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

The beginning is here

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

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.