COMPUTATIONAL ANALYSIS OF CANCER SPHEROID INVADING THE MESOTHELIUM

Lim Shi Yue Phoebe

A0225620R

Undergraduate Research Opportunities in Science

PROJECT REPORT

submitted to the

Department of Biological Sciences

National University of Singapore

ZB3288 (4 MC)

April, 2023

2966 words

Table of Contents

Abstract

During ovarian cancer metastasis, cancer spheroids have been found to invade and clear the mesothelial layer. Though tissue-level models of spheroid invasion of the mesothelium have been proposed, the pattern of cell dynamics in spheroids during invasion has been relatively unexplored. Using tracked motion data of cells in invading and non-invading spheroids, temporal autocorrelation and spatial correlation analyses were carried out to quantify the pattern of cell migration. This study provides a theoretical basis for single cell-directed migration in invading spheroids, as well as collective cell migration in non-invading spheroids. Regional differences in the duration of directed cell migration show that the bottom of spheroids undergo invasive spreading. Results also suggest that either scattered cell migration or a combination of collective cell migration and neighbour exchange occur in invading spheroids.

130 words

1. Introduction

In ovarian cancer, the mesothelium, which is a single cell layer that covers the peritoneal cavity, is an early protective barrier against the metastatic colonisation of cancer tumours (Lengyel, 2010; Kenny et al., 2011). During metastasis, aggregated tumour cells form multicellular spheroids that attach to the peritoneal surface and invade the mesothelium. Present literature has found that cancer spheroids form metastatic sites and have a survival advantage over single cells (Al Habyan et al., 2018, pp. 5130-5131). Al Habyan et al. (2018, p. 5134) also demonstrated that tumour spheroids maintain E-cadherin heterogeneity that may facilitate late events in metastasis, such as peritoneal attachment and growth. Given that cancer spheroids have a seemingly greater role in metastasis as compared to single cells, this project focused on studying mesothelial invasion by ovarian cancer spheroids. In Iwanicki et al.'s (2011, pp. 146-147) in-vitro study of spheroid invasion of the mesothelium, cancer spheroids were found to attach and intercalate into the mesothelial layer, displacing mesothelial cells in the process. In addition, spheroid myosin IIA and IIB, together with $\alpha5\beta1$ integrin, were reported to be crucial for mesothelial clearance (Iwanicki et al., 2011).

As outlined above, the clinical relevance and spheroid-level mechanism of mesothelial invasion has been well-studied. Despite this, the pattern of cell dynamics in cancer spheroids during invasion of the mesothelium remains unclear. Moreover, while cell migration on a two-dimensional (2D) monolayer is well-understood (Lintz et al., 2017; Liu et al., 2022, p. 1; Pawluchin & Galic, 2022), the three-dimensional (3D) analysis of cell motion is less common but would be immensely valuable for understanding the mechanism of cell migration during spheroid invasion in a bona fide biological system.

This study hence aimed to quantify the pattern of cell migration during spheroid invasion of the mesothelium. Specifically, whether single cells within invading spheroids undergo directed migration, as well as whether clusters of cells in invading spheroids undergo collective migration, was of interest. Directed cell migration is the movement of individual cells in a particular direction and is typically mediated by chemical signals. Single cancer cells are thought to employ two main migration modes, namely amoeboid and mesenchymal migration. As characterised by Lintz et al. (2017, p. 1), amoeboid migration is defined by blebbing, weak adhesions and rapid motility, while the hallmarks of mesenchymal migration are strong stress fibers and slow speed. In many cases, successful tumour invasion is dependent on the directed migration of cancer cells through the extracellular matrix and tissues (Lintz et al., 2017, p. 1). On the other hand, collective cell migration involves a group of cells travelling together under certain patterns with a degree of coordination among their movement. In cancer, collective cell migration is fronted by a few leader cells and has been reported to have a greater potential for metastasis in some cancer types like lung cancer (Yang et al., 2019). Studies have also found that collectively migrating cancer cells are more aggressive and resistant to chemotherapy (Lintz et al., 2017, p. 5). Furthermore, collective cell migration is useful in understanding certain phase transitions in cancer tissues, such as the transition between unjammed and jammed states (Spatarelu et al., 2019, p. 3769). In the jammed state, cells are thought to be confined in a solid-like cage.

Investigating whether single cell-directed migration and collective cell migration occurs in ovarian cancer spheroids would thus shed light on the cellular mechanism behind spheroid invasion and its effect on cancer metastasis.

2. Materials and Methods

2.1. Tracked Cell Position Data

Cancer spheroids from the OVCA433 ovarian cancer cell line were cultured and seven experiments were set up. In four of these experiments (Experiments 1 to 4), the spheroids were allowed to invade H2b-mNeon-labelled mesothelial cells. The timeframe at which spheroid invasion occurred was recorded. As for the other three experiments (Experiments 5 to 7), integrin α5 was knocked down in mesothelial cells to prevent spheroid invasion. The spheroids were imaged and labelled using *StarDist-3D*, a pipeline that employs deep learning to detect and segment cell nuclei (Weigert et al., 2020). Their 3D positions were then tracked over timeframes of 3 minutes (with the exception of Experiment 5, which was tracked over timeframes of 5 minutes) using *Imaris*, an Interactive Microscopy Image Analysis software (*Imaris*, 2023), prior to my analysis. For the invading spheroids, only cell position data after invasion was considered.

2.2. Libraries Used for Data Analysis and Visualisation

The Python programming language was used for the analysis and visualisation of the tracked cell position data obtained in Section 2.1. In particular, the *NumPy* (Harris et al., 2020), *pandas* (McKinney, 2010) and *SciPy* (Virtanen et al., 2020) libraries were used for data processing and statistical analysis, and the *Matplotlib* (Hunter, 2007) and *seaborn* (Waskom, 2021) libraries were used for graph plotting.

2.3. Cell Velocity Calculation

Using the cell position data, absolute cell velocity was first calculated by taking the change in displacement over time. Relative velocity was then obtained by subtracting centroid velocity from the absolute velocity of each cell. This was done to eliminate the

effect of drift, which is an overall shift in the translational position of a cell. Finally, cell velocity was normalised to isolate the direction of cell motion. The eventual relative, normalised cell velocity was then used in subsequent analyses.

2.4. Temporal Autocorrelation Analysis

Temporal autocorrelation analysis was carried out to investigate the relationship between the velocities of individual cells over time. The autocorrelation function quantifies the self-similarity of a signal over varying lag times (dt) (Vilela et al., 2013, p. 264). Figure 1a outlines the steps that were performed on the data obtained from each experiment. In Step 3, the inner product between two normalised velocities can be interpreted as the correlation between them (Figure 1b) (Suga, 2018).

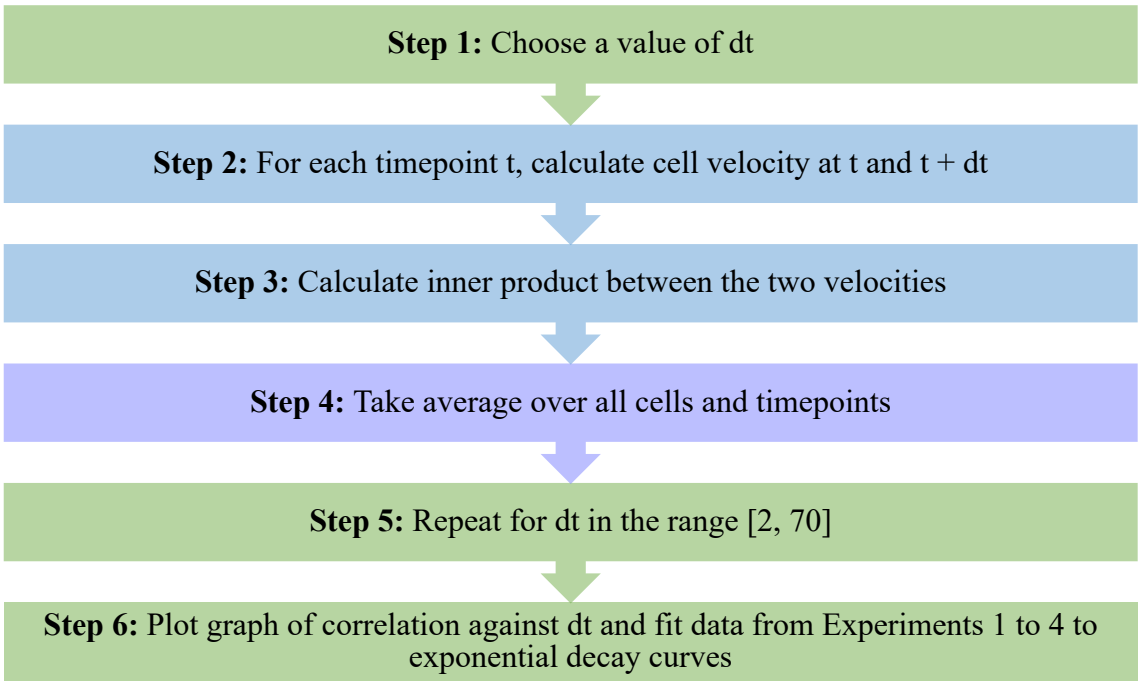


Figure 1a: Steps involved in the temporal autocorrelation analysis

$$r = \cos(\theta) = \frac{a \cdot b}{|a||b|}$$

Figure 1b: Deriving a correlation value from the inner product of two normalised vectors

2.4.1. Characteristic Timescale Analysis

Characteristic timescale analysis aimed to quantify 'temporal persistence' (Vilela et al., 2013, p. 264), or the duration which individual cell velocity persisted for in invading spheroids. This was done by carrying out the steps in Figure 2 on the graphs of correlation against lag time obtained in Section 2.4. This procedure was adapted from similar analyses done by other authors (Eilertsen et al., 2019, p. 29). The Savitzky-Golay filter in the first step is a processing method that performs a least-squares polynomial fit to a moving window of data points (Press & Teukolsky, 1990).

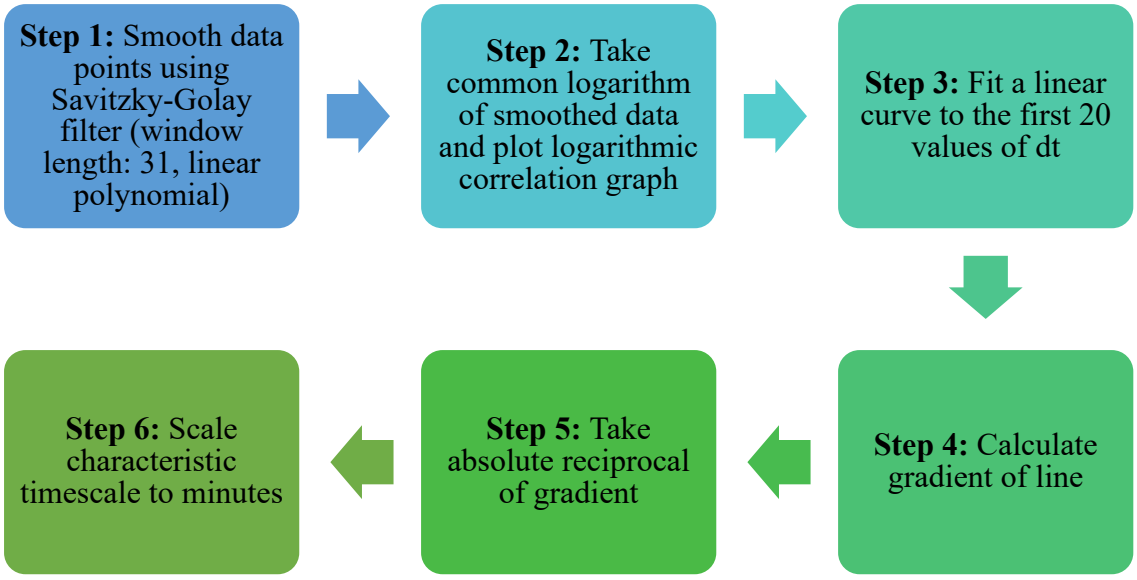


Figure 2: Steps involved in the calculation of characteristic timescale

2.4.2. Regional Analysis of Spheroids

To gain a better understanding of the regional cell dynamics in invading spheroids, the cells were separated based on their positions in the z-axis. Cells that had a z-position below the median z-position of the spheroid were considered to be in the bottom of the

spheroid, whereas the rest of the cells were considered to be in the top. The analyses in Sections 2.4 and 2.4.1 were then carried out on these two regions.

2.5. Spatial Correlation Analysis

From the cell velocities obtained in Section 2.3, spheroid rotation was removed using the approach in Figure 3a. The rationale for this will be discussed in Section 4.3. Subsequently, spatial correlation analysis was carried out by performing the steps outlined in Figure 3b. This analysis examined the relationship between the velocities of different cells at fixed points in time.

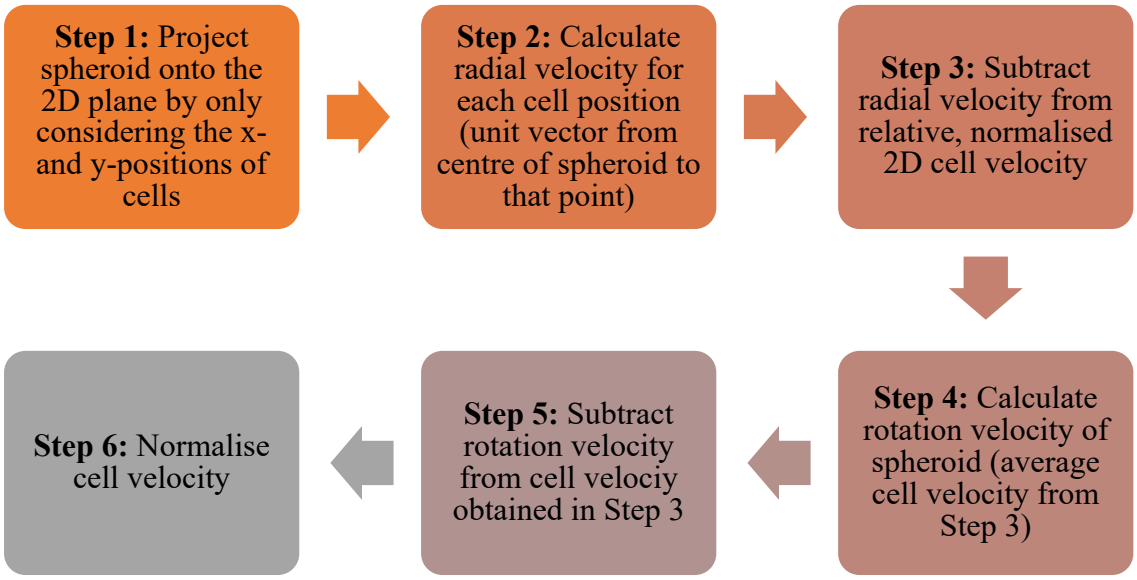


Figure 3a: Approach taken to remove the effect of global rotation

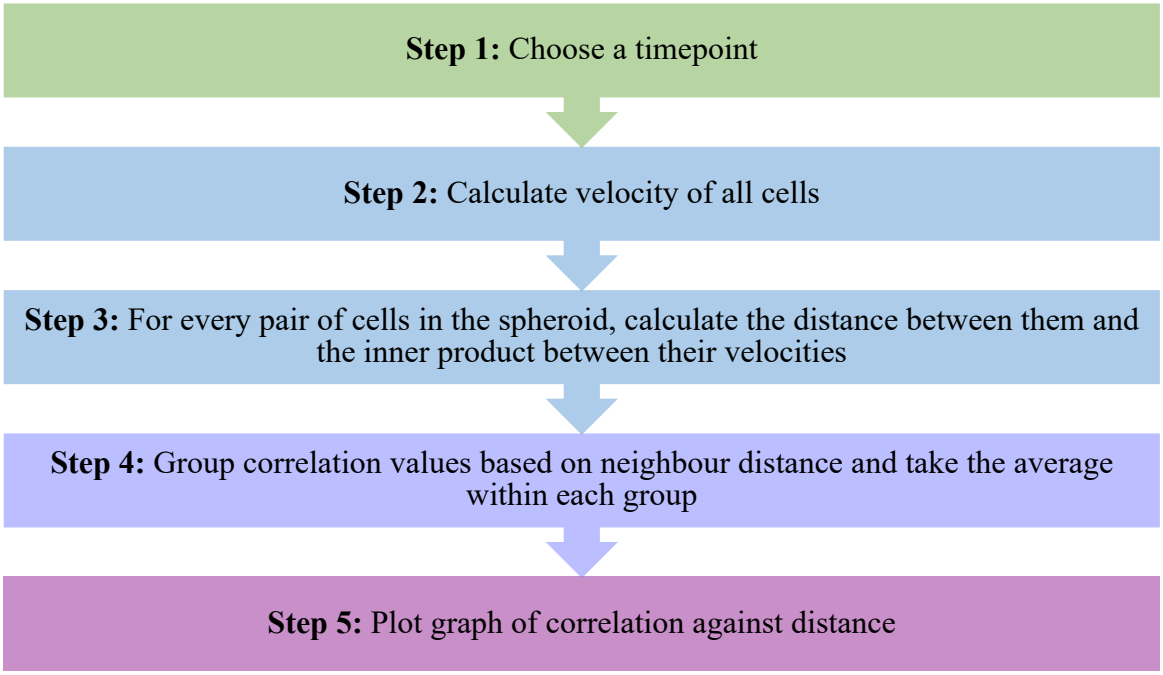


Figure 3b: Steps involved in the spatial correlation analysis

3. Results

3.1. Temporal Autocorrelation Functions of Invading and Non-Invading Spheroids

Figures 4a and b show one representative graph of the temporal autocorrelation function from the invading and non-invading experiments respectively. All the experiments with invading spheroids (Experiments 1 to 4) could be fitted to exponential decay curves while the non-invading experiments (Experiments 5 to 7) could not. The legend in figure 4a indicates the coefficients of the exponential equation ($ae^{-\frac{x}{b}} + c$) that the data points were fitted to.

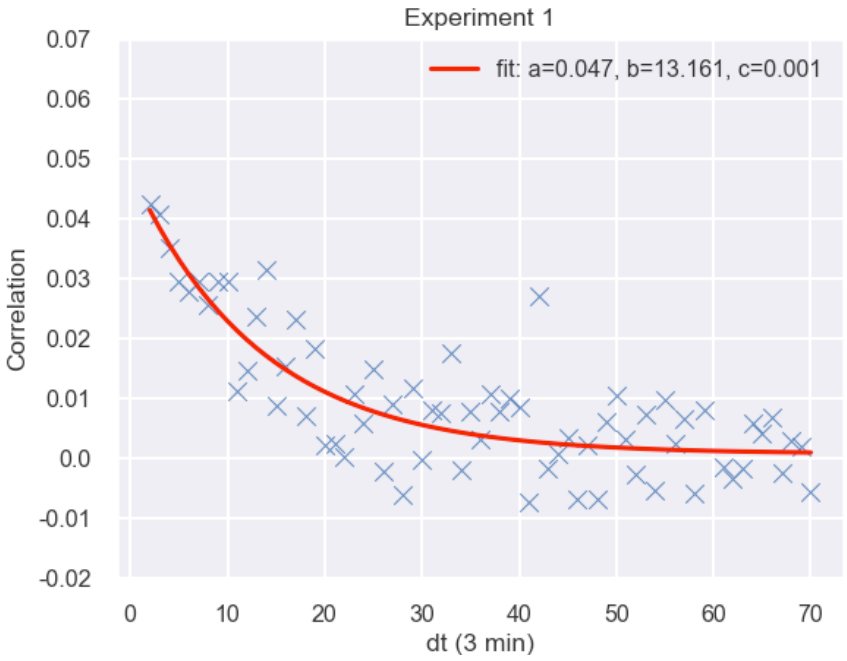


Figure 4a: Graph of the temporal autocorrelation function from one invading experiment

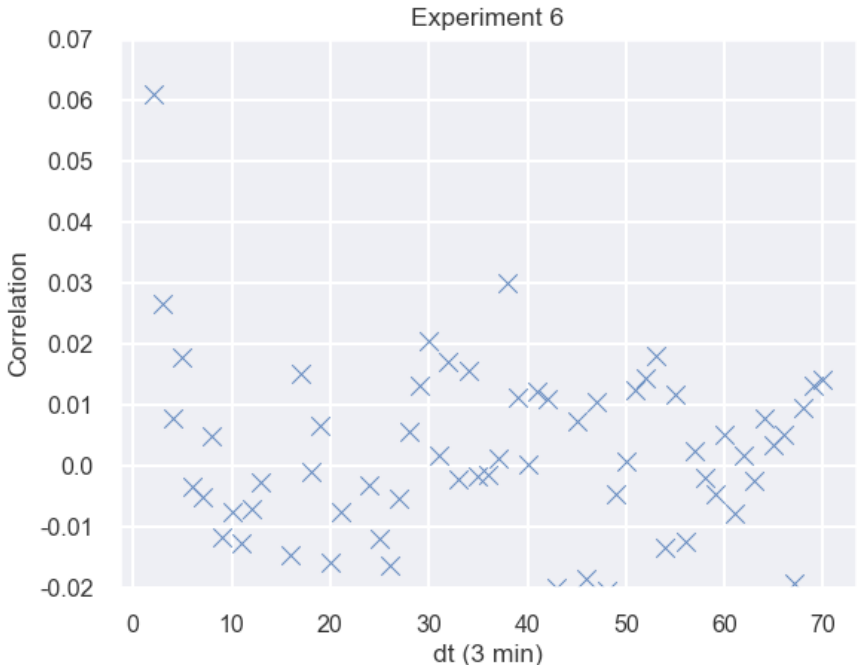


Figure 4b: Graph of the temporal autocorrelation function from one non-invading experiment

3.1.1. Characteristic Timescales of Invading Spheroids

The logarithmic graphs of correlation against lag time obtained from the method outlined in Section 2.4.1 are presented in Figures 5a to c. The overall characteristic timescales of the four invading experiments, as well as the characteristic timescales of the top and bottom of the four spheroids, are summarised in Table 1.

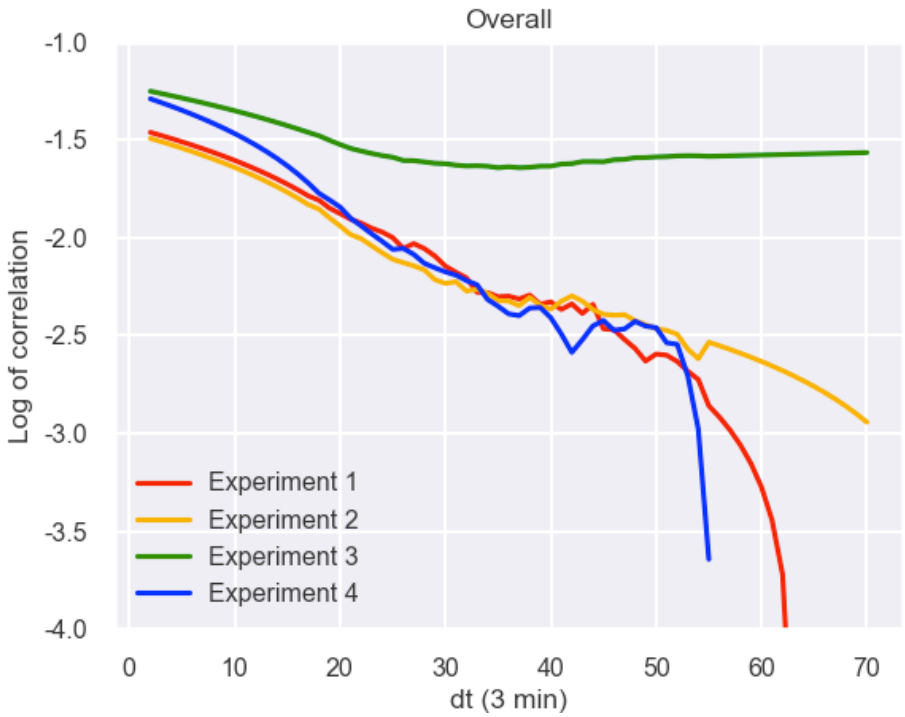


Figure 5a: Graph of the logarithmic temporal autocorrelation function from all invading experiments

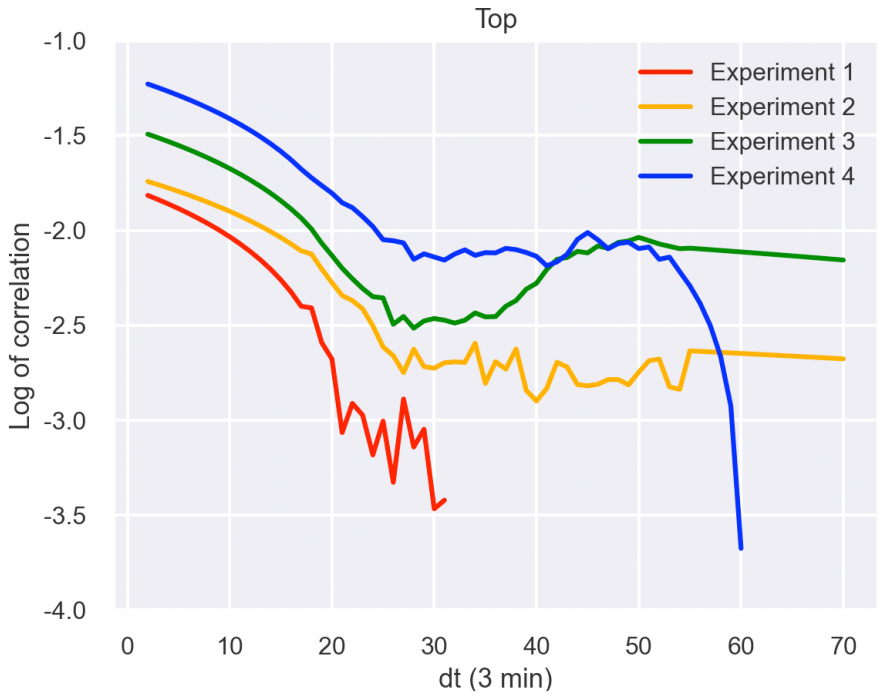


Figure 5b: Graph of the logarithmic temporal autocorrelation function for the top of spheroids from all invading experiments

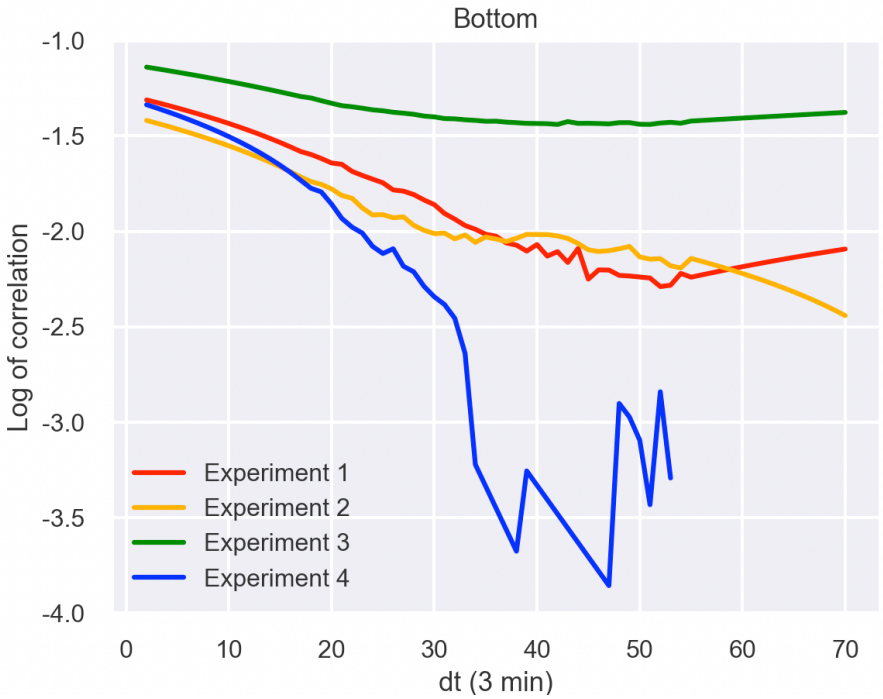


Figure 5c: Graph of the logarithmic temporal autocorrelation function for the bottom of spheroids from all invading experiments

Table 1: Overall, top and bottom characteristic timescales of all invading experiments

Characteristic timescale (min)	Overall	Top	Bottom
Experiment 1	131	68.0	162
Experiment 2	124	110	148
Experiment 3	202	89.4	286
Experiment 4	97.1	93.6	106
Average	139	90.4	175

3.2. Spatial Correlation Functions of Invading and Non-Invading Spheroids

From the analysis in Section 2.5, graphs of the spatial correlation functions from all the experiments were generated for multiple timepoints and the trend was similar throughout. Figure 6a shows the graph obtained from one representative invading experiment at the 120th timepoint after invasion while Figure 6b shows the graph obtained from one representative non-invading experiment at the 40th absolute timepoint. These timepoints were chosen as they were the approximate mid-points of the invading and non-invading experiments, taking into account that the experiments were tracked for varying durations. Regional analysis was also conducted for all the experiments but the results were inconsistent across the invading experiments (Appendix 7.3), and did not present a significant difference between the top and bottom of the spheroids in the case of the non-invading experiments (Appendix 7.4).

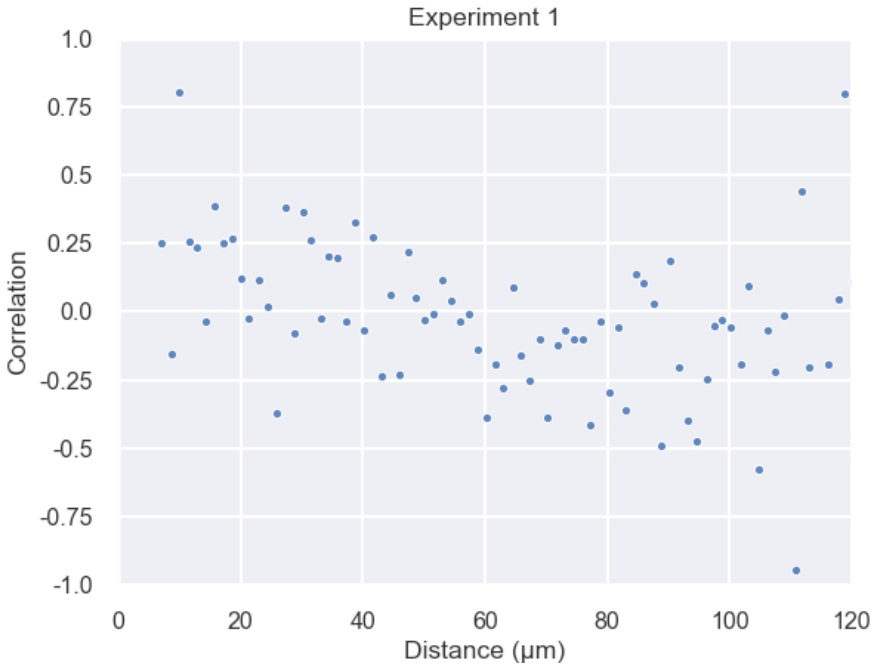


Figure 6a: Graph of the spatial correlation function from one invading experiment

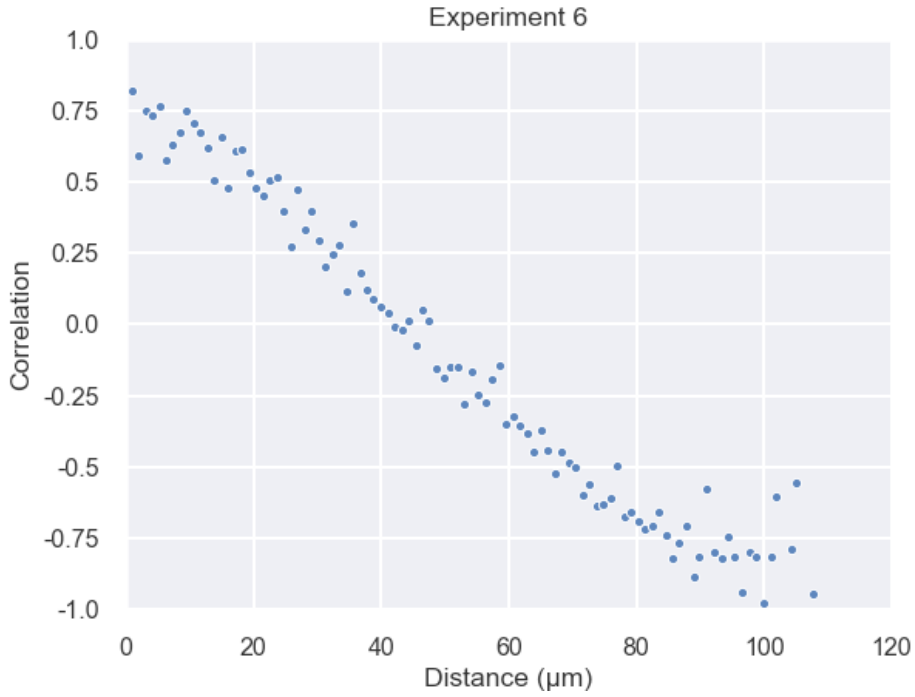


Figure 6b: Graph of the spatial correlation function from one non-invading experiment

4. Discussion

In this section, results obtained from the two main analyses will be examined separately. The temporal autocorrelation results obtained in Section 3.1 will be used to determine whether single cell-directed migration occurs within spheroids in Section 4.1. On the other hand, the spatial correlation results obtained in Section 3.2 will be used to ascertain whether collective cell migration occurs within spheroids in Section 4.2.

4.1. Cell-Directed Migration in Invading Spheroids

From Section 3.1, the temporal autocorrelation functions of the invading experiments followed a trend of exponential decay (Figure 4a) whereas the correlation values for the non-invading experiments fluctuated around 0 (Figure 4b). For the non-invading experiments, the uncorrelated cell velocities at consecutive timepoints meant that there was an absence of any directed cell motion over time and that the cells were exhibiting random motion within the spheroids.

As for cells in invading spheroids, their velocities were self-correlated at short timescales, indicating that cell migration in invading spheroids was directed. Other authors have established that directed cell migration occurs as a series of processes, including gradient sensing, the establishment of cell polarity and the conversion of traction forces to drive cell motion in accordance with the molecular clutch framework (Fortunato & Sunyer, 2022). In this study, the cells could be individually migrating out of the spheroid as they spread on the mesothelium. The decay in the correlation values at longer timescales can be understood intuitively as a cell would be moving in a direction that differs from its direction of motion many timepoints ago, causing individual cell velocities at two timepoints far away from each other to be weakly correlated.

As shown in Table 1, the average duration of directed migration in the four invading spheroids was 139 minutes.

4.1.1. Regional Difference in Duration of Cell-Directed Migration

Comparing Figures 5b and c, the gradients of the logarithmic temporal autocorrelation functions at short lag times for the bottom of spheroids was observed to be gentler than that of the top of spheroids. The characteristic timescales of the respective regions of the spheroids, reflected in Table 1, support this observation, with cells in the bottom of spheroids having an average duration of directed migration of 175 minutes, which is much longer than the average duration of 90.4 minutes for cells in the top of spheroids. This result supports existing observations that spheroids invade and spread on the mesothelium from the bottom, and that mesothelial clearance occurs under cancer spheroids (Iwanicki et al., 2011, p. 152).

4.2. Collective Cell Migration in Invading and Non-Invading Spheroids

The results of the spatial correlation analysis shown in Section 3.2 reflect that the velocities of cells that are relatively close to each other in non-invading spheroids are highly correlated. The correlation values in Figure 6b range from around 0.25 to 0.8 for neighbouring cell distances of up to 30 μm. This indicates that cells in non-invading spheroids migrate collectively. For invading spheroids, the correlation values in Figure 6a fluctuate around 0 and largely range from -0.4 to 0.4, showing a lack of correlation between the direction of motion of neighbouring cells. This result suggests that cell migration in invading spheroids does not appear to be collective, which is particularly puzzling as Section 4.1 found that individual cells in invading spheroids undergo directed migration. In directed cell migration, cells are traditionally thought to be tightly

coordinated (Fortunato & Sunyer, 2022, p. 548), and hence would be expected to migrate collectively in a single direction. Sections 4.2.1 and 4.2.2 present two possible explanations for this finding.

4.2.1. Scattered Migration in Invading Spheroids

The first possibility is that cells in invading spheroids migrate in a scattered and uncoordinated manner relative to neighbouring cells. Shellard and Mayor (2019, p. 4) reported that 'individually migrating cells tend to be randomly polarised' and undergo uncoordinated migration in the absence of external signals such as chemokines. They also characterise collective cell migration as both coordinated, in that neighbouring velocities are highly correlated, and cooperative, in that neighbouring cells influence each other during migration (Shellard & Mayor, 2019, p. 1). From the quantitative analyses performed above, it would perhaps be more appropriate to only conclude that cell migration in non-invading spheroids is coordinated. It should also be noted that a spectrum of cell migration patterns exist, thus it is possible that cells in invading spheroids migrate in a directed manner in response to certain chemical cues but lack the coordinated aspect of collective cell migration.

4.2.2. Neighbour Exchange in Invading Spheroids

Alternatively, cells in invading spheroids could be migrating collectively but undergo a process known as neighbour exchange. Neighbour exchange results from a topology change that occurs with the remodelling of cell-cell contacts when an epithelial cell loses or gains new contact with neighbouring cells (Rauzi, 2020, p. 1). During neighbour exchange, cell intercalation leads to the shrinking of one cell junction and the formation and extension of a new cell junction (Rauzi, 2020, p. 2). It has been found to be important

for collective cell migration through dense environments, and this is especially so for invading cancer cells migrating through narrow spaces in rigid tissues (Das et al., 2021, p. 2). This would be highly applicable to ovarian cancer spheroids that intercalate and squeeze between mesothelial cells during invasion of the mesothelial layer.

Consequently, neighbour exchange would explain the weak correlation between the direction of motion of neighbouring cells since cells in invading spheroids are constantly rearranging themselves and interchanging neighbours. Directed and collective migration of cell clusters in invading spheroids could thus still be occurring despite the results observed in Figure 6a.

4.3. Limitations

This study had a number of limitations. Firstly, the correlation data obtained in Section 3 had been greatly summarised before interpretation. In Step 4 of both the temporal autocorrelation (Figure 1a) and spatial correlation analyses (Figure 3b), average cell velocity was taken either over all cells and timepoints or over all cells within each group. This could cause positive and negative correlation values to negate each other, leading to a correlation value of a lower magnitude that did not accurately reflect the correlation between individual cell velocities. One way to mitigate this would be to plot graphs of the correlation functions of individual cells in the spheroids.

Secondly, it should be noted that correlation values obtained from the temporal autocorrelation analysis in Section 3.1 were generally low (less than 0.1), especially when compared to the results obtained from the spatial correlation analysis in Section 3.2. Nevertheless, the correlation values for invading spheroids were still higher as compared

to the non-invading spheroids. These low correlation values could also be due to the average being taken over all cells and timepoints.

In addition, the appropriateness of the exponential fit utilised in Section 3.1 should be critically evaluated to quantify the duration of directed migration more accurately. As outlined in Section 2.4.1, the data points of the invading experiments had to first be smoothed before their common logarithm was taken, even though the data was expected to follow an exponential decay trend with equation $ae^{-\frac{x}{b}} + c$. These methods were performed as the initial data was noisy and a natural logarithm of the data points did not yield a good fit. For some experiments such as Experiments 3 and 4, the autocorrelation values did not tend to 0 at longer timescales, in particular from lag times of 40 units onwards (Appendix 7.1). This suggests that an exponential fit could only be suitable up to a certain lag time. In other words, fitting of the data points to two separate mathematical functions should be done to ensure a more accurate quantification of the spheroids' characteristic timescales.

Importantly, in non-invading spheroids, the spatial correlation values obtained for neighbour distances of 40 μm onwards (Figure 6b) were observed to be negative, which could not be intuitively understood in terms of the dynamics of cell migration. Initially, these negative correlation values were hypothesised to arise from the rotation of entire spheroids, which explains why the effect of spheroid rotation was removed via the approach in Figure 3a. However, the negative correlation values remained, which is a result that this study could not resolve via a theoretical approach.

Lastly, experimental outliers, specifically Experiments 3 and 5, were present in the results of the temporal autocorrelation analysis. The logarithmic graph for Experiment 3 in Figure 5a did not follow the expected linear trend and the correlation values for

Experiment 5 in Appendix 7.1 were found to be relatively high compared to the rest of the non-invading experiments. The robustness of the analyses in this project could thus be further improved by considering more experiments.

4.4. Future Work

Some future work that could be done to supplement this project includes cell position tracking over more minute timescales, such as in intervals of 1 minute. This is because at very short timescales, cells in non-invading spheroids are still expected to undergo some degree of directed motion. Hence, the duration of directed cell migration in invading spheroids could be more precisely determined by comparing the lag time at which only cells in invading spheroids were observed to have high autocorrelation values.

Moreover, the explanations proposed in Sections 4.1 and 4.2 are mainly theoretical, especially for Sections 4.2.1 and 4.2.2. To ascertain whether directed and collective migration or neighbour exchange occurs in cancer spheroids, cell imaging data and experimental observations should be considered together with the quantitative cell motion data from this study.

Finally, it would be pertinent to investigate whether cell number in invading spheroids affects mesothelial invasion to ensure that it does not contribute as a confounder towards the analyses in this work.

5. References

Al Habyan, S., Kalos, C., Szymborski, J., & McCaffrey, L. (2018). Multicellular detachment generates metastatic spheroids during intra-abdominal dissemination in epithelial ovarian cancer. *Oncogene*, *37*(37), 5127–5135. https://doi.org/10.1038/s41388-018-0317-x

Das, A., Sastry, S., & Bi, D. (2021). Controlled neighbor exchanges drive glassy behavior, intermittency, and cell streaming in epithelial tissues. *Physical Review X*, *11*(4), 041037. https://doi.org/10.1103/PhysRevX.11.041037

Eilertsen, J., Stroberg, W., & Schnell, S. (2019). Characteristic, completion or matching timescales? An analysis of temporary boundaries in enzyme kinetics. *Journal of Theoretical Biology*, *481*, 28–43. https://doi.org/10.1016/j.jtbi.2019.01.005

Fortunato, I. C., & Sunyer, R. (2022). The forces behind directed cell migration. *Biophysica*, *2*(4), 548–563. https://doi.org/10.3390/biophysica2040046

Harris, C. R., Millman, K. J., Van Der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., Van Kerkwijk, M. H., Brett, M., Haldane, A., Del Río, J. F., Wiebe, M., Peterson, P., … Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, *585*(7825), 357–362. https://doi.org/10.1038/s41586-020-2649-2

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, *9*(3), 90–95. https://doi.org/10.1109/MCSE.2007.55

Imaris (9.9.1). (2023). Oxford Instruments.

Iwanicki, M. P., Davidowitz, R. A., Ng, M. R., Besser, A., Muranen, T., Merritt, M., Danuser, G., Ince, T., & Brugge, J. S. (2011). Ovarian cancer spheroids use myosin-generated force to clear the mesothelium. *Cancer Discovery*, *1*(2), 144–157. https://doi.org/10.1158/2159-8274.CD-11-0010

Kenny, H. A., Nieman, K. M., Mitra, A. K., & Lengyel, E. (2011). The first line of intra-abdominal metastatic attack: Breaching the mesothelial cell layer. *Cancer Discovery*, *1*(2), 100–102. https://doi.org/10.1158/2159-8290.CD-11-0117

Lengyel, E. (2010). Ovarian cancer development and metastasis. *The American Journal of Pathology*, *177*(3), 1053–1064. https://doi.org/10.2353/ajpath.2010.100105

Lintz, M., Muñoz, A., & Reinhart-King, C. A. (2017). The mechanics of single cell and collective migration of tumor cells. *Journal of Biomechanical Engineering*, *139*(2), 021005. https://doi.org/10.1115/1.4035121

Liu, Q., Muralidharan, A., Saateh, A., Ding, Z., Ten Dijke, P., & Boukany, P. E. (2022). A programmable multifunctional 3D cancer cell invasion micro platform. *Small*, *18*(20), 2107757. https://doi.org/10.1002/smll.202107757

McKinney, W. (2010). Data structures for statistical computing in Python. *Proceedings of the 9th Python in Science Conference*, *445*, 56–61. https://doi.org/10.25080/Majora-92bf1922-00a

Pawluchin, A., & Galic, M. (2022). Moving through a changing world: Single cell migration in 2D vs. 3D. *Frontiers in Cell and Developmental Biology*, *10*. https://www.frontiersin.org/articles/10.3389/fcell.2022.1080995

Press, W. H., & Teukolsky, S. A. (1990). Savitzky-Golay smoothing filters. *Computers in Physics*, *4*(6), 669. https://doi.org/10.1063/1.4822961

Rauzi, M. (2020). Cell intercalation in a simple epithelium. *Philosophical Transactions of the Royal Society B: Biological Sciences*, *375*(1809), 20190552. https://doi.org/10.1098/rstb.2019.0552

Shellard, A., & Mayor, R. (2019). Supracellular migration – beyond collective cell migration. *Journal of Cell Science*, *132*(8), jcs226142. https://doi.org/10.1242/jcs.226142

Spatarelu, C.-P., Zhang, H., Nguyen, D. T., Han, X., Liu, R., Guo, Q., Notbohm, J., Fan, J., Liu, L., & Chen, Z. (2019). Biomechanics of collective cell migration in cancer progression: Experimental and computational methods. *ACS Biomaterials Science & Engineering*, *5*(8), 3766–3787. https://doi.org/10.1021/acsbiomaterials.8b01428

Suga, N. (2018, September 14). Geometric interpretation of the correlation between two variables. *Medium*. https://medium.com/@ns2586/geometric-interpretation-of-the-correlation-between-two-variables-4011fb3ea18e

Vilela, M., Halidi, N., Besson, S., Elliott, H., Hahn, K., Tytell, J., & Danuser, G. (2013). Chapter Nine—Fluctuation analysis of activity biosensor images for the study of information flow in signaling pathways. In S. Y. Tetin (Ed.), *Methods in Enzymology* (Vol. 519, pp. 253–276). Academic Press. https://doi.org/10.1016/B978-0-12-405539-1.00009-9

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., Van Der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., … Vázquez-Baeza, Y. (2020). SciPy 1.0: Fundamental algorithms for scientific computing in Python. *Nature Methods*, *17*(3), 261–272. https://doi.org/10.1038/s41592-019-0686-2

Waskom, M. (2021). Seaborn: Statistical data visualization. *Journal of Open Source Software*, *6*(60), 3021. https://doi.org/10.21105/joss.03021

Weigert, M., Schmidt, U., Haase, R., Sugawara, K., & Myers, G. (2020). Star-convex polyhedra for 3D object detection and segmentation in microscopy. *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 3655–3662. https://doi.org/10.1109/WACV45572.2020.9093435

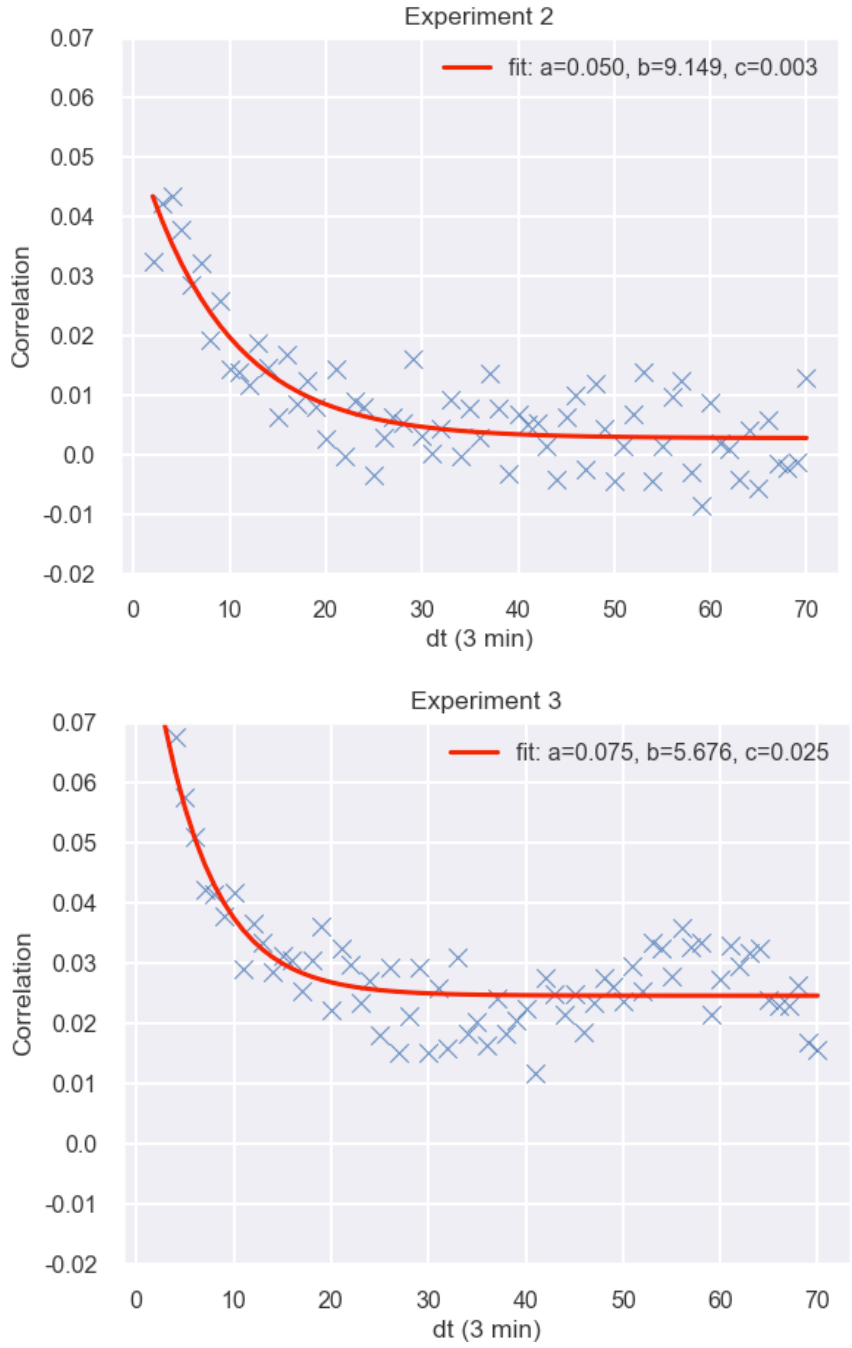
Yang, Y., Zheng, H., Zhan, Y., & Fan, S. (2019). An emerging tumor invasion mechanism about the collective cell migration. *American Journal of Translational Research*, *11*(9), 5301–5312. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6789225/

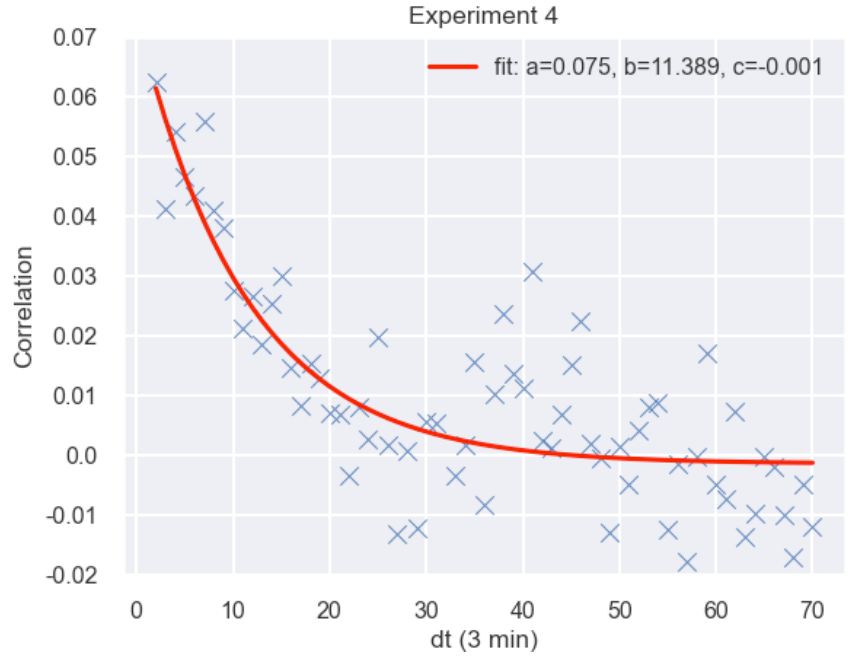
6. Acknowledgements

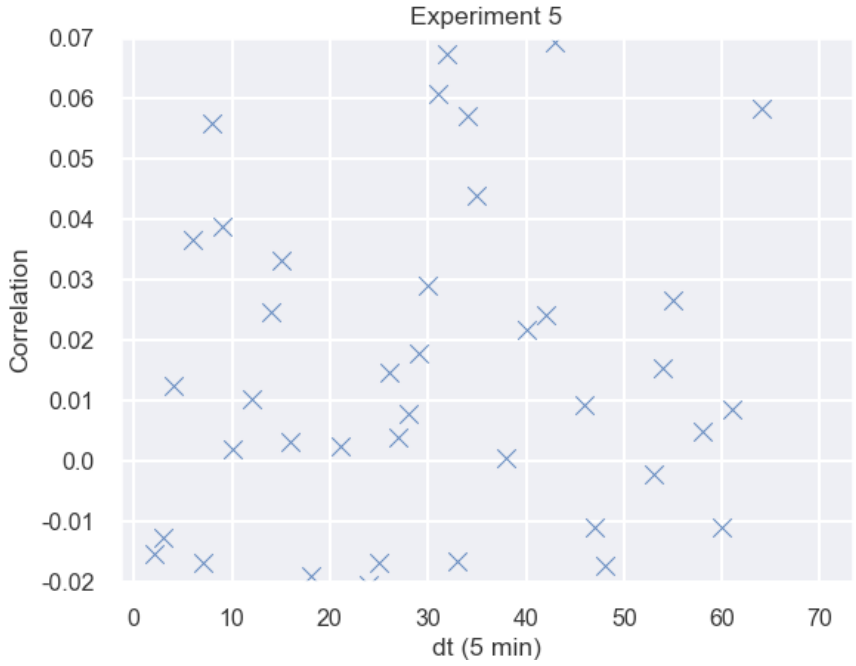
I would like to express my gratitude to various individuals, including Professor Low Boon Chuan and his laboratory members at the Mechanobiology Institute, who gave me much-needed feedback during laboratory meetings, as well as Celestine and Daniella, who were responsible for the spheroid segmentation and tracking prior to my analysis. I would also like to thank Dr Tetsuya Hiraiwa and Dr Lou Yuting, who guided me on the Physics concepts and computational methods relevant to my analysis. Last but not least, much appreciation goes to my mentor, Dr Selwin Wu, who helped me better understand the biological significance of my project and was always ready to give me feedback on my work.

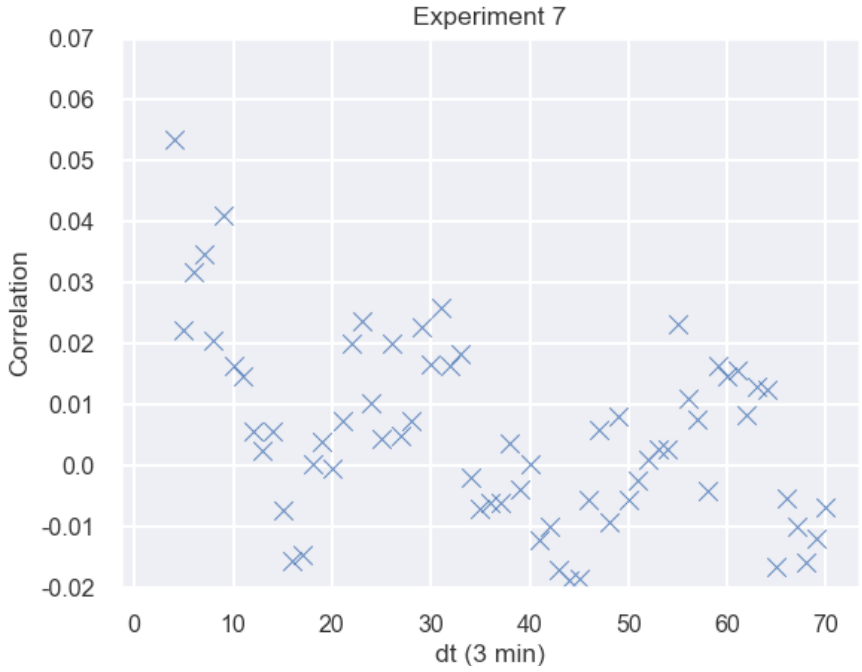
7. Appendices

Appendix 7.1. Temporal Autocorrelation Functions of Experiments 2, 3, 4, 5 and 7

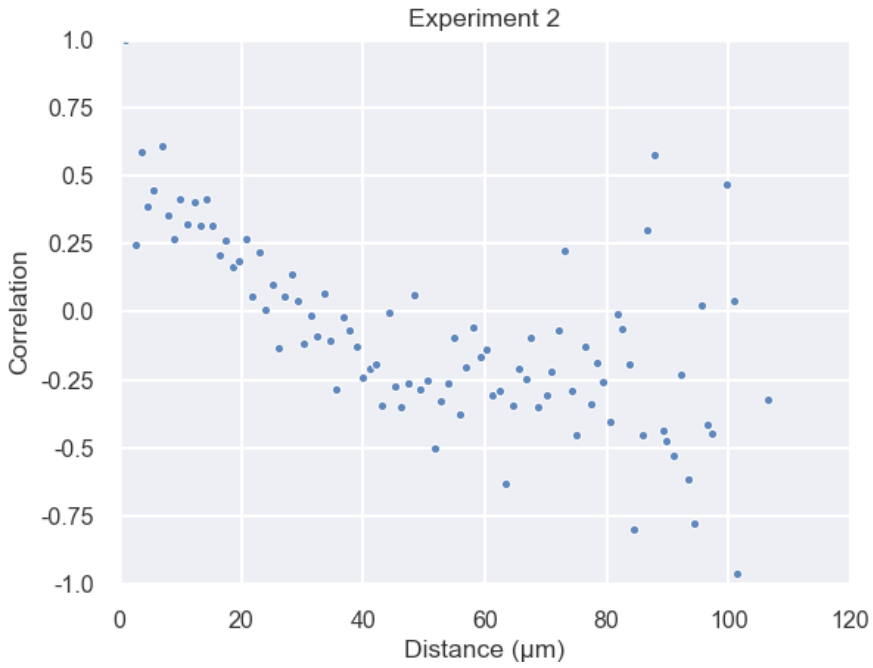


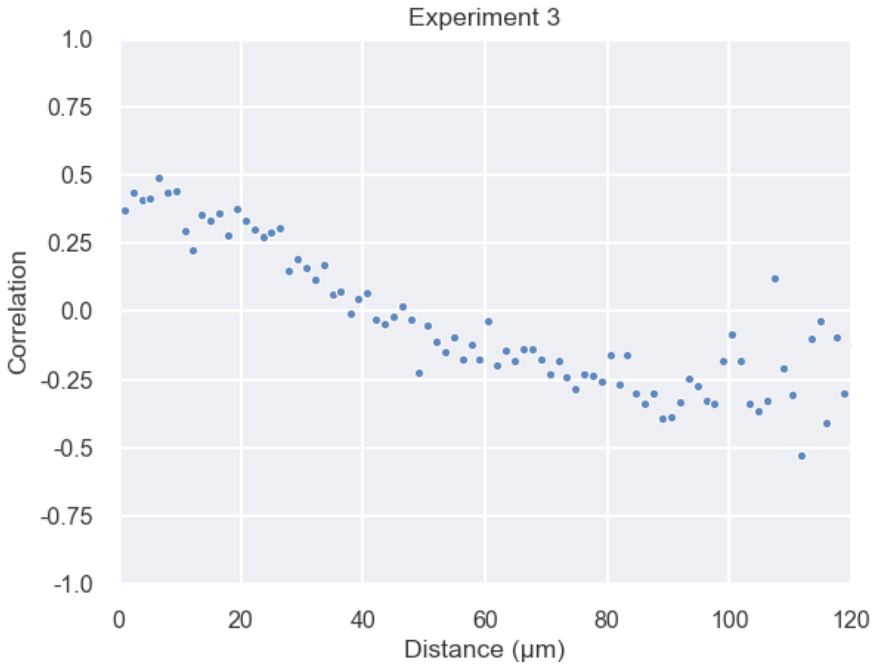

Experiment 4
fit: a=0.075, b=11.389, c=-0.001
Correlation
dt (3 min)
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0.0
-0.01
-0.02
0
10
20
30
40
50
60
70

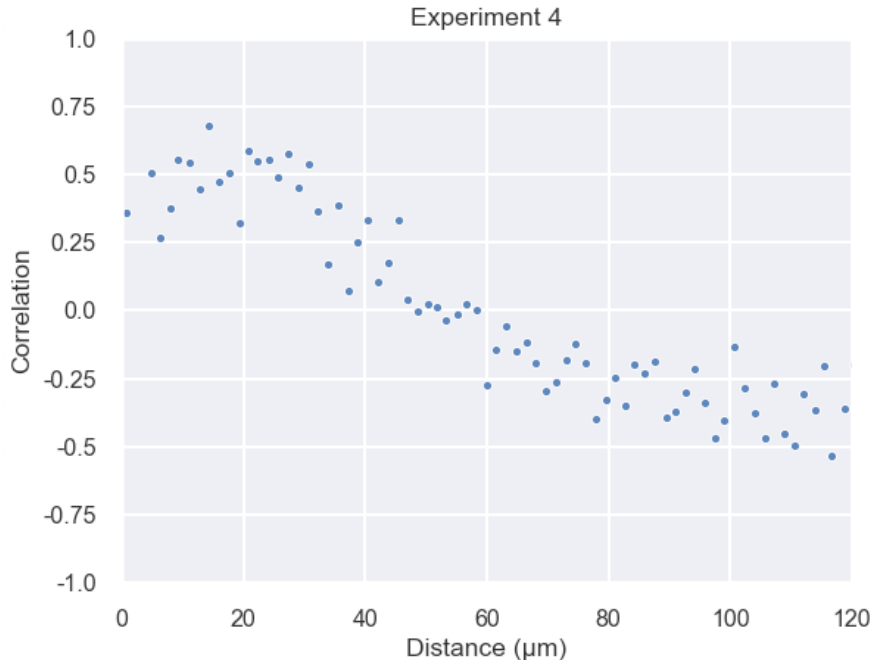

Experiment 5
Correlation
dt (5 min)
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0.0
-0.01
-0.02
0
10
20
30
40
50
60
70

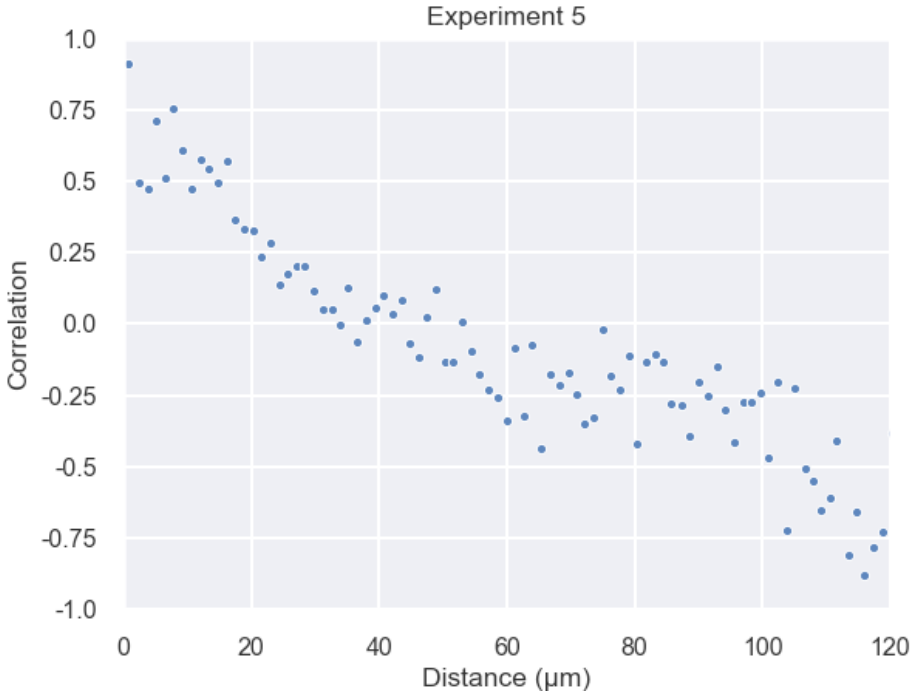

Experiment 7
Correlation
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0.0
-0.01
-0.02
0
10
20
30
40
50
60
70
dt (3 min)

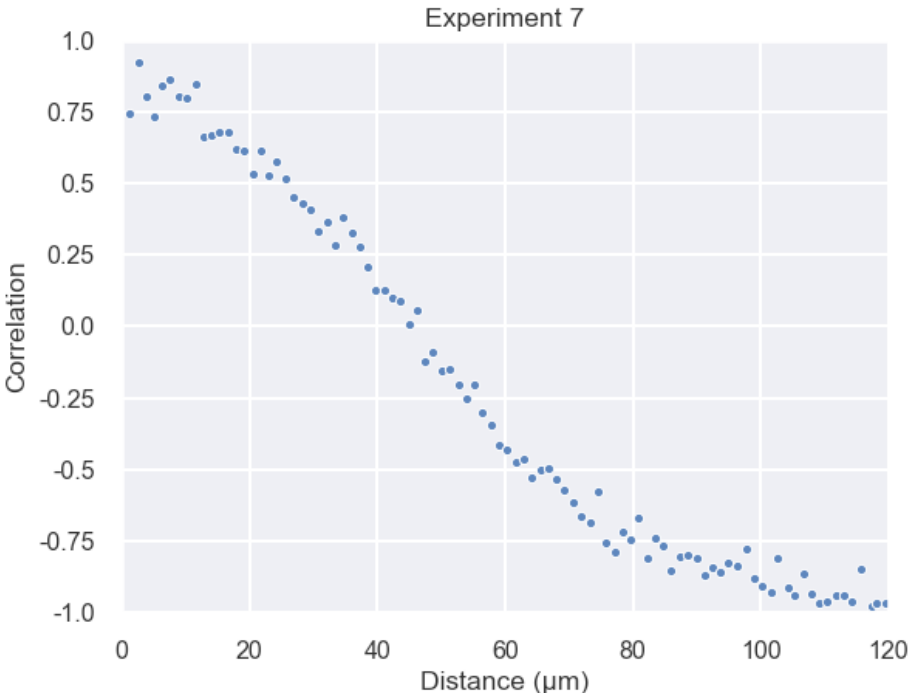
Appendix 7.2. Spatial Correlation Functions of Experiments 2, 3, 4, 5 and 7



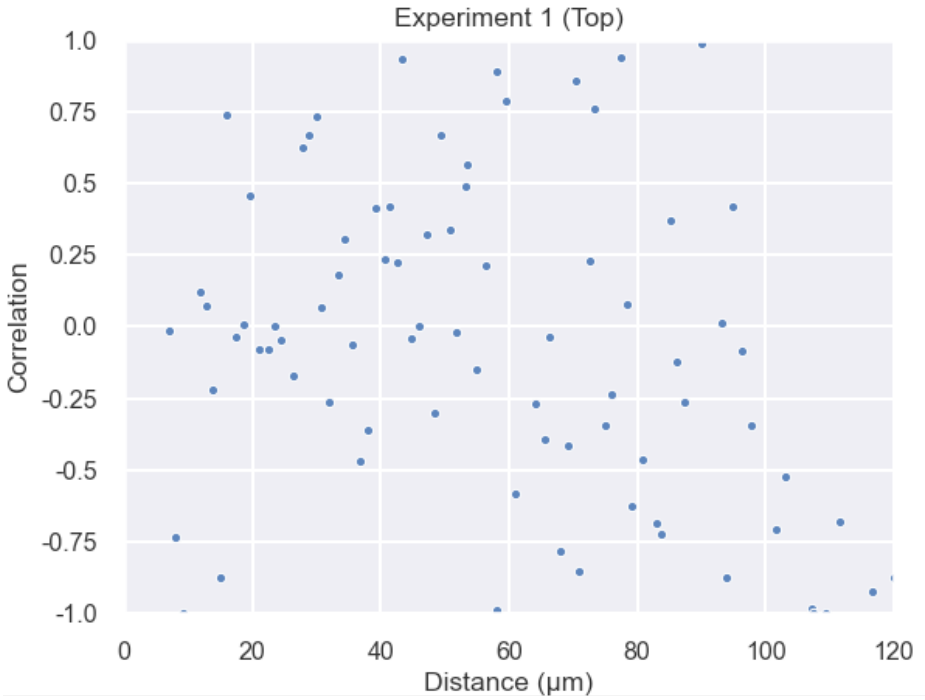


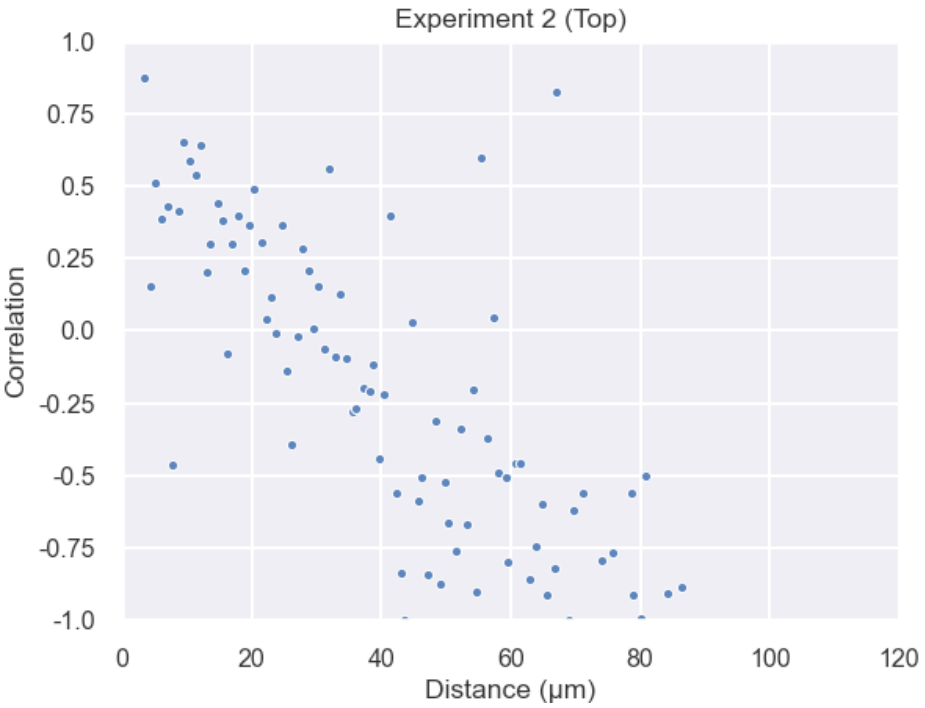

Experiment 4
Correlation
1.0
0.75
0.5
0.25
0.0
-0.25
-0.5
-0.75
-1.0
0
20
40
60
80
100
120
Distance (µm)

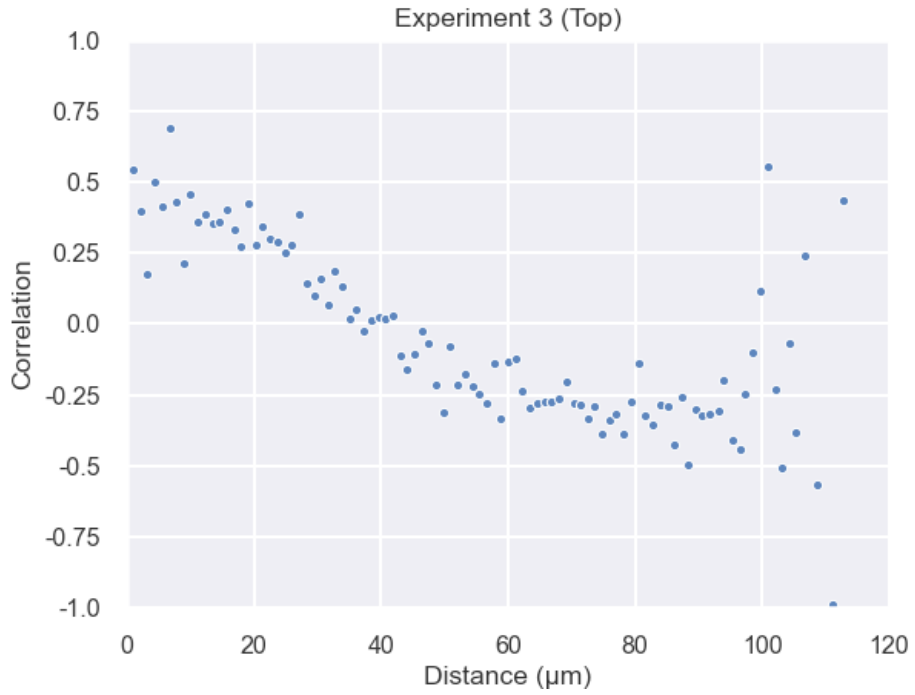

Experiment 5
Correlation
1.0
0.75
0.5
0.25
0.0
-0.25
-0.5
-0.75
-1.0
0
20
40
60
80
100
120
Distance (µm)

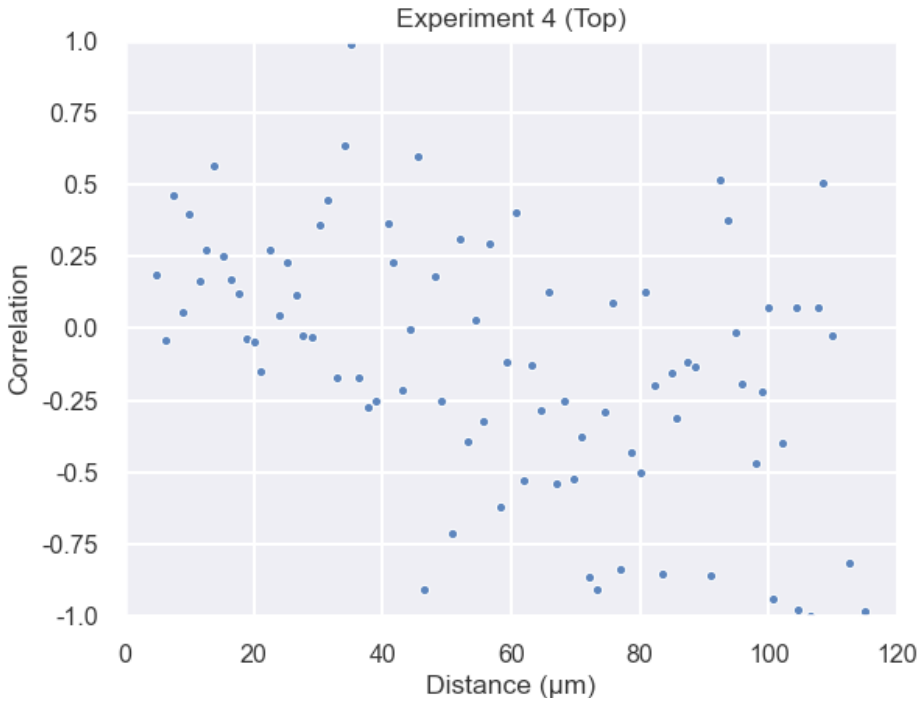

Experiment 7
Correlation
1.0
0.75
0.5
0.25
0.0
-0.25
-0.5
-0.75
-1.0
0
20
40
60
80
100
120
Distance (µm)

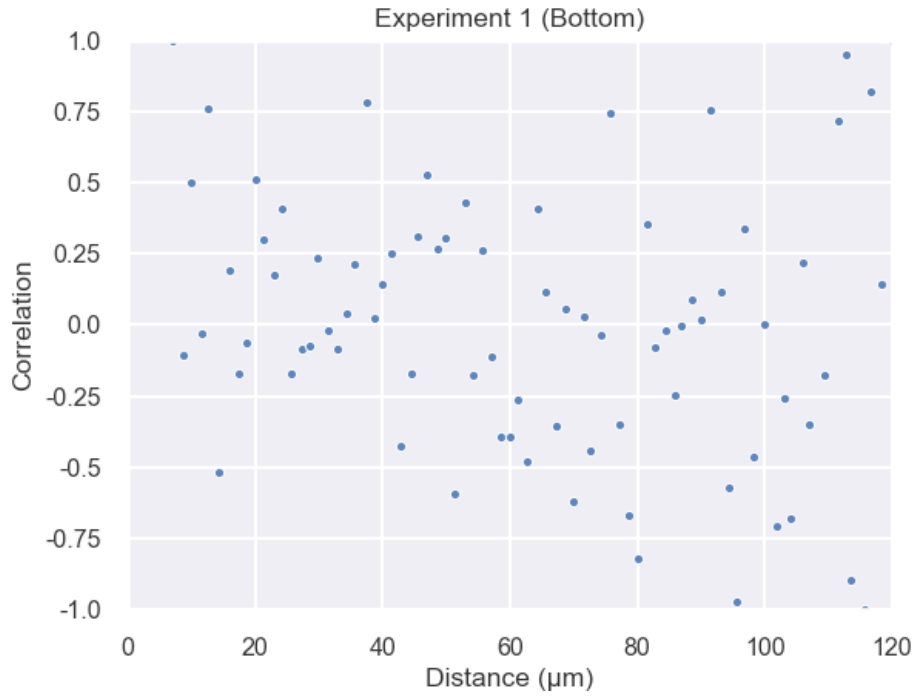
Appendix 7.3. Spatial Correlation Functions of Top and Bottom of Spheroids in Experiments 1, 2, 3 and 4

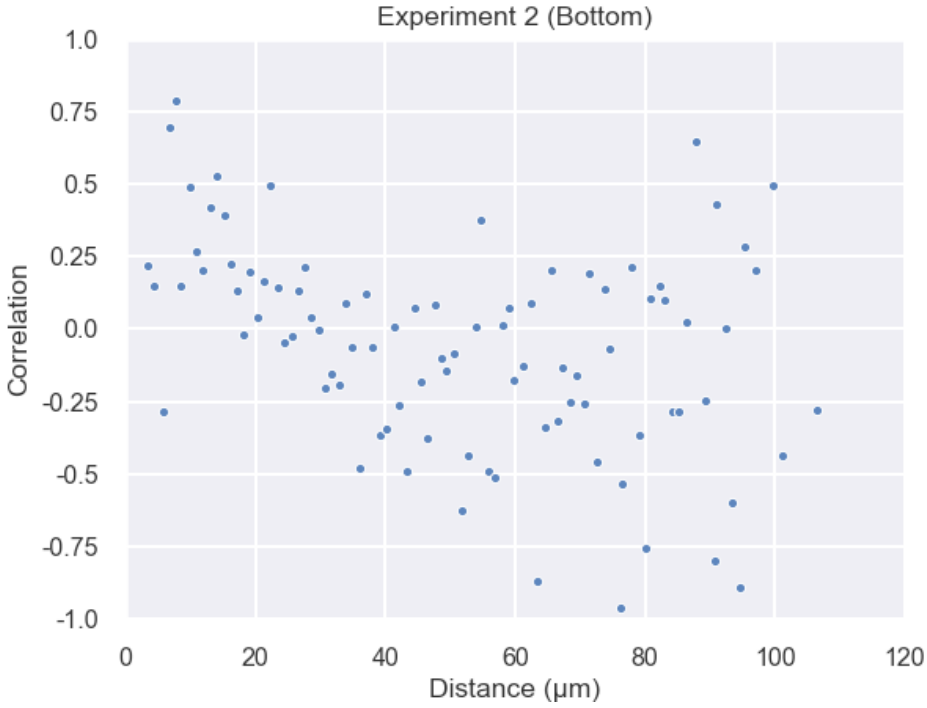


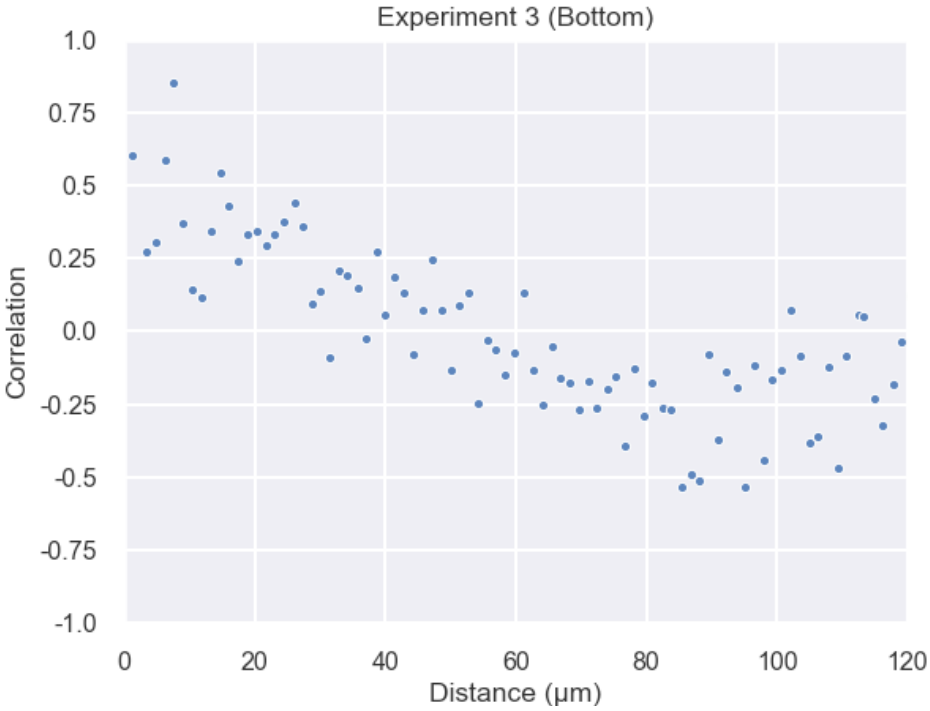


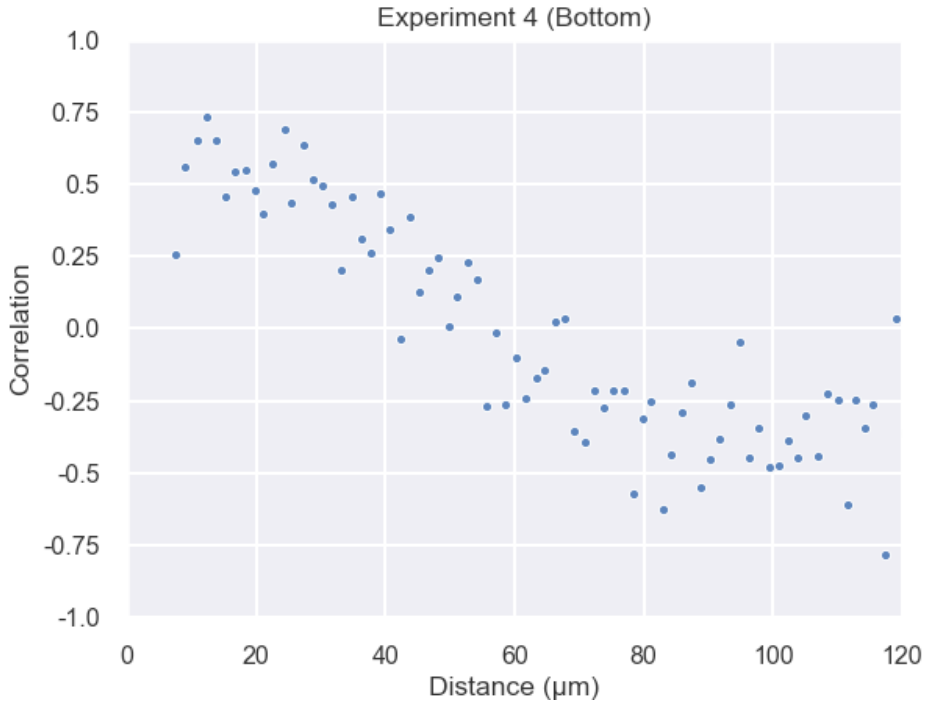

Experiment 3 (Top)
Correlation
1.0
0.75
0.5
0.25
0.0
-0.25
-0.5
-0.75
-1.0
0
20
40
60
80
100
120
Distance (µm)


Experiment 4 (Top)
Correlation
1.0
0.75
0.5
0.25
0.0
-0.25
-0.5
-0.75
-1.0
0
20
40
60
80
100
120
Distance (µm)

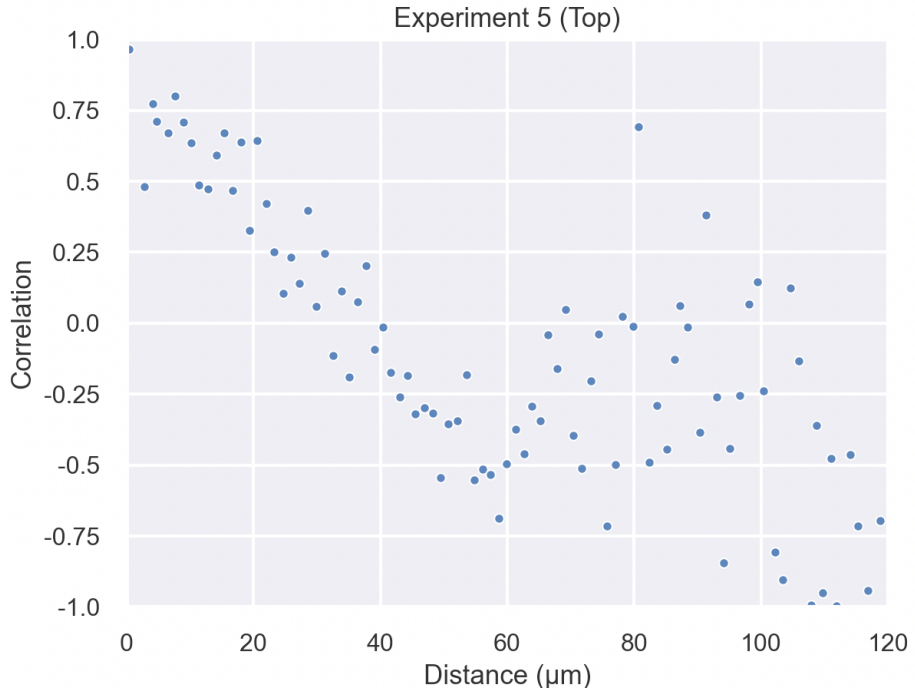

Experiment 1 (Bottom)
Correlation
1.0
0.75
0.5
0.25
0.0
-0.25
-0.5
-0.75
-1.0
0
20
40
60
80
100
120
Distance (μm)

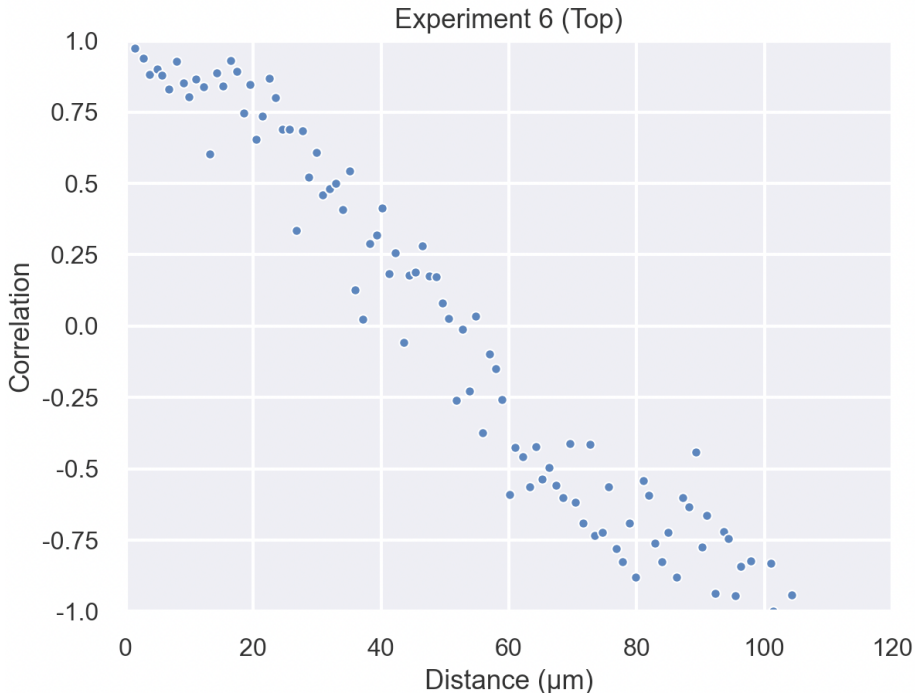

Experiment 2 (Bottom)
Correlation
1.0
0.75
0.5
0.25
0.0
-0.25
-0.5
-0.75
-1.0
0
20
40
60
80
100
120
Distance (μm)

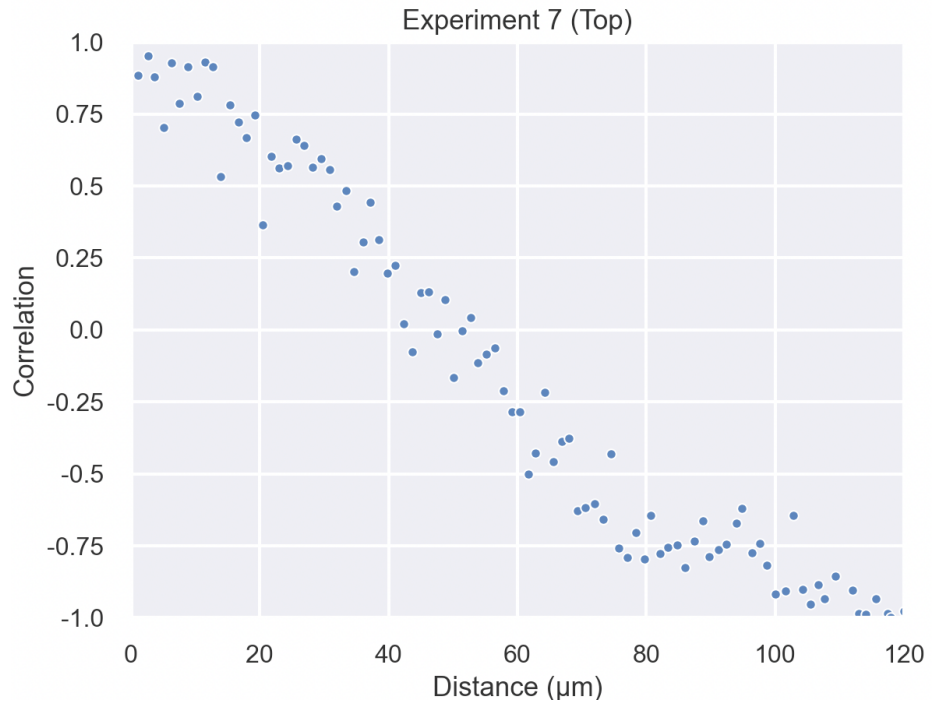

Experiment 3 (Bottom)
Correlation
1.0
0.75
0.5
0.25
0.0
-0.25
-0.5
-0.75
-1.0
0
20
40
60
80
100
120
Distance (μm)

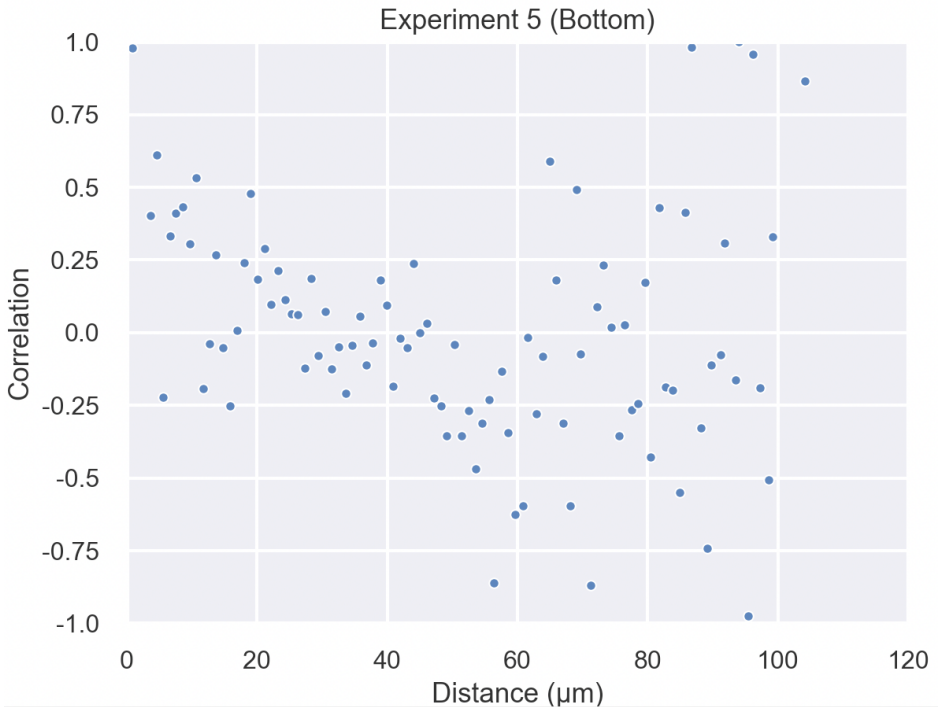

Experiment 4 (Bottom)
Correlation
1.0
0.75
0.5
0.25
0.0
-0.25
-0.5
-0.75
-1.0
0
20
40
60
80
100
120
Distance (μm)

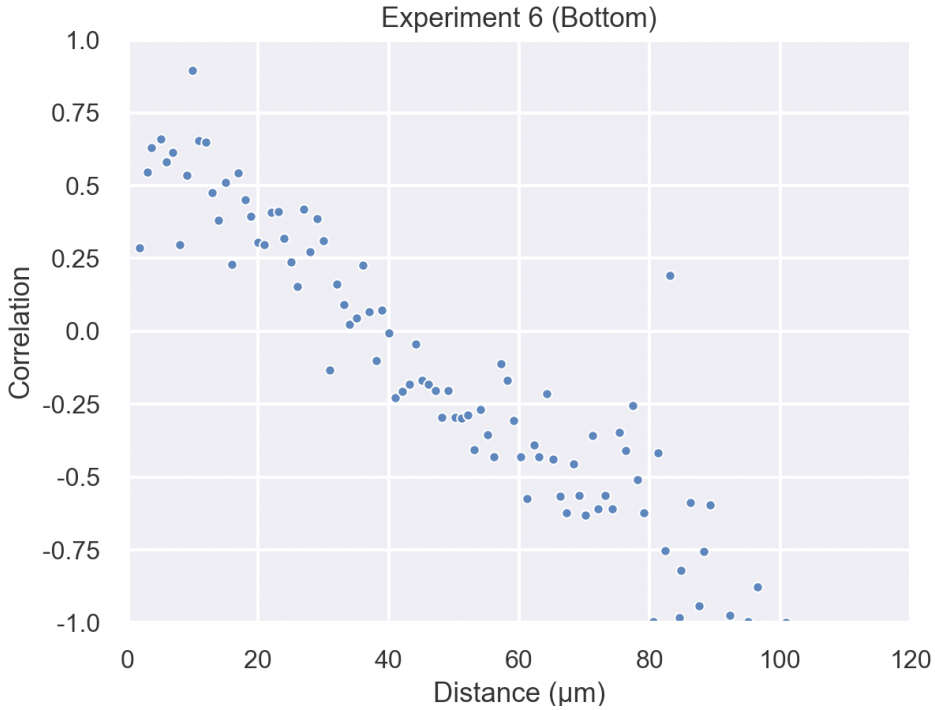
Appendix 7.4. Spatial Correlation Functions of Top and Bottom of Spheroids in Experiments 5, 6 and 7

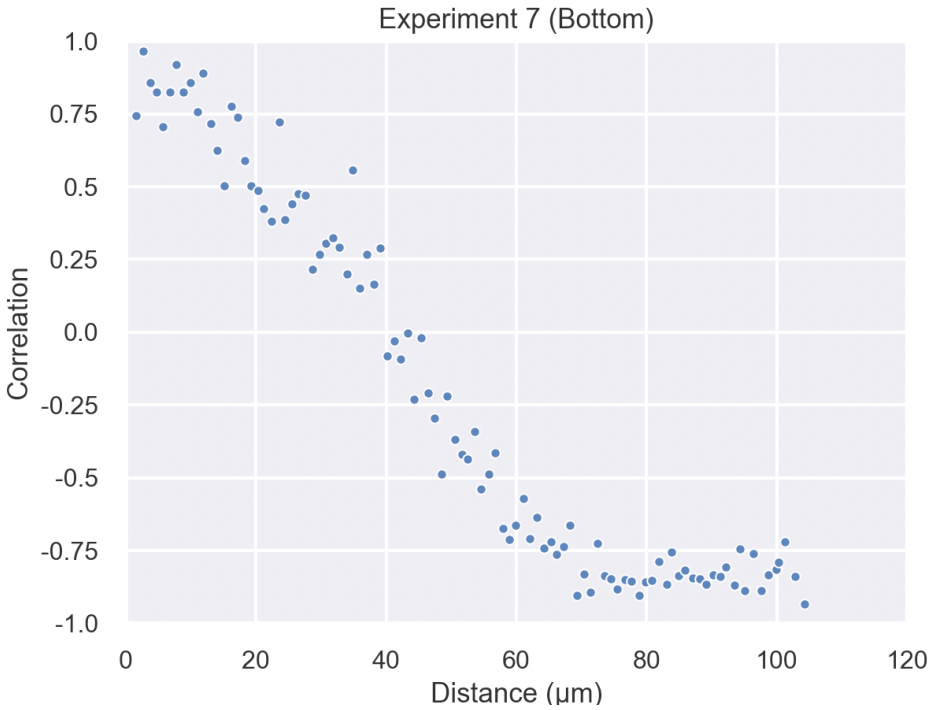





Experiment 7 (Top)
Correlation
1.0
0.75
0.5
0.25
0.0
-0.25
-0.5
-0.75
-1.0
0
20
40
60
80
100
120
Distance (μm)


Experiment 5 (Bottom)
Correlation
1.0
0.75
0.5
0.25
0.0
-0.25
-0.5
-0.75
-1.0
0
20
40
60
80
100
120
Distance (μm)


Experiment 6 (Bottom)
Correlation
1.0
0.75
0.5
0.25
0.0
-0.25
-0.5
-0.75
-1.0
0
20
40
60
80
100
120
Distance (μm)


Experiment 7 (Bottom)
Correlation
1.0
0.75
0.5
0.25
0.0
-0.25
-0.5
-0.75
-1.0
0
20
40
60
80
100
120
Distance (μm)

Appendix 7.5. Source Code

```
'''In the following code, Experiment 1 = 220322, Experiment 2 = 160822
spheroid 1, Experiment 3 = 160822 spheroid 2,
Experiment 4 = 240322, Experiment 5 = 120822, Experiment 6 = 091122 and
Experiment 7 = 171122.'''

import numpy as np
import os
import pandas as pd
import math
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.optimize import curve_fit
from scipy.signal import savgol_filter

directory = './data/invading/'
filenames = os.listdir(directory)
sns.set_theme(context='talk', font_scale=0.7)

def input_csv(filename): # read raw position data files
    df = pd.read_csv(directory + filename)
    df = df[['TrackID', 'Time', 'Position X', 'Position Y', 'Position Z']] #
TrackID is a unique cell identifier

    # rescale as required
    if filename in ['120822.csv', '160822 spheroid 1.csv', '160822 spheroid
2.csv', '091122.csv', '171122.csv']:
        unit = 0.325
    else:
        unit = 1
    df['Position X'] = df['Position X'] * unit
    df['Position Y'] = df['Position Y'] * unit

    # set invasion time for invading spheroids
    if filename == '220322.csv':
        onset = 12
    elif filename == '240322.csv':
        onset = 75
    elif filename == '160822 spheroid 1.csv':
        onset = 23
    elif filename == '160822 spheroid 2.csv':
        onset = 35
    else:
        onset = 0 # set as 0 for non-invading spheroids
    df = df[df['Time'] > onset]
    return df

def drop(file, threshold): # data cleaning
    file.drop(file[file['TrackID'].isnull()].index, inplace=True)
    cells = file['TrackID'].unique()
```

```
    for cell in cells:
        cell_file = file[file['TrackID'] == cell]
        if len(cell_file) < threshold: # drop rows with false signals
            file.drop(file[file['TrackID'] == cell].index, inplace=True)

    x_c, y_c = file['Position X'].mean(), file['Position Y'].mean()
    time = file['Time'].unique()
    for t in time:
        time_file = file[file['Time'] == t]
        time_file['2D dis to C'] = np.sqrt((time_file['Position X'] - x_c) **
2 + (time_file['Position Y'] - y_c) ** 2)
        mean_dis = time_file['2D dis to C'].mean()
        std_dis = time_file['2D dis to C'].std()
        far_cells = time_file[time_file['2D dis to C'] > mean_dis + 3 *
std_dis] # exclude cells that are too far away from centre of spheroid
        ids = far_cells['TrackID'].unique()
        for id in ids:
            file.drop(file[(file['Time'] == t) & (file['TrackID'] ==
id)].index, inplace=True)
    return file

'''Temporal autocorrelation analysis'''
def velocity_temp(file, dt):
    cells = file['TrackID'].unique()
    file_velocity = []
    for cell in cells:
        cell_file = file[file['TrackID'] == cell].sort_values(by='Time')
        cell_file.drop_duplicates(subset="Time", inplace=True)
        # velocity at time t
        cell_file['dt'] = cell_file['Time'].diff()
        cell_file['vx_t'] = cell_file['Position X'].diff() / cell_file['dt']
        cell_file['vy_t'] = cell_file['Position Y'].diff() / cell_file['dt']
        cell_file['vz_t'] = cell_file['Position Z'].diff() / cell_file['dt']
        # velocity at time t + dt
        timepoints = cell_file['Time']
        vx_dt = []
        vy_dt = []
        vz_dt = []
        for t in timepoints:
            time_slice = cell_file[cell_file['Time'] == t + dt]
            if time_slice.empty: # for timepoints with no corresponding dt
position data
                vx_dt.append(None)
                vy_dt.append(None)
                vz_dt.append(None)
                continue
            ind = time_slice.index.to_numpy()[0]
            vx_dt.append(time_slice.at[ind, 'vx_t'])
            vy_dt.append(time_slice.at[ind, 'vy_t'])
            vz_dt.append(time_slice.at[ind, 'vz_t'])
        cell_file['vx_dt'] = vx_dt
```

```
        cell_file['vy_dt'] = vy_dt
        cell_file['vz_dt'] = vz_dt
        file_velocity.append(cell_file.dropna())
    file_v = pd.concat(file_velocity, ignore_index=True)

    time = sorted(file_v['Time'].unique())
    file_norm = []
    for t in time:
        time_file = file_v[file_v['Time'] == t]
        # for velocity at time t
        # calculate centroid velocity
        vcx, vcy, vcz = time_file['vx_t'].mean(), time_file['vy_t'].mean(),
time_file['vz_t'].mean()
        # calculate relative velocity
        time_file['vx_t'] = time_file['vx_t'] - vcx
        time_file['vy_t'] = time_file['vy_t'] - vcy
        time_file['vz_t'] = time_file['vz_t'] - vcz
        # normalise velocity
        mag = np.sqrt(time_file['vx_t'] ** 2 + time_file['vy_t'] ** 2 +
time_file['vz_t'] ** 2)
        time_file['vx_t'] = time_file['vx_t'] / mag
        time_file['vy_t'] = time_file['vy_t'] / mag
        time_file['vz_t'] = time_file['vz_t'] / mag
        time_file['v_t'] = time_file.apply(lambda x: np.array([x['vx_t'],
x['vy_t'], x['vz_t']]), axis=1)
        # for velocity at time t + dt
        # calculate centroid velocity
        vcx, vcy, vcz = time_file['vx_dt'].mean(), time_file['vy_dt'].mean(),
time_file['vz_dt'].mean()
        # calculate relative velocity
        time_file['vx_dt'] = time_file['vx_dt'] - vcx
        time_file['vy_dt'] = time_file['vy_dt'] - vcy
        time_file['vz_dt'] = time_file['vz_dt'] - vcz
        # normalise velocity
        mag = np.sqrt(time_file['vx_dt'] ** 2 + time_file['vy_dt'] ** 2 +
time_file['vz_dt'] ** 2)
        time_file['vx_dt'] = time_file['vx_dt'] / mag
        time_file['vy_dt'] = time_file['vy_dt'] / mag
        time_file['vz_dt'] = time_file['vz_dt'] / mag
        time_file['v_dt'] = time_file.apply(lambda x: np.array([x['vx_dt'],
x['vy_dt'], x['vz_dt']]), axis=1)
        file_norm.append(time_file.dropna())
    file_v = pd.concat(file_norm, ignore_index=True)
    return file_v

def correlation_temp(file):
    file['inner'] = file.apply(lambda x: np.inner(x['v_t'], x['v_dt']),
axis=1) # inner product between adjacent timepoints in each cell
    corr = file['inner'].mean()
    return corr
```

```
def plot_temp(file, time_range, name): # input desired range of dt
    data = pd.DataFrame({'Time': list(time_range)})
    corr = []
    for t in time_range:
        inner = correlation_temp(velocity_temp(file, t)) # plot correlation
against lag time
        corr.append(inner)
    data['Correlation'] = corr

    # curve fitting
    model = lambda x,a,b,c: a * np.exp(-x/b) + c # fit exponential decay curve
    par_opt, par_cov = curve_fit(model, data['Time'], data['Correlation'])
    data['Fitted'] = model(data['Time'], *par_opt)

    # plot observed and fitted curves
    plt.figure()
    plt.ylim(-0.02, 0.07)
    s = sns.scatterplot(data=data, x='Time', y='Correlation', marker='x')
    sns.lineplot(data=data, x='Time', y='Fitted', color='red', label='fit:
a=%5.3f, b=%5.3f, c=%5.3f' % tuple(par_opt)) # do not fit to non-invading
    s.set_xticks(range(0, 71, 10))
    s.set_xticklabels(list(range(0, 71, 10)))
    s.set_yticks([-0.02, -0.01, 0.00, 0.01, 0.02, 0.03, 0.04, 0.05, 0.06,
0.07])
    s.set_yticklabels([-0.02, -0.01, 0.00, 0.01, 0.02, 0.03, 0.04, 0.05, 0.06,
0.07])
    s.set(xlabel='dt (3 min)', ylabel='Correlation', title='Experiment 1')
    plt.legend(loc='upper right', frameon=False)
    plt.show()

def plot_log_temp(files, names): # files is a list of dataframes, names is a
list of filenames
    data = pd.DataFrame({'Time': list(range(2, 71))})
    colours = ['red', 'orange', 'green', 'blue']
    labels = ['Experiment 1', 'Experiment 2', 'Experiment 3', 'Experiment 4']
    plt.figure()
    plt.ylim(-4.0, -1.0)
    for ind, file in enumerate(files):
        filename = names[ind]
        corr = []
        mid = file['Position Z'].median()
        # file = file[file['Position Z'] <= mid] # select top
        # file = file[file['Position Z'] > mid] # select bottom
        for t in range(2, 71):
            inner = correlation_temp(velocity_temp(file, t)) # correlation
values for all files
            corr.append(inner)
        # smoothing
        data[filename + '_smoothed'] = savgol_filter(corr, 31, 1)
        # smoothed curve in log scale
        pos_data = data[data[filename + '_smoothed'] > 0]
```

```
        pos_data[filename + '_log'] = np.log10(pos_data[filename +
'_smoothed'])
        l = sns.lineplot(data=pos_data, x='Time', y=filename + '_log',
label=labels[ind], color=colours[ind])
    l.set_xticks(range(0, 71, 10))
    l.set_xticklabels(list(range(0, 71, 10)))
    l.set_yticks([-4.0, -3.5, -3.0, -2.5, -2.0, -1.5, -1.0])
    l.set_yticklabels([-4.0, -3.5, -3.0, -2.5, -2.0, -1.5, -1.0])
    l.set(xlabel='dt (3 min)', ylabel='Log of correlation', title='Overall')
    plt.legend(loc='upper right', frameon=False) # overall and top plots
    # plt.legend(loc='lower left', frameon=False) # bottom plot
    plt.show()

files, names = [], []
for file in filenames:
    if file.endswith('.csv'):
        files.append(drop(input_csv(file), 3))
        names.append(file[:-4])
plot_log_temp(files, names)

def get_char_timescale(file, time_range, name):
    mid = file['Position Z'].median()
    top = file[file['Position Z'] <= mid]
    bottom = file[file['Position Z'] > mid]
    data, data_top, data_bottom = pd.DataFrame({'Time': list(time_range)}),
pd.DataFrame({'Time': list(time_range)}), pd.DataFrame({'Time':
list(time_range)})
    corr, corr_top, corr_bottom = [], [], []
    for t in time_range:
        inner = correlation_temp(velocity_temp(file, t))
        corr.append(inner)
        inner_top = correlation_temp(velocity_temp(top, t))
        corr_top.append(inner_top)
        inner_bottom = correlation_temp(velocity_temp(bottom, t))
        corr_bottom.append(inner_bottom)
    data['Correlation'] = corr
    data_top['Correlation'] = corr_top
    data_bottom['Correlation'] = corr_bottom

    # smoothing
    data['Smoothed'] = savgol_filter(data['Correlation'], 31, 1)
    data_top['Smoothed'] = savgol_filter(data_top['Correlation'], 31, 1)
    data_bottom['Smoothed'] = savgol_filter(data_bottom['Correlation'], 31, 1)

    # smoothed curve in log scale
    pos_data = data[data['Smoothed'] > 0]
    pos_data['Logarithm'] = np.log10(pos_data['Smoothed'])
    pos_data_top = data_top[data_top['Smoothed'] > 0]
    pos_data_top['Logarithm'] = np.log10(pos_data_top['Smoothed'])
    pos_data_bottom = data_bottom[data_bottom['Smoothed'] > 0]
    pos_data_bottom['Logarithm'] = np.log10(pos_data_bottom['Smoothed'])
```

```
    # coefficients of linear fit
    # overall
    sub_data = pos_data[pos_data['Time'] <= 20] # fit linear curve up to dt =
20
    x = sub_data['Time']
    y = sub_data['Logarithm']
    gradient, intercept = np.polyfit(x, y, 1) # linear fit
    print(f"Coefficients for top of sample {name}:", gradient, intercept)
    # top
    sub_data_top = pos_data_top[pos_data_top['Time'] <= 20]
    x_top = sub_data_top['Time']
    y_top = sub_data_top['Logarithm']
    gradient, intercept = np.polyfit(x_top, y_top, 1)
    print(f"Coefficients for top of sample {name}:", gradient, intercept)
    # bottom
    sub_data_bottom = pos_data_bottom[pos_data_bottom['Time']<=20]
    x_bottom = sub_data_bottom['Time']
    y_bottom = sub_data_bottom['Logarithm']
    gradient, intercept = np.polyfit(x_bottom, y_bottom, 1)
    print(f"Coefficients for bottom of sample {name}:", gradient, intercept)

'''Spatial correlation analysis'''
def velocity_spat(file):
    # get relative, normalised 2D velocity
    cells = file['TrackID'].unique()
    file_velocity = []
    for cell in cells:
        cell_file = file[file['TrackID'] == cell].sort_values(by='Time')
        cell_file.drop_duplicates(subset="Time", inplace=True)
        cell_file['dt'] = cell_file['Time'].diff()
        cell_file['vx'] = cell_file['Position X'].diff() / cell_file['dt']
        cell_file['vy'] = cell_file['Position Y'].diff() / cell_file['dt']
        file_velocity.append(cell_file.dropna())
    file_v = pd.concat(file_velocity, ignore_index=True)

    time = sorted(file_v['Time'].unique())
    file_norm = []
    for t in time:
        time_file = file_v[file_v['Time'] == t]
        # calculate centroid velocity
        vcx, vcy = time_file['vx'].mean(), time_file['vy'].mean()
        # calculate relative velocity
        time_file['vx'] = time_file['vx'] - vcx
        time_file['vy'] = time_file['vy'] - vcy
        # normalise velocity
        mag = np.sqrt(time_file['vx'] ** 2 + time_file['vy'] ** 2)
        time_file['vx'] = time_file['vx'] / mag
        time_file['vy'] = time_file['vy'] / mag
        # calculate radial (unit) vector
```

```
        center_x, center_y = time_file['Position X'].mean(),
time_file['Position Y'].mean()
        time_file['vrx'] = time_file['Position X'] - center_x
        time_file['vry'] = time_file['Position Y'] - center_y
        mag = np.sqrt(time_file['vrx'] ** 2 + time_file['vry'] ** 2)
        time_file['vrx'] = time_file['vrx'] / mag
        time_file['vry'] = time_file['vry'] / mag
        # velocity - radial velocity
        time_file['vx'] = time_file['vx'] - time_file['vrx']
        time_file['vy'] = time_file['vy'] - time_file['vry']
        # calculate rotation velocity
        vrotx, vroty = time_file['vx'].mean(), time_file['vy'].mean()
        time_file['vx'] = time_file['vx'] - vrotx
        time_file['vy'] = time_file['vy'] - vroty
        # normalise velocity
        mag = np.sqrt(time_file['vx'] ** 2 + time_file['vy'] ** 2)
        time_file['vx'] = time_file['vx'] / mag
        time_file['vy'] = time_file['vy'] / mag
        time_file['Velocity'] = time_file.apply(lambda x: np.array([x['vx'],
x['vy']]), axis=1)
        file_norm.append(time_file.dropna())
    file_v = pd.concat(file_norm, ignore_index=True)
    return file_v

def correlation_spat(file, t): # input desired timepoint
    file_t = file[file['Time'] == t]
    pair_file = pd.DataFrame()

    # correlation between all neighbour pairs
    dr = []
    inner = []
    for ind_i, row_i in file_t.iterrows():
        for ind_j, row_j in file_t.iterrows():
            if ind_i < ind_j:
                dr.append(math.sqrt((row_i['Position X'] - row_j['Position
X']) ** 2 + (row_i['Position Y'] - row_j['Position Y']) ** 2))
                inner.append(np.inner(row_i['Velocity'], row_j['Velocity']))
    pair_file['dr'] = dr
    pair_file['inner'] = inner

    # group inner product values into bins based on distance between
neighbours
    pair_file['bin'] = pd.cut(pair_file['dr'],
bins=np.linspace(pair_file['dr'].min(), pair_file['dr'].max(), 100),
include_lowest=True)

    # take average correlation within each bin
    bins = sorted(pair_file['bin'].unique())
    corr_file = pd.DataFrame()
    dist = []
    corr = []
```

```
    for bin in bins:
        bin_file = pair_file[pair_file['bin']==bin]
        dist.append(bin_file['dr'].mean())
        corr.append(bin_file['inner'].mean())
    corr_file['Distance'] = dist
    corr_file['Correlation'] = corr
    return corr_file

def plot_spat(file, name):
    mid = file['Position Z'].median()
    # file = file[file['Position Z'] <= mid] # select top
    # file = file[file['Position Z'] > mid] # select bottom
    data = correlation_spat(velocity_spat(file), 40) # plot correlation
against dr
    plt.figure()
    plt.ylim(-1.00, 1.00)
    s = sns.scatterplot(data=data, x='Distance', y='Correlation', marker='.')
    s.set_xticks(range(0, 121, 20))
    s.set_xticklabels(list(range(0, 121, 20)))
    s.set_yticks([-1.00, -0.75, -0.50, -0.25, 0.00, 0.25, 0.50, 0.75, 1.00])
    s.set_yticklabels([-1.00, -0.75, -0.50, -0.25, 0.00, 0.25, 0.50, 0.75,
1.00])
    s.set(xlabel='Distance (\u03bcm)', ylabel='Correlation', title='Experiment
6')
    plt.show()
```