TARP Code Explanation

Raghav Venkataramanan

September 2025

TARP (Tests of Accuracy with Random Points) is a method for evaluating the accuracy of posterior estimators (especially generative models) that only relies on samples. The method draws samples from the generative model, chooses a reference sample, then measures the "distance" between each posterior sample and both the reference sample as well as the true sample. Then, the coverage probability is computed based on these distances to determine how often a credible region contains the true sample. In other words, coverage probability is used to calibrate the uncertainty of the model, and by running the TARP test for a given model, its hyperparameters can be tuned so that the uncertainty is what we expect and that model bias is under control. As such, by relying only on the samples generated by the diffusion model, TARP allows for rigorous accuracy testing and calibration for generative models.

To run the code, either run TARP_test.py or run the batch script TARP_test.sh to launch a non-interactive run.

TARP code methodology:

1. Sample prior maps from the model using DES-Y3 simulations. These will be considered the "true" samples. They are referenced as "simulation samples" in the code.
2. Create noisy data for each true sample. Each data.h5 file will be used to sample a set of posterior and reference samples corresponding to each true sample.
3. Sample posterior and prior maps from the true maps. The prior maps in this case will be considered the "reference samples" as they are called in the code.
4. For each reference sample, calculate the "distances" between said reference sample and every simulation and posterior sample. This is done to create more data points (number of simulation samples * number of reference samples) from which to calculate the coverage probability. The distance is defined as the normalized difference in κ values between two samples x and y: $\sum_{pixels}(\kappa_x - \kappa_y)/(\kappa_{max} - \kappa_{min})$. The distances are appended to a numpy array.

5. For each distance result, calculate the "closer fraction" or "f-closer" values. These are effectively the percentage of reference-posterior distances that are less than the true-posterior distance. These values will be used to calculate the coverage probability. At this point in the code, all distance and f-closer results are saved to a .csv file.

6. Calculate the coverage probability based on the choices of region centers, labeled "alpha" in the code. After, probability plots are made and saved.

Other notes:

- Currently, the script calls other scripts (e.g., sample.py) as subprocesses in order enact these steps. This allows the other files to be updated without needing to significantly change the TARP code.
- The samples and results are saved in unique subfolders (based on choice of hyperparameters) to allow multiple runs in parallel without having any files overwrite each other.
- In the config file, the number of simulation, posterior, and reference samples generated can be set. In the current iteration of the code, these values must be divisible by the batch size.