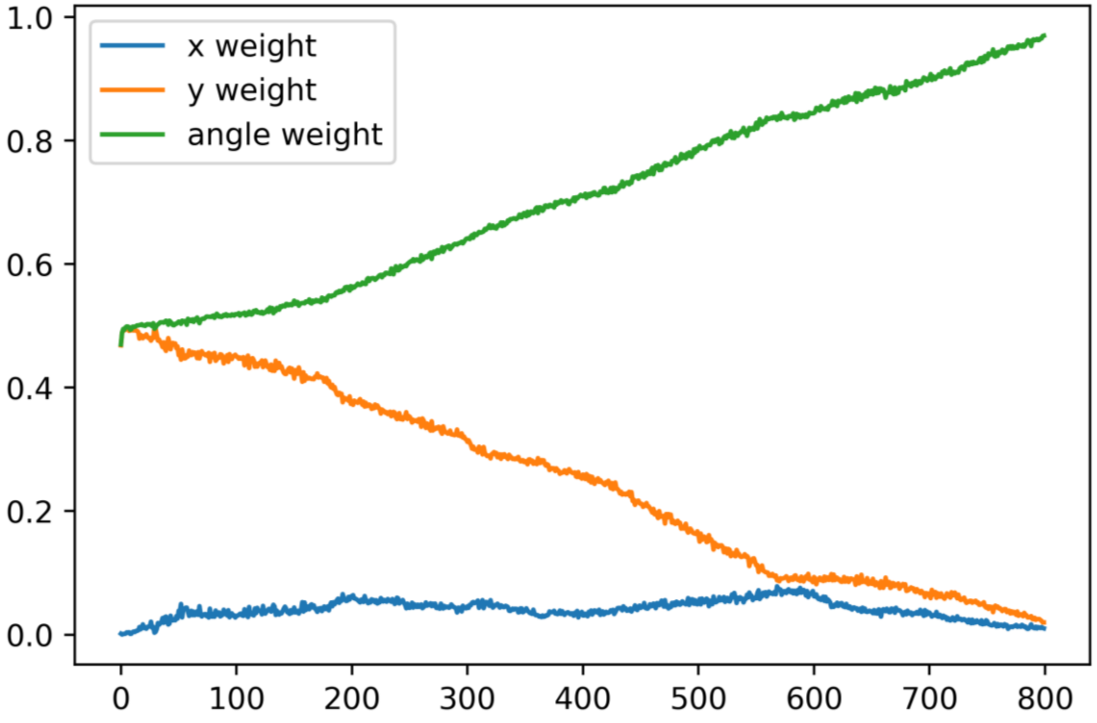


REAP and TSLC Counts of Percentage of Circle Explored after 700 iterations



Typical weights for collective variables chosen by algorithms (REAP on left, TSLAC on right)





Thanks for listening!