MULTISCALE MODELING OF DIFFUSION PROCESSES IN THE BRAIN

by

Fredrik E Pettersen
f.e.pettersen@fys.uio.no

THESIS
for the degree of
MASTER OF SCIENCE

Faculty of Mathematics and Natural Sciences
University of Oslo

June 2013

Abstract

This is an abstract text.

To someone

This is a dedication to my cat.

Acknowledgements

I acknowledge my acknowledgements.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Start your chapter by writing something smart. Then go get coffee.
The very first approach was to simply try the problem on a bit. That is to try and substitute some small part of the mesh in a Finite Difference Diffusion solver (Froward Euler scheme) with a stochastic diffusion solver. A random walk method was implemented on part of the mesh to take over the equation-solving. This was done in 1 and 2 spatial dimensions with the aim of finding potential difficulties so that we can further investigate them.
Upon switching length-scales a fundamental question arises almost immediately; what is the contiuum limit? In our case this question takes a slightly different, and possibly more answearble form; what is the conversion rate beween the continuum model and the microscopic model, and by extension, what does a walker correspond to? The first instinct of this candidate was to just try some conversion rate (say some value corresponds to som enumber of walkers), and this was implemented in both 1 and 2 dimensions.

Chapter 2

Random walks

In this chapter we will take a closer look at random walks, both in general and the transition from the statistical view to partial differential equations. We will take a look at different algorithms to produce random walks, and discuss their pros and cons in light of this project.

2.1 Introduction to random walks

The most basic random walk is a walker on the x-axis which will take a step of a fixed length to the right with a probability p, or to the left with a probability $q = 1 - p$. Using (pseudo-) random numbers on a computer we can simulate the outturn of a random walk. For each step (of which there are N) we draw a random number, r, between 0 and 1 from some distribution (say a uniform one) which will be the probability. If $r \leq p$ the walker will take a step to the left, otherwise it will take a step to the left. After the N steps the walker will have taken R steps to the right, and $L = N - R$ steps to the left. The net displacement from the origin will be $S = R - L$.

2.1.1 Further discussion and analysis of the introduction

If we do sufficiently many walks, the net displacement will vary from $S = +N$ to $S = -N$ representing all steps to the right and all steps to the left respectively. The probability of all steps beeing to the right is $P_N(N) = p^N$. Should one of the steps be to the left, and the rest to the right we will get a net displacement of $S = N - 2$ with the probability $P_N(R = N - 1) = Np^{N-1}q$. We can generalize this to finding the probability of a walk with a R steps to the right as

$$P_N(R) = \binom{N}{R} p^R q^{N-R} \tag{2.1}$$

where $\binom{N}{R} = \frac{N!}{R!(N-R)!}$ is the number of walks which satisfy the net displacement in question, or the multiplicity of this walk in statistical mechanics terms. Equation 2.1 is the Bernoulli probability distribution, which is normalized.

$$\sum_{R=0}^{N} P_N(R) = (p+q)^N = 1^N = 1$$

We can use this distribution to calculate various average properties of a walk consisting of N steps. For example, the average number of steps to the right is

$$\langle R \rangle = \sum_{R=0}^{N} R P_N(R) = \sum_{R=0}^{N} \binom{N}{R} R p^R q^{N-R} =$$

$$p\frac{d}{dp}\sum_{R=0}^{N} \binom{N}{R} p^R q^{N-R} = p\frac{d}{dp}(p+q)^N = Np(p+q)^{N-1} = Np$$

From this we can also find the average value of the net displacement using $S = R - L = R - (N - R) = 2R - N$.

$$\langle S \rangle = \langle 2R \rangle - N = 2Np - N\underbrace{(p+q)}_{=1} = N(2p - p - q) = N(p - q)$$

We notice that the average net displacement is greatly dependent on the probability of the walk and that any symmetric walk will have an expected net displacement of zero. In many cases

we will be more interrested in the mean square displacement than the displacement itself. This can also be calculated rather straightforwardly.

$$\langle R^2\rangle = \sum_{R=0}^{N} R^2 P_N(R) = \sum_{R=0}^{N} \binom{N}{R} R^2 p^R q^{N-R} =$$

$$\left(p\frac{d}{dp}\right)^2 \sum_{R=0}^{N} \binom{N}{R} p^R q^{N-R} = \left(p\frac{d}{dp}\right)^2 (p+q)^N$$

$$= Np(p+q)^{N-1} + p^2 N(N-1)(p+q)^{N-2} = (Np)^2 + Np(1-p) = (Np)^2 + Npq$$

Like before, the average nett displacement is given as $S^2 = (2R-N)^2$ and we obtain

$$\langle S^2\rangle = 4\langle R^2\langle -4N\langle R\rangle + N^2 = 4((Np)^2 + Npq) - 4N^2 p + N^2$$
$$= N^2(4p^2 - 4p + 1) + 4Npq = N^2(2p-1)^2 + 4Npq = N^2(p-q)^2 + 4Npq$$

which for the 1D symmetric walk gives $\langle S^2\rangle = N$

Chapter 3

Analysis

In this numerical setup we will potentially introduce several new error sources in addition to the nomal errors introduced by numerical solution of any equation. When a part of the solution acquired is replaced by the solution from another model, which in this case is stochastic, we will change the initial condition to the next iteration in time. This might have a number of effects on our final solution. When we solve a differential equation numerically we only get an approximation to the actual solution because we are using approximate derivatives (see figure **??**). We can investigate this type of error by doing two simulations of the same problem, but replacing the solution in one of the simuations in some area by the random walk model for one timestep.

```
 path = '/home/fredriep/Dropbox/uio/thesis/doc/results/
    experiment_03102013_1218/results/'
import glob

i = [];e = [];s = []

for excl in sorted(glob.glob(path+'Excl*')):
  e.append(np.load(excl))

for incl in sorted(glob.glob(path+'Incl*')):
  i.append(np.load(incl))

for j in xrange(len(i)):
  s.append(np.max(np.abs(i[j]-e[j])))
  print s[-1]
```

As we can see this will print the maximum absolute difference between the two solutions. Since we are dealing with a diffusion process we expect that the first timestep should have the largest error and that the error will in time be killed by the diffusion process.

```
[...]
0.0
0.025832
0.0054384
0.00360192
0.00212352
0.0016940416
0.0012704
0.001173512704
0.0009942699008
0.00093037021184
0.00083786862592
0.000782433954202
0.000723955260948
0.000680186522141
0.000638864266527
0.000604769629612
```

The error is larger than Δt by a factor of 10 ($\Delta t = 0.002$)... This might be improved by increasing the number of walkers, or finding a better steplength for the walkers as well as improving the walk-algorithm etc.. The above results were obtained by using the previous timestep as input to the random walk solver. In most numerical algorithms (Euler-Chromer etc) one will benefit from using the newest timestep as early as possible. Doing the same experiment with the modification of using the latest timestep as input to the random walk solver we obtain the following:

```
[...]
0.0
0.06
0.014
0.0072
0.0036
0.00232
0.0014912
0.0011744
0.000803712
0.000703744
0.00059240448
0.000536887296
0.0004960012288
0.00047304564736
0.000437414641664
0.00041956655104
```

That is, the initial error is larger, but it is reduced faster. However, this is a stochastic process, and one should average over many experiments. An average over (only) ten experiments using twice the number of walkers (M=100) shows that the errors seem to be converging towards the same value regardless of which timestep is used as input.

3.1 Manifactured Solutions

A normal way of checking that our sceme of choise is implemented correctly is by making an exact solution to the equation and checking that the error is of the expected order. As a first, simple implenetation we have worked with the explicit Forward Euler discretization of the simplest form of the diffusion equation **??**. This discretization is expected to have an error-term of the order of Δt, which again is limited by a stability criterion. We can now decide that the solution to equation **??** should be

$$u(x,t) = e^{-t\pi^2}\cos(\pi x) + 1 \tag{3.1}$$

which satisfies our equation if we set the diffusion constant to 1.

$$\frac{\partial}{\partial t}e^{-t\pi^2}\cos(\pi x) + 1 = D\frac{\partial^2}{\partial x^2}e^{-t\pi^2}\cos(\pi x) + 1 \tag{3.2}$$

$$-\pi^2 e^{-t\pi^2}\cos(\pi x) = -\pi^2 e^{-t\pi^2}\cos(\pi x) + 1 \implies 1 = 1 \tag{3.3}$$

As we saw in section **??** the Forward Euler scheme is expected to have an error of the order of Δt. Testing only the scheme first, in 1D we get the following plot 3.1 of the maximum of the absolute value of the difference between the exact solution and the numerical solution to the equation.

Figure 3.1: Numerical error for 1D Forwar Euler discretization.

We then introduce an area on the domain where we swich models from the normal PDE to an average of the PDE solution and the result of a random walk simulation where the initial

condition is the last timestep from the PDE converted to walkers by the conversion rate given in equation 3.4. In this case we have used the parameters $a = 3$, $\Delta t = \frac{\Delta x^2}{3.0}$, $\Delta x = \frac{1}{20}$. These parameters makes one unit of $u(x,t)$ equal to some 1000 walkers.

$$C_{ij} = \frac{a}{\Delta t} U_{ij} \tag{3.4}$$

The area where the model has been replaced is between $x = 0.6$ and $x = 0.7$, which is three mesh points. In the same way as for only the simple 1D PDE case we compare the combined numerical solution from the two models to the exact solution. Figure **??** shows that the error is still of the order of Δt, and the difference between the two models are negligable.

Figure 3.2: Numerical error for 1D Forward Euler discretization combined with random walk model between $x = 0.6$ and $x = 0.7$.