UNIVERSIDADE DE LISBOA
FACULDADE DE CIÊNCIAS
DEPARTAMENTO DE FÍSICA

Modeling cell adhesions and proliferation in complex environments

Tatiana Sofia Carvalho Rebocho

Mestrado em Engenharia Biomédica e Biofísica

Dissertação orientada por:
Dr. Cristóvão de Sousa Dias
Prof. Dr. Nuno Miguel Azevedo Machado de Araújo

2023

Abstract

Keywords: in silico modeling, mechanotransduction, metastasis

Tissue engineering is always looking for new approaches to control and modulate tissues. To find these approaches, there is a need to understand tissue activity and its interactions with different media. These processes can be studied through experimental studies, as it has been for years. However, in silico approaches are gaining relevance. To successfully use computational tools, it is necessary to have good knowledge of the biophysical mechanisms involved, such as mechanotransduction or metastasis, but also of the developed models and strategies.

In this work, we develop two in silico models, for two different problems with an experimental motivation. The first model was adapted from a previous numerical approach, but with an open-source tool. The model means to illustrate a deformable cell falling on a relaxed granular bed, where the particles of the bed have variable sizes. The cell needs to adapt to the granular bed over cycles of stretching and contraction. We can observe a transition in the cell fate between beds with large particles and beds with small particles. The differences to the previous model are discussed.

The second model describes a tumor spheroid, which includes the formation of protrusions, studied to understand the biophysical parameters that can be relevant for protrusion dynamics. We start with an analytical analysis, where we can establish a parallel between the force of protrusion and the elastic constant of bonds between cells with the quantity of paxillin and E-Cadherin, respectively. The spheroid's size effect on the cell's behavior is evaluated through simulations. Cell dynamics present a non-monotonic behavior as a function of the size of the spheroids.

Resumo

Palavras chave: modelos *in silico*, mecano-transdução, metástase

A engenharia de tecidos está sempre a evoluir com o objetivo de encontrar novas formas de controlar e modular tecidos. Para se conseguir atingir este fim, é importante perceber a atividade coletiva das células e as suas interações com diferentes ambientes e entre si. Estes processos podem ser investigados através de estudos experimentais, como tem sido feito ao longo dos anos. No entanto, as abordagens computacionais estão a ganhar cada vez mais relevância para ajudar a compreensão de diversos fenómenos biológicos e biofísicos. Para que se usem ferramentas computacionais de forma adequada neste tipo de aplicações, é necessário estudar os problemas biofísicos do ponto de vista teórico, mas também os modelos e estratégias já estudados.

Nesta dissertação foram desenvolvidos dois modelos computacionais, para problemas de base experimental. O primeiro modelo é construído com o objetivo de tentar compreender a influência no comportamento celular do tamanho de microparticulas esféricas organizadas numa cama granular. O segundo modelo, por sua vez, desenvolve-se para estudar a correlação de parâmetros biofísicos relacionados com prolongamentos celulares em metástases, como o seu comprimento e o seu tempo de estabilidade, com o potential invasivo de células.

Com o primeiro modelo pretende-se estudar uma célula deformável a cair numa cama granular, e perceber de que forma o tamanho das partículas da cama granular vai influenciar o comportamento da célula, e consequentemente, a sua taxa de sobrevivência. Este modelo é desenvolvido com o objetivo de adaptar um modelo numérico já existente através do uso de uma ferramenta open-source, o Large-scale Atomic/Molecular Massively Parallel Simulator (LAMMPS). A célula vai precisar de se adaptar à forma da cama ao longo de ciclos de alongamento e contração, para que se possa simular o seu crescimento e a sua adesão às partículas da cama granular. Para se ter sucesso na construção do modelo, é necessário conhecer o processo de adesão celular e a forma como as células transduzem estímulos mecânicos em sinais químicos ou em expressão genética, através de um processo denominado mecano-transdução. A célula foi modelada de forma a sofrer ciclos de alongamento celular seguidos de uma contração, com o objetivo de modelar a fixação da mesma nas partículas pertencentes à cama granular. Para simular a adesão celular com um momento de fixação inicial e um segundo momento de reforço da adesão, foram adicionados "patches" nos pontos de contato entre elementos subcelulares e partículas granulares, para que a adesão seja representada à superfície da célula. Estes "patches" simulam o comportamento das integrinas na adesão celular. Podemos observar uma transição entre dois regimes, o de partículas muito maiores que a célula e o de partículas muito menores que a célula. Apenas os resultados para células significativamente menores que as partículas da cama granular correspondem ao esperado, mostrando que neste caso a célula irá conseguir fixar-se, reforçar a sua adesão e alongar-se de forma a cobrir uma maior área da partícula a que adere. Seria esperado que células significativamente maiores que as partículas da cama granular não conseguissem manter a sua

adesão, e que por isso iriam acabar por não alongar e, consequentemente, teriam uma taxa de sobrevivência menor.

O segundo modelo representa esferóides de um tumor, onde as células que o compõe irão sofrer uma força que provoca prolongamentos, promovendo assim a metástase. O objetivo da investigação experimental é compreender a correlação de parâmetros biofísicos destes prolongamentos celulares, tais como o seu comprimento, o tempo em que estes se conseguem manter estáveis e a frequência com que ocorrem. Para estudar este problema, nesta dissertação, usa-se uma abordagem analítica e simulações numéricas. Na abordagem analítica, o foco é a equação da Dinâmica de Langevin e parâmetros que podem ser estudados com base na mesma. Observa-se que o deslocamento máximo de uma célula ativa pela força de prolongação tem um comportamento não-linear e depende tanto da força de formação dos prolongamentos, quanto da constante elástica das conexões entre as células. Quanto maior a constante elástica, maior será a resistência ao movimento da célula e, por isso, menor será o deslocamento atingido pela mesma. Por outro lado, uma força de formação do prolongamento maior, irá aumentar o quanto a célula ativa consegue afastar-se da sua posição inicial e, consequentemente, o seu deslocamento máximo irá ser superior. Para além disso, é observado que o tempo necessário para estabilizar o movimento da célula vai depender tanto da constante elástica das ligações entre células quanto da viscosidade do meio. Um meio com a viscosidade mais baixa irá implicar menos tempo até se atingir o deslocamento máximo, uma vez que a célula sente menos resistência ao seu movimento. No entanto, uma constante elástica maior também vai levar a que a célula ativa atinja o seu deslocamento máximo de forma mais rápida, uma vez que as células vão estar ligadas de forma mais estável e o esferóide vai sofrer menos restruturações. Com o estudo analítico é possível representar o deslocamento máximo de uma célula ativa e o tempo necessário para atingir o deslocamento máximo em unidades dimensionais, dependentes da força de formação do prolongamento, da constante elástica das ligações intercelulares e da viscosidade do meio. Também são estudados os regimes de amortecimento do modelo em função da força efetiva sentida pela célula ativa, e consequentemente em função da constante elástica efetiva das ligações entre células e do deslocamento sofrido pela célula ativa. Para além disso, consegue-se relacionar a força que forma os prolongamentos e a constante elástica das ligações intercelulares com a quantidade de paxilina, uma proteína presente nas interações entre a matriz extracelular e as células, e a quantidade de E-caderina, uma proteína presente nas interações entre células, respetivamente. Estas quantidades foram base de estudo no trabalho experimental usado como inspiração para o desenvolvimento do modelo apresentado nesta dissertação.

Para avaliar o efeito do tamanho do esferoide no comportamento da célula ativa, realizam-se simulações numéricas. Para isso, avaliam-se configurações de esferoides regulares, mas também de esferoides amorfos. Para ambas as configurações, encontra-se uma dependência não-monótona do deslocamento máximo da célula ativa com o tamanho do esferóide. No entanto, a configuração amorfa parece ter uma dependência contínua, ao contrário do que acontece com esferoides de configurações regulares. A variação verificada no deslocamento máximo de células em esferoides regulares deve-se à geometria dos mesmos, uma vez que estes têm uma forma similar a poliedros. O deslocamento máximo da célula ativa é aproximadamente constante após $N \approx 55$, sendo N o número de células num esferoide, o que pode indicar que esferoides com menos de 55 células não devem ser considerados como tecidos compostos por células que apresentam um comportamento coletivo. O número de vizinhos da célula ativa também influencia o deslocamento máximo alcançado pela mesma. O tempo necessário para atingir o deslocamento máximo parece

aumentar, mesmo após o deslocamento deixar de sofrer alterações significativas. Isto pode ser explicado pelo comportamento coletivo do agregado celular, uma vez que a força efetiva sentida pela célula ativa irá aumentar com o tamanho do esferóide devido à cadeia de ligações intercelular existente. Além disso, os parâmetros biofísicos estudados com o auxílio de simulação numérica são estudados de forma a serem comparados às unidades adimensionais retiradas na fase analítica do estudo deste modelo, com o objetivo de verificar se o comportamento da célula ativa seria semelhante ao observado do ponto de vista analítico.

Agradecimentos

Quero começar por agradecer ao Professor Nuno Araújo e ao Professor Cristóvão Dias pela oportunidade de trabalhar numa área que tanto me fascinou quanto me desafiou e por me terem guiado no percurso que foi a minha Tese de Mestrado.

Deixo também o meu agradecimento à restante equipa do CFTC, também pelas discussões sobre Física, mas principalmente pelo companheirismo, pelos conselhos, e pelas longas conversas sobre tudo e nada nas horas de pausa. Foi incrível trabalhar convosco. Tiraram muito do peso do que poderia ser esta fase do Mestrado.

Aos meus amigos, agradecer é pouco! Vocês foram uma peça essencial para a conclusão desta fase. Por todas as vezes que acreditaram em mim, quando eu duvidava. Por todas as vezes que atenderam as minhas chamadas para eu falar sobre Física, mesmo sendo chinês para vocês. Por todas as vezes que olharam para vídeos de bolinhas a mexer e me achavam doida por estar entusiasmada. Às que eu conheci na residência e levo para a vida, obrigada. Ao grupo de LCS, obrigada. À Márcia e à Beatriz por serem as melhores madrugadeiras que eu podia ter conhecido no curso, obrigada. Ao Ivan, pela paciência, por me ajudar a encontrar as palavras que eu não tenho e pelo olho crítico a avaliar os meus rabiscos no Paint, obrigada.

Não podia deixar de agradecer à minha família pela capacidade inesgotável de aturarem todas as minhas crises académicas. Em especial, aos meus pais, por me deixarem voar e por terem sempre a certeza que sou capaz de mais e melhor. Ao meu irmão, por gritar aos sete ventos aquilo que eu andei a fazer, mesmo não sabendo o nome do meu curso. À minha irmã, pelos olhos a brilharem de cada vez que me vê entusiasmada. Aos meus avós, maternos e paternos, por serem mais que avós, pelo "netinha, com calma, saúde e sem estragar a cabeça com tanto estudo", e por serem *casa* sempre que me senti perdida. Sem todos vocês, eu não tinha chegado aqui.

Por último, quero agradecer à Fundação para a Ciência e a Tecnologia e à FCiências.ID pelo apoio financeiro através do projeto UIDB/00618/2020.

Contents

List of Figures

Chapter 1

Introduction

Tissue engineering is in constant progress to find new ways to control and modulate tissues. When this goal is achieved, it will be possible to replace damaged organs and even conduct experiments that will help to understand tissue activity and develop new therapies [2, 3]. To better understand tissues' formation, it is relevant to study cell behavior and how they aggregate. This knowledge may be acquired through experimental studies, but also with the help of numerical simulations and theoretical analysis of the cell motion and interactions. In order to perform this type of investigation, it is important to consider different phenomena such as the formation and spread of aggregates, the mechanism of cell-cell adhesion, changes in shape, and even cell proliferation. The use of in silico models is increasing in order to decrease the materials costs of experimental investigations and provide quantitative knowledge about cell behavior [3, 1, 4, 5, 6, 7, 8, 9]. However, there are limitations in the use of computational techniques. To achieve more details, it is necessary to limit the size of the cell population and it is impossible to consider all interactions due to computational costs. To overcome the imposed limitations, several models were developed taking into account the level of detail needed to achieve the study's goal. Each model can be used to infer conclusions about specific questions about the system. The models can go from describing each cell on a lattice to describing it as a spherical particle, or even describing a group of cells by fields (more details in Chapter 2).

In this work, there are two different approaches to cellular problems. In the first one, we approach a model developed by Cunha et al. [4], to study the way a cell adheres to a granular bed accordingly to the aspect ratio between cells and the bed's particles. The goal is to have the same model with an open-source tool approach that replicates their results (Chapter 3). In the other case, the protrusion phenomena leading to metastasis from spheroid tumors are studied from an analytical point of view, but also by means of simulations (Chapter 4). To understand the topics that will be approached during this work, below there is a description of the biological phenomena in cell mechanotransduction and in cancer metastasis.

1.1 Cell responses via mechanotransduction

The properties of biomaterials can influence and modulate relevant biological performances. By defining specific physical properties of particles, such as size, geometry, anisotropy, topography, stiffness, and porosity, it is possible to achieve specific biological responses. For example, by modeling the size of granular particles, it is possible to understand the behavior of cells during growth [4, 10].

The spreading of cell aggregates on a substrate is analogous to the spreading of a viscous droplet: after contact with an adherent substrate, the aggregate flattens and surrounds itself with a precursor film consisting of a cellular monolayer [3]. This is possible only by proper signaling after the cells have recognized their environment [11]. The conversion of mechanical signals into chemical signals or gene expression is called mechanotransduction. The mechanisms underlying cellular mechanotransduction are still not well-understood [12].

The contractile and adhesive molecular machinery of cells consists of dynamic molecular bonds between the extracellular matrix, integrins, adaptor proteins, and the force-generating actomyosin cytoskeleton. The perception of and response to rigidity occurs in several steps. First, the cell transfers force to the surrounding matrix and then converts the forces into biochemical signals. Talin sets a threshold for rigidity that triggers force transmission [13].

When talin unfolds, vinculin binds, leading to an increase in adhesion and growth and the recruitment of integrins. When talin does not unfold, integrins are not recruited, but the force is still thought to be transferred between integrins and actin via other adapter proteins. Above a stiffness threshold, force loading becomes fast enough to allow unfolding of talin before integrin binding is abolished [4, 13].

Cell-extracellular matrix adhesion and mechanotransduction are mediated by a molecular clutch mechanism that characterizes the connection between the cell cytoskeleton and the extracellular matrix and its relationship to cell movement[13, 14]. As it regulates force transmission and cell movement, it controls the mechanical balance in a tissue, its remodeling and mechanotransduction events[14]. The mechanical stimuli received by cells can help regulate phenomena such as migration, contractility, gene expression, and cell fate [15]. In this way, cell adhesion to the extracellular matrix is central to the development and organization, maintenance, and repair of tissues by providing anchorage and triggering signals that enable these processes [15].

Mechanotransduction occurs through several types of cellular responses. There are primary mechanoreceptors, such as ion channels and integrins, but also mechano-signaling complexes, such as caveolae or focal adhesions, that respond rapidly to forces. The signal integrator response is necessary to transmit extracellular forces into the nucleus. Nuclear shuttling proteins enter the nucleus and alter gene expression, leading to adaptations in cell organization and function [16].

1.2 Cancer Metastasis Cascade

The origin of metastasis is not clear [17]. There are several hypotheses to explain it, such as the epithelial-mesenchymal transition, an accumulation of mutations in stem cells, a macrophage facilitation process, and a macrophage origin involving either transformation or fusion hybridization with neoplastic cells [18]. The most accepted theory in literature is the epithelial-mesenchymal transition [8, 19, 20, 21, 22]. Epithelial-mesenchymal transition is a process by which epithelial cells are transformed into cells with the ability to invade, resist stress, and migrate, this means that they gain a mesenchymal phenotype [23].

Hanahan et al. [24, 25, 26] defined essential alterations in cell physiology that will define a malignant form. It includes self-sufficiency in growth signals in order to create an active proliferative state, insensitivity to growth-inhibitory signals so the proliferation will not be blocked, and evasion of programmed cell death, which will lead to limitless replicative potential. Besides that, there will be sustained angiogenesis to supply oxygen and nutrients for tumor cells

survival, tissue invasion and metastasis so cells can escape the primary tumor mass and colonize other tissues, reprogramming cellular metabolism by adjusting the energy metabolism in order to fuel cell growth and division and avoiding immune destruction [24, 25, 26].

Our focus will be on cancer metastasis, and the hallmarks that promote it. Metastasis is the ultimate reason for cancer mortality, being responsible for 90% of cancer deaths [27, 17, 23], and is associated with a rapid decrease in treatment prospects [28]. This mechanism has a stochastic nature and is a multi-phase process in which tumor cells detach from the primary tumor and travel through the body in the circulatory and lymphatic systems to form a secondary tumor in other tissues of the body [28]. It is a complex process coordinated at the subcellular level by biochemical signals and at the cellular and tissue level by the biophysical properties of cells and force interactions between cells, the extracellular matrix, and the interstitial fluid [27, 17, 28, 29].

Tumor growth and tumor-induced angiogenesis are necessary for initiating the metastasis cascade. On one hand, heterogeneous tumor growth is essential, so the mass is more probable to survive [27]. On the other hand, the non-uniform distribution of oxygen and nutrients will increase the need to activate the angiogenesis, in order to sustain the abnormal tumor growth [25]. The metastasis cascade has four steps: the intravasation of an individual tumor cell into the circulatory system, the circulation of the cell, arrest, and extravasation, and eventually colonization and formation of a metastatic tumor [27].

The metastasis process initiates with the transformation of tumor cells from non-invasive cells to cells with invasion potential [30, 27]. When cells acquire invasive phenotypes and have a greater migration ability, they will detach from the primary tumor. One of the modifications these cells will suffer is the formation of protrusions [30]. These elongations are essential for the migration and invasion of tumor cells and the biophysical parameters that influence it are the focus of the work done in Chapter 4. They will enable the cells to degrade the extracellular matrix by using specialized enzymes - proteases - and to move by providing mechanical forces. The invasion of a tumor can happen with single cells or with clusters. Cells that invade in aggregates can stay connected to the main cluster, or detach. Besides that, they have a higher probability of survival, not only because of the cluster's size, which will enable the inner cells to be protected but also because they have immune components and are not homogeneous [23]. For this to happen, there is a need to generate force, by mechanotransduction, which was mentioned before [30].

There is a moment when tumor cells invade nearby tissues and intravasate into the circulatory or lymphatic vessels [27, 17]. This increases with tumor-induced angiogenesis, mainly because the new vessels have weak and leaky basement membranes, which facilitate the penetration of tumor cells [27]. Most of these cells do not survive, due to the immune response, shear stresses and cell deformations [27, 23]. The small number of circulating tumor cells that can attach to the vessel walls and extravasate out of the circulatory system is enough for initiating a secondary tumor [27].

Cell adhesion to the vessel's wall depends on mechanical, chemical, and biological factors. It will determine whether and where the tumor cell will initiate a secondary tumor since it only will happen if cells find the necessary stability [27, 23]. There are two main hypotheses explaining the patterns of progression of metastasis and the location of the secondary tumor based on the most possible locations of tumor cell arrest and interactions with the microenvironment of the metastatic site. Paget [31] proposed the "seed and soil" hypothesis which refers that cancer cells (seed) can go through every organ, but only will stay if the recipient tissue (soil) have the

necessary characteristics to support the secondary tumor growth. On the other hand, Ewing [32] suggested that mechanical factors related to blood-flow patterns are responsible for metastasis and the number of metastatic tumors in a specific organ depends on the number of cancer cells reaching the organ and staying in it.

After anchoring to the vessel wall, circulating tumor cells start to extravasate the vessel through a drastic shape-changing process called transmigration, which is possible due to cancer cell plasticity [23, 27].

Chapter 2

Computational Models of Cells

Computational modeling is now a relevant part of research on cell and tissue behavior. One of the main reasons for the increasing importance of this approach is that models allow the description of several cell mechanisms from a quantitative point of view, such as cell migration, and cell interactions between them and with the surrounding medium [28]. Besides that, the data achieved with this type of approach can be validated experimentally [28]. With this in mind, there is a growing need for computational models that can integrate biochemical and biophysical factors, since these models enable the conception and simulation of medical processes and even the design of new experiments [27, 28].

There are several challenges in choosing the best model to solve the problem at hand. The process starts by finding the right approach to the model construction. The time and space scales needed are explored, and all the model assumptions are established [33, 34]. Another relevant step is to define the appropriate parameter values and the initial conditions. The next step should be to understand the tools that will allow an efficient, stable, and accurate way to solve the model [33]. Finally, it is necessary to validate the model, for example, with experiments, to understand the predictive capacity of the model [33, 35].

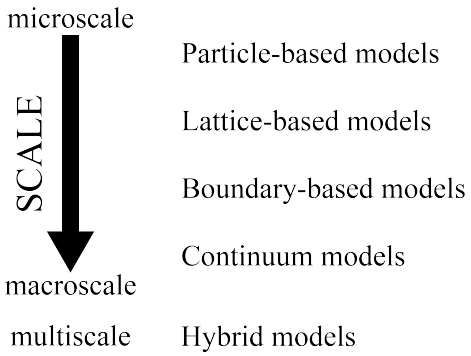

Figure 2.1: Schematic representation of the explored model's categories (particle-based models, lattice-based models, boundary-based models, continuum models, and hybrid models) that are used for several length scales.

Next, there is a description of several computational models that have been used during the past years to model cell and aggregate behavior (continuum, discrete, and hybrid continuum-discrete models), and in Figure 2.1 there is a representation of which models are useful in different length scales.

2.1 Continuum Models

Continuum models are cell population-based, which means that the properties of cells are averaged over a population and gradually change in space, and are described by their densities or volumes fractions [28, 36, 9]. This type of model is used to evaluate the dynamics of nutrients, chemical factors, and the extracellular matrix, and is capable of capturing larger-scale volumetric dynamics [28, 37, 38].

In continuum models, systems of partial differential equations and ordinary equations are used to describe motion and modifications that exist in cell populations and/or chemical processes [27]. The model variables are described as continuous fields [37].

An example of a mathematical representation used in a continuum model for the change in the extracellular matrix is:

$$\frac{\partial e}{\partial t} = -\alpha\beta e \tag{2.1}$$

where e is the density of the extracellular matrix, α is its degradation rate by matrix-degrading enzyme and β is the density of this enzyme in the medium [27].

These models are computationally less expensive and are useful for investigating general behaviors of biological systems at the macroscale. Continuum models are more suitable to describe tissues' evolution since they will not include the heterogeneity of cells and subcellular elements, being used averaging several characteristics over scales larger than the cells' diameter [37, 27, 36, 9].

2.2 Discrete Models

Discrete models, also known as agent-based models, individual-based models, or cell-based models, are based on the units [28, 27, 29]. They are useful to understand how single-cell behaviors have an influence on a system's dynamics and can include a few cells or a larger group of cells [29, 38]. It is possible because in discrete models there is an inclusion of individual cell characteristics that allow one to define tissue heterogeneity [28, 29]. Besides that, there is the implementation of interaction rules between cells and between cells and their surroundings [29, 27, 37, 38].

Agent-based models are used to describe cell systems at the micro- and mesoscale, being relevant to understand heterogeneity phenomena, like the ones created in a system composed of different types of cells, and to describe cell growth, migration, and interactions between cells and their surroundings [27, 37, 28]. However, these models are computationally expensive, and this cost increases with the number of modeled cells [36]. Because of that, it is extremely important to understand the model assumptions, and have caution when deciding which are the needed details [37].

Discrete models can be divided into two different categories: lattice-based models and off-lattice models.

2.2.1 Lattice-based Models

Lattice-based models allow one to track cells in a rigid grid, that can be regular (Figure 2.2) or non-regular [29]. While regular grids are simpler to implement and visualize, non-regular

grids are useful to avoid grid biases [29].

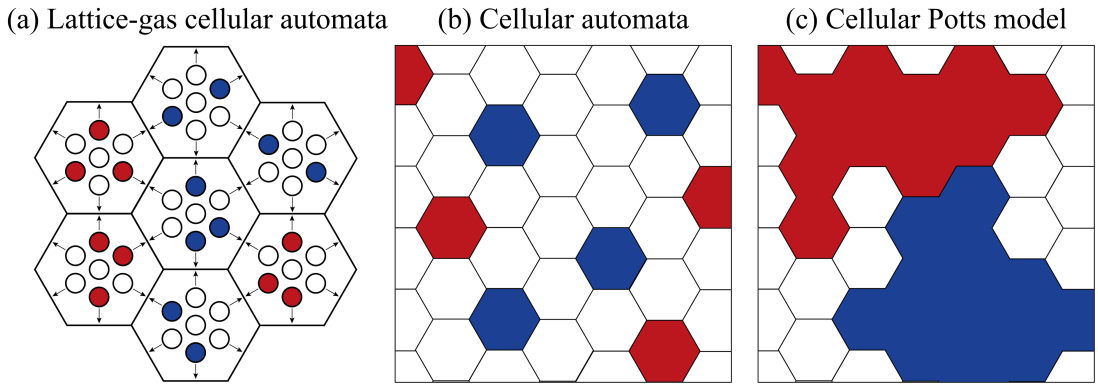

Figure 2.2: Schematic representation of lattice-based models. For every scheme, different colors represent different cells type. (a) Geometric representation of lattice-gas cellular automata models. (b) Geometric representation of cellular automata models. (c) Geometric representation of cellular Potts model.

These models can include several rules for cell behavior, which allow one to mimic complex tissue patterns, so they are usually used to model deformations or rigid-body motions [27]. With lattice-based models, it is possible to understand this characteristic using less memory with a reduced computational cost [39].

2.2.1.1 Lattice-gas cellular automata models

In lattice-gas cellular automata models each lattice site represents multiple cells (Figure 2.2 (a)) [29, 40]. These models track the number of cells that move through channels between individual lattice sites, allowing the simulation of many cells over time [29].

Lattice-gas cellular automata models help to study the physical interactions of groups of particles based on the conservation laws of mass and momentum [27, 41]. Usually, the state of the system is updated based on collision and propagation [41]. These systems are discrete in space, time and state [41]. There are two possible states for each cell (empty or occupied) and since there is a conservation of mass, it is impossible to define rules for single cells, to avoid violating the conservation, that is why the rules are defined for a node of cells [41, 42]. After each step, each particle is transferred along links to the nearest neighbor node (mass and momentum propagate like a gas between steps) [41]. The evolution of each site is assigned based on the state of the sites in the neighborhood, and after that, the state of each node is propagated to a neighboring node [41].

These models are used mainly to understand the spatio-temporal evolution of a system of a huge cell population interacting and migrating [43]. They are a bridge to continuum models [29, 40].

2.2.1.2 Cellular automata models

In cellular automata models each lattice of a grid represents a single cell (Figure 2.2 (b)) [29, 27, 40]. The behavior of each cell is modeled by discrete rules based on its surrounding state, and there is a finite number of possible states [39]. At each timestep, a cell can stay in its place, move to an empty space, die or divide [29, 27, 44].

If we want to simulate the persistent random walk of a cell in a cellular automata model, with cell proliferation, it is necessary to define its motion's direction at the beginning, its average

speed, the time it has to change direction, and the time until the next cell division. The simulation starts by defining random initial conditions, and at each timestep, each site is randomly picked to be evaluated. This evaluation implies that if the site has a cell, the cell will suffer a division, will change direction, or will move, accordingly with the parameters before [39].

Cellular automata models replace the laws of motion with simpler rules for cells movement, still exhibiting behaviors similar to the real ones [9].

2.2.1.3 Cellular Potts model

In the Cellular Potts model each cell is represented by multiple lattices (Figure 2.2 (c)) [29, 27, 40]. With this model, it is possible to simulate the shape of a cell and its deformations with more resolution and accuracy than the models seen before [27, 9]. The evolution of the system is done with the goal of reducing its effective energy [27, 44]. At each timestep, the model tests a random swap of a cell with a neighboring site and evaluates if the energy will be minimized, and the swap will occur only if it is the case [29].

From a physics point of view, the model describes a system of interacting spins σ in a lattice. Adjacent sites with the same spin are part of the same cell. In the simpler approach, E_0 represents an energy that is related to cell-cell interaction, cell volumes, and cell perimeters. Considering a system with N cells, where i and j represent different lattices, and M represents the cell type, the energy of any configuration of cells is given by:

$$E_0 = \sum_{i,j} J_{\sigma_i\sigma_j}(1-\delta_{\sigma_i\sigma_j}) + \sum_{M=1}^{N} \kappa(A_M - A_0)^2 + \sum_{M=1}^{N} \Gamma(P_M - P_0)^2, \tag{2.2}$$

where $J_{\sigma_i\sigma_j}$ is the surface energy between spins σ_i and σ_j, $\delta_{\sigma_i\sigma_j}$ is the Kronecker delta ($\delta_{\sigma_i\sigma_j} = 1$ if $\sigma_i = \sigma_j$ and $\delta_{\sigma_i\sigma_j} = 0$ if $\sigma_i \neq \sigma_j$, κ models cells' volume incompressibility, A_M is cell M area, A_0 is the desired area, Γ models the contractility of the cell, P_M is cell M perimeter and P_0 is the desired perimeter. The first term represents the boundary energy of the interacting cells, and the second and the third terms represent the energies of the area and perimeter elasticities of the cells [45].

This model is used when the shape dynamics due to mechanical contact of each cell is relevant [34].

2.2.2 Off-Lattice models

Off-lattice models represent each cell as an individual non-overlapping point and the rules are based on the potential energy of particles [27]. They include biomechanics and off-lattice cell-cell interactions, and can be divided into center-based models and boundary-based models (Figure 2.3). However, they are computationally expensive, and generally have many parameters to take into account [29].

2.2.2.1 Particle-based models

Particle-based models track each cell's center of mass or volume and can also be named center-based models (Figure 2.3 (a)) [29]. Typically, the cells' positions are updated by formulating the adhesive, repulsive, locomotive, and drag-like forces exchanged between cell centers

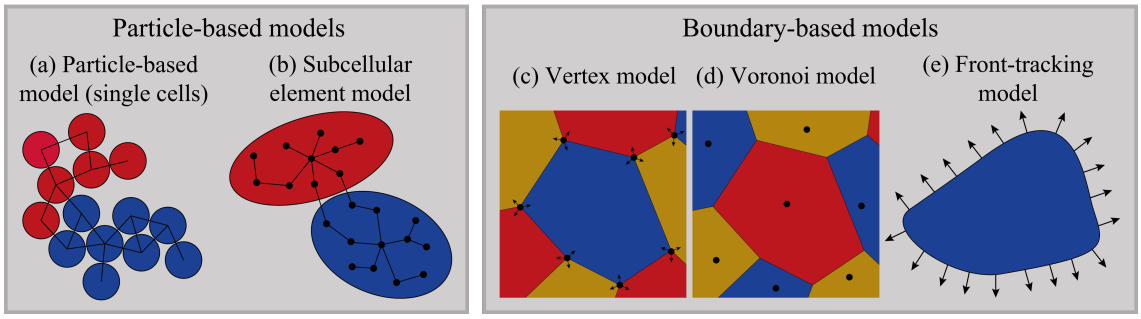

Figure 2.3: Schematic representation of off-lattice models. For every scheme, different colors represent different cells type. (a) Geometric representation of particle-based models for single cells. (b) Geometric representation of subcellular element models. (c) Geometric representation of Vertex models. (d) Geometric representation of Voronoi models. (e) Geometric representation of front-tracking models.

[29]. Usually, these models describe cell movement using the Langevin Dynamics equation, given by:

$$m\frac{\partial \vec{v}_i(t)}{\partial t} = -\mu\vec{v}_i(t) + \vec{F}_i^R(t) + \vec{F}_i^D(t) \tag{2.3}$$

where m is the mass of the entity i, $\vec{v}_i$ is its velocity, μ is the friction coefficient, $\vec{F}_i^R$ is the random stochastic forces applied to it, and $\vec{F}_i^D(t)$ is the deterministic force acting on it [27].

Center-based models can represent more details if each point represents subcellular elements (Figure 2.3 (b)), so each cell is represented by multiple center-based agents that interact with each other [29]. Each subcellular element has an individual entity that contributes to the behavior of the cell. Besides that, the interaction potentials in this case are defined for the subcellular elements, instead of being described for the whole cell. Subcellular element models can simulate in a more realistic manner the changes in cell shape in response to mechanical stimuli and to long-ranged interactions between intra and extracellular elements [46].

This type of model is used when it is important to look at the migrating capacity of cells [34], since center-based models are the only models, from the ones here described, where the cells move freely in a medium since they are not enclosed in a lattice and are not represented as compact tissues. Subcellular element models are computationally expensive but are a better approximation to the real cell biomechanics since they can include the contributions of each cell part [29].

2.2.2.2 Boundary-Based models

Boundary-based models represent cells as polygons (2D) or polyhedra (3D) and compute the forces that act on their vertices. They are very useful to model confluent tissues. These models are computationally intensive but are useful for coupling detailed cell mechanics to fluid and solid tissue mechanics [29].

Vertex models Vertex models represent cells as polygons or polyhedral, that do not overlap, and the different forces in the system are computed as acting on vertices, so the cell vertices are the entities that move (Figure 2.3 (c)) [44, 9]. Each vertex is contained in three or more cells [9, 47, 48]. These models can be useful for modeling confluent tissues, like epithelia [40, 48].

In this type of model, the evolution of each cell is governed by a typical equation of motion:

$$\eta_i \frac{d\vec{r_i}}{dt} = \vec{F_i}, \tag{2.4}$$

where η_i is the drag coefficient, $\vec{r_i}$ is the displacement vector for vertex i, and $\vec{F_i}$ is the net force applied to vertex i [9, 47]. The changes in the system can be evaluated by taking into account two different methods. On one hand, it is possible to define a work function, U, that includes the deformation energy, the membrane surface tension, and the cell adhesion energy, and in this case $-\nabla U = \vec{F_i}$, and can be included in Equation 2.4. On the other hand, the forces of the system can be explicitly defined and its sum equals $\vec{F_i}$ [9, 47].

While using this type of model we need to consider several restructuring rules, to apply to update the system (T1, T2, and T3 transitions and element division). T1 transition, also known as cell neighbor exchange, happens when the distance between two connected vertices is less than a minimum defined distance. When this occurs, the vertices are moved are placed at the defined threshold or, if they are associated with a single cell, they are merged. allows cell connectivity to change as the cell grows and moves. The T2 transition (element removal) is applied to remove a small triangular element or an existing void. This allows the addition of two mechanisms of cell death - necrosis (instantaneous death) and apoptosis (cell will decrease size until being destroyed). The T3 transition, also known as element transition, occurs when a vertex intersects a cell edge, so it will be included in that edge. Element division occurs to mimic mitosis, dividing a cell into two equal cells. In this case, two new vertices are placed in the intersection between the edges and a newly added edge in the center of the parent cell [44, 9, 47].

Voronoi models Voronoi models are similar to Vertex models, in the sense each cell is represented by a polygon or a polyhedron, but the spacial position of each cell is represented by its center (Figure 2.3 (d)). Since the cell centers describe the mesh, Voronoi models have fewer degrees of freedom [49].

In these models, the positions of the element's center are used to define the boundaries using a method called Voronoi tesselation [40, 49]. The mesh will be more dynamic than in Vertex models, since the boundaries are updated at each time step, and do not depend on the rules defined for Vertex models [49]. The partition of space is defined according to the neighborhood relations of a given set of points in the space [50]. The system dynamics is governed by an energy function that depends on the area and perimeter of each Voronoi cell, such as in Vertex models, but focusing on the element center instead of the vertices. The mechanical properties of the tissue will depend on the activity and shape parameter of the cells [51].

Front-tracking models Front-tracking models use partial differential equations to describe flows inside cells, outside cells, and even between them. With the description of flows, it is possible to determine boundary points along the cells' membranes, and include elastic properties of the membrane and its deformation (Figure 2.3 (e)) [29, 52]. A set of conservation equations is defined to solve the flow field, taking into account the intra and extracellular mediums [52].

2.3 Hybrid Discrete-Continuum models

Hybrid Discrete-Continuum models appear due to a necessity to overcome the computational cost of discrete models for complex systems [27]. These models allow one to integrate different lengths and time scales, so they are considered multiscale [28, 27, 34]. This characteristic makes hybrid models a helpful tool that allows using the benefits of both continuum and discrete descriptions (Figure 2.4) [37].

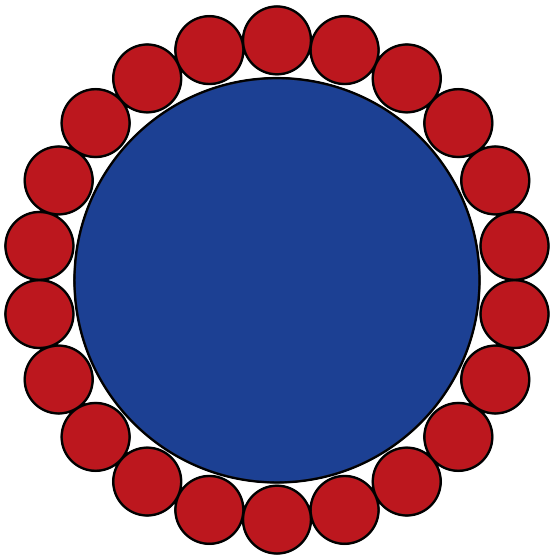

Figure 2.4: Visual representation of a hybrid model. A tumor can be represented by a continuum necrotic center (blue) with a discrete periphery that represents the active cells (red).

The union of continuum and discrete models can be done in two different ways. The first is to treat cells discretely, and they interact and evolve in response to other chemical and mechanical continuum fields (e.g. concentration of oxygen, nutrients, or drugs) - these models are called composite hybrid models. The other way is to define cells with both types of representations, depending on the cell function and/or spatial distribution. These models are called adaptative hybrid models and are used, for example, in tumors with a necrotic center. The model will represent the bulk as a continuum field, and the periphery by individual cells [37].

2.4 Selection Criteria for Cell Modeling

Knowing how to choose an adequate model to approach a cellular behavior problem is a big challenge. We need to evaluate the necessary level of detail for our research. This means deciding if we need to describe molecular details and know how they will evolve during a certain timescale, or if we are only interested in a bigger scale like for example understanding how tissues will grow in response to the environment [35]. If our main goal is to study tissue and its evaluation, without going into greater detail about the entities that compose it, the best choice would be a continuum model. If, on the other hand, the individual cell behavior is relevant to the problem, a discrete model would be a better approach.

If our choice is a discrete model, there are different things that we need to have into consideration:

- Are we facing a confluent tissue?

- Do we want to observe complex shape changes or the cell can be represented as point-like?
- Is motility important?
- Which type of interactions will be included in our model?

If the answer to the first question is positive, the best choice would be one of the boundary-based models. Otherwise, a lattice-based model or a center-based model should work. If we want to observe complex shape changes, a cellular-Potts model, a subcellular element model, or a front-tracking model is more suitable. If it is not an important feature, lattice-gas cellular automata, a cellular automata model, or a center-based model for single cells, may be a better choice. On the other hand, if we want to consider cell motility, the best choices would be lattice-gas cellular automata or center-based models. Finally, if we want to have complex interactions between cells, the best models are the center-based models, since those models can include a wider range of potentials, that can be explicitly defined.

For more complex systems, where we need to take into consideration continuum characteristics, but also have some time of discretization hybrid models are a better choice.

Our choice for both models described in this thesis was discrete models. For the first one, described in Chapter 3 a subcellular element model was chosen since it was necessary to have a cell with an adaptable shape and analyze its mechanical interaction with a granular bed. The model described in Chapter 4 is a center-based model for single cell, since we only wanted to analyze the motility of individual cells.

Chapter 3

Cell mechanotransduction in a granular bed

Cell aggregates spread analogously to the spreading of a viscous droplet since, after contact with the substrate, it will flatten and spread [3]. This is possible due to a process called mechanotransduction, described in Section 1.

Granular systems are of greater interest due to their fluid-like behavior upon external stimuli and due to their several applications. In health fields, specifically, granular materials are used as carriers for drug delivery and for regenerative medicine, as building blocks for cell scaffolds, and even as surfaces for support cell expansion [4]. It is known that cell interactions with particles that are smaller than the cell will include particle internalization (phagocytosis), while particles that are much larger than the cell will support cell adhesion. However, the cell behavior when interacting with particles in between those sizes is not well documented yet [4, 3].

To understand how cells interact with inert materials, we need to focus on smaller scales, since cell adhesion will be mediated by integrins, in an initial moment. Due to these bonds, internal forces will arise, in order to regulate adhesion, spreading, and maturation. At this point, another protein stands out, and it is talin. When unfolded, it reinforces the adhesion by linking the actin cytoskeleton and integrins via vinculin recruitment. This process will allow the cell to exert enough force for the formation of focal adhesion. Without it, the adhesion disassembles and the cell will detach [4].

Cunha et al. [4] studied the influence of a mobile granular bed with different particle sizes in cell response, due to the relevance of granular systems in cell expansion and focusing on cell adhesion. Their hypothesis is that microparticles may offer enough support for cells to interact in a fluid-like environment, where they will pull microparticles and move them. To prove that the initial adhesion and its reinforcement are relevant processes for this mechanism, experimental and numerical approaches were used [4].

The goal of the work reported in this Chapter is to adapt the numerical approach already used in Cunha et al. [4] work, by using an open-source tool, in order to enable other researchers to use and develop research related to this subject.

3.1 Model

Computational approaches to cell adhesion have gained importance as a helpful tool to understand its dynamics and role in morphogenesis. In this model, we pretend to study how a cell adapts to a granular bed according to its relative size. This model was adapted from the model in the work reported in Cunha et al. [4], with a new approach. Our main goal is to develop this model with an open-source tool, in this case, Large-scale Atomic/Molecular Massively Parallel Simulator [53] (LAMMPS). The Langevin dynamics equations define the movement of the particles and cell elements and we integrate the equations of motion, using a velocity Verlet scheme implemented in LAMMPS. All files mentioned in this Chapter are available in `https://github.com/tatiana-rebocho/mechanotransduction`.

3.1.1 Implementation of the bed

We consider a granular bed with 1200 particles, that allow us to have a bed with 3 layers of spheres. These spheres have a diameter with a random Gaussian distribution around d and variance 0.05 and are added with random positions in a box with size $20d \times 20d \times 20d$.

To create a data file with the random positions for the particles, with the desired diameter distribution, we use a script in Python called particles-data.py. In line 1 we import the library random since we use it to generate the random diameter. Line 3 initializes a for cycle that will enable us to create 16 data files, to have different bed configurations. Each data file will be named i.bed.data, where i is the bed index as shown in line 8. In lines 16 to 23, we print in the data file the initial information about the system. This includes the number of particles and bonds and the number of particle types and bond types.

particles-data.py

```
1  import random
2
3  for i in range(1,17):
4
5    n = 1200 #number of particles
6
7    #data file name, where i is the bed index
8    file_name = str(i)+ ".bed" + ".data"
9
10   #open file file_name to write it
11   f = open(file_name, "w+")
12
13   tp = 1 #number of particle types
14
15   #printing of information needed in data file
16   f.write("# lammps molecular data \n \n")
17
18   #definition of number and types of particles and bonds
19   f.write(str(n) + " atoms \n \n")
20   f.write(str(0) + " bonds \n \n")
21
22   f.write(str(tp) + " atom types \n \n")
23   f.write(str(0) + " bond types \n \n")
```

In lines 26 to 29 we defined the mean diameter of the particles, and the size of the simulation box, that like described before is $dim = 20d$. Lines 31 to 33 print in the data file the size of the simulation box for each dimension. Next, we open the Atoms section (line 35), where we will be printing the coordinates and parameters of each particle. Each particle will be defined with a molecule ID, since we will add a patch per particle when an adhesion occurs, so in line 38 we initialize the molecule counting. In lines 40 to 52, we have a for cycle that runs for each particle, where we calculate its coordinates, its diameter, and its density, and then we print it in the data file. We finalize the script by closing the data file in line 54.

particles-data.py

```
25    #definition of the mean diameter
26    diameter = 1.00
27
28    #definition of the dimensions of the box
29    dim = 20 * diameter
30
31    f.write("0 "+str(dim)+" xlo xhi \n") #dimensions in x
32    f.write("0 "+str(dim)+" ylo yhi \n") #dimensions in y
33    f.write("0 "+str(dim)+" zlo zhi \n \n \n") #dimensions in z
34
35    f.write("Atoms \n \n")
36
37    #initializes the molecule counting (each particle will be part of a molecule)
38    mol = 1
39
40    for k in range (1, n+1): #cycle for particle ID
41      x = round(random.uniform(0,dim), 4) #position in x
42      y = round(random.uniform(0,dim), 4) #position in y
43      z = round(random.uniform(2/20*dim,10/20*dim), 4) #position in z
44      diam = random.gauss(diameter, 0.05) #particle diameter
45      den = 1.000 #particle density
46
47      #prints a line per particle
48      print (k, 1, x, y, z, mol, diam, den, file=f)
49      #particleID particleType x y z moleculeID diameter density
50
51      #increases molecule ID
52      mol = mol + 1
53
54    f.close()
```

After having the 16 initial configurations for the bed, we need to run it in LAMMPS, in order to have relaxed granular beds. The granular-bed.run script will enable us to do it. To generate a granular bed, the particles have a gravitational acceleration applied in the vertical direction. The interactions between the particles and between the particles and the surface are given by the Hooke potential, without the tangential part, defined as:

$$\vec{F}_{Hooke} = K_{pp}\delta_{ij}\vec{n} - \eta_{pp}am_{eff}\vec{v}_{n,rel}, \tag{3.1}$$

where K_{pp} is the spring stiffness, $\delta_{ij} = R_i + R_j - ||\vec{r}_{ij}||$ is the particle overlap, R_i, R_j are the

particle radii, $\vec{r}_{ij} = \vec{r}_i - \vec{r}_j$ is the vector that separates the two center points of the particles, $\vec{n} = \frac{\vec{r}_{ij}}{||\vec{r}_{ij}||}$, η_{pp} is the damping prefactor, a is the radius of contact, given by $a = \sqrt{R_{ij}\delta}$, $m_{eff} = m_i m_j/(m_i + m_j)$ and $\vec{v}_{n,rel} = (\vec{v}_j - \vec{v}_i)\cdot\vec{n}$.

The simulations to generate the bed go for enough time to have a compact and relaxed granular bed (Figure 3.1).

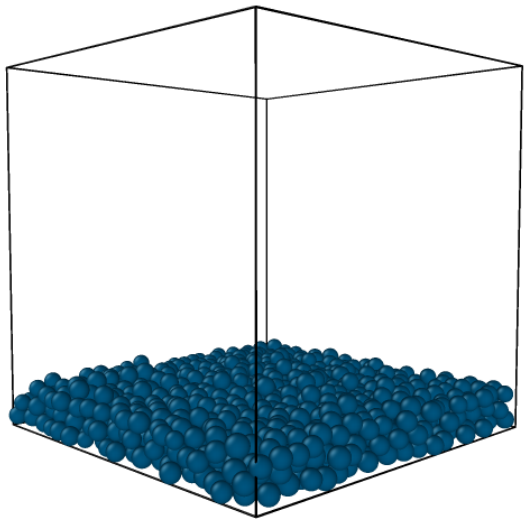

Figure 3.1: Representation of a relaxed bed with three layers.

We start by defining a loop variable indx (line 1) to run the script to each initial conditions data file. Line 3 will enable us to see the progress of the simulation on screen. In line 5 we define the system units as reduced Lennard Jones units. Lines 7 to 10 define the dimensions of the simulation as three-dimensional, the atom style as spheres that can be grouped in clusters, and the boundaries as periodic with the exception of the one in the z dimension. The command in line 10 turns Newton's third law off, so if two interacting atoms are on different processors, both processors compute their interaction. In line 12 we include the data file with the desired index.

granular-bed.run

```
1  variable indx loop 1 16
2
3  echo screen
4
5  units lj
6
7  dimension 3
8  atom_style hybrid molecular sphere
9  boundary  p p f
10 newton  off
11
12 read_data  ${indx}.bed.data
```

In lines 16 to 30, we define the variables that are constant and will be used for future commands using variable equal. Here it is possible to find variables related to the subcellular elements and the particles, like the radius, but also simulation variables, like the temperature,

the damping parameter, the timestep and the number of steps, the elastic constants, and the random number generator seed. To have the output of the simulation we use the command dump that saves the atom ID, the atom type, the coordinates of its center of mass, and its radius at every 1000 steps in a file named indx.bed.dump. This file can be read in external software, like Ovito (https://www.ovito.org), or its data can be extracted and used for different types of studies.

granular-bed.run

```
14  #simulation variables
15
16  variable  Rcell      equal  0.75
17  variable  Rparticle  equal  1.00/2
18  variable  Li         equal  ${Rcell}*4
19
20  variable  temp       equal  0.0
21  variable  damp       equal  ${Rcell}^2
22  variable  timestep   equal  0.001
23  variable  numsteps   equal  1000000
24
25  variable  Kpp        equal  (2*${Rparticle})*10^2
26  variable  gravity    equal  1.0
27  variable  Fmax       equal  (((82/32)*(${Li}/2))^3)/43
28  variable  dampK      equal  (${Fmax}/(5*10^(-2)*${Rcell}))
29
30  variable  seed       equal  82514877
31
32    dump positions all custom 1000 ${indx}.bed.dump id type x y z radius
```

In lines 34 to 39, we add a new command, called fix. These commands are applied to the system during time-stepping. In this case, in line 34 we set the interaction between the particles and the surface in z=0, as a granular Hooke potential, without considering the tangential part of the interaction, as described before, which is represented by setting the values related to it as 0.0

Next, we define the time integration system that will be used to update the position and velocity of each particle at each timestep (velocity Verlet scheme), and we apply a Langevin thermostat to the system, where we define the starting and stopping temperature as constant, and the damping time. We needed to add a fix that define a gravitational acceleration to all particles to simulate their falling movement, in the direction (0,0,-1).

granular-bed.run

```
34  fix wall all wall/gran granular hooke ${Kpp} ${dampK} tangential linear_nohistory
0.0 0.0 zplane 0.0 NULL
35
36  fix integrator all nve
37  fix 1 all langevin ${temp} ${temp} ${damp} ${seed}
38
39  fix grav all gravity ${gravity} vector 0 0 -1
```

Lines 41 to 47 are used to include compute commands. These are used to calculate some parameters from the simulation. The compute msd calculates the mean-squared displacement of the group, and outputs a vector with 4 quantities - the first three elements are the squared displacements for each coordinate, summed and averaged over atoms in the group and the fourth is the total squared displacement, summed and averaged over atoms in the group. The computes ke/atom and pe/atom are used to calculate the per-atom translational kinetic energy and the potential energy for each atom in a group, respectively, and output a per-atom vector. The computes used in lines 46 and 47 are used to reduce the per-atom vectors from the computes described before in scalar values, which are equal to the total sum of each element of the vector.

granular-bed.run

```
41  compute  msda all msd
42
43  compute  atmKE  all  ke/atom
44  compute  atmPE  all  pe/atom
45
46  compute  sumKE  all  reduce sum c_atmKE
47  compute  sumPE  all  reduce sum c_atmPE
```

The interaction potential between particles is defined in lines 49 and 50, where we set it as a granular Hooke potential, without considering the tangential part of the interaction. The command in line 52 is used to enable the velocity information to be communicated with ghost particles, so ghost atoms will store quantities related to the velocity. It can include the translational velocity, angular velocity, and angular momentum of a particle.

We define that the information that goes to the log file will be printed at every 10000 steps in line 54, and the timestep in line 55. The name used for the log file is set in line 56. In line 57, we determine the content that will be printed in it. In this case, it will be the step, the sum of kinetic and potential energies calculated in the computes defined before, and the total mean-squared displacement of the system. The simulation will run for 10000000 steps, as set in line 58.

granular-bed.run

```
49  pair_style granular
50  pair_coeff * * hooke ${Kpp} ${dampK} tangential linear_nohistory 0.0 0.0
51
52  comm_modify vel yes
53
54  thermo 10000
55  timestep ${timestep}
56  log  ${indx}.bed.thermo
57  thermo_style  custom step c_sumKE c_sumPE c_msda[4]
58  run 10000000
```

These conditions are not the desired ones for the relaxed bed, since it was necessary to use a lower elastic constant in the interaction at the beginning, due to numerical problems. In lines 60 and 61 we update the elastic constant used in the pair style, and we update it in lines 63 and 64. Next, we update the timestep and run the simulation again, with the new conditions.

In line 69, we give the instruction to write a data file named indx.data.granular_bed that will include the state of each particle in the system at the last timestep of the simulation.

granular-bed.run

```
60  variable  Kpp  delete
61  variable  Kpp  equal  (2*${Rparticle})*10^4
62
63  pair_style granular
64  pair_coeff * * hooke ${Kpp} ${dampK} tangential linear_nohistory 0.0 0.0
65
66  timestep 0.001
67  run ${numsteps}
68
69  write_data ${indx}.data.granular_bed
70
71  clear
72  next           indx
73  jump           SELF
```

In lines 71 to 73, we restore all settings to the default values, delete all particles and free up all memory, start using the next bed configuration and jump to the beginning of the script, to re-run it to all the bed configurations.

3.1.2 Implementation of the cell

We model the cell as a group of subcellular elements, and it has a maximum of three layers (Figure 3.2). The initial cell has 7 elements, organized in a hexagonal configuration, with diameter L_i.

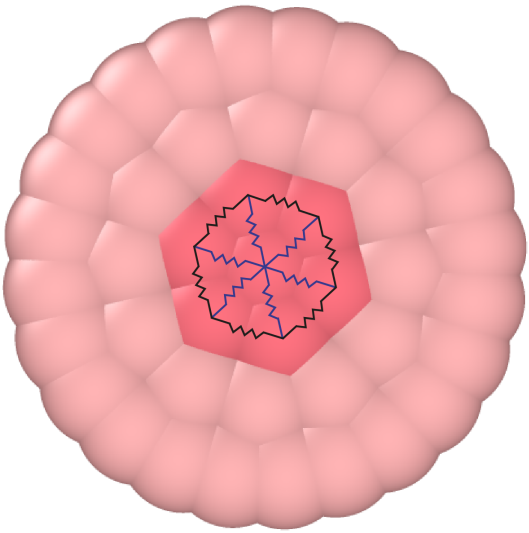

Figure 3.2: Representation of a cell at the beginning of the simulation, and of the springs that connect them. The dark elements are the active subcellular elements.

The cell will suffer cycles of stretching and contraction, in order to mimic the growth of the cell and its fixation with the microparticles. During the stretching phase, two elements will be added per subcellular element, in the perimeter of the cell. After this, the cell will

relax and contract. All subcellular elements are implemented at the beginning, however, all the deactivated elements have their mass reduced by 10^3, and all other properties are re-scaled accordingly.

To generate the cell data files, we use a Python script named cell-data.py. We start it the same way the one for the initial conditions for the bed configurations was initiated. We have a cycle for all the data files, there will be 10. We define its name, the number of atoms and bonds, and the number of atoms types and bond types. Next, we print it in the data file. All these steps are similar to the ones at the beginning of the file particles-data.py, described before.

When we start the Atoms section, we initialize the particle counting, and the molecule counting and we set the mean diameter for bed particles and the dimensions of the simulation box. To locate the cell in a place that will not intersect with the granular bed or with the boundaries, we set inferior and superior limits for the axis x and y in lines 42 to 45. In lines 48 and 49 we choose a random position for the center subcellular elements of the cell. In line 51, we set the height where all subcellular elements will be as $z = diam\_mp \times 5$. In lines 54 to 56, we set the diameter and the radius of each subcellular entity and its density. At this point, the settings for the central subcellular element will be printed to the data file, in line 59.

cell-data.py

```
31    #type 1
32
33    k = 1 #initializes the particle counting
34
35    mol = 1 #defines the molecule counting (1 per cell)
36
37    diam_mp = 1.00 #definition of the mean diameter for microparticles
38    dim = 20 * diam_mp #definition of the dimensions of the box
39
40    #definition of the limits where the cell can exist
41
42    x_inflim = 6.5
43    x_suplim = dim - 6.5
44    y_inflim = 6.5
45    y_suplim = dim - 6.5
46
47    #randomly chooses the center of the cell (x_o,y_o)
48    x_o = random.uniform(x_inflim,x_suplim)
49    y_o = random.uniform(y_inflim,y_suplim)
50
51    z = diam_mp * 5 #defines the at which height the cell will start z
52
53    #definition of the diameter of each subcellular elements its radius and its density
54    diam = 1.5
55    radius = diam / 2
56    den = 1.000
57
58    #prints a line per particle
59    print (k, 2, x_o, y_o, z, mol, diam, den, file=f)
60    #atom-ID atom-type x y z moleculeID diameter density
```

The information for each subcellular element will be processed by layer, so next, we will

explain how we treat it for the first layer (lines 62 to 85), and the others are treated in a similar way. First, we define the number of entities per layer. To have the right position of each subcellular element, since the cell is disk-like, it is necessary to define angles for each subcellular element position. So in the next step, we initialize the list that will include the angles at which each element will be from the center and define the angle, θ, for the first considered subcellular element. The first for cycle will run over all the elements, to complete the list of angles. In it, we append the θ value to the list, and next, we calculate the next θ. In the next for cycle, we run over the angles list, set the coordinates for the considered entity, and we print its settings in the data file. This process is repeated for the next layers, modifying only the type of particle and the initial angle considered.

cell-data.py

```
62    #type 2
63
64    n_layer = 6 #number of cells per layer
65    #initializes the list at which angle the subcellular element will be
66    teta_list = []
67    #defines the first angle (in rad)
68    teta = 0
69
70    #for each element there will be an angle teta
71    for n in range(n_layer):
72      teta_list.append(teta)
73      teta = teta + 2 * math.pi/ n_layer
74
75    #for each angle we define the element position x and y
76    for teta in teta_list: #particle ID
77      x = x_o + radius * math.sin(teta)
78      y = y_o + radius * math.cos(teta)
79
80      #increases atom ID
81      k = k + 1
82
83      #prints a line per particle
84      print (k, 2, x, y, z, mol, diam, den, file=f)
85      #atom-ID atom-type x y z moleculeID diameter density
```

Next, we start the Bonds section, where we will set which subcellular elements are connected, taking into account that each element generates two new elements in the next layer and that it is bonded to the two elements it is in between in the same layer. We start by initiating the section in the data file, generating lists for each layer of subcellular IDs (lines 144-147), and starting the bond ID and bond type counters (lines 150 and 151).

cell-data.py

```
144   type_1 = [1]
145   type_2 = [*range(2,7+1)]
146   type_3 = [*range(8,19+1)]
147   type_4 = [*range(20,43+1)]
```

cell-data.py

```
149    #initializes the IDbond and bondType counting
150    ID_bond = 1
151    bond_type = 1
```

In lines 156 to 159 we define the bonds between the central subcellular element and all the elements in the second layer, with a for cycle that will run for every entity in the second layer, and prints the necessary information (bond ID, bond type, central cell ID and second layer element). From line 161 to line 178 the bonds between elements from the second layer are included in the data file. This is done with a for cycle, like before. If the index of the list is zero, we are considering the first element of the layer, and it will be connected to the last element, and to the next one. Otherwise, we only include the bond between the considered element and the next one, so there will not exist repeated bonds in the data file.

cell-data.py

```
156    for x in type_2: #prints a line per bond
157      print(ID_bond, bond_type, type_1[0], x, file=f)
158      #bondID bondType elementID1 elementID2
159      ID_bond = ID_bond + 1 #increases bond ID
160
161    #bonds for type2 with type2
162
163    bond_type = bond_type + 1 #increases bond type
164
165    #for each element in type2 there will be a bond with the two elements next to it in
the same layer
166    for n in range(0,len(type_2)-1):
167      #for the first element in the list it will be bonded with the last element in it
and with the next one
168      if n == 0:
169        print(ID_bond, bond_type, type_2[n], type_2[len(type_2)-1], file=f)
170        #bondID bondType elementID1 elementID2
171        ID_bond = ID_bond + 1 #increases bond ID
172        print(ID_bond, bond_type, type_2[n], type_2[n+1], file=f)
173        #bondID bondType elementID1 elementID2
174        ID_bond = ID_bond + 1 #increases bond ID
175      else:
176        print(ID_bond, bond_type, type_2[n], type_2[n+1], file=f)
177        #bondID bondType elementID1 elementID2
178        ID_bond = ID_bond + 1 #increases bond ID
```

Next, in lines 182 to 198, we include the bonds between the second layer and the third one. We use k as the first element ID in the third layer, and this constant will increase. While k is less than the last element ID in the third layer, we will connect two elements from this layer to one element in the second layer. To do this, we use j to count 2 bonds per element, with a while cycle.

cell-data.py

```
180    #bonds for type2 with type3
181
182    bond_type = bond_type + 1 #increases bond type
183
184    k = type_3[0] #k is the first element ID for list type3
185
186    #while k is less or equal to the last element ID of type3
187    while k <= type_3[len(type_3)-1]:
188      for x in type_2: #for each element in type2
189        #two elements of type3 will be bonded to an element of type2
190        #j counts the two bonds
191        j = 1
192        while j <=2:
193          print(ID_bond, bond_type, x, k, file=f)
194          #bondID bondType elementID1 elementID2
195          ID_bond = ID_bond + 1 #increases bond ID
196          #increases k and j
197          k = k + 1
198          j = j + 1
```

The bonds between elements in the third layer, between the third layer and the fourth, and between the elements in the fourth layer are defined in a similar way. This definition is finished by closing the data file.

3.1.3 Implementation of the simulation

With the cell data files created and all the granular beds relaxed configurations generated, it is possible to run the model where we lay a cell in a granular bed and observe how it adheres and grows. In order to optimize the script, we compiled LAMMPS as a Python2.7 library. This requirement appeared from the constant need to update the distances between different elements in the simulation. Since LAMMPS is not optimized for this kind of cycle, we needed to find a better solution. So, we created a main file model.py, which will call several LAMMPS scripts, to update the simulation.

We start this script by importing the several needed libraries (lines 1-5). Next, we define the threshold to freeze the growth of the cell, because this is the only one that is constant during the whole simulation, and will be used in this script. The constants defined in lines 9-11 and related to the particles' size, the bed file index, and the cell file index.

model.py

```
1  import shutil
2  import math
3  import os.path
4
5  from lammps import lammps
6
7  thr_freeze = 10**(-4)
```

model.py

```
 9  size_dim = SIZE
10  bed_ID = BEDID
11  runidx = [ID_FILE]
```

We then start a for cycle in line 13 to the considered cell. This cycle initiates with calling LAMMPS in line 15. Each time that we will use a LAMMPS command it will start with lmp. In line 17 we include the variable.run file, where all the necessary constant variables are set. Then in lines 19 to 21, we set variables related to the necessary indexes, mentioned before. In lines 23 to 25, we include three different files in sequence, which we describe below.

model.py

```
13  for run in runidx:
14
15    lmp = lammps()
16
17    lmp.file("variables.run")
18
19    lmp.command("variable bedID equal "+str(bed_ID))
20    lmp.command("variable runindx equal "+str(run))
21    lmp.command("variable size equal "+str(size_dim))
23    lmp.file("initialization.run")
24    lmp.file("bed-definition.run")
25    lmp.file("cell-inclusion.run")
```

The file initialization.run is used to set the principal characteristics of the system, in a similar way as the simulation for relaxing the granular bed, described before. The main difference is in the command read data, in line 10, where we read the relaxed bed file, and set the system to allow 100 extra bonds per particle and 150 extra 1-2,1-3,1-4 interactions per atom.

initialization.run

```
 1  echo screen
 2
 3  units lj
 4
 5  dimension 3
 6  atom_style hybrid molecular sphere
 7  boundary  p p f
 8  newton  off
 9
10  read_data  ${bedID}.data.granular_bed extra/bond/per/atom 100 extra/special/per/atom 150
```

In file bed-definition.run we define a group called bed, for every particle from type 1, which includes all the particles in the granular bed (line 1). Lines 3 to 21 are similar to what was described before for the relaxation of the granular bed. In line 22 we set the considered cutoff for the system, which is equal to a variable defined in the file variables.run. Lines 24 and 25 are used to set an empty group called FreezeNow, which will be used to define a fix freeze that

removes force and torque on a granular particle, so we are able to include particles in this group, in order to stop entities movement. The information about the evolution of the system will be saved in a dump file, as defined in line 27. Besides that, there will be a log file, and the system will run with the defined characteristics for a determined number of steps, as set in lines 29 to 33.

bed-definition.run

```
1  group bed type 1
2
3  fix wall all wall/gran granular hooke ${Kpp} ${dampK} tangential linear_nohistory 0.0
0.0 zplane 0.0 NULL
4
5  fix integrator all nve
6  fix 1 all langevin ${temperature} ${temperature} ${damp} ${seed}
7
8  fix grav bed gravity 1.0 vector 0 0 -1
9
10 compute  msda all msd
12 compute  atmKE  all  ke/atom
13 compute  atmPE  all  pe/atom
14
15 compute  sumKE  all  reduce sum c_atmKE
16 compute  sumPE  all  reduce sum c_atmPE
17
18 pair_style granular
19 pair_coeff * * hooke ${Kpp} ${dampK} tangential linear_nohistory 0.0 0.0
20
21 comm_modify vel yes
22 comm_modify cutoff ${cutoff}
23
24 group FreezeNow empty
25 fix Freeze FreezeNow freeze
26
27   dump positions all custom 1500 ${bedID}.${runindx}.model.dump id type x y z radius
28
29 thermo 1000000
30 timestep  0.001
31 log  ${bedID}.${runindx}.model.thermo
32 thermo_style  custom step c_sumKE c_sumPE c_msda[4]
33 run  ${numstepsBed}
```

When the simulation stops running, the file cell-inclusion.run will be included. In this file, we add the cell, saved in the file runindx.cell.data, and we sum 1200 to the atom ID and to the molecule ID saved in this file, using the commands in line 1. The cell will be at a height high enough to not touch the granular bed. The commands in lines 3 to 5 are used to define the different subcellular elements' masses, so only the active part of the cell (central element and the first layer, at this point) has relevance in the simulation. These masses are set in the file variables.run, as constants. In line 7, we include the file interaction.run, where each interaction in the simulation can be set. All the interactions are defined also by a Hooke potential without the tangential part, as in file granular-bed.run, however, the spring stiffness' is different for each

case (interaction between particles and subcellular elements, active subcellular elements, and nonactive subcellular elements, and patches with every other entity).

cell-inclusion.run

```
1  read_data       ${runindx}.cell.data add 1200 1200
2
3  set type 2 mass ${cellMass}
4  set type 3 mass ${reducedMass}
5  set type 4 mass ${reducedMass}
6
7  include interaction.run
```

In lines 9 to 17 from file cell-inclusion.run we set the harmonic bond type to every bond, with $E = K_{cell}(r - r_0)^2$, where r_0 is the equilibrium distance between entities and r is the actual distance between them. There will be different equilibrium distances and elastic constants for each bond type, and these constants were set in file variables.run. In line 19, we set the gravity for all the entities in the system, which will allow the cell to fall over the granular bed. Then, in line 21 and 22 we modify the timestep of the simulation and the command run 0 allow us to add the new information, without running the simulation.

cell-inclusion.run

```
9  bond_style harmonic
10 bond_coeff 1 ${bondK} ${r1}
11 bond_coeff 2 ${bondK} ${r2}
12 bond_coeff 3 ${bondKdr} ${r3}
13 bond_coeff 4 ${bondKdr} ${r4}
14 bond_coeff 5 ${bondKdr} ${r5}
15 bond_coeff 6 ${bondKdr} ${r6}
16 bond_coeff 7 ${bondmc} ${r7}
17 bond_coeff 8 ${bondinit} ${r7}
18
19 fix grav all gravity 1.0 vector 0 0 -1
20
21 timestep  ${timestep}
22 run  0
```

At this point, we come back to file model.py. In lines 29 and 30 we define two different lists. The first one has all the cell IDs, and the second has only the IDs of the active subcellular elements. The command in line 32 is used to save the local properties of the simulation, like coordinates at that moment and the entities' radius. Since it does not have entity IDs, we use the command in line 33 to save it. The object IDs do have not a list format, so we create one, and we use a for cycle to define a list of entities IDs (lines 34-37). In line 39 we save the total number of entities since in LAMMPS there are no spaces between indexing at the beginning of the simulation. The lists initiated in lines 41 to 43 are necessary for the creation of patches, which will be the adhesion point between subcellular elements and particles from the granular bed. These bonds are illustrated by adding a patch to the particle that touches the subcellular element (Figure 3.3).

In line 45 we start a for cycle, that will evaluate the system 600 times. During this cycle,

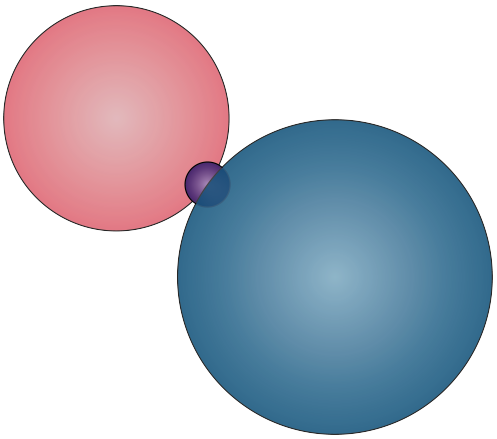

Figure 3.3: Representation of an activated subcellular element (pink) connected to a patch (purple), that belongs to a particle (blue) that is bonded to the subcellular element.

the cell will fall over the granular bed, and we will be creating patches in the contact points. Since the local properties from the system alter their order at each step, we need to initiate the cycle by calling all those properties and creating the IDS list, since our calculations will be made based on it, because we know that all subcellular elements have IDs between 1201 and 1243, and all the particles before it belong to the granular bed (lines 46-53).

model.py

```
29  cell = range(1201,1243+1)
30  active_layer = range(1201,1207+1)
31
32  nlocal = lmp.extract_global("nlocal")
33  ids = lmp.extract_atom("id")
34  ids_list = []
35
36  for index in range(nlocal):
37    ids_list.append(ids[index])
38
39  n_atoms = max(ids_list)
40
41  pair_cell_patch = []
42  patches_IDs = []
43  mp_patch =[]
44
45  for i in range(0,600):
46    nlocal = lmp.extract_global("nlocal")
47    x = lmp.extract_atom("x")
48    ids = lmp.extract_atom("id")
49    rad = lmp.extract_atom("radius")
50    ids_list = []
51
52    for index in range(nlocal):
53      ids_list.append(ids[index])
```

In line 55 we start a for cycle that will run over all the particles in the granular bed and save the index of this element in line 56. Next, for each particle, we will run over all the subcellular

elements (line 58), to save the coordinates from each considered entity and calculate the distance between each subcellular element and granular particle (lines 59-69). At this point, we save the sum between the radius of the considered elements and the pair. If the distance between the pair is less or equal to this sum, and the pair is not in the list that saves each pair that checks it (line 73), we will append the pair to the list, and position a patch at the contact point between them using the command in line 82. All the patches will have particle type 8. Then we increase the particle counting, which will be equal to the patch ID, and we calculate the distance between the patch and the cell at this moment, to save them in lines 86 and 87. In line 85, we include the patch in the molecule to which the granular particle in this interaction belongs to. In lines 89 to 93 we decide which bond type will be considered, taking into account if the subcellular element in the bond is active or not, since if it is not, it will not influence the dynamics of the system.

model.py

```
55    for IDmp in range(1,1201):
56      i_mp = ids_list.index(IDmp)
57
58        for ID in range(1201,1244):
59          i_cell = ids_list.index(ID)
60          Xmp = x[i_mp][0]
61          X = x[i_cell][0]
62          Ymp = x[i_mp][1]
63          Y = x[i_cell][1]
64          Zmp = x[i_mp][2]
65          Z= x[i_cell][2]
66          DX = (Xmp-X)**2.0
67          DY = (Ymp-Y)**2.0
68          DZ = (Zmp-Z)**2.0
69          dist = (DX+DY+DZ)**(0.5)
70          rad_sum = rad[i_cell] + rad[i_mp]
71          pair = [IDmp,ID]
72
73          if dist <= rad_sum and pair not in mp_patch:
74            mp_patch.append(pair)
75            frac = rad[i_cell]/ dist
76            vec_x = x[i_mp][0]-x[i_cell][0]
77            vec_y = x[i_mp][1]-x[i_cell][1]
78            vec_z = x[i_mp][2]-x[i_cell][2]
79            x_p = x[i_cell][0] + frac * vec_x
80            y_p = x[i_cell][1] + frac * vec_y
81            z_p = x[i_cell][2] + frac * vec_z
82            lmp.command("create_atoms 8 single " +str(x_p)+ " "+str(y_p)+ " "+str(z_p))
83            n_atoms = n_atoms + 1
84            dist_patch = ((x[i_cell][0]-x_p)**2.0+(x[i_cell][1]-y_p)**2.0+(x[i_cell][2]
-z_p)**2.0)**0.5
85            lmp.command("set atom "+str(n_atoms)+ " mol " + str(IDmp))
86            pair_cell_patch.append([ID,n_atoms,dist_patch])
87            patches_IDs.append(n_atoms)
```

model.py

```
89            if ID in active_layer:
90              lmp.command("create_bonds single/bond 7 " + str(ID) + " " + str(n_atoms))
91
92            else:
93              lmp.command("create_bonds single/bond 8 " + str(ID) + " " + str(n_atoms))
```

Next, we set the patch diameter, and its mass (lines 95-96) and we freeze the patches, so they will not rotate, and we will update the integrator of the simulation, so each molecule that includes a patch and a granular particle is considered as a rigid particle, and not as two individual entities (lines 97-103). Then, the simulation will run for 100 steps. The main for cycle, will run 600 times until the cell relaxes completely. When this happens, we update all patches, by deleting them and repositioning them in an identical way to the one described before.

model.py

```
 95    lmp.command("set type 8 diameter 0.1")
 96    lmp.command("set type 8 mass 0.000001")
 97    lmp.command("group MPpatches type 8")
 98    lmp.command("group FreezeNow union FreezeNow MPpatches")
 99    lmp.command("group MPpatches include molecule")
100    lmp.command("group langevin subtract all MPpatches")
101    lmp.command("fix integrator langevin nve")
102    lmp.command("fix 1 langevin langevin ${temperature} ${temperature} ${damp} ${seed}")
103    lmp.command("fix patches MPpatches rigid/small molecule langevin ${temperature}
${temperature} ${damp} ${seed}")
104
105    lmp.command("run 100")
```

At this moment in the simulation, with the cell relaxed and all the adhesion points updated, we start to prepare the first contraction. To do it, we extract the equilibrium distances of the bonds between subcellular elements (r1-r6), in lines 190 to 195, we set 4 lists (central subcellular elements IDS, 3 layers IDs) and other 3, only for the cell layers IDs, since one of the lists will be modified during the simulation, while the other will remain intact (lines 197-204). In lines 206 to 224, we create a dictionary where to each subcellular element we add the elements to which it is bonded, to be possible to delete the whole tree in the next steps.

model.py

```
190  r1 = lmp.extract_variable("r1",group=None, vartype=0)
191  r2 = lmp.extract_variable("r2",group=None, vartype=0)
192  r3 = lmp.extract_variable("r3",group=None, vartype=0)
193  r4 = lmp.extract_variable("r4",group=None, vartype=0)
194  r5 = lmp.extract_variable("r5",group=None, vartype=0)
195  r6 = lmp.extract_variable("r6",group=None, vartype=0)
196
197  centerID = 1201
198  layer1_ID = range(1202,1207+1)
```

model.py

```
199  layer2_ID = range(1208,1219+1)
200  layer3_ID = range(1220,1243+1)
201
202  layer1 = range(1202,1207+1)
203  layer2 = range(1208,1219+1)
204  layer3 = range(1220,1243+1)
205
206  k = layer2[0]
207  q = layer3[0]
208
209  tree_bonds = {i:[] for i in layer1_ID}
210
211  for ID in layer1_ID:
212    tree_bonds[ID].append(ID)
213    j = 1
214
215    while j<=2:
216      tree_bonds[ID].append(k)
217      k = k + 1
218      j = j + 1
219    l = 1
220
221    while l<=4:
222      tree_bonds[ID].append(q)
223      q = q + 1
224      l = l + 1
```

From line 230 to line 244 we calculate the distance between each subcellular element and the central subcellular element, and two different lists are created - one for the distances, and the other for the cell IDs. In line 246 we create a list of all the deactivated subcellular elements that will be deleted during contractions.

model.py

```
230  for ID in layer1_ID:
231    i_c = ids_list.index(centerID)
232    i_n = ids_list.index(ID)
233    centerX = x[i_c][0]
234    X = x[i_n][0]
235    centerY = x[i_c][1]
236    Y = x[i_n][1]
237    centerZ = x[i_c][2]
238    Z= x[i_n][2]
239    DX = (centerX-X)**2.0
240    DY = (centerY-Y)**2.0
241    DZ = (centerZ-Z)**2.0
242    dist = (DX+DY+DZ)**(0.5)
243    distancesBegin.append(dist)
244    IDBegin.append(ID)
```

In line 248 we start the first contraction, with a for cycle that will run 10000 times. Line

249 is used to save the timestep, to be used next. The contraction starts by decreasing the equilibrium distances from the bonds related to active particles down to 0 over the entire cycle (lines 250 and 251). The harmonic bonds are updated with these new distances in lines 252 to 260. Even if we only need to update two bond types, all the bonds need to be redefined with the constants they had before, to avoid errors. In lines 261 to 267, we save the local properties and create a list of IDs. In line 269 we start a for cycle where we will evaluate the situation for each subcellular element. If this element is in the active layer and has a patch associated with it, we will calculate the distance between it and the patch to which it bonds. If the difference between this distance and the initial one is bigger than the threshold set at the beginning of the simulation, the subcellular elements will change their type and will be frozen. This entity no longer belongs to the active layer (lines 273-296).

model.py

```
248 for i in range(0,10000):
249   dt = lmp.get_thermo("dt")
250   dr1 = r1-r1*0.1*i*dt
251   dr2 = r2-r2*0.1*i*dt
252   lmp.command("bond_style harmonic")
253   lmp.command("bond_coeff 1 ${bondK} "+str(dr1))
254   lmp.command("bond_coeff 2 ${bondK} "+str(dr2))
255   lmp.command("bond_coeff 3 ${bondKdr} ${r3}")
256   lmp.command("bond_coeff 4 ${bondKdr} ${r4}")
257   lmp.command("bond_coeff 5 ${bondKdr} ${r5}")
258   lmp.command("bond_coeff 6 ${bondKdr} ${r6}")
259   lmp.command("bond_coeff 7 ${bondmc} ${r7}")
260   lmp.command("bond_coeff 8 ${bondinit} ${r7}")
261   nlocal = lmp.extract_global("nlocal")
262   x = lmp.extract_atom("x")
263   ids = lmp.extract_atom("id")
264   ids_list = []
265
266   for index in range(nlocal):
267     ids_list.append(ids[index])
268
269   for ID in range(1201,1244):
270
271     if ID in active_layer:
272
273       if ID in cell_patch:
274         idx_cell = cell_patch.index(ID)
275         i_cell = ids_list.index(ID)
276         i_p = ids_list.index(patches_IDs[idx_cell])
277         d0 = distance_patchs[idx_cell]
278         cellX = x[i_cell][0]
279         X = x[i_p][0]
280         cellY = x[i_cell][1]
281         Y = x[i_p][1]
282         cellZ = x[i_cell][2]
283         Z= x[i_p][2]
```

model.py

```
284           DX = (cellX-X)**2.0
285           DY = (cellY-Y)**2.0
286           DZ = (cellZ-Z)**2.0
287           dist = (DX+DY+DZ)**(0.5)
288           diff = abs(d0-dist)
289
290           if diff >= thr_freeze:
291             lmp.command("set atom " + str(ID)+" type 7")
292             lmp.command("group zeroforce type 7")
293             lmp.command("group langevin subtract all zeroforce MPpatches")
294
295             if ID in active_layer:
296               active_layer.remove(ID)
```

On the other hand, if the considered subcellular element exists in the initial list of IDs, when calculating its distance to the central subcellular elements, and also belongs to the active layer, we calculate the distance between them and if the distance is less than $0.96d_0$, being d_0 the initial distance, this subcellular elements will change its type and it will be removed from all the lists related with active elements (lines 298-320). Next, all the elements that should grow from it will be saved, the element will be added to the deactivated list, and the elements saved before will change types and also be added to the deactivated list (lines 322-329). In lines 331 to 333, after running all the entities, we create a group of subcellular elements that should be deleted, and we delete all of them and their bonds. In line 335 we set the simulation to run 5 steps. After the contraction, the variables dr1 and dr2 will be saved, so the bonds 1 and 2 will not change anymore during the simulation (lines 337-338).

model.py

```
298         if ID in IDBegin:
299
300           if ID in active_layer:
301             i_begin = IDBegin.index(ID)
302             i_c = ids_list.index(centerID)
303             i_n = ids_list.index(ID)
304             centerX = x[i_c][0]
305             X = x[i_n][0]
306             centerY = x[i_c][1]
307             Y = x[i_n][1]
308             centerZ = x[i_c][2]
309             Z= x[i_n][2]
310             DX = (centerX-X)**2.0
311             DY = (centerY-Y)**2.0
312             DZ = (centerZ-Z)**2.0
313             dist = (DX+DY+DZ)**(0.5)
314             thr = distancesBegin[i_begin] * 0.96
```

model.py

```
316             if dist < thr:
317               lmp.command("set atom "+str(ID)+" type 6")
318               active_layer.remove(ID)
319               layer1_ID.remove(ID)
320               del_atoms = tree_bonds[ID]
321
322               if ID not in desactive:
323                 desactive.append(ID)
324
325               for new_ID in del_atoms:
326                 lmp.command("set atom "+str(new_ID)+" type 6")
327
328                 if new_ID not in desactive:
329                   desactive.append(new_ID)
330
331     lmp.command("group del type 6")
332     lmp.command("delete_atoms group del compress no bond yes")
333     lmp.command("group del delete")
334
335     lmp.command("run 5")
336
337   lmp.command("variable dr1 equal "+str(dr1))
338   lmp.command("variable dr2 equal "+str(dr2))
```

After the contraction, a new stretching moment occurs. To this, the patches need to be removed (lines 342-344), and the next layer will be activated in file next-layer-1.run called in line 346, by increasing the mass of the subcellular elements from the new active layer and re-setting the harmonic bonds. Also, the deactivated elements from this layer need to be removed from the necessary lists for the next simulation steps (lines 350-358).

The steps after this point will be very similar to the ones described before. The patches will be added while the cell relaxed, next, they will be re-adapted and a new contraction occurs. This cycle repeats for one more layer.

model.py

```
342   lmp.command("group del type 8")
343   lmp.command("delete_atoms group del compress no bond yes")
344   lmp.command("group del delete")
345
346   lmp.file("next-layer-1.run")
347
348   active_layer = range(1208,1219+1)
349
350   for ID in desactive:
351     if ID in active_layer:
352       active_layer.remove(ID)
353
354   cell = range(1201,1243+1)
```

model.py

```
356  for ID in desactive:
357    if ID in cell:
358      cell.remove(ID)
```

At the end of the simulation, we save the cell's final state, with the file SAVElastStep.run, we clear the information saved during the simulation and close LAMMPS (953-957). The file SAVElastStep.run will group the subcellular elements from types 7 and 2 (active cell) in line 1 and the command in line 2 will save its state in a file called bedID.runindx.finalCell.dump. In line 3, we set the dump command to the final cell to save the current step of the simulation. These commands will only be run with a run command. Since we do not want to run over more steps, we set a run 0 in line 4.

model.py

```
953  lmp.file("SAVElastStep.run")
954
955  lmp.command("clear")
956
957  lmp.close()
```

SAVElastStep.run

```
1  group finalCell type 7 2
2    dump finalCell finalCell custom 100 ${bedID}.${runindx}.finalCell.dump id x y z radius
3    dump_modify finalCell first yes
4  run 0
```

3.2 First validation

In order to validate our model, we replicated the numerical analysis done in Cunha et al. [4]. We evaluated the cell fate by studying the dependence of the ratio between the final cell volume V and the estimated maximum volume if the cell had not suffered contraction (Figure 3.4). To have these results, we also used 10 different initial positions for the cell, and considered 16 different bed configurations, resulting in 160 samples per aspect ratio. In Cunha et al. [4], it was found through numerical work that when the aspect ratio between the cell and the bed particles is larger, the cell is able to pull the particles and cannot establish stable adhesion. On the other hand, when the aspect ratio is smaller, they observed the strengthening of the bonds, resulting in a higher average volume ratio and a transition between two different regimes of survival rate. While a transition between two regimes is also observed using our model, the regime for cells that are significantly larger than the particles is not well defined. However, it is clear that smaller cells are able to almost completely adhere to the particles, indicating that the cell can enhance its adhesion and grow, similar to the original model.

Our model was developed to adapt the original numerical model using an open-source tool, making it possible for other researchers to utilize it for studying similar systems from a quan-

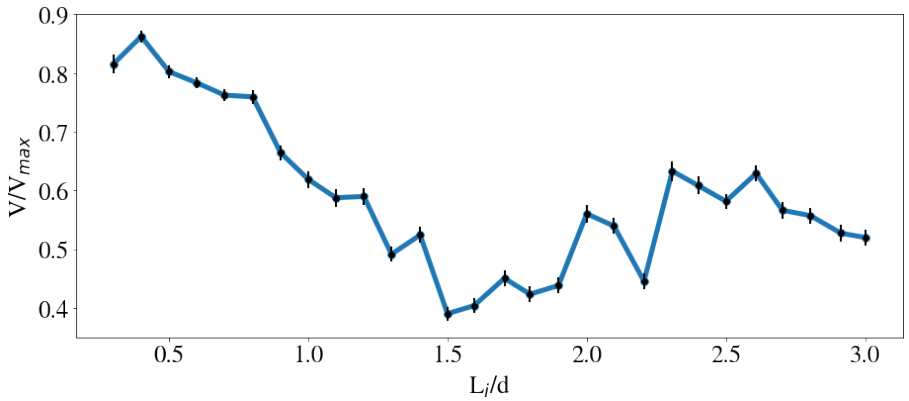

Figure 3.4: Variation of the the fraction of cell volume V/V_{max} with fraction of dimensions L_i/d. The values are averaged over 160 simulations.

titative perspective. The primary differences between the model described in this thesis and the original one are the interaction potentials between the elements of the system, the timescale for the integration and the way they bond with each other. The numerical model described in Cunha et al. [4] includes two elastic constants for the bed particles, one for when they are approaching and the other for when they are moving away, so the relaxation of the granular bed is faster. Besides that, the interactions are also viscous, so they have viscoelastic interaction between entities. Our model only has a single elastic constant, does not consider the proximity of the particles to define the used elastic constant and instead of having viscoelastic interactions, we have elastic interaction in a viscous medium. Besides that, while in the original model, the system is evaluated at each timestep, in our model we evaluated at every 5 steps, due to computational costs. In contrast, the bonds between subcellular elements and bed particles in the original model are implemented at the surface of the elements, while in our model we added patches to the surface of the particles to mimic surface-surface bonds, as LAMMPS only allows for center-center bonds. All these characteristics are possible reasons for the difference in the results since the first one can influence the way microparticles behave and consequently lead to unpredictable behavior of the cell if the granular bed is not correctly relaxed; the second one can also result in different behaviors since we are working in a different timescale; the lasts one can originate a difference in the reinforcement of the adhesion, and consequently a different result in the final structure.

3.3 Future perspective

The main goal for the future of the implemented model is to understand the reasons for the significant differences in the results for higher aspect ratios and the impact of the main differences between the two models. This will help us determine if the definition of the elastic constant for bonds between cells and bed particles or the use of patches is contributing to the discrepancy in the results. The location of the patches may need to be reevaluated as they are the bonding element between subcellular elements and particles. Additionally, the motion of the patches is frozen to prevent rotation, which may affect the way adhesion is modeled. Once these issues are thoroughly investigated and addressed, we intend to study the effect of granular beds containing a mixture of mobile and non-mobile particles on cell survival.

Chapter 4

Metastasis in a spheroid

Tumors are a complex system that involves interactions between cancer cells and their tissue microenvironment. Unexpected behaviors such as resistance to therapeutics, tissue invasion, or treatment failures emerge from the dynamical characteristics of cancer tissues. Mathematical models can be used as "in silico laboratories", where it is possible to explore different hypotheses for this type of behavior in controlled conditions. By using numerical methods, researchers are capable to quantify mechanical, chemical, and biophysical factors affecting tumor cells and their dynamics. In addition to that, this knowledge helps to understand different phases of tumors' lifetime such as cancer initiation, tumor growth, interactions between the tumor and the host tissue, and different steps of metastasis.

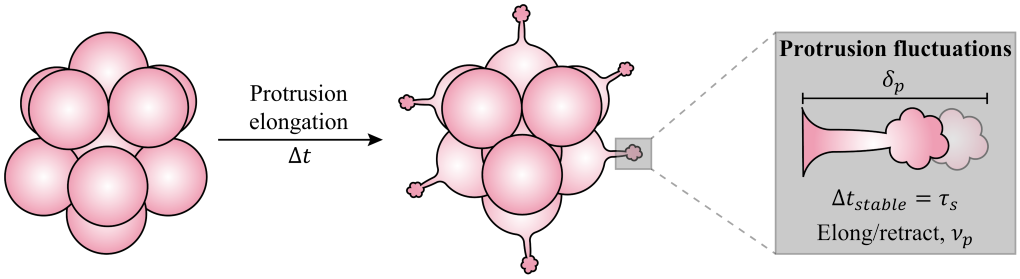

Figure 4.1: Figure inspired by [1]. In order to trigger metastasis, tumorogenic cells in spheroids will elongate to evaluate the surrounding space, and protrusions will appear in Δt. The protrusions fluctuations can be described by several biophysical parameters that enable its evaluation, such as the elongation length, δ_p, the duration of its stabilization, τ_s, and the frequency of protrusions formation, υ_p.

Recently, experiments have been focused on understanding a phenomenon that occurs before a metastasis: the protrusions [1, 7]. Protrusions appear in the first steps of tumorigenesis. They are elongations suffered by a cell, to explore the microenvironment and to apply mechanical forces in the extracellular matrix to start the metastatic process [1, 7]. The invasive cells modulate the protrusions' dynamics to maximize their invasion efficiency [1]. To understand how tumor invasion potential affects the protrusions' dynamics, Caballero et al. [1] used key biophysical parameters (frequency of probing, υ_p, the stability lifetime, τ_s and the probing distance, δ_p) to describe protrusions' activity (Figure 4.1).

To understand the correlation between these biophysical parameters and the invasion potential they evaluated three cell types, with different invasion capabilities in different conditions (normal conditions, applying a chemotherapeutic drug, and misregulating the cytoskeleton machinery involved in force generation). Their results indicate that the phenotype of protrusions

depends on the invasion potential of cells, meaning that cells with a higher metastatic potential have protrusions that elongate more, and are more active. To interpret these results from a quantitative point of view and to highlight the correlation between the chosen biophysical parameters, Caballero et al. [1] defined a new biophysical parameter they called invasion index I_{inv}:

$$I_{inv} = \frac{\langle A_{Pax}/A_{Sph} \rangle}{\langle A_{ECad}/A_{Sph} \rangle} v_p \tau_p, \tag{4.1}$$

where A_{Pax}/A_{Sph} is the fraction of the total spheroid area occupied by paxillin, a protein involved in the interaction between the cells and the extracellular matrix and A_{ECad}/A_{Sph} is the fraction of the total spheroid area occupied by E-cadherin, a protein involved in the interaction between cells. It is a dimensionless parameter that will depend on the protrusions' dynamics and on the cytoskeleton structures.

With this in mind, the purpose of the work reported in this Chapter is to build a model that represents a tumor spheroid based on the described experimental setup, which allows for the evaluation of parameters correlated with protrusions' dynamics from a theoretical and quantitative point of view.

4.1 Model and Methods

We consider N spheres, with radius r, in a viscous medium, organized in a 3D compact spheroid to represent a tumor (Figure 4.2 (a)). In this model, each sphere represents a cell. The movement of the cell is described by Langevin dynamics, and it is influenced by cells' velocity, the interaction between cells, and the force of protrusion formation. The interaction between cells is described by an elastic force connecting neighbor spheres, while the interaction between the cells and the medium is given by the force of protrusion formation and viscosity.

We control the force of protrusion formation, $\vec{F}_p$, the elastic constant of the springs K that connect the neighbor cells, and the viscosity of the medium, μ. Regarding the concepts defined at the beginning of this Chapter, the force of protrusion magnitude is indicative of how invasive a cell can be. Furthermore, the springs' elastic constant correlates with the quantity of E-cadherin, while the force of the protrusion correlates with the quantity of paxillin.

The force of protrusion formation is applied in the outward direction to a random cell (Figure 4.2 (b)). If the cell is in the spheroid periphery, it will leave the spheroid when this force exceeds the binding force between cells during the protrusion lifetime, and the cell starts to be an invasive cell. Otherwise, it will stay in the spheroid. The protrusion will be represented by the applied force of protrusion. Its effect depends on its lifetime, the medium's viscosity, and cell-cell interaction.

In the simpler case, each periphery cell has 6 neighbors, and only the movement of the cell to which the force of protrusion is applied is considered. The Langevin dynamics equation that describes this situation is represented as follows:

$$m\ddot{\vec{y}}(t) = -\mu\dot{\vec{y}}(t) + \sum \vec{F}, \tag{4.2}$$

where m is the cell mass, $\vec{y}$ is its displacement, μ is the Stokes coefficient and $\sum \vec{F}$ is the total force applied in the cell. The total force applied in the cell is given by the sum of the spring

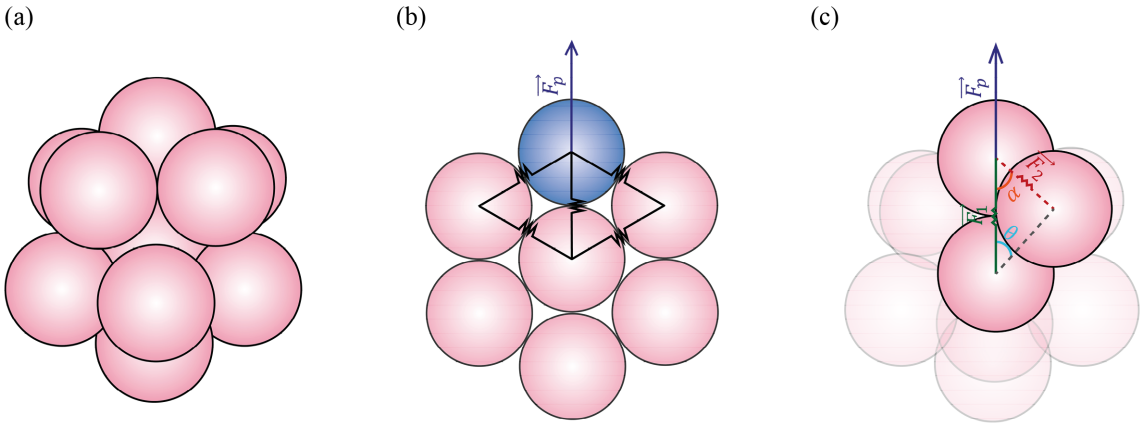

Figure 4.2: Representation of the simpler case of a regular spheroid. At left, a spheroid with 13 cells. At the center is the representation of the springs that connect the cells. The blue sphere is considered the active cell, where the force of protrusion is applied. At right, it is the representation of the forces considered to study the movement of the active cell.

forces that pull it to its 5 neighbors in the periphery, $\vec{F}_2$, the spring force that pulls it to the central cell, $\vec{F}_1$, and the force of protrusion $\vec{F}_p$ (Equation 4.3).

$$\sum \vec{F} = -(5\vec{F}_2 + \vec{F}_1) + \vec{F}_p, \tag{4.3}$$

where $\vec{F}_1 = Ky\vec{u}_y$ and $\vec{F}_2 = K(\Delta r)(cos\alpha)\vec{u}_y$, with K being the spring constant, y the magnitude of the elongation suffered by the springs associated with $\vec{F}_1$ and Δr the magnitude of the elongation suffered by the springs associated with $\vec{F}_2$. The angle α is represented in Figure 4.2 (c). The value Δr will depend on Δy accordingly with:

$$\Delta r(y) = \sqrt{(\delta r_0 + y)^2 + (\delta r_0)^2 - 2(\delta r_0 + y)(\delta r_0)cos\theta} - r_0, \tag{4.4}$$

where δ is $sin\frac{2\pi}{5}$, r_0 is the distance between periphery cells and the central cell, and θ is the angle represented in Figure 4.2 (c).

4.1.1 Configuration of the Spheroids

We have two types of spheroids in this study. The first one is a spheroid with a regular shape. In order to achieve the proper configuration, we considered the Cambridge Cluster Database for Lenard Jones Cluster Global Minima [1], to define the coordinates of each element of the spheroid. In the simulations conducted, we consider reduced units, standard default as Lennard Jones units in LAMMPS. LAMMPS sets all fundamental quantities (mass m, distance σ, energy ϵ, and the Boltzmann constant k_B) equal to 1, and all quantities are given in reference to these reduced units. This is how the quantities will be presented throughout this Chapter. The central sphere is always fixed to its position and harmonic bonds are created between any two cells that have $2r$ of the distance between their center of mass.

The other type of spheroids have an amorphous configuration, that would be more comparable with the real case. The approach to this case was different. One sphere is fixed in the center of a box and all the other spheres were randomly positioned in it. A force was applied toward

[1] The Cambridge Cluster Database, D. J. Wales, J. P. K. Doye, A. Dullweber, M. P. Hodges, F. Y. Naumkin F. Calvo, J. Hernández-Rojas and T. F. Middleton, URL `http://www-wales.ch.cam.ac.uk/CCD.html`

the center of the box, in order to form a compact spheroid. When the kinetic energy of the system is lower than 10^{-10}, this force is deactivated, and a harmonic bond is created between any two cells that have $2r$ of the distance between their center of mass.

To calculate the number of neighbors each active cell has, we calculate the distance between all the cells and the active cell, and we count the number of cells for which the distance is equal to or less than $2r$. To define if a cell is in the periphery of the spheroid or not, we calculate the distance between the cells and the spheroid's center and found the maximum distance (D_{max}). If the cell is in the range between $D_{max} - 2r$ and D_{max}, we say that this cell can be found in the periphery of the spheroid.

4.1.2 Cells interactions

In order to avoid the overlap between spheres, it was necessary to add an interaction between them. Therefore, the interactions between two cells are given by the Hooke potential, without the tangential part, defined as:

$$F_{Hooke} = k_n \delta_{ij} \vec{n} - \eta_{n0} a m_{eff} \vec{v}_{n,rel}, \tag{4.5}$$

where k_n is the spring stiffness, $\delta_{ij} = R_i + R_j - ||\vec{r}_{ij}||$ is the particle overlap, R_i, R_j are the particle radii, $\vec{r}_{ij} = \vec{r}_i - \vec{r}_j$ is the vector that separates the two center points of the particles, $\vec{n} = \frac{\vec{r}_{ij}}{||\vec{r}_{ij}||}$, η_{n0} is the damping prefactor, a is the radius of contact, given by $a = \sqrt{R\delta}$, $m_{eff} = m_i m_j/(m_i + m_j)$ and $\vec{v}_{n,rel} = (\vec{v}_j - \vec{v}_i) \cdot \vec{n}$.

In order to simulate the protrusion's formation a random cell is activated with a force F_p, toward the periphery of the spheroid.

4.2 Results

4.2.1 Numerical and Analytical Studies

We performed a numerical analysis of the simpler regular spheroid, with 13 cells, where only the active cell's movement is considered. The next results arise from the numerical solution of Equation 4.2.

4.2.1.1 Maximum displacement and time needed to steady-state

One of the relevant parameters that result from the numerical solution of Equation 4.2, is the evolution of the active cell's steady-state displacement Δy_{stable} depending on system conditions. In Figure 4.3 (a) we present the variation of Δy_{stable} with the strength of protrusion F_p, taking into consideration different system's Stokes coefficients and spring constants. The main characteristic shared by all the cases is that Δy_{stable} varies non-linearly with the strength of the force of protrusion $\vec{F}_p$. Possibly, this non-linear behavior is due to the dependence of Δr on Δy, which will imply that this variation will be linear only when $\Delta r \approx \Delta y$ ($\Delta r \gg 1.0$). For cells with the same spring constant K, but different viscosity μ, Δy_{stable} will depend on F_p in a similar manner, having the same curve. This means that Δy_{stable} will have a dependence on K, besides the one it has on the applied F_p. Furthermore, higher K will lead to stronger resistance to cell movement, since the bonds between cells will be stronger, and due to this, the achieved displacement will be lower. In Figure 4.3 (b) we represent the dependence of Δy_{stable}

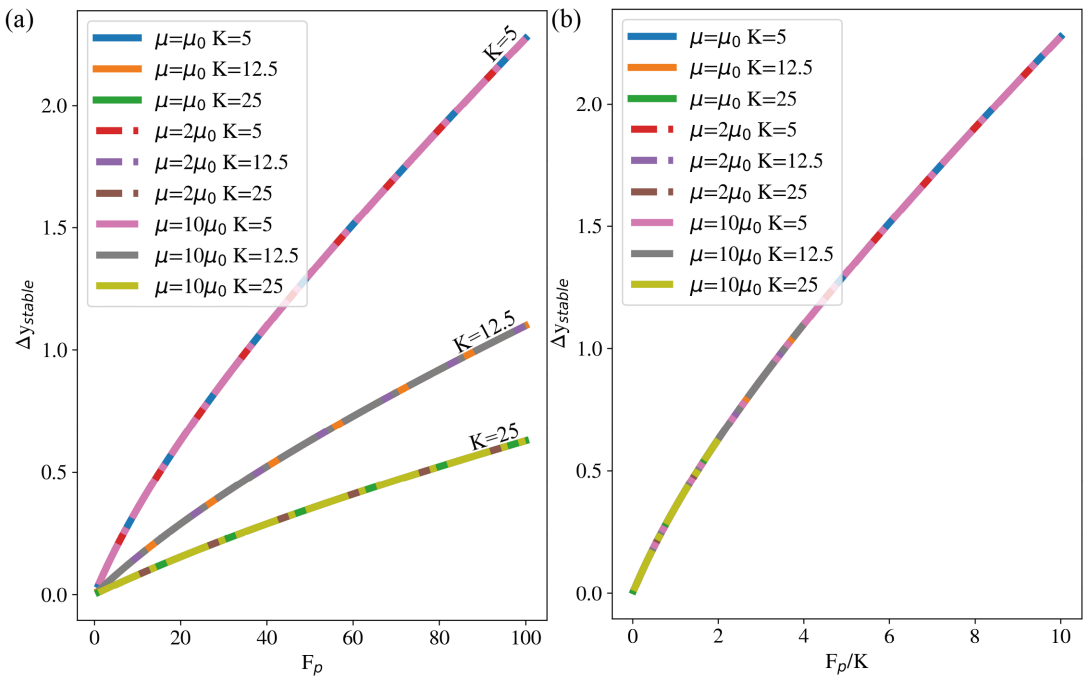

Figure 4.3: (a) Variation of steady state displacement with force of protrusion for several pair of viscosity coefficient μ and spring constant K ($\mu_0 = 235\ \sqrt{m\epsilon/\sigma^2}$). (b) Variation of steady state displacement with F_p/K for several pair of viscosity coefficient μ and spring constant K.

on F_p/K to show that even in systems with different characteristics, the achieved displacement will depend only on the force of protrusion and on how strong bonds between cells are. Since the F_p can correlate with the interaction between the cells and the extracellular matrix, K can correlate with the interaction between cells, and the displacement increases with the constant F_p/K, it is possible to establish a parallelism between F_p/K and $\frac{\langle A_{Pax}/A_{Sph}\rangle}{\langle A_{ECad}/A_{Sph}\rangle}$ in Equation 4.1.

The time it takes to achieve a stable displacement is also important. Figure 4.4 represents the variation of time needed to achieve Δy_{stable} (T_s) with F_p. We observe that T_s depends on μ and K. Lower μ will imply less time to achieve the desired displacement since the medium will offer less resistance to the movement. For example, in the cases where $K = 5\ \epsilon/\sigma$, Δy_{stable} is equal for every μ (Figure 4.3 (a)), however, it will take more time to a cell in a system with a higher viscosity to achieve it. Besides that, T_s will depend on K too. Cells that bond to their neighbors with a higher elastic constant will achieve their stable displacement quicker than cells that are poorly bonded to their neighbors. This behavior stems from the fact that Δy_{stable} is lower for these cases, as stated before, so the cell will need less time to reach the wanted position.

In Figure 4.5 we represent Δy_{stable} variation with T_s in adimensional units. For this, we considered the conclusions from the analysis done before. The cell maximum displacement depends on cell-cell interaction (K) and cell-extracellular matrix interaction (F_p), while the time to reach this displacement depends on cell-cell interactions (K) and on the medium viscosity (μ).

4.2.1.2 Damping regimes

In order to understand the different regimes in our model, we performed a numerical analysis of the effective force of the springs, the effective elastic constant present in the cell-cell interactions, and the damping ratio, considering the simpler case, previously mentioned.

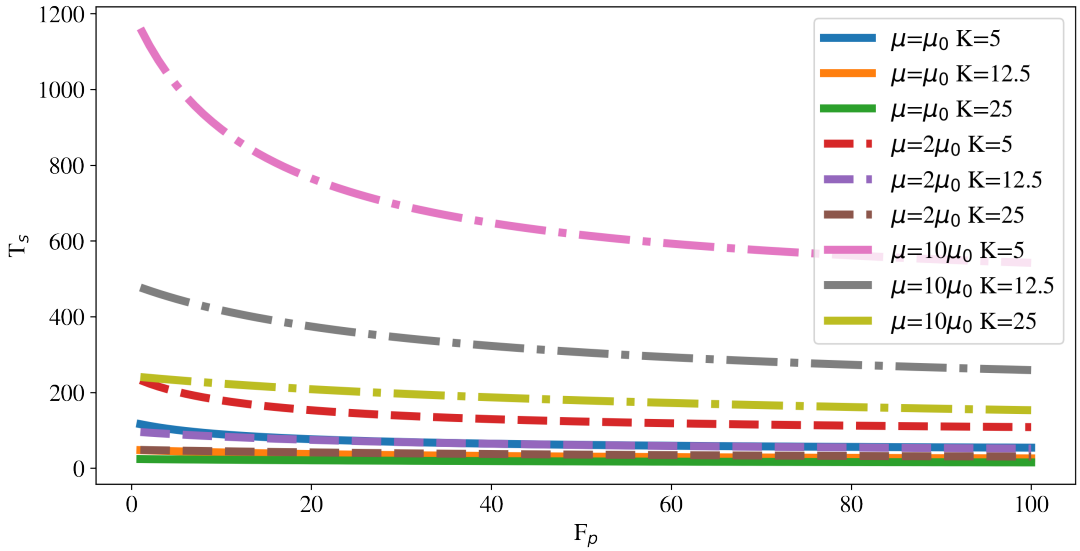

Figure 4.4: Analytical variation of time to reach the steady state with the force of protrusion F_p for several pairs of viscosity coefficient μ and spring constant K ($\mu_0 = 235\ \sqrt{m\epsilon/\sigma^2}$).

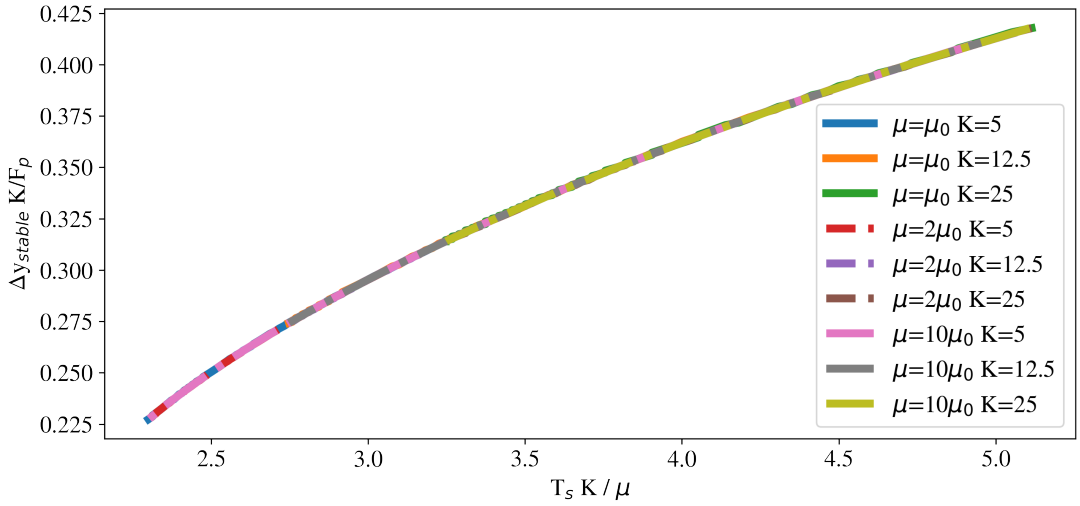

Figure 4.5: Analytical variation of $\Delta y_{stable} K/F_p$ with $T_s K/\mu$ for several pair of viscosity coefficient μ and spring constant K. It represents the dependence of the maximum displacement suffered by the active cell on the time needed to reach it, in adimensional units.

First, we focused on the springs' effective force acting on the active cell, given by

$$F_{eff} = -(5F_2 + F_1) = 5K\Delta r(\Delta y)\cos\alpha + K\Delta y. \tag{4.6}$$

In Figure 4.6 (a) we have the effective force variation with the displacement suffered by the active cell in a system where $K = 1\ \epsilon/\sigma$ and $\mu = 42\ \sqrt{m\epsilon/\sigma^2}$. Here, it is possible to observe a transition between two regimes in F_{eff}, where each one has a characteristic effective elastic constant. The first regime is related to small deformations in the spheroid, where Δr is much different than Δy, at the beginning of the displacement. Focusing on the analytical analysis we can consider that $\Delta y \approx 0$, so we will have a slope equal to $5K\cos\alpha$. We have $\cos\alpha \approx 0.52$, so the slope will be nearly 2.6. The other regime is related to larger deformations in the spheroid, where $\Delta r \approx \Delta y$. In this case, $\cos\alpha \approx 1$. If we replace Δr by Δy in F_{eff}, we will have $F_{eff} = 5K\Delta y + K\Delta y = 6.0\Delta y$,

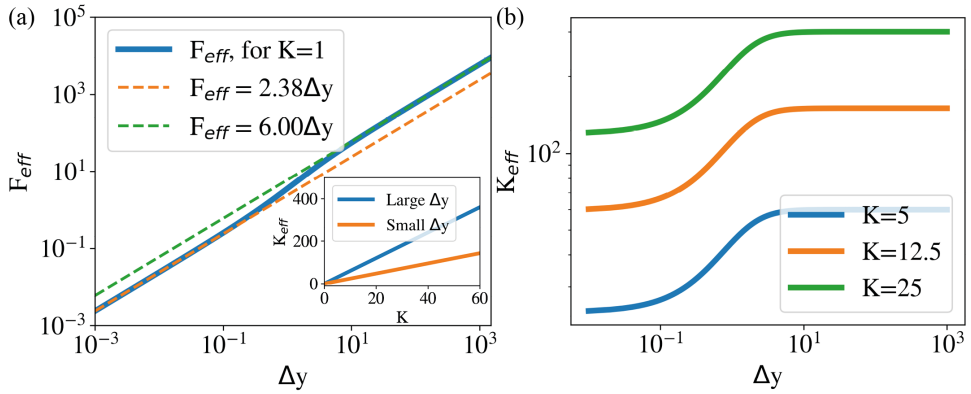

Figure 4.6: (a) Effective force dependence of cell displacement. In the inset, there is the dependence of the effective spring constant on K. (b) Effective spring constant dependence of cell displacement, for three different K (10,25,50). In gray, the transition between small and large Δy is highlighted.

as was stated in the numerical analysis. Another relevant parameter is K_{eff}. In the inset of Figure 4.6 (a) we can see the graphical representation of the numerical variation of K_{eff} with K for the two present regimes, which can be inferred from the previous discussion. So, for small displacements K_{eff} will depend on K accordingly with $K_{eff} = 2.38K$. On the other hand, for large displacements, it will depend on K accordingly with $K_{eff} = 6.00K$.

We performed a numerical analysis of K_{eff} to observe its variation with Δy. We considered that $F_{eff} = K_{eff}\Delta y$, so we calculated the variation of the F_{eff} slope in function of Δy, using the expression $K_{eff} = \frac{F_{eff}(\Delta y) - F_{eff}(\Delta y + 0.001)}{\Delta y - (\Delta y + 0.001)}$, and represented it in Figure 4.6 (b), for three different K. We observe a transition in this variation, similar to the one observed in F_{eff}. Besides that, the described regimes maintain their described behavior. For larger displacements, there is a baseline equal to $6K$, after a transition. The position of the transition will not depend on K, only depends on Δy, as stated before.

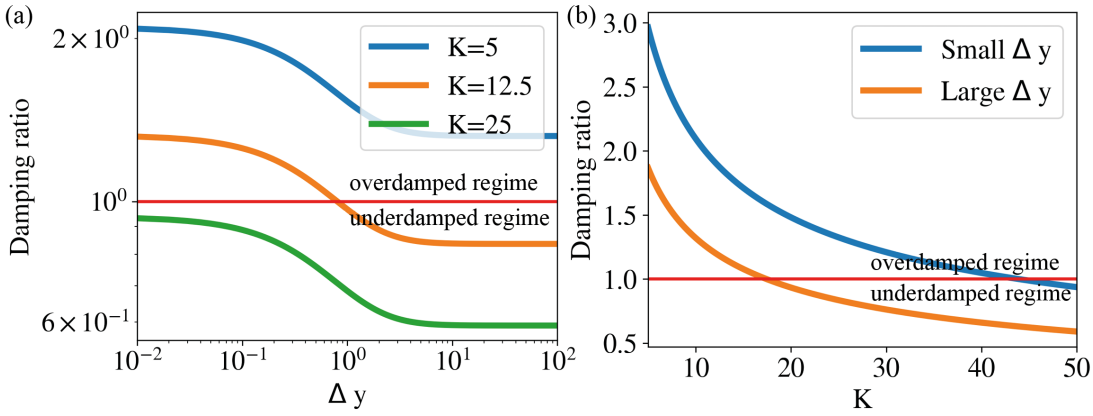

Figure 4.7: (a) Damping ratio dependence on Δy, for three different spring constants (10,25,50). In red is the threshold that delimits the transition between the overdamped regime and the underdamped regime. (b) Damping ratio dependence on K, for small and large Δy. In red is the threshold that delimits the transition between the overdamped regime and the underdamped regime.

Based on the damped harmonic oscillation equations, we studied the damping ratio $\zeta = \frac{\gamma}{2\sqrt{K_{eff}(\Delta y)m}}$ dependence on Δy and on K, to understand the different damped regimes of the system. When $\zeta < 1$, we are in the underdamped regime; otherwise, when $\zeta > 1$ the system is in the overdamped regime. In Figure 4.7 (a) we represented the damping ratio's variation with Δy,

for three different K. The red line represents the threshold that divides the two different regimes ($\zeta = 1$). When depending on Δy (Figure 4.7 (a)), the damping ratio suffers a transition, similar to the one observed for K_{eff}, since ζ depends on $\sqrt{K_{eff}}$. However, in the case of the damping ratio, the spring constant K controls the transition from an overdamped to an underdamped regime. In this case, a larger K implies that the system is in an underdamped regime, while a system where the K is lower is in an overdamped regime. Intermediate values of spring constants indicate that the active cell starts moving in the overdamped regime, but if it achieves a large displacement, will behave accordingly with the underdamped regime.

Numerically, we were able to understand the variation of the damping ratio depending on K for large and small Δy. This way it was possible to find a range of K that has both overdamped and underdamped regimes, depending on Δy. Its representation is in Figure 4.7 (b) and the red line represents the threshold that divides the two different damping regimes ($\zeta = 1$). Between $K \approx 18$ and $K \approx 45$, depending on the suffered deformation, the system will suffer a transition from the overdamped regime to the underdamped regime.

4.2.2 Effect of the spheroid's size

To evaluate the collective effect of the cells present in the spheroid and correlate the variation of the parameters discussed before with the spheroid's size we integrate the equations of motion, using a velocity Verlet scheme implemented in Large-scale Atomic/Molecular Massively Parallel Simulator [53] (LAMMPS). In the simulations, we used two different configurations, regular and amorphous, as referred to in Section 4.1.1. For the regular configuration, we considered spheroids with sizes from 10 to 150 cells, while for the amorphous configuration we considered spheroids with sizes from 10 to 215 cells. For the next results, we considered $\mu = 66$, $K = 5$, and $F_p = 40$. For both cases, we evaluated the steady-state displacement of active cells (Δy_{stable}), the number of neighbors each active cell has, and the time needed to achieve Δy_{stable} (T_{stable}), all depending on the number of cells in the spheroid, N.

The results for spheroids with regular configurations are in Figure 4.8. Figure 4.8 (a) shows a non-monotonic dependence of Δy_{stable} on the spheroid size. After $N \approx 55$ the displacement starts to be almost constant, meaning that the spheroid's size has no effect after this one. The number of neighbors each active cell has (Figure 4.8 (b)) is non-monotonic too, and a higher number of neighbors corresponds to a lower displacement. The variations in these values are due to the geometry of the clusters since regular clusters do not mean that the spheroids have a perfect spherical shape, in fact, they are polyhedrons. T_{stable} in function of the spheroid size is represented in Figure 4.8 (c). Its behavior is similar to the behavior of Δy_{stable}, their minimums and maximums correspond, meaning that there is a dependence between them or they have common dependences on the studied parameters. A larger displacement corresponds to more time to achieve it. The exterior active cells (in orange) behaved as expected: they have larger displacements, fewer neighbors and they need less time to achieve stable displacement when compared with cells in the interior of the spheroid. In the regular case, we can take into account the Mackay structural icosahedron magic numbers. In the sizes we studied, these structures have 13, 55, and 147 cells. For these cases, we can observe a minimum in the stable displacement. Even if the number of neighbors is not large, the geometry of these structures is more stable and makes it difficult to deform the structure.

The results for spheroids with amorphous configurations are in Figure 4.9. Figure 4.9 (a)

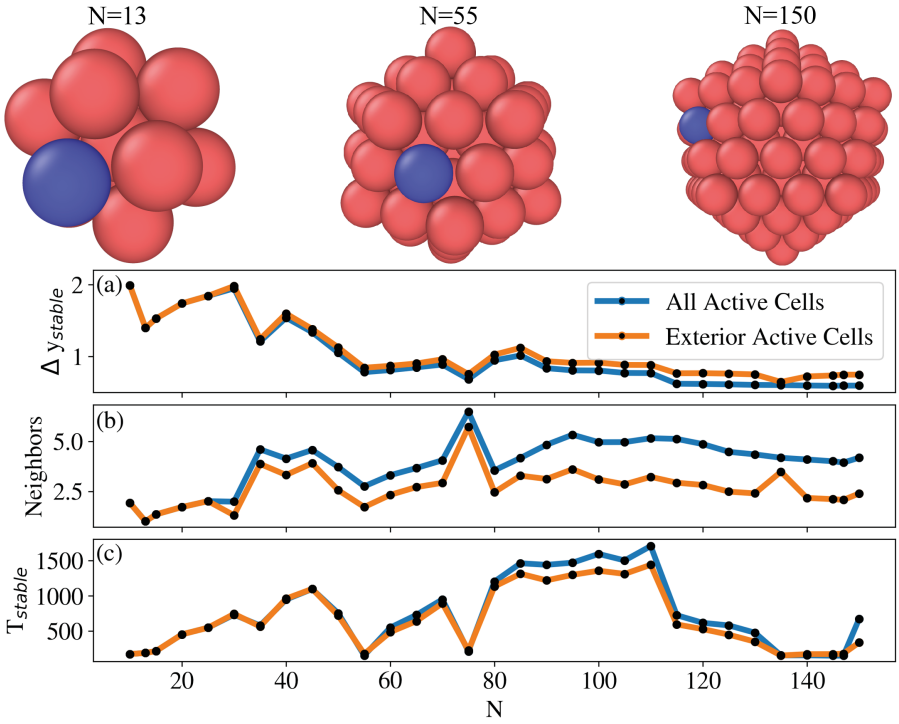

Figure 4.8: Variation of steady state displacement and a number of neighbors in function of the spheroid's size for regular configuration. (a) Variation of the steady-state displacement with the number of cells in the spheroid. (b) Variation of the number of neighbors each active cell has with the number of cells in the spheroid. In blue all the active cells were taken into account, in orange, only the active cells in the periphery of the spheroid were considered. The values are averaged over 1000 simulations. The error bars are the same size as the points.

shows a non-monotonic dependence of Δy_{stable} on the spheroid size. For smaller spheroids, Δy_{stable} tends to decrease. It goes accordingly with the number of neighbors of the active cell (Figure 4.9 (b)) that increases with the number of cells per spheroid. Since the active cell has more neighbors, the effective force attaching it to the spheroid is larger, so the cell will feel more resistance when it tries to move away from the spheroid. After $N \approx 55$ the displacement starts to be constant, so the size will have no effect on the behavior of the active cell. The number of neighbors will not change after $N \approx 20$, since the spheroids are very compact, which maximizes the number of neighbors. On the other hand, T_{stable} (Figure 4.9 (c)) has a non-monotonic behavior too, but after $N \approx 55$ keeps increasing. This phenomenon can be explained by the fact that even if the number of cell's neighbors is constant, these neighbors are connected to other cells. If we observe the phenomenon as a chain of springs since all cells are connected with their neighbors, every cell in the chain will contribute to the effective force felt by the active cell, so there will be needed more time to achieve exactly the same displacement. Besides that, we separate the collected data into two sets - a set with all the cells we activated (in blue) and the other one with all the exterior active cells (in orange). The behavior for both sets was similar, however, as expected, exterior cells achieve larger displacements, have fewer neighbors, and take less time to achieve stable displacement, when compared with cells from the interior of the spheroid.

The behavior of regular and amorphous configurations is very similar. However, regular configurations seem to not have a continuous dependence on the other factors, since the geometry of the spheroid will have bigger effects on the final results of activating a cell. Nonetheless, in both configurations, the displacements seem to achieve a constant value after $N \approx 55$. This can

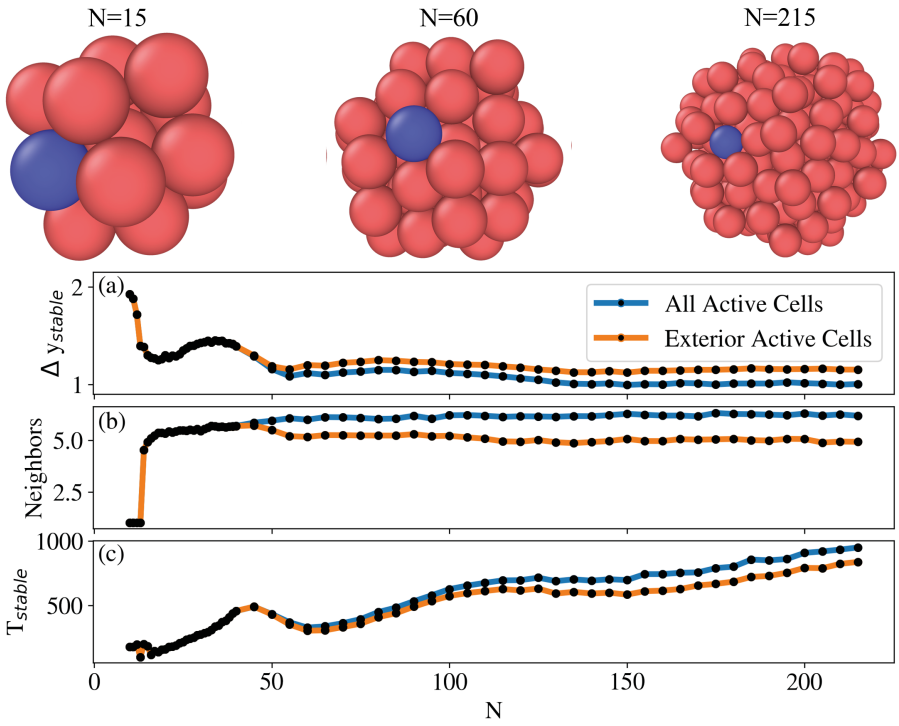

Figure 4.9: Variation of steady state displacement and the number of neighbors in the function of the spheroid's size for amorphous configuration. (a) Variation of the steady-state displacement with the number of cells in the spheroid. (b) Variation of the number of neighbors each active cell has with the number of cells in the spheroid. In blue all the active cells were taken into account, in orange, only the active cells in the periphery of the spheroid were considered. The values are averaged over 2000 simulations. The error bars are the same size as the points.

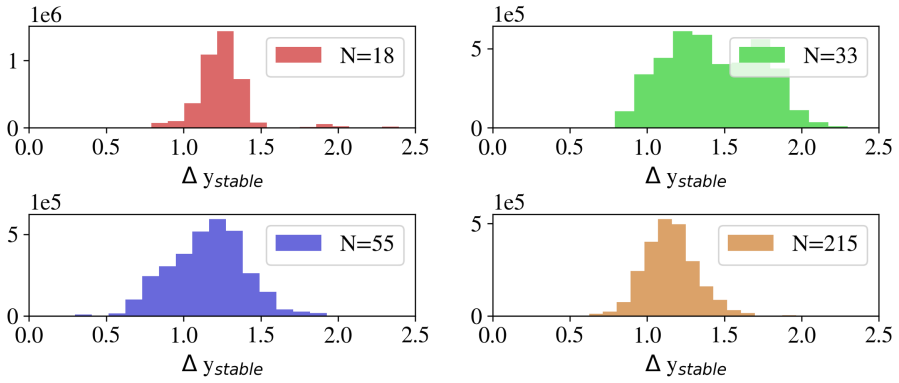

Figure 4.10: Histogram of the distribution of the maximum displacement for all the activated cells in the exterior of the spheroid in amorphous configuration. (a) N=18 (b) N=33 (c) N=55 (d) N=215

imply that only spheroids with more than 55 cells can be considered as behaving like a tissue since only at that moment the cells start to have a more evident collective behavior.

To help us understanding the minima and maxima in the previous analysis, we evaluated the distribution of Δy_{stable} (Figure 4.10) and T_{stable} (Figure 4.11) for that specific sizes, only for exterior active cells, since they are the ones that can provoke metastasis. Besides that, these distributions are only for amorphous configurations. Regarding Δy_{stable}, the minimum is at $N = 18$, and the maximum is at $N = 33$. What we can observe, is that when comparing both distributions, the one for the minimum is more narrow around the peak, while for the maximum it does not happen. There is a wider distribution for $N = 33$, and there are two peaks. When we look for the distribution for $N = 55$ and $N = 215$, it becomes more narrowed around the Δy_{stable}

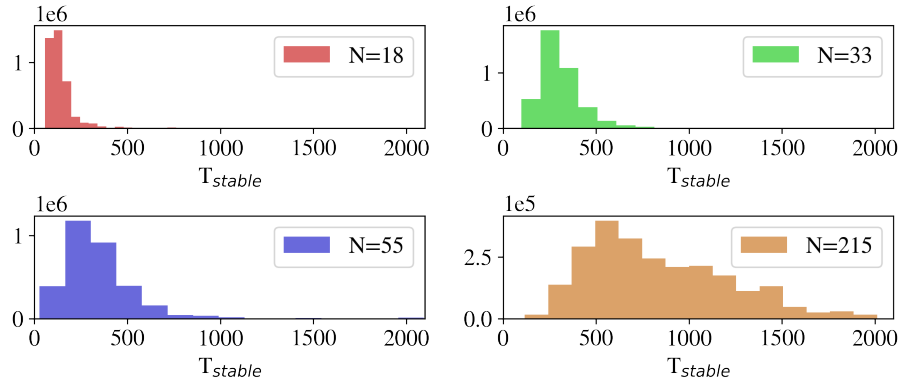

Figure 4.11: Histogram of the distribution of the time needed to reach the maximum displacement for all the activated cells in the exterior of the spheroid in amorphous configuration. (a) N=18 (b) N=33 (c) N=55 (d) N=215

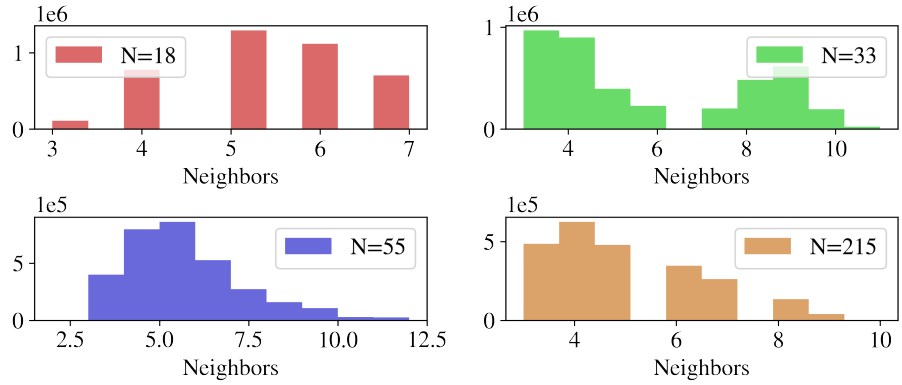

Figure 4.12: Histogram of the distribution of the number of neighbors for all the activated cells in the exterior of the spheroid in amorphous configuration. (a) N=18 (b) N=33 (c) N=55 (d) N=215

peak again. T_{stable} distributions become wider to the right with increased sizes of the spheroid, reinforcing the increase in time even when Δy_{stable} is constant.

We evaluated the distribution of neighbors for each active cell in the periphery of the spheroids for the same structures (Figure 4.12), in order to understand the behavior in the maximum at $N = 33$. The structures have only one peak in the number of neighbors, except $N = 33$ which has a peak for less than 4 neighbors per cell and another for more than 8 neighbors. Having fewer bonds per active cell will imply that the effective force applied to cells will have a lower magnitude, so the resistance to the movement will be smaller and the cell will be able to get away from the spheroid more easily.

In order to understand the conclusion taken before about the effect of F_p and K, on the cell's behavior, we performed simulations around the experimental range of spheroids' size. In Figure 4.13 we compiled the dependence on N of Δy_{stable}, $\Delta y_{stable}K/F_p$, T_{stable} and $T_{stable}K/\mu$, for several pairs of F_p and K. In fact, a higher F_p promotes a larger displacement, and systems with the same F_p/K will have similar displacements, as was noted in Section 4.2.1. On the other hand, a higher K will promote a lower displacement, as noted before. Regarding T_{stable}, lower K will imply that the cell will need to take less time to achieve a stable displacement since this displacement will be smaller when compared with systems with higher K. Besides that, lower F_p will imply more time to achieve the stable displacement, since the force that goes against the effective elastic force is smaller, it will need more effort to achieve a stable position.

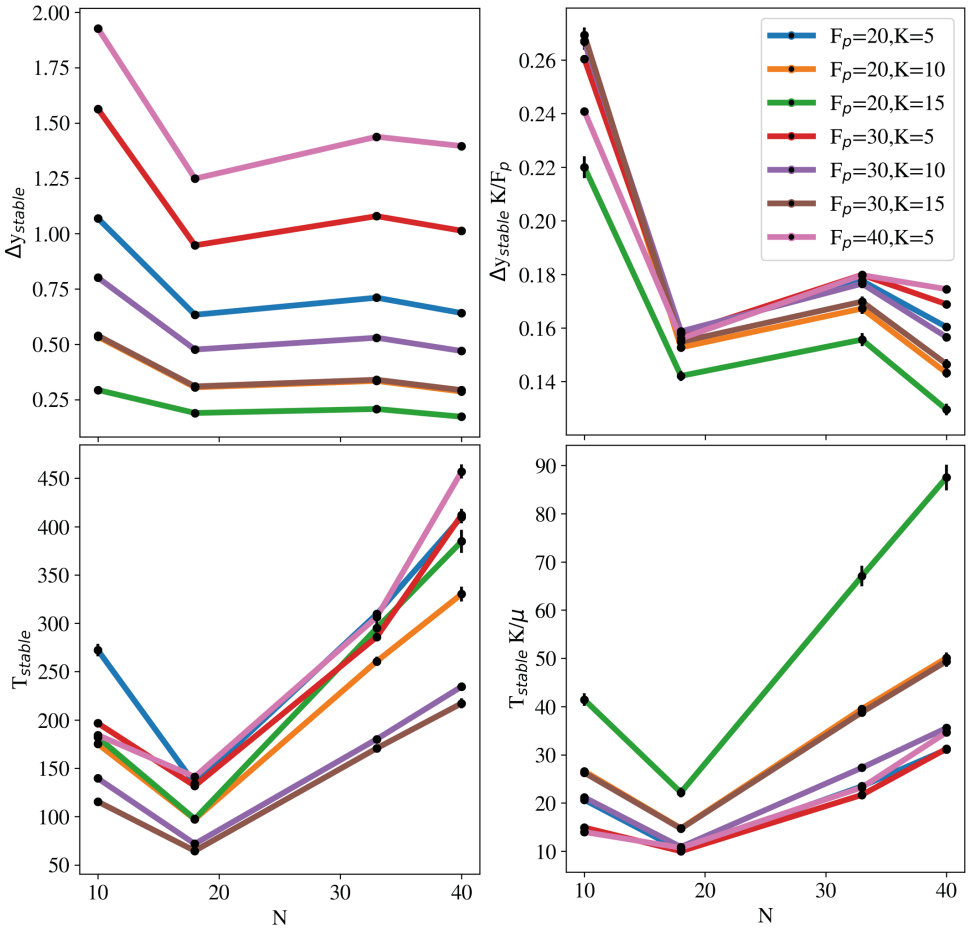

Figure 4.13: Variation of Δy_{stable}, $\Delta y_{stable}K/F_p$, T_{stable} and $T_{stable}K/\mu$ in function of the spheroid's size for amorphous configuration in the experimental range, with different forces of protrusion and different elastic constants. The values are averaged over 1000 simulations. The error bars are the same size as the points.

We used the adimensional equivalences to the evaluated values, to understand if we would get the same phenomenon - overlapping curves. It does not happen for $\Delta y_{stable}K/F_p$, nor for $T_{stable}K/\mu$. One possible justification is the fact that in the numerical analysis, only the active cell's movement is considered. In this case, we take into account every contribution of each cell, and for that reason, probably the behavior will not be exactly the same, but the general rules can be taken into account.

4.3 Future perspective

In the future, we pretend to evaluate how we can establish a parallelism between our parameters and the parameters studied in Callabero et al. [1] work, specifically, the parameters in Equation 4.1. Our interpretation is that F_p/K can be used as a correlated term to $\frac{\langle A_{Pax}/A_{Sph}\rangle}{\langle A_{ECad}/A_{Sph}\rangle}$. We believe it because F_p translates the interaction between cells and extracellular matrix and paxillin is a protein involved in this interaction, while K translates the bonds between cells, and E-cadherin promotes this interaction. The frequency of protrusion formation and the duration of its stabilization are still hard to find in our model since we do not explicitly define the protrusions. Instead, its behavior is translated into a force of protrusion.

Chapter 5

Conclusion

In this thesis, we developed two in silico models based on experimental results. The first model was based on Cunha et al. [4] research about the influence of a mobile granular bed with a range of particle diameters in cell response. The second model was based on Caballero et al. [1] research about the correlation of several biophysical parameters related to the protrusion activity present in metastasis with the cell's invasion potential.

The goal in Chapter 3 was to develop a numerical model with an open-source tool to illustrate the interaction between a deformable cell and a granular bed. The cell would fall into a relaxed granular bed and suffer cycles of stretching and contraction, to mimic its growth and a phenomenon called mechanotransduction. Due to the complexity of the modeled phenomena, we needed to compile LAMMPS, the open-source tool, as a Python2.7 library, in order to do some numerical approaches with Python tools. The several phases of the cell's behavior were modeled. It includes the creation of patches in the contact points between subcellular elements and granular particles (to simulate the integrins recruitment in adhesion), the stretching of the cell and its contraction, and also the granular bed relaxation and the cell's initial configuration. However, after the first validation, we understood that our cell fate does not correspond to the results presented in Cunha et al. [4] work. Nonetheless, we can observe a transition between the two regimes, but only the results for cells much smaller than the granular bed particles are similar to their results. This means that in these cases, the cell can reinforce its adhesion and spread over the particles. We pretend to improve our model and study the effect of granular beds composed of a mix of mobile particles and non-mobile particles on the cell's survival.

The purpose of the model described in Chapter 4 was to represent a tumor spheroid, where cells will suffer a force of protrusion formation. We started with an analytical analysis of our system. With it, we found that the maximum displacement of a cell has a non-linear behavior and will depend on the force of protrusion, and on the spring constant. The higher the spring constant, the stronger will be the resistance to the movement, and the lower will be the achieved displacement. On the other hand, an increase in the force of protrusion will increase how long the cell can go. Besides that, we observed that the time needed to stabilize the cell movement will depend on the spring constant and on the viscosity of the medium. A lower medium's viscosity means less time to achieve a stable displacement. However, a larger elastic constant will also lead to the quicker achievement of the stable position, since the cells will have higher stability in their connections. With this study, we found how to represent the stable displacement and the time needed for achieving it in dimensional units. We also studied the damping regimes of our model and learned that they will depend on the effective force of the system and consequently

on the effective elastic constant and on their achieved displacement.

To evaluate the effect of the spheroid's size on the cell's behavior we performed numerical simulations for our model. For this, we studied regular and amorphous spheroid configurations. For both configurations, we found a non-monotonic dependence of the stable displacement on the spheroid size. However, the amorphous configuration seems to have a continuous dependence on the spheroid's size. It shows to be constant after $N \approx 55$, which can signify that spheroids with less than 55 cells cannot be considered as behaving like tissues since only after this size the aggregate starts to show a clearer collective behavior. The number of neighbors will also influence the achieved displacement. The time needed to achieve a stable displacement seems to keep increasing, which can be explained by the collective behavior of the aggregate since the effective force felt by the active cell will increase with the size of the spheroid. Besides that, we evaluated the adimensional units of the studies performed in the analytical part of this work, to observe if we would get overlapping curves as before, but it did not happen. It can be because in the simulations all cell's movements contribute to the effective force sensed by the active cell, which does not happen in the analytical analysis. Notwithstanding the fact that we were able to establish some correlations between our parameters and the parameters defined in Caballero et al. [1], we pretend to improve our model in order to reproduce the results provided by Equation 4.1.

Bibliography

[1] D. Caballero, V. Brancato, A. C. Lima, C. M. Abreu, N. M. Neves, V. M. Correlo, J. M. Oliveira, R. L. Reis, and S. C. Kundu, "Tumor-Associated Protrusion Fluctuations as a Signature of Cancer Invasiveness," *Advanced Biology*, vol. 5, no. 9, p. 2101019, 2021.

[2] C. S. Dias, C. A. Custodio, G. C. Antunes, M. M. Telo Da Gama, J. F. Mano, and N. A. Araujo, "Modeling of cell-mediated self-assembled colloidal scaffolds," *ACS Applied Materials and Interfaces*, vol. 12, no. 43, pp. 48321–48328, 2020.

[3] U. Nagarajan, G. Beaune, A. Y. Lam, D. Gonzalez-Rodriguez, F. M. Winnik, and F. Brochard-Wyart, "Inert-living matter, when cells and beads play together," *Communications Physics*, vol. 4, no. 1, p. 2, 2021.

[4] A. F. Cunha, A. F. V. Matias, C. S. Dias, M. B. Oliveira, N. A. M. Araújo, and J. F. Mano, "Cell Response in Free-Packed Granular Systems," *ACS Applied Materials Interfaces*, vol. 14, pp. 40469–40480, sep 2022.

[5] L. Adenis, E. Gontran, C. Deroulers, B. Grammaticos, M. Juchaux, O. Seksek, and M. Badoual, "Experimental and modeling study of the formation of cell aggregates with differential substrate adhesion," *PLoS ONE*, vol. 15, p. e0222371, 2020.

[6] A. R. A. Anderson, "A hybrid mathematical model of solid tumour invasion: the importance of cell adhesion," *Mathematical Medicine and Biology: A Journal of the IMA*, vol. 22, pp. 163–186, 6 2005.

[7] W. Kang, J. Ferruzzi, C.-P. Spatarelu, Y. L. Han, Y. Sharma, S. A. Koehler, J. A. Mitchel, A. Khan, J. P. Butler, D. Roblyer, M. H. Zaman, J.-A. Park, M. Guo, Z. Chen, A. F. Pegoraro, and J. J. Fredberg, "A novel jamming phase diagram links tumor invasion to non-equilibrium phase separation," *iScience*, vol. 24, no. 11, p. 103252, 2021.

[8] D. Bi, X. Yang, M. C. Marchetti, and M. L. Manning, "Motility-driven glass and jamming transitions in biological tissues," *Physical Review X*, vol. 6, no. 2, p. 021011, 2016.

[9] A. G. Fletcher, J. M. Osborne, P. K. Maini, and D. J. Gavaghan, "Implementing vertex dynamics models of cell populations in biology within a consistent computational framework," *Progress in Biophysics and Molecular Biology*, vol. 113, no. 2, pp. 299–326, 2013.

[10] M. D. Neto, M. B. Oliveira, and J. F. Mano, "Microparticles in Contact with Cells: From Carriers to Multifunctional Tissue Modulators," *Trends in Biotechnology*, vol. 37, no. 9, pp. 1011–1028, 2019.

[11] V. Swaminathan and C. M. Waterman, "The molecular clutch model for mechanotransduction evolves," *Nature Cell Biology*, vol. 18, pp. 459–461, 5 2016.

[12] N. Wang, "Review of cellular mechanotransduction," *Journal of Physics D: Applied Physics*, vol. 50, no. 23, p. 233002, 2017.

[13] A. Elosegui-Artola, R. Oria, Y. Chen, A. Kosmalska, C. Pérez-González, N. Castro, C. Zhu, X. Trepat, and P. Roca-Cusachs, "Mechanical regulation of a molecular clutch defines force transmission and transduction in response to matrix rigidity," *Nature Cell Biology*, vol. 18, no. 5, pp. 540–548, 2016.

[14] A. Elosegui-Artola, X. Trepat, and P. Roca-Cusachs, "Control of mechanotransduction by molecular clutch dynamics," *Trends in Cell Biology*, vol. 28, pp. 356–367, 5 2018.

[15] N. D. Gallant, K. E. Michael, and A. J. García, "Cell Adhesion Strengthening: Contributions of Adhesive Area, Integrin Binding, and Focal Adhesion Assembly," *Molecular Biology of the Cell*, vol. 16, pp. 4329–4340, 2005.

[16] P. Van Liedekerke, M. M. Palm, N. Jagiella, and D. Drasdo, "Simulating tissue mechanics with agent-based models: concepts, perspectives and some novel results," *Computational Particle Mechanics*, vol. 2, no. 4, pp. 401–444, 2015.

[17] W. C. Lee, S. Kopetz, I. I. Wistuba, and J. Zhang, "Metastasis of cancer: When and how?," *Annals of Oncology*, vol. 28, pp. 2045–2047, 9 2017.

[18] T. N. Seyfried and L. C. Huysentruyt, "On the origin of cancer metastasis," *Critical Reviews in Oncogenesis*, vol. 18, pp. 43–73, 2013.

[19] T. Brabletz, R. Kalluri, M. A. Nieto, and R. A. Weinberg, "Emt in cancer," *Nature Reviews Cancer*, vol. 18, pp. 128–134, 1 2018.

[20] S. Douezan, K. Guevorkian, R. Naouar, S. Dufour, D. Cuvelier, and F. Brochard-Wyart, "Spreading dynamics and wetting transition of cellular aggregates," *Proceedings of the National Academy of Sciences*, vol. 108, 2011.

[21] Y. Zhang and R. A. Weinberg, "Epithelial-to-mesenchymal transition in cancer: complexity and opportunities," *Frontiers of Medicine*, vol. 12, pp. 361–373, 8 2018.

[22] Y. Iwadate, "Epithelial-mesenchymal transition in glioblastoma progression," *Oncology Letters*, vol. 11, pp. 1615–1620, 3 2016.

[23] J. Fares, M. Y. Fares, H. H. Khachfe, H. A. Salhab, and Y. Fares, "Molecular principles of metastasis: a hallmark of cancer revisited," *Signal Transduction and Targeted Therapy*, vol. 5, p. 28, 12 2020.

[24] D. Hanahan and R. A. Weinberg, "The hallmarks of cancer," *Cell*, vol. 100, pp. 57–70, 1 2000.

[25] D. Hanahan and R. A. Weinberg, "Hallmarks of cancer: The next generation," *Cell*, vol. 144, pp. 646–674, 2011.

[26] D. Hanahan, "Hallmarks of cancer: New dimensions," *Cancer Discovery*, vol. 12, pp. 31–46, 2022.

[27] S. Anvari, S. Nambiar, J. Pang, and N. Maftoon, "Computational Models and Simulations of Cancer Metastasis," *Archives of Computational Methods in Engineering*, vol. 28, no. 7, pp. 4837–4859, 2021.

[28] P. M. Altrock, L. L. Liu, and F. Michor, "The mathematics of cancer: Integrating quantitative models," *Nature Reviews Cancer*, vol. 15, no. 12, pp. 730–745, 2015.

[29] J. Metzcar, Y. Wang, R. Heiland, and P. Macklin, "A Review of Cell-Based Computational Modeling in Cancer Biology," *JCO Clinical Cancer Informatics*, no. 3, pp. 1–13, 2019.

[30] X. Guan, "Cancer metastases: Challenges and opportunities," *Acta Pharmaceutica Sinica B*, vol. 5, pp. 402–418, 9 2015.

[31] S. Paget, "The distribution of secondary growths in cancer of the breast," *The Lancet*, vol. 133, pp. 571–573, 3 1889.

[32] J. Ewing, *Neopastic diseases, A Text-book On Tumors*. W.B. Saunders Company, 1919.

[33] A. G. Fletcher and J. M. Osborne, "Seven challenges in the multiscale modeling of multicellular tissues," *WIREs Mechanisms of Disease*, vol. 14, no. 1, p. e1527, 2022.

[34] M. Meier-Schellersheim, I. D. Fraser, and F. Klauschen, "Multiscale modeling for biologists," *Wiley Interdisciplinary Reviews: Systems Biology and Medicine*, vol. 1, no. 1, pp. 4–14, 2009.

[35] M. P. Stumpf, "Statistical and computational challenges for whole cell modelling," *Current Opinion in Systems Biology*, vol. 26, pp. 58–63, 2021.

[36] J. S. Lowengrub, H. B. Frieboes, F. Jin, Y. L. Chuang, X. Li, P. MacKlin, S. M. Wise, and V. Cristini, "Nonlinear modelling of cancer: Bridging the gap between cells and tumours," *Nonlinearity*, vol. 23, no. 1, pp. R1–R91, 2010.

[37] T. S. Deisboeck, Z. Wang, P. MacKlin, and V. Cristini, "Multiscale cancer modeling," *Annual Review of Biomedical Engineering*, vol. 13, pp. 127–155, 2011.

[38] P. J. Murray, C. M. Edwards, M. J. Tindall, and P. K. Maini, "From a discrete to a continuum model of cell dynamics in one dimension," *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, vol. 80, no. 3, pp. 24–29, 2009.

[39] Y. Lee, S. Kouvroukoglou, L. V. McIntire, and K. Zygourakis, "A cellular automaton model for the proliferation of migrating contact-inhibited cells," *Biophysical Journal*, vol. 69, no. 4, pp. 1284–1298, 1995.

[40] A. Montagud, M. Ponce-de Leon, and A. Valencia, "Systems biology at the giga-scale: Large multiscale models of complex, heterogeneous multicellular systems," *Current Opinion in Systems Biology*, vol. 28, p. 100385, 2021.

[41] B. Chopard, R. Ouared, A. Deutsch, H. Hatzikirou, and D. Wolf-Gladrow, "Lattice-Gas Cellular Automaton Models for Biology: From Fluids to Cells," *Acta Biotheoretica*, vol. 58, no. 4, pp. 329–340, 2010.

[42] D. Dab, A. Lawniczak, J. P. Boon, and R. Kapral, "Cellular-automaton model for reactive systems," *Physical Review Letters*, vol. 64, no. 20, pp. 2462–2465, 1990.

[43] A. Deutsch, H. Hatzikirou, and C. Mente, "Lattice-Gas Cellular Automaton Models," in *Encyclopedia of Systems Biology*, pp. 1106–1108, New York, NY: Springer New York, 2013.

[44] J. M. Osborne, A. G. Fletcher, J. M. Pitt-Francis, P. K. Maini, and D. J. Gavaghan, "Comparing individual-based approaches to modelling the self-organization of multicellular tissues," *PLoS Computational Biology*, vol. 13, no. 2, p. e1005387, 2017.

[45] N. Guisoni, K. I. Mazzitello, and L. Diambra, "Modeling active cell movement with the Potts model," *Frontiers in Physics*, vol. 6, no. JUN, 2018.

[46] T. J. Newman, "Modeling Multicellular Systems Using Subcellular Elements," *Mathematical Biosciences and Engineering*, vol. 2, no. 3, pp. 613–624, 2005.

[47] A. G. Fletcher, M. Osterfield, R. E. Baker, and S. Y. Shvartsman, "Vertex models of epithelial morphogenesis," *Biophysical Journal*, vol. 106, no. 11, pp. 2291–2304, 2014.

[48] S. Alt, P. Ganguly, and G. Salbreux, "Vertex models: From cell mechanics to tissue morphogenesis," *Philosophical Transactions of the Royal Society B: Biological Sciences*, vol. 372, no. 1720, 2017.

[49] R. Alert and X. Trepat, "Physical Models of Collective Cell Migration," *Annual Review of Condensed Matter Physics*, vol. 11, pp. 77–101, 2020.

[50] M. Bock, A. K. Tyagi, J. U. Kreft, and W. Alt, "Generalized Voronoi Tessellation as a Model of Two-dimensional Cell Tissue Dynamics," *Bulletin of Mathematical Biology*, vol. 72, no. 7, pp. 1696–1731, 2010.

[51] D. E. Pinto, M. M. Da Gama, and N. A. Araújo, "Cell motility in confluent tissues induced by substrate disorder," *Physical Review Research*, vol. 4, no. 2, p. 023186, 2022.

[52] Z. Y. Luo, S. Q. Wang, L. He, T. J. Lu, F. Xu, and B. F. Bai, "Front tracking simulation of cell detachment dynamic mechanism in microfluidics," *Chemical Engineering Science*, vol. 97, pp. 394–405, 2013.

[53] A. P. Thompson, H. M. Aktulga, R. Berger, D. S. Bolintineanu, W. M. Brown, P. S. Crozier, P. J. in 't Veld, A. Kohlmeyer, S. G. Moore, T. D. Nguyen, R. Shan, M. J. Stevens, J. Tranchida, C. Trott, and S. J. Plimpton, "Lammps - a flexible simulation tool for particle-based materials modeling at the atomic, meso, and continuum scales," *Computer Physics Communications*, vol. 271, p. 108171, 2 2022.