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 2014

Abstract

This is an abstract text.

Acknowledgements

I acknowledge my acknowledgements.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This thesis is an attept at modeling duffusion processes in which part of the process takes place on a length scale so small that the continuum approximation becomes invalid. In this part we will therefore try and introduce anoter model of the diffusion process in the hope that this will give us something extra. It is the hope of Hans Petter that this thesis will be an introduction to a new research project to understand mesoscale physics in collaboration with Gaute Einevoll at UMB.

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

Basic Theory

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. Then we will take a quick look at partial diffential equations and numerical solution of them.

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 right. 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 $S = R - L$.
This simple approach is easily generalizable to two and three dimensions by having $2d$ possible outcomes from the random number, where d is the dimensionality. In two d the walker will step up if $r \in (0.75, 1]$ and left if $r \in [0, 0.25)$, for example.

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 relationship between p and q, 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, because many important large scale parameters can be related to the root-mean-sqare displacement. This can also be calculated rather straightforwardly.

$$\langle R^2\rangle = \sum_{R=0}^{N} R^2 P_N(R) = \sum_{R=0}^{N} \binom{N}{R} R^2 p^R q^{N-R} =$$
$$\left(p\frac{d}{dp}\right)^2 \sum_{R=0}^{N} \binom{N}{R} p^R q^{N-R} = \left(p\frac{d}{dp}\right)^2 (p+q)^N$$
$$= Np(p+q)^{N-1} + p^2N(N-1)(p+q)^{N-2} = (Np)^2 + Np(1-p) = (Np)^2 + Npq$$

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

$$\begin{aligned}\langle S^2\rangle &= 4\langle R^2\langle -4N\langle R\rangle + N^2 = 4((Np)^2 + Npq) - 4N^2p + N^2\\ &= N^2(4p^2-4p+1) + 4Npq = N^2(2p-1)^2 + 4Npq = N^2(p-q)^2 + 4Npq.\end{aligned}$$

which for the 1D symmetric walk gives $\langle S^2\rangle = N$ and the variance, denoted $\langle \Delta S^2\rangle = \langle\langle S^2\rangle - \langle S\rangle^2\rangle$, is found by instertion as

$$\langle \Delta S^2\rangle = \langle N^2(p-q)^2 + 4Npq - (N(p-q))^2\rangle = 4Npq \tag{2.2}$$

When the number of steps gets very large we can approximate the Bernoulli distribution 2.1 by the Gaussian distribution. This is most easily done in the symmertric case where $p = q = \frac{1}{2}$, but it is sufficient for the steplengths to have a finite variance (*find something to refer to*). The Bernoulli distribution then simplifies to

$$P(S,N) = \left(\frac{1}{2}\right)^N \frac{N!}{R!L!} \tag{2.3}$$

on which we apply Stirlings famous formula for large factorials $n! \simeq \sqrt{2\pi n}n^n e^{-n}$.

$$\begin{aligned}P(S,N) &= \left(\frac{1}{2}\right)^N \frac{N!}{R!L!}\\ &= \exp\left(-N\ln 2 + \ln\sqrt{2\pi N} + N\ln N - \ln\sqrt{2\pi R} - R\ln R - \ln\sqrt{2\pi L} - L\ln L\right)\\ &= \sqrt{\frac{N}{2\pi RL}}\exp\left(-R\ln\frac{2R}{N} - L\ln\frac{2L}{N}\right)\end{aligned}$$

Where we have used $R + L = N$. We now insert for $\frac{2R}{N} = 1 + \frac{S}{N}$ and $\frac{2L}{N} = 1 - \frac{S}{N}$ and expand the logarithms to first order, $RL = \frac{N^2-S^2}{4}$ in the prefactor, and approximate $1 - \frac{S^2}{N^2} \simeq 1$. This gives

$$P(S,N) = \sqrt{\frac{2}{\pi N}}\exp\left(\frac{-S^2}{2N}\right) \tag{2.4}$$

which is an ordinary, descrete Gaussian distribution with $\langle S\rangle = 0$ and $\langle S^2\rangle = N$. If we keep assuming that the walker is on the x-axis, and let the step length, a, get small the final position will be $x = Sa$ which we can assume is a continous variable. Similarly, we let the time interval between each step, τ, be small and let the walk run for a contious time $t = N\tau$. This changes the distribution 2.4 to

$$P(x,t) = \frac{1}{2a}\sqrt{\frac{2\tau}{\pi t}}\exp\left(-\frac{x^2\tau}{2a^2t}\right). \tag{2.5}$$

The prefactor $\frac{1}{2a}$ is needed to normalize the continous probability distribution since the sepatation between each possible final position in walks with the same number of steps is $\Delta x = 2a$. We also introduce the diffusion constant

$$D = \frac{a^2}{2\tau} \tag{2.6}$$

making the distribution

$$P(x,t) = \sqrt{\frac{1}{4\pi Dt}} \exp\left(-\frac{x^2}{4Dt}\right) \tag{2.7}$$

Introducing x also gives us the expected value and variance of x on a form which will be usefull later. We have $x = Sa$ which means

$$\langle x \rangle = a\langle S \rangle$$

and

$$\langle x^2 \rangle = a^2 \langle S^2 \rangle$$

Finally by insertion we find the variance $\langle \Delta x^2 \rangle$

$$\langle \Delta x^2 \rangle = \langle \langle x^2 \rangle - \langle x \rangle^2 \rangle = \langle a^2 \langle S^2 \rangle - a^2 \langle S \rangle^2 \rangle = 4Npqa^2 \tag{2.8}$$

2.1.2 More general Random Walks

In the more general case, the position of a random walker,$\mathbf{r}$ at a time t_i is given by the sum

$$\mathbf{r}(t_i) = \sum_{j=0}^{i} \Delta \mathbf{x}(t_j) \tag{2.9}$$

where $\Delta \mathbf{x}(t_j) = (\Delta x(t_j), \Delta y(t_j), \Delta z(t_j))$ in 3D. Each $\Delta x, y, z$ is a random number drawn from a distribution with a finite variance $\sigma^2 = \langle \Delta x^2 \rangle$. By the central limit theorem, any stochastic process with a well defined mean and variance can, given enough samples, be approximated by a Gaussian distribution. This means that the probability of finding the walker at some position x after M steps is

$$P(x, M) \propto e^{-\frac{x^2}{2M\sigma^2}} \tag{2.10}$$

Remember that the actual gaussian distribution is

$$\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{(n-\mu)^2}{2\sigma^2}\right)$$

We can introduce an Einstein relation $\sigma^2 = 2D\Delta t$ and the obvios relation $t = M\Delta t$ to get a more desirable exponent. We see that $\langle \Delta x^2 \rangle = 2Dt$. *The introduction of the Einstein relation might put some restrictions on our model.* Normalizing the expression gives us

$$P(x,t) = \sqrt{\frac{1}{4Dt}} \exp\left(-\frac{x^2}{4Dt}\right) \tag{2.11}$$

If we have a large number, N, of walkers, their concentration will be $C(x,t) = NP(x,t)$. The concentration is conserved, so any amount that flows out of an area must reflect as a decrease in concentration. We can express this by the flow of concentration

$$\frac{\partial C}{\partial t} - \nabla \cdot \mathbf{J} = S \tag{2.12}$$

where $\mathbf{J}$ is the flow vector and S is a source term which in our case will be zero. From Fick's first law we know that $\mathbf{J} = -D\nabla C$. Inserting this gives us

$$\frac{\partial C}{\partial t} = \nabla \cdot (D \cdot \nabla C) \tag{2.13}$$

which is the diffusion equation. By insertion we can check that this version (2.11) of the gaussian distribution fulfills the diffusion eqution. Starting with only the time derivative gives us

$$\frac{\partial P}{\partial t} = -\frac{4\pi D \exp\left(-\frac{x^2}{4Dt}\right)}{2\sqrt{(4\pi Dt)^3}} + \frac{x^2 \exp\left(-\frac{x^2}{4Dt}\right)}{4Dt^2\sqrt{4\pi Dt}}$$

$$= \exp\left(-\frac{x^2}{4Dt}\right)\left(\frac{8Dx^2}{2\sqrt{\pi}(4Dt)^{5/2}} - \frac{(4D)^2 t}{2\sqrt{\pi}(4Dt)^{5/2}}\right) = \frac{4D\exp\left(-\frac{x^2}{4Dt}\right)(x^2 - 2Dt)}{\sqrt{\pi}(4Dt)^{5/2}}$$

We then finish by doing the spatial derivative

$$D\frac{\partial^2 P}{\partial x^2} = \frac{D}{\sqrt{4\pi Dt}}\frac{\partial}{\partial x}\left[-\exp\left(\frac{-x^2}{4Dt}\right)\left(\frac{-2x}{4Dt}\right)\right]$$

$$= \frac{2D}{4Dt\sqrt{4\pi Dt}}\exp\left(\frac{-x^2}{4Dt}\right)\left[1 - x\left(\frac{2x}{4Dt}\right)\right] = \frac{4D\exp\left(-\frac{x^2}{4Dt}\right)(x^2 - 2Dt)}{\sqrt{\pi}(4Dt)^{5/2}}$$

and see that they are equal, meaning that the diffusion equation is satisfied.

2.1.3 Choosing random walk algorithm

As this article points out [1] the simplest random walk model, which places walker on discrete mesh points and uses a fixed step length, has been used with great succsess to model diffusion processes. However, this model will struggle with reproducing anisotropic diffusion, that is $D = D(x)$. Farnell and Gibson also suggest a method for improving the results by adjusting the step length according to to position, thus effectively adjusting the diffusion constant of the walk as well. We see then that the simplest model is rather robust, and well tested. However, the aim of this project is to combine two realistic models for diffusion on different length scales, and the simplest random walk model has one fundamental flaw in that view; it is not a realistic model for diffusing particles. Brownian motion is a more realistic physical model for diffusing particles, and can (I think) quite easily be modified to model anisotropic diffusion as well. We can model brownian motion simply by using equation 2.9, and we can with a bit of work expand it to model collisions between walkers as well.
That beeing said, by the central limit theorem both models will after some timesteps be described by a gaussian distribution meaning that on the PDE scale we will not know the difference. Hence it will make no sense to not use the simplest random walk model.

2.1.4 Potential problems or pitfalls with combining solutions

There are a few obvious difficulties we can expect to run into in our planned project. Future ones will be added here as well.

- Different timescales
  The PDE-solver will be operating with some timestep Δt which will, depending on the discretization of the PDE, have some constraints and will definately have an impact on

the error. The walkers will, as we have just seen, solve the diffusion equation as well, but with some different $\Delta\tilde{t}$ which is smaller than the timestep on the PDE level. Depending on the couplig chosen between the two models this difference will have some effect or a catastrophic effect on the error. Running some number of steps, N, on the random-walk level should eventually sum up to the timestep on the PDE level, $\sum_{i=0}^{N} \Delta\tilde{t} = \Delta t$. It turns out, as we will see in section 2.3.2 that we can make sure the coupling is as good as it gets by restricting the step length of the walkers.

- Boundary conditions
  To combine the two models we will need to put restricting boundary conditions on the random walks. This is not usually done (as far as I have seen), but not very difficult. Finding a boundary condition that accurately models the actual system turns out to be quite straightforward, so long as the walk-domain is not on the actual boundary of the whole system. We can assume that the number of walkers in the walk-domain is conserved for each PDE-timestep, and thus no walkers can escape the domain. Implementing perfectly reflecting boundaries solves this quite well. This means that the flux of walkers out of a boundary is zero, which is the same as Neumann boundary conditions on the PDE level.
  Dirichlet boundaries can (probably) be implemented by adding or removing walkers on the boundaries (or in a buffer-zone around them) untill we have the desired concentration of walkers.

- Negative concentration of walkers
  The concentration of walkers is calculated as $NP(x, t)$ where $P(x, t)$ is really only an estimate of the actual probability distribution, calculated by dividing the number of walkers in one area $x \pm \frac{\Delta x}{2}$ by the total number of walkers. Seeing as negative probabilities does not make sense, and neither does a negative number of walkers, we will eventually run into some problems when the solution of the PDE takes negative values (which it might do). The solution to this is simply to store the signs of the solution to the PDE in an array, and send only positive values to the random walk solver. When we convert the number of walkers back to a PDE solution we still have the sign from before and can multiply the concentration by the sign it had in the last timestep.

- Smooth solutions
  A diffusion process is very effective when it comes to dampig fast fluctuations, and so any solution of the diffusion equation will be smooth. When we itroduce a stochastic process, we will potentially also introduce fast fluctutations from one timestep to the next. In this case we are faced with a dilemma; on the one hand there is the smoothness of the solution to consider, on the other hand we have introduced the stochastic term believing that it adds detail to our model. The approach we use to this is to do some curve-fitting using both of the solutions. This will give us some difference between the two models and some smoothness.

2.1.5 Random walks and anisotropy

Any real problem where parts of the diffusion process cannot be modelled by the continuum approximation is bound to be anisotropic. There is reason to belive that an anisotropic diffusion process on the PDE level will lead to an aisotropic random walk model as well, but how do we modell this. Equation 2.24 shows the step length as a function of the diffusion constant. If we simply replace the diffusion constant by a function $D = D(\mathbf{x})$ we are at least started, but this will not quite be sufficient as

2.1.6 Random walks and drift

Another point we have yet to say something about is diffusion that has a drift term, $\frac{\partial u}{\partial x}$. Initially one thought that diffusion in the ECS of the brain was governed by a drift term , but the modern preception is that this drift term is in the very least negligible [2]. The drift term might be of importance in other applications, however, and som we should look into it.
How do we model random walks with drift?
A first instinct is to simply add some vector to the brownian motion model, thus forcing all walkers to have a tendency to walk a certain direction. This approach can also be used in the fixed steplength (or variable steplength in the anisotropic case) if we express the new step, $\mathbf{s}$, as

$$\mathbf{s} = (\pm l \text{ or } 0, \pm l \text{ or } 0) + \mathbf{d}$$

where $\mathbf{d}$ denotes the drift of the walker.
We can set up the continuity equation for a concentration, $C(x,t) = NP(x,t)$ of random walkers which are affected by a drift.

$$\frac{\partial C}{\partial t} + \nabla \cdot \mathbf{j} = S \tag{2.14}$$

Where $\mathbf{j}$ denotes the total flux of walkers through some enclosed volume and S is a source/sink term. Since the walkers are affected by drift the flux will consist of two terms; $\mathbf{j} = \mathbf{j}_{diff} + \mathbf{j}_{drift}$. From Fick's first law we know that $\mathbf{j}_{diff} = -D\nabla C$. The second flux term is the advective flux which will be equal to the average velocity of the system; $\mathbf{j}_{drift} = \mathbf{v}C$. Inserting this in the continuity equation gives us the well known convection diffusion eqution (2.15).

$$\frac{\partial C}{\partial t} = \nabla \cdot (D\nabla C) - \nabla \cdot (\mathbf{v}C) + S \tag{2.15}$$

Which in many cases will simplify to

$$\frac{\partial C}{\partial t} = D\nabla^2 C - \mathbf{v} \cdot \nabla C \tag{2.16}$$

2.2 Some words about partial differential equations

2.2.1 Discretizing

To maintain a bit of generality we will look at the (potentially) anisotropic diffusion equation in 2d. The extension to 3d is trivial, as is the 1d version.

$$\frac{\partial u}{\partial t} = \nabla D \nabla u + f \tag{2.17}$$

where f is some source term. The final expression and scheme will depend on how we chose to approximate the time derivative, but the spatial derivative will mostly have the same approximation.
We start off by doing the innermost derivative in one dimension. The generalization to more dimesions is trivial, and will consist of adding the same terms for the y and z derivatives.

$$\left[\frac{d}{dx}u\right]^n \approx \frac{u^n_{i+1/2} - u^n_{i-1/2}}{\Delta x}$$

Where we have made the approximate derivative around the point x_i. We then set $\phi(x) = D\frac{du}{dx}$ and do the second derivative

$$\left[\frac{d}{dx}\phi\right]^n \approx \frac{\phi^n_{i+1/2} - \phi^n_{i-1/2}}{\Delta x}$$

and insert for ϕ

$$\frac{\phi^n_{i+1/2} - \phi^n_{i-1/2}}{\Delta x} = \frac{1}{\Delta x^2}\left(D_{i+1/2}(u^n_{i+1} - u^n_{i+1}) - D_{i-1/2}(u^n_i - u^n_{i-1})\right)$$

Since we can only evaluate the diffusion constant at the mesh points (or strictly speaking since it is a lot simpler to do so) we must approximate $D_{i\pm1/2} \approx 0.5(D_{i\pm1} + D_i)$. Inerting this gives us

$$\begin{aligned}\nabla D \nabla u \approx & \frac{1}{2\Delta x^2}\left((D_{i+1,j} + D_{i,j})(u_{i+1,j} - u_{i,j}) - (D_{i,j} + D_{i-1,j})(u_{i,j} - u_{i-1,j})\right) \\ & + \frac{1}{2\Delta y^2}\left((D_{i,j+1} + D_{i,j})(u_{i,j+1} - u_{i,j}) - (D_{i,j} + D_{i,j-1})(u_{i,j} - u_{i,j-1})\right)\end{aligned}$$

2.2.2 Stability

In section 2.2.1 we used the Forward Euler approximation to the time derivative. Unfortunately the resulting scheme is potentially unstable, as we shall now see. We start out by assuming that the solution $u(x, t)$ is on the form

$$u(x, t) = A^n \exp(ikp\Delta x) \tag{2.18}$$

where $i^2 = -1$ is the imaginary unit and A^n is an amplification factor which, for the solution **??** ideally should be $\exp(-\pi^2 t)$, but will be something else in the numerical case. We notice that we must have $|A| \leq 1$ if u is to not blow up. Inserting 2.18 in the simplified version of the variable coefficient scheme (where the coefficient is constant) gives us the following

$$\begin{aligned}\exp(ikp\Delta x)\left(A^{n+1} - A^n\right) &= A^n\frac{D\Delta t}{\Delta x^2}\left(\exp(ik(p+1)\Delta x) - 2\exp(ikp\Delta x) + \exp(ik(p-1)\Delta x)\right) \\ A^n \exp(ikp\Delta x)\,(A - 1) &= A^n \exp(ikp\Delta x)\frac{D\Delta t}{\Delta x^2}\left(\exp(ik\Delta x) - 2 + \exp(-ik\Delta x)\right)\end{aligned}$$

Using the well known identies $\exp(iax) + \exp(-iax) = \frac{1}{2}\cos^2\left((\frac{ax}{2}\right)$ and $\cos^2(ax) - 1 = \sin^2(ax)$ gives us

$$A - 1 = \frac{D\Delta t}{\Delta x^2}\sin^2\left(\frac{k\Delta x}{2}\right) \tag{2.19}$$

We now insert for the "worst case scenario" $max(\sin^2\left(\frac{k\Delta x}{2}\right)) = 1$

$$A = \frac{D\Delta t}{2\Delta x^2} + 1 \implies \Delta t \leq \frac{\Delta x^2}{2D} \tag{2.20}$$

In 2d this criterion is halved, and for the anisotropic case we must insert for the maximum value of D which, again, will be the "worst case scenario".

2.2.3 Truncation error

As we know the numerical derivative is not the analytical derivative, but an approximation. This approximation has a well defined residual, or truncation error which we can find by Taylor expansion.

$$R = \frac{u(t_{n+1}) - u(t_n)}{\Delta t} - u'(t_n)$$

Remember Taylor expansion of $u(t+h) = \sum_{i=0}^{\infty} \frac{1}{i!}\frac{d^i}{dt^i}u(t)h$

$$\begin{aligned} R &= \frac{u(t_n) + u'(t_n)\Delta t + 0.5u''(t_n)\Delta t^2 + \mathcal{O}(\Delta t^3) - u(t_n)}{\Delta t} - u'(t_n) \\ &= u''(t_n)\Delta t + \mathcal{O}(\Delta t^2) = \mathcal{O}(\Delta t) \end{aligned}$$

We can do better than this by using another discretization scheme for the PDE, but in our case the PDE is not the only error source seeing as we will combine it with a random walk solver. Quantifying an error term for the random walk solver is not straightforward, but naturally it will be closely coupled to the number of walkers used. So far the error seems to behave as expected, meaning that introducing very many walkers might reduce the error to $\mathcal{O}(\Delta t^2)$ if the number of walkers, N is proportionate to $N \propto \frac{1}{\Delta t^2}$. Since $\Delta t \leq \frac{D\Delta x^2}{2}$ by the stability constraint (in 1D), we will already for small meshes of some 20 points need to introduce ~ 600000 walkers per unit “concentration” per meshpoint in the walk-area. This will be such a costly operation that it will not necesarily be worth it.

2.3 Combining the two solvers

This section will deal with the actual combination of the two models.

2.3.1 The basic algorithm

The basic structure of the program is to have one solver-object which contains one PDE-solver for the normal diffusion equation, and a linked list of random walk-solvers and their relevant areas. Before we start we must add an initial condition along with some parameters such as the diffusion constant (/tensor) and Δt, and we have the opportunity to mark areas on the mesh where we want random walk solvers. The method for adding walk-areas will map them to an index and set the initial condition for the walk. In the future we plan to add the possibility of setting boundary conditions and having anisotropy follow into the random walk solvers as well. At each timestep we call the solve-method of the combined solver, which in turn calls the solve method for the PDE-solver. We then loop over the random walk solvers and call their solve-methods. The results of these are inserted in the solution from the PDE using some rutine (e.g. the average of the two) and the timestep is done. A schematic of the algoritm is provided in figure 2.1.

Figure 2.1: Schematic diagram of the algorithm.

2.3.2 Probability distribution and timesteps

As we saw in section 2.1.2 the probability of finding a walker at a position x_i after some N timesteps (on the walk-scale) is (in the limit of large N) given as the gaussian distribution.

In our application, however, we are not interrested in finding the walker at an exact position, but in an interval around the meshpoints sent to the walk-solver. This interval is (for obvious reasons) $x_i \pm \frac{\Delta x}{2}$ where Δx is the mesh resolution on the PDE level. We will also run the walk solver for some N timesteps on the random-walk scale (where N steps on the random walk scale is the same as one step on the PDE scale). This slightly modifies our distribution into

$$P(x_i \pm \Delta x, t_{n+1}) = \frac{1}{\sqrt{4\pi DN\Delta\tilde{t}}} \exp\left(\frac{(x \pm \Delta x)^2}{4DN\Delta\tilde{t}}\right) \tag{2.21}$$

This makes the concentration of walkers $C(x,t) = MP(x,t)$

$$C(x_i \pm \Delta x, t_{n+1}) = \frac{M}{\sqrt{4\pi DN\Delta\tilde{t}}} \exp\left(\frac{(x \pm \Delta x)^2}{4DN\Delta\tilde{t}}\right) \tag{2.22}$$

For each PDE-timestep we reset the walkers to have some new initial condition. This is done to make sure that statistical fluctuations will not put the diffusive process "off course". The point is that $C(x_i \pm \Delta x, t_{n+1})$ will be dependent on the initial condition $C(x_i \pm \Delta x, t_n)$.

Looking at the difference in timestep size between the two lengthscales we see from equation 2.4 that the stepsize on the random walk scale is dependent on the variance in the actual steps (This is in principle the Einstein relation).

$$\sigma^2 = \langle \Delta x^2 \rangle = 2DN\Delta\tilde{t} \implies \Delta\tilde{t} = \frac{\langle \Delta x^2 \rangle}{2DN} \tag{2.23}$$

Equating this with 2.8 gives us a first order approximation to the steplength, l

$$\begin{aligned} \langle \Delta x^2 \rangle &= 4pqNl^2 = 2DN\Delta\tilde{t} \\ l &= \sqrt{2D\Delta\tilde{t}}. \end{aligned} \tag{2.24}$$

Of course this is assuming that we use a random walk algorithm of fixed steplength.

Chapter 3

Analysis

3.1 Some discussion

This chapter will concern most of the numerical error analysis and som of the discussion of this analysis as well as an introduction to the methods used for error analysis in general, and how they are adapted to this particular problem.

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 (see section 2.2). 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 **??**). To investigate the error we are introducing we will first need to test that the truncation error of the numerical PDE solver behaves as expected.

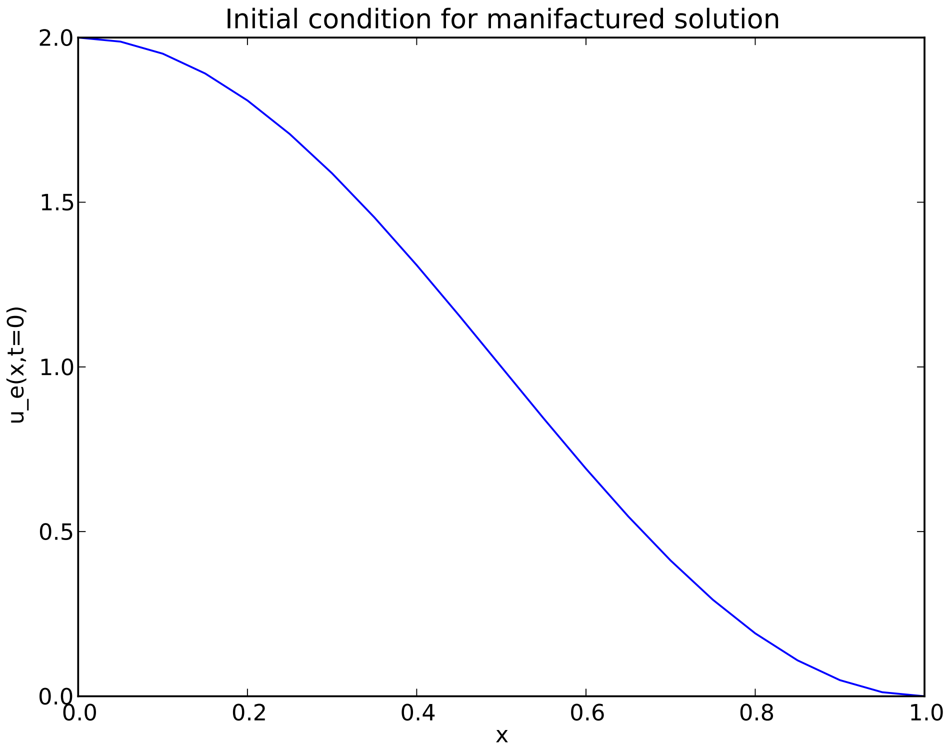

Figure 3.1: Initial condition of manifactured solution in 1d and the simulation.

3.2 Manifactured Solutions

A normal way of checking that our scheme 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 2.13. 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 2.13 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 2.2.3 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 3.2 of the maximum of the absolute value of the difference between the exact solution and the numerical solution to the equation.

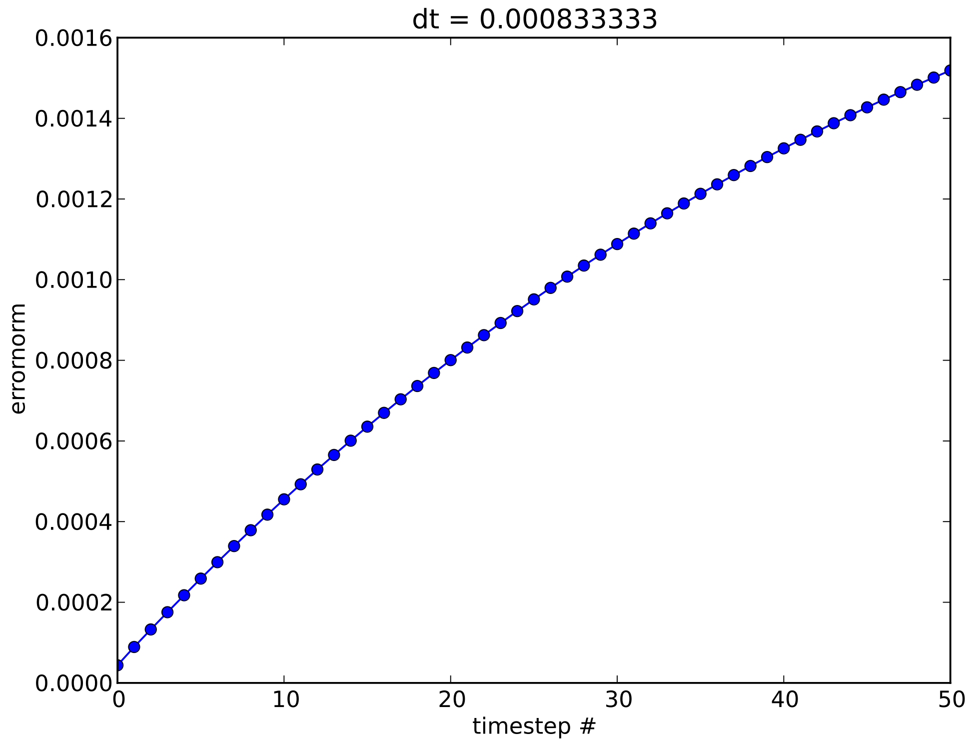

Figure 3.2: Numerical error for 1D Forwar Euler discretization of the PDE. Nothing else is done to the simulation.

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 3.3 shows that the error is still of the order of Δt, and the difference between the two models are negligable.

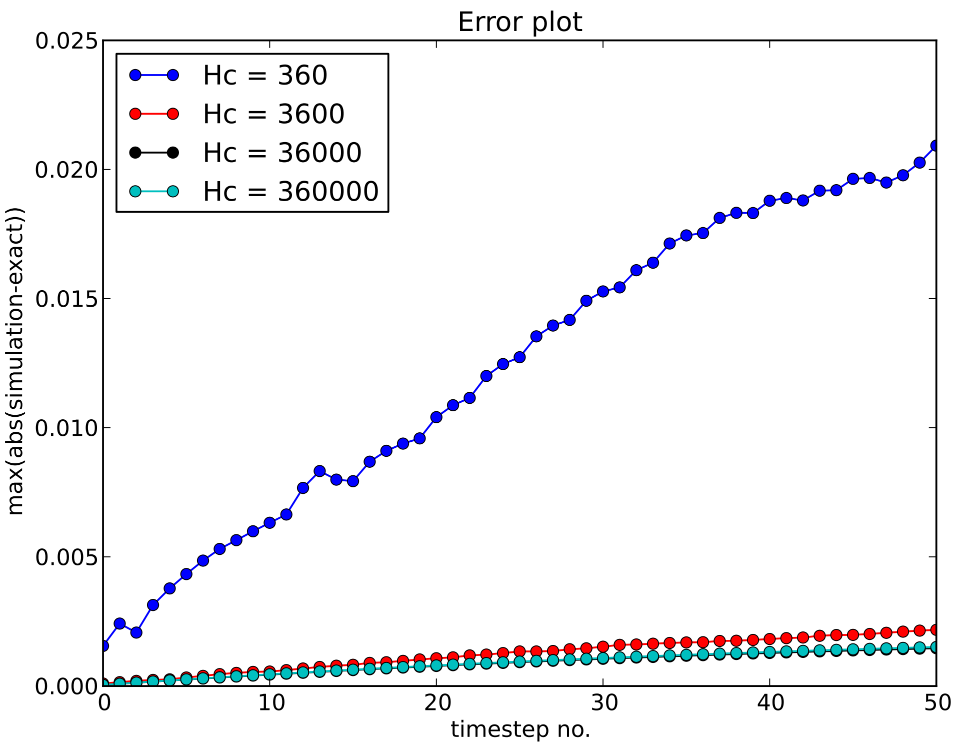

Figure 3.3: Numerical error for 1D Forward Euler discretization combined with random walk model between $x = 0.6$ and $x = 0.7$. Other parameters of importance are $\Delta t \approx 0.00083333$, $\Delta x = 0.05$.

As we can se from figure 3.3 increasing the number of walkers that each "unit concentration" corresponds to has a positive effect on the errornorm up to a certain point. After we reach $Hc \sim 10^4$ the error is dominated by the truncation error of the Forward Euler scheme. This tells us that the error associated with introducing a random walk model on parts of the mesh behaves roughly as we hoped; it tends to zero for large numbers of walkers. Meaning of course that the calculations in section **??** are not hopeless.

3.2.1 The effects of adding drift to the walkers

As shown in section 2.1.6 adding drift to the walkers will modify our model to represent the convection diffusion equation (2.15) rather than the simple diffusion equation. In our analysis so far we have completely ignored the spatial truncation error because it is of second order, and the truncation error in time is of first order. When discretizing the convection diffusion equation however, we must take care to use an approximation to the first order spatial derivative that has a truncation error of second order. Otherwise the truncation error in this term will completely dominate seeing as $\Delta t < \Delta x$. We must also find a new stability criterion for the scheme.
As for now, the Leap-frog discretization will do (though it is not by far a perfect choise seeing as it is unstable). As a sidenote we can also note that the Neumann boundary condition will be very clear in this scheme. $\frac{\partial C}{\partial n} = 0 \implies \frac{\partial C}{\partial x} = 0$ on the boundary, leading to $C_{-1} = C_1$ on the boundary and cancelling the drift term on the boundary.

$$C^{n+1} = \frac{D\Delta t}{\Delta x^2}\left(C_{i+1}^n - 2C_i^n + C_{i-1}^n\right) - \frac{v\Delta t}{2\Delta x}\left(C_{i+1}^n - C_{i-1}^n\right) + C^n \tag{3.5}$$

The truncation error for the first order derivative using the Leap-frog scheme is obtained as follows

$$
\begin{aligned}
u(t, x + \Delta x) &= u(t, x) + \frac{\partial u(t, x)}{\partial x}\Delta x + \frac{\partial^2 u(t, x)}{2\partial x^2}\Delta x^2 + \mathcal{O}(\Delta x^3) \\
u(t, x - \Delta x) &= u(t, x) - \frac{\partial u(t, x)}{\partial x}\Delta x + \frac{\partial^2 u(t, x)}{2\partial x^2}\Delta x^2 + \mathcal{O}(\Delta x^3)
\end{aligned}
$$

which we recognize as the same error we got from the second order spatial derivatives.
If we now do the same analysis as we have already done, by finding a manifactured solution to the convection diffusion equation, implementing a numerical scheme like 3.5 to solve it after and taking the errornorm.
The errornorm does not have to go as Δt, but it must be halved (approximately) if we halv Δt. Figure **??** shows two simulations of equation 2.15 for $D = 1$ and $v = 1$ compared to the manifactured solution 3.1 without adding walkers. Before the simulation we must find a source term so the manifactured solution will solve the equation.

$$
\begin{aligned}
-\pi^2 \exp\left(-\pi^2 t\right)\cos(\pi x) &= -\pi^2 D \exp\left(-\pi^2 t\right)\cos(\pi x) + \pi v \exp\left(-\pi^2 t\right)\sin(\pi x) + f(x, t) \\
-\pi^2 \cos(\pi x) &= \pi^2 D \cos(\pi x) + \pi v \sin(\pi x) + \tilde{f}(x) \\
\tilde{f}(x) &= -\pi \sin(\pi x)
\end{aligned}
$$

Where $f(x, t) = \exp\left(-\pi^2 t\right)\tilde{f}(x)$.

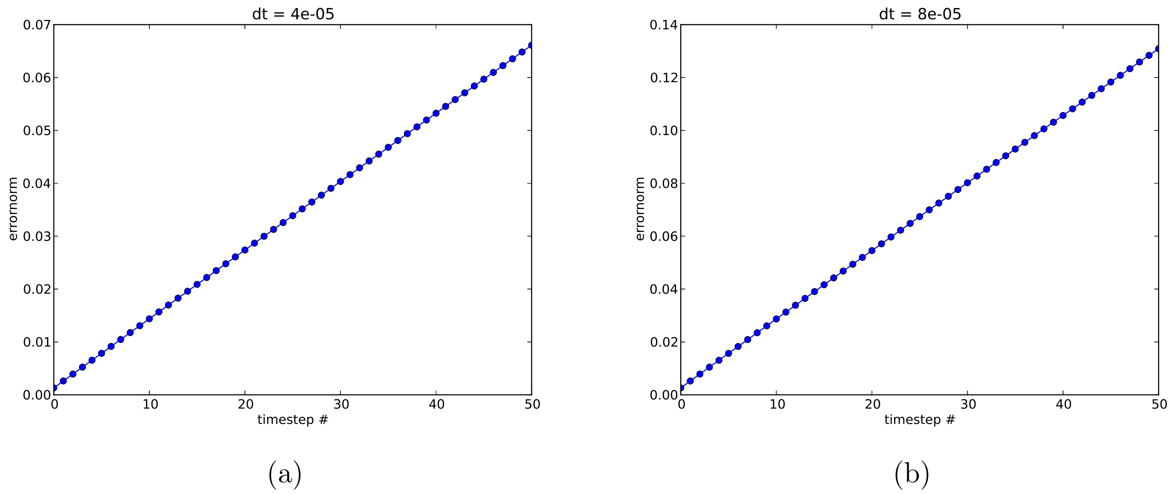

Figure 3.4: Verification of Convection diffusion equation implenetation

As we see from figure 3.4 the errornorm is halved when Δt is halved, just as we expected. We can then advance to testing the effect of adding an area of walkers for different values of Hc as before. Note that we have now added some drift to the walkers so the model will fit better.

3.3 2D

Doing the same tests is 2D gives slightly different results; adding a 2D walk-domain has an influence on the error, but a rather small one. This can, however be tweaked by increasing the conversion parameters.

Chapter 4

Results

Bibliography

[1] L Farnell and WG Gibson. "Monte Carlo simulation of diffusion in a spatially nonhomogeneous medium: A biased random walk on an asymmetrical lattice". In: *Journal of Computational Physics* 208.1 (2005), pp. 253–265.

[2] Charles Nicholson. "Diffusion and related transport mechanisms in brain tissue". In: *Reports on progress in Physics* 64.7 (2001), p. 815.