600093 – Computational Modelling

Table of Contents

Task 1 ..... ***2***

Uniform Distribution 2

Normal Distribution 2

Bernoulli Distribution 2

4-direction movement **3**

8-direction movement **5**

Task 2 ..... ***7***

Gompertz Model **7**

Plotting tumour growth using Gompertz Model **9**

Run 1 9

Run 2 10

Run 3 10

Run Summary 10

The following report outlines how simple instruction, and simulation can be used to create more complex systems. Random number distributions will be investigated, focusing on the laws of large numbers, complexity, and accuracy.

These investigations will be completed using a series of simple simulated environments to depict the growth of a tumour, starting by plotting random movement of a cell in a 100 x 100 grid, with 4 possible directions, and progressing to 8, exploring distribution uniformity in stochastic models. The Gompertz growth model will then be used to deterministically simulate tumour cell growth, analysing parameter changes and their impact on progression to a steady state. The two processes will then be integrated to simulate tumour propagation, providing insight into nonlinear systems.

Task 1

Uniform Distribution

Uniform distribution is a type of probability distribution in which all outcomes have an equal chance of occurring. Examples of this include rolling a fair six-sided dice or drawing from a deck of cards. This is the most appropriate distribution for the simulations as there is an even chance of each direction being picked, ensuring that movement is random.

Notation:

$$X \sim U(a,\ b)$$

Probability mass function:

$$f(x) = \frac{1}{n} \qquad a \leq x \leq b$$

Normal Distribution

Normal (or Gaussian) distribution is a continuous probability distribution that is symmetric around its mean, creating a bell curve. The mean, median and mode are all equal in normal distribution. Examples of normal distribution can be seen in human physical characteristics such as height and weight distribution. This is not appropriate for the simulations as the movement would show bias toward directions, instead of random and even selection.

Notation:

$$X \sim \mathcal{N}(\mu, \sigma^2)$$

μ = mean
σ^2 = standard deviation.

Probability mass function:
e = base of natural logarithm.

$$f(x) = \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right) \cdot e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Bernoulli Distribution

Bernoulli distribution has a binary, or binomial distribution of 1 or 0, where each outcome has a 50% chance of occurring. An example of this distribution can be seen in a coin toss. This distribution could be used for the simulation; however, could decrease computational efficiency, with more calculations required in each step.

Probability mass function:

$$\begin{aligned} P(X = 1) &= p \\ P(X = 0) &= q \\ &= 1 - p \end{aligned}$$

4-direction movement

The movement of the cell is determined by the ".choice" function from the "random" Python library, where it can make a "uniform selection (from) a random element" (Python, 2025). The function will select a movement value at random from the "directional_moves" tuple, which contains (x, y) values for going up (0, 1), down (0, -1), left (-1, 0) and right (1, 0). When selected, these values will be added to the current (x, y) value of the cell, which will plot a new point on a grid. Boundary checks are in place to ensure the plots do not exceed the 100 x 100 size from (-50, -50) to (50, 50). The simulation complexity is O(n), where n is the number of runs, making the process computationally efficient.

The distribution of selected moves will be examined across a 10,000-step simulation, analysing the results at 100, 500, 1000, 5000 and 10,000 moves.

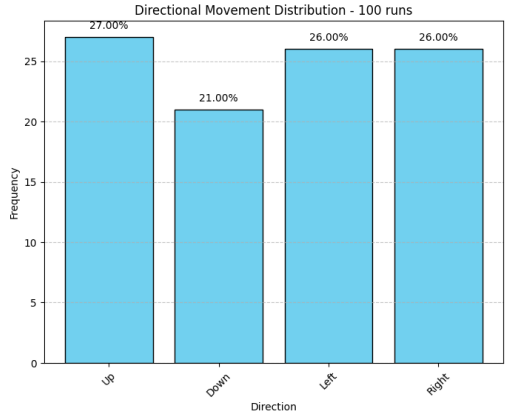

Figure 1.1 - Distribution after 100 runs

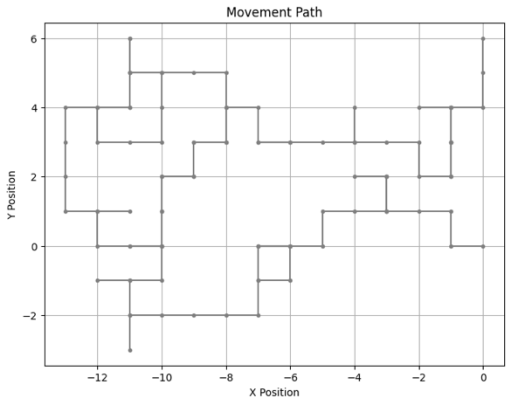

Figure 1.2 - Plot after 100 runs

After 100 steps, the results do not show uniformity,
with up and down deviating from an even distribution at 27% and 21% respectively.
Randomness is likely to dominate at a smaller number of steps.

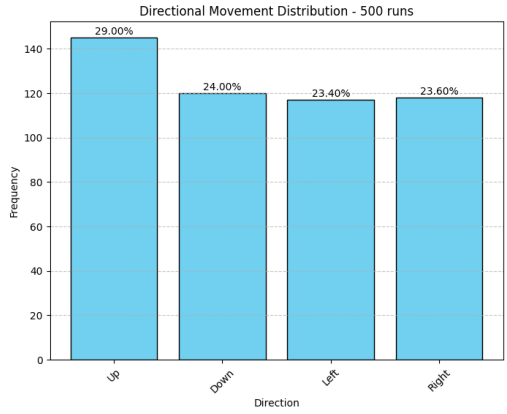

Figure 2.1 - Distribution after 500 runs

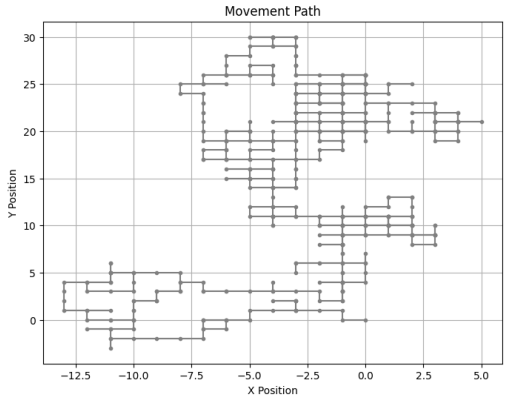

Figure 2.2 - Plot after 500 runs

The distribution begins to stabilise after 500 steps, but still shows noticeable variation, favouring up at 29%

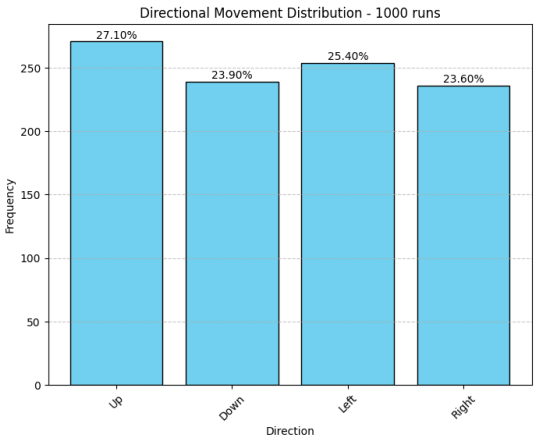

Figure 3.1 - Distribution after 1000 runs

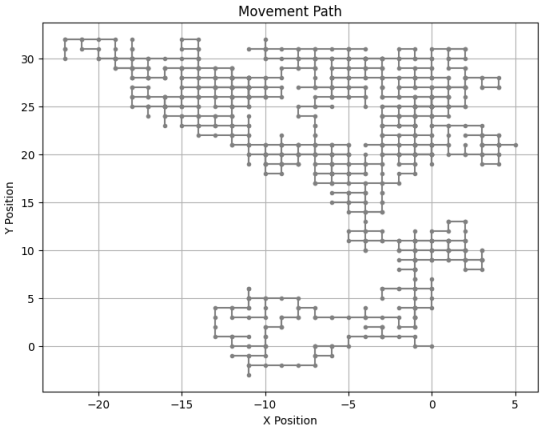

Figure 3.2 - Plot after 100 runs

Small deviations persist at 1000 steps; however, the results are beginning to converge towards 25%.

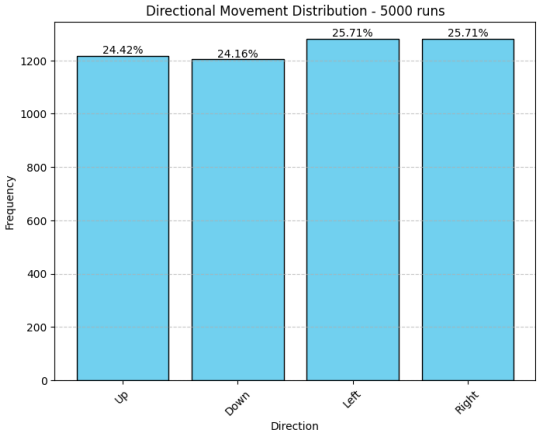

Figure 4.1 - Distribution after 5000 runs

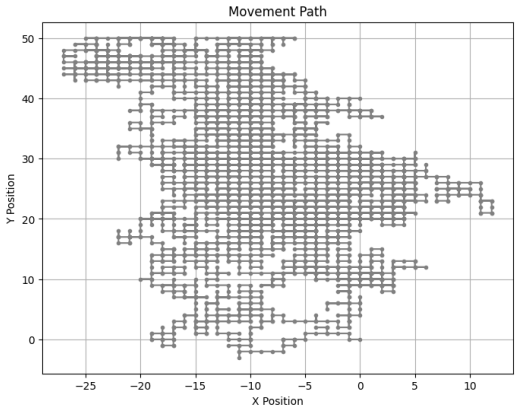

Figure 4.2 - Plot after 5000 runs

The values are much closer to uniform distribution, which demonstrates the impact of the law of large numbers, where larger sample sizes yield more balanced results (Sedor, 2015).

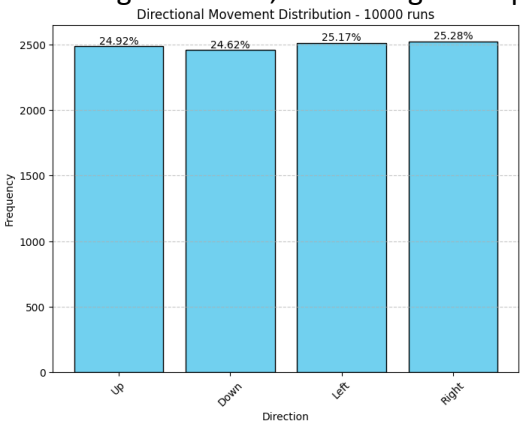

Figure 5.1 - Distribution after 10,00 runs

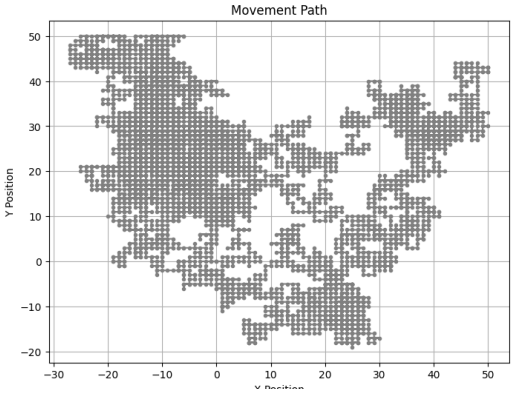

Figure 5.2 - Plot after 10,000 runs

At 10,000 runs, the results closely resemble the theoretical expectation of 25% in each direction, suggesting that the random distribution is uniform.

8-direction movement

This simulation operates very similarly to the 4-direction movement, with the addition of diagonals. The diagonal moves were added to the “directional_moves” tuple, which would now include up-left (-1, 1), up-right (1, 1), down-left (-1, -1) and down-right (1, -1), in addition to up, down, left and right.

The movement of the cell will be plotted over 10,000 steps, outputting move distribution graphs after 100, 1000 and 10,000 moves.

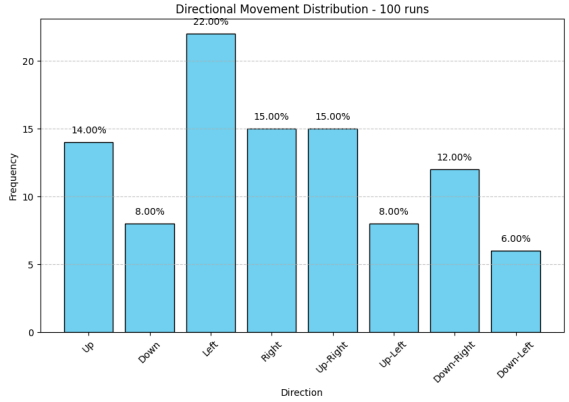

Figure 6.1 - Distribution after 100 runs

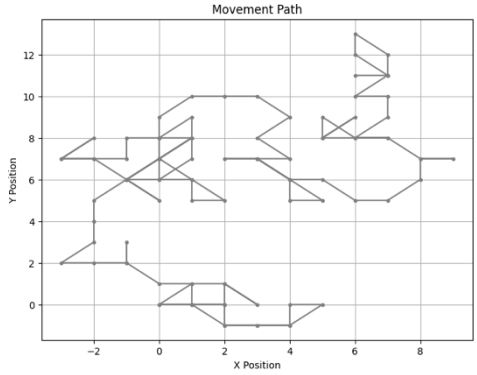

Figure 6.2 - Plot after 100 runs

After 100 runs, the distribution lacks uniformity, with a range of distributions from 6% on down-left to 22% on left. These biases can be attributed to the smaller sample size and it being more affected by random fluctuations.

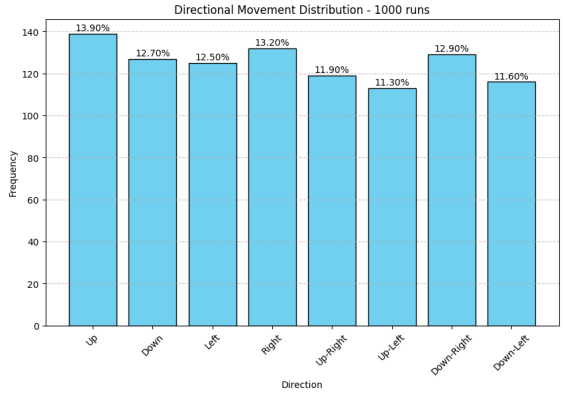

Figure 7.1 - Distribution after 1000 runs

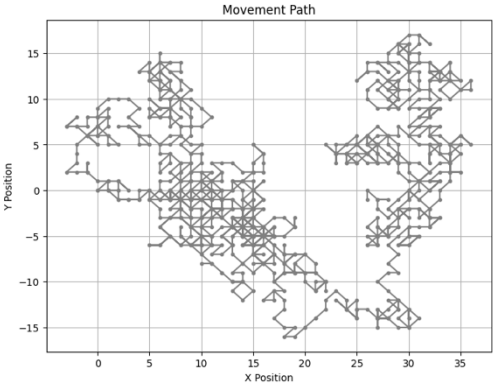

Figure 7.2 - Plot after 1000 runs

After 1000 steps, there is some bias present in the distribution, seen in up and right. However, the distribution can be seen to tend toward uniformity.

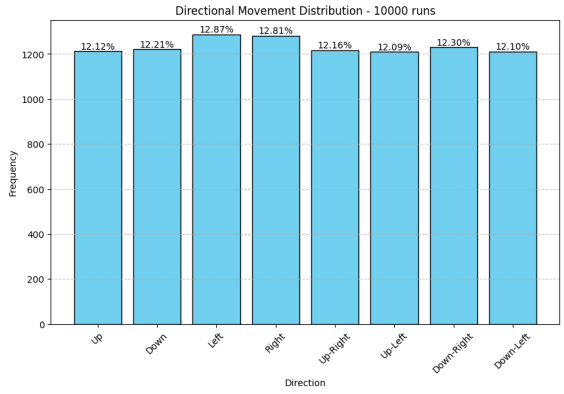

Figure 8.1 - Distribution after 10,000 runs

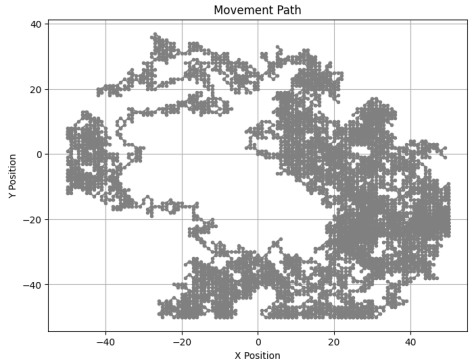

Figure 8.2 - Plot after 10,000

The distribution across all directions is close to uniform, as all values converge close to 12.5% after 10,000 moves.

Point	Frequency
48, -20	20
48, -19	19
27, -14	17
-18, -49	17
42, -21	16

Table 1 - Top 5 Most Visited Points

After 10,000 steps, the most common points of traversal were around the edges, where x and y were ≈±50. This occurred due to the cell reaching the limits of the grid, meaning that moves exceeding ±50 would not be possible.

This may introduce bias, as the number of possible moves is reduced, seen in Figure 8.1, where left and right are at a higher frequency.

The plots seen in Figure 8.2 show where the cell reaches the outer limits of the grid, creating flat lines along y=-50, x=-50 and x=50.

Task 2

Gompertz Model

Tumour growth can be simulated using Gompertz model (Tatro, 2018), which is used to mathematically show how a population, such as bacteria or a tumour, grows similarly to a sigmoid function.

The differential equation for the Gompertz growth model is represented with the following:

Value	Definition
N(*t*)	Population at time *t*
k	Growth rate constant (*k* > 0)
M	Carrying capacity (maximum population)
$\frac{dN}{dt}$	Growth rate

Table 2 - Gompertz Growth Equation Definitions

$$\frac{dN}{dt} = kN \ln\left(\frac{M}{N}\right)$$

The natural logarithm occurs as the growth rate is influenced by the current population, meaning as *N* tends toward *M*, the growth rate decreases exponentially, giving the curve a horizontal asymptote.

A steady state is reached when the population approaches carrying capacity, causing the growth rate to increase by a negligible amount. In this exercise, the steady state has been defined as 99% of *M* and the time taken to reach this point will be recorded. The inflection point will also be calculated, which is the point in the growth curve, where the rate begins to slow down significantly, shifting from an exponential growth phase to a more gradual approach towards *M*. For this simulation, the inflection point will be { $M * (1 - e^{-1})$ }, where *e* is Euler's number, equalling to ≈63.21%.

The time taken to reach the inflection point and steady state will be assessed across different sizes of *M* – 10^{12}, 10^{13}, 10^{14}, with an initial tumour cell population (*N*) of 10^{9}.

The values of *N(t)* will be calculated in steps of 0.001, ensuring a balance between reduced error and improved precision, to computational efficiency, as complexity increases at a rate of *O*(1/*h*), where *h* = step size. The error calculation is proportional to *machine precision* / *h*, as Euler's method can introduce truncation errors and machine precision can introduce round-off errors.

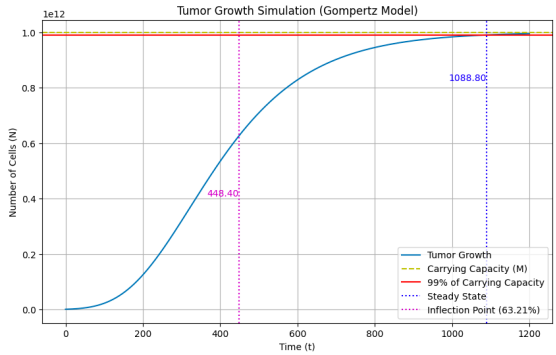

Figure 10 - Gompertz Model, M = 10^{12}

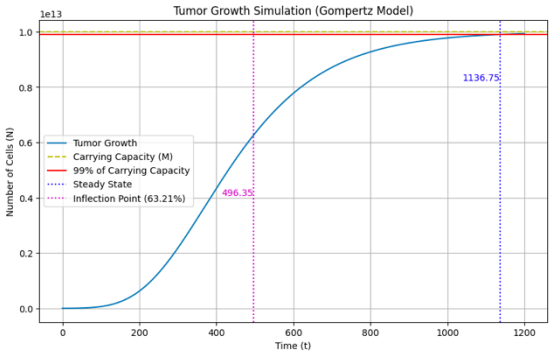

Figure 9 - Gompertz Model, M = 10^{13}

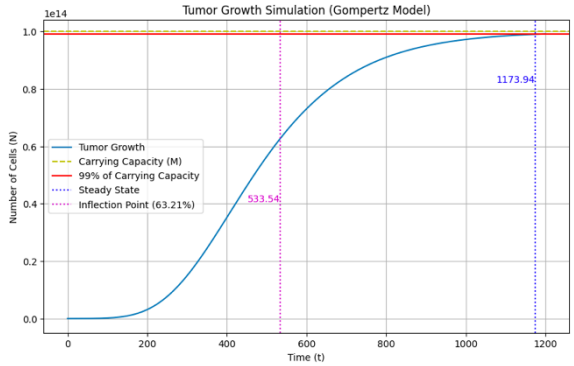

Figure 11 - Gompertz Model, M = 10^{14}

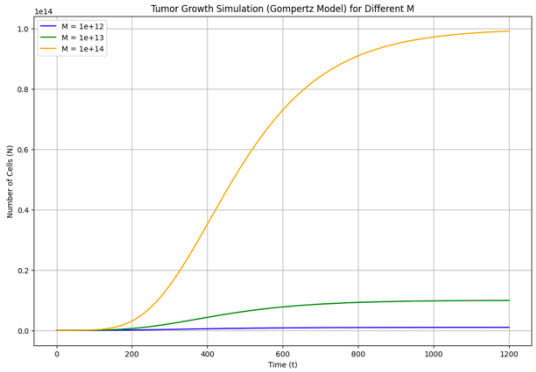

Figure 12 - Gompertz Model, All Values

M	**Inflection Point (*t*)**	**Steady State (*t*)**
10^{12}	448.40	1088.80
10^{13}	496.35	1136.75
10^{14}	533.54	1173.94

Table 3 - Gompertz Model Simulation Results

The simulations show that the time taken to reach the inflection point and steady state increases with *M*, following a logarithmic trend, as Gompertz model is based on exponential growth and decay.

Plotting tumour growth using Gompertz Model

The Gompertz growth model will be combined with the 8-direction random movement, forming a simplified simulation of tumour growth, where tumour growth starts at the centre of a grid, and spreads to adjacent cells after reaching a steady state, with the initial population of the cell resetting after each growth.

As the growth model will be reinitialised after every step, the computational efficacy of the simulator will be affected significantly more by changes in parameters, particularly the step size and carrying capacity { O(ln(*M*)/*h*) }.

The steady state in this simulation is defined as 66% of the carrying capacity. The cells will not be able to grow back on themselves, meaning that if all adjacent cells are occupied, the simulation will end.

The simulation will be run with an increased carrying capacity and step size, exploring the change in run time and complexity.

The distribution of movement and number of runs is random and unaffected by the change in parameters of the simulation. However, they may display more bias due to the cells being unable to move to previously occupied cells.

Run 1

Run 1 had a step size of 0.001 and a carrying capacity of 10^{13}. The simulation lasted 147 runs and took 139 seconds to complete, ≈0.95 seconds per run.

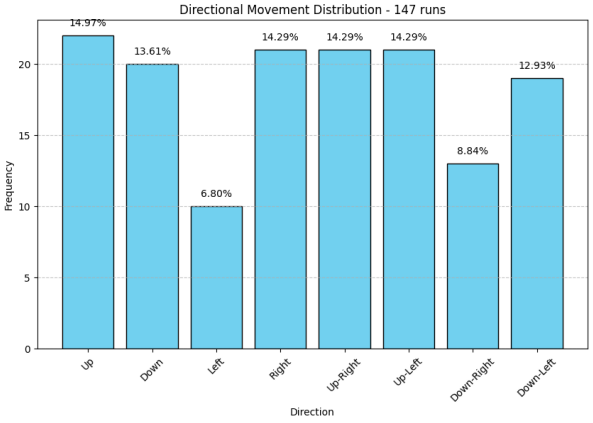

Figure 13.1 - Distribution of Run 1

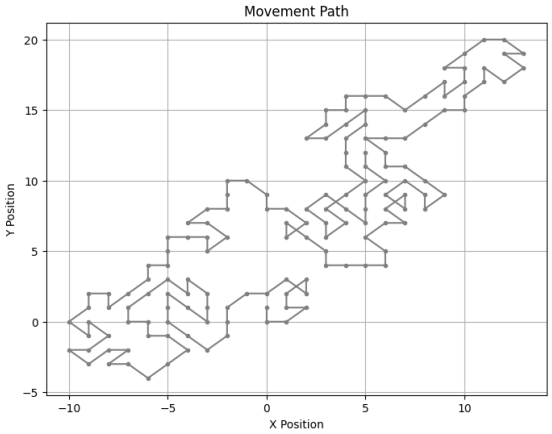

Figure 13.2 - Plot of Run 1

Run 2

In run 2, the step size was 0.001 and the carrying capacity was increased to 10^{14}. The simulation lasted 155 runs and took 139 seconds to complete, ≈1.02 seconds per run.

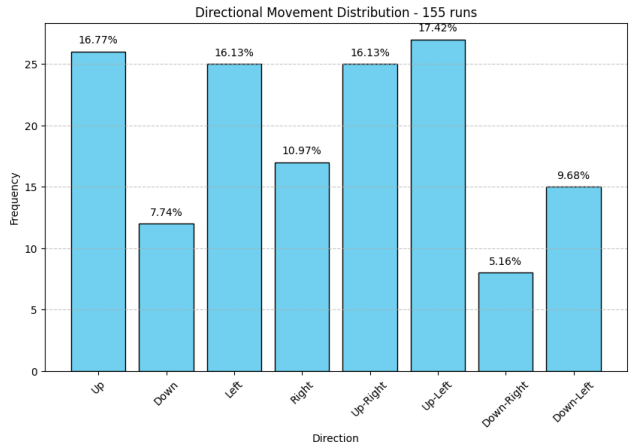

Figure 14.1 - Distribution of Run 2

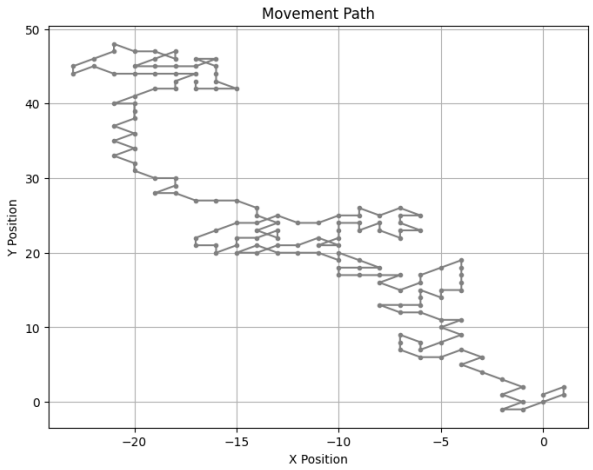

Figure 14.2 - Plot of Run 2

Run 3

A larger step size of 0.01 was used in run 3, with a carrying capacity of 10^{13}. The simulation took 20 seconds to run and completed 187 runs, ≈0.11 seconds per run.

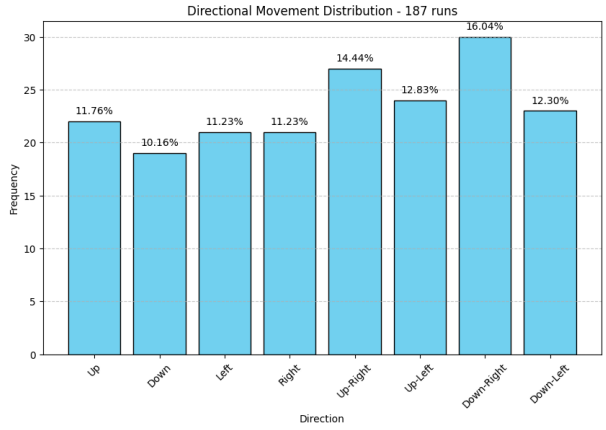

Figure 15.1 - Distribution of Run 3

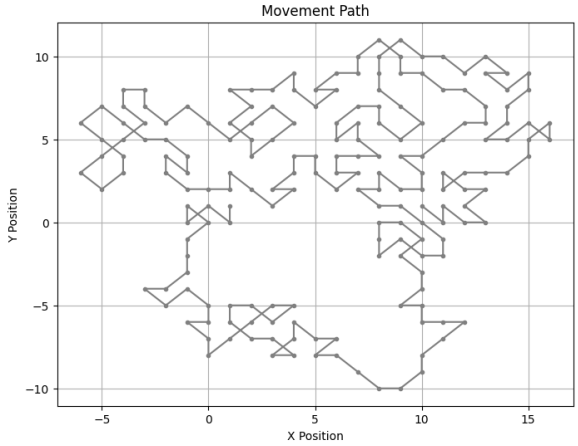

Figure 15.2 - Plot of Run 3

Run Summary

The change in runtime with the variation of *M* and *h* demonstrates the proportional relationship between the machine performance and the time to reach a steady state divided by the step size.

M	***h***	**≈ Seconds / Run**
10^{13}	0.001	0.95
10^{14}	0.001	1.02
10^{13}	0.01	0.11

Table 4 - Gompertz Cell Growth Simulation Summary

A larger value of *M* increases the time taken by a cell to reach a steady state. An increase of step size decreases runtime linearly, with 1/*h*. However, a decreased step size could lead to the introduction of rounding and truncation errors.

Python (2025) *Random - generate pseudo-random numbers.* https://docs.python.org/3/library/random.html [Accessed Jan 15 2025].

Sedor, K. (2015) The law of large numbers and its applications.

Tatro, D. (2018) The mathematics of cancer: Fitting the gompertz equation to tumor growth.