Summer Project 2021: Keller-Segel Model of Chemotaxis

Mentor: Professor Dond Asha Kisan, School of Mathematics, Indian Institute of Science Education and Research, Thiruvananthapuram

Ira Zibbu

May-July 2021

1 Objective

Chemotaxis is the phenomenon of directed motion of cells up or down a chemical gradient. Cells utilise chemotaxis to navigate complex environments to locate regions that are ideal for survival and reproduction. Various mathematical models have been proposed to describe chemotaxis, the most popular being the Keller-Segel Model. This project aims to explore the formulations of the Keller-Segel model and its derivatives, solutions and their behaviours, obtaining numerical solutions using finite difference schemes and interpreting them in a biologically meaningful way.

2 Introduction

Cells are constantly surrounded by heterogeneous, fluctuating local environments. In this context, the ability to locomote to favourable regions represents an advantage. Chemotaxis is defined as any cell motion that is affected by a chemical gradient in a way that results in net propagation along that chemical gradient. If motion is up a chemical gradient, then the chemical is a termed a chemoattractant, and if motion is down a chemical gradient, then the chemical is termed a chemorepellant. Its two characteristic features are directed motion, and movement either up or down a gradient. This is in contrast with chemokinesis, which is an undirected motion in response to a chemical stimulus, and chemotropism, which is the growth of a part of an organism towards/away from a chemical stimulus [1]. Chemotaxis is critical for survival and is a nearly universal phenomenon among motile prokaryotes and eukaryotes. It fulfils a variety of functions such as searching for optimal environments and resources, colonisation of new areas, cell aggregation and other social activities. In mutlicellular organisms, chemotaxis enables cell movements during development, differentiation, growth and cell migration in vasculature.

The process of chemotaxis involves signal perception, processing, integration and a response that alters cell movement. Chemoreceptors that are responsible for detecting particular chemicals can be transmembrane or cytoplasmic. Signal perception is followed by transduction, amplification and activation of an effector molecular that interacts with the locomotory machinery to induce a change in motion.

In humans, chemotaxis directs the motion of sperm cells towards the ovum, organises cells for events in development such as gastrulation, allows fibroblasts to locate wounds etc. Cancerous cells also utilise chemotaxis to invade surrounding tissue and to initiate angiogenesis [6].

3 Chemotaxis in some model organisms

3.1 Escherichia coli

E. coli is a gram-negative bacteria that is present in the soil and animal gastrointestinal tracts. Being a model organism, the signalling pathways and mechanisms associated with chemotaxis in *E. coli* are well characterized. The flagella of this bacterium are responsible for its motility, and the arrangement of its flagella is peritrichous [15]. When the flagellar motors rotate counterclockwise, the flagella form a bundle that propels the cell forward in a straight line. When some, or all flagellar motors rotate clockwise, the flagella unbundle and the cell tumbles in place. In the absence of chemical gradients, the movement of *E. coli* is modelled as a random walk, reflecting the combination of straight runs and tumbles it performs. When a chemical gradient is present, the random walk is biased such that it results in a net propagation in the direction of, or against, the chemical gradient [9]. On average, bacterial cells change direction every second by tumbling. A higher tumbling frequency causes the cell to steer away from its current direction while a lower frequency enables to cell to maintain longer straight runs in current direction.

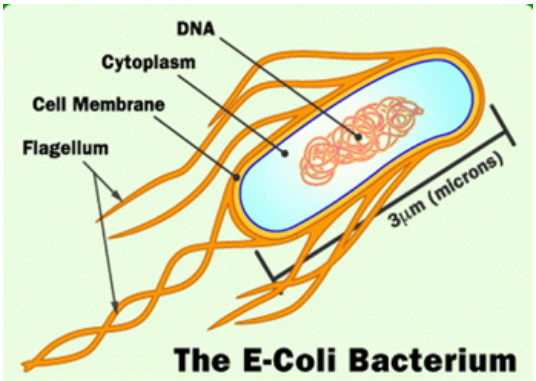

Figure 1: Structure of *E. coli* [18]

The sensitivity of the chemoreceptors of bacteria is remarkable– they have the ability to detect changes as small as a few molecules in the volume of the cell [19]. As bacterial cell sizes tend to be a few micron, they rely on temporal sensing of chemical gradients rather than measuring absolute concentrations along their lengths [12]. The receptors can be thought of as possessing 'memory' wherein they sense chemical concentrations relative to a previous time point.

In *E. coli*, chemical signal perception is performed by transmembrane methyl accepting proteins (MCPs). The proteins downstream to the MCPs are a system of chemotaxis proteins (Che proteins). CheA is a dimeric, cytosolic histidine protein kinase, which is associated with the MCPs via an adaptor CheW. Two response regulators, CheY and CheB bind with CheA. CheY is a single-domain protein that binds to flagellar motor proteins to induce clockwise rotation. CheB is a methylesterase that plays a role in the adaptation of the MCPs. A decreasing concentration of an attractant decreases ligand binding to the MCPs, resulting in the trans-autophosphorylation of histidine residues of the monomers of CheA. CheY is phosphorylated to CheY-P, which acts as the effector molecule by reversing the direction of rotation of the flagellar motors. The cell now tumbles at a greater frequency,

resulting in direction change. The CheY-P signal is terminated by phosphotase CheZ. In addition to phosphorylating CheY, CheA also phosphorylates CheB. CheB-P exhibits increased methylesterase activity and demethylates the MCPs in order to adapt the receptors. Demyethylated MCPs show decreased ability to induce autophosphorylation in CheA, which returns the cell to its pre-stimulus state. The cell can now respond to further increments to decrements in the chemical concentrations. The opposite occurs when attractant concentrations increase. Autophosphorylation of CheA is inhibited, CheY-P concentrations decrease, the rate of reversal of flagella decreases, the cell tumbles less frequently and tends to move in a straight line. CheB activity is also reduced, allowing the constitutive methyltransferase CheR to methylate the MCPs. Methylated MCPs show increased ability to stimulate autophosphorylation in CheA, allowing further elevations in attractant concentration to induce the pathway all over again [20].

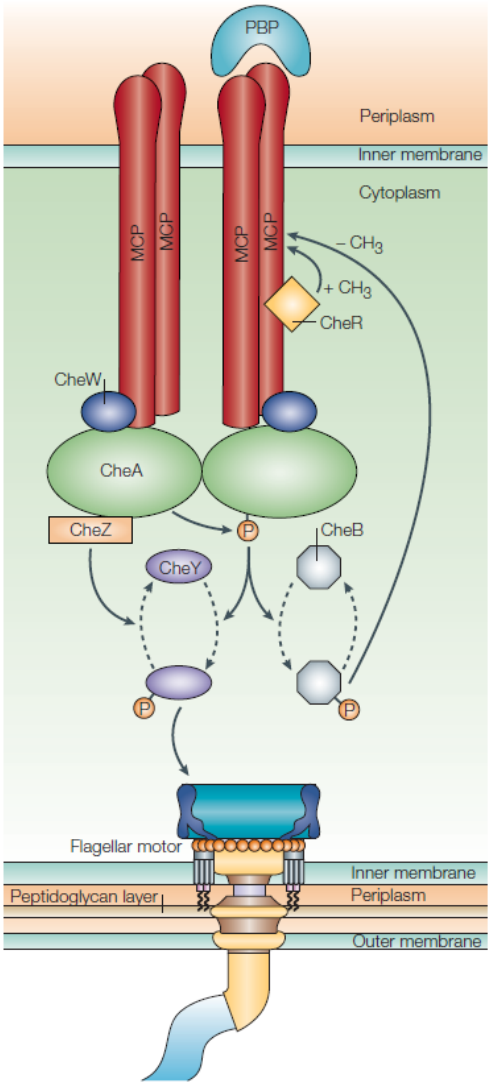

Figure 2: A schematic diagram representing the chemosensory pathway in *E. coli* [20]

E. coli has five chemoreceptors T_{ar} (aspartate and maltose chemoreceptor), T_{sr} (serine, leucine chemoreceptor), T_{rg} (ribose and galactose chemoreceptor), T_{ap} (dipeptide chemoreceptor), and *Aer* (oxygen). The cytoplasmic domains of these receptors are highly conserved, even across bacterial species, while their periplasmic domains are variable and adapted to the specific target ligands.

3.2 Dictyostelium discoiduem

Dictyostelium discoideum, commonly referred to as slime mould, is a species of soil-dwelling amoeba. The life cycle of *D. discoideum* begins as a spore, which germinates into the amoeba. The amoeba feed on bacteria in their surroundings and divide by mitosis during this vegetative state. Once food sources are depleted, starvation initiates the aggregation phase where cells aggregate to a central location to form a cell mass called a slug. The completion of aggregation initiates the migration phase, and this motile slug, comprising of over 100,000 cells, moves in search of a suitable environment. Finally, the slug differentiates to form a fruiting body which releases spores upon maturation, thus completing the life cycle.

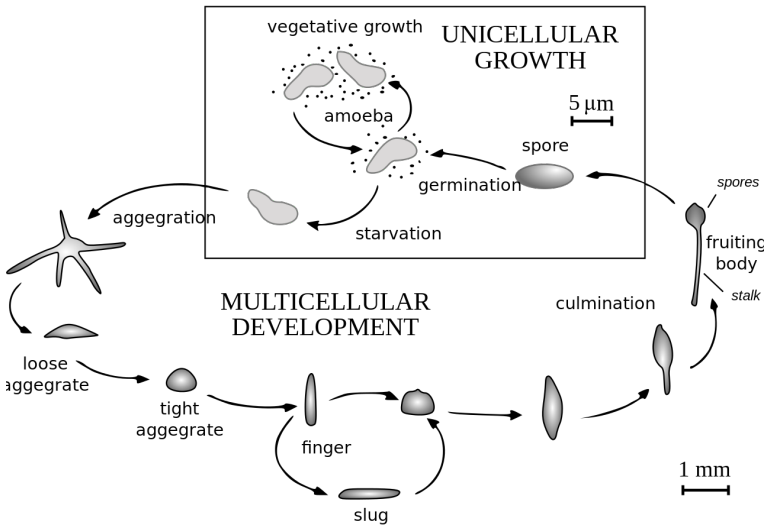

Figure 3: Life cycle of *D. discoideum*

The aggregation phase is of particular interest as it involves the chemotactic locomotion of individual cells in response to a self-secreted chemoattractant, cyclic adenosine monophosphate (cAMP). During chemotaxis, the cells become elongated, with a leading and trailing edge. This asymmetric shape allows for more effective response to chemical cues [21]. Periodic waves of cAMP propagate as pulses moving away from the aggregation centre at a speed of around $300\mu/min$. Each pulse stimulates movement of the amoeba at a speed of $25\mu/min$ for $100s$. Movement stops after the cAMP pulse has passed and the cells remain at rest until the next pulse arises [7].

4 Keller-Segel Model

This justification has been adopted from [8] and [6]. The model assumes that both the cells and the chemoattractant move via diffusion through the medium. This is distinct from the biased random walk that is used to model *E. coli* but over large spatial scales, the two modes of movements can be considered the same. Although cells are discrete and distinct entities, when they aggregate, as in chemotaxis and the distances between them is comparable to the cell size, they can be regarded as existing in a smooth continuum and differential equations describing cell concentration can be used.
Let $u(x,t)$ denote the mass of cells and $v(x,t)$ denote the concentration of the chemoattractant at a point x at a time t, where $x \in \Omega$, Ω is a bounded subset of $\mathbb{R}^n$ and $t > 0$. Here the cells both produce the chemoattractant and move up the chemoattractant gradient. Let the chemoattractant be produced per cell at a rate of $f(v)$ per cell.

There is an extracellular enzyme that degrades the chemoattractant. The concentration of the enzyme is denoted by $p(x,t)$ and is produced by the cells at a rate $g(v,p)$. The reaction of the degradation of the chemoattractant is represented as follows

$$v + p \underset{r_{-1}}{\overset{r_1}{\rightleftarrows}} \epsilon \xrightarrow{r_2} p + \text{degraded product}$$

Where ϵ is the enzyme-chemoattractant complex whose concentration is $\eta(x,t)$ and r_1, r_{-1}, r_2 represent the rate constants of the respective reactions.

Let $J^{(u)}(x,t)$ denote the flux density of cells. $J^{(u)}(x,t)$ comprises of a diffusive flux (in accordance with Fick's first law) and an advective flux (analogous to Fourier's law of heat conduction).

$$J^{(u)}(x,t) = k_2(x,t)\nabla v - k_1(x,t)\nabla u \tag{1}$$

The chemoattractant, enzyme and enzyme complex diffuse according to Fick's law. Thus the equations for the respective flux densities are

$$\begin{aligned} J^{(v)}(x,t) &= k_c \nabla v \\ J^{(\eta)}(x,t) &= k_\eta \nabla \eta \\ J^{(p)}(x,t) &= k_p \nabla p \end{aligned} \tag{2}$$

The rate of change of any concentration c is given by

$$\frac{\partial c}{\partial t} = -\nabla . J^{(c)}$$

For some control volume D, the conservation of mass equations are represented as follows:

$$\frac{d}{dt}\int_D u(x,t)dx = -\int_{\partial D} J^{(u)}(x,t).n(x).dS \tag{3}$$

$$\frac{d}{dt}\int_D v(x,t)dx = Q^{(v)}(D,t) - \int_{\partial D} J^{(v)}(x,t).n(x).dS \tag{4}$$

Where $Q^{(v)}(D,t)$ denotes the quantity of chemoattractant produced per unit volume and time.

An additional term to model cell division and death $k_4(u,v)$ is added. Upon imposing a Neumann zero-flux boundary condition, the following system of equations is generated $\forall x \in \Omega, t > 0$

$$\begin{aligned} u_t &= \nabla(k_1(u,v)\nabla u - k_2(u,v)\nabla v) + k_4(u,v) \\ v_t &= k_c \Delta v - r_1 v p + r_{-1}\eta + u f(v) \\ p_t &= k_p \Delta p - r_1 v p + (r_{-1} + r_2)\eta + u g(v,p) \\ \eta_t &= k_\eta \Delta \eta + r_1 v p - (r_{-1} + r_2)\eta \\ \frac{\partial u}{\partial n} &= 0, \frac{\partial v}{\partial n} = 0, \frac{\partial p}{\partial n} = 0, \frac{\partial u}{\partial n} = 0 \\ u(0,x) = u_0(x), v(0,x) &= v_0(x), p(0,x) = p_0(x), \eta(0,x) = \eta_0(x) \end{aligned} \tag{5}$$

This represents the most general form the equations describing chemotaxis. A majority of models can be derived from these equations by selecting the appropriate functions k_j.

Further applying the assumptions that the enzyme-complex is in a steady state, the total concentration of the free and bounded enzyme is constant, and there is no cell division or death, then two coupled differential equations are obtained that represent the Keller-Segel model in its standard form.

$$
\begin{aligned}
u_t &= \nabla(k_1(u,v)\nabla u - k_2(u,v)\nabla v) \\
v_t &= k_c\Delta v - k_3(u,v) + k_5(u,v) \\
\frac{\partial u}{\partial n} &= 0, \frac{\partial v}{\partial n} = 0 \\
u(0,x) &= u_0(x), v(0,x) = v_0(x)
\end{aligned} \tag{6}
$$

$k_1(u,v)$ is the diffusion coefficient or motility coefficient for the cells, $k_2(u,v)$ is the chemotactic sensitivity or chemotactic coefficient. $k_c(u,v)$ is the diffusion coefficient of the chemoattractant and depends on the nature of the chemoattractant and the surrounding medium. $k_3(u,v)$ and $k_5(u,v)$ denote the degradation and production of the chemoattractant respectively.

4.1 Possible Forms for the Parameters

The next task is to identify candidate functions for the parameters k_j. Many approaches have been utilised to determine these parameters; in this section we restrict ourselves to empirical studies that have been conducted on the topic. The equations (6) can be better represented as follows:

$$
\begin{aligned}
u_t &= \nabla(\mu\nabla u - \chi\nabla v) \\
v_t &= \nabla(D\nabla v) + f(u,v) - g(u,v)
\end{aligned} \tag{7}
$$

Where μ is motility coefficient, χ chemotactic coefficient, D is the diffusion coefficient. For a given chemoattractant, if the substrate is homogeneous and isotropic, then D is a constant that can be experimentally determined and is generally well-defined in existing literature. For example, the D for glucose is $500\mu m^2/s$ [4]. The cell motility coefficient μ has also been described as a constant for bacteria [14]. Specifically, μ is given by

$$\mu = \frac{1}{2}Ts^2$$

where s is the mean cell speed; T is the mean persistence time (the average time period for which a cell moves in a given direction before making a significant direction change)[16]. The assumption made here is that cell speed remains constant irrespective of the chemical gradient. Experiments have shown that μ depends on the local concentration of the chemoattractant [11], suggesting that μ has a functional dependence on v rather than being a constant. Nonetheless, approximating the cell motility coefficient to a constant is useful as it is a measurable quantity. For *Pseudomonas flourescens* in the absence of a chemoattractant, μ is $60\mu m^2/s$, for motile strains of *E. coli* in the absence of a chemoattractant, μ is $8\mu m^2/s$ [16]. For rat alveolar macrophages, in the absence of C5a (complement component), μ is $0.011\mu m^2/s$, and is $0.019\mu m^2/s$ at $10^{-11}M$ C5a [2].

The chemotactic coefficient χ is hypothesised to have a dependence on v as follows [16]

$$\chi(v) = \frac{K_d T s^2 \sigma}{(K_d + v)^2} = \frac{\chi_0 K_d}{(K_d + v)^2} \tag{8}$$

where $\chi_0 = Ts^2\sigma$. K_d is the receptor-chemoattractant equilibrium dissociation constant, and σ is the decrease in the probability of change in the mean direction for a unit change in the rate of change of fractional bound enzyme. Experiments done on bacteria have shown that χ_0 more or less remains a constant over different concentrations of the chemoattractant. For example, in the response of *E. coli* to αmethylaspartate, $\mu = 15\mu m^2/s$, $\chi_0 = 750\mu m^2/s$ and $K_d = 10^{-4}M$ [2].

Not much is known about the functional form of $f(u,v)$, the rate of production of chemoattractant by the cells. It is often represented as $f(u,v) = a_1 u$ where a_1 is the rate of production of the per cell, implying that the chemoattractant is produced at a constant rate per cell [14]. This is not biologically accurate as the production of the chemoattractant is expected to reduce at high u and v. For the function $g(u,v)$ that describes the degradation of the chemoattractant, utilising Michaelis-Menten kinetics is appropriate

$$g(u,v) = \frac{V_{max} v}{K_m + v} \tag{9}$$

Where V_{max} is the maximum velocity and K_m is Michaelis constant, both of which relate to the enzyme kinetics and are constants for a given substrate-enzyme couple.

While these forms of the parameters are reflective of biological processes, their non-linear nature introduces difficulties in the mathematical analysis of the Keller-Segel model. For the remainder of this project, we will deal with variations of the Keller-Segel model that may appear to be biologically naive, but can still offer insight into the phenomenon of chemotaxis.

4.2 The Minimal Model

As its name suggests, the minimal model is the simplest model and is given below [6]. It is derived from 6 using non-dimensionalization.

$$\begin{aligned} u_t &= \nabla(\mu \nabla u - \chi u \nabla v) \\ v_t &= \Delta v + u - v \end{aligned} \tag{10}$$

In one dimension, (10) admits a global solution that is uniformly bounded [6] [13]. Thus we can write $\forall x \in [0,1], t > 0$

$$\begin{aligned} u_t &= (\mu u_x - \chi u v_x)_x \\ v_t &= v_{xx} + u - v \end{aligned} \tag{11}$$

We apply the following initial value conditions and boundary condition

$$u(x,0) = u_0(x), v(x,0) = v_0(x) \tag{12}$$

$$v_x|_{x=0,1} = 0 \tag{13}$$

$$[\mu u_x - \chi u v_x]|_{x=0,1} = 0 \tag{14}$$

Condition (14) is equivalent to $u_x|_{x=0,1} = 0$ because $v_x|_{x=0,1} = 0$

4.3 The Signal Dependent Sensitivity Model

The cells rely on receptors to determine local concentrations of chemoattractants. The minimal model technically allows for 'infinite sensing' and does not relfect the fact that above a particular threshold concentration of v, the receptors are fully occupied and are unable to sense further increases in v. The sensitivity dependant model incorporates this phenomenon of saturation of receptors:

$$\begin{aligned} u_t &= \nabla(\mu \nabla u - \frac{\chi u}{(1+\alpha v)^2} \nabla v) \\ v_t &= \nabla^2 v + u - v \end{aligned} \tag{15}$$

α is a parameter that is determined by the receptor-ligand binding kinetics. As expected, at high concentrations of v, the receptors are occupied and are unable to resolve changes to the concentration gradient. (15) also shows that the contribution by the chemotaxis term depends on the gradient of the fraction of bound receptors, rather than the local gradient of v, which is a more accurate descriptor of chemotaxis in general.

4.4 The Cell Kinetics Model

Generally, chemotaxis occurs at a faster rate than cell division, hence the term modelling cell death and growth is often neglected. However, in certain contexts where the movement is slow enough, adding terms describing population growth/death becomes necessary. The choice of the functional form of the population kinetics depends on the biological system. One of them is elaborated below:

$$\begin{aligned} u_t &= \nabla(\mu\nabla u - \chi u\nabla v) + ru(1-u) \\ v_t &= \Delta v + u - v \end{aligned} \tag{16}$$

This equation has a single, non-trivial steady state, and with the limit $r \longrightarrow 0$, the model reduces to the minimal model [6].

5 Finite Difference Methods

5.1 Explicit Finite Difference Scheme for the Minimal Model

The system of equations (11) can be represented via an explicit finite difference scheme. Assuming $u(x,t)$ and $v(x,t)$ are sufficiently smooth, a grid of step h is introduced for the spatial variable x and step τ for time variable t. Let $N, K \in \mathbb{N}$ such that $h = 1/N$ and $\tau = 1/K$. The grid points are represented as follows

$$x_i = ih, i = \{0, 1....N\}$$
$$t_n = n\tau, n = \{0, 1....K\}$$

The derivatives are approximated as follows

$$\frac{{u_i}^{n+1} - {u_i}^n}{\tau} = \mu\frac{{u^n}_{i+1} - 2{u_i}^n + {u^n}_{i-1}}{h^2} - \chi\frac{{u^n}_{i+1} - {u^n}_i}{h}\frac{{v^n}_{i+1} - 2{v_i}^n + {v^n}_{i-1}}{h^2} \tag{17}$$

$$\frac{{v_i}^{n+1} - {v_i}^n}{\tau} = \frac{{v^n}_{i+1} - 2{v_i}^n + {v^n}_{i-1}}{h^2} + {u_i}^n + {v_i}^n \tag{18}$$

Rewriting (17) and (18) with ${u_i}^{n+1}$ and ${v_i}^{n+1}$ as the subjects respectively

$${u^n}_{i+1} = \mu F({u^n}_{i+1} - 2{u^n}_i + {u^n}_{i-1}) - \chi F(1/h)({u^n}_{i+1} - {u^n}_i)({v^n}_{i+1} - 2{v^n}_i + {v^n}_{i-1}) + {u^n}_i \tag{19}$$

$${v_i}^{n+1} = F{v^n}_{i+1} + (\tau - 1 - 2F){v^n}_i + F{v^n}_{i-1} + \tau{u^n}_i \tag{20}$$

where $F = \tau/h^2$
From the initial condition (13)

$$\begin{aligned} \frac{{v^n}_1 - {v^n}_{-1}}{h} = 0 &\implies {v^n}_1 = {v^n}_{-1} \\ \frac{{v^n}_{N-1} - {v^n}_{N+1}}{h} = 0 &\implies {v^n}_{N-1} = {v^n}_{N+1} \end{aligned} \tag{21}$$

Similarly from condition (14)

$$\begin{aligned} \frac{{u^n}_1 - {u^n}_{-1}}{h} = 0 &\implies {u^n}_1 = {u^n}_{-1} \\ \frac{{u^n}_{N-1} - {u^n}_{N+1}}{h} = 0 &\implies {u^n}_{N-1} = {u^n}_{N+1} \end{aligned} \tag{22}$$

The numerical algorithm starts with the following computation

$$\begin{aligned}\mathbf{v}^0 &= [v^0{}_0, v^0{}_1 .... v^0{}_N], v^0{}_i = v_0(x_i) \\ \mathbf{u}^0 &= [u^0{}_0, u^0{}_1 .... u^0{}_N], u^0{}_i = u_0(x_i)\end{aligned} \tag{23}$$

Then starting with $n = 0$, knowing u^n and v^n, calculate u^{n+1} and v^{n+1} recursively till $n = K - 1$. At each time level, evaluate (20) from $i = 0$ till $i = N$, making use of (21) at the boundaries. Similarly then evaluate (19) from $i = 0$ till $i = N$, making use of (22).

In order for the solution obtained to be meaningful, it should satisfy two criteria- the conservation of total mass of the cells and that u and v should be non-negative throughout the domain. Mathematically, these criteria can be written as:

$$\int_0^1 u_0(x)dx = \int_0^1 u(x,t)dx, t > 0$$

$$u_0(x) \geq 0 \implies u(x,t) \geq 0$$

$$v_0(x) \geq 0 \implies v(x,t) \geq 0$$

Furthermore, the finite difference scheme should be stable. Often explicit schemes suffer from stability issues and have restrictions on the values of the mesh size and parameters to ensure stability [10]. There is no proof that the given scheme satisfies these criteria or is stable, although simulations suggest that $\mu < \chi$ may be necessary for stability. The MATLAB code is given below:

```
function [u,v] = KS_explicit (m,c,dx,dt,T)
%solve the minimal keller-segel model. m is parameter \mu, c is \chi, dx is
%the space mesh size, dt is the mesh size for time and T is the final
%time. The initial conditions are u(x,o) = (x^2)/2 - (x^3)/3 and u_x(x,t) =
%v_x(x,0) = 0 for x = 1 and 0.
t = [0:dt:T]; %creating mesh points in time domain
x = [0:dx:1];%creating mesh points in the space domain
Nx = size(x,2); %no of mesh points in time
Nt = size(t,2);% no of mesh points in space
u = zeros(Nt,Nx); %stores solution for cell concentrations
v = zeros(Nt,Nx); %stores solution for attractnt concentration
F = dt/(dx^2);
for i = 1:Nx %loop to set the initial conditions at time level 0
    u(1,i) = ((x(i))^2)/2 - ((x(i))^3)/3;
    v(1,i) = 0;
end
for j = 2:Nt %loop to compute the approximate solution
    for k = 1:Nx
    if k == 1
        u(j,k)=m*F*(2*u(j-1,k+1) - u(j-1,k)) - c*F*(1/dx)*(u(j-1,k+1) -u(j-1,k))
   *(2*v(j-1,k+1) - 2*v(j-1,k)) + u(j-1,k);
        v(j,k) = 2*F*v(j-1,k+1)+ (dt-1-2*F)*v(j-1,k) + dt*u(j-1,k);
    end
    if k == Nx
        u(j,k)=m*F*(2*u(j-1,k-1) - u(j-1,k)) - c*F*(1/dx)*(u(j-1,k-1) -u(j-1,k))
   *(2*v(j-1,k-1) - 2*v(j-1,k)) + u(j-1,k);
        v(j,k) = 2*F*v(j-1,k-1)+ (dt-1-2*F)*v(j-1,k) + dt*u(j-1,k);
    end
    if ~(k == 1 | k == Nx)
         u(j,k)=m*F*(u(j-1,k+1) - u(j-1,k) + u(j-1,k-1)) - c*F*(1/dx)*(u(j-1,k+1)
   -u(j-1,k))*(v(j-1,k+1) - 2*v(j-1,k) + v(j-1,k-1)) + u(j-1,k);
        v(j,k) = F*v(j-1,k+1)+ (dt-1-2*F)*v(j-1,k) + dt*u(j-1,k) + F*v(j-1,k-1);
```

```
    end
    end
end
[A, B] = meshgrid(x,t);
fig_u = figure;%plots approximate solution
surf(A,B,u)
title ('cell concentration')
xlabel ('x')
ylabel ('t')
saveas(fig_u,'KS_explicit_u.png');
fig_v = figure;%plots approximate solution
surf(A,B,v)
title ('attractant concentration')
xlabel ('x')
ylabel ('t')
saveas(fig_v,'KS_explicit_v.png');
sum_u = sum(u') %stores the total sum of cell mass at each time level
sum_v = sum(v')
end
```

When run with $\mu = 0.0001$, $\chi = 0.01$, $dx = 0.1.$, $dt = 0.001$, $T = 0.01$, and setting the initial conditions $u(x,0) = \frac{x^2}{2} - \frac{x^3}{3}$ and $v(x,0) = 0$ the following result is obtained fig(4), fig(5). The solution for u appears to remain almost the same as the initial values, possibly due to the small final time T. The solution is positive everywhere. The solution for v presents as interesting oscillating solution. It attains small negative values at particular points in the domain. When T is large, the solution become unstable fig(6). All of this motivates us to seek to another finite difference scheme to solve the minimal model. A variety of implicit methods have been formulated for this purpose.

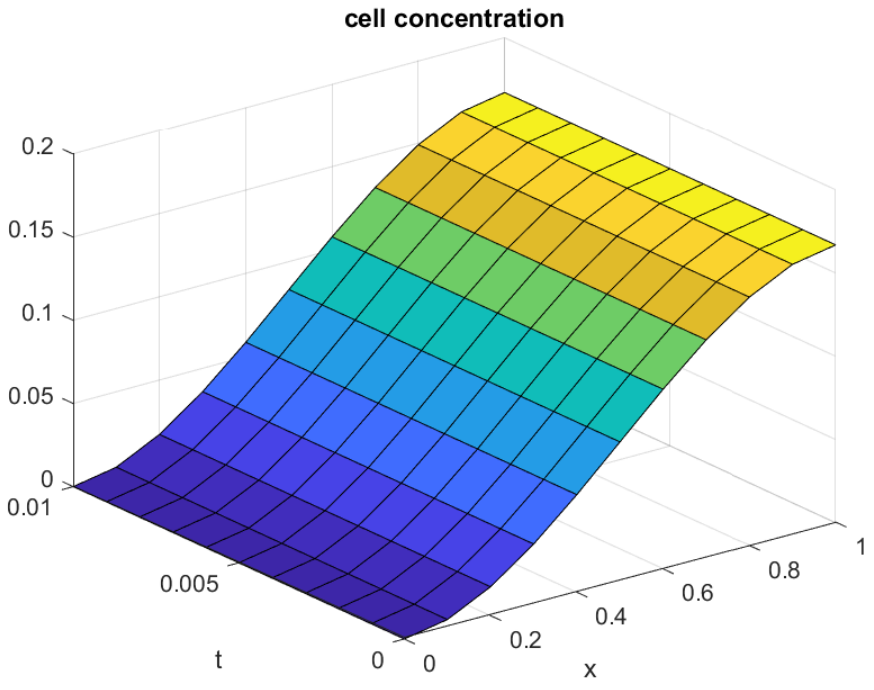

Figure 4: Plot of solution for u

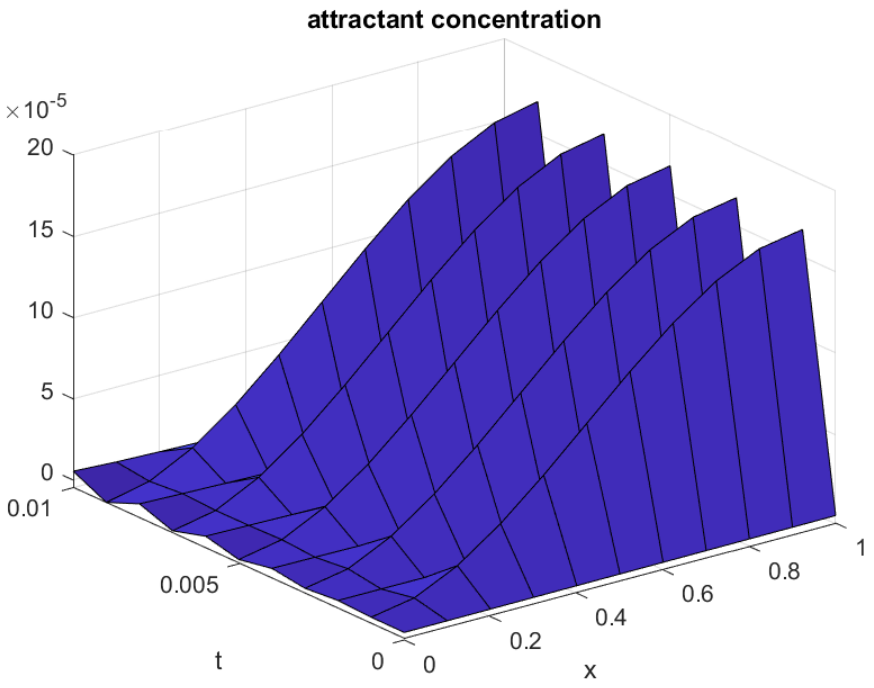

Figure 5: Plot of solution for v

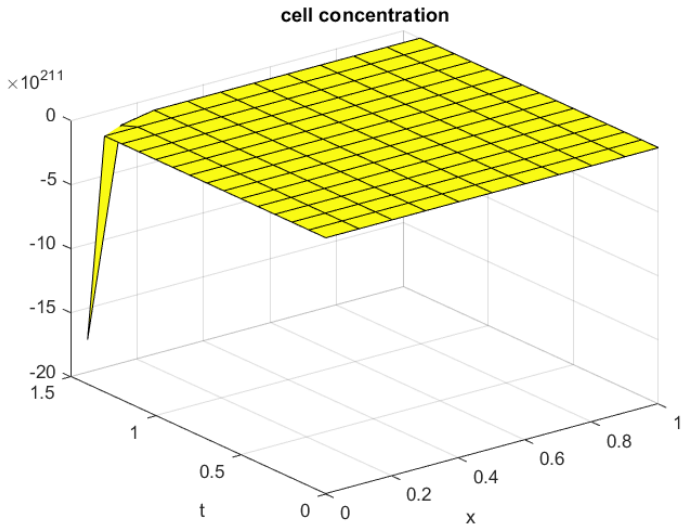

Figure 6: Plot of solution for u when T = 2

5.2 Implicit Finite Difference Scheme for the Minimal Model

The following implicit scheme is adapted from [17]. It is an upwind finite element method that has been shown to preserve the L^1 norm, as well as the positivity criteria. Let $N \in \mathbb{N}$ and set $h = 1/N$. Two kinds of grid points are introduced over the domain $[0, 1]$

$$x_i = (i - \frac{1}{2})h, i = 1, 2...N$$

$$\hat{x_i} = ih, i = 0, 1...N$$

The grid points over the time domain $[0, \infty)$

$$t_n = \tau_1 + \tau_2.. + \tau_n$$

with $\tau_j > 0 \forall 1 \leq j \leq n$

$$\mathbf{u}^n = ({u_1}^n, ...{u_N}^n)^T$$
$$\mathbf{v}^n = ({v_0}^n, ...{v_N}^n)^T$$
$$\hat{\mathbf{u}}^n = (\hat{u}_0^n, ...\hat{u}_N^n)^T$$
$$\hat{u}_i^n = \begin{cases} {u_i}^n & i = 0 \\ \frac{1}{2}({u_{i+1}}^n + {u_i}^n) & 1 \leq i \leq N-1 \\ {u_N}^n & i = N \end{cases}$$

Applying the standard theta scheme, the equation for v can be written as follows: $\theta = [0, 1], \forall 0 \leq i \leq N$

$$\frac{{v_i}^n - {v_i}^{n-1}}{\tau_n} = \theta \frac{{v^n}_{i+1} - 2{v_i}^n + {v^n}_{i-1}}{h^2} + (1-\theta)\frac{{v^{n-1}}_{i+1} - 2{v_i}^{n-1} + {v^{n-1}}_{i-1}}{h^2} + {u_i}^{n-1} + {v_i}^{n-1} \tag{24}$$

From the boundary conditions,

$${v_{-1}}^n = {v_1}^n$$
$${v^n}_{N-1} = {v^n}_{N+1}$$

In the vector form,

$$[I + \theta F_n]\mathbf{v}^n = [I - (1-\theta)F_n L]\mathbf{v}^{n-1} + \tau_n \hat{\mathbf{u}}_i^n - \tau_n {\mathbf{v}^n}_i \tag{25}$$

Where I is the $N+1 \times N+1$ identity matrix, $F_n = \tau_n/h^2$, and the $N+1 \times N+1$ tridiagnol matrix L is given by:

$$\begin{bmatrix} 2 & -2 & 0 & ... & & & 0 \\ -1 & 2 & -1 & ... & & & 0 \\ ... & & & & & & \\ 0 & 0 & -1 & 2 & -1 & ... & 0 \\ 0 & ... & & & -1 & 2 & -1 \\ 0 & ... & & & & -2 & 2 \end{bmatrix}$$

We define the flux of u as $G = \mu u_x - [\chi(v)]_x u$. We define a new quantity, $\mathbf{b}^n = ({b_i}^n, ..., {b_N}^n)^T$ where

$${b_i}^n = \frac{\chi({v_i}^n) - \chi({v_{i-1}}^n)}{h^2} \text{ where } 1 \leq i \leq N$$

Set ${b_i}^{n,+} = \max\{0, {b_i}^n\}$ and ${b_i}^{n,-} = \max\{0, -{b_i}^n\}$. Therefore ${b_i}^n$ represents $[\chi(v)]_x$ at $x = x_i$

$$G = \mu u_x + [b]_+ u - [b]_- u$$

where $[b]_{\pm} = \max\{0, \pm b\}$. Thus the flux ${G_i}^n$ of $\mathbf{u}^n$ at $x = \hat{x}_i$ is given by

$$ {G_i}^n = -\mu \frac{{u^n}_{i+1} - {u_i}^n}{h} + {b_i}^{n-1,+} {u_i}^n - {b_{i+1}}^{n-1,-} {u^n}_{i+1} \text{ where } i = 1, 2..., N-1 $$

Similarly, $\tilde{G}_i^{n-1}$ of $\mathbf{u}^{n-1}$ at $x = \hat{x}_i$

$$ \tilde{G}_i^{n-1} = -\mu \frac{{u^{n-1}}_{i+1} - {u_i}^{n-1}}{h} + {b_i}^{n-1,+} {u_i}^{n-1} - {b_{i+1}}^{n-1,-} {u^{n-1}}_{i+1} \text{ where } i = 1, 2..., N-1 $$

From the boundary conditions,

$$ {G_0}^n = 0, {G_N}^n = 0 $$
$$ \tilde{G}_0^n = 0, \tilde{G}_N^n = 0 $$

We can represent the equation for u as follows:

$$ \frac{{u_i}^n - {u_i}^{n-1}}{\tau_n} = -\theta \frac{{G_i}^n - {G^n}_{i-1}}{h} - (1-\theta) \frac{\tilde{G_i}^{n-1} - \tilde{G_{i-1}}^{n-1}}{h} \text{where } i = 1...N \tag{26} $$

Set $\mathbf{H}$ as an $N \times N$ matrix as follows

$$ H_{i,j} = \begin{cases} \mu + h{b_1}^{n,+} & i = j = 1 \\ -\mu - h{b_2}^{n,-} & i = 1, j = 2 \\ -\mu - h{b_{i-1}}^{n,+} & 2 \le i \le N-1, j = i-1 \\ 2\mu + h[{b_i}^{n,+} + {b_i}^{n,-} & 2 \le i \le N-1, j = i \\ -\mu - h{b_{i+1}}^{n,-} & i = N, j = N-1 \\ \mu + {b_N}^{n,+} & i = j = N \end{cases} $$

Equation (26) can be vectorised as follows:

$$ [I + \theta F_n \mathbf{H}^{n-1}]\mathbf{u}^n = [I + (1-\theta) F_n \mathbf{H}^{n-1}]\mathbf{u}^{n-1} \tag{27} $$

The numerical algorithm begins by defining ${u_i}^0 = u_0(x_i)$ and ${v_i}^0 = v_0(\hat{x}_i)$. For each time level, starting from $n = 1$, $\mathbf{b}^{n-1}$ is computed. The time mesh size is selected as follows:

$$ \tau_n = \min\{\tau, \frac{\epsilon h^2}{2(1-\theta)(\mu + ||\mathbf{b}^{n-1}||_\infty)}, \frac{\epsilon h}{2\theta ||\mathbf{b}^{n-1}||_\infty}\} \tag{28} $$

$$ \tau = \begin{cases} \text{an arbitrary fixed positive constant} & \theta = 1 \\ \frac{h^2}{2\mu(1-\theta)} & \theta \neq 1 \end{cases} \tag{29} $$

where $\epsilon \in (0, 1]$. Recursively compute $\mathbf{u}^n$ and $\mathbf{v}^n$ using (26) and (25) respectively. If $t_n \ge T$ finish the computation, or else renew n by $n + 1$. τ_n does not converge to 0.

6 Interpretation of the Keller-Segel Model

The last step in building mathematical models for biological processes is the interpretation of the results. Although meaningful results were not obtained here, existing work on the Keller-Segel model can be examined to see how well it captures key behaviours of chemotaxis. At its core, the model captures the tug-of-war between the tendency of cells to diffuse away from each other and their response to a self-secreted chemoattractant, which tends to gather the cells at a common location.

One of the most notable features of the Keller-Segel model is the occurrence of blow-up. A solution is said to be a blow-up solution if $\exists\ 0 < T_{\max} \leq \infty$ such that $||u(T_{\max}, x)||_\infty$ or $||v(T_{\max}, x)||_\infty$ becomes unbounded [8]. The steady state of the equations (10) is given below:

$$\begin{aligned} 0 &= \nabla(\mu\nabla u - \chi u \nabla v) \\ 0 &= \Delta v + u - v \end{aligned} \tag{30}$$

The trivial solution satisfies (30) piecewise, but the conservation of mass for u necessitates the formation of a singularity. The formation of δ function singularities is referred to as blow up. The question of if solution blows up, and if yes, then where and when, depends on the dimension and shape of Ω , initial mass and the parameters. In general, there exists a critical number D_Ω such if the total density in Ω is greater than D_Ω, then the solution blows up in finite time [14].
In $\mathbb{R}^2$, there exist radially symmetric solutions to (10) which blow up up at the centre if the following criteria is met [5].

$$\overline{u_o}\chi > 8\pi, \overline{u_o} = \frac{1}{|\Omega|}\int_\Omega u_0(x)dx$$

This is supported by simulations performed by [3]. Using a gaussian function as the intial datum,

$$u_0(x,y) = \frac{\overline{u_0}}{2\pi Z}\exp(\frac{(x-x_0)^2 - (y-y_0)^2}{2Z})$$

Setting $\Omega = [-0.5, 0.5] \times [-0.5, 0.5]$, $Z = 5 \times 10^{-3}$, $\overline{u_0}\chi = 10\pi$, $dt = 10^{-3}$ and $(x_0, y_0) = (0, 0)$, the following plot is obtained fig(7). In case of the non-symmetric solution, blowup occurs at the boundary, rather than the centre. Using the same initial values as before and with $(x_0, y_0) = (0.1, 0.1)$ (fig(8))

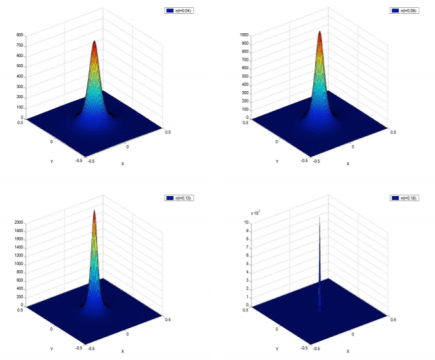

Figure 7: Simulation of blow up with symmetric solution, at t = 0.04, t = 0.09, t = 0.13, t = 0.18

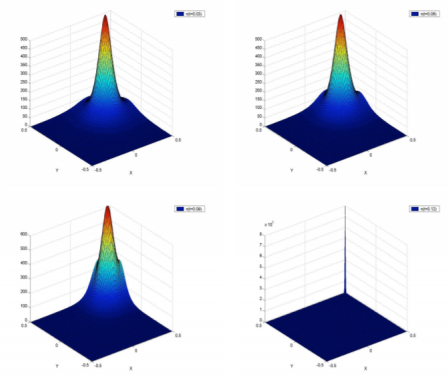

Figure 8: Simulation of blow up with asymmetric solution, at t = 0.03, t = 0.06, t = 0.09, t = 0.12

In the context of self-aggregating species like *D. discoideum* that both secrete and respond to the chemoattractant, blow-up corresponds to the stage of right before slug formation. It represents the precursor to the morphological changes accompanying the migratory phase.

7 Supplementary Work

In addition to the Keller-Segel models, finite difference schemes for some standard ODEs and PDEs were also explored. These algorithms served as a base for the algorithms in section 5.1.

7.1 Vibration ODE

The vibration ODE $u_{tt} + \omega^2 u = 0, t \in [0, T]$ is a standard differential equation that appears in many places, particularly in physics. Additionally, the initial conditions $u(0) = I$, $u_t(0) = 0$ lead to a well-defined exact solution $u(t) = Icos(\omega t)$. There are several explicit and implicit finite difference methods with varying accuracies that can be used to solve this equation. For the explicit schemes, the mesh size is restricted by the stabilty criteria $dt \leq$ These algorithms have been adapted from [10].

7.1.1 Verlet Method

The Verlet or Stromer method is an explicit method that can solve the vibration ODE. The differentials are replaced by finite differences:

$$\frac{u^{n+1} - 2u^n + u^{n-1}}{dt^2} = -\omega^2 u^n$$

Initial condition $u_t = 0 \implies u^{-1} = u^1$. The MATLAB code is given below

```
function [u,e,error_mat,t] = verlet_FD(I,w,dt,T)
%%Solves the equation u_tt = -w^2 u, u(0) = I, u'(0) = 0 the standard vibration
    ODE using the
%%stromer method or verlet method. I and w are constants of any value, dt
%%is step size and T is the final time
t = [0:dt:T];%Creating a mesh grid on the time domain of step dt
N = size(t,2)
u = zeros(1,N);%vector that will store the approximate solution
e = zeros(1,N);%vector that will store the exact solution
u(1) = I;
u(2) = u(1) - 0.5*dt*dt*w*w*u(1);
for i = 2:(N-1) %loop to calculate approximate solution
    u(i+1) = 2*u(i) - u(i-1) - dt*dt*w*w*u(i);
end
for j = 1:N %loop to compute exact solution and store it in vector e
    x = t(j);
    e(j) = I*cos(w*x)
end
error_mat = e-u
f1 = figure
plot(t,u,'b')%plots approximate solution
hold on
plot(t,e,'r' )%plot exact solution
legend('Approximation', 'exact')
xlabel('t')
ylabel('u')
hold off
```

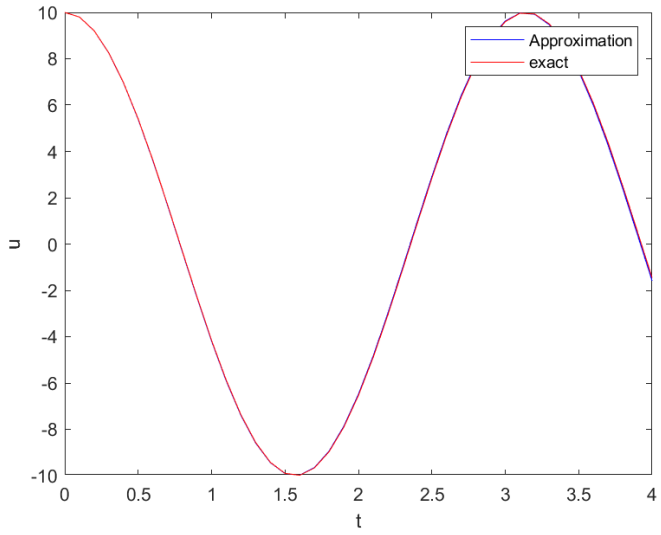

Figure 9: Plot of approximate and exact solutions for the verlet method with $I = 10$, $\omega = 2$, $dt = 0.1$ and $T = 4$

```
saveas(f1,'verlet_FD_graph.png')
end
```

The error for this method increases with increasing t, as indicated in the graph.

7.1.2 Euler Forward Method

In order to use the euler forward method, we introduce another variable $u_t = v$. The details of the method can be found in section 1.5.1 of [10]. The MATLAB code is given:

```
function [u,e,error_mat,t] = euler_fwd_vib(I,w,dt,T)
%%Solves the equation u_tt = -w^2 u, the standard vibration ODE using the
%%euler forward method
t = [0:dt:T];%Creating a mesh grid on the time domain of step dt
N = size(t,2)
u = zeros(1,N);%vector that will store the approximate solution
e = zeros(1,N);%vector that will store the exact solution
u(1) = I;
u(2) = 2*u(1)/(2+dt*dt*w*w);
for i = 2:N-1
    u(i+1) = 2*u(i) - u(i-1) - dt*dt*w*w*u(i-1);
end
for j = 1:N %loop to compute exact solution and store it in vector e
    x = t(j);
    e(j) = I*cos(w*x)
end
error_mat = e-u
maxnorm = max(error_mat)
l2norm = sqrt(dt*sum(error_mat.^2))
f2 = figure
plot(t,u,'b')%plots approximate solution
hold on
plot(t,e,'r' )%plot exact solution
legend('Approximation', 'exact')
xlabel('t')
ylabel('u')
```

```
hold off
saveas(f2,'vib_euler_fwd.png')
end
```

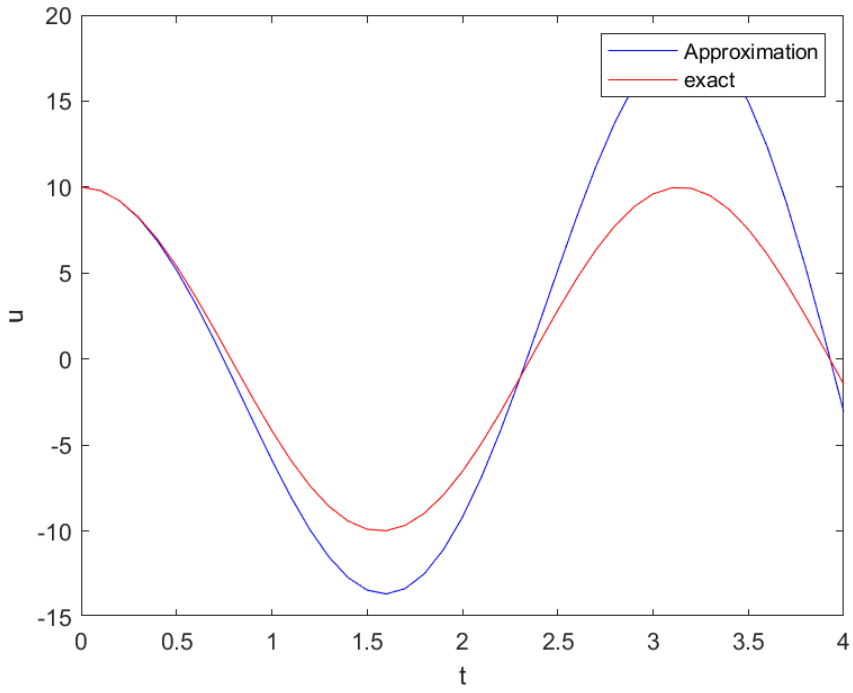

Figure 10: Plot of approximate and exact solutions for the euler forward method with $I = 10$, $\omega = 2$, $dt = 0.1$ and $T = 4$

The euler forward method results in an increase in the amplitude and an unstable solution.

7.1.3 Euler Backward Method

Similar to the euler forward method, another variable $u_t = v$ is introduced. Further details are located in section 1.5.2 of [10]. The MATLAB code:

```
function [u,e,error_mat,t] = euler_back_vib(I,w,dt,T)
%%Solves the equation u_tt = -w^2 u, the standard vibration ODE using the
%%euler backward method
t = [0:dt:T];%Creating a mesh grid on the time domain of step dt
N = size(t,2)
u = zeros(1,N);%vector that will store the approximate solution
e = zeros(1,N);%vector that will store the exact solution
u(1) = I;%initial values
u(2) = 2*u(1)/(2+dt*dt*w*w);
for i = 2:N-1
    u(i+1) = (2*u(i)-u(i-1))/(1 + w*w*dt*dt);
end
for j = 1:N %loop to compute exact solution and store it in vector e
    x = t(j);
    e(j) = I*cos(w*x);
end
error_mat = e-u;
maxnorm = max(error_mat)
l2norm = sqrt(dt*sum(error_mat.^2))
```

```
f3 = figure
plot(t,u,'b')%plots approximate solution
hold on
plot(t,e,'r' )%plot exact solution
legend('Approximation', 'exact')
xlabel('t')
ylabel('u')
hold off
saveas(f3,'vib_back_fwd.png')
end
```

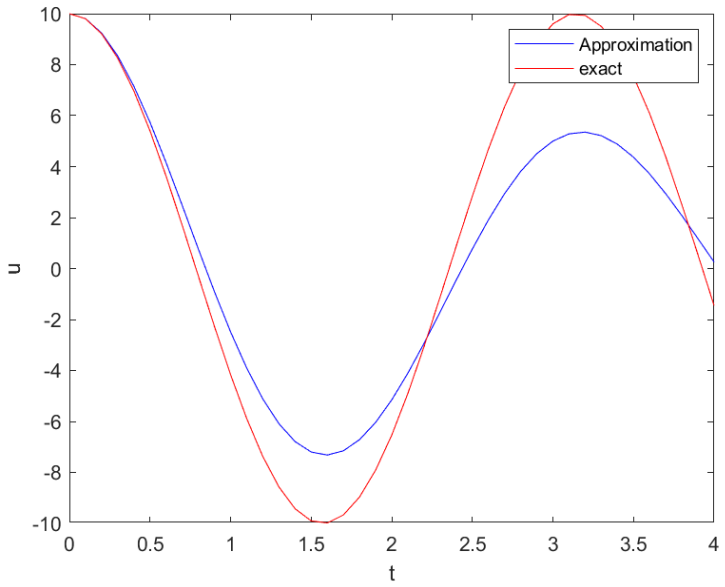

Figure 11: Plot of approximate and exact solutions for the euler backward method with $I = 10$, $\omega = 2$, $dt = 0.1$ and $T = 4$

It is evident that the verlet method outperforms both euler methods by a large margin.

7.2 Heat Equation

The heat equation (also called the diffusion equation) $u_t = Du_{xx}$ where $x \in [0, X]$, is a PDE that elegantly describes the diffusion of a substance through its surroundings. It relates the rate of change of the quantity of the substance with the change in the concentration gradient per unit of space. Here D is regarded as constant if the environment is homogeneous. We set the initial conditions to $u(x, 0) = I(x)$, $u(0, t) = 0$ and $u(0, X) = 0$. By setting $I(x) = Qsin(kx)$ and $X = \pi$ we obtain a well-defined exact solution $u(x, t) = Qsin(kx)exp(-Dk^2t)$. Having a test problem where the exact analytical solution is known allows us to check the functioning of the algorithm and measure the convergence.

7.2.1 Euler Forward Method

The euler forward method for the heat equation is an explicit finite difference scheme. The differentials of the heat equation are replaced by finite difference with a forward approximation in time.

$$\frac{{u_i}^{n+1} - {u_i}^n}{\tau} = D\frac{{u^n}_{i+1} - 2{u_i}^n + {u^n}_{i-1}}{h}$$

Since this is an explicit method, to ensure stability, $0.5 \geq \frac{\tau}{h^2}$ is necessary. The MATLAB code is given below and has been adapted from section 3.1.2 of [10].

```
function [U,E,error_mat,x,t] = diffusion_euler_fwd_sine(dt,dx,a,kp,Q,T)
%%The diffusion equation u_t = a*u_xx  where a is a scalar, x is in the closed
    interval [0,pi] with initial
%%conditions u(x,0) = I(x), u(0,t) = 0, u(1,t) = 0. k,Q are scalars and can
%%be any number

x = [0:dx:pi]; %creating mesh in space domain of step size dx
t = [0:dt:T]; %creating mesh in the time domain of step size dt
s1 = size(x);
s2 = size(t);
Nx = s1(1,2); %number of elements in x
Nt = s2(1,2); %number of elements in t
u1 = zeros(1,Nx); %creating a blank row vector to store the solution for a time
    level n
u2 = zeros(1,Nx); %creating a blank row vector to store the solution for a time
    level n+1
U = zeros(Nt,Nx); %to store the total solution
F = a*dt/(dx^2);
b = a*kp*kp;
if F >= 0.5 %stability criteria
    error('Error F value not small enough');
end
syms z;
I(z) = Q*sin(kp*z); %initial function
for i = 1:Nx %loop to assign the initial value to our solution at time level n =0
    zi = dx*(i-1);
    u1(i)= I(zi);
end
U(1,:) = u1; %assiging the solution of n= 0 time level to our solution matrix
for j = 1:(Nt-1) %loop that passes over all time levels
    for k  = 2:(Nx-1) %loop that passes over all space points for a given time
    level
        u2(k) = u1(k) + F*(u1(k+1) - 2*u1(k) + u1(k-1)); %FD formulation of
    solution
    end
    u2(1) = 0; %inital condition u(0,t) = 0
    u2(Nx) = 0;%intial condition u(1,t) = 0
    U((j+1),:)= u2;
    u1 = u2;
end
E = zeros(Nt,Nx); %empty matrix to store the exact solution
for i = 1:Nt
    ti = dt*(i-1);
    for j = 1:Nx
        xi = dx*(j-1);
        E(i,j) = Q*exp(-b*ti)*sin(kp*xi);
    end
end
error_mat = E - U; %matrix that computes the error
[A, B] = meshgrid(x,t);
fig_approx = figure;%plots approximate solution
surf(A,B,U)
title ('Approximate')
xlabel ('x')
ylabel ('t')
```

```
saveas(fig_approx, 'approx_euler_fwd_diff.png');
fig_exact = figure;%plots eact solution
surf(A,B,E)
title ('exact');
xlabel ('x')
ylabel ('t')
saveas(fig_exact, 'exact_euler_fwd_diff.png');
end
```

When run with $dx = 0.1$, $dt = 0.1$, $a = 0.005$, $Q = 10$, $kp = 10$, $T = 2$ the following results are obtained:

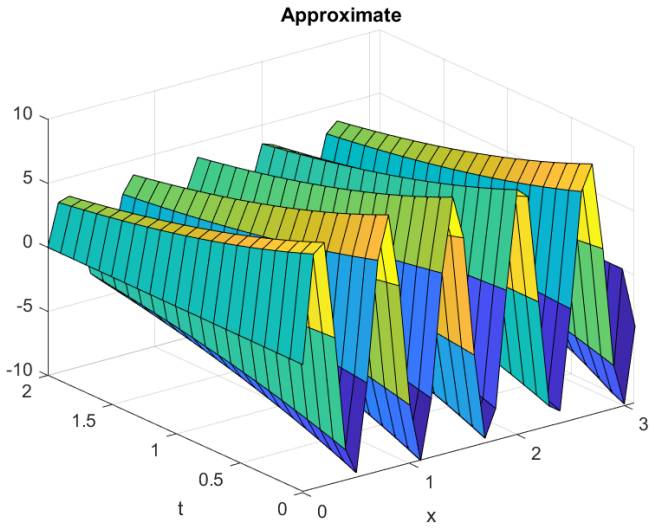

Figure 12: Plot of the approximate solution of the heat equation using the euler forward method

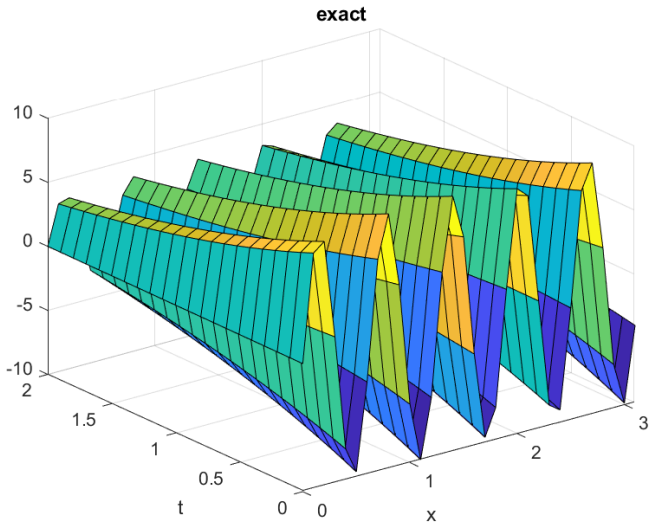

Figure 13: Plot of the exact solution of the heat equation using the euler forward method

At $t = 0$ the solution has a shape corresponding to a sine wave which 'flattens' out as t increases, as expected with the diffusion equation. There is good correspondence between the exact and approximate solution, suggesting that the algorithm works correctly.

7.2.2 Euler Backward Method

The euler backward method is an implict method that yields a system of linear equations that needs to be solved at each time level to obtain the solution. Replacing the differentials with finite differences:

$$\frac{{u_i}^n - {u_i}^{n-1}}{\tau} = D\frac{{u^n}_{i+1} - 2{u_i}^n + {u^n}_{i-1}}{h}$$

Unlike the forward method, there is no stability criteria, which makes the backward method more powerful. Complete details can be found in section 3.2.1 of [10]. The MATLAB code is given as follows:

```
function [Total_sol,E,error_mat,x,t] = diffusion_euler_back_sine(dt,dx,a,kp,Q,T)
%%The diffusion equation u_t = a*u_xx  where a is a scalar, x is in the closed
   interval [0,pi] with initial
%%conditions u(x,0) = I(x), u(0,t) = 0, u(1,t) = 0. k,Q are scalars and can
%%be any number and are paramters for the exact solution. The euler
%%backward method is an implicit scheme AU=B

x = [0:dx:pi]; %creating mesh in space domain of step size dx
t = [0:dt:T]; %creating mesh in the time domain of step size dt
s1 = size(x);
s2 = size(t);
Nx = s1(1,2); %number of elements in x (no. of mesh points)
Nt = s2(1,2); %number of elements in t (no. of mesh points)
Total_sol = zeros(Nt,Nx); %to store the total solution
F = a*dt/(dx^2);
b = a*kp*kp;
B = zeros(1,Nx); %stores the solution to n-1 time level
syms z;
I(z) = Q*sin(kp*z); %initial function
for i = 1:Nx %loop to assign the initial value to our solution at time level n =0
    zi = dx*(i-1);
    B(i)= I(zi);
end
u = zeros(1,Nx);
Total_sol(1,:) = B; %assiging the solution of n= 0 time level to our solution
   matrix
A = zeros(Nx); %tridiagnol matrix of coefficients
A(1,1) = 1;
A(1,2) = 0;
A(Nx,(Nx-1))=0;
A(Nx,Nx) = 1;
for i = 2:(Nx-1) %loop to out values for A
    A(i,(i-1)) = -F;
    A(i,i) = 2*F+1;
    A(i,(i+1)) = -F;
end
for k = 2:Nt %loop to solve for U at each time level
    U = linsolve(A,B');
    B = U';
    Total_sol(k,:) = U;
end
%[p,q] = meshgrid(0:0.1:1,0:0.1:1);
%Z = exp(-pi*pi.*q)*sin(pi.*p) + 0.1*exp(-pi*pi*10000.*q)*sin(100*pi.*p);
%surf(p,q,Z)
E = zeros(Nt,Nx); %empty matrix to store the exact solution
for i = 1:Nt
    ti = dt*(i-1);
```

```
    for j = 1:Nx
        xi = dx*(j-1);
        E(i,j) = Q*exp(-b*ti)*sin(kp*xi);
    end
end
error_mat = E - Total_sol; %matrix that computer the error
[A, B] = meshgrid(x,t);
fig_approx = figure;%plots approximate solution
surf(A,B,Total_sol)
title ('Approximate')
xlabel ('x')
ylabel ('t')
saveas(fig_approx,'approx_euler_back.png');
fig_exact = figure;%plots eact solution
surf(A,B,E)
title ('exact');
xlabel ('x')
ylabel ('t')
saveas(fig_exact,'exact_euler_back.png');
end
```

When run with $dx = 0.1$, $dt = 0.1$, $a = 0.005$, $Q = 10$, $kp = 10$, $T = 2$ the following results are obtained:

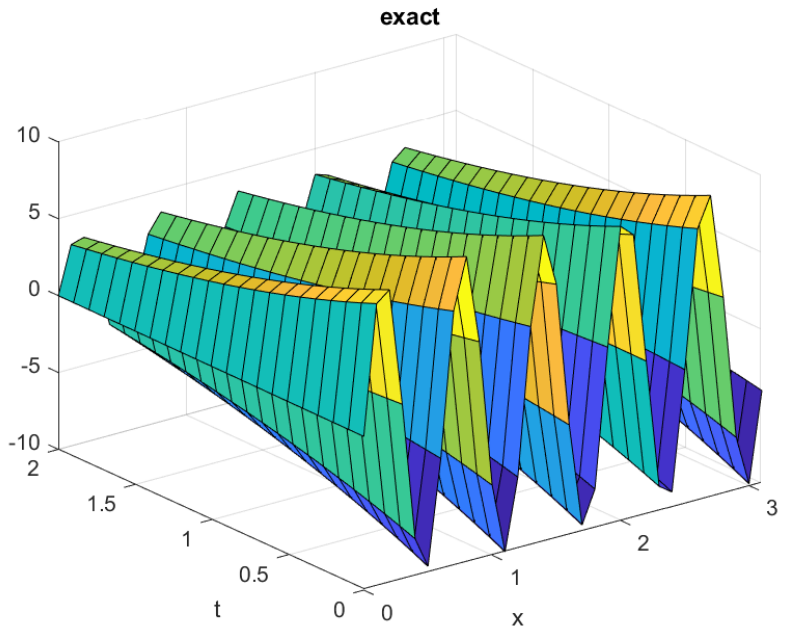

Figure 14: Plot of the exact solution of the heat equation using the euler backward method

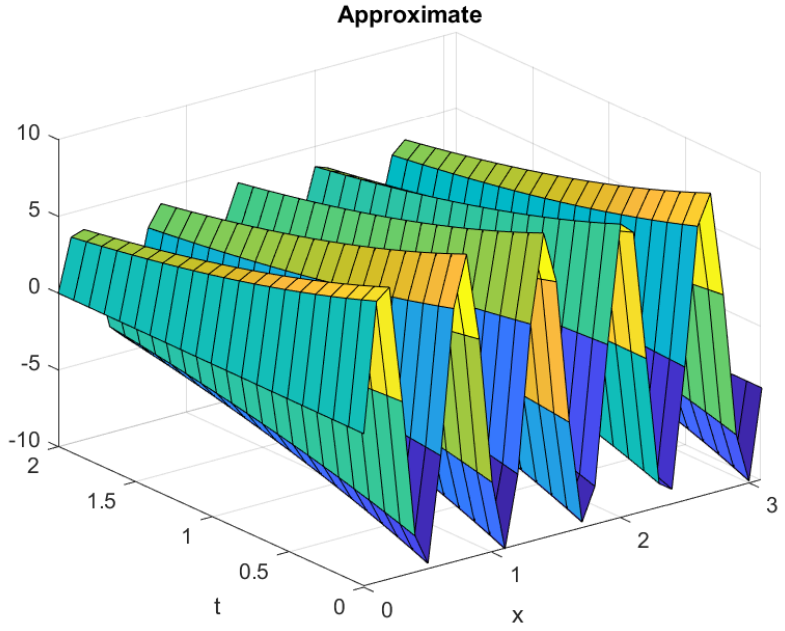

Figure 15: Plot of the approximate solution of the heat equation using the approximate backward method

References

[1] M. Eisenbach and J.W. Lengeler. *Chemotaxis*. Imperial College Press, 2004. ISBN: 9781860944130.

[2] Brian E. Farrell, Ronald P. Daniele, and Douglas A. Lauffenburger. "Quantitative relationships between single-cell and cell-population model parameters for chemosensory migration responses of alveolar macrophages to C5a". In: *Cell Motility* 16.4 (1990), pp. 279–293. DOI: https://doi.org/10.1002/cm.970160407.

[3] Francis Filbet. "A finite volume scheme for the Patlak–Keller–Segel chemotaxis model". In: *Numerische Mathematik* 104 (4 Sept. 2006). ISSN: 0029-599X. DOI: 10.1007/s00211-006-0024-3.

[4] K. Groebe, S. Erz, and W. Mueller-Klieser. "Glucose Diffusion Coefficients Determined from Concentration Profiles in Emt6 Tumor Spheroids Incubated in Radioactively Labeled L-Glucose". In: *Advances in Experimental Medicine and Biology* (1994). DOI: 10.1007/978-1-4615-1875-4_114.

[5] Miguel A. Herrero and Juan J. L. Velazquez. "A blow-up mechanism for a chemotaxis model". en. In: *Annali della Scuola Normale Superiore di Pisa - Classe di Scienze* Ser. 4, 24.4 (1997), pp. 633–683. URL: http://www.numdam.org/item/ASNSP_1997_4_24_4_633_0/.

[6] T. Hillen and K. J. Painter. "A user's guide to PDE models for chemotaxis". In: *Journal of Mathematical Biology* 58 (1-2 Jan. 2009), pp. 183–217. ISSN: 03036812. DOI: 10.1007/s00285-008-0201-3.

[7] T. Hofer et al. "Resolving the Chemotactic Wave Paradox: A Mathematical Model of Chemotaxis of Dictyostelium Amoeba". In: *Journal of Biological Systems* 03 (04 Dec. 1995). ISSN: 0218-3390. DOI: 10.1142/S0218339095000861.

[8] Dirk Horstmann. *From 1970 until present: the Keller-Segel model in chemotaxis and its consequences*. 2003.

[9] Basarab G. Hosu and Howard C. Berg. "CW and CCW Conformations of the E. coli Flagellar Motor C-Ring Evaluated by Fluorescence Anisotropy". In: *Biophysical Journal* 114 (3 Feb. 2018). ISSN: 00063495. DOI: 10.1016/j.bpj.2017.12.001.

[10] Hans Petter Langtangen and Svein Linge. *Finite Difference Computing with PDEs*. Vol. 16. Springer International Publishing, 2017. ISBN: 978-3-319-55455-6. DOI: 10.1007/978-3-319-55456-3.

[11] D Lauffenburger, C Rothman, and S H Zigmond. "Measurement of leukocyte motility and chemotaxis parameters with a linear under-agarose migration assay." In: *The Journal of Immunology* 131.2 (1983), pp. 940–947.

[12] Gabriele Micali and Robert G Endres. "Bacterial chemotaxis: information processing, thermodynamics, and behavior". In: *Current Opinion in Microbiology* 30 (Apr. 2016). ISSN: 13695274. DOI: 10.1016/j.mib.2015.12.001.

[13] Toshitaka Nagai and Takasi Senba. "Behavior of radially symmetric solutions of a system related to chemotaxis". In: *Nonlinear Analysis: Theory, Methods Applications* 30.6 (1997). Proceedings of the Second World Congress of Nonlinear Analysts, pp. 3837–3842. ISSN: 0362-546X. DOI: https://doi.org/10.1016/S0362-546X(96)00256-8. URL: https://www.sciencedirect.com/science/article/pii/S0362546X96002568.

[14] Vidyanand Nanjundiah. "Chemotaxis, Signal Relaying and Aggregation Morpholm". In: *J. theor. Biol* 42 (1973), pp. 63–105.

[15] Singleton P. *Bacteria in Biology, Biotechnology and Medicine*. 5th ed. 1999. ISBN: 978-0-471-98880-9.

[16] Mercedes Rivero-Hudec and Douglas A. Lauffenburger. "Quantification of bacterial chemotaxis by measurement of model parameters using the capillary assay". In: *Biotechnology and Bioengineering* 28.8 (1986), pp. 1178–1190. DOI: https://doi.org/10.1002/bit.260280808.

[17] Norikazu Saito. *Conservative finite-difference scheme to a chemotaxis system*. 2006. URL: http://www.infsup.jp/saito.

[18] Hamidreza Shirzadfar. "Design and Evaluation of a GMR Biosensor for Magnetic Characterisation of Biological Medium". PhD thesis. June 2014. DOI: 10.13140/RG.2.1.2350.7361.

[19] Victor Sourjik and Howard C. Berg. "Receptor sensitivity in bacterial chemotaxis". In: *Proceedings of the National Academy of Sciences* 99.1 (2002), pp. 123–127. ISSN: 0027-8424. DOI: 10.1073/pnas.011589998. eprint: https://www.pnas.org/content/99/1/123.full.pdf. URL: https://www.pnas.org/content/99/1/123.

[20] George H. Wadhams and Judith P. Armitage. "Making sense of it all: bacterial chemotaxis". In: *Nature Reviews Molecular Cell Biology* 5 (12 Dec. 2004). ISSN: 1471-0072. DOI: 10.1038/nrm1524.

[21] Yu Wang, Chun-Lin Chen, and Miho Iijima. "Signaling mechanisms for chemotaxis". In: *Development, Growth Differentiation* 53 (4 May 2011). ISSN: 00121592. DOI: 10.1111/j.1440-169X.2011.01265.x.