1 Rare-event sampling with the Giardinà-Kurchan-Lecomte-Tailleur algorithm

The general idea of the Giardinà-Kurchan-Lecomte-Tailleur algorithm **?** is to propagate an ensemble of trajectories forwards in time, iteratively cloning or killing them according to a metric of 'goodness'. This way, the ensemble will gradually get 'better' at sampling a particular rare event of interest. What follows is a detailed description of the algorithm.

Let us assume we are interested in a stochastic system $\mathrm{d}X = f(X,t)\mathrm{d}t + \sigma(X,t)\mathrm{d}B(t)$ with $X \in \mathbb{R}^d$, and an observable $O(X(t)) \in \mathbb{R}$. We can then identify a region $\mathcal{A} \subset \mathbb{R}^d$ of the phase space that is not often visited by the system and has particular values of the observable O. We then define a score function $V(\{X\},t)$ that will work as a proxy for the probability of trajectory $\{X\}$ reaching $\mathcal{A}$ **?**.

In our case, the observable is the AMOC strength itself $O = \mathrm{AMOC}$, possibly time-averaged. For simplicity here, we will use as score function the observable itself, $V(X,t) = -\mathrm{AMOC}(X(t))$, which is a quite common choice in the literature (see for instance Wouters and Bouchet [2016]; Ragone et al. [2018]). This choice makes the assumption of persistence to approximate probabilities, namely: "the AMOC is more likely to collapse in the future if it is already weak right now".

After we have defined the score function, we initialize N ensemble members with trajectories $\{X_j^{(0)}\}$, $j = 1,\dots,N$, each ending at time $t_j^{(0)}$. For simplicity and without loss of generality, we can assume $t_j^{(0)} = t_0 = 0$, and, since the dynamics is Markovian, we can ignore everything that happens before t_0. Finally, to complete the initialization of our ensemble we compute the initial scores $V_j^{(0)} = V(\{X_j^{(0)}\})$, and assign to each trajectory an unbiasing weight $\pi_j^{(0)} = 1$. This weight will track the likelihood to observe trajectory j when running the climate model with no REA, and it will be crucial to have access to the unbiased probabilities of events sampled by the biased trajectories.

Once we have our initial ensemble, we need to select a resampling time τ and a selection strength k. Then, for each iteration $i = 1,\dots,I$, we will perform the following steps:

1. Extend each trajectory integrating forward in time for one resampling step τ, obtaining $\{\tilde{X}_j^{(i)}\}$, $j = 1,\dots,N$ that run from t_0 to $t_i = i\tau$. Then compute for each trajectory its score function at the end of the resampling step $\tilde{V}_j^{(i)} = V(\{\tilde{X}_j^i\}, t_i)$.

2. Compute for each trajectory a weight according to the improvement of its score in the last resampling step $\tilde{w}_j^{(i)} = \exp\left(k\left(\tilde{V}_j^{(i)} - V_j^{(i-1)}\right)\right)$, and then normalize the weights so that they sum up to N:

$$w_j^{(i)} = \frac{1}{Z^{(i)}}\tilde{w}_j^{(i)}, \quad Z^{(i)} = \frac{1}{N}\sum_{j=1}^{N}\tilde{w}_j^{(i)}$$

3. Assign to each trajectory a random number of clones $m_j^{(i)}$ with expectation $w_j^{(i)}$. This is achieved by drawing N random numbers $u_j^{(i)} \sim \mathcal{U}(0,1)$, uniformly distributed between 0 and 1, and then setting $m_j^{(i)} = \lfloor w_j^{(i)} + u_j^{(i)} \rfloor$. Trajectories with $m_j^{(i)} = 0$ will be discarded. In case $\Delta N^{(i)} = \sum_{j=1}^{N} m_j^{(i)} - N \neq 0$, $|\Delta N^{(i)}|$ additional trajectories are either cloned or killed to ensure that the ensemble remains of size N.

4. Update the trajectories in the ensemble by creating the parent mapping function $p^{(i)}(j)$, which links each resampled trajectory j to the one it was cloned from. Then,

set

$$\{X_j^{(i)}\} = \{\tilde{X}^{(i)}_{p^{(i)}(j)}\}, \quad V_j^{(i)} = \tilde{V}^{(i)}_{p^{(i)}(j)} \quad \text{and} \quad \pi_j^{(i)} = \frac{\pi^{(i-1)}_{p^{(i)}(j)}}{w^{(i)}_{p^{(i)}(j)}}$$

In the original paper by Giardina et al. **?**, the weights in point 2 are computed not as the improvement of the score function but as the score function itself at the end of the resampling step. However, in **?**, the authors suggest using the improvement of the score function.

We implemented the algorithm both for a fully deterministic ocean-only Veros configuration, as well as for a configuration with a temperature noise component. In the deterministic case, when cloning trajectories (step 4), we add a small random perturbation to the temperature T and salinity S fields, which allows them to diverge. The perturbation is defined as

$$\begin{aligned} T &\mapsto T + T\epsilon_T\eta \\ S &\mapsto S + \epsilon_S\eta, \end{aligned} \tag{1}$$

where η is sampled from a normal distribution with mean 0 and variance 1, independently for each field and grid point. The noise amplitudes are chosen to be $\epsilon_T = 0.001$ and $\epsilon_S = 0.002\ \mathrm{g\,kg^{-1}}$. Typical salinity values are around $34.5\ \mathrm{g\,kg^{-1}}$ with fluctuations of the order of $0.2\ \mathrm{g\,kg^{-1}}$, while temperature values have a mean of 5 deg C with a standard deviation of the order of 10 deg C, though the temperature distribution is strongly skewed due to the heating from the top. Thus, in Eq. 1 we employ a perturbation that is multiplicative in temperature, such that the deep ocean temperatures that are close to 0 degrees are perturbed less than the warm surface ocean.

Now, once we reach the final iteration of the algorithm, we can use the unbiasing weights $\pi_j^{(I)}$ to compute the proper average of any observable $U(\{X\})$ over our biased ensemble:

$$\bar{U}_N = \frac{1}{N}\sum_{j=1}^{N} \pi_j^{(I)} U(\{X_j^{(I)}\}) \sim_{N\to\infty} \mathbb{E}\left[U(\{X\})\right], \tag{2}$$

where $\mathbb{E}$ is the expectation over the stationary measure. Since the trajectories are not independent, the central limit theorem doesn't apply, and thus the typical error can be larger than $1/\sqrt{N}$.

Then, the probability that the AMOC is weaker than a given threshold a is simply given by

$$\mathbb{P}(O(X(T)) \le a) \equiv \mathbb{E}\left(\mathbb{I}_{O(X(T))\le a}\right) \approx \frac{1}{N}\sum_{j=1}^{n} \pi_j^{(I)} \mathbb{I}_{O(X_j^{(I)}(T))\le a}. \tag{3}$$

In the main text, we give results for a chosen as the median of the current distribution of the ensemble, as this uses most of the ensemble members and thus has low statistical uncertainty. Given ordered realizations in the ensemble at iteration i, this yields the unbiased probability $p_{1/2}^{(i)}$ of being below the median $a_{1/2}^{(i)}$:

$$p_{1/2}^{(i)} = \mathbb{P}(O(X(i\tau)) < a_{1/2}^{(i)}) = \frac{1}{N}\sum_{l=1}^{N/2} \pi_{j_l}^{(i)}, \quad O(X_{j_1}^{(i)}) \le \ldots \le O(X_{j_l}^{(i)}) \le \ldots \le O(X_{j_N}^{(i)}). \tag{4}$$

The algorithm has four 'hyperparameters': the ensemble size N, the resampling time τ, the selection strength k and the score function V.

As already discussed, we chose to use as score function the amoc strength itself, and in particular its 5-year average for the deterministic model and its yearly average for the model with temperature noise.

The resampling time should be of the order of the Lyapunov time **?**. Long enough to allow clones of the same ensemble member to separate sufficiently, but not so long that they relax back to the attractor. Namely, we don't want segments of trajectories to completely lose memory of their initial condition. By analyzing the autocorrelation function of the AMOC strength we decided to use $\tau = 50$ yr for the deterministic model and $\tau = 1$ yr for the model with atmospheric noise.

Concerning the selection strength k, it should be chosen based on how extreme the event of interest is: the more extreme the event, the higher the selection strength. **?** suggests an empirical way to get the right order of magnitude for k: if over the integration period $T = I\tau$ the AMOC strength fluctuates with a standard deviation σ, then to push the system n standard deviations from the mean, we should set $k \sim n/\sigma$. In our case this gives a suggested k of the order of 10 Sv^{-1}.

Finally, the choice of the ensemble size is a matter of compromise between accuracy and computational cost, and in this work we chose to use $N = 50$ trajectories.

The practical implementation of the GKTL algorithm and its application to the Veros model is available at `https://github.com/AlessandroLovo/REA-Veros`.

References

Ragone, F., Wouters, J., and Bouchet, F. (2018). Computation of extreme heat waves in climate models using a large deviation algorithm. *Proceedings of the National Academy of Sciences*, 115(1):24–29.

Wouters, J. and Bouchet, F. (2016). Rare event computation in deterministic chaotic systems using genealogical particle analysis. *Journal of Physics A: Mathematical and Theoretical*, 49(37):374002.