Computational Statistics final project report: the Metropolis-Hastings algorithm

Yulia Grajewska
jg6403@nyu.edu

December 15, 2022

1 Introduction - MCMC methods

The Metropolis-Hastings algorithm is a Markov chain Monte Carlo (MCMC) method used for sampling from a probability distribution. In general, MCMC methods rely on constructing a Markov Chain that has the desired distribution as its equilibrium distribution. MCMC methods have applications in finding numerical approximations of multi-dimensional integrals.

2 The Metropolis-Hastings algorithm

The Metropolis-Hastings algorithm can be used to draw from a distribution with density $P(x)$, if we have a function $f(x)$ that is proportional to the density (and we can calculate the values of f). We generate a sequence of samples such that, as the number of samples increases, the distribution gets closer and closer to the desired distribution. The next sample depends only on the current one (the Markov chain part). At each step, depending on the value of f, the candidate new sample will be either accepted or rejected.

2.1 Pseudocode

Variable list:

- $P(x)$ - desired density, up to a constant
- x_t - current sample
- x' - candidate sample
- α - acceptance ratio. Indicates how probable the candidate sample is with respect to the current sample in the distribution $P(x)$
- $g(x'|x_t)$ - density used to find the new candidate x', usually a Gaussian

The Pseudocode:

$x_0 \leftarrow initial\ guess$
for `each t` **do**
  $x' \leftarrow drawn\ from\ g(x'|x_t)$
  $\alpha \leftarrow f(x')/f(x_t) * g(x'|x_t)/g(x_t|x')$
  $u \leftarrow drawn\ from\ U[0,1]$
  if $u \leq \alpha$ **then**
    $x_{t+1} \leftarrow x'$
  else
    $x_{t+1} \leftarrow x_t$
  end if
end for

In case where the proposition density (g) is symmetric, the $g(x'|x)/g(x|x')$ part of α cancells out.

2.2 drawbacks of the MH algorithm

The biggest drawback of the MH algorithm is that the samples are, very obviously, correlated - the next draw is based on the current one. In the long run, the sampling gives an accurate representation of the source distribution, but successive samples will be next to each other. Additionally, if our choice of the original point is poor, the MH algorithm may need time to find the high density region of the distribution.

3 Experiments with the algorithm

I implemented the Metropolis-Hastings with two different proposition densities - Gaussian ($N(x, 1)$) and Uniform($U[x-1, x+1]$. Below, I present the results of running the algorithm on a 2D Gaussian target distribution with Gaussian proposition density and a 1D distribution with density proportional to $exp[sin^2(x) + sin^2(3x) - 0.5x^2]$ with both proposition densities.

3.1 2D Gaussian

First, I ran the algorithm on a 2D Gaussian with the following density:

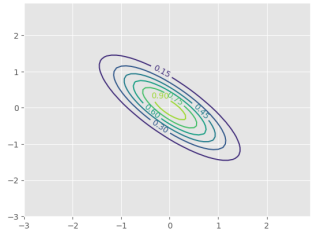


Figure 1: Target Gaussian distribution

The result is presented below.

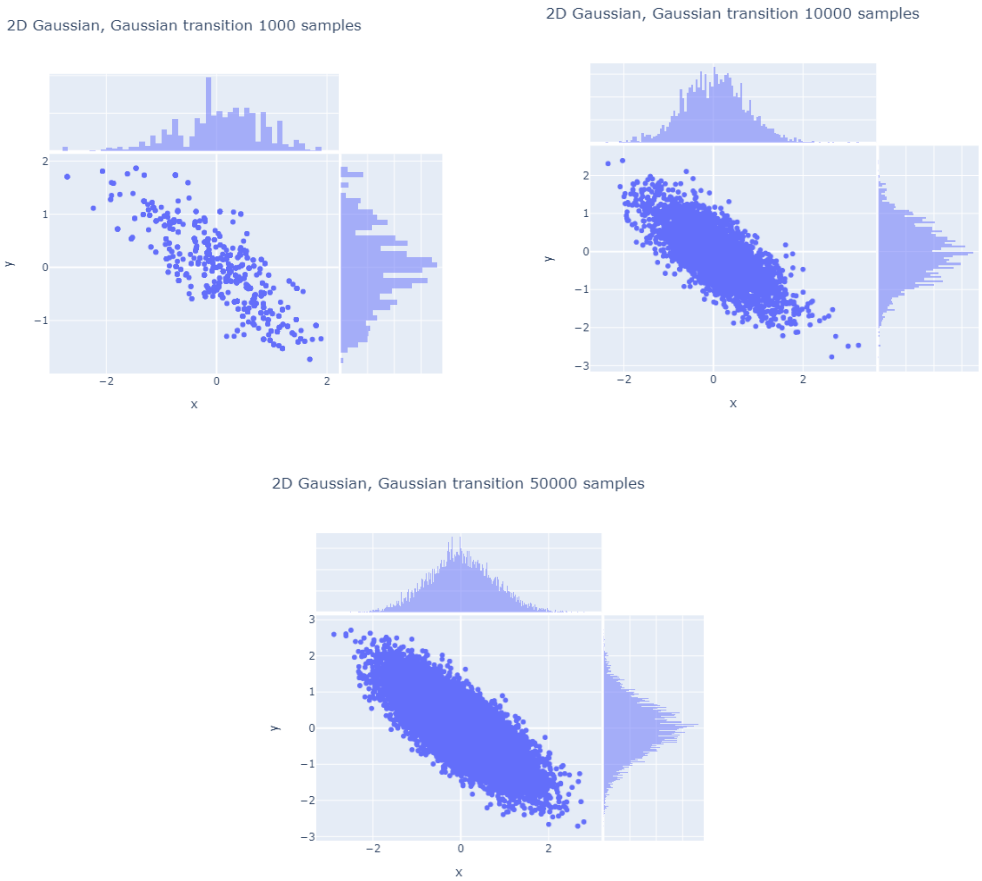
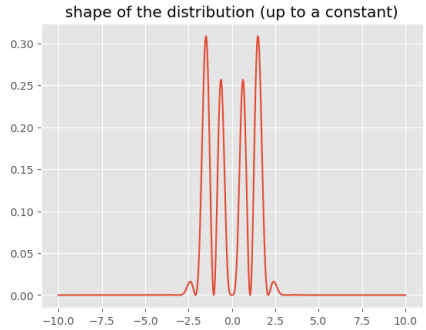


Figure 2: sampling from 2D Gaussian, for 1000, 10000 and 50000 samples

As we can see, the result distribution starts resembling the target one fairly early on - even with a 1000 samples, the result gives us a decent idea of what the target distribution looks like.

3.2 1D sinusoidal distribution

Next, I decided to test the algorithm against a more difficult distribution. The proportional function is $f(x) = exp[sin^2(x) + sin^2(3x) - 0.5x^2]$. I first run the MH algorithm with a Gaussian proposition density, and then with a uniform one. The results are presented below.



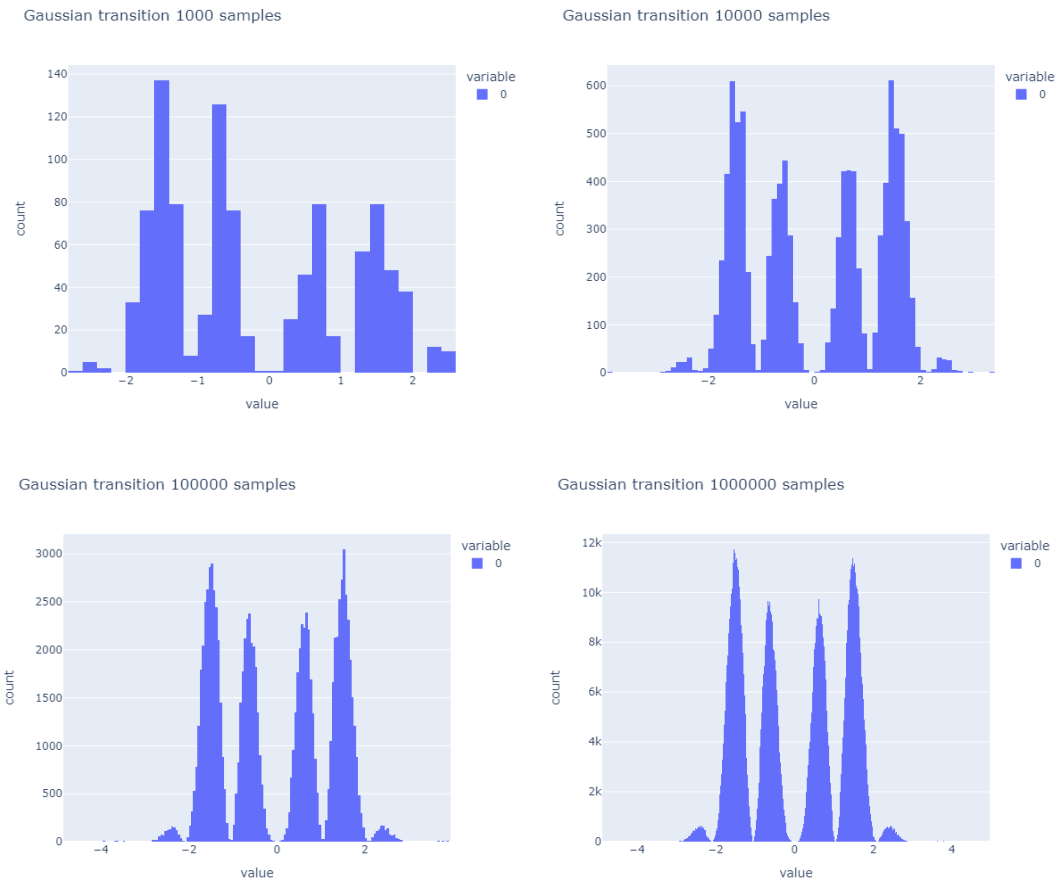


Figure 3: sampling from 1D sinusoidal/exponential, for 1000, 10000, 100000 and 1000000 samples, Gaussian proposition density

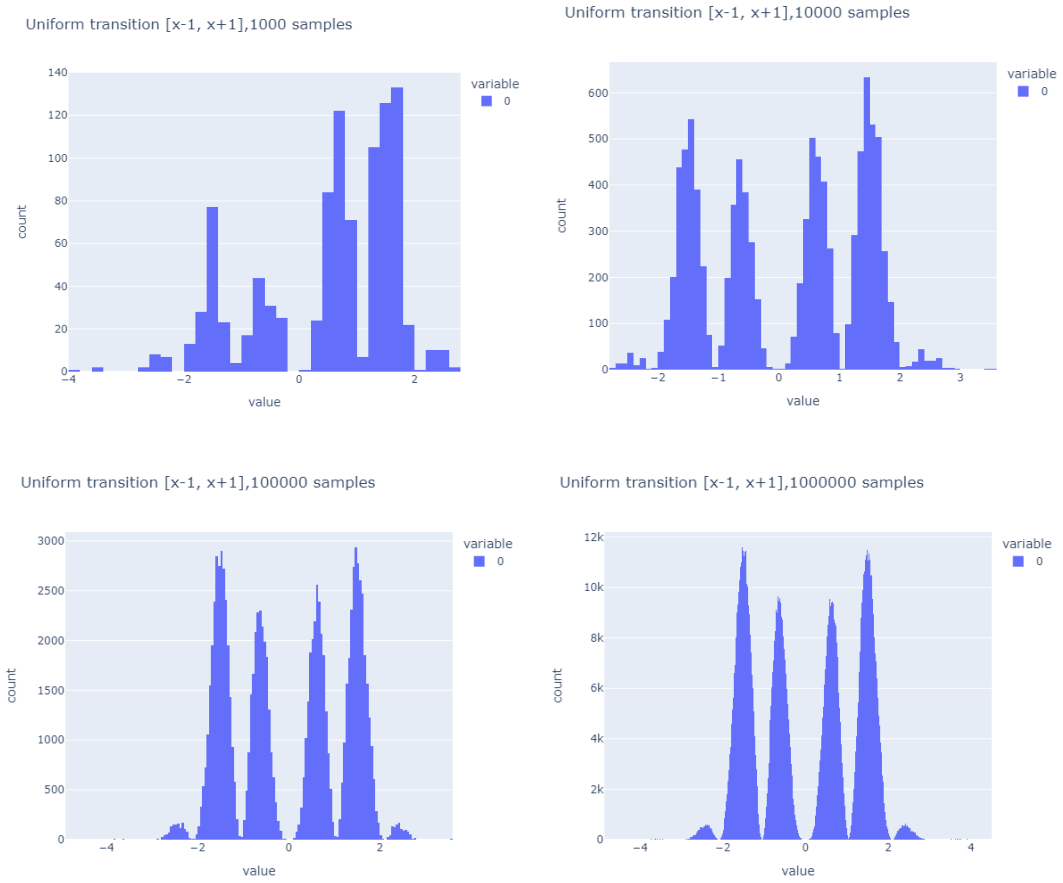


Figure 4: sampling from 1D sinusoidal/exponential, for 1000, 10000, 100000 and 1000000 samples, uniform proposition density

For the chosen parameters (st.d. = 1 for Gaussian and [x-1,x+1] as the range for uniform), it does not look like there is a significant difference in performance.

4 Shortcomings spotted in practice

How to make the Metropolis-Hastings algorithm fail? Turns out that giving it a distribution with a more complicated shape and a more limiting transition function can result in the algorithm not returning an accurate sample. Below is an example of same sinusoidal distribution with transition density being uniform over [x-0.2, x+0.2] - even with 100 000 draws, the algorithm completely misses the right part of the graph!

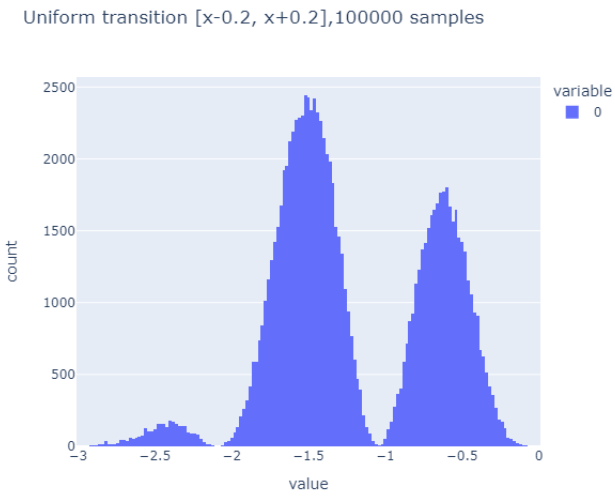


Figure 5: Metropolis-Hastings fails to discover the second half of the graph, despite having 100 000 draws.

5 Conclusion

In summary, I implemented the Metropolis-Hastings algorithm and tested it on two different distributions. For the 2D Gaussian, the result quickly converged to the desired shape. For a more complicated distribution, more samples were needed. I tested out 2 different proposition densities (Gaussian and uniform), but I was not able to observe any significant differences for my choice of parameters. However, with some experimenting, I discovered that it is possible for the algorithm to fail to accurately sample from the distribution, if the chosen proposition function is too constrictive.

6 References

For writing this paper, I have referenced:

- the class notes
- Bishop's *Pattern Recognition and Machine Learning*, section 11.2