Assignment 1: History Driven Walk

Jelle Hendriks
Boston University
jelleh@bu.edu

Abstract

We designed a random walk type MCMC sampler which changes in variance based on its mean variance and pushes away from the sample mean to try to create a method that explores well. The method was generally worse than Hamiltonian Monte Carlo in our tests but could outperform a regular random walk dependent on the parameters. It had a significantly worse acceptance rate than either comparison method but would easily explore without much parameter changes, making it quite robust to changes. The way the walk works with a push away from the mean and an additional adapting variance, it would likely work best on distributions with a wide top and close to normal farther away. Though not better than HMC it could be used as a method rarely if HMC could not be used.

1 Introduction

The method creates a push away from the current sample mean by a multiplier named drift strength. It also changes the variance based on the sample variance. It was designed this way as a modification on the baseline random walk but trying to push it out of the sample mean to let it explore more easily. The changing variance was added to allow one to not have to work with the step size as much and increase the robustness of the method to just work. It is also simpler than adaptive metropolis method for how to work out the variance and it allows one to change the amount that it can vary from the starting value from zero to one.

2 Method

It is mostly a regular random walk but it samples normally with a different mean and standard deviation. The mean of current position plus a push away based on the sample mean times some drift strength multiplier is its sampled mean. The standard deviation is some starter variance plus the diagonal of the variance of the accepted samples times some multiplier. Afterwards the log proposal densities are changed to account for these to not cause runaway effects.

3 Experiments

The method generally was exploring better than a regular random walk but worse than a Hamiltonian Monte Carlo method but with a very low acceptance probability which generally is either lower or the same as a well mixed random walk and seems best around 10%.

Now for the trace plots to indicate how well our method mixes which generally mixes pretty well. It often contains some indicators that display that it is not just random noise. This only shows up in the Rosenbrock distribution, so we will only show that distribution. Additionally for our effective sample size for this it seems to be similar to slightly less of what HMC outputs, but it also seems that ArviZ effective sample size does not entirely work correctly as the values are notably too small to take seriously for every method even on just a normal gaussian it would at best get a 7% effective

Preprint.

sampling rate with the comparison random walk having only a 2% effective sampling rate. On both the distributions it only ended as a 1-2% effective sample size compared to the number of samples.

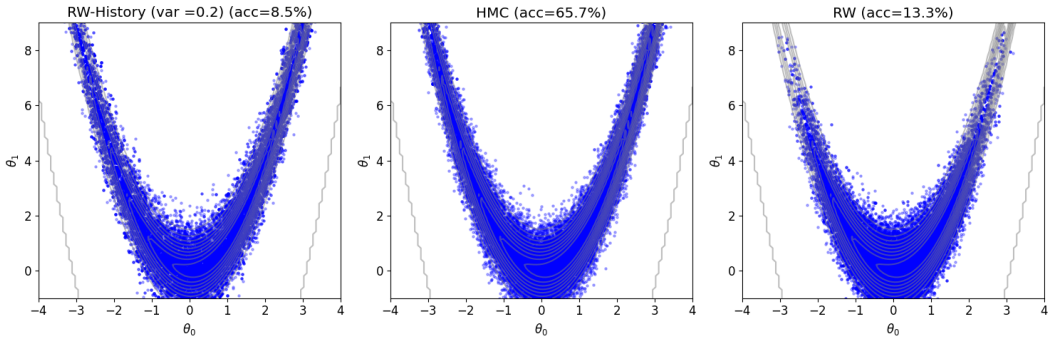

Figure 1: Rosenbrock probability distribution sample comparison of our method versus a regular HMC and Random Walk

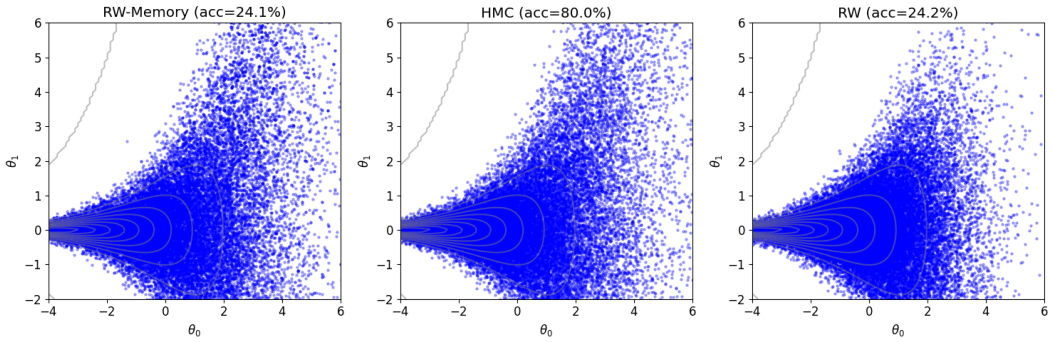

Figure 2: A Neal's Funnel probability distribution sample comparison of our method versus a regular HMC and Random Walk

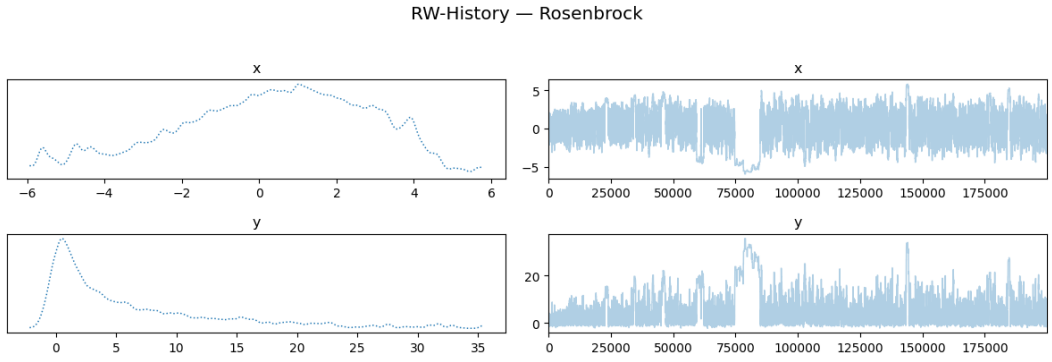

Figure 3: The Rosenbrock sample's trace plot after 200 000 samples.

4 Discussion

The method is an improvement over random walk in the distributions we tried and so might work as a general starter point if HMC was not doable for some reason. The method definitely works best on things close to regular Gaussians but is still generally somewhat effective on others. It seems to do better in places where everything is more spread out instead of a concentration within one area but can struggle to explore the shapes like the Rosenbrock sample. We would like to try this method on

other probability distributions that are not as concentrated in particular locations to observe if the method could work as an actual improvement over HMC in some areas. We could also try to see if there could be some other improvements that could be made to the adaptive aspects so that a user would not need to work with the parameters as much to get a good result.

5 AI Collaboration

For AI Use, within my coding software there was an automatic code completion from GitHub Copilot installed into Visual Studio Code. Additionally, we used the chatting feature with Claude to generate ideas and write the functions to create the method. We came in with some idea of what to write into the program and then asked Claude for additional ideas and settings to add upon that. Additionally it wrote most of the functions code and some plots. We had to modify all of the plots and parameters but had to make no major code changes within any function. Attempting to use GitHub Copilot in the sidebar of visual studio code led to unsuccessful pieces of code that often did not function. Thus we gave up on using that for anything other than automatic code completion and requested for Claude to generate new methods on its own without reading any of our code. It likely would have been better if Claude could have read what was already there but all of its code functioned after moving into the python notebook with just some variable changes so it was still an improvement on speed. Claude actually did not fall for the trap of not updating the Hastings' correction without any particular prompting for it. Throughout this project we have not copied any code from any source outside of Claude or Copilot. So the main benefit and thing to learn from this is to remove the need to search up functions and methods created by others to start with in coding and to instead be able to get right to the more iterative aspects of changing parameters and other minor changes needed to get good results..