University of Hull

600093 - Computational Science

Cellular Automata

Nathan Rignall

April 29, 2024

Word count: 1933

Contents

1 Introduction . . . **1**

1.1 Simulation Techniques . . . 1

2 Methodology . . . **2**

2.1 Cell Movement . . . 2

2.1.1 Simulation Heatmap . . . 4

2.1.2 Movement Complexity . . . 6

2.1.3 Simulation Steps . . . 7

2.2 Cell Growth . . . 9

2.2.1 Changing the Capacity . . . 10

2.2.2 Accuracy of the Simulation . . . 10

2.2.3 Model Complexity . . . 12

2.2.4 Random Walk . . . 13

2.2.5 Changing the Grid Size . . . 14

3 Conclusion . . . **15**

List of Figures . . . **16**

List of Tables . . . **17**

Bibliography . . . **18**

A Appendix . . . **19**

1 Introduction

This report shall show the steps taken to simulate the growth of cancer cells in a tissue using the Gompertz Equation to model growth.

Growth for individual cells can be modelled using a differential equation so that the number of cells is a function of time. The Gompertz Model of Cell Growth works well in this application because it can accurately model the non exponential nature of cell growth (Tatro n.d.). Once an area has reached it's capacity no more tumour cells can form, hence the cells must begin to move through a tissue. This movement can be simulated using simple random walk algorithm or more complex genetic algorithms.

This paper specifically shall analyse the computational complexity of such models and evaluate the differences between simulation techniques. The underlying implementation shall be investigated, including random algorithms and accuracy of simulations on computer systems.

1.1 Simulation Techniques

Simulations shall use Rust as the programming language. It is a good choice for this project because it is a low level language that can be used to write high performance code. Allows for defined control over types and memory management.

Analysis of data shall be done using Python with Matplotlib and Numpy.

2 Methodology

2.1 Cell Movement

The random movement of cells should be modelled using a probability distribution

There are several probability distributions that can be classified as discrete or continuous, we will analyse uniform, bernoulli and beta.

The uniform distribution is usually regarded as a continuous distribution but can also be used to model discrete variables.

Area under the graph is one

$$f(x) = \begin{cases} a & \text{if } x = 1 \\ b & \text{if } x = 0 \end{cases}$$

$$f(x) = \begin{cases} 1 & \text{for } 0 \le x \le 1 \\ 0 & \text{otherwise} \end{cases}$$

Uniform distribution Definition of distributions Uniform distributions Bernoulli Distribution

Bias introduced with different distributions

When all directions are equally probable a uniform distribution could be used.

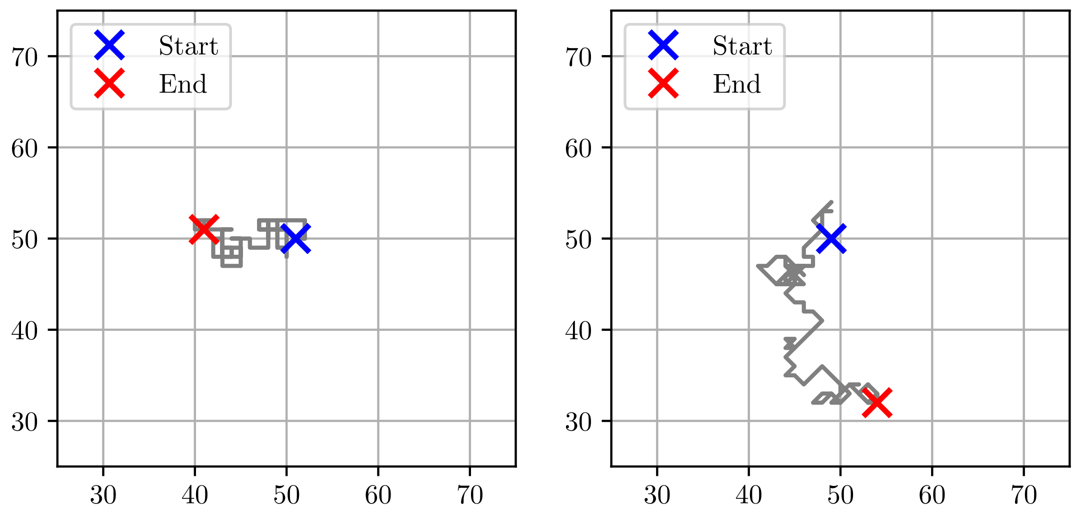
If a Bernoulli distribution was selected, it would introduce a bias to the direction of movement This could be desired in a complex model where factors such as surface tension affect the direction of movement

Both directions have bias

Square goes to right, diagonal goes down

This is because it is a single step run, the number generation is random, hence direction is random

100 steps, diagonal moves further than square



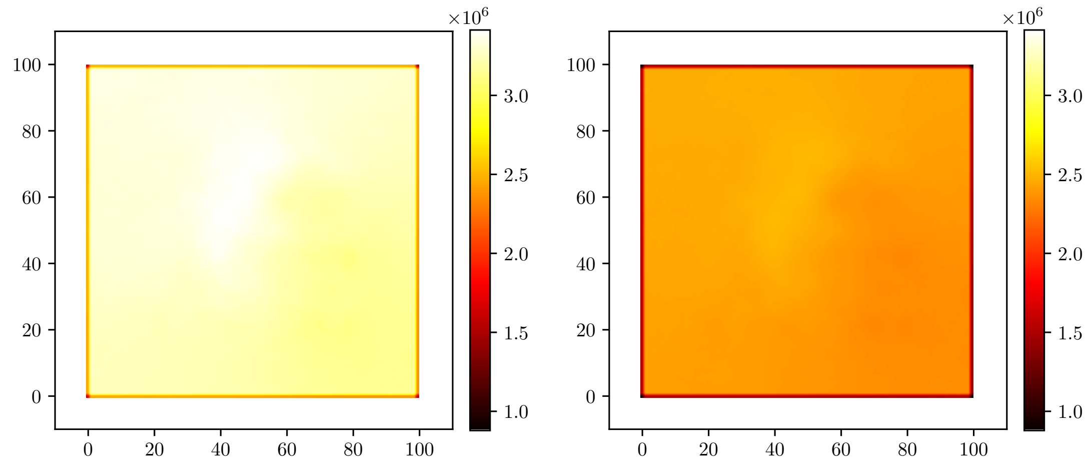
(a) Square   **(b)** Diagonal

Figure 2.1: Cell movement plots

2.1.1 Simulation Heatmap

The movement of the cells can be visualised using a heatmap to show the validate the uniform distribution of the movement. A number of start and end points can be selected using another uniform distribution. Next the random walk can execute until the end point is reached, this process can be repeated multiple times to gather a mean and record the visited cells.

The results are shown in Figure 2.2, there are small hotspots but overall the distribution is uniform. With more iterations, the distribution would become more even. This validates the uniform distribution of both the square and diagonal movements, and validates the uniform distribution of the random start and end points



(a) Square **(b)** Diagonal

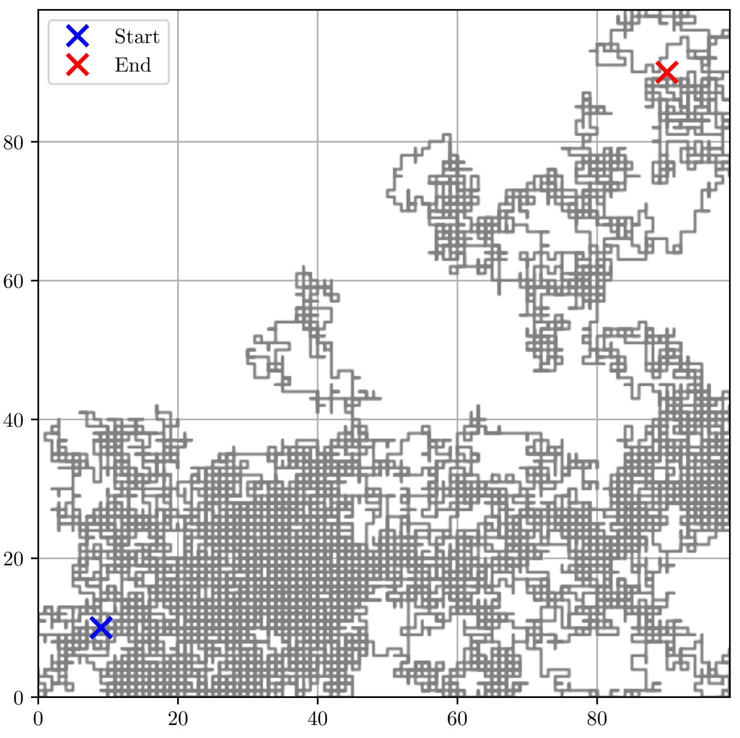
Figure 2.2: Visited cells simulation heatmap

There is a significant difference in the number of visited cells between the square and diagonal movements. This is because the diagonal movement can move further in a single step than the square movement.

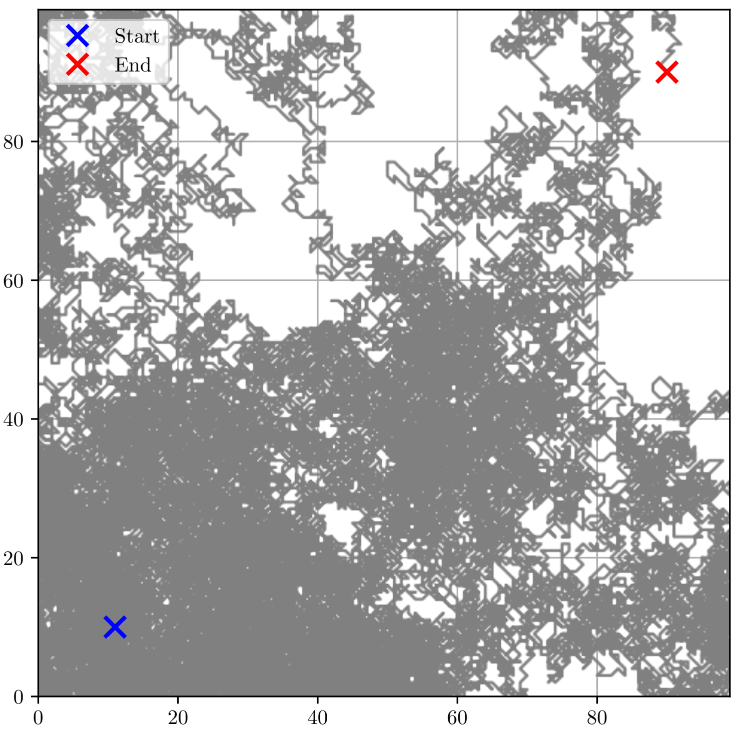
The square can perform one step to the left, right, up or down. The diagonal can perform one step to the left, right, up, down, up-left, up-right, down-left or down-right. Any diagonal movement is equivalent to $\sqrt{1^2 + 1^2} = \sqrt{2}$ square movements.

Hence square movement is on average $(1 \times 4)/4 = 1$ steps The diagonal movement is on average $(\sqrt{2} \times 4 + 1 \times 4)/8 = 1.2071$ steps Square is factor of $1/1.2071 = 0.8284$ compared to diagonal

Hence in the scene of moving from one point to another, the diagonal movement is more efficient than the square movement. However, this is different from computational complexity, where the square movement is more efficient than the diagonal movement.



(a) Square



(b) Diagonal

Figure 2.3: Cell movement fill plots

2.1.2 Movement Complexity

Using criterion, the computational complexity of the different movements can be measured. The results are shown in Figure 2.4.

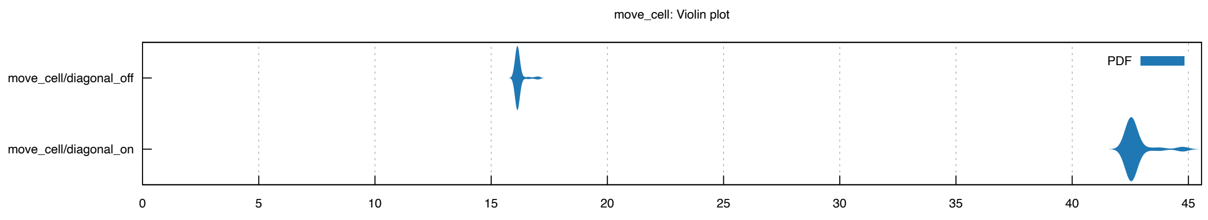


Figure 2.4: Cell movement criterion

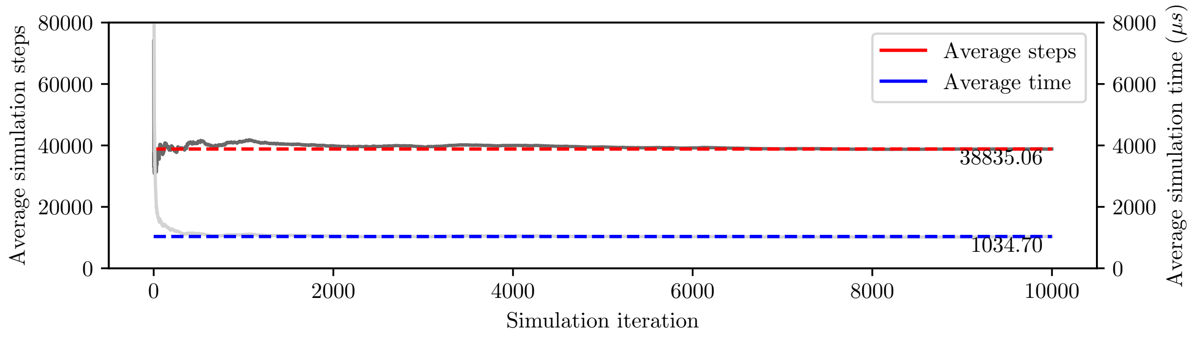
Square movement takes 16.2ns compared to diagonal movement which takes 42.7ns, this is a ratio of 0.3794. Hence, diagonal movement is more computationally complex than square movement.

This is because the diagonal movement requires more checks to ensure the cell does not move off the grid. The act of generating a random number is the same for both movements as a single random number is generated to determine the direction of movement, regardless of the movement method.

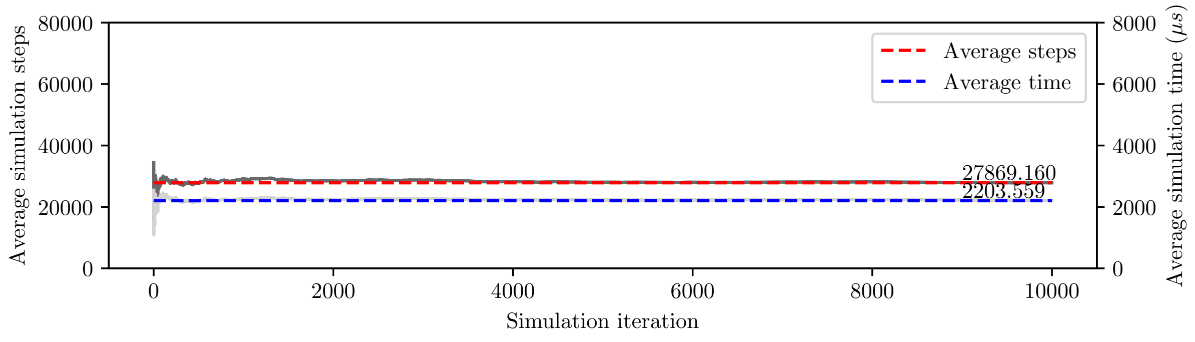
2.1.3 Simulation Steps

By selecting a random start and end point, we can compare the number of steps and time taken by the square and diagonal movements. If this movement simulation is performed multiple times, the average number of steps and time taken by the square and diagonal movements can be compared.

Given enough iterations, the average should stabilise and the difference in the number of steps and time taken by the square and diagonal movements can be calculated.



(a) Square



(b) Diagonal

Figure 2.5: Comparison of square and diagonal simulation steps

Figure 2.5 shows the number of steps taken by the square and diagonal movements for a single simulation with start and end points of (74, 67) and (55, 40).

As suggested in the previous sections, the diagonal movement takes less steps than the square movement to reach the end point. However, the diagonal movement takes longer to complete than the square movement as the computational complexity is higher.

By taking the same process and applying it to a random selection of start and end points, the average number of steps and time taken by the square and diagonal movements can be compared for different distances travelled.

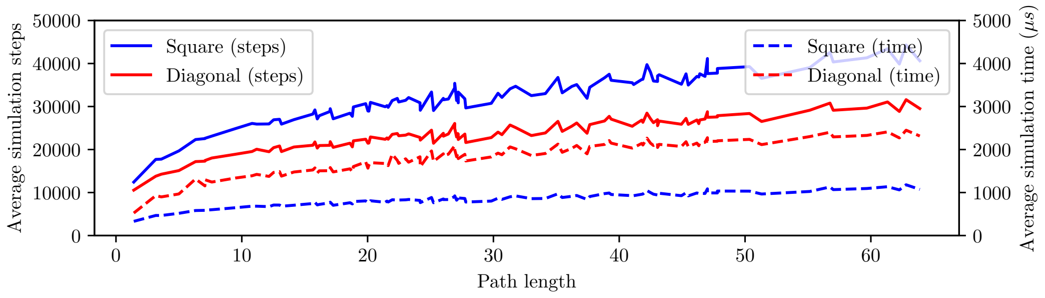


Figure 2.6: Comparison of multiple square and diagonal simulation steps

Next, by plotting the ratio of the average number of steps and time taken by the square and diagonal movements we can establish a factor by which the diagonal movement is more efficient than the square movement.

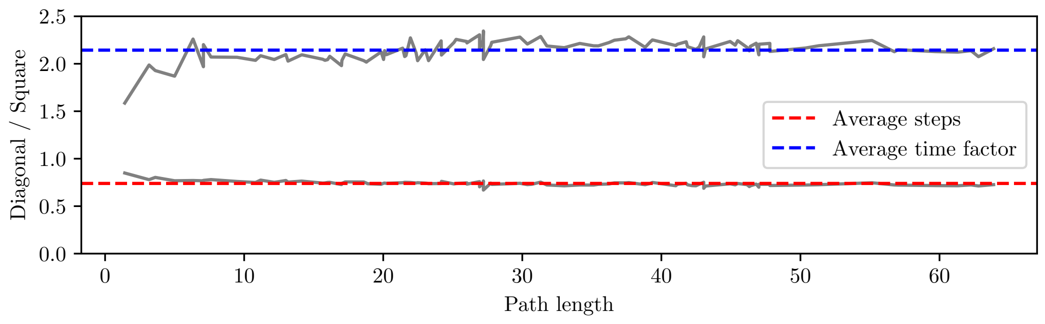


Figure 2.7: Comparison of multiple square and diagonal simulation steps (factor length)

The average factor for simulation steps is 0.7393 which is within the order of magnitude of the expected factor of 0.8284. The discrepancy is likley due to the non-linear behaviour in the interactions at the edge of the grid to prevent the cells from moving off the grid.

The average factor for simulation time is 2.1436. Given the square movement is 0.3794 times faster than the diagonal movement, and the diagonal movement takes 0.8284 times as many steps as the square movement, the expected factor for simulation time is $0.3794 \times 0.8284 = 2.184$. This is exceptionally close to the measured factor of 2.1436.

In conclusion, the diagonal movement is more efficient than the square movement in terms of the number of steps taken, but less efficient in terms of computational complexity. The computational complexity is significantly higher for the diagonal movement than the square movement, this overwhelms the number of steps taken. Hence, the square movement is more efficient than the diagonal movement overall.

2.2 Cell Growth

Using the Euler method, we can simulate the differential equation for cell growth in Figure 2.8. This uses the rate of growth at a point in time to calculate the number of cells at the next time step.

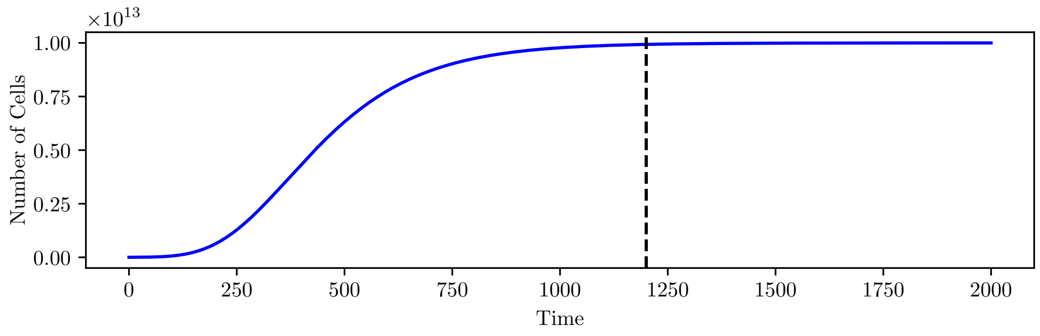


Figure 2.8: Cell growth simulation

The growth appears to reach a steady state at around 1200 time units, at this time the percentage of cells filled is 99.31%. Given the growth is modelled using an exponential function, the growth fills 100% at $t = \infty$, this applies if the simulation has an infinite accuracy. At a certain point, the percentage of cells filled will be 100% as the value is rounded up. Steady state can be defined numerically as:

- The percentage of cells filled is near 100%.
- The rate of change of cell growth is near 0.
- The rate of change of the rate of change of cell growth is near 0.

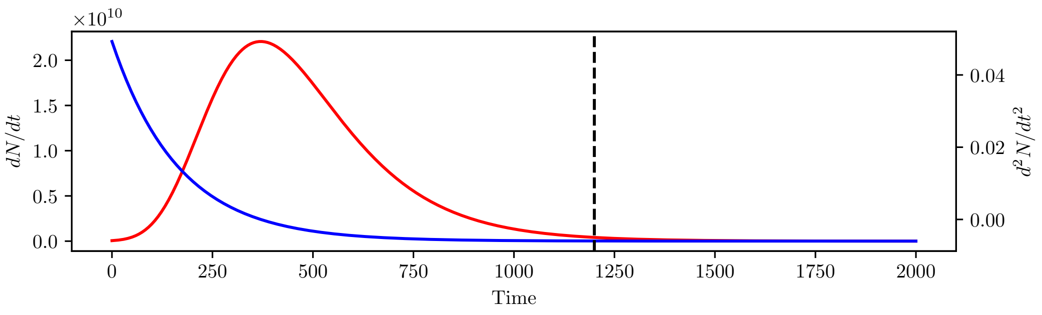


Figure 2.9: Cell growth simulation (dN/dt)

Figure 2.9 shows the rate of growth ($\frac{dN}{dt}$) and the rate of change of the rate of growth ($\frac{d^2N}{dt^2}$) for the simulation, solved analytically. When we reach a value of $t = 1200$, the rate of growth is 4.083e+08, and the rate of change of the rate of growth is -0.005959.

2.2.1 Changing the Capacity

Changing the capacity value M for the differential equation will change the rate of growth proportionally. Figure 2.10 displays that at 1200 time units, the percentage of cells filled is 99.31% across all values of M.

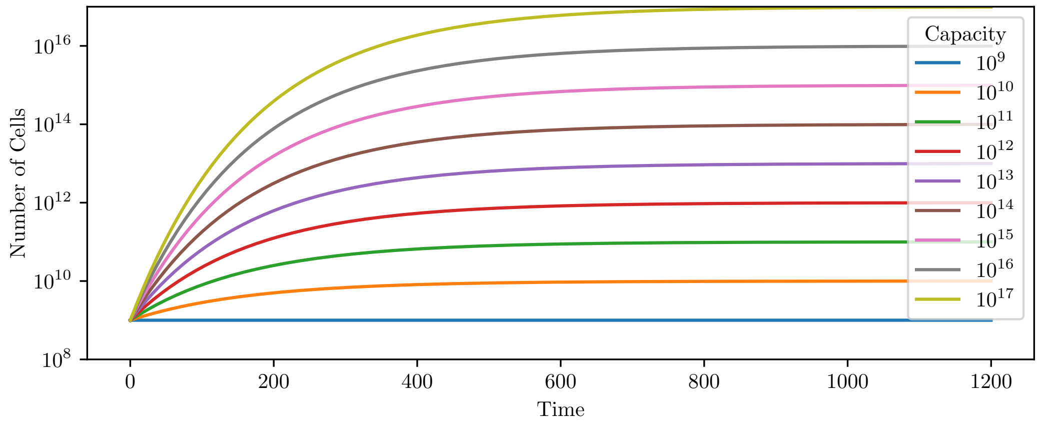


Figure 2.10: Cell growth simulation with different capacity values

The different values for M should not change the time to execute the simulation, as the resolution and time steps are the same.

2.2.2 Accuracy of the Simulation

When using the Euler method, the time step size (h or Δt) is important for the accuracy of the simulation. Larger values of h will result in larger computation errors relative to the analytical solution. Figure 2.11 and Figure 2.12 show the growth simulation with different values of h.

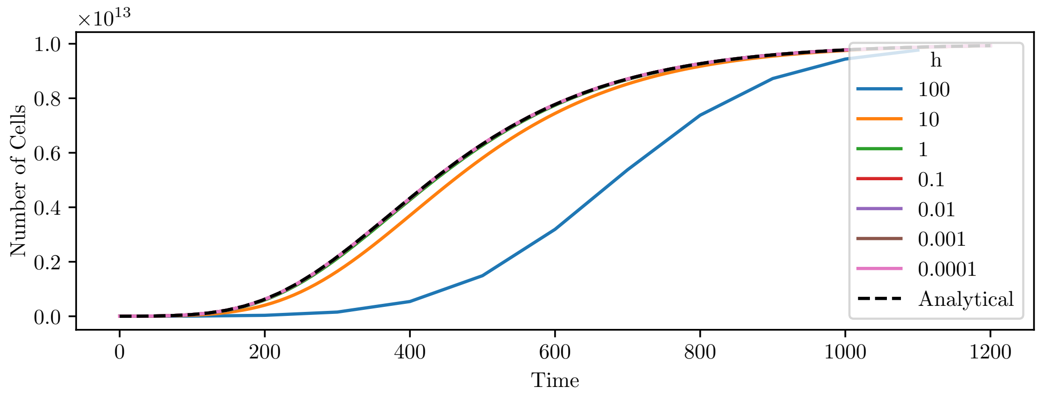


Figure 2.11: Cell growth simulation with different dt values

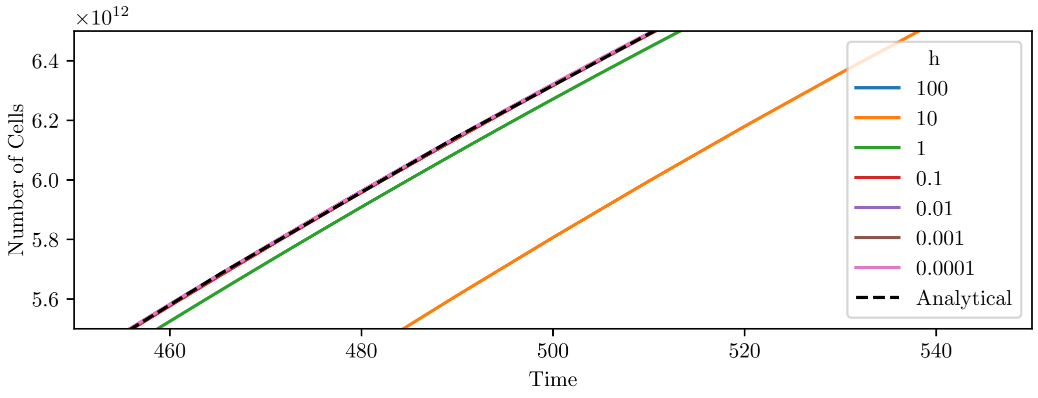


Figure 2.12: Cell growth simulation with different dt values (zoomed)

The mean absolute percentage error for different values of h relative to the analytical solution allows us to determine the optimal value of h for the simulation. Smaller values of h result in a more accurate simulation. However, smaller values of h also result in a longer computation time.

h	Error
100	66.69%
10	5.296%
1	0.5238%
0.1	0.05233%
0.01	0.005233%
0.001	0.0005239%
0.0001	5.82e-05%
Analytical	0.0

Table 2.1: Cell growth simulation error

2.2.3 Model Complexity

While the model is of $\mathcal{O}(n)$ complexity, the value for n is inversely proportional to the value of h ($N = 1/h \times t$). Decreasing h by a factor of 10 will increase the computation time by a factor of 10, this is a reciprocal relationship.

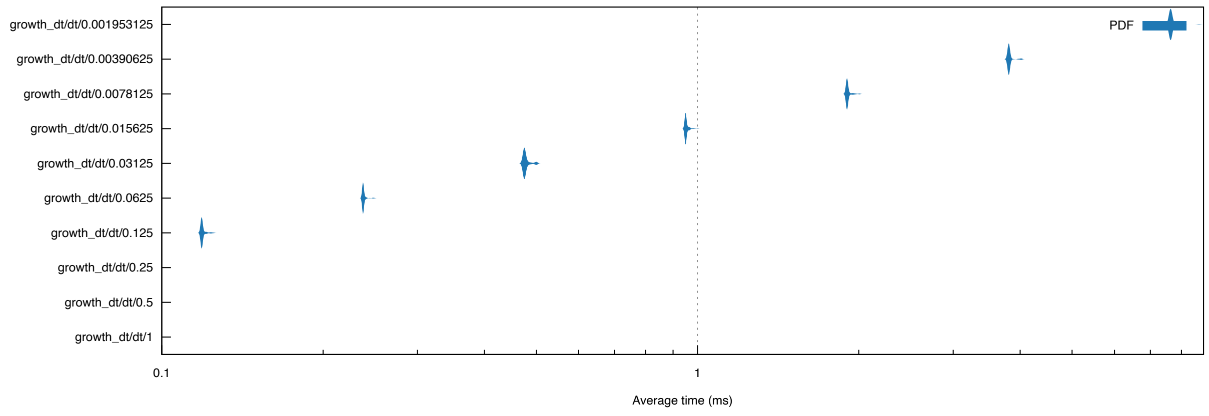


Figure 2.13: Cell growth criterion

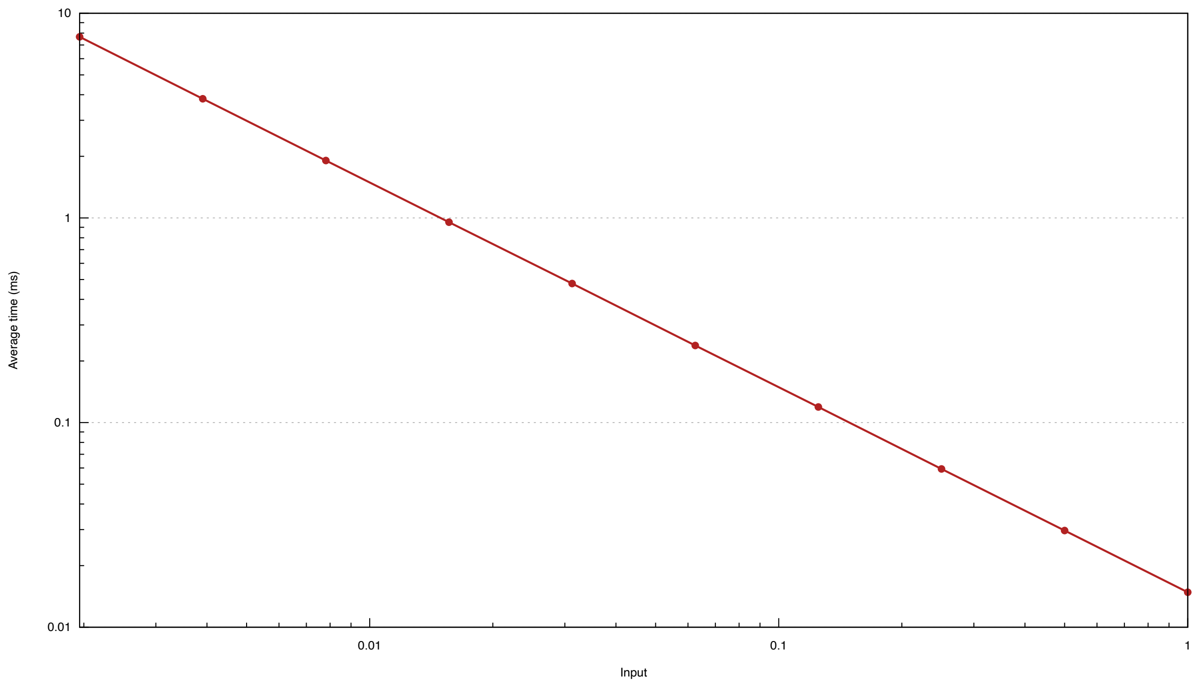


Figure 2.14: Cell growth criterion

Plotting both the value for h and t with logarithmic scales, we can see the relationship between the two values. It appears that the relationship is linear.

2.2.4 Random Walk

Now we can take the random walk and apply it to the cell growth model to simulate the movement of cells within a tissue. Before movement, the cells grow to a maximum capacity or a steady state. Once full, the cell moves in a random direction. Given the random walk is independent of the growth model, we can expect the same results as the independent random walk, displayed in Figure 2.15.

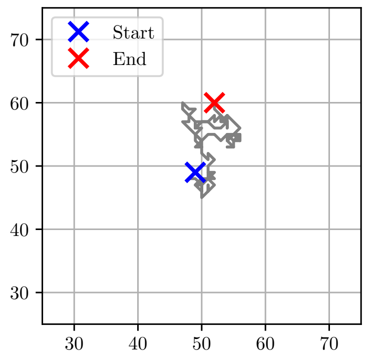


Figure 2.15: Cell growth simulation with diagonal movement

When the random walk reaches a cell it has already visited, it will instantly move to a new cell as the percentage full will be already above the threshold. Hence if the total number cells is plotted, we can expect a linear growth as shown in **??** with 100 walk steps. This linear growth can be estimated with a linear function, the growth is always at a steady state, until the tissue/grid is full.

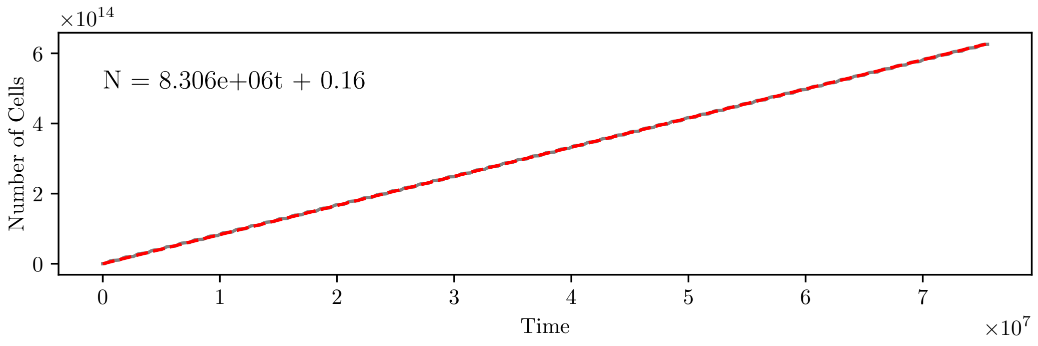


Figure 2.16: Cell growth simulation with diagonal movement (total cells) linear estimation

2.2.5 Changing the Grid Size

All simulations to this point used a grid size of 100x100, with a maximum capacity of 10,000 cells areas. Figure 2.17 shows the simulations steps required to fill the grid for different grid sizes, the graph is plotted against grid area.

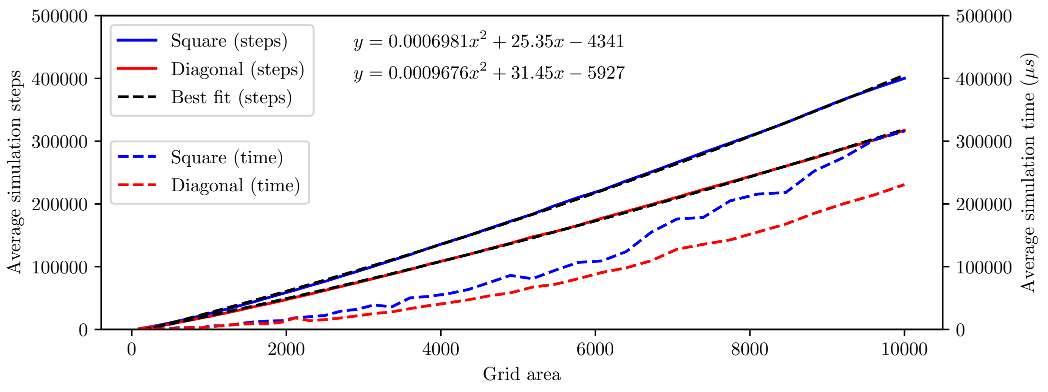


Figure 2.17: Cell growth grid comparison

The relationship between the grid area and the number of steps required to fill the grid is polynomial. The growth model and random walk remain complexity of $\mathcal{O}(n)$, however the grid size is now a factor in the complexity. This added complexity is related to the concept of filling the grid and is of $\mathcal{O}(n^2)$ if n is the grid area.

3 Conclusion

Some text

Also talk about edge processing and limited requirements

Issues with the numerical strategy include the accuracy of the simulation and the computation time.

List of Figures

2.1 Cell movement plots 3
2.2 Visited cells simulation heatmap 4
2.3 Cell movement fill plots 5
2.4 Cell movement criterion 6
2.5 Comparison of square and diagonal simulation steps 7
2.6 Comparison of multiple square and diagonal simulation steps 8
2.7 Comparison of multiple square and diagonal simulation steps (factor length) . . . 8
2.8 Cell growth simulation 9
2.9 Cell growth simulation (dN/dt) 9
2.10 Cell growth simulation with different capacity values 10
2.11 Cell growth simulation with different dt values 10
2.12 Cell growth simulation with different dt values (zoomed) 11
2.13 Cell growth criterion 12
2.14 Cell growth criterion 12
2.15 Cell growth simulation with diagonal movement 13
2.16 Cell growth simulation with diagonal movement (total cells) linear estimation . . 13
2.17 Cell growth grid comparison 14

List of Tables

2.1 Cell growth simulation error 11

Bibliography

Tatro, Dyjuan (n.d.). “The Mathematics of Cancer: Fitting the Gompertz Equation to Tumor Growth”. In: ().

A | Appendix

Integration using substitution

$$\begin{aligned}
\frac{dN}{dt} &= kNln\left(\frac{M}{N}\right)\\
\frac{dN}{Nln\left(\frac{M}{N}\right)} &= kdt\\
\int\frac{dN}{Nln\left(\frac{M}{N}\right)} &= \int kdt
\end{aligned}$$

with $u = ln\left(\frac{M}{N}\right)$ and $\frac{du}{dN} = -\frac{1}{N}$

$$\begin{aligned}
\int\frac{-Ndu}{Nu} &= \int kdt\\
\int -\frac{1}{u}du &= \int kdt\\
-ln\left(|u|\right) &= kt + c\\
ln\left(|u|\right) &= -kt - c\\
ln\left(\left|\ln\left(\frac{M}{N}\right)\right|\right) &= -kt - c\\
ln\left(\frac{M}{N}\right) &= e^{-kt-c}\\
\frac{M}{N} &= e^{e^{-kt-c}}\\
N &= \frac{M}{e^{e^{-kt-c}}}
\end{aligned}$$

Calculate c using the initial values

$$M = 10^{13} \qquad k = 0.06 \qquad N = 10^{9} \qquad t = 0$$

$$\begin{aligned}
10^9 &= \frac{10^{13}}{e^{e^{-0.006(0)-c}}} \\
e^{e^{-c}} &= \frac{10^{13}}{10^9} \\
e^{e^{-c}} &= 10^4 \\
-c &= ln\left(\left|\ln\left(10^4\right)\right|\right) \\
c &= -ln\left(\left|\ln\left(10^4\right)\right|\right)
\end{aligned}$$

Substitute c back in and simplify

$$\begin{aligned}
N &= \frac{M}{e^{e^{-kt+ln\left(\left|\ln\left(10^4\right)\right|\right)}}} \\
&= \frac{M}{e^{e^{-kt}e^{ln\left(\left|\ln\left(10^4\right)\right|\right)}}} \\
&= \frac{M}{e^{e^{-kt}\ln\left(10^4\right)}} \\
&= \frac{M}{\left(e^{\ln\left(10^4\right)}\right)^{e^{-kt}}} \\
N &= \frac{M}{10^{4e^{-kt}}}
\end{aligned}$$