Abstract

Cell motility plays a vital role in many biological processes and organisms. This project focuses on computationally modeling pseudopod growth, which is one mechanism by which cell motility occurs. Pseudopods grow through actin polymerization, a process by which actin molecules synthesize into chains, or filaments. These filaments form a network of actin within the cell that pushes the cell membrane outward, creating pseudopods. The growth of pseudopods depends on the concentrations of activators and inhibitors, which promote or stunt the growth of pseudopods. Using a pair of nonlinear partial differential equations, pseudopod growth was modeled with respect to the time and two-dimensional space. The computer programming language MATLAB was used to generate graphs of the concentrations of activators on a one dimensional view of a cell, as well as a two dimensional surface. The reaction rates of decay for the activators and inhibitors as well as their proportionality were found to be essential to the existence of pseudopods based on the equations used to describe pseudopod growth.

1 Introduction

All organisms are highly mobile on a cellular level. Understanding cell motility, the motion of cells, can have several applications in pathobiology. Cell mobility is essential to life; it is accountable for a wide array of biological processes including wound healing, nervous system reactions, immune response, embryonic development and cancer metastasis. Defective cell migration can lead to complications in cell-based therapy. When cell motility takes place in cancer cells, they metastasize, or translocate, and move to different organs, thus becoming deadly. If immune cells fail to migrate properly, an organism may be afflicted with illnesses such as asthma and rheumatoid arthritis. Amoebas grow pseudopods to assist them in moving towards food or escaping predators. Even some sessile cells have significant motion within them, as with chloroplasts in plants. Cell migration occurs through a variety of mechanisms. For the most part, cell motility, is beneficial to life. There are a variety of mechanisms which enable cellular motion. This project focuses on modeling the mechanism of pseudopod growth.
2 Background

2.1 Cell Motility using Pseudopods

Pseudopods are cellular projections that can be thought of as false feet, consisting of actin filaments extending from the cytoskeleton. Actin is a semi-flexible polymer that is found in the cytoskeleton of a cell. Crawling motion is achieved through three steps: first, a pseudopod is formed at the leading edge of the cell, causing it to protrude in its direction of motion; then, the leading edge adheres to the surface, while the trailing edge detaches from the substrate; finally, the trailing edge is contracted, resulting in a net forward motion of the cell.3

![Figure 1](image_url)

Figure 1: This image shows the process of cell motility. A pseudopod protrudes in the direction of motion, adheres to a substrate, and detaches itself on the other side, which causes the entire cell to move.

2.2 Actin Polymerization

Pseudopods are formed through the polymerization of actin microfilaments, enabling the cell to move. This phenomenon mostly occurs in the cytoskeleton of a cell. The process of pseudopod formation is initiated by the membrane sensory proteins PIP2 and WASp, which detect signals from the extracellular environment using chemosensing. This triggers the Arp2/3 protein to initiate nucleation, which is the process of combining with three actin monomers to begin polymerizing an actin filament. Following nucleation, actin monomers connect to the Arp2/3 complex to form a chain. Actin filaments continue to grow and the pointed end of the filaments polymerize, however, since one actin filament cannot exert enough force to create a pseudopod, several actin filaments are necessary. To facilitate this process, Arp2/3 attaches to the existing filaments and forms branches. The Arp2/3 protein can bind anywhere along the actin filament and at the site of combination, a new branch is formed. This phenomenon is referred to as filament branching, due to its resemblance of a tree. This process speeds up the production of actin filaments, causing a network of actin to form. Ultimately, this branching allows the actin network to provide enough force to make a pseudopod grow. Finally, actin depolymerization occurs, causing actin to be removed from the trailing end of the filament. Because there is a finite amount of actin monomers within the cell, proteins such as cofilin and actin depolymerizing factor (ADF) remove monomers from the barbed end of the chain which can then be re-attached later at the pointed end, acting as a means for recycling actin.
2.3 Dynamic Systems

The growth of actin filaments in a cell is an example of a dynamic system, in which an activator and inhibitor determine the growth and decay of the system. There are many dynamic systems in nature, such as sand dunes, stripes on seashells, and spots on a leopard, all of which rely on activators and inhibitors to create variation in these systems. For instance, a sand dune forms when a stone is eroded by wind and water, which are growth factors, but is then similarly worn away by other inhibiting factors. The concentration of sand particles in the air may be lowered as the sand dune increases in size, which in turn, slows down the growth of the sand dune. The inhibiting factors only become present when the dune is sizable. Likewise, Arp2/3 serves as an activator, while coflin, which depolymerizes actin, serves as the inhibitor, with the inhibitor’s presence relying on that of the activator. As the concentration of the activator increases, the concentration of the inhibitor increases as well, helping the system reach equilibrium. The activator and inhibitor must maintain a balance. If the activator concentration exceeds that of the inhibitor, the actin filament would, in theory, grow infinitely. In contrast, if the inhibitor exceeds the activator, the pseudopod would not grow. This project aims to examine the growth of pseudopods by mathematically modeling the interactions of the activators and inhibitors.

3 Procedure

3.1 Modeling Pseudopod Growth

The growth of pseudopods depends on the concentration of the different proteins associated with actin. The concentration of activators, such as Arp2/3, can be expressed as a function with respect to space and time, $A(x, t)$. Similarly, the concentration of inhibitors, such as coflin and actin depolymerization factor (ADF), can be expressed as the function $B(x, t)$.

To better understand this system, both functions were first analyzed in one-
dimensional space using the diffusion equation to solve for A at a specific space and time. \( \partial A / \partial t \) represents the rate of change of the activator concentration over time. This is equated to the rate of diffusion of A, represented by the coefficient of diffusion, \( D_A \), multiplied by the second derivative of A with respect to space, \( \partial^2 A / \partial x^2 \).

However, this is insufficient in accurately describing the growth of A. As previously mentioned, actin filaments are capable of branching into new protein chains anywhere along the original strand. In addition, the growth of these actin rods is impeded by the presence of inhibitor proteins. The concentration of inhibitor proteins depends on the presence of activator proteins at a given time and location. The growth of the inhibitor is represented by the term \( A^2 / B \), so that as B increases, the concentration of A decreases. In addition, the natural decay of A must be represented as \( R_A \times A \), where \( R_A \) is the coefficient for the reaction rate of decay. This results in the final equation

\[
\frac{\partial A}{\partial t} = D_A \frac{\partial^2 A}{\partial x^2} + \frac{A^2}{B} - R_A A.
\]

A similar equation can be used to solve for B, which represents the concentration of inhibitors such as cofilin. However, although B relies on the value of A, it is also uninhibited. This causes the second term to vary slightly. The partial differential equation for the inhibitor, B, can be expressed as

\[
\frac{\partial B}{\partial t} = D_B \frac{\partial^2 B}{\partial x^2} + A^2 - R_B B.
\]

Together, these two functions represent a system of coupled nonlinear partial differential equations called the reaction-diffusion system.

### 3.2 Finite Difference Method

Rather than solve this system using calculus, the process was simplified through numerical methods, which can be analyzed using computers. The finite difference method was used to express the equations algebraically by approximating derivatives as discrete intervals.

In calculus, the derivative, or the slope of a line, is equal to

\[
\frac{dy}{dx} = \lim_{\Delta x \to 0} \frac{dy}{dx}
\]

Given a discrete value of \( x \), namely \( x_i \), the instantaneous rate of change at that point can be approximated using two values of \( x \) in close proximity to \( x_i \), \( x_{i-1} \) and \( x_{i+1} \), and their corresponding \( y \) values, \( y_{i-1} \) and \( y_{i+1} \). The formula for the derivative would then be

\[
\frac{dy}{dx} = \frac{y_{i+1} - y_{i-1}}{x_{i+1} - x_{i-1}}.
\]

The distance between the points \( x_i \) and \( x_{i+1} \) equals the distance between \( x_i \) and \( x_{i-1} \), and thus, the denominator can be simplified to \( 2\Delta x \). In order to approximate the second derivative at the same point, two more points can be plotted between the previous nodes. If these intermediary points are called \( x_{i-\frac{1}{2}} \) and \( x_{i+\frac{1}{2}} \) and their corresponding \( y \) values \( y_{i-\frac{1}{2}} \) and \( y_{i+\frac{1}{2}} \), the first derivative, \( \frac{dy}{dx} \), at the point \( i \) using these two points changes to

\[
\frac{dy}{dx} = \frac{y_{i+\frac{1}{2}} - y_{i-\frac{1}{2}}}{\Delta x}.
\]

The second derivative, represented as

\[
\frac{d}{dx} \left( \frac{dy}{dx} \right)
\]

can be equated to

\[
\frac{d}{dx} \left[ \frac{y_{i+\frac{1}{2}} - y_{i-\frac{1}{2}}}{\Delta x} \right] = \frac{\Delta x}{\Delta x}
\]

Taking the derivative at the points \( x_{i-\frac{1}{2}} \) and \( x_{i+\frac{1}{2}} \) results in the following

\[
\frac{y_{i+1} - y_{i}}{\Delta x} - \frac{y_{i} - y_{i-1}}{\Delta x}
\]

When simplified, the second derivative is
\[
y_{i+1} - 2y_i + y_{i-1} = \frac{\Delta x^2}{A}
\]
This can be further simplified because it is known that
\[
\frac{\partial A}{\partial t} = D_A \frac{\partial^2 A}{\partial x^2},
\]
which also enables this derivative to be expressed in terms of both time and space. \(\frac{\partial A}{\partial t}\) can be expressed as
\[
\frac{A_i^{(n)} - A_i^{(n-1)}}{\Delta t},
\]
where \(i\) is the index of the node in space and \(n\) is the index for time and not an exponent. Likewise, \(\frac{\partial^2 A}{\partial x^2}\) can be expressed as
\[
\frac{A_{i+1}^{(n-1)} - 2A_i^{(n-1)} + A_{i-1}^{(n-1)}}{\Delta x^2},
\]
where \(i\) represents space and \(n\) represents time. Thus,
\[
\frac{\partial A}{\partial t} = D_A \frac{\partial^2 A}{\partial x^2}
\]
can be represented as
\[
\frac{A_i^{(n)} - A_i^{(n-1)}}{\Delta t} = \frac{A_{i+1}^{(n-1)} - 2A_i^{(n-1)} + A_{i-1}^{(n-1)}}{\Delta x^2}.
\]
In this representation, \(A_i^{(n)}\) is the only unknown because all the other terms have index \(n-1\) for time, meaning their values are already known. Hence, solving for \(A_i^{(n)}\), the final representation for the rate of change in concentration of activator with respect to both time and space is
\[
A_i^{(n)} = A_i^{(n-1)} + \frac{\Delta t}{\Delta x^2} D_A [A_{i+1}^{(n-1)} - 2A_i^{(n-1)} + A_{i-1}^{(n-1)}].
\]
The same method was then applied to \(B\), the inhibitor, resulting in
\[
B_i^{(n)} = B_i^{(n-1)} + \frac{\Delta t}{\Delta x^2} D_B [B_{i+1}^{(n-1)} - 2B_i^{(n-1)} + B_{i-1}^{(n-1)}].
\]
A special consideration when using the finite difference method is the periodicity of the function. This means that when \(i\) is 0, is equal to \(i+1\), when \(i\) is the maximum number of nodes.

### 3.4 Programming in MATLAB

In the computer program MATLAB, code was written with the intention of understanding pseudopod growth in one dimension. By inputting the numerical equations into for-loops for both time and space, an array can be created to represent concentrations at a specific node in time and space.

Using the equations which were simplified using the finite difference method, an algorithm was written that compiled values into vectors from 101 different nodes in space for both the activator and inhibitor. This was done for only one dimension in space, the \(x\)-axis, but was repeated for 500,000 iterations of time which were measured in tenths of seconds. The concentration of \(A\) was then plotted over time at a specific node.

After finishing the algorithm, the goal was to achieve a steady state of actin concentration. This steady state of actin represents equilibrium between the activator and inhibitor, which allows for pseudopod formation. This steady state is modeled as a horizontal line in the concentration of actin.
at a particular node in space. To accomplish this, the reaction rate of decay constants, $R_A$ and $R_B$, were varied until a steady state was reached.

### 3.5 Modeling Pseudopod Growth in Two Dimensions

To simulate pseudopod growth on a two-dimensional patch of the surface of a cell, a y-component was added to the equations. Another for-loop was added to account for this new dimension. In addition, a constant $C$ was added to both the activator and inhibitor equations to ensure that neither concentration would decrease to zero. Similarly to the one dimensional model, the periodicity of the function required that the boundary values, all edges and corners, be equated. Nine if-statements were created to define values of the activator and inhibitor in normal space, along the four edges, and the four corners. This is required because of the nature of numerical equations. Two dimensional pseudopod growth was modeled on MATLAB using the contour feature. The surface command was used to create a graph that appeared three dimensional, with the third dimension indicating the magnitude of the concentration. The contour command was used to produce a graph with colored contours showing the changes in concentration over the area.

### 4 Results and Discussion

#### 4.1 One-Dimension

Once a code to model one dimensional growth was written, certain variables were changed in order to achieve a steady state. A low $R_B$ and a high $R_A$ caused the inhibitor to overpower the activator. This caused the concentration of actin to decrease to 0, indicating no growth or existence of any pseudopods. To fix this, the magnitude of $R_B$ was increased and the magnitude of $R_A$ was decreased, in order to increase the overall concentration of actin. This resulted in the value of $A$ increasing towards infinity. This situation would result in infinite pseudopod formation and growth, a situation that is impossible due to its inefficiency. The variables were further manipulated until a steady state of actin concentration was reached. This steady state would represent a constant presence of pseudopods on the cell surface, which enables efficient cell motility.

Following this, a study was completed to observe how certain variables can affect the actin concentration. The results of this study are shown in the chart below.

<table>
<thead>
<tr>
<th>Effects of changing one variable</th>
<th>Increase in coefficient value</th>
<th>Decrease in coefficient value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$R_A$</td>
<td>Inhibitor is lower</td>
<td>Inhibitor is higher</td>
</tr>
<tr>
<td>$R_B$</td>
<td>All concentrations increase</td>
<td>All concentrations decrease</td>
</tr>
<tr>
<td>$D_A$</td>
<td>Not applicable</td>
<td>Varies peaks, nonsymmetrical, concentration goes up rapidly</td>
</tr>
</tbody>
</table>

Figure 4: A chart detailing the effects of changing certain constants

It was determined that the biggest factors for change in pseudopod growth were the rates of decay of activators and inhibitors. $D_A$, which was the diffusion rate of the activator, was important in terms of the number of pseudopods that formed during that trial, but overall it was kept fairly constant, with
values in close proximity to 0.001. \( R_A \) and \( R_B \) both had notable impacts on the results. In particular, it was critical that the ratio between their values did not fall under 0.75 or exceed 1.1. In fact, if the ratio was not maintained within this range, either the activator could not be controlled by the inhibitor whatsoever or the inhibitor would immediately kill off the activators, making it impossible for steady pseudopod growth. It is also important to note that both \( R_A \) and \( R_B \) were kept as low as possible. Initially, they had values of 0.75 and 1, respectively, but pseudopod growth markedly improved when both values were decreased by a factor of 5, to 0.15 and 0.20, respectively.

### 4.2 Two Dimensions

![Figure 5: A color-coded contour map detailing concentration of actin in space. The image on the left represents the concentration of actin at the beginning and the image on the right represents the concentration of actin after the code completed 500,000 iterations in time.](image)

These two graphs display the activator concentration on the two-dimensional surface of a cell. The graph on the left shows the initial concentrations of the activator that occur when random initial values are present. The graph on the right shows the concentrations after significant diffusion occurs. On both graphs, red areas represent areas of higher concentration, and blue areas represent areas of lower concentration. As depicted by the graph, the random signals in the beginning eventually result in a few areas of relatively higher concentration. Certain areas have peaks of activator concentration, which means that there is a high change of pseudopod existence in those areas. This particular set of graphs actually has a relative decrease in the concentration of actin activators, despite there being peaks at certain areas. The overall activator concentration at the beginning was 0.09 but after the system had reached a steady-state, the activator concentration was about 0.056. This decreasing activator concentration represents one particular type of behavior.

Additional behaviors of the system include an overall increase in activator concentration and an oscillatory behavior where the concentration of activator continuously moves up and down. By changing the values of \( R_A, R_B, D_A \), it was also possible to model an increasing behavior for activator concentrations. It was not, however, possible to achieve an oscillatory behavior. An oscillatory behavior most closely models true cell motility and requires immense computing power to model, which was not available at the time of testing.

Initially, the concentrations were random across the area of the cell’s surface. However, as time went on, certain areas began to have higher concentrations overall, while other areas had lower concentrations. Furthermore, the range of concentrations decreased as time went on, since the activator diffuses. This means that the areas with relatively high concentrations in the
second graph actually have lower concentrations than those in the first graph, as shown by the maximum values on the color scale to the right of the models.

5 Conclusion

Cell motility, the motion of cells, is crucial to a wide array of biological processes. In this study, pseudopod growth, a mode of cell motility, was modeled mathematically. Using the finite difference method, this system of differential equations was converted into an algebraic version, which could be solved by a computer. The simplified algebraic functions were programmed in MATLAB, first in one dimension, then in two dimensions. After extensive experimentation with the values of the reaction rates of decay for both the activator and the inhibitor, the one dimensional graph approached a steady state. This signified that a pseudopod would form. This concept was then extended to a two dimensional model with random chemical signals, which was animated to show the change in activator concentration over time. As a result of diffusion, the concentrations gravitated toward a relatively medium value, but there were distinct regions with higher concentrations, representing the growth of a pseudopod.

Throughout the study, two of three possible graph behaviors were observed. First, the actin concentration was found to decrease over time. After manipulating variables, the actin concentration was found to increase over time. In the future, the goal is to observe an oscillatory actin concentration, which most accurately represents the pseudopod growth mechanism within a cell. To accomplish this, a better understanding of how various parameters affect the behavior of pseudopod growth must be found by conducting more rigorous experiments.

6 Acknowledgements

The authors would like to thank Professor Prosenjit Bagchi of Rutgers Mechanical Engineering for his guidance and support throughout this project. The authors would also like to recognize graduate student Eric Campbell of Rutgers University for his assistance.

The authors are grateful to the Director of the New Jersey Governor’s School of Engineering and Technology, Ilene Rosen, the Assistant Director, Dean Jean Patrick Antoine, and all the instructors and Residential Teaching Assistants for coordinating this program. Special thanks to the Residential Teaching Assistant assigned to this project, Eamon Collins, for his supervision, support, and advice. Additional thanks to head RTA Taylor Au, Research Coordinator Edmund Han, and RTA Nitika Yadlapalli for reviewing this paper. Finally, the authors would like to acknowledge their sponsors, Rutgers, the State University of New Jersey, Rutgers School of Engineering, the state of New Jersey, Silverline Windows, Lockheed Martin, South Jersey Industries, Novo Nordisk Pharmaceuticals, Inc., and NJ resources for supporting this program.

7 References

1 D. A. Fletcher, J. A. Theriot, “An introduction to cell motility for the physical scientist.” Department of Bioengineering and Biophysics Program, University of California at Berkeley.


8 Appendix

The final two-dimensional code in the programming language MATLAB is presented below:

```matlab
clear all % Clear any residual values
num = 51; % Number of Nodes
dx = 0.02; % Spacing of x-axis nodes
x = 0:dx:1; % Space x at equal intervals of dx from 0 to 1
dy = 0.02; % Spacing of y-axis nodes
y = 0:dy:1; % Space y at equal intervals of dy from 0 to 1
A = 0.1*rand(num, num); % Set A equal to a random number from 0 to 1
A0 = A; % Store initial value of A
B = 0.1*ones(num, num); % Set matrix of B all equal to 0.1
B0 = B; % Store initial value of B
Da = 0.00005; % Diffusion Coefficient of A
Db = 0.02; % Diffusion Coefficient of B
Ra = .105; % Reaction rate of decay of A
Rb = .0250; % Reaction rate of decay of B
dt = 0.003; % Time intervals
C = 0.0001; % Constant actin concentration
for n = 1:500000 % Iterations of Time
    A1 = A;
    B1 = B;
    for i = 1:num % Y-dimension iterations (Space)
        for j = 1:num % X-dimension iterations (Space)
            if (i ~= 1 && j ~= 1 && i ~= num && j ~= num) % Defining all non-boundary values
                A(j, i) = A1(j, i) + dt * (Da * ((1/dx^2* (A1(j, i+1) - 2*A1(j, i) + A1(j, i-1)) + (1/dy^2* (A1(j+1,i) - 2*A1(j, i) + A1(j-1, i)))) + A1(j, i)^2/(B(j, i)+0.0001) - Ra * A1(j, i)+ C);
                B(j, i) = B1(j, i) + dt * (Db * ((1/dx^2* (B1(j, i+1) - 2*B1(j, i) + B1(j, i-1)) + (1/dy^2* (B1(j+1,i) - 2*B1(j, i) + B1(j-1, i)))) + A1(j, i)^2 - Rb * B1(j, i))+ C);
            end
            if (i == 1 && j~=1 && j~=num) % Defining bottom-edge boundary values
                A(j, i) = A1(j, i) + dt * (Da * ((1/dx^2* (A1(j, i+1) - 2*A1(j, i) + A1(j, i-1)) + (1/dy^2* (A1(j+1,i) - 2*A1(j, i) + A1(j-1, i)))) + A1(j, i)^2/(B(j, i)+0.0001) - Ra * A1(j, i)+ C);
                B(j, i) = B1(j, i) + dt * (Db * ((1/dx^2* (B1(j, i+1) - 2*B1(j, i) + B1(j, i-1)) + (1/dy^2* (B1(j+1,i) - 2*B1(j, i) + B1(j-1, i)))) + A1(j, i)^2 - Rb * B1(j, i))+ C);
            end
            if (i == num && j~=1 && j~=num) % Defining top-edge boundary values
                A(j, i) = A1(j, i) + dt * (Da * ((1/dx^2* (A1(j, i+1) - 2*A1(j, i) + A1(j, i-1)) + (1/dy^2* (A1(j+1,i) - 2*A1(j, i) + A1(j-1, i)))) + A1(j, i)^2/(B(j, i)+0.0001) - Ra * A1(j, i)+ C);
                B(j, i) = B1(j, i) + dt * (Db * ((1/dx^2* (B1(j, i+1) - 2*B1(j, i) + B1(j, i-1)) + (1/dy^2* (B1(j+1,i) - 2*B1(j, i) + B1(j-1, i)))) + A1(j, i)^2 - Rb * B1(j, i))+ C);
            end
        end
    end
end
```
if (j == 1 && i~=1 && i~=num) % Defining left-edge boundary values
A(j, i) = A1(j, i) + dt * (Da * ((1/dx^2* (A1(j, i+1) - 2 * A1(j, i) + A1(j, i-1)) + (1/dy^2* (A1(j+1, i) - 2 * A1(j, i) + A1(num, i))))) + A1(j, i)^2/(B(j, i)+0.0001) - Ra * A1(j, i)) + C);

B(j, i) = B1(j, i) + dt * (Db * ((1/dx^2* (B1(j, i+1) - 2 * B1(j, i) + B1(j, i-1)) + (1/dy^2* (B1(j+1, i) - 2 * B1(j, i) + B1(num, i))))) + A1(j, i)^2/Rb * B1(j, i))+ C);
end

if (j == num && i~=num && i~=1) % Defining right-edge boundary values
A(j, i) = A1(j, i) + dt * (Da * ((1/dx^2* (A1(j, i+1) - 2 * A1(j, i) + A1(j, i-1)) + (1/dy^2* (A1(1, i) - 2 * A1(j, i) + A1(num, i))))) + A1(j, i)^2/(B(j, i)+0.0001) - Ra * A1(j, i)) + C);

B(j, i) = B1(j, i) + dt * (Db * ((1/dx^2* (B1(j, i+1) - 2 * B1(j, i) + B1(j, i-1)) + (1/dy^2* (B1(1, i) - 2 * B1(j, i) + B1(num, i))))) + A1(j, i)^2/Rb * B1(j, i))+ C);
end

if ((i == 1 && j == 1)) % Defining bottom-left corner values
A(j, i) = A1(j, i) + dt * (Da * ((1/dx^2* (A1(j, i+1) - 2 * A1(j, i) + A1(j, i-1)) + (1/dy^2* (A1(j+1, i) - 2 * A1(j, i) + A1(num, i))))) + A1(j, i)^2/(B(j, i)+0.0001) - Ra * A1(j, i)) + C);

B(j, i) = B1(j, i) + dt * (Db * ((1/dx^2* (B1(j, i+1) - 2 * B1(j, i) + B1(j, i-1)) + (1/dy^2* (B1(j+1, i) - 2 * B1(j, i) + B1(num, i))))) + A1(j, i)^2/Rb * B1(j, i))+ C);
end

if ((i == 1 && j == num)) % Defining bottom-right corner values
A(j, i) = A1(j, i) + dt * (Da * ((1/dx^2* (A1(j, 1) - 2 * A1(j, i) + A1(j, i-1)) + (1/dy^2* (A1(j+1, i) - 2 * A1(j, i) + A1(num, i))))) + A1(j, i)^2/(B(j, i)+0.0001) - Ra * A1(j, i)) + C);

B(j, i) = B1(j, i) + dt * (Db * ((1/dx^2* (B1(j, 1) - 2 * B1(j, i) + B1(j, i-1)) + (1/dy^2* (B1(j+1, i) - 2 * B1(j, i) + B1(num, i))))) + A1(j, i)^2/Rb * B1(j, i))+ C);
end

if ((i == num && j == 1)) % Defining top-left corner values
A(j, i) = A1(j, i) + dt * (Da * ((1/dx^2* (A1(j, 1) - 2 * A1(j, i) + A1(j, i-1)) + (1/dy^2* (A1(j+1, i) - 2 * A1(j, i) + A1(num, i))))) + A1(j, i)^2/(B(j, i)+0.0001) - Ra * A1(j, i)) + C);

B(j, i) = B1(j, i) + dt * (Db * ((1/dx^2* (B1(j, 1) - 2 * B1(j, i) + B1(j, i-1)) + (1/dy^2* (B1(j+1, i) - 2 * B1(j, i) + B1(num, i))))) + A1(j, i)^2/Rb * B1(j, i))+ C);
end

if ((i == num && j == num)) % Defining top-right corner values
A(j, i) = A1(j, i) + dt * (Da * ((1/dx^2* (A1(j, 1) - 2 * A1(j, i) + A1(j, i-1)) + (1/dy^2* (A1(j+1, i) - 2 * A1(j, i) + A1(j, i))))) + A1(j, i)^2/(B(j, i)+0.0001) - Ra * A1(j, i)^2 + C);

B(j, i) = B1(j, i) + dt * (Db * ((1/dx^2* (B1(j, 1) - 2 * B1(j, i) + B1(j, i-1)) + (1/dy^2* (B1(j+1, i) - 2 * B1(j, i) + B1(j-1, i))))) + A1(j, i)^2/Rb * B1(j, i))+ C);
end

if (n==1) % Plot initial values
figure(1);
colorbar;
contourf(x, y, A0);
end

if (mod(n, 10) == 0) % Plot every 5 iterations in time
figure(2);
colorbar;
contourf(x, y, A1);
end