Random Walk for Solving PDEs
- Report for Computational Physics Final Project

Yucheng Zhang*
Department of Physics, New York University
(Dated: December 1, 2017)

This is the report for the Computational Physics final project.

INTRODUCTION

There are many numerical methods for solving different types of Partial Differential Equations (PDEs) in Boundary Value Problems (BVPs), such as the relaxation method and the Gauss-Seidel method. These methods are all deterministic methods. In this report, we explore the Monte Carlo method for solving PDEs in BVPs.

We mainly focus on Laplace's Equation with Dirichlet Boundary Conditions,

$$\nabla^2 u = 0 \tag{1}$$

inside the boundary, and

$$u = f(x) \tag{2}$$

on the boundary. In 2D, the Laplace's Equation Eq. 1 can be discreted with finite difference approximation, the centered difference reads,

$$u_{i,j} = \frac{1}{4}(u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1}). \tag{3}$$

For the relaxation method, we use this equation, start from the boundary, and iterate until we get the desired accuracy. Here we look at this equation in a different way. We should notice that the value at one point is the average of its neighbours' values. So we can regard the $\frac{1}{4}$ as the statistical weight of each neighbour while calculating the value of the center point. Then we can construct the Monte Carlo method based on this point of view.

Simple Random Walk Method

Given the point P_0 whose value we want to evaluate on the grid,

1. Pick a neighbour point, e.g. P_1, with proper probability randomly, here it's $\frac{1}{4}$ for each point.

2. We don't know the value at P_1, but we know it's the average of its neighbours, so pick one of it's neighbours, e.g. P_2. Continue this process, until we reach a point on the boundary, e.g. P_n, whose value is given.

3. We know that P_n is an estimate of P_{n-1}, which is an estimate of P_{n-2}, etc, etc, all the way back to P_0. So after this random walk process, we get an estimate of P_0.

4. Do a large amount of random walks from P_0 to the boundary, then average all the estimates, we can get a good estimate of the value at P_0, whose accuracy depends on the number of random walks, i.e. the number of estimates.

Walk on Spheres

As we should notice, for the Laplace Equation, one estimate given by one random walk is just the value at the point on the boundary where the random walk ends. So the estimate doesn't depend on the path of the random walk. If we let the grid spacing go to zero, which obviously will increase the accuracy of the estimate, then the random walk here becomes simple Brownian motion.

Consider a sphere centered at $\boldsymbol{r}$, for a simple Brownian motion starting from $\boldsymbol{r}$, the possibility for the motion to reach any point on the sphere is equal. So since the path doesn't matter here for the Laplace equation, why do we simulate the Brownian motion step by step? We can jump to one point on the sphere directly!

Characteristics of Monte Carlo method

Both the Simple Random Walk (SRW) and Walk on Spheres (WoS) methods are Monte Carlo methods. The Monte Carlo methods here have some characteristics and advantages that normal deterministic methods don't have.

Independence of points

The first thing to notice is that unlike the relaxation or Gauss-Seidel methods, the points on the grid for the SRW or on the grid-free WoS, they are all independent. So it's very fast to estimate some point values.

Boundary & Dimension

For the Monte Carlo methods here, the boundary doesn't matter a lot. The only thing is to decide whether the random walk reaches the boundary, and at which point. So it's relatively easy for these methods to handle some problems with complex boundaries. Also, it's easier for these methods to be extended to higher dimensions.

Parallel Computing

Not only different points, as mentioned above, but also different estimates for one point are independent. So these Monte Carlo methods are naturally parallel. The two most straightforward ways to parallelize the program are,

- Parallelize the different estimates of one point, then average the results from different processes.
- Parallelize the set of points to be evaluated.

In this project, we take the first way.

IMPLEMENTATION & ANALYSIS

SUMMARY & FUTURE WORK

ACKNOWLEDGEMENT

* yz4035@nyu.edu