Implementing Molecular Diffusion in Agent-Based Models of The Immune System

Simon Jarrett*, Jason Cosgrove†, Kieran Alden†, Mark Coles* and Jon Timmis†
*Department of Biology
University of York, UK
†Department of Electronics
University of York, UK
Email: sjj509@york.ac.uk

***Abstract*—The immune microenvironment shapes functional immune responses through the secretion of soluble molecules such as cytokines, chemokines and signalling lipids. Increasingly, theoretical models are used to study these factors with emphasis on how these factors are distributed within tissues. While theoretical frameworks to model diffusion within tissues exist, there are a dearth of studies highlighting implementation challenges and how to mitigate them, particularly when modelling diffusion in a discrete environment. To address this we explore key parameters of a discrete form of the heat equation. We provide data to compare different implementations of the model, quantifying trade-offs in efficiency and accuracy and incorporate them into a hybrid agent-based model capable of bridging molecular and cellular levels of understanding.**

I. Introduction

To form a robust response to immunisation and infection the immune microenvironment must communicate with immune cells through direct contact or through soluble activatory, migratory and survival factors. In the context of secondary lymphoid tissues, T-cell responses are coordinated by specialised stromal cells known as fibroblastic reticular cells which secrete the chemokines CCl19 and CCL21 cite fletcher and link. Mice deficient in these factors show a paucity of lymph node T-cells with defects in the migration of naive T cells and activated dendritic cells within lymphoid organs, [5]. Despite the importance of these molecules, they are difficult to study *in situ* due to a dynamic regulatory network occurring across multiple spatiotemporal scales []. As such, experimental studies are often complemented by the use of modelling and simulation (M & S) techniques (ref).

Agent-based models are a M & S technique composed of individual entities which interact with each other and their environment through a predetermined rule-set [1]. The aggregate effects of these interactions gives rise to emergent behaviours at the population level. Agent-based models are spatially resolved and can incorporate heterogeneity, making them a suitable platform for studying immune cells (ref alden frontiers, MMH, bocharov, kirschner).Relative to immune cells, molecules exist in much larger numbers, and move on faster timescales. As such it is often computationally intractable to model both molecules and cells using an agent-based approach. A common approach to study diffusion in this context is through functions which relate molecular concentration to distance from a source [5], partial differential equations [3][4] and discretised partial differential equations [6] (Figure 1). However, PDEs and sigmoids are of limited value in hybrid agent-based systems with multiple moving cells as they become difficult to solve when molecules are dynamically added and removed. This is relevant to lymphoid tissues where chemokine fields are dynamic and ligand-scavenging is required to shape functional chemotactic gradients ref rot.

For hybrid agent-based systems where model outputs are sensitive to the molecular processes, molecular concentrations can be represented on discretised grids. This approach which can be adapted to model more complex behaviours, such as the fluid dynamics of lymph or in reaction-diffusion schemes [6]. To model diffusion in a discrete environment, grajdeanu have developed a discretised form of the heat equation. The mathematical construct is well-suited to hybrid agent-based systems in that it is capable of isotropic diffusion, can diffuse to an arbitrary number of neighbours and is applicable to linear, planar, spatial and n-dimensional constructs. While providing a general framework to model diffusion,there are limited data available to guide the most appropriate implementation of this scheme for a given research context.

A. Aims

The aims of this paper are to address a number of technical considerations of the gradjeanu model. Specifically, we seek to explore the relationship between the diffusion timestep and the diffusion constant, highlighting where this complex relationship may become unstable. In addition we quantify the effect of grid and neighbourhood sizes on the efficiency and accuracy of the approach, and compare two different instantiations of the scheme which are relevant to implementation within agent-based models of the immune system. Finally, we demonstrate the broader implications of these analyses through a hybrid agent based model of T-cell migration in response to chemokine diffusion.

II. Methods

In the discretised grid model, the area or volume within a compartment is split into discrete, identical grid spaces, with each grid space having an associated molecular concentration.

We focus on three-dimensional grids, but many of the techniques discussed here are equally relevant to two-dimensional grids. Diffusion is modelled in the grid by calculating the movement of molecules between individual grid spaces; the grid spaces that a grid space interacts with in diffusion is called its neighbourhood. Each grid space has a set of coordinates representing its position within its compartment. For all experiments we use a diffusion constant value of $140m^2$. This value was derived using the einstein-stokes relation where we assume the fluid environment is a homogeneous intracellular media in the absence of binding effects.

A. Implementating Diffusion

Convolution matrices are a useful way of describing how a grid space interacts with its neighbourhood. The centre of the matrix corresponds to the diffusing grid space, and the other values in the matrix represent the proportion of the grid space's molecular concentration that moves to the surrounding grid spaces. The overall process of diffusion can then be simulated by applying the resulting convolution matrix to each grid space in the grid. We will be looking at two different schemas for creating convolution matrices: linear and Grajdeanu.

In the linear model of diffusion, molecules move only to the immediate, Von Neumann neighbours in the grid. An example convolution matrix for the 2-dimensional case can be seen in Equation 1. The

$$\frac{1}{5}\begin{bmatrix}0 & 1 & 0\\ 1 & 1 & 1\\ 0 & 1 & 0\end{bmatrix} \tag{1}$$

The Grajdeanu model of diffusion is a discretisation of Fick's law of diffusion - a mathematical model of diffusion. Instead of diffusing to just the immediate neighbours, the Grajdeanu model is capable of diffusing to an arbitrary number of neighbours; the more neighbours the more accurate the solution. For example, in the two-dimensional convolution matrix in Equation 2, the diffusion neighbourhood consists of all grid spaces within 2 units of distance.

$$\begin{bmatrix}0 & 0 & f(2) & 0 & 0\\ 0 & f(\sqrt{2}) & f(1) & f(\sqrt{2}) & 0\\ f(2) & f(1) & f(0) & f(1) & f(2)\\ 0 & f(\sqrt{2}) & f(1) & f(\sqrt{2}) & 0\\ 0 & 0 & f(2) & 0 & 0\end{bmatrix} \tag{2}$$

The function f is derived from the heat equation [**?**]:

$$f(d) = A\ \exp\left(\frac{-d^2}{4Dt}\right) \tag{3}$$

Where D is the diffusion coefficient, t is the length of a single timestep, and A is chosen such that the sum of all values in the convolution matrix is 1, ensuring the process of diffusion neither adds nor removes molecules from the grid.

B. Common Implementation Problems

1) Anisotropy: Anisotropy is where diffusion occurs faster in certain directions than others. For example, the convolution matrix given in Equation 1 has biased diffusion along the x and y axes, and so diffuses slower along the diagonal directions. Anisotropy was one of the motivations for Grajdeanu's discretisation of Fick's Law [**?**]; the Grajdeanu method reduces the problem of anisotropy to negligible levels by diffusing along additional axes.

The problem of anisotropy, then, needs to be addressed in the linear model of diffusion. By comparing the linear model with a control, we can quantify the error introduced by anisotropy. In Figure ??, we graph the average error for the grid spaces in the linear approach after the convolution matrix is applied different numbers of times; it can be seen that as the number of diffusion steps is increased, the error approaches zero. Because the process of diffusion happens on a much smaller timescale than cell interactions, the number of diffusion steps per simulation timestep is typically high, and so the error introduced by anisotropy becomes negligible.

2) Diffusion Coefficient: When implementing a model of diffusion, typically a specific diffusion coefficient is desired. It is not immediately obvious how to get a predefined diffusion coefficient as the output of a model of diffusion. Indeed, even the Grajdeanu diffusion model, which takes the diffusion coefficient as a parameter, does not give the specified diffusion coefficient as the output, unless the diffusion neighbourhood is excessively large.

The output diffusion coefficient of a convolution matrix can be calculated using the mean squared displacement (MSD). Equation 4 relates the diffusion coefficient, D, with the mean squared distance, $\langle\mu^2\rangle$ and the timestep length, t, in n dimensions:

$$\langle\mu^2\rangle = 2nDt \tag{4}$$

The diffusion coefficient is conserved over multiple iterations of a convolution matrix; by calculating the mean squared displacement of the convolution matrix, we can choose the timestep, t, to give the required diffusion coefficient.

However, when using the Grajdeanu model of diffusion, the timestep is a parameter for the convolution matrix. This can be ignored, but it can result in a less accurate solution where values closer to the source of molecules are higher, and values at extreme distances are much lower than they should be. Using the Grajdeanu model, the only way to accurately model diffusion is to use a large neighbourhood: the input diffusion coefficient is actually an upper bound on the resulting diffusion coefficient, and using a smaller neighbourhood brings the resulting diffusion coefficient further away from this upper bound. Increasing neighbourhood size quickly ramps up the computational complexity, and so an alternative way to increase the output diffusion coefficient, other than increasing the neighbourhood size, is to increase the input diffusion coefficient above the desired amount. This can give a convolution matrix with the desired diffusion

coefficient, but without a sufficient number of diffusion steps, it can lead to the same problems as adjusting the timestep size.

3) Estimating Continuous Points: It isn't always possible to create a grid such that we only have to access it on the known discrete points, and so a way to estimate the concentrations of points between the known discrete points is needed. A simple but naive approach would be to assume the concentration within a given grid-space is constant, but a better solution is to use interpolation. Interpolation is the process of estimating continuous points of a data set when given nearby discrete points. For a 2-dimensional square-based grid and a 3-dimensional cube-based grid, we can use bilinear and trilinear interpolation respectively.

Linear interpolation is the process of reducing the number of discrete points by estimating additional points using a linear function. The process is explained fully in Figure **??**. n-dimensional interpolation takes n steps to reduce 2^n points down to one, but each step is computationally trivial, and so is efficient even in models that require frequent interpolation.

The method of interpolation described has been for estimating continuous values on the discrete grid, but the reverse of the process can be used to add values to the grid at a continuous point; for example, when a cell secretes chemokine, it secretes to a continuous point which is then distributed to the nearby discrete grid points according to reverse linear interpolation.

Other methods, such as tricubic interpolation, which use more than the immediately adjacent data points can provide better results, but are much more computationally complex.

C. Parameters

1) Grid Size: The larger and more fine-grained the grid is, the more memory and computational power is needed to simulate diffusion, but the more accurate the solution will be. By using a smaller grid space size, we reduce the errors introduced by interpolation. Ideally, we would pick a grid space size such that we access the grid only on the corners of the grid - removing the need for interpolation; practically, this is usually impossible. Thus, the choice of the size of the grid is a trade-off between accuracy and complexity; we should choose the largest grid-space size that meets our accuracy requirements, and the smallest grid-space size that remains computationally tractable.

2) Grid Shape: The grid is made up of tessellated regular polygons, and the polygon you choose can affect the complexity of the system, the accuracy of the result, and the kind of problems affecting the implementation.

In a 2-dimensional system, squares and hexagons are commonly used. Squares are very easy to implement using a Cartesian coordinate system, but can lead to problems with anisotropy due to the diagonally adjacent grid spaces being a different distance than the immediately adjacent grid spaces (See §II-B1). A hexagonal grid avoids the problem of anisotropy because all the adjacent grid spaces are equidistant; however, the coordinate system and implementation are more complex, and interpolating (See §II-B3 between grid spaces is non-trivial.

In a 3-dimensional grids, cubes are used, being the only regular polyhedron that will tessellate to fill space. Just like the squares in the 2-dimensional system, cubes are simple to implement using a Cartesian coordinate system, but suffer the same problem with anisotropy.

3) Diffusion Neighbourhood: The number of other grid spaces each grid space interacts with is called the neighbourhood. Choosing the neighbourhood affects both the accuracy and complexity of the solution; a larger neighbourhood is more computationally complex, but means the diffusion is more accurate in a larger area.

4) Timestep Length: In the Grajdeanu system of simulating diffusion, the timestep is used in calculating the convolution matrix. A consequence of this is that the relationship between the timestep and the resulting diffusion coefficient is non-linear.

In the linear system, decreasing the timestep always increases the resulting diffusion coefficient, but under the Grajdeanu system, there is a maximum diffusion coefficient for a given input diffusion coefficient and neighbourhood.

5) Diffusion Coefficient:

D. Metrics

1) Computational Complexity: To measure computational complexity, we calculate the number of operations required to complete one second of diffusion. An operation here means a single interaction between two grid spaces. When calculating the diffusion coefficient, we have to iterate through all the grid spaces, and then interact with grid spaces according to the convolution matrix being used; note that since all interactions are symmetrical, we only use half the neighbourhood. Thus, the computational complexity is the product of the grid width, height, depth, and half the size of the neighbourhood.

2) Accuracy: We use the relative difference to measure the accuracy of the diffusion method. The relative difference can be calculated with the formula,

$$d_r = 2\,\frac{|c_{\text{meas}} - c_{\text{ref}}|}{c_{\text{meas}} + c_{\text{ref}}}. \tag{5}$$

Where c_{meas} is some measured value, and c_{ref} is some reference value. The relative difference allows us to calculate the accuracy of the results even when they are approaching zero. However, we also do also exclude the values closest to zero, lower than $1E-5$, as the relative difference starts to become less reliable.

To get the reference values, we use the Grajdeanu diffusion model with a very large diffusion neighbourhood and a dense grid. The resulting model is very accurate, being based on the mathematical model of diffusion, but impractically slow. This model of diffusion was simulated once, and the output was stored to give the c_{ref} values.

III. Results

We simulated the two models of diffusion in three dimensions, and varied the inputs to compare both the computational complexity and accuracy of the models. First, we looked at the Grajdeanu model when compared to the linear model of diffusion. In Figure 3a, we plot each grid space's concentration against its distance from the molecule source; this gives us some intuition about the differences between the models, and the effects of changing the size of the grid. Both of the dense grid models follow the reference model closely, but the sparse grid models start to deviate from the reference. Notably, the sparse Grajdeanu can be seen to be closer to the reference model at almost all distances besides the grid space the molecule source is located at. This is most likely because, while the Grajdeanu approach eliminates anisotropy by diffusing in more directions, the linear approach requires more time steps which, as was seen in Figure 2b, reduces the error. The computational complexity and the mean percentage difference was calculated for each of the models and is shown in Figure 3b. The computational complexity of the Grajdeanu model is much greater than that of the linear model, but it is more accurate. As the diffusion coefficient output of the model is increased, both models become much more accurate, likely because the number of diffusion steps increases. Confirming our intuition, the sparse grid is much less accurate than the dense grid; however, the computational complexity is much lower in the sparse grid.

Secondly, we look at how changing the input parameters affects the complexities and accuracies of the models. Of our five parameters, we look at the effect of changing the grid size and the neighbourhood size. Changing the grid shape is non-trivial, but there are also no good alternatives in a three dimensional system. The timestep length and the diffusion coefficient both affect the output diffusion coefficient the most; however, they are the parameters that need to be changed to normalise the output diffusion coefficient and so it does not make sense to vary them. In Figure 3c, we see the results of changing the grid size and the neighbourhood size; note the logarithmic scale on the y-axis. Reconfirming what we saw in the Figure 3b, increasing the grid space size has a significant effect on both of our metrics: decreasing both the accuracy and the computational complexity. Conversely, increasing the neighbourhood distance increase both the complexity and the accuracy. In Figure 3d, we get a better idea of the relationship between the complexity and the accuracy when changing the two parameters. Changing the neighbourhood size is much less effective at reducing the complexity of the system when compared to changing the grid space size.

A. Discussion

In summary, we have looked at various models diffusion; all the models looked at have approximated the mathematical model of diffusion in discrete space with varying degrees of accuracy. However, when choosing which model to use, there is a trade-off between accuracy and complexity. Our results have shown that the Grajdeanu model is much more accurate than the linear model; however, we have also seen that it is much more effective to decrease the size of the grid spaces to increase the accuracy of the system than it is to use the Grajdeanu approach with a large neighbourhood. Additionally, when reducing the computational complexity, it is much more effective to increase the size of the grid spaces than it is to change the size of the neighbourhood.

As such, we suggest that the best way to model diffusion in discretised time and space is to use the linear model, and change the grid size according to the dependency of the agent based model on the accuracy of diffusion.

Acknowledgment

The authors would like to thank...

References

[1] J. Cosgrove, J. Butler, K. Alden, M. Read, V. Kumar, L. Cucurull-Sanchez, J. Timmis, and M. Coles, "Agent-based modeling in systems pharmacology," *CPT: Pharmacometrics and Systems Pharmacology*, vol. 4, no. 11, pp. 615–629, 2015.

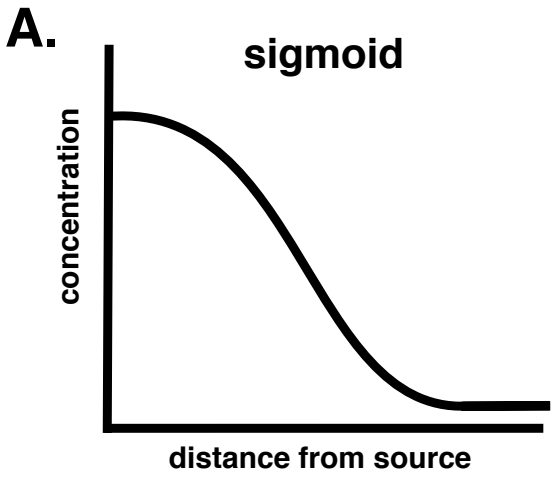
[2] J. O. Dada and P. Mendes, "Multi-scale modelling and simulation in systems biology," *Integrative Biology*, vol. 3, no. 2, pp. 86–96, 2011. [Online]. Available: http://pubs.rsc.org/en/content/articlelanding/2011/ib/c0ib00075b

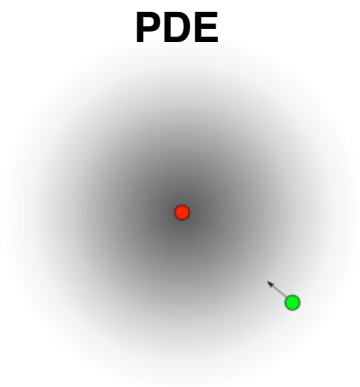
[3] G. Bocharov, A. Danilov, Y. Vassilevski, G. Marchuk, V. Chereshnev, and B. Ludewig, "Reaction-Diffusion Modelling of Interferon Distribution in Secondary Lymphoid Organs," *Mathematical Modelling of Natural Phenomena*, vol. 6, no. 7, pp. 13–26, 2011. [Online]. Available: http://www.scopus.com/inward/record.url?eid=2-s2.0-79960934740&partnerID=tZOtx3y1

[4] Y. Mei, V. Abedi, A. Carbo, X. Zhang, P. Lu, C. Philipson, R. Hontecillas, S. Hoops, N. Liles, and J. Bassaganya-Riera, "Multiscale modeling of mucosal immune responses," *From IEEE International Conference on Bioinformatics and Biomedicine*, pp. 2–5, 2015.

[5] K. Alden, J. Timmis, P. S. Andrews, H. Veiga-Fernandes, and M. C. Coles, "Pairing experimentation and computational modeling to understand the role of tissue inducer cells in the development of lymphoid organs," *Frontiers in Immunology*, vol. 3, no. JUL, pp. 1–20, 2012.

[6] Z. Guo, P. M. A. Sloot, and J. C. Tay, "A hybrid agent-based approach for modeling microbiological systems," *Journal of Theoretical Biology*, vol. 255, no. 2, pp. 163–175, 2008.





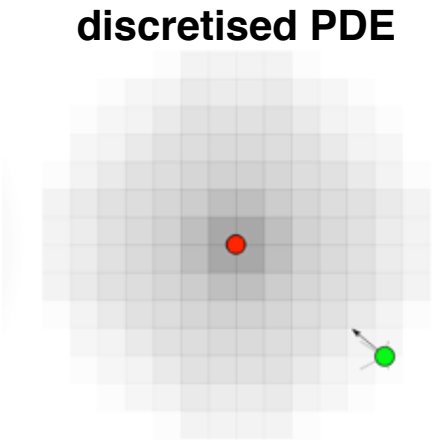


Fig. 1. Overview of the different approaches to implement diffusion in silico (A) Common approaches to implement diffusion in silico include representing chemokine concentrations as a sigmoid function which decays as the distance from the source increases. In addition, continuous and discretised partial differential equations have also been used.

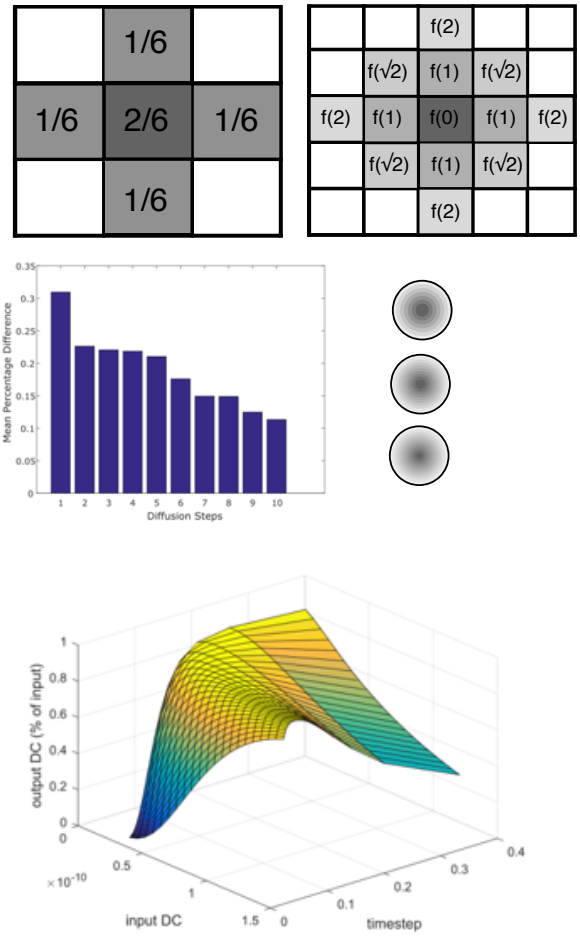


Fig. 2. Technical considerations when implementing diffusion within a discrete environment(B) The linear model of diffusion was simulated after a number diffusion steps, and the percentage difference was calculated based on a reference model without anisotropy and with the same diffusion coefficient. As the number of diffusion steps increases, the accuracy of the linear model improves; this is primarily because of a lessening of the effect of anisotropy.(C) A 3d plot showing the relationship between the chosen timestep length, the input diffusion coefficient, and the resulting diffusion coefficient of the model. For a given input diffusion coefficient, the timestep length has a non-linear relationship to the output coefficient. Likewise, for a given timestep length, the input diffusion coefficient has a non-linear relationship with the output diffusion coefficient. NEED TO COMBINE PARTS OF FIGURE 1 AND FIGURE 2 TO GET ACROSS SOME OF THE TECHNICAL ASPECTS. FIGURE 1A CAN BE ON ITS OWN THEN WE MENTION THAT WE FOCUS ON DIFFERENT ASPECTS OF DISCRETISED DIFFUSION

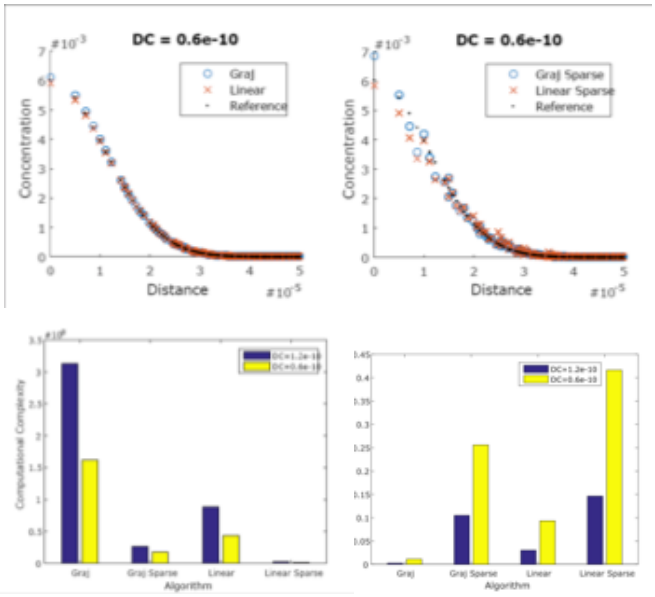


Fig. 3. (A)The computational complexity (left) and mean percentage difference (right) was calculated for four models of diffusion with two different diffusion coefficients. The models with sparse grids are less computational complex but less accurate than the models with dense grids, and the Grajdeanu models are more accurate but more computational complex then the linear model. Generally, the higher the accuracy, the higher the computational complexity.

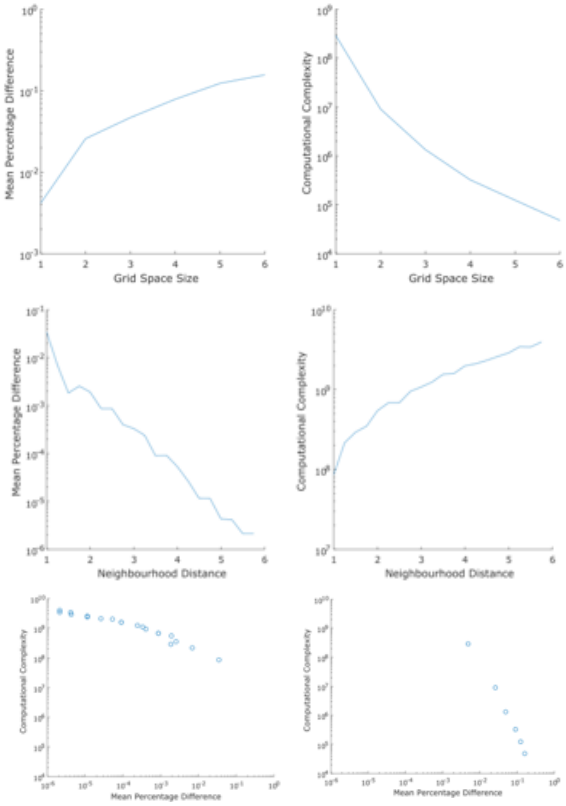


Fig. 4. Graphs showing the effects of increasing grid space size (top) and neighbourhood size (bottom) on the accuracy (left) and the computational complexity (right). The computational complexity increases and the grid space size decreases and as the neighbourhood size increases, and the accuracy increases as the neighbourhood size increases and the grid space size decreases. Changing the neighbourhood size has a larger effect on the accuracy, whereas changing the grid space size has a larger effect on the computational complexity. Graphs showing the relationship between accuracy and complexity of changing the neighbourhood size (left) and the grid space size (right). Changing the grid space size results in a much steeper reduction in computational complexity when compared to the mena percentage difference.