Implementing Markov-Chain Monte Carlo

Anusha Muley

January 14 2020

1 Introduction

In this project I attempted to explore the Markov-Chain Monte Carlo algorithm to generate possible Keplerian orbits and see if they fit a model. I used Python to write the code and matplotlib to generate the graphics for this project.

2 Background

A Markov chain is a mathematical system that experiences transitions from one state to another according to certain probabilistic rules. The defining characteristic of a Markov chain is that no matter how the process arrived at its present state, the possible future states are fixed. In other words, the probability of transitioning to any particular state is dependent solely on the current state and time elapsed. The state space, or set of all possible states, can be anything: letters, numbers, weather conditions, baseball scores, or stock performances.

Monte Carlo is a method that we touched on in class. Monte Carlo (MC) methods are a subset of computational algorithms that use the process of repeated random sampling to make numerical estimations of unknown parameters. They allow for the modeling of complex situations where many random variables are involved, and assessing the impact of risk. The uses of MC are incredibly wide-ranging, and have led to a number of groundbreaking discoveries in the fields of physics, game theory, and finance. There are a broad spectrum of Monte Carlo methods, but they all share the commonality that they rely on random number generation to solve deterministic problems. I hope to outline some of the basic principles of Monte Carlo and how I built my own MCMC code.

3 Historical Context

The concept was invented by Stanislaw Ulam, a mathematician who devised these methods as part of his contribution to the Manhattan Project. He used the tools of random sampling and inferential statistics to model likelihoods of outcomes, originally applied to a card game (Monte Carlo Solitaire). Ulam later

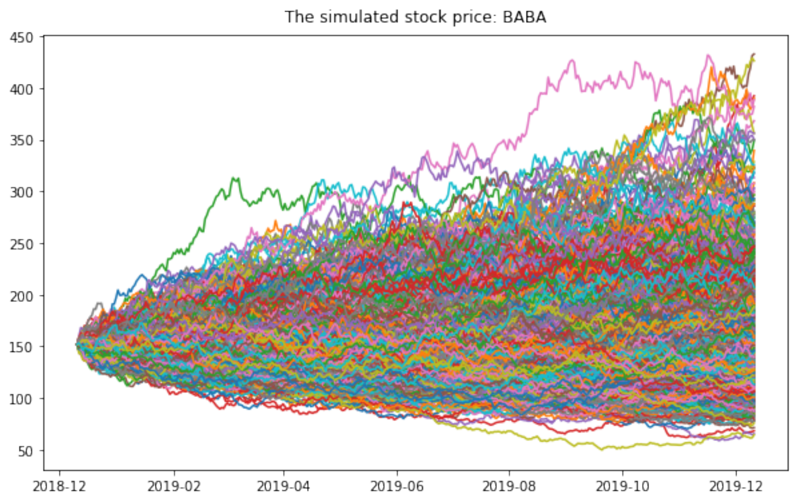


Figure 1: Possible stock simulation using MCMC

worked with collaborator John von Neumann, using newly developed computer technologies to run simulations to better understand the risks associated with the nuclear project. As you can imagine, modern computational technology allows us to model much more complex systems, with a larger number of random parameters, like so many of the scenarios that we encounter during our everyday lives.

4 Basic MCMC Program

Before I made a made a MCMC code for a large scale project, I experimented with a simple code. I searched online a bit and one of the simplest implementations was to make a PI estimation in python by randomly distributing points.

Code:

```
import numpy as np
import matplotlib.pyplot as plt
n = 10;
x = np.random.rand(n, 2)
inside = x[np.sqrt(x[:, 0]**2 + x[:, 1]**2)<1]
estimate = 4*len(inside)/len(x)
print('Estimate of pi: {}'.format(estimate))
plt.figure(figsize = (8,8))
plt.scatter(x[:, 0], x[:, 1], s = .5, c = 'red')
plt.scatter(inside[:, 0], inside[:, 1], s = .5, c = 'blue')
plt.show()
```

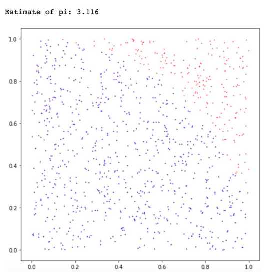


Figure 2: n = 1

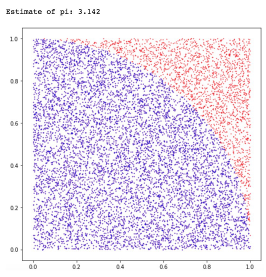


Figure 3: n = 10

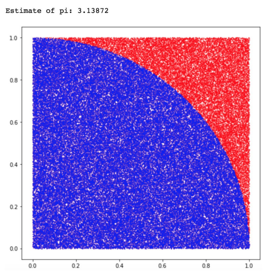


Figure 4: n = 1000

5 Astronomy Implementation

I generated a program to determine if randomized data could fit the eccentricity of Keplerian orbits. I had initially envisioned using data from the Gaia database but I realized that would take a lot more time and since there wasn't much data that I would've been able to use, I decided to make my own data instead. This also allowed me to test how robust my algorithm was and at which point the model could not be applied.

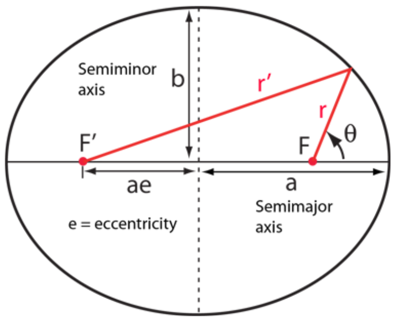
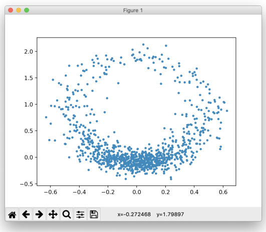
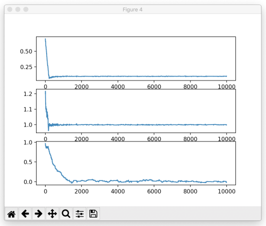


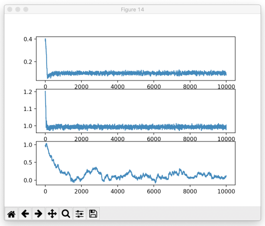
Figure 5: Keplerian Orbit

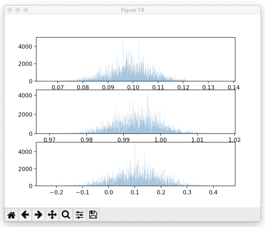
. Make data method:

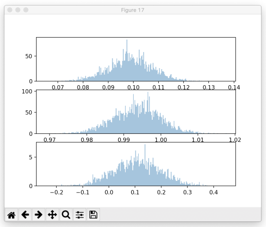
```
def makeData(angle, ecc, a, arg):

    ...: ellerr = np.random.normal(size=len(angle))*0.1

    ...: ellbase = model(angle, ecc, a, arg)

    ...:  return ellbase + ellerr
```

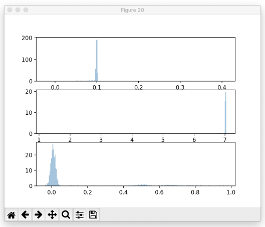












I made a semirandom collection of trials based on the model. Then I attempted to fit that model to them and determine if the eccentricity of the planet would fit the model.

Run MCMC method:

I will go into this method more in class because it is one of the main ones and it is a bit long.

Images from my simulations are depicted above.